Institut für Geographie
Fakultät für Angewandte Informatik
Universität Augsburg

Universität Augsburg University

# From Space to Place – A Computational Model of Functional Place

Inauguraldissertation zur Erlangung des Doktorgrades

vorgelegt von

David Jonietz

| | |
|---|---|
| Gutachter: | Prof. Dr. Sabine Timpf |
| | Prof. Dr. Christian Freksa |
| | Prof. Dr. Jukka Krisp |

| | |
|---|---|
| Tag der mündlichen Prüfung: | 22.03.2016 |

Für meine Eltern

# Danksagung

Mein größter Dank gilt Prof. Dr. Sabine Timpf, die mir stets eine hervorragende Mentorin, Betreuerin und Chefin war. Sowohl im Rahmen meiner Promotion als auch in meiner sonstigen beruflichen Tätigkeit konnte ich mich immer voll und ganz auf ihre uneingeschränkte Unterstützung, ihr vollstes Vertrauen und ihre große Geduld verlassen. Dafür meinen herzlichsten Dank!

Ebenfalls danken möchte ich meinen Kolleginnen und Kollegen an der Universität Augsburg, speziell Carolin von Groote-Bidlingmaier und Wolfgang Schuster für die einmalige Arbeitsatmosphäre und die zahlreichen Gespräche und Diskussionen, sowie Jean Damascene Mazimpaka und Andreas Keler für viele hilfreiche Tipps.

Ein großes Dankeschön geht an Prof. Dr. Christian Freksa für seine Bereitschaft dieses Promotionsvorhaben als Zweitbetreuer zu begleiten, sowie Prof. Dr. Jukka Krisp für seine Gutachtertätigkeit.

Ganz besonderer Dank gebührt meiner Familie, insbesondere meinen Eltern, die mich auf meinem gesamten Bildungsweg begleitet und immer in allen erdenklichen Belangen und Lebenslagen unterstützt haben.

Schließlich danke ich von ganzem Herzen meiner Frau Barbara für ihre stete Geduld, die unzählbaren Momente der Aufmunterung und den immerwährenden Rückhalt.

# Abstract


This research is motivated by the discrepancy between the spatial model underlying current geographical information systems (GIS) and the human way of structuring space into meaningful spatial units, which are generally referred to as places. Focusing on their functional dimension with regards to human action, this thesis presents a computational model of functional place, and conceptualizes it as a subset of geo-atoms, primitives of geographic representations, which are bound by a functional unity condition with regards to a complex spatial action.

Of the variety of challenges which accompany the endeavor of integrating place into GIS, two are explicitly addressed: the compound nature of places, and the subjectivity of place formation processes. Thus, in contrast to the high presence of places in our daily life, activities, and communication, they typically do not refer to a distinct geo-object, but rather consist of multiple geo-spatial entities, and might therefore not exist as a distinct entity in a geo-spatial database. Furthermore, functional places are mentally constructed by the individual, and thus prone to subjective variations in terms of their location, usage pattern and allocated meaning among different people. Inherited from its Euclidean spatial tradition, however, space is generally assumed to be an objective, mathematical reality in GIS, which is why possibilities to incorporate subjective differences in human perception and use of the world are still rare.

Applying an agent-based approach to modeling place, the concepts developed in this thesis allow for the dynamic construction of functional, agent-specific places from geo-spatial data. For this, a simulation framework is created which, based on concepts derived from ecological psychology, describes how an individual human agent assesses the suitability of its environment with regards to complex actions. On this basis, suitability, expressed as a scaled numeric value, can be computed in a way which is specific to the particular system of agent, environment and action. In a second step, this framework is embedded within a larger model of functional place formation, which enables the dynamic generation of functional places based on optimizing their resulting suitability with regards to a complex action.

On this conceptual basis, a software agent with the ability to dynamically construct functional places from its sensory inputs, and demonstrate the according spatial behavior, is developed and tested for its functionality in the context of a pedestrian simulation.

# Zusammenfassung


Die Motivation für die vorliegende Dissertation ergibt sich aus der existierenden Diskrepanz zwischen dem heutigen Geographischen Informationssystemen (GIS) zugrunde liegenden Raummodell einerseits, und der menschlichen Konzeptualisierung von Raum, welche eine Strukturierung desselben in einzelne räumliche Bedeutungseinheiten, bezeichnet mit dem englischen Begriff der *places*, beinhaltet, andererseits. Mit dem Fokus auf der funktionalen Dimension von *places* präsentiert diese Arbeit ein computer-basiertes Modell eines *functional place* als Teilmenge sogenannter *geo-atoms*, Primitive geographischer Repräsentationen, die in Bezug auf eine komplexe räumliche Aktion eine funktionale Einheit darstellen.

Von den vielfältigen Herausforderungen die sich bei der Integration von *places* in GIS ergeben werden 2 explizit adressiert: ihr Kompositumcharakter und die Subjektivität ihrer Formierungsprozesse. Tatsächlich verweisen *places* im Kontrast zu ihrer starken Präsenz in unserem täglichen Leben, unseren Aktivitäten und unserer Kommunikation typischerweise nicht auf ein distinktes Geo-Objekt, sondern bestehen aus mehreren geographischen Entitäten. Es ist daher möglich dass sie nicht als eindeutige Entität in einer räumlichen Datenbank existieren. Zusätzlich dazu werden *places* individuell mental konstruiert, weswegen in ihrer Wahrnehmung durch verschiedene Personen Unterschiede bezüglich ihrer Lokalität, ihren Nutzungsmustern und ihrer allokierten Bedeutung auftreten können. Im GIS wird Raum jedoch, bedingt durch das Erbe des traditionellen Euklidischen Raumverständnisses, generell als objektive mathematische Realität konzeptualisiert, was dazu führt dass Möglichkeiten der Einbeziehung subjektiver Unterschiede in der menschlichen Wahrnehmung und Nutzung ihrer Umwelt noch weitgehend fehlen.

Einem agenten-basierten Ansatz folgend werden in dieser Arbeit Konzepte entwickelt die eine dynamische Konstruktion von funktionalen, agenten-spezifischen *places* auf der Basis geographischer Daten ermöglichen. Für diesen Zweck wird, fundiert durch Konzepte der ökologischen Psychologie, simuliert wie ein individueller menschlicher Agent die Eignung seiner Umwelt bezogen auf komplexe Aktionen evaluiert. Auf dieser Basis kann eine räumliche Eignung, repräsentiert durch skalierte numerische Werte, spezifisch für das spezielle System aus Agent, Umwelt und Aktion berechnet werden. Diese Methode wird in einem zweiten Schritt in ein größeres Modell der Formierung von *functional places* integriert. Dieses ermöglicht schließlich deren dynamische Generierung basierend auf einer Optimierung der resultierenden Eignungswerte in Hinblick auf eine komplexe Aktion.

Diese Konzepte dienen als Grundlage für die Entwicklung eines Software Agenten mit der Fähigkeit, *functional places* in dynamischer Weise aus seinen sensorischen Eindrücken zu konstruieren und das entsprechende Raumverhalten zu demonstrieren. Im Anschluss wird dieser im Rahmen einer Fußgängersimulation auf seine Funktionalität hin getestet.

# Contents

# List of Figures

# List of Tables



# Listings

# 1 Introduction

*Perhaps much of the confusion that lies at the heart of geography today results from an awareness that there are simply many geographies and many possible worlds. Uncertainty arises because we know not which geography to choose, nor which possible reality we should aim for. We run the risk of becoming dogmatic by trying to force all worlds into one very limited format, and in doing so we ignore, belittle, or forget the others* (Golledge 1981, p. 21).

Our everyday live takes place in places. We constantly dwell and act in, decide upon, or move between various places, such as our home, our work place, or our favored restaurant. While the meaning of some places is unique and exclusive to us, others are shared with other people, however, all provide a "context for everyday action" (Jordan et al. 1998, p. 2). From the perspective of Geographical Information Science (GISc), understanding the cognitive processes which lead to places to be perceived, formed and used is of relevance with regards to place-related communication between a Geographical Information System (GIS) and its user, but also the GIS-based recommendation or prediction of functional places which provide the utilities for our intended activities, such as *EATING* (restaurant) or *SHOPPING* (mall).

The perception of the functional dimension of places, however, is highly subjective. Thus, the perceived opportunities for action provided by a place, but also its suitability, meaning its appropriateness with regards to a particular activity, tend to vary among its prospective users. Although the process of suitability evaluation is based on a place's objective spatial characteristics, which are either immediately perceived or recalled from memory, their resulting interpretation differs. This, in turn, influences the allocation of place utility, the level of satisfaction expected from interacting with this particular place, which correlates with the probability of it being selected and actually used (Golledge and Stimson 1997, Malczewski 2006). As the above citation by Golledge (1981) suggests, accordingly, there is no absolute reality, but instead many different geographies exist which, being created in our minds, provide the true context for our actions, a fact which, however, contradicts the assumption of an objective spatial truth which underlies current GIS. As a result, GIS are presently unable to represent such inter-individual differences in human spatial perception and the resulting behavior.

A further challenge is posed by the computational representation of places with regards to their functionality. Since they have typically emerged as a direct result of their manifold interactions with human agents, the action-related component of places is a determining factor with regards to their location and identity. Thus, places are typically functional units which consist of several geo-spatial entities, a fact which impedes their representation in a database. A place's potential for human action, however, due to the fact that it is not static and inherent in the environment but arises only from the dynamic interplay with an agent, cannot simply be stored as other thematic attributes, but instead requires an alternative form of computational representation. In GISc, there is still a need for the development of corresponding modeling concepts.

This work is set in the broader context of cognitive research in GISc. In particular, it is concerned with the development of a computational model of functional place which allows for its dynamic generation from geo-spatial data stored in a GIS while explicitly considering the subjectivity of place formation processes. On this conceptual basis, a software agent is developed which is able to dynamically construct and act in functional places, and tested for its functionality in the context of a pedestrian simulation.

## 1.1 Motivation

Simply put, geographers study places. This, however, does not imply that the concept of place is either simple or well-defined, in contrary, it is complex and in its terminological use marked by a high degree of ambiguity (May 1971, Cresswell 2009). From the phenomenological perspective of human geographers in the 1960s and 70s, place can be described as "a particular location that has acquired a set of meanings and attachments" (Cresswell 2009, p. 1). In accordance with the phenomenological approach to human-environment studies, which places the focus on the meaning of things instead of their objective, physical structure, place is conceptualized as space which is experienced, structured and semantically enriched (Graumann 2002). Places are created in the mind of the individual, but there are certain locations where shared meaning and activity overlap and converge. The formation of these collective places in space is in the focus of human geography, whereas questions regarding the involved cognitive processes of the individual person are of more interest to fields such as psychology or spatial cognition (Seamon 1980, Golledge and Stimson 1997, Carmona et al. 2010). Other disciplines including urban design, planning or architecture, in turn, are concerned with examining potential strategies for the effective creation of such "successful people places", rather than their mere description and analysis (Carmona et al. 2010, p. 201).

There are three aspects which determine a place's identity: its physical setting, the particular actions which can be conducted there, and any additional meaning which it is allocated, such as shared cultural values or personal memories (Relph 1976, Cresswell

2009). Of these, the functional component of a place is particularly important, which is due to the central role of interaction for human-environment relations in general, as frequently described in environmental psychology (Graumann 2002). The activities which we perform at places, therefore, have a great effect on how we mentally define their spatial location, and what meanings we attribute to them (Relph 1976). As a result, places are often equated to action spaces, partitions of the environment which are used by individuals for their spatial behavior (Golledge and Stimson 1997, Goodchild 2011).

Apart from the scientific observation, description and analysis of real-world places, however, their fundamental importance for human spatial cognitive processes raises the relevance of them being represented in a computer, a topic which, as mentioned previously, lies well within the scope of GISc. In fact, there is a need for research, which is due to an existing discrepancy between place as experienced by humans and the abstract, mathematical, and objective Euclidean model of space, which traditionally provides the basis for GIS (Couclelis 1999). Representing a place as a regular geo-object in a current GIS, therefore, would imply it to exist as an atomic, unique entity in the geo-spatial database, be universal, absolute and static in terms of its geometrical, topological and thematic properties, and allow appropriate modeling as a point, line, polygon or raster field (Goodchild 2011, Winter and Freksa 2012). These criteria, however, are typically not applicable to places, which results in problems regarding the use of and interaction with GIS, as for instance in the following exemplary situations:

- References to places by name or denomination, e.g. "*Is my hotel located in Downtown?"*
- Place-based qualitative queries, e.g. "*Where can I go to sit in the shade and have a coffee?"*
- GIS-based analysis of human spatial behavior, e.g. "*Why are some areas of the city more frequented by pedestrians than others?*"

A particular problem for the appropriate treatment of place in GIS is posed by its compound nature. In fact, rather than referring to a distinct geo-object, a place typically emerges from relationships between multiple geo-spatial entities (Winter and Freksa 2012). As it has been argued, these relations are often of functional nature, and result in an indeterminate location and footprint of a place (Scheider and Janowicz 2014). The true shape of *Downtown*, for instance, can only be approximated by a polygonal or field-based representation, which, however, would be a prerequisite for further spatial operations and queries. Moreover, a user reference to a place called *Downtown* might fail due to its absence as a distinct entity in the database (Goodchild 2011, Scheider and Janowicz 2014).

Further, the notion of a place is a mental construct, and therefore prone to subjective variations among individual persons, which, as will be argued in this thesis, affect its location, action potential and meaning (Jonietz and Timpf 2013a). The subjectivity of place is of relevance for action-related queries, thus, to refer back to the previous example, depending on the personal preference, a suitable place to *SIT IN THE SHADE* may be restricted to benches and chairs, or include informal seating opportunities such as stair steps (Carmona et al. 2010). Apart from recommendations, a place's individuality also concerns the value of GIS for the analysis and prediction of human spatial behavior, which, as has been stated earlier, is based not on an objective, mathematical model of space with fixed locations and universal meaning, but rather our own subjective reality (Golledge and Stimson 1997, Kwan 2000). It has been observed, for instance, that the route choice of pedestrians in urban areas depends significantly on the degree by which the perceived environment meets their specific walking needs, and is not always mathematically optimized in terms of distance minimization (Özer and Kubat 2007, Agrawal et al. 2008).

In accordance, the main challenges for GISc with regards to the concept of place include its computational representation and automated localization (Janowicz et al. 2012). In the past, the role of places as action spaces has been identified as one potential approach towards modeling place in GIS. In this context, work has built on a concept of ecological psychology, Gibson's (1979) theory of visual spatial perception, in which the process of human perception of action potentials, termed affordances, within the environment is described. According to the author, the meaningful environment, the part of the environment within our perceptual limits, is composed of basic primitives of type substance, surface and media, the affordances of which are directly perceivable but always relative to the observer. Similar to the notion of compound places, more complex, higher-order affordances may require a combination of such environmental primitives (Gibson 1979). Thus, from a functional perspective, places can be described as "any meaningful spatial configuration of shared affordances to the human body" (Richter and Winter 2014, p. 15), a view which, though, has been accused of simplifying the place concept (Carmona et al. 2010, Goodchild 2011).

Interwoven with their computational representation, but also in itself a critical issue is the localization of places. In our daily communication and behavior, places play a central role, and are referred to by names or descriptions, photographs or other depictions. For a user-GIS communication to be possible, the automated georeferencing of places is a central prerequisite. Thus, work from the area of GISc has focused on the development of methods to infer the location of places from various data on human communication (Vasardani, Winter et al. 2013). Further, the observation and analysis of human spatial behavior can provide the basis for the identification and discovery of places, such as neighborhoods or city centers (e.g. Hollenstein and Purves 2010, Zhong et al. 2013). If no such data is available, however, place locations can also be predicted based on computational models of human spatial behavior, for instance using GIS-based spatial analysis or agent-based models (ABM) (Batty 2001, Benenson and Torrens 2005).

Despite previous research activities, however, the goal of integrating place into GIS has not yet been fully achieved. For instance, the problem of modeling affordances, which arise from the interplay of several spatial entities in compound places and the agent itself, has only recently been addressed (Scheider and Janowicz 2014). Due to the fact that these cannot be represented as simple attributes of a geo-object, novel forms of their computational representation are required. Moreover, the actions which are afforded by and conducted at places are typically of complex nature, and mutually connected in manifold causal inter-dependency relationships. Current methods of conceptualizing agent-environment interactions, in contrast, often apply a purely reactive approach, which falls short of incorporating such complexities. In the context of ABM, for instance, human spatial actions are often modeled as simple transition functions which, determined by an agent's percepts of the environment, change the current states of itself and the environment (Russel and Norvig 2003). The affordance concept has been identified as a potential conceptual approach for modeling more complex agent behavior (Klügl 2015).

A further yet unsolved problem is posed by the subjectivity of place perception and formation. In this context, the particular challenge lies in the semantic translation of objective information stored in a spatial database to the unique perceptual context of individual users, or in other words, the simulation of the subjective process of spatial information interpretation. In the context of places, a particular emphasis is put on functional, action-relevant information such as their affordances or the evaluated suitability. Currently, the value of GIS as basis for recommender or assistive systems is reduced by the assumed objectivity of the stored geo-spatial data. Although personalization is a prominent issue in GIS-related research, most efforts are put into the adaptation of the system itself, the method for visualizing the geo-spatial information, or learning systems which predict future behavior of users based on their previous actions (Aissi and Gouider 2012). To the best of our knowledge, for instance, there is presently no framework which allows for the automated inference of personalized suitability values of places for actions. Instead, the personalization of recommended places, for instance for *WALKING*, is usually based on rules of thumb and the personal opinion of an expert who is involved in the manual evaluation process (Jonietz et al. 2013). Modeling inter-individual differences in how the world is perceived and used, however, are also of relevance for the value of GIS as a tool for analyzing human spatial behavior, which is still mostly based on objective space, and follows an aggregated, macroscopic approach. Instead of assuming objectivity and perfect spatial knowledge, however, the explicit consideration of a person's actual cognitive environment would likely increase the realism of behavioral models (Kwan 2000).

Against this background, three distinct motivations for writing this thesis arise:

A first motivation lies in the fact that GISc is in need of a method to computationally represent places which explicitly addresses their compound and subjective nature. This is due to place not being inherent in the environment, but arising as a result of dynamically changing human-environment interactions. In contrast to the traditional procedure of representing places as polygons or via their representative points, an alternative approach would be to construct places on-the-fly by simulating the process of individual place formation, which includes the perception of affordances related to complex actions as well as the according behavior. The dynamic generation of functional places from spatial primitives would eliminate the necessity for the representation of a place as a distinct entity in a geo-spatial database, and allow the incorporation of subjectivity, since the places can be constructed with regards to each specific user or agent. In the context of modeling compound places, a concept of geographic representation primitives could be useful, such as the notion of geo-atoms by Goodchild et al. (2007), who introduce them as abstract associations between a point location in space-time, a property and the value of that property, which provide the fundamental building blocks for higher-dimensional geo-objects such as lines and areas.

Secondly, the central importance of a place's functional suitability for the process of place formation and the resulting spatial behavior has already been emphasized. Nevertheless, and despite the fact that suitability assessment is a standard method in GIS (e.g. Malczewski 2006), there is currently no conceptual framework which describes how an individual human agent evaluates the spatial suitability of places with regards to specific actions. Such a concept, however, would allow for the automated computation of personalized suitability values from static spatial data stored in a GIS. This would be of high relevance for the development of personalized assistive or recommender services, and likely increase the explanatory potential of GIS as tool for the analysis of human spatial behavior (Jonietz and Timpf 2013a).

A third motivation links the topic of place to ABM. So far, an ABM paradigm has not been used to approach the problem of modeling place, however, can be expected to be valuable due to various reasons. Thus, the possibility to create individual, heterogeneous agents allows for modeling on a disaggregate, microscopic level, in which subjective differences can be incorporated. Furthermore, the percepts, cognition and behavior of each artificial agent can be modeled with a high level of detail, and, when being used in combination with a GIS, it is also possible to situate artificial agents in a detailed model of the environment, with which they can interact. Finally, ABM are well suited to modeling emerging phenomena, such as collective places. A potential area of application is pedestrian simulation, where the current practice of simplifying agent cognition and inter-individual differences stands largely in contrast to empirical findings on the conceptual richness of walkability, the suitability for *WALKING* (e.g. Sallis et al. 2012).

## 1.2 Aims, Premises and Hypotheses

The main aim of this work is to develop a computational model of place, which allows for the dynamic generation of individual places from spatial data. It encompasses three subgoals:

1. Develop a simulation framework for individual spatial suitability assessment based on an extended notion of affordances and a multi-level hierarchical action model.
2. On this foundation, develop a conceptual model of place as a functional subset of geo-atoms.
3. Implement and test the place concept as part of a reasoner of an agent-based model of pedestrian movement in an urban area.

For this, we base our work on several premises:

- The semantic dimension of places can be reduced to their affordances. Following the current practice in disciplines such as urban design or GISc, the focus is restricted to their functional dimension. Throughout this thesis, thus, we will explicitly focus on functional places.
- Places are individually perceived and used by human agents, when shared affordances and the resulting usage patterns overlap, collective places emerge (Seamon 1980, Golledge and Stimson 1997, Carmona et al. 2010).
- Human agents are able to perceive action potentials within their meaningful environment. These are not universal qualities, but depend on the observer as well as the environment. This concept of agent-environment mutuality is a core concept of ecological psychology (Gibson 1979).
- Human spatial decisions and the resulting behavior are guided by a homo economicus paradigm. Despite the fact that a satisfier approach might in some cases be closer to the reality of human behavior and decision making, the assumption of utility maximization is still is a useful simplification for behavioral models (Golledge and Stimson 1997).

On the basis of the premises stated before, the main hypothesis of this work is as follows:

*A functional place can be defined as a subset of geo-atoms, which are bound by a functional unity condition with regards to a complex, high-level spatial action. Assuming a suitability-based optimization strategy of human spatial behavior, this conceptualization allows the dynamic generation of individual places from spatial data.*

The main hypothesis can be further broken down into the following sub-hypotheses:

1. *Spatial suitability must be modeled as an abstract quality of the system of agent, geo-atom and action.*

2. *In order to assess the suitability provided by a place for reaching a desired goal state, an agent needs to evaluate its suitability with regards to the contributing actions.*
3. *A functional place which affords reaching a desired goal state consists of the geo-atoms which afford the contributing subordinate actions with the highest possible suitability.*
4. *The generated places are sensitive to the needs and characteristics of the individual agent.*

## 1.3 Approach

This work comprises three steps: First, a conceptual model of the process of individual spatial suitability assessment is developed based on Gibson's (1979) affordances and activity theory (Leontiev 1978, Kemke 2001). We build on post-Gibsonian work on the formalization and measurement of affordances to extend the notion of affordances from expressing binary true/false-statements to scaled suitability values (Warren 1984, Stoffregen 2003). For this, we define suitability as an abstract quality of the system consisting of agent, environmental primitive and action. Drawing from activity theory, complex actions are modeled as hierarchical constructs which can be represented on three different levels of abstraction, the goal state to be reached by the agent, the necessary state changes to fulfill the goal state, and the operational actions to be performed by the agent in order to cause the state changes (Leontiev 1978, Kemke 2001). This, in combination with the extended affordance-concept, allows for the dynamic calculation of individual suitability levels for each distinct combination of agent, environmental primitive and action based on static spatial data (Jonietz et al. 2013, Jonietz and Timpf 2013a).

In a second step, this concept is used to develop a computational model of place and simulate the process of place formation. For this purpose, places are conceived of as subsets of atomic spatial entities (e.g. raster cells), which broadly refer to environmental primitives but, due to the fact that at this stage of model development, the context is digital information modeling, are called geo-atoms in reference to Goodchild et al. (2007). These are bound by a functional unity condition with regards to a complex, high-level action, as defined in the suitability assessment framework described above. With regards to a specific pair of agent and geo-atom, and based on their respective properties, possible action strategies aiming at reaching the goal state and their expected suitability values can be computed. Following a suitability maximization strategy, it is thus possible to calculate an optimal course of action, which defines the necessary agent behavior in terms of *what* it has to do and *where* in order to perform the highest-level action, or, in other words, reach its goal state. The associated geo-atoms constitute the elements of the individual place formed with reference to this particular high-level action.

In the third step, the conceptual model is implemented in Java as the extension package PlaceBuilder to the NetLogo simulation environment. It is then tested in the context of a pedestrian movement simulation, where it serves as part of the agent reasoner. One aim is to demonstrate the conceptual strength of the proposed approach by incorporating complex interactions between the pedestrian agent and its walking environment. Thus, in contrast to prior work on pedestrian simulations, the behavior of the virtual pedestrians is not restricted to movement, but, based on a review of empirical literature on walking, a model of subjective walkability assessment is developed, and used to compute suitability values for the complex action *WALK*. During the trip, the software agent computes walkability-maximized places for *WALKING* through its immediate environment, which are tailored to its individual characteristics. The model results are tested for formal correctness and plausibility by means of observing its behavior in pre-defined scenarios, and conducting a local sensitivity analysis. Further, using the example of important, highly-frequented pedestrian places, potentials for the further analysis of collective places are briefly demonstrated.

## 1.4 Contributions

In contrast to previous research on representing place in GIS with affordances, this work proposes a method for the dynamic generation of individual places from spatial data. In particular, it is concerned with the challenges of representing compound functional places and their complex affordances, and incorporating subjectivity in the process of place formation.

Therefore, the main contributions are:

1. A simulation framework for individual spatial suitability assessment is developed based on an extended notion of affordances and activity theory. This allows for the automated calculation of individual suitability values for each agent-environment-action-system.
2. A computational model of functional place is developed, which represents a novel approach to represent complex functional relations between the agent and the environment. Based on an innovate representation of higher-order affordances as properties of compositions of numerous agent-environment-action-systems, it is possible to dynamically generate functional places.

In addition, there are significant practical outcomes resulting from this work:

1. The conceptual model of place formation is implemented as an extension for the agent-based simulation environment NetLogo (PlaceBuilder), and is expected to be valuable for modeling different kinds of behavior apart from walking.

2. To the best of our knowledge, this work represents the first approach to develop a model of subjective walkability and use it for navigation in an agent-based pedestrian simulation.
3. The developed pedestrian model provides a potential basis for a practical planning tool to assist urban planners or designers. In comparison to previous approaches, it enables modelers to acknowledge the heterogeneity of pedestrians to a high degree.

## 1.5 Thesis Outline

This thesis is organized as follows: Chapter 2 presents the theoretical background from an interdisciplinary perspective, gradually shifting the focus from ecological psychology (Chapter 2. 1), and the introduction of the concept of affordances, via human geography and urban design (Chapter 2. 2), where the phenomenological concept of place is explained, to GISc (Chapter 2. 3), where previous work on place integration in GIS is discussed.

Drawing from this theoretical foundation, in Chapter 3, the focus is placed on the methodological underpinning of this research, starting with the discussion of geosimulation and agent-based modeling as a potential approach for microscopic modeling of human behavior (Chapter 3. 1), and, in anticipation of the thematic scope of our case study, moving on to prior research on pedestrian-environment interactions (Chapter 3. 2). First, the focus is on approaches and methods for pedestrian simulation. Rather than giving a full review, however, the aim is to discuss basic methods and techniques, especially with a focus on modeling pedestrian-environment interactions and subjectivity. Then, empirical research on walkability is reviewed.

Chapter 4 presents the main contribution of this thesis. Thus, an approach for modeling place and simulating the process of place formation is presented, first on the conceptual level (Chapter 4. 1) and then followed by the computational model (Chapter 4. 2).

With pedestrian simulation, Chapter 5 presents an exemplary application of the concepts developed in this thesis. As a first step, building on the empirical research on walkability presented in Chapter 3. 2. 2, the complex action *WALK* is modeled in accordance with our framework (Chapter 5. 1). In the following, the simulation is described, tested and the results presented (Chapter 5. 2).

In Chapter 6, the main aims of this thesis are discussed with regards to the achieved results, before our work is finally concluded in Chapter 7.

# 2 Theoretical Background

This chapter provides an overview on prior work of relevance to this thesis. On the basis of literature originating from the fields of Spatial Cognition, Human Geography and GIScience, it sets the foundation for the development of a conceptual model of place, which is described in Chapter 4. Accordingly, this chapter is further separated as follows:

Chapter 2. 1 approaches the topic from the viewpoint of ecological psychology, and, based on Gibson's (1979) affordance theory, describes a fundamental model of how humans perceive action potentials in their environment. In order to further demonstrate the usefulness of this concept for computational models of human-environment interactions, relevant research from the fields of GISc and ABM is briefly reviewed.

Chapter 2. 2 shifts the focus from the individual’s perception of his or her ecological environment to the phenomenological study of human experience on the geographical scale, and presents the concept of place by reviewing literature originating mainly from the fields of human geography and urban design. The central role of the functional dimension and the influence of subjectivity for place formation processes is emphasized. Further, it is discussed how collective places emerge from individual place perception processes.

Chapter 2. 3 relates the two concepts of affordance and place from the perspective of GISc. Prior work on the computational representation and automated localization of place is discussed. Particular challenges and unresolved problems are explicitly addressed.

## 2.1 Ecological Psychology and Gibson’s Affordances

Ecological psychology, a movement in perceptual psychology which emerged in the early 1940s, originates from ecological science, a more general, multidisciplinary endeavor to study energy transactions between living organisms and their environments. In this particular case, however, it is not energy but the flow of information between an animal and its physical surroundings which is in the center of attention, with a particular focus on functional relations relevant for planning or performing actions (Morris 2009).

Although some authors trace it back to James’s radical empiricism, the emergence of an ecological approach to spatial cognition is generally attributed to the works of James J. Gibson and Roger Barker, respectively. While Gibson (1979) focused on environmental perception on the individual level, Barker (1968) chose an extra-individual approach to describe and predict environmentally situated human behavior. His theory of behavior settings states that collective human behavior is not independent of the environmental setting, but rather influenced by sociocultural structures which consist of environmental and human components as well as control circuits (Barker 1968, Heft 2001).

Despite the existence of similarities shared by both approaches, the fact that Gibson focuses explicitly on spatial perception of the individual level makes it more suitable as a theoretical foundation for our notion of individual place formation. Accordingly, it is further explained in the following.

### 2.1.1 The Concept of Affordances

Being interested in the perceptual processes of animals, including humans, J. J. Gibson investigated how they extract the meaning of environmental objects from visual stimuli. In contrast to the prevalent psychological theories of his time, which emphasized the role of mental calculation processes in creating meaning from otherwise meaningless sensory information, Gibson, inspired by Gestalt theorists, most notably Kurt Lewin (1935) and Kurt Koffka (1935), postulated perception to be a direct, unmediated process (Jones 2003). Gestalt theory, embracing a holistic view of perception in contrast to Elementarism, provided a conceptual basis for ecological psychology, since it highlighted the central role of meaning for human environmental perception and described objects with regards to the action possibilities provided by them (Jenkins 2008). Referring to concepts such as Koffka's (1935) *demand character* or Lewin's (1935) *Aufforderungscharacter*, Gibson himself noted, "[t]he gestalt psychologists recognized that the meaning […] of a thing seems to be perceived just as immediately as its color" (Gibson 1979, p. 138).

Thusly inspired, Gibson published several works on visual perception in which he gradually developed the concept of affordances. Although prefigured in his early writings on motion perception in the specific situations of driving an automobile (Gibson and Crooks 1938) or landing an airplane (Gibson 1947), the most elaborate explanation of notion of affordances can be found in *The Ecological Approach to Visual Perception*, his last piece of writing before his death:

"The affordances of the environment are what it *offers* the animal, what it *provides* or *furnishes*, either for good or ill. The verb *to afford* is found in the dictionary, but the noun *affordance* is not. I have made it up. I mean by it something that refers to both the environment and the animal in a way that no existing term does. It implies the complementarity of the animal and the environment." (Gibson 1979, p.127, emphasis in original).

In Gibson's view, the environment of animals, that is their perceived surroundings, can be distinguished from the physical world by referring to the size-level that their perceptual systems are able to detect, which is relative to their own size. Compared to the physical world, which ranges from atoms to galaxies, only a narrow band can be directly perceived by animals. According to Gibson, on this ecological level, the environment cannot be sufficiently described in terms of physical bodies, but rather as consisting of media, substances and, most importantly, surfaces, all of which afford certain actions for animals. A medium, such as air, affords breathing, locomotion, as well as visual, auditory or olfactory perception for terrestrial species. Substances, including liquids or solids, may afford nutrition, drinking or manufacture. Rather than the substances themselves, however, their surfaces, which separate them from the medium, reflect light and are visually perceived. The surface, therefore, “is where most of the action is”, and may afford standing or walking or act as a barrier (Gibson 1979, p. 23). When topologically closed, a surface can define an object which is detached from the ground and which, depending on its size, shape or structure, may afford actions such as lifting, throwing, hitting or cutting (Gibson 1979). In Gibson’s view, the process of visual perception can be described as the extraction of invariants, constant components of perceptual stimulation patterns, from the stimulus flux, which make the apprehension of affordances possible (Sheehy, Chapman and Conroy 1997).

One of the key innovations of Gibson’s work and a core concept of ecological psychology is the principle of agent-environment mutuality (Varela and Thompson, 1991). On the one hand, affordances are determined by environmental properties, such as the composition and layout of surfaces. On the other hand, however, an affordance is also related to the properties of the animal, since, being neither objective nor subjective, “an affordance points both ways, to the environment and to the observer.” (Gibson 1979, p. 128). The question whether a terrestrial surface is stand-on-able, for instance, depends on its properties such as its extent or rigidity in relation to the size or weight of the respective animal (Gibson 1979). A stone affords throw-ability only if its size and weight fit the grasp size and strength of a human agent. Higher-order affordances, which refer to more complex actions, are related to not just one but numerous properties of the environment, which is why Gibson introduced the notion of compound invariants as invariant combinations of invariants, which can also be extracted by the visual system (Gibson 1979).

### 2.1.2 Approaches for Formalizing and Measuring Affordances

In the past, numerous experimental studies have provided support for the theory of affordances, most notably Warren (1984) who, with his stair climbing experiment, could not only identify the specific environmental and person-related components related to an affordance, but also provide an approach to measure or predict its existence. In an experimental setup, test persons were asked to visually rate the perceived climbability of a number of stair steps of varying step height. When analyzing the resulting data, the author found that the perceptions of this particular affordance did not merely depend on the environmental attribute, the step height, but correlated to a ratio

$$\pi = \frac{R}{L} \tag{1}$$

where the riser height $R$ is related to the person’s leg length $L$ (Warren 1984). In accordance with Gibson's theory, a relation described by a ratio can remain constant even if the involved properties change.

As illustrated in figure 1, Warren (1984) further identified a critical maximum threshold value of .88, which is not to be exceeded by $\pi$ since from that point on, the action is no longer perceived as possible. This value was termed $\pi_{max}$ by the author. There is, however, also an optimal point $\pi_O$, which denotes the ratio at which the energy expenditure required for the respective action is at a minimum (Warren 1984). In the context of a different example, namely the evaluation of the potential for walking through apertures, Warren (1995) also defined a $\pi_{min}$, the critical minimum threshold value of the ratio of aperture and shoulder width below which the affordance no longer exists.

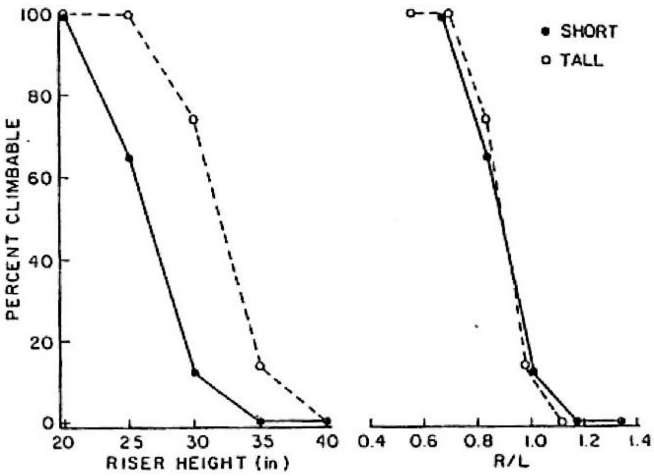


Fig. 1: Perceived Climbability and Riser Height (left) and $\pi$ (right)
(adapted from Warren 1995)

In addition, a range of experimental studies have examined other affordances, including the depth perception of infants (Gibson and Walk 1960), the sit-ability of surfaces of varying height under the effect of altered personal capabilities (Mark et al. 1990), the reach-ability of objects (Carello et al. 1989) and the cross-ability of roads (Oudejans et al. 1996).

In psychology, however, the notion of affordances remains a controversial issue, which is mostly due to its radical departure from established cognitive theories which emphasize the importance of memory-based inferences on the interpretation of sensory input (Gaver 1991). Thus, several researchers have denied the sufficiency of Gibson's theory to fully explain human perceptual processes. Eco (1999), for instance, criticized the fact that Gibson leaves no place for an integration of previous knowledge or other cognitive interpretations of stimuli. Another example is Oliver (2005), who placed Gibson's thesis on the abstract level of a whole species, and consequently questioned its relevance for explaining an individual's actions. Apart from doubts regarding its fundamental assumptions, the affordance concept has been accused of being conceptually vague due to the fact that Gibson tended to provide illustrating examples instead of a formal definition (Wells 2002). Moreover, Gibson himself described his theory as "subject to revision" and, until his death in 1979, constantly modified it (Gibson 1977, p. 67).

Apart from merely criticizing its conceptual vagueness, however, several ecological psychologists have proposed further refinements and formalizations of Gibson's theory. The first approach came from Turvey (1992):

> "Let $Wpq$ (e.g., a person-climbing-stairs system) $= j(Xp, Zq)$ be composed of different things $Z$ (e.g., person) and $X$ (e.g., stairs). Let $p$ be a property of $X$ and $q$ be a property of $Z$. Then $p$ is said to be an affordance of $X$ and $q$ the effectivity of $Z$ (i.e., the complement of $p$), if and only if there is a third property $r$ such that
>
> (i) $Wpq = j(Xp, Zq)$ possesses $r$
>
> (ii) $Wpq = j(Xp, Zq)$ possesses neither $p$ nor $q$
>
> (iii) Neither $Z$ nor $X$ possesses $r$ where $j$ is a joining or juxtaposition function"
>
> (Turvey 1992, p. 180).

In his example, Turvey (1992) illustrated his notion of an affordance $p$ as a property of the environment, which is determined in its existence by what he termed the effectivity of an involved person. The definition of an affordance as a property of the environment is shared by numerous other authors (e.g. Heft 2001, Michaels 2003).

In later approaches, however, such formalization attempts have been criticized. In the view of Stoffregen (2003), for instance, an affordance as an environmental characteristic would naturally imply the need for further mental processing by the observing animal, a fact which clearly contradicts direct perception, one of Gibson's (1979) basic assumptions. The author therefore proposed an alternative definition of affordances as "properties of the animal-environment system […] that do not inhere in either the environment or the animal" (Stoffregen 2003, p. 123). In reference to Turvey (1992), he argued for a revised formalization of affordances:

> "Let $Wpq$ (e.g., a person-climbing-stairs system) $= (Xp, Zq)$ be composed of different things $Z$ (e.g., person) and $X$ (e.g., stairs). Let $p$ be a property of $X$ and $q$ be a property of $Z$. The relation between $p$ and $q$, $p/q$, defines a higher order property (i.e., a property of the animal–environment system), $h$. Then $h$ is said to be an affordance of $Wpq$ if and only if
>
> (i) $Wpq = (Xp, Zq)$ possesses $h$
>
> (ii) Neither $Z$ nor $X$ possesses $h$"
>
> (Stoffregen 2003, p. 123).

This view is also shared by Chemero (2003), despite the fact that in his opinion, affordances should be defined as relations between an animal's abilities and environmental features rather than properties of the animal-environment system.

### 2.1.3 The Use of Affordances in GISc and ABM

Due to its combination of conceptual simplicity and strength, the notion of affordances has served as a theoretical basis for a range of studies originating from a variety of disciplines other than psychology including GIS and GISc (Llobera 1996, Jordan et al. 1998, Frank and Raubal 1999, Kuhn 2001, Kuhn and Raubal 2003, Kuhn 2007, Janowicz and Raubal 2007, Timpf 2008, Scheider and Kuhn 2010, Alazzawi et al. 2012, Jonietz et al. 2013, Ortmann et al. 2014), artificial intelligence and ABM (Raubal 2001a, Raubal 2001b, Turner and Penn 2002, Michael and Chrysanthou 2003, Raubal and Moratz 2006, Kapadia et al. 2009, Kim et al. 2011, Jonietz and Timpf 2013a), assistive technologies (e.g. Mastrogiovanni et al. 2010), robotics (e.g. Rome et al. 2008), as well as the design of everyday things (Norman 1988) or graphical user interfaces (Gaver 1991, Kuhn 1996).

In the context of GISc, two aspects of Gibson's (1979) work have particularly contributed to its popularity, namely its action-centered view on spatial entities and the notion of agent-environment mutuality. While the first characteristic has been particularly useful for the identification and grounding of semantic primitives for geo-ontology engineering, the formalization and modeling of place, and the development of models of human spatial behavior, the latter has contributed to issues such as information integration with Semantic Reference Systems (SRS) and the incorporation of subjectivity in GIS (Jonietz and Timpf 2015).

In terms of ontologies, the theory of affordance can be useful since it emphasizes the central role of action potentials for human spatial perception and provides a semantic connection between the afforded action and the geo-object (Janowicz et al. 2012). Thus, affordances can support the grounding and interpretation of semantic primitives and therefore improve the usefulness of geo-ontologies and, as a result, the usability of GIS (Janowicz et al. 2012, Kuhn 2001). In Kuhn (2001), for instance, human actions are deployed as basic ontological elements which are connected to environmental objects via affordances. Inspired by work by Raubal et al. (1996) on structuring space for wayfinding, in a later study, the same author combined affordances with Johnson’s (1987) image schemata, recurring image-like reasoning patterns which form a foundation for human cognitive processes, as basic primitives for the categorization of geospatial entities (Kuhn 2007). In the same context of categorizing geo-spatial entities, Janowicz and Raubal (2007) proposed a method for the measurement of their semantic similarity. Based on the premise that the similarity of entities depends on their common functionalities which are provided to a user, the authors argued in favor of the use of affordances for such purposes (Janowicz and Raubal 2007). Scheider and Kuhn (2010) focused on locomotion-related affordances to categorize elements of road networks. The question of how to ontologize affordances was addressed by Ortmann and Kuhn (2010), who relate them to observations by defining them as observable environmental qualities.

Since affordances express action potentials together with their environmental or agent-related preconditions, their value for the formulation of interactions between agents and their environment in the context of human spatial behavior modeling has been identified. Thus, affordances enable a modeler to structure agent behavior which involves a particular spatial entity on a higher conceptual level than traditional reactive approaches by incorporating the agent's exact motivation for interacting with this particular spatial entity. It is expected that this conceptual shift leads to higher levels of behavioral complexity to be modeled in ABM or other methodological approaches (Klügl 2015). An early example based on GIS can be found in archaeology, when Llobera (1996) uses spatial analysis in order to identify landscape affordances from the perspective of an individual, in particular the visibility of ditches which served as visual territorial markers for prehistoric people. In the context of ABM, Raubal (2001b) built on Gibson's work to model an ontology and epistemology for a simulation of human wayfinding in airports. Regarding ontological issues, the author distinguished between media, such as light or sound, substances like fellow travelers, counters or gates, and surfaces, which separate the substances from the media. Particular properties of substances and surfaces can provide affordances such as movement or wayfinding to certain agents, who are then able to act upon the perceived action potentials. The author extended Gibson's (1979) ideas by introducing physical, social-institutional and mental affordances, which correspond to physical properties, social and institutional rules and mental processes of the agent. By simulating wayfinding tasks for various agents, the proposed tool can be used to identify wayfinding difficulties or inconsistent signage in an airport space (Raubal 2001a). A more general approach is followed by Raubal and Moratz (2006), who proposed a functional model for affordance-based agents as an alternative to an object-oriented paradigm. The authors integrate an extended notion of affordances (Raubal 2001b) with the HIPE theory of function as a basis for representing the agent's functional knowledge (Barsalou et al. 2005). Concerning the agent, the authors distinguish between its physical structure, its spatial and cognitive abilities and its goal, properties which, in relation to environmental properties, social-institutional context and spatio-temporal setting, determine which affordances are perceived (Raubal and Moratz 2006). Ksontini et al. (2014) proposed an affordance-based agent model for a road traffic simulation, in which they distinguish between occupied and unoccupied road space based on its affordance for movement. Klügl (2015) developed a methodology for affordance-based agent interaction design on the example of an agent simulation of human behavior after an earthquake. Furthermore, there are several approaches in which affordances are used to determine the spatial behavior of artificial pedestrian agents, none of which, however, moves beyond basic navigational issues such as obstacle avoidance and local path planning (Turner and Penn 2002, Michael and Chrysanthou 2003, Kapadia et al. 2009, Kim et al. 2011).

Another aspect of geospatial semantics in which, according to Janowicz et al. (2012), the inclusion of affordances represents a promising approach is the creation of formal models of place, which, however, will be discussed in detail in the following chapter.

A topic which is closely related to ontologies but yet deserves to be treated separately is information integration and Semantic Reference Systems (SRS). SRS were first proposed by Kuhn (2003), who, in analogy to spatial or temporal reference systems, developed the idea of a common basis for values or symbols related to geospatial data. Relevant aspects of geographic information include for instance feature names, attribute values or relationships which need a reference in order to be appropriately interpreted, translated and integrated across different information systems (Kuhn 2003, Kuhn and Raubal 2003). In this context, it is particularly the notion of agent-environment mutuality which justifies the usefulness of the affordance-concept. The insight that affordances are always relative to the observing organism provides a conceptual basis for the translation of semantic information among different users or information systems (Frank and Raubal 1999). Kuhn and Raubal (2003) applied SRS on a road navigation application as a simple use case. Although the authors refrain from explicitly using the term affordances, the generic image schemata are linked to the specific actions which they afford. In an information system built accordingly, it is possible to simplify, a process which the authors term semantic projection, or transform the data model in the sense of a semantic transformation (Kuhn and Raubal 2003). Building on this concept, Ortmann et al. (2014) developed an egocentric reference system for affordances, which allows the translation of affordances expressed as ordinal values across different users. Based on the agent as an egocentric datum, their theory enables the transformation of user observations and ratings to another person's reference frame, which, due to differing capabilities, will result in a different interpretation of the observed binary or graded affordance. Thus, based on their previous ratings of different hiking tours as more or less challenging, it is possible to deduce from one user's rating of a particular tour to its most probable evaluation by a different person (Ortmann et al. 2014).

### 2.1.4 Conclusion: Affordances for Modeling Human-Environment Interactions

This chapter approached the process of human perception of action potentials within their environment from a psychological perspective, and focused on Gibson's (1979) view of a direct pick up of functional information from the environment, with affordances as potentials for action which are determined by the interplay of properties of the agent as well as the environment. Since post-Gibsonian work on the formalization, measurement and representation of the concept, such as Warren's (1984) stair climbing experiment, or formalization proposals by Turvey (1992) or Stoffregen (2003), have further reduced its ambiguity, the usefulness of the affordance concept as a basic theoretical foundation for modeling human-environment relations has been increased. Thus, in the context of GISc, the notion of affordance as a fundamental cognitive element which connects an individual agent, its environment, and an action in a functional relationship has been identified as one potential approach to identify and ground the semantic primitives for spatial modeling. Moreover, the hypothesis of compound invariants allows for the conception of higher-order affordances as being constituted of lower-level affordances. In addition, the notion of agent-environment mutuality has assisted the incorporation of subjective differences in the user- or system-based interpretation of spatial information. Especially in the context of human behavior modeling, for instance in ABM, affordance-based approaches to agent-environment interaction design are a promising approach to develop agents with more complex reasoning and behavioral capabilities.

## 2.2 The Concept of Place

Taking the perspective of ecological psychology, the previous chapter described a theory concerning the processes of human perception of action potentials within the environment. In this context, space was conceptually restricted to the ecological level of the perceiving organism, meaning the narrow bandwidth of the physical world which affords sensory perception and consists of media, substances and surfaces (Gibson 1979). In this chapter, the scope is extended to the geographical scale by introducing the concept of place, which describes how humans perceive and structure their environment in a meaningful way. Due to the phenomenological perspective, the focus is put less on the cognitive processes of the individual but rather on the resulting meaning structures (Graumann 2002).

### 2.2.1 From Space to Place – A Conceptual Demarcation

Both space and place are central concepts in geography, yet marked by a conceptual elusiveness (Couclelis 1992). In the Merriam-Webster Dictionary, for instance, space is more or less defined as a synonym for empty area (Merriam-Webster’s online dictionary, n.d.). Regarding the use of the word place, May (1971) distinguished between four distinct uses by geographers alone, namely as a reference to the entire surface of the earth or a clearly defined unit of space such as a city, in the sense of a part of space and its occupying entities, or when speaking of an exact location or position. The fact that both terms describe formal concepts but, at the same time, refer to naïve geographical experiences, is certainly contributing to this ambiguity and source of considerable debate about their nature (Relph 1976, Cresswell 2009). Due to their conceptual nearness, demarcating place from space is necessary to approach a definition of place (Relph 1976).

#### *2.2.1.1 A Short History of Space and Place*

Although it can be assumed that humans have always to some degree systematically observed their environment in order to gain knowledge about its spatial phenomena, relations and changes, since antiquity, philosophers, mathematicians, physicists and geographers have discussed the metaphysical nature of space on a higher conceptual level (Couclelis 1999).

Among the first were Greek philosophers, who already distinguished between space and place (Raper 2000). When describing the formation of the universe in his *Timaeus*, for instance, Plato described an empty, limitless space, called the receptable or kenon, which merely provides a 3-dimensional space to be filled. This spatial substratum is then inserted with what might be called places, spatial locations or place holders for either static (topos) or roaming objects (chora) (Johansen 2004). Aristotle, a student of Plato, shared the view of an infinite, void space, but assigned an even more fundamental significance to places. Defining them as the extension of their occupying objects, places represent a fundamental prerequisite to the existence of all things and the only way to detect and describe motion (Raper 2000, Cresswell 2009). Aristotle's works had a great influence on early Greek mathematicians, thus several of his propositions were later included by Euclid in his detailed examination of geometry, a field which, however, had originated earlier in Egypt (Couclelis 1999). Abstracting the notion of a uniform, continuous space and formalizing fundamental elements such as point, line or polygon, Euclidean geometry dominated the field until the 19th century, when non-Euclidean geometries gained in popularity (Raper 2000).

According to Raper (2000), the 17$^{th}$ century marks an era when the concept of place gradually became insignificant due to the rise of Newtonian views on space. Newton, the founder of classical mechanics, based his notion of an absolute space on the theory of space being independent from body. Rejecting contemporary ideas which denied space and time the status as absolute entities, he argued in favor of space as a container for objects and events, which can be located based on a 3-dimensional Cartesian reference frame (Rynasiewicz 2014). His critics, with Leibniz leading the way, advocated the notion of a relative space since, as they argued, space does not exist detached from its objects, but only emerges from spatial relations between these entities (Couclelis 1992, Raper 2000). Thus, physical bodies are not contained in an otherwise empty space, as Newton proposed it, but rather set in a complex structure of relations between them (McDonoguh 2014). Due to the rise of classical mechanics, however, it were Newton's ideas which dominated scientific discourse about space over the course of the following centuries (Couclelis 1999).

An aspect which was shared by both Newton and Leibniz, however, was that of space being an objective reality, be it absolute or relative, but independent and detached from human experience. This notion was, however, compromised by Kant, who, comparing his approach to Copernicus' step from the geocentric, Ptolemaic system to Heliocentrism, argued that space is rather "subjective and ideal, and originates from the mind's nature" (Kant as cited in Janiak 2012). Since it is created in the process of a human interacting with the world, it is synthetic and depends on the properties of the human mind (Couclelis 1992, Golledge and Stimson 1997).

The notion of place, however, did not reappear as a philosophical concept until the rise of phenomenology in the early 20th century, when it was addressed especially in the writings of Martin Heidegger (Cresswell 2009). In general, phenomenology is an approach to study the human consciousness from a person-centered perspective, or, in a spatial context, the "comprehensive study of the lifeworld-that is, the world as it is lived and experienced" (Graumann 2002, p. 97). For Heidegger, one of the key contributors to phenomenology, the phenomenon of existing in the world, or dwelling, was closely tied to our place of being. This place, however, is much more than just a determinate location, but rather involves belonging, familiarity and involvement. This deep, inseparable connection between dwellers and their meaningful place in the world is described on the famous example of a traditional cabin in the Black Forest, where the influence of every aspect of life on the place's appearance is clearly visible (Heidegger 1993).

Influenced by Heidegger's work, in the 1960s, a growing number of geographers rediscovered the concept of place as a form of criticizing the positivist approaches of the quantitative revolution (Cresswell 2009). In contrast to the traditional interest in the analysis of aggregated human behavior, the focus was shifted to disaggregate behavioral research. Moreover, while the predominant view treated humans as rational actors in an objective, measurable space, it was now widely realized that humans base their spatial decisions not on such, but rather on a subjective world in their minds, a behavioral space in the sense of Couclelis (1992), which is generally based on a "naïve geography" (Egenhofer and Mark 1995, p. 1). This stream of thought, which postulates that the meaning of environmental objects varies among different people, for instance depending on their bodily condition, their gender or age, was at that time already established in phenomenological psychology, however, had an increasing influence on geography, architecture and design (Graumann 2002). Depending on their focus areas, this group of geographers eventually divided into a humanistic and a behavioral school, with the former being especially concerned with the dynamic, multi-dimensional and opaque character of human spatial experience, while the interest of the latter lay on the overlap with spatial cognition, which provided a conceptual basis for theories about human spatial decisions and behavior (Golledge and Stimson 1997).

Following the different traditions, geographers have developed different notions of space. Beck (1967), for instance, distinguished between the objective space of physics and mathematics, the ego space as an individual's interpretation of the former, and the immanent space in the sense of subjective, unconsciously perceived space, which provides a basic reference frame such as up and down or left and right. Relph (1976) discriminated between perceptual space, which refers to immediate needs or practices, existential space, which depends on our belonging to a particular cultural group, architectural space, which is deliberately created, cognitive space, an abstract construct created by mentally reflecting upon space, and finally abstract space, the space of logical relations among objects. Sack (1980), in contrast, focused on a scientific, objective view on one hand, and an unsophisticated, subjective view of space, which predominates everyday life, on the other hand. Alternatively, Couclelis (1992) proposed a taxonomy which differentiates between 5 types of space:

- Mathematical space: an abstract, formal description of space using principles of geometry, e.g. Euclidean, discrete or fractal spaces.
- Physical space: both Newtonian notions of absolute space as a neutral container for objects, and extensions originating from physics, describing relative space, whose structure influences the distribution of its objects.
- Socioeconomic space: an abstract relative space used in socio-economic models and spatial analysis, it is largely homogeneous but preserves spatial relations between social and economic activities.
- Behavioral space: the unique conception of the world or cognitive map that each individual has developed, it provides the basis for their spatial decisions, a subjective "world in the head" (Couclelis 1992, p. 226)
- Experiential space: the space as it is actually perceived by people which includes all intuitive forms of basic spatial understanding, such as sensorimotor or perceptual space

The phenomenon of place is allocated to the latter category of space by the author, arguing that "By now, space - space enriched with human experience - has become place" (Couclelis 1992, p. 230). In fact, it has been argued that, rather than attempting to strictly separate the two concepts, it is more feasible to speak of a space-place-continuum which stretches between the poles of objective, abstract and formal on the one hand and subjective, emphatic and intuitive on the other hand (Couclelis 1992, Edwardes 2007). Such a view of the different concepts can, to some degree, be interpreted as a consolidation of these different approaches, and is illustrated in figure 2.

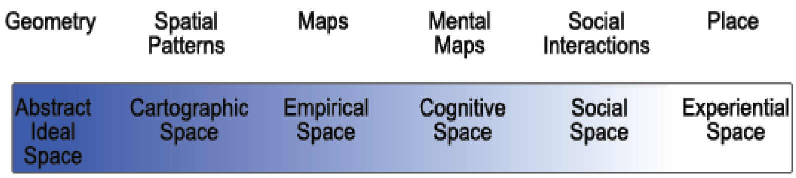


Fig. 2: The Space-Place Continuum
(Edwardes 2007)

#### *2.2.1.2 Defining Place*

Place is a central concept not only of humanistic geography but also of other disciplines such as urban design, architecture, and planning (Jordan et al. 1998). In geography, space is seen as undifferentiated and abstract, but, as soon as humans attribute it meaning and value, places come to existence (Tuan 1975). In his seminal work, *Place and Placelessness*, Relph (1976), for instance, defined places as "centers of meaning or focuses of intention and purpose" and, while still being part of it, set them clearly apart from space (Relph 1976, p. 43). Criticizing what he interpreted as a gradual loss of distinctive places by the design of placeless, standardized landscapes, the author emphasized the affective aspects of place, including feelings of belonging, identity and security.

With regards to geographic terminology, place needs to be differentiated from other, related terms, such as location, landscape or region. According to Relph (1976), while every place has a location, places are semantically richer, since they "are sensed in a chiascuro of setting, landscape, ritual, routine, other people, personal experience, care and concern for home and in the context of other places" (Relph 1976, p. 29). The term landscape, when being used in the context of human geography, is connotated with the emphasis of the tangible, visual-perceptual aspects of built or natural settings (Golledge and Stimson 1997). In accordance, Relph (1976) comprehends landscape as one aspect of place, which relates to its physical, visible form, and includes its natural or built features. Regions, finally, are distinctive areas of the earth's surface. What sets them apart from place, however, is the fact that while there is no restriction in the potential characteristics for delimiting regions, places are always endowed with human meaning (Gregory et al. 2011). Norberg-Schulz (1965), for instance, when structuring experiential space, clearly distinguished between larger regions and smaller places, the latter being of some significance and located within the former. Nevertheless, due to places being largely independent of scale, a region could be a place, as long as it is attributed meaning (Cresswell 2009).

In contrast to human geographers, however, architects, planners and urban designers are typically not interested in describing existing places, but rather in creating successful places. For some, the notion of place even implicates a built environment setting since natural places do not exist (Curry 1996). In urban design, for instance, there is a place-making tradition, which encourages practitioners to acknowledge places as physical and aesthetical units of the urban environment as well as settings for human behavior (Carmona et al. 2010). In general, there are large differences between place as defined by humanistic geographers and those who are concerned with making places. Although the former have acknowledged the ability of skilled architects and planners to create places, the pragmatic approach to the concept has been subject to criticism (Tuan 1979). Thus, architects and planners have been accused of, for the sake of feasibility, simplifying the multi-dimensionality of places to the aesthetic and functional dimension, or in other words, being solely interested in the action potentials of places and their visual appearance, while "real places are complex and messy" (Relph 1976, Carmona et al. 2010, p. 122).

### 2.2.2 Aspects of Place

Places are an integral part of how we perceive our environment, since they provide the building blocks used for structuring space and set the context for our daily actions and interactions. As it has been discussed, however, the exact conceptualization of place differs in complexity and pragmatism depending on the involved discipline. The prevalent view, however, is to distinguish between three aspects of places: the physical setting, the activities, and the meanings (Relph 1976). In the related literature, these three aspects are remarkably consistent, although sometimes conveyed in slightly different form or with differing terminology, for instance, by separating the setting, also referred to as locale, from location or integrating activities and meaning into a sense of place (Canter 1977, Cresswell 2009). These properties, however, are not clearly separated but mutually connected in causal relationships. Thus, the material objects at a certain location have an influence on the activities which the place affords. Activity patterns, on the other hand, can generate meaning, which, if temporally persistent, may influence the tolerated behavior at this place, but also its perceived footprint. Finally, intended meanings can be inscribed into the material topography of a place, for example in the form of a monument (Cresswell 2009). In the following chapters, each of the three place components is further discussed.

#### *2.2.2.1 Location and Setting*

A place and its location are inseparably connected. A location can be interpreted as an absolute point in space which can be clearly determined by an x, y, z coordinate tuple and allows geometrical analysis such as measurements of distances to other locations (Cresswell 2009). Places, however, rather refer to areal regions taken up in space, which include both natural and man-made structures (Casati and Varzi 1995).

Determining the exact location or spatial footprint of a place is challenging for numerous reasons. Thus, for instance, places are largely independent of scale, which means that the spatial extent of a place can range from the size of a corner of a room to the whole planet (Cresswell 2009). Again, the appropriate size of a place is discipline-dependent. While geographers prefer places to have the size of neighborhoods or public plazas, architects label spatial units as small as individual buildings, rooms, or separate vistas as places (Relph 1976, Tuan 1979). Moreover, places can be nested within larger places, such as Time Square within Manhattan, Manhattan within New York and so on, which further complicates the delimitation of individual place boundaries (Richter and Winter 2014). Due to them being the product of human interpretation and activity, their footprints do in many cases not adhere to administrative or other official boundaries, but are intrinsically vague and time-dependent (Curry 1996, Goodchild 2011). Rather, due to the subjectivity in spatial experience and knowledge, memories, activity patterns, preferences and perception, each individual human agent forms its own mental place representation, which is likely to differ in the exact location and spatial extent of a place (Golledge and Stimson 1997, Jordan et al. 1998, Goodchild 2011). Regarding the example of the footprint of the city center of Santa Barbara, for instance, Montello et al. (2003) demonstrated the inter-subjective variations among the representations of different test persons.

The mutual dependency relation between a place's location and its role as an action space, and therefore its affordances, is particularly strong. Thus, successful, collective places emerge from individual place formation processes when subjective action spaces converge at a location (Golledge and Stimson 1997, Carmona et al. 2010). Seamon (1980), for instance, spoke of "time-space-routines", such as walking to work or going shopping, which merge at central places to form a social "place-ballet", out of which arises the place's meaning (Seamon 1980, p. 158). In fact, the movement component of action spaces has a distinctive influence on the emergence of a collective place (Golledge and Stimson 1997). Thus, apart from its affordances, the relative accessibility of a place's location, which can be defined as "the inherent characteristic (or advantage) of a place with respect to overcoming some form of spatially operating source of friction (for example, time and/or distance)" is of relevance (Ingram 1971, p. 101). In fact, a place's central location in the movement network might be a better indicator for high usage frequencies than its design (Carmona et al. 2010).

Apart from the location of a place, however, one of its most apparent and observable characteristics is its material content, also referred to as its landscape, materiality, setting or locale (Relph 1976, Montgomery 1998, Cresswell 2009). Places consist of buildings, plazas, streets and sidewalks, parks or natural features such as mountains, hills, valleys or rivers (Relph 1976). In some cases, the dependency relation between the material setting and the footprint of a place is particularly strong, as in the example of walled towns, see figure 3 for an example, or valleys with clear natural boundaries, but more frequently, its delimitation is ambiguous (Tuan 1975).

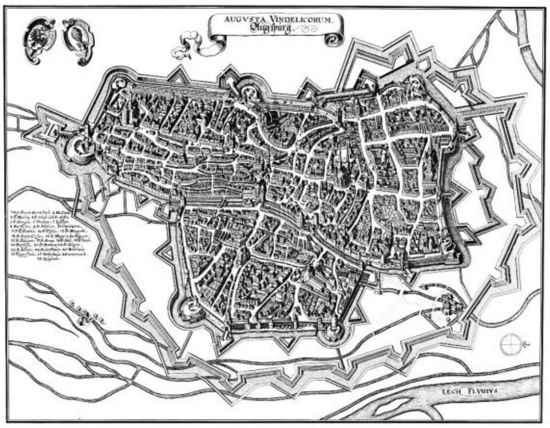


Fig. 3: Historical Map of Augsburg, Germany
(Merian 1650)

Most prominently, however, is the effect of a place's material setting on its affordances. Not only Gibson (1979), but also Heidegger (1993) and others emphasized that the human awareness of space is deeply grounded in activities and the practical value of things. Thus, the actions which are afforded by the material objects of a place define its usage patterns, and also its mentally represented footprint (Tuan 1975, Carmona et al. 2010). In an urban design and planning context, the question of how well the setting affords the intended usage patterns is always of concern for the process of creating new places (Carmona et al. 2010).

#### *2.2.2.2 Activity*

People interact with places on a daily basis. In the phenomenological approach, but especially in ecological psychology, the central role of activity for human spatial apprehension in general has been emphasized (Gibson 1979, Graumann 2002). Via their affordances, thus, the location, footprint and semantic meaning of a place can be defined (Relph 1976, Hart and Conn 1991, Goodchild 2011). In turn, however, the affordances of a place are determined by its material structure and its location (Cresswell 2009).

The use of the terms action, activity and behavior is often ambiguous. For Werlen (1993), however, the term behavior denotes purely mechanical body movements as a result of stimulation, while an action is intentional and conscious, and oriented towards the fulfillment of a particular goal. The term activity is used as a superordinate term which includes actions and behaviors. According to Leontiev (1978), in contrast, activities represent the highest level in a structured hierarchy with contributing actions and operations on the lower levels (Leontiev 1978). Concerning the nature of human-environment interactions, Tolman (1958), for instance, distinguished between three types of environmental attributes relevant for distinct behavioral operations, including for the discrimination between objects relevant for human orientation (discriminanda), the manipulation of objects (manipulanda), and a third group (utilitanda), which denotes the discriminanda and manipulanda which lead to others in a by-means of-relationship (Tolman 1958). Werlen (2000) described the embeddedness of spatial actions within complex cultural, social, economic, psychological and physical systems. Depending on the particular viewpoint, he distinguished between three models of action, which are not mutually exclusive, but rather emphasize different aspects of action:

- The purposive-rational model: models rational decisions made by a homo economicus based on objective spatial knowledge
- The norm-oriented model: models the ability of human actors to consider social and cultural norms when planning an action
- The action model of intersubjective understanding: models the ability of human subjects to constitute and communicate goals and meaning in the world

Depending on the relative influence of a particular aspect of action, the transactions between people and places can be different (Werlen 1993). Actions which focus on some sort of spatial optimization, for instance the identification of a suitable location for a business or a place to live, generally require purposive-rational planning. In this context, a typical strategy involves the formalization of the spatial reality (Werlen 2000). A place's characteristics, such as its location or material setting, are abstracted in order to allow for spatial calculations, for instance distance measurements or the conversion into monetary values. Places, however, are also subject to social and cultural norms, which often determine what actions are approved of or deemed inappropriate behavior. National states and their individual legislation, for instance, are a result of prescriptive, spatially bounded territorialization processes. Finally, actions have a strong social component, which can be referred to the notion of collectively shared meaning or affordances of places (Werlen 2000).

The actions conducted at places are manifold, and, from the perspective of urban design, can be related to comfort (e.g. finding physical relief by sitting and being sheltered from environmental influences, psychological and social relief by feeling safe and undisturbed), relaxation (e.g. by perceiving natural elements such as trees or water), passive engagement (e.g. people-watching), active engagement (e.g. socializing and interaction), and discovery (e.g. participating in events such as concerts or exhibitions) (Carr et al. 1992, Carmona et al. 2010). In general, the visual perception of the urban environment represents a purposeful action as well, and not a passive endeavor (Sell et al. 1984). Established urban design qualities such as imageability, visual enclosure, human scale, transparency or complexity are used to describe the aesthetic dimension of places (Ewing et al. 2006).

A particularly important role, however, is played by movement, which acts as a generator of urban life and activity. Especially in urban settings, pedestrian flows generate feelings of urbanity, activity and security, and are an important economical factor for local businesses. In order to attract pedestrian flows, places may be important destinations themselves (go-to-places) or located on the way to other places (go-through-places) (Carmona et al. 2010). The detailed infrastructural requirements for walking are discussed in chapter 3. 2. 2 of this thesis.

In accordance with the ecological notion of agent-environment mutuality, however, place affordances are perceived differently by different people (Golledge and Stimson 1997). There is an "indivisible unity" of place, person, time and act, resulting in the fact that people perform similar actions at different places or use one place in different ways (Wagner 1972, p. 49). Each person, thus, has its subjective action space, which, in this context, denotes the spatial extent of all locations he or she is aware of as potential places for actions, and an activity space, which is constituted of the subset of locations which are actually visited (Horton and Reynolds 1971). The variations of action and activity spaces are due to differences in spatial knowledge and the expected place utility (Wolpert 1964, Horton and Reynolds 1971). Thus, to be used by a person, a basic requirement for a place is to be known, meaning to be included in the person's cognitive map (Tolman 1948). The utility or expected level of satisfaction of a place user is relevant for the success of a place. Representing a determinate factor for the probability of a place to be part of a person's activity space, it is often indicated by the amount of time spent there (Golledge and Stimson 1997). The perceived place utility, in turn, depends on the suitability of this place, the appropriateness of its material setting with regards to a particular use (Malczewski 2006). The distinction of different uses or actions is vital, since, for instance, the suitability of a place with regards to movement will differ for a pedestrian in comparison to a car driver (Relph 1976).

#### *2.2.2.3 Meaning*

Due to the conceptual closeness of the two aspects, a clear separation of a place's meaning and its activity potential is not always possible. In general, however, the notion of a sense of place involves, apart from its activities, additional associations to place which are typically of affective and intangible nature (Tuan 1975, Relph 1976). Place identity has an individual component, for instance in terms of personal memories or feelings of belonging to or rootedness in a place, but can also be culturally constructed, such as London's Piccadilly Circus or San Francisco's Haight-Ashbury. In many cases, places derive their meaning from events, as in the cases of Tiananmen Square or Waterloo (Carmona et al. 2010). For urban designers, predicting the impact of the built environment on the meaning or sense of place is extremely challenging, since human experience and interpretation is hard to foresee. By adapting the urban infrastructure to the activities and needs of human users, however, the chances of a place to be successful, its potential, can be increased (Carmona et al. 2010). Since "individuality and distinction from other places" is critical for place identity, Relph (1976) interpreted the creation of standardized landscapes as indicators of placelessness (Lynch 1960, p. 6).

### 2.2.3 Conclusion: The Subjectivity and Functionality of Places

In this chapter, place formation was introduced as a mental process, which involves the meaningful structuring of space. Places come to existence when humans subjectively attribute complex meaning to locations, and typically refer to a vague, ambiguous spatial footprint with a physical setting. The role of a place as action space is particularly important for its formation and identity, which is why its semantic dimension has often been reduced to the functional perspective. Therefore, from an urban designer's point of view, a place's potential for success is mainly determined by its affordances, which result from the interplay of its users, actions and physical setting. Apart from the mere affordance, however, purposive-rational decisions are typically based on the perceived place utility, which guides the process of choosing an activity location. In addition to a place's suitability with regards to the action, however, its spatial accessibility is also critical for its potential usage. Although places are created in the mind of the individual, therefore, when activity spaces or shared human meaning converge, collective places emerge.

## 2.3 Towards the Integration of Place in GIS

Recently, place was described as "yet another fundamental cognitive concept that is missing in spatial information systems" (Richter and Winter 2014, p. 123). In fact, formal representations of space in current GIS differ profoundly from human spatial conceptualizations, as presented in the previous chapters, causing problems for the use of and interaction with GIS such as difficulties in handling place-related queries, challenges for database design, ambiguities in data integration and spatial analysis, and finally complications for modeling and analyzing human spatial behavior (Jordan et al. 1998, Abdelmoty et al. 2007, Winter and Freksa 2012, Gao et al. 2013, Scheider and Janowicz 2014). As a consequence, research on place-based GIS has become one of the major research directions in geospatial semantics (Janowicz et al. 2012). In this chapter, the current state of research is presented from the perspective of GISc.

### 2.3.1 Fundamental Challenges and Main Research Areas

Despite much research effort especially in the last decade, computational models of place are still at an early stage (Goodchild 2011). Recent work focused on such diverse topics as the formalization of place (e.g. Goodchild 2011, Abdelmoty et al. 2007), affordance-related approaches to represent places (e.g. Jordan et al. 1998, Scheider and Janowicz 2010, Scheider and Janowicz 2014), or the automated identification and localization of places from sources such as textual place descriptions (e.g. Khan et al. 2013, Scheider and Purves 2013, Vasardani, Timpf et al. 2013), geo-tagged pictures (e.g. Shirai et al. 2013), or movement trajectories (e.g. Ying et al. 2011). In general, the two main directions of research focus on the representation of place in GIS on the one hand, and the development of methods for automated place discovery on the other hand.

#### *2.3.1.1 Representing Place in GIS*

The particular challenges for a true integration of the concept of place into modern GIS arise from their foundation in an absolute, mathematical spatial model. Following the traditions of its originating disciplines such as cartography, computer-aided design, landscape architecture or remote sensing, GIS are typically based on an Euclidean spatial model, with vector and raster as the two standard data models (Jordan et al. 1998, Couclelis 1999). Thus, the inherited way of representing places is by attributing thematic information to specific coordinate locations (Jordan et al. 1998). This procedure, however, does not take into account the conceptual richness and complexity of places, and is often accused of excessive simplification for the sake of formalization and precision (Goodchild 2011).

In their work on place reference systems, Scheider and Janowicz (2014) provided an orientation with regards to the requirements imposed on place-based spatial information systems, when they list essential kinds of place inference.

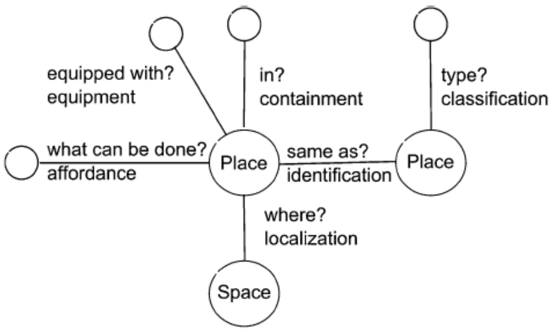


Fig. 4: Essential Kind of Place Inference
(Scheider and Janowicz 2014)

As illustrated in figure 4, relevant tasks include identifying and distinguishing places by name, applying classification schemes, determining what objects places are equipped with and whether these are within or outside the spatial extent of the place, localizing them in space, and finally, determining what actions a place affords (Scheider and Janowicz 2014). Since this thesis aims to contribute to the tasks of localization and affordance, they are discussed in the following.

