Robust vSLAM for dynamic scenes

Jun SHIMAMURA, Masashi MORIMOTO, and Hideki KOIKE
NTT Cyber Space Laboratories, NTT Corporation
1-1 Hikari-no-oka Yokosuka Kanagawa 239-0847, Japan
{shimamura.jun, morimoto.masashi, k.hideki}@lab.ntt.co.jp

Abstract

This paper proposes a robust visual Simultaneous Localization and Mapping(vSLAM) method that consists of two parts: (1)an initialization method for a 3D map and camera poses, and (2)an outlier rejection method for moving objects. We introduce weighted tentative initial values of a camera pose and 3-D map to reduce the user operation load. We also distinguish outliers of feature points between moving objects and errors to stably estimate camera poses in dynamic scenes. To achieve this, we first construct an angle histogram based on outlier flows at a current frame, then approximate the obtained angle histogram using a mixture of Gaussian functions. Finally, we estimate the parameters for Gaussian mixtures using the Expectation-Maximization algorithm. Results are shown to illustrate the superior performance of the proposed method.

1 Introduction

Augmented reality (AR) systems incorporate virtual information into actual environments to achieve enhanced functionality. One goal of AR is to overlay virtual information on a camera image to provide the end-user with additional knowledge about objects in the scene.

The vision-based real-time camera tracking technique called "visual Simultaneous Localization and Mapping" (vSLAM) helps one to browse virtual information without special equipment or prior information (e.g., CAD models) about the environment[1, 2, 3, 4, 5]. Typically, the camera pose is estimated from frame-to-frame feature points tracking over the short term, along with the position of the 3-D map points recovered using bundle adjustment. They usually run fast. However, drifting and relocalization problems could be produced when there are moving objects in an environment, because they mostly have imposed some constraints on the scene to be tracked: it should be mostly static, i.e. not dynamic.

In order to estimate camera poses in dynamic scenes stably, we propose a new vSLAM method that first distinguishes outliers between feature points upon the moving objects and errors, then eliminates the moving objects' 3-D points from the 3-D map. We also propose a map and pose initialization method without user operation that is necessary for determining initial camera pose and scene 3-D structure.

This paper is structured as follows. In Section 2, we describe a method of camera pose estimation vis-a-vis dynamic scenes by tracking natural features. Then, we demonstrate experimental results of camera pose estimation from actual environment image sequences to show the feasibility of the proposed method in Section 3. Finally, Section 4 summarizes the paper.

2 Robust vSLAM for dynamic scenes

2.1 Framework overview

Table 1 overviews the proposed method, which consists of two modules, i.e., the tracking thread and mapping thread. The tracking thread is responsible for estimating the camera poses for each input frame in real-time. The mapping thread is responsible for recovering 3-D points by triangulation and optimizing them. This design is almost the same as that of Parallel Tracking and Mapping (PTAM) [4]. Our method modifies map and camera pose initialization (Step 2.1), takes in outlier segmentation that distinguish between moving objects' points and errors (Step 1.5), then eliminates 3-D points from a map corresponding to the moving objects' points (Step 1.6).

Table 1. Framework overview

1.	**Tracking thread**
1.1	Estimate a tentative current camera pose by using motion model.
1.2	Project 3-D map points into current frame.
1.3	Search matching features nearby the projected point.
1.4	Estimate a camera pose and detect outliers with the matched features by using robust estimator.
1.5	*Segment outliers into moving object points and errors.*
1.6	*Eliminate moving object points from 3-D map.*
2.	**Mapping thread**
2.1	*Map and camera pose initialization at initial frame.*
2.2	Receive a new keyframe from tracking thread.
2.3	Extract FAST[6] features from the keyframe, then triangulate and add to map points.
2.4	Optimize map and keyframe poses by applying local or global bundle adjustment.

2.2 Map and camera pose initialization

While PTAM [4] needs end-user help for 3-D map and camera pose initialization, we have imposed some constraints on the initialized scene for excluding intrusive operation. It seems that end-users will train the camera on an object that lies on a plane when they would like to see virtual information of the objects. Therefore, we express an initial 3-D map with a sphere that lies on a plane and express initial camera pose so that the camera might direct to the sphere center in

the lower side. Figure 1(c) shows the map and camera pose initialization status described above.

