2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

# Vision-Only Autonomous Navigation Using Topometric Maps

Feras Dayoub, Timothy Morris, Ben Upcroft and Peter Corke

*Abstract*— **This paper presents a mapping and navigation system for a mobile robot, which uses vision as its sole sensor modality. The system enables the robot to navigate autonomously, plan paths and avoid obstacles using a vision based topometric map of its environment. The map consists of a globally-consistent pose-graph with a local 3D point cloud attached to each of its nodes. These point clouds are used for direction independent loop closure and to dynamically generate 2D metric maps for locally optimal path planning. Using this locally semi-continuous metric space, the robot performs shortest path planning instead of following the nodes of the graph — as is done with most other vision-only navigation approaches. The system exploits the local accuracy of visual odometry in creating local metric maps, and uses pose graph SLAM, visual appearance-based place recognition and point clouds registration to create the topometric map. The ability of the framework to sustain vision-only navigation is validated experimentally, and the system is provided as open-source software.**

## I. INTRODUCTION

In order to achieve vision-only autonomous navigation, mobile robots require the competencies of mapping, localisation, path planning and obstacle avoidance. These competencies must be achieved using monocular or stereo vision rather than a laser range finder. In recent years visual navigation systems have been able to create accurate large-scale maps whilst maintaining precise position and orientation estimates for indoor and outdoor mobile robots [1], [2], [3], [4]. Vision has also become critical for ‘closing the loop’ — forming connections from the current view to other corresponding views elsewhere in the map that might not be co-located in the current map due to sensor error or odometry drift [5]. Through visual pose estimation, and with the aid of visual loop closure detection, highly accurate global maps can be computed using graph optimisation techniques to correct the map. Despite the advantages of vision as a sensor for mapping, a reliable end-to-end mapping and autonomous visual navigation system that includes loop closure detection and subsequent optimisation has not yet been available. This work provides such a system in an open source platform.

This paper progresses the state-of-the-art in vision-only topometric mapping and navigation. Our system does not use a laser range finder (LRF) and operates in a visually challenging indoor environment with problems such as poor light conditions and scarcity of features. The contributions and novelty of this work are:

The authors are with the CyPhy Lab, School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, Australia. first.last@qut.edu.au

*This research was supported under Australian Research Council’s Discovery Projects funding scheme (project number DP110103006).

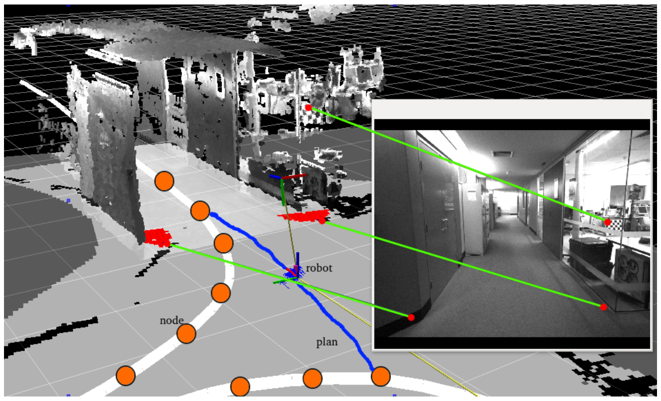

Fig. 1. The hybrid topometric map consists of two levels: global and local. On the global level, the world is represented as a globally consistent graph (the white line). Whereas on the local level each node is the centre of an occupancy grid that covers the immediate neighbourhood surrounding the node. The occupancy grids are generated from visual point clouds which were ray traced and projected to ground level. Red regions represent obstacles in front of the robot. The blue line represents the local plan in the local metric map. The figure shows also a point cloud generated from the stereo head of the robot.

- A mobile robot equipped with only a stereo vision sensor able to perform an autonomous navigation using a topometric map of its environment.
- A system that exploits the local accuracy of visual odometry (VO) from stereo imagery and 3D point cloud registration in constructing local metric maps dynamically and on-demand.
- A robust recovery behaviour to handle the periods of vision blackout during navigation.

