Shape-coded ArUco: Fiducial Marker for Bridging 2D and 3D Modalities

Lilika Makabe Hiroaki Santo Fumio Okura Yasuyuki Matsushita
Graduate School of Information Science and Technology, Osaka University
{makabe.lilika,santo.hiroaki,okura,yasumat}@ist.osaka-u.ac.jp

Abstract

We introduce a fiducial marker for the registration of two-dimensional (2D) images and untextured three-dimensional (3D) shapes that are recorded by commodity laser scanners. Specifically, we design a 3D-version of the ArUco marker that retains exactly the same appearance as its 2D counterpart from any viewpoint above the marker but contains shape information. The shape-coded ArUco can naturally work with off-the-shelf ArUco marker detectors in the 2D image domain. For the 3D domain, we develop a method for detecting the marker in an untextured 3D point cloud. Experiments demonstrate accurate 2D-3D registration using our shape-coded ArUco markers in comparison to baseline methods.

1. Introduction

Registration of two-dimensional (2D) images and three-dimensional (3D) shapes is an essential problem in computer vision. While 3D scanners are becoming widely available, many high-resolution laser scanners do not record texture information[1]. For texturing the untextured 3D shapes, accurate registration of 2D images and 3D shapes is needed. However, in general, such cross-modality registration is difficult due to the lack of features that are common in both modalities, *i.e.*, images and point clouds.

In this work, instead of estimating the 2D-3D correspondences, we propose to actively *define* feature points that both 2D and 3D sensors can reliably detect. To achieve this goal, we propose a *shape-coded ArUco* marker whose 2D appearance is exactly the same as ordinary 2D ArUco from any viewpoint but contains shape information that can be detected by 3D sensors (Fig. 1). The shape-coded ArUco thus can naturally work with off-the-shelf 2D ArUco detectors in the 2D image domain. For the 3D domain, we develop a method for detecting and localizing the markers in a recorded 3D point cloud based on a hybrid approach of point cloud processing and 2D marker detection.

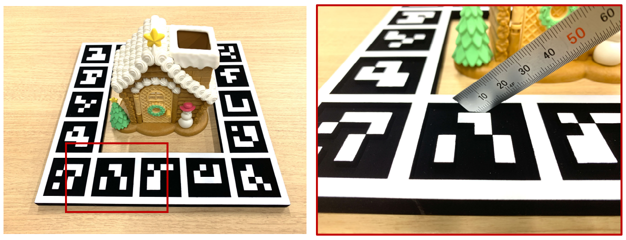


Figure 1: Our *shape-coded* markers give accurate 2D-3D correspondences by their embossed shape without changing the appearance on 2D images. Our method can be extended to any 2D fiducial marker.

To ensure the property of retaining the same appearance as the 2D ArUco markers from any viewpoint, we study the possible 3D deformations of the original 2D ArUco markers. We show that the *embossing* deformation satisfies the property, which can be used to encode the shape information to the marker, as shown in Figure 1.

We fabricated the shape-coded ArUco and assessed its effectiveness using both synthetic and real-world datasets. Experimental results show a higher registration accuracy in comparison to the state-of-the-art method that is based on a 2D planar marker board [19]. We provide 3D models, fabrication instruction, and the marker detector code for the shape-coded ArUco in the project page[2].

Contributions. The chief contributions of this work are:

- We introduce a practical fiducial marker for bridging 2D and 3D modalities, resulting in highly accurate 2D-3D registration for mapping colors on untextured 3D models.
- We study the class of deformations that retains the same 2D appearance of the original 2D markers while having the shape information that can be decoded by 3D scanners.

[1] *E.g.*, HandySCAN 3D, https://www.creaform3d.com/en/portable-3d-scanner-handyscan-3d, last accessed on August 17, 2021.

[2] https://github.com/lilika-makabe/shape-coded-aruco

- We demonstrate the effectiveness of the shape-coded ArUco markers in the application of color mapping on untextured 3D scans.

2. Related work

Our goal is to develop a fiducial marker for 2D-3D registration tasks, where texture information is unavailable in 3D data. We first briefly recap common methods relying on the texture information on 3D shapes, and then categorize the 2D-3D registration methods designed for untextured 3D models that work with/without fiducial markers.

When the texture information on 3D shapes is available, the 2D-3D registration problem is commonly achieved by finding the 2D-3D correspondences based on the texture. A straightforward approach is to solve the perspective-n-point (PnP) problem from the given correspondences [15] and is successfully applied for large-scale objects [11]. Some recent methods use deep neural networks to infer the 6 degrees-of-freedom (DoF) object poses from a textured 3D shape and image observations [13, 39, 41, 27, 9].

2D-3D registration for *untextured* 3D shapes is a harder problem because no obvious common feature points are available. The previous works for this problem are categorized into marker-less and marker-based methods.

Marker-less methods. Since the marker is not available in this setting, previous studies aim at identifying features from the scene that are geometrically consistent. One of the major approaches is to assume the correlation between 2D texture and 3D shape features, such as silhouette and boundaries [21, 23]. Point correspondences [28, 30] and line/plane correspondences [1, 40, 17, 36] are also known useful particularly for urban scenes. Gong *et al.* [7] use a trihedron in the scene while it requires manual intervention. A recent study [18] uses a deep neural network to overcome the difficulty of associating cross-modal feature descriptors. Instead of explicitly finding 2D-3D correspondences, their method turns the problem into a classification problem about whether the part of the point cloud is visible or not from the camera. While the marker-less methods have a merit of not requiring an explicit marker in the scene, their accuracy is inferior to the marker-based methods [24].

Marker-based methods. The problem of 2D-3D registration becomes easier when a marker is available. One of the major approaches is to use planar fiducial markers. Among them, most methods use the boundaries of the marker board as the 3D cues, whose relative positions to the 2D markers are known. For example, Geiger *et al.* [6] propose to use checkerboard markers printed on a plane. While most of the traditional methods [26, 29] require manual selection of the four corner points of the board from 3D scans, the state-of-the-art method proposed by Zhou *et al.* [42] automates the pipeline by detecting the 3D boundaries of the marker board and estimates the camera poses by line-to-line correspondences. Some other studies use tailored reference objects, such as holes [12, 4], spherical target [37, 14], or retroreflective target [10]. Another example of the 3D markers is a cube with marker faces, which are used in augmented reality applications [38]. Along the line of these works, we propose a simple yet effective shape-coded ArUco marker, which explicitly provides a 2D-3D correspondence at every feature point.

