FCV2016, the 22nd Korea-Japan Joint Workshop on Frontiers of Computer Vision

Camera Pose Estimation Based on Keypoints Matching with Pre-Captured Set of Real Images

Kei Obata
Graduate School of Science and Technology
Keio University
Yokohama, Japan
Email: obata@hvrl.ics.keio.ac.jp

Hideo Saito
Keio University
Yokohama, Japan
Email: saito@hvrl.ics.keio.ac.jp

***Abstract*—In this paper, we propose a method for camera pose estimation based on keypoints matching with images capturing the target scene from multiple viewpoints. For estimating camera pose from an image, 3D structure of the target scene is needed. Then a set of correspondences between 2D position in the image and 3D positions in the target scene provides the camera pose by solving Perspective-n-Point problem. Image keypoints are used as these positions, but finding corresponding 3D positions of the 3D structure is not easy problem even if a 3D textured model of the target scene is given. This is because the appearance of each 3D position is not always the same. In order to solve this problem, we adopt real images as a source of keypoints contained in the database. Real image represents accurate appearance for each 3D position, so keypoint matching with the input images can be more accurate. We constructed database based on the keypoint using the local image feature derived from real images and evaluated the accuracy of camera pose estimation. The result showed that the database constructed by our method is effective to accurate camera tracking.**

I. Introduction

Estimation of camera poses from images captured by the camera is one of the essential issue in computer vision. A typical approach for the camera pose estimation is based on the data of the real scene such as the shape and texture obtained in advance, then comparing the data with the captured images for computing the camera pose. The correspondence of the 3D geometry of the target scene and 2D image enable us to compute the camera pose by using Perspective-n-Point (PnP) problem. To find the points for the matching in the image, various local invariant features have been proposed. However, they are not robust against affine transformations and change of appearance of the object caused by change of the viewpoint of the camera, therefore they cannot be applied to 3D object tracking including dynamic changes of camera pose.

Yoshida et al.[1] proposed the keypoint which contains the multiple image features derived from the projection images captured in various viewpoints. The keypoint contains the image features described in the multiple viewpoints at its position, which indicates that it is detected robustly as a keypoint from various viewpoints. They called such keypoint "stable keypoint", and constructed the database which contains the stable keypoints for camera pose estimation with respect to a flat textured surface. Tachasongtham et al.[2] applied them to the 3D object tracking. For obtaining the images of the 3D model, they use the viewpoint generative learning (VGL) method. In VGL method, a 3D textured model of the target object is used as an input, and the projected images from various viewpoint are synthesized by a regular manner in OpenGL[3].

However, using the synthesized images from the 3D textured model as an input of the database includes some problems. When a 3D model is used, we cannot render the same as appearance as the real scene. Even a synthesized image captured from a camera pose is not completely coincident with a real image captured at the same viewpoint as the synthesized image. There are many reasons of the difference. Even if we use one of the state-of-the-art 3D modeling technologies such as Structure from Motion (SfM) methods that have recently been studied by a lot of researchers[4][5][6] to create the 3D model, the shape of the 3D model isn't always accurate. In addition, it is necessary to set the surface property of each portion of the model since the material of the object influences the appearance of its surface. The setting of the property of the light source is also needed to reproduce the real appearance.

In order to solve these problems, we propose a method for camera pose estimation by matching with real images used as input of the creation of the 3D model. When the 3D textured model is reconstructed using them by SfM, the camera pose of each real image can also be computed, so that we can obtain each depth image captured in the pose in which the real image is taken. Using the sets of the real image and the depth image, we can obtain the 3D position and the image feature of each keypoint based on the real appearance. These images are the input of the SfM, therefore they capture the object in various viewpoints as the generated images do in VGL method.

We construct the keypoint database based on the real image sequence which is the input of SfM. Using the database, we estimated the camera pose. Our experimental result show that the database constructed by our method indicates more stable result than the one using the database based on the projected images in estimating camera poses.

II. Related Works

In this section, we introduce related works of camera pose estimation and relevant keypoint matching methods.

SIFT[7] and SURF[8] are widely used as keypoint detectors and descriptors. They are robust against the change of scale, rotation and translation. However, they are weak against the dynamic affine transformation.

