Submitted (12/9/2008) to
2009 IEEE Int Conf on Robotics and Automation, Kobe, Japan

Avoiding moving outliers in visual SLAM by tracking moving objects

Somkiat Wangsiripitak and David W Murray

***Abstract*— To work at video rate, the maps that monocular SLAM builds are bound to be sparse, making them sensitive to the erroneous inclusion of moving points and to the deletion of valid points through temporary occlusion. This paper describes the parallel implementation of monoSLAM with a 3D object tracker, allowing reasoning about moving objects and occlusion. The SLAM process provides the object tracker with information to register objects to the map's frame, and the object tracker allows the marking of features, either those on objects, or those created by their occluding edges, or those occluded by objects. Experiments are presented to verify the recovered geometry and to indicate the impact on camera pose in monoSLAM of including and avoiding moving features.**

I. Introduction

The various techniques of simultaneous localization and mapping [1] [2] [3] lie at the core of many successful autonomous robotic navigation systems: those firmly grounded (e.g. [4]) where laser sensing predominates, those on and in the water (e.g. [5] [6]) using radar and sonar respectively, and those airborne (e.g. [7]) where radar and 2D vision are most used.

Our interest here however lies in using SLAM with hand-held and body-worn cameras. The recovered camera location and map are used not so much for navigation — one can rely on the camera's user to get around — as for graphical augmentation of the scene, where a sense of what is where becomes more important (with emphasis added on the what).

A common assumption in applications of SLAM is that the sensor moves in a rigid environment, and the accidental inclusion of moving features into the processing can degrade performance substantially and vice versa [8]. Several approaches to handling movement in the environment have been reported (e.g. [9] [10]), with all relying on comparison with the uncertainty of prediction of the innovation between predicted and actual sensor measurements, given the motion model of the sensor. The better the odometry available, say when the sensor is on a comprehensively instrumented vehicle, the more certain the resulting segmentation between static and moving features. However, for a hand-held or wearable camera good odometry is often unavailable, and the motion of the camera, though usually and optimistically modelled as constant velocity, is often rather haphazard, making it the more likely that outlying moving points will be included.

The authors are with the Department of Engineering Science, University of Oxford, Oxford OX1 3PJ, UK. Web: www.robots.ox.ac.uk/ActiveVision Email: somkiat,dwm@robots.ox.ac.uk. This research was part supported by Grant GR/S97774 from the UK Engineering and Physical Science Research Council, and by a Thai Government scholarship to SW.

Recent work using wearable cameras in our laboratory [11] [12] has used Davison's single camera approach to localization and mapping [13] [14]. In monoSLAM, even though a single motion model generates all predictions, decisions about correspondence are then taken individually. To avoid mismatches, a quite cautious process of active search is used which often leads to sparse maps, in turn making avoiding mismatches even more important. One different approach to removing mismatches is to be less cautious proposing matches, and then seek rigidly moving points by collective fitting of multifocal tensors using robust methods. Here however we adopt another approach to removing outliers by tracking known 3D objects in the scene, determining whether they are moving, and then using their convex hulls to mask out features.

The ability to track objects is useful not only for deleting moving features, but also for extending the lifetime of features which become occluded. It is common for map features that are lost from the image for several frames to be regarded as unreliable and removed from the map. Knowledge that they are occluded rather than unreliable avoids the need to invoke the somewhat cumbersome process of feature deletion, followed later perhaps by unnecessary re-initialization.

The emphasis on objects is, as our earlier comment suggests, well-suited to wearable vision, but has also been proposed in a different context by [15]. Those authors noted that movement is likely to associated with objects in the scene, and classified them according to the likelihood that they would move. The data used for SLAM was from laser range sensors, and objects were detected and identified using radio-frequency identification tags, allowing computational effort to be concentrated in a particular region to determine whether or not there was motion .

In previous work [16] one of us has reported the automatic identification of 2D objects and their use to augment SLAM maps, both geometrically and semantically. For 3D objects the problems of identification and pose initialization are rather harder, and here we consider only the use of 3D objects for reasoning about motion segmentation and occlusion. The 3D object tracker used is a modified version of Harris' RAPiD [17].

