Using Spatially Distributed Patterns for Multiple View Camera Calibration

Martin Grochulla, Thorsten Thormählen, and Hans-Peter Seidel

Max-Planck-Institut Informatik, Saarbrücken, Germany
{mgrochul, thormae}@mpi-inf.mpg.de

Abstract. This paper presents an approach to intrinsic and extrinsic camera parameter calibration from a series of photographs or from video. For the reliable and accurate estimation of camera parameters it is common to use specially designed calibration patterns. However, using a single pattern, a globally consistent calibration is only possible from positions and viewing directions from where this single pattern is visible. To overcome this problem, the presented approach uses multiple coded patterns that can be distributed over a large area. A connection graph representing visible patterns in multiple views is generated, which is used to estimate globally consistent camera parameters for the complete scene. The approach is evaluated on synthetic and real-world ground truth examples. Furthermore, the approach is applied to calibrate the stereo-cameras of a robotic head on a moving platform.[1]

1 Introduction

Camera parameter estimation is the task of finding the intrinsic and extrinsic camera parameters, which describe the projection of the 3D scene onto the 2D image plane of the camera. From one calibrated camera the line of sight for a given pixel can be computed; in combination with stereo algorithms two calibrated cameras can be used in combination with stereo algorithms to estimate the depth for a given pixel [2]. Furthermore, camera calibration is required for a number of computer vision applications in areas such as augmented reality, robot navigation, or special effects generation.

A common method for camera calibration is the usage of a calibration object or calibration pattern for which the geometry is known. The knowledge of the geometry of the pattern provides points in 3D space, while the corresponding 2D points are extracted from the image. With these extracted 2D-3D correspondences the camera parameters can be estimated. Popular approaches for camera calibration were presented by Tsai [3, 4] and Zhang [5, 6], both using calibration patterns.

To compute the intrinsic parameters of the camera Zhang uses at least two images of the pattern from different orientations. Tsai on the other hand uses only a single image of a calibration pattern to estimate the extrinsic and intrinsic camera parameters in a two stage approach. In both approaches the calibration pattern consists of squares arranged in a grid. The corners of the squares are the 3D points used for calibration. Thus, four 3D points for each square are obtained. However, finding the correct 2D position of the corner points in the image can be difficult and error-prone considering the noise and blur present in the images.

Tsai and Zhang both use a single pattern for camera parameter estimation. However, if multiple cameras or if cameras in a larger environment have to be calibrated, the problem arises that this may not be possible with a single pattern, since only cameras that see the pattern can be calibrated.

If the task is to calibrate multiple cameras in a scene, one possibility is to use the approach of Ueshiba and Tomita [7]. This approach uses a single calibration pattern, which is placed at three or more locations, where a separate set of images is taken for each location. Another possibility is to use multi-camera self-calibration [8]. In this approach, instead of calibration patterns, a single laser pointer is moved in the calibration volume. Tracking the position of the laser pointer in each image of each camera allows to self-calibrate the cameras. However, both approaches require static cameras and, thus, can not handle multiple images of a single moving camera.

If the task is to calibrate a moving camera in the scene, self-calibration can be employed. This approach does not require a special calibration object. Intrinsic camera parameters are computed from multiple uncalibrated images taken by the camera. The movement of the camera provides enough constraints for computing the intrinsic parameters [9, 10]. However, camera self-calibration is a complex and difficult task, where degenerate cases can occur.

For the application of augmented reality, Fiala et al. [11, 12] developed a system called ARTag that employs multiple coded markers to calibrate the camera. This system consists of a set of different markers and algorithms to detect the orientation and the position of the markers in the image. The goal is to augment the image/video with rendered 3D virtual content by detecting the relative position and orientation of several markers to each other.

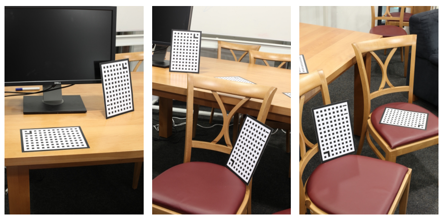


Fig. 1: The suggested approach allows camera calibration from images that have a partial overlap. Each one of the shown images contains two calibration patterns where one of these patterns is also visible in the next image.

In this paper, we present an approach for performing camera calibration from a series of images or from a video with multiple patterns. We neither restrict our approach to require a pattern to be visible in all views nor a camera to see all patterns. In contrast to existing work, the approach is very general and works with one or multiple moving or static cameras.

The approach is easy to apply in practice, as a user only has to distribute the calibration patterns in the scene such that in each view some of them are visible (see Fig. 1 for an example). The patterns are coded so that they can be identified in different images. It is not necessary for all patterns to be visible in all views; a few patterns per view are sufficient as long as relative position and orientation between every pattern or camera can be computed. Instead of using the corners of the squares on the pattern as 2D points, we use the centers of gravity of the projected squares. This has the advantage of a more reliable detection. However, the center of gravity does not always coincide with the geometric center of the squares under perspective projection. Therefore, these 2D points must be refined after an initial parameter estimation of cameras and patterns, in order to achieve a camera calibration with high accuracy.

