# Multiple Cameras Calibration with Occlusion Based on a Marker Tracking Algorithm

Yuji Oyamada*
Keio University
Technische Universität München

Pascal Fallavollita†
Technische Universität München

Nassir Navab‡
Technische Universität München

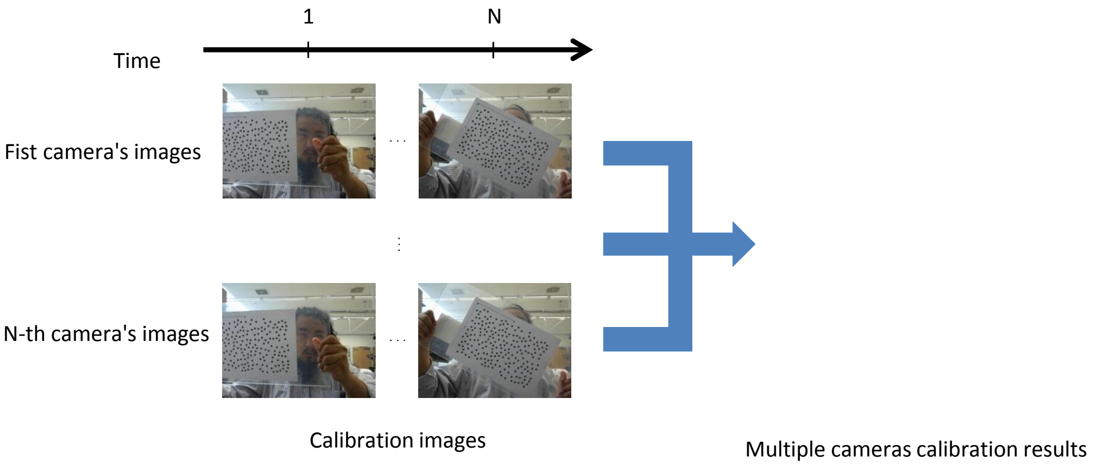

Fig. 1. Multiple cameras calibration with occlusion.

**Abstract**— Multiple cameras calibration is a necessary process for computer vision application such as video based augmented reality. Such augmented reality applications require accurate calibration result so that the applications can achieve geometrical consistency between the real world and virtual objects. In this sense, camera calibration takes important parts of such augmented reality applications. Accurate calibration requires accurate and points correspondence between cameras. With typical calibration objects such as chessboards, it gets more difficult to take good calibration images when we have more cameras or less overlap of cameras' view. This disadvantage makes the image acquisition step longer and may need manual operations to construct points correspondence.

In this paper, we propose a multiple cameras calibration method using a planar calibration object. The proposed method has two key features. One is to introduce a state-of-the-art planar marker tracking algorithm into multiple cameras calibration process. The tracking algorithm localizes control points on planar objects even with partial occlusion. Thus, points correspondence between cameras is automatically achieved even if part of the calibration object is occluded. This advantage allows to increase number of available calibration images and also to put calibration objects around image's boundary. Therefore, we can expect more accurate calibration result while time-saving at the image acquisition step. The other feature is to describe camera parameters by a combination of a projection matrix and lens distortion parameters. Traditional description uses a combination of a calibration matrix, lens distortion parameters, and extrinsic parameters. The proposed method uses a projection matrix in stead of using a calibration matrix and extrinsic parameters so that we can avoid modeling error. The proposed method is validated with both simulation and real world experiments. The experimental results show that the proposed method achieves more accurate result.

**Index Terms**—Radiosity, global illumination, constant time.

## 1 Introduction

[Charmie: Drawbacks] [ToDo: add potential scenario with existing method and Sbovoda's work]

[Charmie: Our idea] Introducing planar object tracking algorithm [24] that

1. uses randomly positioned circle dots as control points
2. is based on a rotation and scaling invariant feature descriptor

*e-mail: charmie@hvrl.ics.keio.ac.jp
†e-mail: fallavol@in.tum.de
‡e-mail: navab@in.tum.de

*Manuscript received 15 September 2011; accepted 3 January 2012; posted online 4 March 2012; mailed on 27 February 2012.*
*For information on obtaining reprints of this article, please send email to: tvcg@computer.org.*

similar to SIFT [17] and SURF [2]

3. handles partial occlusion

- reduces number of photo shooting
- Friendly for unskilled =
- Flexibility = calibration object and points correspondence algorithm

full-automatic multiple cameras calibration = Planar marker tracking + camera calibration same spirit with Pilet *et al.* [20].

Camera(s) calibration is a process that finds camera parameters describing the relationship between 3D world and 2D images observed by a camera (single camera calibration) and/or one between 2D images observed by multiple cameras (multiple cameras calibration). Calibration is a necessary step to run 3D computer vision application such as Augmented Reality and 3D reconstruction.

Typical calibration methods consist of two steps: control points localization and then solving the camera parameters. Basically, there are two choices for calibration object. One type of calibration objects is known shape object. At the beginning of the history of the calibration researches, we use known 3D control points [5, 10]. One of the difficult problems of this type of calibration objects is the control points localization step. One solution is a hardware solution that measures accurate 3D position of control points with special devices such as a turn table and a moving stage. Another solution is to rely on manual operation such as manual correspondence by mouse click. Since this kind of solutions is sometimes difficult to introduce existing systems, an alternative solution had been strongly expected. Then, Zhang proposed a more user-friendly camera calibration method using a known 2D planar object [27, 28]. By designing the planar object to be detectable, *e.g.*, chessboard and circle grid, control points on the planar object can be automatically localized with simple pattern recognition algorithms such as line fitting. Thus, this calibration method has been widely used in vision community. For example, OpenCV [9] provides calibration functions based on Zhang's method with a chessboard and a circle grid and well-known MatLab Camera Calibration Toolbox [3] is also based on the method. The other type of methods, so-called self-calibration, does not assume the knowledge on known shape objects [19, 22, 23]. Self-calibration methods use feature points detected on static objects in the scene as control points, thus the methods can calibrate cameras up to scale factor.