After defining frames of reference, Sections II-A and II-B give a brief outline of the underlying SLAM and tracking processes. Their combination, in which object motion is detected and object pose recovered in the map's frame allowing occlusion masks to be set, is proposed in Section III. Experiments and results are described in Section IV, and the

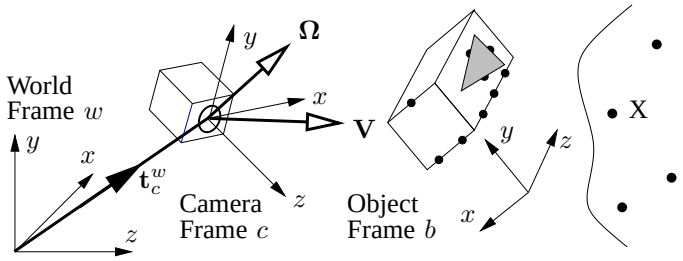


Fig. 1. The world frame w is that in which background points are stationary. The moving camera has a frame c attached to it. Each object is modelled within its own coordinate frame b as a set of control points lying on edges, which may be crease or albedo edges.

paper ends with a discussion of future work.

II. Underlying processes

As illustrated in Fig. 1, points X in the scene (which may include points on the static background and any object) are observed by a moving camera whose optic centre and image plane define the coordinate frame c. The camera's pose at some time is one or other representation of the six degrees of freedom in $[\mathtt{R}^{wc}|\mathbf{t}_c^w]$, which defines the relationship between the coordinates of X referred to the world and camera frames as $\mathbf{X}^w = \mathtt{R}^{wc}\mathbf{X}^c + \mathbf{t}_c^w$, where $\mathtt{R}^{wc}$ is a rotation matrix and $\mathbf{t}_c^w$ is the origin of the camera's frame referred to the world frame.

A. Visual SLAM

Monocular visual SLAM introduced and developed by Davison and colleagues [13] [14] uses an EKF to map a sparse set of natural features in the scene. The camera may move freely, and (typically) no odometry is available, making the localization more involved than that for a robot moving on a surface. As the filter must complete within the constant inter-frame time, a fixed bound is placed on the maximum map size. The use of several tens of features is typical for video rate operation on current processors.

The state $\mathbf{p}_k$ of the system at timestep k comprises a description of the camera $\mathbf{C}$ and the coordinates of $N(k)$ 3D points referred to the world frame

$$\mathbf{p}_k = \left[\mathbf{C},\ \mathbf{X}_1^w, \ldots \mathbf{X}_{N(k)}^w\right]_k \ .$$

The points are assumed to be stationary, so that prediction of the prior at timestep $k+1$ is idempotent, and no noise is added to their associated part of state covariance $\mathtt{P}_k$ during the prediction stage of the filter.

In the camera state, orientation is represented by a unit quaternion $\mathtt{q}^{wc}$, the position is that of the camera's optic centre in the world frame, $\mathbf{t}_c^w$, and the instantaneous angular and rectilinear velocities vectors $\mathbf{\Omega}^c$ and $\mathbf{V}^w$ are referred to the camera and world frames, respectively. The motion model is that of constant velocity, so that prediction of the prior over an interval Δt is given by

$$\mathbf{C}_{k+1} = \begin{bmatrix} \mathtt{q}^{wc} \\ \mathbf{t}_c^w \\ \mathbf{\Omega}^c \\ \mathbf{V}^w \end{bmatrix}_{k+1} = \begin{bmatrix} \mathtt{q}^{wc} \times \mathtt{q}\left(\mathbf{\Omega}^c \Delta t\right) \\ \mathbf{t}_c^w + \mathbf{V}^w \Delta t \\ \mathbf{\Omega}^c \\ \mathbf{V}^w \end{bmatrix}_k .$$

As new structure appears or old structure disappears, feature estimates $\mathbf{X}_i^w$ can be added to or deleted from the map as required. Otherwise, the cycle of prediction, measurement, and update in monoSLAM follows that of a standard EKF [18].

The measurements themselves are the image positions of features located using the detector of Shi and Tomasi [19]. Predicted measurement positions are generated by projecting all potentially visible 3D features in the current map into the image space using the predicted camera pose and calibrated values for the camera instrinsics and radial distortion parameters. Image search to establish an actual match occurs within the 3σ limit of the projected prior covariance, and uses sum of squared differences between the current image and an 11×11 image patch which is stored as a descriptor for each map point.

Of importance here is the process of deleting a feature. Standard monocular SLAM takes no account of occlusion. It assumes a transparent world and the criterion for deleting an observed feature resorts to using a threshold ratio of actual observability to predicted observability over the long term (the usual ratio is 50%). As well as the informational and computational costs associated with deleting a feature, there is the associated cost which arise from the need, or pressure, to replace it with a new feature.

