

A Unified Representation for Application of Architectural Constraints in Large-Scale Mapping

Paul Amayo, Pedro Piniés, Lina M. Paz, Paul Newman

Abstract—This paper is about discovering and leveraging architectural constraints in large scale 3D reconstructions using laser. Our contribution is to offer a formulation of the problem which naturally and in a unified way, captures the variety of architectural constraints that can be discovered and applied in urban reconstructions. We focus in particular on the case of survey construction with a push broom laser + VO system. Here visual odometry is combined with vertical 2D scans to create a 3D picture of the environment. A key characteristic here is that the sensors pass/sweep swiftly through the environment such that elements of the scene are seen only briefly by cameras and scanned just once by the laser. These qualities make for an ill-constrained optimisation problem which is greatly aided if architectural constraints can be discovered and appropriately applied. We demonstrate our approach in an end-to-end implementation which discovers salient architectural constraints and rejects false loop closures before invoking an optimisation to return a 3D model of the workspace. We evaluate the precision of this model by comparison to a ground truth provided by a 3rd party professional survey using high-end (static) 3D laser scanners.

I. INTRODUCTION

Within this work we are interested in the creation of large-scale dense and accurate 3D metric maps of both indoor and outdoor urban environments as the one shown in Figure 1. We utilise a platform equipped with a stereo camera and a 2D laser sensor for the creation of the dense 3D maps. The lasers are configured in a push-broom orientation resulting in the laser scans capturing a vertical snap-shot of the platform’s environment as shown in Figure 2. By combining these scans with the forward motion of the platform a 3D dense map of the surrounding environment can be created.

The motion model of the platform is generated from the stereo camera through Visual Odometry (VO). VO consists of frame-to-frame motion estimation from visual input, through a geometric hypothesise-and-test architecture and least-squares optimisation [1]. While the motion estimation from VO is accurate over small regions it accumulates error and ‘drifts’ over time. The accuracy of the maps generated using this method are intrinsically tied to the accuracy of the motion estimation and this drift reduces the accuracy of the maps. In this paper we present a framework that leverages higher-level features found in indoor and urban environments to create architectural constraints that improve the estimation of motion and subsequent accuracy of the dense maps.

In this framework a graph-formulation similar to that proposed by Lu and Milios [2] is employed. This involves the

Authors are from the Mobile Robotics Group, Dept. Engineering Science, University of Oxford, UK. {pamayo, ppinies, linapaz, pnewman}@robots.ox.ac.uk



Fig. 1: A large scale optimised map obtained as solution of the proposed graph optimisation on a urban environment that renders over a 2km trajectory (top). The combination of two different sensor modalities, a push broom laser and a stereo rig, allows us to create very dense coloured models of the traversed environment. Our main contributions is in the formulation of an unified framework to handle multiple architectural constraints driven by the different geometric features discovered in the environment (e.g. points, planes, etc). A close up of lower corner is depicted (bottom).

construction of a graph whose nodes represent robot poses or features/landmarks and the edges represent constraints between nodes [3]. In our system, the sensors pass swiftly through the environment so that scene elements are briefly seen by the cameras and scanned once by the laser. Architectural constraints in this case allow us to better estimate the motion of the platform and from this, more accurate dense maps of the workspace can be generated even when it has been observed briefly.

The main contribution of this paper is to offer a unified representation of architectural constraints in a graph optimisation formulation. To do that, we extend the Symmetries and Perturbations (SP) model [4] to a graph optimisation framework. The SP model allows for the unified representation of

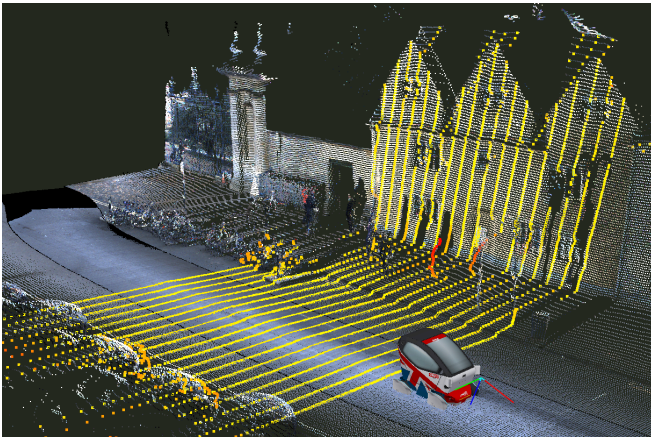


Fig. 2: Figure showing our system collecting non-overlapping laser scans using a push-broom 2D Hokuyo laser. The motion of the platform is estimated using VO.

geometric entities/features by using a reference coordinate frame attached to them. In addition, we present a method of validating loop closures found from the camera’s image stream based on the generated geometric features.

In section II we present the related work. Section III describes the representation of the architectural constraints, how to incorporate them into the graph formulation and how to seamlessly use them in the optimisation process. In section IV we explain the generation of the architectural constraints and they can be automatically associated. Section V presents the results and in section VI we draw the conclusions.

