Calibrating Visual Sensors and Actuators in Distributed Platforms

E. Hörster, R. Lienhart
Multimedia Computing Lab
University of Augsburg
Augsburg, Germany

W. Kellermann
LNT
U. of Erlangen-Nuremberg
Erlangen, Germany

J.-Y. Bouguet
Intel Research
Intel Corporation
Santa Clara, CA, USA

Abstract

Many novel multimedia, home entertainment, visual surveillance and health applications use multiple audio-visual sensors and actuators. In this paper we present a novel approach for position and pose calibration of visual sensors and actuators, i.e. cameras and displays, in a distributed network of general purpose computing devices. It complements our work on position calibration of audio sensors and actuators in a distributed computing platform [14]. The approach is suitable for a wide range of possible - even mobile - setups since (a) synchronization is not required, (b) it works automatically, (c) only weak restrictions are imposed on the positions of the cameras and displays, and (d) no upper limit on the number of cameras and displays under calibration 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 and displays and has proven itself to work extremely accurate.

1 Introduction

Many audio-visual sensor and actuator array processing algorithms require precise knowledge about the positions and poses of the sensors and actuators. Current systems achieve that either by place the sensors and actuators at known locations in a known pose or by calibrating them manually, a very tedious process. The omnipresence of microphones, cameras, loudspeakers and displays nearly everywhere in our lives today demands a simple and convenient calibration approach to put all sensors and actuators into a common time and space. [10] proposes a means to provide a common time reference to multiple distributed GPCs. In [14] a method for automatically calibrating audio sensors and actuators is presented. In this paper we focus on providing a common space for multiple cameras and flat panel-displays by actively estimating their 3D position and pose. We also address the problem of effortlessly calibrating the intrinsic parameters of multiple cameras. Fig. 1 shows an sample setup of our distributed computing platform. A room or area is instrumented with $N \geq 3$ static cameras connected to networked GPCs. Further $M \geq 1$ flat-panel screens are at least partially visible from one camera. No precise synchronization of the different devices is required.

Related Work: Camera calibration is a well researched topic in computer vision. Fundamentally there are two different methods of camera calibration: photogrammetric calibration and self-calibration [18]. The first method uses a 3D, a 2D (planar), or a virtual calibration object of precisely known geometry. Important approaches are described in [18] [9] [17] [4]. Planar methods

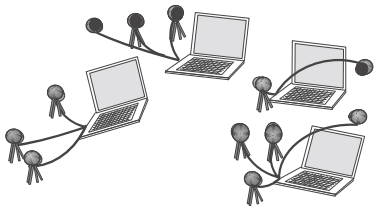

Figure 1. Distributed computing platform with cameras and displays

are very popular because it is easy to obtain a calibration target by just printing the pattern and fixing the paper on a flat surface. Although providing good results, the major drawback of these calibration methods is that they require special equipment or precise manual measurements. Virtual calibration objects are constructed over time by tracking an easily identifiable object through a 3D scene. The cameras usually have to be synchronized and thus the setup requires special equipment. Self calibration techniques ([7][16][13]) 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 can be extracted between images. Point correspondences are extracted automatically from the images by identifying 2D features and tracking those between the different perspective views. Different feature extraction algorithms exist (see [6] [15] [11]). Multiple camera calibrations can be solved globally in one step, or multiple 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 procedure to automatically calibrate the positions and poses of sensors and actuators without using calibration objects. Thus no special equipment is required. In addition the setup does not have to be synchronized. It only requires to filter out temporally unstable salient points and keep only stationary features. 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. only some cameras share a common view.
- The usage of an active display as our calibration target for intrinsic calibration giving us control over the calibration pattern to be displayed. As a result the extraction of feature points is easier and more reliable. The calibration pattern can be made adaptive to the distance between the camera and the pattern's image on the LCD screen.
- Control points and point correspondences across images are extracted automatically. Our solution works for cameras whose viewpoints differ less than 15°.

