Global Localization in Meshes

Marc Dreher[1], Hermann Blum[1], Roland Siegwart[1], and Abel Gawel[1]

[1]Autonomous Systems Lab, ETH Zurich, Switzerland

dreherm@ethz.ch, blumh@ethz.ch, rsiegwart@ethz.ch, gawela@ethz.ch

Abstract -

Safely waking up a robot at an unknown location and subsequent autonomous operation are key requirements for on-site construction robots. In this regard, single-shot global localization in a known map is a challenging problem due to incomplete observations of the environment and sensor obstructions by unmapped clutter. In this work, we address global localization of sparse multi-beam LiDAR measurements in a 3D mesh building model, a typical setup for construction robots. Our solution extracts and summarizes planes from the LiDAR scan and matches them to the building mesh. We evaluate different options for the registration problem, and evaluate the system on simulated and real-world datasets. The best performing system uses a combination of the Randomized Hough Transform (RHT) and a modified version of the Plane Registration based on a Unit Sphere (PRRUS) algorithm. For sparse and noisy robotic sensors, our system outperforms contemporary systems like GoICP by a large margin.

Keywords -

Construction Robotics; Plane Extraction/Matching; Global Localization

1 Introduction

Localisation in 3D building models with onboard sensors is a necessary skill for autonomous construction robots, and it finds application even outside of construction tasks e.g. for indoor service robots. In this work, we consider a particular part of the localisation task, the robot wake-up problem: A robot has to find its location in a map without any prior knowledge, e.g. because its localisation routines diverged or it was just switched on. While it may be safe for a small household robot to blindly explore and risk collisions while performing the localization, this is too dangerous for e.g. heavy construction robots. Ideally, these could perform one-shot global localization at the location of wake-up. This is a difficult problem as the robot may only partially observe the environment and have incomplete information in its map, e.g. the building model only represent the raw state of a environment without equipment, temporary structures, or people. Fortunately, it is sufficient to find an approximate guess of the actual robot pose in a global map. Local methods (that require such an initial guess) then enable fine registration.

In this paper we consider the problem of 3D global localization in meshes, which is a common representation for buildings in architecture and can be easily and automatically created from ubiquitous 2D floor plans. However, they do not contain any visual information, ruling out visual global localization such as [1], and also generally do not contain the necessary information for semantic localization methods [2]. Better suited for the presented problem are methods based on 3D geometry [3, 4, 5]. Unfortunately, as these were designed for localization on robot maps instead of building meshes, we found that they do not gracefully account for data mismatches due to clutter, have long processing times, or are prone to converge to (wrong) local minima. We investigate a solution to localize sparse LiDAR scans in meshes using a plane extraction and matching algorithm. We propose and evaluate different methods for the individual components of such a localization system, and show its superior performance against state-of-the-art methods operating on point clouds [3] and volumetric features [5]. Our experiments highlight in particular the difficulties of on-board sensors like sparse LiDARs, as opposed to e.g. dense 3D scanners used in surveying, and show a need to use specific methods for such sensors. Overall, we present the following contributions:

- Heterogeneous global localization system to localize sparse and noisy 3D point clouds in 3D mesh models
- A modified version of the PRRUS algorithm described in [6], which does not maximize the best alignment of two sets of planes by checking all possible assignments, but assigns three orthogonal planes in the scan to the map planes and checks this assignment for consistency
- An extensive ablation study of our design choices on simulated and real-world data

2 Related Work

The problem of globally localizing between 3D data is well studied in literature. Existing approaches can be categorised into local descriptor-based approaches, Iterative Closest Point Algorithm (ICP)-based approaches, Neural Network-based, object-based, and topological approaches, e.g., using plane-matching.

2.1 Registration based on Local Descriptors

Methods facilitating local descriptors typically aim to extract features of selected keypoints from point clouds to form correspondences and estimate transformations [7]. This is inspired by the 2D approach well studied in image-based place recognition, using local descriptors such as SIFT, SURF, FREAK, BRISK, BRIEF and ORB descriptors [1]. 3D approaches are e.g. (I)FSD, FPFH and (I-)SHOT, forming compact volumetric descriptors. However, these features perform best on dense point clouds and only to a limited extent on sparse point clouds [8]. The back-end for calculating the relative transformations is typically based on Random Sample Consensus (RANSAC) or defined as a cost minimization problem[4, 9, 10].

