Interactive Floor Mapping with Depth Sensors

Yang Cai(✉) and Uma Arunachalam

Carnegie Mellon University, Pittsburgh, USA
ycai@cmu.edu

Abstract. Simultaneous Localization and Mapping (SLAM) algorithms have been growing popular in indoor navigation and mapping. However, it often fails in many real-world environments, such as low-lighting, fast motion, featureless walls, and large buildings. There are also usability issues with the 3D point clouds for actual indoor localization and mapping for humans and autonomous robots. In this study, we use depth sensor to generate 3D point cloud and then register that to the 2D building floor plan or footprint. We extract the ground plane from the point cloud and create a 2D point cloud and contours to be registered to the map. The experiments show that 2D map is more intuitive than 3D point cloud. Furthermore, the contour map reduces computational time in orders of magnitude. We also developed a graphical user interface to enable the user to register the 2D point cloud interactively. It is a new way to use SLAM data. Our case studies in large office buildings demonstrate that this approach is simple, intuitive, and effective to enhance the localization and mapping in the real-world.

Keywords: Indoor navigation · Simultaneous Localization and Mapping · SLAM · Localization · Mapping · Features · RANSAC · Floor map · Floor plan · Point cloud · 3D · Registration · LiDAR · Depth sensor · Depth camera · Edge detection

1 Introduction

Indoor navigation is a challenge when the GPS signals and infrastructural positioning sensors are not available. Visual Simultaneous Localization and Mapping (Visual-SLAM) algorithms have attracted broad attention because it relies on an affordable camera without any infrastructural positioning sensors. Visual-SLAM is designed to localize the user's moving trajectory and orientation, while simultaneously mapping the environment with a 3D point cloud. However, there is a significant gap between the SLAM results and the practical applications. Visual-SLAM works well in "normal" environments, e.g., a well-lighted space and full of "features" in the scene. It often fails in many real-world environments, such as low-lighting, fast motion, and "featureless" walls. There are also usability issues about how to use the 3D point clouds for indoor navigation.

In this study, we explore depth cameras to generate a point cloud and register it to a floor plan or a footprint of the building from open sources or satellite images. The point cloud down-sampling, floor plane detection, and projection, as well as edge detection algorithms, are implemented to generate a floor map for mapping and localization.

T. Z. Ahram and C. S. Falcão (Eds.): AHFE 2021, LNNS 275, pp. 19–25, 2021.
https://doi.org/10.1007/978-3-030-80091-8_3

An interactive graphical user interface is developed to enable the user to register the scanned floor map to the floor plan or building footprint. We tested the method in two extremely realistic environments: a large office building with a long route including multiple turns, low feature walls, low-lighting, and a loop, and a large hospital building. Compared to the conventional RGB-camera-based Visual-SLAM, this method shows better mapping and localization accuracy, flexibility, and potential applications for first responders, building inspection, and subterranean mapping.

2 Review of SLAM Algorithms and Depth Sensors

Early probabilistic approaches were introduced in the paper by Lu and Milios [1], followed by Gutmann and Konolige [2], in which local frames, as well as the spatial relationship between local frames, was derived by modeling it as a random variable and matching pairwise scans for odometry. Further approaches [3] formulate SLAM as a maximum a posteriori estimation problem with factor graphs to represent interdependence among variables. Extended Kalman Filters (EKF), followed by FastSLAM [4] are also methods that have been increasingly used for this problem, and these are feature-based methods that use maximum likelihood algorithms for exploiting interdependency between frames. These methods also involved the representation of the model/environment as an occupancy grid map, that has been extended to 2D and 3D grid-based structures representing the probability of a cell being occupied.

