Single Camera Calibration using partially visible calibration objects based on Random Dots Marker Tracking Algorithm

Yuji Oyamada[*]
Keio University
Technische Universität München

Pascal Fallavollita[†]
Technische Universität München

Nassir Navab[‡]
Technische Universität München

Abstract

Camera calibration is one of the long existing research issues in computer vision domain. Typical calibration methods take two steps for the procedure: control points localization and camera parameters computation. In practical situation, control points localization is a time-consuming task because the localization puts severe assumption that the calibration object should be visible in all images. To satisfy the assumption, users may avoid moving the calibration object near the image boundary. As a result, we estimate poor quality parameters.

In this paper, we aim to solve this partial occlusion problem of the calibration object. To solve the problem, we integrate a planar marker tracking algorithm that can track its target marker even with partial occlusion. Specifically, we localize control points by a RANdom DOts Markers (RANDOM) tracking algorithm that uses markers with randomly distributed circle dots. , and localizes the dots by a RANdom DOts Markers (RANDOM) tracking algorithm. Once the control points are localized, they are used to estimate the camera parameters. The proposed method is validated with both synthetic and real world experiments. The experimental results show that the proposed method realizes camera calibration from image on which part of the calibration object is visible.

Index Terms: I.4.9 [Computing Methogologies]: Image Processing and Computer Vision—Applications

1 Introduction

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 [26, 27]. 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 [18, 20, 21]. 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 [13]. 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 [14]. Using this kind of marker tracking algorithms for control points localization, we can provide more user-friendly camera calibration method in which the users don't have 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 [22] 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.

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.

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

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
Heikkilä [12]	Circles grid	X
Vo *et al.* [23]	Rings grid	X
Datta *et al.* [4]	Circles/rings grid	X
Fiala and Shu [7]	ARTag [6]	O
Atcheson *et al.* [1]	CALTag	O
Pilet *et al.* [19]	Natural images	O

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, 23]. 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, 16] Pilet *et al.* proposed geometric and photometric calibration of multiple cameras based on natural image tracking algorithm [19]. Based on robust tracking algorithm running in real-time [14], 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 Proposed method

The proposed method assumes a scene that the user moves calibration object, *i.e.*, a random dots marker, in front of a camera to be calibrated. As a calibration object, our method uses a planar object that has M randomly distributed circle dots, the 3D position of which is described as $\boldsymbol{X}_j = [X_j, Y_j, Z_j]^\top$ $j = 1, \dots, M$. The camera observes a sequences of N images $\boldsymbol{I}_i$ $i = 1, \dots, N$. On an image $\boldsymbol{I}_i$, the center of j-th circle dot appears as ellipse at $\boldsymbol{x}_{i,j} = [x_{i,j}, y_{i,j}]^\top$. As shown in Fig. 1, the proposed method can handle input images on which the calibration object is partially visible.

Figure 1 and Alg. 1 briefly explain the overview of the proposed method. The proposed method calibrates the camera with two steps given the N input images. The first step (Sec. 3.2) localizes the control points on each captured image. The 3D-2D points correspondence is built based on random dots tracking algorithm [22]. In Fig. 1, red circles represent the localized control points while blue circles represent the other detected key-points not control points. Once the points correspondences are built, they are used to estimate the camera parameters, namely the intrinsic parameters and the lens distortion parameters $(\boldsymbol{K}, \boldsymbol{\delta})$, based on Zhang's calibration algorithm [26, 27].

Algorithm 1 Single cameras calibration algorithm.

```
Input: 3D position of control points $\boldsymbol{X} = \{\boldsymbol{X}_j\}$ and sets of image sequences $\{\boldsymbol{I}_i\}$.
  Step1: Control points localization
  for $i = 1$ to $N$ do
      Find key-points on an image $\boldsymbol{u}_i = \{\boldsymbol{u}_{i,1} \dots \boldsymbol{u}_{i,M_i}\}$.
      Make points correspondence $\boldsymbol{C}_i = \{\boldsymbol{X}_j \leftrightarrow \boldsymbol{x}_{i,j}\}$.
  end for
  Step2: Single Camera Calibration
  for $i = 1$ to $N$ do
      Compute a homography $\boldsymbol{H}_i$.
  end for
  Compute intrinsic parameters $\boldsymbol{A}$.
  for $i = 1$ to $N$ do
      Compute extrinsic parameters $(\boldsymbol{R}_i, \boldsymbol{t}_i)$.
  end for
  Estimate lens distortion parameters $\boldsymbol{\delta}$ and refine $\boldsymbol{A}$ by minimizing Eq. (11).
Output: Estimated parameters $\boldsymbol{A}$ and $\boldsymbol{\delta}$.
```