3. Shape-coded marker for 2D-3D registration

We design a 3D fiducial marker for 2D-3D registration that retains the same appearance as the original 2D marker. To be detectable by 3D scanners, we deform the original 2D marker to encode the shape information. We first discuss the required conditions for the deformations that preserve the same appearance as the 2D markers from any viewpoint. Hereafter, we call the property a *projective invariant appearance* property. We then design a shape-coded version of ArUco that satisfies the conditions. A similar shape coding can be applied to any other types of markers that are akin to ArUco markers.

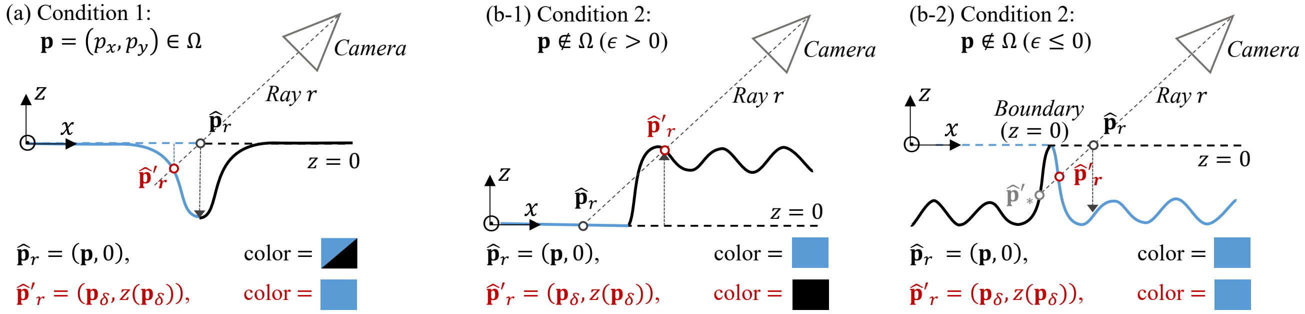
3.1. Required conditions for marker deformation

Let us assume a planar marker composed of two distinct colors $\{c_1, c_2\}$. As illustrated in Fig. 2, we use the 3D coordinate system, where the original 2D marker plane locates on $z = 0$ and assume that the marker is viewed from any location in $z > 0$. For a 2D point coordinate on the marker $\mathbf{p} = (p_x, p_y)$, we denote its 3D position using $\hat{}$ as $\hat{\mathbf{p}} = (\mathbf{p}, 0)$. For simplicity of fabrication, we only consider the deformation of the marker surface along the z-axis. Thus, the deformation is restricted to a morphism from the original marker point $\hat{\mathbf{p}} = (\mathbf{p}, 0)$ to the deformed point $\hat{\mathbf{p}}' = (\mathbf{p}, z(\mathbf{p}))$, where $z(\cdot)$ denotes the amount of deformation along z axis for the given 2D point. We further assume that the 3D shape of the deformed marker is piece-wise smooth, and cast/attached shadows do not alter the binary colors c_1 and c_2.

We use r to represent a camera ray that passes through the viewpoint and the point on the marker. Suppose that a camera ray r intersects the original 2D marker at point $\hat{\mathbf{p}}$ and that the same camera ray r intersects the deformed 3D marker at point $\hat{\mathbf{p}}'$. The marker has a projective invariant appearance if and only if, for any camera ray from any viewpoint at $z > 0$, the color of the deformed marker is the same as the original 2D marker, *i.e.*,

$$c\left(\hat{\mathbf{p}}, r\right) = c\left(\hat{\mathbf{p}}', r\right), \tag{1}$$

where $c(\cdot)$ returns the color $\{c_1, c_2\}$ at the given 3D point and camera ray.

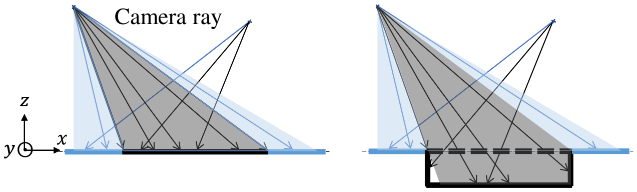


(a) Condition 1:
$\mathbf{p} = (p_x, p_y) \in \Omega$
Camera
Ray r
z
x
$\hat{\mathbf{p}}_r$
$\hat{\mathbf{p}}'_r$
$z = 0$
$\hat{\mathbf{p}}_r = (\mathbf{p}, 0)$,
color = ◩
$\hat{\mathbf{p}}'_r = (\mathbf{p}_\delta, z(\mathbf{p}_\delta))$,
color = 🟦
(b-1) Condition 2:
$\mathbf{p} \notin \Omega \ (\epsilon > 0)$
Camera
Ray r
$\hat{\mathbf{p}}'_r$
z
x
$\hat{\mathbf{p}}_r$
$z = 0$
$\hat{\mathbf{p}}_r = (\mathbf{p}, 0)$,
color = 🟦
$\hat{\mathbf{p}}'_r = (\mathbf{p}_\delta, z(\mathbf{p}_\delta))$,
color = ■
(b-2) Condition 2:
$\mathbf{p} \notin \Omega \ (\epsilon \leq 0)$
Camera
Ray r
z
x
Boundary
$(z = 0)$
$\hat{\mathbf{p}}_r$
$z = 0$
$\hat{\mathbf{p}}'_*$
$\hat{\mathbf{p}}'_r$
$\hat{\mathbf{p}}_r = (\mathbf{p}, 0)$,
color = 🟦
$\hat{\mathbf{p}}'_r = (\mathbf{p}_\delta, z(\mathbf{p}_\delta))$,
color = 🟦

Boundary points must be located at z = 0.

Non-boundary points must be located at z ≤ 0 (*i.e., embossed* deformation is allowed).

Figure 2: Required conditions to deform planar markers while preserving the projective invariant appearances.



(a) 2D planar marker

(b) Shape-coded marker

Figure 3: Side-view illustrations of (a) 2D planar marker and (b) our shape-coded marker. White patterns in an original 2D marker are on $z = 0$ plane, and black patterns are *embossed* to be distinguishable from 3D scanners. Off-the-shelf 2D marker detectors work for our marker because our marker has projective invariant appearances.