For accurate camera calibration, both control points localization and camera parameters estimation steps should be done as accurate as possible. Practically, users should be careful on the localization step. For example, control points should fill the entire camera view volume. Otherwise, estimated parameters may lack the accuracy, especially lens distortion parameters. This requirement is a dilemma for the users because existing calibration methods assume that entire calibration object is visible on calibration images for control points localization.

For this issue, introducing marker tracking algorithms can be a feasible solution. Well-known marker tracking algorithm is ARToolKit proposed by Kato and Billinghurst [14]. Focusing on binary pattern marker, ARToolKit can find 2D planar objects from camera images based on the marker's constraint. Even though original ARToolKit cannot handle partial occlusion, extended algorithms have been proposed to handle partial occlusion [6]. Recently, even more natural image can be used as a marker object [15]. Using this kind of marker tracking algorithms for control points localization, we can provide more user-friendly camera calibration method in which the operators don't have to take care the visibility of calibration object.

The purpose of this paper is to realize a full automatic single camera calibration method by localizing control points on partially occluded calibration objects. For the purpose, we integrate a marker tracking algorithm that identifies 2D planar markers on input images into a camera calibration procedure. Specifically, we use a RANdom DOts Marker (RANDOM) tracking algorithm [24] for control points localization. With the proposed method, the user only has to move the marker in front of the cameras to be calibrated. Once the control points on the calibration object are localized by the tracking algorithm, the corresponding points are used for camera parameters estimation. The proposed method is validated by two types of experiments: synthetic experiment and real world experiment. The synthetic experiment is performed to numerically evaluate the benefit of the proposed method. The real world experiment is to confirm the proposed method works in practical situation.

Table 1. Calibration objects and their capability for partial occlusion.

| Paper | Calibration objects | Partial occlusion |
|---|---|---|
| Bouguet [3] | Chessboard | X |
| OpenCV [9] | Chessboard/circles grid | X |
| Pilet *et al.* [20] | Natural images | O |
| Svoboda *et al.* [21] | A laser pointer | O |

## 2 Calibration Objects Used in the Literature

This section briefly overviews 2D calibration objects used in the related works. Table 1 summarizes what kind of calibration objects the related works used and their capability for partial occlusion problem.

- Calibration with known shape object [ToDo: Ref to papers].
- Self-calibration solves up to scale factor [21].

Basically, the 2D calibration objects consist of binary color for ease of detection and correspondence. One of the well-used objects is a chessboard [3, 9]. To be robust for varying light conditions, circles and rings grid are usable [4, 9, 12, 25]. The control points localization is done by combination of simple corners/circles detection algorithms and matching algorithms. For unique matching, these methods assume the prior knowledge on the calibration objects, *e.g.*, number of corners/circles. Therefore, if part of the calibration object is invisible due to limited camera's view angle or imperfect feature detection algorithm, the control points localization algorithm fails. As mentioned above, this is one of the big limitation of this type of calibration objects.

To solve the partial occlusion problem, there exist two types of solutions. One solution is to use multiple markers on a planar object [1, 7]. They use a set of planar markers as a 2D calibration object. Embedding a unique ID into each small marker, the calibration object can be identifiable even if part of the calibration object is invisible. Thus, using such calibration objects and their tracking algorithm makes the control points localization algorithm be robust for partial occlusion of the control points. One thing to be considered for this type of calibration objects is its texture. Since the marker tracking algorithm utilizes its texture, texture of the markers should be recognizable.

The other type of solutions relies on the recent development of natural image tracking algorithm [2, 17] Pilet *et al.* proposed geometric and photometric calibration of multiple cameras based on natural image tracking algorithm [20]. Based on robust tracking algorithm running in real-time [15], their calibration method can localize control points on a natural image in real-time.

Considering the above methods, Pilet *et al.*'s method seems to be most robust for partial occlusion. However, if the control points localization algorithm fails, it is difficult for the users to specify which process causes the failure due to the complexity of the tracking algorithm. For analyzability of practical try and error, simpler calibration object such as chessboard and circles grid are more acceptable.

## 3 Camera Model

In this section, we remind the single view/multiple views geometry.

### 3.1 Single view geometry

[Charmie: add the purpose of SingleCamCalib] Let $\boldsymbol{X}_c = [X_c, Y_c, Z_c]^\top$ denotes a point in the camera reference frame and $\boldsymbol{x} = [x, y]^\top$ denotes its projection onto the image plane in the camera coordinate. The homogeneous coordinate of a point is described by its tilde as $\tilde{\boldsymbol{x}} = [\boldsymbol{x}^\top, 1]^\top$.

