Describing, Navigating and Recognising Urban Spaces - Building An End-to-End SLAM System

Paul Newman[1], Manjari Chandran-Ramesh, Dave Cole, Mark Cummins, Alastair Harrison, Ingmar Posner, Derik Schroeter

Oxford University Mobile Robotics Research Group

Summary. This paper describes a body of work being undertaken by our research group aimed at extending the utility and reach of mobile navigation and mapping. Rather than dwell on SLAM estimation (which has received ample attention over past years), we examine sibling problems which remain central to the mobile autonomy agenda. We consider the problem detecting loop-closure from an extensible, appearance-based probabilistic view point and the use of visual geometry to impose topological constraints. We also consider issues concerning the intrinsic quality of 3D range data / maps and finally describe our progress towards substantially enhancing the semantic value of built maps through scene de-construction and labeling.

1 Introduction and Architectural Overview

This paper describes the techniques we are employing to build an end-to-end SLAM system. To the best of our knowledge at the time of writing no group of researchers have built an embedded SLAM system capable of repetitively and reliably mapping large urban areas time and time again — this is our goal. Our concerns range from the low-level control of sensors and filtering their output through to perception, estimation and inference, longevity, introspection, loop closing, data management, software architectures and up to semantic labeling of maps. In the spirit of ISRR, we intend to provide the reader with a panorama of how these components work together and while doing so, direct the reader to more detailed technical accounts, and discuss their strengths and weaknesses and, where applicable, any open questions.

The structure of this paper is summarised by a walk-through of the major components of our system with forward declarations of the relevant sections. At the heart of the system are two database-like entities one of which is concerned with short-term memory and implements a publish and subscribe architecture for all on-line processes. The other provides a mechanism by which processes can sequester data for later use or search and retrieve data hours after it was written. For example, the loop closure detection module may want to retrieve image data from many hours earlier. The entire system is driven by two principal sensors: a 3D laser scanner and a camera on a pan tilt unit (inter-changeable with a stereo pair). Both sensors stream data to the long-term memory and run-time notification server. We have adopted the Exactly Sparse Delayed State formulation, as proposed by Eustice [12], as the core SLAM estimation technique and this component is discussed further in Section 2. However good the SLAM engine is loop-closure detection and prosecution will be an issue. Our loop closure detection component [11] is probabilistic and appearance-based and has an extremely low false-positive rate; it is discussed further in Section 3. We have a computer vision competency which, in this paper, we only use to impose loop

closing geometry on an in-error state vector — see Section 3.3. However nothing about our architecture design precludes the use of vision as the only navigation sensor. The SLAM estimation module produces a vehicle trajectory over time. From this we are able to generate globally aligned point clouds of 3D laser data (maps); see for example Figure 6 where the laser clouds have been coloured using registered images. At this stage we are able to run two high-level processes which add value to the map. The first [7] is an analysis of the intrinsic quality of the map in which a classifier is run over the globally aligned point cloud to highlight regions that appear to possess implausible scene geometry - for example overlapping walls. The second seeks to use the geometry of the point cloud and the scene appearance captured in images to generate higher level labels for the workspace, which we discuss further in Section 5.

2 Trajectory Estimation

We employ the Exactly Sparse Delayed State formulation as proposed by Eustice [12] for online estimation of the vehicle trajectory. This inverse (sparse) formulation is particularly suitable for exploration which induces a very thinly connected pose graph. The filter is implemented in C++ and shows no notable computational bottleneck over the workspace scales (one or two km) we are interested in. Typically we end a SLAM run with a few thousand poses in the state vector which makes the use of sparse linear algebra structures and solvers imperative. Our numerical tool set is built upon the *vnl* library found with the VXL vision software package.