II. RELATED WORK

Within the Simultaneous Localisation And Mapping (SLAM) literature attempts to include higher-level features to the problem of localising a robot within a constructed map have been reported with the majority of the work concentrating on planar features.

Early work into this was reported by Weingarten and Siegwart [5] who incorporated planes into a filter-based approach to the SLAM problem. Using a 3D laser scanner planes were generated through a RANSAC algorithm [6] and included into an Extended Kalman Filter (EKF)-formulation via a Symmetry and Perturbations (SP) model [4]. A similar EKF approach was also taken in [7], [8] and [9]. While the inclusion of these features improved the solution of the problem the intrinsically large computational cost associated with using an EKF severely limited the application of this approach.

Planar features have also been incorporated to graph-based SLAM approaches within the literature with the major reported difference being the representation of the planes and the effect it has on the optimisation process. Lee et al. [10] utilised planes obtained from a RGB-D camera for indoor mapping by representing them within the spherical coordinate system prior to graph optimisation. The use of this spherical coordinate system however leaves the optimisation subject to singularities as it is a minimal representation of

orientation [11]. Trevor et al. [12] and Taguchi et al. [13] represented planar features using the plane equations, which are the normal and distance from the origin. As compared to the spherical coordinate system which is a minimal representation, the plane equations are an over-parametrised representation and hence not allow for unconstrained optimisation. Using the Levenberg-Marquadt technique optimisation was carried out, this technique includes an internal regulariser which constrains the optimisation at the cost of convergence speed. Kaess [14] includes planes obtained from a RGB-D camera to the graph-formulation by defining a minimal representation for the planes similar to that used to represent quaternions.

In this work all geometric features are represented using the SPmodel framework. This model allows for unconstrained optimisation without the need to define a special representation for geometric entities like planes. In addition, it is not subject to singularities as is shown in the following sections.

III. SPMODEL

In the SPmodel a geometric entity e is represented by a reference coordinate frame \mathbf{T}^e attached to it. This reference encapsulates both the entity’s position and orientation as expressed below.

$$\mathbf{T}^e = \begin{pmatrix} \mathbf{R}^e & \mathbf{t}^e \\ \mathbf{0}^T & 1 \end{pmatrix} \in SE(3) \quad (1)$$

Where $\mathbf{R}^e \in SO(3)$ is the rotation matrix and $\mathbf{t}^e \in \mathbb{R}^3$ is the 3 dimensional translation vector of the transformation. The SPmodel is based on the following two principles:

- The error in the position of a geometric entity is represented locally
- A geometric entity is not allowed to move locally along its underlying symmetries

In our case to accomplish the first requirement we denote the current estimate of the location of an entity by $\hat{\mathbf{T}}^e$ whereas its local error is represented by the differential Δ^e . The current estimation $\hat{\mathbf{T}}^e$ belongs to the Special Euclidean $SE(3)$ Lie Group while the error Δ^e is defined in the Lie algebra $\mathfrak{se}(3)$ which allows for manifold optimisation (no constraints required). The true position of the entity is then given by:

$$\mathbf{T}^e = \hat{\mathbf{T}}^e \oplus \Delta^e \quad (2)$$

with \oplus representing the composition of the transformations described later.

The Lie algebra $\mathfrak{se}(3)$ is the tangent space of $SE(3)$ at the identity. The elements of this algebra are the 6-vectors $(\mathbf{w}, \mathbf{v})^T$ where $\mathbf{w} = (w_x, w_y, w_z)$ is the axis-angle representation of rotation and \mathbf{v} is a rotated version of the translation \mathbf{t} . An exponential mapping can be used to map the elements of the $\mathfrak{se}(3)$ algebra back to the $SE(3)$ group as shown in Equation 3.

$$\exp_{SE(3)}(\mathbf{w}, \mathbf{v}) = \begin{pmatrix} \exp_{SO(3)}(\mathbf{w}) & \mathbf{V}\mathbf{v} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{pmatrix} \quad (3)$$

with

$$\exp_{SO(3)}(\mathbf{w}) = \mathbf{I} + \frac{\sin(\theta)}{\theta}[\mathbf{w}]_{\times} + \frac{1 - \cos(\theta)}{\theta^2}[\mathbf{w}]_{\times}^2$$

$$\mathbf{V} = \mathbf{I} + \frac{1 - \cos(\theta)}{\theta^2}[\mathbf{w}]_{\times} + \frac{\theta - \sin(\theta)}{\theta^2}[\mathbf{w}]_{\times}^3$$

$$\theta = \|\mathbf{w}\|^2$$

and $[\cdot]_{\times}$ is an operator which maps a 3-vector to its skew-symmetric matrix.

