A Linear Generalized Camera Calibration from Three Intersecting Reference Planes

Mai Nishimura∗ Shohei Nobuhara Takashi Matsuyama
Graduate School of Informatics, Kyoto University, Japan
{nismura,nob,tm}@vision.kuee.kyoto-u.ac.jp

Shinya Shimizu Kensaku Fujii
NTT Media Intelligence Laboratories, NTT Corporation, Japan
{shimizu.shinya,fujii.kensaku}@lab.ntt.co.jp

Abstract

This paper presents a new generalized (or ray-pixel, raxel) camera calibration algorithm for camera systems involving distortions by unknown refraction and reflection processes. The key idea is use of intersections of calibration planes, while conventional methods utilized collinearity constraints of points on the planes. We show that intersections of calibration planes can realize a simple linear algorithm, and that our method can be applied to any ray-distributions while conventional methods require knowing the ray-distribution class in advance. Evaluations using synthesized and real datasets demonstrate the performance of our method quantitatively and qualitatively.

1. Introduction

3D analysis and motion measurement of underwater objects enable various applications such as discovering the process of fertilized egg development and kinematic analysis of fish motion, etc. Such applications are expected to contribute for bioinformatics or other industrial fields, however, there have been no general techniques which can handle complicated refraction, reflection, and attenuation of underwater environments. We therefore aim at establishing a new general technique and realizing multi-view camera system for underwater environment.

In computer vision, real cameras do not exactly follow ideal projection models such as perspective or orthographic projections, and captured images always show geometric distortions. For some special camera systems, such distortions can be modeled by displacements on the imaging plane[4, 1] in practice. However, such distortion models cannot fit with general cases such as distortions by catadioptric systems or refractive housings of unknown geometry as shown in Figure 1.

To handle such general cases involving unknown distortions, Grossberg and Nayer[3] have proposed the concept of generalized (or ray-pixel, raxel) camera model in which each pixel is associated with a 3D ray outside of the system as shown in Figure 2, and realized 3D ray modeling in the target space without modeling the refraction and/or reflection processes explicitly.

However, conventional methods for such 3D ray modeling require either calibration objects whose global 3D positions are given, or prior knowledge on the ray-distribution class (e.g. axial, central, etc.) of the system in order to switch the algorithm to be applied.

The goal of this paper is to propose a new linear calibration algorithm that can overcome these limitations, and the key idea on realizing such algorithm is use of intersections of calibration planes.

Our contribution is twofold: (1) a new linear calibration of generalized camera model that utilizes intersections...
of reference planes, and (2) a practical algorithm that detects intersections of reference planes from observed images. In what follows, we show that intersections of calibration planes can realize a distribution-independent formulation, and evaluate its performance in comparison with the state-of-the-art quantitatively and qualitatively with synthesized and real datasets.

2. Related Works

In order to model geometric distortions without explicitly modeling catadioptric systems or refractive housings of unknown geometry, Grossberg and Nayer have proposed the concept of generalized (or ray-pixel, raxel) camera model[3], and showed a calibration method using reference planes whose positions are given a priori. This idea is successfully applied for underwater photography to account for the refraction by housings or water tanks[9, 15, 2, 16]. However, these approaches require that the 3D geometry of reference planes in the target space, e.g., water, is given in a unified coordinate system, and they used extra markers exposed in the air or mechanical devices such as sliders or rotation tables.

Contrary to this approach, Ramalingam and Sturm have shown another approach utilizing three calibration planes of unknown postures [14, 11, 12, 10, 13]. These methods do not require such 3D geometry of calibration planes. Instead, they use collinearity constraints of 3D points projected to a same pixel in order to estimate the plane poses. Considering the distribution of rays in the target space, they showed that generalized cameras are classified into four classes (central, axial, crossed-slits, fully non-central), and proposed class-specific algorithms by assuming distribution-dependent degeneracies of the system. That is, they realized image-based calibrations, but such methods can be applied only to known ray-distributions.

Compared with these state-of-the-arts, our method does not require 3D geometry of the calibration objects and the knowledge on the ray-distribution class in advance. Similarly to them, our method also utilizes three reference planes of unknown postures, and estimates their postures from observed images in order to calibrate the rays in the target space.