After a mapping stage and during a navigation stage, the robot uses the topological level of the map to plan a global path along a graph to its target, when possible taking shortcuts using the local metric map instead of following the graph. The local metric maps are occupancy grids generated from the point clouds by ray casting and projected to ground level. (See Fig. 1). The system stores 3D point clouds rather than occupancy grids. Storing the clouds instead of the grids allows the grids to be rendered dynamically when needed and the possibility of updating the clouds over time in response to changes in the environment. To ensure robust VO during dropouts — e.g when the robot is turning on the spot inside a narrow corridor and facing a plain wall – the system integrates the velocity commands issued by the robot’s base planner. The integration is done only when the number of image features is very low. The system does not use wheel odometry, only visual odometry. The system is built as an open source package[1] for the Robot Operating System (ROS) platform [6].

The rest of the paper is constructed as follows. In Section II, we discuss related work in the field. Section III contains details of the architecture of the system. Section IV presents the experimental evaluation. Finally we draw conclusions and discuss future work in Section V.

## II. Related Work

Regardless of the sensor used, mobile robots need to be able to plan and navigate in their environment in order to do useful work, and in order to do this must construct some form of map. Many methods have been proposed to build such a map, but a simple taxonomy groups the methods into three classes based on the way they represent the environment. *Metric mapping* generates a geometric map of the environment where the positions of landmarks or data are stored in the map relative to a global coordinate system. *Topological mapping* represents the environment as a graph, and the nodes of the graph correspond to places in the real environment. *Hybrid methods* represent the environment as a combination of two or more distinct maps — most commonly a combination of topological and metric maps, often referred to as *topometric maps*.

Metric maps, when represented as occupancy grids, provide a dense representation of the environment and are particularly suitable for precise trajectory planning, but they do not scale well to large-scale environments. On the other hand, pure topological maps provide a more abstract level of representation, which still permits goal-directed planning tasks to be performed. The resulting maps are also more compact and scale better with the size of the environment, but they cannot be used for accurate, shortest path navigation. The complementary strengths and weaknesses of metric and topological methods are the motivation for the topometric mapping approach, which has seen significant recent increase in interest [7], [8]. The environment is typically represented by a global topological map, which connects relatively small local metric maps (sub-maps). The main emphasis is on creating accurate metric sub-maps which are anchored to certain places in the environment and cover the sensor range of the robot at these places.

One of the most comprehensive approaches to hybrid mapping was presented by [9], where the environment of the robot was mapped in a hierarchical fashion, referred to as the *Spatial Semantic Hierarchy* (SSH) consisting of several layers of topological and metric maps. Later, [10] presented an extension to the spatial semantic hierarchy, using an occupancy grid to represent local metric maps of small-scale space created using a particle filter method, while topological methods were used to represent the structure of large-scale space. The method creates a set of topological map hypotheses and can handle multiple nested, large-scale loops. Recently, [11] formulated the hybrid mapping problem as a unified hybrid metric-topological (HMT) Bayesian estimation via the factorisation of the general Simultaneous Localization and Mapping (SLAM) problem. This means that the global map produced by the SLAM system is divided into a set of metric sub-maps, which can then be estimated theoretically from conditionally-independent sequences of observations.

This work is closely related to the work by [12]. However, we differ by using only vision sensor to build the topometric map and use the map for vision-based navigation. Our system also maintains a database of point clouds attached to the nodes of the graph. These point clouds are used to create the local occupancy grids after a registration process and can serve as a base for persistent navigation - by updating the point cloud at each node in the pose graph over time, we will be able to model the persistent structure around each node, which consequently can be reflected in its occupancy grid.

## III. System Architecture

The system presented in this paper is composed of two layers. The first layer is a visual pose-graph SLAM system which takes as input a stream of stereo images captured by on-board cameras, and produces a globally optimized pose-graph online using state-of-the-art methods. The pose-graph is built and optimised using visual odometry [4], visual vocabulary [13] for appearance-base loop closure, wide-baseline stereo to geometrically verify the loop closures triggered by the appearance matcher, and the optimisation algorithm `g2o` by [14] for graph optimisation.

The second layer of the system, which is the focus of this paper, is the visual topometric mapping and navigation system. The topometric layer uses the pose-graph produced from the visual pose-SLAM system as a skeleton for a topometric map. Using the topometric map, this layer allows the robot to perform vision-only navigation, which involves localization and path planning. Fig 2 shows the two layers of the system and the most important relations between their components.

The following sections describe the methodology used for mapping, localization and planning in the topometric navigation system.