**Intrinsic parameters.**
[Charmie: add what intrinsic parameters do more clearly] The intrinsic parameters transform a point in 3d camera coordinate to a 2d corresponding point. The 3d point $\boldsymbol{X}_c$ is projected onto the normalized image plane as

$$\boldsymbol{x}_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \frac{1}{Z_c} \begin{bmatrix} X_c \\ Y_c \end{bmatrix}. \tag{1}$$

Following a polynomial lens distortion model [5, 26], the lens distorted point $\boldsymbol{x}_d = [x_d, y_d]^\top$ is described as

$$\boldsymbol{x}_d = \boldsymbol{x}_n + \boldsymbol{d}, \tag{2}$$

where lens distortion $\boldsymbol{d}$ consists of two components, the radial distortion component $\boldsymbol{d}_{\text{rad}}$ and the tangential distortion component $\boldsymbol{d}_{\tan}$ as

$$\boldsymbol{d} = \boldsymbol{d}_{\text{rad}} + \boldsymbol{d}_{\tan}, \tag{3}$$

$$\boldsymbol{d}_{\text{rad}} = (\delta_1 r^2 + \delta_2 r^4 + \delta_5 r^6)\boldsymbol{x}_n, \tag{4}$$

$$\boldsymbol{d}_{\tan} = \begin{bmatrix} 2\delta_3 x_n y_n + \delta_4(3x_n^2 + y_n^2) \\ 2\delta_4 x_n y_n + \delta_3(x_n^2 + 3y_n^2) \end{bmatrix}, \tag{5}$$

$$r = \sqrt{x_n^2 + y_n^2}, \tag{6}$$

where $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_5]^\top$ denotes lens distortion parameters. The final pixel coordinate $\boldsymbol{x}$ is described using calibration matrix $\boldsymbol{A} \in \mathbb{R}^{3\times3}$ as

$$\begin{aligned} \tilde{\boldsymbol{x}}_p &= \boldsymbol{A}\tilde{\boldsymbol{x}}_d \\ &= \boldsymbol{A}(\tilde{\boldsymbol{x}}_n + \tilde{\boldsymbol{d}}) \\ &= \tilde{\boldsymbol{x}}_u + \tilde{\boldsymbol{d}}_p, \end{aligned} \tag{7}$$

where $\boldsymbol{A}$ consists of five parameters as

$$\boldsymbol{A} = \begin{bmatrix} f_x & \theta & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{8}$$

where $[f_x, f_y]^\top$ denotes the focal length along $x$ and $y$ axes respectively, $\theta$ the skew parameter, and $[c_x, c_y]^\top$ the principle point.

**Extrinsic parameters.**
[Charmie: add what extrinsic parameters do more clearly] The extrinsic parameters transform the 3d world coordinate to the 3d camera coordinate. A point $\boldsymbol{X}_w$ in the world coordinate is transformed to one in the camera coordinate $\boldsymbol{X}_w$ by extrinsic parameters as

$$\tilde{\boldsymbol{X}}_c = [\boldsymbol{R}|\boldsymbol{t}]\tilde{\boldsymbol{X}}_w, \tag{9}$$

where $\boldsymbol{R} \in \mathbb{R}^{3\times3}$ denotes the rotation matrix and $\boldsymbol{t} \in \mathbb{R}^3$ denotes the translation vector.

**Projection function.**
Considering the all above components (Eqs. (1), (2), (7), and (9)), the 2D pixel position of a given 3d point is obtained with a 3d point projection function $\text{Proj}(\cdot)$ as

$$\begin{aligned} \tilde{\boldsymbol{x}}_p &= \text{Proj}(\tilde{\boldsymbol{X}}_w, \boldsymbol{A}, \boldsymbol{\delta}, \boldsymbol{R}, \boldsymbol{t}) \\ &= \boldsymbol{A}[\text{Dist}(\tilde{\boldsymbol{X}}_c, \boldsymbol{\delta})] \\ &= \boldsymbol{A}[\text{Dist}([\boldsymbol{R}|\boldsymbol{t}]\tilde{\boldsymbol{X}}_w, \boldsymbol{\delta})], \end{aligned} \tag{10}$$

where the function $\text{Proj}(\cdot)$ projects a 3d point at a world coordinate to a 2d point at a pixel coordinate and the function $\text{Dist}(\cdot)$ adds lens distortion effect.

**Projection matrix and lens distortion model**
Here, we derive a model that describe $\boldsymbol{x}_p$ by using a combination of a projection matrix $\boldsymbol{P}$ and lens distortion $\boldsymbol{d}_p$ instead of $\boldsymbol{A}, \boldsymbol{R}, \boldsymbol{t}$, and $\boldsymbol{\delta}$.

Lens distortion at pixel coordinate can be derived from Eq. (7) and Eq. (8) as

$$\boldsymbol{d}_p = \begin{bmatrix} f_x d_x + \theta d_y \\ f_y d_y \end{bmatrix}. \tag{11}$$

Therefore, $\boldsymbol{x}_p$ is rewritten with a projection matrix as

$$\begin{aligned} \tilde{\boldsymbol{x}}_p &= \boldsymbol{P}\tilde{\boldsymbol{X}}_w + \tilde{\boldsymbol{d}}_p \\ &= \boldsymbol{A}[\boldsymbol{R}|\boldsymbol{t}]\tilde{\boldsymbol{X}}_w + \tilde{\boldsymbol{d}}_p. \end{aligned} \tag{12}$$

### 3.2 Multiple views geometry

[Charmie: add the purpose of MulCamCalib] Here, we consider the multiple views geometry with different descriptions of camera parameters. Suppose we have $N$ cameras $\{C_i\}$ $i = 1, \ldots, N$. Knowing the multiple views geometry means that we find the geometrical relation between views. We choose one of the cameras as a reference camera $C_{\text{ref}}$.

