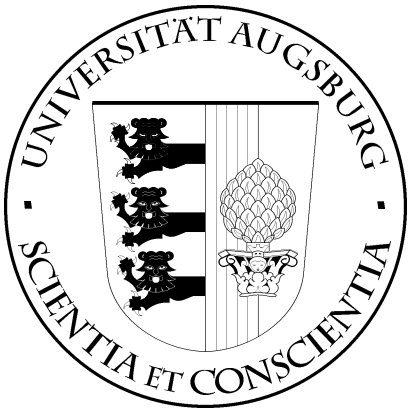
Calibration of Visual Sensors and Actuators in Distributed Computing Platforms

Eva Hörster, Rainer Lienhart, Wolfgang Kellermann, Jean-Yves Bouguet

Angaben zur Veröffentlichung / Publication details:

Hörster, Eva, Rainer Lienhart, Wolfgang Kellermann, and Jean-Yves Bouguet. 2005. "Calibration of Visual Sensors and Actuators in Distributed Computing Platforms ." Augsburg: Universität Augsburg.

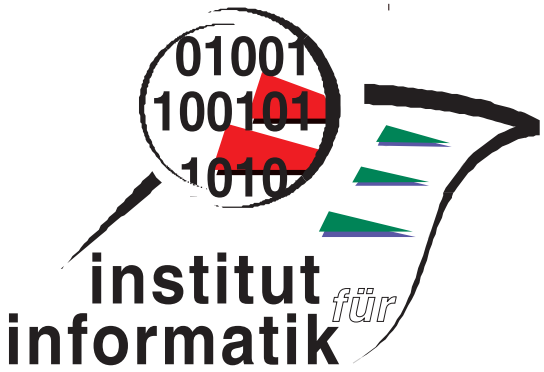
Universität Augsburg



Calibration of Visual Sensors and Actuators in Distributed Computing Platforms

E. Hörster, R. Lienhart, W. Kellermann and J.-Y. Bouguet

Report 2005-11 Juni 2005



Institut für Informatik
D-86135 Augsburg

Calibration of Visual Sensors and Actuators in Distributed Computing Platforms

E. Hörster[*] , R. Lienhart
University of Augsburg
Augsburg, Germany
Hoerster/Lienhart@informatik.uni-augsburg.de

W. Kellermann
University of Erlangen-Nuremberg
Erlangen, Germany
wk@LNT.de

J.-Y. Bouguet
Intel Corporation
Santa Clara, CA, USA
jean-yves.bouguet@intel.com

ABSTRACT

Many novel multimedia applications such as virtual immersive environments use multiple sensors and actuators. We present in this paper a novel approach for position calibration of visual sensors and actuators, i.e. cameras and displays, in a distributed network of general purpose computing devices. The proposed approach is very suitable for the calibration of mobile setups since (a) synchronization of the setup is not required, (b) it works fully automatic, (c) only weak restrictions are imposed on the positioning of the cameras and displays, and (d) no upper limit on the number of cameras and displays to calibrated is imposed. Corresponding points across different camera images are established automatically and found with subpixel accuracy. Cameras do not have to share one common view, only a reasonable overlap between camera subgroups is necessary. The method has been sucessfully tested in numerous multi-camera environments with a varying number of cameras. It has proven itself to work extremely accurate. Performance results are reported.

1. INTRODUCTION

Many audio-visual sensor/actuator array processing applications, including video conferencing and smart conference rooms, video surveillance, Multiple Perspective Video [10], games, e-learning, home entertainment and image based rendering, require that the positions of the sensors and actuators are known precisely. Current systems either place the sensors and actuators in known locations or manually calibrate their positions making mobile systems with multiple sensors and actuators impossible.

Today we find microphones, cameras, loudspeakers and displays nearly everywhere - in public, at home and at work. These audio/video sensors and actuators often are a component of computing and communication devices such as laptops, PDAs and tablets, which we refer to as General Purpose Computers (GPCs). Often GPCs are networked using high-speed wired or wireless connections. The resulting array of audio/video sensors and actuators along with array processing algorithms offers a set of new features for multimedia applications such as mentioned above.

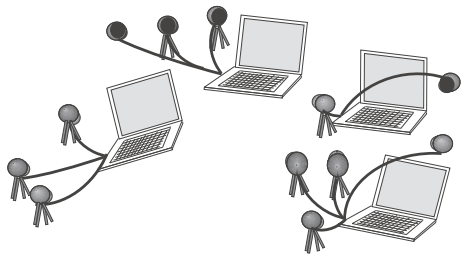


Figure 1: Distributed computing platform consisting of networked general purpose computers with visual sensors (cameras) and actuators (flat-panel displays)

Such a setup demands a simple and convenient calibration approach to put all sensors and actuators into a common time and space. This is an important prerequisite to enable such a platform for powerful multimedia-applications. [11] proposes a means to provide a common time reference for multiple distributed GPC's; in [15] a method for automatic calibration of audio sensors and actuators is presented. In this paper we focus on providing a common space (coordinate system) for multiple cameras and flat panel-displays, i.e. video sensors and actuators, by actively estimating their 3D position and pose. We also adress the problem of effortless calibrating the intrinsic parameters of the multiple cameras.

Fig. 1 shows one setup of our distributed computing platform. A room or area is instrumented with N ($N \geq 3$) static cameras, which are connected to networked GPCs of sufficient computational power. Further M ($M \geq 1$) flat-panel screens are at least partially visible from one camera. Given such a distributed platform, the goal is to (a) determine automatically the intrinsic parameters of each camera and (b) the 3D position and pose of the cameras and flat-panel displays in a common coordinate system. Therefore it is assumed that at any instant the number of cameras and the number of flat-panel displays in the setup is known. Our dis-

9[*]This work was performed while she was an intern at Intel Labs, Intel Corporation, Santa Clara, CA, USA.

tributed system will automatically calibrate the positions of the visual sensors and actuators. A precise synchronization of the different devices is not necessary for our framework.

Related Work: Camera calibration is a well researched topic in computer vision. It is defined as the estimation of the intrinsic and extrinsic parameters f a camera. Broadly there are two different methods of camera calibration, photogrammetric calibration and self-calibration [20].
The first method applies a 3D or planar calibration object whose precise geometry is known. Planar methods are more popular because it is easy to obtain a calibration target by just printing the pattern on paper and fixing it on a flat surface. Some different approaches are described in [20], [9], [19]. Although the results of calibrating a camera are good, the major drawback of these calibration methods is that they require special equipment or precise measurements by hand.
A relatively new approach is the use of a virtual calibration object [18], [4]. The calibration object is constructed over time by tracking an easily identifiable object through a 3D scene. Here the disadvantages usually are that the setup has to be synchronized and special equipment is required.
Self calibration techniques ([7], [17], [14]) do not require any special calibration target. They simultaneously process several images from different perspectives of a scene and are based on point correspondences across the images. The accuracy of these methods depends on how accurately those point correspondences between images can be extracted. Point correspondences can be extracted automatically from the images by identifying 2D features and tracking those between the different perspective views. Different algorithms exist [6], [16], [12].
The multiple camera calibration can be solved globally in one setup or subsets of cameras and displays are calibrated first and then merged into a global coordinate system. Since the first method is only suitable if all cameras share a common view, we follow the second more general approach.