3.1 Camera model

Before going to the detail of the proposed method, this section quickly reminds basic theory of pinhole camera model. 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. The intrinsic parameters transform a point in 3D coordinate into the pixel coordinate. The 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, 24], the lens distorted point $\boldsymbol{x}_d = [x_d, y_d]^\top$ is described as

$$\boldsymbol{x}_d = \boldsymbol{x}_n + \boldsymbol{d}_{\mathrm{rad}} + \boldsymbol{d}_{\tan}, \tag{2}$$

$$\boldsymbol{d}_{\mathrm{rad}} = \begin{bmatrix} (\delta_1 r^2 + \delta_2 r^4 + \delta_5 r^6) x_n \\ (\delta_1 r^2 + \delta_2 r^4 + \delta_5 r^6) y_n \end{bmatrix}, \tag{3}$$

$$\boldsymbol{d}_{\tan} = \begin{bmatrix} 2\delta_3 x_n y_n + \delta_4 (3x_n^2 + y_n^2) \\ 2\delta_4 x_n y_n + \delta_3 (x_n^2 + 3y_n^2) \end{bmatrix}, \tag{4}$$

$$r = \sqrt{x_n^2 + y_n^2}, \tag{5}$$

where $\boldsymbol{\delta} = [\delta_1, \dots, \delta_5]^\top$ denotes lens distortion parameters and $\boldsymbol{d}_{\mathrm{rad}}$ and $\boldsymbol{d}_{\tan}$ denote the radial distortion and tangential distortion vector respectively. The final pixel coordinate $\boldsymbol{x}$ is described using calibration matrix $\boldsymbol{A} \in \mathbb{R}^{3\times 3}$ as

$$\boldsymbol{A} = \begin{bmatrix} f_x & \theta & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

$$\tilde{\boldsymbol{x}} = \boldsymbol{A} \tilde{\boldsymbol{x}}_d, \tag{7}$$

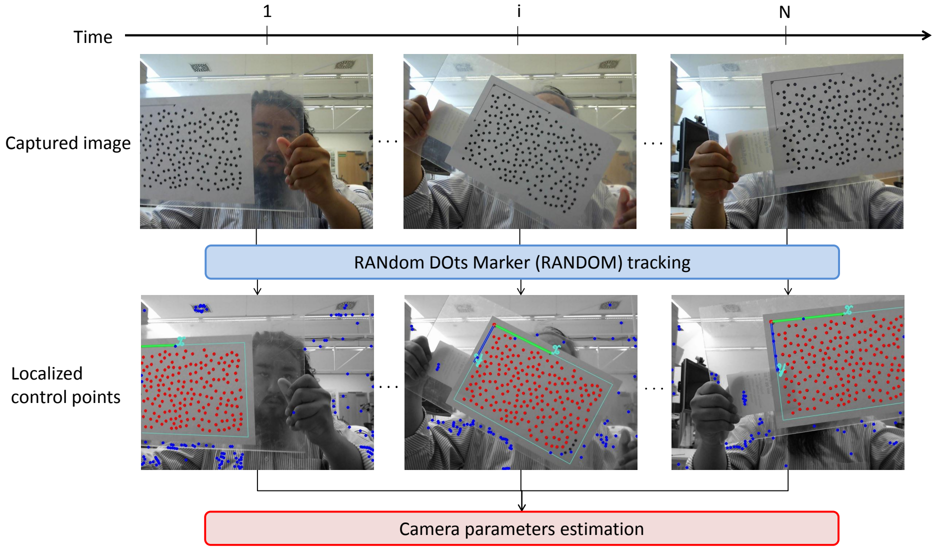

Time
1
i
N
Captured image
RANdom DOts Marker (RANDOM) tracking
Localized control points
Camera parameters estimation

Figure 1: Overview of the proposed method. Step1: Control points localization by RANDOM tracking algorithm. Step2: Camera parameters estimation using the localized control points.