The components of the error Δ^e are given by $d\mathbf{v}_{\bullet}$ for the translation and $d\mathbf{w}_{\bullet}$ for the rotation where the elements in the subscript describe the allowed axis (x, y, z) along which we can translate or rotate to correct the error.

To fulfil the second requirement of the model the axis along which a particular entity can be locally moved depends on the symmetries and real Degrees Of Freedom (DOF) of the entity. Table I shows the corresponding coordinate frames attached to a point, line, plane and robot pose entities. The corresponding differential movements allowed are also defined in the table with the motion composition given by:

$$\hat{\mathbf{T}}^e \oplus \Delta^e \stackrel{def}{=} \hat{\mathbf{T}}^e \exp(\Delta^e) \quad (4)$$

For the purpose of this work we focus primarily on the inclusion of plane entities, which can be used to generate architectural constraints in the context of the large data sets being analysed. As can be seen from Table I the coordinate frame attached to the plane has its x and y vectors lying along the plane with the z vector being the normal of the plane. From this it can be seen that any translation in the x and y direction as well as rotation around the z axis of this reference will not change the plane equation. Leaving it with the three degrees of freedom. By applying the respective rotations sequentially we have full control of the symmetry of the entities. In this way the $\exp(\Delta^e)$ fully reflects the degrees of freedom available to the specific entity as shown in Table I.

A. Optimisation

The optimisation problem solves for a configuration of parameters $(\mathbf{T}^e)^*$ that explains all the measurements by minimising the following cost function:

$$\mathbf{F}(\mathbf{T}^e) = \sum_{i,j \in C} \mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij})^T \Omega_{ij} \mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij}) \quad (5)$$

where $\mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij})$ is an error function that measures how well the parameter blocks \mathbf{T}_i^e and \mathbf{T}_j^e satisfy the constraint whose mean is given by \mathbf{z}_{ij} and covariance by Ω_{ij} . The set C represents the pairs of geometric entities for which a constraint exists.

The error function \mathbf{e}_{ij} used for all constraints in this

framework is defined as:

$$\mathbf{e}_{ij} \stackrel{def}{=} \mathbf{e}_{ij}(\mathbf{T}_i^e, \mathbf{T}_j^e, \mathbf{z}_{ij}) \quad (6)$$

$$\mathbf{e}_{ij} = \log(\mathbf{E}^{ee}) \quad (7)$$

The $\log(\cdot)$ function is the inverse of the exponential map defined in Equation 3 and maps the Lie group $SE(3)$ to its corresponding Lie algebra $\mathfrak{se}(3)$. In our implementation of our SP model we utilise the residual matrix \mathbf{E}^{ee} to represent the error between different geometric entities.

1) *Pose-Pose residual*: The first constraint used in this framework is that between two poses where the residual matrix is given as:

$$\mathbf{E}^{xx} = (\mathbf{T}_j^x)^{-1} \mathbf{T}_i^x \mathbf{z}_{ij} \quad (8)$$

with \mathbf{z}_{ij} defined as the relative transformation between the poses.

2) *Point-Plane residual*: The first architectural constraint considered is that between a point i and a plane j whose corresponding transformations are given by:

$$\mathbf{T}^{\pi_j} = \begin{pmatrix} \mathbf{n}_x^{\pi_j} & \mathbf{n}_y^{\pi_j} & \mathbf{n}_z^{\pi_j} & \mathbf{p}^{\pi_j} \\ 0 & 0 & 0 & 1 \end{pmatrix} \in SE(3) \quad (9)$$

$$\mathbf{T}^{p_i} = \begin{pmatrix} \mathbf{I} & \mathbf{p}^i \\ \mathbf{0} & 1 \end{pmatrix} \in SE(3) \quad (10)$$

These arise when a point lies on a plane or the distance z_{ij} between a point and a plane is measured. The corresponding residual matrix is given as:

$$\mathbf{E}^{\pi p} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{n}_z^{\pi_j} \cdot (\mathbf{p}^i - \mathbf{p}^{\pi_j}) - z_{ij} \\ 0 & 0 & 0 & 1 \end{pmatrix} \in SE(3) \quad (11)$$

3) *Plane-Plane Angle residual*: Another architectural constraint arises when the angle z_{ij} between two planes is known, like for example for two parallel or orthogonal walls indoors. The corresponding residual matrix between the planes \mathbf{T}^{π_i} and \mathbf{T}^{π_j} is:

$$\mathbf{E}^{\pi \pi} = \begin{pmatrix} \cos(\phi) & -\sin(\phi) & 0 & 0 \\ \sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \in SE(3) \quad (12)$$

$$\phi = z_{ij} - \text{acos}(\mathbf{n}_z^{\pi_i} \cdot \mathbf{n}_z^{\pi_j}) \quad (13)$$