2 Camera Calibration

This chapter describes our approach to estimate the intrinsic and extrinsic parameters of multiple cameras (either static or moving) using multiple coded patterns.

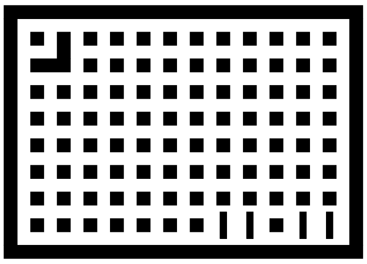


Fig. 2: Calibration pattern used in our approach. The L-shaped marker is used to detect the orientation of the pattern, while in the last row a pattern identifier is encoded binary (here pattern no. $27 = 11011_2$ is shown).

Tsai and Zhang are using the corners of the squares on the calibration patterns as points. Finding those corner points becomes less accurate with smaller size of the pattern in the image. Hence, we use the centers of gravity as initial 2D points, as they are easier to detect. In order to extract enough 2D-3D correspondences for camera parameter estimation, the patterns used in our approach consist of eight rows of squares with twelve squares in a row, arranged in a grid (see Fig. 2). To detect the orientation of the pattern, we use an L-shaped marker in one corner of the pattern, which replaces three squares. To be able to distinguish different patterns, we use an identifier for each pattern. The identifier is coded in the last row of the pattern. It is a binary coded number with squares representing 0s and rectangles representing 1s. The rectangles are twice as long and half as thick as the squares resulting in the same surface area as the squares. The grid of squares, including the coded identifier and the marker, are surrounded by a frame. Using such a calibration pattern, provides enough 2D-3D correspondences for the estimation process.

2.1 Pattern Identification and Point Extraction

To identify the patterns and extract 2D points, we proceed as follows. First, we threshold an image with a specified threshold t resulting in a binary image. Color images are converted to gray-scale before thresholding. In the binary image we perform a region analysis, where a single region consists of all pixels having the same value (black or white) and being 4-connected. Patterns are then identified in the image as regions having a given number of neighbor regions The L-shaped marker is then identified as the second biggest neighbor region within the pattern. Knowing the orientation of the pattern, we are able to identify the last row of the pattern. In order to distinguish squares from rectangles, which encode the pattern identifier, we compute the standard deviation of all 2D pixel positions in the region. In practice, this is already enough to distinguish squares from rectangles since the standard deviation for the rectangles will be significantly bigger than the one of the squares. Thereby, the expected standard deviation of the squares is known from looking at the second last row of the pattern, which only contains squares.

Finally, the initial 2D points for the parameter estimation are computed as the mean of the pixel positions of each region. Note that the mean of the pixel positions of a region corresponds to the center of gravity of the projected square. Although the center of gravity is the same as the geometric center for a square or a rectangle in 2D, this does not hold for projected squares or rectangles in 3D space. However, the center of gravity is a good approximation of the geometric center and our algorithm does work well with these measurements. Nevertheless, once camera parameters are estimated, the measured 2D points can be compensated with the current camera parameters in order to refine the camera parameters.

2.2 Connection Graph Generation

By distinguishing different patterns in the images, we can generate a connection graph. This graph is an abstract representation of the connections between camera views and visible patterns. In the graph cameras and patterns are represented by nodes (see Fig. 3). A camera node is connected to a pattern node by an edge, if the pattern represented by its node is visible in the view of the camera represented by its node. The edge means position and orientation of a camera with respect to a pattern can be estimated.

Our approach uses the following two ideas. On the one hand, if two patterns are visible in one image, the position and orientation between those patterns can be estimated (see section 2.3: single view alignment). On the other hand, if one pattern is visible in two different views, the position and orientation between the two cameras can be estimated (see section 2.3: multiple view alignment).

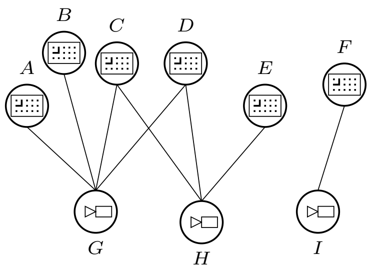


Fig. 3: Connection graph. Edges between camera nodes (G, H, and I) and pattern nodes (A – F) represent the ability to estimate camera parameters. A path from node A to node E indicates the possibility to compute position and orientation of these two patterns relative to each other, while this is not possible for nodes E and F.