Contributions: The main contributions of the paper are:

- A novel setup for a network of multiple visual sensors and actuators, which can be created using ad-hoc connected general purpose devices.
- A procedure to automatically calibrate the positions of sensors and actuators without using calibration objects, neither 3D objects or planes nor virtual ones. Thus we do not need any special equipment such as a laser pointer. Also the setup does not have to be synchronized. Our method is simple and convenient to use and offers mobility of the entire setup. The camera views are assumed to overlap only partly, i.e. not all cameras share a common view.
- The usage of an active display as our calibration target for the intrinsic calibration. Hence our system does not require any special calibration target to be printed. Moreover we have control over the calibration pattern to be displayed and hence extraction of feature points is easier and more reliable.
- Control points and point correspondences accross images are extracted fully automatically. Our solution works for cameras whose viewpoint differ less that 15°.

The rest of the paper is organized as follows. In Section 2 we formulate the problem and present our solution. Section 3 explains how point features are extracted and tracked between images. In Section 4 and Section 5 the camera calibration of the multiple cameras is described. Results on simulated and real-world experiments are presented. In Section 4 the one-by-one calibration of the intrinsic parameters of each camera is outlined. Section 5 presents the algorithm used to determine the extrinsic parameters, i.e. the position of all cameras in a common coordinate system, based on the intrinsic parameters determined in Section 4 and detected point correspondences acrosse the camera images. The estimation of the position and orientation of the flat-panel displays in the 3D scene is described in Section 6. The paper concludes with an outlook in Section 7.

2. PROBLEM FORMULATION

Given M cameras and N flat-panel displays, the goal is to determine the cameras internal parameters and the three dimensional position and pose of the cameras and the flat-panel displays automatically. Therefore we make the following assumptions:

- At any instant we know the number of visual sensors and actuators in the network.
- The displays are sufficiently flat.
- The flat-panel displays' dimensions, i.e. their width and height are known in pixels.

In this work we use an enhanced perspective model considering also lens distortions which describes cameras sufficiently for most applications.
Without considering lens distortion, the mapping performed by a perspective camera between a 3D point $\mathbf{X}$ and a 2D image point $\mathbf{x}$, both represented by their homogeneous coordinates, may be represented by a 3×4 matrix, the camera projective matrix $\mathbf{P}$:

$$\mathbf{x} \simeq \mathbf{P}\mathbf{X} \tag{1}$$

The matrix $\mathbf{P}$ can be written as

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \tag{2}$$

where $\mathbf{K}$ is a 3×3 upper triangular matrix containing the camera intrinsic parameters:

$$\mathbf{K} = \begin{pmatrix} f_x & s & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3}$$

The parameters f_x and f_y are the focal length measured in the width and height of the pixel, p_x and p_y are the coordinates of the principal point in terms of pixel dimensions and s denotes the skew. For most commercial cameras, and hence below, the skew can be considered to be zero. The 3×3 rotation matrix $\mathbf{R}$ and the 3×1 translation vector $\mathbf{t}$ in Equation 2 describe the 3D position and pose of the camera in a certain coordinate system.
In real cameras this perspective model is not sufficient, some desktop cameras exhibit significant distortions. In order to develop a proper model, the perspective model has to be enriched by some distortion components. Our distortion model is chosen according to the model presented in [9]. We account with two coefficients for tangential and radial distortion. This is sufficient to describe distortions that occur in practice.

One goal of our work is to calibrate multiple cameras given a set of corresponding points across their images. Points are said to correspond if they image the same scene point in different views. We assume in the following discussion, that the distortion parameters of each camera are known and the effects of those have been removed from all images. Thus the projective matrix for each camera has to be determined in a common coordinate frame.
As different views of the same scene are related to each other, these relations are used for the calibration. For the remainder of this paper the scene is restricted to be static and for the purpose of mathematical abstraction to consist of points only.
The problem of calibrating multiple cameras is illustrated in Fig. 2. A set of 3D points $\mathbf{X}_i$ is viewed by a set of cameras

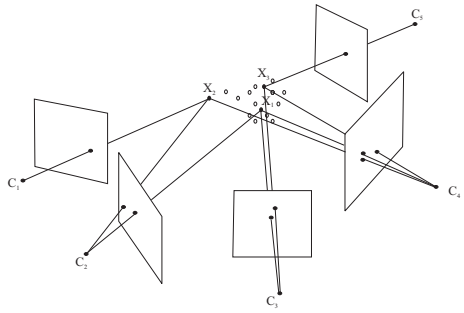


Figure 2: General calibration problem

with matrices $\mathbf{P}^j$. However a 3D point may not be visible in all cameras, thus its corresponding projected point will not be available in all image. Let $\mathbf{x}_i^j$ denote the coordinates of the i-th point as detected in the j-th camera image. The reconstruction problem is then to find the set of camera matrices $\mathbf{P}^j$ and points $\mathbf{X}_i$ such that

$$\mathbf{x}_i^j \simeq \mathbf{P}^j \mathbf{X}_i \tag{4}$$

However, unless additional constraints are given, it is in principle only possible to reconstruct a set of points, or equivalent, to determine the camera matrices, up to a projective ambiguity. Additional constraints arise from knowledge about the cameras (intrinsic or extrinsic parameters) and/or the scene, and can be used to restrict the projective ambiguity. A reconstruction up to an affine, metric or Euclidean transformation can be achieved.

Solution: We solve the camera and display calibration problem in two stages. In a first stage we determine the cameras intrinsic parameters and their position relative to each other up to a global scale factor. Intrinsic calibration is done independently for each camera by using a flat-panel display as a planar calibration object. This is reasonable since our cameras do not allow for varying internal parameters. To accomodate general camera setups it is not required that the calibration object is visible from all cameras at the same time. Then the position estimation is performed in the extrinsic calibration algorithm, i.e. cameras' positions and poses are computed in a common coordinate system. In a typical distributed camera environment each camera can only see a small volume of the total viewing space and different intersecting subsets of cameras share different intersecting views. Hence multiple camera calibration is performed by calibrating subsets of cameras and then building a global coordinate system from individual overlapping views.
In the second stage the display positions in the scene and their respective orientations are estimated by actively displaying a known patterns on the screens. Since all our feature points lie on planes we estimate the homography between each display and the camera image in which the screen is best visible. From those homographies we can extract the position and orientation with respect to the specific camera. Since the position of each camera is known in a common coordinate system, we then know the position and pose of the display with respect to any other camera.

