Path planning of cooperating industrial robots using evolutionary algorithms

Lars Larsen, Alfons Schuster, Jonghwa Kim, Michael Kupke

Angaben zur Veröffentlichung / Publication details:

Larsen, Lars, Alfons Schuster, Jonghwa Kim, and Michael Kupke. 2018. "Path planning of cooperating industrial robots using evolutionary algorithms." *Procedia Manufacturing* 17: 286–93. https://doi.org/10.1016/j.promfg.2018.10.048.

Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 17 (2018) 286–293

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

28th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2018), June 11-14, 2018, Columbus, OH, USA

Path Planning of Cooperating Industrial Robots Using Evolutionary Algorithms

Lars Larsen[a], Alfons Schuster[a], Jonghwa Kim[b,*], Michael Kupke[a]

[a]*German Aerospace Center (DLR), Am Technologiezentrum 4, 86159 Augsburg, Germany*
[b]*University of Science & Technology (UST), 217 Gajeong-ro, 34113 Daejon, Korea*

Abstract

Nowadays, aerospace components made of carbon fiber reinforced plastics (CFRP) which are manufactured by a dry-fiber depositing process are produced almost exclusively by hand. In this type of process, hundreds of textile blanks with different contours are stored in a tool mould. For large components such as an aircraft fuselage, it is also necessary to transport the flexible blanks with cooperating robots. For this purpose, a system was developed which allows to generate robot paths on the basis of Computer Aided Design (CAD) data, which is automatically extended by information such as pickup and drop points for the robot. The system has already been successfully tested with sampling-based algorithms on a robot system with two KUKA KR210 R3100 robots on a common linear axis. However, the use of sampling-based algorithms a heuristic must be developed to solve the redundancy of linear axis which is usually only functional for the current scenario and must be adapted for another one. Evolutionary algorithms offer the advantage that they are also able to find a solution for the linear axis independent of the scenario. In this work. A path planning system that uses evolutionary algorithms to calculate collision-free paths reliably for cooperating robots on a common linear axis will be presented.

Keywords: Path Planning; Cooperating Robots; Evolutionary Algorithm; Sampling-based Algorithm

1. Introduction

Nowadays, industrial robots are increasingly being used in production. Nevertheless, programming is still a major cost driver during commissioning. The robots are often programmed either with the teach-in method or by an off-line tool which requires subsequent adaptation. For small companies, however, robots are often only worthwhile if they can be used for many different components which again leads to cost effective programming. The production of large

* Corresponding author. Tel.: +82-42-865-2475 ; fax: ++82-42-865-2329.
E-mail address: kim@ieee.org

10.1016/j.promfg.2018.10.048

carbon fiber reinforced plastics (CFRP) structures, where many different blanks have to be transported in one mould, also requires a great amount of programming effort. There is a demand from different areas for solutions that enable automated path planning and robot program generation. This work introduces a system that makes it possible to create automated paths for cooperating robots on a common linear track.

2. Related Work

Although path planning for robots is a large field, there are just a few publications dealing with the planning for cooperating robots. In the following some works for sampling-based and Evolutionary Algorithm (EA) are presented.

[11] was the first publication to find a path for a closed kinematic chain consisting of several manipulators transporting a solid object. A procedure is presented which first creates a path for the object to be moved with the Randomized Path Planner (RPP) algorithm. In the second step, for each of these states for all manipulators involved, the Inverse Kinematic (IK) checks whether there is a valid configuration. [18] extends the Probabilistic Roadmap Method (PRM) algorithm for closed chain solutions where the connection between the chains does not change. The basic idea is to use optimization techniques to find a local path for the closed chain. The algorithm is validated using a 2D environment. [9] presents a system in which two PUMA robots on a mobile platform jointly transport a fixed pole in a simulation environment. The procedure is also based on PRM algorithm. The closed chain is broken down into sub-chains. For one of these chains (active) a solution is found with the PRM and for the remaining manipulators (passive) a solution is attempted via the IK. The disadvantage of this procedure is that solutions can be lost by the determination of the active manipulator, since solutions are only sought for the passive manipulator if the active manipulator has a solution in the workspace. [5, 6] introduced the Random Loop Generator (RLG), which extends the approach of [9] to such an extent that a solution for the passive is also found for each generated sample for the active manipulator.