**Traditional parameterization.**
Goal of the multiple cameras calibration Following Eq. (9), the geometrical relation between the world coordinate and $i$-th camera coordinate is described by its extrinsic parameters $(\boldsymbol{R}_i, \boldsymbol{t}_i)$ as

$$\boldsymbol{X}_i = \boldsymbol{R}_i \boldsymbol{X}_w + \boldsymbol{t}_i. \tag{13}$$

To know the relative relation between cameras, we define the relative extrinsic parameters $(\boldsymbol{R}_{\text{rel},i}, \boldsymbol{t}_{\text{rel},i})$ that transforms the reference camera coordinate, 1st camera, to $i$-th camera coordinate as

$$\boldsymbol{X}_i = [\boldsymbol{R}_{\text{rel},i}|\boldsymbol{t}_{\text{rel},i}]\boldsymbol{X}_{\text{ref}}, \tag{14}$$

where the relative extrinsic parameters are written as

$$\begin{cases} \boldsymbol{R}_{\text{rel},i} \doteq \boldsymbol{R}_i \boldsymbol{R}_{\text{ref}}^\top \\ \boldsymbol{t}_{\text{rel},i} \doteq \boldsymbol{t}_i - \boldsymbol{R}_i \boldsymbol{R}_{\text{ref}}^\top \boldsymbol{t}_{\text{ref}}. \end{cases} \tag{15}$$

Note that $\boldsymbol{R}_{\text{ref,ref}}$ is identity matrix and $\boldsymbol{t}_{\text{ref,ref}}$ is zero vector.

**Projection matrix and lens distortion model**
Same as single camera geometry, we describe relative geometric relation between camera views by using a projection matrix instead of extrinsic parameters.

Suppose a scene that two cameras, $C_{\text{ref}}$ and $C_i$, see a 3d point $\boldsymbol{X}_w$. A relative projection matrix $\boldsymbol{P}_{\text{rel},i}$ that projects a 3d point in the reference camera coordinate $\boldsymbol{X}_{\text{ref}}$ to a 2d point in the $i$-th camera's pixel coordinate $\boldsymbol{x}_{p,i}$ as

$$\tilde{\boldsymbol{x}}_{p,i} = \boldsymbol{P}_{\text{rel},i}\tilde{\boldsymbol{X}}_{\text{ref}}. \tag{16}$$

The relative projection matrix $\boldsymbol{P}_{\text{rel},i}$ can be obtained by combining Eq. (7) and Eq. (14) as

$$\boldsymbol{P}_{\text{rel},i} = \boldsymbol{A}_i[\boldsymbol{R}_{\text{ref},i}\boldsymbol{t}_{\text{ref},i}] \tag{17}$$

## 4 Proposed method

$$\{x_{n,m,l} \leftrightarrow X_l\} = \text{findPointsCorrespondence}\left(I_{n,m}, \{X_l\}\right) \tag{18}$$

$$A_n, \boldsymbol{\delta}_n, \{\boldsymbol{R}_{n,m}, \boldsymbol{t}_{n,m}\} = \text{calibrateSingleCamera}\left(\{x_{n,m,l} \leftrightarrow X_l\}\right) \tag{19}$$

$$\{\boldsymbol{R}_{\text{ref},m}, \boldsymbol{t}_{\text{ref},m}\}, \{\boldsymbol{R}_{\text{rel},n}, \boldsymbol{t}_{\text{rel},n} \mid n = 2, \ldots, N\}, \{A_n, \boldsymbol{\delta}_n\} \tag{20}$$

$$= \text{calibrateMultipleCameras}\left(\{A_n, \boldsymbol{\delta}_n, \{\boldsymbol{R}_{n,m}, \boldsymbol{t}_{n,m}\}, \{x_{n,m,l} \leftrightarrow X_l\}\}\right) \tag{21}$$

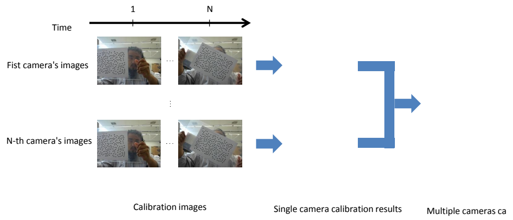

Fig. 2. Overview of the proposed method.

**Algorithm 1** Multiple cameras calibration algorithm.

```
Input: 3d position of control points $\boldsymbol{X} = \{\boldsymbol{X}_k\}$ and sets of image sequences $\{\boldsymbol{I}_{i,j}\}$.
  Step1: Points Correspondence
  for $i = 1$ to $N$ do
      for $j = 1$ to $M$ do
          Find key-points on an image $\boldsymbol{u}_{i,j} = \{\boldsymbol{u}_{i,j,1} \dots \boldsymbol{u}_{i,j,L_{i,j}}\}$.
          Make points correspondence $\boldsymbol{C}_{i,j} = \{\boldsymbol{X}_k \leftrightarrow \boldsymbol{u}_{i,j,k}\}$.
      end for
  end for
  Step2: Single Camera Calibration
  for $i = 1$ to $N$ do
      Compute intrinsic parameters $\boldsymbol{A}_i$ and extrinsic parameters $\{(\boldsymbol{R}_{i,j}, \boldsymbol{t}_{i,j})\}$ by minimizing Eq. (23).
  end for
  Step3: Multiple Cameras Calibration
  Refine $\{\boldsymbol{A}_i\}$ and estimate relative extrinsic parameters $\{(\boldsymbol{R}_{\mathrm{ref},i}, \boldsymbol{t}_{\mathrm{ref},i})\}$ by minimizing Eq. (30).
Output: Estimated parameters $\{\boldsymbol{A}_i\}$ and $\{(\boldsymbol{R}_{\mathrm{ref},i}, \boldsymbol{t}_{\mathrm{ref},i})\}$.
```