2.2 Iterative Closest Point Algorithms

Numerous variants of ICP exist for the local refinement of a localization given a good inital guess [11]. Furthermore, several extensions were proposed to apply the ICP algorithm to global localization. Fitzgibbon [12] use the Levenberg-Marquardt [13] optimizer to escape local minima which are the primary source of ICP failing on arbitrary global localization tasks. Boehnke and Otesteanu [14] perform a coarse-to-fine alignment to find a good prior for ICP. WP-ICP [15] pre-processes the raw data into sets of corner and surface points, which are aligned separately. GoICP [3], which we evaluate in our experiments, uses a branch-and-bound scheme to divide the space of all transformations along upper and lower bounds[3]. It also guarantees the finding of a global minimum given sufficiently long run-times.

2.3 Neural Networks in Registration

Ratz et al. [16] train a 3D Convolutional Neural Network (CNN) to globally localize sparse 3D LiDAR scans in dense point cloud maps. In general, trained descriptors or matchers may lead to strong dependence on the training data, with unpredictable behaviour outside of their trained domain. Scan2CAD [17] uses a CNN to match candidate keypoints between RGB-D scans and meshes.

2.4 Object-based Registration

Several object-based methods use registration of meshes for localisation. Feng et al. [18] match and align objects with a database of class-representative object meshes, but require multiple views. Further works match line segments extracted from a scan. He and Hirose [19] use information about the geometric relations between lines in an interpretation tree for matching, whereas Micusik and Wildenauer [20] match known structures of lines with known structures using efficient matching procedures. Wang et al. developed a long-range localization procedure specifically for a shopping centers using names of visible stores [21].

2.5 Plane Extraction and Matching

Another approach involves the extraction of planes and the calculation of the relative transformation between the data by plane matching. To extract planes from a scan, [22] use a 3 dimensional version of the Hough Transformation. The authors also introduce various designs for the accumulator, which is crucial for the quality of the plane extraction. Fernández-Moral et al. [10] and Pathak et al. [6] use a region growing algorithm for the same task. Plane Sweeping [23], derives a transformation by constructing an axis of rotation through randomly selected points and generating a histogram from it. After extracting the planes, they are then interpreted by e.g. detecting and reconstructing openings [24], calculating plane features [10] or labeling occluded regions on the wall [25] to obtain a more descriptive representation of the planes. Fernández-Moral et al. [10] then construct a graph from the planes and the relative distances between two planes and match subgraphs with the help of a cost function.

We take inspiration from these works to develop a complete pipeline that includes plane extraction from the scan and matching to the mesh of the building model.

3 Method

The presented system (Fig. 1) localises sparse 3D LiDAR scans in 3D building mesh models. In the following, we describe how the LiDAR scan is preprocessed, planes are extracted from both the building mesh and the scan, subsequently matched to find a global localisation, which can finally be refined by local ICP based registration.

3.1 Point cloud filtering

LiDAR data from real environments is subject to different kinds of noise and outliers. We apply a voxel centroid filter that yields even density of the LiDAR scans, by voxelizing data in a fixed grid and merging points within each voxel into the centroid of all contained points.

3.2 Plane Extraction

Extracting the planes from the 3D mesh model is straightforward, i.e., merging neighbouring mesh cells based on parallel surface normals. However, for sparse 3D LiDAR scans, it remains a challenging problem. Due to the sparseness of the scans, the scene can no longer be completely reconstructed, which also makes it difficult to filter out clutter and noise. We evaluate different options, i.e., RANSAC-based[27], region growing-based [28], and

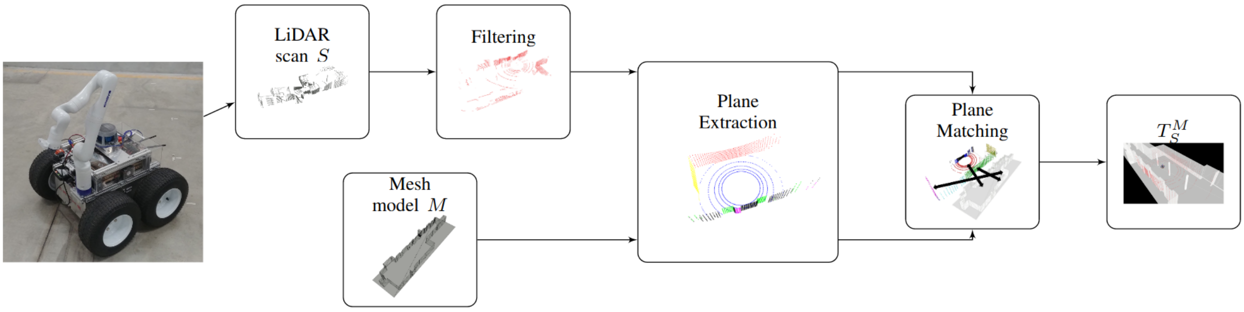


Figure 1. Software system overview of the global localization system: First the raw LiDAR scan input and mesh model are preprocessed in a filtering step and a mesh summarization respectively. On both data, planes are extracted using RHT, and consequently matched. Using the correct assignments, the system outputs the transformation between scan and model for the real robot [26].