##### *2.3.1.1.1 Indeterminacy of Location*

Of all aspects of a geo-object, GIS traditionally put the most emphasis on its geometry. The exact location of a spatial object can be defined as a binary relation it holds with its corresponding 2- or 3-dimensional region in space, such that other entities of the same kind are excluded from being located at the same spatial location at the same time (Casati and Varzi 1995). In GIS, the preciseness with which the location of a spatial entity can be represented merely depends on the maximum allowed number of decimal places for the storage of the coordinates, inherently implying, however, the existence of a distinct boundary, which is basically a linear interpolation between nodes (Goodchild 2011, Winter and Freksa 2012).

Hill (2000) listed several types of potential footprint representations of places, including point, bounding box, line, polygon, and grid, while explicitly stating the approximate character of each method. There are several approaches tailored to representing vague boundaries. Fuzzy logic, for example, defines the spatial extent of a region by use of a membership function with values ranging from 0 - 1 instead of a binary classification with crisp borders (Altmann 1994). To state another example, rough location, proposed by Bittner and Stell (2002), describes the location of a spatial object via mereological relations of its parts to a set of regions which form the regional partition of space, such as a mountain, which can thus be described as being fully located within a core region, partly located within a boundary region, and not located with respect to an exterior region (Bittner 1999).

In other approaches to representing the spatiality of places, however, it has been argued that the notion of boundaries in general is a concept of space, and therefore inappropriate for delimiting places, which, in reference to their material setting, “come in discrete (…) sets of objects” (Winter and Freksa 2012, p. 38). Thus, the spatiality of compound places is defined by the union of the locations of contributing objects and their relations in the geographic environment, and not via geographic coordinates (Winter and Freksa 2012). According to Goodchild et al. (2007), all geographic representations can be reduced to abstracted geo-atoms, which might be aggregated based on rules, such as uniformity, to form higher-dimensional geo-objects (Goodchild et al. 2007). Thus, the representation of a place in a GIS could potentially be based on geo-atoms, which are bound together as parts of a whole according to a unity condition (Simons 2000). Due to the prominent role of activity for place formation, a potentially useful unity relation might be functional unity, as explained by Guarino and Welty (2000) on the example of the different parts of a hammer. Apart from a composition of smaller parts, however, a place could also itself be a part of a larger geo-object, or in the sense of Goodchild et al. (2007), consist of a subset of its geo-atoms, such as, for instance, a particular area within a larger public plaza. Thus, a vector representation of place boundaries in GIS seems inappropriate, since “places have periphery rather than boundaries” (Winter and Freksa 2012, p. 38). At the same time, due to the fact that they are composed of discrete objects, such as buildings, trees or benches, as well as the undefined space within, places are not continuous fields. Moreover, due to the subjectivity of place perception, any concept of a footprint is inherently vague and prone to temporal changes (Goodchild 2011).

#### 2.3.1.1.2 *Affordance-Based Approaches*

As a reaction to the inherent vagueness and individuality in the process of place representation, and as an acknowledgement of the central role of places as action spaces, there have been affordance-based approaches towards modeling place in GIS (Jordan et al. 1998, Scheider and Janowicz 2010, Ortmann and Michels 2011, Scheider and Janowicz 2014).

Understanding places mainly as action spaces, Jordan et al. (1998) built on Gibson (1977) in their work on creating an affordance-based model of place in GIS (Jordan et al. 1998, p. 2). In reference to Warren (1984), and illustrated in figure 5, they identify the three aspects of affordances which must be modeled: the agent, the environment and the task. To determine, for example, whether a restaurant is suitable for a potential customer, the authors claim that it is necessary to note not only the agent's capabilities and preferences, but also the actual task, such as eating, socializing or reading. A combined analysis of these aspects is the only approach to realistically represent a place's meaning (Jordan et al. 1998).

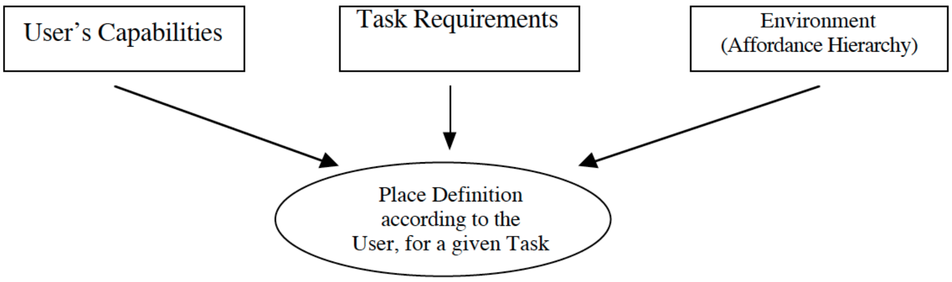


Fig. 5: Affordance-Based Model of Place in GIS
(Jordan et al. 1998)

Scheider and Janowicz (2010) aim to provide a basis for geo-ontologies by conceiving of places as media in the sense of Gibson (1977). In a similar fashion, a place can be traversed, has a location but can also move, and is bounded by a surface, for instance the walls of a building. Concerning the afforded action, Scheider and Janowicz (2010) propose the use of image schemas (Johnson 1987). In particular, the authors relate places to the containment schema, which denotes an inside-relation between an object and a place. As they argue, this conceptualization can be used as a basis for data-schemas of place in GIS (Scheider and Janowicz 2010).

The work done by Ortmann and Michels (2011) is not directly related to place, but to a related phenomenological concept, von Uexküll and Kriszat's (1956) Umwelten, and presents a theory to structure spaces based on human activities, which is why it is of relevance to this research. Inspired by von Uexküll and Kriszat (1956), the authors model activity Umwelten, physical-behavioral units which comprise all objects that are required for an activity. More complex, compound activities require the notion of compound activity Umwelten, which can also be nested within each other. Thus, a cycling Umwelt can exist within a driving Umwelt which can again be part of some higher-level Umwelt (Ortmann and Michels 2011). With their work, the authors aim to provide a conceptual basis for assistive systems, however, the notion of compound activity Umwelten has similarities to the place concept.

Finally, in recent work by Scheider and Janowicz (2014), the focus is on place reference systems. In analogy to spatial reference systems, the authors propose to reference places by localizing objects (locatums) which participate in simulated activities in relation to other objects (locators) involved in the activities. Postulating that "to be in a place is to act relative to certain referents involved in this act", place localization is based on these affordances, such that locatums serve to locate the locators in space (Scheider and Janowicz 2014, p. 103). Regarding a visibility affordance, for instance, the place consists of and is localized by all sites (locators) from which a concrete referent (locatum) is visible. Similar to the approach proposed by Ortmann and Michels (2011), it is also possible to combine simple affordances, such as *WALKING* or *SITTING*, to more complex compound affordances, for instance by concatenation, such as *STANDING* plus *WALKING* plus *STANDING* in order to describe the process of accessing a train from a platform (Scheider and Janowicz 2014).

As the brief review of related work has demonstrated, a potential approach to modeling places in GIS is provided by the use of affordances, which propose a departure from purely space-based notions. The particular strengths of the affordance-concept for this purpose can be listed as follows:

- The spatiality of places can be grounded in Gibson's (1979) notion of medium, substance and surface.
- A strong emphasis is put on the functional dimension of places.
- The possibility to formalize and measure affordances facilitates their integration and analysis in GIS.
- The semantic connection between an agent and its environment as described in the affordance theory corresponds to Heidegger's (1993) notion of the situatedness of place perception.
- Compound affordances relate to the compound character of places.
- The concept of agent-environment mutuality allows modeling subjective differences in and the individuality of place perception.

Ecological space, according to Gibson (1979), consists of media, substances and surfaces. When transferred to the context of places, aspects such as their locatedness, physical settings or affordances like insideness or traversability can be grounded in these spatial primitives (Scheider and Janowicz 2010). In addition, the fact that Gibson (1979) allocates human activity a central meaning for the entirety of visual perceptual processes parallels the importance of the functional dimension of places, which is common in phenomenological approaches. Due to work on the formalization and measurement of affordances (e.g. Warren 1984, Stoffregen 2003), the affordance-concept is a promising approach to integrate these largely intangible aspects in the "harsh and unforgiving" digital world of GIS, which requires high levels of precision and formalization (Goodchild 2011, p. 15). A further strength of the affordance concept is its emphasis of the observing agent's situatedness, which, as Heidegger (1993) emphasized, is a fundamental characteristic of place perception and formation.

A particular problem for traditional approaches to modeling place is posed by their compound character, which, however, can be approached based on the notion of compound invariants and higher-level, composed affordances. Accordingly, in approaches such as Ortmann and Michels (2011) and Scheider and Janowicz (2014), it is described on a conceptual level how basic actions could potentially be combined to form more complex activities, which then relate to compound spatial units or places. In both studies, the focus is on the representation of pre-defined places. The question of identifying and localizing unknown places based on the notion of compound affordances, and by connecting lower-level environmental units or object to higher-level places, has not yet been addressed. This, however, would be desirable in contexts such as a user querying the GIS for a particular place to perform some activity ("*Where can I go to sit in the shade and have a coffee?"*), defining and localizing an unknown place for spatial queries or analyses ("*Is my hotel located in Downtown?"*) or, in the context of human spatial behavior modeling, providing an artificial agent with the capability to autonomously identify places for the performance of spatial behavior ("*Why are some areas of the city more frequented by pedestrians than others?*"). Further, the issues of conceptualizing and modeling the actions on different levels of complexity, as well as representing their dependencies in composite sequences, for instance in the form of by-means-of- or either-or-relationships, have not yet been addressed. Also, the computational representation of affordances is still an unresolved issue, since the current practice of either attributing it to the environmental object (e.g. Kuhn 2001, Michael and Chrysanthou 2003) or representing it as an agent-independent dynamic field (e.g. Kapadia et al. 2009, Kim et al. 2011) does not correspond well to Gibson's (1979) original concept of agent-environment mutuality, as emphasized in the formalization of Stoffregen (2003).

Regarding the last point, the perceptual subjectivity of places, with the exception of Ortmann et al. (2014), it has so far been neglected, and is still missing in current GIS. The concept of agent-environment mutuality, however, can be used as a theoretical foundation to develop computational models of subjectivity.

#### *2.3.1.2 Automated Place Discovery*

Apart from the representation of places in GIS, research has also focused on the discovery of places in terms of extracting their location from data obtained from human communication or behavior. While we constantly refer to places by using place names or descriptions, or act in and interact with places, for instance by photographing them, moving in and to them, or otherwise use them as settings for our spatial activities, georeferencing places represents a fundamental challenge to spatial information systems (Vasardani, Winter et al. 2013). The development of methods for automated place detection and localization, therefore, is an essential step towards truly intelligent communication about places between users and computer systems, as well as a deepened understanding of the relationships between humans and their environment (Vasardani, Winter et al. 2013, Zhong et al. 2013).

##### *2.3.1.2.1 Inferring Places from Human Communication*

One possibility to obtain information about the location of places is from human communication in verbal or textual form. These data can include references to places either by their name, or by a spatial description.

The straightforward way of referring to a place is by its name. Place names are an essential part of our communication of spatial information, since they provide a qualitative reference system for locations (Liu et al. 2009). Information systems and services such as navigation or routing systems, web search engines, map services, public transport portals or trip planners for tourists need the functionality to link place names used in user queries to spatial entities stored in a database, in order to, for instance, retrieve their coordinates or other additional information. For this purpose, these systems are generally based on a gazetteer, a depository of georeferenced geographic names. In many cases, however, communication problems between users and the system arise due to the ambiguity of place names, caused for instance by the fact that a place name might refer to several locations, such as the name Bachem, which refers to three different cities in Germany. A single location, however, can also have multiple names, for example differing authoritative and unofficial names (Vasardani, Winter et al. 2013). Resolving this ambiguity is a major requirement for successful place localization based on their name. Approaches for allocating one of certain potential locations to a place name can for instance be based on the size, population or relative importance of a location (e.g. Li et al. 2003), the spatial context of a document (Rauch et al. 2003), or deduced from the spatio-temporal clustering of place names used in geo-tagged photos (Kessler et al. 2009). When place names are georeferenced, they are in most cases represented as points. In order to derive spatial regions, which, as has already been argued, are more realistic spatial referents to places, from a set of points with the same place name, several methods have been developed, such as using Voronoi diagrams (Alani et al. 2001), Delauney triangulation (Arampatzis et al. 2006), or density surfaces (Purves et al. 2005).

Another way of communicating about places is by describing them in terms of their spatial extension, which is in most cases done relative to another reference object. The human way of describing places, however, is qualitative rather than numeric, and marked by vagueness and uncertainty (Vasardani, Timpf et al. 2013). There have been several approaches to bridge the gap between natural language (NL) place descriptions and locations stored in a GIS, for instance using speech recognition in combination with models of qualitative spatial reasoning (QSR) in order to extract place names and spatial relations (Yao and Thill 2006), learning systems to interpret the notion of "near" (Robinson 1990) or combining vague regions with spatial constraints from relational information (Schockaert et al. 2005). In a recent study, a method was proposed to construct sketches of places from verbal descriptions (Vasardani, Timpf et al. 2013).

##### *2.3.1.2.2 Inferring Places from Human Behavior*

Apart from communication, systematic observation of our spatial behavior can also provide an informational basis for place discovery. Possible data sources for actual behavior include datasets of geo-referenced photographs and various data describing human activities in space, such as movement trajectories, traditional travel surveys, mobile phone usage, or public transportation data.

Georeferenced photographs are a potential source for discovering places of interest, and very present on photo-sharing websites such as Flickr or Panoramio (Vasardani, Winter et al. 2013). Thus, for instance, the density of Flickr photographs has been used in combination with land cover datasets to define the vague boundary of a place (Martins 2011). In other approaches the popularity of places was deduced from the frequency of them being photographed, in one study based on a large dataset of images (Schlieder and Matyas 2009) and in the other based on pictures taken by tourists in an experimental setting (Kremer and Schlieder 2012). Hollenstein and Purves (2010) used Flickr Photos to describe city neighborhoods, while Rattenbury and Nathan (2009) attempted to determine the coherence of place semantics from the tags of photographs.

Tracking, in the sense of automated mapping of x, y, z coordinate tuples at a predefined time interval, is one of the possibilities to capture human spatial behavior at a very detailed level (Hess et al. 2015). The positioning can be based on technologies based on Global Navigation Satellite System (GNSS) such as the NAVSTAR Global Positioning System (GPS), Global System for Mobile Communications (GSM), Radio Frequency Identification (RFID) or sensor techniques (Parent et al. 2013). Due to the rise of GPS-enabled devices such as smart phones or car-based navigation systems, a variety of sets of vehicle or person tracking data have been collected and are in many cases freely available for researchers and planners. There have been numerous approaches to infer from movement trajectories the location and usage patterns of places, such as Alvares et al. (2007), who created semantic trajectories by integrating the metric data obtained from GPS with the underlying geographic information. In their exemplary application in the context of tourism management, it is therefore possible to identify stops at certain locations such as hotels, sights or airports, and analyze tourist behavior at a detailed level (Alvares et al. 2007). The notion of semantic trajectories has been taken up by others, for instance Andrienko et al. (2013), who aimed to identify people's personal places of interest (POI) by detecting stops in their movement patterns as obtained from GSM and GPS-based trajectories, and spatially relate them to Twitter data for place interpretation, or in work done by Ying et al. (2011) on predicting the next location a user will move to based on semantic trajectory similarity and frequency of patterns. Additional work has focused on analyzing the spatio-temporal pattern of vehicle movement trajectories or GSM-data for the identification of functional regions within a city (Yuan et al. 2012, Rinzivillo et al. 2012, Zheng et al. 2012).

Apart from movement trajectories, traditional travel surveys can also be used for movement and activity analysis. Zhong et al. (2013), for instance, propose a method to detect functional urban centers based on travel behavior received via interviews. Analyzing the density and diversity of urban activity patterns in combination with urban form, the authors calculate a centrality index as well as an attractiveness index, which describe the relative importance of activity places within a intra-urban system. Zheng et al. (2012) focus on clustering individual persons based on their daily spatial activity patterns, and describe their group-specific behavior in particular sub-regions of a city.

Data on mobile phone usage has already been discussed in terms of GPS- or GSM-based positioning for trajectory generation. There are, however, also other potential use cases of such data, such as, for instance, in Ratti et al. (2010). Here, the authors focus on data of phone calls between geo-referenced locations in order to identify regions of human interaction, where the interactional links between people are particularly strong. Remarkably, on a study of Great Britain, the authors show that the derived regions correspond to a high degree to administrative areas.

As a final example of data sources for human spatial behavior, information regarding public transportation transactions are presented. Due to the increased usage of smart card payment systems in cities such as London or Singapore, detailed spatio-temporal information about customers movements are available (Ordonez et al. 2012). Analyzing the strength of passenger flows, for example, Roth et al. (2011) identify activity centers in London. Charkirov and Erath (2012) and Ordonez et al. (2012) combine smart card data with travel surveys and land use information to identify human activity patterns or people's work places, respectively.

#### 2.3.1.2.3 *Predicting Places*

So far, approaches have been discussed to discover places based on existing data about human communication about or usage of places. Such data, however, are often difficult or cost-intensive to obtain, and in many cases do not exist at all. For instance, in cases where the potential impact of newly planned developments or intended changes to existing spatial structures must be assessed, actual human reactions to these changes cannot be studied. Thus, there are many situations in which planners, without having access to actual data, are interested in predicting spatial patterns of human behavior, which might also include the formation of places. In Geodesign, a planning method which connects geographical analysis with the design process, for instance, the simulation and assessment of potential impacts of a proposed change to the environment is part of the planning which takes place in an iterative process (Steinitz 2012). Depending on the specific context, the full range of analytic capabilities of current GIS can be deployed to infer potential human behavior from spatial data. As Kwan (2000) argues, however, spatial analysis conducted in these contexts should be based on the subjective environment of individuals. Apart from agent-based simulations, which are covered in the next chapter, accessibility modeling and suitability analysis are particularly important with regards to detecting places.

In the previous chapter, accessibility has already been introduced as a highly relevant spatial quality of places. Having been a standard method in urban and transportation planning since the 1920s, a range of potential approaches to measure a location's potential for spatial interaction have been developed, for instance based on the distance of a place to opportunities or to all other places, or on measures of centrality with regards to the entire road network (Batty 2001, Batty 2009). Several GIS-based models use accessibility to assess a place's potential for development and activity (Batty 2001, Geurs 2014). Of particular importance for urban places, since it describes how people use urban space, is the theory of Space Syntax, originally developed by Hillier and Hanson (1984), which applies methods derived from graph theory to spatial networks in architectural or urban settings (Carmona et al. 2010). In numerous occasions, a strong relationship was found between the spatial configuration of the movement network and the relative frequency of pedestrians or vehicles (e.g. Penn et al. 1998, Schwander and Law 2012). Apart from the fact that cognitive processes are ignored, the purely topological approach of Space Syntax, which, for instance, does not incorporate the magnet effect of prominent destinations (e.g. large shopping centers), is often criticized (Carmona et al. 2010). Nevertheless, it still demonstrates the importance of a place's centrality in the network for its usage.

Perceived place utility has already been described as a determinate factor for the probability of a place to be part of a person's activity space. For our daily activities, we have several potential places to choose from, which each represent a feasible spatial alternative.

In any city center, for instance, there will be numerous places which afford the action *RESTING*, some, however, will be more accessible, provide more and better sitting accommodation, are quiet and so on, and therefore more suitable with regards to our individual needs. These are the places that will also be more frequently visited, and become true collective places (Carmona et al. 2010). Based on these insights, suitability analysis, a common application area of GIS in which spatial locations are ranked according to their appropriateness with regards to a particular use, can be used to detect potential places where activities will probably be performed. Originating from the traditional map overlay techniques deployed manually by landscape architects, current approaches are frequently based on GIS functionality (Malczewski 2006). An intuitive and easily implemented method is GIS-based overlay mapping. The standard procedure involves defining the intended use or activity, choosing, operationalizing and standardizing a set of appropriate suitability criteria, assigning relative weight factors in case of a weighted linear combination (WLC), and applying an overlay procedure (Eastman 1999, Malczewski 2006). The operationalization of criteria, which are generally received from empirical studies or expert knowledge, is a necessary step to transform them to measurable indicators and often involves a range of spatial analysis techniques. In the following standardization process, the criteria can either be collapsed to Boolean true/false-statements or maintained as continuous variables expressing various degrees of suitability, for instance on a numeric scale 0-1 (Eastman 1999). There are various possibilities to determine the weights assigned to the individual criteria, such as ranking or pairwise comparison, a method in which the relative importance of each criteria is received by comparing it to all others (Saaty 1980). The overlay process, finally, results in a combined suitability map, and is often based on map algebra techniques (Tomlin 2013). Such techniques are regularly incorporated into multi-criteria evaluation methods (MCE), processes which support spatial decision making in case of multiple criteria, yet have, however, also been criticized due to an alleged over-simplification of complex spatial problems and the reliance on untested assumptions about the independence of the chosen criteria (Malczewski 2006). Exemplary applications in urban contexts include numerous studies on the suitability for walking (e.g. Leslie et al. 2007, Brownson et al. 2009), bicycling (e.g. McNeil 2011, Jonietz and Timpf 2012), recreation (e.g. Kienast et al. 2012, Jonietz and Rathmann 2013), or tourism (e.g. Pareta 2013). Apart from overlay and WLC techniques, there are other potential approaches to MCE, for instance the ideal point method, which compares alternative solutions in terms of their distance to an hypothetical ideal solution (Pereira and Duckstein 1993).

Recently, geocomputational methods derived from artificial intelligence (AI) have been applied to suitability modeling and analysis, such as fuzzy logic (e.g. Wang et al. 1990), evolutionary algorithms (e.g. Brookes 1997), artificial neural networks (e.g. Zhou and Civco 1996), cellular automata (e.g. Wu 1998) and finally, artificial agents, which, however, will be discussed in more detail in the following chapter.

### 2.3.2 Conclusion: Methods for Representing and Localizing Places

Taking the perspective of GISc, this chapter focused on the challenges accompanying the development of digital representations of place. Motivated by the discrepancy between the spatial model generally used in GIS, which, in Couclelis' (1992) terminology, can be described as mathematical space, and the human way of mentally structuring space into places, prior work on representing and georeferencing places has been discussed. As a synthesis of the two preceding chapters, the affordance-concept was presented as a potential approach to modeling places in GIS which allows for the composition of functional places from lower-level spatial entities. So far, however, there has been no approach to generate, and therefore localize, unknown places based on their affordances. Place localization is mostly based on existing data on human communication or behavior, in the case of missing data, however, methods such as accessibility analysis and suitability mapping can be used to infer or predict the location of places. However, the subjectivity of place formation processes has so far not been incorporated into GIS.

# 3 Methodology

In this chapter, the methodological basis of this thesis is described. The first Chapter 3. 1 introduces ABM as an approach which allows for the development of disaggregate, microscopic models of human spatial behavior. Extra emphasis is put on the representation of inter-individual differences among agents, their complex action strategies and their environment.

In anticipation of the thematic scope of our case study, which is presented in Chapter 5, Chapter 3. 2 begins with a brief review of selected prior work on pedestrian movement simulation, with a particular focus on studies based on an ABM framework. The list of reviewed approaches is not exhaustive, since the main aim is to provide fundamental information about common methods, techniques and challenges of agent-based pedestrian models. This is followed by a review of empirical research on pedestrian-environment interactions, in particular micro-scale walkability.

## 3.1 Geosimulation with Artificial Agents

The previous chapter has emphasized the central role of subjectivity for spatial cognitive processes and the generation of meaning in the world, an aspect which, as has been shown, is still neglected in GIS. Accordingly, the integration of inter-individual differences in human spatial behavior modeling requires a disaggregate, microscopic approach. This chapter, therefore, introduces geosimulation, and especially agent-based simulation, as a methodological approach which allows the development of high-resolution models of geographical phenomena for the purpose of fostering and testing hypotheses about their functional principles and, ultimately, to solve practical problems situated in geographical space (Benenson and Torrens 2005).

### 3.1.1 The Agent Paradigm

With the rise of the digital computer in the 1950s, the potential of geosimulation as a tool has been noted by geographers and deployed for modeling and experimenting with hypothetical scenarios. An implicit prerequisite for traditional methods for modeling geographical systems such as multiple regression, location-allocation and spatial interaction models involves the treatment of their respective parts as aggregates in order to keep the analytic process simple and efficient (Batty et al. 2012). In the real world, however, geographic phenomena which are observable on the macro-scale emerge from the activities and interactions of elementary geographic objects, a fact which has led to criticism of said approaches and a partial turn towards bottom-up methods (Benenson and Torrens 2005, Crooks and Heppenstall 2012).

Geosimulation, in contrast, allows for a disaggregate, micro-scale perspective on modeling and represents geographical system behavior as a result of individual actions of a multitude of its basic elements. These, depending on the thematic context, can be individual humans, sub-groups of humans, objects such as buildings or cars, but also bacteria and other natural phenomena, or even abstract concepts like trends and ideas, which are each modeled as automata, computational objects which change their internal characteristics dynamically on the basis of pre-defined rules and external inputs (Benenson and Torrens 2005, Crooks and Heppenstall 2012). In the past, two types of automata have proven particularly useful for an application to geographical problems, namely cellular automata (CA), tessellated structures of spatially located static automata, and agent-based models (ABM), on which the focus will be in this chapter.

Artificial agents are a particular class of automata, which, due to additional functionality such as spatial mobility or the possibility to store more than one property, have been described as more sophisticated than their CA counterparts (Benenson and Torrens 2005). Restricting our view to applications which involve agents set in geographical space, in contrast to other, non-spatial agent-based systems such as expert systems or web crawlers, the possibilities are various, and include transportation modeling (e.g. Bernhardt 2007), archaeological simulation (e.g. Axtell et al. 2002), political science (e.g. Kollmann et al. 1992), economics and retail modeling (e.g. Heppenstall et al. 2007), epidemical spread prediction (e.g. Perez and Dragicevic 2009), pedestrian simulation (e.g. Torrens 2012) and many more.

#### *3.1.1.1 What are Agents?*

There is no unambiguous, universal definition of the term agent, but in fact, numerous explanations have been proposed to date which often differ in their degree of generality or specificity, concreteness or abstractness, as well as their thematic focus or discipline-specific viewpoint (Raubal 2001a). For Russel and Norvig (2003), an agent is “anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators” (Russel and Norvig 2003, p. 34). Their basic notion of an agent is illustrated in figure 6, and can be interpreted as a rather broad definition which includes not only artificial, but all kinds of agents, and formulates no restrictions concerning the nature of the demonstrated behavior.

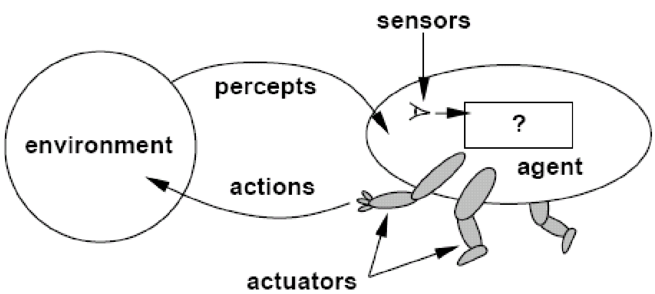


Fig. 6: Basic Agent Concept
(Russel and Norvig 2003)

Woolridge (1999), for instance, narrows the scope by explicitly speaking of agents as computer systems with the capability for behavior which is to some degree autonomous and goal-directed. A further distinction should be made regarding the qualification of an agent as intelligent. Thus, the author introduces behavioral flexibility as a further qualification for an intelligent agent, meaning that, in addition to autonomy, an agent should have the ability to respond to changes in the environment and interact with other agents (Woolridge 1999). In a similar fashion, Russel and Norvig (2003) speak of a rational agent as one who is capable of selecting an action in a manner which, on the basis of the agent's current knowledge, is optimized with regards to a performance measure, a term the authors use to describe the desirability of environmental states resulting from the agent's behavior. For this task, the agent's knowledge can be either built-in or created from its percept sequence, which includes the entirety of its previous perceptions. A distinction can be made at this point with regards to the extent of the agent's knowledge, which may comprise all the other model elements or be bounded by restrictions such as the range of sensory inputs. Focusing on geosimulation, the following set of qualities of artificial agents have been stated (Benenson and Torrens 2005, Crooks and Heppenstall 2012, Abdou et al. 2012):

- Heterogeneity: agents can differ in their properties, goals or behavioral rules
- Proactiveness: agents act to realize certain goals
- Perception: agents perceive their environment
- Interaction: agents interact with each other or the environment
- Adaption: agents can change the rules of their behavior based on their percepts
- Memory: agents record information regarding previous states or actions

In general, however, the notion of an agent should not be interpreted as an absolute category, but rather as a general concept for the development of models and software (Benenson and Torrens 2005).

#### *3.1.1.2 A Formal Description of an Agent*

Formally, according to Woolridge (1999), a standard agent can be described as a function

$$action: S^* \rightarrow A \qquad (2)$$

where a set of states $S = \{s_1, s_2, ..., s_n\}$ of the agent's environment are mapped to the agent's actions with regards to its action capabilities $A = \{a_1, a_2, ..., a_n\}$. The sequence of environmental states encountered by an agent provide the basis for its behavioral decisions. Apart from a purely reactive agent, which responds directly to its environment by mapping a single state to an action (for instance a thermostat), an agent architecture can also incorporate perception and action subsystems such as a function

$$see: S \rightarrow P \tag{3}$$

where environment states are mapped to percepts, and, resultantly

$$action: P^{*} \rightarrow A \tag{4}$$

where sequences of percepts are mapped to actions. A state, however, is not a quality restricted to the environment, but rather, applies to agents as well. Figure 7 shows the resulting model, where a set of internal states $I$ of an agent is mapped to actions

$$action: I \rightarrow A \tag{5}$$

and a state and percept are used to update the internal state in a function $next$

$$next: I \times P \rightarrow I \tag{6}$$

so that an agent has the ability to choose its behavior based on its initial state and the input of new percepts which lead to a state transformation.

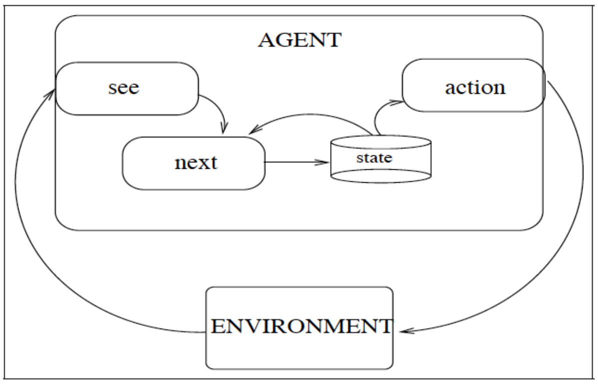


Fig. 7: Architecture of an Agent with State
(Woolridge 1999)

#### *3.1.1.3 Types of Agents*

There are different ways to implement the formalisms discussed in the previous chapter. Russel and Norvig (2003), for instance, distinguish between four types of agents:

- Simple reflex agents: agents which behave merely on the basis of the current percept, such as the purely reactive thermostat agent described in Woolridge (1999).
- Model-based reflex agents: agents which have some sort of model of the world, which enables them to use their percept sequence to infer states of the environment which were not actually perceived and mapped to percepts.

- Goal-based agents: agents which select their actions with regards to achieving a particular goal, a situation which is somehow desirable.
- Utility-based agents: agents which, in addition to a goal, plan their behavior according to a utility function. Due to their thematic relevance with regards to this thesis, they will be described in more detail in the following.

The introduction of a utility function as an additional behavioral parameter can be interpreted as an extension of purely goal-based agent architectures. Instead of a particular goal, which the agent can either achieve or not, the allocation of utilities to different world or agent states, and the according action sequences, enables the agent to plan its behavior with regards to maximizing the expected utility of the outcome. The utility function, therefore, is basically an internalized performance measure and allows for much more complex behavioral strategies than purely goal-based behavior. Formally, a utility-based agent $A$ has the ability to calculate and compare the utilities $U(A, C_i)$ of each member of a set $\{C_i\}$ of potential choices regarding desirable state changes or behavioral sequences. If the utility values of each member of the choice set are known, the agent can use these as basis for its decision-making (Russel and Norvig 2003).

### 3.1.2 The Agent Environment

Traditionally, the main efforts in the development of ABM have been placed on the representation of the agent's behavior and characteristics, and less on modeling the environment in which it is situated and operates (Stanilov 2012, Crooks and Heppenstall 2012). Still, however, the possibility to model complex interdependencies, mutual influences and feedbacks between an agent and its surrounding environment has been described as one of the main strengths of ABM (Parker et al. 2003). Depending on the scope of the simulation model, one can distinguish between spatially implicit or explicit agents. In case of the former, which can often be found in the context of social or cultural simulations, their actual location is irrelevant for the simulation outcome, which is why spatial relationships are largely neglected. The latter, in contrast, explicitly consider the location of an agent in geometrical space, which represents an essential factor for the modeled system, and are often focused on the analysis of geographic phenomena (Stanilov 2012, Crooks and Heppenstall 2012).

One of the challenges when developing a spatially explicit ABM is to create an appropriate environmental model. Russel and Norvig (2003) list general characteristics of agent environments:

- Fully observable vs. partially observable: depends on the agent's ability to sense the state of the whole environment or have a segmented perception
- Single agent vs. multi-agent: depends on the number of agents existing simultaneously in the environment

- Deterministic vs. stochastic: depends on the degree of predictability of the environment's state based on its current state and the agent's actions
- Episodic vs. sequential: depends on the degree of influence an agent's action sequence has on its current action
- Static vs. dynamic: depends on the changeability of the environment's state during the simulation
- Discrete vs. continuous: depends on the representation of states, time, and the agent's percepts and actions as discrete or continuous phenomena
- Known vs. unknown: depends on whether the outcomes of actions are known to the agent or the designer of the model

Naturally, the majority of these characteristics should be understood as extremes on a scale with various intermediate stages rather than mutually exclusive alternatives.

In addition to the previous criteria, there are several extra considerations to be made in the context of geosimulation. On the one hand, this is due to the complexity of geographical processes, while on the other, there is a causal relation to the specific requirements of geographical information modeling. Concrete modeling concerns include issues of scale, type of agent-space relationships, and the choice of a raster- or vector-based data model (Stanilov 2012, Benenson and Torrens 2005).

The issue of scale is an important aspect for spatially explicit ABM. As Dungan et al. (2002) criticize, the term scale is often used as a synonym for differing concepts, such as extent, grain or resolution. Nevertheless, all these aspects are to be explicitly considered in the context of ABM. Thus, the spatial extent of a study area has been shown to be a sensitive factor which can drastically alter the results of spatial analysis processes, as it has been demonstrated on the examples of calculating landscape metrics and spatial statistical analyses (Saura and Millan 2001, Dungan et al. 2002). For ABM, a particular challenge is posed by the fact that the spatial area at which an agent's behavior takes place is not always corresponding to the spatial extent of the processes and phenomena which influence and are influenced by the behavior (Stanilov 2012). The grain size describes the size of the smallest, fundamental sampling unit, and is closely related the term resolution, which, however, also incorporates the precision of the attribute measurement scale (Legendre and Legendre 2012, Dungan et al. 2002). Since, as we have argued, one strength of ABM is its potential to develop disaggregate, micro-scale models of geographical phenomena, there is a tendency towards fine-grain representations of the environment, contributing, however, to the fact that ABM, in comparison to other comparable methods, is especially "data-hungry" (Benenson and Torrens 2005, p. 13). Further, a fine-grain resolution might lead to inefficient computational load as well as produce unrealistic, fragmented results (Menard and Marceau 2005).

Another aspect to be considered is the question of how the relationships between agents and the environment are conceptualized. In principle, one can distinguish between one-to-one relationships, where an agent is associated with only one spatial object, such as a household and its place of residence, and one-to-many relationships, such as a household and a set of places, including residence, workplace or school locations. Further, the agent-environment relationship can be unidirectional, such that one entity, either agent or environmental object, is affected by but may not affect the other entity, or multidirectional, which allows cyclic interactions and feedbacks between agent and environment (Stanilov 2012).

Concerning the data model, a regular tessellation of space, particularly as a grid cell structure, is used by a majority of existing ABM, which can partly be explained as being inherited from grid-based CA (Benenson and Torrens 2005). While the raster data model is particularly well-suited for the representation of continuous geographic fields, however, discrete objects are more appropriately modeled as vector-based structures. Since both types of spatial phenomena exist in the real world, identifying the best approach can be challenging, and always depends on the thematic scope of the study. In general, however, Stanilov (2012) favors the use of vectors with reference to traditional pitfalls of raster-based modeling, such as the influence of cell sizes on the results, the problem of cell heterogeneity when objects are smaller than the cell size, and the potential need to define more complex neighborhood relationships than simple distance-based concepts.

In general, in the effort of maturing ABM from a purely experimental application to a predictive, testable tool, it has been recognized that environmental models, too, must evolve from being abstract and theoretical to more realistic and detailed representations (Stanilov 2012). Recent developments, such as an increased data availability or a steady rise in computation power, support these efforts.

### 3.1.3 Agent-Based Modeling of Human Behavior

Depending on the real-world system to be modeled, agents, as we have argued, can represent a variety of entity types (Benenson and Torrens 2005). In many cases, however, they refer to acting humans as elements of a geographical system. Modeling the behavior of individual agents, however, due to its complexity, heterogeneity and variety of potential influences, is a non-trivial issue, which needs to be based on insights derived from psychology. Since the cognitive revolution in the mid-1950s, which included a departure from purely stimulus-response-based explanations of human behavior towards a computational theory of mind which involves inputs, processes and outputs, two distinct views of human behavior modeling have been developed in AI and cognitive science. Therefore, while AI focuses on replicating human cognitive strategies in order to equip technical systems with our strengths in problem solving, cognitive science pursues to examine human cognition in its entirety, including non-rational or erroneous aspects of our behavior (Kennedy 2012). Accordingly, depending on the specific perspective, the aim in designing artificial agents may be either to enable them to solve problems in an intelligent way, or mimic human behavior.

#### *3.1.3.1 Approaches to Modeling Human Behavior*

According to Kennedy (2012), there are several basic principles for human behavior modeling. Thus, our actions are based on processed sensory information, clearly motivated, as expressed by conceptual frameworks such as Maslow's (1954) hierarchy of needs, and rational. In addition, human behavior is also partly emotional, intuitive and unconscious, as well as influenced by social aspects (Kennedy 2012). In terms of current modeling approaches, Kennedy (2012) distinguishes between:

- mathematical approaches: Mathematical simplifications, such as random number generation or threshold values, are used to trigger an action sequence.
- conceptual frameworks: These abstract concepts include belief-desire-intention (BDI) frameworks or PECS (Physis, Emotion, Cognition, Social Status), and describe agent reasoning processes.
- cognitive architectures: Such models explain the actual cognitive functioning of an individual, and are developed with the aim to understand human cognition itself, for example SOAR (Laird et al. 1987)

Due to its relevance in the context of this thesis, the BDI approach, which is still one of the most popular frameworks of practical reasoning agents, is described in detail in the following (Rao and Georgeff 1995). Practical reasoning itself is concerned with deciding which action to perform in order to achieve a certain goal, and involves choosing goals (deliberation) and the appropriate actions (means-end reasoning) which are expected to lead to the goals (Woolridge 1999). The BDI framework distinguishes between three mental modules of an agent, including its beliefs in terms of its knowledge of the world, its desires or goals, and its intentions, which can be described as its deliberative states (Rao and Georgeff 1995). Figure 8 shows a schematic diagram of a BDI architecture. The agent's percepts are generated via its sensors, and, together with its current beliefs, are used to update the belief knowledge base in a belief revision function (*brf*). On the basis of its actualized beliefs and current intentions, the agent then generates options or desires which are available to the agent at that time, filters them in a deliberation process in order to update its intentions on the basis of its current beliefs, desires and intentions, and, finally, infers an action to be performed (Woolridge 1999).

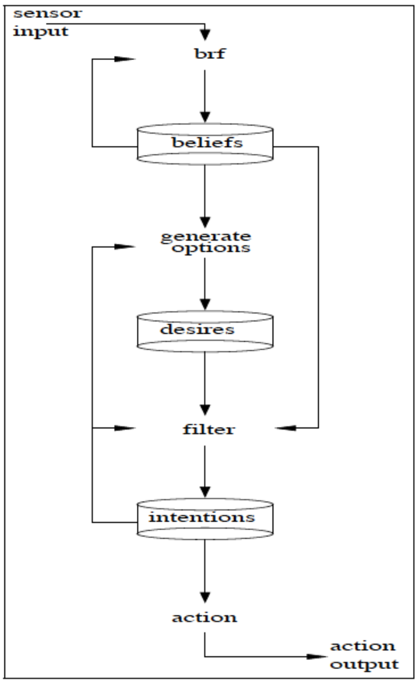


Fig. 8: A BDI-Architecture

(Woolridge 1999)

The BDI model is based on work on observations of human practical reasoning, and therefore intuitive (Bratman 1987). Further, the clear-cut separation of deliberation and means-end reasoning allows for more complex behavior to be modeled, such as the weighing of trade-offs between conflicting desires, or reorientation if a goal has become impossible to achieve (Woolridge 1999). Due to its generality, however, the BDI framework can be implemented in different ways and only provides a broad orientation schema for modeling human behavior.

#### *3.1.3.2 A Few Words on Searching Solutions for Problems*

According to Russel and Norvig (2003), rationality involves the ability of an agent to perform means-end reasoning in a manner which is somehow optimized with regards to a certain performance measure. In other words, an agent should be capable of developing an action strategy which results in reaching a particular goal, while at the same time maximizing the performance measure (Russel and Norvig 2003). For AI, developing methods for problem solving which mimic or even outperform human capabilities is a key challenge. In the following, therefore, some fundamentals regarding such strategies are briefly presented. It should be noted that, despite the conceptual richness of this particular topic in AI, the discussion is restricted to the most basic case of a search problem in a fully observable, discrete, and deterministic environment which is known to the agent.

According to Russel and Norvig (2003), a problem can typically be abstracted to five components:

- initial state: the present state of the agent
- set of possible actions: the set of actions possible for the agent in the initial state
- transition model: a description of the resulting state changes for each possible action; together, the three preceding components form a state space, which can be represented as a directed graph, with states as nodes and actions as edges
- goal test: a procedure to test each state for being a goal state
- path cost: a path through the state space graph is a sequence of actions and states; each path is typically assigned a numeric cost

Based on the state space, an agent can choose a solution, which is represented by a path through the graph and denotes the sequence of actions and states which is expected to lead to a goal state. A cost-optimized path represents an optimal solution. There are several algorithms for the identification of a solution, such as breadth-first or several depth-first search strategies, and some which identify the optimal solution, such as the Dijkstra- or the A*-algorithm (Russel and Norvig 2003). Apart from such uninformed search methods, which are restricted to the problem definition, there are also informed search methods which apply heuristic functions to come up with a solution. These, however, are omitted here.

#### *3.1.3.3 Modeling Actions – Activity Theory*

So far, it has been discussed how action strategies, as solutions to the problem of reaching a goal state, can be represented and identified by means of a graph structure. Actions, however, are not trivial to model, but rather characterized by complexity and interdependencies. A common problem for researchers dealing with actions, for instance, is to unambiguously decide whether an agent's behavior consists of just one or more actions. This is generally referred to as the action individuation problem (Trypuz 2008). As an example, Kemke (2001) provides the following three action descriptions: *MOVING ONES FINGER IN A CERTAIN WAY TO PRESS THE LIGHT SWITCH*, *SWITCHING THE LIGHT ON* and *LIGHTING A ROOM*. Adopting a fine-grained view, these actions can be interpreted as separate actions, since they might have different modal, temporal and causal properties (Trypuz 2008). In case of our example, accordingly, the three actions are connected in "by-means-of" relationships, meaning that one action is performed by doing the other (Searle 1983, p. 128). Thus, *LIGHT A ROOM* is done by *SWITCHING THE LIGHT ON*, which is in turn accomplished by *MOVING ONE'S FINGER IN A CERTAIN WAY*. In contrast, a coarse-grained view takes all three actions as different descriptions of one and the same action (Trypuz 2008).

Activity theory, a conceptual psychological framework which was developed by psychologists in the former Soviet Union, most notably Leontiev (1978), represents a useful approach to cope with the action individuation problem (Jonietz and Timpf 2013a). Thus, one of the basic principles of this conceptual system is the three-level hierarchical structure of activity (Leontiev 1978). According to the author's terminology, the highest hierarchical level is represented by activities which are oriented towards motives that correspond to basic human needs. In order to execute an activity and fulfill a motive, however, it is necessary to perform a number of separate actions which follow subordinate goals and are, in turn, realized by lower-level actions with subordinate goals. When the lowest hierarchical levels of activity are reached, unconscious, automated processes take place. This is the level of operations, which do not follow specific goals, but serve only to implement the corresponding actions (Leontiev 1978).

Until today, activity theory and the notion of hierarchically structured actions has served as a valuable theoretical basis for work for instance in the context of human computer interaction (HCI) (e.g. Nardi 1996), ontologies (e.g. Kuhn 2001, Kemke 2001) and GIS (e.g. Timpf 2003). In an approach very similar to Leontiev's work, Kemke (2001), for example, differentiates between different levels of abstraction when describing actions:

- the realization level
- the semantic level
- the pragmatic level

Actions on the realization level of abstraction are described in terms of their physical, motoric realization, such as the bodily movement of *MOVING ONES FINGER IN A CERTAIN WAY*. The semantic level of abstraction refers to the environmental effect as the outcome of the action, such as *SWITCHING THE LIGHT ON*. Rather than the actual realization or the motivation for acting, the resulting state change regarding the environment or the agent is described. Finally, on the pragmatic level of abstraction, a direct reference to the intended goal of acting is given. The action description *LIGHT A ROOM* would be an example for this level of abstraction, and can be referred to a goal state (Kemke 2001).

When modeling actions, therefore, it is necessary to acknowledge their hierarchical structure. A three level structure as proposed by Leontiev (1978) or Kemke (2001) helps to identify the subordinate actions which contribute to the execution of a higher-level action and can also be transferred to the context of modeling higher-order affordances.

### 3.1.4 Verification and Validation of Agent-Based Models

Model verification and validation are two important steps in the development of an ABM, and among the greatest challenges (Ngo and See 2012, Crooks and Heppenstall 2012). In this specific context, verification refers to testing whether the internal logic of the model is consistent and it behaves as originally intended by its developers. This is normally followed by validation, an evaluation of whether the model's output fits the observed behavior of the corresponding real world system in a satisfactory way (Batty et al. 2012, Ngo and See 2012). In other words, "model verification deals with building the model *right*. (…) Model validation deals with building the *right* model" (Balci 1998, p. 135).

In the literature, however, the two terms are not always used unambiguously. Thus, for instance, verification has been defined as testing the "inner validity" of a model (Crooks et al. 2007, p. 10). An alternative definition conceptualizes it as a subordinate part within the process of structural validation, the testing of whether the internal functioning of the model corresponds to the real system, and as a superordinate term for face validation and sensitivity analysis (Parker et al. 2003, Ngo and See 2012).

Apart from the modelers themselves, testing the face validity often requires the consultation of domain experts, who are asked to visually assess the model's behavior as logical and appropriate. This is typically done either at an early stage of model testing, or in case of a lack of real-world data against which the model could be tested statistically (Ngo and See 2012, Evans 2012). By conducting multiple model runs with varying initial conditions or parameter settings, a sensitivity analysis systematically tests these changes for their effect on the model outcomes (Crooks and Heppenstall 2012). This is done in order to identify non-effective and therefore unnecessary parameters, which can then be excluded from the model, or to test the overall robustness of the results. Due to its wide use, numerous methods have been developed for this purpose, a general distinction can be made, however, between local and global sensitivity analyses. While the former estimates the sensitivity of each model parameter separately by altering their values sequentially with the others kept constant, a global sensitivity analysis examines the effect of combined changes over the entire parameter distribution (Hamby 1995). In the next steps, the model is calibrated by identifying a unique parameter setting which maximizes the goodness-of-fit, and finally validated sensu stricto by comparing its output against a real-world dataset, a process which typically involves the use of statistical methods (Ngo and See 2012).

Keeping in mind that ABM model geographical processes as phenomena emerging from the behavior of individual agents on the micro-scale, it is not surprising that traditional model validation techniques are in many cases not feasible. Thus, for instance, there are often no data available which fit the required level of detail. Further, some processes, for instance human spatial choice and behavior, involve non-observable components which cannot be tested against real-world data in a traditional sense. There is still discussion between those who interpret this as a general limitation of ABM as a research method, and others who emphasize the strengths of ABM as a "tool to think with" rather than to make exact predictions about geographical processes (Crooks and Heppenstall 2012, p. 90). Batty et al. (2012), for instance, emphasize the fact that theories which are plausible but, due to a lack of data, untestable by classical means, should not be distorted or omitted. Instead of overly fitting the model to the available data, the potential of ABM to explore possible futures and alternative systems should be preserved by focusing on its plausibility, sensitivity and face validity (Batty et al. 2012, Crooks and Heppenstall 2012).

### 3.1.5 Conclusion: Agent-Based Models of Subjective Spatial Behavior

This chapter introduced ABM as a powerful tool for geosimulation, and, widening the scope to the previous chapter, as a potentially well-suited methodological approach to model place with affordances:

Firstly, geosimulation in general applies a disaggregate, microscopic perspective on modeling geographic phenomena and processes. In combination with the heterogeneity of agents as a core concept of ABM, this allows for the incorporation of the subjective, individual component of perception and action which is emphasized in both the affordance and the place concept. Thus, individual differences can be incorporated in ABM for instance by attributing different attributes to agents, or by changing the function which controls the mapping of environmental states to percepts and percepts to actions, or in the case of a BDI framework, sensory inputs to beliefs and the resulting intentions, or different desires.

Secondly, agents can be developed to create and follow individual action strategies, depending on their percepts. By means of graph structures, complex hierarchical action models, and utility-based optimization heuristics, the modeling of multifaceted human activity is possible, a fact which acknowledges the centrality of activity for ecological psychology and place formation.

Thirdly, the combination of GIS and ABM allows for the creation of a detailed model of the environment, which is perceived and acted upon by a situated agent, and necessary for modeling spatial perceptual processes. In terms of data models, a raster-based approach particularly corresponds to Gibson's (1979) notion of surfaces. A practical problem for such detailed models is model validation, the need for which, however, has been relativized to a certain degree in the past.

## 3.2 Research on Pedestrian-Environment Interactions

Whereas the previous chapters 2 and 3. 1 have presented the theoretical background and the methodological basis for the development of a computational model of functional place, which is discussed in Chapter 4, this chapter narrows the methodological scope in anticipation of the case study presented in Chapter 5. Thus, the topic of pedestrian modeling is introduced, with a particular focus on shortcomings of current approaches which are addressed with the methods developed in this thesis.

### 3.2.1 Pedestrian Simulation – Approaches and Methods

Although being long neglected by transportation planners and scientists alike, since the early 1990s, the topic of pedestrian traffic has experienced a remarkable renaissance of interest (Handy et al. 2002). Following several decades of almost exclusively catering for the needs of motorized transport, a rethinking process was triggered by the negative consequences of excessive car use becoming evermore apparent, including traffic congestions, increased levels of pollution, unsustainable consumption of resources, effects on climate change, the need for costly infrastructure, the degeneration of urban areas, social exclusion and health risks due to low levels of physical activity (Cervero and Kockelmann 1997, Handy et al. 2002, Frank and Engelke 2001). Walking, in contrast, is environmentally friendly, requires less infrastructure, is affordable for everyone, healthy, and contributes to lively communities with lower levels of crime and social decay (Litman 2010). In general, the modeling and simulation of pedestrian behavior and the resulting movement patterns receives interest from the retail industry, computer science, emergency services, urban and transportation planning, geography, urban designers and architects (Haklay et al. 2001).

The traditional disregard of walking in transportation planning and modeling can partly be explained by their original focus on automobile travel. Thus, similar to geographical modeling in general, car-based approaches were marked by a macro-scale view and the simulation of aggregates, while naturally, pedestrian movement appears to happen on much finer scales and on a disaggregate level. However, as Batty (2001) notes, since the very first development of traffic models in the 1950s, there have been attempts to incorporate the pedestrian realm, all of which, however, follow the tradition of an aggregated view on pedestrian movement while at the same time ignoring or drastically simplifying the cognitive-behavioral component of walking (Batty 2001). Fueled by trends such as a conceptual turn towards disaggregate, bottom-up approaches to modeling, the recognition of the limitations of predictability, as well as technological innovations resulting in increased computational power and the improved availability of pedestrian movement data on the micro-scale, the first agent-based pedestrian models appeared (Kwan 2000, Batty 2001).

Despite their undeniable improvements, however, there are two shortcomings of present-day pedestrian ABM, which have been subject to criticism in the past, and are explicitly addressed in this thesis. Firstly, despite the recognition that pedestrians are heterogeneous in terms of their physical ability, social roles and economic constraints, in the majority of models, they are still treated as a homogeneous group (Buchmueller and Weidmann 2006). There are, however, also exceptions which are further discussed in the following (e.g. Kerridge et al. 2001, Haklay et al. 2001, Durupinar et al. 2008). Further, since empirical research has demonstrated the richness and variability of potential environmental influences on pedestrian behavior, manifested for instance in their route choice, there have been demands to incorporate walkability into ABM (Zacharias 2001b, Johansson and Kretz 2012). Apart from a distinction between the sidewalk and the street, however, most models drastically restrict the complexity of their environmental models, as well as the cognitive abilities of the pedestrian agent, thus focusing on merely distance-minimizing and obstacle-avoiding agents.

#### 3.2.1.1 *Overview on Approaches to Pedestrian Simulation*

One of several possible ways to classify the existing approaches on pedestrian simulation is by their level of abstraction. Macroscopic models typically focus on modeling aggregate movement patterns such as overall pedestrian densities or flows (Johansson and Kretz 2012, Torrens 2012). Of these, several models are inspired by observed similarities between pedestrian flow patterns and physical phenomena, with examples including relations to the behavior of fluids or gas (e.g. Henderson 1971, Chenney 2004), or gravity forces to describe the potential spatial interaction between two locations (e.g. Ness et al. 1969). Space Syntax, which has already been mentioned, is another stream of research which focuses on the effects of spatial configurations of urban layouts on pedestrian activity levels on the macroscopic level (e.g. Hillier and Hanson 1984, Hillier 1996). In general, while macroscopic models are well suited to modeling general movement patterns on a coarse level, and in many cases relatively efficient in terms of computational effort, they are inappropriate for modeling individual behavior on the micro-scale of buildings or streets, as well as for incorporating inter-individual differences among pedestrians (Haklay et al. 2001, Torrens 2012).

These gaps are attempted to be filled by microscopic models. Here, the aspiration is to simulate each pedestrian as an individual, which enables more detailed representations of cognition, behavior and emerging local patterns, such as line formation or crowding at bottlenecks (Helbig et al. 2001, Johansson and Kretz 2012). Somewhat related to the previous approaches inspired by physics, Helbing and Molnar's (1995) social force model draws from an analogy to Newtonian mechanics, and describes an individual's motion as the result of an equation of attracting and repelling forces. As has been argued, geographic automata, especially CA or ABM, are frequently used to model individual pedestrian behavior. In the case of CA, the walkable area is tessellated into discrete cells, which can be either occupied or unoccupied by a pedestrian (Johansson and Kretz 2012). The movement of an individual pedestrian is then represented by changing states of neighboring cells (e.g. Batty et al. 2003, Varas et al. 2007, Teknomo and Millonig 2007, Iltanen 2012). Although CA-based methods for pedestrian simulation have proven especially useful for modeling movement patterns of large crowds, drawbacks are posed by the focus on local neighborhoods, which limits the incorporation of global influences on movement behavior, practical restrictions related to modeling perceptual and cognitive abilities of virtual pedestrians, and problems caused by the central role of the underlying lattice, such as local minima or artificial symmetries in movement patterns (Benenson and Torrens 2005, Johansson and Kretz 2012). The most recent development in pedestrian simulation are ABM, which model each pedestrian as one individual entity (e.g. Haklay et al. 2001, Kerridge et al. 2001, Turner and Penn 2002, Gloor et al. 2003, Gloor et al. 2004, Zachariadis 2005, Antonini et al. 2006, Ronald et al. 2007, Klügl and Rindsfüser 2007, Durupinar et al. 2008, Teknomo 2008, Kapadia et al. 2009, Klügl et al. 2009, Kitazawa and

Batty 2010, Kim et al. 2011, Torrens 2012, Vizzari et al. 2013). It has been argued that several systemic characteristics of pedestrian traffic, such as the fact that it can be performed in different manners (e.g. scurrying versus strolling), its less constrained movement, its individuality and autonomy, and its smaller scale, make it a particularly suitable application area for ABM (Ronald et al. 2007).

#### *3.2.1.2 Representing the Environment in Agent-Based Pedestrian Models*

Pedestrians move in and are influenced by their environment, which is why particular attention has to be paid to modeling it an appropriate and feasible way (Antonini et al. 2006). Since modeling pedestrian movements is relevant in various contexts, including evacuation scenarios of buildings or trains (e.g. Klügl et al. 2009, Kim et al. 2011), crowd modeling at large-scale public events (e.g. Durupinar et al. 2008, Kapadia et al. 2009, Vizzari et al. 2013), individual pedestrian movements in large buildings such as art galleries (e.g. Turner and Penn 2002), train stations (e.g. Klügl and Rindsfüser 2007), shopping centers (e.g. Kitazawa and Batty 2010) or outdoor areas such as town centers (e.g. Haklay et al. 2001), public parks (e.g. Zachariadis 2005) or in hiking areas (e.g. Gloor et al. 2003, Gloor et al. 2004), different types of environments need to be modeled. Ronald et al. (2007), for instance, classifies pedestrian environments based on their environmental features, typical walking behaviors and the expected volume of pedestrians:

- Small-scale enclosed spaces (e.g. rooms in buildings)
- Large-scale enclosed spaces (e.g. sports arenas or train stations)
- Mixed mode (e.g. urban area which is shared with other traffic modes)
- Open space (e.g. pedestrian areas, public parks)
- Hybrid (e.g. university campuses)

Apart from such classification of real-world environments, however, the computational representation of space is also critical, since it can be expected to have profound effects on the simulation method as well as the results (Antonini et al. 2006). Previous studies have treated space either as a discrete structure (e.g. Kerridge et al. 2001, Turner and Penn 2002, Gloor et al. 2003, Torrens 2012) or as a continuous phenomenon (e.g. Helbing et al. 2001, Antonini et al. 2006). In the former case, according to Torrens (2012), a feasible cell size should be smaller than a person’s physical footprint. Typical grid structures have resolutions of 10 cm (Batty 2005), 40 cm (Chen et al. 2009, Chu 2011) or 75 cm (Kerridge et al. 2001). In the context of CA, Chen et al. (2009) proposed an approach in which cell sizes are elastic with regards to differing system conditions, such as pedestrian density. In other studies, the walking environment is reduced to a graph representation of its path network (e.g. Teknomo 2008). A typical problem of such approaches, however, is posed by the fact that, compared to motorized traffic modes, pedestrian movement is less restricted to a network, which raises the question of how to represent large pedestrian areas (Gaisbauer and Frank 2008). Examples for hybrid approaches include Haklay et al.

(2001), who incorporate raster, vector and network data in their model, Gloor et al. (2004), who combine a graph and a field-based approach, or Zachariadis (2005), who represents space with a grid structure and a network, but models agent movement in a continuous manner.

A different question is posed by the level of detail for modeling the environment. Depending on the complexity of agent perceptions and cognitive processes, as well as the available spatial data, extremes are posed by pure network representations on the one hand, where the walking environment is reduced to its path network's topology, and semantically rich models on the other hand, which can include obstacles (Kim et al. 2011), buildings, streets and sidewalks, and typical pedestrian gateways such as car parks or public transport stations (e.g. Haklay et al. 2001), path characteristics such as its steepness (Gloor et al. 2004) or even information about the visual quality of the visible landscape (Gloor et al. 2003). A number of studies acknowledge the fact that real environments are not static, but rather subject to change, and propose approaches for simulating pedestrian behavior in dynamic environmental conditions (e.g. Teknomo and Millonig 2007, Klügl et al. 2009). Apart from geo-objects and fields, more abstract information used for modeling agent behavior can also be pre-calculated and embedded within the environmental model for the agent to pick up, such as information about visibility relations (de Berg et al. 1997, Gloor et al. 2004), steering force fields (Helbing and Molnar 1995), the distance to the destination (Kretz et al. 2010), expected utility or cost (Zachariadis 2005), or movement affordances (Turner and Penn 2002, Kapadia et al. 2009, Kim et al. 2011).

#### *3.2.1.3 Modeling the Pedestrian Agent*

Several of the shortcomings of macroscopic approaches to pedestrian modeling can be explained by the complexity of pedestrian behavior (Batty 2001). The exact manner in which walking is performed, for instance, depends to a high degree on its predominant purpose:

"Commuters scurry; shoppers meander; bushwalkers trek; power-walkers stride; lovers stroll; tourists promenade; protesters march … But we all walk" (Australian Pedestrian Council, 2015).

Such are not merely semantic subtleties, but of a practical relevance for modelers since they result in observable behavioral differences in terms of speed, directness, tolerated walking distances or specific environmental preferences and route choice (Buchmueller and Weidmann 2006, Zacharias 2001a). Thus, according to Ronald et al. (2007), different types of walking behavior can be distinguished:

- Purposeful (destination-bound) movement in a familiar environment
- Purposeful movement in an unfamiliar environment
- Purposeless wandering

- Evacuation/panic situation
- Forced waiting in an ordered (ticket vendor) or unordered queue (traffic light)
- Temporal constraints as influencing elements (e.g. train schedule)

For modeling pedestrian behavior, Hoogendorn and Bovy (2004) propose a distinction into three behavioral levels. Before starting the trip, decisions related to departure time and activity pattern choice are made on the strategic level. Activity scheduling and route choice take place during the walking process, which, according to the authors, can be described as the tactical level of walking behavior. Finally, the microscopic walking behavior, such as obstacle avoidance, forms the operational level (Hoogendorn and Bovy 2004). In a similar approach, Antonini et al. (2006) sketch three separate models of walking behavior, namely a destination choice model, a route choice model, and a collision avoidance model. Using the example of human navigation on the U. S. Interstate Highway Network, Timpf et al. (1992) and Timpf and Kuhn (2003) distinguish between a planning, an instructional and a driver level, on which wayfinding decisions are made.

Especially in microscopic, agent-based models, a particular focus is often put on modeling pedestrian navigation (Papadimitriou et al. 2009). According to Montello (2005), navigation consists of the two sub-processes wayfinding, which is based on some representation of the environment (e.g. a mental map) and involves various decision making and planning tasks, and locomotion, the act of physically moving in response to sensory input of the immediate environment, which incorporates tasks such as steering or obstacle avoidance. Transferred to the context of pedestrian simulation, this distinction results in two separate questions:

- How to represent spatial knowledge of a pedestrian agent?
- How to model sensory inputs of a pedestrian agent?

For the first case, abstract wayfinding information has often been attributed to the environmental model, as has been described in the previous chapter. A possibility are visibility graphs, in which corners of obstacles are connected with edges based on their mutual visibility, and which can then be used to calculate shortest paths which have similarities with actual human movement patterns (de Berg et al. 1997, Johansson and Kretz 2012). Of course, instead of a visibility graph, actual path networks can be used in a similar manner (e.g. Gloor et al. 2004, Teknomo 2008). A different method involves the use of floor fields, which store information such as distance (Kretz et al. 2010) or walking time (PTV Planung Transport Verkehr 2010) to the destination, and are then used as a look-up table by the agents (Kretz et al. 2010). Apart from the distance or time, however, other environmental aspects can be relevant for pedestrian route choice as well. Hoogendorn and Bovy’s (2004) theory of pedestrian utility maximization represents an approach to model such influences. For each pedestrian, an individual utility function is calculated, which represents the expected trade-off between the utility of performing an activity at a specific location, and the predicted cost of walking there. The walking cost can depend on factors such as distance, number of obstacles, number of sharp turns, level of crowdedness or attractiveness of the environment (Hoogendorn and Bovy 2004). Expected utility values can either be calculated on runtime by each agent individually, or embedded in the environment, for instance as grid cell attributes (Gloor et al. 2004). In general, approaches in which wayfinding information is pre-computed and stored within the environment face the difficulty to incorporate individual differences among pedestrians. As an alternative, for instance, Ronald et al. (2007) propose a BDI-framework for pedestrian simulation. Thus, differences in spatial knowledge, wayfinding strategies or destinations can be represented in a straight-forward way (Ronald et al. 2007). Apart from the area of ABM, other researchers have provided computational models of spatial learning and cognitive map formation (e.g. Kuipers 1978).

Concerning the locomotion aspect of navigation, the representation of sensory information input is relevant. To a high degree, pedestrian movement is based on vision, which is why its appropriate representation is vital (Torrens 2012). Guidelines can be obtained from empirical studies, such as Kitazawa and Fujiyama (2010), who used eye tracking technology to examine how far, wide and fixated pedestrians observe their environment. Their results demonstrated that the field of view is cone shaped, and fixations seldom exceed 5 meters (Kitazawa and Fujiyama 2010). Typical angles for the modeled viewfield are 90° (Kerridge et al. 2001), 115° (Torrens 2012) or 170° (Antonini et al. 2006). There have also been approaches to model the immediate environment and further areas as two separate awareness zones (Torrens 2012).

A particularly important aspect in pedestrian simulation is road crossing. From 1996-2006, for instance, around one third of all accidents involving cars and pedestrians in the OECD countries occurred at pedestrian crossings (La Feypell-De Beaumelle et al. 2010). In the past, there have been approaches to explicitly simulate crossing behavior, such as Bönisch and Kretz (2009), who used the micro-simulation software VISSIM (PTV Planung Transport Verkehr 2010) and model conflict areas (crossings) with the parameters street visibility, gap size between successive vehicles and safety distance. Another example is Fi and Igazvölgyi (2014), who use the same software and compare the gap time between two successive vehicles to the critical gap time for a pedestrian, which is based on the crossing's width and the pedestrian's walking speed.

Finally, ABM provide the possibility to model individual differences among agents. In the context of pedestrian simulation, however, there have been only few approaches to incorporate such subjectivity, which can relate to attributes, perceptions, experiences or attitudes. A particular challenge is posed by the difficulty in recording such aspects, which require sophisticated methods such as in-depth interviews or questionnaires (Kerridge et al. 2001). There have been, however, approaches in which inter-personal differences were explicitly modeled, such as Kerridge et al. (2001), whose pedestrian agents differ in terms of static awareness, meaning the front length of their awareness field, preferred gap size when passing through bottlenecks, and walking speed. Haklay et al. (2001) incorporate socio-economic and behavioral differences, and their effects on all aspects of pedestrian movement. In their simulation of crowd behavior, Durupinar et al. (2008) equip their agents with different personality traits such as openness, extroversion or agreeableness, and test for effects on the emerging systemic behavior.

### 3.2.2 The Concept of Walkability

Aiming to explain and predict the spatial patterns of distribution and concentration of pedestrian activity within urban areas, researchers and planners have often reduced the nature of the relationship between walkers and their surrounding environment to its topological, geometrical or distance-based dimension (Zacharias 2001b). What has since become apparent, however, is the inability of such models to fully explain pedestrian behavior, which is often attributed to a conceptual neglect of the much wider range of pedestrian perceptual and cognitive processes assumed to be equally important in guiding their behavior (Zacharias 2001b, Owen et al. 2004, Johansson and Kretz 2012). As a result, the image that pedestrians form of their urban environment, and which is expressed in the form of opinions and preferences, has gained increasing attention (Zacharias 2001b).

#### *3.2.2.1 Walkability and Pedestrian Behavior*

Pedestrians, due to their systemic characteristics such as low speed, direct contact with the environment and a lack of a protecting hull in case of accidents, pose particular needs to their walking environment, and should be treated separately from other modes of transportation (Frank and Engelke 2001, Saelens et al. 2003). Since the early 1990s, the superordinate term walkability, although lacking a concrete universal definition, has been established to describe the degree to which the urban environment meets these demands, or the “quality of walking environment perceived by the walkers” (Park 2008, p. 22). Although other environmental characteristics, such as climatic conditions or local concentrations of incivilities and crime, have been demonstrated to influence walking behavior as well, walkability, especially when being used in an urban design and planning context, is often restricted in scope to aspects of the built environment, a term which refers to human-made infrastructure and includes transportation systems, land-use patterns and micro-scale urban design features (Renalds et al. 2010, Cunningham et al. 2004).

In accordance with the results of an extensive body of empirical research, walkability indicators, attributes of the built environment from which its walkability can be inferred, exist on different scale-levels, ranging from the micro-scale of the individual street or vista to the macro-level of the entire city or region (Handy 2004, Cervero and Kockelman 1997). Indicators on the latter level can be allocated to the first two of the three Ds of built environment dimensions, (building) density and (land-use) diversity (Cervero and

Kockelman 1997). After their introduction, more macro-scale aspects were found to have a significant effect on people's propensity to walk, and were subsequently added to the list, including destination accessibility, distance to transit, and demand management, for instance in terms of variable parking costs (Zook et al. 2011). A common explanation for the relevance of such urban characteristics is provided by pedestrians' sensitivity to distances. Due to systemic characteristics of walking such as its low speed and resulting physical strain, maximum walking distances are typically low, and have been found to lie roughly between from 250 – 800 meters, with dissimilarities having been found for different North American cities (Pushkarev and Zupan 1975). With the exception of demand management, which aims at stimulating pedestrian or public transport by decreasing the perceived attractiveness of motorized individual transportation, macro-scale aspects influence the effective distances of walking trips (Handy 2004).

With its effect less unambiguously documented, the third D, design, captures urban design elements at the micro-scale level (Cervero and Kockelman 1997). Such aspects are assumed to create a sense of uniqueness and evoke a pedestrian's interest, and improve his or her general image of an urban setting (Cunningham and Michael 2004). Although the fact that well-designed pedestrian environments are generally positively apprehended has been demonstrated in various studies (e.g. Borst et al. 2008, Adkins et al. 2012), there are differing views on the potential of micro-scale elements as actual stimulators of walking transportation. Thus, for instance Cervero (1993) concluded that "micro-design elements are too 'micro' to exert any fundamental influences on travel behavior" (Cervero 1993, p. XVI). To date, however, due to the fact that a range of studies have found significant relationships between the design-related walking quality and pedestrian frequencies, the relevance of such urban environmental characteristics is widely accepted (Papadimitriou et al. 2009, Adkins et al. 2012). According to Handy (2005), the generally mixed nature of empirical findings can be explained by issues such as the use of inappropriate design measures or the fact that the magnitude of influence depends on the actual purpose for walking, with a higher effect when walking for recreational than for transportation purposes.

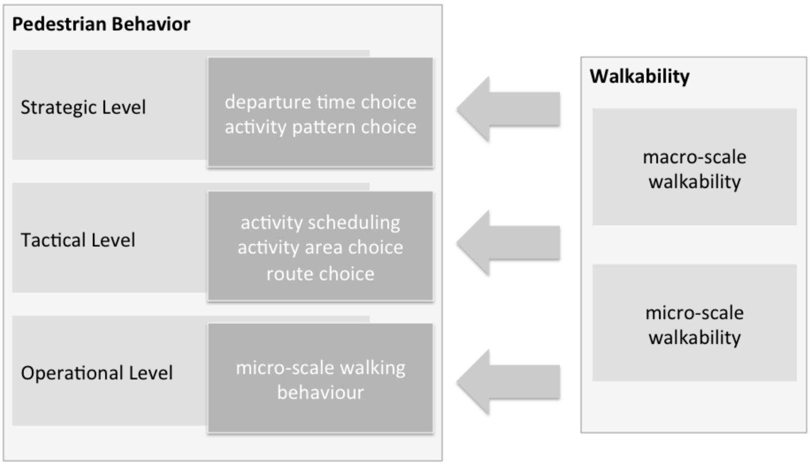


Fig. 9: Influences of Walkability on Pedestrian Behavior

Figure 9 illustrates the potential influences of walkability on pedestrian behavior. As can be seen, an area's perceived suitability for walking can influence a person's walking behavior on all of its three levels (Hoogendoorn and Bovy 2004). Thus, being encoded as spatial memory in the cognitive map, it influences the decisions made at the strategic level, including mode and destination choice, as has been demonstrated by numerous empirical studies. On the tactical level, as has been shown by Özer and Kubat (2007) and Agrawal et al. (2008), the walkability of differing alternative paths or places is perceived at decision points and has an effect on route and activity area choice, while microscopic walking behavior can also be influenced by attracting or repulsing forces due to qualitative differences (Zacharias 2001a, Papadimitriou et al. 2009).

#### *3.2.2.2 Empirical Research on Micro-Scale Walkability*

This chapter aims to identify a set of concrete micro-scale walkability indicators and, for this purpose, briefly reviews relevant evidence which has been collected mainly in the last two decades. In fact, since the mid-1990s, an extensive body of literature has developed on the topic of walkability, with various involved disciplines such as geography, urban design, transportation planning, public health, psychology and sports medicine. The majority of studies focus on identifying macro- and micro-scale environmental as well as pedestrian-related determinants of the recorded amount of walking. Due to their high number, the evidence produced by these empirical studies is consolidated in more than 50 literature reviews and even two reviews of these review papers (Baumann and Bull 2007, Gebel et al. 2007).

A different approach, which is more common in Urban Design, involves surveying pedestrians for their perceptions of walkability instead of measuring the actual walking outcome (e.g. Brown et al. 2007, Adkins et al. 2012). It has been argued, in fact, that this approach may be better suited to examinations of micro-scale walkability elements (Cervero and Kockelman 1997). In addition, a few studies have moved beyond the mere identification of such behavior-environment correlations to the development of theoretical frameworks to conceptualize the relation between pedestrian perceptions and higher-level characteristics of the urban environment (Sarkar 1993, Pikora et al. 2002, Hodgson et al. 2004, Alfonzo 2005). There are numerous cases in which the insights gained from such research activities provided the basis for the development of walkability indexes, that means GIS- or audit-based practical tools to measure the predicted walkability of an urban study area (e.g. Leslie et al. 2007, Schlossberg et al. 2007, Kelly et al. 2007, Clark and Davis 2009).