The keypoint matching method proposed by Lepetit et al. [9] is effective to estimate camera pose against planar surface. In learning phase, affine transformation is applied randomly to the target image, and keypoint detection is executed in generated images. By using image patches around the detected keypoints, randomized trees are made. In keypoint matching, point-to-point matching is done by training the patch around the keypoint detected in input image in the trees. In this method, the random affine transformation could lead to cause unfair data for matching, and the amount of affine-transformed images is large.

Yoshida et al.[1] also proposed a method of camera pose estimation to planar surface robust against viewpoint change. This method uses the viewpoint generative learning (VGL). In learning phase, the images from various viewpoints are generated virtually. After executing keypoint detection in each image, the keypoints detected in many images are used for camera pose estimation because of their robustness against viewpoint change. For this reason, these keypoint are called "Stable Keypoint". For coordinating the image features in the database, k-means clustering is done in each stable keypoint. Tachasongtham et al.[2] applied Yoshida et al.'s method to 3D object. By treating stable keypoint in 3D space, camera pose estimation to 3D object is enabled. In [1] and [2], it is assumed that the target is covered with Lambertian surface. Therefore, both of them cannot be applied to objects whose appearance change depending on the viewpoint.

Shinozuka et al.[10] proposed a method in which all image features in each stable keypoint are saved into the database without clustering in order to deal with the change of appearance. However, the input of this method is still the set of synthesized images from textured 3D models, therefore the image features of keypoints aren't matched with real appearances.

In our proposed method, we don't use synthesized images from textured 3D model, but use real images captured for reconstructing the 3D model of the scene, so that the image features of each stable keypoints are computed from the same same appearance as the real images taken at various viewpoints.

III. Database Construction

In this section, we describe the algorithm of constructing the database for the camera pose estimation. The algorithm is shown in Fig.1. First of all, we make a 3D model of the target scene using the real images by SfM. By looking the 3D model in the equal camera pose in which the real image is captured, we can obtain the 3D position of the 2D keypoints in each real image. The 2D keypoints which are detected repeatedly from many viewpoints are summarized in the 3D keypoints, then 3D keypoints are stored into the database.

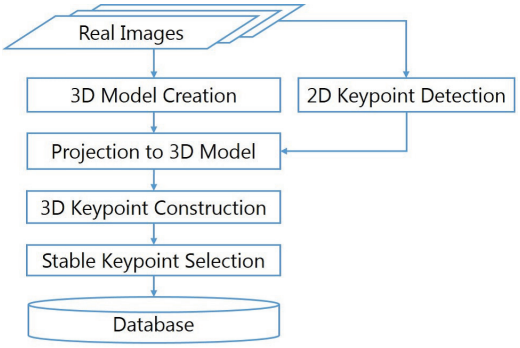


Fig. 1. Overview of Database Construction

A. Obtaining the 3D Positions of 2D Keypoints

A real image sequence of a target scene is captured from various viewpoints. Then a 3D model can be reconstructed by the captured images using SfM. In our method, we employ Autodesk 123D Catch[12] as a SfM for 3D model reconstruction. When the 3D model is obtained, camera pose of every image in the input image sequence can also be estimated by Autodesk 123D Catch. They are represented in the world coordinate system which is set in the reconstruction process. Fig.2 shows the example images of a input real image sequence for reconstructing the 3D model shown in Fig.3(a) with the estimated camera poses shown in Fig.3(b). Fig.4 shows the depth images of the 3D model synthesized at the same camera pose as the part of input images shown in Fig.2. The depth image provides 3D positions of 2D keypoints detected in the input real image at the same viewpoint as the depth image. In this way, we can estimate 3D positions of all keypoints detected in all input images.

B. Making 3D Keypoints

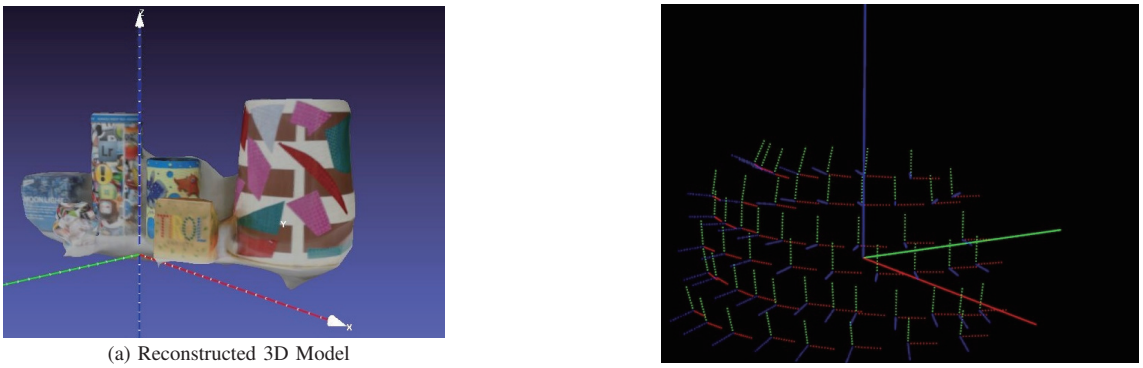
By regarding keypoints detected in the different images with similar 3D positions, we can collect 2D keypoints in different images for the same 3D position in the scene.