RHT-based plane segmentation [22]. Our final choice is RHT, we therefore only briefly describe the other options.

In RANSAC-based plane extraction, hypotheses about the planes contained in the scan are made by considering only a small subset of the points and calculating the plane parameters from this. These are then checked for consistency with the remaining points in the scan. Several iterations are carried out and finally the solution with the most inliers returned. The run-time depends on the size of the considered subset that is used to derive the hypothesis, the inlier ratio and the number of LiDAR points [29].

Region growing is stated to show low sensitivity to noise [28], and is therefore an attractive option for the considered problem. Seed regions are determined, which are then checked for local coplanarity, e.g., by using RANSAC or areas of low local curvature evaluation. Then, points in the neighbourhood of the considered subset of LiDAR data points are checked for their association to the corresponding plane and added to the current subset as further plane inlier points. Efficient implementations facilitate a kd-tree for nearest neighbor search [28]. Planes can be further regularized comparing the calculated parameters of the detected planes to correct the found parameters towards parallelism, orthogonality or coplanarity.

RHT is a subtype of the 3D Hough Transformation and is characterized by single data points voting for single geometric primitives (in our case planes) instead of voting for all. Amongst different Hough Transformation variants, RHT is a good trade off for high efficiency [22].

3.2.1 RHT Voting Phase

RHT randomly samples three points p_i $i \in \{1,2,3\}$ from the point cloud and checks for their proximity. This proximity metric prevents detection of wrong planes lying crosswise in space. The plane defined by these three points is then found as

$$n_{plane} = \frac{(p_2 - p_1) \times (p_3 - p_1)}{|(p_2 - p_1) \times (p_3 - p_1)|} \tag{1}$$

The norm in the denominator is used to filter out sets of three points arranged on a line. Next, the plane parameters in 3D Hough space $\rho \in \mathbb{R}_{\geq 0}$, pitch $\psi \in [-\pi, \pi)$ and yaw $\theta \in [0, 2\pi)$ are determined. If the scalar product $p_1 \cdot n_{plane}$ is negative, the plane normal is flipped before calculation of θ and ψ. Votes for plane parameters are collected from the samples into an accumulator. The accumulator describes an arrangement of discrete bins, each of which represents a point in the 3D Hough space. After the plane parameters of the given sample p_1, p_2, p_3 have been determined, the bin representing the closest value in 3D Hough space is selected and its vote is incremented. Then, the next voting cycle starts with sampling three points. Since many points are discarded due to the requirements of the sampled points, good plane extraction results require a high number of voting cycles.

3.2.2 RHT Evaluation Phase

After several voting cycles, the bins with highest voting scores are selected from the accumulator. We assume a uniform distribution of reference points, therefore parameters associated with these bins correspond to the most dominant planes. We iteratively perform RHT, removing already detected planes, enabling detection of less dominant planes in subsequent iterations. Due to noisy LiDAR scans it may happen that reference points from the same plane can be assigned to different bins with similar parameters. Therefore, the same plane may be detected several times. To prevent this, a Non-Maxima Suppression is implemented, which is applied to the bins of the maximum values and their neighbors. The neighborhood of a bin A is defined as all bins B with $\max\{|\rho_A - \rho_B|, |\theta_A - \theta_B|, |\psi_A - \psi_B|\} < k$. Votes in the neighborhood are credited to the bin under consideration.

3.2.3 RHT Accumulator Design

Choice of the accumulator has a large effect on the plane detection performance due to the different quantizations of the 3D Hough space. This choice is largely influenced by the LiDAR scan characteristics and the specific point sampling pattern. Here, we consider array and ball accumulator [22]. While array accumulators are easy to implement, they come with the downside of largely differently sized bins. Especially small bins at the poles lead to fragmented plane segmentation in θ, however, non-maximum suppression can reduce this effect. The ball accumulator on the other side counteracts this effect by applying a binning that couples θ and ψ resolution, according to:

$$\theta'_{\psi_i} = \frac{max_{\psi} U(\psi)}{U(\psi_i)} \cdot \frac{1}{N_\theta} = \frac{1}{N_\theta \cdot cos(\psi_i)} \tag{2}$$

where $U(\psi)$ is the accumulator circumference for a specific ψ and N_θ the number of bins of a comparable array accumulator in the θ-direction.