For the example shown in Fig. 3, this means relative positions and orientations between these nodes can be estimated, as there is an edge between each pattern node A, B, C, and D and the camera node G. By looking at the edges of the transitive closure of the connection graph, it is possible to determine if it is also possible to estimate the relative transformation between two camera nodes in the graph. For the example shown in Fig. 3 it is possible to relate camera nodes G and H, but not H and I.

2.3 Camera Parameter Estimation

Having generated the connection graph, we will now show how positions and orientations of cameras and patterns in the scene are estimated. For simplicity, in the following we will describe the problems as if the scene is observed by multiple static cameras. However, a single moving camera or multiple moving cameras can be handled in the same way by just generating a new virtual static camera for each point in time.

The estimation of the camera parameters is done in six steps (compare Fig. 4):

1. Estimation of position and orientation between a single pattern and the camera using Tsai's approach,
2. Alignment of all patterns visible in one image,
3. Estimation of positions and orientations between *all* patterns in an image and the camera,
4. Alignment of all cameras and all patterns,
5. Estimation of positions and orientations between *all* patterns and *all* cameras, and
6. Refinement of 2D points and re-estimation of camera parameters (optional)

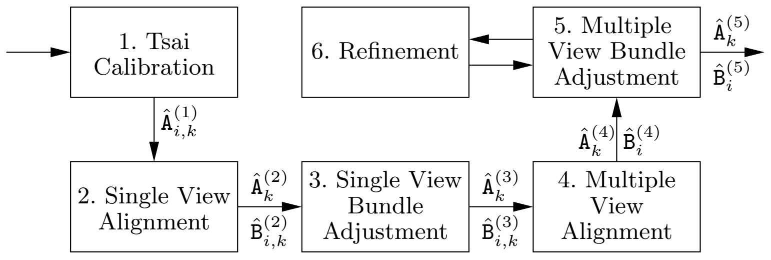

1. Tsai Calibration
6. Refinement
5. Multiple View Bundle Adjustment
$\hat{\mathtt{A}}_k^{(5)}$
$\hat{\mathtt{B}}_i^{(5)}$
$\hat{\mathtt{A}}_{i,k}^{(1)}$
$\hat{\mathtt{A}}_k^{(4)}$
$\hat{\mathtt{B}}_i^{(4)}$
2. Single View Alignment
$\hat{\mathtt{A}}_k^{(2)}$
$\hat{\mathtt{B}}_{i,k}^{(2)}$
3. Single View Bundle Adjustment
$\hat{\mathtt{A}}_k^{(3)}$
$\hat{\mathtt{B}}_{i,k}^{(3)}$
4. Multiple View Alignment

Fig. 4: Algorithm overview.

In our approach, camera view k is represented by its projection matrix $\mathtt{A}_k$:

$$\mathtt{A}_k = \begin{bmatrix} f_k & 0 & p_{x,k} \\ 0 & f_k & p_{y,k} \\ 0 & 0 & 1 \end{bmatrix} [\mathtt{I}\,|\,\mathbf{0}] \begin{bmatrix} \mathtt{R}_k & -\mathtt{R}_k \underset{\sim}{\mathbf{C}}_k \\ \mathbf{0}^\top & 1 \end{bmatrix} = \mathtt{K}_k [\mathtt{I}\,|\,\mathbf{0}] \begin{bmatrix} \mathtt{R}_k & -\mathtt{R}_k \underset{\sim}{\mathbf{C}}_k \\ \mathbf{0}^\top & 1 \end{bmatrix}, \quad (1)$$

with the 3×3 calibration matrix $\mathtt{K}_k$, the 3×3 rotation matrix $\mathtt{R}_k$, and the 3-vector $\underset{\sim}{\mathbf{C}}_k$. The calibration matrix $\mathtt{K}_k$ contains the intrinsic camera parameters, where f_k is the focal length and $p_{x,k}$ and $p_{y,k}$ are the principal point offsets in x- and y-direction, respectively. The rotation matrix is composed of consecutive rotations around the y-, x- and z-axis with Euler angles φ, ϑ, and ρ: $\mathtt{R} = \mathtt{R}_z(\rho) \cdot \mathtt{R}_x(\vartheta) \cdot \mathtt{R}_y(\varphi)$. The camera center is represented by $\underline{\mathbf{C}}_k$. A pattern i is represented by a 4×4 transformation matrix $\mathtt{B}_i = \left[\begin{smallmatrix} \mathtt{S}_i & -\mathtt{S}_i \underline{\mathbf{D}}_i \\ \mathbf{0}^\top & 1 \end{smallmatrix} \right]$, with 3-vector $\underline{\mathbf{D}}_i$ representing the pattern center and 3×3 rotation matrix $\mathtt{S}_i$ composed of consecutive rotations around the y-, x- and z-axis with Euler angles α, β, and γ. The projection of a 3D point $\mathbf{P}$ of pattern i given in homogeneous coordinates in the pattern coordinate system is then given by

$$\mathbf{p} = \mathtt{K}_k \left[\mathtt{I} \,|\, \mathbf{0} \right] \begin{bmatrix} \mathtt{R}_k & -\mathtt{R}_k \underline{\mathbf{C}}_k \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathtt{S}_i & -\mathtt{S}_i \underline{\mathbf{D}}_i \\ \mathbf{0}^\top & 1 \end{bmatrix} \mathbf{P}, \tag{2}$$

where $\mathbf{p}$ is the corresponding 2D point in the image plane of camera k given in homogeneous coordinates.