### *A. Map Construction*

While the robot is performing visual pose-graph SLAM, a record of data associated with each node in the graph is maintained. For each new node added to the SLAM pose-graph, an entry is also added to the topometric database which contains a 3D point cloud and its associated node ID number. The point clouds are generated using the state-of-the-art dense disparity estimation method (`Efficient LArge-scale Stereo Matching`) ELAS [15]. Using ELAS, we construct a rolling window of point clouds for the local area around the robot Fig. 3. Successive point clouds are registered simply using visual odometry, and this is made possible due to the local accuracy of the visual odometry [4]. The snap shot of the rolling window for a node in the graph is taken when the VSLAM system declares a new node.

[1]`http://www.ros.org/wiki/cyphy_vis_nav`

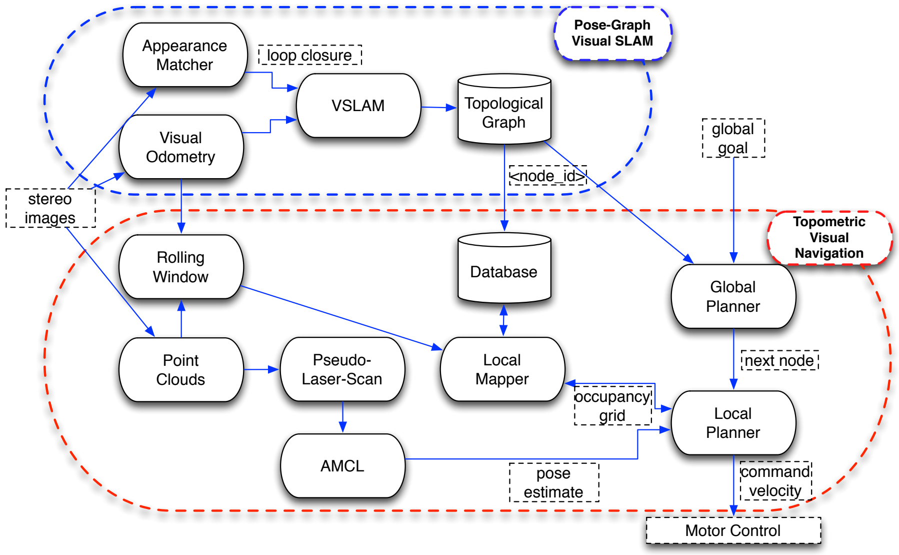

Fig. 2. The two layers of the propsed system. The first layer is a visual pose-graph SLAM system which takes as an input a stream of stereo images captured by the cameras on-board the robot while moving through the environment, and produces a globally optimized pose-graph. The second layer is the visual topometric mapping and navigation system, which uses the pose-graph as a skeleton for a topometric map. Using the topometric map, this layer allows the robot to perform vision-only navigation, which involves localization and path planning.

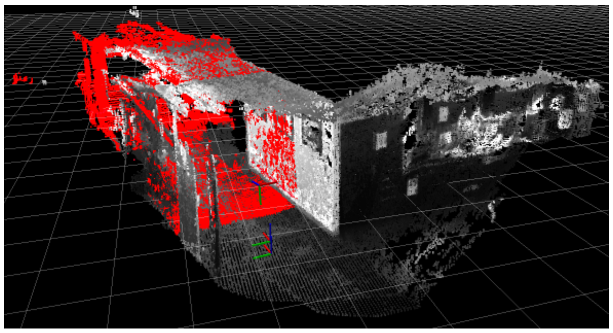

Fig. 3. Grey: Rolling window of point clouds. Red: The point cloud produced from ELAS from one stereo frames of the robot current view.

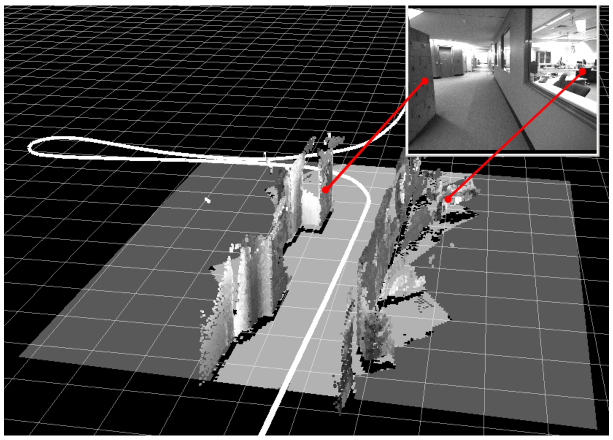

Fig. 4. Local metric map centred on one of the nodes of the pose-graph. The pose-graph is represented as a white line in the figure. The figure shows the occupancy grid and extruded above it the point cloud, which has been used to create it.