In the actual scene, these constraints are slightly broken. However, 3-D map and camera poses, including those in initial frames, are modified through a bundle adjustment process[7] in Table 1, Step 2.4, as in the Figure 1(d). We introduce a weighting coefficient into the sum of re-projection errors as in Equation 1, which is minimized by a Levenberg-Marquardt optimizer, to improve the optimization accuracy and convergence speed of bundle adjustment.

$$E = \sum_{f} W(f) \sum_{p} |\mathbf{x_{fp}} - \hat{\mathbf{x}}_{\mathbf{fp}}|^{\mathbf{2}}. \quad (1)$$

where $\mathbf{x_{fp}}$ is the position of a matched feature and $\hat{\mathbf{x}}_{\mathbf{fp}}$ is the re-projected position of a corresponding 3-D map point. $W(f)$ is a weighting function for the frame. It assigns a larger value to former frames than to later frames, since the map and camera pose nearby the initial frame are considered to be unreliable. We use a Gaussian function as the $W(f)$.

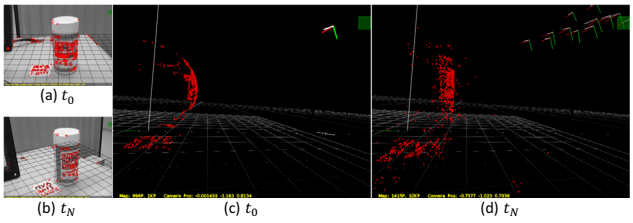

Figure 1. Map and camera pose initialization. (a),(b) shows video image overlaid grid CG. (c),(d) shows 3-D map and camera pose.

2.3 Segmenting outliers into moving object points and errors

To cope with camera pose drifting and relocalization problems that could be produced when there are moving objects in an environment, our key idea is eliminating moving objects points from the 3-D map and estimating later camera pose by using only static 3-D points at Step1.4 shown in Table 1.

This idea will be implemented to eliminate all matched features and corresponding 3-D map points that were judged as outliers by a robust estimator (e.g., the Tukey M-Estimator). This judgment is conducted at the camera pose estimation procedure. However, in textureless environments particularly, it sometimes causes drifting and relocalization problems, because accuracy of camera pose estimation often worsens as the number of matched features decreases. Additionally, outliers caused by occlusions, specular reflections, and other reasons within a certain frame have the possibility of becoming inliers in other frames. This is why we segment outliers into moving object points and errors. We describe below how to extract moving object points from outliers.

2.3.1 Classification of cause of outliers

The cause of outliers on the camera pose estimation process is separated into two types: (1) moving objects in the scene and (2) matching errors. The second can be further separated into five types, i.e., the existence of (2-i) occlusions, (2-ii) specular reflections, (2-iii) textureless features, (2-iv) aperture problems, and (2-v) repeating textural regions in the scene. Our goal in this section is to detect the first type, i.e., moving objects in the scene from outliers.

The reason for (2-iii), (2-iv), and (2-v) is mostly that similar patterns exist near a correct matched feature. These can be eliminated by using self-correlation in a current frame. If a feature having a high correlation score exists nearby a matched feature, we can exclude it as a matching error falling under categories (2-iii), (2-iv), and (2-v).

The problem is how to segment and extract features upon moving objects with matching errors belonging to occlusions and specular reflections. Fortunately, flow vectors consist of projected points and matched features on occlusions and specular reflections tend to emanate angles because of the forced search, though there are no correspondence points. In contrast, flow vectors on moving objects tend to unify the angles when the object moves, except for translation and rotation around the camera's optical axis. Considering that flow vectors include Gaussian noise, we can approximate the distribution of angles of flow vectors as a Gaussian Mixture Model (GMM).

2.3.2 Optical flow segmentation observed by a fixed camera

To simplify our explanation, we first describe a method of segmenting optical flows observed by a fixed camera in a dynamic scene.