3. POINT CORRESPONDENCES

2D point correspondences between projections of the same 3D point onto different camera planes, i.e. camera images, can be generally used to recover the calibration matrices of the cameras. Therefore establishing such correspondences is the first step in determining the cameras' intrinsic parameters and their relative positions.
To determine point correspondences each image is at first described and represented by a set of features, each describing a specific image point, also interest point, and its neighborhood. Subsequently the features of each image are input to a matching procedure, which identifies in different images features that correspond to the same point in the observed scene.
There are various approaches for extracting a set of interest points and features from an image. Our approach uses the so called SIFT (Scale Invariant Feature Transform)-features proposed in [12] to establish correspondences. SIFT-based feature descriptors were identified in [13] to deliver the most suitable features in the context of matching points of a scene under different viewing conditions such as different lighting and changes in 3D viewpoint.

SIFT-Features Extraction: The SIFT-feature extraction method combines a scale invariant region detector and a descriptor based on the gradient distribution in the detected regions.
The following steps are performed for computing a set of features of an image [12]:

- Scale-space extrema detection
- Accurate keypoint localization
- Orientation assignment
- Keypoint descriptor calcultaion

The first three steps detect a set of interest points - also called keypoints. They are characteristic for a specific image. The keypoints are filtered to preserve only those, which are stable under a certain amount of additive noise. An image location, scale and orientation is assigned to each keypoint. This enables the construction of a repeatable local 2D coordinate system, in which the local image (pixel and its surrounding region) is described invariantly from these parameters based upon image gradients.
However this approach has its limitations. To ensure a sufficient number of reliable matching points, the displacement between the cameras should not exceed 15°. The resulting correspondences are within pixel accuracy.

SIFT-Feature Matching: Point-to-point correspondences

between two images are established by first extracting a set of SIFT-features from each image and subsequently comparing their keypoint descriptors. The matching technique used for the SIFT-features has also been proposed in [12].
The matching is performed by first individually measuring the Euclidean distance of each feature vector (representing a certain keypoint) of one image to each feature vector of the other image. Then the best matching candidate for a specific keypoint is identified by the interest point in the other image with the minimum distance. A valid match is found in the second image, if the distance ratio between the nearest and the second nearest neighbor (closest/next closest) is below a certain threshold. This validation is necessary due to features caused by noise or perspective phenomenons that match incorrectly in the other image.
Fig. 3 shows an example of matched points between two images.

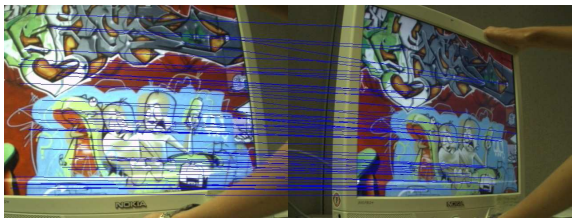


Figure 3: Matched feature points visualized ny a connecting line between images of the same scene

Subpixel Accuracy: The matching result of Lowe's algorithm is only at pixel accuracy. For position estimation of multiple cameras extensive experiments have shown that it is essential to keep all computations at a subppixel accuracy level.
So far the approximate positions of the corresponding points are known and can be used as an initial guess about the exact position. Subpixel precision can then be achieved through the Affine Lucas Kanade Feature Tracker [16]. This feature tracker assumes for simplicity reasons an affine transformation between the viewpoint of both images. This approximation is valid, if the viewpoints of the different cameras are sufficiently close together.
The basic optimization problem which is solved by the feature tracker is the following:

$$\min_{\mathbf{d},\mathbf{D}} \sum_{x=-\omega_x}^{\omega_x} \sum_{y=-\omega_y}^{\omega_y} \left(I(\mathbf{x}+\mathbf{u}) - J((\mathbf{D}+\mathbf{I_{2\times 2}})\mathbf{x}+\mathbf{d}+\mathbf{u})\right)^2 \quad (5)$$

where $I(\mathbf{u})$, $J(\mathbf{u})$ represent the grey-scale values of the two images at location $\mathbf{u}$, the vector $\mathbf{d} = [d_x \quad d_y]^T$ is the optical flow at location $\mathbf{u}$, and the matrix $\mathbf{D}$ denotes an affine deformation matrix characterized by the four coefficients $d_{xx}, d_{xy}, d_{yx}, d_{yy}$:

$$\mathbf{D} = \begin{pmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{pmatrix} \quad (6)$$

The objective of affine tracking is then to choose $\mathbf{d}$ and $\mathbf{D}$ in a way that minimizes the dissimilarity between feature windows of size $2\omega_x + 1$ in x and size $2\omega_y + 1$ in y direction around the point $\mathbf{u}$ and $\mathbf{v}$ in I and J respectively. $\mathbf{v}$ denotes here the corresponding point of $\mathbf{u}$ of image I in image J and can be expressed in terms of $\mathbf{u}$ according to $\mathbf{v} = \mathbf{u}+\mathbf{D}\mathbf{u}+\mathbf{d}$. To handle changes in brightness and contrast a normalization is applied to the image patches in the iteration process.
An example result obtained by the subpixel feature tracking algorithm is shown in Fig. 4. The corresponding feature points shown in Fig. 3 were used to initialize the algorithm. It can be seen, that the matching points are shifted to a slightly different position by the algorithm. The improvment in accuracy is especially obvious in the region marked with a circle in both images. SIFT-feature matching in the case of two very close points in the first image resulted in the same feature in the second image. With this initial guess the Lucas Kanade feature tracker finds the two different corresponding points and herewith it significantly improves the accuracy of the image matching process.

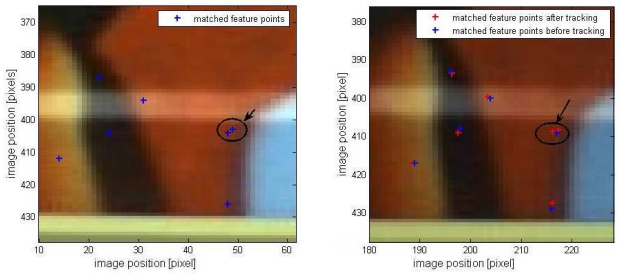


Figure 4: Matched feature points before and after the tracking algorithm (points in the left image were taken as reference and tracked in the right image)