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

Extrinsic parameters. The extrinsic parameters transform the 3D world coordinate to the 3D camera reference coordinate. A point $\boldsymbol{X}_c$ in the world coordinate is transformed to one in the camera reference coordinate by extrinsic parameters as

$$\boldsymbol{X}_c = \boldsymbol{R}\boldsymbol{X} + \boldsymbol{t}, \tag{8}$$

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

Considering the all above components (Eqs. (1), (2), (7), and (8)), the 2D pixel position of a given 3D point is obtained with a 3D point projection function $\mathrm{Proj}(\cdot)$ as

$$\tilde{\boldsymbol{x}} = \mathrm{Proj}(\tilde{\boldsymbol{X}}, \boldsymbol{A}, \boldsymbol{\delta}, \boldsymbol{R}, \boldsymbol{t}). \tag{9}$$

3.2 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 $\{\boldsymbol{X}_j \leftrightarrow \boldsymbol{x}_{i,j}\}$ for all images.

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. 1. Thus, combination of binarization and ellipse fitting is enough for this case. We first binarize an input image $\boldsymbol{I}_i$ to extract dot regions in the image. Then, ellipse fitting is performed to the binarized image. Let $\boldsymbol{u}_{i,j} = [u_{i,j}, v_{i,j}]^\top$ denote the 2D position of the center of an extracted ellipse and M_i the number of the extracted ellipses. Note M_i may vary along time i due to partial occlusion and background object.

Feature vector computation Next, the proposed method computes a feature vector for every key-point and then performs 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 [22].

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.

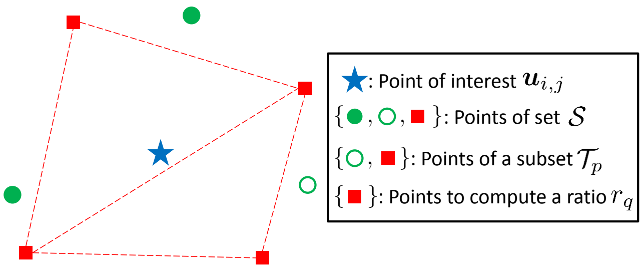

★: Point of interest $\boldsymbol{u}_{i,j}$
{●, ○, ■}: Points of set $\mathcal{S}$
{○, ■}: Points of a subset $\mathcal{T}_p$
{■}: Points to compute a ratio r_q

Figure 2: Sets of key-points to compute the feature vector for a key-point (an example of $n = 7, m = 5$).

To obtain better recognition ability, we assign multiple feature vectors for a key-point as shown in Fig. 2. For an extracted key-point $\boldsymbol{u}_{i,j}$, we make a set of n neighboring key-points $\mathcal{S} = \{\boldsymbol{u}_{i,j+1} \ldots \boldsymbol{u}_{i,j+n}\}$, $n = 7$ in Fig. 2. From the set $\mathcal{S}$, we make all combinations of subsets, one of which consists of m of n points, $\mathcal{T} = \{\mathcal{T}_1 \ldots \mathcal{T}_p \ldots \mathcal{T}_{{}_nC_m}\}$, $m = 5$ in Fig. 2. For a subset $\mathcal{T}_p$, we compute a feature vector $\mathcal{V}_p$. One dimension of $\mathcal{V}_p$ is the ratio of two triangles' area computed from 4 of m points. Therefore, the feature vector $\mathcal{V}_p$ is ${}_mC_4$ dimensional vector written as $\mathcal{V}_p = [r_{p,1} \ldots r_{p,q} \ldots r_{p,{}_mC_4}]^\top$ 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.

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 $\boldsymbol{C}_{\text{init},i} = \{\boldsymbol{X}_j \leftrightarrow \boldsymbol{u}_{i,j}\}$ denote a set of corresponding points obtained by the initial matching. Due to imperfectness of the feature descriptor, this matching result $\boldsymbol{C}_{\text{init},i}$ 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 $\boldsymbol{C}_{\text{RANSAC},i}$, 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 $\boldsymbol{C}_{\text{RANSAC},i}$ and reproject the known control points $\boldsymbol{X} = \{\boldsymbol{X}_1 \ldots \boldsymbol{X}_M\}$ onto the image. Then, the reprojected points $\boldsymbol{x}_i = \{\boldsymbol{x}_{i,1} \ldots \boldsymbol{x}_{i,M}\}$ are matched to its neighboring points. Furthermore, we assign a visibility term $v_{i,j}$ for each point. If j-th control point $\boldsymbol{X}_j$ has its corresponding point on image $\boldsymbol{I}_i$, the visibility term $v_{i,j}$ is set to 1, and 0 otherwise. Finally, we obtain a set of refined corresponding points $\boldsymbol{C}_i = \{\boldsymbol{X}_j \leftrightarrow \boldsymbol{x}_{i,j}\}$.