3.3 Plane Matching

After plane extraction, the extracted two sets of planes are matched. The algorithm developed in this paper is inspired by PRRUS [6] which aims to identify the geometrically most consistent assignment between two sets of planes. Our approach differs from the original PRRUS, i.e. restricting the plane assignments by a set of rules.

3.3.1 Assignments Generation

To generate the initial set of candidates, groups of planes are gathered from the model and the scan. These groups consist of three planes with approximately orthogonal normals. Then all possible combinations between the groups of the scan and those of the model are formed. Since the direction of the normals can be ambiguous, we also combine all (6) possible permutations of the groups. If no assignments of two groups of orthogonal planes are found, the algorithm fails to find the exact pose and terminates with a corresponding message.

3.3.2 Calculation and Consistency of Rotations

The set of candidates obtained from the previous step is further reduced by consistency checks with regard to rotational alignment. The rotation is determined by two scan plane normals and the corresponding model plane normals. This problem is known as Wahba's problem. Solving this problem generally requires matrix inversion, but since we only consider two vector pairs the problem can be solved efficiently [30]. We introduce the convention that plane normals in the plane extraction and the map planes point outwards with respect to the robot pose. Therefore, we check for consistency of the rotation determined above with the third plane normal of the group by rotating the third normal from the scan group $n_{scan,3}$. If $n_{scan,3} \cdot n_{map,3} > \delta$, the rotation aligns $n_{scan,3}$ well with its counterpart $n_{map,3}$. Otherwise, the assignment is rejected.

3.3.3 Calculation of the Translation

Once a valid assignment and rotation between three orthogonal planes of scan and map has been found, the translation can be determined. The rotated scan planes are approximately parallel to those in the map. Thus, we find the translation as the mean distance of the plane centroids $c_{\text{map}}, c_{\text{scan}}$ as $(c_{\text{map}} - c_{\text{scan}}) \cdot n_{\text{map}}$.

3.3.4 Cost function

After determining the transformation between scan and model, we calculate a cost of the assignment. For this purpose, the translation error $(c_{\text{map}} - c_{\text{scan}}) \cdot n_{\text{map}}$ between all scan and map planes is considered. We match planes between scan and map by the minimum error. If no corresponding model plane is found for a scan plane, it's translation error is set to a fixed penalty. Finally, all errors of the individual scan planes are summarized into the overall assignment cost C_A (Fig. 2 and Fig. 3).

$$\begin{aligned} &\text{Initialize } C_A = 0, \delta = 0.8, \text{penalty} = 20 \text{ m} \\ &\text{for } i \in 1, ..., N_{scan} : C_A \mathrel{+}= \\ &\begin{cases} \min_{\forall j \in I_i} \mid (c_{scan,i} - c_{map,j}) \cdot n_{map,j} \mid, & \text{if } \mid I_i \mid > 0 \\ \text{penalty}, & \text{else} \end{cases} \end{aligned} \tag{3}$$

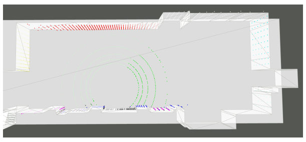


Figure 2. Correct C_A: 0.3 m

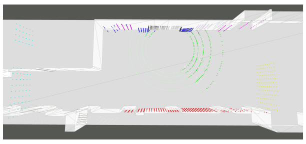


Figure 3. Incorrect C_A: 14.5 m

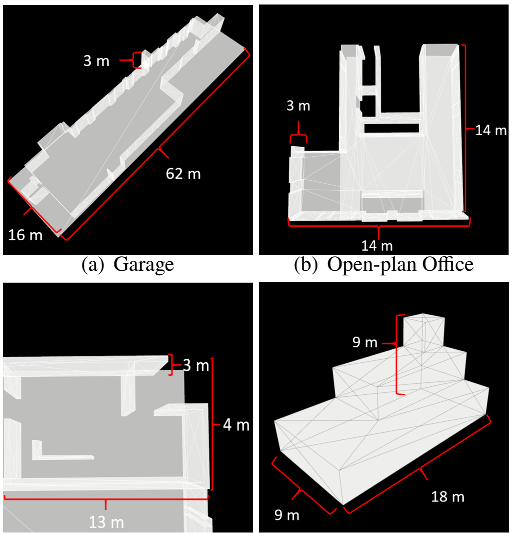
The values of δ and penalty were chosen based on a series of experiments. I_i is the set of all map planes j which have a similar plane normal to scan plane i (i.e. as in 3.3.2) and on which the projection of $c_{scan,i}$ is within the known boundaries, except for i in the group of 3 reference planes that were used to determine the transformation and which therefore have a well-defined corresponding map plane. Finally, the algorithm outputs the transformation with the lowest cost. As the alignment from the plane matching phase is calculated based on groups of 3 planes and does not optimize over all the data available, the transformation yielded from our PRRUS variant can be further refined by ICP alignment after the global localisation.