4) *Error minimisation*: With the error function defined in Equation 7 a numerical solution to perform least squares optimisation using the Gauss-Newton (GN) or Levenberg-Marquardt (LM) algorithms can be derived [15]. In this section we refer to Δ^e as the state vector consisting of all the differentials with Δ_i^e referring to the individual differential of entity i . Similarly we define the shorthand

$$\mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \Delta^e) \stackrel{def}{=} \mathbf{e}_{ij}(\hat{\mathbf{T}}_i^e \oplus \Delta_i^e, \hat{\mathbf{T}}_j^e \oplus \Delta_j^e, \mathbf{z}_{ij}) \quad (14)$$

The idea employed by LM and GN is to approximate the error function by linearising around the current estimate





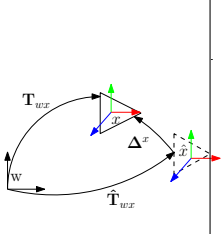
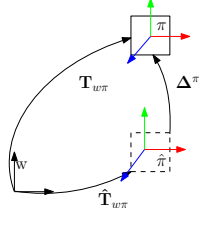
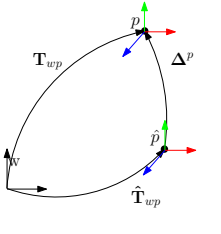
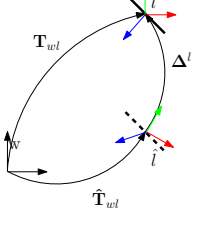
Entity	Pose	Plane	Point	Line
Reference				
Graph constraint				
$\exp(\Delta^e)$	$\exp(d\mathbf{v}_{xyz}d\mathbf{w}_{xyz})$	$\exp(d\mathbf{v}_z)\exp(d\mathbf{w}_y)\exp(d\mathbf{w}_x)$	$\exp(d\mathbf{v}_{xyz})$	$\exp(d\mathbf{v}_{yz})\exp(d\mathbf{w}_z)\exp(d\mathbf{w}_y)$
DOF	6	3	3	4

TABLE I: Description of the proposed geometric entities and constraints: a point, line, plane and pose entity with their respective representation within the proposed lie algebra framework. The attached reference frame as well as the graphical representation of the entities are described together with the construction of $\exp(\Delta)^e$, the exponential mapping of the differential to preserve the symmetry of each of the individual entities.

using a first order Taylor expansion.

$$\mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \Delta^e) \approx \hat{\mathbf{e}}_{ij} + \mathbf{J}_{ij}\Delta^e \quad (15)$$

The Jacobian \mathbf{J}_{ij} is given by

$$\mathbf{J}_{ij} = \left. \frac{\partial \mathbf{e}_{ij}(\mathbf{T}^e \oplus \Delta^e)}{\partial \Delta^e} \right|_{\Delta^e=0} \quad (16)$$

Building up from Equation 5.

$$\begin{aligned} F_{ij}(\mathbf{T}^e) &= \mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \Delta^e) \Omega_{ij} \mathbf{e}_{ij}(\hat{\mathbf{T}}^e \oplus \Delta^e) \\ &\approx (\mathbf{e}_{ij} + \mathbf{J}_{ij} \oplus \Delta^e) \Omega_{ij} (\mathbf{e}_{ij} + \mathbf{J}_{ij} \oplus \Delta^e) \\ &= \underbrace{\mathbf{e}_{ij}^T \Omega_{ij} \mathbf{e}_{ij}}_{c_{ij}} + 2 \underbrace{\mathbf{e}_{ij}^T \Omega_{ij} \mathbf{J}_{ij}}_{b_{ij}} \Delta^e \\ &\quad + (\Delta^e)^T \underbrace{\mathbf{J}_{ij}^T \Omega_{ij} \mathbf{J}_{ij}}_{H_{ij}} \Delta^e \\ &= c_{ij} + 2b_{ij}\Delta^e + (\Delta^e)^T H_{ij}\Delta^e \end{aligned}$$

Applying this to the full Equation

$$\begin{aligned} \mathbf{F}(\mathbf{T}^e) &= \sum_{\langle i,j \rangle \in C} F_{ij}(\mathbf{T}^e) \\ &= \sum_{\langle i,j \rangle \in C} c_{ij} + 2b_{ij}\Delta^e + \Delta^e H_{ij}\Delta^e \\ &= \mathbf{c} + 2\mathbf{b}^T \Delta^e + (\Delta^e)^T \mathbf{H} \Delta^e \end{aligned}$$

This form is obtained by setting $\mathbf{c} = \sum c_{ij}$, $\mathbf{b} = \sum b_{ij}$ and $\mathbf{H} = \sum H_{ij}$. The solution can be found as below

$$\mathbf{H}\Delta^e = -\mathbf{b} \quad (17)$$

The update is obtained by composing the solution Δ^e obtained in the current iteration k with the previous estimate $\hat{\mathbf{T}}_k^e$.

$$\hat{\mathbf{T}}_{k+1}^e = \hat{\mathbf{T}}_k^e \oplus \Delta^e \quad (18)$$

The linearisation, solution and update step can be iterated

through until convergence. In our implementation we use the Ceres solver [16] with automatic differentiation to calculate the Jacobians in Equation 16.