3.3 Single camera calibration

This step estimates the intrinsic parameter $(\boldsymbol{K}, \boldsymbol{\delta})$ of the target camera given the set of corresponding points $\{\boldsymbol{X}_j \leftrightarrow \boldsymbol{x}_{i,j}\}$ obtained in the previous step. The basic strategy of this step is based on Zhang's calibration method [26, 27] and the implementation is based on Bouguet's Camera Calibration Toolbox for MatLab [3].

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

$$v_i = \begin{cases} 1 & \text{if } \displaystyle\sum_{j=1}^{M} v_{i,j} \geq \tau_{\text{homo}} \\ 0 & \text{otherwise} \end{cases}, \tag{10}$$

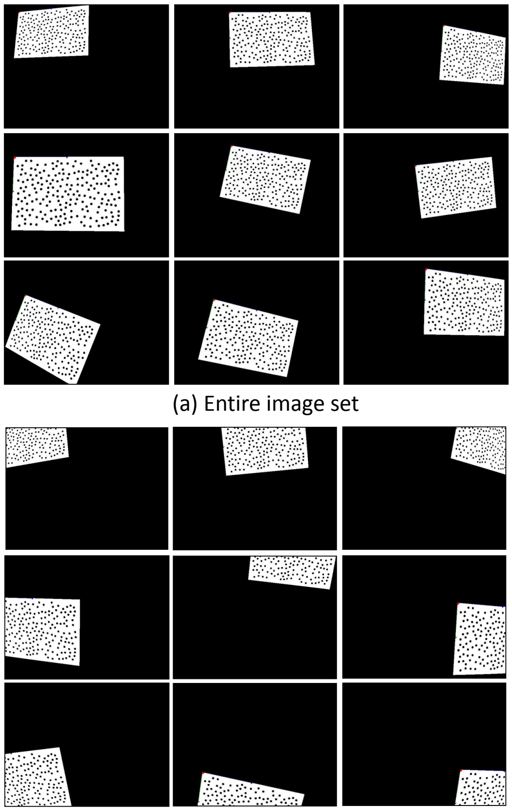
where the thresholding value τ_{homo} represents minimum number of points to compute a homography. The thresholding value τ_{homo} is theoretically 4, however we empirically assign 8. Using the homography availability term v_i, we can decompose a set $\mathcal{I} = \{1, \ldots, N\}$ into two subsets $\mathcal{I}_{\text{homo}}$ and its complement $\mathcal{I}^{\complement}_{\text{homo}}$. The subset $\mathcal{I}_{\text{homo}}$ contains indexes whose v_i is 1 while $\mathcal{I}^{\complement}_{\text{homo}}$ contains indexes whose v_i is 0.

From a set of the computed homographies $\{\boldsymbol{H}_1, \ldots, \boldsymbol{H}_N\}$, we estimate the intrinsic parameter $\boldsymbol{A}$ using orthogonality of vanishing points obtained from homographies [11]. Once the intrinsic parameters are estimated, we compute the extrinsic parameters $(\boldsymbol{R}_i, \boldsymbol{t}_i)$ given $\boldsymbol{A}$ and $\boldsymbol{H}_i$

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 $\boldsymbol{A}$ and $\{(\boldsymbol{R}_i, \boldsymbol{t}_i)\}$ 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 [15, 17] is performed to estimate/refine the all parameters by minimizing the following energy term:

$$\sum_{i \in \mathcal{I}_{\text{homo}}} \sum_{j=1}^{M} v_{i,j} \left\| \tilde{\boldsymbol{x}}_{i,j} - \text{Proj}(\tilde{\boldsymbol{X}}_j, \boldsymbol{A}, \boldsymbol{\delta}, \boldsymbol{R}_i, \boldsymbol{t}_i) \right\|^2 . \tag{11}$$

Note that this step uses input images categorized into the set $\mathcal{I}_{\text{homo}}$.



(a) Entire image set

(b) Partial image set

Figure 3: Two types of image sets for the synthetic experiment. (a) Entire image set on which entire marker is visible. (b) Partial image set on which part of marker is visible.

4 Experiments

The proposed method is validated by using both synthesized images and real world images. Synthesized images are used to evaluate the performance of the proposed method while real world images are to validate that the proposed method practically works.