2.1 From Impressions to Observations

We have taken care to make the central "SLAM Engine" domain neutral: it knows nothing of computer vision, laser scanning or loop closing; it simply knows about pose-graphs, optimisation routines and the existence of a sensor data abstraction we refer to as a "scene impression". Impressions are references (hash keys) to persistent objects stored in long-term memory, typically images, stereo depth maps or 3D laser clouds (both corrected and raw — see Section 4.1). Poses have impressions attached to them as and when notifications of scene capture are received. When it is required to derive the relationship between two or more poses possessing compatible impressions, a suitable external process is forked. It is this third-party executable that possesses the relevant specialisation to interpret the impressions and return a rigid body transformation. This architecture was motivated by the fact that our software base and sensor suite is constantly changing and that having an ever-increasing volume of sensor interpretation and management code in the central estimation framework is a software management problem. It also means that a new sensor can be adopted with little change to the code-base.

2.2 Non-linear optimisation

While the information filter at the heart of the system provides excellent quiescent performance, when a large loop is unexpectedly closed we must take care to handle the baked-in linearisations residing in the information matrix. What is needed is a full non-linear optimisation over the whole pose graph [26]. At completion of the optimisation the Hessian can be copied into the run-time information matrix and one can proceed as before. Again the moderate size of the system (thousands of variables) dictates the use of sparse numerical tools.

Fig. 1. A portion of a trajectory and a rendered map. The quadrangle shown here (around which the vehicle drives twice) is about 40 by 30 m. We choose to show this part of the map in detail because it was more of a challenge than expected. The buildings are visually very repetitive (which makes loop closure detection harder) and for substantial periods of time all of the walls but one are out of range of the laser. A consequence of this is that registrations immediately following a loop closure could only be achieved using the vision system described in Section 3.3.

We have implemented a sparse form of the common-place Levenberg-Marquadt algorithm with a line-search to speed convergence [1] in C++. This large-scale optimisation is at the time of writing the most fragile part of our system. The loops we are closing are often very long and skinny and seem, from time to time, to cause instabilities in the optimisation. We have had some success with the relaxation techniques proposed by Olson [22] but we have not been able to achieve failsafe operation. We note that in the presence of long, loosely constrained pose-graph loops a second pass is needed that considers not just the inter-pose constraints but also the quality of the rendered map (typically millions of laser points thrown at a G.P.U). Looking at the global picture, and equipped with a prior on what are likely point cloud properties (e.g. crispness when seeing workspace surfaces) it should be possible to apply constraints over scales greatly exceeding those over which, for example, scan matching or image alignment can be directly applied. We discuss some of our current work on ensuring the fidelity of the maps generated by our pose based system in Section 6.

2.3 Zipping

It is generally the case that for a large sparse pose graph, imposing a single loop closing constraint does not yield a fully corrected trajectory. Look for example at Figure 2, shown just after application of a single loop closure constraint. This example is illustrative because it highlights the common situation in which the loop closure advisory is not issued at the earliest possible moment, i.e. when the vehicle starts its second loop around the quadrangle, but at some later time when the probability of false loop closure given the input image sequence is sufficiently small (see Section 3 for a discussion on our appearance-based loop closure module). Immediately after loop closing we begin propagating an association wavefront out from the point of loop closure throughout the pose graph, searching for new pairs of poses that following the loop closure application may now be close enough to have their impressions matched to produce a new inter-pose constraint. Every time a new pair is found the pose-graph is updated - possibly by a complete relinearisation in the case of a substantial residual - and the process continues until no further new pairings are discovered. The outcome of this process is that portions of the trajectory become zipped together by a mesh of inter=pose constraints as shown in the lower figure of Figure 2, resulting in a greatly improved map.

[1] with thanks to Andrew Fitzgibbon for input on this

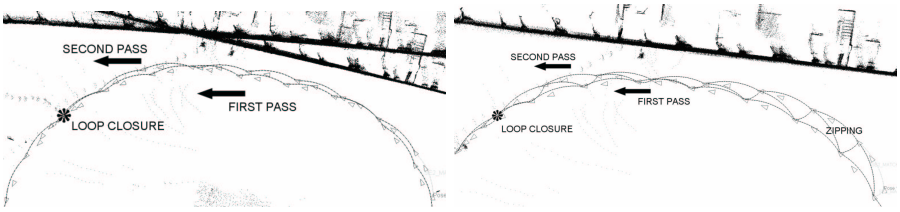

Fig. 2. Loop closure prosecution: in the top figure a loop closure is detected and imposed via a single constraint. The ensuing zipping operation binds previously unrelated sections of the vehicle trajectory together.