The largest number of walkability-related contributions compares the walking outcome to a set of potentially influential environmental variables. Given the high number of published studies, a review of reviews approach is chosen to approach this body of knowledge, following two earlier studies (Baumann and Bull 2007, Gebel et al. 2007). In the course of an exhaustive literature search, 51 review papers could be identified of which 18 fitted the inclusion criteria, namely that they reviewed the empirical research in a systematic, not merely narrative way, reported results for walking for transportation purposes, and included micro-scale walkability indicators (Ewing and Cervero 2001, Humpel et al. 2002, Saelens et al. 2003, Lee and Moudon 2004, Cunningham and Michael 2004, Handy 2004, Owen et al. 2004, Sallis et al. 2004, Badland and Schofield 2005, Handy 2005, Davison and Lawson 2006, Wendel-Vos et al. 2007, Saelens and Handy 2008, Renalds et al. 2010, McCormack and Shiell 2011, Sugiyama et al. 2012, Sallis et al. 2012, Van Holle et al. 2012). Of these, 10 originated from the area of public health or sports medicine, 4 from transportation planning and 4 involved researchers from both disciplines. All 18 review papers recapitulated the results of empirical studies which measured a selection of whether objectively assessed or subjectively perceived environmental qualities as the independent and, in most cases, self-reported walking outcome as the dependent variable.

| Variable | Review |
|---|---|
| aesthetics | Sugiyama et al. (2012)<br>McCormack and Shiell (2011)<br>Saelens and Handy (2008)<br>Badland and Schofield (2005)<br>Owen et al. (2004)<br>Handy (2004)<br>Handy (2005) |
| security (crime) | Saelens and Handy (2008)<br>Sugiyama et al. (2012)<br>Owen et al. (2004)<br>McCormack and Shiell (2011)<br>Saelens and Handy (2008)<br>Handy (2004)<br>Handy (2005) |
| public parks | Saelens and Handy (2008) |
| presence of sidewalks | Sugiyama et al. (2012)<br>McCormack and Shiell (2011)<br>Owen et al. (2004)<br>Van Holle (2012)<br>Saelens and Handy (2008)<br>Handy (2004)<br>Handy (2005)<br>Sallis et al. (2012) |
| sidewalk condition/ maintenance | Sugiyama et al. (2012)<br>McCormack and Shiell (2011) |
| traffic | Sugiyama et al. (2012)<br>McCormack and Shiell (2011)<br>Owen et al. (2004)<br>Van Holle (2012)<br>Handy (2005) |
| street lighting | McCormack and Shiell (2011)<br>Sallis et al. (2012) |
| street furniture | McCormack and Shiell (2011) |

Table 1: Variables Found Significant in Review Papers

Table 1 lists the micro-scale environmental criteria which were found significant in one or more review papers. It should be noted, however, that several of the factors listed here were contrastingly assessed as insignificant in other literature reviews. Moreover, the review papers reported very abstract environmental criteria rather than concrete characteristics, which is why the results can give only limited insights in the context of identifying actual, workable indicators.

A more detailed picture of the effect of micro-scale design elements on people's perceptions is provided by a range of pedestrian surveys (Lynch and Rivkin 1959, Clifton and Livi 2004, Brown et al. 2007, Sanches et al. 2007, Borst et al. 2008, Agrawal et al. 2008, Kaufmann et al. 2010, Samarasekara et al. 2011, Adkins et al. 2012). These studies apply one of two distinct strategies, thus either relate test persons' walkability ratings to objectively measured environmental criteria, or ask pedestrians what aspects they rate as important for walking. Several concrete criteria could be identified, such as the presence of urban greenery (Lynch and Rivkin 1959, Borst et al. 2008, Kaufmann et al. 2010 Adkins et al. 2012), aesthetics (Lynch and Rivkin 1959, Brown et al. 2007, Agrawal et al. 2008, Samarasekara et al. 2011), traffic safety (Brown et al. 2007, Agrawal et al. 2008), here especially a physical separation from traffic (Lynch and Rivkin 1959, Clifton and Livi 2004, Kaufmann et al. 2010, Samarasekara et al. 2011, Adkins et al. 2012), appropriate street crossing facilities (Lynch and Rivkin 1959, Agrawal et al. 2008, Borst et al. 2008, Kaufmann et al. 2010) and curb cuts (Clifton and Livi 2004), low traffic speed and volume (Sanches et al. 2007, Borst et al. 2008, Samarasekara et al. 2011), low levels of noise or air pollution (Sanches et al. 2007), safety from crime (Brown et al. 2007, Clifton and Livi 2004, Agrawal et al. 2008), here especially appropriate lighting conditions (Clifton and Livi 2004, Sanches et al. 2007, Kaufmann et al. 2010) and a lack of signs of decay such as vacant buildings or litter (Borst et al. 2008, Kaufmann et al. 2010), the existence of sidewalks (Clifton and Livi 2004, Kaufmann et al. 2010) with appropriate width (Lynch and Rivkin 1959, Samarasekara et al. 2011), appropriate surface structure (Lynch and Rivkin 1959, Sanches et al. 2007, Agrawal et al. 2008) and a lack of obstructions (Sanches et al. 2007, Samarasekara et al. 2011), low rates of slopes and stairs (Borst et al. 2008), as well as the presence of benches and other street furniture (Brown et al. 2007, Borst et al. 2008, Kaufmann et al. 2010).

Grounded on this empirical basis, several authors have proposed theoretical frameworks of walkability, in which potential indicators were classified to higher-level attributes of the urban environment (Sarkar 1993, Pikora et al. 2002, Hodgson et al. 2004, Alfonzo 2005). Since in general, there is much agreement among the authors, with differences being found only in the terminology or exact classification of lower level indicators, Alfonzo’s (2005) hierarchical model of walking needs is discussed as a representative, and is shown in figure 10. Setting out to develop a “social-ecological model for how both urban and non-urban form factors may interact to affect walking”, the author bases her work on Maslow’s (1954) theory of human motivation (Alfonzo 2005, p. 817). In a similar fashion, it is argued that walking needs are hierarchically organized, with basic needs having to be fulfilled, or afforded in the sense of Gibson (1979), before higher-level needs are even considered by pedestrians. On the most fundamental level, Alfonzo (2005) places the feasibility of a walking trip, which refers to the question whether it is practical with regards to time or mobility constraints, and can therefore be allocated to the pedestrian rather than the walking environment. This is followed by accessibility, which is determined by the presence and completeness of the walking path network, the distribution of destinations and potential barriers to movement. If this need is satisfied, according to the author, safety issues relating to the perceived risk of crime such as certain types of land-uses or loitering people are considered. Then, the expected level of comfort is assessed, which refers to the ease, convenience and contentment experienced while walking, and may be operationalized by traffic calming or separation features, street furniture and others. The final level is pleasurability, the level of appeal an urban environment evokes in a pedestrian. This is determined by aesthetical attributes and liveliness (Alfonzo 2005).

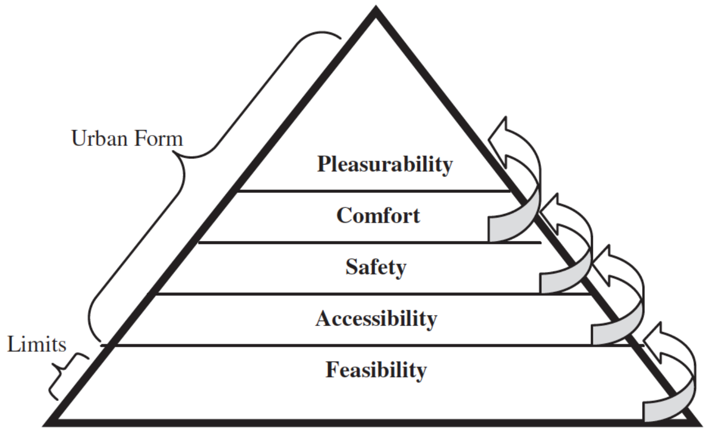

Fig. 10: Hierarchy of Walking Needs
(adapted from Alfonzo 2005)

#### *3.2.2.3 Pedestrian Factors*

In accordance with ecological psychology, it has been argued that the existence of an affordance or the evaluated suitability of an environmental object are not determined by its characteristics alone, but arise from their functional relations to properties of the perceiving organism. Transferring this notion to micro-scale walkability, it appears obvious that this abstract quality should not be narrowed to an objective, purely environmental view as well, but rather incorporate the perspective of differing types of pedestrians (Jonietz and Timpf 2013a). In fact, there is a growing awareness of this issue among planners and researchers (Buchmueller and Weidmann 2006).

Since there is far less research on the pedestrian compared to the environmental side, the involved criteria and particularly their interactions with the environment are less clear. Most frequently, pedestrians are classified according to their physical abilities, social roles and economic constraints (Hodgson et al. 2004). Hodgson et al. (2004) list potentially important aspects including physical impairments, age, additional luggage, gender, social status and ethnicity. Buchmueller and Weidmann (2006) provide very detailed information on the statistical distribution of pedestrian body dimensions (see figure 11), additional space required when walking due to wavering movement, typical shy-away-distances to walls, traffic lanes or other obstacles, walking speed and energy expenditure in situations, for instance under the influence of gradients or stairs.

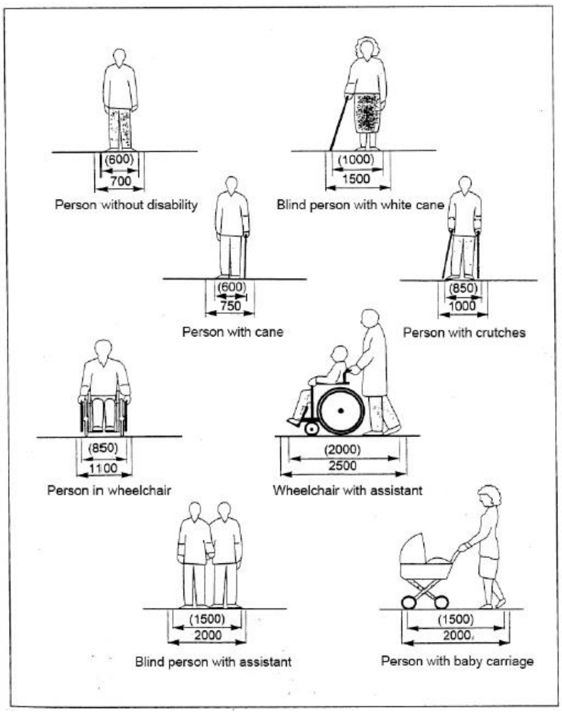


Fig. 11: Body Dimensions of Pedestrians
(Ackermann 1997)

An alternative distinction can be based on different purposes of walking, thus, differences in walking behavior and perceptions of walkability were found between goal-directed and exploratory walking (Handy 2005). Boesch (1992) mixes walking purposes with socio-demographic characteristics, and describes different sub-networks for pupils, common or mobility-impaired commuters, shoppers and several more. These sub-networks, examples include a way-to-school-network or shopping network, should primarily meet the needs of their particular user-group (Boesch 1992). In the model of walking needs proposed by Alfonzo (2005), the author discusses a range of personal attributes which include biological and demographic factors, however, relates them more to the mode choice process than to the perception of walkability. Basbas et al. (2010) and Vukmirovic (2010) provide very detailed lists of pedestrian abilities, which include aspects related to physical, physio-motor, sensorial and cognitive abilities, and affect various aspects of the walking process, as can be seen on the example of Basbas et al. (2010) in table 2.

| Ways in which pedestrians differ | Affecting | Impacting upon |
|---|---|---|
| Height | Ability to see over objects<br>Ability to be seen by others | Sight lines |
| Speed of reflexes | Inability to quickly avoid dangerous situations | Crossing opportunities |
| Stamina | Journey distance between rests | Resting places |
| Visual perception | Ability to scan the environment and tolerate glare | Legibility of signs<br>Detection of kerbs and crossing locations<br>Trip hazards<br>Tactile paving<br>Judging traffic |
| Attention span, and cognitive abilities | Time required to make decisions<br>Difficulties in unfamiliar environments<br>Inability to read or comprehend warning signs | Positive directions<br>'Legibility' of streetscape<br>Consistency of provision<br>Use of symbols |
| Tolerance for adverse temperature and environments | Preference for sheltered conditions | Location and exposure of routes |
| Balance and stability | Potential for overbalancing | Provision of steps and ramps<br>Kerb height<br>Gradients<br>Crossfall |
| Fear for personal safety and security | Willingness to use all or part of a route | Lighting<br>Surveillance<br>Lateral separation from traffic<br>Pedestrian densities<br>Traffic speed and density |
| Manual dexterity and coordination | Ability to operate complex mechanisms | Pedestrian activated traffic signals |
| Accuracy in judging speed and distance | Inopportune crossing movements | Provision of crossing facilities |
| Difficulty localising the direction of sounds | Audible clues to traffic being missed | Need to reinforce with visual information |
| Energy expended in movement | Walking speed | Crossing times<br>Length of journey<br>Surface quality |

Table 2: Pedestrian Abilities

(Basbas et al. 2010)

As a reaction to an increased risk of accidents at pedestrian crossings, there has also been research on inter-pedestrian differences in these situations. Yannis et al. (2013), for instance, found that gender affects road crossing behavior, with male test persons showing more risk-taking behavior. Oxley et al. (1995) found differences in the road crossing behavior of younger and elderly test persons, which they explained with age-related deficits in sensory, perceptual and cognitive abilities. In general, the authors argued that in complex crossing situations, elderly people are more at risk (Oxley et al. 1995).

Finally, Ovstedal and Ryeng (2002) examined inter-pedestrian differences in the relative importance attributed to different higher-level environmental characteristics, including safety and security, attractiveness, traffic conditions, social meeting and pleasantness, efficiency of moving, and many more. Based on interviews and a factor analysis, they identified four pedestrian types: the easy-going type, who especially values the weather and easy orientation, a safety- and security-oriented pedestrian, those seeking fresh air, space and light and a pedestrian type which is particularly oriented towards social pleasure (Ovstedal and Ryeng 2002).

### 3.2.3 Conclusion: Modeling Pedestrian-Environment Interactions

Due to the thematic scope of the case study of this thesis, this chapter has presented relevant prior work on pedestrian behavior modeling, with a particular emphasis on pedestrian-environment interactions and walkability. A juxtaposition of current modeling practice and empirical research demonstrated that although micro-scale characteristics of the environment have been identified as significant for pedestrian movement, they have, apart from a few exceptions, not yet been incorporated systematically into pedestrian simulation models. Further, despite the capabilities of ABM to represent inter-personal differences, the actual heterogeneity of pedestrians has so far been largely neglected, a shortcoming which is also present in current walkability research. Instead, it is necessary to understand walking as a high-level action which involves a pedestrian agent equipped with a specific set of physical, physio-motor, sensorial and cognitive abilities as well as individual preferences, which is situated in a complex walking environment with differing degrees of perceived walkability. Transferred to the context of this thesis, the actual route chosen by pedestrians can be understood as their functional place for walking, which is constructed based on its perceived overall suitability for walking. Locations with high frequencies of overlapping emerge as important pedestrian places, the identification or prediction of which is of interest for planners.

# 4 A Suitability-Based Model of Functional Place

In this chapter the main contribution of this thesis is described in detail. Based on the theoretical and methodological foundations laid in the previous chapters, a computational model of functional place is developed. In the first subchapter, our notion of place is explained on the conceptual level. The second subchapter describes how the theoretical concepts are implemented as a computational model in Java and NetLogo (Wilensky 1999). Please note that only exemplary code segments are presented in the text, whereas the complete program code can be found in Appendices A and B.

## 4.1 Conceptual Model

This chapter starts with the development of a model of human spatial suitability assessment. The resulting simulation framework is then embedded within a superordinate conceptualization of place, which is described afterwards.

### 4.1.1 A Model of Human Spatial Suitability Assessment

As has already been discussed in Chapter 2. 2, the perceived utility which humans allocate to places plays a central role in their spatial choice processes. Further, the relation between abstract utility values and the suitability of a place has been presented together with established GIS-based methods for suitability calculation and mapping which, however, are based on an objective, mathematical notion of space and therefore inappropriate for modeling human suitability perception and assessment. Although the work done by Ortmann et al. (2014), which has already been discussed in Chapter 2. 1. 2, allows for the translation of affordances expressed in ordinal values from one person to another, there is currently no conceptual approach to formally describe how a human agent interprets the suitability of his or her environment with regards to a complex action. As a result of the importance of suitability assessment for place formation, however, this represents a precondition for computationally modeling functional places. The approach presented in this chapter aims to fill this gap and, building on Gibson's (1979) affordance concept and activity theory, proposes a simulation framework which allows for the automated calculation of agent-, environment-, and action-specific suitability values. Concerning the type of actions, our focus is at this stage restricted to object manipulation in the sense of Tolman (1958). Further, since we assume a suitability-based homo economicus paradigm, a purposive-rational model of action in the sense of Werlen (2000) is applied.

#### *4.1.1.1 An Extended Notion of Affordances*

Gibson's (1979) affordance concept describes how humans directly perceive action potentials in their ecological environment, which, however, are not inherent as preset qualities within the environment, but rather determined by mutual dependencies of the perceiving organism and the perceived medium or surface. So far, affordances have been mostly restricted to binary statements, which means that they can either exist or not at all (Gibson 1979, Warren 1984). Only Greeno (1995) briefly mentions the general possibility of affordances to be graded phenomena. In this thesis, thus, the notion of affordances is extended to incorporate a notion of grade, which, other than being restricted to the extremes of true or false, expresses the suitability of a medium, surface or object with regards to an action.

For this, the formal notion of affordance is based on the definition proposed by Stoffregen (2003), who, in contrast to Turvey (1992), allocates an affordance to the system of agent and environment, a procedure which corresponds more closely to Gibson's (1979) original idea, and, as will be argued, allows for a higher flexibility when modeling affordances. Accordingly, in this thesis, an affordance is understood as a higher-order property of a system

$$\mathrm{W_{ij\propto}} = \left(\mathrm{agent_i}, \mathrm{environmental\ primitive_j}, \mathrm{action_\propto}\right) \tag{7}$$

which consists of an agent, an environmental primitive, which could be any perceived medium, surface or object, and an action to be performed by the agent. In fact, other than in Stoffregen's (2003) approach, the respective action is explicitly treated as part of the system. This is due to the fact that all system components, and in particular their mutual dependencies, are of decisive importance for the emergence of an affordance.

According to the principle of agent-environment mutuality, the existence or non-existence of an affordance of $action_\propto$ in $W_{ij\propto}$ is determined by certain agent- and environment-related properties which we term agent capabilities $cap_{ij\alpha}$ and environmental dispositions $disp_{ij\alpha}$, and which are interconnected in dependency relationships. The exact properties which are of relevance for the affordance are hereby determined by the action. To state an example, whereas the existence of a sidewalk in an urban environment is certainly of relevance for the affordance of *WALKING*, this environmental property can be neglected with regards to *DRIVING*. It is thinkable, though, that some situations require the calculation of capabilities or dispositions from multiple properties of the agent or the environment. Thus, for instance, the total space needed by a pedestrian in order to pass a bottleneck comprises not only its body width, but also the extent of its additional wavering movement while walking (Buchmueller and Weidmann 2006). Accordingly, our notion of agent capability $cap_{ij\alpha}$ and environmental disposition $disp_{ij\alpha}$ can be interpreted as situation-specific dynamic representations of the agent or the environmental primitive, respectively, which are restricted to include only the sub-set of properties which are relevant with regards to the $action_\alpha$.

For the process of evaluating affordances and suitability values in $W_{ij\alpha}$, we draw from the approach proposed by Warren (1984). In accordance, the dependency relationship between the relevant agent- and environment-related properties are expressed in the form of a ratio value $\pi$, which can be further analyzed to determine whether an $action_\alpha$ is afforded in $W_{ij\alpha}$. Furthermore, however, the $suitability_{ij\alpha}$ provided within a system $W_{ij\alpha}$ with regards to the performance of $action_\alpha$ can be inferred from the relative location of the ratio $\pi_{ij\alpha}$ with regards to the optimal point $\pi_O$ and the critical threshold values $\pi_{max}$ or $\pi_{min}$, respectively, as defined by Warren (1995). The action of *STAIR-CLIMBING*, for instance, may be afforded in two instances of an $agent_i$-$climbing_\alpha$-$stairs_j$ system $W_{ij\alpha}$. If, however, one had a ratio $\pi$ closer to $\pi_{max}$, it could be expected that humans would perceive this system as less suitable for *STAIR-CLIMBING* than another system with a value closer to $\pi_O$, where less energy would be required to perform the action. Similarly, a narrow passage for pedestrians, such as a door, is certainly more suitable for the action *PASSING THROUGH* if it is wider than the pedestrian's physical space, and provides additional buffer space between the agent and the doorway (Buchmueller and Weidmann 2006). Accordingly, when expressed on a scale from 0 – 1, the suitability value $suitability_{ij\alpha}$ should tend toward the maximum value 1 if $\pi$ approaches $\pi_O$, while at $\pi_{max}$ or $\pi_{min}$, it should reach 0, meaning that $action_\alpha$ is no longer afforded in this system.

Consequently, it can be argued that by setting certain action-specific agent- and environment-related properties $cap_{ij\alpha}$ and $disp_{ij\alpha}$ in a relation in the form of

$$\pi = \frac{disp_{ij\propto}}{cap_{ij\propto}} \tag{8}$$

and comparing the received ratio values to known threshold values $\pi_O$ and $\pi_{max}$ or $\pi_{min}$, it is possible to derive scaled values for affordances, as illustrated on the example of a linear dependency relationship in figure 12.

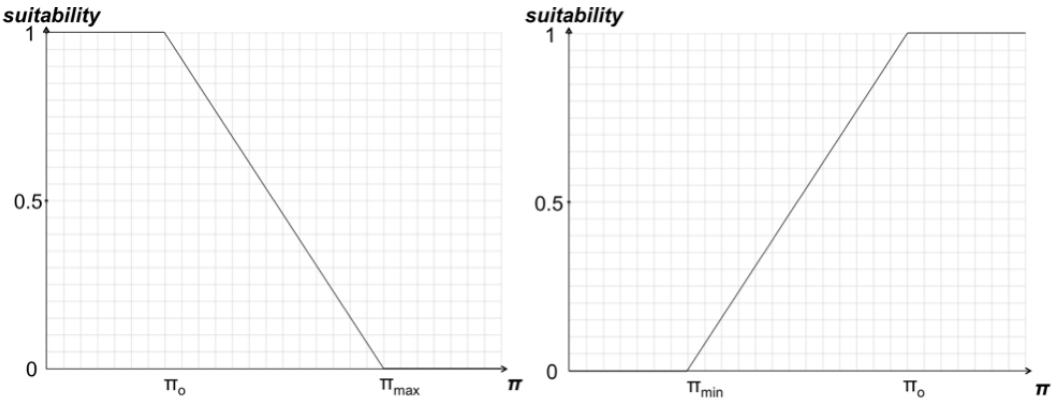


Fig. 12: Suitability Calculation in Case of $\pi_{max}$ (left) or $\pi_{min}$ (right)

However, since the assumption of a linear mapping from $\pi$ to $suitability_{ij\alpha}$ might not correspond to all systems, it could be replaced by another function. The resulting values can be interpreted as the system-specific values of *suitability$_{ij\alpha}$* which are provided within system $W_{ij\alpha}$.

#### 4.1.1.2 A Hierarchical Model of Action

In our view, of the components of a system $W_{ij\alpha}$, it is the $action_\alpha$ that establishes the semantic connection between the $agent_i$ and the $environmental\ primitive_j$, meaning that its exact execution procedure determines not only which capabilities and dispositions are involved but also how they relate to each other. In order to acknowledge the fact that complex actions can be composed of simpler, basic actions, as Ortmann and Michels (2011) or Scheider and Janowicz (2014) propose, actions are modeled as hierarchically structured phenomena, such that more complex actions refer to Gibson's (1979) higher-order affordances. In order to identify the relevant $cap_{ij\alpha}$ and $disp_{ij\alpha}$, therefore, a precondition is to identify the subordinate actions, in the following referred to as sub-actions, which contribute to the respective higher-level action. For this, inspired by activity theory (Leontiev 1978), and following the conceptualization and terminology proposed by Kemke (2001), we distinguish between the three different hierarchical levels of abstraction pragmatic (referring to a goal state), semantic (referring to a change of state) and realization (referring to an operational realization) when modeling actions.

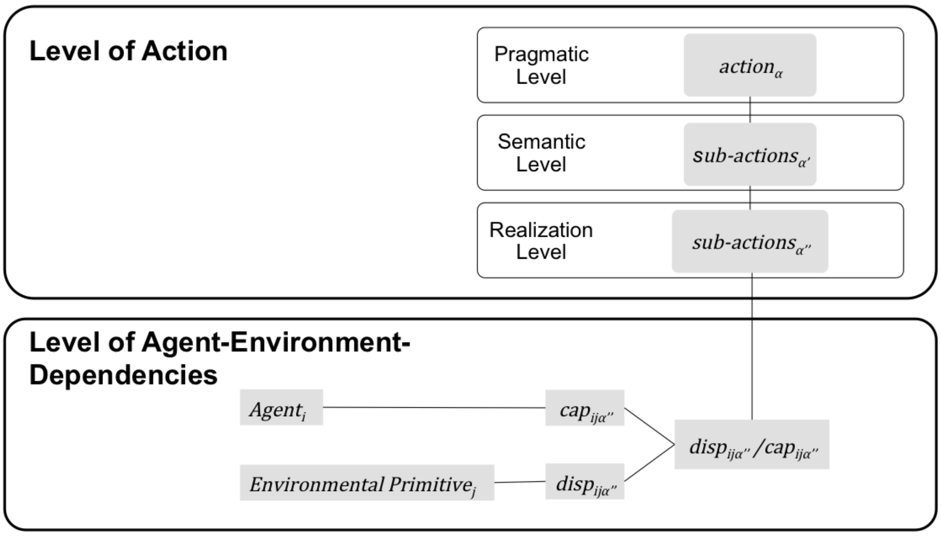


Fig. 13: Action Model
(adapted from Jonietz and Timpf 2013a)

Figure 13 illustrates our model of action. As can be seen, a distinction is made between the two modeling levels action and agent-environment-dependencies. On the superordinate action level, the relationships between actions on the different levels of abstraction are described. Thus, a pragmatic $action_\alpha$ is carried out by performing one or more semantic $sub\text{-}actions_{\alpha'}$. Actions on the semantic level of abstraction, in turn, must be physically realized by operational $sub\text{-}actions_{\alpha''}$ on the realization level. Usually, there will be a 1:n or a n:m relationship between the actions on the different hierarchical levels.

Thus, using the abstraction proposed by Russel and Norvig (2003), the problem can be described as follows: A pragmatic $action_\alpha$ describes a state of either agent or environment, or a combination of both, which represents a goal state for *agent*$_i$. Based on the initial state of the agent itself and the environment, a transition model must be created, which describes all possible state changes which would eventually lead to the intended goal state, expressed by semantic *sub-actions*$_{\alpha'}$. In the next step, based on the initial state of agent and environment, a set of possible realization *sub-actions*$_{\alpha''}$ must be defined, which would, in their entirety, cause all state changes necessary to achieve the goal state, that means to realize the pragmatic *action*$_\alpha$. To refer back to the example of lighting a room, in case of a broken light bulb, for instance, *LIGHT A ROOM* (*action*$_\alpha$) could involve the following *sub-actions*$_{\alpha'}$: *GET A NEW LIGHT BULB*, *REPLACE THE BROKEN LIGHT BULB* and *SWITCH THE LIGHT ON*. Similarly, *REPLACE THE BROKEN LIGHT BULB* requires a range of *sub-actions*$_{\alpha''}$ to be performed by the agent. In general, sub-actions on the semantic and the realization level can be conducted either sequentially, as in the previous example, or simultaneously, as *HOLD THE LIGHT BULB* and *SCREW THE LIGHT BULB IN SOCKET*. Sub-actions might also provide alternative procedures for higher-level actions, such as *LIGHT A ROOM* (*action*$_\alpha$), which could also be carried out by *LIGHT A CANDLE* (*sub-action*$_{\alpha'}$) instead of *SWITCH THE LIGHT ON* (*sub-action*$_{\alpha'}$) (Jonietz and Timpf 2013a).

The choice of an action strategy can be based on the overall suitability of each alternative. Since it, as conceptualized in this thesis, arises from the interplay of agent, environment and action, it is calculated on the modeling level of agent-environment-dependencies. As shown in figure 13, having arrived at the lowest level of the hierarchical action model, at the realization level with the *sub-actions*$_{\alpha''}$, it is possible to define and compare the action-specific agent- and environment-related properties $cap_{ij\alpha''}$ and $disp_{ij\alpha''}$, calculate $\pi$, determine $\pi_O$ and $\pi_{max}$ or $\pi_{min}$, compare the resulting values, and derive scaled values for suitability. This process is described in detail in the following chapter.

#### *4.1.1.3 An Affordance-Based Simulation Framework for Spatial Suitability Assessment*

In combination, the described concepts allow agents to be equipped with the ability to evaluate whether a complex pragmatic *action*$_\alpha$ is afforded in a system $W_{ij\alpha}$, and express the provided suitability for reaching the goal as a scaled value. In a further step, an agent could adapt its behavior in order to optimize the suitability, for instance by modifying its action strategy in terms of realization *sub-actions*$_{\alpha''}$, choosing a different environmental primitive for performing the pragmatic *action*$_\alpha$, or somehow changing its own attributes in order to enhance its capabilities and the resulting suitability. The conceptual framework for the suitability assessment process is illustrated in figure 14.

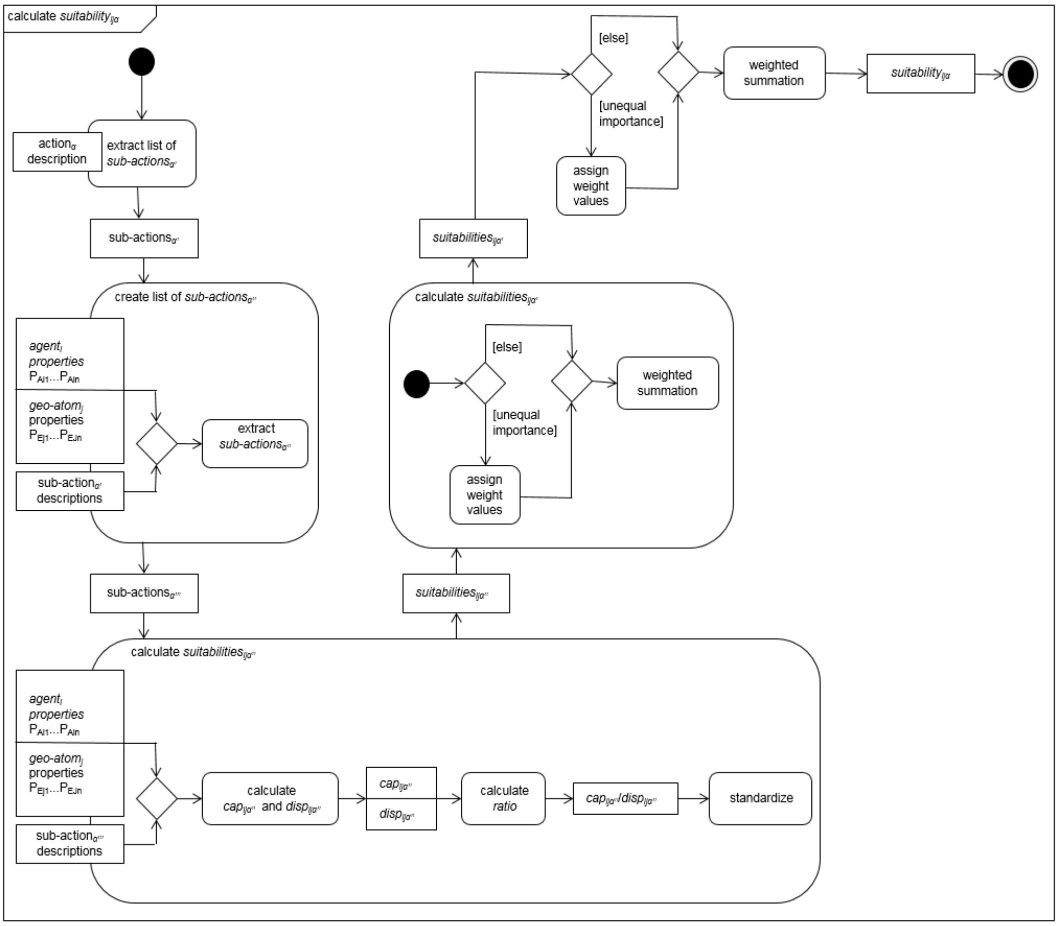


Fig. 14: The Process Model of Spatial Suitability Assessment (adapted from Jonietz and Timpf 2013a)

Based on the previous explanations, we assume a discretionary system $W_{ij\alpha}$ composed of an *agent$_i$* with the goal of performing a pragmatic *action$_\alpha$*. As environmental counterpart, the system includes exactly one *geo-atom$_j$*, as an atomic spatial entity. At this stage of the model description, the context has shifted from the description of a mental process to a computational setting, which is why the term *environmental primitive$_j$* is replaced by *geo-atom$_j$* to denote an atomic spatial entity in a geo-spatial model (Goodchild et al. 2007).

A further assumption is based on the notion that, if the realization of *action$_\alpha$* is possible in $W_{ij\alpha}$, implying that there is an *affordance$_{ij\alpha}$* as a higher-order property of the system, then the *suitability$_{ij\alpha}$* which is provided can be computed and expressed in the form of a standardized numeric value. As described above, this requires the detailed modeling of the hierarchical structure of pragmatic *action$_\alpha$*, meaning that its potentially contributing *sub-actions$_{\alpha'}$* and *sub-actions$_{\alpha''}$* on the lower levels of abstraction must be identified and analyzed separately. The *suitability$_{ij\alpha}$* can then be calculated from the received values for *suitabilities$_{ij\alpha''}$*, meaning the suitability which is provided for each realization-level sub-action in $W_{ij\alpha''}$, which are combined on the basis of the hierarchical action structure.

As a first step, as illustrated in figure 14, all $sub\text{-}actions_{\alpha'}$ on the semantic level of abstraction, as well as their dependencies as alternatives or complements must be identified based on the description of $action_{\alpha}$. This structure represents the state transition model and needs to be pre-defined and stored as part of the knowledge base of $agent_i$. Since the basic suitability values are calculated on the level of the realization actions, however, it is necessary to move to the lowest level of abstraction. Thus, for each semantic $sub\text{-}action_{\alpha'}$, the list of $sub\text{-}actions_{\alpha''}$ on the realization level and their corresponding relationships must be identified. In other words, it must be defined what must be physically done by $agent_i$ in order to realize an intended state change. This information needs to be accessible to the agent.

Differently to the semantic level, on the realization level, each $sub\text{-}action_{\alpha''}$ is atomic, meaning that it is not to be broken down any further. When modeling realistic human spatial behavior, naturally, one could excessively refine every motoric action until finally arriving at a level of detail where each single muscular contraction is acknowledged. For most purposes, however, there is a feasible level of model precision, which is likely determined by the limited level of detail of input data, restrictions in computational effort or the required model accuracy. Nevertheless, at this fundamental level, each $sub\text{-}action_{\alpha''}$ can be related to one specific pair of a capability $cap_{ij\alpha''}$ of $agent_i$ and an environmental disposition $disp_{ij\alpha''}$ of the $geo\text{-}atom_j$. These capabilities and dispositions are derived from the total of properties $P_{i1}...P_{in}$ of $agent_i$ and $P_{j1}...P_{jn}$ of $geo\text{-}atom_j$. Their dependency relationship with regards to $sub\text{-}action_{\alpha''}$ is analyzed by calculating ratio values $\pi$, which, based on a comparison with $\pi_0$ and $\pi_{max}$ or $\pi_{min}$ and a predetermined dependency function, are mapped to $suitability_{ij\alpha''}$ values for each $sub\text{-}action_{\alpha''}$. It should be noted, however, that in particular contexts, it might be preferable from a modeler's perspective to allocate not one but several capability-disposition pairs to one realization action, and calculate an average suitability value, a procedure which might be worthwhile for reasons of feasibility and simplicity. Still, although this does not contradict the adequacy of the model, a rigorous restriction to exactly one pair of capability and disposition is to be generally preferred, since it is straight-forward and corresponds more closely to the original ideas of Gibson (1979) and Warren (1984).

The following step is necessary to acknowledge the fact that agents might attribute different relative importance to different sub-actions. It is realistic to assume that as a result of individual preferences or specific needs, there are situations in which a high suitability value is especially desirable with regards to particular sub-actions, while for others, a low suitability would be acceptable. Similar to the standard procedure applied in GIS-based suitability mapping, these individual differences are incorporated by the assignment of individual weight coefficients (Malczewski 2006). Thus, each $suitability_{ij\alpha''}$ value is assigned a distinct weight coefficient to express the relative importance of the realization action for the final $suitability_{ij\alpha}$ of the pragmatic action, and, during the following calculation process, a second coefficient is added, which denotes the different relative importance of the semantic actions on the final result. When assessing the walkability of two routes, for instance, a pedestrian agent might place higher relative weight on the sub-suitability which is provided for *FEELING SAFE* than for *CROSSING A ROAD* ($sub\text{-}action_{\alpha'}$). For a mobility-impaired person, in terms of physical movement, it can be of a higher relevance to be able to *OVERCOME A SLOPE* than to *MOVE ON THE SHORTEST PATH* ($sub\text{-}action_{\alpha''}$). With the weight coefficients assigned, a suitability value $suitability_{ij\alpha}$ can be calculated for each possible course of action based on the possible combinations of realization $sub\text{-}actions_{\alpha''}$, which are retrieved from the action descriptions of the sematic $sub\text{-}actions_{\alpha'}$, and finally their potential sequences to achieve pragmatic $action_{\alpha}$. For this, the additive weighted additive score traditionally used in suitability mapping can be deployed (Malczewski 2006):

$$suitability_{ij\alpha} = \frac{\sum_{i,j,\alpha''=1}^{n} w \times suitability_{ij\alpha''}}{\sum_{i,j,\alpha''=1}^{n} w} \tag{9}$$

### 4.1.2 A Model of Functional Place Formation

In this thesis, suitability is conceptualized as a systemic quality which arises from the interplay of agent, environment and action. On this foundation, place is modeled as a subset of geo-atoms bound by a functional unity with regards to a pragmatic action. By simulating the process of human formation of functional places, a method is described to dynamically extract them from spatial data.

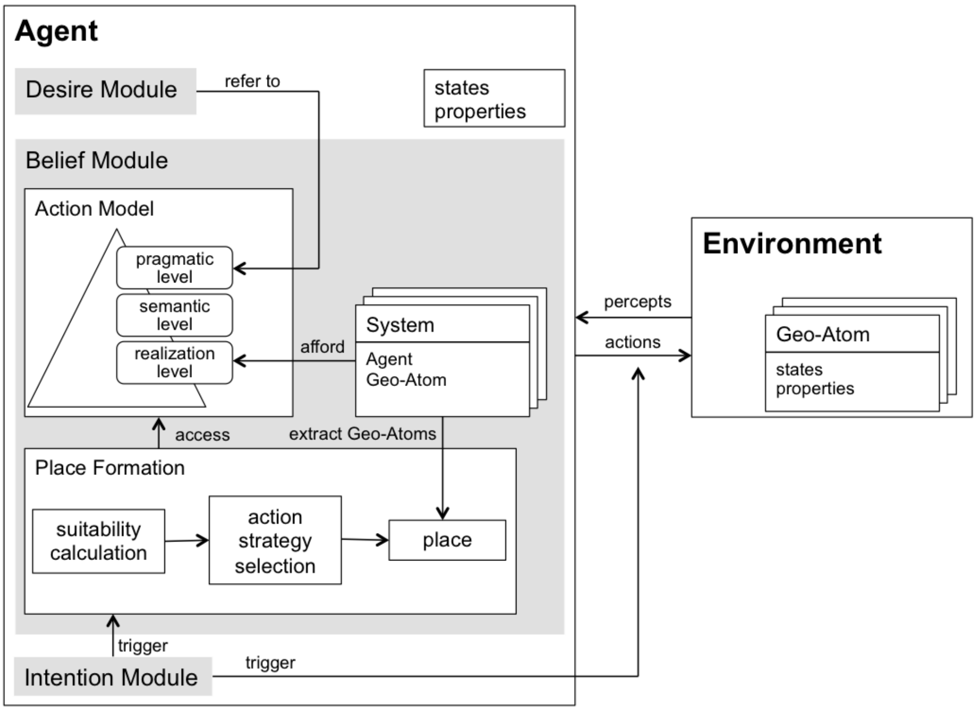


Fig. 15: Conceptual Model of Place Formation

The conceptual model is illustrated in figure 15. In accordance with the basic agent concept of Russel and Norvig (2003), we assume an agent which is equipped with one or more sensors, and thus, has the ability to perceive its environment. Further, it has the necessary means to perform actions which cause changes, both to its own and the environment's state. Apart from perception and action, the agent is unique in terms of its set of characteristics, and is capable of proactive behavior towards a certain goal. Its behavior is based on the pre-assessed suitability which results from different strategies, which makes it a utility-based, adaptive agent. The agent is restricted with regards to agent interaction, since the presence and influence of other agents is neglected at the present stage. Further, in our conceptual model, the agent has no prior memory of the environment. It is, though, capable of practical reasoning which is modeled with a BDI-inspired framework.

The agent is spatially explicit, and set within its ecological environment, with which it has a multi-directional relationship (Gibson 1979, Stanilov 2012). The environmental model consists of a set of discrete geo-atoms, which have a set of properties and states. The environment is partially observable to a single agent, and deterministic in the sense that its future state can be fully predicted based on its current state and the agent's actions (Russel and Norvig 2003).

The agent's behavior is motivated by its desire, which refers to an abstract pragmatic action. In accordance with the described hierarchical action model, each possible sequence of agent- or geo-atom-state transitions which would lead to the desired goal state is stored in the agent's belief structure in the form of semantic actions. Further, any required spatial dependencies between geo-atoms are described explicitly. Thus, in many cases, it is, for instance, not sufficient that two required state changes affect any two geo-atoms, but it might be necessary for them to take place on exactly the same entity, or occur in neighboring geo-atoms. To give an example, imagine an agent's goal state of an open door to be *CLOSED* and *LOCKED*, which only makes sense if the two state changes involve not different doors but the same one, or an agent *CHANGING ITS PHYSICAL LOCATION* from one room to the other, which requires them to be topologically connected. For each semantic action, in turn, the information about its contributing realization actions is also stored in the agent's beliefs, again, together with any spatial requirements concerning the involved geo-atoms. Each realization action, finally, has attributed information which describes its individual procedure of suitability assessment, including the involved capabilities and dispositions, the corresponding $\pi_O$, $\pi_{max}$ or $\pi_{min}$ values and the dependency function between the suitability and $\pi$. If needed, individual weight coefficients are also attributed to each realization and semantic action.

The place formation process starts when, in order to realize its desire, the agent executes the intention of purposefully perceiving its environment, which means it becomes aware of a subset of separate geo-atoms together with their respective properties and states. In an internal process, the agent then forms a system with each perceived geo-atom, and, for each of these systems, tests every realization action, using its pre-defined procedure, for affordance and suitability. If an affordance exists in a system, meaning that $suitability_{ij\alpha''} > 0$, the suitability value is stored together with the corresponding system in the belief structure of the agent. When the process has terminated for the subset of perceived geo-atoms, all possible action strategies can be determined based on the available systems, afforded realization actions, and semantic connections as well as spatial dependencies as stored in the hierarchical action descriptions of the actions on the pragmatic and semantic level. Incorporating the suitability values and weight coefficients, overall suitability values for each possible course of action can be calculated and stored as a belief. Since we assume a homo economicus paradigm, the agent then selects the strategy which provides the highest suitability and performs the corresponding realization actions which involve the according geo-atoms. As a result, the state changes occur which, in their entirety, lead to a goal state and thus realize the pragmatic action. The associated geo-atoms at which the realization actions take place, finally, represent the individually formed functional place with regards to the complex, pragmatic action.

### 4.1.3 Conclusion: A Suitability-Based Model of Functional Place

This chapter has presented a conceptual model of place and its formation process. As a first step, a concept for automated spatial suitability assessment was presented based on an extended notion of the affordance-concept and a hierarchical action model. This framework allows for the simulation of an agent evaluating the suitability of functional places and its constituting geo-atoms with regards to complex, pragmatic actions. In accordance with the notion of a subjective interpretation of the world which guides our decisions and behavior, these suitability values are unique for the specific combination of agent, environment and action, but clearly defined and computable.

The conceptual model of suitability assessment provides the basis for a model of functional place, which defines them as meaningful subsets of geo-atoms that can be dynamically extracted from a digital representation of the geo-spatial environment. On this conceptual basis, computer systems can be developed which are able to identify agent-specific higher-order affordances without the need to hard-code them as attributes of geo-spatial entities. Instead, the dynamic computation of affordances on the basis of pre-defined suitability calculation procedures explicitly acknowledges their agent-dependence and enables the detection of personalized affordances for human or artificial agents. It therefore allows to model differences in human perception of the same environment and the resulting behavior. In addition, it increases the flexibility and adaptivity of the model, which is due to the fact that new affordances can easily be introduced without the need to manipulate the original geo-spatial data. Moreover, it is thus possible to detect higher-order affordances which arise from the dynamic interplay of multiple spatial primitives, referred to as functional place, in an automated fashion, a task which would not be possible if affordances were pre-stored as environmental attributes. The construction of these functional places is based on the deployment of a suitability-based optimization strategy and the determination of a situation-sensitive action strategy, and provides novel ways for modeling possible behavioral adaptations, for instance by altering the set of used geo-atoms, or in other words, changing the extent of the functional place used for reaching the goal state, by following a different action strategy, or by changing the attributes of the agent in order to increase the resulting suitability.

## 4.2 Computational Model

This chapter describes one possible way to computationally implement the theoretical concepts. It explains the development of a software agent with the ability to detect higher-order affordances within its environment, calculate individual suitability values, generate a functional place with regards to a complex pragmatic action, and act accordingly. The process is implemented in two separate programming environments: While the agent and its environment are programmed in the NetLogo multi-agent simulation environment (Wilensky 1999), the place formation process exceeds NetLogo's functionality and is thus, apart from the suitability calculation procedures, programmed separately in the programming language Java, resulting in PlaceBuilder, a new extension package for NetLogo. A class diagram can be seen in figure 16 which, however, includes only the classes related to the conceptual model while omitting less relevant elements of the program such as factories, reporters, additionally used software packages, or primitive managing classes. For the full code please see Appendix A. In the following, each aspect of the computational model is explained in detail, using code listings where appropriate for clarification.

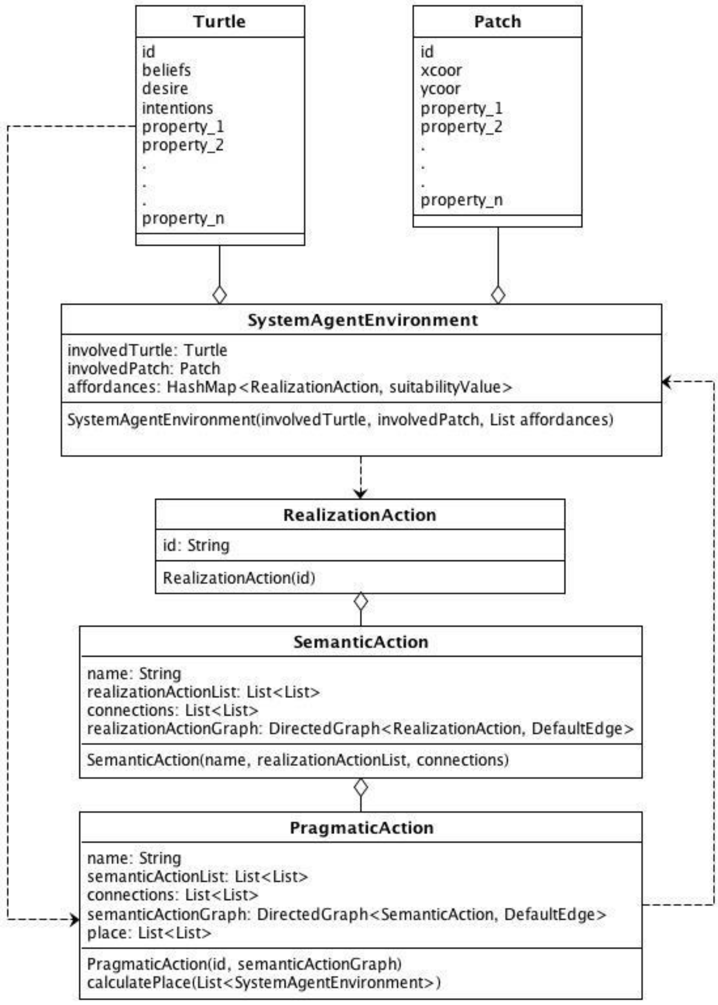


Fig. 16: PlaceBuilder Class Diagram

### 4.2.1 Agents and the Environment

NetLogo is an agent-based, open-source simulation environment which, since its development in 1999, has constantly risen in popularity as a tool for researchers from various disciplines (Wilensky 1999). Due to its flexibility, ease of use and capability for the creation of models of high complexity, it is currently considered one of the best agent-based modeling environments (Abdou et al. 2012). At the same time, the name NetLogo refers to the multi-agent modeling language used for programming the simulations, which runs on the Java virtual machine. In NetLogo, virtual agents (turtles) act in and interact with the environment, which is modeled as a tessellation of grid cell objects (patches), that can take on attributes and behave in accordance to predefined rules (Wilensky 1999). The class declarations of *Turtle* and *Patch* are accessible online[1], and are not further discussed here. Instead, code listing 1 shows an example creation of a new turtle in our model.

```
;creates one turtle at the origin of the spatial reference system
to createTurtles

  create-turtles 1
  [
    ;;set turtle properties
    set property_1 ... ;some property
    set property_2 ...
    set property_n ...

    ;;initialize beliefs and intentions as empty lists
    set beliefs []
    set intentions []

    ;;set agent desire
    set desire goalTest self ;calls specific goalTest function

    ;;add initial beliefs and intentions
    add-belief create-belief "beliefType_1" ... ;belief type and any NetLogo datatype
    add-intention "intention_1" "intentionRealized?" ;intention and its condition (done)
  ]

end
```

Listing 1: Exemplary Turtle Instantiation (NetLogo)

The turtle has several variables, which are assigned values during the instantiation, including a set of properties, which can refer to any characteristic or store distinct agent states. Further, as implementation of the BDI concept, the beliefs and intentions are stored as lists, while a desire variable calls an agent-specific goal test function. For the representation of beliefs and intentions, our model is based on work done by Sakellariou et al. (2008), who developed a BDI framework for NetLogo which is freely accessible[2]. The procedures *add-belief*, *create-belief*, and *add-intention* which are used in our example refer to their implementation, which is included in the full NetLogo code in Appendix B. Beliefs are represented as lists consisting of two elements, a type which takes a string to denote the class of the belief, in our example "*beliefType_1*", and the belief content, which can be of any data type supported in NetLogo. Similarly, intentions are stored as a two element list, and include the name of a NetLogo procedure and a done-condition which refers to a NetLogo reporter, in listing 1 "*intention_1*" and "*intentionRealized?*". Intentions are stored on a stack and executed in order until their respective done-condition reports true (Sakellariou et al. 2008).

The environmental model consists of NetLogo patches, which, in accordance with the notion of geo-atoms of Goodchild et al. (2007), associate a location in space-time with

[1] https://github.com/NetLogo/NetLogo

[2] http://users.uom.gr/~iliass/projects/NetLogo/code/bdi.nls

one or more properties and values for these properties. Although not a point but a raster cell, patches are the smallest, atomic spatial entity of our model, which can form larger objects based on rules. Although the use of vector objects is supported in NetLogo, a raster-based representation is preferred since it corresponds more closely to Gibson's (1979) emphasis of the role of surfaces for the perception of affordances.

### 4.2.2 The Suitability Calculation Procedures

As has been stated already, the process of suitability calculation is detached from the actual place formation process done within the PlaceBuilder extension, but instead takes place within the main program in NetLogo. The reason for this separation lies in the fact that the exact procedure of calculating the $suitability_{ij\alpha''}$ of a system $W_{ij\alpha''}$ can be highly specific to each part of the system. Thus, every realization action defines a different way of suitability calculation, but also, for instance, different agents with their individual properties might require dissimilar suitability calculation methods with regards to the same realization action $sub\text{-}action_{\alpha''}$ and $environmental\ primitive_j$. Imagine, for instance, two agents walking up a steep, slippery ramp, with the first one wearing non-slip work shoes, while the second one is not. Then, in a simplified model, the suitability calculation procedure with regards to the first agent could potentially neglect the slippery character of the surface, and focus exclusively on its steepness. Due to the high potential for variation in suitability calculation methods, therefore, it would not make sense to define a fixed procedure for the entirety of possible situations. Rather, the modeler is given the freedom to define specific methods directly within NetLogo, and simply pass the received suitability values to PlaceBuilder for the calculation of an optimized place, which is further described in Chapter 4. 2. 5. This functional place describes the selected sub-set of turtle-patch-action systems $W_{ij\alpha}$ which afford the respective pragmatic $action_\alpha$ with the highest possible value for $suitability_{ij\alpha}$. This information is then returned to the main program in NetLogo to inform the turtle which realization $sub\text{-}actions_{\alpha''}$ to perform, and at which patches.

Despite the specificity of each suitability function, however, listing 2 shows a general framework for a suitability calculation procedure for illustration purposes. Typically, the procedure starts with the initialization of the suitability variable, followed by the definition of the relevant capability value $cap_{ij\alpha''}$, for instance, a measure of the maximum tolerated slope of an agent walking up a ramp, and the disposition value $disp_{ij\alpha''}$, which could for example point to the actual gradient of the ramp. As can be seen in the example code, these variables might either refer to just one turtle or patch property, or combine several to a new value, as in the case of *capRealAction_1*. In our example, a multi-value $disp_{ij\alpha''}$ could for instance combine the ramp's steepness and slipperiness to one descriptive index value. In the next step, the values for $\pi$, $\pi_0$ and $\pi_{max}$ or $\pi_{min}$ are defined, referring for example to $\pi_0 = 0$ and $\pi_{max} = 1.01$ for a ramp's steepness with regards to the maximum tolerated value, meaning here that the optimal suitability value of 1 should eventually be reached with the ramp's gradient either tending towards 0 or being largely exceeded by the maximum tolerance value of the agent. If the ramp's steepness, however, exceeds the agent's capability, meaning that $\pi \geq \pi_{max}$, the suitability will be set to $0$ and the realization action *WALKING UP THE RAMP* is not afforded in this system $W_{ij\alpha}$. This dependency relationship between $\pi$ and the resulting suitability is based on a specific dependency function, in the exemplary case a linear one.

```
;calculate suitability values for realizationAction_1
to-report calcSuit_realAction_1  [involvedTurtle involvedPatch]

  ;;initialize suitability for realization action
  let suit_realAction_1  0

  ;;define capability and disposition values for realization action
  let capRealAction_1 [property_1] of involvedTurtle + [property_2] of involvedTurtle ;capability
  let dispRealAction_1 [property_1] of involvedPatch ;disposition

  ;;define Pi-Value, P-Max and Pi-O values
  let PiRealAction_1 dispRealAction_1 / capRealAction_1 ;Pi-Value
  let PiMaxRealAction_1  1.01 ;Pi-max, critical upper Pi-Value
  let PiORealAction_1 0 ;Pi-optimal, optimal Pi-Value

  ;;calculate suitability value
  if PiRealAction_1  < PiMaxRealAction_1
  [
    ifelse PiRealAction_1  <= PiORealAction_1
    [ set suit_realAction_1 1 ]
    [ set suit_realAction_1
      ((PiRealAction_1 - PiMaxRealAction_1) / (PiORealAction_1 - PiMaxRealAction_1)) ]
  ]

  ;;report suitability value
  report suit_realAction_1

end
```

Listing 2: Exemplary Suitability Calculation Procedure (NetLogo)

### 4.2.3 The SystemAgentEnvironment Class

In accordance with the conceptual model, a turtle forms a dynamic system with each perceived patch. This process is implemented as the instantiation of a new *SystemAgentEnvironment* object. Listing 3 shows the class definition within the PlaceBuilder extension. Apart from the involved turtle and patch, each *SystemAgentEnvironment* has an *affordances* attribute, which stores instances of *RealizationAction* together with their respective suitability value in a Java *HashMap*. This approach to computationally representing affordances is novel and provides the semantic connection between agent, environment and action in a way which corresponds closely to Stoffregen's (2003) formalized notion of affordance. In listing 3, please note that the constructor takes an input of type *List*, which is then used to generate the *HashMap* for the *affordances* attribute. This is necessary since the *HashMap* data type is not supported in NetLogo.

```java
public class SystemAgentEnvironment {

    public Turtle involvedTurtle;   //involved Turtle instance
    public Patch involvedPatch; //involved Patch instance
    public HashMap affordances; //HashMap with RealizationAction
                                //(key), and suitability (value)

    public SystemAgentEnvironment(org.nlogo.api.Turtle turtle,
            org.nlogo.api.Patch patch, List affordances) {
        this.involvedTurtle = (Turtle) turtle;
        this.involvedPatch = (Patch) patch;
        this.affordances = new HashMap();

        //create HashMap from input (nested list from NetLogo)

        for (int i = 0; i < affordances.size(); i++) {
            List sublist = (List) affordances.get(i);
            RealizationAction key = (RealizationAction)
                    sublist.get(0);
            Double value = (Double) sublist.get(1);
            this.affordances.put(key, value);
        }
    }
}
```

Listing 3: SystemAgentEnvironment Class (Java)

In listing 4, an exemplary instantiation of a new *SystemAgentEnvironment* instance in NetLogo is shown. For each realization action, the respective suitability calculation procedure is called which takes the involved turtle and patch as inputs. If the returned value is above 0, the action is afforded, and added to a list of affordances together with its suitability value. When the process is finished, a PlaceBuilder extension primitive is used to create a new instance of *SystemAgentEnvironment*.

```
;calls suitability calculation functions for realization actions,
;and creates SystemAgentEnvironment instance for an involved turtle and a patch
to createSystemAgentEnvironment [ involvedTurtle involvedPatch ]

   ;;calculate suitability for realizationAction_1, and add to affordanceList if afforded
   if calcSuit_realAction_1 involvedTurtle involvedPatch != 0
    [ set affordanceList lput (list realAction_1 calcSuit_realAction_1 involvedTurtle involvedPatch)
      affordanceList ]

   ;;calculate suitability for realizationAction_2, and add to affordanceList if afforded
   if calcSuit_realAction_2 involvedTurtle involvedPatch != 0
    [ set affordanceList lput (list realAction_2 calcSuit_realAction_2 involvedTurtle involvedPatch)
      affordanceList ]

   ;;create an instance of SystemAgentEnvironment
   PlaceBuilder:createSystemAgentEnvironment involvedTurtle involvedPatch affordanceList

end
```

Listing 4: Initialize an Instance of SystemAgentEnvironment (NetLogo)

### 4.2.4 The Action Model

The action model is implemented within PlaceBuilder, and consists of the three classes *RealizationAction*, *SemanticAction* and *PragmaticAction*. Before explaining the class declarations, however, listing 5 shows how an exemplary action model could be created in

NetLogo. When instantiating the realization actions, only a string input is needed to set the name attribute.

```
;creates an hierarchically structured action model
to createActionModel [ involvedTurtle ]

  ;;create realizationAction instances
  set realAction_1 PlaceBuilder:createRealizationAction "realAction_1"
  set realAction_2 PlaceBuilder:createRealizationAction "realAction_2"
  set realAction_3 PlaceBuilder:createRealizationAction "realAction_3"

  ;;create semanticAction instances
  set semAction_1 PlaceBuilder:createSemanticAction "semAction_1"
      (list (list realAction_1 0.5) (list realAction_2 1))
      (list (list realAction_1 realAction_2 "sameSystem"))

  set semAction_2 PlaceBuilder:createSemanticAction "semAction_2"
      (list (list realAction_3 1))
      []

  ;;create pragmaticAction instance
  set pragAction PlaceBuilder:createPragmaticAction "pragAction"
      (list (list semAction_1 1) (list semAction_2 0.75))
      (list (list semAction_1 semAction_2 "sameSystem"))

  ;;store action model as belief
  add-belief create-belief "actionModel" pragAction ;store as belief

end
```

Listing 5: Creation of an Action Model (NetLogo)

For the creation of *SemanticAction* instances, however, two further input variables of type *List* are needed. As can be seen in the class declaration of *SemanticAction* in listing 6, in the attribute *realizationActionList*, each *RealizationAction* instance which contributes to the respective *SemanticAction* is stored together with their corresponding weight coefficient in a nested list, and passed to the constructor. Since the weight coefficient, however, is not hard-coded as an attribute of the *RealizationAction* instance, but only defined in the context of instantiating a *SemanticAction*, it is possible that, in case that a realization action contributes to different semantic actions, an agent attributes different relative importance to the same realization action, a measure which provides additional freedom to the modeler. How the contributing realization actions are functionally connected, however, for instance as sequential steps or interchangeable alternatives, is stored in the second list variable, which is named *connections*. In addition, this variable contains information about their spatial dependencies, for example “*sameSystem*” as in listing 5, which means that the realization actions must occur at the same patch. However, in the particular case that an agent is required to perform only one action to realize a semantic action, or in our implementation, if a *SemanticAction* instance takes only one contributing *RealizationAction* instance, no connections would be needed, as can be seen on the example of *semAction_2* in listing 5.

From these inputs, a graph is created using the Java graph library JGraphT[3]. As already described in Chapter 3. 1. 3. 2, the use of a graph structure is a common approach to represent planning problems in AI. In our action model, as can be seen in listing 6, a *SemanticAction* instance stores all possibilities of its realization in an attribute *realizationActionGraph*. In this graph, all contributing *RealizationAction* instances are stored as nodes which, depending on their functional dependency, are connected with directed edges. In addition to simply connecting the realization actions, each edge carries requirements concerning the spatial dependencies between the patches which can be involved in the performance of the *RealizationAction* instance, as described in the previous Chapter 4. 1. 2, using the example of an open door to be *CLOSED* and *LOCKED*, which must involve not different doors but the same one to be successfully performed. Every path through the graph from a source (no incoming edges) to a sink node (no outgoing edges) represents one possible sequence of *RealizationAction* instances which would lead to the semantic action to be realized, causing the intended state change.

[3]http://www.jgrapht.org

```java
public class SemanticAction {
    public String name; //name of the SemanticAction
    public List<List> realizationActionList;    //contributing RealizationAction
                                                //and their weight factors in list
    public List<List> connections;  //list with connected RealizationAction instances
                                    //and their spatial dependencies

    //stores connections of contributing RealizationAction instances as directed graph
    public DirectedGraph realizationActionGraph =
            new DefaultDirectedGraph<RealizationAction, DefaultEdge>(DefaultEdge.class);

    public SemanticAction(String name, List<List> realizationActionList,
            List<List> connections) {
        this.name = name;
        this.realizationActionList = realizationActionList;
        this.connections = connections;

        //build realizationActionGraph

        for (int i = 0; i < realizationActionList.size(); i++) {

            //create a vertex for each RealizationAction instance

            RealizationAction element =
                    (RealizationAction) realizationActionList.get(i).get(0);
            realizationActionGraph.addVertex(element);
        }

        for (int i = 0; i < connections.size(); i++) {

            //create a LabeledEdge instance for each tuple in connections variable
            //and put the spatial dependency (string) as label

            List edge = connections.get(i);
            Object from_node = edge.get(0);
            Object to_node = edge.get(1);
            String label = (String)edge.get(2);
            realizationActionGraph.addEdge(from_node, to_node, new LabeledEdge(label));
        }

    }

}
```

Listing 6: SemanticAction Class (Java)

Every state change, or *SemanticAction* instance, in turn, serves the purpose of achieving the agent's goal, the *PragmaticAction*. In a similar fashion, therefore, as shown in listing 7, the constructor takes the lists *semanticActionList* and *connections* as inputs, which define the contributing *SemanticAction* instances, their weight coefficients which are specific with regards to their contribution to the *PragmaticAction*, the functional connections and spatial dependencies. Again, these inputs are used to create a directed graph, the *semanticActionGraph*, for each pragmatic action. In contrast to the *SemanticAction* class, however, *PragmaticAction* has an additional attribute *place*, and a method *calculatePlace*, which serve for the calculation of a place for the respective pragmatic action. Since this process is a vital part of this thesis, however, it is described separately in the following chapter.

```java
public class PragmaticAction {
    public String name;     //name of the PragmaticAction

    //stores connections of contributing SemanticAction instances as directed graph
    public DirectedGraph semanticActionGraph =
            new DefaultDirectedGraph<SemanticAction, DefaultWeightedEdge>(DefaultWeightedEdge.class);
    public List<List> semanticActionList;   //contributing SemanticActions
                                            //and their weight factors in list
    public List<List> connections;  //list with connected SemanticAction instances
                                    //and their spatial dependencies
    public List<List> place = new ArrayList();  //stores place as list of lists

    public PragmaticAction(String name, List<List> semanticActionList, List<List> connections) {
        this.name = name;
        this.semanticActionList = semanticActionList;
        this.connections = connections;

        //build semanticActionGraph based on semanticActionList and connections

        for (int i = 0; i < semanticActionList.size(); i++) {

            //create a vertex for each SemanticAction instance in semanticActionList

            SemanticAction element = (SemanticAction) semanticActionList.get(i).get(0);
            semanticActionGraph.addVertex(element);
        }

        for (int i = 0; i < connections.size(); i++) {

            //create a LabeledEdge instance for each tuple in connections variable
            //and put the spatial dependency (string) as label

            List edge = connections.get(i);
            Object from_node = edge.get(0);
            Object to_node = edge.get(1);
            String label = (String)edge.get(2);
            semanticActionGraph.addEdge(from_node, to_node, new LabeledEdge(label));
        }
    }
```

Listing 7: PragmaticAction Class Excerpt (Java)

### 4.2.5 The Place Calculation Process

From within NetLogo, the place calculation process is triggered by using PlaceBuilder's primitive *reportPlace*, which takes a *PragmaticAction* instance and a list of *SystemAgentEnvironment* instances, and calls the pragmatic action's method *calculatePlace*, as can be seen in listing 8. In the exemplary procedure, which would be called by a turtle, the list of *SystemAgentEnvironment* instances could for instance refer to the current percepts of the turtle. The *PragmaticAction* instance is received from the belief structure.

```
;calculates a place for a list of SystemAgentEnvironment instances
to-report calculatePlace [ involvedSystems ]

  ;;receive pragmaticAction instance from agent belief
  let pragmaticAction last read-first-belief-of-type "pragmaticAction"

  ;;call calculatePlace method from pragmaticAction instance
  let place PlaceBuilder:reportPlace pragmaticAction involvedSystems

  ;;store place as belief
  add-belief create-belief "place" place ;store as belief type "place"

end
```

Listing 8: Exemplary place calculation (NetLogo)

The place calculation process is too extensive to have its full code presented here, which is why its general workflow is illustrated in figure 17. For further details see Appendix A.

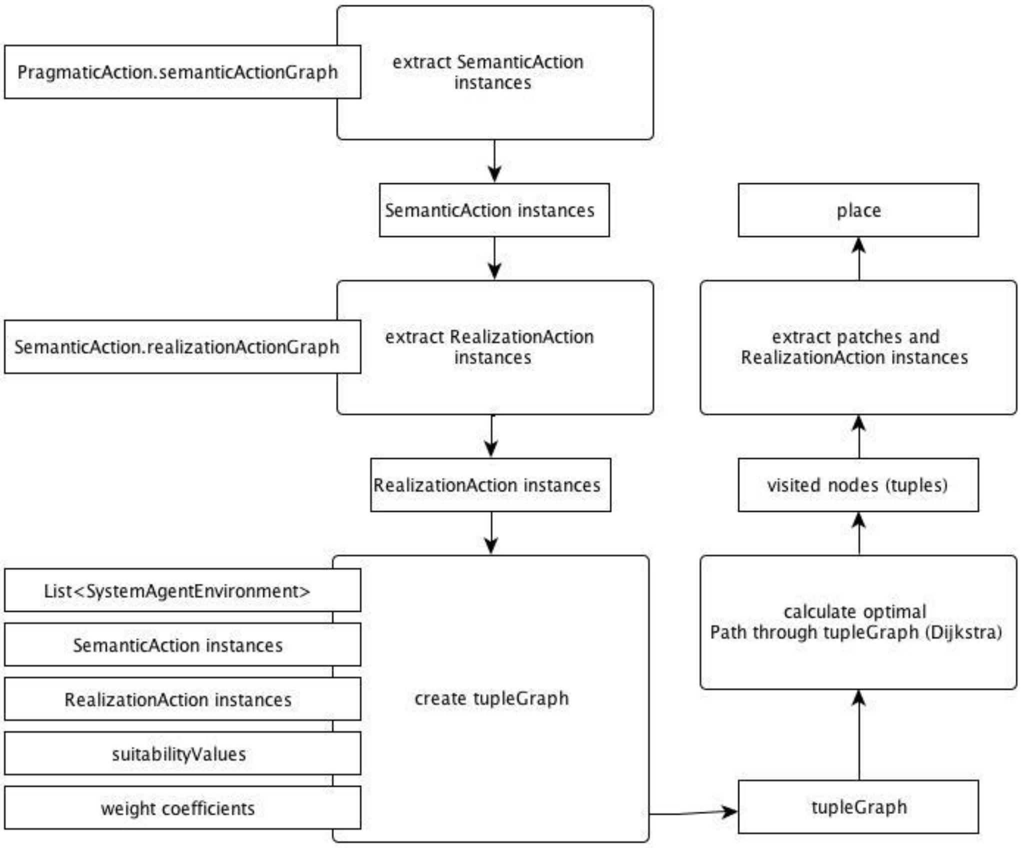

Fig. 17: Process of Place Calculation

By default, the *PragmaticAction* instance's *place* attribute will be empty. Based on a list of all *SystemAgentEnvironment* instances available for performing the pragmatic action, the procedure starts off with an analysis of the *semanticActionGraph* attribute, identifying each contributing instance of *SemanticAction.* For each of these semantic actions, in turn, their contributing *RealizationAction* instances are received from the *realizationActionGraph* attribute. On the basis of the available instances of *SystemAgentEnvironment*, their afforded realization actions as well as any spatial dependencies (such as "*sameSystem*"), all possibilities of achieving the pragmatic action in the respective situation can now be identified. For this purpose, a new graph *tupleGraph* is built with tuples as nodes. Each of these tuples consists of one instance of *SystemAgentEnvironment*, *RealizationAction* and *SemanticAction*, and is connected with edges to other tuples according to the existing connections in the *realizationActionGraph* and *semanticActionGraph* attributes. For the creation of *tupleGraph*, in a first step, for each *SemanticAction* instance in *semanticActionGraph*, all contributing *RealizationAction* instances are identified, and tested for their affordance with regards to each available *SystemAgentEnvironment* instance. If they are afforded, a node is created in *tupleGraph* which consists of a HashMap which stores a tuple of *SystemAgentEnvironment* instance, the *RealizationAction* instance, and the corresponding *SemanticAction* instance as key and the respective suitability value as value.

After the nodes have been created, the *tupleGraph* edges need to be determined. This requires a two-step process: First, the edges are created between vertices which correspond to the same *SemanticAction* instance based on the functional connections of the *realizationActionGraph* attribute of this particular *SemanticAction* instance. Thus, for each *SemanticAction* instance, all *RealizationAction* instances are identified which are connected in its *realizationActionGraph*, and their respective tuples tested for spatial dependency. If the spatial requirement is fulfilled with regards to the patches contributing in the *SystemAgentEnvironment* instance, an edge between these *tupleGraph* nodes is established, and the inverse suitability value, after being multiplied with the weight coefficients of both the *RealizationAction* and the *SemanticAction* instance, added as cost value.

In a second step, the vertices of *tupleGraph* must be connected which correspond to *SemanticAction* instances which are connected in the pragmatic action's *semanticActionGraph* attribute. For this, the connections are extracted from *semanticActionGraph*, and the sink and source nodes in their *realizationActionGraph* are identified. The corresponding *tupleGraph* nodes, meaning the ones which correspond to the correct *SemanticAction* and *RealizationAction* instances, are identified and, if they fulfill the spatial requirements, connected with edges, which are again attributed the inverse suitability value multiplied with the weight coefficients of both action instances.

After its creation, it is possible to use a Dijkstra-Algorithm to calculate the least cost path through the *tupleGraph*, which represents the optimal action strategy for the agent in this particular situation. The corresponding instances of *SystemAgentEnvironment* and *RealizationAction* are extracted and stored as pairs in the *place* attribute of the respective *PragmaticAction*. In NetLogo, the PlaceBuilder's primitive *reportPlace* returns this *place* variable, which can then be used by a turtle to perform the appropriate actions involving the right patches.

### 4.2.6 Conclusion: Generating Places with PlaceBuilder

This chapter described a possible computational implementation of our conceptual model of functional place in the specific context of an agent-based simulation. The development and functionality of a new extension package PlaceBuilder for the multi-agent simulation environment NetLogo was explained. In comparison to the current methods of modeling agent-environment interactions, as described in Chapter 3. 1, PlaceBuilder allows to model high-level actions as hierarchical structures, with several supporting sub-actions on lower levels which are semantically connected in complex causal and spatial dependency relationships. Furthermore, instead of hard-coding the semantic connection between an agent, its actions and the environment, for instance by pre-defining affordances as attributes of the environment, it is computed on-the-fly based on specific suitability calculation procedures. Incorporated in the artificial agent's reasoning engine, therefore, it enables it to actively interpret its percepts in terms of identifying higher-order affordances, assessing the subjective suitability, and determining an optimized behavioral strategy and the corresponding functional place. In our implementation, the modeler is free to define action models, suitability calculation procedures, individual, context-dependent weight coefficients or spatial dependencies. Thus, PlaceBuilder provides a powerful framework to model complex spatial actions, and, since the suitability calculation delivers individual values tailored to the specific combination of agent and environment, a step forward for the integration of subjectivity in ABM and GIS, as well as a novel approach to utility-based agent modeling.