We first choose optical flows that have magnitudes. Second, we eliminate optical flows caused by the existence of similar patterns by using self-correlation in a current frame. A template is generated from pixels surrounding each target pixel in the current frame, and then the second-best match for this template within a fixed rectangle region around a target pixel is found. This is done by evaluating sum of squared difference (SSD) scores. If the score of the second-best match point is higher than that of a threshold, we exclude it. Subsequently, we build an angle histogram based on the angles of optical flows at consecutive frames, then estimate the parameters for the GMM using the Expectation-Maximization (EM) algorithm [8] to divide the remaining optical flows into optical flows upon moving objects and errors. Note that the changes in the angle parameter are periodic and that there is an adverse situation in which 0 degrees and 360 degrees are both possible parameter values even though they yield the same angle. To avoid this, an angle is divided into two sub parameters as input data for the EM algorithm. In particular, when the periodic parameter is θ, sub parameters are given by $\theta_1 = sin\theta$, $\theta_2 = cos\theta$. Lastly, Gaussians whose mixing proportion is sufficiently-high will be chosen as flows upon moving objects in the scene.

An example of an angle histogram and the estimated GMM is shown in Figure 2. In this example, GMM containing four Gaussians is used to serve the segmentation purpose. Figure 3 also shows the segmentation results of optical flows. The left column of this figure shows original flows within a current frame, and

the right column shows segmented flows relative to the estimated GMM. The red and green flows mean segmented flows as moving objects. The black flow means error flows caused by the existence of similar patterns, and other flows mean error flows caused by other reasons such as occlusions and specular reflections.

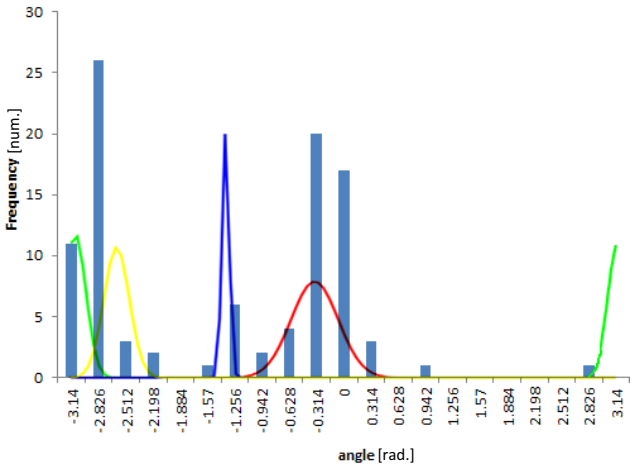

Figure 2. Angle histogram and estimated GMM

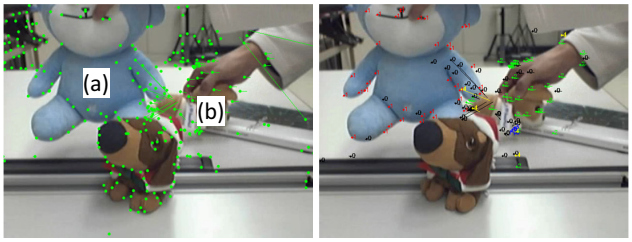

Figure 3. Segmentation result of optical flows observed by fixed camera. In this sequence, objects (a) and (b) moved to right and left, respectively.

2.3.3 Introducing outlier segmentation into the vSLAM process

In order to extend the optical flow segmentation described in Section 2.3.2 to the outlier segmentation observed by a motion camera in a dynamic scene, we use an estimated camera pose at the current frame. If no matching errors occur, re-projected 3-D map points that lie on the static scene are consistent with matched feature points but those that lie on moving objects are not, because the system's map remains unchanged even if moving objects exist in an actual scene. Therefore, we can treat the motion camera's flows in the same way as those of a fixed camera's when we re-project 3-D map points (time $t-n$) to the current frame (time t) by using the estimated current camera pose.