2 Problem Formulation

Given M cameras and N flat-panel displays, the goal is to determine the cameras' internal parameters and the 3D positions and poses of the cameras and the flat-panel displays automatically. Therefore we make the following assumptions: (1) We know the number of visual sensors and actuators in the network. (2) The displays are sufficiently flat. (3) The flat-panel displays' dimensions, i.e. their width and height are known in pixels.

In this work we use an enhanced perspective model to describe our cameras. The mapping performed by a perspective camera between a 3D point $\mathbf{X}$ and its 2D image point $\mathbf{x}$, both represented by their homogeneous coordinates, is usually represented by a 3×4 matrix, the camera projective matrix $\mathbf{P}$: $\mathbf{x} \simeq \mathbf{P}\mathbf{X}$. The matrix $\mathbf{P}$ can be written as $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$ where $\mathbf{K}$ is a 3×3 upper triangular matrix containing the camera intrinsic parameters:

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

The parameters f_x and f_y denote the focal length, p_x and p_y denote the coordinates of the principal point, each in terms of pixel dimensions. s denotes the skew. For most commercial cameras, and hence below, the skew is considered to be zero. The 3×3 rotation matrix $\mathbf{R}$ and the 3×1 translation vector $\mathbf{t}$ describe the 3D position and pose of the camera. As some desktop cameras exhibit significant distortions, this model has to be enriched by some distortion components. The distortion model introduced in [9] accounts for tangential and radial distortions using two coefficients. It describes distortions occuring in practice sufficiently precise. In the following discussion we assume that the distortion parameters of each camera are known and the effects of those have been removed from all images.

Different views of the same scene are related to each other. These relations can be used for our multiple camera calibration task. Therefore we need to determine a set of corresponding points across the different images. Points are said to correspond if they represent the same scene point in different views. This general calibration problem is illustrated in Fig. 2.

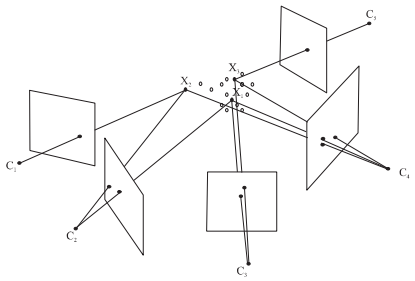

Figure 2. General calibration problem

A set of 3D points $\mathbf{X}_i$ is viewed by a set of cameras with matrices $\mathbf{P}^j$. Let $\mathbf{x}_i^j$ denote the coordinates of the i-th point as detected in the j-th camera image. A 3D point may not be visible in all cameras, thus its corresponding projected point will not be available in all images. The calibration problem is then to find the set of camera matrices $\mathbf{P}^j$ and points $\mathbf{X}_i$ such that for all image points $\mathbf{x}_i^j \simeq \mathbf{P}^j\mathbf{X}_i$ holds. However, unless additional constraints are given, it is in principle only possible to determine the camera matrices up to a projective ambiguity. Additional constraints arising from knowledge about the cameras' parameters and/or the scene can be used to restrict this ambiguity up to an affine, metric or Euclidean transformation.

Solution: We solve the camera and display calibration problem in two stages. In a first stage we determine the cameras' intrinsic parameters and their positions 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 the planar calibration object. Then camera positions and camera poses are computed in a common coordinate system (extrinsic calibration). 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 calibrations are 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 pattern. The pose of a specific display is calculated with respect to the camera in which it is best visible. 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 can be generally used to recover the calibration matrices of the cameras. Therefore establishing such correspondences is the first step in determining the cameras parameters. To establish point correspondences each image is at first represented by a set of features. Each feature describes a specific image point, and its neighborhood. Subsequently these features are input to a matching procedure, which identifies features in different images 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-features proposed in [11]. SIFT-based feature descriptors were identified in [12] 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. In order to compute a set of caracteristic image features, first a set of interest points - also called keypoints - is found by detecting scale-space extremas. Only keypoints that are stable under a certain amount of additive noise are preserved. 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. Finally a descriptor for each keypoint is calculated based upon image gradients in the local image. 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: The matching technique used for the SIFT-features has been proposed in [11]. Point correspondences between two images are established by comparing their respective keypoint descriptors. 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. The best matching candidate for a specific keypoint is identified by the keypoint belonging to the feature vector with the minimum distance. A match is found in the second image, if the distance ratio between the nearest and the second nearest neighbor (closest/second closest) is below a threshold. An example of matched points between two images is shown in Fig. 3.

