Combining monoSLAM with Object Recognition for Scene Augmentation using a Wearable Camera

R O Castle, G Klein, D W Murray*

Active Vision Laboratory, Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK.

Abstract

In wearable visual computing, maintaining a time-evolving representation of the 3D environment along with the pose of the camera provides the geometrical foundation on which person-centred processing can be built. In this paper, an established method for the recognition of feature clusters is used on live imagery to identify and locate planar objects around the wearer. Objects' locations are incorporated as additional 3D measurements into a monocular simultaneous localization and mapping process, which routinely uses 2D image measurements to acquire and maintain a map of the surroundings, irrespective of whether objects are present or not. Augmenting the 3D maps with automatically recognized objects enables useful annotations of the surroundings to be presented to the wearer. After demonstrating the geometrical integrity of the method, experiments show its use in two augmented reality applications.

Key words: Wearable vision, augmented reality, object recognition, simultaneous localization and mapping.

1. Introduction

The availability of miniature hand-held and body-worn cameras coupled to high-performance portable processors is providing technological impetus to the study of egocentric wearable vision and visual augmented reality (AR). While position and orientation data from GPS and magnetometers (available in modern smart-phones, for example) allows spatially-aware annotation of imagery, more precise localization from visual sensing and analysis is required to establish "see-through" AR, where 3D graphical entities project seamlessly with the real scene.

The visual analysis required shares much with that for vehicle navigation in markerless environments. In both fields, recovery of the camera's location and the disposition of its environment is paramount. At a more detailed level, however, an important difference emerges. Whereas robot vehicles require structural information before they can make any movement at all, human wearers can usually be relied upon to avoid obstacles, to undertake intelligent exploration, and so on. Of more immediate importance in wearable vision is the delivery of a sense of *what* is where in the camera's environment.

However, a shortcoming of many applications of vision in AR are the presumptions that object identities are already determined, and that objects can be represented by highly-carpentered geometric models suited to model-based tracking, often using edges. Whether such models are built entirely by hand or semi-automatically (either on-line [1] or from a stored sequence [2]) there is a significant cost in time, and the model builder usually has to guess which geometric elements on an object will actually provide reliable features to track.

This paper takes a contrasting approach, and describes a real-time wearable system which avoids the use of strong CAD geometrical features, and instead combines recent advances in point-based visual localization and map-building with recognition using appearance models. Objects in the scene are detected and identified using appearance models learned from SIFT features [3], and also located using a single view. In parallel, their locations are used as 3D measurements, in addition to commonly used 2D image features, in a monocular simultaneous localization and mapping process [4] to build a map of the surroundings populated with recognized objects. The paper demonstrates the utility of the approach for scene augmentation, both in the large, where the map dimensions exceed the object dimensions, and at finer scale, where locations within an object are of particular interest to the wearer.

The underlying processes of monoSLAM and object identification using SIFT are mostly standard, and are outlined in Sections 3 and 4, where the specializations required in this work are introduced. The transfer of information between the two processes is more involved, and is described in Section 5.

The implementation of the method on our wearable system is described in Section 6, and results from laboratory and application experiments are given in Sections 7 and 8. Closing remarks are made in Section 9. First, however, we provide motivation for the approach with reference to previous work.

*Corresponding author. Tel +44 1865 273106.
Email addresses: bob@robots.ox.ac.uk (R O Castle), gk@robots.ox.ac.uk (G Klein), dwm@robots.ox.ac.uk (D W Murray)

2. Related work

Those working in the area of wearable systems have often used GPS for localization and activity analysis: examples are found in [5], [6], and [7]. Visual sensing in established world frames is also common: in [8] external cameras provide a fix on the wearer's location, and in [9] and [10] inertial sensing combined with visual measurement of pre-mapped fiducial targets does the same.

A more general approach is to use markerless structure from motion (SFM) or simultaneous localization and mapping (SLAM) techniques. The emphasis in SFM has tended to be placed on obtaining dense structure. Image feature tracks are built up incrementally over a sequence, and, from initial estimates given by multifocal geometry, the camera poses and 3D scene positions are optimised in a batch using off-line bundle adjustment (*e.g.* [11], [12], [13]). Optimality requires all observations that project from a particular scene point to be in proper correspondence, which is difficult to assure using a method which, in its basic form, does not supply frame-by-frame estimates of structure and camera position. By contrast, SLAM (*e.g.* [14], [15], [16]) places emphasis on the continual recovery of the state of the camera and scene structure, and on maintaining information about the correlation between state members. This makes re-matching after neglect easier, and also allows uncertainty to be reduced immediately throughout the camera and map state vector upon loop closure, again assisting feature matching.

Recent advances in the recovery of high quality visual odometry and the intelligent use of keyframes have eroded SLAM's apparent advantage over SFM for real-time operation (*e.g.* [17], [18], [19]). Nonetheless, in this paper, recovery of location is achieved using the monocular SLAM method introduced in [20] and refined in [4] by Davison and co-workers. Like the original methods using sonar [21] [22], it is based on the extended Kalman filter (EKF), and it uses much of the formalism developed for stereo visual SLAM in [23].

The quadratic computational complexity of EKF-SLAM has made finding other methods to handle large-scale maps a major concern [24], [25], [26]. It has been argued though that the EKF remains well-suited to vision using sparse landmark points [4]. This is perhaps particularly the case for vision for wearables [27]. First, sparseness of representation is no hindrance to navigation because, as noted earlier, one can usually rely on the wearer to move around without mishap. Secondly, the need to impose a limit on the growth of the feature map in order to maintain video-rate performance is quite compatible with the notion of a local workspace of fixed volume around the wearer (as seen later in the experiments).

MonoSLAM was used earlier in our laboratory [27] to facilitate fixation using an active wearable camera. Once part of the camera's environment had been mapped the wearer could select a 3D location of interest, usually lying on an object, on which the camera would fixate. However, the link between the fixation point and any underlying object was not established automatically, and, being based on just one point, was intrinsically fragile. While it is certainly more robust to make that link using many points, this conflicts with the requirements of complexity discussed above. The compromise made here is to establish the identity and location of an object using many points, but to install it in the map using the minimal number.

Others have hung richer information onto sparse maps once geometrical integrity is established. For example, [28] uses methods from machine learning and attempts to divide a map into a small number of given classes such as office, corridor, or doorway. More ambitious work attempts to determine even these labels automatically. For example, in [29] images sequences are categorized in similar shots, and then used to characterize areas of the map. At a slightly lower level is geometric map augmentation: one recent example used geometric primitive fitting combined with model selection to produce a reduced representation of a map [30]. However, these approaches perform post-processing and run apart from the SLAM process itself. The approach developed here allows both to progress in parallel.

A last highly relevant work is that of Gordon and Lowe [31], who applied Lowe's method of object recognition using SIFT to augment imagery. However, their approach is more akin to model-based tracking but using models which are built beforehand using SIFT features in a dense structure from motion reconstruction. During a live run, when a known object is detected, its pose can then be tracked over time. In this paper, by contrast, detected objects are not strictly tracked, but are installed and localized in a 3D map as the map continues to be built.

