Reconstruction of Three Dimensional Spatial Clusters Using Monocular Camera

Kai Zhou, Michael Zillich and Markus Vincze
Institute of Automation and Control
Vienna University of Technology
Gusshausstrasse 27-29, A-1040 Vienna, Austria
{zhou, zillich,vincze}@acin.tuwien.ac.at

Abstract—Vision is an increasingly important sensor modality for mobile robots. Visual SLAM (simultaneous localization and mapping) has been used successfully for indoor and outdoor scenarios, where the focus typically lies in localizing the robot and the environment only serves to provide sparse landmarks for a large scale map. Structure from motion (SFM) techniques put emphasis on the actual fine-grained 3D structure of the environment, resulting in a denser more detailed map. For our mobile robot scenario we are interested in detecting objects in such a map. To this end we build on a state-of-the-art SFM technique and use it to build a series of maps, which we stitch together after correcting for scale and finally use the combined map to detect flat surfaces and objects resting on those surfaces.

Keywords—Simultaneous localization and mapping; Structure from motion; 3D reconstruction; 3D spatial clustering.

I. INTRODUCTION

Starting with pioneering work [1] in the 1990s, SLAM (Simultaneous localization and mapping) in robotics community and SFM (structure from motion) in computer vision community have both made impressive progress during the last two decades. Although these two technologies originally came from different scientific communities, the current trend of using monocular cameras as sensors makes them more and more similar. SLAM has been widely applied for mobile robots and autonomous vehicles in outdoor and indoor environments. In these scenarios the focus lies on robot localization and real-time capabilities and hence only a sparse set of easily distinguishable landmarks is used to build the maps; an accurate three-dimensional representation of the scene usually is not necessary or feasible. SFM on the other hand focuses on detailed scene reconstruction rather than observer motion. Classical SFM typically works offline in batch mode and uses computationally expensive techniques such as bundle adjustment. But ongoing improvements of computer hardware make it possible to start focusing on full three-dimensional scene reconstruction in real time. So the boundaries between SLAM and SFM start to blur.

We aim to build a system that allows a mobile robot to detect and eventually grasp generic 3D objects. We make the simplifying assumption that objects of interest rest on flat horizontal surfaces. This makes the otherwise very difficult task of segmenting arbitrarily shaped objects from the background tractable. We build upon a state-of-the-art SFM technique [4] and extend it to fit our particular needs. We build a series of fine-grained maps and stitch these together to form a combined map. As the scale of maps generated from SFM is undefined (moving a camera over a model landscape is indistinguishable from flying over a real landscape) we have to correct for the arbitrary scale of the partial maps. We then use the combined map to robustly detect the dominant horizontal plane and finally segment the objects resting on that plane.

The SFM technique we employ relies on tracking corner points in the images. Points matching over several images are used to build a 3D map and simultaneously estimate the camera motion. To provide at least a rough estimation of scale an initial map is built from two initial images using the 5-point stereo algorithm, where the motion between these images is assumed to be large enough (say 10 cm) and known approximately from an external source such as the robots odomotry. Successive camera motion is estimated per image using map points projected into the image and their matching detected corner points. At the same time the map is extended as new corner features enter the field of view. Some well-known problems arising in this context are:

- *Difficult initialization* Simultaneous estimation of camera pose from map points and building of the map from camera motion leads to a typical chicken-and-egg problem.
- *Expensive computation* The whole estimation process gets increasingly expensive as the map grows.
- *Enormous redundancy* All the captured landmarks usually are recorded and stored without efficient suppression of outliers.

The SFM method we employ handles these well for small AR workspaces. To adapt the system to our mobile manipulation scenarios we slightly extend it and then use the results to detect tables and objects resting on them.

II. RELATED WORK

3D scene reconstruction from vision data has a long tradition in computer vision. Binocular stereo is among the most widely used techniques. Two (or sometimes also three or even more) cameras with a fixed, known relative position allow 3D triangulation of points matched in both images, where the known epipolar geometry is used constrain the search for matching points. A long base-line between the cameras will lead to a more accurate 3D reconstruction but at the same time makes the matching of image points more difficult and computationally expensive. Some approaches use active sensors such as structured light or line lasers. These methods however suffer from susceptibility to strong external light sources and limited range. Using a moving monocular camera allows simple matching of image features as the base-line between consecutive pairs of images is small but offers an infinitely large base-line over the whole sequence of images and thus high accuracy.

Visual SLAM and SFM are two closely related techniques of that category.