Prominent visual SLAM methods include Large Scale Direct SLAM (LSD-SLAM) [5] and Oriented FAST and Rotated BRIEF SLAM (ORB SLAM) [6]. LSD-SLAM, in contrast to ORB-SLAM, is a featureless method that uses monocular RGB data to build large-scale consistent maps based on image alignment, and pose-graph of keyframes with associated semi-dense depth maps. Tracking of new camera images is based on mapping of pose between current and previous frame initialization. A depth map is further estimated to refine or replace the current keyframe. It is followed by optimization for loop closures and scale drifts. ORB-SLAM is a feature-based method for the generation of camera trajectory and sparse reconstruction of a 3D point cloud. It involves automatic robust initialization of Scale Invariant Feature Transform (SIFT) features from planar and non-planar scenes using homograph transformation and fundamental matrices. Thus, they require the cameras to be calibrated. Overall, similar to LSD-SLAM, it also involves tracking, mapping, and optimization.

Light Detection and Ranging (LiDAR) based SLAM methods involve the use of depth information to develop scan mapping and matching methods [7]. A commonly used method is Normal Distributions Transform (NDT) matching [8] that involves scan matching based on observed features such as points and lines. Originally developed for 2D applications, NDT models laser scan data as a collection of local normals, and divides the space around a SLAM agent into grids. It then estimates the mean and standard deviation between points in that particular grid cell for modeling as a normal distribution. Scan alignments are then performed for different scans with reference to the coordinate system based on which the transform was built. Loop detection and closure based on the bag-of-words model was proposed in [8], in which range data in 3D was used to robustly detect previously seen places and estimate relative orientation.

Another popular method recently developed was Visual-LiDAR Odometry and Mapping (V-LOAM) [9], in this work, Zhang and Singh propose a method for fusion of odometry and LiDAR data as a principal step for mapping. The LiDAR odometry framework matches point clouds at lower frequencies to refine motion estimates while accounting for distortion and scaling parameters. Another extension of this line of work is visual-inertial SLAM (VI-SLAM) methods [10] that fuse IMU data with estimated pose based on conventional SLAM methods. This helps improve the accuracy and reliability of estimation.

Shin, Sik, and Kim [11] present a thermal infrared-based SLAM system that is aided by range measurements from LiDAR scan data. This method is relatively robust against fog, smoke, and dim-lit conditions in comparison to RGB-based methods that exploit well-lit environments for mapping. Their proposed method focuses on the reliable estimation of 6-DOF camera pose for tracking and loop detection based on raw 14-bit features obtained from the thermal sensor by using a thermographic error model for optimization.

The most commonly used commercially available LiDAR is Velodyne, VLP32, etc. These have ROS-packages that can be used for easy interfacing. It generates a 3D visualization of the surroundings by shooting pulses of the wavelength of 905 nm at a 600 kHz frequency. It has a range of about 200m and has a full 360̊ field of view [12]. The primary sensor used in this work is the Intel RealSense™ D345i sensor that is equipped with an RGB camera, depth camera, accelerometer, and gyroscope. Depth-based SLAM [14] methods involve the use of this depth data for the triangulation of 3D points in the frame of the camera. These methods then exploit matching features and aligning scans relative to the camera pose of the previous frame for tracking and mapping.

3 Depth-Based SLAM Algorithm

The main objective of this task is to create a 3D point cloud of the surrounding environment from 3D depth scans. We estimate the trajectory using point cloud registration with NDT and reduce drift using pose-graph optimization when the agent revisits a place using a trust region solver [15].

The algorithm takes an $N \times 3$ positional data as input per scan. This corresponds to the depth data back-projected as points based on triangulation and camera intrinsic by RealSense™. Parameters such as the maximum range of 3D laser scan data, reference vectors that act as normal to ground plane, minimum Euclidean and angular distances for plane fitting, down sampling ratio for point clouds, voxel grid size corresponding to NDT registration, the minimum distance between poses for a scan to be considered valid, loop closure radius, submaps scan count, submap threshold, RMSE error threshold are initialized. These parameters required a considerable amount of tuning depending on how the data is structured.

After initialization, the point cloud is filtered to remove outliers, points on the ground, and ceiling plane. The point cloud is also down sampled to improve the speed and accuracy of point cloud registration. Point cloud registration then estimates the relative pose (translation and orientation) between the previous and current scan. A pose graph 3D object is used to store the estimated poses. The next step is to perform loop closure, the agent searches within a small radius, it is sufficient to do so since drift in LiDAR data is low and it is time-consuming to exhaustively search all scans. Submaps are created from consecutive scans. These submaps are matched with the current scan and are scored to determine whether or not they are probable loop candidates. The relative pose is estimated and accepted if the root means square (RMSE) is within the threshold set in initialization. After a sufficient number of loop edges are accepted, pose graph optimization is performed to reduce drift in the estimated trajectory. The point clouds are then combined based on the estimated poses and visualized.