The local metric maps produced by our system are Occupancy Grids [16]. These grids are a powerful map representation for accurate metric navigation, which are usually built from laser sensory data. In our case, we use point clouds from stereo vision instead. When requested to build a local map centred on a particular node, the visual pose-graph SLAM system is queried for a set of nodes that lie within a predefined graph distance from that particular node. The point clouds of the nodes in the local regions are recalled from the database and ray cast across the occupancy grid. Ray casting on point clouds is performed only in the ground plane dimensions (x,y) — no filtering based on height is completed other than a threshold of maximum distance from the sensor in three dimensional space. Rays which exceed the maximum distance are discarded. For a cell within the occupancy grid to be classified as either “free” or “occupied” a minimum number of rays must pass through or end in the cell otherwise it will be classified as “unknown”. Given sufficient observations, a cell in the occupancy grid is marked as “occupied” if the ratio of rays ending and passing through is greater then the specified threshold. Fig. 4 shows an example of one of the occupancy grids centred on one of the nodes from the pose-graph and the point cloud which was used to fill it.

Each point cloud is defined with respect to the local frame of its node, so point clouds from several nodes must be registered before the occupancy grid can be generated. Due to noisy point clouds, we find the Iterative Closest Point (ICP) [17] registration method fails. Instead we use a recently introduced method based on 3D Normal distribution Transform (3D-NDT) by [18], which is proven to outperform the current state-of-the-art registration methods. In this work, we use 3D-NDT registration with the relative pose of the

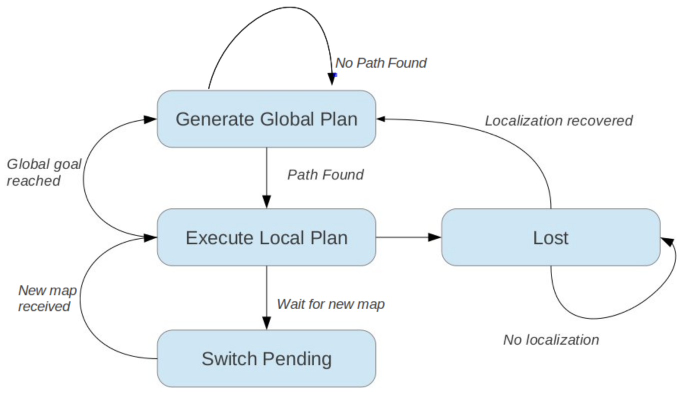

Fig. 5. Block diagram of the topometric planner state machine architecture.

nodes in the graph as an initial guess. Compared to ICP, 3D-NDT registration is almost an order of a magnitude faster. In order to account for any errors in registration, a score based on a closed-form expression for the covariance of the solution obtained from the registration is used [18] .

## B. Localization

In order to track the position of the robot inside the local 2D occupancy grid, the system uses `AMCL`, an adaptive particle filter localizer [19] provided in ROS. We convert the point cloud generated in the current view of the robot to a `pseudo-laser-scan` required by `AMCL` by cutting a horizontal slice out of the point cloud and selecting the closest points to fill in the scan ranges.

While the robot is moving towards a local goal inside the current map, the localizer monitors the distance between the robot's position and the local goal in the current local map. When a pre-defined distance to the goal is reached, the localizer sends a message indicating that a local goal is about to be reached. The localizer then requests a new map centred at a node on the path towards the global goal that covers the current position of the robot. After the new map is delivered, the localizer re-initializes the robot's position in the local frame of the current map with the covariances estimated from the previous local map. A new local goal is subsequently generated within the current map.

## C. Vision Blackout

It is often the case that the robot temporarily lose salience visual information resulting in localization error. This can occur when the robot is turning on the spot inside a narrow corridor with plain walls. To overcome this, we supplement VO with open loop integration of velocity commands sent to the motors. In other words, we assure that the robot accurately performs the requested commands in the short periods of localization failure. This recovery behaviour has proven very effective as we show is section IV-D.

## D. Planning and navigation

The topometric planning in the system is implemented as a state machine to control the execution of a global path through the series of local metric maps that are retrieved on-demand. Figure 5 illustrates the states of the planner.