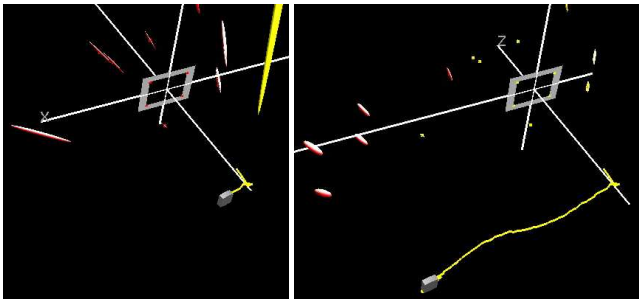
3. MonoSLAM for wearable vision

As noted earlier, the version of monoSLAM used here is adapted from that described by Davison *et al.* [4]. The EKF has as its state and covariance

$$\boldsymbol{p} = \begin{bmatrix} \boldsymbol{c} \\ \boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{X}_n \end{bmatrix}; \mathrm{P} = \begin{bmatrix} \mathrm{P}_{\boldsymbol{c}\boldsymbol{c}} & \mathrm{P}_{\boldsymbol{c}\boldsymbol{X}_1} & \cdots & \mathrm{P}_{\boldsymbol{c}\boldsymbol{X}_n} \\ \mathrm{P}_{\boldsymbol{X}_1\boldsymbol{c}} & \mathrm{P}_{\boldsymbol{X}_1\boldsymbol{X}_1} & \cdots & \mathrm{P}_{\boldsymbol{X}_1\boldsymbol{X}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{P}_{\boldsymbol{X}_n\boldsymbol{c}} & \mathrm{P}_{\boldsymbol{X}_n\boldsymbol{X}_1} & \cdots & \mathrm{P}_{\boldsymbol{X}_n\boldsymbol{X}_n} \end{bmatrix}, \quad (1)$$

where the $\boldsymbol{X}_i$, $1 \leq i \leq n$, are 3D locations of the map features, and $\boldsymbol{c} = [\boldsymbol{t}_c^w, \mathbf{q}^{wc}, \boldsymbol{v}, \boldsymbol{\omega}]$ describes the camera state. Position $\boldsymbol{t}_c^w$ is that of the camera's optic centre in the world frame, orientation is represented by the unit quaternion $\mathbf{q}^{wc}$ which would realign the camera frame with the world frame, and $\boldsymbol{v}$ and $\boldsymbol{\omega}$ are the instantaneous rectilinear and angular velocities defined in the world and camera frames, respectively. The state's non-linear evolution over time Δt from step k to $(k+1)$ is modelled by $\boldsymbol{p}_{k+1} = \boldsymbol{f}(\boldsymbol{p}_k, \boldsymbol{\lambda}_k) + \boldsymbol{e}_k$, where $\boldsymbol{e}_k$ is an uncorrelated zero-mean Gaussian noise sequence and $\boldsymbol{\lambda}_k$ is a control input which, as there is no source of odometry here, is set to zero. The 3D map positions are stationary and their associated noise is zero, and the part of $\boldsymbol{f}()$ dealing with the camera is a constant velocity model [4]

$$[\boldsymbol{t}_c^w,\ \mathbf{q}^{wc},\ \boldsymbol{v},\ \boldsymbol{\omega}]_{k+1} = [\boldsymbol{t}_c^w + \boldsymbol{v}\Delta t,\ \mathbf{q}^{wc} \times \mathbf{q}(\boldsymbol{\omega}\Delta t),\ \boldsymbol{v},\ \boldsymbol{\omega}]_k\ . \quad (2)$$



(a) (b)

Figure 1: Graphical output from the first few seconds of a monoSLAM run without objects, showing the camera track and the characteristic deflation of the covariance ellipsoids of the 3D map points.

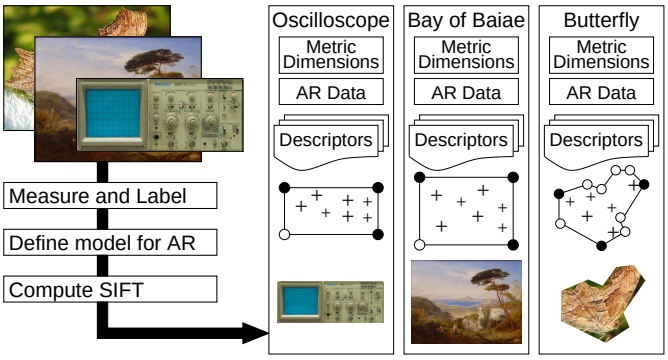


Figure 2: The object database contains planar objects (here pictures), the list of SIFT descriptors and their locations (crosses), and the locations of boundary points (circles), three of which (filled circles) are tagged for use in the SLAM map.

In monoSLAM, measurements are associated with the predicted projections of 3D map points given the current camera pose

$$\tilde{\boldsymbol{x}} = \boldsymbol{h}(\boldsymbol{p}) + \boldsymbol{d}\,, \tag{3}$$

where $\boldsymbol{d}$ is an uncorrelated zero-mean Gaussian noise sequence, and $\boldsymbol{h}()$ models perspective projection into the camera whose intrinsic and radial distortion parameters are already calibrated. Existing features are matched using active search within the predicted match region using normalized sum-of-squared difference correlation between the image and 11×11 pixel appearance templates stored with each map feature. New features for inclusion in the map are detected with the Shi-Tomasi saliency operator [32] and are initialized using the inverse depth representation of Montiel *et al.* [33]. The usual EKF update of the state and covariance matrix is followed [34].

To illustrate typical output from monoSLAM alone, without objects, Fig. 1 shows views of the recovered structure points and covariances, along with the camera's track, as they developed during a monoSLAM run over a period of a few seconds. The rectangular calibration plate serves to define the world coordinate system at the outset, and also resolves the depth/speed scaling ambiguity inherent in monocular motion processing.

In this work, the first change made to monoSLAM is that certain sets of triples of map features in the state vector, $\boldsymbol{X}_i$, $\boldsymbol{X}_{i+1}$, $\boldsymbol{X}_{i+2}$, say, arise from objects rather than from the general scene. Measurements of these features are not made via the image, but directly in 3D and are assumed to arise from the measurement model

$$\tilde{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{D}\,, \tag{4}$$

where $\boldsymbol{D}$ is again a Gaussian noise sequence. However, as described in Section 5.2, care has to be taken with these measurements as they become available only after several image-based updates of the filter have occurred.

4. Object detection and identification

4.1. The object database

A database of objects is constructed beforehand using single views of planar objects. To avoid confusion with the "live" scene and image, we refer to object coordinates as $\boldsymbol{U}$ and their projections while building the database as $\boldsymbol{u}$.

To construct an entry, an image of the object is captured and, after correcting for radial distortion, SIFT descriptors $\boldsymbol{\sigma}_i$ and their image positions $\boldsymbol{u}_i$, $1 \le i \le I$, are derived [3]. The captured image need not be aligned parallel with the object, and so the homography H between the object and image is found by choosing four or more image points whose corresponding planar object points ($\boldsymbol{U} = [U, V, 1]^\top$ in 2D homogeneous coordinates) can be located by hand in a object-based Cartesian coordinate frame. The object's extent is defined by hand with a set of M boundary points $\boldsymbol{U}^{\mathrm{B}}_m$ in this object-based frame. These can be either be defined directly on the object, or in the image and transferred to the object plane via the homography.

The top-level database entries, illustrated in Fig. 2, have the form

$$\begin{aligned} O_j \;=\; \Big\{\; & \mathcal{I}_{\mathrm{R}};\;\; \{\boldsymbol{\sigma}_i,\, \boldsymbol{U}_i = \mathrm{H}^{-1}\boldsymbol{u}_i\}_{i=1\ldots I}; \\ & \{\boldsymbol{U}^{\mathrm{B}}_m\}_{m=1\ldots M};\;\; \{m_1, m_2, m_3\} \in \{1\ldots M\}; \\ & \{\boldsymbol{U}^{\mathrm{AR}}_n,\, \langle \text{AR Markup} \rangle_n\}_{n=1\ldots N} \quad \Big\}_j\,. \end{aligned}$$

Each comprises (i) the boundary-clipped image $\mathcal{I}_{\mathrm{R}}$ rectified by the homography so that it appears as a fronto-parallel view; (ii) the SIFT features and their object plane locations $\boldsymbol{U}_i$; (iii) the boundary points; (iv) the indices of just three boundary points flagged for use in the SLAM map, as explained later; and (v) a number of positions tagged with annotations for AR. Each object entry typically contains around 2 000 descriptors, and each database holds tens of objects. To facilitate efficient on-line search, all SIFT descriptors are structured in a kd-tree, following [3].