[19] introduces a co-evolutionary algorithm that plans a path for 2-Degrees of Freedom (DOF) robots in a 2D environment sharing the same workspace. The algorithm is initialized with two populations with one population for each robot. As fitness the number of collisions of the two robots, the path length, the movement profile and the absolute movement of the axes is taken. [4] represent an adaptive multi-chromosome EA that is able to plan tasks for modular systems with n robots in 3D moving in the same workspace. A chromosome consists of n lists representing robot poses. [7] introduce a Pareto-based co-evolutionary algorithm that provides the path for two 6-DOF FANUC robots. However, planning is not focused on a cooperation between the robots but rather to coordinate the two robots in such a way that they do not collide.

3. Automatic Path Planning of Cooperating Industrial Robots

3.1. Implementation

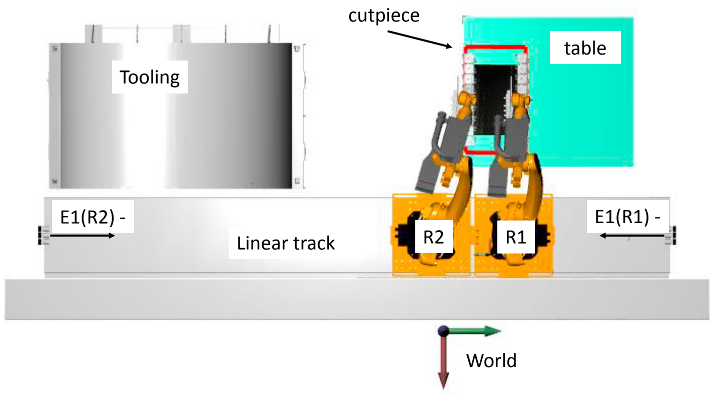
In [2, 15, 17, 14] the Collision free Cooperating robot path planning system (CoCo), which has been developed in C#, has been introduced and stepwise improved. It uses Helix Toolkit [10] for 3D visualization and BEPUphysics [3] for collision detection of the simulated objects. Since many collision checks are necessary for path planning, convex hulls are internally used to represent the robots, linear track and the pick-up table (see fig. 1b). Only the tool shape is represented as a grid structure, since it is intended to be used for lay-up.

For path planning sampling-based as well as evolutionary algorithms are used. For sampling-based algorithms Open Motion Planning Library (OMPL) [20] has been integrated into CoCo which itself does not have a visualization or collision detection mechanisms which allows integrating the library into systems that provide these functionalities. OMPL consists of many state-of-the art sampling-based motion planning algorithms like PRM, Rapidly Exploring Random Trees (RRT), RRT-connect, RRT*, Sparse Roadmap Spanner algorithm (SPARS) and many more. As basis for the implementation of the EAs the AForge [1] library has been used which is also written in C# and provides the base function for implementing EAs.

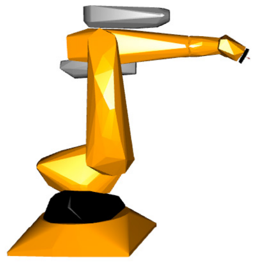
The validation tests of the CoCo planning system were performed on a Airbus A321 lower fuselage (1977 mm diameter and a length of 2000 mm) production scenario. The simulated robot facility consists of two KUKA QUANTEC KR210 R3100 ultra industrial robots on a common 8000 mm linear track. Figure 1a shows a plan view of the structure of the robot cell in the CoCo simulation environment. For both robots there is a common world coordinate system (x=red, y=green, z=blue) which lies below the linear axis. In the lower area in gray from left to right, the linear track (E1) can be seen. For the right robot R1 the origin of the axis is on the right and for the left robot R2 on the left side. When the value of the axis decreases R1 moves from right to left and R2 from left to right. In the upper right corner a green table is located on which the blanks are positioned. The upper left corner shows the lay-up tooling of the fuselage. In [16] an EA for path planning of single robots was introduced. In this work it adopted to work for cooperating robots on a common linear track.