4 Evaluation

We perform multiple experiments to validate the method for single-shot global localization. An ablation study of the presented algorithm on simulated data in several different environments validates our design choices. Finally, we quantitatively and qualitatively evaluate the system in real-world experiments on a mobile robotic platform.

4.1 Experimental data

For quantitative evaluation, we generate simulated LiDAR data of various configuration. The model data is taken from real-world 3D building mesh models, i.e., a large parking garage of ~ 62 × 16 × 3 m (Fig. 4(a)), an open-plan office space of ~ 14 × 14 × 3 m (Fig. 4(b)), a utility building of ~ 13 × 4 × 3 m (Fig. 4(c)) and a synthetic object called *Construct* which consists of three stacked cuboids of ~ 18 × 9 × 9 m (Fig. 4(d)). The simulated LiDAR data is generated by sampling poses inside these models and then ray casting beams from this pose in different patterns analogous to real multi-beam sensors, e.g., a 16 beam LiDAR with an opening angle of 30 degrees. Since the LiDAR used in the real measurements had an inaccuracy in the range-bearing of ±2cm, we add Gaussian noise with a standard deviation of 0.01 in the same direction. Furthermore, we gathered data with a real robotic platform [26] equipped with a Robosense RS-16 LiDAR sensor. The robot is deployed in the parking garage and the utility building. The ground-truth position of the robot is tracked with a Multistation providing mm-accurate measurements. Note that it is not possible to resolve the ground-truth orientation of the robot with this sensor. In contrast to the simulated environments, the real data also contains both small and large clutter objects and dynamic obstacles that are not mapped in the models.



(a) Garage (b) Open-plan Office

(c) Utility building (d) Construct

Figure 4. Overview of the building models.

4.2 Plane Extraction

Firstly, we perform an ablation study on the plane extraction method. Using a semi-automatic method, we create a data set of 30 simulated 16 beam LiDAR scans taken from different positions in the garage map and a matching ground truth regarding the contained planes. In addition, three pillars and a mesh of an excavator are added in another set of experiments to assess the effects of clutter on the different methods. We compare array and ball accumulator for the RHT. Alternatively, we test a RANSAC-based approach and a region growing algorithm [28]. Quantitative results are reported in Table 1 and examples in Fig. 5. The fractional numbers correspond to the average of the correctly detected planes divided by the planes contained in the individual LiDAR scans. The results suggest that PCL RANSAC is the fastest algorithm. However, the measurements indicate that it reacts more sensitively to clutter and Fig. 5 also reveals bad performance on the real data. CGAL Region Growing produces better results in real data, but the method also finds fewer planes in case that clutter artifacts were not removed. Our experiments on the full localisation pipeline will show that RHT yield the best results on real data due to lower sensitivity to noise and clutter. Similar observations were reported in [22].

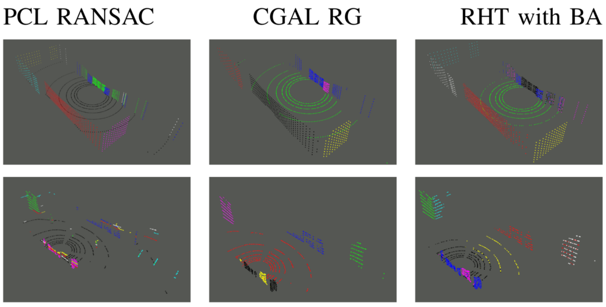


Figure 5. Comparison of different plane extraction methods on simulated data (top row) and real LiDAR scans (bottom row).

method	clutter-free	cluttered	time [ms]
RANSAC	0.87	0.76	3.0625
RG	0.84	0.76	96.9688
RHT Ball Acc.	0.91	0.87	270.469
RHT Array Acc.	0.90	0.86	353.938

Table 1. Quantitative comparison of plane extraction methods.