4.2. Detection and identification

During a live run, and in parallel with monoSLAM, a video frame is selected at intervals and SIFT features extracted. Their locations are corrected for radial distortion (giving $\boldsymbol{x}$), and the detected features are matched to the stored keypoints of the known objects. (This is faster than undistorting the whole image, and the distortion is not large enough to significantly af-

fect the SIFT descriptors.) Candidate matching descriptors are found by search using the pre-computed kd-tree [3].

If the number of matched points from any given object's database entry to the current image is greater than a threshold (the value 8 is used here), that object is deemed potentially visible. Now because of repeated structure or other scene confusion, some of the features may be incorrectly matched. However, as the database objects are known to be planar, the positions of the *database* SIFT feature points $\boldsymbol{U}$ and the locations $\boldsymbol{x}$ of their matches in the live image should be related by a plane-to-plane homography $\boldsymbol{x} = \mathtt{H}'\boldsymbol{U}$. RANSAC is used to estimate the homography $\mathtt{H}'$ and if a substantial consensus set is found it is inferred that the database object is indeed visible in the current frame.

5. Object localization, and its incorporation in monoSLAM

Having determined a particular object is visible, its location is recovered by decomposing the homography between scene and current image.

In the object-centred coordinate frame, the object lies in the plane $W = 0$, and 3D homogeneous points on the object are $[U, V, 0, 1]^\top$. Thus, in any view the projection can be written, up to scale, in terms of extrinsic and intrinsic parameters as

$$\boldsymbol{x} = \mathtt{K}\,[\mathtt{R}^{cb}|\boldsymbol{t}_b^c]\,[U, V, 0, 1]^\top = \mathtt{K}\left[\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{t}_b^c\right]\,[U,\ V,\ 1]^\top . \qquad (5)$$

The first two columns of the rotation matrix and the translation are found from the homography already computed as the output of RANSAC and assuming known camera calibration $\mathtt{K}$,

$$[\ \boldsymbol{r}_1 \quad \boldsymbol{r}_2 \quad \boldsymbol{t}_b^c\ ] = \mathrm{sign}\left(\mathtt{K}^{-1}\mathtt{H}'\right)_{33}\,\mathtt{K}^{-1}\mathtt{H}' , \qquad (6)$$

now modulo some positive scale. Because the estimate $\mathtt{H}'$ is noisy, there is no guarantee that $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ found above will be orthogonal. The closest rotation matrix, and hence the overall scale for the translation, is determined using singular value decomposition

$$\mathtt{U}\Sigma\mathtt{V}^\top \leftarrow [\boldsymbol{r}_1\ \boldsymbol{r}_2\ \boldsymbol{r}_1 \times \boldsymbol{r}_2] , \quad \mathtt{R}^{cb} = \mathtt{U}\mathtt{V}^\top , \quad \boldsymbol{t}_b^c \leftarrow \boldsymbol{t}_b^c / \sqrt{\Sigma_{11}\Sigma_{22}} . \qquad (7)$$

Any point $\boldsymbol{U}$ in the object's frame can now be transformed into monoSLAM's world frame using the estimate of camera's pose at the time of capture

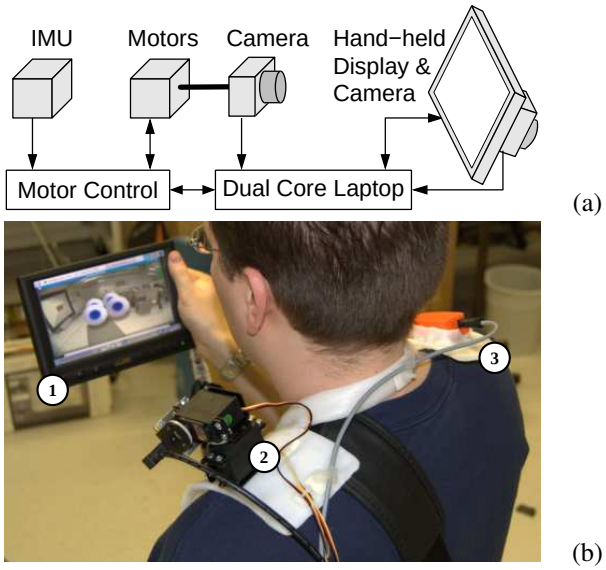
$$\boldsymbol{X} = \begin{bmatrix} \mathtt{R}^{wc} & \boldsymbol{t}_c^w \\ \boldsymbol{0}^\top & 1 \end{bmatrix} \begin{bmatrix} \mathtt{R}^{cb} & \boldsymbol{t}_b^c \\ \boldsymbol{0}^\top & 1 \end{bmatrix} \boldsymbol{U} , \qquad (8)$$

where rotation matrix $\mathtt{R}^{wc}$ is derived from $\mathbf{q}^{wc}$.

5.1. Adding recognized object locations to the map

A number of methods for entering objects to monoSLAM's 3D map can be envisaged but, as noted in Section 2, the pressure of computational complexity suggests using a minimal description.

Three points are used. However, these are not feature positions, but the three points $\boldsymbol{U}_m^{\mathrm{B}}, m = m_1, m_2, m_3$, from the object's boundary, as designated in the object database entry. One benefit of this approach is that no reliance placed on any particular SIFT features being re-measured over time: the position of the boundary points is computed collectively from all of them. Another is that no additional mechanism is required in the SLAM process to make the method applicable to both two and three dimensional objects.



(a)

(b)

Figure 3: Schematic (a) and realization (b) of the wearable camera system. (1) Hand-held display with camera mounted on the rear for AR applications. (2) Active camera capable of pan and tilt, mounted on left shoulder. (3) Inertial measurement unit (IMU) on right shoulder.

After transformation using Eq. (8), the measurement covariances of the three points are estimated in monoSLAM's world frame as ellipsoids with their major axes aligned along the ray from the point to the position of camera's optic centre esimated at the time of image capture.