Now we describe the conditions of deformations that preserve the property of projective invariant appearance. We denote a set of the 2D marker points on the boundary of two distinct colors $\Omega \subset \mathbb{R}^2$ and discuss the conditions for each of 2D marker point $\mathbf{p} \in \Omega$ and $\mathbf{p} \notin \Omega$.

Condition 1 **($\mathbf{p} \in \Omega$)** *Marker points on the boundary of two colors must be located at the original marker plane, $z(\mathbf{p}) = 0$.*

Suppose a 2D marker point $\mathbf{p}$ is on the color boundary. If the original marker point $\hat{\mathbf{p}}$ moves along z axis as illustrated in Fig. 2(a), the camera ray r passes through a different marker point $\hat{\mathbf{p}}'$ on the deformed surface. Since the moved point is no longer on the color boundary after deformations, it indicates that no deformations are allowed for points on the color boundaries.

Condition 2 **($\mathbf{p} \notin \Omega$)** *Marker points off the boundary of two colors must be located on or beneath the original marker plane, $z(\mathbf{p}) \leq 0$.*

If the deformation toward $z(\mathbf{p}) > 0$ is allowed (see Fig. 2(b-1)), the color of the original 2D marker point can vary due to occlusions introduced by the deformation, resulting in $c(\hat{\mathbf{p}}, r) \neq c(\hat{\mathbf{p}}', r)$ that breaks the condition of Eq. (1). Therefore, the projective invariant appearance cannot be preserved for deformations toward $z(\mathbf{p}) > 0$.

On the other hand, deformation toward $z(\mathbf{p}) < 0$ is allowed because the same color can always be observed even after deformations as illustrated in Fig. 2(b-2), maintaining the projective invariant appearance property. It implies that the embossing deformation is always allowed for marker points that are not on the color boundaries.

3.2. Designing shape-coded ArUco marker

Based on the deformation constraints described in Sec. 3.1, we fabricated a shape-coded ArUco marker [5] by deforming the original 2D ArUco . In real-world situations, the top surface casts shadows on the embossed area that may cause undesirable color variations. To avoid the issue, we place white regions as the top surface and black regions embossed because the shadowing effect does not affect the black appearance. To ease fabrication and detection, both the top surface (*i.e.*, $z = 0$) and embossed part (*i.e.*, $z = -\epsilon$) are made planar, where we set ϵ large enough to be distinguishable by commercially available 3D scanners. All surfaces except the top surface are colored black, as visualized in Fig. 3 so that the fabricated marker meets the above conditions.

For making our marker practical for 2D-3D registration, we fabricate a marker board where a set of markers placed surrounding the target object. Figure 1 shows the appearance of the shape-coded ArUco board as well as a target object located in the middle of the marker board.

4. 2D-3D registration with shape-coded ArUco

In this section, we describe the pipeline of 2D-3D registration by our shape-coded ArUco marker. Since our markers have projective invariant appearances, we can directly use off-the-shelf 2D marker detectors (*e.g.*, implemented

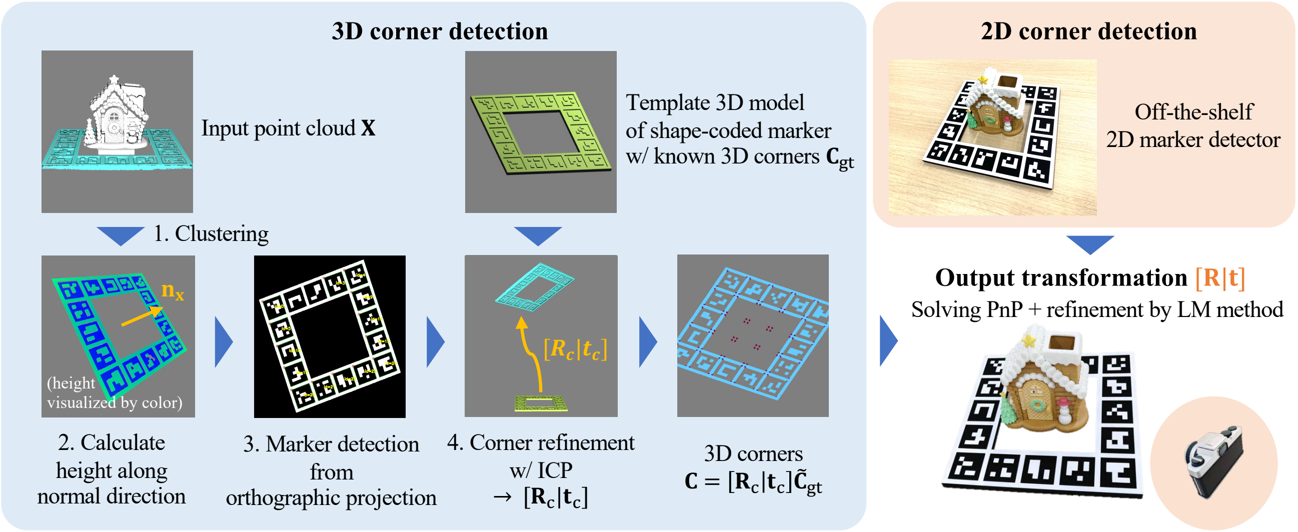

3D corner detection
Input point cloud $\mathbf{X}$
Template 3D model of shape-coded marker w/ known 3D corners $\mathbf{C}_{\mathrm{gt}}$
1. Clustering
$\mathbf{n}_{\mathrm{x}}$
(height visualized by color)
$[R_c|t_c]$
2. Calculate height along normal direction
3. Marker detection from orthographic projection
4. Corner refinement w/ ICP → $[\mathbf{R}_{\mathrm{c}}|\mathbf{t}_{\mathrm{c}}]$
3D corners $\mathbf{C} = [\mathbf{R}_{\mathrm{c}}|\mathbf{t}_{\mathrm{c}}]\tilde{\mathbf{C}}_{\mathrm{gt}}$
2D corner detection
Off-the-shelf 2D marker detector
Output transformation $[\mathbf{R}|\mathbf{t}]$
Solving PnP + refinement by LM method

Figure 4: A 2D-3D registration pipeline using the shape-coded ArUco marker.

in OpenCV) to detect the markers in 2D images. For the marker detection in the 3D domain, we develop a hybrid method of point cloud processing and 2D marker detection as illustrated in Fig. 4.