Sampling-based planners initially only consider the robot without the linear axis. For this reason, a heuristic has been implemented which allows to solve the redundancy of the external axis. First, a linear function was set up which determines the position of the linear axis depending on the X-position of the Tool Center Point (TCP) of the robot R1. The corresponding points were determined by manually approaching different position and setting a valid and collision free value of the linear axis of the robots. For $X_1 = -3961$ and $E_1 = 525$ the robots are in X-direction above the middle of the tool form or table. For $X_2 = -1180$ and $E_2 = 1290$ the robots are located in X-direction above the linear axis. Inserting these values into a linear equation results in the following function whereby X is the x-position of the TCP and the result is the corresponding E1 value:

$$f(x) = \frac{255 \cdot X}{887} + \frac{1475730}{887} \tag{1}$$



(a) Cooperating robot fuselage production scenario.



(b) Convex Hull representation.

Fig. 1: Production scenario on the left and convex hull representation of a robot on the right.

3.2. Sampling-based Planners

Sampling-based algorithms generate a roadmap of sampled configurations in the configuration space. This kind of algorithm works very well for path planning and should therefore be used to compare the developed EA.

Single Query algorithms: The main characteristic of single query algorithms is that they search a path from a start to a goal configuration without previous knowledge an no preprocessing phase is necessary. For another query in the same scene the algorithm completely has to recalculate even if the start and goal configuration just changed a little bit. Most famous representatives are RRT, Expansive Space Trees (EST), Kinematic Planning by Interior-Exterior Cell Exploration (KPIECE), Potential Field Methods, Search Tree with Resolution Independent Density Estimation (STRIDE), Path-Directed Subdivision Trees (PDST) and Fast Marching Tree algorithm (FMT).

Multi Query algorithms: These kinds of algorithms have a preprocessing phase where the configuration space is extensively examined for collision states which are stored in a data structure which can be used in the planning

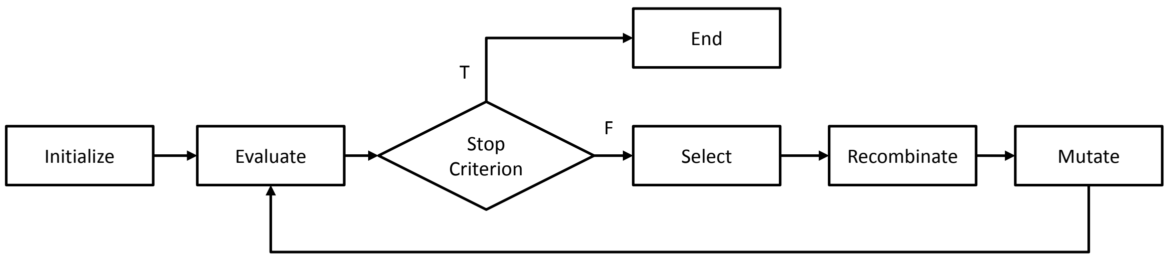


Fig. 2: Sequence of an Evolutionary Algorithm (EA).

phase. At the beginning these kinds of planners are slower than single query algorithms but for new calculation on the same scene the stored data can be used. Multi query algorithms are optimal for static environments and famous representatives are PRM, SPARS and SPArse Roadmap Spanner Version 2.0 (SPARStwo).

3.3. Evolutionary Algorithms

EAs are naturally inspired algorithms which were introduced 1975 in [8] and can be applied to various application. They are assigned to the class of stochastic and meta-heuristic methods, because they normally dont find the optimal solution for a problem. Originally there existed four main distinctions of evolutionary algorithms which cannot be clearly distinguished today (genetic algorithms, evolutionary stategies, genetic programming and evolutionary programming). Often the names are used as synonym for the entire area of the EA.