# 5 Case Study - Modeling Places for Walking

In this chapter, an agent-based simulation is presented which puts the PlaceBuilder extension to a concrete use. This is done in order to demonstrate the practical value of the computational model of place and test the assumptions made when developing the theoretical concepts. For this, in accordance with the thematic focus of Chapter 3. 2, the concrete application area is pedestrian movement. Thus, as a first step, based on the empirical findings about micro-scale walkability as presented in Chapter 3. 2. 2, and in accordance with the theoretical concepts developed in Chapter 4. 1, the complex action *WALK* and its according suitability, walkability, is modeled. Then, on this basis, an agent-based pedestrian simulation is conducted in a study area located within the city center of Augsburg, Germany. In order to test the simulation model, test scenarios are run as well as a sensitivity analysis conducted. Finally, in order to connect our notion of individually perceived functional places to emerging collective places, a brief example is provided for the analysis and visualization of highly-frequented pedestrian places which emerge from multiple simulation runs.

## 5.1 A Model of Subjective Micro-Scale Walkability

In prior approaches to pedestrian simulation, as presented in Chapter 3. 2. 1, *WALKING* has often been equated with physical movement. The variety of environmental factors which have been identified as significant for our walking behavior, as discussed in Chapter 3. 2. 2, however, demonstrates the richness of the concept of walkability, which is one motivation for developing a more complex action model of *WALKING*. Also, walkability has so far been treated as an objective quality of the environment. At the same time, however, the need to incorporate individual differences among pedestrians and their perceptions of the urban walking environment has been formulated (e.g. Alfonso 2005, Buchmueller and Weidmann 2006). In this chapter, therefore, the aim is to develop a model of subjective micro-scale walkability as an exemplary application of our concept of spatial suitability to a real-world topic, the suitability of the urban environment for walking. The model development is based on empirical knowledge on micro-scale walkability.

According to the conceptual requirements of our theoretical framework for suitability assessment explained in Chapter 4. 1, the process of modeling walkability requires several steps: First, an hierarchical action model has to be developed, which means that walking has to be modeled as a complex pragmatic action, with all its potentially contributing sub-actions on the lower hierarchical levels being identified and modeled separately. Then, for each sub-action on the realization level of abstraction, the suitability needs to be calculated, which includes the identification of relevant capabilities of the pedestrian and the dispositions of the walking environment, and the definition of $\pi$,

$\pi_O$ and $\pi_{max}$ or $\pi_{min}$ values. Further, an appropriate dependency function for mapping $\pi$ to suitability values needs to be determined.

### 5.1.1 Modeling the Hierarchical Structure of $action_\alpha$ WALK

In accordance with Kemke's (2001) approach, walking is understood as a pragmatic action, which can be further broken down into its contributing sub-actions. The action model of walking is developed on the basis of both Alfonzo's (2005) walking needs model, and the review of empirical literature on micro-scale walkability, as described in Chapter 3. 2. Transferring the concept of Alfonzo (2005) to our conceptual framework, the walking needs can be understood as agent states which, in their entirety, represent the goal state that the $action_\alpha$ *PA_WALK* describes. Thus, apart from feasibility, which can be omitted since it does not refer to the built environment, the remaining pedestrian needs can be directly translated into semantic $sub\text{-}actions_{\alpha'}$. The state changes are restricted to the agent, since, in the particular case of walking, the purpose is not to change the environment, but rather one's own state, for instance concerning one's location.

The need for accessibility describes the requirement of pedestrians to be able to reach destinations, both with regards to the overall trip distance as well as to potential barriers for walking (Alfonzo 2005). In our action model, we translate this need to the semantic $sub\text{-}action_{\alpha'}$ *SA_AGENTMOVE*. After the definition of a semantic action, its contributing realization $sub\text{-}actions_{\alpha''}$ need to be identified, which is done based on the results of empirical studies on walkability, as reviewed in Chapter 3. 2. 2. 2. The overall trip distance, not being a micro-scale walkability indicator, is omitted here. A fundamental action of a pedestrian on the micro-scale, however, is to physically move on the sidewalk. For this, the sidewalk needs to be wide enough (Lynch and Rivkin 1959, Samarasekara et al. 2011) with a lack of obstructions (Sanches et al. 2007, Samarasekara et al. 2011). Thus, the first realization action *RA_ACCESSPATCH* denotes the agent's ability to physically fit on the unobstructed sidewalk space. Further barriers to movement might be represented by insufficient surface quality (Lynch and Rivkin 1959, Sanches et al. 2007, Agrawal et al. 2008, Sugiyama et al. 2012, McCormack and Shiell 2011). Notably, in an empirical study which asked pedestrians to evaluate the overall quality of the inner city of Augsburg, our study area, this particular aspect has been shown to be a particular nuisance (Heller and Monheim 2004). In fact, an uneven surface might lead to falls, which actually account for 80% of pedestrian injuries in OECD countries (La Feypell-De Beaumelle et al. 2010). Therefore, the next realization $sub\text{-}action_{\alpha''}$ to be introduced is *RA_KEEPBALANCE*. Movement can also be hindered by slopes and stairs (Borst et al. 2008), which especially concerns mobility-impaired persons. As several empirical studies have shown, the energy consumption required for walking of regular pedestrians increases sharply with the gradient of the walking surface, and doubles at around 10% inclination (Weidmann 1993). To acknowledge the additional effort which is necessary to walk up a hill, a further $sub\text{-}action_{\alpha''}$

is *RA_OVERCOMESLOPE*. Concerning stairs as potential barriers for movement, it is less the gradient, but rather the height of the individual steps which represents a particular challenge, especially for wheelchair drivers. Apart from stairs, curb stones have also been mentioned as problematic, and might drastically restrict the potential road crossing locations for mobility-impaired persons (Clifton and Livi 2004). Since the action of stepping over an obstruction is different from regular walking, it is modeled as a separate *sub-action*$_{\alpha''}$ *RA_OVERCOMEHEIGHTDIFFERENCE*. Finally, apart from purposeless wandering in the terms of Ronald et al. (2007), walking is normally destination-bound, which is why an additional *RA_REACHDESTINATION* is created, which captures the pedestrian's action of actively reducing the distance to its goal.

Apart from a change in location, however, *PA_WALK* affects additional agent states, and accordingly, consists of other semantic *sub-actions*$_{\alpha'}$ as well. According to Alfonzo (2005), in relative importance, accessibility is followed by safety. With the semantic action *SA_AGENTUPDATESAFETY*, therefore, a pedestrian's mental state of feeling safe is described. Potential sources of threat are risks related to traffic accidents and crime (Painter 1996, Martin 2006). Both aspects have been found to strongly correlate with observed walking frequencies (e.g. Saelens and Handy 2008, Sugiyama et al. 2012). Critical safety-related aspects have been found to include physical separation from traffic (Lynch and Rivkin 1959, Clifton and Livi 2004, Kaufmann et al. 2010, Samarasekara et al. 2011, Adkins et al. 2012), and appropriate street crossing facilities (Lynch and Rivkin 1959, Agrawal et al. 2008, Borst et al. 2008, Kaufmann et al. 2010). Since in our model, road crossing is treated as a separate semantic action, for safety, realization *sub-actions*$_{\alpha''}$ *RA_KEEPDISTANCETOMOTORTRAFFIC* and *RA_KEEPDISTANCETOCYCLETRAFFIC* are modeled. The two transportation modes are distinguished due to the fact that although bicycles pose risks to pedestrians, and there is a need for an adequate buffer between both cyclists and pedestrians, accident statistics show that pedestrian-motorized vehicle accidents are still far more frequent, although under-reporting seems to have an influence as well (Martin 2006, Hakkert 2010). Due to its prominence in empirical and qualitative studies on pedestrian perceptions, the need for street lighting is also included in our model as *RA_STAYNEARLIGHT*, which reduce the risk of traffic accidents and crime (Hakkert 2010).

Furthermore, all frameworks include some measures of walking comfort and pleasurability (Sarkar 1993, Pikora et al. 2002, Alfonzo 2005). Compared to the previous aspects, the effects of comfort and pleasurability on pedestrian perceptions are less undisputed in the literature and more complicated to operationalize. There are, in fact, inconsistent findings regarding whether people walk more or longer in comfortable, attractive environments, although studies have shown that walking distances in central urban areas grow to a certain degree with an increasing restorative and aesthetic value of the surrounding environment (e.g Monheim and Raab 2008). Both comfort and pleasurability are related to the pedestrian's ability for mental restoration while walking.

Although originally developed in the context of natural settings, Kaplan and Kaplan's (1989) attention restoration theory provides a conceptual framework to explain stress reduction and has also been applied to urban environments (e.g. Lindal and Hartig 2013). Transferring the concept to walkability, it can be postulated that comfortable walking should neither require a pedestrian's focused attention nor hinder the effortless attention necessary for recovery. Accordingly, as table 1 in Chapter 3. 2. 2. 2 illustrates, environmental characteristics related to comfort or pleasurability, such as aesthetics, public parks and greenery, the absence of disturbances by motorized traffic, as well as the availability of street furniture such as benches have been found significant for walking outcome and perceptions of walkability (e.g. Badland and Schofield 2005, Handy 2005, Saelens and Handy 2008, McCormack and Shiell 2011). Alfonzo (2005) distinguishes walking comfort from pleasurability, defining comfort as the "level of ease, convenience, and contentment" for walking, while pleasurability refers to the appealing character of the urban setting, which depends to a high degree on its visual quality (Alfonzo 2005, p. 828).

Transferred to our conceptual framework, the need for comfortable walking conditions translates to a semantic action *sA_AgentUpdateComfort*, which is realized by *rA_NotSeeTraffic* and *rA_StayNearBenches*. The first realization action describes the pedestrian trying to keep either at a certain distance from streets, or try to have an object, such as a building, in between him- or herself and the road, in order to be protected from negative sensory inputs. The second sub-action focuses on the possibility to take breaks while walking.

The aspect of pleasurability is captured in *sA_AgentUpdatePleasure*. In order to achieve this state, and evoke mental restoration, a pedestrian should maintain a visibility relation with aesthetically valuable environmental settings (Hidalgo et al. 2006, Lindal and Hartig 2013). The operationalization of aesthetics, however, is not a trivial task. In the literature, a variety of indicators for aesthetic environments have been proposed, including urban vegetation (Alfonzo 2005), architectural variation (Lindal and Hartig 2013), the historical value of buildings (Hidalgo et al. 2006) and complex urban design attributes such as imageabiliy, human scale or complexity (Ewing et al. 2006). In pedestrian studies, the presence of urban greenery has been particularly significant for the perception of the aesthetical value of urban areas (Lynch and Rivkin 1959, Borst et al. 2008, Kaufmann et al. 2010 Adkins et al. 2012). For assessing the visual quality of buildings, however, the results have been less unambiguous, however, several empirical studies have found a strong relation between their perceived attractiveness and their historical character. In one study, for instance, pictures of buildings of different ages and states of maintenance were shown to test persons. It was found that in general, old but well-maintained buildings were preferred to newer ones (Herzog and Shier 2000). Similar findings were made when test persons were asked to identify the most visually attractive and unattractive place of their city, which showed that attraction correlates with historical or recreational value (Hidalgo et al. 2006). Thus, in the context of this study, it appears sufficient to introduce the historical value of the adjacent buildings as a substitute quality for the attractiveness of the urban environment (Jonietz and Timpf 2013b). Therefore, *SA_AGENTUPDATEPLEASURE* is realized by *RA_SEEGREENERY* and *RA_SEEAESTHETICALBUILDING*.

Finally, during the trip, a pedestrian will in most cases be required to cross a road. Although on average, crossings amount to less than 10 % of the walked distance, 74% of all accidents which involve pedestrians take place there (Walter et al. 2007). The presence of appropriate street crossing facilities has been found to be of relevance for pedestrian perceptions of safety (Lynch and Rivkin 1959, Agrawal et al. 2008, Borst et al. 2008, Kaufmann et al. 2010). Crossing a road is different from regular walking with regards to the actions required by a pedestrian, which is why we introduce a separate $sub\text{-}action_{\alpha'}$ *SA_AGENTCROSSROAD*. Based on models of the road crossing process such as Bönisch and Kretz (2009) or Fi and Igazvölgyi (2014), and observational studies such as Oudejans et al. (1996) or Oxley et al. (1995), three distinct realization actions are defined, *RA_EVALUATESTREETCROSSING*, *RA_ASSESSTRAFFICSITUATION*, and *RA_REACHSIDEWALK*. The first one is related to wayfinding, and describes the process of assessing whether a street crossing process is sensible in terms of effectively reducing the distance to the destination. The second $sub\text{-}action_{\alpha''}$ is related to the action of observing the traffic flow, and detecting an appropriate gap to safely cross the road. The actual process of moving across the street is captured by the third realization action.

### 5.1.2 Suitability Calculation Procedures

On the basis of the hierarchical structure of $action_{\alpha}$ *PA_WALK*, the suitability needs to be calculated for every sub-action on the realization level of abstraction. For this, the relevant properties of both agent and environment must be identified and values for the environmental disposition $disp_{ij\alpha''}$ and the pedestrian capability $cap_{ij\alpha''}$ calculated. Further, the critical threshold values $\pi_0$ and $\pi_{max}$ or $\pi_{min}$ values need to be defined and an appropriate dependency function for mapping $\pi$ to suitability values developed.

Table 3 lists the properties $P_{i1}...P_{in}$ of $agent_i$ and $P_{j1}...P_{jn}$ of $environmental\ primitive_j$, from which the action-specific agent- and environment-related attributes are extracted in our model. Regarding the definition of capabilities, it is deliberately avoided to develop an overly detailed pedestrian model, separately incorporating, for instance, the full range of factors listed by Vukmirovic (2010) or Basbas et al. (2010). In contrast, due to issues of feasibility, and with regards to maintaining the practical value of the model as a planning tool, we rather choose to develop aggregated capabilities. Thus, to give an example, the ability to walk up an inclined surface depends on a bundle of pedestrian properties, such as strength, stamina and balance. If the influences of all factors were to be modeled separately, a detailed biometrical model would be needed to determine the exact dependency relationships between the particular capability and the environmental disposition. Further, the process of calculating suitability values would become overly complex, and require very detailed input data. Instead, the use of a less detailed, abstract property, such as the maximum slope which a pedestrian is able to overcome, aggregates the entirety of potentially influential pedestrian factors, and can be related to properties for which reference values already exist in the planning literature. Thus, for instance, in order to improve the accessibility of pedestrian infrastructure for everyone, official guidelines in Germany recommend sidewalk gradients of no more than 3% (FGSV 2011). The definition of higher-level capabilities allows the incorporation of such established values, and therefore improves the practical value, feasibility and straight-forward character of the model.

The action *RA_ACCESSPATCH* is related to a pedestrian's ability to fit on the sidewalk and through bottlenecks. The related capability, therefore, can be described as the total space needed for walking by a pedestrian. According to Buchmueller and Weidmann (2006), it is composed of the body width, the additional width which results from wavering movement while walking, and certain shy-away- or buffer distances kept to obstacles and building walls. In our model, the sum of these pedestrian-specific values defines the capability for *RA_ACCESSPATCH*. The corresponding environmental disposition is determined by the distance to the nearest obstacle. Streets and cycle lanes, however, are not counted as obstacles in this step, since, although not desirable, it is theoretically possible for a pedestrian to step off the sidewalk to walk around an obstacle. In the next step, the critical threshold values need to be determined. The action is afforded if the minimum space needs for walking are met, however, since additional space would be desirable, with a low suitability value. The $\pi_{min}$ value, therefore, is set to the smallest unit below 1, depending on the level of model accuracy for instance to 0.99, which has the effect that with a $\pi$ of 1, the suitability is above 0. The $\pi_O$ value defines the point at which the suitability reaches its optimum of 1, and determines the gradient of the mapping function. In our case, it is arbitrarily set to 5, which means that the provided space for walking is 5 times higher than needed by a pedestrian. Based on these values, the suitability can be calculated. We assume a linear mapping function:

$$suitability_{ij\propto''} = \frac{\pi - \pi_{min}}{\pi_O - \pi_{min}} \tag{10}$$

| Pedestrian Properties | Environmental Properties |
|---|---|
| minimum acceptable surface quality | classification into sidewalk, street, crossing, and cycleway |
| maximum walkable slope | slope |
| body width | surface quality |
| additional width due to dynamic body movements | walking distance to closest bench |
| preferred minimum buffer distance to streets | distance to closest streetlight |
| preferred minimum buffer distance to cycleways | number of visible historical buildings |
| preferred minimum buffer distance to walls | number of visible trees |
| maximum tolerated distance to the next street light | longest distance from which approaching cars are visible when crossing the road |
| maximum walking distance between benches | distance to nearest, unblocked street |
| curb delay | distance to nearest, unblocked cycleway |
| walking speed | allowed maximum traffic speed |
| maximum step height | minimum crossing distance |
| | curb/step height |
| | amount of visible street |
| | distance to nearest obstacle |

Table 3: Pedestrian and Environmental Properties

*RA_KeepBalance* corresponds to the ease of walking safely on surfaces with differing quality. There are notable differences among pedestrians, thus, while mobility-impaired persons require an even, well maintained walkway, for other persons, the walking quality is not considerably diminished by walkways of less quality, such as cobblestone or gravel paths (La Feypell-De Beaumelle et al. 2010). Apart from the rather permanent surface structure, a sidewalk’s walking quality can also be influenced by temporary phenomena, such as ice and snow, rain or fallen leaves, which might cause slipperiness, or the increase of tripping hazards due to litter or dirt on the ground. Such factors, however, are not incorporated in this model, the focus of which is put on aspects of the permanent, built environment. In particular, the unevenness of the surface structure has been identified as the major cause for falling incidents (LaFeypell-De Beaumelle et al. 2010). Based on this property, surfaces can be classified into several types, which then represent the environmental disposition for *RA_KeepBalance*. Table 4 shows the classification scheme applied in this study as an example. Regarding the pedestrians, differences in balance and stability, speed of reflexes, visual perception and energy expended in movement can result in variability concerning the maximum tolerated surface quality, which serves as higher-level capability in this model (Basbas et al. 2010). Since the capability denotes the minimum surface quality at which walking is still possible, the $\pi_{min}$ value is set to 0. The perfect case would be that a pedestrian has the lowest possible requirements, but the best surface structure type is provided. In accordance with table 4, therefore, $\pi_O$ is set to 5. For $\pi$ values above or equal to 1, suitability values can then be calculated using formula 10.

| Value | Surface Type |
|---|---|
| 1 | pebble or dirt<br>grass |
| 2 | pavement (uneven)<br>cobblestone (uneven, fugue > 2 cm) |
| 3 | cobblestone (even, fugue > 2 cm)<br>cobblestone (uneven, fugue < 2 cm) |
| 4 | cobblestone (even, fugue < 2 cm) |
| 5 | pavement (even)<br>tiles |

Table 4: Classification of Surface Types

The realization action *RA_OVERCOMESLOPE* refers to the higher levels of physical effort needed for movement along inclined surfaces. The biomechanical process of moving is complex, and expends energy for lifting and lowering as well as accelerating and decelerating the body. In order to change the potential energy of a body with regards to moving it to a different location, a higher force is needed in case of a positive slope. However, walking down a hill might also increase the energy expenditure due to additional force needed for reducing the body's potential energy caused by the gravitational pull (Weidmann 1993). Depending on their physical strength, balance and stamina, pedestrians can equal out the required additional force, and can thus tolerate higher gradients (Vukmirovic 2010). The suitability for *RA_OVERCOMESLOPE*, therefore, depends on the correspondence of the environmental disposition slope and an aggregated capability which denotes the maximum slope which is still possible or tolerated to be overcome by a pedestrian. Accordingly, $\pi_{max}$ is set to 1.01, in order to denote that in a system with a $\pi = 1$, the action is still afforded. The best suitability can be expected when there is no slope at all, so that $\pi_O = 0$. Again, formula 10 is used to calculate suitability values.

Climbing a stair step or curbstone, or in other words, performing *RA_OVERCOMEHEIGHTDIFFERENCE*, changes the pendulum movement of the leg, causing a higher lift as well as an alteration of the pedestrian's mass center. It therefore increases the necessary energy expenditure, which correlates with the step height, the step depth, as well as the gradient and length of the stair set (Weidmann 1993). Apart from the energy expenditure, however, height differences represent barriers to numerous pedestrians due to their inability to lift their body to this height, which depends on physical aspects such as muscular strength, balance and stability (Basbas et al. 2010). In this study, the influence of stairs on pedestrian movement is reduced to the step height, which is due to model simplicity and a lack of available data. Thus, an aggregated capability which describes the maximum step height a pedestrian is able to surmount is opposed by the actual height difference of a curb or a stair step relative to their surroundings as the environmental disposition. Similar to the previous case, $\pi_{max}$ is set to 1.01, and $\pi_O = 0$. Also, formula 10 is used to calculate suitability values.

The realization action *RA_REACHDESTINATION* describes the process of pedestrians directing their movement more or less directly towards a destination. Therefore, the suitability of an agent-environment-action system will depend on the distance of the environmental primitive to the destination in relation to the alternative systems. In this specific context, however, the related pedestrian characteristic is unlikely to be related to a capability, except for differences in cognitive wayfinding skills, which are omitted here, but rather depends on the relative importance which the person attributes to the directness of his or her route. Such differences can for instance be expressed by the contrast between the purposeful movement of a commuter and the rather purposeless wandering of tourists (Ronald et al. 2007). In our framework, however, such variations are not modeled as a capability but by allocating different weight factors to the according realization action. Accordingly, the suitability of each system with regards to *RA_REACHDESTINATION* can be inferred in an agent-independent way, which means directly from the normalized relative distance of its corresponding $environmental\ primitive_j$ to the final or an intermediate destination of the pedestrian, so that the suitability value grows from 0 - 1 one with decreasing distance. This can be achieved with the following formula:

$$suitability_{ij\propto\prime\prime} = 1 - \frac{dist_j - dist_{min}}{dist_{max} - dist_{min}} \tag{11}$$

*RA_KEEPDISTANCETOMOTORTRAFFIC* and *RA_KEEPDISTANCETOCYCLETRAFFIC* denote the need of a pedestrian to keep a safety distance to the traffic or cycle lanes. Here, the suitability can be determined by the distance of an environmental primitive to the nearest road or cycle lane on the one hand, and the total space needed by the pedestrian on the other hand. In a similar fashion as *RA_ACCESSPATCH*, the individual walking space for a pedestrian can be calculated from summing the body width, wavering movement and shy-away-distance to the road or the cycle lane (Buchmueller and Weidmann 2006). Also, the realization action is possible if the minimum space requirements are met, which is why the $\pi_{min}$ value is set to 0.99. The $\pi_O$ value is arbitrarily set to 5, relating to the provided space to the road or cycle lane being 5 times higher than needed. For the suitability calculation, formula 10 is used in both cases.

At night, being able to see and be seen by others is a critical need for pedestrians, as expressed in *RA_STAYNEARLIGHT*. Since 2003, for EU member states, minimum standards for illumination intensities are defined in the standard EN 13201, which distinguishes between different types of urban areas based with according standards based on characteristics such as predominant modes of transport, frequencies and many more (Richter and TRILUX 2005). For the suitability calculation, therefore, the exact light intensity of urban areas would be a desirable environmental disposition, however, is difficult to obtain due to a lack of available data. Based on the fact, however, that the brightness decreases with rising distance from the light source, and that street light density has been identified as significant for walking frequency, this characteristic can be used as a proxy (Forsyth et al. 2008). Pedestrians, in turn, differ in their need for illumination levels, either due to differences in visual perceptive abilities or fear of crime (Basbas et al. 2010, Ovstedal and Ryeng 2002). The preferred maximum distance to a streetlight, therefore, is used for the definition of a capability. In accordance with the used pattern, $\pi_{max}$ is set to 1.01, and the optimal case $\pi_O$ is given if $\pi$ approaches 0. A linear mapping function is deployed with the use of formula 10.

Especially when walking for recreational purposes, disturbances by motorized traffic modes should be minimized (Sanches et al. 2007). *RA_NOTSEETRAFFIC*, therefore, describes the action of a pedestrian trying to keep a certain minimal distance or a blocked view from the traffic lanes. The relative amount of road which is visible from a location can therefore be used as an environmental disposition. However, similar to *RA_REACHDESTINATION*, the definition of a corresponding pedestrian capability is not necessary. For the action of keeping within a distance to, or a blocked view of traffic, the suitability does not depend on any pedestrian property, but rather on how much importance is placed on this aspect. Thus, for a person in a hurry, the aspect of not being disturbed by motorized traffic is less important than for someone going for a recreational walk (Ovstedal and Ryeng 2002). The suitability function, therefore, can be based on an inversion of a normalized value which describes the relative road visibility.

Walking is an activity which puts physical strain on the human body and therefore requires possibilities to rest (Weidmann 1993). Thus, the suitability of a system with regards to *RA_STAYNEARBENCHES* depends on how close a pedestrian is to the next bench as a possibility to recover. Other possibilities for resting, such as cafes or restaurants, are non-public space, and therefore omitted in this context. The environmental disposition is defined by the distance to the next bench. The definition of a pedestrian capability is necessary, since maximum tolerated walking distances between resting places differ based on the stamina of the pedestrian (Basbas et al. 2010, Vukmirovic 2010). Accordingly, the environmental disposition is opposed by the maximum distance a pedestrian is able to walk without taking a rest. Thus, $\pi_{max}$ is set to 1.01, and $\pi_O = 0$. Formula 10 is used as suitability mapping function.

The aesthetical value of the walking environment has been identified to depend to a high degree on the presence of greenery and aesthetical buildings (e.g. Hidalgo et al. 2006, Adkins et al. 2012). Thus, the realization actions *RA_SEEGREENERY* and

*RA_SEEAESTHETICALBUILDING* were introduced to describe a pedestrian keeping a visibility relation with such environmental entities. Again, we argue that no explicit pedestrian capabilities are needed here, since the suitability is rather determined by the corresponding weight coefficient, which differ depending on the trip purpose (Ovstedal and Ryeng 2002). Similar to *RA_REACHDESTINATION*, the suitability can be directly inferred from the environmental attribute, in the first case the number of visible trees and in the second the number of visible aesthetical buildings. Using an adapted version of formula 11, however, omitting the value inversion, the suitability can then be calculated:

$$suitability_{ij\propto''} = \frac{visibleObjects_j - visibleObjects_{min}}{visibleObjects_{max} - visibleObjects_{min}} \tag{12}$$

Finally, the realization actions related to street crossing, *RA_EVALUATESTREETCROSSING*, *RA_ASSESSTRAFFICSITUATION*, and *RA_REACHSIDEWALK*, require suitability calculation procedures. The first realization action merely describes the process of assessing whether a street crossing process would actually reduce the distance to the destination. An environmental disposition is represented by the distance to the destination, with no capability being necessary, while the suitability can be calculated from the difference of distances to the destination measured from the current position of the pedestrian compared to the expected position after crossing the street:

$$suitability_{ij\propto''} = 1 - \frac{dist_{afterCrossing}}{dist_i} \tag{13}$$

For the second sub-action, the detection of a sufficient gap in the traffic flow for crossing, a pedestrian needs a certain amount of time. Studies have shown that this curb delay differs among pedestrians, with elderly persons generally taking longer due to reduced sensory, perceptual and cognitive abilities (Oxley et al. 1995). Further, the critical gap time depends on the walking speed (Fi and Igazvölgyi 2014). Therefore, the capability of a pedestrian for assessing the traffic situation is given by the time needed for the whole crossing process, as a sum of curb delay and the time needed for crossing the road. This corresponds to the time provided for crossing as environmental disposition, which depends on the distance from which an approaching car can be spotted, and the allowed maximum traffic speed. From these properties, the time window can be calculated, which a pedestrian has from seeing the car until it arrives at his or her present location. This time should not be less than the time needed for the crossing process, as expressed in a $\pi_{min}$ value of 0.99. The $\pi_O$ value is arbitrarily set to 2, meaning that the suitability reaches its highest value if the given time window is double the time which a pedestrian would need for the crossing process.

Finally, a pedestrian has to be able to actually move across the road at a certain location, meaning that potential barriers must be overcome. For *RA_REACHSIDEWALK*, therefore, the suitability can be assessed by incorporating all capabilities and dispositions associated with the realization actions allocated to the semantic action *SA_AGENTMOVE*, which have already been discussed above, except for *RA_REACHDESTINATION*. For this, since pedestrians tend to minimize the distance when crossing the street (Kadali and Vedagiri 2013), the direct path to the nearest sidewalk on the other side of the road can be checked for its accessibility, and a combined suitability value for *SA_AGENTMOVE* calculated, with the inclusion of a normalized coefficient which denotes the effect of a longer crossing distance:

$$suit_{ijReachSidewalk} = \frac{\left( \frac{\sum_{j=1}^{n} \frac{suit_{ijAccessPatch} + \cdots + suit_{ijOvercomeSlope}}{4}}{n} \right) + \frac{crossDist_j}{crossDist_{max}}}{2} \quad (14)$$

### 5.1.3 Conclusion: Modeling Walking as a Subjective Experience

In this chapter, in preparation for the development of an agent-based pedestrian simulation, a model of subjective micro-scale walkability has been presented which is based on the framework for suitability assessment described in Chapter 4. 1. 1. Accordingly, walking was modeled as a pragmatic action, with several contributing semantic and realization actions being involved. This was done based on the conceptual framework of walking needs by Alfonzo (2005) and the results from empirical studies on micro-scale walkability, as presented in Chapter 3. 2. 2. In addition, for each sub-action on the realization level of abstraction, the suitability calculation procedure was defined. To the best of our knowledge, our model represents the first approach to incorporate subjective perceptual differences into a model of walkability. Due to data limitations, as well as with regards to feasibility and model simplicity, however, several restrictions had to be made in defining the model. Thus, due to the fact that we strictly oriented on walkability indicators which had been found as significant in the reviewed literature, it is possible that other aspects of walkability are neglected. Further, drastic simplifications had to be made with regards to the definition of the suitability calculation procedures concerning several realization actions, for instance by incorporating the step height as sole relevant aspect of stairs, the operationalization of aesthetically pleasing urban settings or the use of proximity to street lights instead of actually measured values on the light intensity. Still, however, due to the focus of this thesis, the aim of the simulation is not to achieve perfect accuracy, but to demonstrate the practical value of the model of functional place.

## 5.2 Simulating Pedestrian Movement in Augsburg, Germany

In this chapter, an agent-based model of pedestrian movement in Augsburg, Germany, is described as an exemplary application of the PlaceBuilder extension. Based on the subjective walkability model developed in the previous chapter, the aim is to demonstrate the formal correctness of the conceptual model of place and its computational implementation, as well as its practical usefulness for generating pedestrian agents with the ability to form functional places for walking which are plausible and individualized. Modeling *WALKING* as a complex pragmatic action, a further goal is to incorporate walkability as an influential factor into a pedestrian movement simulation. In the following subchapter, the model's components and processes are described in detail before the approach is discussed with regards to central design concepts. Please note that the full code is provided in Appendix B.

### 5.2.1 Overview of the Model Components and Processes

In the context of our simulation, a NetLogo turtle represents a pedestrian who is situated in a patch-based model of the urban environment, and moves between predefined locations such as shops, cafes, touristic attractions and public transport stations. The focus of interest is set on the turtle's navigation process, which is influenced by its preset world knowledge, including a representation of the movement network and the location of itself and its destination, as well as its immediate visual perceptions of its surrounding environment. The functional place for *WALKING* which is chosen by a turtle, meaning its path, depends on the evaluated walkability, which is in turn influenced by the interplay of properties of the turtle and the involved patches as well as the action model for *WALKING*. In the following, with the agent and the environment, the main components of the model are described. Then, the simulation procedures and its general schedule are explained.

#### *5.2.1.1 Modeling the Environment*

Figure 18 shows the extent of the study area in Augsburg, Germany. It is set in the inner city and encompasses part of the pedestrian precinct with the primary shopping area in the south, as well as sidewalks along streets in the north and east. Accordingly, in reference to the terminology proposed by Ronald et al. (2007), it can be classified as a mixture between mixed mode and open space. In total, 152 potential origins and destinations are located within the study area, including shops, restaurants and cafes, bars and touristic attractions, and public transportation stations. All locations were extracted from the Open Street Map[4] (OSM), except for the shops, which were manually mapped in the course of a field audit. In general, similar to Haklay et al. (2001), the study area is treated as a fixed and closed system with a limited number of pedestrian entry points, which are represented by the described locations.

[4] http://www.openstreetmap.de/

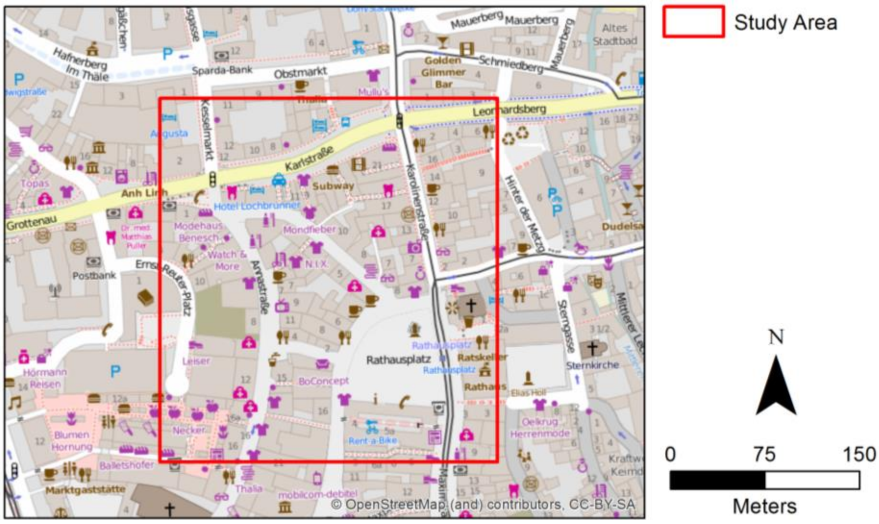


Fig. 18: Study Area

In accordance with Russel and Norvig (2003), the environmental model can be classified according to a number of criteria. First, it is based on a discrete representation of space in the form of a regular tessellation with NetLogo patches. In reference to the conceptual model, each individual patch, being the smallest, fundamental sampling unit, represents a geo-atom in the context of this simulation. The spatial resolution is determined by the patch size, which should be smaller than a person's physical footprint (Torrens 2012). In accordance with Chen et al. (2009) and Chu (2011), cells with 40 centimeter edge length are used. Further, the environment is partially observable, which means that an agent is only aware of the patches within its current viewfield. At the present stage, the environment can be classified as single-agent, although the general concept would allow for the incorporation of more, simultaneously acting agents. It is also static in the sense that its properties are not subject to change during a simulation run, and deterministic since the result of an agent's action is not of stochastic nature. With regards to the agent-environment relationships, it is unidirectional, since the environment influences the agent's actions, but the agent cannot alter the environment (Stanilov 2012).

Concerning the level of semantic detail, the environmental model used in this simulation is semantically rich, with numerous spatial characteristics being encoded as patch attributes. For the full list, please see table 3 in the previous chapter. Apart from environmental attributes such as slope or surface structure, more abstract information is also pre-calculated and embedded within the environmental model, including visibility and distance relations with other patch types (e.g. street) or objects (e.g. trees), which themselves are not explicitly represented within the NetLogo model. Although these properties could theoretically be computed by the agents during run-time, their pre-calculation reduces the computational load (Haklay et al. 2001).

The pre-processing of the spatial data for the environmental model required several steps, which were conducted using ESRI's ArcGIS (ESRI 2012). First, it was necessary to determine the geometry of the walkable area. This involved several steps: First, using ESRI's Erase tool, all building footprints, as derived from OSM, were cut from a polygon of the study area. Then, aided by GPS-enabled devices, a field audit was conducted in May 2014, during which smaller permanent obstacles such as lanterns or fixed trash cans were identified and deleted from the walkable polygon. Furthermore, the polygon was segmented based on the classification into sidewalk, street, cycle lane, and traffic-light controlled crossing, as well as the surface structure, which was assessed based on the classification scheme shown in table 4. Moreover, locations with drastic height differences such as curb stones and stair steps were digitized as line features and had their height measured.

With regards to the data import into NetLogo, the aim was to produce one separate raster file in the ASCII Grid data format for each environmental property. In order to avoid inconsistencies among raster layers, for instance caused by slight differences in cell size, variations in areal extent or unaligned cell boundaries, a reference point dataset was computed, in which each point represents the centroid of a raster cell with 40 centimeters cell size. All further geo-processing steps needed for the calculation of the environmental attributes could thus be based on the same point dataset, which was attributed the resulting values, and then converted to an ASCII Grid file. In general, the environmental attributes contributing to an environmental disposition with regards to a realization action involved in the achievement of *sA_AgentMove* or *sA_AgentUpdateSafety* were calculated for all point features, regardless of their classification into sidewalk, street, cycle lane or crossing. This was based on the thought that, for instance when stepping on the road, an agent must be able to physically move over these areas and be as safe as possible. In such situations, environmental attributes related to realization actions contributing to *sA_AgentUpdateComfort* or *sA_AgentUpdatePleasure*, in contrast, are less important, and were thus only computed for sidewalks and traffic light controlled crossings.

A number of environmental attributes could be calculated with the use of standard geo-processing tools included in ArcGIS. The slope, for instance, was calculated based on a digital elevation model with 1 meter resolution (DGM 1), using the tool Slope which is included in ESRI's Spatial Analyst extension (ESRI 2012). The resulting values were then attributed to the point dataset for sidewalks, streets, cycle lanes and street crossings, and converted into raster. Also, the walking distance to the closest bench could be computed using standard tools. The benches, which were extracted from OSM, served as input for the Path Distance tool, which calculates for each cell the accumulative cost or distance to the nearest destination, but, not based on Euclidean distance but using a network-like cell representation with links and nodes, in our case based on a cost raster which included the sidewalk area and the cellsize of 0.40 of each cell as cost value (ESRI 2012). To receive more realistic results, benches in a buffer zone of 100 meters around the boundaries of the study area were also included in the analysis. A further attribute, the distance to the nearest street light, was based on Euclidean distance, and thus could be computed with the tool Near from ESRI, which calculates the distance between input features and their closest feature in another input dataset (ESRI 2012). Accordingly, for each point feature with the classification sidewalk, the distance to the closest street light, which could be obtained as a shapefile from the Department for Civil Engineering (Tiefbauamt) Augsburg, was calculated and added as an attribute. The distance to the nearest obstacle, such as a building wall, was calculated using the tool Euclidean Distance, which computes for each cell the distance to the nearest source (ESRI 2012). Finally, for all points which represent a potential start location for a jaywalking process, meaning being themselves of type sidewalk but located next to a street point, the maximum allowed traffic speed of the neighboring street was identified and allocated as an attribute. In all of the above cases, the resulting values were attributed to the point dataset and converted to a separate ASCII GRID file.

The remaining environmental attributes were calculated using the programming language Python and the arcpy site-package (ESRI 2012). Since it is not within the immediate focus of this thesis, the procedures will not be discussed in full detail. For the complete Python code, however, please see Appendix C. The calculation of the number of historical buildings visible from each point involved several steps, and was done using the function *calculateVisibilityFrequency*. Due to a lack of actual data on the building heights, the DEM used for the visibility calculations was computed based on the DGM 1 by simply extruding the building footprints with a fixed height of 20 meters. Since buildings block the view of a pedestrian in eye height, more detailed data is not necessary for these purposes. For the computation of the number of historical buildings visible from each point, a standard height of 1.7 meters of a pedestrian was assumed, and, due to the lack of detailed 3D building data, the visibility of a building was restricted to eye height of the pedestrian. Although this procedure reduces the data needs for the analysis, it is clear that our model deviates from reality at this step, which may cause inaccuracies. The environmental attribute number of visible trees was calculated in a similar fashion.

The distance to nearest unblocked street or cycle way was calculated using the function *calculateVisibilityDistance*. Here, for each feature of the input point dataset, the distance was calculated to the nearest point of classification street, with which a visibility relation exists. Thus, the positive blocking effect of buildings or other obstacles on perceived traffic disturbances could be incorporated in the analysis. A cut-off value of 25 meters was defined, above which, the distance from the street or cycle lane was assumed to be sufficient to marginalize the negative effects.

The relative amount of visible road area for each point could be identified with the Python function *calculateTrafficVisibility*. Here, the number of visible street points was identified for each sidewalk or crossing point and normalized to a range from 0 – 1.

Finally, the attributes related to *SA_AGENTCROSSROAD* needed to be calculated, in particular the longest distance from which approaching cars can be spotted, and the minimum crossing distance. In the first case, this occurred by using the Python function *calculateSightDistance*, which identifies from the sidewalk points the potential starting locations for a crossing action, computes their viewshed, overlays it with a polygon feature of the street to be crossed to identify their overlapping area, and calculates the length of the longest possible sightline within this intersecting polygon. This value is then stored as an attribute of the respective point, and later converted to a raster cell value. In the direct vicinity of street intersections, no potential crossing locations were defined, which is due to the fact that they are unlikely places for pedestrian crossing, but also because the described method is only applicable to computing the longest sightline if there is only one relevant street polygon. For the second attribute, the corresponding crossing destination for a starting point needs to be known. Thus, in the function *calculateCrossingDistance*, for each starting point, the nearest sidewalk point on the opposite side of the street is identified, the distance calculated and stored and, since the cell index of the corresponding patch is needed for simulating agent movement in the NetLogo simulation, the row and column numbers of the cell calculated and stored as attributes. Again, for each attribute, ASCII Grid datasets are exported.

Listing 9 shows the list of patch attributes defined in NetLogo, which mostly correspond to the subjective model of walkability defined in the previous chapter.

```
patches-own
[
  classification ;classification into sidewalk, street, crossing, and cycleway
  slope ;gradient in percent
  surface ;surface structure classified into 5 types
  benchDist ;path distance to closest bench in meters
  lightDist ;euclidean distance to closest streetlight in meters
  buildVis ;number of historical protected buildings visible
  treeVis ;number of visible trees
  crossSightDist ;longest sightline from which approaching cars are visible in meters
  minDistStreet ;euclidean distance to closest, unblocked street patch in meters
  minDistCycle ;euclidean distance to closest, unblocked cycleway patch in meters
  trafficSpeed ;allowed maximum speed in km/h
  crossingDistance ;minimum crossing distance in meters
  crossDestRow ;row of nearest crossing destination
  crossDestCol ;column of nearest crossing destination
  patchHeight ;curb height in centimeters
  trafficVis ;amount of street visible (standardized values from 0-1)
  buildDist ;distance to nearest unwalkable cell in meters
]
```

Listing 9: Patch Attributes (NetLogo)

During the model initialization, the created ASCII Grid files are imported using the NetLogo GIS extension, and then copied to the corresponding patch attributes during the setup procedure. Figure 19 shows the study area as modeled within NetLogo.

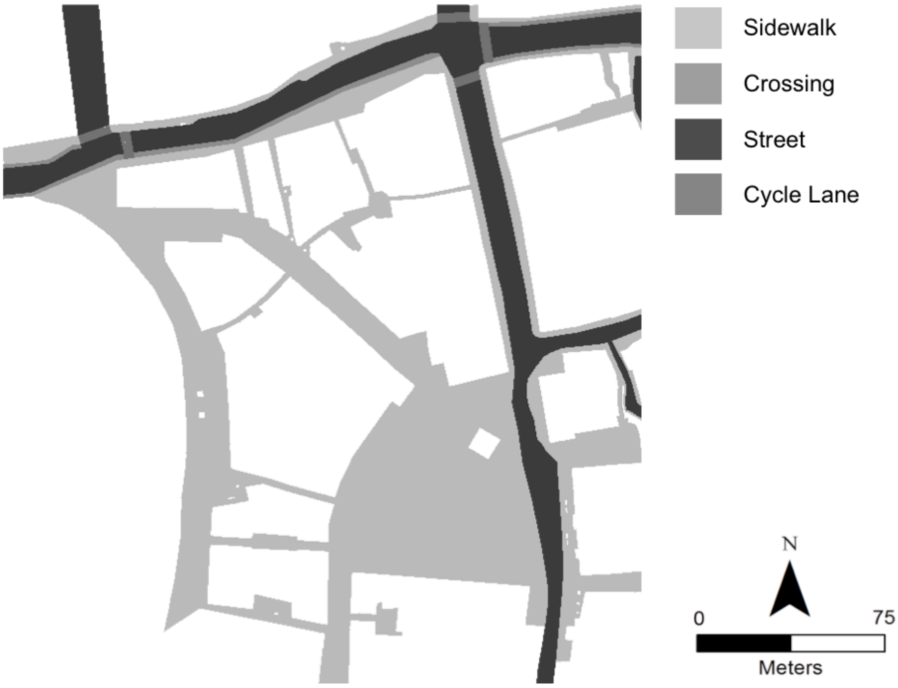


Fig. 19: NetLogo Model of the Walking Environment

#### *5.2.1.2 Modeling the Agent*

The pedestrian agent is realized as a NetLogo breed, a subclass of Turtle with its own behavior and attributes (Wilensky 1999). Listing 10 shows the pedestrian attributes, which mostly correspond to the subjective model of walkability as described earlier, but also include attributes for its beliefs, desire and intentions, as well as an origin-destination-pair. Further, as a pedestrian places individual relative importance on the different sub-actions contributing to *PA_WALK*, additional attributes are introduced to store the weight coefficients for the semantic and the realization actions.

In this simulation, the aim is to equip the agent with the ability to navigate its walking environment based on the individually perceived walkability. In terms of the behavioral levels proposed by Hoogendorn and Bovy (2004), thus, our focus is on the route choice behavior on the tactical level and the microscopic walking behavior which takes place on the operational level. Accordingly, it can be differentiated between wayfinding, for which a representation of spatial knowledge is required, and locomotion, which must be based on some sort of sensory input of the environment (Montello 2005). For the representation of spatial knowledge, a variety of potential methods have been discussed in chapter 3. 2. 1. For this simulation, a visibility graph is created, in which corners of obstacles form the nodes of a network, and are connected with edges based on their mutual visibility (de Berg et al. 1997, Johansson and Kretz 2012). This approach is preferred to a foot path network representation due to the fact that, especially within the pedestrian precinct in our study area, walking zones are rather of areal than network-bound shape. In general, however, our approach can be described as hybrid, since, similar to Haklay et al. (2001) or Gloor et al. (2004), a graph and field-based approach is combined. Thus, the spatial knowledge used for wayfinding is represented as a visibility graph, while the immediate perceptions needed for locomotion are based on the patch variables.

```
pedestrians-own
[
  beliefs ;represents the knowledge-base of the agent
  intentions ;defines the intended agent behavior
  desire ;defines the agent desire
  OD-Pair ;its origin-destination pair

  minSurface ;minimum acceptable surface quality, refers to code used in surfaceRaster
  maxSlope ;maximum slope, critical threshold value
  width ;body width in meters
  waveringMovement ;additional width due to dynamic body movements in meters
  shyAwayDistTraffic ;preferred minimum buffer distance to streets in meters
  shyAwayDistCyc ;preferred minimum buffer distance to cycleways in meters
  shyAwayDistWall ;preferred minimum buffer distance to walls in meters
  maxLight ;maximum euclidean distance to street light in meters
  maxBench ;maximum path distance to bench in meters
  minTimeKerbDelay ;time spend for gap detection in seconds
  walkingSpeed ;walking speed in km/h
  maxHeight ;refers to the height the pedestrian is able to surmount in meters (e.g. a curb)

  weight_sA_AgentMove ;weight for sA_AgentMove
  weight_sA_AgentUpdateSafety ;weight for sA_AgentUpdateSafety
  weight_sA_AgentUpdatePleasure ;weight for sA_AgentUpdatePleasure
  weight_sA_AgentUpdateComfort ;weight for sA_AgentUpdateComfort
  weight_sA_AgentCrossStreet ;weight for sA_AgentCrossStreet
  rA_KeepBalance_weight_sA_AgentMove ;weight for rA_KeepBalance (sA_AgentMove)
  rA_OvercomeSlope_weight_sA_AgentMove ;weight for rA_OvercomeSlope (sA_AgentMove)
  rA_OvercomeHeightDifference_weight_sA_AgentMove ;weight for rA_OvercomeHeightDifference (sA_AgentMove)
  rA_AccessPatch_weight_sA_AgentMove ;weight for rA_AccessPatch (sA_AgentMove)
  rA_ReachDestination_weight_sA_AgentMove ;weight for rA_ReachDestination (sA_AgentMove)
  rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety ;weight for rA_KeepDistToMotorTraffic (sA_AgentUpdateSafety)
  rA_StayNearLight_weight_sA_AgentUpdateSafety ;weight for rA_StayNearLight in the context of sA_AgentUpdateSafety
  rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety ;weight for rA_KeepDistToCycleTraffic (sA_AgentUpdateSafety)
  rA_SeeGreenery_weight_sA_AgentUpdatePleasure ;weight for rA_SeeGreenery (sA_AgentUpdatePleasure)
  rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure ;weight for rA_SeeAestheticBuild (sA_AgentUpdatePleasure)
  rA_NotSeeTraffic_weight_sA_AgentUpdateComfort ;weight for rA_NotSeeTraffic (sA_AgentUpdateComfort)
  rA_StayNearBenches_weight_sA_AgentUpdateComfort ;weight for rA_StayNearBenches (sA_AgentUpdateComfort)
]
```

Listing 10: Pedestrian Agent Attributes in NetLogo

The visibility graph, which is shown in figure 20, was pre-computed in ArcGIS (ESRI 2012). In order to reduce the elements of a visibility graph without decreasing its usefulness for navigation purposes, it is possible to compute convex hulls for the obstacles, in our case the building footprints (De Berg et al. 1997). Thus, the ESRI tool Minimum Bounding Geometry with the parameter setting *Convex_Hull* was used to create the according shapes. Placing the nodes directly on their vertices, however, would lead to problems in the simulation since, due to their need for space for walking, a pedestrian agent would not be able to actually reach a node. Thus, buffers were computed for the convex hulls with 0.5 meters distance, and, in order to reduce the number of vertices, simplified using ESRI's Simplify Polygon tool with the parameter setting *Point Remove* and a maximum allowed offset of 0.5 meters. Then, the vertices were converted to points. Before computing the mutual sightlines, however, the 152 locations which provide potential origins and destinations were added to the point dataset. Then, all-pair sightlines could be created with the tool Construct Sight Lines of ESRI's 3D-Analyst extension (ESRI 2012). By intersecting the resulting sightlines with all obstacle footprints, the non-visible features could be identified and deleted from the dataset. Further, all sightlines which intersected roads with a maximum allowed traffic speed above 50 km/h were also deleted from the dataset, except at traffic-light controlled crossing locations. This was done to ensure that these safe crossing locations are preferred by pedestrian agents, which, however, still have the option to cross the road anywhere. With regards to streets with a maximum speed below 50 km/h is was assumed that most pedestrians would not divert from the direct path to use a traffic light controlled crossing, which in many cases does not exist at all. Thus, for such streets, the sightlines were retained. In order to further reduce the resulting network, an all-pair shortest path analysis was conducted for the potential origins and destinations using ESRI's Network Analyst extension (ESRI 2012). All edges which were not used by any of the routes were deleted from the visibility graph.

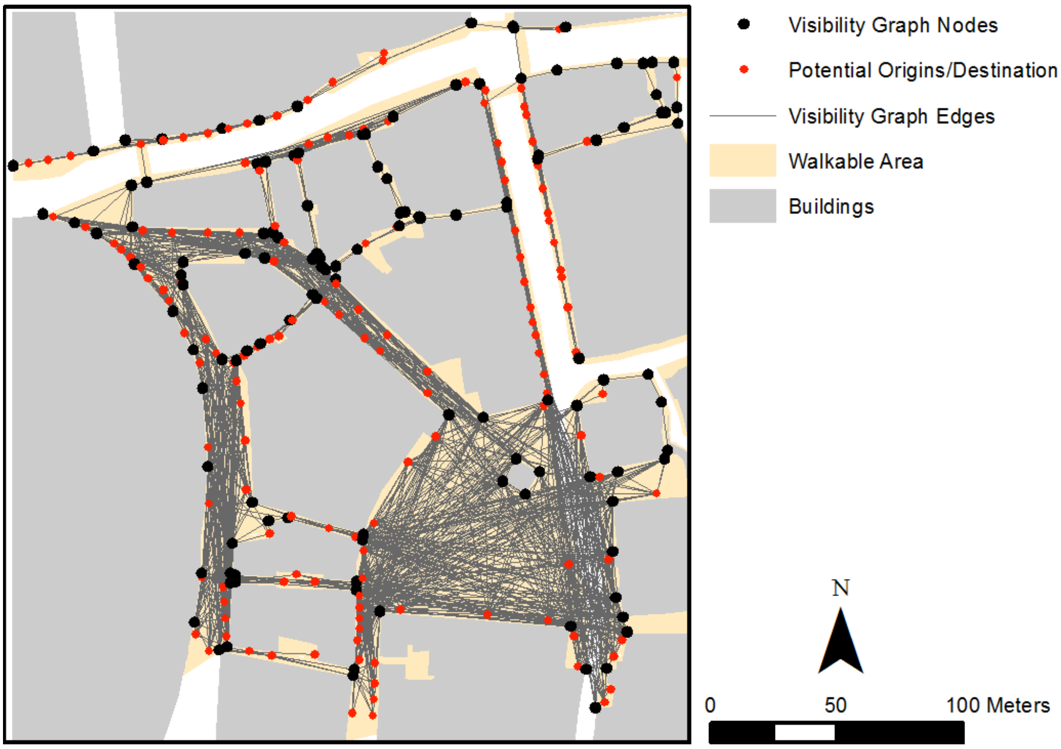


Fig. 20: Visibility Graph

### 5.2.2 Procedures and Scheduling

Figure 21 shows the general scheduling of procedures during a simulation run. The simulation is initialized by importing the geo-spatial data, in particular the pre-processed ASCII Grid files related to the patch attributes and the visibility graph shapefiles which represent the pedestrian agent's spatial knowledge. For this, the NetLogo GIS extension is used, more accurately its primitive *gis:load-dataset*. In the following setup procedure, applying the *gis:apply-raster* primitive, the cell values, as received from the raster files, are mapped to the respective attributes of the corresponding patches. Further, the subset of visibility graph nodes which represent potential origins or destinations for a walking trip are identified and their primary keys stored in a list. Due to the fact that, as discussed in Chapter 5. 1. 2, some realization actions such as *RA_NOTSEETRAFFIC* require no explicit pedestrian capabilities for the suitability calculation, but rather use normalized values of environmental characteristics, the maximum and minimum values for these selected patch attributes are calculated during the setup procedure. After the setup has been completed, a pedestrian agent is created at the location of its predefined origin node and its attributes are set according to the modeler's inputs. Then, the initial beliefs of the agent are set, before the model is finally ready to run. Each of these procedures is discussed in the following chapters.

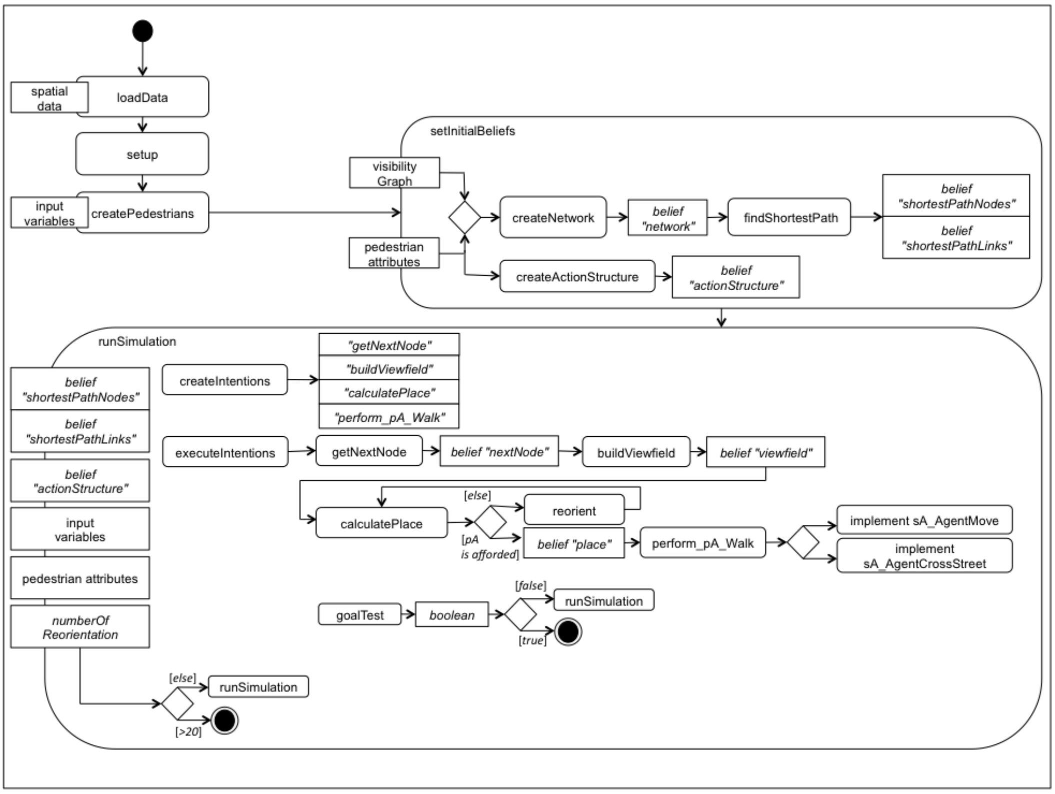


Fig. 21: Model Overview and Scheduling

#### 5.2.2.1 Set Initial Beliefs

The procedure *setInitialBeliefs* serves to create the knowledge base necessary for the pedestrian agent prior to its walk. In particular, this involves the level of spatial knowledge necessary for wayfinding, and the action model for the pragmatic action *PA_WALK*.

##### 5.2.2.1.1 Create Network Knowledge

As a first step, an internal representation of the visibility graph network needs to be constructed for the agent based on the imported shapefiles. Thus, *setInitialBeliefs* calls the procedure *createNetwork*, which computes a NetLogo network representation, using a special turtle breed as nodes which are connected with links. The links are attributed the real length as obtained from the link shapefile. Since the resulting network is stored as an agent belief, it is generally possible to represent varying levels of spatial knowledge of different agents by using a reduced network. This, however, has not been yet implemented at the present stage of the simulation.

In the following, the "*network*" belief provides the input for a *findShortestPath* procedure, which calculates the shortest path between the agent's origin and destination node in the visibility graph network, based on the actual edge length. For this, the NetLogo network extension is used, in particular the primitive *nw:weighted-path-to*. The nodes and the links which constitute the received shortest path are stored in separate beliefs "*shortestPathNodes*" and "*shortestPathLinks*". It should be noted at this point that this routing procedure does not determine the agent's wayfinding process on the tactical level, but rather, the ordered sequence of nodes and links provide a general frame of orientation for the pedestrian agent which, however, chooses its actual path based on its perceptions and the perceived walkability. This, however, is discussed in more detail in Chapter 5. 2. 2. 2. 2.

##### *5.2.2.1.2 Create the Action Model for* PA_WALK

Apart from spatial knowledge, the pedestrian agent needs an action model of the pragmatic action in question, in our case *PA_WALK*. Thus, the procedure *createActionStructure* is called from *setInitialBeliefs*. In this method, the action structure is created based on the general framework explained in Chapter 4. 2. 4 and in accordance with the subjective model of walkability discussed in chapter 5. 1. Figure 22 shows the final action model.

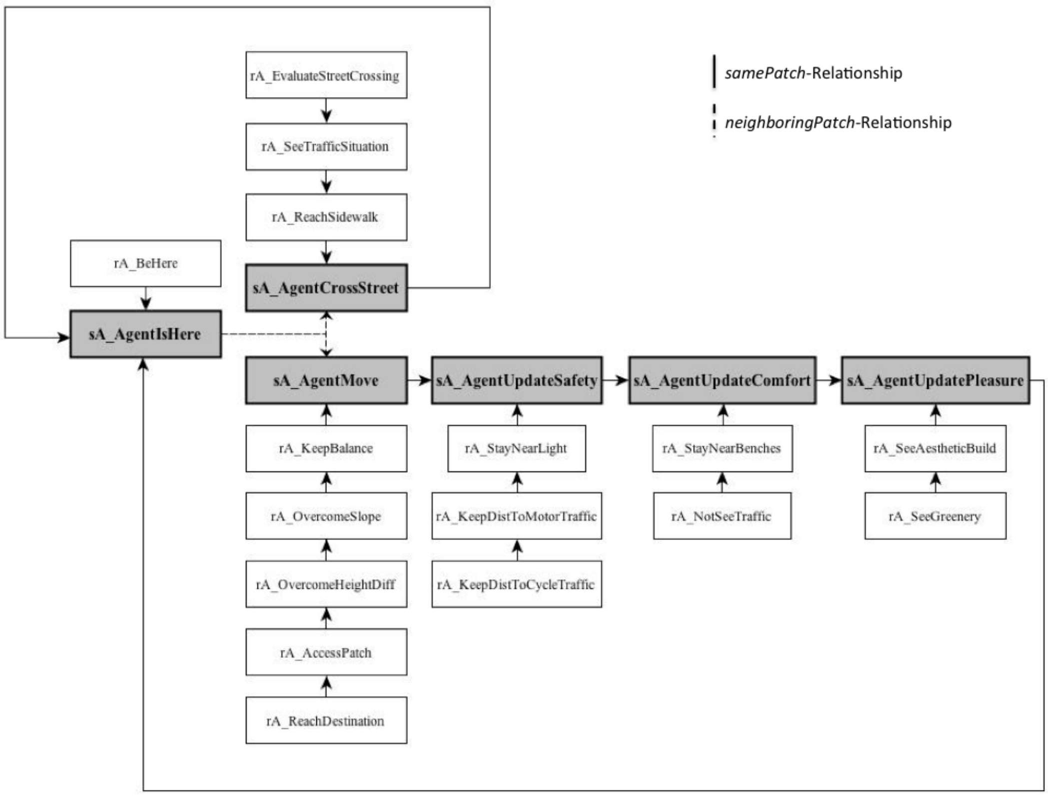


Fig. 22: Action Model for *PA_WALK*

In comparison to the subjective model of walkability, an additional semantic action *SA_AGENTISHERE* and its *RealizationAction* *RA_BEHERE* have been introduced. For the simulation process, these are necessary to serve as initialization actions, which inform the agent about its own current location to avoid starting the place calculation process anywhere within the viewfield. Accordingly, the suitability for *RA_BEHERE* has only two possible values, 1 for the patch at the agent's present location and NoData at all other patches.

Another aspect of importance are the spatial dependencies between the actions. Thus, the realization actions contributing to *SA_AGENTMOVE* and *SA_AGENTCROSSSTREET* can only be afforded by *SystemAgentEnvironment* instances which involve patches in a topological Moore neighborhood in relation to the patch involved in *RA_BEHERE*. In other words, a pedestrian can only move one patch at a time. The realization actions which contribute to the other semantic actions, naturally, must occur in the same patch as the one the pedestrian has moved to.

So far, the pragmatic action consists of only one step or one street crossing process. This, however, is neither feasible nor realistic. Other than shown in figure 22, therefore, the pragmatic action is modeled to involve 15 repeating steps, which means that for each semantic action, 15 separate instances are created. The number of repeating steps refer to the maximum distance until which a pedestrian is able to perceive its immediate environment, which is set to 15 patches or 6 meters in our model, and roughly corresponds to the findings of Kitazawa and Fujiyama (2010). It is, however, thinkable that a pedestrian agent is not able to walk the full 15 steps, for instance due to being too close to an obstacle. To cope with this situation, an additional alternative pragmatic action *PA_WALKONESTEP* is introduced, which actually consists of only one step, and involves no crossing possibility. In order to test for the influence of the *viewDistance* parameter on the simulation result, we also create the possibility to instantiate a version of pragmatic action *PA_WALK* with only 5 steps and a crossing possibility. The weight coefficients attributed to the semantic and realization actions are received from the pedestrian agent properties. For further details, please see Appendix B.

#### *5.2.2.2 Run Simulation*

When the initial beliefs have been created, the model is ready to run. The main *runSimulation* procedure controls the agent's behavior via its intentions, which, using the method *createIntentions*, are created and stored on a stack, for which we reuse code by Sakellariou et al. (2008).

The first step of the agent's behavior consists of identifying from the "*shortestPathNodes*" belief the one node which it should approach next. This "*nextNode*" belief is computed using the *getNextNode* procedure, and, in combination with the shortest path calculation in *setInitialBeliefs*, represents the wayfinding behavior of the pedestrian agent. After the next immediate destination has been identified, the agent orients towards the respective node and, triggered by the intention "*buildViewfield*", partially perceives its environment, with the extent of the viewfield being determined by the input parameters *viewfieldWidth*, which defines its maximum angle, and *viewDistance*, the length of the viewfield in meters. The perceived patches are stored as a belief "*viewfield*", and provide the input for the next intention "*calculatePlace*", which calls the respective method *calculatePlace*. In this procedure, as explained in chapter 4. 2, instances of *SystemAgentEnvironment* are computed for each pair of pedestrian and visible patch, the suitability values are calculated for the realization actions by calling their corresponding procedures, and, finally, the place is generated and stored as a belief of type "*place*". This belief is then used to perform the pragmatic action, which in our case study results in movement of the pedestrian agent, and represents its locomotion level of navigation. How both aspects of navigation, wayfinding and locomotion, are implemented in our model is discussed in detail in the following.

##### *5.2.2.2.1 Wayfinding*

In Chapter 5. 2. 2. 1. 1, it has been described that the shortest path through the visibility graph network is pre-calculated and stored within the knowledge base of a pedestrian agent. It was, however, also noted that this does not represent the entire wayfinding process, but only provides an orientation framework for the agent. In our simulation model, an agent is free to take short cuts, for instance by crossing the road outside of traffic light-controlled crossing locations, or simply by taking a more direct path to the destination. This process is controlled in the *getNextNode* procedure.

The procedure commences by checking whether a belief "*nextNode*" exists. If positive, the distance between the pedestrian agent's present location and the current *nextNode* is computed in order to assess whether the current *nextNode* has been reached. In our case, this reports true if the agent is within 2 meters distance of the node. In case of the current *nextNode* being reached, its incoming link and all links which occur at an earlier stage of the sequence which forms the shortest path are deleted from the belief "*shortestPathLinks*". This step is necessary to avoid the agent moving back and away from the destination node at any time of the simulation.

In general, the identification of the *nextNode* can occur either based on the sequence of visited nodes and links in the shortest path, or on the basis of the visibility of these nodes. Thus, if a pedestrian agent is able to see its destination node or a node which is closer to the destination node than the one which is next in order according to the sequence of the "*shortestPathNodes*" belief, it is free to ignore the sequence and directly approach the respective node, meaning to take a more direct path to the destination. This requires a distinction between the immediate environment (viewfield) and further areas as two separate awareness zones (Torrens 2012). Thus, in the *findShortestPath* procedure, visibility links are created between the pedestrian agent and each node of the shortest path. Using another procedure, *sightlinePatchRelation*, it can be assessed whether these links intersect buildings, thus are not inter-visible.

In the *getNextNode* procedure, accordingly, the sightlines of the pedestrian agent and the *shortestPathNodes* are checked for inter-visibility. Then, from the set of visible nodes, the one is chosen which is closest to the destination node, and therefore likely to provide a direct path to the destination. Before being set as new *nextNode*, however, it must be checked whether a street has to be crossed when approaching this particular node. If not, the node is set as the new *nextNode*. If a street has to be crossed and is within view distance, its crossing affordance is checked with regards to the agent by calculating the respective suitability values for *RA_EVALUATESTREETCROSSING*, *RA_ASSESSTRAFFICSITUATION*, and *RA_REACHSIDEWALK*. If the road can be crossed by the agent, the respective node is set as *nextNode*, and all earlier links are deleted from "*shortestPathLinks*". If this is not the case then the visibility relation to this particular node is deleted in order to avoid a pedestrian agent testing the crossing affordance infinitively, and the next regular node according to the sequence in the shortest path is chosen as new *nextNode*. In all cases, the *nextNode* is stored as belief "*nextNode*".

#### *5.2.2.2.2 Locomotion*

In the previous chapter, it was explained how a pedestrian agent can identify its next intermediate destination, the *nextNode*. This, however, does not determine the agent's path, but merely controls its orientation and the suitability calculation procedures for *RA_REACHDESTINATION* and *RA_EVALUATESTREETCROSSING*. Thus, *SystemAgentEnvironment* instances involving patches which, in case of the first action, are closer to the *nextNode* or, with regards to the second action, from which a crossing process would reduce the distance to the *nextNode*, would receive higher suitability values for these two realization actions. The remaining realization actions and their suitability values, however, are not influenced by the *nextNode*. Thus, the microscopic walking behavior, or locomotion, is mostly based on the perceived walkability of the walking environment. Thus, a functional place for walking is generated, which can also be described as a set of patches within the viewfield which provide a path that optimizes the suitability with regards to the sub-actions involved in *PA_WALK*. Since, however, the exact position of the agent determines its visibility relations to the shortest path nodes, the two processes have mutual influences.

After the *nextNode* has been identified, the procedure *buildViewfield* is called. The pedestrian agent is oriented towards the *nextNode*, and a cone of angle and length as defined by the parameters *viewfieldWidth* and *viewDistance* is created. All patches within the cone and with the classification sidewalk or traffic-light controlled crossing form the viewfield, and are stored as a belief. In the procedure *calculatePlace*, these are used to instantiate *SystemAgentEnvironment* instances, and compute the suitability values. The suitability calculation procedures correspond to the subjective model of walkability described in Chapter 5. 1. As explained earlier, the place is generated and stored as a belief of type "*place*". The intention is run until its done-condition, which refers to the existence of a belief "*place*", is met. If a pragmatic action is not afforded with the available *SystemAgentEnvironment* instances, a *reorient* procedure is called which creates a new viewfield of 360°, thus simulates an agent looking around for a potential path, and repeats the place calculation procedure. In this new viewfield, street and cycle lane patches are included as well to acknowledge that real pedestrians are also able to step on the street to walk around an obstacle. If the pragmatic action is still not afforded, the alternative pragmatic action is tried. If it is afforded, the state changes which correspond to the pragmatic action must be realized in the simulation. Since the internal affective states of safety, pleasure and comfort are neglected at the present stage of the simulation, either *SA_AGENTMOVE* or *SA_AGENTCROSSSTREET* are implemented by calling their implementation procedures. In both cases, the location of the agent is changed, in the first case to the respective patch, and in the second case, to the pre-defined crossing destination patch on the other side of the street. The procedure *runSimulation* is repeated until either the agent's desire state is met, in our case defined by a specific *goalTest* procedure reporting true, which checks whether the final destination node has been reached, or, in order to avoid an agent getting stuck in an infinite loop, until a certain threshold value for the number of times the reorient procedure is called is reached.

### 5.2.3 Model Testing

An essential part of the model development process is its verification in the sense of testing its internal functioning for consistency and correspondence to the original intentions of the developers, as described in Chapter 3. 1. 4 (Batty et al. 2012). Although this is normally followed by a validation against real-world data, in the context of this thesis, this step is omitted due to a lack of available data on observed pedestrian movements. At the present stage of this work, however, the case study serves primarily the need to test the fundamental assumptions of the computational model of functional place, rather than to provide an accurate prediction on actual pedestrian movement. Thus, it can be argued that for our purposes, a comprehensive verification is sufficient.

In the following, two approaches are applied to testing the simulation model: first, its face validity is assessed, and its general functionality demonstrated by the definition of two distinct test scenarios and the observation of the resulting agent behavior. In the following, a more comprehensive evaluation of the model behavior is conducted by means of a local sensitivity analysis. Here, the model's parameter settings are systematically varied and tested for their effect on the model outcomes.

#### 5.2.3.1 Test Cases

As a first step towards analyzing the model's functionality, two distinct test scenarios are established, run and visually assessed with regards to their results. The first case focuses on the street crossing behavior of pedestrian agents with differing capabilities, in particular on jaywalking versus crossing at a traffic light-controlled crossing facility. In the second scenario, the model's capabilities for producing different types of walking are evaluated by simulating the walking behavior of a destination-bound, scurrying pedestrian in comparison to a more exploring, experience-oriented walker.

##### *5.2.3.1.1 Scenario 1 - Jaywalking*

In the first scenario, the agents start from node 154, which is located on the western edge of the city's main square, cross the square and walk alongside a street to node 209, which is, however, set on the other side of the street. This test run is simulated for two different pedestrians, agent 1 and 2, the attribute settings of which are listed in table 5. Both agents are similar with regards to their parameter settings, except for one detail, namely the value attributed to *maxHeight*. Thus, whereas agent 1 is able to overcome barriers of a height up to 30 centimeters, agent 2 is in some way mobility-impaired, so that it is only able to overcome 3 centimeters, a threshold which represents a common guideline value for barrier-free planning and design (FGSV 2002).