4.1. Marker detection in 3D point cloud

Given a point cloud from a 3D scanner, we detect 3D ArUco markers and their corner points as illustrated in Steps 1–3 of Fig. 4. In these steps, our method converts the point cloud to a binary image so that existing 2D marker detectors can be employed. We assume that the background 3D points (*i.e.*, points other than the marker and target object) are removed from the input point cloud, which is done automatically by many commercial 3D scanners when the target scene is placed on a planar surface, such as a desk.

Given the point cloud, our method first separates the marker from the target object using a density-based clustering method, DBSCAN [3], as shown in Step 1 of Fig. 4. Since it is unknown which point cloud corresponds to the marker until the 2D detector identifies the marker, the following steps are applied to each point cloud cl usters.

For each point cloud $\mathbf{X} \in \mathbb{R}^{3\times d}$ where d is the number of points, we further segment it into two parts; one is the top part, and the other is the embossed part as shown in Step 2 in Fig. 4. To achieve this, we transform the input point cloud $\mathbf{X}$ to the coordinates aligned to its normal direction $\mathbf{X}_{\mathrm{n}}$. We use Principal Component Analysis (PCA) to obtain a rotation matrix $\mathbf{R}_{\mathrm{n}} \in \mathrm{SO}(3)$ transforming to the new coordinate system spanned by the principal components, $\mathbf{X}_{\mathrm{n}} = \mathbf{R}_{\mathrm{n}}\mathbf{X}$. The third principal component corresponds to the normal $\mathbf{n}$ of the fitted plane and is aligned to the z-axis after the rotation. We then apply thresholding to the z-component of $\mathbf{X}_{\mathrm{n}}$ using Otsu's method [25] so that the marker point cloud is split into the top (white) $\mathbf{X}_{\mathrm{n(top)}}$ and embossed (black) $\mathbf{X}_{\mathrm{n(emb)}}$ parts based on their heights.

In Step 3 of Fig. 4, we project the point cloud to an image plane perpendicular to the normal $\mathbf{n}$ via orthographic projection. The 2×4 projection matrix $\mathbf{P}$ is computed for mapping the 3D points onto the 2D image coordinates $\mathbf{X}_{\mathrm{2D}}$ as $\mathbf{X}_{\mathrm{2D}} = \mathbf{P}\tilde{\mathbf{X}}_{\mathrm{n}}$, where $\tilde{}$ indicates the homogeneous representation, such that all points are properly projected inside the image. In the projected image, the pixels corresponding to the top part $\mathbf{P}\tilde{\mathbf{X}}_{\mathrm{n(top)}}$ are made white, and the rest are set black (see Step 3 in Fig. 4). We then apply a 2D ArUco detector on the image to obtain the maker identification numbers and the corner points of the markers $\mathbf{C}_{\mathrm{2D}} \in \mathbb{R}^{2\times c}$, where c is the number of detected corners. At this point, the point cloud is classified as the marker or the target object based on the number of detected corners c. Generally, zero marker corners are detected in the target object; therefore, the procedure stably works by simply regarding the point cloud with a greater c as the marker point cloud. Once the corner points are detected on the image, we back project the 2D corner points to the plane of the top surface in the point cloud to obtain the 3D corner points $\mathbf{C}_{\mathrm{n}} \in \mathbb{R}^{3\times c}$ in the normal-aligned coordinates. Finally, the 3D corner points $\mathbf{C}_{\mathrm{n}}$ are transformed to the original point cloud coordinates by $\mathbf{C}_{\mathrm{init}} = \mathbf{R}_{\mathrm{n}}^{\top}\mathbf{C}_{\mathrm{n}}$. We treat the 3D corner points $\mathbf{C}_{\mathrm{init}}$ as the initial corner points that are subsequently refined.

4.2. 3D corner refinement

Accurate 3D corner detection is a key to accurate 2D-3D registration. In Step 4 of Fig. 4, given the initial corner points $\mathbf{C}_{\mathrm{init}}$, we refine the corner point localization using a synthetic 3D model of the marker that are used for the production of the physical marker.

Our method computes the transformation between the initial 3D corners $\mathbf{C}_{\text{init}}$ and the corresponding synthetic 3D corners $\mathbf{C}_{\text{gt}}$ by solving the Procrustes problem [33]. The computed transformation is further refined by an iterative closest point (ICP) method [20] between the input point cloud and synthetic marker point cloud. With the refined transformation from the synthetic point cloud to the input cloud $[\mathbf{R}_{\text{c}}|\mathbf{t}_{\text{c}}]$, the 3D positions of marker corners $\mathbf{C}$ are finally determined as $\mathbf{C} = [\mathbf{R}_{\text{c}}|\mathbf{t}_{\text{c}}]\tilde{\mathbf{C}}_{\text{gt}}$.

4.3. Camera pose estimation

We now have accurate 2D-3D correspondences at the corner points on the shape-coded markers. Similar to conventional marker-based camera pose estimation, we use an off-the-shelf linear solver for the PnP problem [2] with calibrated camera intrinsics. Using the PnP-based camera pose as the initial guess, we further refine the estimated pose by minimizing reprojection errors using the Levenberg-Marquardt (LM) method [16, 22].

5. Experiments

In this section, we evaluate our shape-coded ArUco using both synthetic and real-world data.

5.1. Implementation details

We create a shape-coded ArUco marker board that contains 16 square markers as shown in Fig. 1. For both the real-world and synthetic experiments, the board size is set to $252 \times 252 \times 9.9\,\text{mm}$, where the embossed part is $3.3\,\text{mm}$ below the top surface. The marker board is modeled by Blender and fabricated by a 3D printer[3]. To avoid appearance variations caused by reflection and shadowing, we color the black part with low-reflective black paint, while the top surface is painted white. We publish the 3D model and fabrication instructions for our shape-coded ArUco in the project page.