3 Closing Loops

Loop closure detection is a well known difficulty for traditional SLAM systems. Our system employs an appearance-based approach to detect loop closure – i.e. using sensory similarity to determine when the robot is revisiting a previously mapped area. Loop closure cues based on sensory similarity are independent of the robot's estimated position, and so are robust even in situations where there is significant error in the metric position estimate, for example after traversing a large loop where turning angles have been poorly estimated.

Our approach, previously described in [11], is based on a probabilistic notion of similarity and incorporates a generative model for typical place appearance which allows the system to correctly assign loop closure probability to observations even in environments where many places have similar sensory appearance (a problem known as perceptual aliasing).

Appearance is represented using the bag-of-words model developed for image retrieval systems in the computer vision community [24, 21]. At time k our appearance map consists of a set of n_k discrete locations, each location being described by a distribution over which appearance words are likely to be observed there. Incoming sensory data is converted into a bag-of-words representation; for each location, we can then ask how likely it is that the observation came from that location's distribution. We also find an expression for the probability that the observation came from a place not in the map. This yields a PDF over location, which we can use to make a data association decision and either create a new place model or update our belief about the appearance of an existing place. Essentially this is a SLAM algorithm in the space of appearance, which runs parallel to our metric SLAM system.

3.1 A Bayesian Formulation of Location from Appearance

Calculating position, given an observation of local appearance, can be formulated as a recursive Bayes estimation problem. If L_i denotes a location, Z_k the k^{th} observation and $\mathcal{Z}^k$ all observations up to time k, then:

$$p(L_i|\mathcal{Z}^k) = \frac{p(Z_k|L_i, \mathcal{Z}^{k-1})p(L_i|\mathcal{Z}^{k-1})}{p(Z_k|\mathcal{Z}^{k-1})} \tag{1}$$

Here $p(L_i|\mathcal{Z}^{k-1})$ is our prior belief about our location, $p(Z_k|L_i, \mathcal{Z}^{k-1})$ is the observation likelihood, and $p(Z_k|\mathcal{Z}^{k-1})$ is a normalizing term. An observation Z is a binary vector, the i^{th} entry of which indicates whether or not the i^{th} word of the visual vocabulary was detected in the current scene. The key term here is the observation likelihood, $p(Z_k|L_i, \mathcal{Z}^{k-1})$, which specifies how likely each place in our map was to have generated the current observation. Assuming current and past observations are conditionally independent given location, this can be expanded as:

$$p(Z_k|L_i) = p(z_n|z_1, z_2, ..., z_{n-1}, L_i)...p(z_2|z_1, L_i)p(z_1|L_i) \quad (2)$$

This expression cannot be evaluated directly because of the intractability of learning the high-order conditional dependencies between appearance words. The simplest solution is to use a Naive Bayes approximation; however we have found that results improve considerably if we instead employ a Chow Liu approximation [8] which captures more of the conditional dependencies between appearance words. The Chow Liu algorithm locates a tree-structured Bayesian network that approximates the true joint distribution over the appearance words. The approximation is guaranteed to be optimal within the space of tree-structured networks. For details of the expansion of $p(Z_k|L_i)$ using the Chow Liu approximation we refer readers to [11].