The results are illustrated in the upper half of figure 23. As one can see, agent 1, as soon as it comes in sight of the destination node 209, picks it as the next target and takes a short cut by crossing the road outside of a traffic-light controlled crossing. Agent 2 attempts a similar strategy, but is not able to overcome the height barrier posed by the curbstone. In this case, the wayfinding heuristic lets the agent approach the next node in the shortest path sequence. As a consequence, the agent is forced to walk all the way up north to the official traffic-light controlled crossing, which is equipped with a curb cut and therefore a lower height barrier. This is lower than 3 centimeters, and therefore affords road crossing for the mobility-impaired agent, which then heads back in southern direction to finally reach the destination node 209.

| Parameter | Agent 1 | Agent 2 | Agent 3 | Agent 4 |
|---|---|---|---|---|
| maxSlope | 15 | 15 | 15 | 15 |
| maxBench | 1750 | 1750 | 1750 | 1750 |
| maxLight | 67.5 | 67.5 | 67.5 | 67.5 |
| width | 0.5 | 0.5 | 0.5 | 0.5 |
| waveringMovement | 0.15 | 0.15 | 0.15 | 0.15 |
| shyAwayDistTraffic | 0.55 | 0.55 | 0.55 | 0.55 |
| shyAwayDistCyc | 0.3 | 0.3 | 0.3 | 0.3 |
| maxHeight | 0.3 | 0.03 | 0.3 | 0.3 |
| shyAwayDistWall | 0.15 | 0.15 | 0.15 | 0.15 |
| minSurface | 1 | 1 | 1 | 1 |
| minTimeKerbDelay | 2.5 | 2.5 | 2.5 | 2.5 |
| walkingSpeed | 4.2 | 4.2 | 4.2 | 4.2 |
| | | | | |
| weight_sA_AgentMove | 1 | 1 | 1 | 0.1 |
| weight_sA_AgentUpdateSafety | 0.75 | 0.75 | 0.5 | 0 |
| weight_sA_AgentUpdateComfort | 0.5 | 0.5 | 0.25 | 0.5 |
| weight_sA_AgentUpdatePleasure | 0.5 | 0.5 | 0 | 1 |
| weight_sA_AgentCrossStreet | 0.5 | 0.5 | 0.25 | 0.5 |
| | | | | |
| rA_KeepBalance_weight_sA_AgentMove | 0.5 | 0.5 | 0.5 | 0.25 |
| rA_OvercomeSlope_weight_sA_AgentMove | 0.5 | 0.5 | 0.75 | 0.25 |
| rA_OvercomeHeightDifference_weight_sA_AgentMove | 0.5 | 0.5 | 0.5 | 0.25 |
| rA_AccessPatch_weight_sA_AgentMove | 1 | 1 | 0.5 | 0.25 |
| rA_ReachDestination_weight_sA_AgentMove | 1 | 1 | 1 | 0 |
| rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety | 1 | 1 | 0.75 | 1 |
| rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety | 0.5 | 0.5 | 0.5 | 0.5 |
| rA_StayNearLight_weight_sA_AgentUpdateSafety | 0.25 | 0.25 | 0.25 | 0.25 |
| rA_SeeGreenery_weight_sA_AgentUpdatePleasure | 0.5 | 0.5 | 0 | 0.75 |
| rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure | 0.5 | 0.5 | 0 | 1 |
| rA_NotSeeTraffic_weight_sA_AgentUpdateComfort | 0.5 | 0.5 | 0.5 | 0.5 |
| rA_StayNearBenches_weight_sA_AgentUpdateComfort | 0.5 | 0.5 | 0 | 0.5 |
| | | | | |
| viewfieldWidth | 115 | 115 | 115 | 115 |
| viewDistance | 15 | 15 | 15 | 15 |

Table 5: Agent Parameter Settings

#### *5.2.3.1.2 Scenario 2 – Diverting from the Direct Path*

In the second scenario, our intention is to demonstrate the result of different walking preferences, which are expressed in our model as weight coefficients assigned to the respective sub-action. In this test run, the two agents 3 and 4 take a short walk between two locations which leads them across the city's main square.

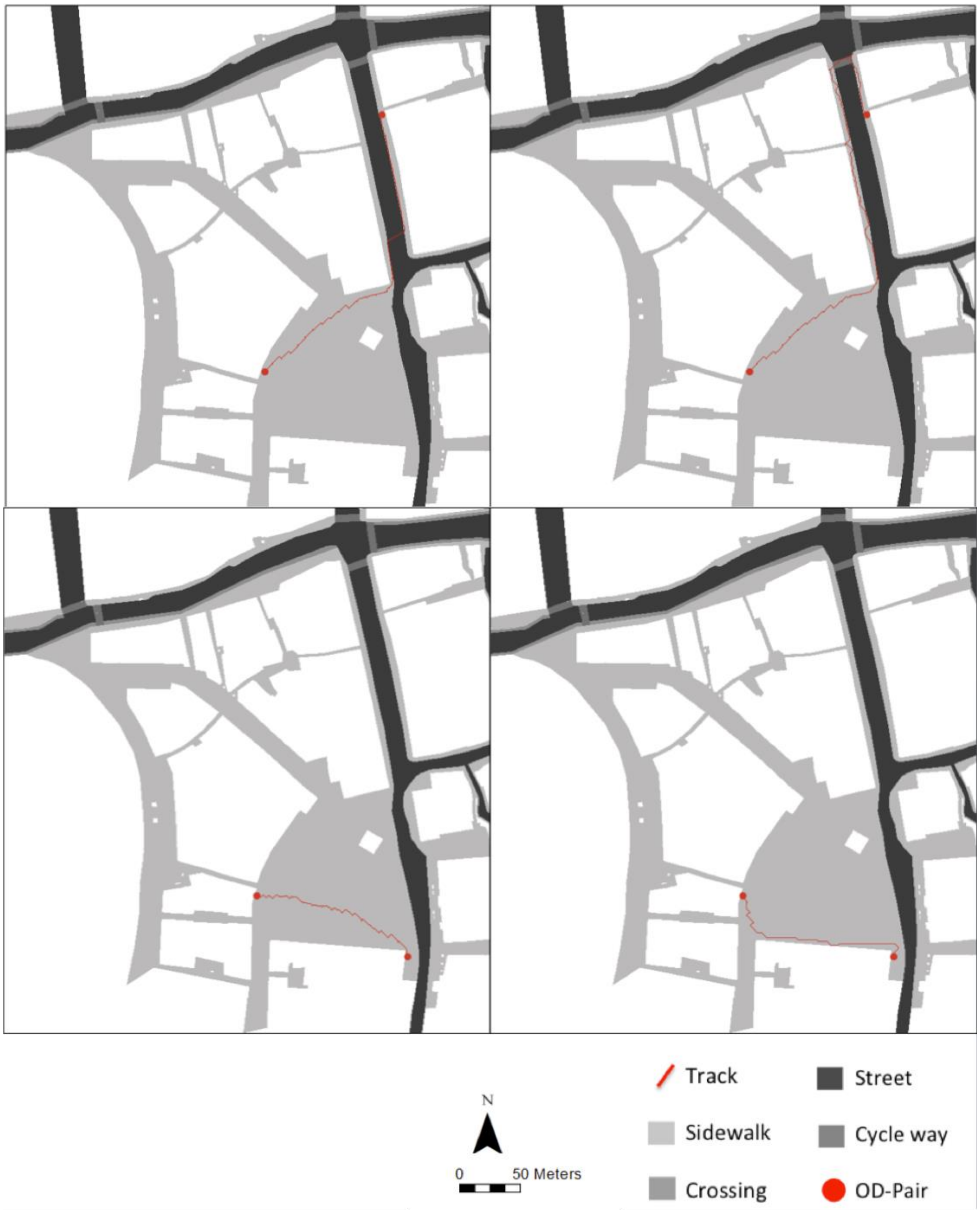


Fig. 23: Results for Agent 1 (upper left), Agent 2 (upper right), Agent 3 (lower left) and Agent 4 (lower right)

With regards to their properties, the agents are similar, but differ to a large degree in the relative importance they attribute to certain aspects of walking. In particular, agent 3 could represent a commuter who places particular emphasis on reaching the destination as quickly as possible, as modeled with high weights being put on *rA_ReachDestination_weight_sA_AgentMove* and *weight_sA_AgentMove*. Agent 4, in contrast, is more prone to diverting from the direct path, since it puts relatively high importance on pleasure-related aspects of the walking trip, in particular *rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure*. A comparison of the chosen paths visible in the lower half of figure 23 demonstrates clear differences. The strong tendency of agent 4 to remain on the southern edge of the square can be explained with a narrow zone which ranges from west to east and affords the visibility of up to 10 aesthetical buildings simultaneously, but ends abruptly after a few meters to the north, when the view is blocked by another building. The agent, due to its high preference to see aesthetical buildings, tends to remain in this zone for as long as possible, until finally heading towards its destination.

#### *5.2.3.2 Local Sensitivity Analysis*

After visually assessing the model's behavior in clearly defined scenarios, it is to be tested in a more systematic and thorough manner. When planning a sensitivity analysis, both local and global techniques can be used, and are often applied in combination. The choice of an appropriate strategy is normally determined by the intended use of the sensitivity analysis. In this context, Pannell (1997) described four categories: decision making, communication of model results, increased understanding and quantification of the system, and model development.

In our context, the motivation for conducting a sensitivity analysis can be related to Pannell's (1997) third point, the understanding and quantification of the system behavior. In particular, the model is to be tested for its correctness, meaning that pedestrian agents should succeed or fail in reaching their destinations in a way which is plausible with regards to their individual capabilities. Further, it is our intention that the functional places which are perceived, formed and used by the agents, are of individual and subjective character, a fact which is expected to result in reasonable behavioral adaptions with altered agent parameter settings. Since the exact reproduction of actual pedestrian behavior is not within our scope, however, a systematic test of interdependencies between individual parameters is unnecessary. On the basis of these preconditions, and with respect to the high computational effort caused by the large number of changeable parameters in our model, a local sensitivity analysis is chosen.

##### *5.2.3.2.1 Procedure*

Table 6 lists the tested model parameters and the values which they are allowed to take on. In order to evaluate the effect of differing pedestrian preferences on the resulting places, the agent-specific weight coefficients are systematically altered. In addition, the agent properties are varied, which has an effect on the capability values used for the suitability calculations. Apart from these, though, three global model parameters are tested, namely *viewfieldWidth*, *viewfieldDistance*, and via *exponent*, the effect of a different suitability calculation formula. With regards to the latter, by adding an exponent $\neq 1$ to the dependency function, the relationship between $\pi$ and the suitability value can be changed from a linear to a non-linear function, which can also be expected to influence the results.

After the identification of the model parameters to be tested, their possible values need to be determined. For this, the weight coefficients are homogeneous, and can take on three possible settings ranging from no influence (0), average influence (0.5) and high influence (1) on the place formation process. Concerning the other agent parameters, however, a definition of appropriate values is much more complicated, since, as Pannell (1997) suggested, these should be realistic and probable. Thus, whenever possible, we base our selection of values on the existing literature on pedestrian characteristics or established reference values and guidelines used in design and planning.

With regards to the first parameter, *minSurface*, large differences in pedestrians' sensitivity to surface unevenness have been demonstrated in the empirical literature (e.g. Sanches et al. 2007, Basbas et al. 2010). Thus, with reference to our classification of surface types as discussed in chapter 5. 1, all scenarios are tested, ranging from a pedestrian who is completely insensitive to surface unevenness to a severely mobility-impaired person requiring the highest possible quality level. Concerning *maxSlope*, the lowest possible value is set to 3%, which is in accordance with a common guideline for barrier free sidewalks (FGSV 2011). The maximum value is set to 15%, which, according to Weidmann (1993), marks the maximum gradient above which the use of stairs is recommended. Concerning the possible body *width* of a pedestrian, values are set to range from 0.4 meters, which has been stated as the minimum space required by a pedestrian in a dense crowd (Chen et al. 2009, Chu 2011), to 0.85 meters, a reference value used for a person in a wheelchair or with crutches (Ackermann 1997). Guidelines for the attribute *waveringMovement* are provided by Buchmueller and Weidmann (2006), who stated values ranging from 0.2 to 0.35 meters, which are adopted in our analysis. The same source, as well as official recommendations, are consulted to define minimum and maximum values for *shyAwayDistTraffic*, *shyAwayDistCycle*, and *shyAwayDistWall*, which are set to range from 0.3 to 0.8 meters for the first two attributes, and 0.15 to 0.45 meters for the third one (Buchmueller and Weidmann 2006, ADFC 2011). The definition of appropriate values for *maxLight* is more problematic, which is due to a lack of empirical studies. Thus, the lower boundary is set to 45 meters, which corresponds to a recommendation with regards to street light density made by Richter and TRILUX (2005). This value is arbitrarily doubled to define a maximum boundary. The highest possible distance between a pedestrian and a bench is defined in *maxBench*, and set to range from 1000-2500 meters, which refers to the average walking distance for different age groups (Ewert 2012). The crossing behavior of pedestrians of different age groups is well researched, as well, which is why the values for *minTimeKerbDelay* can be based on the observations by Oxley et al. (1995), and accordingly lie between 2 and 3 seconds. Concerning the attribute *walkingSpeed*, there are large variations among pedestrians, with typical values ranging from 3.6 to 4.8 km/h (Buchmueller and Weidmann 2006). Finally, the allowed values for the *maxHeight* attribute are based on official guidelines, which state that curb stones up to a height of 3 centimeters are rated as barrier free, even for wheelchair drivers (FGSV 2002). The maximum value of 30 centimeters is chosen arbitrarily.

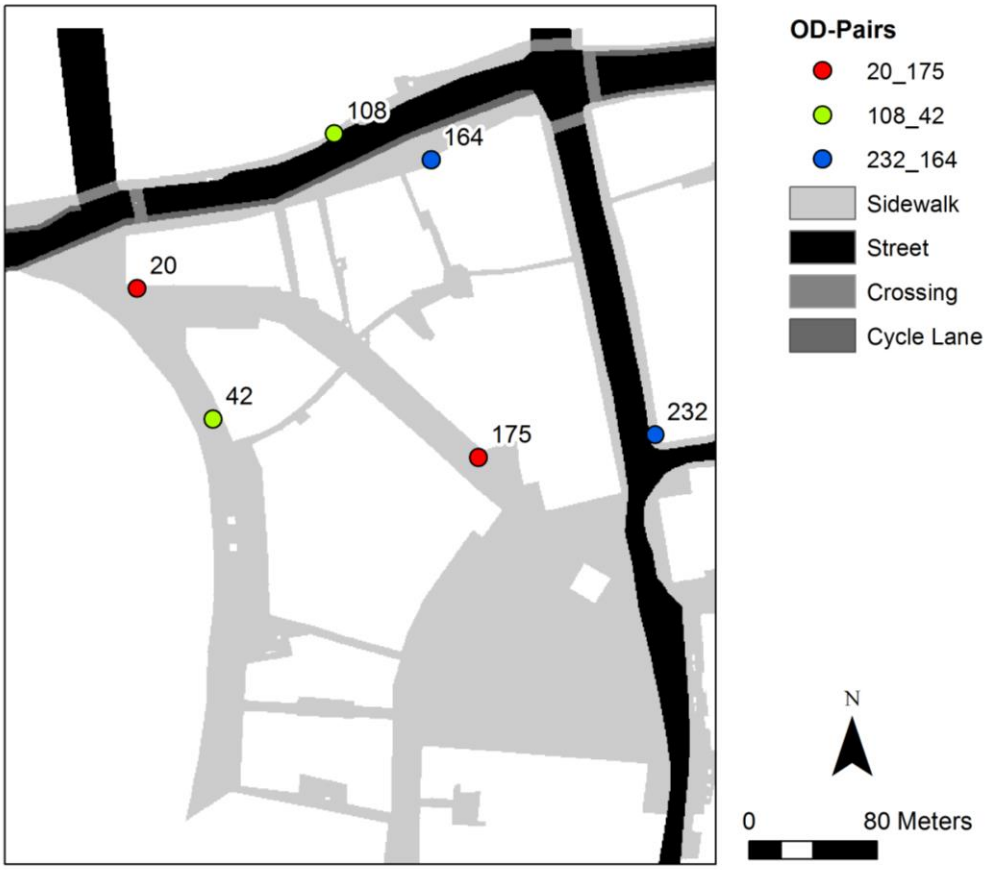


Fig. 24: OD-Pairs for Sensitivity Analysis

For each range between the minimum and the maximum value, 5 equal intervals are set, before each value setting is tested in a separate simulation run for three distinct origin-destination pairs which are shown in figure 24. These were chosen from the set of potential candidates with the aim to represent the variability of the study area with streets, cycle lanes, crossings and sidewalks as much as possible.

Finally, it is necessary to define a set of model outcomes against which the input parameters are to be tested. In the context of this thesis, the characteristics of the resulting places are of interest. Thus, for each simulation run, selected index values for selected attributes of the corresponding patches are calculated and recorded. In general, these attributes, a selection of which is listed in table 7, comprise minimum, maximum and mean values, information about whether the agent was successful in reaching the destination, the overall path length, and the number of jaywalking actions per simulation run. Please note that the full results can be found in Appendix D.

| Input Parameters | Parameter Values |
|---|---|
| Weight Coefficients | |
| weight_sA_AgentMove | 0, 0.5, 1 |
| weight_sA_AgentUpdateSafety | 0, 0.5, 1 |
| weight_sA_AgentUpdatePleasure | 0, 0.5, 1 |
| weight_sA_AgentUpdateComfort | 0, 0.5, 1 |
| weight_sA_AgentCrossStreet | 0, 0.5, 1 |
| rA_KeepBalance_weight_sA_AgentMove | 0, 0.5, 1 |
| rA_OvercomeSlope_weight_sA_AgentMove | 0, 0.5, 1 |
| rA_OvercomeHeightDifference_weight_sA_AgentMove | 0, 0.5, 1 |
| rA_AccessPatch_weight_sA_AgentMove | 0, 0.5, 1 |
| rA_ReachDestination_weight_sA_AgentMove | 0, 0.5, 1 |
| rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety | 0, 0.5, 1 |
| rA_StayNearLight_weight_sA_AgentUpdateSafety | 0, 0.5, 1 |
| rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety | 0, 0.5, 1 |
| rA_SeeGreenery_weight_sA_AgentUpdatePleasure | 0, 0.5, 1 |
| rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure | 0, 0.5, 1 |
| rA_NotSeeTraffic_weight_sA_AgentUpdateComfort | 0, 0.5, 1 |
| rA_StayNearBenches_weight_sA_AgentUpdateComfort | 0, 0.5, 1 |
| Agent Properties | |
| minSurface | 1, 2, 3, 4, 5 |
| maxSlope | 3, 6, 9, 12, 15 |
| width | 0.4, 0.5125, 0.625, 0.7375, 0.85 |
| waveringMovement | 0.20, 0.2375, 0.275, 0. 3125, 0.35 |
| shyAwayDistTraffic | 0.3, 0.425, 0.55, 0.675, 0.8 |
| shyAwayDistCycle | 0.3, 0.425, 0.55, 0.675, 0.8 |
| shyAwayDistWall | 0.15, 0.225, 0.3, 0.375, 0.45 |
| maxLight | 45, 56.25, 67.5, 78.75, 90 |
| maxBench | 1000, 1375, 1750, 2125, 2500 |
| minTimeKerbDelay | 2, 2.25, 2.5, 2.75, 3 |
| walkingSpeed | 3.6, 3.9, 4.2, 4.5, 4.8 |
| maxHeight | 0.03, 0.0975, 0.165, 0.2325, 0.3 |
| Other Model Parameters | |
| exponent | 1, 1.5 |
| viewfieldWidth | 90, 115, 170 |
| viewfieldDistance | 5, 15 |

Table 6: Local Sensitivity Analysis Parameter Setting

*5.2.3.2.2 Results*

In the past, numerous methods to calculate and express the parameter sensitivity of a model have been proposed, and include the calculation of the gradient of the dependency function, elasticities as the ratio of the total percentage of change of both the parameter and the resulting value, and numerous sensitivity indexes (Pannell 1997). A simple yet useful index has been proposed by Hoffman and Gardner (1983):

$$SI = (D_{max} - D_{min})/D_{max} \tag{15}$$

where $D_{max}$ is the resulting output value when the parameter is set to its maximum testable value, and $D_{min}$ when it is on its minimum value .

This index is also used in our analysis, and calculated for each pair of parameter setting and place attribute index. It is, however, slightly modified in the sense that in cases where $D_{max} < D_{min}$, the formula is altered:

$$SI = -(D_{min} - D_{max})/D_{min} \tag{16}$$

This is done in order to be able to infer from the resulting value both the exact magnitude, but also the direction in which the model output is changed. Thus, a positive $SI$ indicates a positive correlation, whereas the opposite is true for a negative $SI$. In order to ensure comparability among different test runs, the results of a simulated walk are only considered for further analysis if the agent has successfully reached its goal.

Table 7 shows the averaged results for all three OD-pairs. For reasons of clarity, each model output value is opposed only by selected input parameters, namely the ones for which an effect was expected and intended. For further details, please see the results for all parameters provided in Appendix D.

The results demonstrate the model's sensitivity to almost all tested parameters, however, with large variations in magnitude. For the majority of parameter settings, it reacts in accordance with our expectations and intentions. Thus, for instance, equipping an agent with a higher *maxSlope* attribute, and therefore with the ability to overcome higher gradients, results in the generated places to be roughly 34% more inclined than with a lower parameter setting. On the contrary, though, an increase of the relative importance which an agent allocated to the realization action *RA_OVERCOMESLOPE* or its superior semantic action *SA_AGENTMOVE* results in gradients which are lower by 12% and 21%, respectively. To state another example, an increase of a pedestrian agent's preference to see urban greenery leads to a growth of the number of visible trees by 3%, whereas a higher general preference for walkability aspects related to pleasurability leads to a 5% rise.

For other parameters, however, no or only minimal model sensitivity is observed. Hence, the effects of *maxBench* or *maxLight* on the mean values for *benchDist* or *lightDist* of the resulting places are negligible. Thus, agents with differing needs with regards to the closeness of benches or to street lights show no notable behavioral differences. In the case of *benchDist*, this can be explained by the relatively high difference between the values set for the agent property *maxBench*, which range from 1000 – 2500 meters, and the actual patch properties *benchDist*, which reach a maximum value of only 212 meters in our study area. This discrepancy results in the general trend of $\pi$ values being very close to $\pi_0$, which, in turn, leads to high suitability values. The same is true for the patch attribute *lightDist*, which reaches a maximum of 42 meters in our study area. In addition, when calculating the suitability values for the corresponding realization actions *RA_STAYNEARLIGHT* and *RA_STAYNEARBENCHES*, the differences in values for *benchDist* and *lightDist* of patches located within the viewfield of an agent can be expected to lie in the range of merely centimeters to a few meters. This leads to very small differences in derived suitability values, which are easily compensated by other factors.

Apart from such insensitivities, there are also results which are unexpected, and require additional explanations and exploration of the model behavior. An example is the positive effect of *weight_sA_UpdateSafety* on the mean attribute *lightDist* of the chosen places. In this case, one would expect a more safety-aware agent to behave in an opposite way, namely to decrease the distance to street lights. A possible explanation for this seemingly erroneous output is provided by the fact that within the study area, street lights are frequently positioned as hanging directly above the center of the street. Therefore, the appropriate agent behavior would in fact result in less safety buffer to the street, and therefore be conflicting with *RA_KEEPDISTANCETOMOTORTRAFFIC* and in some cases also with *RA_KEEPDISTANCETOCYCLETRAFFIC*. In addition, with an increase of *weight_sA_UpdateSafety*, the relative importance of these sub-actions grows as well. Another observation further supports this hypothesis, namely the fact that the unexpected agent behavior is not observable in the test runs for OD-pair 20-175, in which the agent moves mainly outside of the zone of influence of streets and cycle lanes. Nevertheless, an additional test run was conducted in which the weight coefficients of *RA_KEEPDISTANCETOMOTORTRAFFIC* and *RA_KEEPDISTANCETOCYCLETRAFFIC* were set to 0. Without this conflicting influence, *weight_sA_UpdateSafety* showed a negative effect on the mean distance to street lights of -0.6%.

Another aspect which requires further discussion is posed by the negative influence of *shyAwayDistTraffic* to the mean *minDistStreet*. Again, the opposite was expected, since the need for a larger safety buffer should draw an agent away from the street. A possible reason can be found in the sidewalk width. In both OD-pairs 232-164 and 108-42, the sidewalk on which the agent moves is relatively narrow, and therefore restricts its possibilities to maintain the preferred buffer distance to the street. In fact, a further analysis of the resulting places show that in the majority of cases, the suitability calculated for *RA_KEEPDISTANCETOMOTORTRAFFIC* was 0. Consequently, changing the value of *shyAwayDistTraffic* could not have any effect on the chosen path. In order to test the correct functionality of the model, the experiment was repeated with a different OD-pair (5-71), which enabled the agent to move on a more spacious sidewalk, and therefore provided more options for keeping the preferred safety buffer. In this setting, with 6.6% increase, a clear and plausible positive effect of *shyAwayDistTraffic* on *minDistStreet* could be demonstrated.

In a comparable case, an unexpected negative effect of *waveringMovement* on the mean *minDistCycle* and *meanBuildDist* could be observed. This, however, can be explained with the contribution of this agent attribute to the capabilities used for the suitability calculation of *RA_ACCESSPATCH*, *RA_KEEPDISTANCETOMOTORTRAFFIC* and *RA_KEEPDISTANCETOCYCLETRAFFIC*, which are in many cases conflicting actions, as it has already been explained. Thus, additional test runs were conducted for OD-pairs 232-164 and 108-42, at the same time alternating which weight coefficients was set to 0. The results show an effect of *waveringMovement* of 1.3% for *minDistCycle* if conflicts are excluded, and 3% for *meanBuildDist* in the opposite case.

Moreover, there were several parameters which were tested but not included in table 6. Thus, for instance, the number of jaywalking processes was counted for each simulation run. In total, there were 6 such processes, which all occurred in 232-164, and affected 5.3% of the simulation runs for this OD-pair. Due to the low number, though, no particular effect of a model parameter could be identified.

We also tested for the sensitivity of the surface type, however, due to the spatial concentration of predominantly homogeneous surface types in large zones of the study area, there was in many cases no difference or sensitivity visible. Only in 232-164, a positive effect of the *minSurface* attribute on the minimum value found for surface type in the resulting places could be observed.

Apart from parameter related to agent attributes, we also tested the effects of *viewfieldWidth*, *viewDistance* and an exponent added to the suitability function. Here, it was found that a higher angle of *viewfieldWidth* increases the length of the total path by 11%, which can be explained by a higher possibility for an agent to divert from the direct path. Additionally, it increased the *minDistStreet* by 17%. With regards to the *viewDistance*, no clear pattern was found due to the fact that the model reacted mainly insensitive to its alterations. The exponent which influenced the linearity of the suitability calculation function had stronger effects on almost all aspects of the model outcome, however, without a clear pattern to be observed. For the exact results, please see Appendix D.

| Place Characteristics | Input Parameters | SI |
|---|---|---|
| mean slope | maxSlope | 0.3414 |
| | rA_OvercomeSlope_weight_sA_AgentMove | - 0.2146 |
| | weight_sA_AgentMove | - 0.122 |
| mean benchDist | maxBench | 0 |
| | rA_StayNearBenches_weight_sA_AgentUpdateComfort | 0 |
| | weight_sA_AgentUpdateComfort | 0 |
| mean lightDist | maxLight | 0.0048 |
| | rA_StayNearLight_weight_sA_AgentUpdateSafety | - 0.0041 |
| | weight_sA_UpdateSafety | 0.0082 |
| mean buildVis | rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure | 0.009 |
| | weight_sA_UpdatePleasure | 0.0469 |
| mean treeVis | rA_SeeGreenery_weight_sA_AgentUpdatePleasure | 0.0317 |
| | weight_sA_UpdatePleasure | 0.0542 |
| mean minDistStreet | width | 0.0337 |
| | waveringMovement | 0.0066 |
| | shyAwayDistTraffic | - 0.0278 |
| | rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety | 0.0307 |
| | weight_sA_UpdateSafety | 0.0779 |
| mean minDistCycle | width | 0.0134 |
| | waveringMovement | - 0.0089 |
| | shyAwayDistCyc | 0.0029 |
| | rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety | 0.0084 |
| | weight_sA_UpdateSafety | 0.0254 |
| mean patchHeight | maxHeight | 0.6055 |
| | rA_OvercomeHeightDifference_weight_sA_AgentMove | - 0.6596 |
| | weight_sA_AgentMove | - 0.1516 |
| mean trafficVis | rA_NotSeeTraffic_weight_sA_AgentUpdateComfort | - 0.3168 |
| | weight_sA_AgentUpdateComfort | - 0.3177 |
| mean buildDist | width | 0.007 |
| | waveringMovement | - 0.0208 |
| | shyAwayDistWall | 0.0185 |
| | rA_AccessPatch_weight_sA_AgentMove | 0.2925 |
| | weight_sA_AgentMove | 0.1446 |
| Path length | rA_ReachDestination_weight_sA_AgentMove | - 0.0003 |

Table 7: Average Results of the Sensitivity Analysis

### 5.2.4 An Exemplary Visualization of a Collective Place

So far, the concepts developed in this thesis have focused on the process of an individual agent perceiving and using a functional place. However, as discussed in Chapter 2. 2. 2. 1, there are also collective places which emerge from individual place formation processes converging at a location (Golledge and Stimson 1997, Carmona et al. 2010). Since the identification, analysis or prediction of such important places is of interest for geographers and urban planners, in this chapter, the conceptual connection between the simulation of individual place formation processes and emerging collective phenomena is demonstrated on the example of potential collective pedestrian places in the sense of specific areas of the walking path network where the human actions, in this case *WALKING*, overlap. Apart from the pedestrian frequency, in order to, for instance, assess the need for infrastructural improvements to enhance the walking conditions at such places, it would also be worthwhile for planners to receive information about the perceived walkability.

In our example, we choose the north western entrance to the pedestrian precinct as study area. An all-pair routing is conducted with pedestrian agents moving between 27 nodes. Thus, in total, 729 simulation runs are conducted, with agent attributes and preferences set randomly within the appropriate ranges defined in the previous chapter. For all runs, the resulting functional places are recorded and provide the basis for the post-processing in which for each patch, the frequency of being part of a functional place is calculated. Further, the perceived suitability values with regards to all realization actions are averaged for each patch, and added as an attribute. Afterwards, in order to enhance the visual clarity of the resulting map, using ESRI's IDW tool, a raster surface is interpolated for the frequency as well as the suitability values, and visualized in ESRI's ArcScene (ESRI 2012).

The result can be seen in figure 25, which shows the study area as seen from north to south, with the main city square in the background of the picture. The usage frequency is plotted on the z-axis, whereas the average perceived walkability is color-coded from red to green.

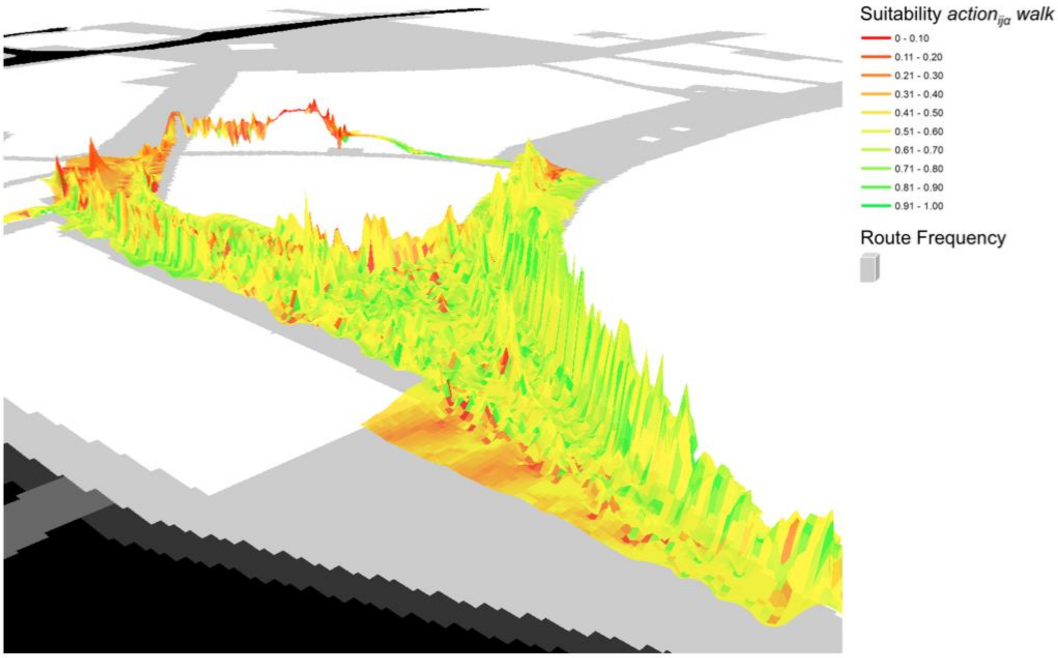


Fig. 25: Collective Places and Perceived Walkability

As one can see, there are considerable differences to be detected, for instance a potentially important pedestrian place directly at the entrance to the pedestrian area at its western edge. Here, high pedestrian frequencies are to be expected, while the walkability is generally perceived as positive by the agents. A different example, however, can be seen further to the south-east, where pedestrian frequencies peak in a narrow passage, due to the existence of stores and other potential destinations, but the walkability is evaluated considerably lower which is primarily due to restricted space. For a planner, these places with high frequencies but lower walkability might be potential candidates for effective infrastructural improvements.

### 5.2.5 Conclusion: Simulating Pedestrian Movement with PlaceBuilder

As a final step in the development of a practical application for the PlaceBuilder extension, this chapter presented an agent-based simulation of pedestrian movement in Augsburg, Germany. Based on a model of subjective walkability, its aim was to demonstrate the correctness and practical value of the computational model of place. For this, an artificial pedestrian agent was developed which is situated within a detailed model of the urban walking environment, and uses PlaceBuilder to generate functional places for walking based on its perceived walkability. Such places, accordingly, consist of a subset of patches which, in their entirety, afford the pragmatic action *PA_WALK* for the agent by forming a walkability-optimized path through the its viewfield.

After the description of its components, processes and general scheduling during a simulation run, the model was tested for its face validity and sensitivity to parameter changes by observing its behavior in pre-defined scenarios and conducting a local sensitivity analysis. The general functionality and sensitivity of the resulting places to the agent's attributes could be demonstrated. Finally, the exemplary analysis and visualization of collective pedestrian places in our study area has indicated the practical value of our approach for urban design and planning.

# 6 Results and Discussion

This thesis was concerned with the computational representation and localization of functional places. Its main aim, thus, lay in the development of a computational model of functional place, which was conceptualized as a sub-set of geo-atoms bound by a functional unity condition with regards to a complex, high-level spatial action. In order to acknowledge the subjective character of human place formation processes, it was proposed in Chapter 1 to dynamically generate these functional places from spatial data by simulating the mental process of individual place formation with ABM. Therefore, this work explicitly addressed two main challenges which arise when integrating the notion of place in GIS: the subjectivity of the formation and perception of places, especially with regards to their suitability for action, and their compound structure.

In this chapter, the central goal is revisited, and discussed against the background of the derived results. Concerning its structure, this chapter refers to the main challenges stated above: First, the focus is set on the evaluation of the spatial suitability of places, and the development of a conceptual framework for simulating such mental processes. Then, the conceptualization of places as functional compounds of geo-atoms and its computational implementation is resumed, before finally, this chapter concludes with a discussion of the implications which result from this work, as well as its limitations.

## 6.1 Modeling Individual Spatial Suitability Assessment

A subordinate aim of this thesis was the development of a simulation framework for individual spatial suitability assessment. On the one hand, this was motivated by the importance of the perceived suitability for place formation and the resulting spatial behavior, and therefore its significance for simulating these processes, as discussed in Chapter 2. 2. On the other hand, however, it was found in Chapter 2. 3 that the problem of modeling how an individual human agent subjectively perceives and evaluates spatial suitability is still largely unaddressed in the GISc community.

### 6.1.1 Revisiting the Approach

Against this motivational background, this work approached the notion of spatial suitability based on the hypothesis that it must be modeled as an abstract quality of a system which incorporates an agent, a geo-atom and a particular action. In order to substantiate this claim, the general process of human perception of functional information within their environment was examined from a psychological perspective in Chapter 2. 1. Based on Gibson's (1979) work, it was argued that an affordance in general is not inherent in the environment or the agent alone, but depends on the correspondence of selected properties of both. In accordance with Stoffregen (2003), thus, an affordance was understood as a systemic property, which, as Warren's (1995) experiments and many others demonstrated, can be measured as a ratio when the corresponding agent- and environment-related properties, as well as the critical threshold values, are known.

On this theoretical basis, spatial suitability was conceptualized in Chapter 4. 1 as an extended notion of affordance, which, instead of merely expressing binary statements about its existence or non-existence in an agent-environment-action system, allows for the derivation of graded values. Thus, the $suitability_{ij\alpha}$ which is provided within a system $W_{ij\alpha}$ consisting of an $agent_i$, an environmental $primitive_j$ and an $action_\alpha$ can be inferred from the relative value of its ratio $\pi_{ij\alpha}$ with regards to an optimal point $\pi_O$ and a critical threshold value $\pi_{max}$ or $\pi_{min}$, respectively.

The present conceptualization of affordances, however, largely restricts its focus to simplistic actions on a fundamental level, such as Johnson's (1987) image schemata (Kuhn 2007). Actions typically conducted at places, however, are of much higher complexity, and often refer to a higher level of abstraction, such as *SHOPPING* or *WALKING*. Although Gibson (1979) described more complex, higher-order affordances, they are mostly neglected by existing work on the formalization or measurement of affordances (Warren 1995, Scheider and Janowicz 2014). In this thesis, therefore, it was hypothesized that, in order to assess the overall suitability of a functional place for conducting a complex action, which we equated with the achievement of a desired goal state, an agent needs to evaluate the suitability which is provided for each contributing action. A basic assumption, therefore, was that it is possible to define and model complex actions as hierarchically structured phenomena, which consist of numerous contributing sub-actions on several levels. A corresponding framework was developed in Chapter 4. 1. 1 on the basis of activity theory (Leontiev 1979), drawing in particular from Kemke's (2001) description of three levels of action abstraction. Concretely, notions from activity theory and abstractions used in ABM were synthesized to model pragmatic actions (goal state) with contributing semantic actions (transition model), and realization actions (operational agent behavior). Since agent-environment interactions sensu stricto occur on the fundamental realization action level, suitability values can be calculated there and be transferred to the higher-level actions by means of dependency relationships between the hierarchical levels. Although the specificity of an appropriate method for suitability calculation with regards to the particular system has been emphasized, a general approach has been developed, which starts with the initialization of the suitability variable, which is followed by the definition of the relevant capability and disposition values $cap_{ij\alpha}$ and $disp_{ij\alpha}$ from the properties of agent and geo-atom, and the values for $\pi$, $\pi_O$ and $\pi_{max}$ or $\pi_{min}$, before the $suitability_{ij\alpha''}$ can be calculated based on a linear or non-linear mapping function.

### 6.1.2 Discussion of the Results

The conceptual framework of spatial suitability assessment was implemented as part of a software (PlaceBuilder) which enables artificial agents to dynamically generate functional places in their virtual environment. Concretely, it was demonstrated in Chapter 5. 1 that, based on the empirical literature, it is possible to model a real-world action, in our case *WALKING*, as an hierarchically-structured phenomenon in accordance with our notion of suitability. During the simulation, when forming places, the agents used the developed suitability model to pre-calculate walkability values of perceived geo-atoms which, based on a suitability optimization strategy, were encapsulated to functional units.

This exemplary application of the framework and the results derived via testing the simulation model demonstrate the usefulness and functionality of our approach, as well as the necessity to model suitability as a systemic quality. Its potential of generating heterogeneous, but at the same time plausible places as demonstrated by the behavior of agents with differing attributes and preferences was revealed both in the pre-defined scenarios and in the sensitivity analysis in Chapter 5. 2. 3. In the first scenario, for instance, a clear behavioral adaption can be observed based on the perceptual differences with regards to the suitability for road crossing. Thus, whereas the first agent easily overcomes the barrier posed by the curb stone, and chooses to jaywalk to make a short cut, the second agent has no other possibility than to follow the sidewalk until it reaches a curb cut.

In the second example, the influence of differing preferences of pedestrians, incorporated in the suitability calculation as weight coefficients, are demonstrated. Thus, two pedestrian agents, similar in capabilities but with different preferences, which move between the same OD-pair, show considerable differences in their place formation strategies. Thus, the commuter-type agent is not interested in other walkability aspects apart from simply being physically able to move towards its destination, whereas the strolling-type agent is much more prone to diverting from the direct path. These conflicting interests are mirrored in their chosen places for walking.

A more systematical assessment of the effect of changes to the agent parameter settings on the resulting suitability calculations, and ultimately the generated functional places, was conducted by means of a sensitivity analysis. Again, the results support the appropriateness of our approach, since they demonstrate clear and reasonable differences in the computed suitability values and, consequently, behavioral adaptions of the involved agents. There have been, however, large variations in the magnitude of the observed effects of individual agent parameters, which depend primarily on the heterogeneity of the involved patch attributes simultaneously perceived by the agent. The minor effects of *maxBench* and *maxLight* on the respective place attributes, for instance, results from the largely homogeneous character of proximal patches.

In general, the possibility to compute suitability values which are tailored to the specific agent, environmental setting and action represents an extension of GIS-based suitability modeling and utility-based agent approaches. The definition of unambiguous dependency functions helps to be "objective in measuring the subjective", and represents a novel approach to the translation of objective space to subjective realities (Couclelis 1992, p. 228-229). As our results have demonstrated, our method of incorporating agent subjectivity and heterogeneity by defining functions which map percepts to individual beliefs about spatial suitability allows to simulate the process of individual suitability evaluation. In the case of artificial agents, it is thus possible to equip them with the ability to develop and pursue individually optimized action strategies, such as *CROSSING A ROAD* or continue *WALKING ON THE SIDEWALK*, or generally choosing a more suitable path.

## 6.2 Places as Functional Compounds of Geo-Atoms

Apart from incorporating the subjective aspect of place, a further challenge addressed in this thesis was the development of a method for modeling the compound nature of functional places. A motivation for this endeavor was posed by the fact that the places which are used or referred to by humans often do not exist as a distinct spatial entity in a database, which represents a common problem for GIS. This is due to the fact that places are often defined by the relationships between multiple geo-spatial entities rather than a fixed boundary (Winter and Freksa 2012). In particular for the generation of functional places, as intended in this thesis, methods for the identification and encapsulation of appropriate geo-atoms to a composite functional unit are needed, have, however, not been explicitly addressed before.

### 6.2.1 Revisiting the Approach

In an extension of sub-hypothesis 2, which has been discussed in the previous chapter, the compound nature of places was explicitly addressed by postulating that a functional place which affords a pragmatic action consists of the geo-atoms that afford the contributing sub-actions with the highest possible suitability. In order to support this hypothesis, the phenomenological concept of place was examined in Chapter 2. 2. Based on work done by geographers, urban designers and psychologists, places were defined as meaningful spatial units, which are a result of our mental structuring of space. From these individual place formation processes, collective places can emerge. In particular, the role of places as action spaces was identified as highly relevant with regards to their formation and identity. Actions, as has been discussed in the previous chapter, are afforded by the physical setting, meaning the objects which exist at, and ultimately define the footprint of a place. For a functional place to be selected and used, however, apart from its affordance, the suitability with regards to a particular use is of critical importance, since it is perceived by a prospective user and influences the perceived place utility (Golledge and Stimson 1997). In GISc, the affordance-concept has been identified as a potential approach to modeling places, in particular because it allows for the composition of functional places from lower-level spatial entities, an idea which, however, has not yet been used to generate or localize places, as it was discussed in Chapter 2. 3.

For our approach, ABM were identified in Chapter 3. 1 as a potential modeling framework since they apply a disaggregate, microscopic perspective on modeling geographic phenomena and processes. Thus, on the basis of the framework for simulation suitability assessment revisited in the prior chapter, functional places were conceptualized in accordance with the hypothesis, and a model of functional place formation developed in Chapter 4. 1. 2 which describes how an agent, equipped with sensors, actuators and a BDI-framework, created functional places from its percepts and with regards to its individual capabilities and preferences.

### 6.2.2 Discussion of the Results

One main result of this thesis is a conceptual model of functional places and its computational implementation, as presented in Chapter 4. Using the NetLogo multi-agent simulation environment, the extension package PlaceBuilder was developed. This software allows NetLogo modelers to incorporate complex actions, which are modeled as hierarchical structures with several supporting sub-actions on lower levels that are mutually connected in causal and spatial relationships. A major contribution is represented by the possibility to develop NetLogo agents with the ability to perceive higher-order affordances which are the result of the combinatory use of compounds of several patches, or functional places. Each of these geo-atoms affords one or more contributing realization actions, and provides the agent's counterpart for the evaluation of the suitability. A functional place might therefore be optimized by using alternative behavioral strategies which may incorporate an alteration of the course of action or the place itself.

The results of the pedestrian simulation described in Chapter 5. 2. 3 support the hypothesis that the functional places generated according to our framework are sensitive to the needs and characteristics of the individual agent, as is indicated by its different behavioral strategies discussed in the previous chapter. Further, the observation of the agent behavior demonstrates the correctness of the place formation processes. Thus, in most simulation runs, the pedestrian agent succeeded in reaching its destination and demonstrated the intended walking behavior. On the other hand, however, agents with insufficient capabilities were not able to reach their destination, which is plausible. Furthermore, the spatial dependencies among realization and semantic actions function as intended, which is evident from the fact that the agents move only among neighboring patches, and conduct the actions with a *sameSystem* relationship at the same patch. The problem which occurred when an agent is located within a narrow passage, or close to a building wall or street, and which consisted in the agent not perceiving enough patches to be able to conduct the pragmatic action, in our case to walk 15 steps, could be solved by the introduction of the alternative *PA_WALKONESTEP*.

A noteworthy aspect are mutual influences of conflicting sub-actions, such as *RA_ACCESSPATCH* and *RA_KEEPDISTANCETOMOTORTRAFFIC*, which influence the agent to move in opposite directions. In order to avoid a complete compensation of the contradictory motivational drives, the corresponding weight coefficients can be adjusted to define a dominant influence. In general, a comparison of the three OD-pairs has shown that the model behavior depends to a considerable degree on local environmental conditions which can cause independent actions to come into conflict with each other, for instance at sidewalks with nearby streets and buildings.

In general, in the course of developing the hierarchical action model, and defining the corresponding suitability calculation procedures, a modeler needs to carefully consider the comparability of alternative action strategies. Imagine two parallel branches of an action graph, for instance, which each provide an alternative action strategy to reach the goal state. One of them, however, consists of 2 separate sub-actions whereas the other requires the performance of 4 sub-actions. Then, naturally, an agent would be likely to choose the one with the fewer actions to be performed for reaching the goal. Of course, this is fully intended, however, can be problematic since the effect of lower suitability values in the first action sequence can be compensated. Thus, for instance, in the action model of *PA_WALK*, for instance, whereas the choice to jaywalk requires only one semantic action $sub\text{-}action_{\alpha'}$ *SA_AGENTCROSSROAD* with three corresponding realization actions, walking on the sidewalk comprises 4 semantic actions and many more realization actions. Therefore, in order to make sure that jaywalking is not over-represented in the agent's behavior, its corresponding weight needs to be set to a very low value.

Finally, the exemplary analysis of the usage frequencies of patches and their perceived suitability values for walking in Chapter 5. 2. 4 provided a brief illustration of our notion of an emerging collective place. Although at this stage, our work focused on the simulation of individual place formation, the analyses demonstrated the potential value of the developed simulation tool for localizing not only individual, but also collective places, and assisting planners in not only discovering prominent pedestrian places in the transportation network, but also assessing whether infrastructural improvements are necessary and would be worthwhile.

All things considered, the results show that the computational model of functional place developed in this thesis allows for the calculation of places which are plausible, sensitive to the capabilities and preferences of the individual agent, functional with regards to a complex action, and spatial in the sense that they are composed of spatially located geo-atoms. Therefore, they fulfill major characteristics of real-world places. In the context of ABM, our approach allows for a higher level of behavioral complexity to be incorporated, since the agents are able to adapt their action strategy and generate different places based on their individual needs. The fact that the complex affordances are created by the agents and not simply embedded in the environment results in the fact that the same environment can be interpreted and used differently by the agents.

## 6.3 Implications and Limitations

In accordance with its thematic scope, the implications of this thesis can be placed in GISc and ABM. In particular, it contributes to the ongoing endeavor to integrate the concept of place into traditionally space-based GIS. The proposed computational model of functional place, which addresses explicitly the subjectivity in place perception with regards to the evaluation of spatial suitability as well as the individual composition of optimized places, can be interpreted as a step in that direction. Thus, the proposed approach to simulate human perceptual processes by translating objective spatial data to subjective meaning is of relevance for a range of current place-related computational challenges.

Hence, with our approach, a basic methodology is provided which can improve the analysis of human spatial behavior with GIS or ABM. On the one hand, the heterogeneity of human agents can be incorporated in much greater detail as in current studies, since the same objective, mathematical model of space can be interpreted in a different manner. This allows to model agents which differ in their capabilities and preferences, and which behave accordingly. The strengths of the approach have been demonstrated in the exemplary context of pedestrian simulation, it can, however, be expected that they persist when the model is transferred to other simulation scenarios or domains. On the other hand, the possibility to compute higher-level affordances which require the combinatory use of numerous spatial entities is innovative and represents a new way of modeling functional human-environment relations. Since these affordances are not pre-defined and allocated to spatial entities, but instead computed on-the-fly by an agent or another software program, it is possible to explore the provided functionality of space with much more creative flexibility, and identify solutions which are tailored to the specific situation.

Apart from the analysis of human spatial behavior, however, the notion to define and localize functional places based on the location of geo-atoms involved in a pragmatic action also addresses the challenge of localizing places on the basis of user queries or references, and is thus of relevance for spatial assistive or recommender services. Hence, a query such as "*Where can I go to sit in the shade and have a coffee?"*, to refer back to the example used in Chapter 1, could be answered in a similar manner. Thus, for instance, interpreting the involved *GO*, *SIT IN THE SHADE* and *HAVE A COFFEE* as semantic actions, and their combination as a pragmatic action, a pre-stored realization action model could be deployed in order to compute and return the most suitable place in the city which affords the pragmatic action. Other undefined places, such as *Downtown*, could be localized by identifying its pre-dominant affordances and computing a probable collective place. For this, the immediate perception of an agent is not of importance, but rather the whole area of interest would provide the input for the place calculation process.

Especially for pedestrian simulation, finally, the inclusion of a subjective notion of walkability is novel and, as empirical research on walking behavior implies, closer to reality than current simplified approaches. It provides new opportunities to simulate the behavior of different sub-types of pedestrians, for instance commuters, tourists, or mobility-impaired persons, and has the potential, if properly calibrated with data on real pedestrian observations, to assist planners in analyzing the pedestrian flows in their city. Thus, collective, important places for walking can be identified, and further analyzed for their perceived walkability, for instance in order to evaluate the sense of infrastructural improvements.

Despite its strengths, however, the approach presented in this thesis has also several limitations. Firstly, from the perspective of humanistic geography, it is clear that our restriction to the functional dimension represents a drastic simplification of the complexity of place. As it has been discussed, there can be various reasons for a place to be used in a certain way or neglected, ranging from collective phenomena such as social norms, territoriality or symbolic meanings to personal memories or even errors and gaps in individual spatial knowledge. Since, however, we clearly restrict our scope to functional places, and follow the current strategy of GISc or urban design and planning, this limitation is acceptable at the present stage, however, further work on these issues is still necessary. Moreover, functional places can certainly be stored in people's spatial memory, and do not require to be evaluated for their suitability prior to every use. In our current conceptual model, spatial memory of functional places has not yet been acknowledged. This, however, is a shortcoming which is to a certain degree inherited from the affordance theory, which has also been criticized for ignoring previous knowledge of the observing organism (e.g. Eco 1999). A related problem is our assumption of a homo economicus paradigm, which, although being a useful simplification, is not representative of the entirety of human behavior and decision making.

A practical problem for modeling functional places with our framework lies in the high level of detail which is required for modeling the environment, the agent and the action. While spatial or agent-related data might be difficult to obtain, the development of the action model with the inclusion of all possible feasible alternatives to reach the goal is certainly challenging, and must be based on profound knowledge of the particular domain. For the sake of feasibility, but also to reduce the high computational load, simplifications will be necessary, as they have been discussed with regards to our case study in Chapter 5. 1. 2. A related implementation-specific problem is the fact that a pre-computed suitability value needs to be passed to PlaceBuilder for the place formation process. In case of the *RA_OVERCOMESLOPE*, however, it would make sense to distinguish between a positive and a negative gradient. This, however, due to the fact that during the suitability calculation, it is not known whether the agent is walking up- or downhill, is currently not possible in our simulation model.

In addition, validating the model outputs at this high level of detail is difficult at best, which is due to the required data being not available in many cases, and time- and cost-intensive to be collected manually. Thus, without a proper calibration and validation, the results of our simulation are to be interpreted with care. This, as has been discussed in Chapter 3. 1. 4, is not a specific problem of our model, but a typical limitation of ABM in general. Moreover, without denying the importance of traditional means of model validation, the value of geosimulation in general and ABM in particular should not be tied too closely to such conceptual restrictions. Thus, for instance, since our model's potential of generating plausible agent behavior has been demonstrated, despite the lack of a full validation against real-world data, it can still be of high value for planners as a tool to explore potential implications of infrastructural alterations.

# 7 Conclusions and Future Work

This final chapter concludes the thesis and discusses potential directions for future work. It begins with summarizing the presented work and its motivations, and commences with a presentation of the results and major findings. Finally, possibilities for future work are provided.

## 7.1 Summary

This work followed the aim of developing a computational model of functional place, and simulating the process of its formation in order to dynamically generate it from static spatial data. For this, a functional place was conceptualized as a subset of geo-atoms, which are bound by a functional unity condition with regards to a complex, pragmatic spatial action. The model was implemented as part of a reasoner in an ABM of pedestrian movement.

The motivation for this study arose from several facts:

Firstly, based on the discrepancy between the model of space used in GIS and human conceptualizations of place, the necessity for methods to explicitly incorporate the subjectivity and compound structure of place into their computational representation was identified. The dynamic generation of functional places from spatial data was proposed as a potential approach to this problem.

Secondly, the process of human agents assessing the spatial suitability of places with regards to specific actions has a central meaning in place formation and perception. This motivated the development of a conceptual framework to describe, formalize and finally simulate these processes.

Thirdly, it was expected that an ABM paradigm would be well suited to modeling place formation due to its disaggregate approach as well as the possibility to model heterogeneous agents and their environment with a high level of detail. Pedestrian simulation was identified as a potential application domain which would particularly profit from these strengths.

Against this motivational background, the goals for this thesis were defined as follows:

- develop a simulation framework for individual spatial suitability assessment
- use it as a basis for creating a computational model of functional place as an action-related subset of geo-atoms
- implement and test this model on the example of an agent-based pedestrian simulation

In Chapter 2, a theoretical foundation for the concepts to be developed in this work was set. As a start, in Chapter 2. 1, the process of human perception of action potentials within their environment was approached from a psychological perspective, with a focus on Gibson's (1979) concept of affordances. It was discussed how, according to the principle of agent-environment mutuality, an affordance is determined by the correspondence of properties of the agent as well as of the environment. Further, the notion of higher-order affordances, which refer to more complex actions and are constituted of lower-level affordances, was discussed. In respect of the computational approach of this thesis, particular emphasis was placed on methods to formalize and measure affordances, as well as prior applications of the concept in GISc, ABM and related areas.

This was followed by the introduction of the phenomenological concept of place in Chapter 2. 2. Here, places were described as results of a mental structuring of space, in which complex meaning is attributed to locations with a vague, ambiguous spatial footprint and a physical setting. Due to our focus being put on functional places, the role of a place as action space was discussed in detail, and related to the affordance-concept. Furthermore, the importance of a place's suitability with regards to an action, as well as its spatial accessibility were presented as critical predictors of its potential usage frequency. A distinction was made between individual places as a result of mental place formation processes, and collective places with converging activity spaces and shared human meaning.

Then, in Chapter 2. 3, the perspective was shifted to GISc, when the explicit challenges for integrating place in GIS were addressed. The discrepancy between the spatial model generally used in GIS and the human way of structuring space into places was identified as a main barrier towards reaching this goal. Prior work was discussed regarding the topics of place representation and localization, with an emphasis on studies which were based on the affordance-concept as a potential approach to modeling places. In terms of place localization, methods to infer or predict unknown places by modeling human spatial behavior were presented, in particular accessibility analysis and suitability mapping.

In Chapter 3, the methodological basis of this research was set. Thus, in Chapter 3. 1, ABM was presented as a powerful tool for modeling human spatial behavior on a disaggregate, microscopic level, and with an explicit acknowledgement of the subjective, individual component of perception and action. Methods to model the cognitive structures, and action strategies of artificial agents were discussed, as well as the notion of utility-based agents.

Afterwards, in Chapter 3. 2, a brief recapitulation of established methods and approaches, but also shortcomings of the current practice in pedestrian simulation was presented, and empirical research on micro-scale walkability reviewed.

In Chapter 4, a conceptualization of functional place was developed on these theoretical and methodological foundations, and implemented as a computational model.

In Chapter 4. 1, as a first step, a concept to simulate the human mental process of spatial suitability assessment was presented based on an extended notion of the affordance-concept and a hierarchical action model. Then, this was integrated in a model of functional place as meaningful subset of geo-atoms which are bound by a functional unity condition and which can be dynamically extracted from a model of the environment. In Chapter 4. 2, these concepts were implemented in the form of a new Java-based extension package PlaceBuilder for the multi-agent simulation environment NetLogo. The structure and functionality of PlaceBuilder in combination with NetLogo was explained in detail.

In Chapter 5, in order to demonstrate its usefulness and practical value, an agent-based pedestrian simulation was presented as an exemplary application of the developed PlaceBuilder extension. Based on the empirical evidence on walkability presented in Chapter 3. 2. 2, a model of subjective walkability assessment was developed in accordance with our conceptual framework in Chapter 5. 1, and implemented in an agent-based pedestrian simulation in Chapter 5. 2.

Chapter 6 revisited the main aims and hypotheses of this thesis, and critically discussed them with regards to the derived results in Chapter 6. 1 and Chapter 6. 2. Moreover, in Chapter 6. 3, major implications of this work as well as the limitations of our approach were presented.

## 7.2 Results and Major Findings

This thesis produced a number of main results and major findings, which are briefly restated in this chapter.

A main outcome of this thesis is the conceptualization of spatial suitability as an abstract quality of the system of agent, geo-atom and action, which depends on the correspondence of agent- and environment-related properties with regards to a particular action. This premise describes suitability as something abstract but still measurable by means of the relative value of its ratio of disposition and capability with regards to the optimal and critical threshold values of the particular system. Ecological psychology, in particular Gibson's (1979) affordance theory, has served as a conceptual foundation for this approach towards modeling spatial suitability. The notion of graded suitability values has been based on the work by Warren (1984), who described an affordance as a ratio $\pi$ of an environmental and agent-related property. Further, the author demonstrated that there is a range of values between the two poles $\pi_O$ and $\pi_{max}$ or $\pi_{min}$ at which the action is afforded. Consequently, it takes only a small conceptual step to deduce from that the existence of graded affordances, or suitability. Naturally, the assumption of a linear dependency relationship between $\pi$- and suitability values is worth discussion, which is why the sensitivity of the model to a non-linear mapping function has been tested in Chapter 5. 2. 3. 2. Other than in the original formalization by Stoffregen (2003), the action was included as part of the system, which is due to its role as a semantic connector between agent and the environment, in our model represented by geo-atoms as primitives, as described by Goodchild et al. (2007). We proposed a computational representation of the system as a distinct, dynamically constructed object which has an attribute *affordances* which makes the connection to a *RealizationAction* instance. This approach towards modeling affordances is novel and helps to avoid the necessity to hard-code an affordance as an attribute of an environmental object, which would contradict its agent-relatedness and dynamic character. The usefulness of this conceptualization of affordances and suitability could be demonstrated by the practical application in the context of a pedestrian simulation. Thus, it was possible to identify relevant agent- and environment-related attributes, determine useable threshold values $\pi_O$ and $\pi_{max}$ or $\pi_{min}$ as well as define dependency functions based on the empirical literature on walkability, which indicates the feasibility of our approach. Further, suitability values could be computed which were specific to the agents and their respective environmental setting, and which led to plausible agent behavior.

In order to incorporate complex actions into our model, we further hypothesized that in order to assess the suitability provided by a place for reaching a desired goal state, an agent needs to evaluate its suitability with regards to the contributing actions. Since in the past, affordances have been used for modeling place in GIS, we deploy our extended affordance-based notion of suitability for the same purpose. Due to the importance of the perceived suitability for place formation, as presented in Chapter 2. 2, this approach is legitimate. The view of complex actions to be composed of hierarchically structured contributing sub-actions, and their representation as a graph structure, is a common approach in AI but can also be grounded in psychological theories such as activity theory (Leontiev 1978). The approach presented by Kemke (2001), who distinguishes between different levels of abstractions when referring to actions, represents a useful way to model complex actions and cope with the action individuation problem. On the example of the real world action *WALK*, it has been demonstrated that it is feasible and useful to model a complex action according to our framework, although a high level of domain knowledge is necessary.

The conceptual framework for spatial suitability assessment presented in this thesis is based on its conceptual closeness to Gibson's (1979) affordance concept, as well as the transfer and extension of Warren's (1984) way of measuring affordances. In combination with activity theory, a novel framework for modeling complex actions and evaluating their suitability has been presented which allows for the computation of suitability values which are tailored to the individual pair of agent and geo-atom. While building on established methods for GIS-based suitability mapping and utility-based agents, these concepts were substantially extended by incorporating the subjectivity of agent perception and by modeling the action as a complex construct to be performed by differing behavioral strategies. As a result, our framework facilitates the simulation of the semantic interpretation of objective attributes of the environment to receive subjective evaluations of suitability. It might therefore serve as a basis for the development of personalized recommender and other spatial assistive systems, as well as provide new methods for human spatial behavior modeling.

Similar to previous work by Kuhn (2001) or Scheider and Janowicz (2014), we used affordances to create a semantic connection between the action and the environment, and accordingly hypothesized that a functional place which affords reaching a desired goal state consists of the geo-atoms which afford the contributing sub-actions with the highest possible suitability. This conceptualization of place, as well as the focus on its functional dimension, was justified with the central importance of spatial actions for the perception of places in general, and represents a feasible approach to cope with its compound nature. Instead of referring to higher-order geo-objects such as buildings or streets as constituents of places, they are constructed from primitive geo-atoms. This is done in order to increase the conceptual soundness of our model, as well as ease its potential application to other domains and scenarios. Furthermore, this notion of compound functional places can be paralleled to Gibson's (1979) concept of higher-order affordances, which result from the extraction of compound invariants from the stimulus flux, and relate to multiple properties of the environment. Thus, representing places as dynamically constructed, functional units of the environment is closer to human spatial conceptualizations than pre-defining them as objective, distinct geo-objects. The assumption of a homo economicus paradigm and utility maximization strategies is worth discussion, but still a useful simplification used for modeling human behavior.

Against this conceptual background, we finally hypothesized that the generated places are sensitive to the needs and characteristics of the individual agent. In accordance with the notion of agent-environment mutuality, differing characteristics or preferences of agents should therefore result in variations concerning their specific functional places. Thus, such dissimilarities would result in different capabilities with regards to the action in question, which, in turn, lead to altered suitability values to be computed, and, finally, variations in terms of either the chosen realization or semantic actions, or the set of geo-atoms used to conduct the pragmatic action, the functional place. In fact, testing the model sensitivity indicated the intended variations in agent behavior which were plausible rather than random.

The approach presented in this thesis explicitly addressed the challenge of places to be subjectively perceived and of compound nature. In order to incorporate the first aspect of place, a conceptual model of the process of individual spatial suitability assessment was developed, which was then, in order to cope with the second place characteristic, incorporated into a larger computational model of functional place as a subset of geo-atoms bound by a functional unity condition with regards to a complex, high-level action. As our results have demonstrated, this approach allows for the construction of functional spatial units without explicitly keeping them as entities or their affordances as attributes. As intended, these functional places are plausible, sensible to the individual agent, functional since they afford a complex action, and spatial in terms of them being composed of geo-atoms. As expected, an ABM paradigm has proven useful for modeling place due to its disaggregate, microscopic approach and the possibility to create heterogeneous agents and a detailed environmental model.

To conclude, this research has proposed to bridge the gap between the objective, mathematical spatial model used in GIS and the way humans conceptualize the world by dynamically computing the subjective reality of each individual person on the basis of spatial data, or, to refer back to the introductory citation by Golledge (1981), to construct different geographies instead of enforcing one which is assumed to be universal. For this, it is needed to derive conceptual models which describe the involved perceptual and cognitive processes, and which can be formalized and computationally implemented. Thus, for instance, whereas the interaction with the environment has been allocated central meaning for human apprehension of space, it plays only a minor role in the current, object-biased way of representing geographical information (Kuhn 2001). As has been discussed in this thesis, however, actions require alternative forms of computational representation. In GISc, there is still a need for research which improves its compatibility with human spatial concepts.

## 7.3 Future Work

In this chapter, possible directions for future work are presented. A first step would be to extend the model by taking a more holistic viewpoint on place, thus, in contrast to the present restriction on functional places, to incorporate other aspects of place. Thus, for instance, different levels of spatial knowledge could be incorporated by changing the beliefs an agent has about its environment, thereby intentionally distorting or reducing the agent's spatial knowledge. Further, personal preferences or aversions towards certain locations could also be introduced, as well as socially-created place-related meaning which, for instance, influences behavioral norms. Also, at the present stage, there is no acknowledgement of a agents memorizing places and accessing this memory at a later stage. An extension of the model in that direction can be expected to improve its realism.

Another potential direction of future work would be to adapt the implementation of the conceptual model to serve as a basis for a spatial assistant system, such as an activity planner or a routing device, which aims at providing highly personalized recommendations with regards to a user. As described in Chapter 6. 3, based on a detailed user and action model, such a system could compute individual suitability values and generate optimal functional places in response to user queries. Due to the high level of detail necessary with regards to the user model, a learning system could be worthwhile, which continuously observes the user's behavior and refines its internal model.

Another possible step would be to improve the pedestrian simulation model, for instance by calibration with real world data, for instance pedestrian trajectories. Thus, by constantly comparing the output with the actual pedestrian movement patterns, one could adjust the model parameters and identify a best fit. Further, specifically targeted empirical research on the different sub-actions of *WALK*, and their corresponding properties could assist in further improving the validity of the pedestrian model. Also, the wayfinding algorithm could be extended to incorporate walkability as early as in the shortest path analysis on the basis of the visibility graph. Thus, one could model the effect that a pedestrian agent avoids a particular sidewalk segment, which somehow poses a barrier to it, from the very beginning of the path planning process. Also, the outputs of our model could be compared to other approaches for pedestrian simulation to determine the exact magnitude of differences.

Apart from pedestrian simulation, it would be interesting to apply PlaceBuilder to different problem domains and study areas of different scale, such as residential selection in urban areas. It would be interesting to examine whether the required level of detail of the input data as well as the models for agent, environment and action, allow feasible modeling in application domains which are less well researched than walking and walkability. In principle, although we have restricted our simulation to only one single agent at a time, the framework allows for the inclusion of numerous agents acting simultaneously in the same environment. Moreover, since the places are described in terms of agent-environment-action systems, it would in general be possible to model cooperative behavior of agents with differing capabilities to reach a common goal state. Furthermore, since, at present, we have focused on static environments, it would be interesting to introduce agents acting in dynamically changing surroundings, which would require a constant re-evaluating and -planning of the action strategy.

# References

Abdelmoty AI, Smart P, Jones CB (2007) Building place ontologies for the semantic web: Issues and approaches. In: Proceedings of the workshop on geographic information retrieval (GIR'07), CIKM 2007, Lisboa

Abdou M, Hamill L, Gilbert N (2012) Designing and building an agent-based model. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 141 - 166

Ackermann K, Bartz C, Feller G (1997) Behindertengerechte Verkehrsanlagen, Planungshandbuch für Architekten und Ingenieure, Werner, Neuwied

Adkins A, Dill J, Luhr G, Neal M (2012) Unpacking walkability: Testing the influence of urban design features on perceptions of walking environment attractiveness. Journal of Urban Design 17(4):499–510

Agrawal AW, Schlossberg M, Irvin K (2008) How far, by which route and why? A spatial analysis of pedestrian preference. Journal of Urban Design 13(1):81–98

Aissi S, Gouider MS (2012) Personalization in geographic information systems: a survey. International Journal of Computer Science Issues 9(4): 291-298

Alani H, Jones CB, Tudhope D (2001) Voronoi-based region approximation for geographical information retrieval with gazetteers. International Journal of Geographical Information Science 15(4):287–306

Alazzawi AN, Abdelmoty AI, Jones CB (2012) What can I do there? Towards the automatic discovery of place-related services and activities. International Journal of Geographical Information Science 26(2):345–364

Alfonzo MA (2005) To walk or not to walk? The hierarchy of walking needs. Environment and Behavior 37(6):808–836

Allgemeiner Deutscher Fahrrad-Club e.V. ADFC (ed) (2011) Fachwissen für den Fahrradalltag – Seitliche Sicherheitsabsstände. ADFC

Altman D (1994) Research Article. Fuzzy set theoretic approaches for handling imprecision in spatial analysis. International journal of geographical information systems 8(3):271–289

Alvares LO, Bogorny V, Kuijpers B, Moelans B, Macedo J de, Palma A (2007) Towards semantic trajectory knowledge discovery. Technical Report, Hasselt University

Andrienko G, Andrienko N, Fuchs G, Raimond AMO, Symanzik J, Ziemlicki C (2013) Extracting semantics of individual places from movement data by analyzing temporal patterns of visits. ACM SIGSPATIAL COMP'2013, Orlando

Antonini G, Bierlaireb M, Weberb M (2006) Discrete choice models of pedestrian walking behavior. Transportation Research Part B 38 40(8):667–687

Arampatzis A, van Kreveld M, Reinbacher I, Jones CB, Vaid S, Clough P, Joho H, Sanderson M (2006) Web-based delineation of imprecise regions. Computers, Environment and Urban Systems 30(4):436–459

Australian Pedestrian Council (2015) The Pedestrian Council: Pedestrian Council Home. http://www.walk.com.au

Axtell RL, Epstein JM, Dean JS, Gumerman GJ, Swedlund AC, Harburger J, Chakravarty S, Hammond R, Parker J, Parker M (2002) Population growth and collapse in a multiagent model of the Kayenta Anasazi in Long House Valley. Proceedings of the National Academy of Sciences 99(Supplement 3):7275–7279

Badland H, Schofield G (2005) Transport, urban design, and physical activity: an evidence-based update. Transportation Research Part D(10):177–196

Balci O (1998) Verification, validation and accreditation of simulation models. In: Proceedings of the 1998 Winter Simulation Conference, Washington DC

Barker RG (1968) Ecological psychology: Concepts and methods for studying the environment of human behavior. Stanford University Press, Stanford

Barsalou LW, Sloman SA, Chaigneau SE (2005) The HIPE theory of function. In: Carlson L, van der Zee E (eds) Representing functional features for language and space: Insights from perception, categorization and development, Oxford University Press, Oxford

Basbas S, Konstantinidou C, Ribas DJM (2010) Factors and mechanisms which determine the outcome of strategic decisions with regard to walking. In: COST 358 Pedestrian Quality Needs Final Report, Part B, p 127-156

Batty M (2001) Agent-based pedestrian modeling. Environment and Planning B: Planning and Design 28(3):321-326

Batty M (2005) Cities and complexity - understanding cities with cellular automata, agent-based models, and fractals. The MIT Press, Cambridge, London

Batty M (2009) Accessibility: in search of a unified theory. Environment and Planning B: Planning and Design 36(2):191-194

Batty M, Crooks AT, See LM, Heppenstall AJ (2012) Perspectives on agent-based models and geographical systems. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 1–18

Batty M, Desyllas J, Duxbury E (2003) Safety in numbers? Modelling crowds and designing control for the Notting Hill Carnival. Urban Studies 40(8):1573–1590

Baumann AE, Bull F (2007) Environmental correlates of physical activity and walking in adults and children: A review of reviews. National Institute of Health and Clinical Excellence

Beck R (1967) Spatial meaning, and the properties of the environment. In: Lowenthal D (ed) Environmental perception and behavior. University of Chicago, Chicago, p 18-41

Benenson I, Torrens PM (2005) Geosimulation. Automata-based modeling of urban phenomena, Wiley, Chichester

Bernhardt K (2007) Agent-based modeling in transportation. Artificial Intelligence in Transportation EC113:72–80

Bittner T (1999) On ontology and epistemology of rough location. In: Goos G, Hartmanis J, van Leeuwen J, Freksa C, Mark DM (eds) Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, p 433–448

Bittner T, Stell JG (2002) Vagueness and rough location. Geoinformatica 6(2): 99-121

Boesch H (1992) Die Langsamverkehrs-Stadt. Bedeutung, Attraktion und Akzeptanz der Fussgängeranlagen - Eine Systemanalyse. ARF Arbeitsgemeinschaft Recht für Fußgänger, Zürich

Bönisch C, Kretz T (2009) Simulation of pedestrians crossing a street. Proceedings traffic and granular flow 09, Shanghai

Borst HC, Miedema HM, Vries SI de, Graham JM, van Dongen JE (2008) Relationships between street characteristics and perceived attractiveness for walking reported by elderly people. Journal of Environmental Psychology 28(4):353–361

Bratman M (1987) Intention, plans, and practical reason. Harvard University Press, Cambridge

Brookes CJ (1997) A parameterized region-growing programme for site allocation on raster suitability maps. International Journal of Geographical Information Science 11(4):375–396

Brown BB, Szalay C (2007) Walkable enroute perceptions and physical features converging evidence for en route walking experiences. Environment and Behavior 39:34–61

Brownson RC, Hoehner CM, Day K, Forsyth A, Sallis JF (2009) Measuring the built environment for physical activity: State of the science. American Journal of Preventive Medicine 36(4 suppl):99–123

Buchmueller S, Weidmann U (2006) Parameters of pedestrians, pedestrian traffic and walking facilities. IVT-Report Nr. 132, Institute for Transport Planning and Systems, ETH Zurich

Canter D (1977) The psychology of place. Architectural Press, London

Carello C, Grosofsky A, Reichel F, Solomon HY, Turvey MT (1989) Visually perceiving what is reachable. Ecological Psychology 1(1):27-54

Carmona M, Tiesdell S, Heath T, Oc T (2010) Public places - urban spaces. The dimensions of urban design. Second Edition. Routledge, New York

Carr S, Francis M, Rivlin LG, Stone AM (1992) Public space. Cambridge series in environment and behavior. Cambridge University Press, Cambridge, New York

Casati R, Varzi AC (1995) The structure of spatial localization. Philosophical Studies 82:205-239

Cervero R (1993) Transit-supportive development in the United States: Experiences and prospects. Final Report, Federal Transit Administration, Washington DC

Cervero R, Kockelmann K (1997) Travel demand and the 3Ds: Density, diversity, and design. Transpn Res.-D. 2(3):199-219

Charkirov A, Erath A (2012) Activity identification and primary location modeling based on smart card payment data. In: Proceedings of the 13th International conference on travel behavior research, Toronto.