Fig. 2 shows a scene assumed in this paper that $N$ cameras observe a moving calibration object. Each camera observes a sequences of $M$ images $\boldsymbol{I}_{i,j}$ $j = 1, \dots, M$, thus, we have totally $N \times M$ input images. As a calibration object, our method uses a planar object that has $L$ randomly distributed circle dots, the 3d position of which is described as $\boldsymbol{X}_k = [X_k, Y_k, Z_k]^\top$ $k = 1, \dots, L$. Those dots are used as control points. On an image $\boldsymbol{I}_{i,j}$, the center of $k$-th circle dot appears as ellipse at $\boldsymbol{x}_{i,j,k} = [x_{i,j,k}, y_{i,j,k}]^\top$.

Algorithm 1 briefly explains the overview of the proposed method. Our task is to calibrate all $N$ cameras with three steps given the $N \times M$ input images. Specifically, our calibration method estimates intrinsic parameters of all cameras $\{(\boldsymbol{K}_i, \boldsymbol{\delta}_i)\}$ $i = 1, \dots, N$ and relative extrinsic parameters $\{(\boldsymbol{R}_{\mathrm{ref},i}, \boldsymbol{t}_{\mathrm{ref},i})\}$ $i = 1, \dots, N$, here the 1st camera is used as the reference camera. In the first step (Sec. 4.1), the proposed method makes 3d-2D points correspondence correspondence based on circle dots tracking algorithm [24]. Next, we perform a single camera calibration (Sec. 4.2) for each camera given the set of corresponding points. This step estimates intrinsic parameters $\{(\boldsymbol{K}_i, \boldsymbol{\delta}_i)\}$ and extrinsic parameters of each image w.r.t. the world coordinate $[(\boldsymbol{R}_{i,1}, \boldsymbol{t}_{i,1}), \dots, (\boldsymbol{R}_{i,M}, \boldsymbol{t}_{i,M})]$ for each camera using a combination of solving direct linear transformation and non-linear optimization. Finally, multiple cameras calibration step (Sec. 4.3) refines the intrinsic parameters $\{(\boldsymbol{K}_i, \boldsymbol{\delta}_i)\}$ and estimates the relative extrinsic parameters $\{(\boldsymbol{R}_{\mathrm{ref},i}, \boldsymbol{t}_{\mathrm{ref},i})\}$ by using non-linear optimization.

### 4.1 Points correspondence based on tracking

The first step makes points correspondence between known 3d position of control points on the calibration object and 2d position of their projection detected on input images. In other words, this step makes sets of corresponding points $\{\boldsymbol{X}_k \leftrightarrow \boldsymbol{x}_{i,j,k}\}$ for all images.

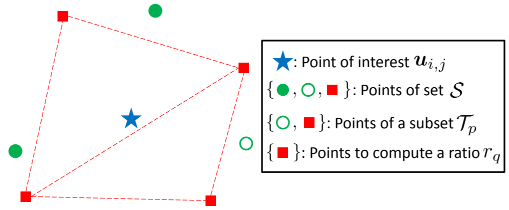
★: Point of interest $\boldsymbol{u}_{i,j}$
{●, ○, 🟥}: Points of set $\mathcal{S}$
{○, 🟥}: Points of a subset $\mathcal{T}_p$
{🟥}: Points to compute a ratio $r_q$

Fig. 3. Feature vector for a key-point (an example of $n = 7, m = 5$).

**Key-points extraction** We first detect control points on the calibration object by key-points extraction algorithm. The control points are circle and the calibration object consists of two colors, *e.g.*, black circle dots on white background color in Fig. 2. Thus, combination of binarization and ellipse fitting is enough for this case. We first binarize an input image $\boldsymbol{I}_{i,j}$ to extract dot regions in the image. Then, ellipse fitting is performed to the binarized image. Let $\boldsymbol{u}_{i,j,k} = [u_{i,j,k}, v_{i,j,k}]^\top$ denote the 2d position of the center of an extracted ellipse and $L_{i,j}$ the number of the extracted ellipses. Note $L_{i,j}$ may vary along time $j$ due to partial occlusion and background object.

**Feature vector computation** Next, we compute a feature vector for every key-point and then perform feature vector matching between the input image and the reference image. This sub-step is based on RANdom DOts Markers (RANDOM) tracking algorithm proposed by Uchiyama and Saito [24].

A feature descriptor used in this step is a ratio between two bordering triangles given four key-points. Suppose we have a point of interest and its four neighboring points. A ratio of two triangles' area made with the neighboring points is the feature descriptor. Since the descriptor uses the ratio of areas, it is rotation and scale invariant descriptor.