Radial distortion is modeled as follows. Let $(x_u, y_u)^\top$ be the undistorted position of the projection of the 3D point $\underline{\mathbf{P}}$. The distorted position of the projection of $\underline{\mathbf{P}}$ is then modeled as $x_d = \left(1 + \kappa_3 r_u^2 + \kappa_5 r_u^4\right) x_u$ and $y_d = \left(1 + \kappa_3 r_u^2 + \kappa_5 r_u^4\right) y_u$, where r_u is the distance of $(x_u, y_u)^\top$ from the principal point $(p_x, p_y)^\top$. Here, $(1 + \kappa_3 r_u^2 + \kappa_5 r_u^4)$ is an approximation of the real radial distortion function with a Taylor series and κ_3 and κ_5 are the parameters describing the lens distortion.

For the sake of clarity, we are denoting the resulting cameras matrices $\mathtt{A}_k$ and pattern transformations $\mathtt{B}_i$ of the m-th processing step with an additional index: $\mathtt{A}_k^{(m)}$ and $\mathtt{B}_i^{(m)}$ (compare Fig. 4).

Tsai Calibration Having identified all patterns i in a camera view k, we use Tsai's approach [4] to estimate the position and orientation of the camera $\hat{\mathtt{A}}_{i,k}^{(1)}$ relative to the pattern i. To be able to estimate the parameters, we have to provide 2D-3D correspondences between the image and the pattern. The (measured) 2D points $\tilde{\mathbf{p}}_i$ of pattern i are given by the 2D-3D point extraction (see section 2.1). Here we use the centers of gravity of the found regions. The corresponding 3D points $\mathbf{P}_i$ in the pattern coordinate system are given by the known structure of the pattern. Since our calibration pattern is planar, we assume for the sake of simplicity and without loss of generality that all 3D points lie in the x-y-plane and that the geometric center of the whole pattern lies at the origin. During this first processing step we define that the local coordinate system of the pattern coincides with the world coordinate system. Thus, we have: $\hat{\mathtt{B}}_{i,k}^{(1)} = \mathtt{I} \quad \forall\, k$.

Since we assume that the pattern lies in the x-y-plane around the origin, Tsai's algorithm provides estimated parameters $\hat{\mathtt{A}}_{i,k}^{(1)}$ for the location and orientation of camera k with respect to pattern i by minimizing the cost function:

$$\underset{\hat{\mathtt{A}}_{i,k}^{(1)}}{\operatorname{argmin}} \sum_{i,j,k} d\left(\tilde{\mathbf{p}}_{j,k},\, \hat{\mathtt{A}}_{i,k}^{(1)} \mathtt{B}_{i,k}^{(1)} \mathbf{P}_j\right)^2 \qquad \forall\, i,\, k, \tag{3}$$

where $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between homogeneous points $\mathbf{x}$ and $\mathbf{y}$.

Single View Alignment Having an estimate $\hat{\mathtt{A}}^{(1)}_{i,k}$ of the position and orientation of the camera k with respect to every visible pattern i in the image, we are now able to compute parameters $\hat{\mathtt{B}}^{(2)}_{i,k}$ for all patterns in the image. Since there is only one camera view for each image in reality, the different estimated camera parameters $\hat{\mathtt{A}}^{(1)}_{i,k}$ are in fact resulting from different positions of the patterns in the scene. Therefore, we align the different estimated cameras to a single reference camera with $\mathtt{R} = \mathtt{I}, \underset{\sim}{\mathbf{C}} = \mathbf{0}^\top$ for every camera view k. From the different camera parameters

$$\hat{\mathtt{A}}^{(1)}_{i,k} = \hat{\mathtt{K}}^{(1)}_{i,k} \left[\, \mathtt{I} \,|\, \mathbf{0} \,\right] \begin{bmatrix} \hat{\mathtt{R}}^{(1)}_{i,k} & -\hat{\mathtt{R}}^{(1)}_{i,k}\, \hat{\underset{\sim}{\mathbf{C}}}^{(1)}_{i,k} \\ \mathbf{0}^\top & 1 \end{bmatrix} \tag{4}$$