B. Object pose tracking

The feature density that can be sustained by monoSLAM at video rate is far too low to indicate much about surfaces, let alone objects. Instead a separate object tracker is used — here a modified version of Harris' RAPiD tracker [17], [20].

As illustrated in Fig. 1, three-dimensional polyhedral objects are modelled by a set of control points which lie on edges. These may be either surface creases or surface albedo markings, edges fixed to the geometry that allow the projections to be detected in the image. RAPiD makes the assumption that the pose change required between current and new estimates is sufficiently small, first, to allow a linearization of the solution and, second, to make trivial the problem of inter-image correspondence. The correspondences used are between predicted point to measured image edge, allowing search in 1D rather than 2D within the image. This makes very sparing use of image data — typically only several hundred pixels per image are addressed.

Each object's geometry is described in its local frame b. The object's pose $[\mathtt{R}^{wb}|\mathbf{t}_b^w]$ is such that a point on the object is given in the world frame by

$$\mathbf{X}^w = \mathtt{R}^{wb}\mathbf{X}^b + \mathbf{t}_b^w = \mathbf{X}^a + \mathbf{t}_b^w .$$

If the object's angular and rectilinear velocities are $\mathbf{\Omega}$ and $\mathbf{v}$, in a small time Δt the object will move to a new position $\mathbf{X}^{w\prime} = \mathbf{X}^a + (\mathbf{\Omega}\Delta t)\times\mathbf{X}^a + \mathbf{t}_b^w + \mathbf{v}\Delta t$. Writing the change in pose as $\Delta\mathbf{s} = \Delta t[\mathbf{v}^\top, \mathbf{\Omega}^\top]^\top$ and composing the matrix $\mathtt{G} = \left[\mathtt{I}_3|\mathbf{X}_\times^a\right]$ the new position can be written as $\mathbf{X}^{w\prime} = \mathbf{X}^w + \mathtt{G}\Delta\mathbf{s}$, and transformed into the camera frame as

$$\mathbf{X}^{c\prime} = \mathbf{X}^c + \mathtt{R}^{cw}\mathtt{G}\Delta\mathbf{s} \ .$$

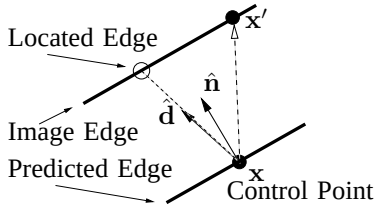


Fig. 2. Because of the aperture problem, only the perpendicular distance along $\hat{\mathbf{n}}$ is measurable. It is not necessary to search for the edge along $\hat{\mathbf{n}}$: it is quicker to search along one of the eight cardinal directions $\hat{\mathbf{d}}$, here a diagonal.

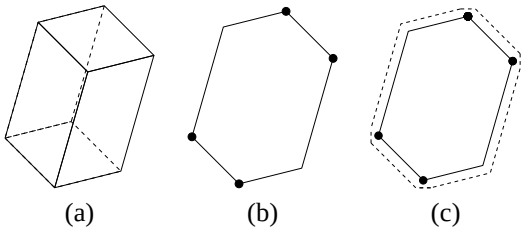


Fig. 3. The process of creating the extended convex hull of 3D object. (a) The projection of visible corner points of 3D object. (b) The 2D convex hull of the object is found by gift wrapping from one of the extreme point. (c) The convex hull is uniformly dilated by an amount that corresponds to the projection of the typical change in pose.

Projecting into the camera, and assuming the depth change is small, gives the movement of projected point as (e.g. [21])

$$(\mathbf{x}' - \mathbf{x}) \approx \frac{1}{Z^c} \left[\mathtt{R}^{cw} \mathtt{G} \Delta\mathbf{s} + ([001] \mathtt{R}^{cw} \mathtt{G} \Delta\mathbf{s}) \mathbf{x} \right] \ .$$

This equation can be used to recover the change of pose when point to point matches are available. However, as noted earlier, it is more usual to use control point to line matches, measuring distances along the cardinal direction $\hat{\mathbf{d}}$ closest to the perpendicular to a line, as sketched in Fig. 2. The measurement equation becomes

$$\hat{\mathbf{d}}^\top (\mathbf{x}' - \mathbf{x}) = \frac{1}{Z^c} \hat{\mathbf{d}}^\top \begin{bmatrix} 1 & 0 & x \\ 0 & 1 & y \end{bmatrix} \mathtt{R}^{cw} \mathtt{G} \Delta\mathbf{s} \ ,$$

and the optimum change in pose found as

$$\Delta\mathbf{s}^* = \arg\min_{\Delta\mathbf{s}} \sum_i \hat{\mathbf{d}}_i^\top (\mathbf{x}' - \mathbf{x})_i \qquad (1)$$

using least squares. In practice a correction for radial distortion is applied.