4.1 Synthetic experiments

The purpose of this synthetic experiments is to validate the benefit of the proposed method that control points localization can be done with images on which only part of the calibration object is visible. As mentioned above, control points localized near the image boarder is necessary to achieve accurate camera parameters estimation, especially, lens distortion parameters. To validate this benefit, we use two types of image sets for camera calibration, some of those images are shown in Fig. 3. One set consists of images on which entire marker is visible (entire image set) while the other set consists of images on which part of marker is visible (partial image set).

We first perform the proposed camera calibration method only with images randomly chosen from entire image set. Since the en-

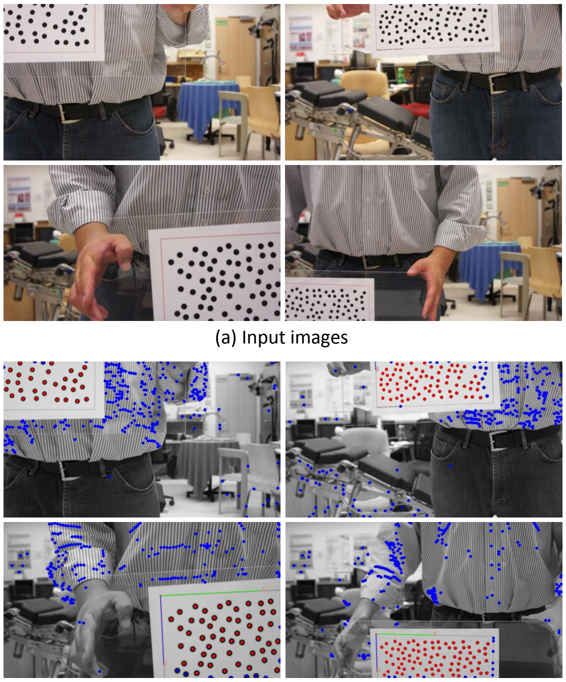
Table 2: Experimental results of synthetic experiment: numerical error of the estimated camera parameters comparing with the ground truth data. From left to right, number of added images chosen from partial image set is increased.

# added images		0	1	2	3	4	5	6	7	8	9	10
f_x	Mean	1.41	1.43	1.48	1.39	1.35	1.31	1.31	1.33	1.29	1.27	1.25
	σ	1.05	1.11	1.15	1.10	1.05	1.01	0.99	0.95	0.93	0.93	0.89
f_y	Mean	1.40	1.42	1.47	1.39	1.34	1.30	1.31	1.32	1.28	1.26	1.24
	σ	1.03	1.10	1.13	1.08	1.03	0.99	0.98	0.93	0.91	0.92	0.87
o_x $[\times 10^{-1}]$	Mean	5.06	5.17	5.24	5.03	4.91	4.94	4.89	4.96	4.79	4.53	4.60
	σ	3.86	3.74	3.93	3.64	3.53	3.53	3.13	3.01	3.02	2.99	2.98
o_y $[\times 10^{-1}]$	Mean	6.17	6.17	5.81	5.99	5.98	5.97	5.80	5.80	5.77	5.82	5.74
	σ	4.67	4.37	4.31	4.28	4.13	4.02	3.97	3.91	3.90	3.79	3.73
δ_1 $[\times 10^{-3}]$	Mean	1.14	1.13	1.05	1.00	0.98	0.95	0.93	0.93	0.92	0.86	0.92
	σ	0.89	0.86	0.77	0.76	0.71	0.71	0.64	0.64	0.61	0.63	0.62
δ_2 $[\times 10^{-3}]$	Mean	5.30	5.83	5.58	5.50	5.30	5.56	5.51	5.23	5.05	4.96	5.05
	σ	4.24	5.07	5.13	4.96	4.52	4.26	3.96	3.87	3.84	3.52	3.76
δ_3 $[\times 10^{-4}]$	Mean	1.11	1.11	1.05	1.10	1.08	1.12	1.15	1.08	1.06	1.09	1.09
	σ	0.79	0.76	0.80	0.85	0.89	0.92	0.95	0.96	0.93	0.88	0.85
δ_4 $[\times 10^{-4}]$	Mean	2.12	2.01	2.00	1.86	1.83	1.82	1.85	1.85	1.79	1.79	1.78
	σ	1.51	1.51	1.59	1.51	1.49	1.49	1.49	1.42	1.32	1.29	1.33

tire image set contains less control points near the image boarder as shown in Fig. 3 (a), the estimated camera parameters may be less accurate. Then, we choose images from partial image set and run the proposed method. Since the partial image set contains more control points around the image boarder as shown in Fig. 3 (b), the estimated camera parameters should get more accurate in proportion to number of added images. We randomly generates the combination of calibration images and compare the estimated parameters with the ground truth. Table 2 shows the average of 100 random trials w.r.t. the number of added images chosen from partial image set. As the result shows, both mean and standard deviation of numerical error is decreasing in proportion to number of added images. This result indicates that adding images of partially visible markers can contribute to increase the accuracy of the camera parameters estimation and to decrease the instability of the parameters estimation, especially the lens distortion parameters $\boldsymbol{\delta}$.