Subpixel Accuracy: The result of SIFT-feature matching is only at pixel accuracy. For position es-

Figure 3. Matched points are visualized by a connecting line between images

timation of multiple cameras experiments have shown that it is essential to keep all computations at a subpixel accuracy level. So far the approximate positions of the corresponding points are known. To achieve subpixel precision we use the Affine Lucas Kanade feature tracker [15]. This tracker assumes an affine transformation between the viewpoint of both images. This approximation is valid, if the viewpoints of the different cameras are sufficiently close.
The basic optimization problem solved by the feature tracker is:

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

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 $d_{xx}, d_{xy}, d_{yx}, d_{yy}$:

$$\mathbf{D} = \begin{pmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{pmatrix} \tag{3}$$

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 the corresponding point to $\mathbf{u}$ 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

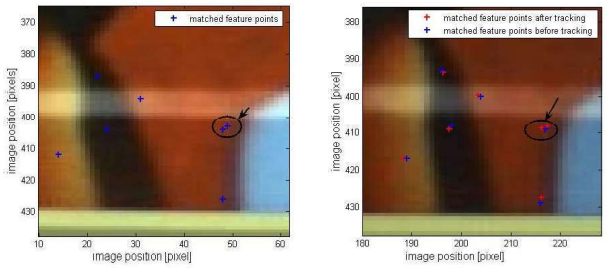

Figure 4. Matched feature points before and after the tracking algorithm (points in the left image were taken as reference and tracked in the right image)

corresponding points obtained by SIFT-feature matching were used to initialize the algorithm. They are shifted to a slightly different position by the tracker. The improvement 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 feature tracker regains the two different corresponding points and herewith significantly improves the accuracy of the image matching process.

4 Intrinsic Calibration

Intrinsic calibration is done independently for each camera using J.-Y. Bouguets Camera Calibration Toolbox [3] in OpenCV [2]. 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 needs not to be known. Therefore the target can be freely moved in front of each camera separately. As the 2D geometry of the calibration plane is known with 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. The Maximum Likelihood estimate of the camera parameters is obtained 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, i.e. by minimizing the following error:

$$\varepsilon = \sum_{j=1}^{n} \sum_{i=1}^{m} \parallel \mathbf{x}_i^j - \hat{\mathbf{x}}(\mathbf{K}, \kappa_1, \kappa_2, \rho_1, \rho_2, \mathbf{R}_j, \mathbf{t}_j, \mathbf{X}_i) \parallel^2 \tag{4}$$

where n denotes the number of images taken of the model plane and m denotes the number of corresponding points each images gives rise to. $\hat{\mathbf{x}}(\mathbf{K}, \kappa_1, \kappa_2, \rho_1, \rho_2, \mathbf{R}_j, \mathbf{t}_j, \mathbf{X}_i)$ is the theoretical position 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 is a non-linear optimization problem which requires a proper initialization. Thus the complete calibration procedure consists of an initialization stage, where a closed form solution for the calibration parameters is computed, followed by a nonlinear refinement based on the Maximum Likelihood criterion. For more details on both stages the reader is reffered to [18] and [3].

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. 5 (right; from [1]) as our planar model object, since it gives rise to a large number of SIFT-features. It is displayed on a laptop or any other screen of known size, whose surface is sufficiently flat. Different images of the model plane from different orientations are captured by waving the screen in front of the camera. Projected pattern points in the images are determined by matching extracted SIFT-features from each view with extracted features 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. 5.