Three difficulties using the Harris tracker were summarized in [21]. First, it uses so few pixels that edge location is highly sensitive to changes in lighting, shadows, background texture, and occlusion. Second, linearization can leave a single-shot computation of pose change far from the optimum. Third, the accuracy of the pose recovered in a single camera shows correlation between lateral translation and rotation about an orthogonal axis parallel to the image plane. There is little that can be done about the last using a single camera. The second is readily solved by iteration [21], and typically two iterations suffice. There are several palliatives for the first, some of which are suggested elsewhere [22][21]. For control points on albedo edges, to avoid mismatching search is made for image edge polarities which match those built into the model. For those lying on non-occluding crease edges, search uses a short-term memory of the previously observed edge magnitude and polarity captured after posed has converged, and for occluding edges only edge magnitude is involved. Matches involving faces that are close to edge-on to the camera are down-weighted. In addition, multiple control points are used on each edge and RANSAC [23] applied to test for collinearity amongst located edge positions from controls points on the same model edge [22]. Finally the solution to Eq. (1) is subjected to another robust method, this time least median of squares as the underlying standard deviation is unknown [24].

III. MonoSLAM with tracked objects

The two processes of object tracking and SLAM just described run largely independently, but there are two stages of interaction, the first essential to the RAPiD tracker, the other beneficial to monoSLAM.

A. Information from SLAM to the object tracker

The description of the RAPiD object tracker refers to $\mathtt{R}^{cw}$ and $\mathbf{t}_b^w$. Unfortunately, in the absence of some additional pre-calibration or reference object, the world frame is undefined in a stand-alone implementation, and all that can be recovered is relative pose between the camera and object — in effect, the world frame is replaced by the camera frame itself so that $\mathtt{I} \leftarrow \mathtt{R}^{cw}$ and $\mathbf{t}_b^c \leftarrow \mathbf{t}_b^w$.

However, in this work visual SLAM establishes the world frame and at each frame provides the object tracker with the pose of the camera relative to it, and objects become immediately registered with the 3D map. The SLAM process is started first in a static part of the scene, and once a portion of the map is stable, object tracking commences. The important issues of identification and pose initialization are not addressed here: at present these are done by hand.

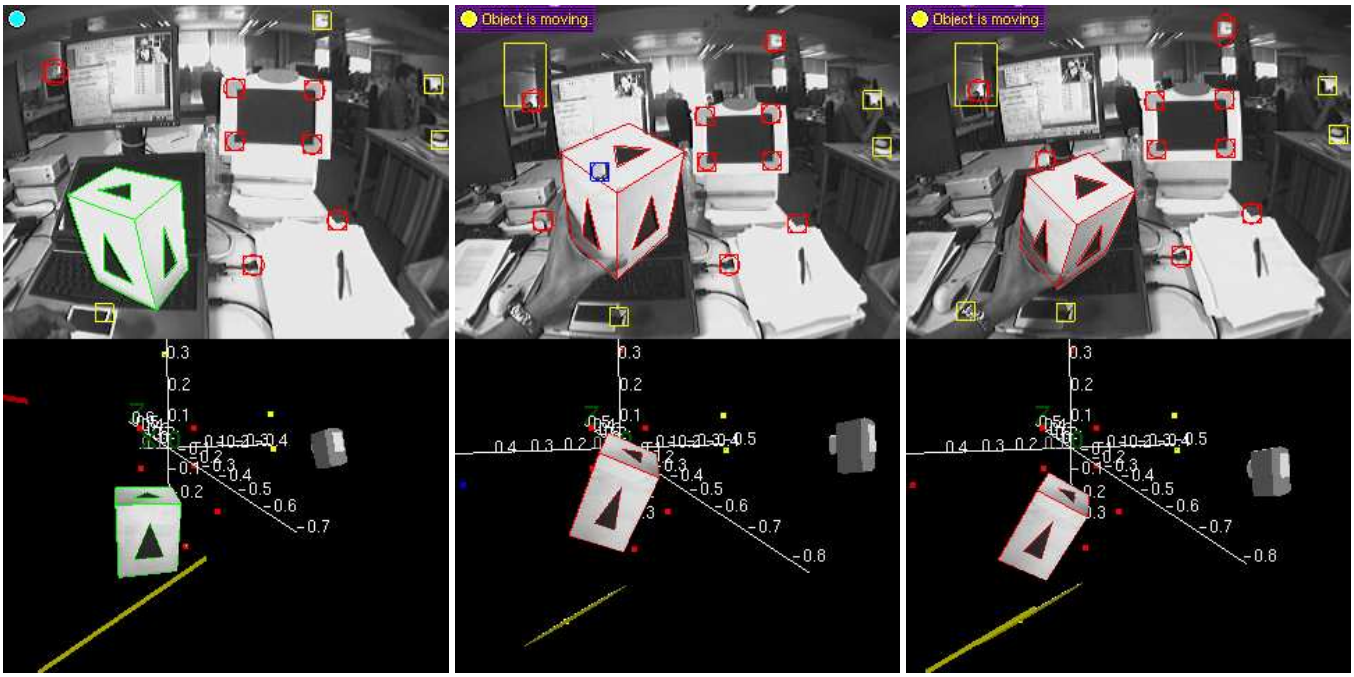
B. Information from the object tracker to SLAM