4. INTRINSIC CALIBRATION

Intrinsic calibration is done independently for each camera. This is reasonable since our cameras do not allow for varying internal parameters. To estimate the internal parameters , the C++ implementation of J.-Y. Bouguets Camera Calibration Toolbox [3] in OpenCV [2] is used. It uses a slightly modified version of Zhang's method [20].
The calibration algorithm requires to record images of a known planar calibration target at a few (at least two) different orientations for each camera, where the motion of the different poses need not to be known. Therefore the target is freely moved in front of each cameras separately.
As the 2D geometry of the calibration plane is known with very high precision, the camera observes in each image the projection of a set of control points with known position in some fixed world coordinate system. Camera calibration is achieved by minimizing the mean-squared distance between measured feature points in the image and their theoretical position with respect to the camera's intrinsic and extrinsic parameters. If n images of a model plane are taken and each image gives rise to m correspondening points with the calibration pattern, then the Maximum Likelihood estimate (assuming measured points are corrupted by independent Gaussian noise) can be obtained by minimizing the following function with respect to the complete set of calibration parameters:

$$\varepsilon = \sum_{j=1}^{n} \sum_{i=1}^{m} \parallel \mathbf{x}_i^j - \hat{\mathbf{x}}(\mathbf{K}, \kappa_1, \kappa_2, \rho_1, \rho_2, \mathbf{R}_j, \mathbf{t}_j, \mathbf{X}_i) \parallel^2 \quad (7)$$

where $\hat{\mathbf{x}}(\mathbf{K}, \kappa_1, \kappa_2, \rho_1, \rho_2, \mathbf{R}_j, \mathbf{t}_j, \mathbf{X}_i)$ is the theoretical posi-

tion of the projection of point $\mathbf{X}_i$ in the image j including distortion effects described by the distortion coefficients $\kappa_1, \kappa_2, \rho_1, \rho_2$. This results in a non-linear optimization problem; a proper initialization of the calibration parameters is required. Thus the complete calibration procedure consists of an initialization stage, where a closed form solution is computed, followed by a nonlinear refinement based on the Maximum Likelihood criterion. For a more in detail discussion of both stages the reader is reffered to [20] and [3].

4.1 Control Point Extraction

In the calibration procedure several different perspectives of a planar model object of known geometry are fed into the calibration routine. We used the pattern shown in Fig. 6 (right) as our planar model object, since it gives rise to a large number of stable SIFT-features. It is displayed on a laptop or any other flat screen of known size, whose surface can be assumed to be sufficiently flat. A number of different images of the model plane are then captured by waving the screen in front of the camera. Some example images of the plane under different orientations are shown in Fig 5.

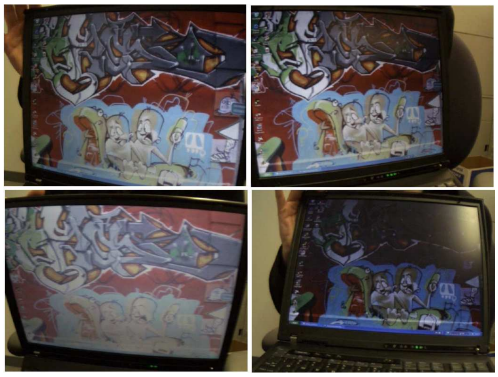


Figure 5: Four sample images of the model plane used for calibration

Projected points from the pattern in the images are then determined by extracting SIFT-features from each view and matching them with the features extracted from the calibration pattern. Subpixel matching was not necessary to obtain sufficiently accurate results. A matching example between the calibration pattern and an image of the model plane is illustrated in Fig. 6.

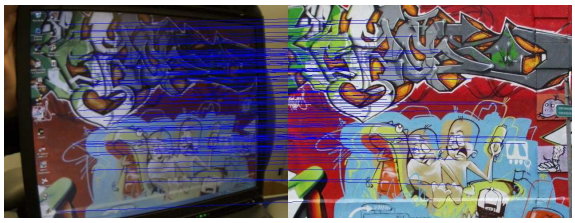


Figure 6: Example match between calibration pattern (right; from [1]) and imaged calibration plane (left)

One remark has to be made about the use of SIFT-features in the field of intrinsic camera calibration: Usually in other calibration procedures ([3], [19]) a checkboard pattern is used requiring some user interaction to obtain matching points (in this cases corners) between the calibration object and the (different) image(s) of the calibration object. With the procedure described here, the subset of image points can be easily detected automatically. Another important advantage of using a flat screen displaying a known pattern together with extracting SIFT-features is that feature matching can be performed also with images containing only a partially visible image of the model plane.

4.2 Experimental Results

In order to evaluate the calibration routine, the algorithm was applied to a different number of images of the model plane. The results are shown in Tabel 1. As the number of pattern points that could be extracted varies per view, only the total number of points used in the calibration procedure is given in the second row of the table for each experiment.

# img	20	30	40	50	60
# pnts	2218	3735	5171	6735	8039
$\mathbf{f}_x$	818.38	831.29	834.17	836.79	838.16
$\mathbf{f}_y$	818.16	830.87	833.64	836.38	837.98
$\mathbf{p}_x$	305.02	307.19	308.77	307.69	308.51
$\mathbf{p}_y$	263.87	257.35	255.35	255.32	254.48
κ_1	−0.421	−0.432	−0.437	−0.433	−0.436
κ_2	0.092	0.108	0.126	0.101	0.111
ρ_1	-0.005	-0.003	-0.002	-0.002	-0.002
ρ_2	0.004	0.003	0.002	0.003	0.003

Table 1: Results obtained for the intrinsic parameters of a camera

The influence of the number of images used for the calibration with respect to the performance of the optimization procedure was investigated in [20]. It was found that the estimation error decreases with an increasing number of images of the model plane. This effect can also be observed in Table 1. For 40 or more images the estimated intrinsic parameters are consistent between the experiments, whereas in the case of only 20 views the calculated values show a relatively large deviation from the calculated values for more views. Compared to algorithms using corner features, the number of views necessary for reliable calibration is higher, as corner correspondences can be extracted more accurate than SIFT feature correspondences.
Once the intrinisc parameters are determiend the distortion in the original images can be corrected as shown in Figue 7.

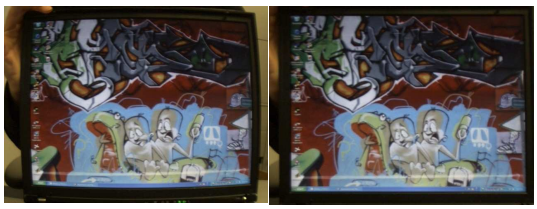


Figure 7: Original and recified image

5. EXTRINSIC CALIBRATION OF MULTIPLE CAMERAS

The main objective of the algorithm presented in this section is to recover the 3D positions and poses of a multiple cameras in a common coordinate system in a fully automatic manner from the captured images of the different cameras. The considered situation is illustrated in Fig. 2. As already metioned in Section 2 we can describe the mapping of $\mathbf{X}_i$ to $\mathbf{x}_i^j$ according to $\mathbf{x}_i^j \simeq \mathbf{P}^j\mathbf{X}_i$ under the assumption, that radial/tangential distortion effects have been already removed. In general a 3D point $\mathbf{X}_i$ might only be observed by a subset of cameras, so the corresponding projected point will not appear in all views.
Since we know the intrinsic parameters of all cameras in the scene, the locations of the projected points can be given in normalized image coordinates, denoted in the following with $\mathbf{x_n}_i^j$. The normalized image coordinates $\mathbf{x_n}$ of a point $\mathbf{x}$ are derived by removing the effects of the internal parameters from the measured image point. The idea is, that the normalized point $\mathbf{x_n}$ is the image of the point $\mathbf{X}$ with respect to a camera that has the identity matrix $\mathbf{I_{3\times 3}}$ as its calibration matrix $\mathbf{K}$. A normalized point may be computed from the image measurement via:

$$\mathbf{x_n} \simeq \mathbf{K}^{-1}\mathbf{x} \tag{8}$$

if the calibration matrix $\mathbf{K}$ of the particular camera is known. The mapping between a point $\mathbf{X}_i$ to its projected and normalized point $\mathbf{x_n}_i^j$ in the j-th image is then described by the normalized camera matrix $\mathbf{P_n}^j$:

$$\mathbf{x_n}_i^j \simeq \mathbf{P_n}^j\mathbf{X}_i \tag{9}$$

where

$$\mathbf{P_n}^j = \mathbf{K}_j^{-1}\mathbf{P}^j = \mathbf{K}_j^{-1}\mathbf{K}_j[\mathbf{R}_j|\mathbf{t}_j] = [\mathbf{R}_j|\mathbf{t}_j] \tag{10}$$