Chemero A (2003) An outline of a theory of affordances. Ecological Psychology 15(2): 181-195

Chen M, Bärwolff G, Schwandt H (2009) Automation model with variable cell size for the simulation of pedestrian flow. In: Proceedings of the 7th international conference on information and management sciences 2008, Urumtschi, p 727-736

Chenney S (2004) Flow tiles. In: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on computer animation, Grenoble, p 233–242

Chu C (2011) Adaptive guidance for emergency evacuation for complex building geometries. In: Zhu Y, Issa RR (eds) Proceedings computing in civil engineering, Miami, p 604–610

Clark S, Davis A (2009) Identifying and prioritizing walking investment through the PERS audit tool. In: Proceedings Walk 21 Conference 2009, New York

Clifton KJ, Livi AD (2004) Gender differences in walking behavior, attitudes about walking, and perceptions of the environment in three Maryland communities. In: Transportation research board conference proceedings 35, Conference on research on women's issues in transportation, Chicago, p 79–88

Couclelis H (1992) Location, place, region, and space. In: Abler RF, Marcus MG , Olson JM (eds) Geography's inner worlds, Rutgers, USA, p 215–233

Couclelis H (1999) Space, time, geography. In: Longley PA, Goodchild MF, Maguire DJ, Rhind DW (eds) Geographical information systems, Wiley, Hoboken, p 29–38

Cresswell T (2009) Place. In: Kitchin R, Thrift N (eds) International encyclopedia of human geography, Elsevier, p 169–177

Crooks AT, Castle CJE, Batty M (2007) Key challenges in agent-based modelling for geo-spatial simulation. CASA Working Paper 121

Crooks AT, Heppenstall AJ (2012) Introduction to agent-based modelling. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 85–105

Cunningham GO, Michael YL (2004) Concepts guiding the study of the impact of the built environment on physical activity for older adults: a review of the literature. Am J Health Promot 18(6):435–443

Curry MR (1996) The work in the world, Geographical practice and the written word. University of Minnesota Press, Minneapolis

Davison K, Lawson CT (2006) Do attributes in the physical environment influence children's physical activity? A review of the literature. Int J Behav Nutr Phys Act 3(19)

de Berg M, van Krefeld M, Overmars M, Schwarzkopf O (1997) Computational geometry. Springer, Berlin

Dungan JL, Perry JN, Dale MRT, Legendre P, Citron-Pousty S, Fortin M, Jakomulska A, Miriti M, Rosenberg MS (2002) A balanced view of scale in spatial statistical analysis. Ecography 25(5):626–640

Durupinar F, Allbeck J, Pelechano N, Badler N (2008) Creating crowd variation with the OCEAN personality model. In: Proc. of 7th Int. Conf. on autonomous agents and multi-agent systems (AAMAS 2008), Estoril

Eastman JR (1999) Multi-criteria evaluation and GIS. In: Longley PA, Goodchild MF, Maguire DJ, Rhind DW (eds) Geographical information systems, Wiley, Hoboken, p 493-502

Eco U (1999) Kant and the Platypus - Essays on Language and Cognition. English translation ed., San Diego, Harvest Book Harcourt, New York

Edwardes AJ (2007) Re-placing location: Geographic perspectives in location based services. Dissertation, University of Zurich

Egenhofer MJ, Mark DM (1995) Naïve geography. In: Frank A, Kuhn W (eds) Lecture notes in computer science, Vol. 988, Springer, Berlin

ESRI (2012) ArcGIS Desktop: Release 10.2. Environmental Systems Research Institute, Redlands, CA

Evans AJ (2012) Uncertainty and error. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 309 - 346

Ewert U (2012) Senioren als Fußgänger. BFU Faktenblatt Nr. 08, BFU – Beratungsstelle für Unfallverhütung, Bern

Ewing R, Cervero R (2001) Travel and the built environment. Transportation research record (1780):87–114

Ewing R, Handy S, Brownson RC, Clemente O, Winston E (2006) Identifying and measuring urban design qualities related to walkability. Journal of Physical Activity and Health(3):223–240

Fi I, Igazvölgyi ZK (2014) Travel time delay at pedestrian crossings based on microsimulations. Civil Engineering 58(1):47-53

Forschungsgesellschaft für Straßen- und Verkehrswesen FGSV (ed) (2002) EFA: Empfehlungen für Fußverkehrsanlagen EFA (R 2), FGSV Verlag, Köln

Forschungsgesellschaft für Straßen- und Verkehrswesen FGSV (ed) (2011) H BVA Hinweise für barrierefreie Verkehrsanlagen, FGSV Verlag, Köln

Forsyth A, Hearst M, Oakes JM, Schmitz KH (2008) Design and destinations: factors influencing walking and total physical activity. Urban Stud 45: 1973-1996

Frank A, Raubal M (1999) Formal specifications of image schemata - A step to interoperability on geographic information systems. Spatial Cognition and Computation 1(1):67-101

Frank LD, Engelke PO (2001) The built environment and human activity patterns: Exploring the impacts of urban form on public health. Journal of Planning Literature 16(2):202–218

Gaisbauer C, Frank A (2008) Wayfinding model for pedestrian navigation. In: Bernard L, Friis-Christensen E, Pundt H (eds) The European information society. Taking geoinformation science one step further. Springer, Berlin

Gao S, Janowicz K, McKenzie G, Li L (2013) Towards platial joins and buffers in place-based GIS. In: ACM SIGSPATIAL COMP'13, Orlando

Gaver W (1991) Technology affordances. In: Robertson SP, Olson GM, Olson JS (eds) Proceedings of the SIGCHI conference on human factors in computing systems: Reaching through technology, New Orleans, p 79–84

Gebel K, Bauman AE, Petticrew M (2007) The physical environment and physical activity: a critical appraisal of review articles. Am J Prev Med 32(5):361–369

Geurs KT (2014) Recent advances and applications in accessibility modelling. Computers, Environment and Urban Systems 49:82-85

Gibson EJ, Walk R (1960) The "visual cliff". Scientific American 202:67-71

Gibson J (1979) The ecological approach to visual perception. Houghton Mifflin Company, Boston

Gibson JJ (1947) Motion picture testing and research. Army Air Force Aviation Psychology Program Research Rep. No. 7, U.S. Government Printing Office, Washington, DC

Gibson JJ (1977) The theory of affordances. In Shaw RE, Bransford J (eds) Perceiving, acting, and knowing: Toward an ecological psychology, Lawrence Erlbaum Associates, Hillsdale, p 67–82

Gibson JJ, Crooks LE (1938) A theoretical field-analysis of automobile-driving. American Journal of Psychology 51:453–471

Gloor C, Mauron L, Nagel K (2003) A pedestrian simulation for hiking in the alps. In: Proceedings STRC 3rd Swiss Transport Research Conference 2003, Ascona

Gloor C, Stucki P, Nagel K (2004) Hybrid techniques for pedestrian simulations. In: Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, Sloot PMA, Chopard B, Hoekstra AG (eds) Lecture notes in computer science. Springer, Berlin, Heidelberg, p 581–590

Golledge RG (1981) Misconceptions, misinterpretations, and misrepresentations of behavioral approaches in human geography. Environment and Planning A 13: 1325-1344

Golledge RG, Stimson RJ (1997) Spatial behavior: A geographic perspective. The Guildford Press, New York, London

Goodchild MF (2011) Formalizing place in geographic information systems. In: Burton LM, Matthews SA, Leung M, Kemp SP, Takeuchi DT (eds) Communities, neighborhoods, and health. Springer, New York, p 21–33

Goodchild MF, Yuan M, Cova TJ (2007) Towards a general theory of geographic representation in GIS. International Journal of Geographical Information Science 21(3):239-260

Graumann CF (2002) The phenomenological approach to people-environment studies. In: Bechtel RB, Churchman A (eds) Handbook of environmental psychology, John Wiley and Sons, New York, p 95-113

Greeno JG (1995) Gibson’s Affordances. Psychological Review 101(2): 336-342

Gregory D, Johnston R, Pratt G, Watts M, Whatmore S (eds) (2011) The dictionary of human geography, 5th edn., John Wiley & Sons, New York

Guarino N, Welty C (2000) Identity, unity and individuality: Towards a formal toolkit for ontological analysis. In: Horn W (ed) The European conference on artificial intelligence, IOS Press, Amsterdam.

Hakkert AS (2010) Pedestrian risk and risk factors. In: COST 358 Pedestrian quality needs final report, Part B, p 55-68

Haklay M, O'Sullivan D, Thurstain-Goodwin M, Schelhorn T (2001) "So go downtown": simulating pedestrian movement in town centres. Environment and Planning B: Planning and Design 28(3):343-359

Hamby DM (1995) A comparison of sensitivity analysis techniques. Health Phys. 68(2):195-204

Handy S (2005) Does the built environment influence physical activity? TRB special report 282, Transportation Research Board, Washington DC

Handy S, Boarnet MG, Ewing R, Killingsworth RE (2002) How the built environment affects physical activity: Views from urban planning. American Journal of Preventive Medicine 23(2):64–73

Handy SL (2004) Critical assessment of the literature on the relationship among transportation, land use, and physical activity. Report, Transportation Research Board, Washington DC

Hart R, Conn M (1991) Developmental perspectives on decision making and action in environments. In: Garling T, Evans G (eds) Environment, cognition and action: An integrated approach, Oxford University Press, Oxford, p 277–294

Heft H (2001) Ecological psychology in context: James Gibson, Roger Barker, and the legacy of William James's radical empiricism. Erlbaum, NJ

Heidegger M (1993) Building, dwelling, thinking. In: Krell DF (ed) Martin Heidegger: Basic writings, HarperCollins Publishers, New York

Helbing D, Molnár P (1995) Social force model for pedestrian dynamics. Phys. Rev. E 51(5):4282–4286

Helbing D, Molnár P, Farkas IJ, Bolay K (2001) Self-organizing pedestrian movement. Environ. Plann. B 28(3):361–383

Heller J, Monheim R (2004) Die Augsburger Innenstadt im Bild von Verhalten und Einstellungen der Besucher und Bevölkerung. Projektbericht, Omniphon mbh, Leipzig

Henderson LF (1971) The statistics of crowd fluids. Nature 229(5284):381–383

Heppenstall AJ, Evans AJ, Birkin MH (2007) Genetic algorithm optimisation of an agent-based model for simulating a retail market. Environ. Plann. B 34(6):1051–1070

Herzog TR, Shier RL (2000) Complexity, age and building preference. Environment and Behavior 32(4):557-575

Hess S, Quddus M, Rieser-Schüssler N, Daly A (2015) Developing advanced route choice models for heavy goods vehicles using GPS data. Transportation Research Part E: Logistics and Transportation Review 77:29–44

Hidalgo CM, Berto R, Galindo MP, Getrevi A (2006) Identifying attractive and unattractive urban places: categories, restorativeness and aesthetic attributes. Medio Ambiente y Comportamiento Humano 7 (2):115-133

Hill LL (2000) Core elements of digital gazetteers: Placenames, categories, and footprints. In: Research and advanced technology for digital libraries, LNCS 1923, p 280-290

Hillier B (1996) Space is the machine. Cambridge University Press, Cambridge

Hillier B, Hanson J (1984) The social logic of space. Cambridge University Press, Cambridge

Hodgson FC, Page MW, Tight MR (2004) A review of factors which influence pedestrian use of the streets. Task 1 report for an EPSRC funded project on measuring pedestrian accessibility, institute of transport studies, University of Leeds, Working Paper 581

Hoffman FO, Gardner RH (1983) Evaluation of uncertainties in environmental radiological assessment models: A textbook on environmental dose assessment. US Nuclear Commission, Washington D.C., p 11.1 – 11.55

Hollenstein L, Purves RS (2010) Exploring place through user-generated content: Using Flickr tags to describe city cores. JOSIS 1: 21-48

Hoogendoorn SP, Bovy PHL (2004) Pedestrian route-choice and activity scheduling theory and models. Transportation Research Part B 38:169–190

Horton FE, Reynolds DR (1971) Effects of urban spatial structure on individual behavior. Economic Geography 47(1):36–48

Humpel N, Owen N, Leslie E (2002) Environmental factors associated with adults' participation in physical activity. American Journal of Preventive Medicine 22(3):188–199

Iltanen S (2012) Cellular automata in urban spatial modeling. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 69 - 84

Ingram DR (1971) The concept of accessibility: A search for an operational form. Regional Studies 5(2):101-107

Janiak A (2012) Kant's views on space and time. In: Zalta EN (ed) The Stanford encyclopedia of philosophy, The Metaphysics Research Lab, Stanford University

Janowicz K, Raubal M (2007) Affordance-based similarity measurement for entity types. In: Winter S, Duckham M, Kulik LKB (eds) Spatial information theory, 8th International Conference, COSIT 2007, Melbourne, Australia, September 19-23, 2007, Proceedings, p 133- 151

Janowicz K, Scheider S, Pehle T, Hart G (2012) Geospatial semantics and linked spatiotemporal data - Past, present, and future. Semantic Web – Interoperability, Usability, Applicability 0(1):1-13

Jenkins HS (2008) Gibson's "Affordances": Evolution of a pivotal concept. Journal of Scientific Psychology 12:34-45

Johansen TK (2004) Plato's natural philosophy. A study of the Timaeus-Critias. Cambridge University Press, Cambridge

Johansson A, Kretz T (2012) Applied Pedestrian Modeling In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 451 - 462

Johnson M (1987) The body in the mind. The bodily basis of meaning, imagination, and reason. University of Chicago Press, Chicago

Jones KS (2003) What is an affordance? Ecological Psychology 15(2):107–114

Jonietz D, Rathmann J (2013) Entwicklung einer Methodik zur GIS-gestützten Analyse therapeutischer Landschaften. In: Strobl J, Blaschke T, Griesebner G, Zagel B (eds) Angewandte Geoinformatik 2013 - Beiträge zum 25. AGIT-Symposium Salzburg, Berlin und Offenbach, p 600 - 609

Jonietz D, Schuster W, Timpf S (2013) Modelling the suitability of urban networks for pedestrians: An affordance-based framework. In: Vandenbroucke, D., Bucher, B., Crompvoets, J. [Eds.]: Geographic information science at the heart of Europe, Lecture Notes in Geoinformation and Cartography, Springer, Heidelberg, p 369 - 382

Jonietz D, Timpf S (2012) Entwicklung eines GIS-basierten Bikeablity-Index am Beispiel der Stadt Augsburg. In: Löwner MO, Hillen F, Wohlfahrt R (eds): Geoinformatik 2012: Mobilität und Umwelt. Konferenzband 28.-30. März 2012 Braunschweig, Aachen, p 83 – 90

Jonietz D, Timpf S (2013a) An affordance-based simulation framework for assessing spatial suitability. In: Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, Tenbrink T, Stell J, Galton A, Wood Z (eds) Lecture notes in computer science. Springer International Publishing, Cham, p 169–184

Jonietz D, Timpf S (2013b) Incorporating the influence of walkability into a model of pedestrian accessibility. In: Wulfhorst G, Büttner B (eds) Transportation demand management – insights from the mobil.TUM 2012 international scientific conference on mobility and transport. TUM Schriftenreihe Heft 2, p 31-43

Jonietz D, Timpf S (2015) On the relevance of Gibson’s affordances for geographical information science (GISc). Cognitive Processing 16(1 Suppl.): p 265-269

Jordan T, Raubal M, Gartrell B, Egenhofer MJ (1998) An affordance-based model of place in GIS. In: Poiker T, Chrisman N (eds) 8th Int. Symposium on spatial data handling (SDH'98), IUG, Vancouver, p 98- 109

Kadali BR, Vedagiri P (2013) Effect of vehicular lanes on pedestrian gap acceptance behavior. Procedia – Social and behavioral sciences 104:678-687

Kapadia M, Singh S, Hewlett W, Faloutsos P (2009) Egocentric affordance fields in pedestrian steering. In: Haines E, McGuire M, Aliaga AG, Oliveira MM (eds) I3D '09 Proceedings of the 2009 symposium on interactive 3D graphics and games, Boston, p 215–223

Kaplan R, Kaplan S (1989) The experience of nature: A psychological perspective. Cambridge university press, New York

Kaufmann C, Papaioannou P, Blaszczyk M, Marques Almeida D de (2010) Preconditions and how they are perceived. In: COST 358 Pedestrian quality needs final report, Part B, p 15–48

Kelly CE, Tight MR, Page MW, Hodgson FC (2007) Techniques for assessing the walkability of the pedestrian environment. In: Walk 21, 8th International conference on walking and liveable communities, Toronto, 2007

Kemke C (2001) About the ontology of actions. Technical Report MCCS-01-328, Computing Research Laboratory, New Mexico State University

Kennedy WG (2012) Modeling human behavior in agent-based models. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 167–179

Kerridge J, Hine J, Wigan M (2001) Agent-based modeling of pedestrian movements: the questions that need to be asked and answered. Environment and Planning B: Planning and Design 28(3):327-341

Keßler C, Maué P, Heuer JT, Bartoschek T (2009) Bottom-up gazetteers: Learning from the implicit semantics of geotags. In: Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, Janowicz K, Raubal M, Levashkin S (eds) Lecture notes in computer science. Springer, Berlin, Heidelberg, p 83–102

Khan A, Vasardani M, Winter S (2013) Extracting spatial information from place descriptions. In: Scheider S, Adams B, Janowicz K, Vasardani M, Winter S (eds) COMP 2013. ACM SIGSPATIAL International workshop on computational models of place: November 5, 2013, Orlando

Kienast F, Degenhardt B, Weilenmann B, Wäger Y, Buchecker M (2012) GIS-assisted mapping of landscape suitability for nearby recreation. Landscape and Urban Planning 105(4):385–399

Kim N, Joo J, Rothrock L, Wysk RA, Son Y (2011) Human behavioral simulation using affordance-based agent model. In: HCII'11 Proceedings of the 14th international conference on human-computer interaction: design and development approaches, Orlando, p 368–377

Kitazawa K, Batty M (2010) Pedestrian behaviour modelling. An application to retail movements using a genetic algorithm. In: Proceedings of the 7th international conference on design and decision support systems in architecture and urban planning, Nijkerk

Kitazawa K, Fujiyama T (2010) Pedestrian vision and collision avoidance behavior: Investigation of the information process space of pedestrians using an eye tracker. In: Klingsch WWF, Rogsch C, Schadschneider A, Schreckenberg M (eds) Pedestrian and evacuation dynamics 2008. Springer, Berlin, Heidelberg, p 95–108

Klügl F (2015) Affordance-based interaction design for agent-based simulation models. In: Bulling N (ed) EUMAS 2014, Lecture notes in artificial intelligence, Springer, Berlin, Heidelberg, p 51-66

Klügl F, Klubertanz G, Rindsfüser G (2009) Agent-based pedestrian simulation of train evacuation integrating environmental data. In: Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, Mertsching B, Hund M, Aziz Z (eds) Lecture notes in computer science. Springer, Berlin, Heidelberg, p 631–638

Klügl F, Rindsfüser G (2007) Large-scale agent-based pedestrian simulation. In: Petta P, Müller J, Klusch M, Georgeff M (eds) Multiagent system technologies, MATES 2007, Lecture notes in artificial intelligence, Springer, Berlin, Heidelberg, p 145-156

Koffka K (1935) Principles of gestalt psychology. Harcourt Brace, New York

Kollman K, Miller JH, Page SE (1992) Adaptive parties in spatial elections. The American Political Science Review 86(4):929

Kremer D, Schlieder C (2012) Identifying touristic places. In: Vasardani M, Winter S, Richter KF, Janowicz K, Mackaness W (eds) Proceedings of the international workshop on place-related knowledge acquisition research (P-KAR) 2012, Monastery Seeon, Germany, August 31, 2012

Kretz T, Bönisch C, Vortisch P (2010) Comparison of various methods for the calculation of the distance potential field. In: Klingsch WWF, Rogsch C, Schadschneider A, Schreckenberg M (eds) Pedestrian and evacuation dynamics 2008, Springer, Berlin, Heidelberg, p 335–346

Ksontini F, Mandiau R, Guessoum Z, Espie S (2015) Affordance-based agent model for road traffic simulation. Autonomous Agents and Multi-Agent Systems 29(5):821-849

Kuhn W (1996) Handling data spatially: Spatializing user interfaces. In: Kraak M, Molenar M (eds) Advances in GIS research-7th international symposium on spatial data handling (SDH'96), Delft

Kuhn W (2001) Ontologies in support of activities in geographical space. International Journal of Geographical Information Science 15(7):613–631

Kuhn W (2003) Semantic reference systems. International Journal of Geographical Information Science 17(5):405-409

Kuhn W (2007) An image-schematic account on spatial categories. In: Winter S, Duckham M, Kulik L, Kuipers B. (eds) Spatial information theory, 8th international conference, COSIT 2007, Melbourne, Australia, September 19-23, 2007, p 152-168

Kuhn W, Raubal M (2003) Implementing semantic reference systems. In: Gould M, Laurini R, Coulondre S (eds) AGILE 2003, 6th AGILE Conference on geographic information science, Lyon, p 63-72

Kuipers B (1978) Modelling spatial knowledge. Cognitive Science 2(2):129-153

Kwan M (2000) Analysis of human spatial behavior in a GIS environment: recent developments and future prospects. Journal of Geographical Systems 2(1):85–90

La Feypell-De Beaumelle V, Papadimitriou E, Granie M (2010) Pedestrian safety data. In: COST 358 Pedestrian quality needs final report, Part B, p 69-106

Laird J (1987) SOAR: An architecture for general intelligence. Artificial Intelligence 33(1):1–64.

Lee C, Moudon A (2004) Physical activity and environment research in the health field: Implications for urban and transportation planning practice and research. Journal of Planning Literature 19(2):147–181

Legendre P, Legendre L (2012) Numerical ecology, 3rd edn. Developments in environmental modelling, vol 24. Elsevier, Amsterdam, Boston

Leontiev AN (1978) Activity, consciousness, and personality. Prentice-Hall, New Jersey

Leslie E, Coffee N, Frank L, Owen N, Bauman A, Hugo G (2007) Walkability of local communities: Using geographic information systems to objectively assess relevant environmental attributes. Health & Place 13(1):111–122

Lewin K (1935) A dynamic theory of personality: selected papers. McGraw-Hill, New York

Li H, Srihari RK, Niu C, Li W (2003) InfoXtract location normalization: a hybrid approach to geographic references in information extraction. In: Proceedings of the HLT-NAACL 2003 workshop on analysis of geographic references, p 39-44

Lindal P, Hartig T (2013) Architectural variation, building height, and the restorative quality of urban residential streetscapes. Journal of Environmental Psychology 33:26-36

Litman TA (2010) Economic value of walkability. Transportation Research Record 1828, p 3-11

Liu Y, Guo QH, Wieczorek J, Goodchild MF (2009) Positioning localities based on spatial assertions. International Journal of Geographical Information Science 23(11):1471–1501

Llobera M (1996) Exploring the topography of mind: GIS, social space and archaeology. Antiquity(70):612–622

Lynch K (1960) The image of the city. MIT Press, Cambridge, London

Lynch K, Rivkin M (1959) A walk around the block. Landscape 8:24–34

Malczewski J (2006) GIS-based multicriteria decision analysis: a survey of the literature. International Journal of Geographical Information Science, 20(7):703-726

Mark LS, Balliett JA, Craver KD, Douglas SD, Fox T (1990) What an actor must do in order to perceive the affordance for sitting. Ecological Psychology 2(4):325-366

Martin A (2006) Factors influencing pedestrian safety: A literature review. Published project report PPR241, TRL Limited

Martins B (2011) Delimiting imprecise regions with georeferenced photos and land coverage data. In: Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, Tanaka K, Fröhlich P, Kim K (eds) Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 219–229

Maslow AH (1954) Motivation and personality. Harper & Brothers, New York

Mastrogiovanni F, Scalmato A, Sgorbissa A, Zaccaria R (2010) Affordance-based planning for assisting humans in daily activities. In: IE '10 Proceedings of the 2010 sixth international conference on intelligent environments, Kuala Lumpur, p 19-24

May JA (1971) Kant's concept of geography and its relation to recent geographical thought. University of Toronto Press, Toronto

McCormack GR, Shiell A (2011) In search of causality: a systematic review of the relationship between the built environment and physical activity among adults. Int J Behav Nutr Phys Act 8(125):1-11

McDonoguh J (2014) Newton’s scholium on time, space, place and motion. http://aardvark.ucsd.edu/grad_conference/mcdonough.doc

McNeil N (2011) Bikeability and the 20-min neighborhood. Transportation Research Record: Journal of the Transportation Research Board 2247(-1):53–63

Ménard A, Marceau DJ (2005) Exploration of spatial scale sensitivity in geographic cellular automata. Environ. Plann. B 32(5):693–714. doi: 10.1068/b31163

Merian M (1650) Topographia germaniae. Schuchard, Göttingen

Merriam-Webster’s online dictionary, n. d. http://www.merriam-webster.com/ Accessed: 05.08.2015

Michael D, Chrysanthou Y (2003) Automatic high level avatar guidance based on affordance of movement. In: Proceedings eurographics 2003, Granada

Michaels CF (2003) Affordances: four points of debate. Ecological psychology 15(2): 135-148

Monheim R, Raab S (2008) Parkstrategien und Aktionsräume von Parkhausnutzern in Innenstädten mit flächenhaftem Fußgängerbereich - Fallstudie Nürnberg. In: Handbuch der kommunalen Verkehrsplanung: Verkehrssysteme und Verkehrsleitsyssteme, VDE Verlag, Heidelberg

Montello DR (2005) Navigation. In: Shah P, Miyake A (eds), The Cambridge handbook of visuospatial thinking, Cambridge University Press, Cambridge, p 257-294

Montello DR, Goodchild MF, Gottsegen J, Fohl P (2003) Where's downtown?: Behavioral methods for determining referents of vague spatial queries. Spatial Cognition and Computation 3(2&3):185-204

Montgomery J (1998) Making a city: Urbanity, vitality and urban design. Journal of Urban Design 3(1):93–116

Morris EK (2009) Behavior analysis and ecological psychology: Past, present, and future. A review of harry heft's ecological psychology in context. Journal Of The Experimental Analysis Of Behavior 92:275–304

Nardi BA (ed) (1996) Context and consciousness – Activity theory and human-computer interaction. The MIT Press,

Ness MP, Morall JF, Hutchinson BG (1969) An analysis of central business district pedestrian circulation patterns. Highway research record 283:11-18

Ngo TA, See LM (2012) Calibration and validation of agent-based models of land cover change. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 181 - 198

Norberg-Schulz C (1965) Intentions in architecture. MIT Press, Cambridge

Norman D (1988) The design of everyday things. Doubleday, New York

Oliver M (2005) The problem with affordance. E-Learning and Digital Media 2(4):402-413

Ordonez SA, Erath A (2012) Estimating dynamic workplace capacities using public transport smart card data and a household travel survey. Transportation Research Record: Journal of the Transportation Research Board 2344:20-30

Ortmann J, Felice G, Wang D, Daniel D (2014) An egocentric semantic reference system for affordances. Semantic Web 5:449-472

Ortmann J, Michels H (2011) Modelling umwelten. In: Geertman S, Reinhardt WP, Toppen FJ (eds) Advancing geoinformation science for a changing world. Springer, Heidelberg, New York, p 293–316

Ortmann J, Kuhn W (2010) Affordances as qualities. In: Proceedings of the 2010 conference on formal ontology in information systems (FOIS 2010), Amsterdam

Oudejans RR, Michaels CF, van Dort B, Frissen EJP (1996) To cross or not to cross: The effect of locomotion on street-crossing behavior. Ecological Psychology 8(3):259–267

Ovstedal L, Ryeng EO (2002) Who is the most pleased pedestrian? In: WALK21 3rd international conference, Donastia-San Sebastián 8-11 May 2002

Owen N, Humpel N, Leslie E, Baumann A, Sallis JF (2004) Understanding environmental influences on walking. review and research agenda. American Journal of Preventive Medicine 27(1):67–76

Oxley J, Fildes B, Ihsen E, Day R, Charlton J (1995) An investigation of road crossing behavior of older pedestrians. Report No. 81, Monash University Accident Research Centre

Özer Ö, Kubat AS (2007) Walking initiatives: a quantitative movement analysis. In: Proceedings of the 6th space syntax symposium, Istanbul

Painter K (1996) The influence of street lighting improvements on crime, fear and pedestrian street use, after dark. Landscape and Urban Planning 35:193-201

Pannell DJ (1997) Sensitivity analysis: strategies, methods, concepts, examples. Agric Econ 16:139–152

Papadimitriou E, Yannis G, Golias J (2009) A critical assessment of pedestrian behaviour models. Transportation Research Part F 12:242–255

Parent C, Pelekis N, Theodoridis Y, Yan Z, Spaccapietra S, Renso C, Andrienko G, Andrienko N, Bogorny V, Damiani ML, Gkoulalas-Divanis A, Macedo J (2013) Semantic trajectories modeling and analysis. ACM Comput. Surv 45(4):1–32

Pareta K (2013) Remote sensing and GIS based site suitability analysis for tourism development. International Journal of Advanced Research in Engineering and Applied Sciences 2(5):43-58

Park S (2008) Defining, measuring, and evaluating path walkability, and testing its impacts on transit users' mode choice and walking distance to the station. Dissertation, University of California, Berkeley

Parker DC, Manson SM, Janssen MA, Hoffmann MJ, Deadman P (2003) Multi-agent systems for the simulation of land-use and land-cover change: A review. Annals of the Association of American Geographers 93(2):314–337

Penn A, Hillier B, Banister D, Xu J (1998) Configurational modelling of urban movement networks. Environ. Plann. B 25(1):59–84

Pereira JMC, Duckstein L (1993) A multiple criteria decision-making approach to GIS-based land suitability evaluation. International Journal of Geographical Information Systems 7(5):407–424

Perez L, Dragicevic S (2009) An agent-based approach for modeling dynamics of contagious disease spread. Int J Health Geogr 8(1):50

Pikora TJ, Bull FCL, Jamrozik K, Knuiman M, Giles-Corti B, Donovan RJ (2002) Developing a reliable audit instrument to measure the physical environment for physical activity. Am J Prev Med 23(3):187-194

PTV Planung und Verkehr (2010) VISSIM user manual. PTV Planung Transport Verkehr AG

Purves R, Clough P, Joho H (2005) Identifying imprecise regions for geographic information retrieval using the web. In: Proceedings of GIS research UK 13th annual conference, Glasgow, p 313–318

Pushkarev BS, Zupan JM (1975) Urban space for pedestrians. A report of the regional plan association. MIT Press, Cambridge

Rao AS, Georgeff M (1995) BDI agents: from theory to practice. In: Lesser V (ed) Proceedings of the first international conference on multi-agent systems, San Francisco, p 312-319

Raper J (2000) Multidimensional geographic information science. Taylor & Francis, London, New York

Rattenbury T, Nathan M (2009) Methods for extracting place semantics from flickr tags. ACM Transactions on the Web 3(1)

Ratti C, Sobolevsky S, Calabrese F, Andris C, Reades J, Martino M, et al. (2010) Redrawing the map of great britain from a network of human interactions. PLoS ONE 5(12)

Raubal M (2001a) Agent-based simulation of human wayfinding: A perceptual model for unfamiliar buildings. Dissertation, Technical University Vienna

Raubal M (2001b) Human wayfinding in unfamiliar buildings: a simulation with a cognizing agent. Cognitive Processing 2(2-3):363-388

Raubal M, Egenhofer MJ, Pfoser D, Tryfona N (1996) Structuring space with image schemata: wayfinding in airports as a case study. In: Proceedings of the Conference of Spatial Information Theory (COSIT) 1997

Raubal M, Moratz R (2006) A functional model for affordance-based agents. In: Rome E, Hertzberg J, Dorffner G (eds) Towards affordance-based robot control, International seminar, Dagstuhl Castle, Springer, Berlin, Heidelberg, p 91-105

Rauch E, Bukatin M, Baker K (2003) A confidence-based framework for disambiguating geographic terms. In: HLT-NAACL-GEOREF '03 Proceedings of the HLT-NAACL 2003 workshop on analysis of geographic references, p 50-54

Relph E (1976) Place and placelessness. Pion Limited, London

Renalds A, Smith TH, Hale PJ (2010) A systematic review of built environment and health. Family & Community Health 33(1):68–78

Richter HJ, TRILUX GmbH & Co. KG (2005) 13 201 Planungshilfe – Licht für Europas Straßen, Beleuchtung von Straßen, Wegen und Plätzen nach DIN EN 13 201, TRILUX GmbH & Co. KG, Arnsberg

Richter K, Winter S (2014) Landmarks. Springer International Publishing, Cham

Rinzivillo S, Mainardi S, Pezzoni F, Coscia M, Pedreschi D, Giannotti F (2012) Discovering the geographical borders of human mobility. Künstl Intell 26(3):253–260

Robinson VB (1990) Interactive machine acquisition of a fuzzy spatial relation. Computers & Geosciences 16(6):857–872

Rome E, Hertzberg J, Dorffner G (eds) (2008) Towards affordance-based robot control. Lecture notes in computer science. Springer Berlin / Heidelberg

Ronald N, Sterling L, Kirley M (2007) An agent-based approach to modelling pedestrian behaviour. International Journal of Simulation 8(1):25–38

Roth C, Kang SM, Batty M, Barthelemy M, Perc M (2011) Structure of urban movements: polycentric activity and entangled hierarchical flows. PLoS ONE 6(1)

Russel S, Norvig P (2003) Artificial intelligence - A modern approach. Third edition, Prentice Hall, New Jersey

Rynasiewicz R (2014) Newton's views on space, time, and motion. In: Zalta EN (ed) The Stanford encyclopedia of philosophy, The Metaphysics Research Lab, Stanford University

Saaty TL (1980) The analytic hierarchy process. Planning, priority setting, resource allocation. McGraw-Hill International Book Co., New York, London

Sack RD (1980) Conceptions of space in social thought. Macmillan, London

Saelens BE, Handy S (2008) Built environment correlates of walking: A review. Medicine and Science in Sports and Exercise 40(7 suppl):550–566

Saelens BE, Sallis JF, Frank LD (2003) Environmental correlates of walking and cycling: Findings from the transportation, urban design, and planning literatures. Annals of Behavioural Medicine 25(2):80-91

Sakellariou I, Kefalas P, Stamatopoulou I (2008) Enhancing NetLogo to simulate BDI communicating agents. In: SETN 2008, LNAI 5138, p 263-275

Sallis JF, Floyd MF, Rodriguez DA, Saelens BE (2012) Role of built environments in physical activity, obesity, and cardiovascular disease. Circulation 125(5):729–737

Sallis JF, Frank LD, Saelens BE, Kraft MK (2004) Active transportation and physical activity: opportunities for collaboration on transportation and public health research. Transportation Research Part A 38(4):249–268

Samarasekara GN, Fukahori K, Kubota Y (2011) environmental correlates that provide walkability cues for tourists: An analysis based on walking decision narrations. Environment and Behavior 43(4):501–524

Sanches SP, Ferreira MAG (2007) How the elderly perceive the quality of Sidewalks. In: ITE 2000 annual meeting and exhibit, Nashville

Sarkar S (1993) Determination of service levels for pedestrians, with European examples. Journal of the Transportation Research Board 1405:35-42

Saura S, Martinez-Millan J (2001) Sensitivity of landscape pattern metrics to map spatial extent. Photogrammetric Engineering & Remote Sensing 67(9):1027–1036

Scheider S, Janowicz K (2010) Places as media of containment. In: Purves R, Weibel R (eds) Extended abstracts volume, GIScience 2010, Zurich

Scheider S, Janowicz K (2014) Place reference systems: A constructive activity model of reference to places. Applied Ontology 9(2):97–127

Scheider S, Kuhn W (2010) Affordance-based categorization of road network data using a grounded theory of channel networks. International Journal of Geographical Information Science 24(8):1249-1267

Scheider S, Purves R (2013) Semantic place localization from narratives. In: ACM SIGSPATIAL COMP'13, Orlando

Schlieder C, Matyas C (2009) Photographing a city: an analysis of place concepts based on spatial choices. Spatial Cognition & Computation 9(3):212–228

Schlossberg M, Agrawal AW, Irvin K, Bekkouche VL (2007) How far, by which route, and why? a spatial analysis of pedestrian preference. Final Report. MTI Report 06-06

Schockaert S, Cock MD, Kerre EE (2005) Automatic acquisition of fuzzy footprints. In: Proceedings of the workshop on semantic-based geographical information systems, LNCS 3762. Presented at the ACM SIGIR, Agia Napa: Springer, p 1077–1086

Schwander C, Law S (2012) Accessibility analysis with space syntax. The pedestrian movement network in the city centre of Munich. In: Wulfhorst G, Büttner B (eds) Transport demand management, Insights from the mobil.TUM 2012 International scientific conference on mobility and transport, TUM Schriftenreihe Heft 2, p 11-30

Seamon D (1980) Body-subject, time-space routines, and place ballets. In: Buttimer A, Seamon D (eds) The human experience of space and place, London

Searle J (1983) Intentionality: An essay in the philosophy of mind(Vol. 9). Cambridge, England: Cambridge University Press

Sell JL, Taylor JG, Zube EH (1984) Toward a theoretical framework for landscape perception. In: Environmental Perceptions and Behavior: An inventory and Prospect, p 61–83

Sheehy, N., Chapman, A. J., & Conroy, W. A. (eds.) (1997) Biographical dictionary of psychology. Routledge, New York

Shirai M, Hirota M, Ishikawa H, Yokoyama S (2013) A method of area of interest and shooting spot detection using geo-tagged photographs. In: ACM SIGSPATIAL COMP'13, Orlando

Simons PM (2000) Parts. A study in ontology. Oxford University Press, Oxford, New York

Stanilov K (2012) Space in agent-based models. In: Heppenstall AJ, Crooks AT, See LM, Batty M (eds) Agent-based models of geographical systems. Springer, Dordrecht, Heidelberg, London, New York, p 253 - 270

Steinitz C (2012) A framework for geodesign. Changing geography by design, 1st edn. ESRI Press, Redlands

Stoffregen TA (2003) Affordances as properties of the animal environment system. Ecological Psychology 15(2):115–134

Sugiyama T, Neuhaus M, Cole R, Giles-Corti B, Owen N (2012) Destination and route attributes associated with adults' walking: A review. Med. Sci. Sports Exerc. 44(7):1275–1286

Teknomo K (2008) Modeling mobile traffic agents on network simulation. In: Transportation science society of the Philippines 2008, Proceedings of the 16th annual of Transportation Science

Teknomo K, Millonig A (2007) A navigation algorithm for pedestrian simulation in dynamic environments. In: Proceedings of the 11th world conference on transport research (WCTR), California, Berkeley

Timpf S (2003) Geographical activity models. In: Goodchild M, Duckham M, Worboys MF (eds) Perspectives in geographic information science, Taylor and Francis, New York

Timpf S (2008) Appropriating places in public spaces - a multi-agent simulation. In: Klügl F, Timpf S, Schmid U (eds) Agent-based simulation - from cognitive modelling to engineering practice. workshop proceedings der KI'08 Kaiserslautern

Timpf S, Kuhn W (2003) Granularity transformations in wayfinding. In: Freksa C, Brauer W, Habel C, Wender KF (eds) Spatial cognition III. LNAI: 77-88

Timpf S, Volta GS, Pollock DW, Egenhofer MJ (1992) A Conceptual Model of Wayfinding Using Multiple Levels of Abstraction. In: Frank A, Campari I, Formentini U (eds) Theory and methods of spatio-temporal reasoning in geographic space. LNCS 639: 348-367

Tolman EC (1948) Cognitive maps in rats and men. Psychological Review 55(4):189

Tolman EC (1958) Behavior and psychological man. University of California Press, Berkeley

Tomlin CD (2013) GIS and cartographic modeling. ESRI Press, Redlands

Torrens PM (2012) Moving agent pedestrians through space and time. Annals of the Association of American Geographers 102(1):35–66

Trypuz R (2008) Formal ontology of action – a unifying approach. Dissertation, Universit'a degli Studi di Trento

Tuan Y (1979) Space and place: humanistic perspective. In: Gale S, Olsson G (eds) Philosophy in Geography. Springer Netherlands, Dordrecht, p 387–427

Tuan YF (1975) Place: An experiential perspective. Geographical Review 65(2):151-165

Turner A, Penn A (2002) Encoding natural movement as an agent-based system: an investigation into human pedestrian behaviour in the built environment. Environment and Planning B: Planning and Design 29:473–490

Turvey MT (1992) Affordances and prospective control: an outline of the ontology. Ecological Psychology 4(3):173–187

van Holle V, Deforche B, van Cauwenberg J, Goubert L, Maes L, van de Weghe N, Bourdeaudhuij I de (2012) Relationship between the physical environment and different domains of physical activity in European adults: a systematic review. BMC Public Health 12

Varas A, Cornejo M, Mainemer D, Toledo B, Rogan J, Muñoz V, Valdivia J (2007) Cellular automaton model for evacuation process with obstacles. Physica A: Statistical Mechanics and its Applications 382(2):631–642

Varela FJ, Thompson E (1991) The embodied mind. Cambridge, MA, MIT Press

Vasardani M, Timpf S, Winter S, Tomko M (2013) From descriptions to depictions: a conceptual framework. In: Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, Naor M, Nierstrasz O, Pandu Rangan C, Steffen B, Sudan M, Terzopoulos D, Tygar D, Vardi MY, Weikum G, Tenbrink T, Stell J, Galton A, Wood Z (eds) Lecture notes in computer science. Springer International Publishing, Cham,p 299–319

Vasardani M, Winter S, Richter K (2013) Locating place names from place descriptions. International Journal of Geographical Information Science 27(12):2509–2532

Vizzari G, Manenti L, Ohtsuka K, Shimura K (2013) An agent-based pedestrian and group dynamics model applied to experimental and real-world scenarios. Journal of Intelligent Transportation Systems 19(1): 32-45

von Uexküll J, Kriszat G (1956) Streifzüge durch die Umwelten von Tieren und Menschen: ein Bilderbuch usichtbarer Welten – Bedeutungslehre. Rowohlt, Hamburg

Vukmirovic M (2010) Functional abilities of humans and identification of specific groups of pedestrians. In: Walk 21 XI International Conference on Walking and Liveable Communities, The Hague, Netherlands, 16-19 November 2010

Wagner PL (1972) Environments and peoples. Prentice-Hall, Englewood Cliffs

Walter E, Cavegn M, Scaramuzza G, Niemann S, Allenbach R (2007) Fussverkehr-Unfallgeschehen, Risikofaktoren und Prävention. Bubenberg Druck- und Verlags-AG, Bern

Wang F, Brent Hall G, Subaryono (1990) Fuzzy information representation and processing in conventional GIS software: database design and application. International journal of geographical information systems 4(3):261–283

Warren WH (1984) Perceiving affordances: Visual guidance of stair climbing. Journal of Experimental Psychology 105(5):683-703

Warren WH (1995) Constructing an econiche. In: Flach J, Hancock P, Caird J, Vicente K (eds) The ecology of human-machine systems, Erlbaum, Hilldsale, p 210-237

Weidmann U (1993) Transporttechnik der Fussgänger. Transporttechnische Eigenschaften des Fußgängerverkehrs (Literaturauswertung), 2. Auflage. Schriftenreihe des IVT, Zürich

Wells AJ (2002) Gibson’s affordances and Turing’s theory of computation. Ecological Psychology 14(3): 141–180

Wendel-Vos W, Droomers M, Kremers S, Brug J, van Lenthe F (2007) Potential environmental determinants of physical activity in adults: a systematic review. Obesity Reviews 8(5):425–440

Werlen B (1993) Society action and space. An alternative human geography. Routledge, London, New York

Werlen B (2000) Sozialgeographie. Eine Einführung. Uni-Taschenbücher, vol 1911. Haupt, Bern

Wilensky U (1999) NetLogo. Center for connected learning and computer-based modeling, Northwestern University. Evanston, IL. http://ccl.northwestern.edu/netlogo/

Winter S, Freksa C (2012) Approaching the notion of place by contrast. JOSIS(5)

Wolpert J (1964) The decision process in spatial context. Annals of the Association of American Geographers 54(4):537–558

Woolridge M (1999) Intelligent agents. In: Weiss G (ed) Multiagent systems. A modern approach to distributed artificial intelligence. MIT Press, Cambridge, London

Wu F (1998) SimLand: a prototype to simulate land conversion through the integrated GIS and CA with AHP-derived transition rules. International Journal of Geographical Information Science 12:63–82

Yannis G, Papadimitriou E, Theofilatos A (2013) Pedestrian gap acceptance for mid-block street crossing. Transportation Planning and Technology 36(5):450-462

Yao X, Thill J (2006) Spatial queries with qualitative locations in spatial information systems. Computers, Environment and Urban Systems 30(4):485–502

Ying JJ, Lee W, Weng T, Tseng VS (2011) Semantic trajectory mining for location prediction. In: Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems - GIS '11. ACM Press

Yuan J, Zheng Y, Xie X (2012) Discovering regions of different functions in a city using human mobility and POIs. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '12. ACM Press

Zachariadis V (2005) An agent-based approach to the simulation of pedestrian movement and factors that control it. In: Proceedings Computers in Urban Planning and Urban Management CUPUM 2005, London

Zacharias J (2001a) Path choice and visual stimuli: signs of human activity and architecture. Journal of Environmental Psychology 21:341–352

Zacharias J (2001b) Pedestrian behavior and perception in urban walking environments. Journal of Planning Literature 16(3):3-18

Zheng Y, Wolfson OE, Toole JL, Ulm M, González MC, Bauer D (2012) Inferring land use from mobile phone activity. In: Proceedings of the ACM SIGKDD International Workshop on Urban Computing - UrbComp '12. ACM Press

Zhong C, Huang X, Müller Arisona S, Schmitt G (2013) Identifying spatial structure of urban functional centers using travel survey data: A case study of Singapore. In: ACM SIGSPATIAL COMP'13, Orlando

Zhou J, Civco D (1996) Using genetic learning neural networks for spatial decision making in GIS. Photogrammetric Engineering & Remote Sensing 62(11):1287–1295

Zook JB, Lu Y, Glanz K, Zimring C (2011) Design and pedestrianism in a smart growth development. Environment and Behavior 44:216-234

# Appendix

## A: PlaceBuilder Code

### SystemAgentEnvironment.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.HashMap;

import java.util.Hashtable;
import java.util.List;

import org.nlogo.agent.Patch;
import org.nlogo.agent.Turtle;

/**
 * Class for system of agent and patch. Takes one instance of Turtle and Patch,
 * and a HashMap with associated RealizationAction instances and the
 * corresponding suitability values.
 *
 * @author David Jonietz
 */

public class SystemAgentEnvironment {

    public Turtle involvedTurtle; // involved Turtle instance
    public Patch involvedPatch; // involved Patch instance
    public HashMap affordances; // HashMap with RealizationAction
                                // (key), and suitability (value)

    public SystemAgentEnvironment(org.nlogo.api.Turtle turtle,
            org.nlogo.api.Patch patch, List affordances) {
        this.involvedTurtle = (Turtle) turtle;
        this.involvedPatch = (Patch) patch;
        this.affordances = new HashMap();

        // create HashMap from input (nested list from NetLogo)

        for (int i = 0; i < affordances.size(); i++) {
            List sublist = (List) affordances.get(i);
            RealizationAction key = (RealizationAction) sublist.get(0);
            Double value = (Double) sublist.get(1);
            this.affordances.put(key, value);
        }
    }
}

```

### RealizationAction.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.List;

/**
 * Corresponds to the lowest level action.
 * Takes a name (string).
 *
 * @author David Jonietz
 */

public class RealizationAction {
    public String name;

    public RealizationAction(String name) {
        this.name = name;
    }
}

```

## SemanticAction.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.List;

import org.jgrapht.DirectedGraph;
import org.jgrapht.graph.DefaultDirectedGraph;
import org.jgrapht.graph.DefaultEdge;

/**
 * Corresponds to the medium-level action. Takes list of RealizationAction
 * instances and their functional and spatial connections as a nested list of
 * lists.
 *
 * @author David Jonietz
 */

public class SemanticAction {
    public String name; // name of the SemanticAction
    public List<List> realizationActionList; // contributing RealizationAction
                                             // and their weight factors in
                                             // list
    public List<List> connections; // list with connected RealizationAction
                                   // instances
                                   // and their spatial dependencies

    // stores connections of contributing RealizationAction instances as
    // directed graph
    public DirectedGraph realizationActionGraph = new DefaultDirectedGraph<RealizationAction,
            DefaultEdge>(
            DefaultEdge.class);

    public SemanticAction(String name, List<List> realizationActionList,
            List<List> connections) {
        this.name = name;
        this.realizationActionList = realizationActionList;
        this.connections = connections;

        // build realizationActionGraph

        for (int i = 0; i < realizationActionList.size(); i++) {

            // create a vertex for each RealizationAction instance

            RealizationAction element = (RealizationAction) realizationActionList
                    .get(i).get(0);
            realizationActionGraph.addVertex(element);
        }

        for (int i = 0; i < connections.size(); i++) {

            // create a LabeledEdge instance for each tuple in connections
            // variable
            // and put the spatial dependency (string) as label

            List edge = connections.get(i);
            Object from_node = edge.get(0);
            Object to_node = edge.get(1);
            String label = (String) edge.get(2);
            realizationActionGraph.addEdge(from_node, to_node, new LabeledEdge(
                    label));
        }

    }

}
```

# PragmaticAction.java

```java
//
// Copyright: David Jonietz, 2015
//

import org.jgrapht.*;
import org.jgrapht.alg.DijkstraShortestPath;
import org.jgrapht.alg.DirectedNeighborIndex;
import org.jgrapht.graph.DefaultDirectedGraph;
import org.jgrapht.graph.DefaultDirectedWeightedGraph;
import org.jgrapht.graph.DefaultEdge;
import org.jgrapht.graph.DefaultWeightedEdge;
import org.nlogo.agent.Patch;
import org.nlogo.api.LogoList;
import java.util.*;

/**
 * Corresponds to the highest-level action. Takes list of semanticAction
 * instances and their functional and spatial connections as a nested list of
 * lists. Has variable place, which is calculated with method calculatePlace.
 *
 * @author David Jonietz
 */

public class PragmaticAction {
    public String name; // name of the PragmaticAction

    // stores connections of contributing SemanticAction instances as directed
    // graph
    public DirectedGraph semanticActionGraph = new DefaultDirectedGraph<SemanticAction,
            DefaultWeightedEdge>(
            DefaultWeightedEdge.class);
    public List<List> semanticActionList; // contributing SemanticActions
                                           // and their weight factors in list
    public List<List> connections; // list with connected SemanticAction
                                   // instances
                                   // and their spatial dependencies
    public List<List> place = new ArrayList(); // stores place as list of lists

    public PragmaticAction(String name, List<List> semanticActionList,
            List<List> connections) {
        this.name = name;
        this.semanticActionList = semanticActionList;
        this.connections = connections;

        // build semanticActionGraph based on semanticActionList and connections

        for (int i = 0; i < semanticActionList.size(); i++) {

            // create a vertex for each SemanticAction instance in
            // semanticActionList

            SemanticAction element = (SemanticAction) semanticActionList.get(i)
                    .get(0);
            semanticActionGraph.addVertex(element);
        }

        for (int i = 0; i < connections.size(); i++) {

            // create a LabeledEdge instance for each tuple in connections
            // variable and put the spatial dependency (string) as label

            List edge = connections.get(i);
            Object from_node = edge.get(0);
            Object to_node = edge.get(1);
            String label = (String) edge.get(2);
```

```
            semanticActionGraph.addEdge(from_node, to_node, new LabeledEdge(
                    label));
        }
    }

    // takes a list of SystemAgentEnvironment instances, creates a tupleGraph
    // based on semanticActionGraph and realizationActionGraph
    // of each involved SemanticAction instance, and calculates and returns
    // place variable

    public List calculatePlace(List systemAgentEnvironmentList) {

        // initialize variables for sink and source nodes in tuple graph
        List tupleGraphSinkNodes = new ArrayList(); // sink nodes as end points
                                                    // of paths through
                                                    // tupleGraph
        List tupleGraphSourceNodes = new ArrayList(); // source nodes as
                                                        // starting points of
                                                        // paths through
                                                        // tupleGraph

        // initialize tupleGraph

        DefaultDirectedWeightedGraph tupleGraph = new DefaultDirectedWeightedGraph<HashMap,
                DefaultWeightedEdge>(
                DefaultWeightedEdge.class);
        List semanticActionGraphSinkNodes = new ArrayList(); // stores sink
                                                                // nodes of

                                                                // semanticActiongraph
        List semanticActionGraphSourceNodes = new ArrayList(); // stores source
                                                                // nodes of
                                                                // semanticActiongraph

        // create lists of semantic actions and their weight coefficients
        List semActionList = new ArrayList();
        List semActionWeightList = new ArrayList();
        for (List sublist : semanticActionList) {
            semActionList.add(sublist.get(0));
            semActionWeightList.add(sublist.get(1));
        }

        // identify sink and source nodes in semanticActiongraph based on number
        // of outgoing or incoming edges and store in variables

        for (Object node : semanticActionGraph.vertexSet()) {
            SemanticAction workedSemanticAction = (SemanticAction) node;

            if (semanticActionGraph.outDegreeOf(workedSemanticAction) == 0) {
                semanticActionGraphSinkNodes.add(workedSemanticAction);
            }

            if (semanticActionGraph.inDegreeOf(workedSemanticAction) == 0) {
                semanticActionGraphSourceNodes.add(workedSemanticAction);
            }

            // retrieve weight of semantic action and store as variable

            Double sAWeightCoeff = (Double) semActionWeightList
                    .get(semActionList.indexOf(workedSemanticAction));

            // create tupleGraph - vertices consist of tuples
            // (SystemAgentEnvironment instance, RealizationAction instance,
            // SemanticAction instance)
            // edges are based on connections from semanticActionGraph and
            // realizationActionGraph

            // retrieve contributing RealizationAction instances and their
            // weight coefficients from current SemanticAction instance and
            // store in lists
            List realizationActionList = new ArrayList();
            List realizationActionWeightList = new ArrayList();
            for (List sublist : workedSemanticAction.realizationActionList) {
                realizationActionList.add(sublist.get(0));
                realizationActionWeightList.add(sublist.get(1));
            }

            // check if SystemAgentEnvironment instances afford
            // RealizationAction instances, if true, create tuple and add to
            // tupleGraph as vertex

            for (Object rA : realizationActionList) {
                RealizationAction workedRealizationAction = (RealizationAction) rA;

                for (Object sys : systemAgentEnvironmentList) {
```

```
                SystemAgentEnvironment workedSystemAgentEnvironment = (SystemAgentEnvironment)
                        sys;

                if (workedSystemAgentEnvironment.affordances
                        .containsKey(workedRealizationAction)) {

                    // retrieve SystemAgentEnvironment, RealizationAction
                    // and SemanticAction instances

                    List key = new ArrayList();
                    key.add(workedSystemAgentEnvironment);
                    key.add(workedRealizationAction);
                    key.add(workedSemanticAction);
                    HashMap tuple = new HashMap();

                    // retrieve suitability value

                    tuple.put(key, workedSystemAgentEnvironment.affordances
                            .get(workedRealizationAction));

                    // add vertex to tupleGraph

                    tupleGraph.addVertex(tuple);
                }
            }
        }

        // create edges in tupleGraph based on realizationGraph variable of
        // each SemanticAction instance

        // retrieve all nodes in realizationActionGraph for each
        // SemanticAction instance

        Set realizationActionGraphNodes = workedSemanticAction.realizationActionGraph
                .vertexSet();

        for (Object rAGraphNode : realizationActionGraphNodes) {
            RealizationAction workedRealizationActionGraphNode = (RealizationAction) rAGraphNode;

            // retrieve realizationAction weight coefficient from list
            Double rAWeightCoeff = (Double) realizationActionWeightList
                    .get(realizationActionList
                            .indexOf(workedRealizationActionGraphNode));

            // store neighboring nodes for each node in
            // realizationActionGraph in cache

            DirectedNeighborIndex rAGraphDirectedNeighborIndex = new DirectedNeighborIndex(
                    workedSemanticAction.realizationActionGraph);

            // store neighboring nodes of current RealizationAction instance
            // in list

            List workedRealizationActionGraphNodeNeighbors = rAGraphDirectedNeighborIndex
                    .successorListOf(workedRealizationActionGraphNode);

            if (!workedRealizationActionGraphNodeNeighbors.isEmpty()) {

                // iterate through tuples from tupleGraph nodes

                for (Object tupleNodeFrom : tupleGraph.vertexSet()) {
                    HashMap tupleGraphNodeFrom = (HashMap) tupleNodeFrom;
                    List nestedKeylistFrom = new ArrayList(
                            tupleGraphNodeFrom.keySet());
                    List workedKeyFrom = (List) nestedKeylistFrom.get(0);

                    // identify the tuple which includes the current
                    // RealizationAction and SemanticAction instances

                    if (workedKeyFrom.get(1) == workedRealizationActionGraphNode
                            && workedKeyFrom.get(2) == workedSemanticAction) {

                        // iterate through tuples from tupleGraph nodes

                        for (Object tupleNodeTo : tupleGraph.vertexSet()) {
                            HashMap tupleGraphNodeTo = (HashMap) tupleNodeTo;
                            List nestedKeylistTo = new ArrayList(
                                    tupleGraphNodeTo.keySet());
                            List workedKeyTo = (List) nestedKeylistTo
                                    .get(0);

                            // identify nodes which are neighbors in
                            // realizationActionGraph and contribute to the
                            // same SemanticAction instance
```

```
                if ((workedRealizationActionGraphNodeNeighbors
                        .contains(workedKeyTo.get(1)) && workedKeyTo
                        .get(2) == workedSemanticAction)) {

                    // get edge label (spatial dependency) from
                    // semanticActionGraph

                    LabeledEdge workedRealizationActionGraphNodeEdge = (LabeledEdge)
                            workedSemanticAction.realizationActionGraph
                            .getEdge(
                                    workedRealizationActionGraphNode,
                                    workedKeyTo.get(1));
                    String workedSemanticActionGraphNodeEdgeLabel =
                            workedRealizationActionGraphNodeEdge.label;

                    // if label is "sameSystem", create only
                    // edges in tupleGraph between tuples with
                    // the same patch

                    if (workedSemanticActionGraphNodeEdgeLabel == "sameSystem") {

                        if (workedKeyFrom.get(0) == workedKeyTo
                                .get(0)) {
                            tupleGraph.addEdge(
                                    tupleGraphNodeFrom,
                                    tupleGraphNodeTo,
                                    new LabeledEdge());

                            LabeledEdge edge = (LabeledEdge) tupleGraph
                                    .getEdge(
                                            tupleGraphNodeFrom,
                                            tupleGraphNodeTo);

                            // set inverse suitability of
                            // systemAgentEnvironment instance
                            // as edge weight; if to-node is
                            // sink node, the weight from
                            // from-node and to-node are added

                            if (tupleGraph
                                    .outDegreeOf(tupleGraphNodeTo) == 0) {
                                Double rAWeightCoeffTo = (Double)
                                        realizationActionWeightList
                                        .get(realizationActionList
                                                .indexOf(workedKeyTo
                                                        .get(1)));
                                double weight = (1.0 - (sAWeightCoeff
                                        * rAWeightCoeff * (Double) tupleGraphNodeFrom
                                            .get(workedKeyFrom)))
                                        + (1.0 - (sAWeightCoeff
                                                * rAWeightCoeffTo * (Double)
                                                tupleGraphNodeTo
                                                    .get(workedKeyTo)));
                                tupleGraph.setEdgeWeight(edge,
                                        weight);
                            }
```

```
                    else {
                        double weight = 1.0 - (sAWeightCoeff
                                * rAWeightCoeff * (Double) tupleGraphNodeFrom
                                .get(workedKeyFrom));
                        tupleGraph.setEdgeWeight(edge,
                                weight);
                    }
                }
            }

            // if label is neighbors, create only edges
            // in tupleGraph between tuples where the
            // patches are topological neighbors

            if (workedSemanticActionGraphNodeEdgeLabel == "neighbors") {
                SystemAgentEnvironment systemFrom = (SystemAgentEnvironment)
                        workedKeyFrom
                        .get(0);
                SystemAgentEnvironment systemTo = (SystemAgentEnvironment)
                        workedKeyTo
                        .get(0);
                Patch patchFrom = (Patch) systemFrom.involvedPatch;
                Patch patchTo = (Patch) systemTo.involvedPatch;

                if (patchFrom.getNeighbors().contains(
                        patchTo)) {
                    tupleGraph.addEdge(
                            tupleGraphNodeFrom,
                            tupleGraphNodeTo,
                            new LabeledEdge());
                    LabeledEdge edge = (LabeledEdge) tupleGraph
                            .getEdge(
                                    tupleGraphNodeFrom,
                                    tupleGraphNodeTo);

                    // set inverse suitability of
                    // SystemAgentEnvironment instance
                    // as edge weight; if to-node is
                    // sink node, the weight from
                    // from-node and to-node are added

                    if (tupleGraph
                            .outDegreeOf(tupleGraphNodeTo) == 0) {
                        Double rAWeightCoeffTo = (Double)
                                realizationActionWeightList
                                .get(realizationActionList
                                        .indexOf(workedKeyTo
                                                .get(1)));
                        double weight = (1.0 - (sAWeightCoeff
                                * rAWeightCoeff * (Double) tupleGraphNodeFrom
                                    .get(workedKeyFrom)))
                                + (1.0 - (sAWeightCoeff
                                        * rAWeightCoeffTo * (Double)
                                        tupleGraphNodeTo
                                            .get(workedKeyTo)));
```

```
                                        tupleGraph.setEdgeWeight(edge,
                                                weight);
                                    }

                                    else {
                                        double weight = 1.0 - (sAWeightCoeff
                                                * rAWeightCoeff * (Double) tupleGraphNodeFrom
                                                .get(workedKeyFrom));
                                        tupleGraph.setEdgeWeight(edge,
                                                weight);
                                    }
                                }
                            }

                            // if label is noSpatialRestriction, create
                            // edges in tupleGraph regardless of spatial
                            // relation between patches

                            if (workedSemanticActionGraphNodeEdgeLabel ==
                                    "noSpatialRestriction") {
                                tupleGraph.addEdge(tupleGraphNodeFrom,
                                        tupleGraphNodeTo,
                                        new LabeledEdge());
                                LabeledEdge edge = (LabeledEdge) tupleGraph
                                        .getEdge(tupleGraphNodeFrom,
                                                tupleGraphNodeTo);

                                if (tupleGraph
                                        .outDegreeOf(tupleGraphNodeTo) == 0) {
                                    Double rAWeightCoeffTo = (Double)
                                            realizationActionWeightList
                                            .get(realizationActionList
                                                    .indexOf(workedKeyTo
                                                            .get(1)));
                                    //set edge weight; if to-node is sink-node,
                                    //the weight from from- and to-nodes are
                                    //summed
                                    double weight = (1.0 - (sAWeightCoeff
                                            * rAWeightCoeff * (Double) tupleGraphNodeFrom
                                                .get(workedKeyFrom)))
                                            + (1.0 - (sAWeightCoeff
                                                    * rAWeightCoeffTo * (Double)
                                                    tupleGraphNodeTo
                                                        .get(workedKeyTo)));
                                    tupleGraph.setEdgeWeight(edge,
                                            weight);
                                }

                                else {
                                    double weight = 1.0 - (sAWeightCoeff
                                            * rAWeightCoeff * (Double) tupleGraphNodeFrom
                                            .get(workedKeyFrom));
                                    tupleGraph.setEdgeWeight(edge,
                                            weight);
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}

// create connections in tupleGraph based on semanticActionGraph

// retrieve SemanticAction instances which exist as nodes in
// semanticActionGraph

for (Object node : semanticActionGraph.vertexSet()) {
    SemanticAction workedSemanticAction = (SemanticAction) node;

    // retrieve weight of semantic action and store as variable

    Double sAWeightCoeff = (Double) semActionWeightList
            .get(semActionList.indexOf(workedSemanticAction));

    // store neighboring nodes for each node in semanticActionGraph in
    // cache

    DirectedNeighborIndex sAGraphDirectedNeighborIndex = new DirectedNeighborIndex(
            semanticActionGraph);

    // store neighboring nodes of current SemanticAction instance in
```

```
        // list

        List workedSemanticActionGraphNodeNeighbors = sAGraphDirectedNeighborIndex
                .successorListOf(workedSemanticAction);

        if (!workedSemanticActionGraphNodeNeighbors.isEmpty()) {
            List workedSemanticActionRealizationActionGraphSinkNodes = new ArrayList();
            List workedSemanticActionTupleGraphSinkNodes = new ArrayList();

            // identify sink nodes of RealizationActionGraph of current
            // SemanticAction instance

            for (Object rANode : workedSemanticAction.realizationActionGraph
                    .vertexSet()) {

                if (workedSemanticAction.realizationActionGraph
                        .outDegreeOf(rANode) == 0) {
                    workedSemanticActionRealizationActionGraphSinkNodes
                            .add(rANode);
                }
            }

            // identify nodes in tupleGraph which correspond to the same
            // RealizationAction as a sink node and the current
            // SemanticAction instance

            for (Object tuple : tupleGraph.vertexSet()) {
                HashMap tupleGraphNode = (HashMap) tuple;
                List nestedKeylist = new ArrayList(tupleGraphNode.keySet());
                List workedKey = (List) nestedKeylist.get(0);

                if (workedSemanticActionRealizationActionGraphSinkNodes
                        .contains(workedKey.get(1))
                        && workedKey.get(2) == workedSemanticAction) {
                    workedSemanticActionTupleGraphSinkNodes.add(tuple);
                }
            }

            for (Object sANeighbor : workedSemanticActionGraphNodeNeighbors) {
                SemanticAction workedSemanticActionGraphNodeNeighbor =
                        (SemanticAction) sANeighbor;
                List workedSemanticActionGraphNodeNeighborRealizationActionGraphSourceNodes =
                        new ArrayList();
                List workedSemanticActionGraphNodeNeighborTupleGraphSourceNodes =
                        new ArrayList();

                // identify source nodes of realizationActionGraph of
                // neighboring nodes of current SemanticAction instance in
                // semanticActionGraph

                for (Object rANode : workedSemanticActionGraphNodeNeighbor.realizationActionGraph
                        .vertexSet()) {

                    if (workedSemanticActionGraphNodeNeighbor.realizationActionGraph
                            .inDegreeOf(rANode) == 0) {
                        workedSemanticActionGraphNodeNeighborRealizationActionGraphSourceNodes
                                .add(rANode);
                    }
                }

                // identify nodes in tupleGraph which correspond to the same
                // RealizationAction as a sink node and the current
                // SemanticAction instance

                for (Object tuple : tupleGraph.vertexSet()) {
                    HashMap tupleGraphNode = (HashMap) tuple;
                    List nestedKeylist = new ArrayList(
                            tupleGraphNode.keySet());
                    List workedKey = (List) nestedKeylist.get(0);

                    if (workedSemanticActionGraphNodeNeighborRealizationActionGraphSourceNodes
                            .contains(workedKey.get(1))
                            && workedKey.get(2) == workedSemanticActionGraphNodeNeighbor) {
                        workedSemanticActionGraphNodeNeighborTupleGraphSourceNodes
                                .add(tuple);
                    }
                }

                // get edge label (spatial dependency) from
                // semanticActionGraph

                LabeledEdge workedSemanticActionGraphNodeEdge = (LabeledEdge) semanticActionGraph
                        .getEdge(workedSemanticAction,
                                workedSemanticActionGraphNodeNeighbor);
```

```java
String workedSemanticActionGraphNodeEdgeLabel =
        workedSemanticActionGraphNodeEdge.label;

// if label is "sameSystem", create only edges in tupleGraph
// between tuples with the same patch

if (workedSemanticActionGraphNodeEdgeLabel == "sameSystem") {

    for (Object tupleFrom : workedSemanticActionTupleGraphSinkNodes) {
        HashMap tupleGraphNodeFrom = (HashMap) tupleFrom;
        List nestedKeylistFrom = new ArrayList(
                tupleGraphNodeFrom.keySet());
        List workedKeyFrom = (List) nestedKeylistFrom
                .get(0);

        for (Object tupleTo :
            workedSemanticActionGraphNodeNeighborTupleGraphSourceNodes) {
            HashMap tupleGraphNodeTo = (HashMap) tupleTo;
            List nestedKeylistTo = new ArrayList(
                    tupleGraphNodeTo.keySet());
            List workedKeyTo = (List) nestedKeylistTo
                    .get(0);

            if (workedKeyFrom.get(0) == workedKeyTo.get(0)) {
                tupleGraph
                        .addEdge(tupleGraphNodeFrom,
                                tupleGraphNodeTo,
                                new LabeledEdge());
                LabeledEdge edge = (LabeledEdge) tupleGraph
                        .getEdge(tupleGraphNodeFrom,
                                tupleGraphNodeTo);

                // set inverse suitability of
                // systemAgentEnvironment instance as edge
                // weight; if to-node is sink node, the
                // weight from from-node and to-node are
                // added

                if (tupleGraph
                        .outDegreeOf(tupleGraphNodeTo) == 0) {

                    // retrieve weight of TO-semantic action
                    // and store as variable
                    Double sAWeightCoeffTo = (Double) semActionWeightList
                            .get(semActionList
                                    .indexOf(workedKeyTo
                                            .get(2)));

                    //set edge weight; if to-node is sink-node,
                    //the weight from from- and to-nodes are
                    //summed
                    double weight = (1.0 - (sAWeightCoeff *
                            (Double) tupleGraphNodeFrom
                            .get(workedKeyFrom)))
                            + (1.0 - (sAWeightCoeffTo * (Double) tupleGraphNodeTo
                                    .get(workedKeyTo)));
                    tupleGraph.setEdgeWeight(edge, weight);
                }

                else {
                    double weight = 1.0 - (sAWeightCoeff *
                            (Double) tupleGraphNodeFrom
                            .get(workedKeyFrom));
                    tupleGraph.setEdgeWeight(edge, weight);
                }
            }
        }
    }
}

// if label is neighbors, create only edges in tupleGraph
// between tuples where the patches are topological
// neighbors

if (workedSemanticActionGraphNodeEdgeLabel == "neighbors") {

    for (Object tupleFrom : workedSemanticActionTupleGraphSinkNodes) {
        HashMap tupleGraphNodeFrom = (HashMap) tupleFrom;
        List nestedKeylistFrom = new ArrayList(
                tupleGraphNodeFrom.keySet());
        List workedKeyFrom = (List) nestedKeylistFrom
                .get(0);
```

```
        for (Object tupleTo :
            workedSemanticActionGraphNodeNeighborTupleGraphSourceNodes) {
            HashMap tupleGraphNodeTo = (HashMap) tupleTo;
            List nestedKeylistTo = new ArrayList(
                    tupleGraphNodeTo.keySet());
            List workedKeyTo = (List) nestedKeylistTo
                    .get(0);
            SystemAgentEnvironment systemFrom = (SystemAgentEnvironment)
                    workedKeyFrom
                    .get(0);
            SystemAgentEnvironment systemTo = (SystemAgentEnvironment)
                    workedKeyTo
                    .get(0);
            Patch patchFrom = (Patch) systemFrom.involvedPatch;
            Patch patchTo = (Patch) systemTo.involvedPatch;

            if (patchFrom.getNeighbors().contains(patchTo)) {
                tupleGraph
                        .addEdge(tupleGraphNodeFrom,
                                tupleGraphNodeTo,
                                new LabeledEdge());
                LabeledEdge edge = (LabeledEdge) tupleGraph
                        .getEdge(tupleGraphNodeFrom,
                                tupleGraphNodeTo);

                // set inverse suitability of
                // SystemAgentEnvironment instance as edge
                // weight; if to-node is sink node, the
                // weight from from-node and to-node are
                // added

                if (tupleGraph
                        .outDegreeOf(tupleGraphNodeTo) == 0) {

                    // retrieve weight of TO-semantic action
                    // and store as variable
                    Double sAWeightCoeffTo = (Double) semActionWeightList
                            .get(semActionList
                                    .indexOf(workedKeyTo
                                            .get(2)));

                    double weight = (1.0 - (sAWeightCoeff *
                            (Double) tupleGraphNodeFrom
                            .get(workedKeyFrom)))
                            + (1.0 - (sAWeightCoeffTo * (Double) tupleGraphNodeTo
                                    .get(workedKeyTo)));
                    tupleGraph.setEdgeWeight(edge, weight);
                }

                else {
                    double weight = 1.0 - (sAWeightCoeff *
                            (Double) tupleGraphNodeFrom
                            .get(workedKeyFrom));
                    tupleGraph.setEdgeWeight(edge, weight);
                }
            }
        }
    }
}

// if label is noSpatialRestriction, create edges in
// tupleGraph regardless of spatial relation between patches

if (workedSemanticActionGraphNodeEdgeLabel == "noSpatialRestriction") {

    for (Object tupleFrom : workedSemanticActionTupleGraphSinkNodes) {
        HashMap tupleGraphNodeFrom = (HashMap) tupleFrom;
        List nestedKeylistFrom = new ArrayList(
                tupleGraphNodeFrom.keySet());
        List workedKeyFrom = (List) nestedKeylistFrom
                .get(0);

        for (Object tupleTo :
            workedSemanticActionGraphNodeNeighborTupleGraphSourceNodes) {
            HashMap tupleGraphNodeTo = (HashMap) tupleTo;
            List nestedKeylistTo = new ArrayList(
                    tupleGraphNodeTo.keySet());
            List workedKeyTo = (List) nestedKeylistTo
                    .get(0);
            tupleGraph.addEdge(tupleGraphNodeFrom,
                    tupleGraphNodeTo, new LabeledEdge());
            LabeledEdge edge = (LabeledEdge) tupleGraph
                    .getEdge(tupleGraphNodeFrom,
                            tupleGraphNodeTo);
```