IV. MAPPING FRONT END

With the optimisation procedure for the geometric features described it remains to generate and incorporate features, in the case of this paper plane features, into the framework and this is dealt with via our front-end. The generation of planes is done through segmentation of a 3D point-cloud that represents the platform's environment. This 3D point-cloud can be retrieved directly using 3D sensors such as RGB-D cameras or using a 2D laser sensor in a push-broom configuration that is undergoing known motion. While RGB-D cameras have been shown to produce accurate point-clouds their operation is restricted to mostly indoor environments with very limited operation available outdoors. The proposed method is not constrained in this way and can be used in all environments.

To generate the point-cloud the trajectory is divided into sections of user-defined distance with the poses at the start of the sections defined as Key Poses $K_{i \in \{1, \dots, n\}}$ where n is the number of sections in the trajectory. Within these sections the error produced by the effect of drift on the VO is small enough such that the point cloud generated within the section can be considered accurate. Plane segmentation is therefore carried out over the section's point cloud followed by data association between different sections.

A. Constraint Generation

With the 3D point cloud over a section defined the next step is segmenting the point cloud into planes. To do this a curvature-based algorithm is used [17]. The curvature algorithm utilises the relationship between a point in the point cloud and the small local region neighbouring it. If the local region is planar the curvature of the region is expected to be low and the normals of the points within the region will

also point in the same direction. By comparing the residual and normals of a point to certain user-defined thresholds it is possible to ‘grow’ a plane from the local regions.

The addition of the reference coordinate frame used to incorporate the plane into the SPmodel requires the definition of two orthogonal vectors that lie on the plane $\mathbf{n}_x^\pi, \mathbf{n}_y^\pi$, a normal vector \mathbf{n}_z^π and a point \mathbf{p}^π belonging to the plane as shown in Table I and defined in Equation 9.

B. Data Association

In this paper there are two elements of data association that are considered, that between planes found in successive sections and those between planes in sections that have loop closures detected by FABMAP [18]. Planes are associated if they are coplanar.

All the planes generated in a section are represented in the frame of their corresponding Key Pose. To compare two planes represented in different sections with Key Poses K_a and K_b a transformation $\mathbf{T}_{K_a K_b} \in SE(3)$ is used to represent the planes in section b in the frame of Key Pose a. For successive sections we use VO to calculate $\mathbf{T}_{K_a K_b}$ while for the loop closures we use pairs of stereo images matched by FABMAP to compute the transformation.

The coplanar test between two planes \mathbf{T}^{π_a} and \mathbf{T}^{π_b} can be performed by comparing the angle θ_{ab} and distance d_{ab} between the planes with user-defined angle and distance thresholds respectively.

$$\theta_{ab} = \arccos((\mathbf{n}_z^{\pi_a})^T \cdot (\mathbf{R}_{K_a K_b} \cdot \mathbf{n}_z^{\pi_b})) \quad (19)$$

$$d_{ab} = \mathbf{n}_z^{\pi_a} (\mathbf{c}^{\pi_a} - (\mathbf{R}_{K_a K_b} \cdot \mathbf{c}^{\pi_b} + \mathbf{t}_{K_a K_b})) \quad (20)$$

$\mathbf{c}^\pi \in \mathbb{R}^3$ is the centroid of a plane. If θ_{ab} and d_{ab} are less than their respective thresholds the planes are coplanar and can be associated as one.

The association of geometric features found in a loop closure presents us with an opportunity to geometrically validate the loop closures obtained via FABMAP, which uses the appearance of stereo images. This is done via a plane-compatibility test which checks whether the planes and configuration of planes observed from both areas that constitute the loop closure are consistent by analysing the data associations.

V. RESULTS

To test this proposed framework four independent surveys were carried out. We generated planes from the dense laser map and added them as constraints for the optimisation as shown in Figure 5. The results of the two surveys of an indoor environment, the Acland building in Keble College Oxford, and the two of an outdoor environment, a triangular loop in the Jericho area Oxford are shown in Figure 3. Column 1 of the table shows the dense maps created using the motion estimation provided by VO, with the maps created after the optimisation being compared in column 2.