The key difference is that our method utilizes not the collinearity but intersections of planes. That is, we estimate the relative postures of the planes by detecting the 3D locations on each plane where it intersects with another one. Since our method does not require the class-specific knowledge, our method can be applied to any unknown ray-distributions.

Besides, the state-of-the-arts[14, 11, 12, 10, 13] almost consist of linear steps but involve a non-linear process to seek a solution in null spaces inevitably. On the other hand, our method does not require such non-linear process and consists of a simple linear process.

3. Measurement Model

The generalized (or ray-pixel, raxel) camera model[3] is defined as a set of 3D rays in the target space where an object exists, and each of the rays is associated with a pixel of the camera imager (Figure 2).

Figure 2. Generalized (or ray-pixel, raxel) camera model[3]

Let \( \mathcal{X} \) denote the target space. We assume rays \( \mathbf{q} \) go straight in \( \mathcal{X} \), and there exists only a single ray in \( \mathcal{X} \) captured by each pixel by assuming a pinhole camera. The problem we address is to estimate the 3D geometry of such rays in \( \mathcal{X} \) associated with pixels by observing some calibration objects in \( \mathcal{X} \).

Suppose we capture a calibration plane under three unknown different postures \( \Phi_0, \Phi_1, \Phi_2 \) in \( \mathcal{X} \), and the calibration plane has feature points \( \mathbf{p} \) whose position on the plane local coordinate system is given.

Let \( \mathbf{p}^{[k]} = (u, v, 0)^\top \) denote a point on the calibration plane of the \( k \)th posture. By denoting the rigid motion between \( \Phi_k \) and \( \Phi_0 \) by

\[
R_k = \begin{pmatrix}
\tau_{k,1} & \tau_{k,2} & \tau_{k,3} \\
\tau_{k,4} & \tau_{k,5} & \tau_{k,6} \\
\tau_{k,7} & \tau_{k,8} & \tau_{k,9}
\end{pmatrix},
\quad t_k = \begin{pmatrix}
t_{k,1} \\
t_{k,2} \\
t_{k,3}
\end{pmatrix},
\tag{1}
\]

we can describe the points on \( \Phi_1 \) and \( \Phi_2 \) in the \( \Phi_0 \) local coordinate system as

\[
\begin{align*}
\mathbf{p}^{[0]} &= R_1 \mathbf{p}^{[1]} + \mathbf{t}_1, \quad \tag{2} \\
\mathbf{p}^{[0]} &= R_2 \mathbf{p}^{[2]} + \mathbf{t}_2, \quad \tag{3}
\end{align*}
\]

respectively.

4. Linear Generalized Camera Calibration using Three Intersecting Planes

Our calibration using three images of the calibration planes consists of three steps: (1) detection of the intersections of the planes, (2) estimation of the plane postures, and (3) ray generation using collinear points.

The last step is identical to the existing method[14]. That is, once recovered the plane postures in a single coordinate system, we can obtain the 3D ray corresponding to a pixel by detecting the points on the calibration planes such that they are projected to the pixel in question.
Figure 3. Special cases of three planes. (a) parallel to each other, (b) two parallel planes and the other cuts each in a line, (c) intersects in a line, (d) form a prismatic surface

Figure 4. Intersections of three reference planes

In this section, we first introduce the second step, the core of our theoretical contribution, and then introduce the first step with our practical apparatus.

4.1. Pose Estimation of Three Intersecting Planes

Suppose we can detect (1) pixels in the captured images where the calibration planes intersect, and (2) corresponding 3D points on the calibration planes projected to the pixel. The key constraint we use in this section is that if a pixel corresponds to the intersection of two calibration planes, the corresponding 3D points on the different planes are a coincident point in \( \mathcal{X} \).

By ignoring special cases illustrated in Figure 3, three planes always have intersections each other, and such lines intersect at a single point by definition (Figure 4). The goal of this section is to estimate \( R_1, R_2, t_1 \) and \( t_2 \) by using coincident points on such intersections. Notice that the special cases of Figure 3 can be detected by verifying the rank of \( M \) of Eq (9) automatically as described later.