To obtain better recognition ability, we assign multiple feature vectors for a key-point as shown in Fig. 3. For an extracted key-point $\boldsymbol{u}_{i,j,k}$, we make a set of $n$ neighboring key-points $\mathscr{S} = \{\boldsymbol{u}_{i,j,k+1} \dots \boldsymbol{u}_{i,j,k+n}\}$, $n = 7$ in Fig. 3. From the set $\mathscr{S}$, we make all combinations of subsets, one of which consists of $m$ of $n$ points, $\mathscr{T} = \{\mathscr{T}_1 \dots \mathscr{T}_p \dots \mathscr{T}_{{}_nC_m}\}$, $m = 5$ in Fig. 3. For a subset $\mathscr{T}_p$, we compute a feature vector $\mathscr{V}_p$. One dimension of $\mathscr{V}_p$ is the ratio of two triangles' area computed from 4 of $m$ points. Therefore, the feature vector $\mathscr{V}_p$ is ${}_mC_4$ dimensional vector written as $\mathscr{V}_p = [r_{p,1} \dots r_{p,q} \dots r_{p,{}_mC_4}]^\top$ where $r_{p,q}$ denotes the ratio of $q$-th bordering triangles' area of $p$-th subset. For fast matching, the feature vectors on a key-point is converted into a 1D index by a hash function.

**Feature vector matching** Once we compute the feature vectors on the input image, the vectors are matched to ones on the reference image. The proposed method performs two types of matching: initial matching and refinement as shown in Fig. 4.

The initial matching makes points correspondence by comparing the key-points extracted on the input images and ones on the reference image w.r.t. the computed feature vectors. Let $\boldsymbol{C}_{\mathrm{init},i,j} = \{\boldsymbol{X}_k \leftrightarrow \boldsymbol{u}_{i,j,k}\}$ denote a set of corresponding points obtained by the initial matching. Due to imperfectness of the feature descriptor, this matching result $\boldsymbol{C}_{\mathrm{init},i,j}$ may contain two types of errors. One is false-positive error, miss-matching in other words, resulting unreliable calibration. The other one is false-negative error meaning lack of matching. Thus, this initial matching result is refined by two steps.

First step removes false-positive error. Since the calibration object is planar, we omit false-positive matching from the corresponding points based on RANSAC algorithm [8]. The RANSAC based outlier rejection provides a set of corresponding points $\boldsymbol{C}_{\mathrm{RANSAC},i,j}$, which may have less number of corresponding points. Then, false-negative error is removed by using the knowledge of the reference image. First, the proposed method computes a homography given the set of corresponding points $\boldsymbol{C}_{\mathrm{RANSAC},i,j}$ and reproject the known control points $\boldsymbol{X} = \{\boldsymbol{X}_1 \dots \boldsymbol{X}_L\}$ onto the image. Then, the reprojected points

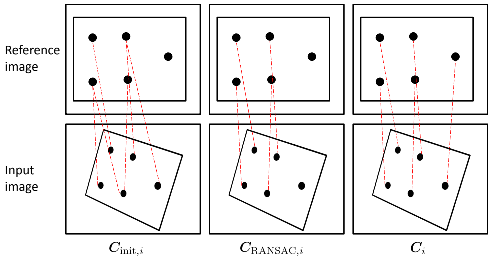

Fig. 4. Overview of the key-points matching algorithm.

$\boldsymbol{x}_{i,j} = \{\boldsymbol{x}_{i,j,1} \dots \boldsymbol{x}_{i,j,L}\}$ are matched to its neighboring points. Furthermore, we assign a visibility term $v_{i,j,k}$ for each point. If $k$-th control point $\boldsymbol{X}_k$ has its corresponding point on image $\boldsymbol{I}_{i,j}$, the visibility term $v_{i,j,k}$ is set to 1, and 0 otherwise. Finally, we obtain a set of refined corresponding points $\boldsymbol{C}_{i,j} = \{\boldsymbol{X}_k \leftrightarrow \boldsymbol{x}_{i,j,k}\}$.

### 4.2 Single camera calibration

Given the set of corresponding points $\{\boldsymbol{X}_k \leftrightarrow \boldsymbol{x}_{i,j,k}\}$, this step estimates intrinsic parameter $(\boldsymbol{K}_i, \boldsymbol{\delta}_i)$ and extrinsic parameters of each image w.r.t. the world coordinate $[(\boldsymbol{R}_{i,1}, \boldsymbol{t}_{i,1}), \dots, (\boldsymbol{R}_{i,M}, \boldsymbol{t}_{i,M})]$ for each camera.

Zhang's planar based calibration [28], Heikkilä and Silvén's calibration method [13] and Camera Calibration Toolbox for MatLab [3]

**DLT for intrinsic and extrinsic parameters initialization** For each input image $\boldsymbol{I}_{i,j}$, we first compute a homography $\boldsymbol{H}_{i,j}$ describing a 2d-2d transformation from the reference image plane $\boldsymbol{I}_0$ to the input image plane $\boldsymbol{I}_{i,j}$ as $\boldsymbol{I}_{i,j} = \boldsymbol{H}_{i,j}\boldsymbol{I}$. Similar to the visibility term $v_{i,j,k}$ for each control point, we also define a term $v_{i,j}$ for each image that describes whether an image $\boldsymbol{I}_{i,j}$ has enough control points to compute homography or not. The term is written as

$$v_{i,j} = \begin{cases} 1 & \text{if } \displaystyle\sum_{k=1}^{L} v_{i,j,k} \geq \tau_{\text{homo}} \\ 0 & \text{otherwise} \end{cases}, \tag{22}$$

where the thresholding value $\tau_{\text{homo}}$ represents minimum number of points to compute a homography. The thresholding value $\tau_{\text{homo}}$ is theoretically 4, however we empirically assign 8. Using the homography availability term $v_{i,j}$, we can decompose a set $\mathscr{J} = \{1, \dots, M\}$ into two subsets $\mathscr{J}_i$ and its complement $\mathscr{J}_i^{\complement}$. The subset $\mathscr{J}_i$ contains indexes whose $v_{i,j}$ is 1 while $\mathscr{J}_i^{\complement}$ contains indexes whose $v_{i,j}$ is 0.