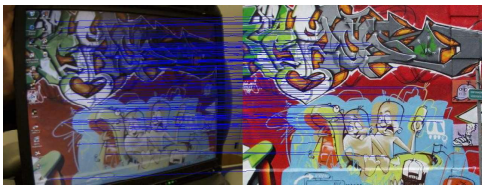

Figure 5. Matches between calibration pattern (right) and its imaged version (left)

Usually in other calibration procedures ([3], [17]) a checkerboard pattern is used as the calibration target requiring some user interaction to obtain matching points (in this case corners) between the

this object and its (different) image(s). The use of SIFT-feature matching in combination with a flat screen displaying a known pattern enables us to easyly and automaticaly detect the subset of image points. Also feature matching can be performed with images containing only a partially visible pattern.

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 Table 1. As the number of pattern points extracted varies per view, only the total number of points used in the calibration procedure is given 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 [18]. They 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 $n \geq 40$ 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 these. Compared to algorithms using corner features, the number of views necessary for reliable calibration is large, as corner correspondences can be extracted more accurately than SIFT feature correspondences. Once the intrinisc parameters are determined the distortion in the original images can be corrected as shown in Fig. 6.

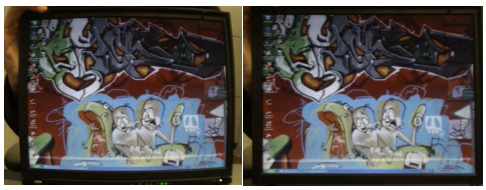

Figure 6. Original and recified image

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 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. The mapping of a 3D point $\mathbf{X}_i$ to a 2D image point $\mathbf{x}_i^j$ can be described according to $\mathbf{x} \simeq \mathbf{PX}$ (assuming distortion effects have been removed). As a 3D point $\mathbf{X}_i$ might only be observed by

a subset of cameras, the corresponding projected point will not appear in every view. 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 measured point $\mathbf{x}$ are derived by removing the effects of the internal parameters from the measured image point: $\mathbf{x_n} \simeq \mathbf{K}^{-1}\mathbf{x}$ where $\mathbf{K}$ is the calibration matrix of the particular camera. 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 $\mathbf{P_n}^j$: $\mathbf{x_n}_i^j \simeq \mathbf{P_n}^j\mathbf{X}_i$ where $\mathbf{P_n}^j = \mathbf{K}_j^{-1}\mathbf{P}^j\mathbf{K}_j^{-1}\mathbf{K}_j[\mathbf{R}_j|\mathbf{t}_j][\mathbf{R}_j|\mathbf{t}_j]$. The matrix $\mathbf{P_n}^j$ only consists of a rotation $\mathbf{R}_j$ and a translation $\mathbf{t}_j$, which define the position of the specific camera. Given a set of image correspondences, represented by their normalized cooridnates $\mathbf{x_n}_i^j$ our goal is now to find the appropriate set of normalized camera matrices $\mathbf{P_n}^j$ and points $\mathbf{X}_i$ such that $\mathbf{x_n}_i^j \simeq \mathbf{P_n}^j\mathbf{X}_i$. As the intrinsic parameters of each camera are known, the relative position of the cameras is computed uniquely up to an overall scale factor, i.e. the position of the cameras is determined up to a metric transformation.

5.1 Algorithm

There are several strategies for solving this 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, which is defined as the - summed squared - distance between the theoretical image positions of the projections of the estimated 3D points $\mathbf{X}_i$ 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 3D 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. 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 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. 7.

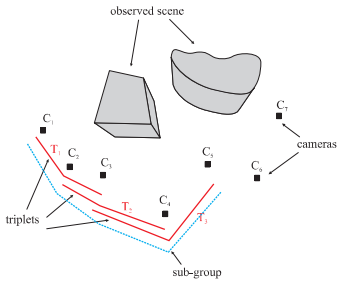

Figure 7. Camera positions and the structure of the scene are computed by registering the basic building block (triplets of cameras) hierarchically

The first step in our algorithm 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 since camera triplets are required to share a common view as a reliable trifocal tensor estimation demandes a sufficient number of point correspondences over the three images. Thus only triplets with a sufficient number of corresponding points are kept. Next the triplets need to be clustered into subgroups. Triplets in a certain subgroup and also different subgroups need to share two cameras to enable their registration in a common coordinate frame. Therefore combinations of the above determined triplets are tested. In a second stage the cameras in each triplet are 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]. To establish point correspondences, one image 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-feature 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 can be simply eliminated by removing all keypoints whose positions are temporally unstable. However, the observed 3D scene needs to be sufficiently textured in order to ascertain detection and tracking of enough point features from the acquired images to ensure reliable results in the trifocal tensor estimation.