SLAM refers to a technology usually used by mobile robots and autonomous vehicles to build up a map within an unknown environment while at the same time keeping track of their current position. To our knowledge, EKF-SLAM [1, 2] and FAST-SLAM [5, 6] are current state-of-the-art recursive approaches. Both of them use range sensors such as ultrasonic sensors or time-of-flight laser sensors and build a 2D ("bird's eye view") map. Extended Kalman Filters (EKF) were the first recursive estimators for estimating map structure and observer position over time. Montemerlo et al. [5, 6] substituted the EKF with a particle filter to increase robustness in ambiguous situations and scalability to large environments by taking advantage of the fact that landmarks become probabilistically independent of each other given a known observer position. This greatly reduces the dimensionality of the estimation problem and allows a larger number of landmarks.

More recently also cameras are used for SLAM. Davisons [3] "MonoSLAM" system shows the feasibility of real-time visual SLAM with a single camera. A FAST-SLAM based approach has also been successfully applied using of a single camera by E. Eade and T. Drummond [7].

Klein and Murray [4] present a novel method to estimate the camera pose and scene structure in an unknown environment, which is called Parallel Tracking and Mapping (PTAM). Tracking and mapping are separated into two threads, running in parallel on a dual-core computer. A brief subjective comparison of PTAM and MonoSLAM in [4, 8] demonstrates that PTAM has a better performance especially for larger maps.

The next section outlines the PTAM system of [4] on which our work is built. Subsequent sections describe in detail our extensions, present results, limitations and also future work.

III. PRELIMINARIES

This section describes the operation of PTAM. We assume that the camera has already been calibrated and all the internal parameters are available known. The system uses FAST-10 [9] corner features and an image pyramid for multi-scale feature detection. When the system is started, the five-point stereo algorithm [10] is employed to initialize the map similar to the processing described in [11-13]. For this the system uses two frames of the video stream assuming a known translation between these two frames (e.g. 10 cm) which can be provided by the robots odometry. So the initial map has a roughly correct scale. The system further consists of two threads, a tracking thread and a mapping thread, running in parallel on a dual-core computer.

Camera pose is estimated for each new image by the tracking thread, without updating the map (circumventing the chicken-and-egg problem mentioned above). The map is only updated when a new so-called key frame comes in. Keyframes are added only when the following conditions are satisfied:

- *Minimum time duration between keyframes* The time since the last keyframe was added must exceed some frames; we use a value of twenty frames.
- *Minimum distance between new features and the features of last keyframe* The minimum distance requirement avoids the common monocular SLAM problem of a stationary camera corrupting the map, and ensures a large enough stereo baseline for robust feature triangulation.
- *Minimum umber of new features* This requirement avoids the interference of noise which can be caused by some irregular tremble of the camera, e.g. for hand-held cameras.

With the features of the new key frame a computationally expensive bundle adjustment procedure is started to calculate 3D positions of matching features from the two key frames and the 3D points are added to the map. This procedure may run for considerable time (hundreds of milliseconds) but does so in its own mapping thread, ensuring that the tracking thread continues to updates camera poses at frame rate.

Figure 1 illustrates the flow of processing between the threads. Note that the colored blocks and especially the third controlling thread constitute our additions to PTAM to fit for our mobile manipulation scenario as explained in the next section.

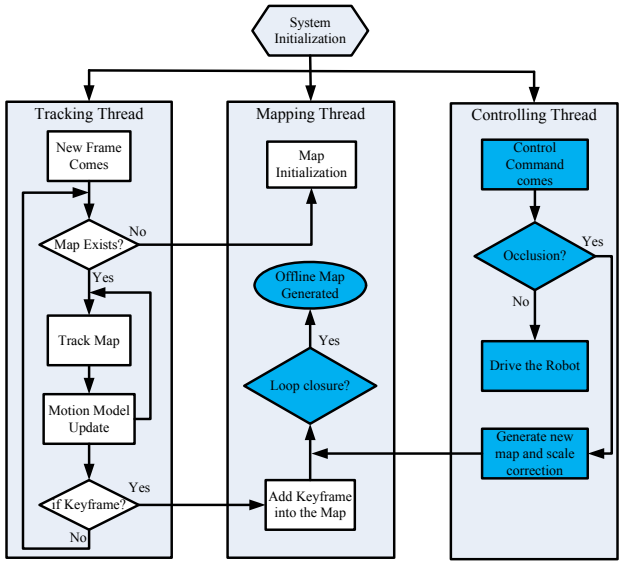

Figure 1. Detailed processing flow gragh of whole system, in the new controlling thread, if the potential occlusion is detected uses the scale correction, otherwise drive the robot normally

IV. MAP STITCHING

PTAM is well suited for augmented reality (AR) situations, where typically the user who wears the AR gear does not move around too much but rather pans and tilts the camera.