4 Registration of the 3D Point Cloud to a Floor Map

4.1 Ground Plane Extraction

Given 3D point cloud data of the generated map, a reduced 2D floor map is generated. The point cloud is formatted where x, y, and z correspond to each 3D point. In no particular order, all points in the generated point cloud are listed. Using a simple python interface, the generated 3D map is stored as an array subsampled depending on the density of the data.

The ground plane is estimated from this data using RANSAC. For a plane parameterized as, $Ax + By + Cz + D = 0$, where, $[A, B, C]$ represent the normal to the plane, D represents the distance of this plane from the origin, all points along the ground plane satisfy the equation. Hence, RANSAC [13] is applied to estimate the set of points that lie within an error bound owing to noisy data. The algorithm is described as follows:

First, extract the ground plane. Only two thresholds need to be initialized [*zmin, zmax*] that describe the range of points to be considered as inliers. The z range is chosen as $[-0.15, 0.2]$ for this problem.

Second, estimate the set of points that fall within the range after subsampling by fitting the z = 0 plane. Since the data must be planar, project the points by taking the dot product of the points, p, with the normal vector n, as $p \cdot n = pncos(\theta)$, where θ represents the angle between the projected point and normal.

Finally, the resulting 2D plot results in the ground plane, which is saved as an image using *Matplotlib* with an equal axis.

4.2 Floor Map Registration

An interactive floor map registration tool is developed in Python for all platforms. The GUI can perform operations such as scaling, translation, and translation. The

interface can be used to register the floor map with the generated 2D map. The transformation performed by the GUI can be summarized as:

$$P = Hp$$

where P is the set of homogeneous transformed points after application of similarity transform H on homogenous points p. H is given by:

$$\begin{bmatrix} Acos(\theta) & -Asin(\theta) & t_x \\ Asin(\theta) & -Acos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

where A represents scaling, θ represents rotation angle about the z-axis, t_x, t_y represents translation along the x-axis and y-axis respectively.

4.3 Floor Mapping with a Floor Plan or Footprint

Assume we have a 3D point cloud and a building's floor map (floor plan). We can use the interface to generate a 2D point cloud to be overlaid on the floor map. The left image in Fig. 1 shows the 3D point cloud from iPhone 12 Pro has been successfully registered onto the building floor map. However, in many cases, we don't have a building floor map. But we can find the building's footprint from Google Map or other open sources such as OpenStreetMap. The image on the right of Fig. 1 shows an example of the 3D point cloud data from iPhone 12 Pro has been registered to the medical center's footprint on Google Map by matching entrance and the dome-shaped structures.

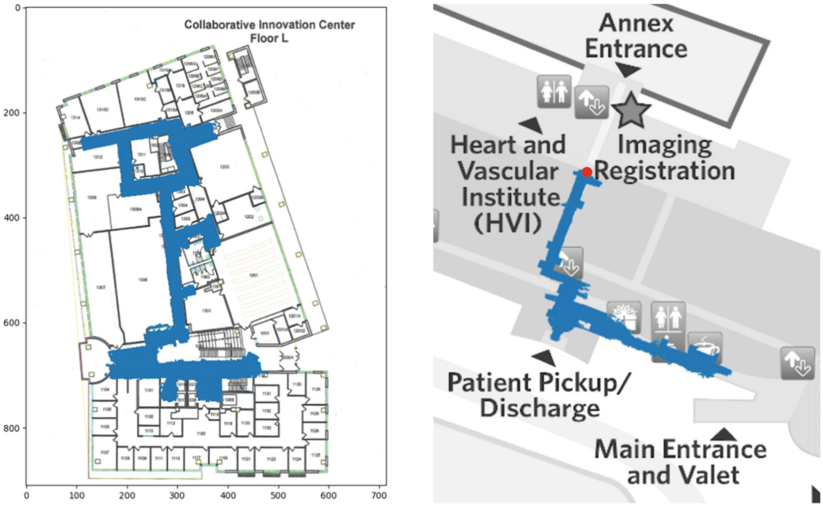

Fig. 1. Long route registration with the office building floor map (left) and the route registration with a hospital footprint (right)