We focus attention on the viewpoint in which each input image is taken in order, and set the 2D keypoint to the 3D coordinate system. We compute the all 3D distances between the 2D keypoint focused on ($kp2D_{tmp}$) and $kp3D_i$ in the set of 3D keypoint in existence $Kp3D = \{kp3D_i \mid 0 \leq i < n\}$. If the Euclidean distance d between $kp2D_{tmp}$ and $kp3D_i$ is closer than the set value D, it is assumed that they have the same 3D position. In this case, $kp2D_{tmp}$ becomes a part of what constitutes $kp3D_i$. If any $kp3D_i$ don't exist around $kp2D_{tmp}$, it is assumed that $kp2D_{tmp}$ is the new 3D keypoint and defined as $kp3D_n$. This process is applied to all 2D keypoints of each viewpoint images in sequence. In this process, the 3D keypoints which contains image features of each viewpoint are accumulated temporarily.

C. Selecting 3D Keypoints

If $kp3D_i$ has many image features, it means that $kp3D_i$ is detected as 2D keypoint from many viewpoints, and it can be

Fig. 2. Part of Input Real Images



(a) Reconstructed 3D Model (Modified Partly)

(b) Estimated Camera Poses

Fig. 3. Output of Autodesk 123D Catch[12]

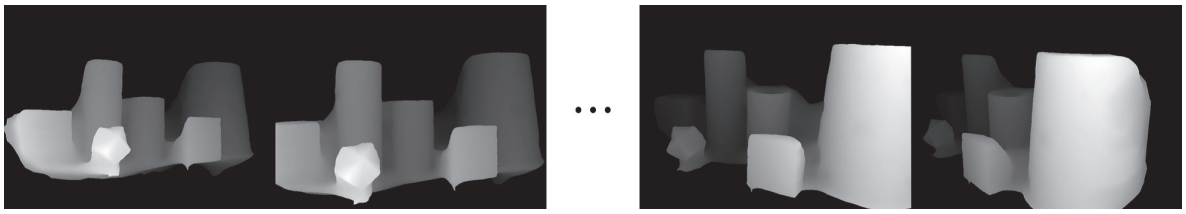


Fig. 4. Part of Depth Images

said that such point is effective to use for keypoint matching in camera pose estimation. Such 3D keypoint is defined as "Stable Keypoint" in [2]. According to this definition, we pick up such 3D keypoints from $Kp3D$ as stable keypoints.

The 3D keypoints are picked up as stable keypoint in descending order of the number of the image features they have. We set two thresholds, $Skpt_{max}$ and $Feature_{min}$, in advance for determining the size of the database. $Skpt_{max}$ is defined as the maximum number of stable keypoints stored into the database. The picking up process continues until the number of picked up 3D keypoints reaches $Skpt_{max}$. $Feature_{min}$ is defined as the minimum number of the image features which are held in each 3D keypoint. When the number of image features which the picked up 3D keypoint has is lower than $Feature_{min}$, this process ends even if the number of the stable keypoint is lower than $Skpt_{max}$.

For the sake of preventing following problems, these thresholds are set. First, too many stable keypoint causes the increase of computation, and it slows down the keypoint matching process in camera pose estimation phase. Second, the 3D point which has few image features means that it is not detected from many viewpoint. If such 3D keypoint is included in the database, it is not only the processing load, but also the cause of the error of keypoint matching.