For each incoming frame, a stand-alone implementation of monoSLAM performs the prediction of prior camera pose, predicts the image projections from the map (determining visibility only by clipping to the image size), finds the innovation, and updates the state of the camera and map features. Here two extra checks are made on the list of features projected into the image.

As sketched in Fig. 3, the corner points of each object being tracked are projected into the image using the current pose of the camera. The 2D convex hull of each is found using gift wrapping [25], and then dilated by an amount to take into account likely image movement due to pose change and to ensure features on occluding edges are marked as such.

The enclosed region is then used as a mask in the following ways:

1) If the object was found by the tracker to be moving, any feature *on* the object that has been detected by Shi-Tomasi, and thence might be included in the map, is deleted.



(a) (b) (c)

Fig. 4. Images and maps from stages in the processing. (a) Early in the processing: the map is established, and the object is tracked but found to be stationary. (b) The object is now detected as a moving, and one map feature is found to be occluded by the object. (c) The object is still moving, but the map feature is no longer occluded but still remains in the map and is immediately re-used. The lower panels show the map point positions and covariances, and the camera's and object's pose. The occluded point is just visible in the colour versions of the maps at the very left hand edge of the lower panels of (b) and (c).

2) Whatever the state of motion of the object, any feature computed close to the occluding boundary is deleted.
3) Whatever the state of motion of the object, any feature predicted to be occluded is marked as such, and is excempted from the usually visibility tests when considering whether to delete a feature.

The first test is an expedient way of allowing features on stationary objects to be included in the map until they move. Their deletion from the list of observed features will lead automatically to their being deleted from the map in monoSLAM's usual way.

The second is a method of supressing the usual monoSLAM rules for deletion. As noted earlier, all standard features in the traditional SLAM are deleted when its long-term ratio of actual to predicted observabilities becomes low, where the actual observability is the number of frames that the feature has actually been measured successfully since it is initialised, while the predicted observability is the number of frames that the predicted projection of the map feature by the camera lies within the image rectangle. Being able to mark a feature as occluded saves time searching for it in the image and does not needlessly reduce the visibility ratio.

IV. Experiments and Results

A. Timing

The object tracker and SLAM methods have been implemented as threaded processes in C++, and, in the experiments reported here, run one on each core of an Intel Pentium Core 2 Duo 1.83 GHz Processor. Grey level images of size 320×240 pixels are provided at 30 Hz from a CMOS camera over a Firewire link, and are read by both threads. In the development system, a considerable part of the available 33 ms is taken up with graphical operations and operating system overheads, around 18 ms in all. Monocular visual SLAM with around 20 point features takes approximately 5 ms, leaving some 10 ms per frame to pursue object tracking. Around 240 control points are used on the box object (size $0.14 \times 0.18 \times 0.11\,\mathrm{m}^3$) which has 12 crease and 18 albedo edges and each iteration takes some 5 ms, allowing two iterations to complete.

B. Occlusion masking

Fig. 4 illustrates three significant stages during a run. Early on during the processing, the map has been established, and the object inserted and tracked. However, it is found to be stationary. Later on, in sub-figure (b), the object is detected to be moving, and a 3D map feature (not present in (a)) has been marked as occluded. It is therefore not deleted from the map, and in (c) the object is still moving but the map feature reappears after occlusion, and is reused immediately.

C. Effect on monoSLAM of avoiding mismatches

To demonstrate the effect that just one mismatch can have on camera pose in monoSLAM Fig. 5 compares the six components of camera pose recovered over time by monoSLAM with and without occlusion detection using the object tracker. At around frame 450 a moving features is included in the processing.