The planner initializes to the Generate Global Plan state. On entry, it waits to localize in the current local occupancy grid. It then calculates which of the topological graph nodes is closest to the robot's current position, giving the start position in the graph. A global path is then planned to the goal position using Dijkstra's search. Once the global path has been obtained, the planner transitions into the Execute Local Plan state. It finds the last pose on the global path that is contained in the current local map and sends this as a local goal to the robot's controller, monitoring its execution. However, if the global goal lies in the current map, its position (rather than that of the nearest topological node) is sent to the controller. The controller then uses the `pseudo-laser-scan` to build an obstacle costmap. Using the costmap, the controller guides the robot safely to its current local goal.

Typically the robot will not reach the local goal in the current map, as the localizer will ask for a new map before this happens. On receiving notification from the localizer that a switch of maps is about to occur, the planner stops the robot by cancelling the goal, and transitions to the Switch Pending state. If however, the global goal lies in the current map, the robot will reach the goal and the planner will transition back to the Generate Global Plan state to await another global goal.

In the Switch Pending state the planner is merely waiting for notification from the localizer that a new local map has been loaded and that the robot is localized within this map. When this occurs, a message will be received and the robot transitions back into the Execute Local Plan state.

If at any point in executing the local plan the robot becomes lost and/or is unable to reach the local goal, it will transition to the Lost state where it will perform pre-planned recovery behaviours (such as in place rotation) in an attempt to re-localize. From here it will transition back to the Get Global Plan state to re-plan and continue navigation towards the goal.

## IV. Experimental Evaluation

Our experimental platform is a MobileRobots' Research GuiaBot shown in Figure 7. The robot is equipped with two Point Grey Grasshopper monochrome cameras which each have a 1.4M pixel $2/3''$ CCD sensor and a FireWire interface. Images are acquired at 12.5 frames per second. The robot has three on-board computers all running ROS.

The experiments took place on *Level 11* floor of the Science and Engineering Faculty building. This floor has featureless walls, narrow corridors, glass walls and lighting variations along the main corridor which greatly impact the quality of the point clouds. In addition, navigating this environment requires the robot to make sharp $180°$ turns which often result in featureless views close to walls.

## A. The topometric map

Fig. 6 shows the topometric map built for *Level 11*. The nodes in the graph are 0.5m apart. During the mapping

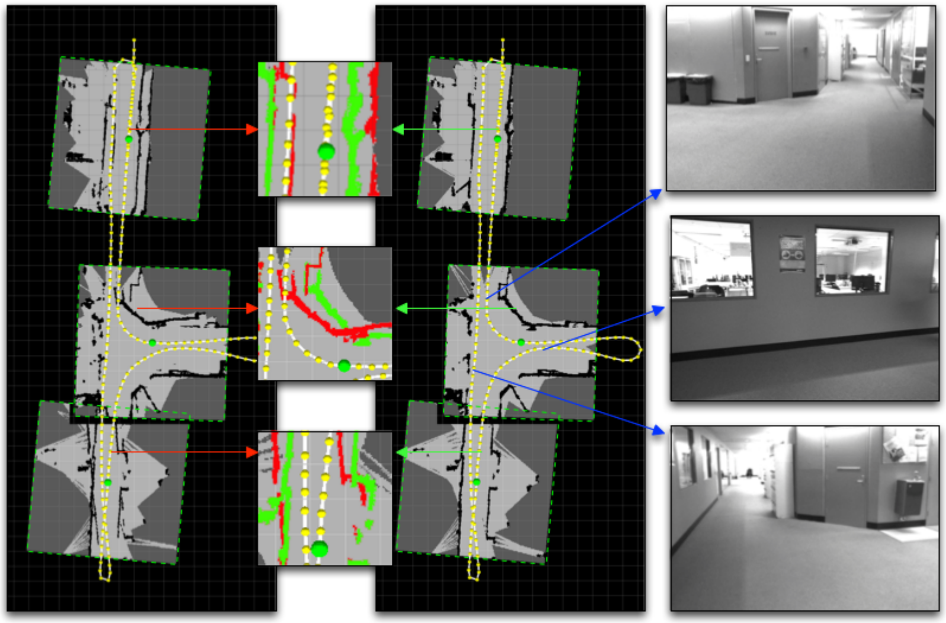

Fig. 6. Sample of local metric maps. Left: the grids created from unregistered point clouds. Centre: enlarged and cropped overlay of the resulting occupancy grids registered (green) and un-registered (red). Right: the same grids but after using 3D_NDT registration on the point clouds before creating the grids. The images on the far right show the registered region from multiple views as observed by the robot. Note that without multiple views occluded regions would be missing from the map.