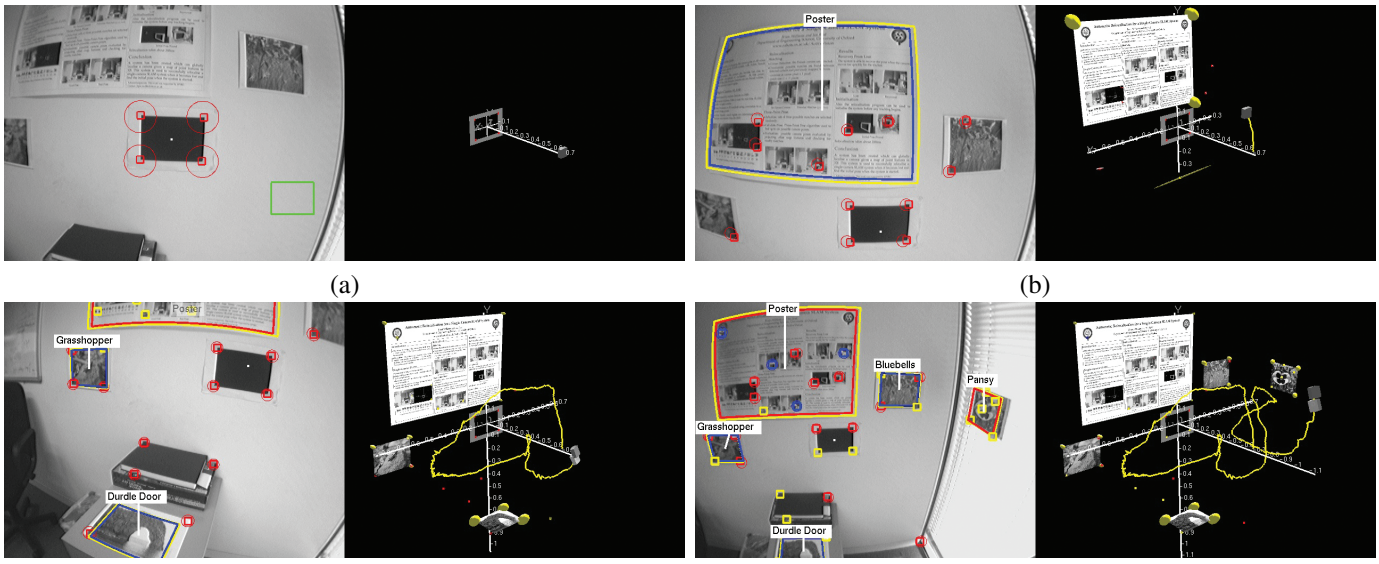
5.2. Delayed object insertion

Because object detection takes a variable amount of time, and because it runs more slowly than SLAM, the detection processing must be performed in the background, always deferring to the immediate needs of monoSLAM to run at frame-rate. A mechanism is required to permit measurement updates using recognized objects at *whatever* time the detection and recognition processes manage to complete processing a frame. We use the technique of delayed decision-making proposed by Leonard and Rikoski [35].

Suppose the single camera SLAM system runs as normal, and that at some time step k the object recognition and object localization module is able to start processing. At this point the current state vector is augmented by the camera pose, $\boldsymbol{s} = [\boldsymbol{t}_c^w, \mathbf{q}^{wc}]$,

$$\boldsymbol{p}^{\mathrm{A}} = [\boldsymbol{c} \quad \boldsymbol{s} \quad \boldsymbol{X}_1 \quad \cdots \quad \boldsymbol{X}_n]^\top , \qquad (9)$$

initialized to the current pose value $\boldsymbol{s}_k$. The covariance matrix



(a) (b) (c) (d)

Figure 4: Frames from the simple scene experiment. The augmented camera view is shown on the left and the associated map on the right. (a) System initialized using the calibration plate. (b) The first object is detected. (c) Further objects are detected and added to the map. (d) All five objects have been detected and localized. Note how the covariances of the three points use to define the poster's position shrink in size.

is similarly augmented

$$\mathrm{P}^{\mathrm{A}} = \begin{bmatrix} \mathrm{P}_{\boldsymbol{cc}} & \mathrm{P}_{\boldsymbol{cs}} & \mathrm{P}_{\boldsymbol{c}\boldsymbol{X}_1} & \cdots & \mathrm{P}_{\boldsymbol{c}\boldsymbol{X}_n} \\ \mathrm{P}_{\boldsymbol{sc}} & \mathrm{P}_{\boldsymbol{ss}} & \mathrm{P}_{\boldsymbol{s}\boldsymbol{X}_1} & \cdots & \mathrm{P}_{\boldsymbol{s}\boldsymbol{X}_n} \\ \mathrm{P}_{\boldsymbol{X}_1\boldsymbol{c}} & \mathrm{P}_{\boldsymbol{X}_1\boldsymbol{s}} & \mathrm{P}_{\boldsymbol{X}_1\boldsymbol{X}_1} & \cdots & \mathrm{P}_{\boldsymbol{X}_1\boldsymbol{X}_n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{P}_{\boldsymbol{X}_n\boldsymbol{c}} & \mathrm{P}_{\boldsymbol{X}_n\boldsymbol{s}} & \mathrm{P}_{\boldsymbol{X}_n\boldsymbol{X}_1} & \cdots & \mathrm{P}_{\boldsymbol{X}_n\boldsymbol{X}_n} \end{bmatrix}, \qquad (10)$$

where $\mathrm{P}_{\boldsymbol{sc}} = \mathrm{P}_{\boldsymbol{cc}}[\partial \boldsymbol{s}/\partial \boldsymbol{c}]^\top$, and so on. After the saved camera pose has been added to the state, its value can no longer be directly measured. However, the correlation values contained in P^{A} between this saved pose and other elements of the state enable its value to be updated as the EKF's cycles continue. Therefore, as the state continues to be updated, the saved pose will be refined such that it remains consistent with newer state estimates. Once the object detection and localization completes, say n frames later, the updated saved camera pose $\boldsymbol{s}_{k+n}$ is used to determine the position of any recognized objects in the world, rather than $\boldsymbol{s}_k$. Then the saved pose is deleted from the state vector and covariance matrix. Although only one saved state is used here, the mechanism allows for multiple detection processes to start and finish at different times, as further processing power permits.

The delayed insertion method provides a faster and less complex update compared with the alternative of rolling back the EKF, inserting the measurement, and then rolling forward by recalculating all measurements from the frame the object detection was performed on.

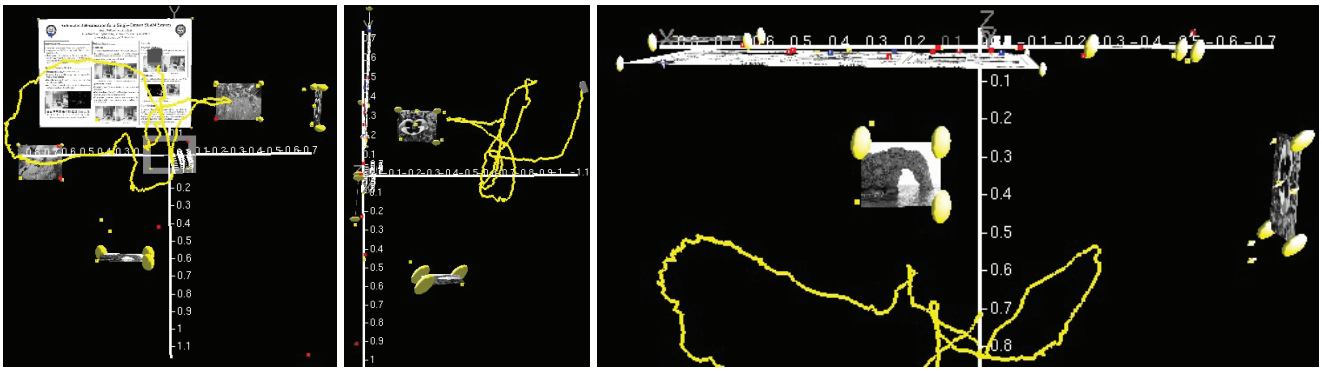
6. Implementation

The method can be applied to any image stream, but here we use one from the wearable system shown in Fig. 3. This consists of a camera mounted on the shoulder; an inertial measurement unit (IMU), which assists in stabilizing the camera and allowing it to make fast redirections of gaze when motion blurs visual feedback; a hand-held touch screen display with a camera mounted on the rear; and a portable computer carried in a shoulder bag. The active camera is broadly similar to that built earlier in this laboratory by Mayol [36], but differs in the detail of the control electronics and mounting, and in overall refinement.