An EA is working according to the steps shown in fig. 2. The detailed implementation for this use-case is explained in below.

Chromosome representation: Each chromosome for cooperating robots consists of two lists of path points, one list for each robot, with the following structure $StartPoint$, $ViaPoint_1$, $ViaPoint_2$,.,., $ViaPoint_n$, $GoalPoint$ and a list of the corresponding axis values. The individual points are stored as 6D coordinates X, Y, Z, Y, B and C and the axes as $A1, A2, A3, A4, A5, A6$ and $E1$.

Initialize: First, a point in the workspace which is marked by a blue frame in fig. 3 is randomly generated for robot R1. Afterwards the external axis of this robot is set to the same height as the Y-position of the TCP (see fig. 3a). In the next step, the linear axis of the robot R1 is moved randomly in the range $[-200, 1500]$ to the left or right until a position within the axis ranges of R1 has been found (fig. 3b). After a valid external axis position has been determined the IK checks whether a valid robot configuration has been generated for R1. If the check was successful, the TCP position for R2 is determined dependent to the TCP of R1 using a catenary function. This function describes the behavior of a chain which is fixed at two points and can be used to approximate the behavior of the blanks. A detailed explanation of the implementation can be found in [14]. After the TCP-position of R2 has been calculated, an robot- and external- axis-configuration is determined with the same procedure as for R1 (fig. 3c). In some cases it can happen that for R1 a configuration was determined, which makes it impossible to find a valid one for R2. This can be the case, for example, if the generated TCP of R1 is located directly near the end stop of the external axis of R2. Therefore, if after 30 attempts no valid axis configuration for R2 is found, a new configuration for R1 will be created.

Evaluate: The most important point in implementing an EA is the fitness function which is essential if the algorithm finds a valid solution. For the application presented here, the fitness consists of three factors: path length, axis movement and collision value. These factors are firstly normalized and afterwards added together with a different weighting. Since a path should be collision-free in any case the Collision Fitness (CF) is weighted with the factor 0.6. Path Lenght Fitness (PLF) and Axis Movement Fitness (AMF) are equally rated with the factor 0.2. The fitness function can be seen in eq. (2).

For the PLF (see eq. (3)) firstly the euclidean distance between Start-, Via and Goal-Points for R1 is calculated. In the next step it is normalize by the fivefold direct path length (5356 mm) and then subtracted from two. For a long path the PLF gets greater which means a bad fitness. If the PLF gets a negative value it is set to 0.001. The calculation of the AMF (see eq. (4)) is similar to the path length. First the movement of the axes is summed up from Start-, Via- and Goal-Points for R1. The resulting value is divided by 30000, which has been determined by experiments, and afterwards subtracted from 1.0. The collision detection of the chromosome path is divided into two steps (see eq. (5)). First, the system checks if there are ray intersections for any of the path segments of the chromosomes, because it is much less expensive to calculate. Are there more than 25 ray collision the CF is set very a very small value 0.001 (see case 1 eq. (5)). If the ray collision value is more than 0 and less equal to 25 the CF is calculated by dividing the ray collisions by 25 and subtract the value from 1 (see case 2 eq. (5)). Are there no more ray collisions, the convex hull collision for this path are calculated. For this purpose, the path is divided into segments of 500 mm length and the robot TCP is placed at each of these points and the collisions that occur are counted. If the value is greater than 40 the collision fitness value is set to 0.001 (see case 3 eq. (5)) otherwise the counted convex hull collisions are divided by 40.0 and subtracted from 1.0 (see case 4 eq. (5)).

$$Fitness = 0.2 \cdot \mathrm{PLF} + 0.2 \cdot \mathrm{AMF} + 0.6 \cdot \mathrm{CF} \tag{2}$$

$$\mathrm{PLF} = 2.0 - \frac{\sqrt{\sum_{i=1}^{\#Points} (PointList[i] - PointList[i-1])}}{5 \cdot DirectPathLength} \tag{3}$$

$$\mathrm{AMF} = 1.0 - \frac{\sum_{i=1}^{\#Points} \sum_{j=1}^{7} AxisList[i]_{A_j} + AxisList[i-1]_{A_j}}{30000} \tag{4}$$