4.3 Full Localisation

We compare the different localisation methods based on accuracy A and true positive rate TPR of the yielded poses. We count as true positives all poses within 2 meters translation and 20 degrees rotation of the ground truth, if orientation is available. Higher errors are counted as false negatives and those scans where the algorithm returned that it cannot find a corresponding transformation are counted as false positives.

$$TPR = \frac{TP}{TP + FN} \qquad A = \frac{TP + TN(=0)}{TP + TN(=0) + FP + FN} \tag{4}$$

method	1000 beam rand Garage	16 beam rand Office	 Garage	16 beam gr Office	 Utility	 Garage	16 beam gr cluttered noise-free	 with noise
GoICP	4 / 4	7 / 7	2 / 2	14 / 14	0 / 0	4 / 4	- / -	- / -
3D SHOT	0 / 0	0 / 0	0 / 0	0 / 0	1 / 1	0 / 0	- / -	- / -
PRRUS RANSAC	90 / 89	67 / 52	76 / **54**	93 / 92	93 / 78	86 / 83	63 / 54	59 / 48
PRRUS RG	**96 / 96**	55 / 31	56 / 26	90 / 82	87 / 82	91 / 86	79 / 59	76 / 56
PRRUS RHT array	89 / 61	**68 / 53**	**78** / 50	**97 / 95**	**88 / 84**	**97** / 95	**85** / 78	**86 / 81**
PRRUS RHT ball	88 / 64	61 / 48	81 / 49	94 / 94	85 / 82	96 / **96**	**85 / 79**	85 / **81**

Table 2. Localisation results (TPR / A) with different simulated LiDARs. LiDARs are simulated with different beam numbers and in either uniformly random (rand) sampled poses or poses simulating a ground robot (gr).

4.3.1 Simulated Data

We simulate LiDAR scans in a range of different configurations to evaluate the dependency of methods with respect to the density of the scans and the configuration with respect to the environment. The simulated scans do not contain random clutter or occlusions, but are generated directly from the building meshes. We test two LiDAR settings, a 16 beam LiDAR with 30° opening angle replicating the sensor used on the real robot and a dense 1000 beam LiDAR with even distribution of the beams over a full 180 degree opening angle. We test the different variants for 300 samples each. Our results are reported in table 2. We find a clear dependence between different plane extraction methods and LiDAR configuration. For GoICP and the 3D descriptor, we in general found very poor performance over all settings. Comparing the different LiDAR variants, we found that a randomly posed 16 beam LiDAR often produces ambiguous measurements that fit well to many different locations, impacting the performance of all methods. This is also illustrated in Fig. 7. Between the plane extraction variants, we find that while RHT provides less accurate plane parameters due to bin discretization, the localisation results of sparse 16 beam LiDAR scans, which is our target sensor, are better with this method. We also can observe an increased resistance of the RHT pipeline with respect to noise and clutter, in the form of a excavator and room pillars.

4.3.2 Registration of sampled point-clouds

In order to investigate the poor performance of GoICP above, we construct a less realistic experiment that may be better suited for classical 3D registration, to confirm that the poor performance is related to the characteristics of our robotic application. We run tests on the *Construct* object shown in Fig. 4(d) and do not simulate LiDAR scans but evaluate the registration of a uniformly sampled point-cloud against the mesh. We test the methods over randomly sampled transformations as before (Fig. 6). GoICP convinces in this test series by a significantly lower registration error and is able to find the desired transformation every time. Our PRRUS-based methods is less accurate, but can also coarsely localise nearly all poses. 3D descriptor based localisation also fails for these more dense point-clouds, indicating that building models do not contain rich enough information for such descriptors. Fig. 6 shows the accuracy gained through the ICP refinement step, as reported above for PRRUS with RHT. Yet, we observe a large gap in the registration error between GoICP and our ICP refinement. Since there is no algorithmical difference, but different implementations, we conclude that the PRRUS registration errors could be further improved by fine-tuning of the used ICP implementation.

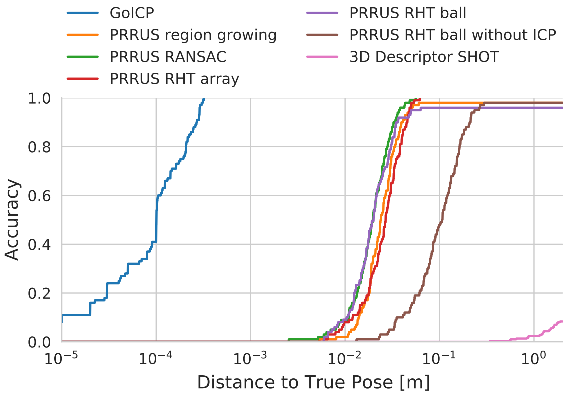
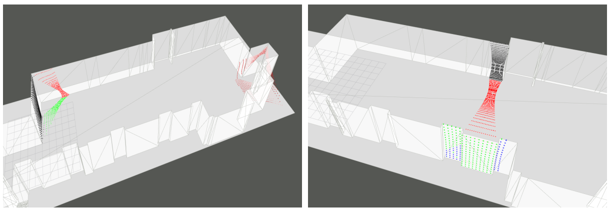


Figure 6. Accuracies of the different methods on the Map *Construct*.