3.2 Loop Closure or New Place?

One of the most significant challenges for appearance-based loop closure detection is calculating the probability that the current observation comes from a place not already in the map. This is particularly difficult due to the repetitive nature of many real-world environments – a new place may look very similar to a previously visited one. While many appearance-based localization systems exist, this extension beyond pure localization makes the problem considerably more difficult [13]. The key is a correct calculation of the denominator of Equation 1, $p(Z_k|\mathcal{Z}^{k-1})$. If we divide the world into the set of mapped places M and the unmapped places $\overline{M}$, then

$$p(Z_k|\mathcal{Z}^{k-1}) = \sum_{m \in M} p(Z_k|L_m)p(L_m|\mathcal{Z}^{k-1}) + \sum_{u \in \overline{M}} p(Z_k|L_u)p(L_u|\mathcal{Z}^{k-1}) \quad (3)$$

where we have applied our assumption that observations are conditionally independent given location. The first summation is simply the likelihood of all the observations for all places in the map. The second summation is the likelihood of the observation for all possible unmapped places. Clearly we cannot compute this term directly because the second summation is effectively infinite. We have investigated a number of approximations. A mean field-based approximation has reasonable results and can be computed very quickly; however we have found that a sampling-based approach yields the best results. If we have a large set of randomly collected place models L_u (readily available from previous runs of the robot), then we can approximate the term by

$$p(Z_k|\mathcal{Z}^{k-1}) \approx \sum_{m \in M} p(Z_k|L_m)p(L_m|\mathcal{Z}^{k-1}) + p(L_{new}|\mathcal{Z}^{k-1}) \sum_{u=1}^{n_s} \frac{p(Z_k|L_u)}{n_s} \quad (4)$$

where n_s is the number of samples used, $p(L_{new}|\mathcal{Z}^{k-1})$ is our prior probability of being at a new place, and the prior probability of each sampled place model L_u with respect to our history of observations is assumed to be uniform. Note here that in our experiments the places L_u do not come from the current workspace of the robot – rather they come from previous runs of the robot in different locations. They hold no specific information about the current workspace but rather capture the probability of certain generic repeating features such as foliage and brickwork.

3.3 Loop Closure Geometry From Vision

While the system architecture is able to use both 3D laser and visual impressions, we currently rely on the laser system for sequential pose-to-pose observations.

Fig. 3. Detected loop closures shown in red using the probabilistic loop closer (from [11]) described in Section 3. Loop Closures are plotted using low quality GPS data for ground truth.

Problems arise immediately after a loop closure has been detected (see Section 3) where after km of driven path we cannot reasonably depend on the SLAM estimation to provide an initial guess for a scan match inside the ICP algorithm's convergence basin. Bosse et al. [3] in a recent work offered a smart approach for global laser point scan alignment using circular convolution over a vector of scene descriptors and integrated it into the Atlas framework. Our current implementation of this approach worked the overwhelming majority of the time but failed to produce correct alignments in some of the long narrow streets in which we were operating. The issue here is that geometry of the environment is impoverished and there is little salient geometry to firmly anchor registration techniques. This issue motivates our work to retire the laser from use as a pose-to-pose measurement sensor and use it only for 3D scene reconstruction. In its place we should use visual geometry techniques which are now mature and swift — see [16, 10, 20] for compelling expositions of what computer vision can offer the mobile robotics community. At the time of writing we use visual geometry to produce a rigid body transformation between two views in the absence of a prior as and when loop closures are detected. While these are now standard computer vision techniques, a degree of care must be taken to yield robust and swift implementations (especially concerning the effects of lens distortion). The procedure is as follows: **POI Detection:** For large displacements, i.e. strong perspective effects on the POI pixel neighbourhoods, we use the scale-invariant Fast Hessian POI detector and SURF descriptors as proposed in [2]. Although it allows us to establish correct correspondences, the estimation of the epipolar geometry fails in a number of cases. We believe this is due to the feature detector being mainly "attracted" by distinctive regions rather than particular points. As a consequence, the corresponding points do not refer to exactly the same position in 3D. Errors can easily be as large as a few pixels. **Establishing Correspondences**: The feature matching supports a restricted search range (useful for small displacements), a threshold for the ratio between the second and first maxima of the correlation function and a cut-off threshold (correlation values below that threshold are discarded). Correspondences are only accepted if they form a clique, i.e. in both directions the corresponding point presents a maximum of the correlation function. Additionally, correspondences are jealous — they are discarded if more than one point has been assigned to the same corresponding point. **Geometry Extraction**: Robust estimation of the two-view epipolar geometry is achieved via MLESAC [27] and the five-point algorithm [19] to extract the essential matrix followed by a nonlinear optimization of the 3D projection matrix based on the back-projection error, referred to as "The Gold Standard Method" [14, page 285]. The essential matrix describes the epipolar geometry between two calibrated views and once found it is decomposed into a 3D rotation