The object recognition and localization process and monoSLAM process are implemented in C++ to take advantage of the capabilities of a dual core processor (here a 2.20 GHz Intel Pentium Core 2 Duo). Including operating system overheads, monoSLAM updates in approximately 10 ms for a 640 × 480 pixel image when executing on one core with around 20 point features, leaving some 20 ms per frame to perform any further computation. Object recognition and localization are run in a separate thread on the second core, continuously grabbing and processing frames. For a typical frame, SIFT detects around 500 keypoints and takes on average 700 ms to complete. Matching against a database of five objects containing 10^4 features takes around 45 ms. Hence, while the point based SLAM runs on our current hardware at 30 Hz, the object detection runs at around 1.5 Hz, at best. These timings of course vary with the size of the database, the number of features found in a frame, and the number of objects found in the scene, making the delayed update method essential.

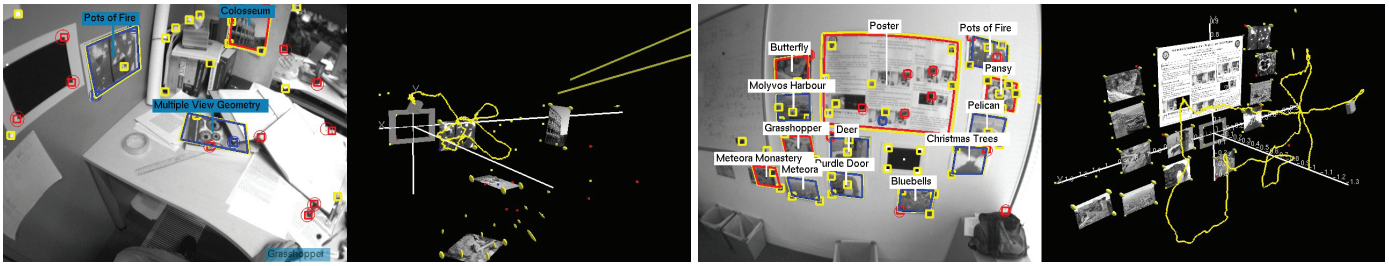
6.1. Relocalization on tracking failure

All the components involved in "smooth running" have been described. However, when monoSLAM is used with a hand-held camera, the camera can become lost through tracking failure. The principal causes observed are erratic camera motion



(a) (b) (c)

Figure 5: Overall views of the recovered structure from the simple scene experiment. (a) A view along the Z-axis within the XY-plane. (b) A view along the X-axis within the YZ-plane. (c) A view along the Y-axis within the XZ-plane.



(a) (b)

Figure 6: Final maps from (a) the cluttered desktop experiment sequence, and (b) the poster experiment sequence.

which confounds the constant velocity model, motion blur, occlusion, cumulative measurement error, and straying into a featureless area. To avoid rebuilding the map from scratch, the relocalization method of Williams *et al.* [37] is invoked when one of the following symptoms is detected: (i) less than three features matched; (ii) camera uncertainty is large; (iii) no visible features; or (iv) too few features matched since a previous recovery. Part of the relocalizer runs continually in the background, using a modified randomized fern classifier to learn the appearance of the monoSLAM map points. When tracking is lost, matches are sought between feature patches from the current view and those in the map, and RANSAC is used with randomly selected sets of three matches to solve for the camera pose. When a valid pose is found (*i.e.*, one with support from many inliers) monoSLAM attempts to continue tracking from this location. The relocalizer has a limited range of distance and angle displacement over which it can function, but it is found highly compatible with the camera being under human control.

7. Experimental evaluation

7.1. Simple scene

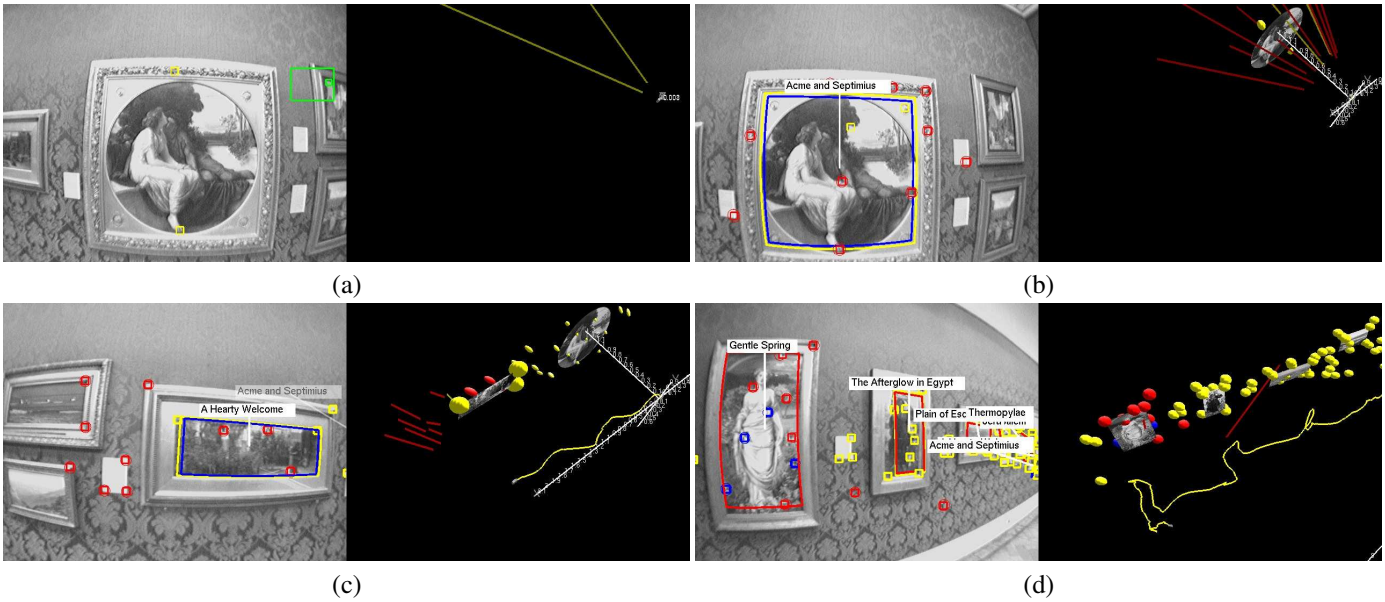
To illustrate the method, a small database is used, consisting of five planar objects with a total of 9,433 keypoints. Details of the objects are given in Table 1. Fig. 4 shows the evolution of processing, from initial calibration of the SLAM system to a time when the five recognized planar objects are all incorporated into the SLAM map[1]. The 2D camera views (on the left) show the object identities and extents of the objects overlaid, while the associated 3D graphics views (on the right) show the evolution of the 3D map, with textures marking the recognized areas. As time proceeds one can see that the quality of the map improves: for example, the poster becomes more vertically aligned with further measurements between panels (b) and (c) of the figure. Fig. 5 shows three graphics views around the recovered 3D map, where three of the objects (Poster, Bluebells, Grasshopper) should be coplanar with the calibration plate (and hence in the XY-plane), and the other two (Pansy, Durdle Door) should by mutually orthogonal and in the ZY and XZ planes respectively. Object detection ran 56 times, and Table 2 shows the number of times each object was successfully measured, as well as the recovered angles between objects and the calibration plate compared with the actual angles.

7.2. Cluttered Scenes

The next two sequences use a database of 16 planar objects with a total of 31,910 keypoints.