Registering all triplets into the same coordinate frame is done 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 objective is to obtain a common set of 3D points and a normalized camera matrix for each view, such that the reprojection error is minimized. In the following only the registration of two triplets that share exactly two cameras is discussed. All registration problems in the algorithm are analogously solved. In general different overlaps are possible, but as our implementation specifically forces the triplets in a subgroup and the different subgroups, to share two cameras, only this case is discussed here. For a detailed description and evaluation of other registration methods the reader is referred to [5]. Given 3D points common in the sets of two different triplets and the homogeneous representation of these points by $\mathbf{X}_i$ in the frame of the first triplet and $\mathbf{X'}_i$ in the second frame (their inhomogeneous representation is denoted with $\bar{\mathbf{X}}_i$, $\bar{\mathbf{X}}'_i$), their representations in the different metric frames are related by a 3-space homography $\mathbf{H}$ according to $\mathbf{X}_i = \mathbf{H}\mathbf{X}'_i$. Equivalently $\mathbf{P_n}^j = \mathbf{P'_n}^j\mathbf{H}^{-1}$ holds for the corresponding normalized camera matrices of both triplets. The homography between two metric frames can be described by a metric transformation:

$$\mathbf{H_s} = \begin{pmatrix} \sigma \cdot \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix} \tag{5}$$

where $\mathbf{R}$ denotes a 3×3 rotation matrix and $\mathbf{t}$ a 3×1 translation vector. σ identifies the relative scale between the structures. Since $\mathbf{R}$ can be parametrized by a 3-vector, 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 further refined in a nonlinear stage. In order to compute a direct solution, the first step is to estimate the relative scale σ via the quotient of the mean distances of the 3D points $\bar{\mathbf{X}}_i$, $\bar{\mathbf{X}}'_i$ to their respective centroid (denoted with inhomogeneous coordinates $\bar{\mathbf{M}}$, $\bar{\mathbf{M}}'$):

$$\sigma \frac{\frac{1}{n}\sum_{i=1}^{n} \parallel \bar{\mathbf{X}}_i - \bar{\mathbf{M}} \parallel}{\frac{1}{n}\sum_{i=1}^{n} \parallel \bar{\mathbf{X}}'_i - \bar{\mathbf{M}}' \parallel} \tag{6}$$

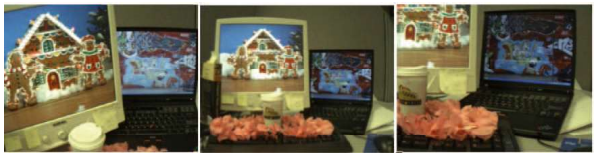

Figure 8. Example images of an office scene from different viewpoints (cameras)

where $\|\cdot\|$ denotes the L2-norm and n is the number of common points. Now the second structure is rescaled according to $\bar{\mathbf{X}}'_{\mathbf{s}i} = \sigma\bar{\mathbf{X}}'_i$ so that $\mathbf{X}_i = \mathbf{H}\mathbf{X}'_i$ becomes $\mathbf{X}_i = \mathbf{H}_\mathbf{s}\mathbf{X}'_{\mathbf{s}i}$ where

$$\mathbf{H}_\mathbf{s} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix} \tag{7}$$