we now compute estimates of the positions of the patterns $\hat{\mathtt{B}}^{(2)}_{i,k}$ in 3D space relative to a reference camera $\hat{\mathtt{A}}^{(2)}_{k}$ for all patterns i in all views k. This is done by setting

$$\hat{\mathtt{B}}^{(2)}_{i,k} := \begin{bmatrix} \hat{\mathtt{R}}^{(1)}_{i,k} & -\hat{\mathtt{R}}^{(1)}_{i,k}\, \hat{\underset{\sim}{\mathbf{C}}}^{(1)}_{i,k} \\ \mathbf{0}^\top & 1 \end{bmatrix} \quad \forall\, i,\, k \quad \text{and} \quad \hat{\mathtt{A}}^{(2)}_{k} := \frac{1}{n_k} \sum_i \hat{\mathtt{K}}^{(1)}_{i,k} \left[\, \mathtt{I} \,|\, \mathbf{0} \,\right] \quad \forall\, k, \tag{5}$$

with n_k denoting the number of patterns visible in view k. Note that we simply average the intrinsic parameters $\hat{\mathtt{K}}^{(1)}_{i,k}$ from Tsai's estimations to get an estimate of the intrinsic parameters of the reference camera $\hat{\mathtt{A}}^{(2)}_{k}$.

For the single view alignment, we have $\hat{\mathtt{A}}^{(1)}_{i,k} \mathtt{B}^{(1)}_{i,k} \mathbf{P} = \mathbf{p}^{(1)} \approx \mathbf{p}^{(2)} = \hat{\mathtt{A}}^{(2)}_{k} \hat{\mathtt{B}}^{(2)}_{i,k} \mathbf{P}$, as can be verified by Eq. (2). Here, $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ are only approximately equal due to the parameter averaging in Eq. (5).

Single View Bundle Adjustment After the single view alignment we perform a bundle adjustment for every camera view k. The bundle adjustment minimizes the reprojection error. The reprojection error is the sum of distances between the measured 2D points $\tilde{\mathbf{p}}_{j,k}$ in the image plane and the projections of corresponding estimated 3D world points $\hat{\mathtt{B}}_{i,k}\mathbf{P}_j$ (both represented in homogeneous coordinates). Bundle adjustment uses non-linear Levenberg-Marquardt optimization ($\hat{\mathtt{A}}^{(2)}_{k}$ and $\hat{\mathtt{B}}^{(2)}_{i,k}$ are used for the initialization of $\hat{\mathtt{A}}^{(3)}_{k}$ and $\hat{\mathtt{B}}^{(3)}_{i,k}$):

$$\underset{\hat{\mathtt{A}}^{(3)}_{k},\, \hat{\mathtt{B}}^{(3)}_{i,k}}{\operatorname{argmin}} \sum_{i,j} d\big(\tilde{\mathbf{p}}_{j,k},\, \hat{\mathtt{A}}^{(3)}_{k} \hat{\mathtt{B}}^{(3)}_{i,k} \mathbf{P}_j\big)^2 \qquad \forall\, k \quad . \tag{6}$$

Multiple View Alignment After having estimated camera parameters $\hat{\mathtt{A}}^{(3)}_{k}$ and pattern parameters $\hat{\mathtt{B}}^{(3)}_{i,k}$ for every single image k with the single view bundle adjustment, we now estimate globally consistent camera and pattern parameters $\hat{\mathtt{A}}^{(4)}_{k}$ and $\hat{\mathtt{B}}^{(4)}_{i}$, respectively. Using the generated connection graph, we are able to select two images k and k' that have at least one pattern i in common.

Consistent parameters are computed for these two images by fixating the camera and pattern parameters of one image ($\hat{\mathtt{A}}_k^{(4)} := \hat{\mathtt{A}}_k^{(3)}, \hat{\mathtt{B}}_i^{(4)} := \hat{\mathtt{B}}_{i,k}^{(3)}$) and transform the camera and pattern parameters of the other image in the following way: $\mathtt{T}_{k,k'} := \hat{\mathtt{B}}_{i,k}^{(3)} \big(\hat{\mathtt{B}}_{i,k'}^{(3)}\big)^{-1}$ is a transformation to align views k and k' where pattern i is the link between those views. The equation $\hat{\mathtt{A}}_{k'}^{(4)} := \hat{\mathtt{A}}_{k'}^{(3)} \mathtt{T}_{k,k'}^{-1}$ transforms the camera of view k' and the equation $\hat{\mathtt{B}}_i^{(4)} := \mathtt{T}_{k,k'} \hat{\mathtt{B}}_{i,k'}^{(3)}$ aligns all patterns of view k'. However, we only transform pattern that have not already been aligned before. Using the connection graph we align all views by consecutively aligning one unaligned view with all views that have already been processed. In addition, using the connection graph we are able to detect constellations where the views cannot be aligned.