From a set of the computed homographies $\{\boldsymbol{H}_{i,1}, \dots, \boldsymbol{H}_{i,M}\}$, we estimate the intrinsic parameter $\boldsymbol{A}_i$ using orthogonality of vanishing points obtained from homographies [11]. Once the intrinsic parameters are estimated, we compute the extrinsic parameters $(\boldsymbol{R}_{i,j}, \boldsymbol{t}_{i,j})$ given $\boldsymbol{A}_i$ and the homography $\boldsymbol{H}_{i,j}$

**Non-linear optimization for all parameters estimation** The remaining parameters to be estimated is lens distortion parameters $\boldsymbol{\delta}$. However, this sub-step also refines $\{\boldsymbol{A}_i\}$ and $\{(\boldsymbol{R}_{i,j}, \boldsymbol{t}_{i,j})\}$ obtained through the above process because the parameters obtained by DLT minimizes an algebraic distance without considering lens distortion. Thus, non-linear optimization such as Levenberg-Marquardt algorithm [16, 18] is performed to estimate/refine the all parameters by minimizing the following energy term:

$$\sum_{j=1}^{M} \sum_{k=1}^{L} v_{i,j,k} \left\| \boldsymbol{x}_{i,j,k} - P(\tilde{\boldsymbol{X}}_k, \boldsymbol{A}_i, \boldsymbol{\delta}_i, \boldsymbol{R}_{i,j}, \boldsymbol{t}_{i,j}) \right\|^2. \tag{23}$$

### 4.3 Multiple cameras calibration

Finally, the proposed method estimates the relative extrinsic parameters between the reference camera and each camera given corresponding points and all parameters estimated through the previous step. This step takes two sub-steps: relative extrinsic parameters initialization and all parameters refinement.

**Relative extrinsic parameters initialization** First sub-step is to roughly estimate the relative extrinsic parameters $(\boldsymbol{R}_{\text{ref},i}, \boldsymbol{t}_{\text{ref},i})$. Let $(\boldsymbol{R}_{\text{ref},i,j}, \boldsymbol{t}_{\text{ref},i,j})$ denote the relative extrinsic parameters at time $j$. The parameters are obtained by using Eq. (15) as

$$\boldsymbol{R}_{\text{ref},i,j} = \boldsymbol{R}_{i,j}\boldsymbol{R}_{1,j}^{\top} \text{ and} \tag{24}$$

$$\boldsymbol{t}_{\text{ref},i,j} = \boldsymbol{t}_{i,j} - \boldsymbol{R}_{i,j}\boldsymbol{R}_{1,j}^{\top}\boldsymbol{t}_{1,j}. \tag{25}$$

As mentioned above, we assume rigid cameras that are relatively fixed. Thus, $(\boldsymbol{R}_{\text{ref},i,j}, \boldsymbol{t}_{\text{ref},i,j})$ should be constant along time $j$. Therefore, initial estimate of $(\boldsymbol{R}_{\text{ref},i}, \boldsymbol{t}_{\text{ref},i})$ is obtained by taking a median value of $(\boldsymbol{R}_{\text{ref},i,j}, \boldsymbol{t}_{\text{ref},i,j})$ along time $j$. Since the calibration object may not appear on some images, the median value should be computed only from images whose indexes are elements of a subset $\mathscr{J}_i$. Thus, we compute the initial estimate of the relative extrinsic parameters as

$$\boldsymbol{R}_{\text{ref},i} = \text{Angle2Mat}(\underset{j \in \mathscr{J}_i}{\text{median}}(\text{Mat2Angle}(\boldsymbol{R}_{\text{ref},i,j}))) \text{ and} \tag{26}$$

$$\boldsymbol{t}_{\text{ref},i} = \underset{j \in \mathscr{J}_i}{\text{median}}(\boldsymbol{t}_{\text{ref},i,j}), \tag{27}$$

where the function Mat2Angle() converts a rotation matrix to three rotation angles, and Angle2Mat() does its inverse.

**All parameters refinement** Then, the final step refines all the parameters: the intrinsic parameters $\{(\boldsymbol{K}_i, \boldsymbol{\delta}_i)\}$ and the relative extrinsic parameters $\{(\boldsymbol{R}_{\text{ref},i}, \boldsymbol{t}_{\text{ref},i})\}$ because the parameters are optimized w.r.t. the reprojection error Eq. (23) that does not considers the rigidity of cameras. In this step, we have to consider which image is available for optimization. To chain all extrinsic parameters, we use the absolute extrinsic parameters of the reference camera $\{(\boldsymbol{R}_{1,1}, \boldsymbol{t}_{1,1}) \dots (\boldsymbol{R}_{1,M}, \boldsymbol{t}_{1,M})\}$ and the relative extrinsic parameters $\{(\boldsymbol{R}_{\text{ref},1}, \boldsymbol{t}_{\text{ref},1}) \dots (\boldsymbol{R}_{\text{ref},N}, \boldsymbol{t}_{\text{ref},N})\}$ instead of the all absolute extrinsic parameters $\{\{(\boldsymbol{R}_{1,j}, \boldsymbol{t}_{1,j})\} \dots \{(\boldsymbol{R}_{N,j}, \boldsymbol{t}_{N,j})\}\}$. The thing we have to consider is the homography availability term on 1st camera $v_{1,j}$. If $v_{1,j}$ is 0, any images at time $j$ is not available. Thus, before running nonlinear optimization, we first interpolate missing extrinsic parameters. For time $j$ such that $j \in \mathscr{J}_1^{\complement}$ and $j \in \mathscr{J}_i$ meaning that the calibration object is not detected by the reference camera but is detected by $i$-th camera at time $j$. Then, the absolute extrinsic parameters $(\boldsymbol{R}_{1,j}, \boldsymbol{t}_{1,j})$ are obtained as