In the implementation of 2D-3D registration, we use OpenCV[4] for intrinsics calibration as well as 2D ArUco marker detection. For point cloud processing, we use Open3D [43]. For solving the PnP problem with refinement using the LM method, we use the implementation in OpenCV [2]. In the process of DBSCAN clustering, we set the maximum neighborhood distance between two samples as $1.5\,\text{mm}$ and the minimum number of samples in a neighborhood as 100 throughout the experiments.

5.2. Baselines

For comparisons, we evaluate a marker-based 2D-3D registration method as well as a 3D registration method.

2D marker-based method. We compare with a method by Zhou *et al*. [42] that uses the line correspondences of the outer edge of the marker board, which shows the state-of-the-art accuracy among the 2D marker-based registration methods. In the synthetic experiment, we use the 2D version of our marker for their method for a fair comparison. For the real-world experiment, since it is difficult to place the 2D marker at exactly the same position as ours, we use the same 2D images of the shape-coded marker for both our method and their method.

We use an implementation of [42] in the Lidar Toolbox of Matlab. Since this method needs to specify a rough position of the marker plane for initialization, we input the position of the marker's top surface estimated by our detection pipeline. Besides, the solution by this method has an ambiguity when using square markers; we thus manually disambiguate by selecting the correct 2D-3D correspondences. Hereafter, we call their method the *2D marker* method.

3D registration method. Although our method is applicable for a single-view input, if images from multiple views are accessible, another promising approach to 2D-3D registration is to cast the problem to well-studied 3D-3D registration by recovering 3D shape from multiple view images [35, 34]. We thus evaluate a *3D registration* method for comparison. We create an image-based 3D model by structure-from-motion (SfM) [31] with multi-view stereo (MVS) [32]. After that, we apply ICP to obtain the transformation between the MVS-based model to the scanned 3D model. For ICP, we give the initial transformation based on manually selected 3D-3D correspondences.

5.3. Evaluation metrics

For synthetic dataset, we assess the errors in relative camera translation $e_{\mathbf{t}}$ and rotation $e_{\mathbf{R}}$ [radian] with respect to the ground truth transformation. While the rotation error $e_{\mathbf{R}}$ is computed as the angular error of the camera direction [8], translation error $e_{\mathbf{t}}$ is calculated as the relative value to the distance between the marker and camera to be insusceptible to the scale of the target scene. Given the estimated translation vector $\mathbf{t}$ and rotation matrix $\mathbf{R}$ from the marker origin to camera, the errors are defined as

$$\begin{aligned} e_{\mathbf{t}} &= \frac{\|\mathbf{t} - \mathbf{t}_{\text{GT}}\|_2}{\|\mathbf{t}_{\text{GT}}\|_2}, \\ e_{\mathbf{R}} &= \arccos\left(0.5\left(\text{tr}\left(\mathbf{R}_{\text{GT}}^{\top}\mathbf{R}\right) - 1\right)\right), \end{aligned}$$

where $\mathbf{t}_{\text{GT}}$ and $\mathbf{R}_{\text{GT}}$ denote the ground-truth translation and rotation, respectively.

[3] HP Jet Fusion 4200, https://www.hp.com/us-en/printers/3d-printers/products/multi-jet-fusion-4200.html, last accessed on August 17, 2021.

[4] OpenCV 4.2.0, https://opencv.org, last accessed on August 17, 2021.

Table 1: Overall accuracy of camera pose estimation for synthetic environment.

		Relative translation error $e_{\mathbf{t}}$			Rotation error $e_{\mathbf{R}}$ [radian]		
Method	Input	Mean ($\times 10^{-2}$)	Median ($\times 10^{-3}$)	SD	Mean ($\times 10^{-1}$)	Median ($\times 10^{-3}$)	SD
2D marker [42]	Single image	4.048	1.220	8.049	6.475	8.098	1.260
Ours	Single image	**0.07875**	**0.2545**	**0.1236**	**0.01460**	**0.3233**	0.002778
3D registration	Multi view	0.3848	3.326	0.2625	0.02165	2.164	**0.0005753**

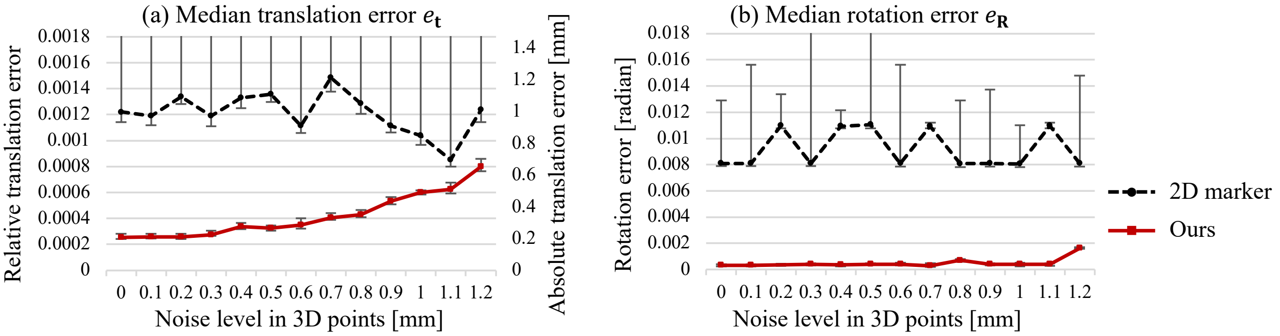


Figure 5: Median errors for different noise levels, with error bars for 25th and 75th percentiles. Our method shows a performance robust to the different noise levels.

For the real-world dataset, we have no access to the ground truth transformations. We thus qualitatively assess the accuracy by mapping colors on the untextured 3D shapes.

5.4. Experiment with synthetic data

In this section, we evaluate our method quantitatively on a synthetic dataset.

Synthetic environment. We use Blender[5] to model and render the synthetic environment. The input images are rendered under a point light source, using cameras with 37.85 [degree] vertical field of view. The cameras distribute uniformly on a spherical cap defined by a polar angle θ and radius r. We set $\theta = \pi/6$ and r as three times the length of one side of the marker board and define 50 viewpoints in the spherical cap. To evaluate the accuracy of camera localization, only the marker board is placed in the scene.