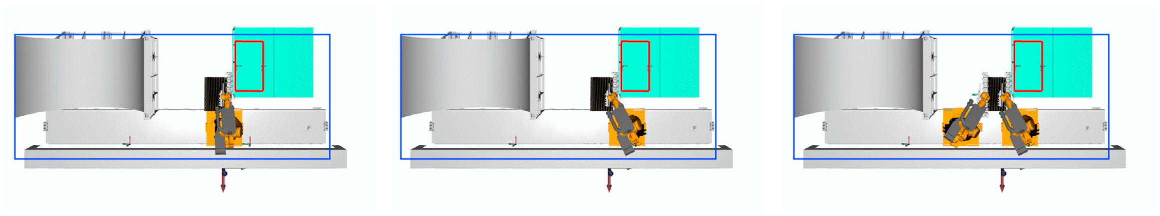
$$\mathrm{CF} = \begin{cases} 0.001 & \text{if RayCollisions} > 25 \\ 1.0 - (RayCollisions/25.0) & \text{if RayCollisions} > 0 \text{ and RayCollisions} \leq 25 \\ 0.001 & \text{if RayCollisions} == 0 \text{ and ConvexHullCollisions} > 40 \\ 1.0 - (ConvexHullCollisions/40.0) & \text{otherwise} \end{cases} \tag{5}$$

Select: Individuals are selected for reproduction. AForge provides Elite- and Rank-selection. Additionally the Tournament- and Roulette-Wheel-selection has been developed in CoCo. More details can be found here in [12].

Recombinate: For the recombination (crossover) a one-point crossover was implemented. First a seed point in the list of points is determined. At this point the path information of two chromosomes is exchanged for both robots.

Mutate: For mutation, a point is randomly selected from the list of Via points (VP) of R1 of a chromosome. This point is changed in a maximum range of 500 mm in x-, y- and z-direction. Thereafter the same procedure which is done during the initialization is performed for finding an axis configuration for R1 and R2.

Stop Criterion: The algorithm will cancel if a collision free path is found or the maximal time is reached.



(a) Generation of a random TCP-position for R1. (b) Determination of a valid E1-position for R1. (c) Determination of a configuration for R2.

Fig. 3: Generation of chromosome configurations inside the working space marked by the blue rectangle [13]

4. Experimental Results

For the evolutionary algorithm there are a lot of parameters which can be changed and have direct influence to the result. In order to determine good parameters a test series had been performed in a single robot scenario in [16] where an EA had to find a path for an industrial robot in a very narrow workspace. First all values were set to the default value and then one value was changed step by step, as described in the following listing. Each parameter set was run ten times and the results have been compared on the basis of the path length, axis movement and the calculation time.

- Mutation rate (MR): 0,1 bis 0,9 (default: 0.3, stepsize: 0.1, best: 0.3)
- Crossover rate (CR): 0,1 bis 0,9 (default: 0.75, stepsize: 0.1, best: 0.6)
- Random Portion Selection (RPS): 0,0 bis 0,8 (default: 0.1, stepsize: 0.1, best: 0.1)
- Individual Count (IC): 60, 100, 150, 200, 250, 300, 400, 500, 600, 700, 800 (default: 100, best: 200)
- Selection Method (SM): Elite Selection, Tournament Selection, Rank Selection and Roulette Wheel Selection (default: Elite Selection, best: Elite Selection)

For RRT-connect the following settings, which were validated in earlier experiments in a single robot scenario, were chosen. Collision Check Resolution: 0.001, Plan with Experience: no, Plan with Constraints: yes, Plan in Joint Space: no, Optimize Path Length: no, Solve Time on Scratch: 20min, State Sampler: Uniform, path simplification (PS): yes and no, Number of points per Path Section: 10, Planner Range SE(3): 100.