4.2 Real world experiments

Next, the proposed method is performed on real world images. In this experiment, we calibrate a Sony Nex-5, a mirrorless interchangeable-lens camera. For the calibration, we use 100 images, each of which has 1920 × 1080 resolution, from image sequences. Some of input images and tracking results are shown in Fig. 4. Detected key-points are drawn as circles: red circles represent the control points localized by RANDOM tracking algorithm while blue circles represent the remaining detected key-points. As the figure shows, the marker tracking algorithm well-detects the calibration object.


(a) Input images

(b) Detected key-points

Figure 4: Tracking results of real world experiments. Drawn circles are detected key-points: Red circles represent localized control points and blue circles represent the remaining key-points.

Since we don't have ground truth data of the real world experiment, the camera parameters estimation result is evaluated based on reprojection error. The reprojection error is 0.16 ± 0.15 pixels and maximum error is 1.30 pixels. Figure 5 plots the reprojection error of all the localized control points used for the calibration. From the plot, the reprojection error of almost control points are less than 0.5 pixel. Considering the image resolution, we can say that the proposed method can provide enough accurate calibration result.

5 Conclusion

This paper proposed a single camera calibration method that solves the control points localization of partially visible calibration object. To realize the localization in such situation, the proposed method integrate a planar marker tracking algorithm into camera calibration procedure. As the synthetic experimental results show, images with control points near the image boarder can contribute to increase the accuracy of the camera parameters estimation and to decrease the instability of the camera parameters estimation.

The potential application that the proposed calibration method can contribute is multiple cameras calibration to build multiple cameras based applications such as camera arrays [25] and multiview stereo reconstruction. For multiple cameras calibration, we have to consider overlap of cameras' view volume and points correspondence between cameras. When the cameras have less overlap area, it gets harder to take calibration images on which entire cal-

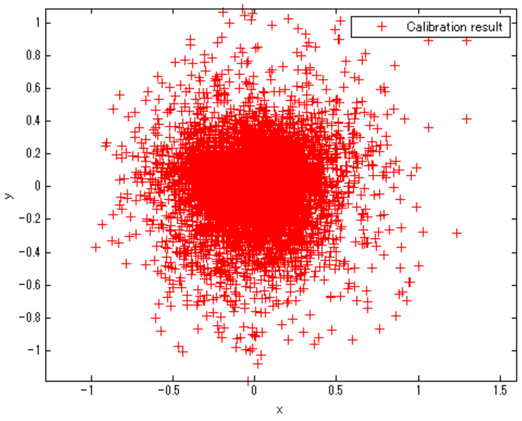


Figure 5: reprojection error with estimated camera parameters.

ibration object is visible. Since the proposed method can perform camera calibration with images of partially visible marker, the proposed method can be useful to calibrate multiple cameras under such condition.

There still remains things to do. Most important thing is control points localization. As Heikkilä mentioned in [12], the center of the circle is not invariant to perspective transform. This indicates that the localized control points $\boldsymbol{x}_{ij}$ may have point detection error. To this problem, we can detect the control points on canonical view not on normal camera view so that the distortion caused by perspective transform is canceled. Another to do is calibration image selection. As mentioned by Pilet *et al*. [19], calibration images on which the calibration object is either occluded or too slanted are not reliable because such images result ambiguous or singular homographies. The proposed method contains outlier rejection step in the control points localization process. However, adding such image selection process contributes to removing such unreliable calibration images. Furthermore, adding some geometrical constraints based on pinhole camera model, and optics is also interesting future direction.

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 renement 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] 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.

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

[15] 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.

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

[17] 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.

[18] 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.

[19] 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.

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

[21] 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.

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

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

[24] 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.

[25] B. Wilburn, N. Joshi, V. Vaish, M. Levoy, and M. Horowitz. High-speed videography using a dense camera array. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 294–301. IEEE Computer Society, 2004.

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

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