D. Storing Keypoints into Database

After all stable keypoints are defined, each stable keypoint has multiple image features (e.g. SURF features) which corresponding 2D keypoints have as shown in Fig.5. The number of image features contained in each stable keypoint is too much to use for matching with image features of 2D keypoints in captured images for camera pose estimation phase. Therefore, the clustering algorithm is applied to reduce the number of image features of each stable keypoint. Because we use SURF as the image feature descriptor, each stable keypoint has its image features in the space of 64 dimensions. Each group of the image features of the stable keypoint is clustered by k-means clustering as shown in Fig.5 (in Fig.5, $k = 3$). By executing this process, each stable keypoint has k image features (the number of the centroids of clusters), which decreases the size of the data and reduces the cost of keypoint matching in camera pose estimation phase. In the end, the

database has the relations of image features and 3D positions as shown in the lower side of Fig.5.

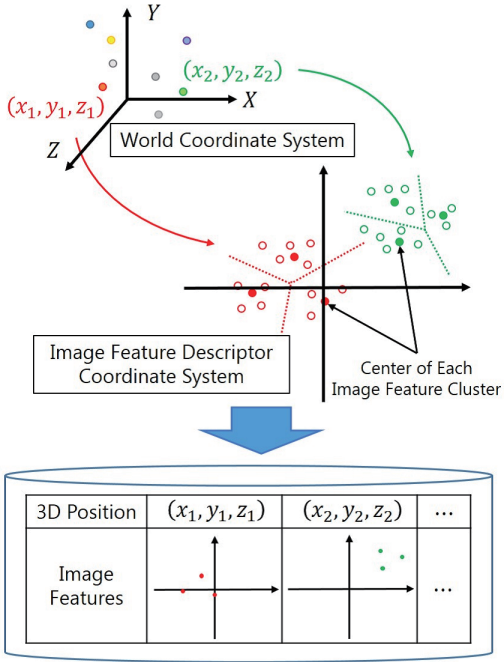


Fig. 5. Image Feature Clustering by k-means

IV. Camera Pose Estimation

The algorithm of camera pose estimation is shown in Fig.6. In camera pose estimation phase, we first get the image captured by the camera and detect the 2D keypoint from it. Then, keypoint matching between the 2D keypoints in the image and the 3D keypoints in the database is executed. For each 2D keypoint in the image, we search the 3D keypoints in the database whose Euclidean distances from the 2D keypoint in their image feature space are nearest($dist_1$) and second nearest($dist_2$). If these distances satisfy eq.(1), we use this matching for computing the camera pose.

$$\frac{dist_1}{dist_2} < \tau \tag{1}$$

Finally, we estimate the camera pose with the sets of 2D-3D keypoint matching pairs by solving PnP problem by using RANSAC algorithm[11].

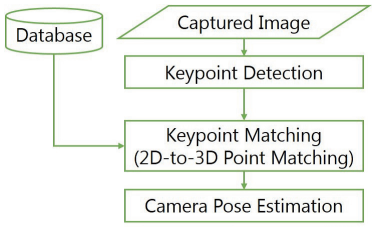


Fig. 6. Overview of Camera Pose Estimation

V. Experimental Result

We conducted camera pose estimation experiments using our method and Tachasongtham et al.'s VGL method[2]. The number of the input images of our method is 54, and the number of the generated images in VGL is 52. The range of generated pose of camera in VGL is adjusted to resemble the one in our method. The number of frames in which camera poses are computed is 95 in estimation phase. Fig.7 is the part of input frames.

The common parameters of the two methods are below : We chose SURF[8] for keypoint detector and descriptor, and defined the value D in III-B as 2.0mm. The number of keypoints in the database is 1000. The number of image features in each keypoint, in other words, the value k in III-D is 10. Then, both database have 1000×10 image features. In camera pose estimation, we set $\tau = 0.7$.

The examples of the results are shown in Fig.8 and Fig.9. In this experiment, we rendered the 3D textured model created in a database construction phase based on the estimated camera poses. If the camera pose is estimated correctly, the rendered model seems to be the same appearance as the input images in Fig.7. The images presented in Fig.8 demonstrate that our method worked well especially when the image was mainly filled with specular object. This was due to the images in which keypoints were detected in a database construction phase. In VGL method, synthesized images reflect the appearance of the 3D textured model. Textured model doesn't always look the same appearance as the real scene's. In particular, if the object don't have Lambertian surfaces (for example, the cylinder in the right side of the scene), this tendency is noticeable. In contrast, we used the real images as input of the database. Real image reflects the appearance of the real scene faithfully. Therefore, our method can deal with the change of the appearance coming from the viewpoint change. Fig.10 shows the example of the application to augmented reality (AR) of our method. We overlaid a penguin image on the front side of the golden box by projecting 3D position of the corner points surrounding the front side of the box onto each image using the estimated camera pose. This example indicates that our method can be applied to AR in the specular circumstances.