Given two points on each of the three intersecting lines as shown in Figure 4, coincident points \( p_i^0, \ldots, p_i^5 \) on such intersections provide the following equations:

\[
\begin{align*}
p_i^0 &= R_1 p_i^1 + t_1, \quad (i = 0, 1), \\
p_i^0 &= R_2 p_i^2 + t_2, \quad (i = 2, 3), \\
R_1 p_i^1 + t_1 &= R_2 p_i^2 + t_2, \quad (i = 4, 5),
\end{align*}
\]

where \( p_0 \) and \( p_1 \) are on the intersection of \( \Phi_0 \) and \( \Phi_1 \), \( p_2 \) and \( p_3 \) are on that of \( \Phi_0 \) and \( \Phi_2 \), and \( p_4 \) and \( p_5 \) are on that of \( \Phi_1 \) and \( \Phi_2 \).

This provides 18 linear constraints on 18 parameters which consist of \( R_1, R_2, t_1, t_2 \) to be estimated except \( r_{k3}, r_{k6}, r_{k9} \) \((k = 1, 2)\) corresponding to the \( z \)-axis. Here, other corresponding points observed on the intersection lines do not provide additional constraints mathematically. However, they can contribute to make the system robust to noise.

The above constraints do not enforce the rotation matrices be \( SO(3) \). Consequently, Eq (4) . . . Eq (6) do not give a unique solution and the rank of the above system always becomes 15. We therefore introduce additional constraints based on inner products as follows.

\[
\begin{align*}
P_0^1 p_1^1 &= p_0^1 p_1^0 R_1 p_1^3 p_1^1, \\
P_2^1 p_3^1 &= p_2^1 p_3^0 R_2 p_3^2 p_3^1.
\end{align*}
\]

where \( p_i^k = p_i^k - p_i^0 \). The key point is that these are defined as linear constraints by utilizing inner products given in different local coordinates of the planes. By adding these two constraints, we have 20 linear equations that form a linear system \( Mx = b \) of rank 17 for 18 parameters, where

\[
M = \begin{pmatrix}
p_0^0 \otimes I_{3 \times 3} & 0_{3 \times 9} \\
p_1^0 \otimes I_{3 \times 3} & 0_{3 \times 9} \\
p_2^0 \otimes I_{3 \times 3} & 0_{3 \times 9} \\
p_3^0 \otimes I_{3 \times 3} & 0_{3 \times 9} \\
p_4^0 \otimes I_{3 \times 3} & 0_{3 \times 9} \\
p_5^0 \otimes I_{3 \times 3} & 0_{3 \times 9} \\
p_0^1 p_1^1 & 0_{1 \times 9} \\
p_1^1 p_2^1 & 0_{1 \times 9} \\
p_2^1 p_3^1 & 0_{1 \times 9}
\end{pmatrix}
\]

and

\[
x = \begin{pmatrix}
r_{11} \\
r_{14} \\
r_{17} \\
r_{12} \\
r_{15} \\
r_{18} \\
r_{11} \\
r_{14} \\
r_{17} \\
r_{18} \\
r_{21} \\
r_{24} \\
r_{27} \\
r_{22} \\
r_{25} \\
r_{28} \\
r_{21} \\
r_{24} \\
r_{27} \\
r_{28} \\
r_{31} \\
r_{34} \\
r_{37} \\
r_{32} \\
r_{35} \\
r_{38} \\
r_{31} \\
r_{34} \\
r_{37} \\
r_{38}
\end{pmatrix}^T, \quad b = \begin{pmatrix}
p_0^0 \\
p_1^0 \\
p_2^0 \\
p_3^0 \\
p_0^1 p_1^1 \\
p_1^1 p_2^1 \\
p_2^1 p_3^1 \\
p_0^1 p_1^1 \\
p_1^1 p_2^1 \\
p_2^1 p_3^1 \\
p_0^1 p_1^1 \\
p_1^1 p_2^1 \\
p_2^1 p_3^1 \\
p_0^1 p_1^1 \\
p_1^1 p_2^1 \\
p_2^1 p_3^1 \\
p_0^1 p_1^1 \\
p_1^1 p_2^1 \\
p_2^1 p_3^1 \\
p_0^1 p_1^1 \\
p_1^1 p_2^1 \\
p_2^1 p_3^1
\end{pmatrix}^T.
\]