matrix and a translation vector (up to scale) describing the relationship between cameras. **Resolving scale**: We now have a similarity transform (correct up to scale) in $Se3$. The last step involves back projecting laser points (known depth) via the lens distortion model into the image and finding associations with visual features whose depth is known up to scale (by triangulation from two views). The ratio of the two yields a scale factor which can be used to upgrade the camera-to-camera transformation to a full, correctly scaled, Euclidean solution.

4 Map Generation

4.1 Roll, Pitch and Slip Compensation

It is the nature of contemporary 3D laser scanners[2] that unless we adopt a stop-scan-move paradigm which is clearly unattractive, we must accept that a scene impression cannot be gathered instantaneously - it has a non-negligible duration - and so is prone to distortion by un-modelled vehicle motion. The 3D laser stream is segmented into discrete clouds of 3D points rendered relative to an origin placed at the start of the segment period, T_s. These chunks can, if desired, be used as scene impressions and fed to the estimation engine to derive pose-to-pose observations. This however is a naive approach - it assumes that we can build a consistent global map by tweaking the geometry of a mechanism of rigid links when in reality the links themselves are flexible. Put another way, we would either be implicitly assuming that there is no trajectory error over the duration of the cloud and all error occurs at the junction between segments or that we can undo the effect of intra-cloud trajectory errors by adjusting the inter-cloud transformations.

To address this we correct the intra-cloud vehicle trajectory and hence the point cloud geometry by estimating the roll, pitch and angular skid rates over the duration of the cloud. A high-level description of the process is as follows. Assume that over the cloud capture period T_s a 3D laser provides a stream of N time-tagged 3D points in the vehicle coordinate frame $P = X_{1:N}$ where $X_i = [xyzt]^t$ is the i^{th} 3D point at time $t \in [0 : T_s]$. We segment P into a set of k planar short scans, S_k. In our case where we are nodding a commonplace 2D SICK laser scanner a scan is simply a set of points corresponding to a $[0 \rightarrow \pi]$ planar sweep (albeit at a time-varying elevation angle). Each scan then has a duration of 1/75 seconds. We then RANSAC over each scan S_k projected into the XY plane to extract linear segments. Grouping these segments over $n = (T_s/t_w)$ multiple time windows of duration t_w where $t_w << T_s$ (we nod at 0.6 Hz and so use $t_w = \frac{1}{1.2}$ seconds) we can extract broadly vertical planes. Bearing in mind that we are hoping to correct for small time varying changes in roll and pitch in the +/-5 degree region we reason that the deviation of large vertical planes away from true vertical can be explained by vehicle roll and pitch[3]. We then undertake a 2-dof non-linear optimisation for each of the n time windows over roll and pitch with the objective of rotating the extracted planes to vertical. Fitting a curve to the resulting n roll and pitch values results in two functions $r(t)$ and $p(t)$ describing the roll and pitch of the vehicle over the duration of the cloud. As a final step we try to explain any apparent rotation of the now vertical walls around the z-up axis as a function of an angular error rate term. By simple differencing of the apparent direction of the wall normals projected in the X-Y plane over the n time windows we can fit a function $w(t)$ which

[2] and in particular ours which nods a 2D laser scanner at 0.6 Hz

[3] There is of course a Nyquist issue here, if the frequency of the true roll and pitch signal exceeds $\frac{1}{7}2 * t_w$ we do see aliasing effects

describes the unmodelled rotational slip (rad/s) of the vehicle over the capture period of the cloud. Care has to be taken in producing a robust implementation of this procedure, in particular when it comes to fitting the polynomials $r(t), p(t)$ and $w(t)$ in the presence of noise and occasionally missing data (not all of the n windows will yield roll pitch or yaw data). Nevertheless the improvements on the data are marked as shown in Figure 4.