Multiple View Bundle Adjustment If all cameras and all patterns have been aligned, we perform another bundle adjustment to minimize the reprojection error of the patterns $\hat{\mathtt{B}}_i^{(5)}$ in all camera views $\hat{\mathtt{A}}_k^{(5)}$ (similarly, $\hat{\mathtt{A}}_k^{(4)}$ and $\hat{\mathtt{B}}_i^{(4)}$ are used to initialize of $\hat{\mathtt{A}}_k^{(5)}$ and $\hat{\mathtt{B}}_i^{(5)}$):

$$\underset{\hat{\mathtt{A}}_k^{(5)},\ \hat{\mathtt{B}}_i^{(5)}}{\operatorname{argmin}} \sum_{i,j,k} d\big(\tilde{\mathbf{p}}_{j,k}, \hat{\mathtt{A}}_k^{(5)} \hat{\mathtt{B}}_i^{(5)} \mathbf{P}_j\big)^2 \quad . \tag{7}$$

Refinement Since the center of gravity of the projected square does not back-project to the geometric center of the square, 2D points are optionally refined.

Having computed globally consistent camera and pattern parameters $\hat{\mathtt{A}}_k^{(5)}$ and $\hat{\mathtt{B}}_i^{(5)}$ and knowing the size of the squares (or rectangles), we are able to project the corners of the squares into the image plane. For each square we get four points forming a quadrilateral. From these four corner points we can then compute the projection of the geometric center $\hat{\mathbf{c}}_{\mathrm{geo}}$ and the projection of the center of gravity $\hat{\mathbf{c}}_{\mathrm{grav}}$ of the square.

The projection of the geometric center is the projection of the intersection of the diagonals of that quadrilateral. For the projection of the center of gravity of the square we first compute the centers of gravity of all four possible triangles of the quadrilateral. The center of gravity of a triangle is the arithmetic mean of its corners. The four computed centers of gravity form another quadrilateral. The projection of the center of gravity of the original quadrilateral is then the intersection of the diagonals of the second quadrilateral.

We then update the initial 2D points $\tilde{\mathbf{p}}_{j,\,\mathrm{init}}$ with

$$\tilde{\mathbf{p}}_{j,\,\mathrm{new}} := \tilde{\mathbf{p}}_{j,\,\mathrm{init}} - \hat{\mathbf{c}}_{\mathrm{grav}} + \hat{\mathbf{c}}_{\mathrm{geo}} \tag{8}$$

and repeat the multiple view bundle adjustment of section 2.3 with the refined 2D points. If necessary, the refinement of the 2D points together with the multiple view bundle adjustment can be iterated. However, we observed the largest improvement to usually occur after the first iteration.

Δf	$\Delta \mathbf{C}$	RMSE of $\{\varphi, \vartheta, \rho\}$	Δf	$\Delta \mathbf{C}$	RMSE of $\{\varphi, \vartheta, \rho\}$
[mm]	[mm]	[rad]	[mm]	[mm]	[rad]
0.00861	0.07769	1.8215E−04	0.00166	0.01962	1.7127E−04
0.06058	0.26194	9.3651E−04	0.01492	0.05603	7.9734E−04
0.01667	0.10513	7.0942E−04	0.00311	0.02640	6.3498E−04
0.00287	0.03357	2.5241E−04	0.00106	0.01355	2.2049E−04
0.00593	0.11377	1.1414E−04	0.00872	0.05367	1.1054E−04
0.00130	0.13067	1.1775E−04	0.00238	0.03902	1.2050E−04
0.00637	0.06130	9.9409E−05	0.00026	0.01868	1.0056E−04
0.00428	0.07432	8.3503E−04	0.00161	0.03465	6.7190E−04
0.05864	0.11363	1.1946E−03	0.06134	0.07239	1.1350E−03
RMSE for whole series of images			RMSE for whole series of images		
0.02936	0.12170	6.3581E−04	0.02852	0.10208	5.6700E−04

Table 1: The synthetic image sequence. **Left**: Comparison between estimated camera parameters and ground truth without 2D point refinement. **Right**: Comparison between estimated camera parameters and ground truth with one iteration of 2D point refinement.

3 Results

This section presents results of tests performed to evaluate our approach. In a first test, we used synthetic data to be able to compare the estimated camera parameters with the ground truth. In a second test, we took images from several calibration patterns located on scale paper to compare estimated pattern positions with measured pattern positions. Finally, we applied our method to calibrate two cameras of a robotic head which can perform human-like movements.

3.1 Synthetic Ground Truth Example

We rendered a series of 9 images of a scene in which we placed six calibration patterns (see Fig. 5). Generating a synthetic series of images enabled us to compare our estimation results with the ground truth (see Tab. 1). The rendered virtual room had a size of approximately 16 square meters, which is important to put the accuracy of results in Tab. 1 into relation. Compared to the overall extend of the scene, the observed errors can be regarded as very low.