The normalized camera matrix $\mathbf{P_n}^j$ only consists of a rotation $\mathbf{R}_j$ and a translation $\mathbf{t}_j$.
In summary, our goal is to find the appropriate set of normalized camera matrices $\mathbf{P_n}^j$ and points $\mathbf{X}_i$ such that $\mathbf{x_n}_i^j \simeq \mathbf{P_n}^j\mathbf{X}_i$, based on a set of image correspondences, which are represented by normalized coordinates $\mathbf{x_n}_i^j$. The relative position of the cameras may be computed uniquly up to an indeterminated overall scale factor.

5.1 Algorithm

There are several strategies for solving the multiple camera calibration problem. The superior method is bundle adjustment. The process of bundle adjustment is an estimation involving the minimization of the reprojection error. The reprojection error is defined as the (summed squared) distances between the theoretical image positions of the projections of the estimated 3D points $\hat{\mathbf{X}}_i$ and the measured image points. Bundle adjustment can handle missing correspondences, which appear, if only a subset of cameras shares a common view. However, it involves a nonlinear optimization process and thus it does not have a direct solution. A sufficiently good starting point is required.
We use a hierarchical method to obtain an initial guess for all camera matrices $\mathbf{P_n}^j$ and the 3D scene represented by the points $\mathbf{X}_i$. The method is mainly based on the approach presented in [5]. It partitions the set of cameras into manageable subgroups that share a common view. A coordinate system is build for each of these subgroups and then, based on points and cameras being common to different subsets, these different coordinate frames are merged in a hierarchical fashion in order to build a global coordinate system from the individual overlapping systems.
The main advantage contributed by a hierarchical procedure is that the error can be distributed evenly over the entire set of estimated camera matrices. As in [5], we also use image triplets as the basic building block. The cameras' positions in such a basic unit and the structure of the scene observed by three cameras can be computed automatically by calculating the associated trifocal tensor from point-to-point correspondences across the three views. Then the triplets are registered into sub-groups, followed by merging these subsets and thus building the entire group. This situation is illustrated in Fig. 8.

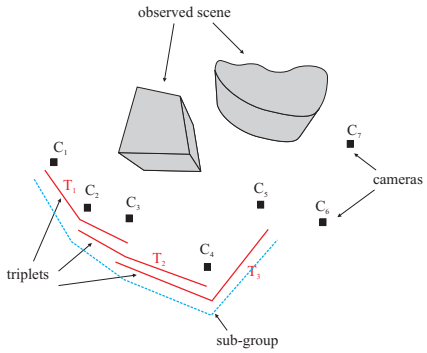


Figure 8: Camera positions and the structure of the scene are computed by registering the basic building block (triplets of cameras) into subgroups

The first task is to segment the set of cameras into appropriate subgroups and those subsets into triplets. Consequently neighboring cameras with sufficient view overlaps have to be determined. Requirements are that cameras building a triplet share a common view and a sufficient number of point correspondences over the three images are available in order to compute the trifocal tensor reliably. Therefore SIFT-feature points are extracted from each image. Then the number of correspondences in each possible triplet combination is computed through a matching procedure. Only triplets with sufficient corresponding points are kept.
Next the triplets need to be clustered into subgoups. The triplets in a certain subgroup and also different sub-groups need to overlap by two cameras to enable stitching them together into a single coordinate frame. Therefore combinations of the above determined triplets are tested.
In the next stage the cameras in each triplet have to be calibrated extrinsically. Robust ways of computing the trifocal tensor and extracting the according camera matrices based on corresponding image points have been extensively studied in [8]. Point correspondences are established by representing each image by its SIFT-feature points. One of the three images of a triple is arbitrarily selected to be the reference image. Two-view point correspondences between this reference image and each of the other two images are then determined by SIFT-matching. These correspondences are refined by using them as initializations for the Affine Lucas Kanade feature tracker. The required three-view correspondences are derived by joining the two-view match sets.
Features which arise from moving objects and are therefore not appropriate for the reconstruction task (as the cameras are not synchronized) can be simply eliminated by observing the location of each feature point over time and eliminating

all temporally unstable SIFT features.
However, if the scene of interest is untextured or contains only sparse textures, then too few point features may be extracted from the acquired images, which may lead to unreliable results in the trifocal tensor estimation. Thus, the observed 3D scene needs to be sufficiently textured in order to ascertain detection and tracking of enough image features.

To proceed from triplets to a description of the complete set of cameras, it is necessary to register all triplets into the same coordinate frame. This is done, as already mentioned, in a hierarchical manner as proposed in [5]. Registration of triplets and sub-groups is achieved by computing a homography of 3-space between the different metric structures.
The considered situation is illustrated in Fig 9. The objective is hereby to obtain a common set of 3D points and a normalized camera matrix for each view, such that the reprojection error is minimized.

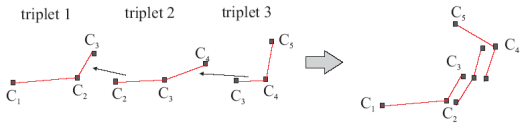


Figure 9: Registration of triplets 2 and 3 in the metric frame of triplet 1

In the following only the registration of two triplets is discussed. All registration problems in the algorithm are analogously solved. Specifically a pair of triplets is considered, where the triplets have exactly two cameras in common. In general different overlaps are possible, e.g. one or zero common cameras. As the implementation in this work specifically forces the triplets in a subgroup, and also the different subgroups, to have two cameras in common, only this case is discussed here. For a detailed description of the other cases and an evaluation of the different registration methods, the reader is referred to [5].
Given some 3D points common in both sets and the homogeneous representation of these points by $\mathbf{X}_i$ in the first frame and $\mathbf{X'}_i$ in the second frame (the inhomogeneous representation are denoted with $\bar{\mathbf{X}}_i$, $\bar{\mathbf{X}}'_i$ in the following), the point representations in the different metric frames are related by a 3-space homography $\mathbf{H}$ according to:

$$\mathbf{X}_i = \mathbf{H}\mathbf{X}'_i \tag{11}$$

Equivalently ${\mathbf{P_n}}^j = {\mathbf{P'_n}}^j\mathbf{H}^{-1}$ holds for the corresponding normalized projection matrices of the cameras common to both triplets. As both triplets are determined up to a metric ambiguity, the homography between the two metric frames can be described by a metric transformation:

$$\mathbf{H} = \begin{pmatrix} \sigma r_{11} & \sigma r_{12} & \sigma r_{13} & t_1 \\ \sigma r_{21} & \sigma r_{22} & \sigma r_{23} & t_2 \\ \sigma r_{31} & \sigma r_{32} & \sigma r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{12}$$

with r_{ij} being the coefficients of a rotation matrix $\mathbf{R}$ and t_i being the coefficients of a translation vector $\mathbf{t}$. σ identifies the relative scale between the structure. Therefore the transformation between the two different metric frames counts 7 unknowns. Two stages are used to derive accurate estimates for those parameters: first a closed-form solution is obtained, which is then further refined in a nonlinear stage.
In order to compute a direct solution for the 7 parameters, the first step is to estimate the relative scale σ. Therefore the centroid of the each structure (consisting of the common 3D points (in inhomogeneous coordinates) $\bar{\mathbf{X}}_i$, $\bar{\mathbf{X}}'_i$ respectively) denoted with inhomogeneous coordinates $\bar{\mathbf{M}}$, $\bar{\mathbf{M}}'$ is computed. Then the distance of each point in the structure to its centroid is calculated. The relative scale between the two structures is determined by the quotient of the mean distances:

$$\sigma = \frac{\frac{1}{n}\sum_{i=1}^{n} \parallel \bar{\mathbf{X}}_i - \bar{\mathbf{M}} \parallel}{\frac{1}{n}\sum_{i=1}^{n} \parallel \bar{\mathbf{X}}'_i - \bar{\mathbf{M}}' \parallel} \tag{13}$$

where $\parallel \cdot \parallel$ denotes the L2-norm and n is the number of common points in both triplets. Thus the second structure may be rescaled according to:

$$\bar{\mathbf{X}}'_{\mathbf{s}i} = \sigma\bar{\mathbf{X}}'_i \tag{14}$$

so that Equation 11 becomes

$$\mathbf{X}_i = \mathbf{H_s}\mathbf{X}'_{\mathbf{s}i} \tag{15}$$

where

$$\mathbf{H_s} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix} \tag{16}$$

In order to obtain an initial estimate for the coefficients of $\mathbf{R}$ and $\mathbf{t}$ the squared distance between these two structures is minimized with respect to the coefficients of $\mathbf{H_s}$ using linear algebraic methods:

$$\min_{\mathbf{R},\mathbf{t}} \sum_i d(\mathbf{X}_i, \mathbf{H_s}\mathbf{X}'_{\mathbf{s}i})^2 \tag{17}$$

where $d(\mathbf{x}, \mathbf{y})$ denotes the Euclidean distance between the inhomogeneous points corresponding to $\mathbf{x}$ and $\mathbf{y}$.
Finally this is followed by a nonlinear minimization stage in order to refine the above derived initial values. This nonlinear estimation minimizes the reprojection error to the original measured and normalized image points with respect to all parameters of $\mathbf{H}$.

$$\min_{\sigma,\mathbf{R},\mathbf{t}} \sum_{ij} d^2({\mathbf{P_n}}^j\mathbf{H}\mathbf{X'}_i, {\mathbf{x_n}}_i^j) + d^2({\mathbf{P'_n}}^j\mathbf{H}^{-1}\mathbf{X}_i, {\mathbf{x_n}}_i^j) \tag{18}$$

$d(\mathbf{x}, \mathbf{y})$ is here the Euclidean image distance between the inhomogeneous points corresponding to $\mathbf{x}$ and $\mathbf{y}$. This nonlinear minimization is solved using a standard techniques, the Levenberg-Marquardt algorithm.
By registering all triplets hierarchically as described above in one common coordinate frame an initial guess for the observed 3D structure (represented by 3D points) and all normalized camera matrices in the entire set of cameras is obtained. Finally a Maximum Likelihood estimate (assuming independent Gaussian noise in the point measurements) of the entire set of camera positions and the 3D-structure is computed via bundle adjustment:

$$\min_{\hat{\mathbf{P}}_{\mathbf{n}}^j, \hat{\mathbf{X}}_i} \sum_{ij} d^2(\hat{\mathbf{P}}_{\mathbf{n}}^{\mathbf{j}}\hat{\mathbf{X}}_i, {\mathbf{x_n}}_i^j) \tag{19}$$

where $d(\mathbf{x}, \mathbf{y})$ denotes here the Euclidean image distance between the homogeneous points $\mathbf{x}$ and $\mathbf{y}$ (for 3D points that are visible in the specific camera image).
Each normalized camera matrix is parameterized by 6 entries, 3 representing the rotation matrix and 3 representing

Figure 10: Some images of the observed office scene taken from the different viewpoints of the cameras

the translation vector. The dimension of the minimization problem adds then up to a total number of $6(M-1)$ parameters for the camera matrices, plus a set of $3L$ parameters for the coordinates of the L reconstructed 3D points.

5.2 Experimental Results

The extrinsic camera calibration algorithm has been implemented for the case of 11 cameras; the size of the subgroups was chosen to be five cameras. Triplets in subgroups and different subgroups have two cameras in common. Fig. 10 shows some of the images taken from the different viewpoints of the cameras. It should be mentioned, that the change of viewpoint between the different locations of the cameras is relatively small. This is due to the matching algorithm, which requires a change of viewpoint less than 15° between the images in order to ensure reliable matching and a sufficient number of corresponding points. A resulting configuration for one subgroup of 5 cameras is shown in Fig. 11 (a),(b) from two different viewpoints. Camera positions are marked with yellow pyramids, reconstructed scene points with blue dots. In Fig. 11 (c) the reprojection error is illustrated. Reprojected points are marked with blue crosses, measured and normalized points with red circles. As can be seen the estimation is highly accurate.
The resulting camera positions and scene reconstruction for all 11 cameras are shown in Fig. 12 from two different perspectives together with the reprojection error for one estimated cameras. The distance between the reprojected points and the measured image points is very small, therefore the overall estimation is highly accurate, again.

Discussion: In the given examples the implementation performs very well. However experiments with different real data sets have shown that sporadically the accuracy of the algorithm can be affected significantly, consequently the estimated cameras and scene point configuration do not represent the actual scene. Thorough analysis showed that misestimations were caused by inaccurately estimated triplets. If the camera positions and/or the reconstructed 3D points of one triplet are estimated inaccurately, the homography estimation to register these triplet in a sub-group fails as well. As a result the whole sub-group configuration is determined incorrectly leading to an initial guess for the entire group too far away from the actual value. As the optimization problem of the final bundle adjustment is of very high dimension, a poor initial guess commonly results in the nonlinear optimization to fail completely, i.e. not to converge at all or to converge to a suboptimal solution.
The sources of the failure in the triplet estimation may be that corresponding image points are not extracted accurately enough, due to the performance limits of the feature extraction and matching algorithm and/or the feature tracker. Those algorithms are only partly invariant to perspective transformations or even assume affine transformation between the images. Another cause of failure arises from the fact, that the intrinsic camera parameters can also only be estimated with a certain accuracy. This may have also an impact on the noise level in the corresponding normalized points.