Here \( I_{n \times n} \), \( I_{2 \times 3} \), and \( 0_{n \times m} \) denote the \( n \times n \) identity matrix, \( \begin{pmatrix} 0 & 0 \end{pmatrix} \), and the \( n \times m \) zero matrix respectively. \( p_i^k = p_i^k - p_i^0 \) denotes the point on \( \Phi_k \), and \( p_i^k = (u_i, v_i, 1)^T \). \( \otimes \) denotes Kronecker product. Besides the rank of \( M \) indicates if the planes are in a special configuration (Figure 3) or not, since such special configurations introduces linearly-dependent equations.

This linear system \( Mx = b \) can be further decomposed...
into two linear systems $M_1 x_1 = b_1$ and $M_2 x_2 = b_2$, where

$$\begin{align*}
M_1 &= \begin{pmatrix}
    p_0^{[1]} \otimes I_{2 \times 2} & 0_{2 \times 6} \\
    p_1^{[1]} \otimes I_{2 \times 2} & 0_{2 \times 6} \\
    0_{2 \times 6} & 0_{2 \times 6} \\
    p_0^{[2]} \otimes I_{2 \times 2} & p_2^{[2]} \otimes I_{2 \times 2} \\
    p_1^{[2]} \otimes I_{2 \times 2} & p_2^{[2]} \otimes I_{2 \times 2} \\
    p_0^{[2]} \otimes I_{2 \times 3} & p_2^{[2]} \otimes I_{2 \times 3} \\
    p_1^{[2]} \otimes I_{2 \times 3} & p_2^{[2]} \otimes I_{2 \times 3} \\
    0_{1 \times 6} & 0_{1 \times 6}
\end{pmatrix},
\end{align*}$$

and

$$\begin{align*}
M_2 &= \begin{pmatrix}
    p_0^{[1]} \otimes I_{1 \times 3} & 0_{1 \times 3} \\
    p_1^{[1]} \otimes I_{1 \times 3} & 0_{1 \times 3} \\
    0_{1 \times 3} & 0_{1 \times 3} \\
    p_2^{[1]} \otimes I_{1 \times 3} & p_3^{[1]} \otimes I_{1 \times 3} \\
    -p_1^{[1]} \otimes I_{1 \times 3} & -p_3^{[1]} \otimes I_{1 \times 3} \\
    p_0^{[2]} \otimes I_{1 \times 3} & p_3^{[2]} \otimes I_{1 \times 3} \\
    p_1^{[2]} \otimes I_{1 \times 3} & p_3^{[2]} \otimes I_{1 \times 3} \\
    0_{1 \times 1} & 0_{1 \times 1} \\
    p_0^{[3]} \otimes I_{1 \times 3} & p_3^{[3]} \otimes I_{1 \times 3} \\
    p_1^{[3]} \otimes I_{1 \times 3} & p_3^{[3]} \otimes I_{1 \times 3} \\
    0_{1 \times 1} & 0_{1 \times 1}
\end{pmatrix}.
\end{align*}$$

The solution $x_1$ is given as $x_1 = (M_1^T M_1)^{-1} M_1^T b_1$, and $x_2$ is given up to a scale factor $\alpha$ using the 6th right singular vector $v_0$ of $M_2$ corresponding to the null space of $M_2$:

$$x_2 = \alpha v_0.$$

This $\alpha$ can be trivially determined by using the orthogonality constraints on $R_1$ and $R_2$:

$$r_{k,1}^T r_{k,2} = 0, \quad |r_{k,1}| = 1, \quad |r_{k,2}| = 1.$$

Up to this point, we obtain the first and the second column vectors of $R_1$ and $R_2$ as well as $t_1$ and $t_2$.

Finally, by recovering the third columns of $R_1$ and $R_2$ as the cross products of each of the first and the second column vectors, we obtain the complete $R_1$, $R_2$, $t_1$ and $t_2$ up to scale.

Notice that the above linear system allows a mirrored solution such that $R_k = \begin{pmatrix} r_{k,1} & r_{k,2} & r_{k,3} \\ r_{k,2} & r_{k,3} & -r_{k,1} \\ r_{k,1} & -r_{k,3} & -r_{k,2} \end{pmatrix}$ and $t_k = \begin{pmatrix} t_{k,1} \\ t_{k,2} \\ t_{k,3} \end{pmatrix}$. Selecting the correct solution can be done by verifying the direction of the z-axis.