(a) PRRUS finds a wrong transformation

(b) PRRUS is not able to make an estimation

Figure 7. Examples of failure of PRRUS

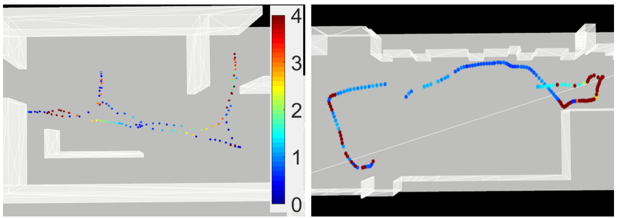
4.4 Robotic Experiments in Cluttered Environments

Finally, we deploy a wheeled robotic platform in the Utility and Garage buildings and take scans at various locations. The correctness of the obtained transformations depends on the position of the LiDAR system as can be seen in Fig.8. PRRUS offers a fast algorithm to perform the plane matching, but it is highly dependent on the plane extraction and the number of planes contained in the scan

method	Utility	Garage	runtime
GoICP	0 / 0	0 / 0	12188
3D SHOT	1 / 2	2 / 2	405
PRRUS RANSAC	10 / 2	25 / 7	66
PRRUS RG	50 / 48	68 / **68**	180
PRRUS RHT array	**90** / 57	**79** / 46	476
PRRUS RHT ball	77 / **77**	75 / 57	387

Table 3. Localisation results (TPR / A) for 16 beam LiDAR scans from the robotic platform, captured in cluttered environments. The runtime was averaged over all scans from the Garage and is given in ms.

and map. In general, PRRUS fails if not enough planes could be detected to determine a unique pose. The PRRUS version with the RHT plane extraction yields the best results. The version with the array accumulator on both maps delivers higher values in the TPR and the ball accumulator in the A, which stems from the RHT version with array accumulator fragmenting the ground plane such that PRRUS cannot find an orthogonal set and does return an "unable to detect" message. The PRRUS version with the RANSAC plane extraction delivers lower values in A and TPR for the real data. GoICP not only required significantly more time, but also resulted in wrong localizations. Similarly, the localization performance of the 3D Descriptor approach is low.



(a) Utility building (b) Garage

Figure 8. Spatial distribution of localisation errors for our RHT ball based method [m].

5 Conclusion

This work proposes a new algorithm for global localisation of robots in mesh based maps of indoor environments. In our evaluation, established point-cloud based methods failed due to the sparse LiDAR scans and clutter even after filtering. Our proposed method based on plane extraction and PRRUS on the other hand achieved high TPR in localisation. The different plane extraction algorithms showed varying advantages and disadvantages, in either run time, precision or clutter-sensitivity. Overall, we found RHT a suitable method to detect static and coarse structures from scans, although other methods were more precise in clutter-free scenarios. Building mesh model based localisation has a huge potential for indoor robotics due to the large availability of floorplans, small memory footprint of the map, and because it is an intuitive representation for operators. With the proposed algorithms, we achieved good performance in simulation and real-world experiments, with and without clutter, and in multiple building meshes.

Acknowledgement

This work was partially supported by the HILTI group.

References

[1] S. Lowry, N. Sünderhauf, P. Newman, J. Leonard, D. Cox, P. Corke, and M. Milford. Visual place recognition: A survey. *IEEE Transactions on Robotics*, 2015. doi:10.1109/TRO.2015.2496823.

[2] J. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler. Semantic visual localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[3] J. Yang, H. Li, D. Campbell, and Y. Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. doi:10.1109/TPAMI.2015.2513405.

[4] Q. Zhou, J. Park, and V. Koltun. Fast global registration. In *ECCV*, 2016.

[5] A. Gawel, R. Dubé, H. Surmann, J. Nieto, R. Siegwart, and C. Cadena. 3d registration of aerial and ground robots for disaster response: An evaluation of features, descriptors, and transformation estimation. In *IEEE International Symposium on Safety, Security and Rescue Robotics*, 2017.