```java

                        // set inverse suitability of
                        // systemAgentEnvironment instance as edge
                        // weight; if to-node is sink node, the weight
                        // from from-node and to-node are added

                        if (tupleGraph.outDegreeOf(tupleGraphNodeTo) == 0) {

                            // retrieve weight of TO-semantic action and
                            // store as variable
                            Double sAWeightCoeffTo = (Double) semActionWeightList
                                    .get(semActionList
                                            .indexOf(workedKeyTo.get(2)));

                            double weight = (1.0 - (sAWeightCoeff *
                                    (Double) tupleGraphNodeFrom
                                    .get(workedKeyFrom)))
                                    + (1.0 - (sAWeightCoeffTo *
                                            (Double) tupleGraphNodeTo
                                            .get(workedKeyTo)));
                            tupleGraph.setEdgeWeight(edge, weight);
                        }

                        else {
                            double weight = 1.0 - (sAWeightCoeff *
                                    (Double) tupleGraphNodeFrom
                                    .get(workedKeyFrom));
                            tupleGraph.setEdgeWeight(edge, weight);
                        }
                    }
                }
            }
        }
    }
}

// identify sink nodes (no outgoing edges) and source nodes (no incoming
// edges) as derived from semanticActionGraph and its
// realizationActionGraph in tupleGraph

for (Object node : tupleGraph.vertexSet()) {
    HashMap tupleGraphNode = (HashMap) node;
    List nestedKeylist = new ArrayList(tupleGraphNode.keySet());
    List keylist = (List) nestedKeylist.get(0);
    SemanticAction tupleSemAction = (SemanticAction) keylist.get(2);
    RealizationAction tupleRealAction = (RealizationAction) keylist
            .get(1);

    List tupleSemActionSinkNodes = new ArrayList();
    for (Object rANode : tupleSemAction.realizationActionGraph
            .vertexSet()) {

        if (tupleSemAction.realizationActionGraph.outDegreeOf(rANode) == 0) {
            tupleSemActionSinkNodes.add(rANode);
        }
    }

    List tupleSemActionSourceNodes = new ArrayList();
    for (Object rANode : tupleSemAction.realizationActionGraph
            .vertexSet()) {

        if (tupleSemAction.realizationActionGraph.inDegreeOf(rANode) == 0) {
            tupleSemActionSourceNodes.add(rANode);
        }
    }

    if (semanticActionGraphSinkNodes.contains(tupleSemAction)
            && tupleSemActionSinkNodes.contains(tupleRealAction)
            && tupleGraph.outDegreeOf(tupleGraphNode) == 0) {
        tupleGraphSinkNodes.add(tupleGraphNode);
    }

    if (semanticActionGraphSourceNodes.contains(tupleSemAction)
            && tupleSemActionSourceNodes.contains(tupleRealAction)
            && tupleGraph.inDegreeOf(tupleGraphNode) == 0) {
        tupleGraphSourceNodes.add(tupleGraphNode);
    }
}

// find shortest path (Dijkstra-algorithm) from each source to each sink
// based on the inverse tuple suitability values as impedance

try {
    List optimalPath = new ArrayList();
```

```
        TreeMap optimalPathsPerSource = new TreeMap();

        for (Object source : tupleGraphSourceNodes) {
            HashMap workedTupleGraphSourceNode = (HashMap) source;
            TreeMap optimalPathsToEachSink = new TreeMap();

            for (Object sink : tupleGraphSinkNodes) {
                HashMap workedTupleGraphSinkNode = (HashMap) sink;

                // compute shortest paths through tupleGraph

                DijkstraShortestPath shortestPath = new DijkstraShortestPath(
                        tupleGraph, workedTupleGraphSourceNode,
                        workedTupleGraphSinkNode);

                // add shortest path from source to each sink to tree map

                optimalPathsToEachSink.put(shortestPath.getPathLength(),
                        shortestPath);
            }

            // identify from all paths to all sinks the one with the lowest
            // cost value and store in treemap

            optimalPathsPerSource.put(optimalPathsToEachSink.firstEntry()
                    .getKey(), optimalPathsToEachSink.firstEntry()
                    .getValue());
        }

        // from each shortest paths of all OD-pairs, identify the shortest
        // one and store in list

        optimalPath.add(optimalPathsPerSource.firstEntry().getValue());

        // get the shortest path as a path object

        DijkstraShortestPath SP = (DijkstraShortestPath) optimalPath.get(0);
        GraphPath path = SP.getPath();

        // for all edges in the shortest path, identify all from-nodes

        for (int i = 0; i < path.getEdgeList().size(); i++) {
            LabeledEdge edge = (LabeledEdge) path.getEdgeList().get(i);
            HashMap fromTuple = (HashMap) path.getGraph().getEdgeSource(
                    edge);

            // for each part of the key in from-tuple, add the instances of
            // SystemAgentEnvironment, RealizationAction,
            // SemanticAction and the suitability value to variable places

            Iterator itrFromTupleKeySet = fromTuple.keySet().iterator();
            while (itrFromTupleKeySet.hasNext()) {
                List tuple = (List) itrFromTupleKeySet.next();
                SystemAgentEnvironment systemAgentEnvironment = (SystemAgentEnvironment) tuple
                        .get(0);
                SemanticAction semanticAction = (SemanticAction) tuple
                        .get(2);
                RealizationAction realizationAction = (RealizationAction) tuple
                        .get(1);
                Double sAWeightCoeff = (Double) semActionWeightList
                        .get(semActionList.indexOf(semanticAction));

                // retrieve weight of realizationAction from
                // semanticAction.realizatioActionList

                List realizationActionList = new ArrayList();
                List realizationActionWeightList = new ArrayList();
                for (List sublist : semanticAction.realizationActionList) {
                    realizationActionList.add(sublist.get(0));
                    realizationActionWeightList.add(sublist.get(1));
                }
                Double rAWeightCoeff = (Double) realizationActionWeightList
                        .get(realizationActionList
                                .indexOf(realizationAction));
                Double value = sAWeightCoeff * rAWeightCoeff
                        * (Double) fromTuple.get(tuple);
                List sublist = new ArrayList();
                sublist.add(systemAgentEnvironment.involvedPatch);
                sublist.add(realizationAction.name);
                sublist.add(semanticAction.name);
                sublist.add(value);
                place.add(LogoList.fromJava(sublist));
            }
        }
```

```java

            // for all edges in the shortestPath, identify all to-nodes

            HashMap toTuple = (HashMap) path.getEndVertex();

            // for each part of the key in to-tuple, add the instances of
            // SystemAgentEnvironment, RealizationAction,
            // SemanticAction and the suitability value to variable places

            Iterator itrToTupleKeySet = toTuple.keySet().iterator();
            while (itrToTupleKeySet.hasNext()) {
                List tuple = (List) itrToTupleKeySet.next();
                SystemAgentEnvironment systemAgentEnvironment = (SystemAgentEnvironment) tuple
                        .get(0);
                SemanticAction semanticAction = (SemanticAction) tuple.get(2);
                RealizationAction realizationAction = (RealizationAction) tuple
                        .get(1);
                Double sAWeightCoeff = (Double) semActionWeightList
                        .get(semActionList.indexOf(semanticAction));

                // retrieve weight of realizationAction from
                // semanticAction.realizatioActionList

                List realizationActionList = new ArrayList();
                List realizationActionWeightList = new ArrayList();
                for (List sublist : semanticAction.realizationActionList) {
                    realizationActionList.add(sublist.get(0));
                    realizationActionWeightList.add(sublist.get(1));
                }
                Double rAWeightCoeff = (Double) realizationActionWeightList
                        .get(realizationActionList.indexOf(realizationAction));

                // multiply with suitability value

                Double value = sAWeightCoeff * rAWeightCoeff
                        * (Double) toTuple.get(tuple);

                List sublist = new ArrayList();
                sublist.add(systemAgentEnvironment.involvedPatch);
                sublist.add(realizationAction.name);
                sublist.add(semanticAction.name);
                sublist.add(value);
                place.add(LogoList.fromJava(sublist));
            }

            // return the variable place
            return place;
        }

        // catches the situation where there is no path though tupleGraph,
        // meaning the PragmaticAction is not possible
        // with the given SystemAgentEnvironment instances; an empty list is
        // returned in this case

        catch (Exception e) {
            List emptyList = new ArrayList();
            return emptyList;
        }
    }
}

```

## RealizationActionFactory.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.List;
import org.nlogo.agent.ArrayAgentSet;
import org.nlogo.agent.ArrayAgentSet.Iterator;
import org.nlogo.api.*;
import org.nlogo.nvm.Workspace;

/**
 * Returns an instance of class RealizationAction.
 *
 * @autor David Jonietz
 */

public class RealizationActionFactory extends DefaultReporter {

    public Syntax getSyntax() {
        return Syntax.reporterSyntax(new int[] { Syntax.StringType() },
                Syntax.ListType());
    }

    public Object report(Argument args[], Context arg1)
            throws ExtensionException, LogoException {
        RealizationAction realizationAction = new RealizationAction(
                (String) args[0].get());
        return realizationAction;
    }
}
```

## SemanticActionFactory.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.List;

import org.nlogo.agent.ArrayAgentSet;
import org.nlogo.agent.ArrayAgentSet.Iterator;
import org.nlogo.api.*;
import org.nlogo.nvm.Workspace;

/**
 * Returns an instance of class SemanticAction.
 *
 * @autor David Jonietz
 */

public class SemanticActionFactory extends DefaultReporter {

    public Syntax getSyntax() {
        return Syntax.reporterSyntax(
                new int[] { Syntax.StringType(), Syntax.ListType(),
                        Syntax.ListType() }, Syntax.ListType());
    }

    public Object report(Argument args[], Context arg1)
            throws ExtensionException, LogoException {

        SemanticAction semanticAction = new SemanticAction(
                (String) args[0].get(), (List) args[1].get(),
                (List) args[2].get());
        return semanticAction;
    }
}
```

## PragmaticActionFactory.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.List;
import org.nlogo.agent.ArrayAgentSet;
import org.nlogo.agent.ArrayAgentSet.Iterator;
import org.nlogo.api.*;
import org.nlogo.nvm.Workspace;

/**
 * Returns an instance of class PragmaticAction.
 *
 * @autor David Jonietz
 */

public class PragmaticActionFactory extends DefaultReporter {

    public Syntax getSyntax() {
        return Syntax.reporterSyntax(
                new int[] { Syntax.StringType(), Syntax.ListType(),
                        Syntax.ListType() }, Syntax.ListType());
    }

    public Object report(Argument args[], Context arg1)
            throws ExtensionException, LogoException {
        PragmaticAction pragmaticAction = new PragmaticAction(
                (String) args[0].get(), (List) args[1].get(),
                (List) args[2].get());
        return pragmaticAction;
    }
}
```

## PlaceReporter.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.ArrayList;
import java.util.*;
import org.nlogo.agent.ArrayAgentSet;
import org.nlogo.agent.ArrayAgentSet.Iterator;
import org.nlogo.api.*;
import org.nlogo.nvm.Workspace;

/**
 * Takes a PragmaticAction instance and a list of SystemAgentEnvironment
 * instances and returns a list with the involved Patch instance, the
 * RealizationAction instance, the corresponding semanticAction instance, and
 * the suitability value (double)
 *
 * @author David Jonietz
 */

public class PlaceReporter extends DefaultReporter {

    public Syntax getSyntax() {
        return Syntax.reporterSyntax(
                new int[] { Syntax.WildcardType(), Syntax.ListType() },
                Syntax.ListType());
    }

    public Object report(Argument args[], Context arg1)
            throws ExtensionException, LogoException {
        PragmaticAction pragmaticAction = (PragmaticAction) args[0].get();
        return LogoList.fromJava(pragmaticAction.calculatePlace((List) args[1]
                .get()));
    }

}
```

## RealizationActionNameReporter.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.ArrayList;
import org.nlogo.agent.ArrayAgentSet;
import org.nlogo.agent.ArrayAgentSet.Iterator;
import org.nlogo.api.*;
import org.nlogo.nvm.Workspace;

/**
 * Returns the name of an instance of RealizationAction.
 *
 * @author David Jonietz
 */

public class RealizationActionNameReporter extends DefaultReporter {

    public Syntax getSyntax() {
        return Syntax.reporterSyntax(new int[] { Syntax.WildcardType() },
                Syntax.ListType());
    }

    public Object report(Argument args[], Context arg1)
            throws ExtensionException, LogoException {
        RealizationAction realizationAction = (RealizationAction) args[0].get();
        return realizationAction.name;
    }
}
```

## SemanticActionNameReporter.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.ArrayList;
import org.nlogo.agent.ArrayAgentSet;
import org.nlogo.agent.ArrayAgentSet.Iterator;
import org.nlogo.api.*;
import org.nlogo.nvm.Workspace;

/**
 * Returns the name of an instance of SemanticAction.
 *
 * @author David Jonietz
 */

public class SemanticActionNameReporter extends DefaultReporter {

    public Syntax getSyntax() {
        return Syntax.reporterSyntax(new int[] { Syntax.WildcardType() },
                Syntax.ListType());
    }

    public Object report(Argument args[], Context arg1)
            throws ExtensionException, LogoException {
        SemanticAction semanticAction = (SemanticAction) args[0].get();
        return semanticAction.name;
    }
}
```

## PragmaticActionNameReporter.java

```java
//
// Copyright: David Jonietz, 2015
//

import java.util.ArrayList;
import org.nlogo.agent.ArrayAgentSet;
import org.nlogo.agent.ArrayAgentSet.Iterator;
import org.nlogo.api.*;
import org.nlogo.nvm.Workspace;

/**
 * Returns the name of an instance of PragmaticAction.
 *
 * @author David Jonietz
 */

public class PragmaticActionNameReporter extends DefaultReporter {

    public Syntax getSyntax() {
        return Syntax.reporterSyntax(new int[] { Syntax.WildcardType() },
                Syntax.ListType());
    }

    public Object report(Argument args[], Context arg1)
            throws ExtensionException, LogoException {
        PragmaticAction pragmaticAction = (PragmaticAction) args[0].get();
        return pragmaticAction.name;
    }
}
```

## ExtensionManager.java

```java
//
// Copyright: David Jonietz, 2015
//

import org.nlogo.api.*;

/**
 * This is the class manager. It defines the primitives for this extension.
 *
 * @autor David Jonietz
 */

public class ExtensionManager extends DefaultClassManager {

    @Override
    public java.util.List<String> additionalJars() {
        java.util.List<String> list = new java.util.ArrayList<String>();
        list.add("jgrapht-core-0.9.0.jar");
        return list;
    }

    public void load(PrimitiveManager primitiveManager) {
        primitiveManager.addPrimitive("createSystemAgentEnvironment",
                new SystemAgentEnvironmentFactory());
        primitiveManager.addPrimitive("createSemanticAction",
                new SemanticActionFactory());
        primitiveManager.addPrimitive("createRealizationAction",
                new RealizationActionFactory());
        primitiveManager.addPrimitive("createPragmaticAction",
                new PragmaticActionFactory());
        primitiveManager.addPrimitive("reportPlace", new PlaceReporter());
        primitiveManager.addPrimitive("reportSystemComponents",
                new SystemComponentReporter());
        primitiveManager.addPrimitive("reportSemanticActionName",
                new SemanticActionNameReporter());
        primitiveManager.addPrimitive("reportRealizationActionName",
                new RealizationActionNameReporter());
        primitiveManager.addPrimitive("reportPragmaticActionName",
                new PragmaticActionNameReporter());
    }
}

```

## LabeledEdge.java

```java
//
// Copyright: David Jonietz, 2015
//

import org.jgrapht.*;
import org.jgrapht.graph.*;

/**
 * Extends DefaultWeightedEdge by adding a label attribute (string), which is
 * used to store spatial relationships between action graph nodes.
 *
 * «
 *
 * @autor David Jonietz
 */

public class LabeledEdge extends DefaultWeightedEdge {
    public String label = new String();

    public LabeledEdge() {
        label = "";
    }

    public LabeledEdge(String setLabel) {
        label = setLabel;
    }

}

```

## B: NetLogo Code

```
;import required extensions
extensions [ GIS nw PlaceBuilder table ]

;set global variables
globals
[
  ;;network data
  nodeShapefile ;shapefile with nodes of visibility graph network
  linkShapefile ;shapefile with edges of visibility graph network

  ;;;vector data
  footprintStudyArea ;shapefile with polygonal footprint of the study area, to be used for world envelope
                     ;determination

  ;;;raster data to be used for patch attribute setting (ESRI ASCII GRID format, NoData = -9999)
  classificationRaster ;attribute classification, code:  sidewalk = 1; street = 2; crossing = 3; cycleway = 4
  slopeRaster ;attribute slope (percent)
  surfaceRaster ;attribute surface, code: 1 = pebble or dirt, grass; 2 = pavement (uneven),
                ;cobblestone (uneven, fugue > 2cm);
                ;3 = cobblestone (even, fugue > 2cm), cobblestone (uneven, fugue < 2cm);
                ;4 = cobblestone (even, fugue < 2cm);
                ;5 = pavement (even), tiles
  benchDistRaster ;includes only cells with classification = sidewalk,
                  ;cell value = path distance to nearest bench in meters
  lightDistRaster ;includes only cells with classification = sidewalk,
                  ;cell value = distance to nearest streetlight in meters
  buildVisRaster ;includes only cells with classification = sidewalk,
                 ;cell value = number of historical buildings visible
  treeVisRaster ;includes only cells with classification = sidewalk, cell value = number of trees visible
  crossSightDistRaster ;includes only cells which are potential starting points for crossing
                       ;(in direct vicinity of street)
                       ;cell value = length of longest sightline into corresponding street segment in meters
  minDistStreetRaster ;includes only cells with classification = sidewalk, cell value = distance to nearest visible
                      ;street/crossing cell in meters; if distance > 25 meters, cell value = -1
  minDistCycleRaster ;includes only cells with classification = sidewalk, cell value = distance to nearest visible cycleways
                     ;cell in meters; if distance > 25 meters, cell value = -1
  trafficSpeedRaster ;includes only cells which are potential starting points for crossing (in direct vicinity of street),
                     ;cell value = speed limit of nearest street patch
  crossingDistRaster ;includes only cells which are potential starting points for crossing (in direct vicinity of street),
                     ;cell value = minimum crossing distance (distance to nearest visible sidewalk cell on other side of road)
  patchHeightRaster ;includes only cells which have a curb or stair step, cell value = height in centimeters
  crossdestrRaster ;includes only cells which are potential starting points for crossing (in direct vicinity of street),
                   ;cell value = row number of crossing destination (nearest visible sidewalk cell on other side of road)
  crossdestcRaster ;includes only cells which are potential starting points for crossing (in direct vicinity of street),
                   ;cell value = column number of crossing destination (nearest visible sidewalk cell on other side of road)
  trafficVisRaster ;includes only cells with classification = sidewalk, cell values = 0 - 1 (pre-standardized depending on the
                   ;number of total visible street cells)
  buildDistRaster ;attribute euclidean distance to nearest unwalkable cell (e.g. buildings)

  ;;pragmatic (pA), semantic (sA) and realization (rA) actions

  ;;;pragmatic Actions
  pA_Walk ;pragmatic action for walking through the viewfield
  pA_WalkOneStep ;alternative pragmatic action for walking a distance of only one patch

  ;;;semantic Actions
  sA_AgentIsHere ;initializing action, makes sure that the agent knows its current position
  sA_AgentMove_1 ;describes a change in environmental state, a location change of the agent (one step into a neighboring patch,
                 ;the number of sequential actions correspond to viewfield length)
  sA_AgentMove_2

  sA_AgentMove_3
  sA_AgentMove_4
  sA_AgentMove_5
  sA_AgentMove_6
  sA_AgentMove_7
  sA_AgentMove_8
  sA_AgentMove_9
  sA_AgentMove_10
  sA_AgentMove_11
  sA_AgentMove_12
  sA_AgentMove_13
  sA_AgentMove_14
  sA_AgentMove_15

  sA_AgentUpdatePleasure_1 ;describes a change in agent state, how pleasurable the environment is perceived
  sA_AgentUpdatePleasure_2
  sA_AgentUpdatePleasure_3
  sA_AgentUpdatePleasure_4
  sA_AgentUpdatePleasure_5
  sA_AgentUpdatePleasure_6
  sA_AgentUpdatePleasure_7
  sA_AgentUpdatePleasure_8
  sA_AgentUpdatePleasure_9
  sA_AgentUpdatePleasure_10
  sA_AgentUpdatePleasure_11
  sA_AgentUpdatePleasure_12
  sA_AgentUpdatePleasure_13
  sA_AgentUpdatePleasure_14
  sA_AgentUpdatePleasure_15

  sA_AgentUpdateSafety_1 ;describes a change in agent state, how safe the environment is perceived
  sA_AgentUpdateSafety_2
  sA_AgentUpdateSafety_3
  sA_AgentUpdateSafety_4
  sA_AgentUpdateSafety_5
  sA_AgentUpdateSafety_6
  sA_AgentUpdateSafety_7
  sA_AgentUpdateSafety_8
  sA_AgentUpdateSafety_9
  sA_AgentUpdateSafety_10
  sA_AgentUpdateSafety_11
  sA_AgentUpdateSafety_12
  sA_AgentUpdateSafety_13
  sA_AgentUpdateSafety_14
  sA_AgentUpdateSafety_15

  sA_AgentUpdateComfort_1 ;describes a change in agent state, how comfortable the environment is perceived
  sA_AgentUpdateComfort_2
  sA_AgentUpdateComfort_3
  sA_AgentUpdateComfort_4
  sA_AgentUpdateComfort_5
  sA_AgentUpdateComfort_6
  sA_AgentUpdateComfort_7
  sA_AgentUpdateComfort_8
  sA_AgentUpdateComfort_9
  sA_AgentUpdateComfort_10
  sA_AgentUpdateComfort_11
  sA_AgentUpdateComfort_12
  sA_AgentUpdateComfort_13
  sA_AgentUpdateComfort_14
```

```
  sA_AgentCrossStreet_1 ;describes a change in environment state, a location change of the agent (jaywalking
  sA_AgentCrossStreet_2
  sA_AgentCrossStreet_3
  sA_AgentCrossStreet_4
  sA_AgentCrossStreet_5
  sA_AgentCrossStreet_6
  sA_AgentCrossStreet_7
  sA_AgentCrossStreet_8
  sA_AgentCrossStreet_9
  sA_AgentCrossStreet_10
  sA_AgentCrossStreet_11
  sA_AgentCrossStreet_12
  sA_AgentCrossStreet_13
  sA_AgentCrossStreet_14
  sA_AgentCrossStreet_15

  ;;;realization Actions
  rA_BeHere ;initializing action, describes the agent existing at a certain position
  rA_KeepBalance ;describes the action of avoiding a fall
  rA_OvercomeSlope ;describe the action of overcoming a positive or negative slope
  rA_OvercomeHeightDifference ;describe the action of stepping up or down on curbs or stair steps
  rA_AccessPatch ;describe the action of physically fitting through obstacles while walking
  rA_ReachDestination ;describe the action of moving towards the destination node
  rA_SeeGreenery ;describe the action of visually perceiving trees
  rA_SeeAestheticBuild ;describe the action of visually perceiving aesthetical buildings
  rA_KeepDistToMotorTraffic ;describe the action of keeping a distance to the street
  rA_KeepDistToCycleTraffic ;describe the action of keeping a distance to the cycleway
  rA_StayNearLight ;describe the action of staying close to streetlights
  rA_NotSeeTraffic ;describe the action of keeping a blocked view to streets
  rA_StayNearBenches ;describe the action of staying close to benches

  rA_EvaluateStreetCrossing ;describe the action of evaluating whether a crossing process makes sense
  rA_SeeTrafficSituation ;describe the action of assessing the traffic situation
  rA_ReachSidewalk ;describe the action of moving across the street to the nearest sidewalk cell

  ;;variable to hold a list with potential Origin-Destination Nodes
  potODList

  ;;variables to hold maximum values of selected patch attributes (used for standardization)
  maxValue_buildVis
  maxValue_treeVis
  maxValue_crossingDistance

  ;;variables to hold minimum values of selected patch attributes (used for standardization)
  minValue_treeVis
  minValue_buildVis

  ;;variable to hold count of re-orientation procedures in a simulation run
  numberOfReorientation

  ;;variable to count how many times a place is calculated or reorient
  generalCounter

  ;;variable to count the number of crossings
  numberOfCrossings

  ;;variable to hold all calculated places in a simulation run (for exporting the results)
  placeList



  ;;variables to serve as identifiers for individual simulation runs
  Agent_ID ;stores an agentID for each run
  Setting_ID ;stores an ID for each setting (weights, ODPair)
  SimulationRun_ID ;stores an ID for each simulation run
]

;set breeds
breed [ nodes node ] ;visibity graph network nodes
breed [ pedestrians pedestrian ] ;pedestrian agents

;set attributes of agents
links-own [ lineID linkLength ] ;lineID and linkLength correspond to linkShapefiles
nodes-own [ nodeID posInSP ] ;nodeID corresponds to nodeShapefile feature ID

patches-own
[
  classification ;classification into sidewalk, street, crossing, and cycleway
  slope ;gradient in percent
  surface ;surface structure classified into 5 types
  benchDist ;path distance to closest bench in meters
  lightDist ;euclidean distance to closest streetlight in meters
  buildVis ;number of historical protected buildings visible
  treeVis ;number of visible trees
  crossSightDist ;longest sightline from which approaching cars are visible in meters
  minDistStreet ;euclidean distance to closest, unblocked street patch in meters
  minDistCycle ;euclidean distance to closest, unblocked cycleway patch in meters
  trafficSpeed ;allowed maximum speed in km/h
  crossingDistance ;minimum crossing distance in meters
  crossDestRow ;row of nearest crossing destination
  crossDestCol ;column of nearest crossing destination
  patchHeight ;curb height in centimeters
  trafficVis ;amount of street visible (standardized values from 0-1)
  buildDist ;distance to nearest unwalkable cell in meters
]

pedestrians-own
[
  beliefs ;represents the knowledge-base of the agent
  intentions ;defines the intended agent behavior
  desire ;defines the agent desire
  OD-Pair ;its origin-destination pair

  minSurface ;minimum acceptable surface quality, refers to code used in surfaceRaster
  maxSlope ;maximum slope, critical threshold value
  width ;body width in meters
  waveringMovement ;additional width due to dynamic body movements in meters
  shyAwayDistTraffic ;preferred minimum buffer distance to streets in meters
  shyAwayDistCyc ;preferred minimum buffer distance to cycleways in meters
  shyAwayDistWall ;preferred minimum buffer distance to walls in meters
  maxLight ;maximum euclidean distance to street light in meters
  maxBench ;maximum path distance to bench in meters
  minTimeKerbDelay ;time spend for gap detection in seconds
  walkingSpeed ;walking speed in km/h
  maxHeight ;refers to the height the pedestrian is able to surmount in meters (e.g. a curb)

  weight_sA_AgentMove ;weight for sA_AgentMove
  weight_sA_AgentUpdateSafety ;weight for sA_AgentUpdateSafety
  weight_sA_AgentUpdatePleasure ;weight for sA_AgentUpdatePleasure
  weight_sA_AgentUpdateComfort ;weight for sA_AgentUpdateComfort
```

```
  weight_sA_AgentCrossStreet ;weight for sA_AgentCrossStreet
  rA_KeepBalance_weight_sA_AgentMove ;weight for rA_KeepBalance (sA_AgentMove)
  rA_OvercomeSlope_weight_sA_AgentMove ;weight for rA_OvercomeSlope (sA_AgentMove)
  rA_OvercomeHeightDifference_weight_sA_AgentMove ;weight for rA_OvercomeHeightDifference (sA_AgentMove)
  rA_AccessPatch_weight_sA_AgentMove ;weight for rA_AccessPatch (sA_AgentMove)
  rA_ReachDestination_weight_sA_AgentMove ;weight for rA_ReachDestination (sA_AgentMove)
  rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety ;weight for rA_KeepDistToMotorTraffic (sA_AgentUpdateSafety
  rA_StayNearLight_weight_sA_AgentUpdateSafety ;weight for rA_StayNearLight in the context of sA_AgentUpdateSafety
  rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety ;weight for rA_KeepDistToCycleTraffic (sA_AgentUpdateSafety
  rA_SeeGreenery_weight_sA_AgentUpdatePleasure ;weight for rA_SeeGreenery (sA_AgentUpdatePleasure)
  rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure ;weight for rA_SeeAestheticBuild (sA_AgentUpdatePleasure)
  rA_NotSeeTraffic_weight_sA_AgentUpdateComfort ;weight for rA_NotSeeTraffic (sA_AgentUpdateComfort)
  rA_StayNearBenches_weight_sA_AgentUpdateComfort ;weight for rA_StayNearBenches (sA_AgentUpdateComfort)
]


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;; CODE FOR CREATING THE ENVIRONMENTAL MODEL
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;create the world
to loadData

  ;;Load spatial data
  set nodeShapefile gis:load-dataset word workspace "/NetworkData/VisGraph_Nodes_RH.shp"
  set linkShapefile gis:load-dataset word workspace "/NetworkData/VisGraph_Edges_RH.shp"
  set classificationRaster gis:load-dataset word workspace "/ASCII Grids/classification_rh.asc"
  set slopeRaster gis:load-dataset word workspace "/ASCII Grids/slope_rh.asc"
  set surfaceRaster gis:load-dataset word workspace "/ASCII Grids/surface_rh.asc"
  set benchDistRaster gis:load-dataset word workspace "/ASCII Grids/benchdist_rh.asc"
  set lightDistRaster gis:load-dataset word workspace "/ASCII Grids/lightdist_rh.asc"
  set buildVisRaster gis:load-dataset word workspace "/ASCII Grids/buildvis_rh.asc"
  set treeVisRaster gis:load-dataset word workspace "/ASCII Grids/treevis_rh.asc"
  set crossSightDistRaster gis:load-dataset word workspace "/ASCII Grids/crossmaxvisdist_rh.asc"
  set minDistStreetRaster gis:load-dataset word workspace "/ASCII Grids/mindiststr_rh.asc"
  set minDistCycleRaster gis:load-dataset word workspace "/ASCII Grids/mindistcyc_rh.asc"
  set trafficSpeedRaster gis:load-dataset word workspace "/ASCII Grids/maxspeed_rh.asc"
  set patchHeightRaster gis:load-dataset word workspace "/ASCII Grids/patchheight_rh.asc"
  set crossingDistRaster gis:load-dataset word workspace "/ASCII Grids/crossdist_rh.asc"
  set crossdestrRaster gis:load-dataset word workspace "/ASCII Grids/crossdestrow_rh.asc"
  set crossdestcRaster gis:load-dataset word workspace "/ASCII Grids/crossdestcol_rh.asc"
  set trafficVisRaster gis:load-dataset word workspace "/ASCII Grids/streetvis_rh.asc"
  set buildDistRaster gis:load-dataset word workspace "/ASCII Grids/builddist_rh.asc"
  set footprintStudyArea gis:load-dataset word workspace "/Shapefiles/footprintStudyArea.shp"

end

;builds environmental model, set additional parameters for simulation
to setup

  ;;set world envelope
  gis:set-world-envelope (gis:envelope-of footprintStudyArea)

  ;;set patch attributes based on GIS raster data
  gis:apply-raster classificationRaster classification
  gis:apply-raster slopeRaster slope
  gis:apply-raster surfaceRaster surface
  gis:apply-raster benchDistRaster benchDist
  gis:apply-raster lightDistRaster lightDist
  gis:apply-raster buildVisRaster buildVis

  gis:apply-raster treeVisRaster treeVis
  gis:apply-raster crossSightDistRaster crossSightDist
  gis:apply-raster minDistStreetRaster minDistStreet
  gis:apply-raster minDistCycleRaster minDistCycle
  gis:apply-raster trafficSpeedRaster trafficSpeed
  gis:apply-raster crossingDistRaster crossingDistance
  gis:apply-raster patchHeightRaster patchHeight
  gis:apply-raster crossdestrRaster crossDestRow
  gis:apply-raster crossdestcRaster crossDestCol
  gis:apply-raster trafficVisRaster trafficVis
  gis:apply-raster buildDistRaster buildDist

  ;;Check for NaN values (noData format used by GIS-extension) and replace with -9999
  ;;Test for being neither smaller nor larger than 0 is the only way to to this.
  ask patches
  [
    if not ((classification <= 0) or (classification >= 0)) [set classification -9999]
    if not ((slope <= 0) or (slope >= 0)) [set slope -9999]
    if not ((surface <= 0) or (surface >= 0)) [set surface -9999]
    if not ((benchDist <= 0) or (benchDist >= 0)) [set benchDist -9999]
    if not ((lightDist <= 0) or (lightDist >= 0)) [set lightDist -9999]
    if not ((buildVis <= 0) or (buildVis >= 0)) [set buildVis -9999]
    if not ((treeVis <= 0) or (treeVis >= 0)) [set treeVis -9999]
    if not ((crossSightDist <= 0) or (crossSightDist >= 0)) [set crossSightDist -9999]
    if not ((minDistStreet <= 0) or (minDistStreet >= 0)) [set minDistStreet -9999]
    if not ((minDistCycle <= 0) or (minDistCycle >= 0)) [set minDistCycle -9999]
    if not ((trafficSpeed <= 0) or (trafficSpeed >= 0)) [set trafficSpeed -9999]
    if not ((crossingDistance <= 0) or (crossingDistance >= 0)) [set crossingDistance -9999]
    if not ((patchHeight <= 0) or (patchHeight >= 0)) [set patchHeight -9999]
    if not ((crossDestRow <= 0) or (crossDestRow >= 0)) [set crossDestRow -9999]
    if not ((crossDestCol <= 0) or (crossDestCol >= 0)) [set crossDestCol -9999]
    if not ((trafficVis <= 0) or (trafficVis >= 0)) [set trafficVis -9999]
    if not ((buildDist <= 0) or (buildDist >= 0)) [set buildDist -9999]
  ]

  ;;set patch colors
  ask patches
  [
    if classification = -9999 ;NoData
    [ set pcolor white ]

    if classification = 1 ;sidewalk
    [ set pcolor 7 ]

    if classification = 2 ;street
    [ set pcolor 2 ]

    if classification = 3 ;crossing
    [ set pcolor 4 ]

    if classification = 4 ;cycleway
    [ set pcolor 5 ]
  ]

  ;;retrieve and store IDs of potential origin or destination nodes in potDOList
  set potODList []

  foreach gis:feature-list-of nodeShapefile
  [
    if gis:property-value ? "potOD" = 1
```

```
  ]

  ;;calculate maximum values for selected patch attributes (used for standardization)
  set maxValue_buildVis [buildVis] of max-one-of patches [buildVis]
  set maxValue_treeVis [treeVis] of max-one-of patches [treeVis]
  set maxValue_crossingDistance [crossingDistance] of max-one-of patches [crossingDistance]

  ;;calculate minimum Values for for selected patch attributes (used for standardization)
  set minValue_treeVis [treeVis] of min-one-of patches with [treeVis != -9999] [treevis]
  set minValue_buildVis [buildVis] of min-one-of patches with [buildVis != -9999] [buildVis]

  ;;reset parameters for sensitivity analyses
  set placeList []
  set Agent_ID 0
  set Setting_ID 0
  set SimulationRun_ID 0

end


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;; CODE FOR CREATING THE INITIAL BELIEFS OF THE AGENTS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;method for creating network knowledge for agent
to-report createNetwork [pedestrian]

;;create nodes based on nodeShapefile
foreach gis:feature-list-of nodeShapefile
 [
  let location gis:location-of (first(first(gis:vertex-lists-of ?))) ;get coordinates from nodeShapefile
  let feat ?

  create-nodes 1 ;create a node for each feature and place at coordinates
  [
    set xcor item 0 location
    set ycor item 1 location
    set shape "circle"
    set color grey
    set size 3
    set nodeID gis:property-value ? "ID" ;get value for attribute nodeID from field "ID" in shapefile
  ]
 ]

;;create links based on linkShapefile
foreach gis:feature-list-of linkShapefile
[
 let shapeLength gis:property-value ? "SHAPE_LENG" ;get length of linkShapefile-Feature
 let ObjectID gis:property-value ? "ID" ;get ID of line feature

 ;;identify start- and endnode for each edge from attribute-table of linkShapefile-Feature
 let agent1 node gis:property-value ? "nodeFrom"
 let agent2 node gis:property-value ? "nodeTo"

 ;;create link between start- and endnode
 ask agent1
 [
   create-link-with agent2
   [

     set linkLength shapeLength ;add length of linkShapefile-Feature as attribute to link
     set lineID ObjectID ;add ID of linkShapefile-Feature as attribute to link
   ]
 ]
]

;;report the network as list of node- and link-agentset
report (list nodes links)

end


;method for routing on the strategic level (shortest path calculation through visibility graph)
to-report findShortestPath [pedestrian]

  ;;get start and end node from attribute OD-Pair of pedestrian
  let node_start first [OD-Pair] of pedestrian
  let node_end last [OD-Pair] of pedestrian

  ;;retrieve network from pedestrian's beliefs
  let network [read-first-belief-of-type "network"] of pedestrian

  ;;initialize shortest path variable
  let shortestPath nobody

  ;;calculate shortest path with network extension
  ask pedestrian
  [
    ask one-of nodes with [nodeID = node_start]
    [ set shortestPath nw:weighted-path-to one-of nodes with [nodeID = node_end] "linkLength" ]
  ]

  ;;identify nodes which are part of shortest Path and add their position in the shortest path
  let nodeIDList []
  let SPNodes []
  let destinationNode one-of nodes with [nodeID = last [OD-Pair] of pedestrian]

  foreach shortestPath
   [
     set SPNodes lput [both-ends] of ? SPNodes
     ask ?
     [
       ask both-ends
       [ set nodeIDList lput nodeID nodeIDList ]
     ]
   ]
  let nodeSet nodes with [member? nodeID nodeIDList = True]
  let nodeList []

  ask nodeSet
  [
    set nodeList lput self nodeList

    foreach SPNodes
    [
      if member? self ?
      [ set posInSP position ? SPNodes ]
    ]

    ;;create sightlines from the agent to each node of shortest Path, except the origin node
```

```
    [
      let distDest distance destinationNode ;for each node of the shortest path, calculate distance to the destination
      set label nodeID
      set label-color red

      create-link-with pedestrian
      [
        set color yellow
      ]
    ]
  ]

  ;;report shortest path (list of link-agentset and node-agentset)
  report (list shortestPath nodeSet)

end

;procedure for creating hierarchical action structure
to-report createActionStructure [ pedestrian ]

  ;;create realization Actions
  set rA_BeHere PlaceBuilder:createRealizationAction "rA_BeHere"
  set rA_KeepBalance PlaceBuilder:createRealizationAction "rA_KeepBalance"
  set rA_OvercomeSlope PlaceBuilder:createRealizationAction "rA_OvercomeSlope"
  set rA_OvercomeHeightDifference PlaceBuilder:createRealizationAction "rA_OvercomeHeightDifference"
  set rA_AccessPatch PlaceBuilder:createRealizationAction "rA_AccessPatch"
  set rA_ReachDestination PlaceBuilder:createRealizationAction "rA_ReachDestination"
  set rA_SeeGreenery PlaceBuilder:createRealizationAction "rA_SeeGreenery"
  set rA_SeeAestheticBuild PlaceBuilder:createRealizationAction "rA_SeeAestheticBuild"
  set rA_KeepDistToMotorTraffic  PlaceBuilder:createRealizationAction "rA_KeepDistToMotorTraffic"
  set rA_KeepDistToCycleTraffic  PlaceBuilder:createRealizationAction "rA_KeepDistToCycleTraffic"
  set rA_StayNearLight PlaceBuilder:createRealizationAction "rA_StayNearLight"
  set rA_NotSeeTraffic PlaceBuilder:createRealizationAction "rA_NotSeeTraffic"
  set rA_StayNearBenches PlaceBuilder:createRealizationAction "rA_StayNearBenches"
  set rA_EvaluateStreetCrossing PlaceBuilder:createRealizationAction "rA_EvaluateStreetCrossing"
  set rA_SeeTrafficSituation PlaceBuilder:createRealizationAction "rA_SeeTrafficSituation"
  set rA_ReachSidewalk PlaceBuilder:createRealizationAction "rA_ReachSidewalk"

  ;;create semantic actions depending on the viewDistance
  if viewDistance = 5
  [
    set sA_AgentIsHere PlaceBuilder:createSemanticAction "sA_AgentIsHere" (list (list rA_BeHere 1)) []

    set sA_AgentMove_1 PlaceBuilder:createSemanticAction "sA_AgentMove_1"
      (list
        (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
        (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
        (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
        (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
        (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
      )
      (list
        (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
        (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
        (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
        (list rA_AccessPatch rA_ReachDestination "sameSystem")
      )
    set sA_AgentMove_2 PlaceBuilder:createSemanticAction "sA_AgentMove_2"
      (list

        (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
        (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
        (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
        (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
        (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
      )
      (list
        (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
        (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
        (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
        (list rA_AccessPatch rA_ReachDestination "sameSystem")
      )
    set sA_AgentMove_3 PlaceBuilder:createSemanticAction "sA_AgentMove_3"
      (list
        (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
        (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
        (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
        (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
        (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
      )
      (list
        (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
        (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
        (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
        (list rA_AccessPatch rA_ReachDestination "sameSystem")
      )
    set sA_AgentMove_4 PlaceBuilder:createSemanticAction "sA_AgentMove_4"
      (list
        (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
        (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
        (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
        (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
        (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
      )
      (list
        (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
        (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
        (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
        (list rA_AccessPatch rA_ReachDestination "sameSystem")
      )
    set sA_AgentMove_5 PlaceBuilder:createSemanticAction "sA_AgentMove_5"
      (list
        (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
        (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
        (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
        (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
        (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
      )
      (list
        (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
        (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
        (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
        (list rA_AccessPatch rA_ReachDestination "sameSystem")
      )

    set sA_AgentUpdatePleasure_1 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_1"
      (list
        (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
        (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
      )
```

```
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_2 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_2"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_3 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_3"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_4 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_4"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_5 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_5"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))

set sA_AgentUpdateSafety_1 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_1"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_2 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_2"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_3 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_3"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_4 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_4"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)


    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_5 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_5"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )

set sA_AgentUpdateComfort_1 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_1"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_2 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_2"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_3 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_3"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_4 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_4"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_5 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_5"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))

set sA_AgentCrossStreet_1 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_1"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_2 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_2"
  (list
```

```
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_3 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_3"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_4 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_4"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_5 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_5"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )

;;create pragmatic actions
set pA_Walk PlaceBuilder:createPragmaticAction "pA_Walk"
  (list
    (list sA_AgentIsHere 1)
    (list sA_AgentMove_1 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_1 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_1 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_1 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_2 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_2 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_2 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_2 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_3 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_3 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_3 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_3 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_4 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_4 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_4 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_4 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_5 weight_sA_AgentMove)

    (list sA_AgentUpdatePleasure_5 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_5 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_5 weight_sA_AgentUpdateComfort)
    (list sA_AgentCrossStreet_1 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_2 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_3 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_4 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_5 weight_sA_AgentCrossStreet)
  )

  (list
    (list sA_AgentIsHere sA_AgentMove_1 "neighbors")
    (list sA_AgentMove_1 sA_AgentCrossStreet_1 "sameSystem")
    (list sA_AgentMove_1 sA_AgentUpdatePleasure_1 "sameSystem")
    (list sA_AgentUpdatePleasure_1 sA_AgentUpdateSafety_1 "sameSystem")
    (list sA_AgentUpdateSafety_1 sA_AgentUpdateComfort_1 "sameSystem")
    (list sA_AgentUpdateComfort_1 sA_AgentMove_2 "neighbors")
    (list sA_AgentMove_2 sA_AgentCrossStreet_2 "sameSystem")
    (list sA_AgentMove_2 sA_AgentUpdatePleasure_2 "sameSystem")
    (list sA_AgentUpdatePleasure_2 sA_AgentUpdateSafety_2 "sameSystem")
    (list sA_AgentUpdateSafety_2 sA_AgentUpdateComfort_2 "sameSystem")
    (list sA_AgentUpdateComfort_2 sA_AgentMove_3 "neighbors")
    (list sA_AgentMove_3 sA_AgentCrossStreet_3 "sameSystem")
    (list sA_AgentMove_3 sA_AgentUpdatePleasure_3 "sameSystem")
    (list sA_AgentUpdatePleasure_3 sA_AgentUpdateSafety_3 "sameSystem")
    (list sA_AgentUpdateSafety_3 sA_AgentUpdateComfort_3 "sameSystem")
    (list sA_AgentUpdateComfort_3 sA_AgentMove_4 "neighbors")
    (list sA_AgentMove_4 sA_AgentCrossStreet_4 "sameSystem")
    (list sA_AgentMove_4 sA_AgentUpdatePleasure_4 "sameSystem")
    (list sA_AgentUpdatePleasure_4 sA_AgentUpdateSafety_4 "sameSystem")
    (list sA_AgentUpdateSafety_4 sA_AgentUpdateComfort_4 "sameSystem")
    (list sA_AgentUpdateComfort_4 sA_AgentMove_5 "neighbors")
    (list sA_AgentMove_5 sA_AgentCrossStreet_5 "sameSystem")
    (list sA_AgentMove_5 sA_AgentUpdatePleasure_5 "sameSystem")
    (list sA_AgentUpdatePleasure_5 sA_AgentUpdateSafety_5 "sameSystem")
    (list sA_AgentUpdateSafety_5 sA_AgentUpdateComfort_5 "sameSystem")
  )
set pA_WalkOneStep PlaceBuilder:createPragmaticAction "pA_WalkOneStep"
(list
  (list sA_AgentIsHere 1)
  (list sA_AgentMove_1 weight_sA_AgentMove)
  (list sA_AgentUpdatePleasure_1 weight_sA_AgentUpdatePleasure)
  (list sA_AgentUpdateSafety_1 weight_sA_AgentUpdateSafety)
  (list sA_AgentUpdateComfort_1 weight_sA_AgentUpdateComfort)
)
(list
  (list sA_AgentIsHere sA_AgentMove_1 "neighbors")
    (list sA_AgentMove_1 sA_AgentUpdatePleasure_1 "sameSystem")
    (list sA_AgentUpdatePleasure_1 sA_AgentUpdateSafety_1 "sameSystem")
    (list sA_AgentUpdateSafety_1 sA_AgentUpdateComfort_1 "sameSystem")
)


: viewDistance = 15

;;create semantic actions
set sA_AgentIsHere PlaceBuilder:createSemanticAction "sA_AgentIsHere" (list (list rA_BeHere :

set sA_AgentMove_1 PlaceBuilder:createSemanticAction "sA_AgentMove_1"
  (list
```

```
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_2 PlaceBuilder:createSemanticAction "sA_AgentMove_2"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_3 PlaceBuilder:createSemanticAction "sA_AgentMove_3"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_4 PlaceBuilder:createSemanticAction "sA_AgentMove_4"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_5 PlaceBuilder:createSemanticAction "sA_AgentMove_5"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)

    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_6 PlaceBuilder:createSemanticAction "sA_AgentMove_6"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_7 PlaceBuilder:createSemanticAction "sA_AgentMove_7"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_8 PlaceBuilder:createSemanticAction "sA_AgentMove_8"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_9 PlaceBuilder:createSemanticAction "sA_AgentMove_9"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
```

```
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_10 PlaceBuilder:createSemanticAction "sA_AgentMove_10"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_11 PlaceBuilder:createSemanticAction "sA_AgentMove_11"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_12 PlaceBuilder:createSemanticAction "sA_AgentMove_12"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_13 PlaceBuilder:createSemanticAction "sA_AgentMove_13"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_14 PlaceBuilder:createSemanticAction "sA_AgentMove_14"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )
set sA_AgentMove_15 PlaceBuilder:createSemanticAction "sA_AgentMove_15"
  (list
    (list rA_KeepBalance rA_KeepBalance_weight_sA_AgentMove)
    (list rA_OvercomeSlope rA_OvercomeSlope_weight_sA_AgentMove)
    (list rA_OvercomeHeightDifference rA_OvercomeHeightDifference_weight_sA_AgentMove)
    (list rA_AccessPatch rA_AccessPatch_weight_sA_AgentMove)
    (list rA_ReachDestination rA_ReachDestination_weight_sA_AgentMove)
  )
  (list
    (list rA_KeepBalance rA_OvercomeSlope "sameSystem")
    (list rA_OvercomeSlope rA_OvercomeHeightDifference "sameSystem")
    (list rA_OvercomeHeightDifference rA_AccessPatch "sameSystem")
    (list rA_AccessPatch rA_ReachDestination "sameSystem")
  )

set sA_AgentUpdatePleasure_1 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_1"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_2 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_2"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_3 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_3"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_4 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_4"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_5 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_5"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_6 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_6"
```

```
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_7 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_7"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_8 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_8"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_9 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_9"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_10 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_10"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_11 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_11"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_12 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_12"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_13 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_13"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_14 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_14"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))
set sA_AgentUpdatePleasure_15 PlaceBuilder:createSemanticAction "sA_AgentUpdatePleasure_15"
  (list
    (list rA_SeeGreenery rA_SeeGreenery_weight_sA_AgentUpdatePleasure)
    (list rA_SeeAestheticBuild rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure)
  )
  (list (list rA_SeeGreenery rA_SeeAestheticBuild "sameSystem"))

set sA_AgentUpdateSafety_1 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_1"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_2 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_2"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_3 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_3"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_4 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_4"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_5 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_5"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_6 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_6"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
```

```
set sA_AgentUpdateSafety_7 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_7"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_8 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_8"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_9 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_9"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_10 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_10"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_11 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_11"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_12 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_12"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )

set sA_AgentUpdateSafety_13 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_13"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_14 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_14"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )
set sA_AgentUpdateSafety_15 PlaceBuilder:createSemanticAction "sA_AgentUpdateSafety_15"
  (list
    (list rA_KeepDistToMotorTraffic rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety)
    (list rA_KeepDistToCycleTraffic rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety)
    (list rA_StayNearLight rA_StayNearLight_weight_sA_AgentUpdateSafety)
  )
  (list
    (list rA_KeepDistToMotorTraffic  rA_KeepDistToCycleTraffic "sameSystem")
    (list rA_KeepDistToCycleTraffic  rA_StayNearLight "sameSystem")
  )

set sA_AgentUpdateComfort_1 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_1"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_2 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_2"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_3 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_3"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_4 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_4"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_5 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_5"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
```

```
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_6 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_6"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_7 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_7"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_8 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_8"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_9 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_9"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_10 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_10"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_11 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_11"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_12 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_12"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_13 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_13"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_14 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_14"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )
  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))
set sA_AgentUpdateComfort_15 PlaceBuilder:createSemanticAction "sA_AgentUpdateComfort_15"
  (list
    (list rA_NotSeeTraffic rA_NotSeeTraffic_weight_sA_AgentUpdateComfort)
    (list rA_StayNearBenches rA_StayNearBenches_weight_sA_AgentUpdateComfort)
  )

  (list (list rA_NotSeeTraffic rA_StayNearBenches "sameSystem"))

set sA_AgentCrossStreet_1 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_1"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_2 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_2"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_3 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_3"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_4 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_4"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_5 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_5"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_6 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_6"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
```

```
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_7 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_7"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_8 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_8"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_9 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_9"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_10 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_10"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_11 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_11"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_12 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_12"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")

    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_13 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_13"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_14 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_14"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )
set sA_AgentCrossStreet_15 PlaceBuilder:createSemanticAction "sA_AgentCrossStreet_15"
  (list
    (list rA_EvaluateStreetCrossing 1)
    (list rA_SeeTrafficSituation 1)
    (list rA_ReachSidewalk 1)
  )
  (list
    (list rA_EvaluateStreetCrossing rA_SeeTrafficSituation "sameSystem")
    (list rA_SeeTrafficSituation rA_ReachSidewalk "sameSystem")
  )

;;create pragmatic actions
set pA_Walk PlaceBuilder:createPragmaticAction "pA_Walk"
(list
  (list sA_AgentIsHere 1)
  (list sA_AgentMove_1 weight_sA_AgentMove)
  (list sA_AgentUpdatePleasure_1 weight_sA_AgentUpdatePleasure)
  (list sA_AgentUpdateSafety_1 weight_sA_AgentUpdateSafety)
  (list sA_AgentUpdateComfort_1 weight_sA_AgentUpdateComfort)
  (list sA_AgentMove_2 weight_sA_AgentMove)
  (list sA_AgentUpdatePleasure_2 weight_sA_AgentUpdatePleasure)
  (list sA_AgentUpdateSafety_2 weight_sA_AgentUpdateSafety)
  (list sA_AgentUpdateComfort_2 weight_sA_AgentUpdateComfort)
  (list sA_AgentMove_3 weight_sA_AgentMove)
  (list sA_AgentUpdatePleasure_3 weight_sA_AgentUpdatePleasure)
  (list sA_AgentUpdateSafety_3 weight_sA_AgentUpdateSafety)
  (list sA_AgentUpdateComfort_3 weight_sA_AgentUpdateComfort)
  (list sA_AgentMove_4 weight_sA_AgentMove)
  (list sA_AgentUpdatePleasure_4 weight_sA_AgentUpdatePleasure)
  (list sA_AgentUpdateSafety_4 weight_sA_AgentUpdateSafety)
  (list sA_AgentUpdateComfort_4 weight_sA_AgentUpdateComfort)
  (list sA_AgentMove_5 weight_sA_AgentMove)
  (list sA_AgentUpdatePleasure_5 weight_sA_AgentUpdatePleasure)
  (list sA_AgentUpdateSafety_5 weight_sA_AgentUpdateSafety)
  (list sA_AgentUpdateComfort_5 weight_sA_AgentUpdateComfort)
  (list sA_AgentMove_6 weight_sA_AgentMove)
  (list sA_AgentUpdatePleasure_6 weight_sA_AgentUpdatePleasure)
  (list sA_AgentUpdateSafety_6 weight_sA_AgentUpdateSafety)
```

```
    (list sA_AgentUpdateComfort_6 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_7 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_7 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_7 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_7 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_8 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_8 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_8 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_8 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_9 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_9 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_9 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_9 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_10 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_10 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_10 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_10 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_11 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_11 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_11 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_11 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_12 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_12 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_12 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_12 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_13 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_13 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_13 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_13 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_14 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_14 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_14 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_14 weight_sA_AgentUpdateComfort)
    (list sA_AgentMove_15 weight_sA_AgentMove)
    (list sA_AgentUpdatePleasure_15 weight_sA_AgentUpdatePleasure)
    (list sA_AgentUpdateSafety_15 weight_sA_AgentUpdateSafety)
    (list sA_AgentUpdateComfort_15 weight_sA_AgentUpdateComfort)
    (list sA_AgentCrossStreet_1 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_2 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_3 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_4 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_5 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_6 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_7 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_8 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_9 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_10 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_11 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_12 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_13 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_14 weight_sA_AgentCrossStreet)
    (list sA_AgentCrossStreet_15 weight_sA_AgentCrossStreet)
)
(list
    (list sA_AgentIsHere sA_AgentMove_1 "neighbors")
    (list sA_AgentMove_1 sA_AgentCrossStreet_1 "sameSystem")
    (list sA_AgentMove_1 sA_AgentUpdatePleasure_1 "sameSystem")
    (list sA_AgentUpdatePleasure_1 sA_AgentUpdateSafety_1 "sameSystem")
    (list sA_AgentUpdateSafety_1 sA_AgentUpdateComfort_1 "sameSystem")
    (list sA_AgentUpdateComfort_1 sA_AgentMove_2 "neighbors")

    (list sA_AgentMove_2 sA_AgentCrossStreet_2 "sameSystem")
    (list sA_AgentMove_2 sA_AgentUpdatePleasure_2 "sameSystem")
    (list sA_AgentUpdatePleasure_2 sA_AgentUpdateSafety_2 "sameSystem")
    (list sA_AgentUpdateSafety_2 sA_AgentUpdateComfort_2 "sameSystem")
    (list sA_AgentUpdateComfort_2 sA_AgentMove_3 "neighbors")
    (list sA_AgentMove_3 sA_AgentCrossStreet_3 "sameSystem")
    (list sA_AgentMove_3 sA_AgentUpdatePleasure_3 "sameSystem")
    (list sA_AgentUpdatePleasure_3 sA_AgentUpdateSafety_3 "sameSystem")
    (list sA_AgentUpdateSafety_3 sA_AgentUpdateComfort_3 "sameSystem")
    (list sA_AgentUpdateComfort_3 sA_AgentMove_4 "neighbors")
    (list sA_AgentMove_4 sA_AgentCrossStreet_4 "sameSystem")
    (list sA_AgentMove_4 sA_AgentUpdatePleasure_4 "sameSystem")
    (list sA_AgentUpdatePleasure_4 sA_AgentUpdateSafety_4 "sameSystem")
    (list sA_AgentUpdateSafety_4 sA_AgentUpdateComfort_4 "sameSystem")
    (list sA_AgentUpdateComfort_4 sA_AgentMove_5 "neighbors")
    (list sA_AgentMove_5 sA_AgentCrossStreet_5 "sameSystem")
    (list sA_AgentMove_5 sA_AgentUpdatePleasure_5 "sameSystem")
    (list sA_AgentUpdatePleasure_5 sA_AgentUpdateSafety_5 "sameSystem")
    (list sA_AgentUpdateSafety_5 sA_AgentUpdateComfort_5 "sameSystem")
    (list sA_AgentUpdateComfort_5 sA_AgentMove_6 "neighbors")
    (list sA_AgentMove_6 sA_AgentCrossStreet_6 "sameSystem")
    (list sA_AgentMove_6 sA_AgentUpdatePleasure_6 "sameSystem")
    (list sA_AgentUpdatePleasure_6 sA_AgentUpdateSafety_6 "sameSystem")
    (list sA_AgentUpdateSafety_6 sA_AgentUpdateComfort_6 "sameSystem")
    (list sA_AgentUpdateComfort_6 sA_AgentMove_7 "neighbors")
    (list sA_AgentMove_7 sA_AgentCrossStreet_7 "sameSystem")
    (list sA_AgentMove_7 sA_AgentUpdatePleasure_7 "sameSystem")
    (list sA_AgentUpdatePleasure_7 sA_AgentUpdateSafety_7 "sameSystem")
    (list sA_AgentUpdateSafety_7 sA_AgentUpdateComfort_7 "sameSystem")
    (list sA_AgentUpdateComfort_7 sA_AgentMove_8 "neighbors")
    (list sA_AgentMove_8 sA_AgentCrossStreet_8 "sameSystem")
    (list sA_AgentMove_8 sA_AgentUpdatePleasure_8 "sameSystem")
    (list sA_AgentUpdatePleasure_8 sA_AgentUpdateSafety_8 "sameSystem")
    (list sA_AgentUpdateSafety_8 sA_AgentUpdateComfort_8 "sameSystem")
    (list sA_AgentUpdateComfort_8 sA_AgentMove_9 "neighbors")
    (list sA_AgentMove_9 sA_AgentCrossStreet_9 "sameSystem")
    (list sA_AgentMove_9 sA_AgentUpdatePleasure_9 "sameSystem")
    (list sA_AgentUpdatePleasure_9 sA_AgentUpdateSafety_9 "sameSystem")
    (list sA_AgentUpdateSafety_9 sA_AgentUpdateComfort_9 "sameSystem")
    (list sA_AgentUpdateComfort_9 sA_AgentMove_10 "neighbors")
    (list sA_AgentMove_10 sA_AgentCrossStreet_10 "sameSystem")
    (list sA_AgentMove_10 sA_AgentUpdatePleasure_10 "sameSystem")
    (list sA_AgentUpdatePleasure_10 sA_AgentUpdateSafety_10 "sameSystem")
    (list sA_AgentUpdateSafety_10 sA_AgentUpdateComfort_10 "sameSystem")
    (list sA_AgentUpdateComfort_10 sA_AgentMove_11 "neighbors")
    (list sA_AgentMove_11 sA_AgentCrossStreet_11 "sameSystem")
    (list sA_AgentMove_11 sA_AgentUpdatePleasure_11 "sameSystem")
    (list sA_AgentUpdatePleasure_11 sA_AgentUpdateSafety_11 "sameSystem")
    (list sA_AgentUpdateSafety_11 sA_AgentUpdateComfort_11 "sameSystem")
    (list sA_AgentUpdateComfort_11 sA_AgentMove_12 "neighbors")
    (list sA_AgentMove_12 sA_AgentCrossStreet_12 "sameSystem")
    (list sA_AgentMove_12 sA_AgentUpdatePleasure_12 "sameSystem")
    (list sA_AgentUpdatePleasure_12 sA_AgentUpdateSafety_12 "sameSystem")
    (list sA_AgentUpdateSafety_12 sA_AgentUpdateComfort_12 "sameSystem")
    (list sA_AgentUpdateComfort_12 sA_AgentMove_13 "neighbors")
    (list sA_AgentMove_13 sA_AgentCrossStreet_13 "sameSystem")
    (list sA_AgentMove_13 sA_AgentUpdatePleasure_13 "sameSystem")
    (list sA_AgentUpdatePleasure_13 sA_AgentUpdateSafety_13 "sameSystem")
    (list sA_AgentUpdateSafety_13 sA_AgentUpdateComfort_13 "sameSystem")
    (list sA_AgentUpdateComfort_13 sA_AgentMove_14 "neighbors")
```

```
      (list sA_AgentMove_14 sA_AgentCrossStreet_14 "sameSystem")
      (list sA_AgentMove_14 sA_AgentUpdatePleasure_14 "sameSystem")
      (list sA_AgentUpdatePleasure_14 sA_AgentUpdateSafety_14 "sameSystem")
      (list sA_AgentUpdateSafety_14 sA_AgentUpdateComfort_14 "sameSystem")
      (list sA_AgentUpdateComfort_14 sA_AgentMove_15 "neighbors")
      (list sA_AgentMove_15 sA_AgentCrossStreet_15 "sameSystem")
      (list sA_AgentMove_15 sA_AgentUpdatePleasure_15 "sameSystem")
      (list sA_AgentUpdatePleasure_15 sA_AgentUpdateSafety_15 "sameSystem")
      (list sA_AgentUpdateSafety_15 sA_AgentUpdateComfort_15 "sameSystem")
    )
    set pA_WalkOneStep PlaceBuilder:createPragmaticAction "pA_WalkOneStep"
    (list
      (list sA_AgentIsHere 1)
      (list sA_AgentMove_1 weight_sA_AgentMove)
      (list sA_AgentUpdatePleasure_1 weight_sA_AgentUpdatePleasure)
      (list sA_AgentUpdateSafety_1 weight_sA_AgentUpdateSafety)
      (list sA_AgentUpdateComfort_1 weight_sA_AgentUpdateComfort)
    )
    (list
      (list sA_AgentIsHere sA_AgentMove_1 "neighbors")
      (list sA_AgentMove_1 sA_AgentUpdatePleasure_1 "sameSystem")
      (list sA_AgentUpdatePleasure_1 sA_AgentUpdateSafety_1 "sameSystem")
      (list sA_AgentUpdateSafety_1 sA_AgentUpdateComfort_1 "sameSystem")
    )
  ]

  ;;report list of pragmatic Actions
  report (list pA_Walk pA_WalkOneStep)

end

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;; CODE FOR IMPLEMENTING STATE CHANGES
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;implementation function for sA_AgentMove
to implement_sA_AgentMove [pedestrian cell]

  ;;agent moves to cell
  ask pedestrian
  [ move-to cell ]

end

;implementation function for sA_AgentCrossStreet
to implement_sA_AgentCrossStreet [pedestrian cell]

  ;;identify crossing destination path from crossing start cell, move to crossing destination
  ask pedestrian
  [
   let crossingDestinationPatch patch [crossDestRow] of cell [crossDestCol] of cell
   move-to crossingDestinationPatch
  ]
  set numberOfCrossings numberOfCrossings + 1

end

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;; CODE FOR THE CREATION OF PEDESTRIAN AGENTS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;create Pedestrians
to createPedestrians

  ;initialize variables for x,y coordinates of origin patch of agent
  let originPatch_X nobody
  let originPatch_Y nobody

  ;check if origin-node and destination-node of agent are potential OD-nodes,
  ;get location of origin-node and identify respective patch
  ifelse member? OriginNodeID potODList = True and member? DestinationNodeID potODList = True
  [
    foreach gis:feature-list-of nodeShapefile
    [
      if gis:property-value ? "ID" = OriginNodeID
      [
        let location gis:location-of (first(first(gis:vertex-lists-of ?)))
        set originPatch_X item 0 location
        set originPatch_Y item 1 location
      ]
    ]

    ;create pedestrian agent at origin-node
    ask patch originPatch_X originPatch_Y
    [
      sprout-pedestrians 1
      [
        set OD-Pair (list OriginNodeID DestinationNodeID) ;;;store ID of origin- and destination-node as attribute

        ;;set pedestrian attributes from global input variables
        set minSurface minSurface_input
        set maxSlope maxSlope_input
        set width width_input
        set waveringMovement waveringMovement_input
        set shyAwayDistTraffic shyAwayDistTraffic_input
        set shyAwayDistCyc shyAwayDistCyc_input
        set shyAwayDistWall shyAwayDistWall_input
        set maxLight maxLight_input
        set maxBench maxBench_input
        set minTimeKerbDelay minTimeKerbDelay_input
        set walkingSpeed walkingSpeed_input
        set maxHeight maxHeight_input

        ;;set pedestrian weight coefficients from global input variables
        set weight_sA_AgentMove weight_sA_AgentMove_input
        set weight_sA_AgentUpdateSafety weight_sA_AgentUpdateSafety_input
        set weight_sA_AgentUpdatePleasure weight_sA_AgentUpdatePleasure_input
        set weight_sA_AgentUpdateComfort weight_sA_AgentUpdateComfort_input
        set weight_sA_AgentCrossStreet weight_sA_AgentCrossStreet_input
        set rA_KeepBalance_weight_sA_AgentMove rA_KeepBalance_weight_sA_AgentMove_input
        set rA_OvercomeSlope_weight_sA_AgentMove rA_OvercomeSlope_weight_sA_AgentMove_input
        set rA_OvercomeHeightDifference_weight_sA_AgentMove rA_OvercomeHeightDifference_weight_sA_AgentMove_input
        set rA_AccessPatch_weight_sA_AgentMove rA_AccessPatch_weight_sA_AgentMove_input
        set rA_ReachDestination_weight_sA_AgentMove rA_ReachDestination_weight_sA_AgentMove_input
        set rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety_input
        set rA_StayNearLight_weight_sA_AgentUpdateSafety rA_StayNearLight_weight_sA_AgentUpdateSafety_input
        set rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety_input
        set rA_SeeGreenery_weight_sA_AgentUpdatePleasure rA_SeeGreenery_weight_sA_AgentUpdatePleasure_input
        set rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure_input
        set rA_NotSeeTraffic_weight_sA_AgentUpdateComfort rA_NotSeeTraffic_weight_sA_AgentUpdateComfort_input
```

```
      set rA_StayNearBenches_weight_sA_AgentUpdateComfort rA_StayNearBenches_weight_sA_AgentUpdateComfort_input
    ]
  ]

  ask pedestrians
  [
    ;;;set shape and size of pedestrian
    set color red
    set size 5
    set shape "pedestrian"

    ;;;set trajectory-draw on
    pen-down

    ;;;initialize beliefs and intentions
    set beliefs []
    set intentions []
  ]
 ]
 [
  ;;if invalid origin- or destination-nodes are set, print error message
  print "Error: Please choose OD-Pair from potential origin-destination-nodes"
 ]

end

;procedure for setting the initial beliefs of the pedestrian, i.e. visibility graph network,
;shortest path to the destination-node, and action structure
to setInitialBeliefs [pedestrian]

  ;;create visibility graph and add as belief
  let network createNetwork pedestrian
  ask pedestrian
  [ add-belief create-belief "network" network ]

  ;;find shortest path for given OD-pair
  let shortestPathList findShortestPath pedestrian
  let shortestPathLinks item 0 shortestPathList
  let shortestPathNodes item 1 shortestPathList

  ask pedestrian
   [
    ;;store links of shortest Path as belief
    add-belief create-belief "shortestPathLinks" shortestPathLinks

    ;;store nodes of shortest Path as belief
    add-belief create-belief "shortestPathNodes" shortestPathNodes

    ;;store action structure as belief
    add-belief create-belief "actionStructure" createActionStructure self
   ]

end

;procedure to allocate a goalTest function to the agent's desire
to setDesire [pedestrian]

  set desire goalTest self

end

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;; CODE FOR CREATING THE INTENTIONS OF THE AGENTS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


; CALCULATE SUITABILITY VALUES
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;calculate suitability values for rA_BeHere (value is always 0, except at the current position of the pedestrian,
;this is to avoid the pedestrian starting his first rA anywhere in the viewfield)
to-report calc_suitability_rA_BeHere [pedestrian cell]

  let suitability_rA_BeHere -9999 ;;NoData value is used, since the affordance is a KO-criterion

  ;;attribute the suitability value 1 to system of pedestrian and the patch at its current Position
  if cell = [patch-here] of pedestrian
  [ set suitability_rA_BeHere 1 ]

  ;;report suitability value
  report suitability_rA_BeHere

end

;calculate suitability values for rA_KeepBalance
to-report calc_suitability_rA_KeepBalance [ pedestrian cell ]

  ;;initialize suitability for realization Action
  let suitability_rA_KeepBalance -9999 ;;;NoData value used, since the affordance is a KO-criterion

  ;;define capability and disposition values for realization action
  let capKeepBalance [minSurface] of pedestrian ;;;capability
  let dispKeepBalance [surface] of cell  ;;;disposition

  ;;define Pi-Value, Pi-Min and Pi-O values
  let PiValueKeepBalance dispKeepBalance / capKeepBalance ;;;Pi Value
  let PiMinKeepBalance 0 ;;;Pi min, critical lower Pi Value
  let PiOKeepBalance 5 ;;;Pi optimal Pi Value

  ;;calculate suitability value, here it is a special case, since the PiMin value is the point
  ;;where the suitability should reach 0.2.
  ;;this is because the minimum surface quality attribute of the pedestrian denotes the minimum quality
  ;;at which walking is still possible.
  ;;therefore, min Pi value is set to 0
  ;;to avoid suitability values < 0.2 for unavoided actions, in case that Pi < 1, the suitability is set to -9999
  if PiValueKeepBalance >= 1
  [
    set suitability_rA_KeepBalance ((PiValueKeepBalance - PiMinKeepBalance) /
      (PiOKeepBalance - PiMinKeepBalance)) ^ exponent
  ]

  ;;return suitability value
  report suitability_rA_KeepBalance

end

;calculate suitability values for rA_OvercomeSlope
to-report calc_suitability_rA_OvercomeSlope [ pedestrian cell ]

  ;;initialize suitabilityy for realization Action
```

```
  let suitability_rA_OvercomeSlope -9999 ;;;NoData value used, since the affordance is a KO-criterion

  ;;define capability and disposition values for realization action
  let capOvercomeSlope [maxSlope] of pedestrian ;;;capability
  let dispOvercomeSlope [slope] of cell ;;;disposition

  ;;define Pi-Value, Pi-Max and Pi-O values
  let PiValueOvercomeSlope dispOvercomeSlope / capOvercomeSlope ;;;Pi Value
  let PiMaxOvercomeSlope 1.000000000000001 ;;;Pi max, critical upper Pi Value
  let PiOOvercomeSlope 0 ;;;Pi optimal Pi Value

  ;;calculate suitability value
  if PiValueOvercomeSlope < PiMaxOvercomeSlope
  [
    ifelse PiValueOvercomeSlope <= PiOOvercomeSlope
    [ set suitability_rA_OvercomeSlope 1 ]
    [
      set suitability_rA_OvercomeSlope ((PiValueOvercomeSlope - PiMaxOvercomeSlope) /
        (PiOOvercomeSlope - PiMaxOvercomeSlope)) ^ exponent
    ]
  ]

  ;;return suitability value
  report suitability_rA_OvercomeSlope

end

;calculate suitability values for rA_OvercomeHeightDifference
to-report calc_suitability_rA_OvercomeHeightDifference [ pedestrian cell ]

  ;;initialize suitabilityy for realization Action
  let suitability_rA_OvercomeHeightDifference -9999 ;;;NoData value used, since the affordance is a KO-criterion

  ;;define capability and disposition values for realization action
  let capOvercomeHeightDifference [maxHeight] of pedestrian ;;;capability
  let dispOvercomeHeightDifference [patchHeight] of cell ;;;disposition

  ;;define Pi-Value, Pi-Max and Pi-O values
  let PiValueOvercomeHeightDifference dispOvercomeHeightDifference / capOvercomeHeightDifference ;;;Pi Value
  let PiMaxOvercomeHeightDifference 1.000000000000001 ;;;Pi max, critical upper Pi Value
  let PiOOvercomeHeightDifference 0 ;;;Pi optimal Pi Value

  ;;calculate suitability value
  if PiValueOvercomeHeightDifference < PiMaxOvercomeHeightDifference
   [
    ifelse PiValueOvercomeHeightDifference <= PiOOvercomeHeightDifference
    [ set suitability_rA_OvercomeHeightDifference 1 ]
    [
      set suitability_rA_OvercomeHeightDifference ((PiValueOvercomeHeightDifference - PiMaxOvercomeHeightDifference) /
        (PiOOvercomeHeightDifference - PiMaxOvercomeHeightDifference)) ^ exponent
    ]
   ]

  ;;return suitability value
  report suitability_rA_OvercomeHeightDifference

end

;calculate suitability values for rA_AccessPatch
to-report calc_suitability_rA_AccessPatch [ pedestrian cell ]

  ;;initialize suitabilityy for realization Action
  let suitability_rA_AccessPatch -9999 ;;;NoData value used, since the affordance is a KO-criterion

  ;;define capability and disposition values for realization action
  let capAccessPatch ([width] of pedestrian + [shyAwayDistWall] of pedestrian +
    [waveringMovement] of pedestrian) ;min space requirement
  let dispAccessPatch [buildDist] of cell

  ;;define Pi-Value, Pi-Min and Pi-O values
  let PiValueAccessPatch dispAccessPatch / capAccessPatch ;;;Pi Value
  let PiMinAccessPatch 0.999999999999999 ;;;Pi min, critical lower Pi Value
  let PiOAccessPatch 5 ;;;Pi optimal Pi Value