For the indoor surveys taken the ground truth is available via a professional survey taken of the same environment. The comparison of these surveys to the professional survey

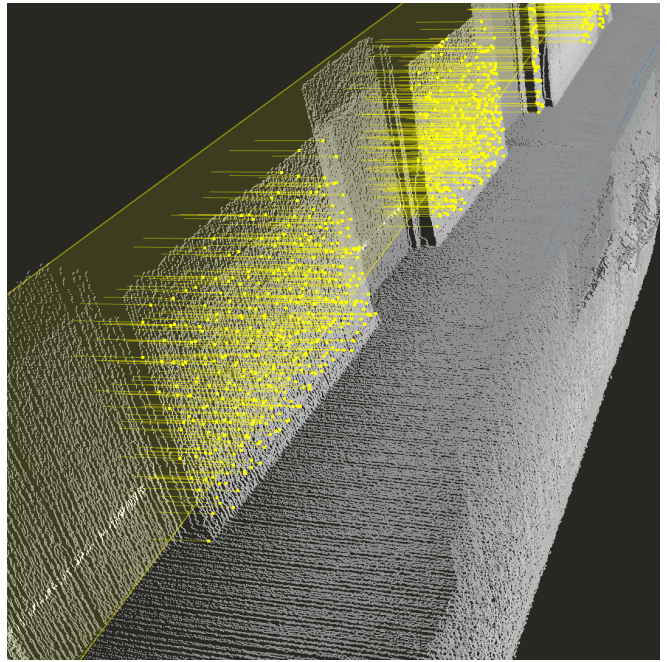


Fig. 5: This figure shows an automatically generated constraint for a set of laser points that belong to a common plane. Observe that laser points belonging to doors (interleaved between the constrained walls) are not taken into account in the plane constraint. Consecutive 2D scans are not related except through the plane constraint as they come from different poses. As a rigid transformation exists between a laser point and the pose it was taken in, we can constrain the trajectory by constraining the laser points.

after alignment via Iterative Closest Point (ICP) algorithm is shown in the second column of Table 3 coupled with the histogram of the error. For the first survey a Root Mean Square (RMS) error of 0.1125 metres was observed for the points with the platform travelling for approximately 200 metres. The second denser survey, as the platform was moved at a slower speed, reported a RMS error of 0.2257 metres over a similar distance travelled. The greatest errors were reported in sections where windows/doors were opened in the ground truth survey but not in our indoor surveys.

In the outdoor environment however ground truth data was not available, for the comparison the two outdoor surveys were compared against each other with the RMS error of this comparison coming to 0.943 metres with the platform covering two loops of approximately 0.9 kilometres each. While this is a significant reduction in error of the maps, it highlights the difference between indoor and outdoor maps. In the indoor case the entire environment can be described by a configuration of planes i.e two sets of orthogonal planes found in corridors as shown in Figure 6 making the final result very accurate while this is not the case in the outdoor environment limiting the final accuracy of the model.

Table II provides details of the number of architectural constraints used during each of the surveys carried out in this unified optimisation framework. Additionally, we list the time required for each iteration in each of the evaluated datasets.