With these setting the performance of RRT-connect with and without PS and of EA with 2, 4 and 6 VP has been compared in the fuselage production scenario. The results can be seen in table 1. Figure 5 shows screenshots of best results of all ten runs. RRT-connect with path simplification gets in median the shortest paths with the shortest calculation time. The time is in median less than ten minutes and for EA with 2 VP more than ten minutes. The success rate for both planners is 90%. The robot- and external-axis movement for EA is considerably smaller compared to RRT-connect. This is also visible in fig. 4. The left side shows the movement of the external axis of robot R1 and R2 with RRT-connect and the right side with EA and 2 VP. For RRT-connect some unnecessary movements can be seen whereas the movement of EA is very smooth. One drawback of the EA is that with a raising amount of VP the path length, the axis movement and calculation time is rising. The calculation time increases as more collision checks are performed for the higher amount of path segments. The path length and axis movement increases as there are unnecessary movements (see figs. 5d and 5e) in the pick-up and lay-up area. This should be remedied in future versions by improving the fitness function, which takes greater account of path length and axial movement.

5. Conclusion

In direct comparison with RRT-connect, EAs also delivers very good results. A very big advantage of EAs is that no fixed heuristics for redundancy resolution of the external axis must be implemented. Thus, the algorithm is easily applicable for different scenarios. Only the calculation time of the EAs is longer in the current implementation compared with RRT-connect, since the collision detection only uses one thread. This drawback will be erased in future versions of CoCo which makes EAs a good alternative for sampling-based algorithms.

Table 1: Overview of planning result of sampling-based and Evolutionary Algorithm (EA) planner.

Planner	success (%)	path length (*mm*)	calc. time (*s*)	axis (°)	E1 (°)
		MED / MIN	MED / MIN	MED / MIN	MED / MIN
RRT-connect (¬ PS)	60	11843 / 11514	407 / 343	1925 / 1682	10758 / 8834
RRT-connect	90	8215 / 7701	519 / 413	1395 / 1337	6476 / 6476
EA (2 VP)	90	9880 / 9108	619 / 167	1098 / 994	6001 / 5391
EA (4 VP)	80	11040 / 9861	1036 / 787	1256 / 1088	6828 / 5683
EA (6 VP)	80	12026 / 10759	1080 / 665	1514 / 1273	7291 / 6652

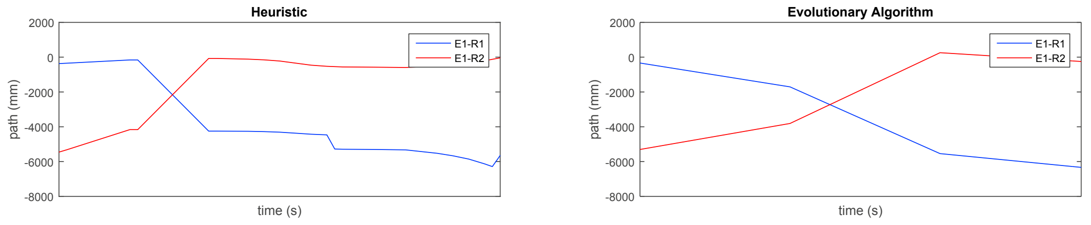
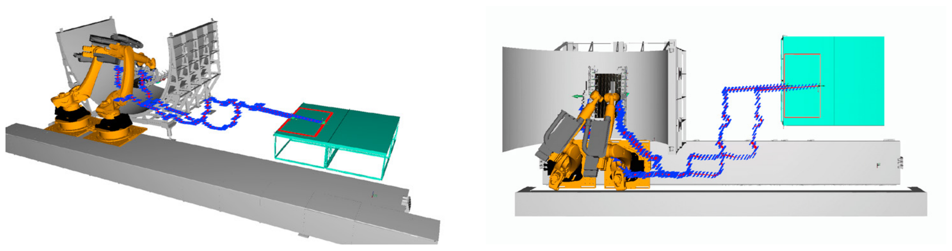