[6] K. Pathak, N. Vaskevicius, J. Poppinga, M. Pfingsthorn, S. Schwertfeger, and A. Birk. Fast 3d mapping by matching planes extracted from range sensor point-clouds. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[7] A. Kaiser, J. Ybanez Zepeda, and T. Boubekeur. Plane pair matching for efficient 3d view registration. *ArXiv*, 2001.07058, 2020.

[8] W. Grant, R. Voorhies, and L. Itti. Finding planes in lidar point clouds for real-time registration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[9] J. Sanchez, F. Denis, P. Checchin, F. Dupont, and L. Trassoudaine. Global registration of 3d lidar point clouds based on scene features: Application to structured environments. *Remote Sensing*, 2017. doi:10.3390/rs9101014.

[10] E. Fernández-Moral, W. Mayol-Cuevas, V. Arevalo, and J. González-Jiménez. Fast place recognition with plane-based maps. In *IEEE International Conference on Robotics and Automation*, 2013.

[11] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat. Comparing icp variants on real-world data sets. *Autonomous Robots*, 2013. doi:10.1007/s10514-013-9327-2.

[12] A. Fitzgibbon. Robust registration of 2d and 3d point sets. *Image and Vision Computing*, 2002. doi:10.1016/j.imavis.2003.09.004.

[13] M. Lourakis. A brief description of the levenberg-marquardt algorithm implemented by levmar. On-line: `http://users.ics.forth.gr/~lourakis/levmar/levmar.pdf`, Accessed: 25/05/2021.

[14] K. Boehnke and M. Otesteanu. Progressive mesh based iterative closest points for robotic bin picking. In *Proceedings of the International Conference on Informatics in Control, Automation and Robotics*, 2008.

[15] C. Peng, Y. Wang, and C. Chen. Lidar based scan matching for indoor localization. In *IEEE/SICE International Symposium on System Integration*, 2017.

[16] S. Ratz, M. Dymczyk, R. Siegwart, and R. Dubé. Oneshot global localization: Instant lidar-visual pose estimation. In *IEEE International Conference on Robotics and Automation*, 2020.

[17] A. Avetisyan, M. Dahnert, A. Dai, M. Savva, A. Chang, and M. NieBnez. Scan2cad: Learning cad model alignment in rgb-d scans. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[18] Q. Feng, Y. Meng, M. Shan, and N. Atanasov. Localization and mapping using instance-specific mesh models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.

[19] T. He and S. Hirose. A global localization approach based on line-segment relation matching technique. *Robotics and Autonomous Systems*, 60:95–112, 2012. doi:https://doi.org/10.1016/j.robot.2011.09.003.

[20] B. Micusik and H. Wildenauer. Descriptor free visual indoor localization with line segments. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[21] S. Wang, S. Fidler, and R. Urtasun. Lost shopping! monocular localization in large indoor spaces. In *IEEE International Conference on Computer Vision*, 2015.

[22] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nuchter. The 3d hough transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, 2011. doi:10.1007/3DRes.02%282011%293.

[23] A. Budroni and J. Boehm. Automated 3d reconstruction of interiors from point clouds. *International Journal of Architectural Computing*, 2010. doi:10.1260/1478-0771.8.1.55.

[24] A. Adan and D. Huber. 3d reconstruction of interior wall surfaces under occlusion and clutter. In *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, 2011.

[25] R.-C Dumitru, D. Borrmann, and A. Nuchter. Interior reconstruction using the 3d hough transform. *Proceedings international archives of the photogrammetry, remote sensing and spatial information sciences*, 2013. doi:10.5194/isprsarchives-XL-5-W1-65-2013.

[26] A. Gawel, H. Blum, J. Pankert, K. Krämer, L. Bartolomei, S. Ercan, F. Farshidian, M. Chli, F. Gramazio, R. Siegwart, M. Hutter, and T. Sandy. A fully-integrated sensing and control system for high-accuracy mobile robotic building construction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.

[27] J. Huang. Lab 31: Planar segmentation. On-line: `https://github.com/cse481sp17/cse481c/wiki/Lab-31:-Planar-segmentation`, Accessed: 22/07/2021.

[28] J. Deschaud and F. Goulette. A fast and accurate plane detection algorithm for large noisy point clouds using filtered normals and voxel growing. *3D Data Processing, Visualization and Transmission*, 2010.

[29] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J. Frahm. Usac: A universal framework for random sample consensus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013. doi:10.1109/TPAMI.2012.257.

[30] J. Hinks and M. Psiaki. Solution strategies for an extension of wahba's problem to spinning spacecraft. *Journal of Guidance, Control, and Dynamics*, 2008. doi:10.2514/6.2008-6459.