In order to obtain an initial estimate for the coefficients of $\mathbf{R}$ and $\mathbf{t}$ the squared distance between the two structures consisting of points $\mathbf{X}_i$ and $\mathbf{X}'_{\mathbf{s}i}$, is minimized with respect to the coefficients of $\mathbf{H}_\mathbf{s}$ using linear algebraic methods:

$$\min_{\mathbf{R},\mathbf{t}} \sum_i d^2(\mathbf{X}_i, \mathbf{H}_\mathbf{s}\mathbf{X}'_{\mathbf{s}i}) \tag{8}$$

where $d(\mathbf{x},\mathbf{y})$ denotes the Euclidean distance between the inhomogeneous points corresponding to $\mathbf{x}$ and $\mathbf{y}$.
Finally the derived initial values are refined in a nonlinear minimization stage where the reprojection error to the originally measured and normalized image points is minimized with respect to all parameters of $\mathbf{H}$:

$$\min_{\sigma,\mathbf{R},\mathbf{t}} \sum_{ij} d^2(\mathbf{P}_\mathbf{n}{}^j\mathbf{H}\mathbf{X}'_i, \mathbf{x}_\mathbf{n}{}_i^j) + d^2(\mathbf{P}'_\mathbf{n}{}^j\mathbf{H}^{-1}\mathbf{X}_i, \mathbf{x}_\mathbf{n}{}_i^j) \tag{9}$$

This non-linear minimization is solved using the Levenberg-Marquardt algorithm.
By registering all triplets hierarchically in one common coordinate frame as described above 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 of the entire set of camera positions and the 3D-structure is computed via bundle adjustment:

$$\min_{\mathbf{P}_\mathbf{n}{}^j,\mathbf{X}_i} \sum_{ij} d^2(\mathbf{P}_\mathbf{n}^\mathbf{j}\mathbf{X}_i, \mathbf{x}_\mathbf{n}{}_i^j) \tag{10}$$

Each normalized camera matrix is parameterized by 6 entries, 3 representing the rotation matrix and 3 representing the translation vector. The dimension of the minimization problem adds then up to a total number of $6(N-1)$ 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 sub-groups was chosen to five cameras. Fig. 8 shows some of the images taken from the different viewpoints of the cameras in one experiment. 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 of corresponding points.

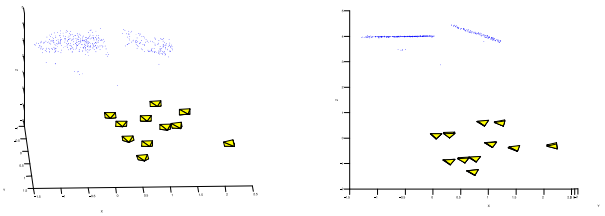

Figure 9. Two different views of the reconstructed 3D scene points and camera positions for the entire group of 11 cameras

The resulting camera positions and scene reconstruction is shown in Fig. 9. Camera positions are marked with yellow pyramids, reconstructed scene points with blue dots. In Fig. 10 the final reprojection error is illustrated for one estimated camera. The distance between the reprojected points and the measured image points is very small. Therefore the overall estimation is highly accurate.

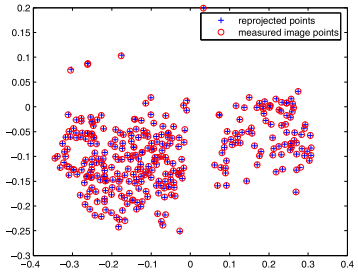

Figure 10. Final reprojection error illustrated in one camera image

Discussion: In the given example the implementation performs very well. However experiments with different data sets have shown that sporadically the accuracy of the algorithm can be severly affected. Thorough analysis showed that mis-estimations were caused by inaccurately estimated triplets. If the camera positions and/or the 3D points in one triplet are estimated inaccurately, the homography estimation to register this triplet in a subgroup fails as well. As a result the whole subgroup 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 non-linear optimization to fail completely, i.e. to converge to a suboptimal solution or to not converge at all.
One sources of failure in the triplet estimation may consist in corresponding image points that are not extracted sufficiently accurate, 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. Another cause of failure arises from the fact that the intrinsic camera parameters can only be estimated with a certain accuracy. This may also have an impact on the noise level in the corresponding normalized points.

6 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. All stages of the calibration algorithms have been implemented 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.

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] 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.

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

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

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

[14] 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.

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

[16] 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.

[17] 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.

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