In order to reconstruct 3D model of objects, a classic solution is multi-view detection which usually uses a calibrated circular motion image sequence. But our system uses a monocular camera which is mounted on a small mobile robot. However we have a slightly different scenario, namely a mobile manipulation scenario with the following characteristics,

- Objects which the robot eventually wants to grasp and pick up rest on horizontal planes, such as tables or simply the floor.
- We require complete information of the objects, i.e. all sides.
- Therefore the robot needs to move around the table or some specific area on the floor to get different views of the objects.
- So the robot is essentially controlled to center the view on the table or specific area on the floor and then move around more or less in a circle.

In such situations PTAM often fails. We assume the reasons are:

- Maps are rather small and dense, i.e. consist mainly of stuff on the table and not much structure in the background. This seems not to be well suited to the bundle adjustment, which obviously would benefit from "wide" bundles for more robust camera pose estimates.
- PTAM does not model occlusion. If the camera mainly pans and tilts and essentially observes the scene from a stationary position, map points are either out of view (i.e. don't project on the current image) or are in view and visible, i.e. should find a matching image feature. If we however move around the objects, objects will start to self-occlude or occlude each other. So map points that are in view (do project on the current image) are actually not visible, because they are occluded. The system does not know about occlusion and thus vainly tries to match these projected map points with image features.
- With most of the map points on our highly textured objects this means that large parts of the map can become occluded and even though the features are distinctive and false matches thus should be rare, the increased number of false matches together with a small number of actually visible (non-occluded) map points seems to deteriorate estimation enough to cause failure.

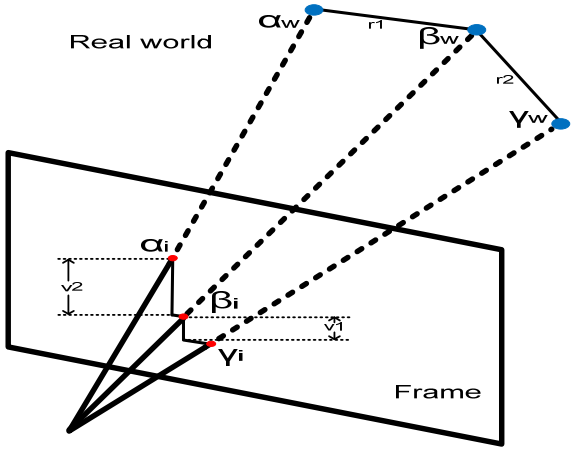

Figure 2. The demonstration graph of matching principle

Therefore we extend PTAM to create several smaller maps and stitch these together. Whenever the failure is detected, a new map should be generated using the following procedure:

- store the current map to disk
- use the current frame and the last key frame to re-initialize the system
- use the distance between camera poses of current frame and last key frame as the

baseline for the 5-point stereo algorithm used in PTAMs initialization

The re-initialization is not accurate enough due to the accumulative error of the pose estimation, so the scale differences in old and new map have to be fixed. Assuming a mostly horizontal movement, the matching points between last keyframe and current frame are searched using the following procedure:

1. Each feature point searches the two "vertically" nearest points in the projected image. Figure 2 shows how the search processes. β_i is the projective point of β_w on the image, and the "vertically" nearest two points are α_i and γ_i , because the vertical distance in the image is this principle for judging, it has been shown as $v1$ and $v2$. $v1/v2$ is stored as a parameter for each point, called vertical ratio.

2. Search the matched pairs of feature points in the old and new maps using vertical ratio as the criteria for judgment. Once the system finds in the new map one feature point that has the same vertical ratio with one point in the old map, it will start to check the nearby points using the same criteria. Using red points in figure 2 as an example, we can show the process. If $\beta_{i,new}$ is matched with $\beta_{i,old}$, $\alpha_{i,new}$ and $\gamma_{i,new}$ are the two "vertically" nearest points to $\beta_{i,new}$, the successful match is that both $\alpha_{i,new}$ is matched with $\alpha_{i,old}$ and $\gamma_{i,new}$ is matched with $\gamma_{i,old}$.