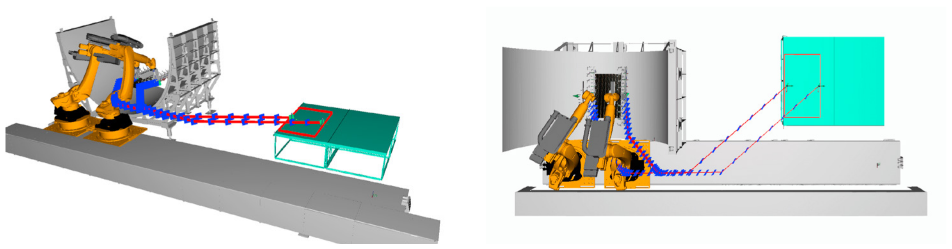
Fig. 4: Comparison of movement of external axis between RRT-connect (with PS and heuristic) and Evolutionary Algorithm (EA).

References

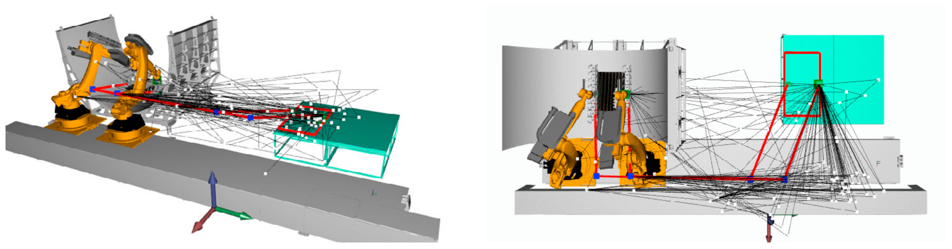
[1] AForge, 2017. Aforge. URL: http://www.aforgenet.com/framework/.
[2] Angerer, A., Hoffmann, A., Larsen, L., Vistein, M., Kim, J., Kupke, M., Reif, W., 2016. Planning and Execution of Collision-free Multi-robot Trajectories in Industrial Applications, in: 47th International Symposium on Robotics, Munich.
[3] BEPU physics, 2018. Bepu physics. URL: https://github.com/bepu/bepuphysics1.
[4] Chocron, O., Bidaud, P., 1997. Evolutionary Algorithms in Kinematic Design of Robotic Systems .
[5] Cortés, J., 2003. Motion Planning Algorithms for General Closed-Chain Mechanisms. Ph.D. thesis.
[6] Cortés, J., Siméon, T., 2005. Sampling-Based Motion Planning under Kinematic Loop-Closure Constraints. Workshop on the Algorithmic Foundations of Robotics 17, 75–90.
[7] Curkovic, P., Jerbic, B., Stipancic, T., 2013. Co-Evolutionary Algorithm for Motion Planning of Two Industrial Robots with Overlapping Workspaces. International Journal of Advanced Robotic Systems , 1doi:10.5772/54991.
[8] De Jong, K.A., 1975. Analysis of the Behaviour of a Class of Genetic Adaptive Systems. Ph.D. thesis. Univeristy of Michigan.
[9] Han, L., Amato, N.M., 2000. A Kinematic-Based Probabilistic Roadmap Method for Closed-Chain Systems .
[10] HelixToolkit, 2017. Helixtoolkit. URL: https://github.com/helix-toolkit.
[11] Koga, Y., Latombe, J.C., 1994. On multi-arm manipulation planning. Proceedings of the 1994 IEEE International Conference on Robotics and Automation , 945–952doi:10.1109/ROBOT.1994.351231.
[12] Kruse, R., Borgelt, C., Braune, C., Klawonn, F., Moewes, C., Steinbrecher, M., 2012. Computational Intelligence - Eine methodische Einführung in Künstliche Neuronale Netze, Evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze.
[13] Larsen, L., 2018. Auf dem Weg zu einer flexiblen Produktion: Automatische und kollisionsfreie Bahnplanung für kooperierende Industrieroboter. Ph.D. thesis. Univeristy Augsburg.
[14] Larsen, L., Kaspar, M., Schuster, A., Vistein, M., Kim, J., Kupke, M., 2017a. Full Automatic Path Planning of Cooperating Robots in Industrial Applications, in: 13th IEEE Conference on Automation Science and Engineering, Xian.
[15] Larsen, L., Kim, J., Kupke, M., 2014. Intelligent Path Panning Towards Collision-free Cooperating Industrial Robots, in: International Conference on Informatics in Control, Vienna.
[16] Larsen, L., Kim, J., Kupke, M., Schuster, A., 2017b. Automatic Path Planning of Industrial Robots Comparing Sampling-Based and Computational Intelligence Methods, in: 27th International Conference on Flexible Automation and Intelligent Manufacturing, Elsevier, Modena.
[17] Larsen, L., Pham, V.L., Kim, J., Kupke, M., 2015. Collision-free Path Planning of Industrial Cooperating Robots for Aircraft Fuselage Production, in: Proceedings - IEEE International Conference on Robotics and Automation, Seattle. doi:10.1109/ICRA.2015.7139466.
[18] LaValle, S.M., Yakey, J.H., Kavraki, L.E., 1999. A Probabilistic Roadmap Approach for System with Closed Kinematic Chains, pp. 6–11.
[19] Petar, Ć., Jerbi, B., 2010. Dual-Arm Robot Motion Planning Based on Cooperative Coevolution , 169–178.
[20] Åucan, I.A., Moll, M., Kavraki, L., 2012. The open motion planning library. IEEE Robotics and Automation Magazine 19, 72–82. doi:10.1109/MRA.2012.2205651.