The first sequence is similar to that in Section 7.1, but is set in a more visually cluttered desktop environment. Four objects are located in the scene, with one partially occluded. The system is able to recognize and localize all correctly, as can be seen in Fig. 6(a). They were recovered within their respective planes

[1] Videos of all experiments are at `http://www.robots.ox.ac.uk/ActiveVision/Publications/castle_etal_ivc2010/`



(a) (b) (c) (d)

Figure 7: Frames showing detected objects setting the scale when no calibration plate is used. (a) System initialization. (b) First painting detected. (c) Point features localized at same depth as paintings. (d) Further paintings detected.

(a) (b) (c)

Figure 8: Views of the map from the gallery sequence where no calibration plate was used. (a) A view along the Z-axis within the XY-plane. (b) A view along the Y-axis within the XZ-plane. (c) A comparison of camera trajectories from runs (i) including object detection and object calibration and (ii) with neither object detection nor any calibration. The uncalibrated lengths (and speeds) are over-estimated here by a factor greater than two.

Object label	Image size	Keypoints	Size (mm)
Bluebells	640×480	1247	247×198
Durdle Door	640×480	3026	246×198
Grasshopper	640×480	1362	246×198
Pansy	640×480	940	246×198
Poster	640×453	2858	841×594
	Total:	**9433**	

Table 1: Objects in the "simple scene" database.

Object label	No. of detections	Angle to plate (°) Actual	Measured
Bluebells	5	0	2.8 ± 6.2
Durdle-Door	1	90	89.9 ± 14.6
Grasshopper	7	0	4.7 ± 8.1
Pansy	6	90	93.7 ± 6.2
Poster	24	0	2.8 ± 1.2

Table 2: Results from the "simple scene" experiment.

to within experimental error (Colosseum, $91 \pm 6°$; Grasshopper, $85 \pm 3°$; Multiple View Geometry, $87 \pm 3°$; Pots of Fire, $5 \pm 9°$). Tuning the performance to the size of the covariance suggests that the lateral and depth errors are of order 10 mm and 20 mm respectively.

In the second cluttered sequence, the number of objects in the scene is increased to 13. All the objects are located on the same wall, and are all recovered correctly within experimental error, as shown in Fig. 6(b). Because monoSLAM maintains 30 Hz video rate only up to some 100 map features, this number of objects is a practical limit for the number that can be tracked simultaneously in this scene. Each object requires three points in the map, leaving less room for the standard features which provide the overall geometric integrity of the map. Because of their less frequent measurement, object features must be regarded as additional to standard features, not as their replacement, and adding more objects must be compensated for by a reduction in the mapped volume.

7.3. Running without scale

In monoSLAM, the calibration plate resolves the depth/speed scaling ambiguity inherent in monocular motion processing by defining 3D positions which are entered into the map with zero uncertainty. If it is omitted, the scale is set implicitly by the size of the process noise added to the velocity in the EKF, and,

in our experience, the filter is easily de-stabilized. In particular, if depths and speeds are both under-estimated, the filter places too much weight on current measurements and the computation tends to one based on instantaneous image motion. Civera *et al.* [38] have described a filter formulation which mitigates this problem. Here we use known objects to set the scale of the world in the absence of a calibration plate. However, because objects are found at different times during a run, and because they already have uncertainty in their map entries, they do not provide an exact replacement for initial calibration.

Fig. 7 shows frames from a run in a museum gallery where no plate was used. The object database here contains 37 paintings from the Ashmolean Museum in Oxford, with a total of 74,452 keypoints. During the sequence, ten of these were observed and seven of them were successfully recognized. The three that were not detected were either too small, or were somewhat featureless with repeating texture[2], and SIFT failed to find sufficient matchable features. It can be seen that at the first detection of a painting, partially initialized features become fully initialized at the correct depth. As the camera continues to explore, further paintings are detected and added to the map, and features detected around them also initialize to commensurate depths.

There is evidence of curvature in the map of Fig. 7(d), but this is not an issue of calibration and scale. Rather, this sequence turns out to be particularly challenging for the underlying monoSLAM process, due to aliased matching from the repeated texture on the wallpaper and on the paintings' frames. This leads to a map that is curved, as more clearly shown in Fig. 8. However, the important point here is that the camera track is *similarly* curved, and paintings and point features are initialized with the correct depth between them.

Fig. 8(c) compares the trajectory obtained when objects are detected and used for calibration with that obtained when running monoSLAM on the same sequence with no calibration whatsoever. The same general shape is evident, reinforcing the conclusion that curvature here is nothing to do with calibration.

8. Augmentation within objects

The experiment of Fig. 9 demonstrates that the robustness and accuracy are sufficient to provide the camera's wearer with spatially referenced augmentations, or overlays, within objects over extended periods.

The scenario is that of an interactive operating manual for the oscilloscope. The system builds the 3D map of the environment from standard point features and those from one detected object, the front panel of an oscilloscope (Fig. 9(a)). The user is guided through powering up the oscilloscope, (b,c), then adjusting various settings (d-i). When the oscilloscope is not visible (Fig. 9(i)), or when the camera is at oblique angles to the oscilloscope (j,k), recognition is impossible, but the AR overlays can still be presented correctly, due the accurate tracking from the underlying monoSLAM. The final state of the tracked features and object is shown in Fig. 9(l).

In this demonstration, the user confirms that each step is completed. One could envisage providing some automatic visual feedback: a first sanity check might be to detect that the forefinger had at least appeared to press the required button, or the thumb and forefinger had turned the relevant knob.

In the introduction it was noted that a difficulty in most applications of vision to augmented reality was their reliance on detailed geometric descriptions of objects, often in terms of a CAD model. A significant uncertainty for the modeller is how detailed should that model be. In the case of the oscilloscope, for example, should one include just the screen surround, or are the switches required too, and so on? Using appearance models removes this responsibility from the programmer, effectively allowing the object to speak for itself in the imagery. The level of expertise and effort required of the modeller is reduced: in the example here, and using only research code, the tutorial was generated in some two hours.

Within the tolerances set by feature matching between images and the accuracy with which overlays must be placed, the requirement of planarity can be relaxed. The oscilloscope's screen surround and the various knobs protrude to a maximum of 20 mm from the panel front, but the model of it in the object database is just the planar fronto-parallel picture. For this example, it was found that the recognition process performs satisfactorily to a maximum angle of some 20° when close enough for recognition to succeed. Beyond this the view of the object was sufficiently changed to cause recognition failure.

9. Conclusions and Discussion

This paper has presented the combination of methods in recognition using appearance models with those in camera localization and 3D scene reconstruction using point-based monocular visual SLAM to identify and locate objects, and then to insert them as 3D measurements into the map for updating by an extended Kalman filter. The method has been demonstrated using planar objects. By reconstructing a minimal number of virtual 3D boundary points to describe an object's location, the measurements entered into the SLAM map are sparse, robust to partial occlusion, less likely to be incorrect through mismatching, and of higher intrinsic accuracy. The relatively slow and variable rate of delivery of geometry from the recognition process has been properly accommodated in SLAM's statistical framework using Leonard and Rikoski's method of delayed decision-making, which inserts a temporary place-holder location in the state and covariance. This is updated during the time the recognition takes to complete, and is then deleted once it has been used to calculate the geometry of recognized objects. After illustrating the basic concept, the paper has demonstrated aspects of system performance, including timing, geometrical integrity, and tolerance to non-planarity. The method has been applied to providing an augmented reality tutorial for operating an oscilloscope, where we find the efficiency of modelling is greatly increased.

[2] An engineer's critique of the early 19th-century Romantic school of landscape painting.