We performed camera parameter estimation using our approach. In the first run we estimated camera and pattern parameters *without* refining the 2D points obtained from Sec. 2.1. In a second run we then used the 2D point refinement from Sec. 2.3. Although we found the estimation results using the initial 2D points to be good, they could be improved by the 2D point refinement (an improvement of approx. 3%, 16%, and 11% percent for focal length, camera center and camera rotation, respectively). As expected, the first iteration of the refinement resulted in the biggest improvements, while further iterations did not improve the results significantly.

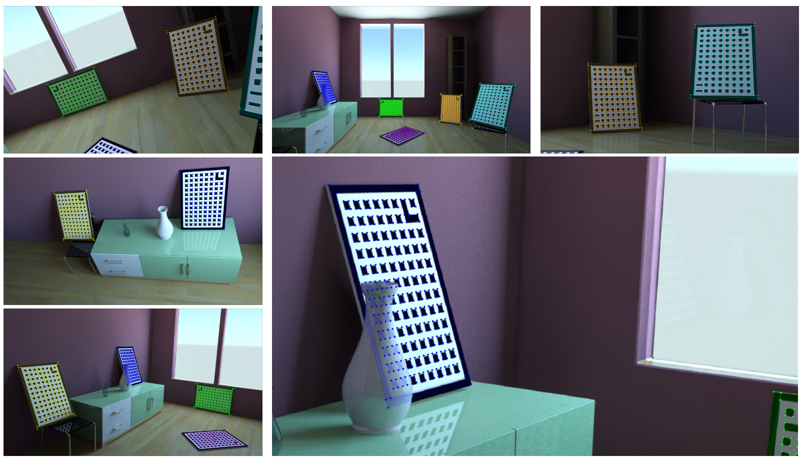


Fig. 5: The synthetic image sequence. **Top row** and **left column**: 5 out of 9 images shown. The positions of the projections of the estimated patterns into the image planes of the estimated cameras can be seen. **Right bottom**: Detail of one of the images. Due to occlusion this pattern is not visible in the image, however, from the other images the position of the pattern can be estimated accurately.

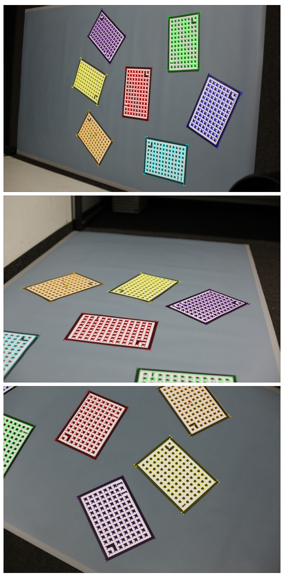
3.2 Real-world Ground Truth Example

For a real-world example we printed seven calibration patterns and arranged them on a paper with millimeter scale. Twelve images of this scene were taken. Since we placed the pattern on the scale paper, in this example all patterns lay in one plane. By using scale paper we were able to measure the corners of the patterns. From the corner points we then computed the centers of the patterns. After applying our camera calibration approach, we were able to compare the estimated pattern positions to the measured ones (see Tab. 6b). The largest absolute difference in Tab. 6b is 1.14 mm in relation to a 583 mm absolute pattern distance (corresponding to a relative deviation of 0.2%).

3.3 Application Example

We applied our approach to the calibration of the stereo-cameras of a robotic head. The cameras of the robotic head are able to move like human eyes and the head of the robot is mounted on a rotating platform. Our goal was to calibrate both cameras for different viewing directions. Because of the ability to look in different direction with the cameras, it is impossible to calibrate the cameras using a single pattern only.

With our method it was possible to calibrate the cameras. The result is shown in Fig. 7. By distributing several calibration patterns in front of the robotic head we managed to see at least one pattern in every image of the robot's cameras. Additionally, we found the calibration procedure to be very easy to apply, as we did not have to pay attention to locate or align the calibration patterns in a specific way with respect to each other.



—	876.94	354.41	368.27	477.54	823.98	829.50
876.94	—	714.22	711.34	399.43	285.58	333.02
354.41	714.22	—	568.29	382.47	559.94	807.66
368.27	711.34	568.29	—	380.13	784.79	540.83
477.54	399.43	382.47	380.13	—	406.04	425.21
823.98	285.58	559.94	784.79	406.04	—	582.95
829.50	333.02	807.66	540.83	425.21	582.95	—

—	876.63	354.20	368.03	477.58	823.16	829.40
876.63	—	713.65	711.24	399.09	285.16	332.37
354.20	713.65	—	567.77	382.19	559.11	806.94
368.03	711.24	567.78	—	380.14	784.00	541.16
477.58	399.09	382.19	380.14	—	405.20	424.76
823.16	285.16	559.11	784.00	405.20	—	581.81
829.40	332.37	806.94	541.16	424.76	581.81	—

—	0.30	0.21	0.23	0.04	0.82	0.11
0.30	—	0.58	0.10	0.34	0.42	0.65
0.21	0.58	—	0.51	0.28	0.83	0.72
0.23	0.10	0.51	—	0.01	0.80	0.33
0.04	0.34	0.28	0.01	—	0.83	0.44
0.82	0.42	0.83	0.80	0.83	—	1.14
0.11	0.65	0.72	0.33	0.44	1.14	—

(a) Image sequence: 3 out of 12 images shown. Patterns placed on paper with millimeter scale. Projections of estimated patterns into the camera images are overlaid.