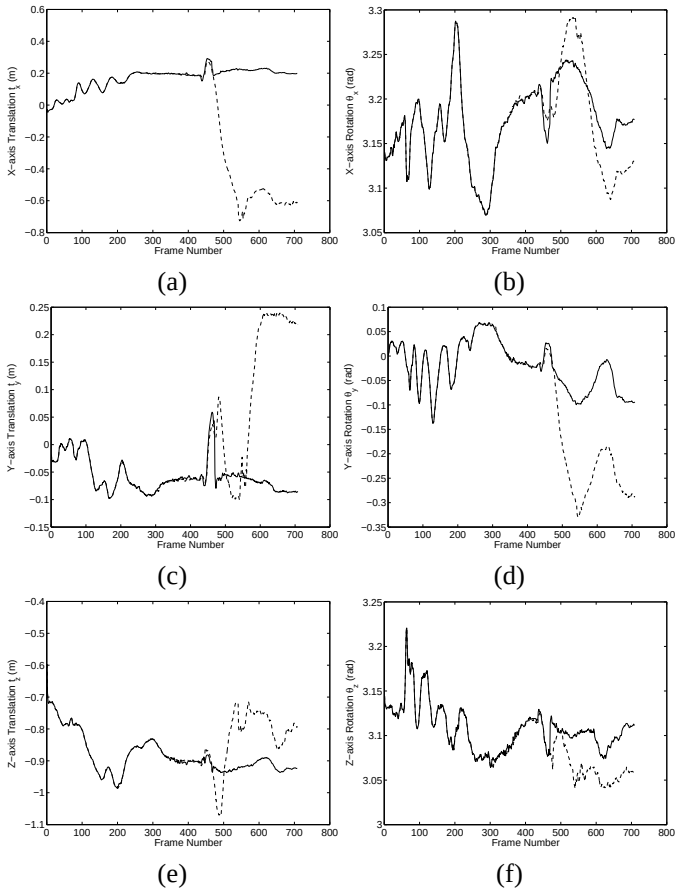


Fig. 5. Estimated position and orientation of the camera along and about the x-, y-, and z-axes with (solid) and without (dashed) occlusion detection.

TABLE I
TRACKING DRIFT

	Drift in Translation (mm)	Drift in Rotation (degree)
x-axis	3	2
y-axis	4	2
z-axis	13	1

D. Tracking drift

Fig. 6 shows snapshots from an experiment to measure tracking drift. MonoSLAM's calibration tile is adhered to a window and establishes the world frame with the y-axis upright in the scene, the x-axis horizontal to the left, and the z-axis mutually perpendicular and pointing away into the scene. While the camera was moved about arbitrarily, the object was first rotated clockwise then anticlockwise around the y-axis returning to its original position. Then the object was translated to and fro along the x-axis. Fig. 6(a) shows that before the object is detected two of the 3D map features are on the stationary object (one on a crease edge, the other on an occluding edge). After the object is instantiated in (b) its dilated convex hull is used to delete both features. Fig. 6(c) illustrates the prevention of initialization of potential map feature on an object which is moving, and (d) illustrates that the modified RAPiD is itself somewhat robust

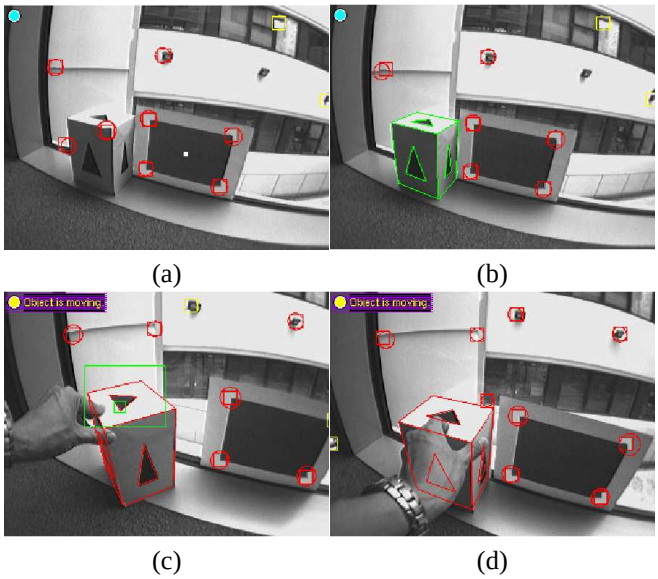


Fig. 6. (a) Before the insertion of the object into the map. (b) The object feature and the object boundary feature are deleted after the object is manually inserted. (c) The object feature is detected but not initialised as it belongs to the moving object, (d) The tracking of object is itself tolerant of occlusion.

to occlusion. Tracking drift was evaluated by examining the object poses before and after movement forward and backward to the original position, and is shown for each axis in Table I. Note that these drift is a combination of boths drifts in the camera pose estimated by SLAM and that in the object relative to the camera.