4.2. Intersection Detection from Distorted Images

The crucial point to realize our method is (1) how to obtain the 2D position on the calibration plane only from the captured images under unknown distortions, and (2) how to detect intersections of the calibration planes under different postures from their images.

To this end, we employ a flat panel display as the calibration plane $\Phi$, and utilize the gray code technique proposed in [3] (Figure 5). That is, we fix the display, and capture gray code patterns shown by it for each pose.

With this gray code technique, the first point is simply achieved by decoding the gray code representing the display pixel location as done by [3]. To achieve the second point, we propose a new method based on the difference of display pixel densities between captured images.

Suppose we have established display pixel to camera pixel correspondences for $\Phi_k$ by decoding the gray code images, and the correspondence is a one-to-many mapping where each display pixel is captured by multiple camera pixels. This assumption can be realized by making the camera to have effectively a higher resolution than the display by binning each $n \times n$ display pixels into a block.

Let $f_k(q)$ denote the display pixel on $\Phi_k$ corresponding to a camera pixel $q$. Once obtained such one-to-many correspondences between display and camera pixels, we can obtain a display density $d_k(q)$ of each camera pixel $q$ by counting the camera pixels sharing the same display pixel $f_k(q)$ (Figure 10).

The key idea on detecting the intersection is to use the difference of $d_k(q)$ and $d_{k'}(q)$ of $\Phi_k$ and $\Phi_{k'}$. Obviously $d_k(q)$ and $d_{k'}(q)$ are not equal or proportional to the real depth since it is affected by both the depth and the refraction/reflection process. However, if the values $d_k(q)$ and $d_{k'}(q)$ are the same, it is guaranteed that the corresponding 3D points are at a same depth as long as the refraction/reflection process is kept static. That is, if $d_k(q) = d_{k'}(q)$, then the corresponding 3D points $f_k(q)$ and $f_{k'}(q)$ on different planes $\Phi_k$ and $\Phi_{k'}$ are a coincident point and hence are on the intersection of the planes.

Based on this idea, we design our intersection detection as follows: (1) establish the display-camera pixel correspondences by decoding the gray code[3], (2) compute the density map $d_k(q)$, (3) find the pixels such that $d_k(q) = d_{k'}(q)$ for each pair of planes $\Phi_k$ and $\Phi_{k'}$ and return the corresponding $f_k(q)$ and $f_{k'}(q)$ as 3D points on the intersection. Section 5.3.2 shows how this process works in our real underwater environment.
5. Evaluations

In order to evaluate the performance of our method, this section first evaluates the performance quantitatively using synthesized datasets. Then we introduce our apparatus for an underwater environment and show a qualitative evaluation using real images with a 3D shape reconstruction in water.

5.1. Error Metrics

Rotation and Translation Errors In case that the ground truth is available, we evaluate the quality of the calibration by estimation errors of \( R \) and \( t \) w.r.t. the ground truth since the ray directions are defined by the plane poses. The estimation error of \( R \) is defined as the Riemannian distance\[8\]. Let \( R_g, t_g \) be the ground truth of \( R, t \), and \( \theta = \cos^{-1}\left(\frac{trR - 1}{2}\right) \). The rotation matrix error \( E_R \) is defined as

\[
E_R = \frac{1}{\sqrt{2}} \| \log(R^T R_g) \|_F, \tag{16}
\]

\[
\log R' = \begin{cases} 0 & (\theta = 0), \\ \frac{\theta}{2\sin(\theta)}(R - R^T) & (\theta \neq 0), \end{cases} \tag{17}
\]

where \( \| \cdot \|_F \) denotes the Frobenius norm.

The translation vector error \( E_T \) is defined by the root mean square error (RMSE)

\[
E_T = \sqrt{\|t - t_g\|^2/3}. \tag{18}
\]

Re-Intersection Error Similarly to the reprojection error widely used in the perspective multi-view calibration, we can consider the re-intersection error that measures the distance between the observed and the synthesized points on the calibration planes.