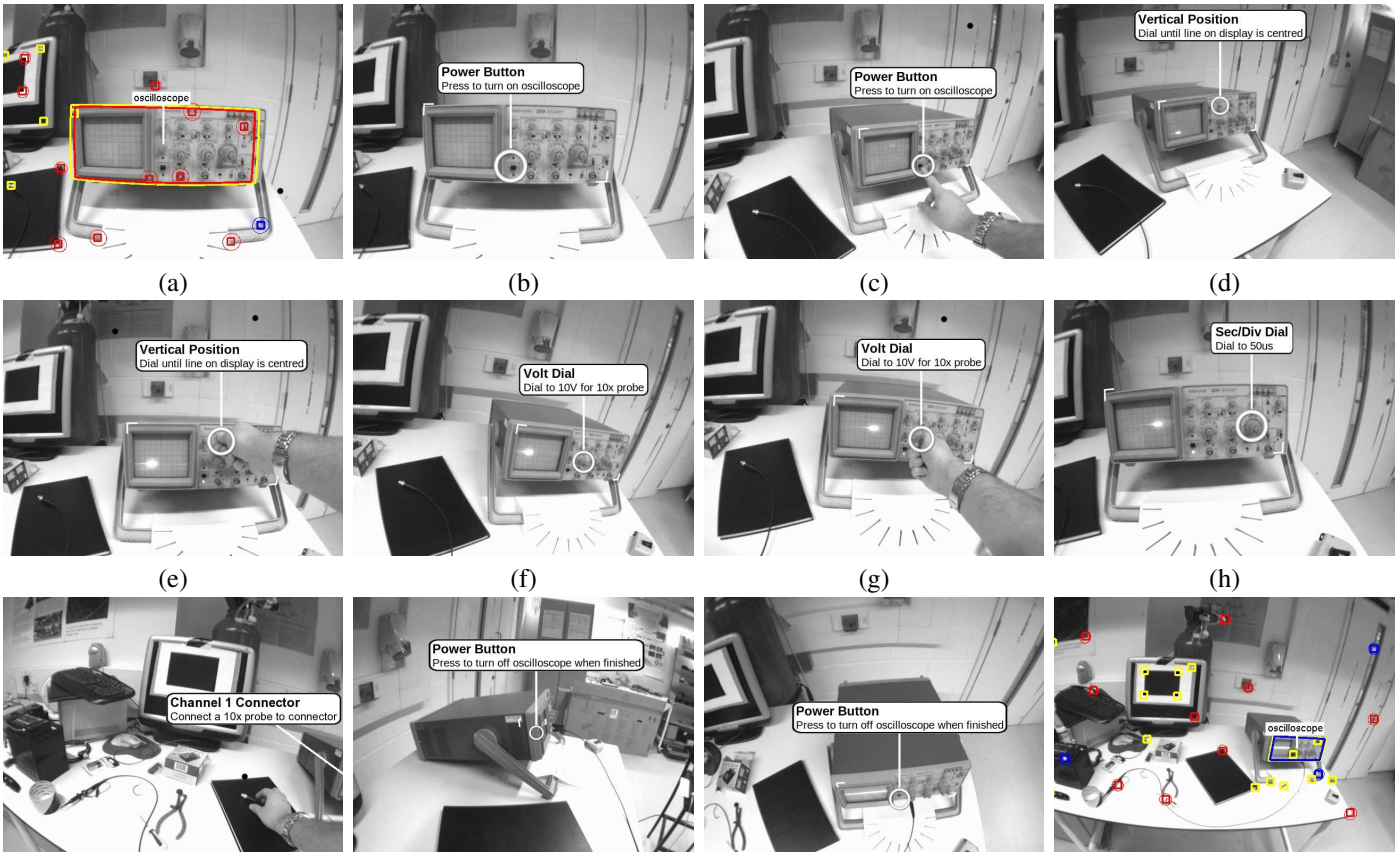


Figure 9: Application to augmented reality. With monoSLAM running and recovering the camera position and surrounding structure, the oscilloscope is recognized at (a). A succession of augmentations is presented, (b) to (k), which guide the user through the setting up of the equipment. Panel (l) shows all the features in the final monoSLAM map.

There are several aspects which are worthy of further development. On the semantic side, object labels allow a more natural interaction between wearable and user. One test of their utility could be made by comparison with the work of Vidal-Calleja [39] who showed that plain directional commands given from moment to moment to the user could be used to improve the orientation of a hand-held camera during a monoSLAM run. However, it was also found that the user struggled to keep up with the detail. Could motion commands referencing objects rather than directions provide a more natural interface?

Another area of interest, explored recently by Reitmayr *et al.* [40], is that of adding new objects to the database while the system is running, allowing one user to annotate the surroundings in preparation for another user. A hand segmentation may be sufficient to generate the object description, and the object's scale estimated from the 3D map. The database image and features might be progressively updated if and when the camera obtains a better view, say when the user moves closer to, and more directly in front of, the object of interest.

This work has combined processes from two extremes. At one end, each object's location is always found from its recognition, while, at the other, the camera's position is always determined from SLAM. Combinations that populate the middle ground might be more appropriate. For example, objects once recognized and installed in the map might be tracked more conventionally rather than always re-recognized. There are also alternative representations of object pose that might be used in the state and covariance of the EKF and different measurement models might be adopted.

Whatever the case, the combination of data is performed here in a principled fashion within monoSLAM's EKF, and we can make two general observations which dampen our enthusiasm for embedding objects in the state.

First, computational complexity does not favour it. Adding objects into monoSLAM limits the physical reach of the map, each removing in our case three points from maximum map size of order 10^2 points. In use, the method is robust in desktop-sized environments with a few objects, but begins to struggle in larger environments like the art gallery, and was found only occasionally successful in street-sized experiments [41]. While using object geometry in the state is of value when the underlying structural information is sparse, our experience suggests that if a denser representation of structure is available — for example, that afforded by the method of video rate bundle adjustment on keyframes [19] — a recognized object's geometry might best be used merely as an augmentation of the 3D map, rather than a measurement into it [42].

Second, running without a calibration plate and relying upon

objects to provide scale in monoSLAM is problematic. If an object is localized early in a run, it acts as a proxy plate and all is well. However, if no object is encountered and the map becomes well-localized (*i.e.*, has small covariances) but at the wrong scale, any object detected afterwards will be inserted at a depth incompatible with surrounding features. There are two outcomes to this conflict, and neither is good. Either the map becomes so corrupted that sufficient matches fail to cause complete tracking failure, or the certainty in the regular scene point features dominate (as they are measured every frame) and the measurements from the object localization process are rejected from the filter as being too far from those expected.

Acknowledgements

This work was supported by grants GR/S97774 and EP/D037077 from the UK's Engineering and Physical Science Research Council. The authors are grateful for discussion with David Lowe during his sabbatical visit to Oxford, and to colleagues Brian Williams and Ian Reid for the use of their camera recovery code.

References

[1] P. Bunnun, W. Mayol-Cuevos, OutlinAR: an assisted interactive model building system with reduced computation effort, in: Proc 7th IEEE/ACM Int Symp on Mixed and Augmented Reality, 2008, pp. 61–64.

[2] A. van den Hengel, A. Dick, T. Thormählen, B. Ward, P. H. S. Torr, VideoTrace: rapid interactive scene modelling from video, ACM Transactions on Graphics 26 (3) (2007) Article 86.

[3] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.

[4] A. J. Davison, I. D. Reid, N. Molton, O. Stasse, MonoSLAM: Real-time single camera SLAM, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (6) (2007) 1052–1067.

[5] D. Ashbrook, T. Starner, Using GPS to learn significant locations and predict movement across multiple users, Personal and Ubiquitous Computing 7 (5) (2003) 275–286.