E. Correlation between lateral translation and rotation

Although the drift is small returned to that same position, the accuracy measuring the absolute change between two distinct poses is less so. The pose data from the translation phase of the previous experiment is shown in Fig. 7. The translation changes along the y- and z-axes are suitable small (c and e), as are the rotations about the x- and z-axes (b and f). However the correlation noted in [21] is quite apparent between (a and d), translation along x and the perpendicular in-plane axis, y. This partial ambiguity in the back-projection into the scene is inherent in the image data: the converse is that the forward projection is not much affected by error in the absolute pose so that, importantly, neither the ability to track, nor the ability to delineate the convex hull is affected. (Note that the offsets in (a), (c) and (e) of around 220 mm, -10 mm and -60 mm are as expected. The origins of the object and calibration tile are at their respective centres.)

V. CONCLUSIONS AND FUTURE WORK

This paper has presented a method of including information about camera pose from monocular visual SLAM into an 3D objects tracker, which in turn is used to prevent SLAM (i) incorporating moving features into its map and (ii) deleting features that are known to be occluded by objects. The paper has presented experiments both to illustrated the operation of the combined processes, to verify geometrical

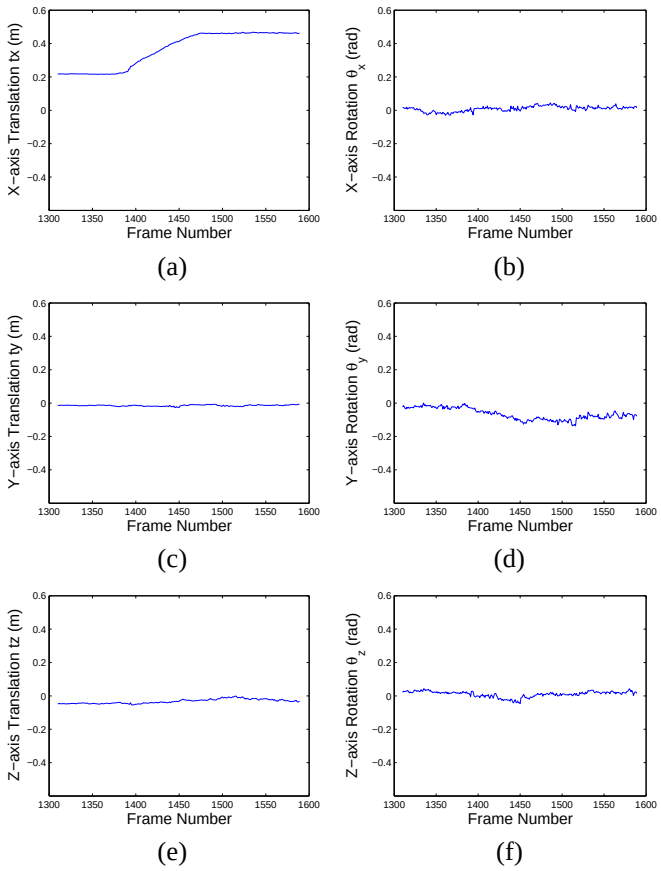


Fig. 7. Estimated position and orientation of the object along and about the x-, y-, and z-axes while the object performs pure translation along the x-axis only. There is correlation evident between (a) translation along x and (d) rotation about y.

integrity of the recovered 3D information, and to indicate the impact on camera pose in monoSLAM using a sparse map of not avoiding moving features. Avoiding moving points, and allowing stationary but temporarily occluded points to survive for immediate reuse once visible again makes monoSLAM more robust to dynamic environments, and avoids unnecessary computation.

Our focus here has been on a geometrical approach in 3D, chiefly to show that information can be exchanged both ways between SLAM and the object tracker. However, althought polyhedron tracking based on edges may be a useful tool in some metrological applications, one must look elsewhere to make this more widely applicable, and our aim in future work is to move to a point-based object tracker where such features can also act as descriptors to allow recognition, pose initialization and tracking, and to allow perhaps a more fundamental integration of the two tasks. Automatic recognition is key to making useful the categorization of objects according to their likelihood of motion, of the sort proposed by [15], which would in turn allow reliably stationary objects to be incorporated into the map. Following the experiment shown in Fig. 5, we note that information about the boundaries of static objects is always useful as it prevents the inclusion of features which appear and move artificially along occlusion boundaries.