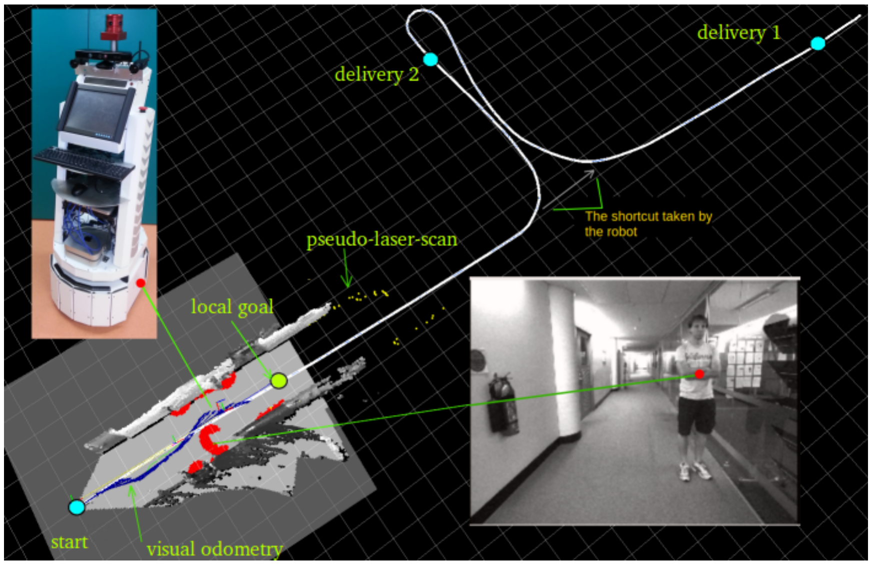

Fig. 7. A topometric map built from one traversal of *Level 11* S-block QUT Gardens Point. The robot starts from one end of the corridor and is required to navigate to the other end (delivery 1) and then return to the elevator area (delivery 2). Please see the attached video of the performing the navigation task. The figure shows a local map centred on node 5 of the global graph. Inside the local map: red represents inflated obstacles, yellow represents the pseudo-laser-scan generated using point clouds from live vision and blue represents the trajectory taken by the robot. The figure shows the local goal obtained from the planner and the two global delivery nodes.

phase the robot stores a point cloud for each node in the graph. When a new map centred on one of the nodes is requested, the point clouds associated with the nodes inside an area of $4 \times 4$m are retrieved to build an occupancy grid of size $10 \times 10$ cells. The clouds used to build local maps can be from distinct views of the same area. The use of 3D geometry enables successful registration of these different views where appearance techniques often fail. Fig. 6 shows sample of the local metric maps with and without the registration step. When no registration is used, the local occupancy grids contain misaligned walls which affect the performance of localization during autonomous navigation and the availability of valid plans through the maps.

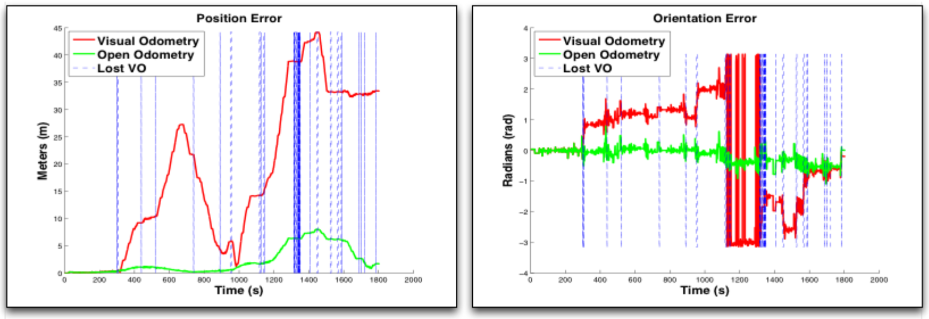

Fig. 8. Results of position and orientation error in Visual Odometry with and without open-loop correction as compared to the localised trajectory observed over 30 minutes of autonomous operation in narrow corridors. Note that the angular error in VO without correction *(red)* significantly increases during periods of VO failure *(blue dashed)*. In contrast the VO with open-loop correction dubbed 'Open Odometry' *(green)* performs much better as angular errors are not as severe and therefore the error in position is less divergent.