5 Edge Detection

Manipulating a large 3D point cloud is computationally expensive to machines and humans! Our eyes in fact prefer smooth, clean, and straight lines, rather than messy point cloud. We use the popular Canny edge detection algorithm to extract the edges from the point cloud. It performs Gaussian filtering to remove noise, followed by finding image gradients and suppressing spurious detections. The gradient thresholds are set to detect edges. This boundary approximately represents the floor plan of the mapped environment. Figure 2 shows examples of the edge detection results.

Fig. 2. Edge detection from 3D point cloud from iPhone 12 Pro with clean contours

6 Conclusions

In this study, we use depth sensor to generate 3D point cloud and then register that to the 2D building floor plan or footprint. We extract the ground plane from the point cloud and create a 2D point cloud and contours to be registered to the map. The experiments show that 2D map is more intuitive than 3D point cloud; furthermore, the contour map reduces computational time in orders of magnitude. We also developed a graphical user interface to enable the user to register the 2D point cloud interactively. It is a new way to use SLAM data. Our case studies in large office buildings and medical centers demonstrate that this approach is simple, intuitive, and effective to enhance the localization and mapping in the real-world.

Acknowledgment. The author would like to thank the discussions with Scott Ledgewood, Neta Ezer, Nick Molino, Justin Vivirito, Dennis Fortner, and Mel Siegel. The author is grateful for the support from NIST PSCR and PSIAP and Northrop Grumman Corporation.

References

1. Lu, F., Milios, E.: Globally consistent range scan alignment for environment mapping. Auton. Robot. **4**(4), 333–349 (1997)
2. Gutmann, J.S., Konolige, K.: Incremental mapping of large cyclic environments. In: Proceedings of 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA 1999 (Cat. No. 99EX375), pp. 318–325. IEEE (1999)

3. Kschischang, F.R., Frey, B.J., Loeliger, H.A.: Factor graphs and the sum-product algorithm. IEEE Trans. Inf. Theory **47**(2), 498–519 (2001)
4. Barrau, A., Bonnabel, S.: An EKF-SLAM algorithm with consistency properties. arXiv preprint arXiv:1510.06263 (2015)
5. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: large-scale direct monocular SLAM. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8690, pp. 834–849. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10605-2_54
6. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Trans. Rob. **31**(5), 1147–1163 (2015)
7. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, pp. 225–234. IEEE (2007)
8. Biber, P., Straßer, W.: The normal distributions transform a new approach to laser scan matching. In: Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2003 (Cat. No. 03CH37453), vol. 3, pp. 2743–2748. IEEE (2003)
9. Zhang, J., Singh, S.: Visual-lidar odometry and mapping: low-drift, robust, and fast. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 2174–2181. IEEE (2015)
10. Mourikis, A.I., Roumeliotis, S.I.: A multi-state constraint Kalman filter for vision-aided inertial navigation. In: Proceedings 2007 IEEE International Conference on Robotics and Automation, pp. 3565–3572. IEEE (2007)
11. Shin, Y.S., Kim, A.: Sparse depth enhanced direct thermal-infrared SLAM beyond the visible spectrum. IEEE Robot. Autom. Lett. **4**(3), 2918–2925 (2019)
12. https://www.agmsys.ru/en/mscan/ultra_puck
13. Li, L., Yang, F., Zhu, H., Li, D., Li, Y., Tang, L.: An improved RANSAC for 3D point cloud plane segmentation based on normal distribution transformation cells. Remote Sens. **9**(5), 433 (2017)
14. https://intel.github.io/robot_devkit_doc/pages/slam.html
15. https://www.mathworks.com/help/nav/ug/perform-lidar-slam-using-3d-lidar-point-clouds.html