6. POSE ESTMATION OF DISPLAYS

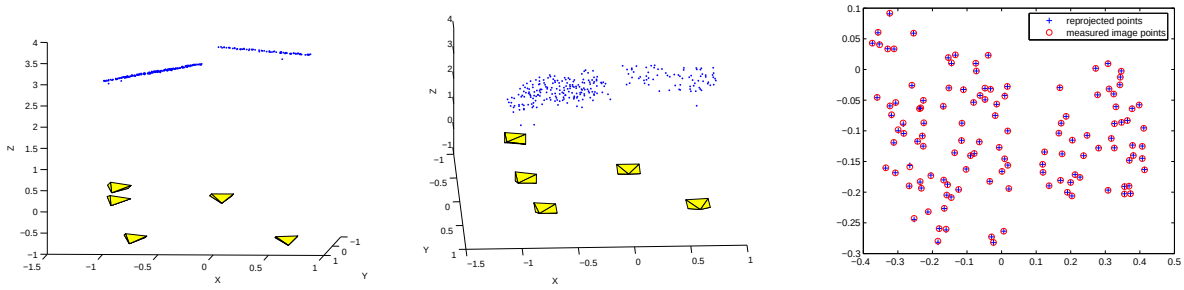
So far only the cameras have been calculated. The task is now to estimate the pose of the flat-panel display in the 3D scene. The pose of an object is defined as its position and orientation in a certain coordinate system.
The following assumptions are made:

- The flat-panel displays are active, i.e. they may be used to display some known pattern. Feature points can be extracted from this pattern, whose relative geometry on the display is known.
- The display is at least partially visible in one camera and a sufficient number of feature points of the displayed pattern can be detected in this camera image.
- All cameras in the scene are extrinsicly and intrinsicly calibrated.

The algorithm requires the screen only to be visible partly in one of the cameras. Therefore if the screen is visible in more than one camera, the screen's position is calculated with respect to the camera from which it can be detected 'best' ('best' is defined later). As the relative position of all N cameras to each other is already known up to a global scale factor, the pose of the specific flat screen with respect to all other cameras is then also determined up to a scale factor. However, if at least two points on a display can be detected in two different cameras, it is possible to determine the overall scale factor of the whole structure, as the absolute distance of those two points on the screen in pixels is known.

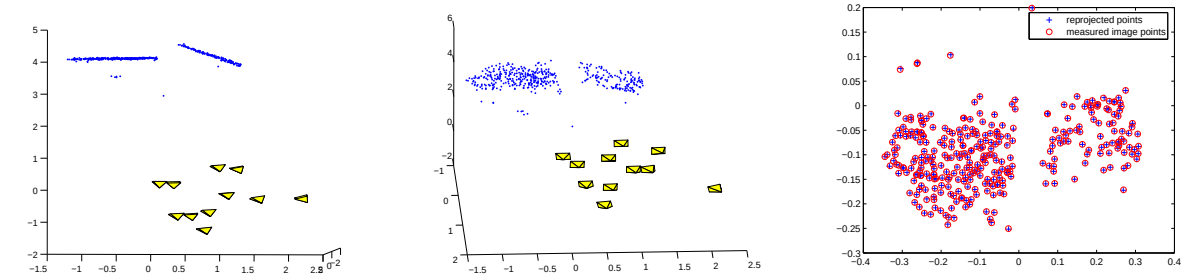
6.1 Algorithm

The considered situation is illustrated in Fig. 13. A known pattern is displayed on the flat-panel screen. Feature points are extracted from the known pattern and the image captured by the camera, and then matched. Thus point correspondences between object points $\mathbf{X}_i$, whose positions in a certain coordinate frame attached to the object are known, and points $\mathbf{x}_i$ in the image can be established.
The relation between the homogeneous coordinates of a 3D point in the object referenece frame attached to the grid (Fig. 13), denoted with $\mathbf{X}_i^O$, and its projection in the image is described by Equation 1, 2 (assuming distortion effects have been removed from the image measurements):
$\mathbf{x}_i \simeq \mathbf{P}\mathbf{X}_i^O = \mathbf{K}[\mathbf{R}|\mathbf{t}]\mathbf{X}_i^O$. $\mathbf{t}$ denotes here a translation vector and $\mathbf{R} = [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3]$ is a 3×3 rotation matrix, where $\mathbf{r}_1, \mathbf{r}_2$ and $\mathbf{r}_3$ are 3×1 vectors representing the rotation axes. The objective of the presented algorithm is now to estimate $\mathbf{R}$ and $\mathbf{t}$ that define the relative position of the active planar screen, i.e. the pattern, with respect to the camera, given



(a) (b) (c)

Figure 11: Two different views of the reconstructed 3D scene points and camera positions for a subgroup of five cameras (a), (b) and the reprojection error for one of the five camera images (c).



(a) (b) (c)

Figure 12: Two different views of the reconstructed 3D scene points and camera positions for the entire group of 11 cameras (a), (b) and the reprojection error for one of the 11 camera images (c).

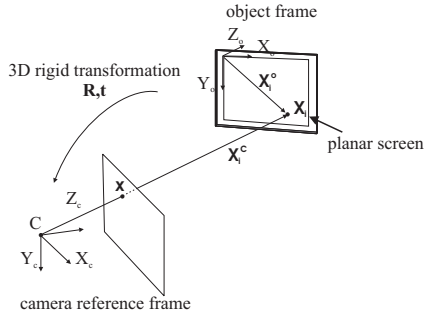


Figure 13: Pose estimation of a planar object of known geometry

the position of feature points on the display, above denoted with $\mathbf{X}_i^O$ and their measured projections onto the image $\mathbf{x_{n}}_i$. The projected coordinates $\mathbf{x}_i$ can now be normalized with respect to the intrinsic parameters, assumed that those are known. In the following the normalized image points are denoted with $\mathbf{x_{n}}_i$. According to Equation 9 and 10, the point $\mathbf{X}_i^O$ is then projected onto $\mathbf{x}_n$ by:

$$\mathbf{x_{n}}_i \simeq= [\mathbf{R}\ \mathbf{t}]\mathbf{X}_i^O \tag{20}$$

In practice Equation 20 will not be exactly satisfied due to noisy image point measurements. Assuming that the extracted image points are corrupted by additive independent white Gaussian noise, the Maximum Likelihood estimate can be derived by minimizing the reprojection error

$$\varepsilon = \sum_{i=1}^{n} \parallel \mathbf{x_{n}}_i - \hat{\mathbf{x}}_{\mathbf{ni}}(\mathbf{R}, \mathbf{t}, \mathbf{X}_i^O) \parallel^2 \tag{21}$$

with respect to $\mathbf{R}$ and $\mathbf{t}$. $\hat{\mathbf{x}}_{\mathbf{ni}}(\mathbf{R}, \mathbf{t}, \mathbf{X}_i^O)$ denotes here the theoretical position of the projection of a point $\mathbf{X}_i^O$ according to Equation 20.