  ;;calculate suitability value
  if PiValueAccessPatch > PiMinAccessPatch ;test to make sure that Pi values 0.99 < Pi < 1 are not afforded
  [
    ifelse PiValueAccessPatch >= PiOAccessPatch
    [ set suitability_rA_AccessPatch 1 ]
    [
      set suitability_rA_AccessPatch ((PiValueAccessPatch - PiMinAccessPatch) /
        (PiOAccessPatch - PiMinAccessPatch)) ^ exponent
    ]
  ]

  ;return suitability value
  report suitability_rA_AccessPatch

end

;calculate suitability values for rA_ReachDestination
to-report calc_suitability_rA_ReachDestination  [ pedestrian cell ]

  ;;initialize suitability for realization action
  let suitability_rA_ReachDestination 0 ;0 value is used, since the affordance is not a KO-Criterion

  ;;get current node the agent is headed towards
  let nextNode last read-first-belief-of-type "nextNode"

  ;;calculate the maximum and minimum distance of cells within viewfield to the nextNode
  let maxDistNextNode distance nextNode ;distance pedestrian - nextNode is the maximum
  let minDistNextNode maxDistNextNode - viewDistance ;end of viewfield is minimum distance, since agent faces nextNode

  ;;calculate standardized suitability values
  ask cell
  [
    let distNextNode distance nextNode
    set suitability_rA_ReachDestination (1 - ((distNextNode - minDistNextNode) / (maxDistNextNode - minDistNextNode)))
  ]

  ;;report suitability
 report suitability_rA_ReachDestination

end


;calculate suitability values for rA_KeepDistToMotorTraffic  (value 0-1)
to-report calc_suitability_rA_KeepDistToMotorTraffic  [ pedestrian cell ]

  ;;initialize suitability for realization action
```

```
  let suitability_rA_KeepDistToMotorTraffic  0 ;0 value is used, since the affordance is not a KO-Criterion

  ;;define capability and disposition values for realization action
  let capKeepDistToMotorTraffic [width] of pedestrian + [waveringMovement] of pedestrian +
      [shyAwayDistTraffic] of pedestrian ;minimum space requirements
  let dispKeepDistToMotorTraffic [minDistStreet] of cell

  ;;define Pi-Value, P-Min and Pi-O values
  let PiValueMotorTraffic dispKeepDistToMotorTraffic / capKeepDistToMotorTraffic ;;;Pi Value
  let PiMinMotorTraffic  0.999999999999999 ;;;Pi min, critical lower Pi Value
  let PiOMotorTraffic 5 ;;;Pi optimal Pi Value

  ;;calculate suitability value
  ;;if classification = 1 and minDistStreet = -9999, then street is over 25 meters away
  ifelse [minDistStreet] of cell = -9999
  [
    if [classification] of cell = 1
    [ set suitability_rA_KeepDistToMotorTraffic 1 ]
  ]
  [
    if PiValueMotorTraffic > PiMinMotorTraffic
    [
      ifelse PiValueMotorTraffic >= PiOMotorTraffic
      [ set suitability_rA_KeepDistToMotorTraffic 1 ]
      [
        set suitability_rA_KeepDistToMotorTraffic ((PiValueMotorTraffic - PiMinMotorTraffic) /
          (PiOMotorTraffic - PiMinMotorTraffic)) ^ exponent
      ]
    ]
  ]

 ;;report suitability value
 report suitability_rA_KeepDistToMotorTraffic

end

;calculate suitability values for rA_KeepDistToCycleTraffic
to-report calc_suitability_rA_KeepDistToCycleTraffic  [ pedestrian cell ]

  ;;initialize suitability for realization action
  let suitability_rA_KeepDistToCycleTraffic  0 ;0 value is used, since the affordance is not a KO-Criterion

  ;;define capability and disposition values for realization action
  let capKeepDistToCycleTraffic [width] of pedestrian + [waveringMovement] of pedestrian +
      [shyAwayDistCyc] of pedestrian ;;; total width needed for walking when street is nearby
  let dispKeepDistToCycleTraffic [minDistCycle] of cell

  ;;define Pi-Value, P-Min and Pi-O values
  let PiValueCycleTraffic dispKeepDistToCycleTraffic / capKeepDistToCycleTraffic ;;;Pi Value
  let PiMinCycleTraffic 0.999999999999999 ;;;Pi min, critical lower Pi Value for distance
  let PiOCycleTraffic 3 ;;;Pi optimal Pi Value

  ;;calculate suitability value
  ifelse [minDistCycle] of cell = -9999
  [
    if [classification] of cell = 1
    [ set suitability_rA_KeepDistToCycleTraffic 1 ]
  ]
  [
    if PiValueCycleTraffic > PiMinCycleTraffic

    [
     ifelse PiValueCycleTraffic >= PiOCycleTraffic
     [ set suitability_rA_KeepDistToCycleTraffic 1 ]
     [
       set suitability_rA_KeepDistToCycleTraffic ((PiValueCycleTraffic - PiMinCycleTraffic) /
         (PiOCycleTraffic - PiMinCycleTraffic)) ^ exponent
     ]
    ]
  ]

 ;;report suitability value
 report suitability_rA_KeepDistToCycleTraffic

end

;calculate suitability values for rA_SeeGreenery
to-report calc_suitability_rA_SeeGreenery [ pedestrian cell ]

  ;;initialize suitability value
  let suitability_rA_SeeGreenery 0 ;0 value is used, since the rA_SeeGreenery-affordance is not a KO-Criterion

  ;;calculate suitability; for comparability, the values are standardized with minimumValue and maximumValue
  if [treeVis] of cell != -9999 ;test for -9999, since when reorienting, the "viewfield" might include street cells
  [
    ask cell
    [
      set suitability_rA_SeeGreenery (([treeVis] of self - minValue_treeVis)  /
        (maxValue_treeVis - minValue_treeVis)) ^ exponent
    ]
  ]

  ;;report suitability value
  report suitability_rA_SeeGreenery

end

;calculate suitability values for rA_SeeAestheticBuild
to-report calc_suitability_rA_SeeAestheticBuild [ pedestrian cell ]

  ;;initialize suitability value
  let suitability_rA_SeeAestheticBuild 0 ;0 value is used, since the affordance is not a KO-Criterion

  ;;calculate suitability; for comparability, the values are standardized with minimumValue and maximumValue
  if [buildVis] of cell != -9999 ;test for -9999, since when reorienting, the "viewfield" might include street cells
  [
    ask cell
    [
      set suitability_rA_SeeAestheticBuild ([buildVis] of self - minValue_buildVis) /
        (maxValue_buildVis - minValue_buildVis)
    ]
  ]

  ;;report suitability value
  report suitability_rA_SeeAestheticBuild

end

;calculate suitability values for rA_StayNearLight
to-report calc_suitability_rA_StayNearLight [ pedestrian cell ]
```

```
  ;;initialize suitability value
  let suitability_rA_StayNearLight 0 ;0 value is used, since the affordance is not a KO-Criterion

  ;;define capability and disposition values for realization action
  let capStayNearLight [maxLight] of pedestrian
  let dispStayNearLight [lightDist] of cell

  ;;define Pi-Value, P-Max and Pi-O values
  let PiValueStayNearLight dispStayNearLight / capStayNearLight ;Pi Value
  let PiMaxStayNearLight 1.000000000000001 ;Pi max, critical upper Pi Value
  let PiOStayNearLight 0  ;optimal Pi Value

  ;;calculate suitability
  if PiValueStayNearLight < PiMaxStayNearLight
  [
    ifelse PiValueStayNearLight <= PiOStayNearLight
    [ set suitability_rA_StayNearLight 1 ]
    [
      set suitability_rA_StayNearLight ((PiValueStayNearLight - PiMaxStayNearLight) /
        (PiOStayNearLight - PiMaxStayNearLight)) ^ exponent
    ]
  ]

  ;;report suitability value
  report suitability_rA_StayNearLight

end

;calculate suitability values for rA_NotSeeTraffic
to-report calc_suitability_rA_NotSeeTraffic [ pedestrian cell ]

  ;;initialize suitability value
  let suitability_rA_NotSeeMotorTraffic 0 ;0 value is used, since the affordance is not a KO-Criterion

  ;;calculate suitability
  if [trafficVis] of cell != -9999
  [ set suitability_rA_NotSeeMotorTraffic 1 - ([trafficVis] of cell) ]

  ;;report suitability value
  report suitability_rA_NotSeeMotorTraffic

end

;calculate suitability values for rA_StayNearBenches
to-report calc_suitability_rA_StayNearBenches [ pedestrian cell ]

  ;;initialize suitability value
  let suitability_rA_StayNearBenches 0 ;0 value is used, since the affordance is not a KO-Criterion

  ;;define capability and disposition values for realization action
  let capProxBench [maxBench] of pedestrian
  let dispProxBenches [benchDist] of cell

  ;;define Pi-Value, P-Max and Pi-O values
  let PiValueProxBench dispProxBenches / capProxBench ;Pi Values
  let PiMaxProxBench 1.000000000000001 ;Pi max, critical maximum Pi Value
  let PiOProxBench 0  ;optimal Pi Value

  ;;calculate suitability
  if PiValueProxBench < PiMaxProxBench

  [
    ifelse PiValueProxBench <= PiOProxBench
    [ set suitability_rA_StayNearBenches 1 ]
    [
      set suitability_rA_StayNearBenches ((PiValueProxBench - PiMaxProxBench) /
        (PiOProxBench - PiMaxProxBench)) ^ exponent
    ]
  ]

  ;;report suitability value
  report suitability_rA_StayNearBenches

end

;calculate suitability value for rA_EvaluateStreetCrossing
to-report calc_suitability_rA_EvaluateStreetCrossing [ pedestrian cell ]

  ;;get current node the agent is headed towards
  let nextNode last read-first-belief-of-type "nextNode"

  ;;initialize suitability value
  let suitability_rA_EvaluateStreetCrossing -9999 ;NoData value used, since the affordance is a KO-criterion

  ;;calculate value which expresses if a crossing action would reduce the distance to the next node
  ask cell
  [
    if crossDestCol != -9999
    [
      if nextNode = nobody [set nextNode one-of nodes with [nodeID = last [OD-Pair] of pedestrian]]

      let crossingDestinationPatch patch crossDestRow crossDestCol ;the destination patch of the crossing action
      let distToNextNodeHere distance nextNode ;the distance to the next Node from the current position of the age
      let distToNextNodeAfterCross 0

      ask crossingDestinationPatch
      [ set distToNextNodeAfterCross distance nextNode ] ;the distance to the nextNode after crossing

      if distToNextNodeHere > distToNextNodeAfterCross ;if crossing makes sense, calculate a suitability value
      [ set suitability_rA_EvaluateStreetCrossing  1 - ( distToNextNodeAfterCross / distToNextNodeHere ) ]
    ]

    ;;if there are any light-controlled crossing patches nearby,
    ;;or agent has already crossed a road in this run, avoid jaywalking
    if (any? patches in-radius viewDistance with [classification = 3]) or numberOfCrossings > 0
    [ set suitability_rA_EvaluateStreetCrossing -9999 ]
  ]

  ;;report suitability value
  report suitability_rA_EvaluateStreetCrossing

end

;calculate suitability values for rA_SeeTrafficSituation
to-report calc_suitability_rA_SeeTrafficSituation [ pedestrian cell ]

  ;;initialize suitability value
  let suitability_rA_SeeTrafficSituation -9999

  if [crossDestRow] of cell != -9999
  [
```

```
    ;;define capability and disposition values for realization action
    let dispSeeTrafficSituation [crossSightDist] of cell / ([trafficSpeed] of cell / 3.6) ;time from when a car can
                                                                                           ;first be spotted until it arrives
                                                                                           ;at the crossing location in seconds
    let capSeeTrafficSituation [minTimeKerbDelay] of pedestrian + ([crossingDistance] of cell
      / ([walkingSpeed] of pedestrian / 3.6)) ;time pedestrian needs for the whole crossing process

    ;;define Pi-Value, P-Min and Pi-O values
    let PiValueSeeTrafficSituation dispSeeTrafficSituation / capSeeTrafficSituation
    let PiMinSeeTrafficSituation 0.999999999999999
    let PiOSeeTrafficSituation 2

    ;;calculate suitability
    if PiValueSeeTrafficSituation > PiMinSeeTrafficSituation
    [
      ifelse PiValueSeeTrafficSituation >= PiOSeeTrafficSituation
      [ set suitability_rA_SeeTrafficSituation 1 ]
      [
        set suitability_rA_SeeTrafficSituation ((PiValueSeeTrafficSituation - PiMinSeeTrafficSituation) /
          (PiOSeeTrafficSituation - PiMinSeeTrafficSituation)) ^ exponent
      ]
    ]
  ]

  ;;report suitability value
  report suitability_rA_SeeTrafficSituation

end

;calculate suitability values for rA_ReachSidewalk
to-report calc_suitability_rA_ReachSidewalk [ pedestrian cell ]

  ;;initialize suitability value
  let suitability_rA_ReachSidewalk -9999

  ;;calculate sutiability value
  if [crossDestRow] of cell != -9999
  [
    ;;initialize variables for id of temporary nodes
    let whoFrom -9999
    let whoTo -9999

    ;;identify crossing destination patch
    let crossingDestinationPatch patch [crossDestRow] of cell [crossDestCol] of cell

    ;;create a temporary node at crossing destination patch
    ask crossingDestinationPatch
    [
      sprout-nodes 1
      [
        set hidden? true
        set whoTo who
      ]
    ]

    ;;create a temporary node at crossing start patch and create link with node at crossing destination
    ask cell
    [
      sprout-nodes 1
      [

        set hidden? true
        create-link-with node whoTo
        set whoFrom who
      ]
    ]

    ;;identify all patches which intersect the link
    let patchesInSightline patches with [ gis:intersects? (link whoFrom whoTo) self = True ]

    ;;delete nodes
    ask node whoFrom
    [ die ]

    ask node whoTo
    [ die ]

    ;;initialize counter for number of patches to be crossed, a crossingPossible boolean and a variable to hold the sum of sub-suitability values
    let counter 0
    let crossingPossible? True
    let sumValues 0

    ;;if moving over each involved patch is possible, calculate the suitability for movement on each patch, and sum them
    ask patchesInSightline
    [
      ifelse calc_suitability_rA_KeepBalance pedestrian self != -9999 and calc_suitability_rA_OvercomeSlope pedestrian self != -9999
      and calc_suitability_rA_OvercomeHeightDifference pedestrian self != -9999
      and calc_suitability_rA_AccessPatch pedestrian self != -9999
      [
        set sumValues sumValues + ((calc_suitability_rA_KeepBalance pedestrian self
            + calc_suitability_rA_OvercomeSlope pedestrian self + calc_suitability_rA_OvercomeHeightDifference pedestrian self
            + calc_suitability_rA_AccessPatch pedestrian self) / 4)
        set counter counter + 1
      ]
      [ set crossingPossible? False ]
    ]

    ;;;calculate suitability from average movement suitability for all patches to be crossed,
    ;;;and add a normalized index for the relative crossing length
    if crossingPossible? = True
    [
      set suitability_rA_ReachSidewalk ((sumValues / counter) +
        (1 - ([crossingDistance] of cell / maxValue_crossingDistance))) / 2
    ]
  ]

  ;;report suitability value
  report suitability_rA_ReachSidewalk

end

; WAYFINDING AND LOCOMOTION
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

to getNextNode

   ;;retrieve destination node of the agent
   let destinationNode one-of nodes with [nodeID = last [OD-Pair] of pedestrian 0]

   carefully
   [
```

```
;;;store the distance of the agent to the destination
let agentDistDest distance destinationNode

;;;retrieve shortest path links
let linksSP last read-first-belief-of-type "shortestPathLinks"

;;;check if the agent has crossed a road
if numberOfCrossings > 0
[
  ;;;;if yes, delete all links from the list, which are farther from the destination than the agent's current position
  ;;;;this is done in order to avoid the agent from going back after a jaywalking process
  ask links with [member? self linksSP]
  [
    let minDistDestNode min-one-of both-ends [distance destinationNode]
    let minDistDest 0
    ask minDistDestNode
    [
      set minDistDest distance destinationNode
    ]

    if minDistDest > agentDistDest
    [ set linksSP remove self linksSP ]
  ]
  update-belief create-belief "shortestPathLinks" linksSP
]

;;check if there is already a nextNode belief
ifelse exist-beliefs-of-type "nextNode" = true and last read-first-belief-of-type "nextNode" != nobody
[
  ;;;retrieve current next Node
  let currentNextNode last read-first-belief-of-type "nextNode"

  ;;;check if current next Node has been reached (within a distance of 2 meters),
  ;;;if yes, delete the sightline to the specific node, and delete "earlier" links from the shortest path links belief
  if distance currentNextNode < 5
  [
    ;;;;retrieve shortest path links belief
    set linksSP last read-first-belief-of-type "shortestPathLinks"

    ;;;;delete sightline between pedestrian and current next node
    if link 0 [nodeID] of currentNextNode != nobody
    [
      ask link 0 [nodeID] of currentNextNode
      [ die ]
    ]

    ;;;;store links which include the current next node in a list
    let linksWithNextNode []
    foreach linksSP
    [
      if member? currentNextNode [both-ends] of ?
      [ set linksWithNextNode lput ? linksWithNextNode ]
    ]

    ;;;;if there is more than one link in the list, store links which are "earlier"
    ;;;;than the to-link to current next node in a list
    if length linksWithNextNode != 1
    [
      let currentLink item 1 linksWithNextNode
      let positionCurrentLink position currentLink linksSP

      let earlierLinks []

      foreach linksSP
      [
        if position ? linksSP < positionCurrentLink
        [ set earlierLinks lput ? earlierLinks ]
      ]

      ;;;;delete sightlines to earlier nodes
      let earlierLinksAgentSet links with [member? self earlierLinks]
      let earlierNodes []
      ask earlierLinksAgentSet [set earlierNodes lput both-ends earlierNodes]

      let earlierNodes_temp []

      foreach earlierNodes
      [
        ask ?
        [ set earlierNodes_temp lput self earlierNodes_temp ]
      ]

      ask nodes with [member? self earlierNodes_temp]
      [
        carefully
        [
          ask link 0 [nodeID] of self
          [ die ]
        ]
        []
      ]

      ;;;;;delete "earlier" links from shortest path link belief
      foreach linksSP
      [
        if member? ? earlierLinks
        [ set linksSP remove ? linksSP ]
      ]
      update-belief create-belief "shortestPathLinks" linksSP

    ]
  ]
]
[ add-belief create-belief "nextNode" nobody ] ;;if there is no current next node, initialize

;;initialize nextNode and variable for the farthest visible node of the shortest path nodes
let nextNode last read-first-belief-of-type "nextNode"
let farthestVisNode nobody

;;retrieve shortest path links
set linksSP last read-first-belief-of-type "shortestPathLinks"

;;identify sightline to the shortest path node which is visible
;;and closest to the destination node (closest in terms of shortest path)
let linkToClosestVisibleNode max-one-of (my-links with
  [sightlinePatchRelation self -9999 = True]) [ [posInSP] of other-end ]

ifelse linkToClosestVisibleNode != nobody
[
  ;;identify respective node
  ask linkToClosestVisibleNode
```

```
[ set farthestVisNode max-one-of both-ends [distance patch-here] ]

;;if there is no street to be crossed to reach farthest visible node:
ifelse sightlinePatchRelation linkToClosestVisibleNode 2 = True
[
  ;;set as nextNode
  set nextNode farthestVisNode
  update-belief create-belief "nextNode" nextNode

  ;;delete all sightlines to closer nodes
  ask (my-links with [sightlinePatchRelation self -9999 = True and link-length <
      [link-length] of linkToClosestVisibleNode])
  [ die ]

  ;;delete all earlier links in the SPLinks
  let linksWithNextNode []
  foreach linksSP
  [
    if member? nextNode [both-ends] of ?
    [ set linksWithNextNode lput ? linksWithNextNode ]
  ]

  ifelse length linksWithNextNode > 1
  [
    let currentLink item 1 linksWithNextNode
    let positionCurrentLink position currentLink linksSP
    let earlierLinks []

    foreach linksSP
    [
      if position ? linksSP < positionCurrentLink
      [ set earlierLinks lput ? earlierLinks ]
    ]

    foreach linksSP
    [
      if member? ? earlierLinks
      [ set linksSP remove ? linksSP ]
    ]

    update-belief create-belief "shortestPathLinks" linksSP
  ]
  [ set nextNode destinationNode ]
]
[
  ;;if there is a street to be crossed:
  ;;if the street is in viewfield
  face farthestVisNode
  ifelse any? patches in-cone viewDistance viewfieldWidth with [classification = 2]
  [
    ;;initialize boolean for crossing possible?
    let boolcross False

    ;;identify all potential starting points for crossing within viewfield
    let patchViewfieldList [self] of patches in-cone viewDistance viewfieldWidth with [classification = 1]

    ;;check if crossing is possible
    foreach patchViewfieldList
    [
      if calc_suitability_rA_EvaluateStreetCrossing myself ? != -9999

         and calc_suitability_rA_SeeTrafficSituation myself ? != -9999
         and calc_suitability_rA_ReachSidewalk myself ? != -9999
         [ set boolcross True ]
      ]

      ifelse boolCross = True
      [
        ;;if crossing is possible
        ;;set nextNode farthest vis node but do not delete the earlier links
        set nextNode farthestVisNode
        update-belief create-belief "nextNode" nextNode
      ]

      [
        ;;else, delete sightline to farthestVisNode to avoid the agent trying constantly
        ask link 0 [nodeID] of farthestVisNode
        [ die ]

        ;;get next node in network path
        let currentLink first linksSP
        let nextLink last linksSP
        set nextNode one-of nodes with [member? self [both-ends] of currentLink
          and member? self [both-ends] of nextLink]
        update-belief create-belief "nextNode" nextNode
      ]
     ]
     [
      set nextNode farthestVisNode
      update-belief create-belief "nextNode" nextNode
     ]
    ]
  ]
  [
   ;;if no node is visible: get next node in shortest path
   let currentLink first linksSP
   ifelse length linksSP > 1
   [
     let nextLink item 1 linksSP
     set nextNode one-of nodes with [member? self [both-ends] of currentLink and member? self [both-ends] of nextLink
   ]
   [ set nextNode destinationNode ]

   update-belief create-belief "nextNode" nextNode
  ]

  ;;if there is still no nextNode set, take next one from shortest Path nodes
  ;;or, if the last link has been reached, the destination node as nextNode
  if last read-first-belief-of-type "nextNode" = nobody
  [
   let currentLink first linksSP
   ifelse length linksSP > 1
   [
     let nextLink item 1 linksSP
     set nextNode one-of nodes with [member? self [both-ends] of currentLink and member? self [both-ends] of nextLink
   ]
   [ set nextNode destinationNode ]
   update-belief create-belief "nextNode" nextNode
  ]
]
[ update-belief create-belief "nextNode" destinationNode ]
```

```
  ;;ask pedestrian to set nextNode ID as label
  ask pedestrian 0 [set label [nodeID] of last read-first-belief-of-type "nextNode" set label-color blue]

end

;done-condition reporter for intention "getNextNode"
to-report nextNodeExists?

  if exist-beliefs-of-type "nextNode" = true
  [ report true ]

end

;reporter function needed by getNextNode, identifies for given sightline the classification of the intersecting patches
to-report sightlinePatchRelation [sightline classificationValue]

  ;;initialize reporter
  let reporter True

  ;;identify patches with intersect sightline
  let intersectPatches patches with [gis:intersects? (sightline) self = True and classification = classificationValue]

  ;;if intersecting patches are of type classificationValue, report false
  if any? intersectPatches [set reporter False]
  report reporter

end

;procedure to create a viewfield
to buildViewfield

  ;;retrieve nextNode belief
  let nextNode last read-first-belief-of-type "nextNode"

  ;;orient towards nextNode
  face nextNode

  ;;create viewfield (includes only sidewalk and crossing patches) and store as belief "viewfield"
  let viewfield [self] of patches in-cone viewDistance viewfieldWidth with [classification = 1 or classification = 3]
  add-belief create-belief "viewfield" viewfield

end

;done-condition reporter for intention "buildViewfield"
to-report viewfieldExists? ;reporter for intention buildViewfield

  if exist-beliefs-of-type "viewfield" = true
  [ report true ]

end

;reorientation procedure in case the pragmatic action is not afforded within viewfield
to reorient

  ;;create new viewfield in 360 degrees (agent "looks around"), now all classifications, also street and cycle lane
  let viewfield [self] of patches in-radius viewDistance with [classification != -9999]
  update-belief create-belief "viewfield" viewfield

  ;;re-try place calculation for pragmatic action

  update-belief create-belief "actionStructure" createActionStructure self

end

;procedure to calculate the place for a pragmatic action
to calculatePlace

  ;;create instances of SystemAgentEnvironment from viewfield and add to list
  let systemList []
  foreach last read-first-belief-of-type "viewfield"
  [
    let affordanceList [] ;empty list for affordances

    ;;;add all non-KO realization actions with their suitability value (non-weighted) to the affordance list
    set affordanceList lput (list rA_ReachDestination (calc_suitability_rA_ReachDestination self ? )) affordanceList
    set affordanceList lput (list rA_SeeGreenery (calc_suitability_rA_SeeGreenery self ?)) affordanceList
    set affordanceList lput (list rA_SeeAestheticBuild (calc_suitability_rA_SeeAestheticBuild self ?)) affordanceList
    set affordanceList lput (list rA_StayNearLight (calc_suitability_rA_StayNearLight self ?)) affordanceList
    set affordanceList lput (list rA_NotSeeTraffic (calc_suitability_rA_NotSeeTraffic self ?)) affordanceList
    set affordanceList lput (list rA_StayNearBenches (calc_suitability_rA_StayNearBenches self ?)) affordanceList
    set affordanceList lput (list rA_KeepDistToCycleTraffic (calc_suitability_rA_KeepDistToCycleTraffic  self ?)) affordanceList
    set affordanceList lput (list rA_KeepDistToMotorTraffic (calc_suitability_rA_KeepDistToMotorTraffic self ?)) affordanceList

    ;;;test all KO realization actions for affordance, if true, add to affordance list
    if calc_suitability_rA_BeHere self ? != -9999
    [ set affordanceList lput (list rA_BeHere calc_suitability_rA_BeHere self ?) affordanceList ]

    if calc_suitability_rA_KeepBalance self ? != -9999
    [ set affordanceList lput (list rA_KeepBalance (calc_suitability_rA_KeepBalance self ?)) affordanceList ]

    if calc_suitability_rA_OvercomeSlope self ? != -9999
    [ set affordanceList lput (list rA_OvercomeSlope (calc_suitability_rA_OvercomeSlope self ?)) affordanceList ]

    if calc_suitability_rA_OvercomeHeightDifference self ? != -9999
    [ set affordanceList lput (list rA_OvercomeHeightDifference (calc_suitability_rA_OvercomeHeightDifference self ?)) affordanceList ]

    if calc_suitability_rA_AccessPatch self ? != -9999
    [ set affordanceList lput (list rA_AccessPatch (calc_suitability_rA_AccessPatch self ?)) affordanceList ]

    if calc_suitability_rA_EvaluateStreetCrossing self ? != -9999 and calc_suitability_rA_SeeTrafficSituation self ? != -9999
    and calc_suitability_rA_ReachSidewalk self ? != -9999
    [
      set affordanceList lput (list rA_EvaluateStreetCrossing (calc_suitability_rA_EvaluateStreetCrossing self ?)) affordanceList
      set affordanceList lput (list rA_SeeTrafficSituation (calc_suitability_rA_SeeTrafficSituation self ?)) affordanceList
      set affordanceList lput (list rA_ReachSidewalk (calc_suitability_rA_ReachSidewalk self ?)) affordanceList
    ]

    ;;;create an instance of SystemAgentEnvironment for each patch within viewfield/pedestrian pair
    ;;;and add afforded realization Actions with their suitability values
    let system PlaceBuilder:createSystemAgentEnvironment pedestrian 0 ? affordanceList

    ;;;add SystemAgentEnvironment instance to list
    set systemList lput system systemList
  ]

  ;;retrieve first pragmatic action from belief
  let pragmaticAction first (last read-first-belief-of-type "actionStructure")

  ;;compute and store place
  let place PlaceBuilder:reportPlace pragmaticAction systemList
```

```
  ;;check if pragmatic action is afforded in viewfield
  ifelse place != []
  [
    ;;;if pragmatic action is afforded, add place as belief
    add-belief create-belief "place" place

    ;;;add place to placeList, for export of the results
    foreach place
    [ set placeList lput ? placeList ]
  ]
  [
    ;;;if pragmatic action is not afforded, try alternative pragmatic action
    set pragmaticAction last (last read-first-belief-of-type "actionStructure")
    set place PlaceBuilder:reportPlace pragmaticAction systemList

    ;;check if alternative pragmatic action is afforded in viewfield
    ifelse place != []
    [
      ;;;if alternative action is afforded, add place as belief
      add-belief create-belief "place" place

      ;;;add place to placeList, for export of the results
      foreach place
      [ set placeList lput ? placeList ]
    ]
    [
      ;;;if alternative pragmatic action is not afforded, call reorient procedure and set counter for reorientation + 1
      reorient
      set numberOfReorientation numberOfReorientation + 1
    ]
  ]

  ;;set generalCounter + 1
  set generalCounter generalCounter + 1

end

;done-condition reporter for intention "calculatePlace"
to-report placeExists?

  ifelse exist-beliefs-of-type "place" = true
  [ report true ]
  [ report false ]

end

;procedure to implemement the pragmatic action
to perform_pA_Walk

  ;;for each tuple within place, call the implementation procedure for each semantic action at the respective patch
  foreach last read-first-belief-of-type "place"
  [
    let cell item 0 ? ;the patch from tuple
    let semAction item 2 ? ;the semantic Action from tuple

    ;;;check semantic action by its name
    if member? "sA_AgentMove" word semAction "" = true [implement_sA_AgentMove self cell]
    if member? "sA_AgentCrossStreet" word semAction "" = true [implement_sA_AgentCrossStreet self cell]
  ]

  ;;reset belief place and belief viewfield
  let temp get-belief "place"
  set temp get-belief "viewfield"

end

;done-condition reporter for intention "perform_pA_Walk"
to-report pA_WalkPerformed?

  if exists-belief "place" = false
  [ report true ]

end

;procedure for the creation of intentions
to createIntentions [ pedestrian ]

  ask pedestrian
  [
   add-intention "perform_pA_Walk" "pA_WalkPerformed?" ;intention to perform the pragmatic action
   add-intention "calculatePlace" "placeExists?" ;intention to calculate the place
   add-intention "buildViewfield" "viewfieldExists?" ;intention to perceive the environment for locomotion
   add-intention "getNextNode" "nextNodeExists?" ;perception to identify the next node for wayfinding
  ]

end

;procedure for the execution of intentions
to executeIntentions [ pedestrian ]

  ask pedestrian
  [ execute-intentions ]

end

;procedure to run the simulation
to runSimulation

  ask pedestrians
  [
    ;;update action structure
    update-belief create-belief "actionStructure" createActionStructure self

    ;;;if stack of intentions is empty, create new ones, else execute them
    ifelse intentions = []
    [ createIntentions self ]
    [ executeIntentions self ]

    ;;;update desire state of agent
    set desire goalTest self

    ;;check desire state of pedestrian agent
    if [desire] of self = true
    [
      ;;;if desire state is fullfilled, stop simulation and export results with "goal achieved" parameter
      exportPlaceList 1
      stop
    ]
```

```
    ;;check number of reorientation procedures of agent during simulation run adn generalCounter
    if (numberOfReorientation > 20) or (generalCounter > 1000)
    [
      ;;;if above threshold, stop simulation and export results with "goal not achieved" parameter
      exportPlaceList 0
      stop
    ]
  ]

end

;procedure to test the agent's desire condition
to-report goalTest [ pedestrian ]

  ;;retrieve destination node
  let destinationNode one-of nodes with [nodeID = last [OD-Pair] of pedestrian]

  ;;calculate distance of present agent location to destination
  let distanceDestination distance destinationNode

  ;;desire is true, if the agent is closer to the destination than viewdistance
  ifelse distanceDestination < viewDistance
  [ report true ]
  [ report false ]

end

; ADDITIONAL PROCEDURES FOR RUNNING THE MODEL AND EXPORTING RESULTS
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;sets up a new simulation run
to newSimulationRun

  ;;reset turtles and variables
  clear-turtles
  clear-drawing
  set placeList []
  set numberOfReorientation 0
  set numberOfCrossings 0

  ;;raises IDs of simulation run by 1
  set Agent_ID Agent_ID + 1
  set Setting_ID Setting_ID + 1
  set SimulationRun_ID SimulationRun_ID + 1

end

;writes results to CSV files
to exportPlaceList [destReached?]

  ;;initialize lists for visited patches and their attributes
  let visitedPatchesList [] ;list for visited patches
  let slopeValueList [] ;list for slope values of visited patches
  let surfaceValueList [] ;list for surface values of visited patches
  let benchDistValueList [] ;list for benchDist values of visited patches
  let lightDistValueList [] ;list for lightDist values of visited patches
  let buildVisValueList [] ;list for buildVis values of visited patches
  let treeVisValueList [] ;list for treeVis values of visited patches
  let crossSightDistValueList [] ;list for crossSightDist values of visited patches
  let minDistStreetValueList [] ;list for minDistStreet values of visited patches

  let minDistCycleValueList [] ;list for minDistCycle values of visited patches
  let trafficSpeedValueList [] ;list for trafficSpeed values of visited patches
  let crossingDistanceValueList [] ;list for crossingDistance values of visited patches
  let crossDestRowValueList [] ;list for crossDestRow values of visited patches
  let crossDestColValueList [] ;list for crossDestCol values of visited patches
  let patchHeightValueList [] ;list for patchHeight values of visited patches
  let trafficVisValueList [] ;list for trafficVis values of visited patches
  let buildDistValueList [] ;list for buildDist values of visited patches

  ;;add patches of each place to visitedPatchesList and remove duplicates (count each patch only once)
  foreach placeList
  [ set visitedPatchesList lput item 0 ? visitedPatchesList ]
  set visitedPatchesList remove-duplicates visitedPatchesList

  ;;compute path length from the number of patches in place list
  let pathLength length visitedPatchesList

  ;;add patch attribute values of visited patches to lists
  foreach visitedPatchesList
  [
    set slopeValueList lput [slope] of ? slopeValueList
    set surfaceValueList lput [surface] of ? surfaceValueList
    set benchDistValueList lput [benchDist] of ? benchDistValueList
    set lightDistValueList lput [lightDist] of ? lightDistValueList
    set buildVisValueList lput [buildVis] of ? buildVisValueList
    set treeVisValueList lput [treeVis] of ? treeVisValueList
    set crossSightDistValueList lput [crossSightDist] of ? crossSightDistValueList
    set trafficSpeedValueList lput [trafficSpeed] of ? trafficSpeedValueList
    set crossingDistanceValueList lput [crossingDistance] of ? crossingDistanceValueList
    set patchHeightValueList lput [patchHeight] of ? patchHeightValueList
    set trafficVisValueList lput [trafficVis] of ? trafficVisValueList
    set buildDistValueList lput [buildDist] of ? buildDistValueList

    ;;;in case of minDistStreet and -Cycle a NoData value means over 25 meters (cut-off distance), thus replace values
    ifelse [minDistStreet] of ? = -9999
    [ set minDistStreetValueList lput 25 minDistStreetValueList ]
    [ set minDistStreetValueList lput [minDistStreet] of ? minDistStreetValueList ]

    ifelse [minDistCycle] of ? = -9999
    [ set minDistCycleValueList lput 25 minDistCycleValueList ]
    [ set minDistCycleValueList lput [minDistCycle] of ? minDistCycleValueList ]

  ]

  ;;remove NoData values from lists
  set slopeValueList remove -9999 slopeValueList
  set surfaceValueList remove -9999 surfaceValueList
  set benchDistValueList remove -9999 benchDistValueList
  set lightDistValueList remove -9999 lightDistValueList
  set buildVisValueList remove -9999 buildVisValueList
  set treeVisValueList remove -9999 treeVisValueList
  set crossSightDistValueList remove -9999 crossSightDistValueList
  set minDistStreetValueList remove -9999 minDistStreetValueList
  set minDistCycleValueList remove -9999 minDistCycleValueList
  set trafficSpeedValueList remove -9999 trafficSpeedValueList
  set crossingDistanceValueList remove -9999 crossingDistanceValueList
  set patchHeightValueList remove -9999 patchHeightValueList
  set trafficVisValueList remove -9999 trafficVisValueList
  set buildDistValueList remove -9999 buildDistValueList
```

```
;;compute indicator values for lists
let meanSlopeValue mean slopeValueList
let maxSlopeValue max slopeValueList
let minSlopeValue min slopeValueList
let medianSlopeValue median slopeValueList
let stdSlopeValue standard-deviation slopeValueList
let varSlopeValue variance slopeValueList

let maxSurfaceValue max surfaceValueList
let minSurfaceValue min surfaceValueList
let medianSurfaceValue median surfaceValueList

let meanBenchDistValue mean benchDistValueList
let maxBenchDistValue max benchDistValueList
let minBenchDistValue min benchDistValueList
let medianBenchDistValue median benchDistValueList
let stdBenchDistValue standard-deviation benchDistValueList
let varBenchDistValue variance benchDistValueList

let meanLightDistValue mean lightDistValueList
let maxLightDistValue max lightDistValueList
let minLightDistValue min lightDistValueList
let medianLightDistValue median lightDistValueList
let stdLightDistValue standard-deviation lightDistValueList
let varLightDistValue variance lightDistValueList

let meanBuildVisValue mean buildVisValueList
let maxBuildVisValue max buildVisValueList
let minBuildVisValue min buildVisValueList
let medianBuildVisValue median buildVisValueList
let stdBuildVisValue standard-deviation buildVisValueList
let varBuildVisValue variance buildVisValueList

let meanTreeVisValue mean treeVisValueList
let maxTreeVisValue max treeVisValueList
let minTreeVisValue min treeVisValueList
let medianTreeVisValue median treeVisValueList
let stdTreeVisValue standard-deviation treeVisValueList
let varTreeVisValue variance treeVisValueList

;;initialize indicator values for cross sight dist values
let meanCrossSightDistValue -9999
let maxCrossSightDistValue -9999
let minCrossSightDistValue -9999
let medianCrossSightDistValue -9999
let stdCrossSightDistValue -9999
let varCrossSightDistValue -9999

;;calculate indicator values for cross sight dist if list is not empty
if length crossSightDistValueList != 0
[
  set meanCrossSightDistValue mean crossSightDistValueList
  set maxCrossSightDistValue max crossSightDistValueList
  set minCrossSightDistValue min crossSightDistValueList
  set medianCrossSightDistValue median crossSightDistValueList
  set stdCrossSightDistValue standard-deviation crossSightDistValueList
  set varCrossSightDistValue variance crossSightDistValueList
]

let meanMinDistStreetValue mean minDistStreetValueList

let maxMinDistStreetValue max minDistStreetValueList
let minMinDistStreetValue min minDistStreetValueList
let medianMinDistStreetValue median minDistStreetValueList
let stdMinDistStreetValue standard-deviation minDistStreetValueList
let varMinDistStreetValue variance minDistStreetValueList

let meanMinDistCycleValue mean minDistCycleValueList
let maxMinDistCycleValue max minDistCycleValueList
let minMinDistCycleValue min minDistCycleValueList
let medianMinDistCycleValue median minDistCycleValueList
let stdMinDistCycleValue standard-deviation minDistCycleValueList
let varMinDistCycleValue variance minDistCycleValueList

;;initialize indicator values for mean traffic speed values
let meanTrafficSpeedValue -9999
let maxTrafficSpeedValue -9999
let minTrafficSpeedValue -9999
let medianTrafficSpeedValue -9999
let stdTrafficSpeedValue -9999
let varTrafficSpeedValue -9999

;;calculate indicator values for mean traffic speed if list is not empty
if length trafficSpeedValueList != 0
[
  set meanTrafficSpeedValue mean trafficSpeedValueList
  set maxTrafficSpeedValue max trafficSpeedValueList
  set minTrafficSpeedValue min trafficSpeedValueList
  set medianTrafficSpeedValue median trafficSpeedValueList
  set stdTrafficSpeedValue standard-deviation trafficSpeedValueList
  set varTrafficSpeedValue variance trafficSpeedValueList
]

;;initialize indicator values for mean crossing distance values
let meanCrossingDistanceValue -9999
let maxCrossingDistanceValue -9999
let minCrossingDistanceValue -9999
let medianCrossingDistanceValue -9999
let stdCrossingDistanceValue -9999
let varCrossingDistanceValue -9999

;;calculate indicator values for mean crossing distance if list is not empty
if length crossingDistanceValueList != 0
[
  set meanCrossingDistanceValue mean crossingDistanceValueList
  set maxCrossingDistanceValue max crossingDistanceValueList
  set minCrossingDistanceValue min crossingDistanceValueList
  set medianCrossingDistanceValue median crossingDistanceValueList
  set stdCrossingDistanceValue standard-deviation crossingDistanceValueList
  set varCrossingDistanceValue variance crossingDistanceValueList
]

let meanPatchHeightValue mean patchHeightValueList
let maxPatchHeightValue max patchHeightValueList
let minPatchHeightValue min patchHeightValueList
let medianPatchHeightValue median patchHeightValueList
let stdPatchHeightValue standard-deviation patchHeightValueList
let varPatchHeightValue variance patchHeightValueList

let meanTrafficVisValue mean trafficVisValueList
let maxTrafficVisValue max trafficVisValueList
```

```
  let minTrafficVisValue min trafficVisValueList
  let medianTrafficVisValue median trafficVisValueList
  let stdTrafficVisValue standard-deviation trafficVisValueList
  let varTrafficVisValue variance trafficVisValueList

  let meanBuildDistValue mean buildDistValueList
  let maxBuildDistValue max buildDistValueList
  let minBuildDistValue min buildDistValueList
  let medianBuildDistValue median buildDistValueList
  let stdBuildDistValue standard-deviation buildDistValueList
  let varBuildDistValue variance buildDistValueList

  ;;write results to csv files
  file-open word Output "SimulationRuns.csv"
  file-print ""
  file-type
  (word
    SimulationRun_ID "," Agent_ID "," Setting_ID "," DestinationNodeID "," OriginNodeID "," destReached? ","
    meanSlopeValue "," maxSlopeValue "," minSlopeValue "," medianSlopeValue "," stdSlopeValue "," varSlopeValue ","
    maxSurfaceValue "," minSurfaceValue "," medianSurfaceValue "," meanBenchDistValue "," maxBenchDistValue ","
    minBenchDistValue "," medianBenchDistValue "," stdBenchDistValue "," varBenchDistValue "," meanLightDistValue ","
    maxLightDistValue "," minLightDistValue "," medianLightDistValue "," stdLightDistValue "," varLightDistValue ","
    meanBuildVisValue "," maxBuildVisValue "," minBuildVisValue "," medianBuildVisValue "," stdBuildVisValue ","
    varBuildVisValue "," meanTreeVisValue "," maxTreeVisValue "," minTreeVisValue "," medianTreeVisValue ","
    stdTreeVisValue "," varTreeVisValue "," meanCrossSightDistValue "," maxCrossSightDistValue "," minCrossSightDistValue ","
    medianCrossSightDistValue "," stdCrossSightDistValue "," varCrossSightDistValue "," meanMinDistStreetValue ","
    maxMinDistStreetValue "," minMinDistStreetValue "," medianMinDistStreetValue "," stdMinDistStreetValue ","
    varMinDistStreetValue "," meanMinDistCycleValue "," maxMinDistCycleValue "," minMinDistCycleValue ","
    medianMinDistCycleValue "," stdMinDistCycleValue "," varMinDistCycleValue "," meanTrafficSpeedValue ","
    maxTrafficSpeedValue "," minTrafficSpeedValue "," medianTrafficSpeedValue "," stdTrafficSpeedValue ","
    varTrafficSpeedValue "," meanCrossingDistanceValue "," maxCrossingDistanceValue "," minCrossingDistanceValue ","
    medianCrossingDistanceValue "," stdCrossingDistanceValue "," varCrossingDistanceValue "," meanPatchHeightValue ","
    maxPatchHeightValue "," minPatchHeightValue "," medianPatchHeightValue "," stdPatchHeightValue "," varPatchHeightValue ","
    meanTrafficVisValue "," maxTrafficVisValue "," minTrafficVisValue "," medianTrafficVisValue "," stdTrafficVisValue ","
    varTrafficVisValue "," meanBuildDistValue "," maxBuildDistValue "," minBuildDistValue "," medianBuildDistValue ","
    stdBuildDistValue "," varBuildDistValue "," pathLength "," numberOfCrossings
  )
  file-close

  file-open word Output "Agents.csv"
  file-print ""
  file-type
  (word
    Agent_ID "," minSurface_input "," maxSlope_input "," width_input "," waveringMovement_input ","
    shyAwayDistTraffic_input "," shyAwayDistCyc_input "," shyAwayDistWall_input "," maxLight_input "," maxBench_input ","
    minTimeKerbDelay_input "," walkingSpeed_input "," maxHeight_input "," viewfieldWidth "," viewDistance "," exponent
  )
  file-close

  file-open word Output "Settings.csv"
  file-print ""
  file-type
  (word
    Setting_ID "," rA_StayNearBenches_weight_sA_AgentUpdateComfort "," rA_NotSeeTraffic_weight_sA_AgentUpdateComfort ","
    rA_StayNearLight_weight_sA_AgentUpdateSafety "," rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety ","
    rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety "," rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure ","
    rA_SeeGreenery_weight_sA_AgentUpdatePleasure "," rA_ReachDestination_weight_sA_AgentMove "," rA_AccessPatch_weight_sA_AgentMove ","
    rA_OvercomeHeightDifference_weight_sA_AgentMove "," rA_OvercomeSlope_weight_sA_AgentMove "," rA_KeepBalance_weight_sA_AgentMove ","
    weight_sA_AgentCrossStreet "," weight_sA_AgentUpdateComfort "," weight_sA_AgentUpdateSafety "," weight_sA_AgentUpdatePleasure ","
    weight_sA_AgentMove

  )
  file-close

  ;;export list of places
  foreach placeList
  [
    let row [pxcor] of item 0 ? ;row for each patch
    let col [pycor] of item 0 ? ;column for each patch
    let rA item 1 ? ;realization action
    let sA item 2 ? ;semantic action
    let suitability item 3 ? ;suitability value

    file-open word Output "Places.csv"
    file-print ""
    file-type (word SimulationRun_ID "," row "," col "," rA "," suitability "," sA)
    file-close
  ]

end


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;; CODE FOR HANDLING BELIEFS (original code from Sakellariou 2010)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;creates a new belief. (does not store it in belief memory)
to-report create-belief [b-type content]
  report (list b-type content)
end

;;; reports type of a belief.
to-report belief-type [bel]
  report first bel
end

;; reports the coontent of belief belief
to-report belief-content [bel]
  report item 1 bel
end

;; Adding information to the beliefs structure
to add-belief [bel]
  if member? bel beliefs [stop]
  set beliefs fput bel beliefs
end
;; Removing a belief from the list of beliefs.
to remove-belief [bel]
 set beliefs remove bel beliefs
end

;;; return true if a specific belief belong to the set of beliefs
to-report exists-belief [bel]
   ifelse member? bel beliefs [report true] [report false]
end

;;; Reports true if a belief in the form of ["b-type" etc etc etc ...] exist in beliefs list
to-report exist-beliefs-of-type [b-type]
  let blfs filter [first ? = b-type] beliefs
  ifelse empty? blfs [report false] [report true]
```

```
end

;;; Returns all beliefs of b-type in a list
to-report beliefs-of-type [b-type]
  report filter [first ? = b-type] beliefs
end

;;; Returns the first belief of a certain type and removes it
to-report get-belief [b-type]
  ifelse exist-beliefs-of-type b-type
  [let bel first filter [first ? = b-type] beliefs
   remove-belief bel
   report bel
  ]
  [report false]
end

to-report read-first-belief-of-type [b-type]
   report first beliefs-of-type b-type
end

to update-belief [bel]
   remove-belief read-first-belief-of-type belief-type bel
   add-belief bel
end


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;; CODE FOR HANDLING INTENTIONS (original code from Sakellariou 2010)
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

to execute-intentions
;;;locals [myInt]  ;; first intentions
  if empty? intentions [stop]
  let myInt get-intention
  run intention-name myInt
  ;print word "running intention: " intention-name myInt ;delete
  if runresult intention-done myInt [remove-intention myInt]
  if show-intentions [set label intentions] ;; Just for debugging.
end

;;;; Intentions Structure Access Functions
;; returns the current intention of the agent
to-report current-intention
  report intention-name first intentions
end

;; Reports the full intention structure
to-report get-intention
  report first intentions
end

;; Returns the intetnion name (the executable)
to-report intention-name [intention]
  report item 0 intention
end

;; return the done-methods (arguments) of the intention. If it evaluates to true
;; then the intention is removed.
to-report intention-done [intention]

  report item 1 intention
end

to pop-intention
  set intentions but-first intentions
end

;; Removes a specific intention from the intention stack
to remove-intention [bdi-lib##intention]
  set intentions remove-item (position bdi-lib##intention intentions) intentions
end


;; Adds an intention in the intentions list. REMEMBER that intentions are
;; stored in a STACK!
;; The first argument is the intention name that should be some executable procedure
;; you encode in NetLogo. The second argument should be a REPORTER that when evaluates to
;; true the intention is removed (either accomplished or dropped).
;; BOTH ARGUMENTS HAVE TO BE STRINGS (see run/runresult primitive procedures in NetLogo)

to add-intention [name done]
  set intentions fput (list name done) intentions
end

;;;; SPECIAL ACTIONS
;;; a null action
to do-nothing
end

;;; wait for something until the timeout expires.
to wait-for-timeout
  do-nothing
end

;;;
to-report timeout_expired [timeout]
report (word "timeout_has_expired " ticks " " timeout)
end

;;; INTERNAL not to be USED.
;;; reports the end of the timeout.
to-report timeout_has_expired [start interval]
  report (start + interval < ticks )
end
```

# C: Python Code for Data Preprocessing

```
'''
Module includes methods for preprocessing the geo-spatial
data for the simulation

@author: David Jonietz
'''
import arcpy
import string
from _sqlite3 import Row
arcpy.CheckOutExtension("Spatial")
arcpy.CheckOutExtension("3d")
import arcpy.sa

#calculates the number of visible points for each view point
def calculateVisibilityFrequency(viewPoints, visibleFeatures, DGM, outputpath, fieldname, workspace):

    #set workspace
    arcpy.env.workspace = workspace

    #set overwrite true
    arcpy.env.overwriteOutput = True

    #check geometry type of visiblefeatures
    viewshedPath = "viewshed"
    if arcpy.Describe(visibleFeatures).shapeType == "Point":
        arcpy.Viewshed_3d(DGM, visibleFeatures, viewshedPath)

    #if geometry type is polygon, iterate through features
    if arcpy.Describe(visibleFeatures).shapeType == "Polygon":


       WSTable = []

       oidFieldname = arcpy.Describe(visibleFeatures).OIDFieldName
       rowsPoly = arcpy.SearchCursor(visibleFeatures)
       for rowPoly in rowsPoly:
           oid = rowPoly.getValue(oidFieldname)
           expression = '{0} = {1}'.format(oidFieldname, oid)
           selectedFeature = "in_memory\\selectedPolygon"
           arcpy.Select_analysis(visibleFeatures, selectedFeature, expression)

           #convert feature vertices to points
           featureVertices_RAW = "in_memory\\vertices_RAW"
           arcpy.FeatureVerticesToPoints_management(selectedFeature, featureVertices_RAW)

           #select only outer vertices, meaning which have a height < 491
           featureVertices = "in_memory\\vertices"
           arcpy.AddSurfaceInformation_3d(featureVertices_RAW, DGM, "Z")
           arcpy.Select_analysis(featureVertices_RAW, featureVertices, '"Z" < 491')

           #calculate viewshed for vertices of polygon feature
           if int(arcpy.GetCount_management(featureVertices).getOutput(0)) != 0:
               viewshedTemp = "viewshedTemp" + str(oid)
               arcpy.AddField_management(featureVertices, "OFFSETA", "FLOAT")
               arcpy.CalculateField_management(featureVertices, "OFFSETA", "1.7", "PYTHON")
               arcpy.AddField_management(featureVertices, "OFFSETB", "FLOAT")
               arcpy.CalculateField_management(featureVertices, "OFFSETB", "1.7", "PYTHON")

               arcpy.Viewshed_3d(DGM, featureVertices, viewshedTemp)
```

```
                #reclassify resulting raster to binary format
                remap = arcpy.sa.RemapRange([[0, 0, 0], [1, 1000, 1]])
                viewshedRECL = arcpy.sa.Reclassify(viewshedTemp, "VALUE", remap)
                viewshedRECLPath = "viewshedRECL" + str(oid)
                viewshedRECL.save(viewshedRECLPath)
                listTemp = [viewshedRECLPath, "VALUE", 1]
                WSTable.append(listTemp)

        #add resulting rasters to receive one in which every building counts only once
        outraster = arcpy.sa.WeightedSum(WSTable)
        outraster.save(viewshedPath)

    #add field to viewpoints with fieldname, and extract rastervalues (number of visible point features)
to viewpoints
    arcpy.sa.ExtractValuesToPoints(viewPoints, viewshedPath, outputpath)
    arcpy.AddField_management(outputpath, fieldname, "LONG")
    expression = "!RASTERVALU!"
    arcpy.CalculateField_management(outputpath, fieldname, expression, "PYTHON")
    arcpy.DeleteField_management(outputpath, "RASTERVALU")

    #clear memory workspace
    arcpy.Delete_management("in_memory")
    arcpy.Delete_management(viewshedPath)

#calculates the distance to the nearest visible "visibleFeature" for each "viewPoint"
def calculateVisibilityDistance(viewPoints, visiblePoints, visiblePolygon, fieldname, maxViewDistance,
DGM, workspace, outputPath):

    #set workspace

   arcpy.env.workspace = workspace

   #set overwrite true
   arcpy.env.overwriteOutput = True

   #add distance attribute and set default as NoData
   arcpy.AddField_management(viewPoints, fieldname, "DOUBLE", "","", "", "","NULLABLE")
   arcpy.CalculateField_management(viewPoints, fieldname, "None", "PYTHON")

   #select only the viewPoints within the maximum Distance to the visible Polygons
   polygonBuffer =  "in_memory\\bufferPoly"
   arcpy.Buffer_analysis(visiblePolygon, polygonBuffer, maxViewDistance)
   viewPointsCLIP = outputPath
   arcpy.Clip_analysis(viewPoints, polygonBuffer, viewPointsCLIP)
   arcpy.Delete_management(polygonBuffer)

   #iterate through clipped viewPoints
   oidFieldname = arcpy.Describe(viewPointsCLIP).OIDFieldName
   rowsPoint = arcpy.UpdateCursor(viewPointsCLIP)
   for rowPoint in rowsPoint:
       OID = rowPoint.getValue(oidFieldname)
       expression = '{0} = {1}'.format(oidFieldname, OID)
       selectedFeature = "in_memory\\selectedPoint"
       arcpy.Select_analysis(viewPointsCLIP, "in_memory\\selectedPoint", expression)

       #identify for each viewpoint all visible features within maximum viewing distance
       pointBuffer = arcpy.Buffer_analysis(selectedFeature, "in_memory\\pointBuffer", maxViewDistance)
       visiblePointsClip = arcpy.Clip_analysis(visiblePoints, pointBuffer, "in_memory\
\visiblePointsClip")
```

```
        arcpy.Delete_management(pointBuffer)

        #calculate sightlines between viewpoint and clipped visiblePoints
        sightlines =  "in_memory\\sightlines"

arcpy.AddGeometryAttributes_management(arcpy.Intervisibility_3d(arcpy.ConstructSightLines_3d(selectedFeatu
re, visiblePointsClip, sightlines), DGM), "LENGTH", "METERS", "", "")

        #iterate through sightlines and store their length in list
        rowsLines = arcpy.SearchCursor(sightlines)
        lengthlist = []
        for rowLine in rowsLines:
            lengthlist.append(rowLine.getValue("LENGTH"))

        #identify the minimum length value and store as attribute
        try:
            minValue = min(lengthlist)
            rowPoint.setValue(fieldname, minValue)
            rowsPoint.updateRow(rowPoint)
        except:
            print "lengthlist empty for feature: " + str(OID)

        arcpy.Delete_management(selectedFeature)
        arcpy.Delete_management(visiblePointsClip)
        arcpy.Delete_management(sightlines)

    #clear memory workspace
    arcpy.Delete_management("in_memory")

#calculates the number of visible street point features for each sidewalk feature and standardize the
results
def calculateTrafficVisibility(viewPoints, visiblePoints, DGM, workspace, outputPath):

    #set workspace
    arcpy.env.workspace = workspace

    #set environment settings
    arcpy.env.overwriteOutput = True

    #create a viewshed for all visiblePoints
    viewshed = arcpy.sa.Viewshed(DGM, visiblePoints)
    viewshed.save("viewshed")

    #reclassify/standardize the viewshed to 0 - 1
    maxValue = float(arcpy.GetCount_management(visiblePoints).getOutput(0))
    standViewshed = viewshed / maxValue
    standViewshed.save("standviewshed")

    #transfer grid values to viewpoints
    arcpy.sa.ExtractValuesToPoints(viewPoints, "standviewshed", outputPath, "INTERPOLATE")

    #clear memory workspace
    arcpy.Delete_management("in_memory")

#calculates distance to nearest nearPoint for each locationPoint
def calculateDistToNearest(locationPoints, nearPoints):
    arcpy.Near_analysis(locationPoints, nearPoints)
```

```
        #create a viewshed for each clipped viewpoint feature
        viewshed = arcpy.sa.Viewshed(DGM, selectedPoint)
        viewshed.save("viewshed")

        #reclassify the viewshed to enable vectorization of only visible areas
        remap = arcpy.sa.RemapValue([[0, "NoData"], [1, 0]])
        viewshedRECL = arcpy.sa.Reclassify(viewshed, "VALUE", remap)
        viewshedRECL.save("viewshedRECL")

        #convert viewshed to vector
        viewshedPoly = "viewshedPoly"
        arcpy.RasterToPolygon_conversion(viewshedRECL, viewshedPoly)

        #select polyline in maximum 10 m distance
        arcpy.MakeFeatureLayer_management(polylines, "polylines_lyr")
        try:
            selPolylines = arcpy.SelectLayerByLocation_management("polylines_lyr", "INTERSECT",
selectedPoint, 10)
            numberPolylines = int(arcpy.GetCount_management(selPolylines).getOutput(0))

            #select from visiblePolygons features which intersect the selected Polylines
            arcpy.MakeFeatureLayer_management(visiblePolygons, "polygons_lyr")
            arcpy.SelectLayerByLocation_management("polygons_lyr", "INTERSECT", selPolylines.getOutput(0))
            selPolygons = "selPolygons"
            arcpy.CopyFeatures_management("polygons_lyr", selPolygons)
            numberPolygons = arcpy.GetCount_management(selPolygons)

            #intersect selected polygons with viewshedPoly

            intersectFeatures = "in_memory\\intersectFeatures"
            arcpy.Intersect_analysis([selPolygons, viewshedPoly], intersectFeatures)

            #convert feature vertices into point features
            intersectPoints = "intersectPoints"
            arcpy.FeatureVerticesToPoints_management(intersectFeatures, intersectPoints)

            #calculate distances between selected viewpoint and intersect points
            distanceTable = "distanceTable"
            arcpy.PointDistance_analysis(selectedPoint, intersectPoints, distanceTable)

            #calculate maximum distance
            summaryStats = "summaryStats"
            arcpy.Statistics_analysis(distanceTable, summaryStats, [["DISTANCE", "MAX"]])
            statRows = arcpy.SearchCursor(summaryStats)
            for statRow in statRows:
                maxValue = statRow.getValue("MAX_DISTANCE")

            #write maximum distance value in field
            pointRow.setValue("MaxVisDis", maxValue)
            pointRows.updateRow(pointRow)

        except:
            print "exception detected for feature: " + str(oid)

    #clear memory workspace
    arcpy.Delete_management(r"in_memory")
```

```python
def calculateCrossingDistance(workspace, sidewalkPoints, streetPolygon, streetLines, referenceRaster, DGM,
outputPath):

    #set workspace
    arcpy.env.workspace = workspace

    #set overwrite true
    arcpy.env.overwriteOutput = True

    #identify the origin and cellsize of the rasterfile
    XOrigin = float(arcpy.GetRasterProperties_management(referenceRaster, "LEFT").getOutput(0))
    YOrigin = float(arcpy.GetRasterProperties_management(referenceRaster, "BOTTOM").getOutput(0))
    cellwidth = float(arcpy.GetRasterProperties_management(referenceRaster, "CELLSIZEX").getOutput(0))

    #create a buffer around the polygon features with the same distance as the cell size
    arcpy.Buffer_analysis(streetPolygon, r"in_memory\\buffer", 0.5)

    #clip viewpoints to receive only the ones in the direct vicinity of the polygon feature
    sidewalkPointsCLIP = outputPath
    arcpy.Clip_analysis(sidewalkPoints, r"in_memory\\buffer", sidewalkPointsCLIP)
    arcpy.Delete_management(r"in_memory\\buffer")

    #add fields for crossing distance, column and row of nearest crossing cell
    arcpy.AddField_management(sidewalkPointsCLIP, "CrossDist", "Double")
    arcpy.AddField_management(sidewalkPointsCLIP, "crossPRow", "Long")
    arcpy.AddField_management(sidewalkPointsCLIP, "crossPCol", "Long")
    arcpy.AddField_management(sidewalkPointsCLIP, "crossDest", "Long")

    #iterate through all sidewalkPointsCLIP

    oidFieldname = arcpy.Describe(sidewalkPointsCLIP).OIDFieldName
    pointRows = arcpy.UpdateCursor(sidewalkPointsCLIP)
    for pointRow in pointRows:
        oid = pointRow.getValue(oidFieldname)
        expression = '{0} = {1}'.format(oidFieldname, oid)
        selectedPoint = "selectedPoint"
        arcpy.Select_analysis(sidewalkPointsCLIP, selectedPoint, expression)

        #identify for each viewpoint all sidewalk points within maximum distance
        pointBuffer = arcpy.Buffer_analysis(selectedPoint, "in_memory\\pointBuffer", 20)
        crossingDestPointsCLIP = "in_memory\\crossingDestPointsCLIP"
        arcpy.Clip_analysis(sidewalkPointsCLIP, pointBuffer, crossingDestPointsCLIP)
        arcpy.Delete_management(pointBuffer)

        #calculate sightlines between sidewalk point and clipped crossingPoints
        try:
            sightlines =  "in_memory\\sightlines"
            arcpy.ConstructSightLines_3d(selectedPoint, crossingDestPointsCLIP, sightlines)
            arcpy.AddSurfaceInformation_3d(sightlines, DGM, "Z_MAX")
            expression = '"Z_Max" = 0'
            visibleLines = "in_memory\\visibleLines"
            arcpy.Select_analysis(sightlines, visibleLines, expression)
            arcpy.AddField_management(visibleLines, "VisFID", "SHORT")
            arcpy.CalculateField_management(visibleLines, "VisFID", "!OID!", "PYTHON")
            arcpy.Delete_management(r"in_memory\\sightlines")

            intersectPoints = "in_memory\\intersectPoints"
            arcpy.Intersect_analysis([visibleLines, streetLines], intersectPoints, "", "", "POINT")
            crossinglineIDList = []
```

```
        intersectPointRows = arcpy.SearchCursor(intersectPoints, "", "", "FID_visibleLines")
        for intersectPointRow in intersectPointRows:
            crossinglineIDList.append(intersectPointRow.getValue("FID_visibleLines"))

        intersectLines = "in_memory\\intersectLines"
        arcpy.Clip_analysis(visibleLines, streetPolygon, intersectLines)

         #iterate through intersected Lines and store the length of the original sightline in list

        rowsLines = arcpy.SearchCursor(intersectLines)
        lengthlist = []
        pointIDList = []
        originID = "OID_OBSERV"
        targetID = "OID_TARGET"
        visibleLinesID = "VisFID"
        lengthField = arcpy.Describe(intersectLines).shapeFieldName

        for rowLine in rowsLines:
            if rowLine.getValue(visibleLinesID) in crossinglineIDList:
                lengthlist.append(rowLine.getValue(lengthField).length)
                pointIDList.append(rowLine.getValue(targetID))


        #identify the minimum length value and store as attribute
        minValue = min(lengthlist)
        pointRow.setValue("CrossDist", minValue)

        #identify the coordinates of the closest visible point feature
        pointID = pointIDList [lengthlist.index(minValue)]


        oidFieldname_3 = arcpy.Describe(crossingDestPointsCLIP).OIDFieldName
        pointsCLIPRows = arcpy.da.SearchCursor(crossingDestPointsCLIP, [oidFieldname_3, "SHAPE@XY"])
        for pointsCLIPRow in pointsCLIPRows:
            if pointsCLIPRow[0] == pointID:
                pointX = pointsCLIPRow[1][0]
                pointY = pointsCLIPRow[1][1]

        #identify the row and column of the rastercell underneath the point
        distanceX = pointX - XOrigin
        distanceY = pointY - YOrigin

        row = round(((distanceX / cellwidth) + 0.01), 0) #add 0.01 to avoid round error of python
        col = round(((distanceY / cellwidth) + 0.01), 0)

        #set row and column as values in point fields
        pointRow.setValue("crossPRow", row)
        pointRow.setValue("crossPCol", col)
        pointRow.setValue("crossDest", pointID)
        pointRows.updateRow(pointRow)

    except:
        print "No possible crossing point for feature found"
```

# D: Results of the Sensitivity Analysis (SI-Index)

| meanBuildVisValue | meanTreeVisValue | meanMinDistStreetValue | meanMinDistCycleValue | meanPatchHeightValue | meanTrafficVisValue | meanBuildDistValue | pathLength | numberOfCrossings |
|---|---|---|---|---|---|---|---|---|
| 0,120942902 | -1 | -0,501128403 | 0,236514002 | -0,694584838 | 0,012673723 | -0,029100219 | -0,345153664 | 0 |
| -0,167437558 | 1 | 0,15133087 | -0,303483095 | 1 | 0,448041263 | 0,461724922 | 0,976359338 | 0 |
| -0,014917102 | -0,016893712 | 0,074293152 | 0,022308299 | -0,745892019 | -0,016112588 | -0,114756513 | -0,016166282 | 0 |
| -0,016139332 | 0,026530612 | 0,076439348 | 0,000924212 | 0,192380952 | -0,014318129 | -0,057252997 | 0,009433962 | 0 |
| 0,003223459 | -0,042473298 | -0,021393397 | 0,006991543 | 0,457737557 | -0,006451058 | 0,02587463 | 0,007009346 | 0 |
| -0,017062767 | -0,01754386 | 0,052250588 | 0,008637583 | 0 | -0,021998952 | -0,055195974 | 0 | 0 |
| -0,008327831 | -0,088892467 | 0,026730434 | 0,005142547 | -0,661916073 | 0,008424105 | -0,017935381 | -0,014051522 | 0 |
| -0,007037462 | -0,061611374 | 0,025615196 | -0,006375683 | 0,192344498 | -0,003126002 | -0,002105315 | 0,009478673 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0,008091623 | -0,004728132 | -0,026053677 | 0,003170864 | 0,946174075 | -0,002558024 | 0,031499681 | 0,004728132 | 0 |
| -0,12677047 | 1 | 0,521874898 | -0,237342133 | -0,17950237 | -0,023669709 | -0,002082504 | 0,343601896 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0,156020844 | -1 | -0,496161652 | 0,284007156 | 0,869174566 | 0,00864443 | 0,003010463 | -0,372037915 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0,164207746 | 1 | 0,529857131 | -0,290271993 | -0,792900259 | -0,046825524 | -0,062082799 | 0,384976526 | -1 |
| 0,007125891 | 0,059382423 | -0,032058346 | 0,004986719 | -0,194258373 | 0,005975935 | 0,011270503 | -0,007125891 | 0 |
| -0,003890972 | 0,05841785 | 0,097139588 | -0,002045335 | -0,706259874 | 1,87069E-06 | -0,105922319 | -0,007058824 | 0 |
| -0,016463415 | -0,375 | 0,192332035 | 0,027669905 | 0 | -0,052360615 | -0,15508636 | 0 | 0 |
| 0,014570657 | -0,053780529 | 0,00356726 | -0,000481103 | 0,757358791 | -0,006645879 | 0,011024261 | 0,018735363 | 0 |
| -0,000705143 | 0,002369668 | 0,000409464 | 0,000596748 | 0,002369668 | -0,000736903 | -0,000210682 | -0,002369668 | 0 |
| -0,00064994 | -0,036954087 | 0,056621348 | 0,010982526 | 0,01627907 | -0,010384456 | -0,082291711 | -0,01627907 | 0 |
| -0,034385906 | 1 | -0,515703344 | -0,117368213 | 0,57277628 | 0,067987728 | 0,347769824 | 0,164037855 | -1 |
| 0,007371007 | 0 | 0,123027761 | -0,00185718 | -0,988950276 | 0,027529177 | -0,113342481 | 0 | 0 |
| 0,154585277 | -1 | -0,374644838 | 0,29009575 | 0,940723 | 0,014606466 | -0,130174102 | -0,387470998 | 1 |
| 0,004291845 | -0,018181818 | -0,01201963 | -0,00058359 | 0 | 0,004665566 | 0,00548119 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0,164207746 | 1 | 0,529857131 | -0,290271993 | -0,792900259 | -0,046825524 | -0,062082799 | 0,384976526 | -1 |
| -0,003100281 | -0,416282353 | 0,228299552 | 0,062569639 | -0,674264706 | -0,061291546 | -0,253225285 | -0,040632054 | 0 |
| 0,069367402 | 0,067616255 | -0,02993318 | 0,013974781 | 0,457603687 | 0,031744363 | 0,012776678 | -0,013824885 | 0 |
| -0,223309301 | 1 | 0,062001071 | -0,307784557 | -0,599083814 | 0,034683889 | 0,417143615 | 0,669833729 | -1 |

| | meanSlopeValue | maxSurfaceValue | minSurfaceValue | meanBenchDistValue | meanLightDistValue |
|---|---|---|---|---|---|
| minSurface_input | 0,049416609 | 0 | 0,2 | -0,200426334 | -0,02857319 |
| maxSlope_input | 0,701645768 | 0 | -0,2 | 0,652244251 | -0,453987804 |
| width_input | -0,019869215 | 0 | 0 | -0,006261995 | 0,011596404 |
| waveringMovement_input | -0,01120917 | 0 | 0 | 0,003239578 | 0,013080761 |
| shyAwayDistTraffic_input | 0,033361129 | 0 | 0 | -0,001537461 | 0,000384304 |
| shyAwayDistCyc_input | -0,014184397 | 0 | 0 | -0,000537515 | 0,013815646 |
| shyAwayDistWall_input | -0,03598443 | 0 | 0 | -0,004828923 | 0,001486512 |
| maxLight_input | 0,006655844 | 0 | 0 | 0,003721444 | 0,007931051 |
| maxBench_input | 0 | 0 | 0 | 0 | 0 |
| minTimeKerbDelay_input | 0 | 0 | 0 | 0 | 0 |
| walkingSpeed_input | 0 | 0 | 0 | 0 | 0 |
| maxHeight_input | 0,013725555 | 0 | 0 | -0,000872201 | -0,000783684 |
| viewfieldWidth | -0,048944532 | 0 | 0 | 0,199634626 | 0,029659514 |
| viewDistance | 0 | 0 | 0 | 0 | 0 |
| exponent | 0,124137565 | 0 | 0,2 | -0,23756292 | -0,064662048 |
| rA_StayNearBenches_weight_sA_AgentUpdateComfort | 0 | 0 | 0 | 0 | 0 |
| rA_NotSeeTraffic_weight_sA_AgentUpdateComfort | -0,079867224 | 0 | -0,2 | 0,239035129 | 0,067694116 |
| rA_StayNearLight_weight_sA_AgentUpdateSafety | -0,006348173 | 0 | 0 | -0,002983977 | -0,007981545 |
| rA_KeepDistToMotorTraffic_weight_sA_AgentUpdateSafety | -0,051681622 | 0 | 0 | 0,000943041 | 0,003946728 |
| rA_KeepDistToCycleTraffic_weight_sA_AgentUpdateSafety | 0,01426025 | 0 | 0 | -0,001266613 | 0,038433436 |
| rA_SeeAestheticBuild_weight_sA_AgentUpdatePleasure | 0,034175711 | 0 | 0 | 0,00028507 | 0,00441732 |
| rA_SeeGreenery_weight_sA_AgentUpdatePleasure | 0,002369668 | 0 | 0 | -0,000676697 | -0,000620082 |
| rA_ReachDestination_weight_sA_AgentMove | 0,017154266 | 0 | 0 | -0,00381882 | 0,009967285 |
| rA_AccessPatch_weight_sA_AgentMove | 0,04778036 | 0 | -0,2 | 0,10297253 | 0,038541728 |
| rA_OvercomeHeightDifference_weight_sA_AgentMove | 0,024107143 | 0 | 0 | 0,000358057 | 0,011533633 |
| rA_OvercomeSlope_weight_sA_AgentMove | -0,212884203 | 0 | 0,2 | -0,229297758 | -0,042355104 |
| rA_KeepBalance_weight_sA_AgentMove | 0,003555556 | 0 | 0 | -0,000142381 | -0,000136953 |
| weight_sA_AgentCrossStreet | 0 | 0 | 0 | 0 | 0 |
| weight_sA_AgentUpdateComfort | -0,079867224 | 0 | -0,2 | 0,239035129 | 0,067694116 |
| weight_sA_UpdateSafety | -0,023717804 | 0 | 0 | -0,014389261 | 0,027688589 |
| weight_sA_UpdatePleasure | 0,007029535 | 0 | 0 | -0,008828762 | -0,021246176 |
| weight_sA_AgentMove | -0,37170662 | 0 | -0,2 | 0,443848342 | 0,041779116 |