The following steps describe the frame-to-frame outlier segmentation method through Steps 1.5–1.6 in Table 1 in more detail (Steps 4–8 are the same as described in Sec.2.3.2.).

1. When the ratio of outliers to matched features exceeds 3%, run the following eight steps. The percentage, which judges whether moving objects is present, has decided through the experiment.
2. Again re-project 3-D map points that correspond to matched features judged as outliers by a robust estimator to a current frame by using the estimated current camera pose.
3. Create a flow vector $\mathbf{f}$ throughout all the outliers. $\mathbf{f}$ is defined by the following equation.

$$\mathbf{f} = \mathbf{x_{fp}} - \hat{\mathbf{x}}'_{\mathbf{fp}}. \quad (2)$$

   where $\mathbf{x_{fp}}$ is a position of matched feature, and $\hat{\mathbf{x}}'_{\mathbf{fp}}$ is the re-projected position by using estimated current camera pose.
4. Choose flow vectors whose magnitudes are larger than 0.
5. Eliminate error flows caused by the existence of similar patterns by using self-correlation in the current frame.
6. Build an angle histogram based on the remaining flow vectors.
7. Estimate the parameters for the GMM using the EM algorithm.
8. Choose Gaussians whose mixing proportion is more than a threshold as flow vectors upon moving objects.
9. Eliminate the 3-D map points that correspond to point-of-flow vectors upon moving objects.

It should be noted that using the robust estimator recursively would also achieve outlier segmentation, but the recursive process would consume much computational time. This is a critical problem in real-time camera pose estimation. In contrast, the EM algorithm can separate a number of objects and errors in one processing, and for this reason we use it.

3 Experimental results

We conducted an outlier segmentation experiment and a camera pose estimation experiment to confirm the feasibility of the proposed method. In these experiments, we captured a number of objects lying on a white table, and moved the camera and the objects on a planar surface. The scene also included a laminated business card that had specular reflection. In both experiments, we used a webcam(Logicool Qcam Pro 4000, 640x480, 30fps) whose intrinsic camera parameters were estimated by Zhang's method [9] in advance. We implemented the proposed method by improving PTAM [4] on a desktop PC with an Intel Core i7 3.33-GHz processor running on Windows 7.

3.1 Segmentation of outliers observed by a motion camera in dynamic scenes

Figure 4 shows the results of outlier segmentation. In this sequence, the camera moved to the left and an object moved to the right foreground. In the experiments, a GMM containing three Gaussians was used to serve the segmentation purpose. The red flow in the figure means segmented flow vectors as a moving object. The black flow means error flows caused by

the existence of similar patterns, and green and blue flows mean error flows caused by other reasons such as occlusions and specular reflections. Outliers segmentation could be performed in around 0.6ms/point, and it took less than 10ms/image. These experimental results indicate that the proposed method can segment outliers in the vSLAM process effectively and rapidly.

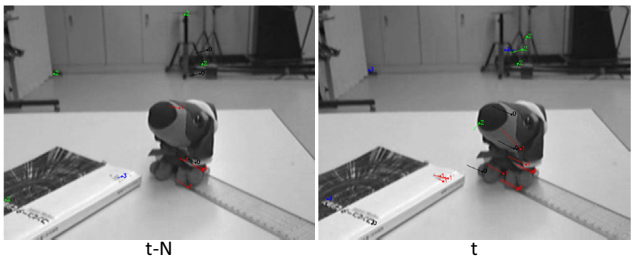

Figure 4. Outliers segmentation results. Red color means segmented flows as a moving object.