Results. Table 1 summarizes the results for synthetic and noise-free environment. We report the mean, median, and the standard deviation (SD) of the evaluation metrics $\{e_{\mathbf{t}}, e_{\mathbf{R}}\}$ calculated from 50 viewpoints. For the methods taking a single image as input (2D marker [42] and Ours), the camera poses are estimated independently for each view. For the 3D registration method, we use all 50 images to reconstruct the marker 3D shape and estimate the camera poses. Our method yields higher accuracies for both camera translation $e_{\mathbf{t}}$ and rotation $e_{\mathbf{R}}$ in comparison to the 2D marker method [42]. The main difference between the 2D marker method and ours is that our marker board yields a larger number of point correspondences, while the 2D marker method only acquires four-line correspondences. We can also see that our single-view method achieves a better estimation than the multi-view 3D registration method that uses 50 images for the camera pose estimation.

To assess the robustness of the methods against noise in point clouds, we add Gaussian noise with varying standard deviations to the point clouds. Figure 5 shows the translation and rotation errors, $e_{\mathbf{t}}$ and $e_{\mathbf{R}}$, for different noise levels. We found the 2D marker method produced large mean errors even for a small noise level. Therefore, to eliminate the effect of outliers, we show the median errors as well as error bars indicating 25th and 75th percentiles in the figure. Besides, to observe the absolute breakdown point of our method for the shape-coded marker, we also show the absolute translation error calculated by $\|\mathbf{t} - \mathbf{t}_{\mathrm{GT}}\|_2$. Overall, our shape-coded ArUco marker achieved accurate and stable estimates of camera poses. When the standard deviation of noise exceeds $1.3\,\mathrm{mm}$, our method starts to break down due to misdetection of the marker from the point cloud. Still, our method is useful with many recent commercial laser scanners because even the casual ones[6] usually achieve $< 0.3\,\mathrm{mm}$ accuracy.

[5] Blender 2.83 LTS, `https://www.blender.org`, last accessed on August 17, 2021.

[6] *E.g.*, Phiz 3D Scanner, `https://www.kiri-innov.com/products/phiz-3d-scanner`, last accessed on August 17, 2021.

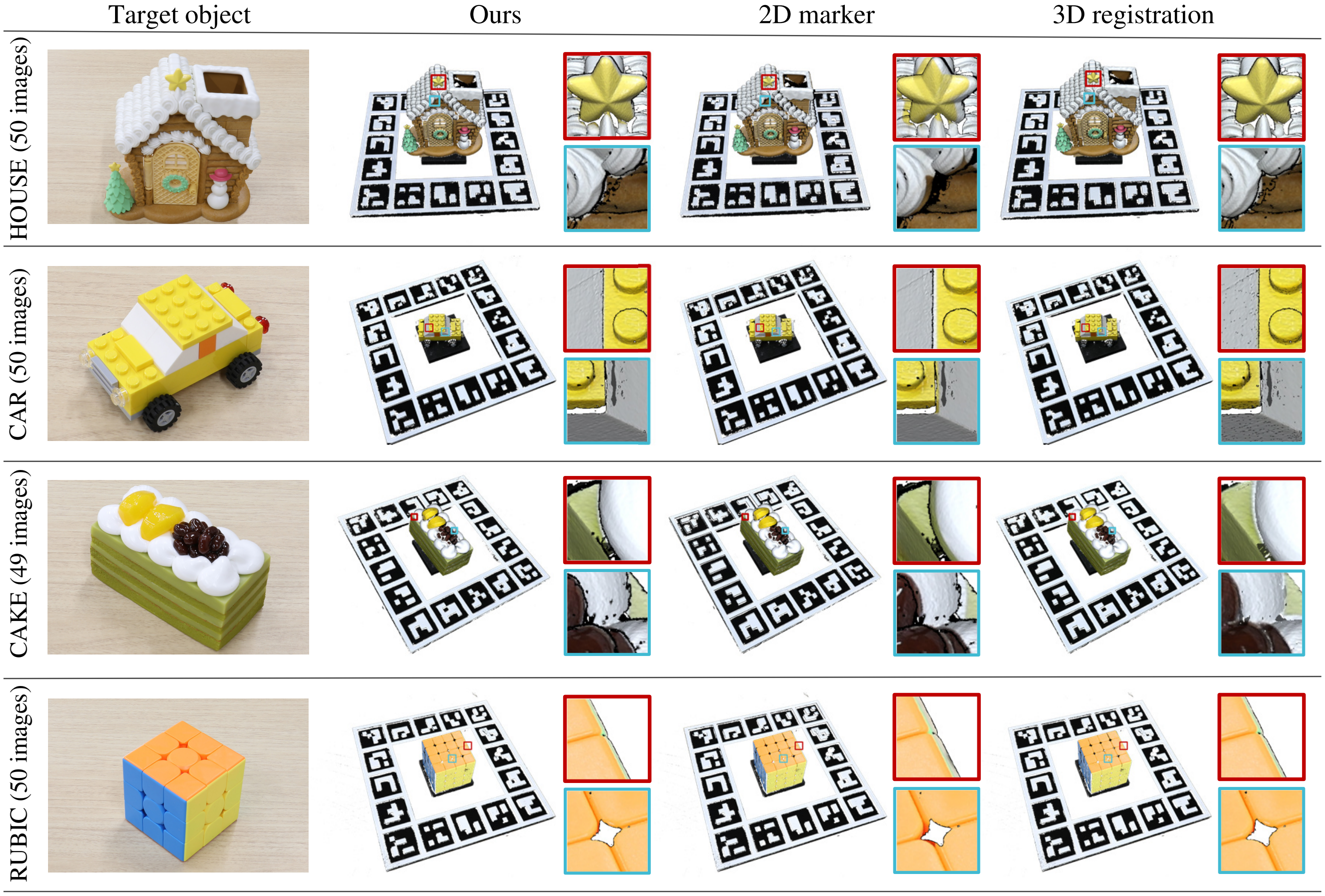


Figure 6: Color mapping results using real-world data (best viewed with color). The first column shows the target objects used in the experiment, and the second to fourth columns visualize the color mapping results by each method. The 2D marker method produces a larger misalignment (*e.g*., around the star in HOUSE). Our single-image method yields visually plausible alignment, which is comparable with the 3D registration method using multi-view input and manual intervention.

5.5. Real-world experiment

We here show the qualitative results for the color mapping using our shape-coded ArUco for real-world data.