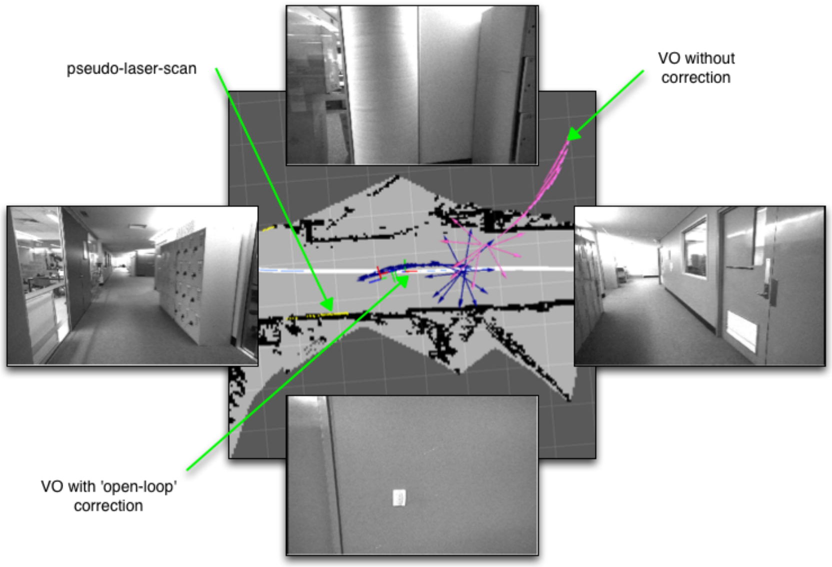

Fig. 9. Visualisation of the effects when visual odometry becomes 'lost' and requires correction. Pink: raw visual odometry. Blue: visual odometry supplemented with command velocity integration during visually ambiguous observations. An example of missing pose estimates due to visual odometry failure, which in-turn causes major corruption of the robots integrated path. The border images are select frames taken from the camera on-board the robot while turning 360 degree in a narrow corridor.

## B. Autonomous navigation

Using the topometric map, the robot was able to navigate autonomously for a period of 30 minutes which was limited by battery life. The robot was given three delivery goals to visit repeatedly in a cyclic manner. The average speed of the robot while navigating was 0.09m/s. The total distance covered was 156.2 meters. One of the most important group of parameters, which has a big effect on the navigation performance, was the particle filter localization parameters. The `pseudo-laser-scan`, published at 3Hz is significantly lower then the usual 10HZ for a real laser scan. This has some effect on localization and limits the speed of the robot. This in turn required the rotational update of the particle filter to be set to $0.5^o$ degrees and the translational movement required before performing a filter update was set to $0.1$ m. Please see the accompanying video of the robot performing the navigation task.

## C. Shortcuts and Obstacle Avoidance

In order to force the system to perform a shortcut in the local metric space, the above navigation experiment was repeated but instead of using the full graph we used only a single direction traversal of the main corridor. Fig. 7 shows a snapshot of the graph shown as a white line, without its nodes for the sake of clarity. The snap shot was taken while the robot autonomously navigated inside a local metric map centred on node number 5. The figure shows the occupancy grid of the local map with the associated 3D point clouds. The figure also shows a person standing in front of the robot, as an obstacle, and the reflection of his existence in the local map as an obstacle shown in red. As mentioned before, the robot dynamically adapts its path to avoid obstacles added to the costmap using the `pseudo-laser-scan` during navigation.

During navigation, the robot starts from one end of the corridor and is required to navigate to the other end (delivery 1) and then return back towards the lift lobby (delivery 2). Please see the accompanying video of the robot performing the navigation task. Fig. 1 demonstrates this case, where the local map is shown as a blue line and the global plan follows the graph, represented as a white line.

### D. Overcoming VO failures

Experimentation in our office environment found that VO failure is common (See Fig 8). This impacts the quality of our topological structure during mapping and our localization during navigation. Analysis shows that discontinuities in VO due to failure are typically brief periods highly correlated with increases in angular error. We found that during short periods of failure supplementing VO with open loop integration of velocity commands sent by the local planner dramatically increases the performance of all odometry dependant processes. Since no other sensors signal is substituted, we maintain the claim of vision only. Throughout the 30 minutes of autonomous navigation we found the average position error of un-corrected VO when compared to the localised position to be 18m . In contrast VO with failures corrected by open loop integration ('Open Odometry') achieves an average position error of less then 2.5m.