By denoting the observed point on \( k \)th plane for camera pixel \( i \) by \( p_{k,i} \), and the intersection of the calibration plane and the corresponding ray by \( \tilde{p}_{k,i} \), we can define the re-intersection error as

\[
E_p = \frac{1}{|I||K|} \sum_{i \in I} \sum_{k \in K} ||\tilde{p}_{k,i} - p_{k,i}||^2, \tag{19}
\]

where \( I \) and \( K \) are the set of the pixels and the calibration planes, and \( |I| \) and \( |K| \) denotes the number of their elements respectively.

5.2. Evaluation using Synthesized Data

To evaluate the performance quantitatively, we used synthesized datasets which simulate our real setup shown in Figure 8. We set the camera and the display resolutions as \( 1280 \times 960 \) and \( 1024 \times 2048 \) respectively. The first display defines the world coordinate system, and the sizes of the other devices are described in the display pixel. The camera is placed at 2662.4 pixels from the center of the water cylinder of radius 2048 pixels. Notice that the refractive index of water is set to 1.3, and this virtual cylinder has no housings so that the rays coming from the air directly refracted at the boundary between water and the air.

For each trial, we synthesized 3 different display postures \( \Phi_0, \Phi_1, \Phi_2 \), and applied our calibration. Here the rotation matrices are generated using uniformly sampled random three Euler angles within \([-\pi/3 : \pi/3]\). The elements of the translation vectors are generated using uniformly sampled random values within \([-50 : 50]\) pixels.

Figure 6 shows the results. Each point is the mean value of 100 trials at each noise level. The red and blue plots indicate the errors estimated with 12 and 60 points (2 and 10 points per intersection per plane, respectively). The magenta indicates the state-of-the-art\[11\] with 60 points (20 points per plane \( \Phi \)). In this evaluation, \[11\] sampled points regularly on the image and ours sampled 2 or 5 points on each intersection as done in Figure 6\[1\].

1Our implementation is available at http://vision.kuee.kyoto-u.ac.jp/~nob/proj/iccv2015/.
Figure 7. Plots of $E_R$, $E_T$, and $E_P$. The cross, circle and triangle markers indicate the errors of the parameters of $\Phi_0$, $\Phi_1$ and $\Phi_2$ respectively.

From these results, we can observe that our method shows more robust performance than the state-of-the-art, while both methods return the exact results $E_R = E_T = 0$ at $\sigma = 0$.

**Bundle Adjustment using Collinearity Constraints** Using the linear calibration result as inputs, we applied the bundle adjustment using the collinearity constraint as done in the conventional methods[14]. That is, our method first utilizes the intersections to obtain the linear solution, and then the solution can be refined non-linearly using the collinearity constraint on other points, while the collinearity constraint itself do not provide a linear solution due to the ray-distribution dependent unknown degeneracy[12].

Figure 7 shows the average errors of 100 trials under different noise levels, where the red plots data in Figure 6 are used as initial inputs. From these results, we can conclude that our method can provide reasonable initial values for the bundle adjustment, and the calibration errors are within a reasonable range in comparison with the injected noise.

### 5.3. Evaluation using Real Images

#### 5.3.1 Apparatus for Underwater Environment

Figure 8 shows our setup. We used 4K cameras (Pointgrey FL3-U3-88S2C-C) to capture an object in an acrylic cylinder water tank of 300mm diameter and 5mm thick. The distance from the camera to the cylinder center was approximately 350mm. To calibrate the mapping between the camera pixels and corresponding rays in water, the cameras capture a waterproof tablet of $1366 \times 768$ resolution showing the gray code[3] as shown in Figure 5.

In this experiment, we exposed approximately the upper-half of the display in the air in order to obtain the reference calibration parameters using a baseline method[17] as shown as the green-boxed areas in Figure 9 (left). Notice that our calibration used only the lower-half of the display in water as shown as the cyan-boxed areas in Figure 9 (right).

#### 5.3.2 Intersection Estimation

As described in Section 4.2, we performed intersection detection by creating the pixel density map. By computing camera pixels corresponding to $8 \times 8$ display pixel bins in the tablet from the captured images, we obtained pixel density maps as shown in Figure 10 (right).