Settings. We put the marker and target object on a desk and capture approximately 50 images with a similar setting as in synthetic experiments. We calibrate the intrinsics of the camera using ChArUco[7] board beforehand. The distance between the object and camera is set approximately 60 cm. Untextured 3D shapes are acquired by a handy laser scanner, HandySCAN 3D[8]. Using the camera poses estimated by each method, we use MeshLab[9] to map the 2D textures on the untextured 3D shapes.

Results. Figure 6 shows the real-world examples of color mapping from a single image by our shape-coded ArUco marker for four objects: CAKE, RUBIC, HOUSE, and CAR. As shown in the close-up views in Fig. 6, the results by the 2D marker method exhibit larger misalignment than ours, typically observed in the star on the roof in HOUSE, the yellow block on CAR, and the boundary of the whipped cream on the CAKE. The 3D registration method yields visually plausible alignment similar to ours, while the 3D method uses all (approximately 50) input images and involves the manual selection of initial correspondences.

To analyze the source of the alignment errors, we show the close-up views around the corner points of the marker board in Fig. 7. The 2D marker method produces a larger misalignment, although it uses the feature correspondences at the marker board boundaries. This result implies the instability of the 2D marker method that only uses a small number of correspondences.

[7] ChArUco, `https://calib.io`, last accessed on August 17, 2021.

[8] HandySCAN 3D, `https://www.creaform3d.com/en/portable-3d-scanner-handyscan-3d`, last accessed on August 17, 2021.

[9] Meshlab 2021.05, `https://www.meshlab.net`, last accessed on August 17, 2021.

Table 2: Comparison of relative errors in 3D corner detection e_{3D}.

Method	3D corner detection error e_{3D} [mm] Mean ($\times 10^{-3}$)	 Median ($\times 10^{-3}$)	 SD ($\times 10^{-1}$)
2D marker [42]	4.654	4.663	1.403
Ours w/o refinement	0.6148	0.6015	0.2173
Ours w/ refinement	**0.002447**	**0.002435**	**0.001011**

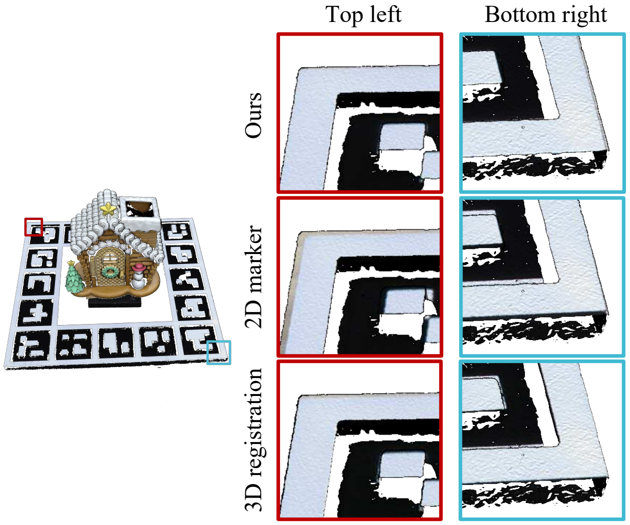


Figure 7: Color mapping results on the marker board.

5.6. Corner detection accuracy

To assess our 3D marker detection method, we assess the 3D corner detection accuracy using the synthetic environment. We here compare our method with and without the 3D corner refinement described in Sec. 4.2 as well as the 2D marker method [42]. Let us denote the ground truth corner point of a marker as $\mathbf{c}_{\text{gt}} \in \mathbb{R}^3$, the corresponding 3D corner estimate as $\mathbf{c} \in \mathbb{R}^3$, and the estimated transformation from the ground-truth marker coordinates to the point cloud coordinates as $[\mathbf{R}_c|\mathbf{t}_c]$. We evaluate the Euclidean distance between the corner points normalized by the length of one side of the square marker board t_m as

$$e_{3D} = \frac{\|\mathbf{c} - (\mathbf{R}_c\mathbf{c}_{\text{gt}} + \mathbf{t}_c)\|_2}{t_m}.$$

Table 2 summarizes the mean, median, and the standard deviation of the errors in 3D corner detection for 50 viewpoints. Compared with the 2D marker method, our method yields better 3D correspondences even without refinement, thanks to the larger number of correspondences. Besides, the ICP-based refinement process greatly contributes the accurate corner detection.

5.7. Effect of marker deformation for 2D detector

The key of our shape-coded ArUco is its projective invariant appearance property. To assess the effect of the shape deformation to the 2D marker detection, we compare the accuracy of the 2D marker detector for 2D and shape-coded ArUco markers in the real-world environment. We use the same pattern and size for both markers, where the 2D marker is printed on matte flat paper. We switch the markers by placing them at approximately the same position while we fix the camera and lighting conditions during the recording. We recorded the images both indoor with ambient light and under sunlight to assess the accuracy variations due to environments.

Table 3: Comparison of marker detection accuracy in 2D images.

Method	Reprojection error [px] Mean	 Median	 SD
2D ArUco	**5.602**	**5.195**	1.234
Shape-coded ArUco	6.000	5.884	**0.6682**

Table 3 summarizes the mean, median, and standard deviation of the reprojection errors, computed from 54 iterations of the capturing processes by changing the camera location. We use Canon EOS 5D Mark IV camera with the image resolution 6720×4480 pixels and the lens with focal length $35\,\text{mm}$. Our shape-encoded ArUco yields a comparable accuracy with the ordinary ArUco markers, indicating that the marker deformation does not affect the marker detection in 2D images. In addition, we have not observed accuracy variations due to environments.

6. Conclusion

We have proposed a fiducial marker named a shape-coded ArUco marker that is useful for the task of 2D-3D registration. We have also developed a method for detecting the marker from the untextured 3D point cloud. The key feature of our marker is its projective invariant appearance property that preserves the same appearance as the 2D ArUco, but shape information is encoded so that it is detectable in the untextured 3D point cloud. The experiments show that our method enables accurate 2D-3D correspondences in comparison to the state-of-the-art 2D marker-based method. Our method works with a single-image input and does not need any human intervention, unlike the 3D-3D alignment method. We believe that the proposed method is useful for bridging the 2D and 3D modalities that is essential to reality modeling.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP19H01123. We express our sincere thanks to Xu Cao for his help in building our shape-coded ArUco board.