(b) Evaluation results. All values are given in millimeters. Distance between pattern i and j is given in column i and row j. **Top**: Measured distances between pattern centers. **Middle**: Distances between estimated pattern positions. **Bottom**: Difference between top and middle table.

Fig. 6: Real-world image sequence and evaluation results.

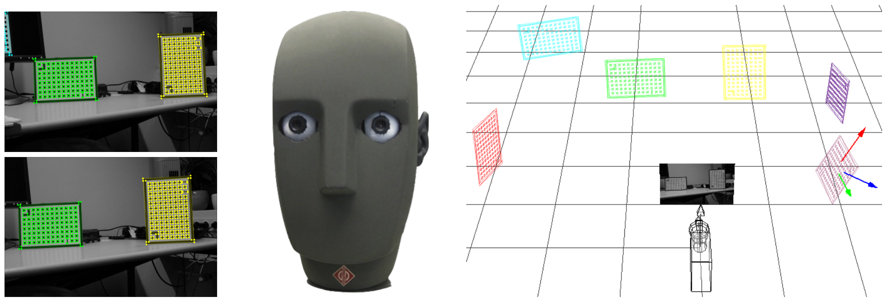


Fig. 7: Application example: This robotic head has two cameras as eyes. The images at the left are two camera images taken with the eye-cameras (calibration patterns are overlaid). The scene at the right shows all reconstructed calibration patterns in 3D space.

4 Conclusion

In this paper we have presented an approach for calibrating multiple camera views in a globally consistent coordinate frame. We make use of multiple calibration patterns that can be distributed in the scene, which makes our approach flexible and easy to use. Furthermore, it addresses shortcomings of single pattern based calibration methods. Intrinsic and extrinsic camera parameters are estimated as well as position and orientation of the calibration patterns. For reliable and accurate estimation results we have simplified the 2D point extraction. After an initial parameter estimation the approximate 2D points are refined to increase accuracy.

Our method has been evaluated on a synthetic and a real-world series of images showing the high flexibility and accuracy of the method. Additionally, the approach has been applied to calibrate the stereo-cameras of a robotic head.

Future work will address how to use several patterns without pattern identifiers, as it should be possible to distinguish the patterns by their relative positions in space.

References

1. Grochulla, M., Thormählen, T., Seidel, H.P.: Using spatially distributed patterns for multiple view camera calibration. In: MIRAGE. (2011)
2. Scharstein, D., Szeliski, R., Zabih, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In: Stereo and Multi-Baseline Vision. (2001) 131–140
3. Tsai, R.: An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision. In: Computer Vision and Pattern Recognition. (1986) 364–374
4. Tsai, R.: A Versatile Camera Calibration Technique for High-accuracy 3D Machine Vision Metrology Using Off-the-shelf TV Cameras and Lenses. IEEE Journal of Robotics and Automation **3** (1987) 323–344
5. Zhang, Z.: Flexible Camera Calibration by Viewing a Plane from Unknown Orientations. International Conference on Computer Vision **1** (1999) 666–673
6. Zhang, Z.: A Flexible New Technique for Camera Calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 1330–1334
7. Ueshiba, T., Tomita, F.: Plane-based Calibration Algorithm for Multi-camera Systems via Factorization of Homography Matrices. In: International Conference on Computer Vision. Volume 2. (2003) 966–973
8. Svoboda, T., Martinec, D., Pajdla, T.: A Convenient Multicamera Self-Calibration for Virtual Environments. Presence: Teleoperators and Virtual Environments **14** (2005) 407–422
9. Maybank, S.J., Faugeras, O.D.: A Theory of Self-Calibration of a Moving Camera. International Journal of Computer Vision **8** (1992) 123–152
10. Hartley, R.I.: An Algorithm for Self Calibration from Several Views. In: Computer Vision and Pattern Recognition. (1994) 908–912
11. Fiala, M.: ARTAG Rev2 Fiducial Marker System: Vision based Tracking for AR. Presented at the Workshop of Industrial Augmented Reality (2005)
12. Fiala, M.: ARTag – augmented reality. Website (2009) http://www.artag.net/, retrieved on March 10th 2011.