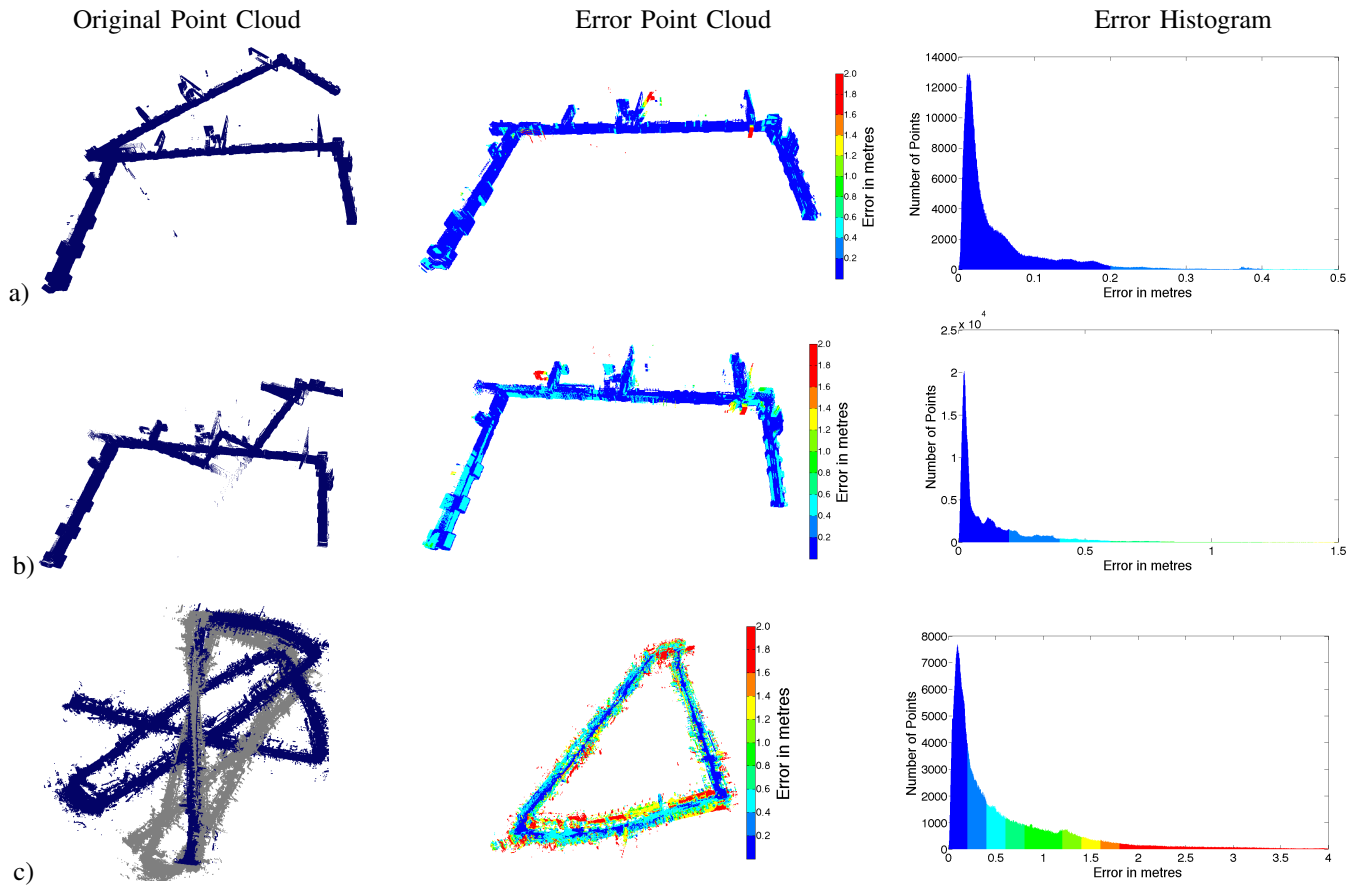


Fig. 3: Figure showing the results of 4 surveys taken and optimised with this framework. The original surveys created using VO are shown in column 1, while a comparison of the surveys after optimisation with architectural constraints together with their subsequent histograms is shown in columns 2 and 3 respectively. Rows 1 and 2 present results from 2 independent indoor surveys of the Acland building in Oxford where the ground truth dense map is available courtesy of a professional survey and used for the comparisons. In row 3 two different independent outdoor maps are presented (grey and blue) of the Jericho triangle in Oxford. Since ground truth is not available the comparison is done between the 2 point clouds optimised independently.

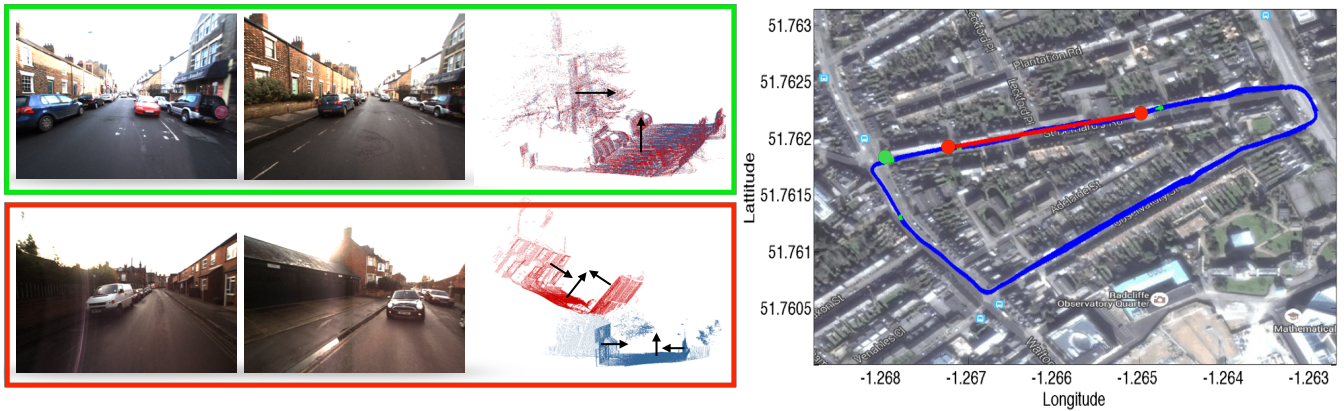


Fig. 4: Figure showing the trajectory of the platform covering two loops of the Jericho triangle of Oxford overlaid on a GPS map. The loop closures generated from FABMAP are geometrically validated using our proposed method. A matched image pair together with its respective transformed 3D sections is shown for the case when the loop closure is validated (green square) and rejected (red square). For the rejected loop closure the transformation of the matched section (red) to the frame of the original section (blue) via the Key Pose transform $\mathbf{T}_{K_2K_1}$ does not result in any overlap between the sections, thus any planes generated from these segments will not be associated and it thus fails the plane association test. In the validated loop closure the Key Pose transformation results in the overlap of the matched (red) and the original (blue) 3D sections, and the subsequent planes generated from this section overlap and complete the plane association test.

Experiment	Pose-Pose	Plane-Point	Plane-Plane	No of Planes	Time per iteration (s)
Acland 1	24523	84524	13	186	0.9
Acland 2	62421	252124	9	194	3.168
Jericho 1	114816	20860	0	125	3.163
Jericho 2	112164	21869	0	133	2.895

TABLE II: Number of constraints used in each of the surveys utilising the proposed unified optimisation framework.