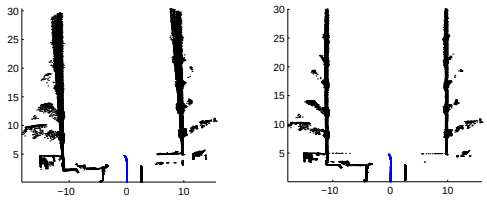

Fig. 4. The effect of the intra-cloud pose alignment scheme described in Section 4.1. On the left is a plan view of a 3D point cloud captured as a vehicle entered a large courtyard (trajectory trail is a small trace running vertical from 0,0). On the right is the roll-pitch-slip corrected point cloud.

4.2 Feature Selection in Point Clouds

Laser scanners can produce vast quantities of points in a single scan, and can exhibit substantial intra-cloud density variation due to inherent sensor geometry and/or viewing angle. This has inhibited the use of traditional navigation techniques, as it is both time-consuming and troublesome to extract robust re-observable features. Instead, view (pose) based approaches have become popular. However, reliable alignment of large, variable density 3D point clouds is difficult. Scene geometry and/or density variations can make scan-matching poorly conditioned, while using all points naively can make the process excessively slow. Furthermore, not all points are equally useful.

Our recent work in [9] deals with these issues by bridging the gap between conventional feature-based representations and those based on 'anonymous' sets of points. The core idea is to re-sample each raw cloud, removing points in overly dense regions, while retaining those in areas deemed useful for a particular function. The re-sampling is achieved using sites of a weighted Voronoi graph generated over a discretized surface fitted to the point cloud. The Voronoi calculation is performed by propagating wave fronts whose speed is modulated by local region properties such as texture, curvature, color or any properties which we deem salient. We refer to this as 'context and feature sensitive re-sampling'. While to date there has been success with this technique, it is still at an early stage of development, and is not yet suitable for integration into our online system. Future work will focus on accelerating this process, and on determining the optimal set of user policies for given environments.

5 Semantic Labelling

The maps we produce are agglomerations of laser points. This representation has only a limited discriminative capacity and fails to adequately represent the subtleties of complex environments. Particularly when navigating in an urban context a more informative, higher-order representation of the environment is indispensable: self-preservation dictates avoidance of highly dynamic regions such

Fig. 5. Typical results from the scene labelling engine in an urban environment. The labels are generated automatically. The classification of individual laser points is omitted for conciseness. A more detailed analysis of results can be found in [23].

as roads; robust localisation depends on distinguishing features beyond the recognition of ubiquitous general objects such as 'ground', 'wall' or 'house'. This motivates the definition of desired classes: in an urban environment places can be distinguished by the type of ground that is present, the colour and texture of surrounding houses (or, more appropriately, of surrounding walls) and the presence or absence of other features such as bushes or trees. Our goal is to add value to maps built by SLAM algorithms by augmenting them with such higher-order, semantic labels. We achieve this by using both *scene-appearance and -geometry* to produce a composite description of the local area.

5.1 The Scene Labelling Engine

Our scene labelling engine proceeds by first performing a plane segmentation on a laser point cloud associated with a particular image. The choice of a plane as a geometric primitive is motivated by its ubiquitous use in man-made environments. This segmentation provides us with a robust estimate of local 3D geometry. Knowledge of the internal camera parameters and the external cross-calibration transformation then enables the projection into the corresponding image of those laser points which fall within the viewing frustum of the camera. Standard appearance features can then be associated with each of these projections. In this case, a histogram for both the hue- and saturation-channel is calculated over a fixed-size neighbourhood around each interest point. Having determined the geometric and visual properties of a particular scene, these are passed through a bank of classifiers, each trained to respond to a given scene attribute – like pavement, tarmac or bush.