$$\boldsymbol{R}_{1,j} = \boldsymbol{R}_{\text{ref},i}^{\top}\boldsymbol{R}_{i,j} \text{ and} \tag{28}$$

$$\boldsymbol{t}_{1,j} = \boldsymbol{R}_{\text{ref},i}^{\top}(\boldsymbol{t}_{i,j} - \boldsymbol{t}_{\text{ref},i}). \tag{29}$$

Then, we refine all the parameters by minimizing the following energy term:

$$\sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{k=1}^{L} v_{i,j,k} \left\| \boldsymbol{x}_{i,j,k} - P(\tilde{\boldsymbol{X}}_k, \boldsymbol{A}_i, \boldsymbol{\delta}_i, \check{\boldsymbol{R}}_{i,j}, \check{\boldsymbol{t}}_{i,j}) \right\|^2, \tag{30}$$

where the recomputed extrinsic parameters $\check{\boldsymbol{R}}_{i,j}$ and $\check{\boldsymbol{t}}_{i,j}$ are obtained as

$$\check{\boldsymbol{R}}_{i,j} = \boldsymbol{R}_{\text{ref},i}\boldsymbol{R}_{1,j} \text{ and} \tag{31}$$

$$\check{\boldsymbol{t}}_{i,j} = \boldsymbol{t}_{\text{ref},i} + \boldsymbol{R}_{\text{ref},i}\boldsymbol{t}_{1,j}. \tag{32}$$

## 5 Experiments

### 5.1 Simulation experiments

### 5.2 Real world experiments

## 6 Conclusion

## References

[1] B. Atcheson, F. Heide, and W. Heidrich. CALTag: High precision fiducial markers for camera calibration. In *International Workshop on Vision, Modeling and Visualization*, 2010.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[3] J.-Y. Bouguet. Camera calibration toolbox for matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`, 2008.

[4] A. Datta, J. Kim, and T. Kanade. Accurate camera calibration using iterative refinement of control points. In *ICCV Workshop on Visual Surveillance (VS)*, 2009.

[5] O. D. Faugeras and G. Toscani. The calibration problem for stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Publ.86CH2290-5, pages 15–20. IEEE, 1986.

[6] M. Fiala. Artag, a fiducial marker system using digital techniques. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 590–596, 2005.

[7] M. Fiala and C. Shu. Self-identifying patterns for plane-based camera calibration. *Machine Vision and Applications*, 19(4):209–216, 2008.

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[9] W. Garage. Open source computer vision library (opencv) ver. 2.4.2. `http://opencv.org/`, 2012.

[10] E. L. Hall, J. B. K. Tio, C. A. McPherson, and F. A. Sadjadi. Measuring curved surfaces for robot vision. *Computer*, 15(12):42–54, 1982.

[11] R. Hartley and A. Zisserman. Determining camera calibration k from a single view. In *Multiple View Geometry in Computer Vision*, chapter 8.8, pages 223–229. Cambridge University Press, 2004.

[12] J. Heikkilä. Geometric camera calibration using circular control points. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(10):1066–1077, 2000.

[13] J. Heikkilä and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1106–1112, Washington, DC, USA, 1997. IEEE Computer Society.

[14] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *2nd IEEE and ACM International Workshop on Augmented Reality (IWAR)*, pages 85–94, Washington, DC, USA, 1999. IEEE Computer Society.

[15] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(9):1465–1479, 2006.

[16] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, 2(2):164–168, 1944.

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[18] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[19] S. J. Maybank and O. D. Faugeras. A theory of self-calibration of a moving camera. *International Journal on Computer Vision*, 8(2):123–151, 1992.

[20] J. Pilet, A. Geiger, P. Lagger, V. Lepetit, and P. Fua. An all-in-one solution to geometric and photometric calibration. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, ISMAR '06, pages 69–78, Washington, DC, USA, 2006. IEEE Computer Society.

[21] T. Svoboda, D. Martinec, and T. Pajdla. A convenient multi-camera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422, 2005.

[22] B. Triggs. Autocalibration from planar scenes. In *European Conference on Computer Vision (ECCV)*, 1998.

[23] Q. tuan Luong and O. D. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal on Computer Vision*, 22(3):261–289, 1997.

[24] H. Uchiyama and H. Saito. Random dot markers. In *IEEE Virtual Reality Conference*, 2011.

[25] M. Vo, Z. Wang, L. luu, and J. Ma. Advanced geometric camera calibration for machine vision. *Optical Engineering*, 50(11):110503, 2011.

[26] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965–980, 1992.

[27] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *International Conference on Computer Vision (ICCV)*, pages 666–673, 1999.

[28] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.