3. The corresponding points in the real world coordinate are used to calculate the ratio of scale correction. As shown in figure 2, α_w , β_w and γ_w are the corresponding points with α_i , β_i and γ_i , respectively. $r1$ and $r2$ are the distances between these three points in the real world coordinate. The values of $r1/r2$ in the new and old maps are utilized to calculate the ratio of scale correction.

V. DETECTION OF OBJECTS

Stitching together all sub-maps we get a combined map and use this for further processing. At first we remove "obviously" wrong points. Occasionally the map will contain point coordinates several orders of magnitude larger than the average (10^8 or something like that). These can be safely ignored. Then the dominant plane is computed using RANSAC, which we assume to be the table or the floor. Next we remove all points belonging to the dominant plane as well as all points below the plane. Finally we cluster the remaining points using the geometrical relationship among nearby points. These clusters are then considered object candidates to eventually be picked up (using of course a bigger robot than the one used in the current experiments).

VI. RESULTS AND LIMITATIONS

We show preliminary results using a small mobile robot (with quite terrible odometry) observing objects on the floor. The final system will be using a larger, more study robot equipped with a manipulator and elevated pan-tilt camera, able to look down on and drive around tables.

In Figure 3, (a) shows our mobile robot "Eddy" used in the experiments, (b) shows the scene of the experiment. The path of the robot is shown in (c) with the white line. (d) and (e) are snapshots of the system during initialization and tracking respectively. The comparison of the estimated trajectory of the robot and the ground truth is shown in (f), using green dots and blue line represent the estimated trajectory and the ground truth, respectively. (g) and (h) show the generated map. The green dots in (h) refer to the dominant plane which is the result of RANSAC. After the elimination of dominant plane points, the rest of the map points are grouped to the objects considering their geometrical relationship. Figure (i) shows the final result of the separated objects with different colors.

As the system relies on tracking feature points, we obviously require the table and objects to contain sufficient texture, which is not a problem for many common household objects. PTAM itself already can handle maps with thousands of features. Adding our stitching procedure further improves scaling to larger maps.

VII. CONCLUSIONS AND OUTLOOKS

This paper presented a system that is capable of recovering textured objects resting on tables using a moving monocular camera. A state-of-

the-art SFM technique was extended to fit our needs for mobile manipulation scenarios, namely the combination of sub-maps into a combined map where a scale correction step was introduced in the stitching procedure. Estimating the table plane then allows segmentation of the objects resting on the table.

At present, the robot odometry is only used during re-initialization for getting an approximate base-line for the stereo image pair. In the future we plan to fuse the pose estimates from visual tracking and robot odometry to increase robustness against temporary tracking failure.

ACKNOWLEDGEMENT

This work was supported by the EU FP7 IST Cognitive Systems Integrated Project "CogX" ICT-215181-CogX. The author Kai Zhou gratefully acknowledges the support from the China Scholarship Council (CSC).

REFERENCES

[1] R. Simith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," Autonomous Robot Vehicles, Springer 1990

[2] G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "An Experimental and Theoretical Investigation into Simultaneous Localisation and Map Building," The Sixth Intl. Symposium on Experimental Robotics VI, pp. 26–274. IEEE Press, Sydney 1999

[3] A. Davison, "Real time simultaneous localisation and mapping with a single camera," ICCV 2003, Nice 2003

[4] G. Klein and D. W. Murray, "Parallel tracking and mapping for small AR workspaces," Proc. IEEE/ACM 6th Int. Symp on Mixed and Augmented Reality, Nara 2007

[5] M. Montemerlo, and S. Thrun, "Simultaneous localization and mapping with unknown data association using fastslam," Proc. of IEEE Intl. Conf. on Robotics and Automation, Taipei 2003

[6] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," IJCAI, Acapulco, Mexico 2003

[7] E. Eade, and T. Drummond, "Scalable monocular slam," Proc. IEEE Intl. Conference on Computer Vision and Pattern Recognition (CVPR06), pp. 469–476. New York 2006

[8] R. Castle, G. Klein, and D. Murray, "Video-rate Localization in Multiple Maps for Wearable Augmented Reality," Proc. Intl. Symposium on Wearable Computers (ISWC08), Pittsburgh 2008

[9] E. Rosten, and T. Drummond, "Machine learning for high-speed corner detection," Proc. 9th European Conference on Computer Vision ,Graz 2006

[10] H. Stewenius, C. Engels, and D. Nister, "Recent developments on direct relative orientation," ISPRS Journal of Photogrammetry and Remote Sensing, pp. 60:284–294 2006