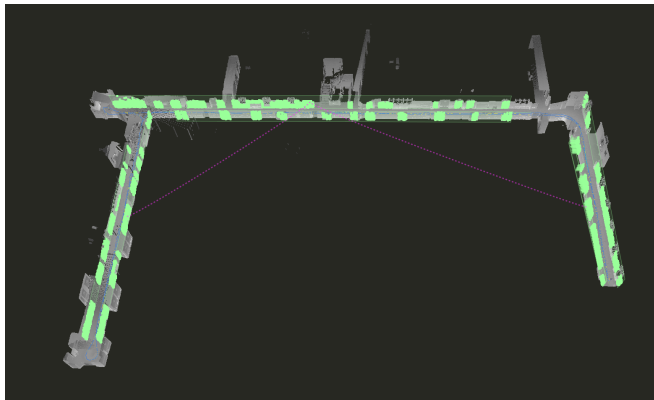


Fig. 6: This figure shows the configuration of planes from the survey of the Acland building. The structure of this environment can be described by the automatically generated orthogonal relationships (in purple) between the three sets of planes (in green) forming the corridors.

The trajectory of the outdoor survey is presented in Figure 4 and with the platform traversing two loops of the Jericho triangle we utilise FABMAP to generate matched images and calculate a loop closure transformation from the matched pair of stereo images. We then geometrically validate these loop closures with our proposed technique and results from a rejected and validated loop closure transformation are shown in Figure 4.

VI. CONCLUSION

We have presented a framework for the creation of dense 3D maps of indoor and outdoor urban environments using automatically detected architectural constraints. These constraints are required since our platform is equipped with a laser oriented in a push-broom configuration where scans do not overlap. Our motion source is given by a visual odometry system. As main contribution of the paper we have presented a unified representation of the architectural constraints that can be seamlessly integrated into a graph optimisation framework. Using this framework, we have demonstrated the generation, representation, association and optimisation of high-level features such as planes to improve the estimation of the trajectory of our system and the creation of accurate maps. This work has been experimentally validated through two indoor datasets obtaining a final accuracy comparable to a professional survey and reporting similar results in outdoor environments.

VII. ACKNOWLEDGEMENTS

The authors acknowledge the following funding sources. Paul Amayo is funded by the Rhodes Trust. Paul Newman is supported by EPSRC Programme Grant EP/M019918/1.

REFERENCES

- [1] D. Nistér, O. Naroditsky, and J. Bergen, “Visual odometry,” in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1. IEEE, 2004, pp. 1–652.
- [2] F. Lu and E. Milios, “Globally consistent range scan alignment for environment mapping,” *Autonomous robots*, vol. 4, no. 4, pp. 333–349, 1997.
- [3] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, “A tutorial on graph-based slam,” *Intelligent Transportation Systems Magazine, IEEE*, vol. 2, no. 4, pp. 31–43, 2010.
- [4] J. Castellanos, J. Montiel, J. Neira, J. D. Tardós *et al.*, “The spmap: A probabilistic framework for simultaneous localization and map building,” *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 5, pp. 948–952, 1999.
- [5] J. Weingarten and R. Siegwart, “3d slam using planar segments,” in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*. IEEE, 2006, pp. 3062–3067.
- [6] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [7] F. Servant, E. Marchand, P. Houlier, and I. Marchal, “Visual planes-based simultaneous localization and model refinement for augmented reality,” in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE, 2008, pp. 1–4.
- [8] A. P. Gee, D. Chekhlov, A. Calway, and W. Mayol-Cuevas, “Discovering higher level structure in visual slam,” *Robotics, IEEE Transactions on*, vol. 24, no. 5, pp. 980–990, 2008.
- [9] J. Martínez-Carranza and A. Calway, “Unifying planar and point mapping in monocular slam,” in *BMVC*. Citeseer, 2010, pp. 1–11.
- [10] T.-k. Lee, S. Lim, S. Lee, S. An, and S.-y. Oh, “Indoor mapping using planes extracted from noisy rgb-d sensors,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 1727–1733.
- [11] F. S. Grassia, “Practical parameterization of rotations using the exponential map,” *Journal of graphics tools*, vol. 3, no. 3, pp. 29–48, 1998.
- [12] A. J. Trevor, J. Rogers, and H. I. Christensen, “Planar surface slam with 3d and 2d sensors,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3041–3048.
- [13] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng, “Point-plane slam for hand-held 3d sensors,” in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 5182–5189.
- [14] M. Kaess, “Simultaneous localization and mapping with infinite planes.”
- [15] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.
- [16] S. Agarwal, K. Mierle, and Others, “Ceres solver,” <http://ceres-solver.org>.
- [17] T. Rabbani, F. van den Heuvel, and G. Vosselmann, “Segmentation of point clouds using smoothness constraint,” *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, no. 5, pp. 248–253, 2006.
- [18] 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.