Obviously this is a non-linear minimization problem, which must be solved iteratively. A standard techniques is the Levenberg-Marquardt algorithm. An approximate pose, which is sufficiently close to the actual position, must be provided to initiate the iteration process.

An initialization is found by a closed-form solution. Since all feature points are positioned on a plane, a homography between the display and the image plane may be estimated. Without loss of generality it is assumed, that the display lies in the plane Z=0 of the object reference frame (see Fig. 13). Then Equation 20 becomes:

$$\mathbf{x_{n}}_i \simeq [\mathbf{R}\ \mathbf{t}]\mathbf{X}_i^O = [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3\ \mathbf{t}] \begin{pmatrix} X_i^O \\ Y_i^O \\ 0 \\ 1 \end{pmatrix} = \mathbf{H} \begin{pmatrix} X_i^O \\ Y_i^O \\ 1 \end{pmatrix} \tag{22}$$

where $\mathbf{H} = [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{t}]$ First this 3×3 homography $\mathbf{H}$ is estimated. Then in a second step a rotation matrix $\mathbf{R}$ and a translation $\mathbf{t}$ are extracted from $\mathbf{H}$ to obtain an initial guess on the pose parameters.

In general such a 3×3 homography has 8 degrees of freedom and can be estimated with linear methods up to a scale

factor by using Equation 22, if four point-to-point correspondences between the displayed pattern and the captured image can be found. However, if more than four feature points are used, insensitivity to measurement errors and image noise is added and the initial guess may be closer to the actual parameters, which finally may give a more accurate result. Therefore the 'best' camera in a set to pick for the pose estimation of a certain screen, is the camera that can establish the most corresponding points between the known pattern on the display and its image.

6.2 Experimental Results

The algorithm described above has been tested in a real setup. In a first step the point correspondences between the known pattern displayed and the camera image are established. Fig. 14 illustrates the extracted matching points, on the top the camera image is shown, on the bottem the displayed pattern. With these correspondences the position

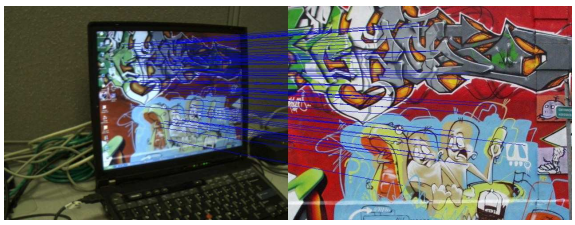


Figure 14: Point correspondences between the displayed pattern and the camera image

and pose of the screen relative to the camera is computed. The result of the reconstruction algorithm is illustrated in Fig. 15.

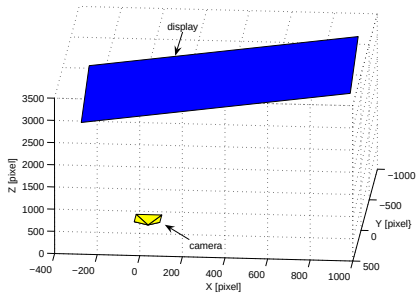


Figure 15: Estimated relative position of the display

7. CONCLUSION

In this paper we have presented a flexible and easy way to calibrate multiple cameras in a distributed platform of GPCs. Our method needs minimal user intervention. Hence the proposed method can be used in a variety of places ranging from single desktop cameras to multi-camera lab setups. We also show how to determine the position of active flat-panel dispalys in the scene. All stages of the calibration algorithms have been implemented in C++ and experimental results on real data showed that the presented methods work very well. As the change in viewpoint between the different cameras is restricted, future work is needed to improve the automatic extraction of point correspondences between images. This will also result in a more robust computation of triplets, leading to an overall robuster estimation of position and pose in the extrinsic calibration routine.

8. REFERENCES

[1] *http://lear.inrialpes.fr/people/Mikolajczyk/.*

[2] Intel corporation. *OpenCV Computer Vision Library*, http://www.intel.com/research/mrl/research/opencv/.

[3] J.-Y. Bouguet. *Camera Calibration Toolbox for Matlab.* http://www.vision.caltech.edu/bouguet/calib_doc/.

[4] X. Chen, J. Davis, and P. Slusallek. Wide area camera calibration using virtual calibration object. In *Proc. CVPR '00*, pages 2520–2527, 2000.

[5] A. Fitzgibbon and A. Zissermann. Automatic camera recovery for closed or open image sequences. In *Proc. European Conference on Computer Vision*, pages 311–326, 1998.

[6] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. 4th Alvey Vision Conference*, pages 147–152, 1988.

[7] R. Hartley. An algorithm for self-calibration from several views. In *Proc. CVPR '94*, pages 908–912, Seattle, USA, 1994.

[8] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge Univ. Press, 2003.

[9] J. Heikkilä and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proc. CVPR '97*, pages 1106–1112, 1997.

[10] P. H. Kelly, A. Katkere, D. Y. Kuramura, S. Moezzi, S. Chatterjee, and R. Jain. An architecture for multiple perspective interactive video. In *Proc. ACM Multimedia '95*, pages 201–212, 1995.

[11] R. Lienhart, I. Kozintsev, S. Wehr, and M. Yeung. On the importance of exact synchronization for distributed audio processing. In *Proc. ICASSP '03*, pages 840–843, 2003.

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal Comput. Vision*, 60(2):91–110, 2004.

[13] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proc. CVPR '03*, volume 2, pages 257–263, 2003.

[14] M. Pollefeys. *Self-Calibration and Metric 3D Reconstruction from Uncalibrated Image Sequences.* PhD thesis, Katholieke Universiteit Leuven, Belgium, 1999.

[15] V. Raykar, I. Kozintsev, and R. Lienhart. Position calibration of audio sensors and actuators in a distributed computing platform. In *Proc. ACM Multimedia '03*, pages 572–581, 2003.

[16] J. Shi and C. Tomasi. Good features to track. In *Proc. CVPR '94*, pages 593 – 600, 1994.

[17] P. Sturm and W. Triggs. A factorization based algorithm for multiple-image projective structure and motion. In *Proc. European Conference on Computer Vision*, pages 709–720, 1996.

[18] T. Svoboda, D. Martinec, and T. Pajdla. A convenient multi-camera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4), 2005. To appear.

[19] R. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lense. *IEEE Journal of Robotics and Automation*, RA-3:323–344, 1987.

[20] Z. Zhang. A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research, Redmond, USA, 1998.