References

[1] Mark Brown, David Windridge, and Jean-Yves Guillemaut. Globally Optimal 2D-3D Registration from Points or Lines without Correspondences. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2111–2119, 2015.

[2] Toby Collins and Adrien Bartoli. Infinitesimal Plane-Based Pose Estimation. *International Journal of Computer Vision (IJCV)*, 109(3):252–286, 2014.

[3] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD)*, pages 226–231, 1996.

[4] Vincent Fremont and Philippe Bonnifait. Extrinsic calibration between a multi-layer LiDAR and a camera. In *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 214–219, 2008.

[5] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.

[6] Andreas Geiger, Frank Moosmann, Ömer Car, and Bernhard Schuster. Automatic camera and range sensor calibration using a single shot. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 3936–3943, 2012.

[7] Xiaojin Gong, Ying Lin, and Jilin Liu. 3D LiDAR-camera extrinsic calibration using an arbitrary trihedron. *Sensors*, 13(2):1902–1918, 2013.

[8] Richard Hartley, Jochen Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *International Journal of Computer Vision (IJCV)*, 103(3):267–305, 2013.

[9] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6D object pose estimation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3385–3394, 2019.

[10] Jiunn-Kai Huang, Shoutian Wang, Maani Ghaffari, and Jessy W. Grizzle. LiDARTag: A Real-Time Fiducial Tag System for Point Clouds. *IEEE Robotics and Automation Letters*, 6(3):4875–4882, 2021.

[11] Katsushi Ikeuchi, Atsushi Nakazawa, Kazuhide Hasegawa, and Takeshi Ohishi. The Great Buddha Project: Modeling Cultural Heritage for VR Systems through Observation. In *Proceedings of the IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 7–16, 2003.

[12] Jiyoung Jung, Joon Young Lee, Yekeun Jeong, and In So Kweon. Time-of-flight Sensor Calibration for a Color and Depth Camera Pair. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 37(7):1501–1513, 2015.

[13] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1521–1529, 2017.

[14] Julius Kümmerle and Tilman Kühner. Unified intrinsic and extrinsic camera and LiDAR calibration under uncertainties. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 6028–6034, 2020.

[15] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. EPnP: An accurate O(n) solution to the PnP problem. *International Journal of Computer Vision (IJCV)*, 81(2):155–166, 2009.

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

[17] Ganhua Li, Yunhui Liu, Li Dong, Xuanping Cai, and Dongxiang Zhou. An algorithm for extrinsic parameters calibration of a camera and a laser range finder using line features. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 3854–3859, 2007.

[18] Jiaxin Li and Gim Hee Lee. DeepI2P: Image-to-Point Cloud Registration via Deep Classification. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15960–15969, 2021.

[19] Lingyun Liu and Ioannis Stamos. Automatic 3D to 2D registration for the photorealistic rendering of urban scenes. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 137–143, 2005.

[20] Kok-Lim Low. Linear least-squares optimization for point-to-plane ICP surface registration. Technical report, Chapel Hill, University of North Carolina, 2004.

[21] David G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, 1991.

[22] Donald 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.

[23] Kenji Matsushita and Toyohisa Kaneko. Efficient and handy texture mapping on 3d surfaces. In *Computer Graphics Forum*, volume 18, pages 349–358, 1999.

[24] Mohammad Omidalizarandi and Ingo Neumann. Comparison of target-and mutual informaton based calibration of terrestrial laser scanner and digital camera for deformation monitoring. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(1W5):559–564, 2015.

[25] Nobuyuki Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[26] Yoonsu Park, Seokmin Yun, Chee Sun Won, Kyungeun Cho, Kyhyun Um, and Sungdae Sim. Calibration between color camera and 3D LIDAR instruments with a polygonal planar board. *Sensors*, 14(3):5333–5353, 2014.

[27] Sida Peng, Xiaowei Zhou, Yuan Liu, Haotong Lin, Qixing Huang, and Hujun Bao. PVNet: Pixel-wise Voting Network for 6DoF Object Pose Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2020. early access.

[28] Tobias Plötz and Stefan Roth. Automatic Registration of Images to Untextured Geometry Using Average Shading Gradients. *International Journal of Computer Vision (IJCV)*, 125(1-3):65–81, 2017.

[29] Zoltan Pusztai and Levente Hajder. Accurate Calibration of LiDAR-Camera Systems Using Ordinary Boxes. In *Proceedings of the International Conference on Computer Vision Workshop (ICCVW)*, pages 394–402, 2017.

[30] Hatem A. Rashwan, Sylvie Chambon, Pierre Gurdjos, Géraldine Morin, and Vincent Charvillat. Using Curvilinear Features in Focus for Registering a Single Image to a 3D Object. *IEEE Transactions on Image Processing (TIP)*, 28(9):4429–4443, 2019.

[31] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[32] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 501–518, 2016.

[33] Peter H Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[34] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3260–3269, 2017.

[35] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 519–528, 2006.

[36] Levente Tamas and Zoltan Kato. Targetless Calibration of a Lidar - Perspective Camera Pair. In *Proceedings of the International Conference on Computer Vision Workshop (ICCVW)*, pages 668–675, 2013.

[37] Tekla Tóth, Zoltán Pusztai, and Levente Hajder. Automatic LiDAR-Camera Calibration of Extrinsic Parameters Using a Spherical Target. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 8580–8586, 2020.

[38] Yuki Uranishi, Akimichi Ihara, Hiroshi Sasaki, Yoshitsugu Manabe, and Kunihiro Chihara. Real-time representation of inter-reflection for cubic marker. In *International Symposium on Mixed and Augmented Reality*, pages 217–218, 2009.

[39] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. 2017.

[40] Huai Yu, Weikun Zhen, Wen Yang, and Sebastian Scherer. Line-based 2-D–3-D registration and camera localization in structured environments. *IEEE Transactions on Instrumentation and Measurement*, 69(11):8962–8972, 2020.

[41] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. DPOD: 6D pose object detector and refiner. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1941–1950, 2019.

[42] Lipu Zhou, Zimo Li, and Michael Kaess. Automatic Extrinsic Calibration of a Camera and a 3D LiDAR Using Line and Plane Correspondences. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 5562–5569, 2018.

[43] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018.