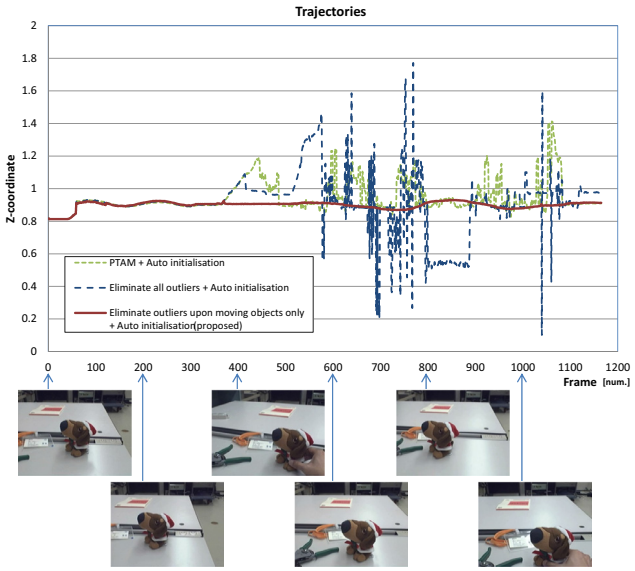

Figure 5. Comparison of camera pose estimation

3.2 Camera poses estimation in dynamic scenes

To evaluate the proposed method, we compared its results in an experiment with those of two other methods[1]. One is a method of modifying PTAM to remove the 3-D map and camera pose initialization described in Section 2.2. The other is a method that eliminates all outliers judged in the camera pose estimation process instead of only those in the object region. Figure 5 shows the resulting Z-coordinate trajectories of a 6-degree of freedom (DOF) camera pose and snapshots of an input sequence for this experiment. In this sequence, the camera moved on a planar surface whose normal vector is toward the Z-axis and the user moved an object. Therefore, the ground truth of this experiment is that the trajectory of Z-coordinate is constant. At about 400–550 frames, the user moved the object in front of the stopped camera. At about 1,000 frames–, both the camera and the object moved. Furthermore, this experimental scene had only a few textures and it included specular reflections' objects.

Throughout this experiment, the proposed method estimated the camera poses accurately, while both the methods used for comparison went into drift and relocalization. The modified PTAM technique would drift when an object moved because the method used the constraint that the scene was static. The method that eliminated all outliers would drift in the same way because there appeared to be few remaining feature points that resulted from eliminating feature points upon the specular object. These results indicate that regardless of even if a moving object is present, the proposed method estimates camera pose stably. Additionally, with our method the sequence could mostly be tracked in real-time.

4 Conclusion

This paper has presented a novel method for a vSLAM approach that is robust to dynamic scenes. The proposed method segments outliers of feature points determined at camera pose estimation steps. It detects features upon moving objects by building an angle histogram based on outliers, and by estimating parameters for the GMM using the EM algorithm. Only features upon moving objects are eliminated from outliers to prevent the drift and relocalization problem that is attributed to an insufficient number of features. Experimental results also indicate that the proposed method is both effective and robust vis-a-vis dynamic scenes.

As future work, we will improve EM algorithm to automatically detect the number of moving objects in a scene. We also intend to create a dynamic 3-D map by using estimated feature points upon moving objects for achieving virtual information overlay.

[1]These comparison results are shown in more detail in the supplemental video.

References

[1] E. Eade, et al.: "Scalable monocular slam," *in Proc. of CVPR*, pp.469–476, 2006.

[2] A. Davison, et al.: "MonoSLAM: Real-time single camera SLAM," *Trans. on PAMI*, vol.26, pp.1052–1067, 2007.

[3] D. Chekhlov, et al.: "Real-time visual SLAM using scale prediction and exemplar based feature description," *in Proc. of CVPR*, pp.1–8, 2007.

[4] G. Klein, et al.: "Parallel tracking and mapping for small AR workspaces," *in Proc. of ISMAR*, 2007.

[5] Z. Dong, et al.: "Keyframe-based real-time camera tracking," *in Proc. of ICCV*, pp.1538–1545, 2009.

[6] E. Rosten, et al.: "Machine learning for high-speed corner detection," *in Proc. of ECCV*, pp.430–443, 2006.

[7] R. Hartley, et al.: "Multiple view geometry in computer vision," *Cambridge Univ. Press*, 2nd edition, 2004.

[8] A. Dempster, et al.: "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc. B*, vol.39, pp.1–38, 1977.

[9] Z. Zhang: "A flexible new technique for camera calibration," *Trans. on PAMI*, vol.22, pp.1340–1334, 2000.