Figure 9 visualizes the divergence of odometry sources in position due to a brief angular discontinuity. In this example the robot performed a $360^o$ in place rotation as a recovery behaviour to clear obstacles. Because the corridor is narrow, the robot faces a featureless wall, shown in the figure as a sequence of images captured from the camera by the robot while turning. The point failure during rotation leads to divergent trajectory, creating a complete dependence on the particle filter to maintain an accurate pose of the robot, in practice requiring absurd levels of particle uncertainty. Whereas in the case of using the "open-loop" correction from the command velocity, the odometry reflected the $360^o$ degree turn. Please note that the figure shows the odometry sources initially offset due to prior visual odometry failures.

## V. Conclusions and future work

This paper presented an open-source framework that enables a mobile robot equipped with only a stereo vision sensor to create a topometric map of its environment and then use the map for autonomous navigation. The aim of the topometric map is to provide a user friendly representation of the environment where a human operator can choose a goal from a topological map consisting of nodes which can easily be labelled with semantic information. In addition, the robot can create local metric maps formed of varying data as the topology changes. The use of local metric maps enables the robot to plan paths / shortcuts within the free space surrounding the graph and perform obstacle avoidance using only vision.

The presented system uses a database to store point clouds attached to each node in a global graph. As future work, this dataset of clouds will serve as a base for persistent navigation within non-static environments. An updating mechanism will be deployed so the point cloud stored at each node in the graph is kept up-to-date with the current state of the environment.

## Acknowledgments

The authors would like to thank Elizabeth Murphy and Patrick Ross for helpful discussion and technical contributions.

## References

[1] K. Konolige and M. Agrawal, "Frameslam: From bundle adjustment to real-time visual mapping," *Robotics, IEEE Transactions on*, vol. 24, no. 5, pp. 1066–1077, 2008.

[2] M. Warren, D. McKinnon, H. He, and B. Upcroft, "Unaided stereo vision based pose estimation," in *Australasian Conference on Robotics and Automation*, 2010.

[3] R. Sim and J. J. Little, "Autonomous vision-based exploration and mapping using hybrid maps and rao-blackwellised particle filters," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, 2006, pp. 2082–2089.

[4] H. Lategahn, A. Geiger, and B. Kitt, "Visual slam for autonomous ground vehicles," in *International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[5] M. Cummins and P. Newman, "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[6] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, vol. 3, no. 3.2, 2009.

[7] A. Murillo, C. Sagues, J. Guerrero, T. Goedemé, T. Tuytelaars, and L. Van Gool, "From omnidirectional images to hierarchical localization," *Robotics and Autonomous Systems*, vol. 55(5), pp. 372–382, 2007.

[8] J.-L. Blanco, J. González, and J.-A. Fernández-Madrigal, "Subjective local maps for hybrid metric-topological slam," *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 64–74, 2009.

[9] B. Kuipers, "The spatial semantic hierarchy," *Artificial Intelligence*, vol. 119, pp. 191–233, 2000.

[10] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, and F. Savelli, "Local metrical and global topological maps in the hybrid spatial semantic hierarchy," vol. 5, New Orleans, USA, April 26 - May 1 2004, pp. 4845–4851.

[11] J. Blanco, J. Fernandez-Madrigal, and J. Gonzalez, "Toward a Unified Bayesian Approach to Hybrid Metric-Topological SLAM," vol. 24, pp. 259–270, 2008.

[12] K. Konolige, E. Marder-Eppstein, and B. Marthi, "Navigation in hybrid metric-topological maps," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2011, pp. 3041–3047.

[13] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," vol. 2, New York, NY, USA, June 17-22 2006, pp. 2161–2168.

[14] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o : A general framework for graph optimization," *Time*, pp. 3607–3613, 2011.

[15] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian Conference on Computer Vision*, Queenstown, New Zealand, November 2010.

[16] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[17] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[18] T. D. Stoyanov, M. Magnusson, H. Andreasson, and A. Lilienthal, "Fast and accurate scan registration through minimization of the distance between compact 3d ndt representations," *The International Journal of Robotics Research*, 2012.

[19] D. Fox, "KLD-sampling: Adaptive particle filters," in *In Advances in Neural Information Processing Systems 14*. MIT Press, 2001, pp. 713–720.