REFERENCES

[1] R. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *International Journal of Robotics Research*, vol. 5, no. 4, pp. 56–86, 1986.

[2] J. Leonard, H. Durrant-Whyte, and I. Cox, "Dynamic map building for an autonomous mobile robot," *International Journal of Robotics Research*, vol. 11, no. 8, pp. 286–298, 1992.

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge MA: The MIT Press, Sep 2005.

[4] J. Nieto, J. Guivant, and E. Nebot, "Denseslam: Simultaneous localization and dense mapping," *International Journal of Robotics Research*.

[5] M. Benjamin, J. Curcio, J. Leonard, and P. Newman, "Navigation of unmanned marine vehicles in accordance with the rules of the road," in *Proc 2006 IEEE Int Conf on Robotics and Automation*, 2006.

[6] R. Eustice, H. Singh, J. Leonard, M. Walter, and R. Ballard, "Visually navigating the RMS Titanic with SLAM information filters," in *Proc Robotics: Science and Systems, Cambridge MA*, 2005.

[7] J. Langelaan and S. Rock, "Passive GPS-free navigation for small UAVs," in *IEEE Aerospace Conference*, 2005, pp. 1–9.

[8] C.-C. Wang and C. Thorpe, "Simultaneous localization and mapping with detection and tracking of moving objects," in *Proc 2002 IEEE Int Conf on Robotics and Automation*, vol. 3, 2002, pp. 2918–2924.

[9] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas," in *Proc 2003 IEEE Int Conf on Robotics and Automation*, vol. 1, 2003, pp. 842–849.

[10] D. Wolf and G. S. Sukhatme, "Online simultaneous localization and mapping in dynamic environments," in *Proc 2004 IEEE Int Conf on Robotics and Automation*, 2004, pp. 1301–1307.

[11] A. J. Davison, W. W. Mayol, and D. W. Murray, "Real-time localisation and mapping with wearable active vision," in *Proc 2nd IEEE/ACM Int Symp on Mixed and Augmented Reality, Tokyo, Japan, Oct 7-10, 2003*, 2003.

[12] W. W. Mayol, A. J. Davison, B. J. Tordoff, and D. W. Murray, "Applying active vision and slam to wearables," in *Robotics Research: The Eleventh International Symposium, Siena Italy, October 19-21, 2003 (Springer Tracts in Advanced Robotics*, P. Dario and R. Chatila, Eds., vol. 15. Springer, 2005, pp. 325–334.

[13] A. J. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Proc 9th IEEE Int Conf on Computer Vision*, 2003.

[14] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 1052–1067, 2007.

[15] H. Zhou and S. Sakane, "Localizing objects during robot SLAM in semi-dynamic environments," in *Proc of the 2008 IEEE/ASME Int Conf on Advanced Intelligent Mechatronics*, 2008, pp. 595–601.

[16] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray, "Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras," in *Proc 2007 IEEE Int Conf on Robotics and Automation*, 2007, pp. 4102–4107.

[17] C. Harris and C. Stennett, "Rapid - a video rate object tracker," in *Proc 1st British Machine Vision Conference*, Sep 1990, pp. 73–77.

[18] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*. Academic Press, 1988.

[19] J. Shi and C. Tomasi, "Good features to track," in *Proc 16th IEEE Conf on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[20] C. Harris, "Tracking with rigid models," in *Active Vision*, A. Blake and A. Yuille, Eds. MIT Press, 1992, pp. 59–73.

[21] R. L. Thompson, I. D. Reid, L. A. Munoz, and D. W. Murray, "Providing synthetic views for teleoperation using visual pose tracking in multiple cameras," *IEEE Transactions on Systems, Man and Cybernetics, Part A*, vol. 31, no. 1, pp. 43–54, 2001.

[22] M. Armstrong and A. Zisserman, "Robust object tracking," in *Proc 2nd Asian Conference on Computer Vision, 1995*, vol. I. Springer, 1996, pp. 58–62.

[23] M. Fischler and R. 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, June 1981.

[24] P. J. Rousseeuw, "Least median of squares regression," *Journal of the American Statistical Association*, vol. 79, no. 388, pp. 871–880, 1984.

[25] R. A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane," *Information Processing Letters*, vol. 2, pp. 18–21, 1973.