[11] D. Nister, O. Naroditsky, and J. R. Bergen, "Visual odometry," Proc. IEEE Intl. Conference on Computer Vision and Pattern Recognition (CVPR04), pp. 652–659, IEEE Computer Society. Washington D.C. 2005

[12] E. Mouragnon, F. Dekeyser, P. Sayd, M. Lhuillier, and M. Dhome, "Real time localization and 3d reconstruction," Proc. IEEE Intl. Conf. on Computer Vision and Pattern Recognition, New York 2006

[13] C. Engels, H. Stewenius, and D. Nister, "Bundle adjustment rules," Photogrammetric Computer Vision (PCV'06) 2006

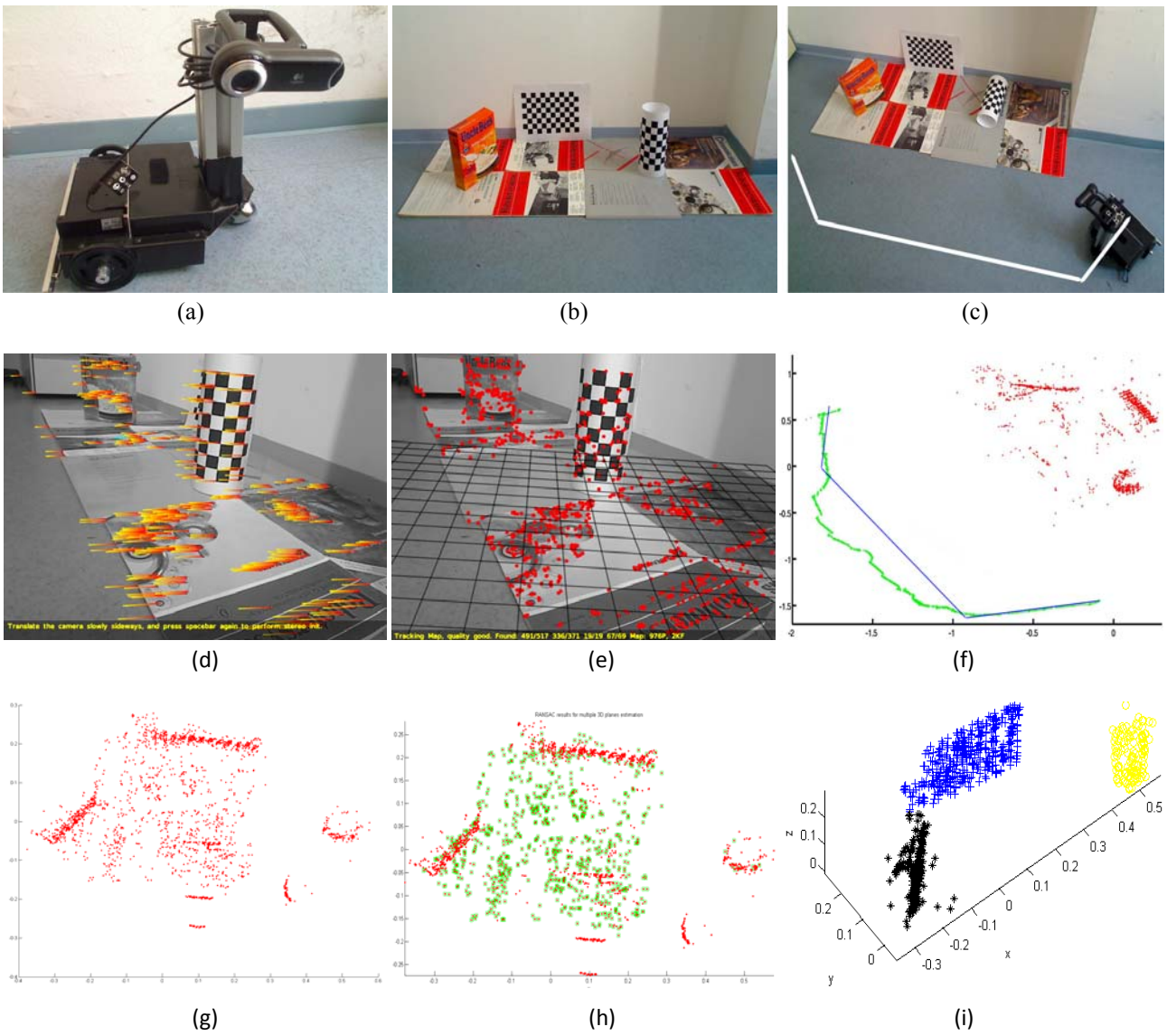

Figure 3. The experiment setting, results and comparison