To evaluate the quality of our intersection estimation, we compare our result with intersections calculated from the baseline method. Figure 11 shows the estimated intersections decoded onto each tablet of different postures. Blue to red color plots (*) indicate pixel-depth maps decoded to the display coordinate system, and Red and blue plots (o) indicate estimated intersections on each display, $\Phi_0$ (left) and $\Phi_1$ (right). Black lines (-) indicate the intersections of each
display estimated by the baseline method. While the estimated intersections are not exactly coincident with the ones from the baseline method, and parallel to each other (Figure 11 left), or intersect each other (Figure 11 right), these could serve as a good enough estimation of intersections as shown in the next result.

### 5.3.3 Linear Calibration from Estimated Intersections

Figure 13 (left) shows the pose estimation result by the baseline method, and Figure 13 (right) shows the result by our proposed method. In Figure 12, black lines (-) indicate intersections calculated by the baseline method. Red, green, and blue markers (×) indicate detected intersections from pixel density maps, and magenta lines (-) are intersections calculated by our proposed method. In our experiment, pixel density map-based intersection detection on the image plane did not perform perfectly in some images as shown in Φ₂ case in Figure 12 (right). This intersection-detection errors appeared as z-axial posture errors in (Figure 13 right).

Even so, these estimated pose parameters can be used as an initial values, and refined in the following multi-view calibration step with a global bundle adjustment.

### 5.3.4 Multi-view Calibration and Underwater 3D Shape Capture

Figure 14 shows the calibration results of one of the cameras after the nonlinear refinement using 3 or 5 cameras, where 5 cameras form a closed loop around the cylinder tank while 3 cameras do not. Figures 15 and 16 show distributions of $E_p$ at each display point on the first plane and the estimated rays in the cases of the 3 and 5 cameras together with the baseline result. In Figure 16, the dots illustrate the intersections with the planes and their color indicate the re-intersection error $E_p$. The average re-intersection error over
the all points of the all planes were 0.21 and 1.59 pixels for the 3 cameras and the 5 cameras setups while the average reprojection error for the baseline method as 0.88 pixels on the camera screen.

From these results, we can observe that the 5 camera case shows a comparable result to the baseline method while the 3 camera case results in a smaller re-intersection error but much different 3D plane postures. This fact indicates that the ray-pixel modeling can produce an overfitted result to the observation because of its flexibility, and multi-view generalized (or ray-pixel, raxel) camera calibration with a close loop can balance such flexibility and the robustness.

3D Shape Reconstruction To evaluate the calibration accuracy qualitatively, we captured a goldfish in the tank as shown in Figure 1, and applied the shape-from-silhouette method[5] to reconstruct the 3D shape. Figure 17 shows the input images taken by the camera 1 to 5 and the reconstructed 3D shape. This result demonstrates that our calibration can serve as a practical technique to realize a multi-view generalized (or ray-pixel, raxel) camera system for 3D measurement under unknown refractions.

6. Conclusion

In this paper, we proposed a new linear generalized (or ray-pixel or raxel) camera calibration algorithm. The key idea was to use intersections of the reference planes and to build a linear system utilizing coincident points on the intersections. Our method first utilizes the ray-independent intersections found where the collinearity constraint degenerates, and then our linear solution can be refined non-linearly using the collinearity constraint found other points on the plane as done in the conventional studies. Also we proposed a new practical method to detect such intersections from images with unknown distortions by exploiting the difference of pixel-density maps.

Compared with the state-of-the-arts[14, 11, 12, 10, 13] that involve non-linear solution finding processes in null spaces of their systems, our method requires only solving a simple linear system and can be applied without knowing the ray-distribution class in advance, since our method does not require adding class-specific constraints to solve the problem.

Limitations Our model does not require any regularizers such as continuities between 3D rays associated with neighboring pixels. This design maximizes its ability to model complex refraction / reflection processes, but such flexibility can also introduce unstable or erroneous results in the bundle adjustment utilizing the collinearity constraint as discussed in the evaluation. Hence integrating smoothness constraints between rays and/or pixels[7, 6] remains to be investigated.

Acknowledgement

This research is partially supported by NTT Media Intelligence Laboratories and by JSPS Kakenhi Grant Number 26240023.

References