[6] R. Hariharan, K. Toyama, Project Lachesis: Parsing and modeling location histories, in: Proc 3rd Int Conf on Geographic Information Science, 2004, pp. 106–124.

[7] L. Liao, D. Fox, H. Kautz, Extracting places and activities from GPS traces using hierarchical conditional random fields, International Journal of Robotics Research 26 (1) (2007) 119–134.

[8] A. Pentland, Looking at people: Sensing for ubiquitous and wearable computing, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (1) (2000) 107–119.

[9] M. Kourogi, T. Kurata, K. Sakaue, A panorama-based method of personal positioning and orientation and its real-time applications for wearable computers, in: Proc 5th IEEE Int Symp on Wearable Computing, 2001, pp. 107–114.

[10] E. Foxlin, Generalized architecture for simultaneous localization, autocalibration and map-building, in: Proc IEEE/RSJ Conf on Intelligent Robots and Systems, 2002, pp. 527–533.

[11] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, R. Koch, Visual modeling with a hand-held camera, International Journal of Computer Vision 59 (3) (2004) 207–232.

[12] A. W. Fitzgibbon, A. Zisserman, Automatic camera recovery for closed or open image sequences, in: Proc 5th European Conf on Computer Vision, Vol. 1406 of LNCS, Springer, 1998, pp. 311–326.

[13] D. Nistér, Automatic dense reconstruction from uncalibrated video sequences, Ph.D. thesis, Royal Institute of Technology KTH, Stockholm, Sweden (March 2001).

[14] R. C. Smith, P. Cheeseman, On the representation and estimation of spatial uncertainty, International Journal of Robotics Research 5 (4) (1986) 56–68.

[15] J. J. Leonard, H. F. Durrant-Whyte, I. J. Cox, Dynamic map building for an autonomous mobile robot, International Journal of Robotics Research 11 (8) (1992) 286–298.

[16] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT Press, Cambridge MA, 2005.

[17] D. Nistér, O. Naroditsky, J. Bergen, Visual odometry for ground vehicle applications, Journal of Field Robotics 23 (1).

[18] E. Mouragnon, F. Dekeyser, P. Sayd, M. Lhuillier, M. Dhôme, Real time localization and 3d reconstruction, in: Proc 24th IEEE Conf on Computer Vision and Pattern Recognition, 2006, pp. 363–370.

[19] G. Klein, D. W. Murray, Parallel tracking and mapping for small AR workspaces, in: Proc 6th IEEE/ACM Int Symp on Mixed and Augmented Reality, 2007, pp. 225–234.

[20] A. J. Davison, Real-time simultaneous localisation and mapping with a single camera, in: Proc 9th IEEE Int Conf on Computer Vision, 2003, pp. 1403–1410.

[21] R. C. Smith, M. Self, P. Cheeseman, Estimating uncertain spatial relationships in robotics, in: I. Cox, G. Wilfong (Eds.), Autonomous Robot Vehicles, Springer-Verlag, New York, 1990, pp. 167–193.

[22] J. J. Leonard, H. F. Durrant-Whyte, Directed Sonar Sensing for Mobile Robot Navigation, Kluwer Academic, Boston MA, 1992.

[23] A. J. Davison, D. W. Murray, Mobile robot localisation using active vision, in: Proc 5th European Conf on Computer Vision, Vol. 1407 of LNCS, Springer, Berlin, Heidelberg, 1998, pp. 809–825.

[24] J. G. H. Knight, A. J. Davison, I. D. Reid, Towards constant time SLAM using postponement, in: Proc IEEE/RSJ Conf on Intelligent Robots and Systems, 2001, pp. I:406–412.

[25] J. J. Leonard, P. M. Newman, Consistent, convergent and constant-time SLAM, in: Int Joint Conference on Artificial Intelligence, 2003, Vol. 18, Morgan Kaufmann, 2003, pp. 1143–1150.

[26] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, H. Durrant-Whyte, Simultaneous localization and mapping with sparse extended information filters, International Journal of Robotics Research 23 (7-8) (2004) 693–716.

[27] W. W. Mayol, A. J. Davison, B. J. Tordoff, D. W. Murray, Applying active vision and SLAM to wearables, in: P. Dario, R. Chatila (Eds.), Robotics Research: The 11th International Symposium, Vol. 15 of Springer Tracts in Advanced Robotics, Springer, 2005, pp. 325–334, siena Italy, October 19-21, 2003.

[28] A. Rottmann, O. Martínez-Mozos, C. Stachniss, W. Burgard, Semantic place classification of indoor environments with mobile robots using boosting, in: Proc AAAI-05: the 20th National Conf on Artificial Intelligence, 2005, pp. 1306–1311.

[29] H. I. Posner, D. Schröter, P. M. Newman, Using scene similarity for place labeling, in: Proc 10th Int Symp on Experimental Robotics, 2006, pp. 85–98.

[30] S. Thrun, C. Martin, Y. Liu, D. Hähnel, R. Emery-Montemerlo, D. Chakrabarti, W. Burgard, A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots, IEEE Transactions on Robotics and Automation 20 (3) (2004) 433–443.

[31] I. Gordon, D. G. Lowe, What and where: 3D object recognition with accurate pose, in: J. Ponce, M. Hébert, C. Schmid, A. Zisserman (Eds.), Toward Category-Level Object Recognition (Proc Sicily Workshop), Springer-Verlag, 2006, pp. 67–82.

[32] J. Shi, C. Tomasi, Good features to track, Proc IEEE Conf on Computer Vision and Pattern Recognition (1994) 593–600.

[33] J. M. M. Montiel, J. Civera, A. J. Davison, Unified inverse depth parametrization for monocular SLAM, in: Proc Robotics: Science and Systems, 2006.

[34] Y. Bar-Shalom, T. E. Fortmann, Tracking and Data Association, Vol. 179 of Mathematics in Science and Engineering, Academic Press Inc, San Diego, California, 1988.

[35] J. J. Leonard, R. Rikoski, Incorporation of delayed decision making into stochastic mapping, in: D. Rus, S. Singh (Eds.), Experimental Robotics VII (Lecture Notes in Control and Information Sciences, Vol 271), Springer Verlag, 2001, pp. 533–542.

[36] W. W. Mayol, B. J. Tordoff, D. W. Murray, Wearable visual robots, Per-

sonal and Ubiquitous Computing 6 (2002) 37–48.
[37] B. Williams, G. Klein, I. D. Reid, Real-time SLAM relocalisation, in: Proc 11th IEEE Int Conf on Computer Vision, 2007, pp. 1–8.
[38] J. Civera, A. J. Davison, J. M. M. Montiel, Dimensionles monocular SLAM, in: Proc 3rd Iberian Conf on Pattern Recognition and Image Analysis, Girona, Spain, June 6-8, 2007, 2007, pp. 412–419.
[39] T. A. Vidal-Calleja, A. J. Davison, J. Andrade-Cetto, D. W. Murray, Active control for single camera SLAM, in: Proc 2006 IEEE Int Conf on Robotics and Automation, 2006, pp. 1930–1936.
[40] G. Reitmayr, E. Eade, T. Drummond, Semi-automatic annotations in unknown environments, in: Proc 6th IEEE/ACM Int Symp on Mixed and Augmented Reality, 2007, pp. 67–70.
[41] R. O. Castle, Simultaneous recognition, localization and mapping for wearable visual robots, DPhil thesis, University of Oxford, Department of Engineering Science (2009).
[42] R. O. Castle, D. W. Murray, Object recognition and localization while tracking and mapping, in: Proc 8th IEEE/ACM Int Symp on Mixed and Augmented Reality, 2009, pp. 179–180.