5.2 Classification Framework

For classification we chose a chain of support-vector machines (SVMs) with a Gaussian kernel. SVMs are based on a linear discriminant framework which aims to maximise the margin between two classes. They are a popular choice since the model parameters are found by solving a convex optimisation problem. This is a desirable property since it implies that the final classifier is guaranteed to be the best feasible discriminant given the training data. SVMs are inherently binary classifiers. In this work, multi-class classification is performed by training a chain of binary classifiers – one for each class – as one-versus-all [5].

Figure 5 shows a small sample of typical results obtained from our scene labelling engine[4]. The quality of the labelling obtained is encouraging, particularly given the very basic set of appearance features used.

6 Assessing Map Quality

The ability to introspectively assess the quality of the map is important. The concept of map quality can be to a degree an abstract measure and vary largely due to the environment being mapped. While maps of non-urban environments, especially with dense foliage are not so well-defined, urban environments with numerous man-made structures yield sharp object borders. Since the concentration here is to map urban environments, the artificial nature of these workspaces is made use of and map quality is defined on the basis of how "crisp" is the image of the environment. Probable and improbable regions of the map are classified using a context-sensitive classifier. The framework of Conditional Random Fields has been shown to work effectively in capturing contextual information [17, 1].

6.1 Classification Pipeline

The goal of the classifier is to classify regions of the map as a probable representation, "plausible", or an improbable representation, "suspicious". The strong geometric structure in urban environments is utilized and the 3D point-cloud is first segmented into plane patches. These plane patches form the nodes in a Conditional Random Field framework. Edges are derived from the alignment between two plane patches. Neighbouring plane patches can occur in various alignments. Using the spatial geometry of the points, the node features intuitively score how well the subset of points fit a plane and the edge features score how reasonable the alignment of a plane patch is with its neighbours. The node and edge features are then used to infer the labels for the planes using the inference technique, graph cuts [15, 4, 25]. Learning of the parameters in this work was done with maximum psuedo-likelihood estimation [18].

Once the plane patches are classified as "plausible" and "suspicious", the error causations for "suspicious" regions can be explored. Using the temporal property in the same framework, missed loop closure has been detected. Further details can be found in [6].

7 Future Work

Much remains to be done; while we certainly have the parts in place to achieve our aims we are not at the stage at which long-term operation is reliable. We also wish to move upwards from point clouds to labelled surfaces and their agglomerations into semantic entities (discrete buildings) and eventually to spoken concepts (joint work with computational linguistics). However, if we were to pick one aspect of this research that needs attention it would be introspection — the ability to look back over past decisions, measurements and optimisations and, armed with several metrics, decide that all is not well and, ideally, plan and execute remedial action. This goes beyond the commonplace day-to-day data association problem where we search for the best way to interpret a given set of measurements (including rejecting them). We should be looking at the final global properties of maps and trajectories (for example compatibility between camera pixels and laser range images) to assess online performance and drive exploration strategies. Our work on map quality analysis is a start down this

[4] A more detailed analysis of results can be found in [23]. Videos can be found at http://www.robots.ox.ac.uk/ posnerhi/HOMProject/compworkspace.html.

path but much remains to be done to provide SLAM systems with the nagging, persistent self-doubt that we believe will lead to the robust implementations we desire.