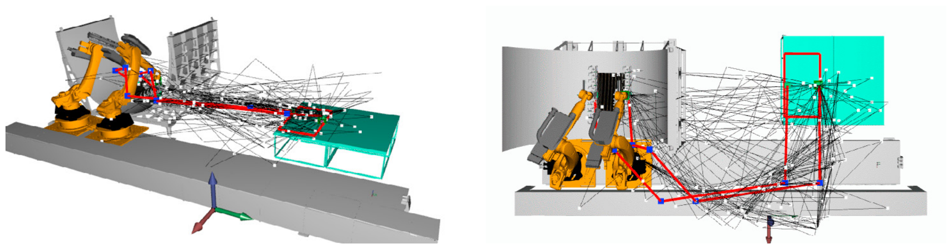
(a) RRT-connect ¬PS – path length 11 514 mm, axis movement 1682°, external axis movement 8834°



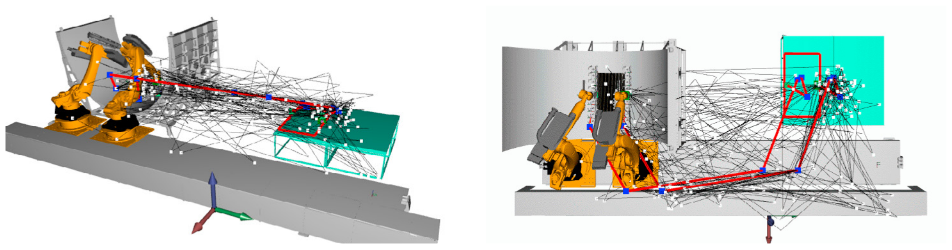
(b) RRT-connect – path length 7701 mm, axis movement 1369°, external axis movement 6476°



(c) EA IC=200, VP=2 – path length 9108 mm, axis movement 1970°, external axis movement 6047°



(d) EA IC=200, VP=4 – path length 9861 mm, axis movement 1195°, external axis movement 7364°



(e) EA IC=200, VP=6 – path length 10 759 mm, axis movement 1452°, external axis movement 6697°

Fig. 5: Comparison of planning results of sampling-based and EA planners. Screenshot shows result with shortest path of all ten runs. For EA the path of the best chromosome is shown in red and the rest of the population in black [13].