Fig.11 shows the number of 2D-to-3D inlier matching pairs of RANSAC algorithm in camera pose estimation phase. The matching pairs in inlier represents that they are correct matching. In almost all of the frames, the number of the correct matching of our method is larger than that of VGL. This fact indicates using real image sequences as input is effective to stable camera tracking.

VI. Conclusion

This paper proposes a method for the scene tracking. The keypoint obtained from a real image reflects real appearance, therefore we construct the database based on the real image sequence. By using these keypoints, we construct the stable keypoint database for estimating the camera pose. It has a robustness against the change of the surface appearance

Fig. 7. Part of the Input Captured Images

Fig. 8. Part of the Estimation Results Using Proposed Method

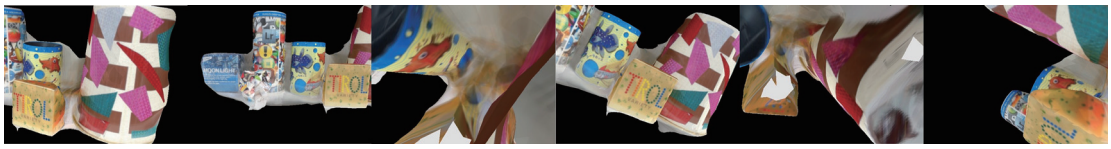


Fig. 9. Part of the Estimation Results Using VGL Method

Fig. 10. Application to AR Using Proposed Method

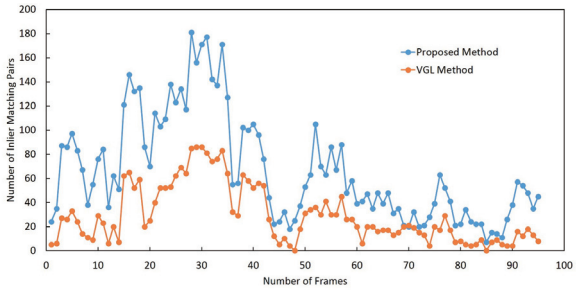


Fig. 11. Inlier matching pairs of both results

caused by the viewpoint. The experimental result shows that the database constructed by using our method achieves more accurate result than the one constructed by previous method based on the projected images of 3D textured model.

ACKNOWLEDGMENT

This work was supported in part by JSPS Grant-in-Aid for Scientific Research(S) 24220004.

REFERENCES

[1] T. Yoshida, H. Saito, M. Shimizu, and A. Taguchi, "Stable Keypoint Recognition using Viewpoint Generative Learning'," in *Proc. 8th International Conference on Computer Vision Theory and Applications*, Feb. 2013, pp. 310-315.

[2] D. Thachasongtham, T. Yoshida, F. de Sorbier and H. Saito, "3D Object Pose Estimation Using Viewpoint Generative Learning," *Image Analysis*, vol. 7944, pp. 512-521, 2013.

[3] OpenGL, ⟨http://www.opengl.org/⟩ (2015/12/1)

[4] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nov. 2007, pp. 225-234.

[5] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proc. 2014 IEEE International Conference on Robotics and Automation*, Jun. 2004, pp. 15-22.

[6] J. Engel, T. Schöps and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," *Computer Vision-ECCV 2014*, pp. 834-849, 2014.

[7] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[8] H. Bay, T. Tuytelaars and L. V. Gool, "SURF: Speeded Up Robust Features," in *Proc. 9th Europian Conference on Computer Vision*, May. 2006, pp. 404-417.

[9] V. Lepetit and P. Fua, "Keypoint Recognition Using Randomized Trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1465-1479, 2006.

[10] Y. Shinozuka, F. de Sorbier and H. Saito "Specular 3D object tracking by view generative Learning," in *Proc. Irish Machine Vision and Image Processing Conference*, Aug. 2014, pp. 9-14.

[11] 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*, vol. 24, no. 6, pp. 381-395, 1981.

[12] Autodesk 123D Catch, ⟨http://www.123dapp.com/catch⟩ (2015/12/1)