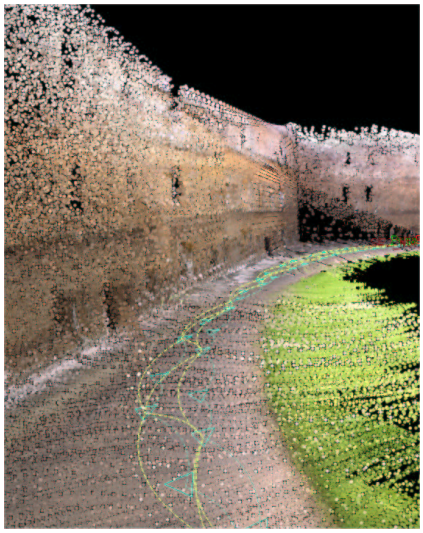

Fig. 6. Part of the 3D SLAM map shown in Figure 1 with the laser points painted with color extracted from the on-board camera. Black areas had no associated camera image.

References

1. Dragomir Anguelov, Ben Taskar, Vassil Chatalbashev, Daphne Koller, Dinkar Gupta, Geremy Heitz, and Andrew Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 169–176, Washington, DC, USA, 2005. IEEE Computer Society.
2. Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. In *Proc. of the 9th European Conference on Computer Vision (ECCV), Graz, Austria*, 2006.
3. Michael Bosse and Jonathan Roberts. Histogram matching and global initialization for laser-only slam in large unstructured environments. In *ICRA*, pages 4820–4826, 2007.
4. Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004.
5. Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.
6. M. Chandran-Ramesh and P. M. Newman. Assessing map quality and error causations using conditional random fields. In *Proc. of the IFAC Symposium Intelligent Autonomous Vehicles*, 2007.
7. M. Chandran-Ramesh and P. M. Newman. Assessing map quality using conditional random fields. In *Proc. of the International Conference on Field and Service Robotics*, 2007.
8. C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, IT-14(3), May 1968.
9. David M. Cole and Paul M. Newman. Context and feature sensitive re-sampling from discrete surface measurements. In *Robotics: Science and Systems (RSS)*, Atlanta, Georgia, USA, June 2007.

10. A.I. Comport, E. Malis, and P. Rives. Accurate quadri-focal tracking for robust 3d visual odometry. In *IEEE Int. Conf. on Robotics and Automation, ICRA'07*, Rome, Italy, April 2007.
11. Mark Cummins and Paul Newman. Probabilistic appearance based navigation and loop closing. In *Proc. IEEE International Conference on Robotics and Automation (ICRA'07)*, Rome, April 2007.
12. Ryan Eustice, Hanumant Singh, John Leonard, Matthew Walter, and Robert Ballard. Visually navigating the rms titanic with slam information filters. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.
13. J. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 318–325, Monterey, California, November 1999.
14. R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, reprinted second edition, 2004.
15. V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):147–159, 2004.
16. K. Konolige and M. Agrawal. Frame-frame matching for realtime consistent visual mapping. *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, April 2007.
17. John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.
18. Lin Liao, Dieter Fox, and Henry Kautz. Extracting places and activities from gps traces using hierarchical conditional random fields. *The International Journal of Robotics Research*, 26(1):119–134, 2007.
19. David Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(6):756–770, 2004.
20. David Nister, Oleg Naroditsky, and James Bergen. Visual odometry. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, pages 652–659, 2004.
21. David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *Conf. Computer Vision and Pattern Recognition*, volume 2, pages 2161–2168, 2006.
22. E. Olson, J. Leonard, and S. Teller. Spatially-adaptive learning rates for online incremental slam. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.
23. I. Posner, D. Schroeter, and P. M. Newman. Describing composite urban workspaces. In *Proc. of the Int. Conference on Robotics and Automation*, 2007.
24. J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, Nice, France, October 2003.
25. R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother. A comparative study of energy minimization methods for markov random fields. *European Conference on Computer Vision*, 2:16–29, 2006.
26. Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *Int. J. Rob. Res.*, 25(5-6):403–429, 2006.
27. Phil Torr and Andrew Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:138–156, 2000.

Acknowledgments

The work reported in this paper was funded by the Systems Engineering for Autonomous Systems (SEAS) Defence Technology Centre established by the UK Ministry of Defence,Guidance Ltd, the Rhodes Trust and by the UK EPSRC (CNA).