Mobile Robot Visual Localization and 3D Map Generation
(Computer Vision Project Final Report)

Xiang He
Mechanical Engineering
University of Utah
hexiang422@gmail.com

Dejun Guo
Mechanical Engineering
University of Utah
dejunguo422@gmail.com

Jacob Harris
Mechanical Engineering
University of Utah
jacob84401@gmail.com

Roya Sabbagh Novin
Mechanical Engineering
University of Utah
roya.sabbaghnovin@utah.edu

Amir Yazdani
Mechanical Engineering
University of Utah
mojtaba.yazdani@utah.edu

Abstract

In this project, the visual localization is implemented on a mobile robot using Kinect. The method used for localization is RGB-D SLAM. In addition, a 3D map is generated based on all collected images. Results show that this method is accurate and comparable to LIDAR and robot odometry.

1. Problem Statement

Mobile robots are commonly used across research and industry. In this context, robot localization is often one of the major challenges in robot control. In an indoor environment with a flat floor plan, localization is identified as a problem of estimating the pose, i.e. position and orientation of a mobile robot, when the map of the environment, sensor readings, and executed actions of the robot are provided [1].

For purpose of this project, we will implement the Visual Odometry method on a Roomba iCreate 2, equipped with a Microsoft Kinect RGB-D sensor, for localization and map generation. We are going to evaluate our approach by comparing localization results with robot odometry results, LIDAR sensor localization, and GPS.

2. Introduction and Motivation

Many existing methods for robot localization are based on GPS, laser odometry, wheel odometry, sonar sensors or artificial landmarks [2, 3]. All of these methods have their strengths and weaknesses. For example, GPS is prone to signal loss, laser based systems are heavy and expensive, and wheel odometry is susceptible to slip and drift overtime [4].

Alternatively, monocular camera visual odometry can be used and has many positive aspects. The system is relatively inexpensive, extremely light, and cameras utilize our rich colored world. It has been demonstrated that Visual Odometry will oftentimes produce better results than Wheeled Odometry and can have as low as 0.1% tracking error [4]. To further reduce drift, a visual odometry system may be combined with GPS, laser, and IMU data [5, 6, 7]. A visual system can also create a map to be used in the future for navigation systems or further analysis.

In this method, it is assumed that a monocular camera is rigidly fixed on a mobile robot. It is not necessary to know the environment. A video is filmed while the robot moves and point features are matched between pairs of frames and linked into image trajectories at video rate using the Harris Corners Technique. Then, the camera motion can be estimated robustly from the feature tracks using a geometric hypothesize-and-test architecture [8]. The key points and images can be stored as a map to be used by the robot to prevent drift over time [9].

In this project, we will implement a Visual Odometry method on a Roomba iCreate 2 which is equipped with a Microsoft Kinect RGB-D sensor for localization and map generation. Although the mobile robot is moving on a 2D plane (ground), the localization and map generation algorithms are for 3D environments, so the final results are in 3D. At the end, we will evaluate our approach by comparing resulted localization with results from robot odometry, LIDAR sensor localization, or GPS data.

Finally, it should be mentioned that we are going to make a ROS package which includes different nodes for various parts of the algorithm.

3. Prior Art

Many solutions have been proposed for the pose estimation of mobile robots employing Kalman filtering [10], particle filtering [11, 12], and Markov localization [13]. Ganganath and Leung in [14] proposed an accurate and low cost mobile robot localization method using odometry and a Kinect sensor. The odometry model they used is capable of tracking any arbitrary robot motion. They have fused odometry and the Kinect sensor measurements using the extended Kalman filter (EKF) and the particle filter (PF) to provide more accurate localization results.

Vision-based localization and mapping algorithm using SIFT features is proposed in [2,15]. Being scale and orientation invariant, SIFT features are good natural visual landmarks for tracking over long periods of time from different views, to correct odometry locally. This algorithm has also been extended for global localization [16].

Compared to the visual odometry method where the map generated is a by-product, visual SLAM (simultaneous localization and mapping) generates a map to help localization. New features will be added into the map as new areas are explored. Andrew, *et al.* [17] first introduced the monocular visual slam algorithm where three known feature points are used to initialize the system. Shi-tomasi features are used due to their efficiency in calculation for initializing patch features. These are then localized based on particle filtration. The patches are stored as the landmarks that form the map. After that, improvements will be made on the visual SLAM. The PTAM (parallel tracking and mapping) [18] project separates localization and mapping with multi threading, and uses bundle adjustment both locally and globally to ensure the convergence. Although visual SLAM can be seen to solve the drifting problem, once the mismatching happens, extra computational work is needed to stop divergence.

Fiala and Ufkes in [19] proposed a visual odometry system that can estimate the 3D pose of a mobile platform using monocular video data and associated 3D depth data, as provided by Microsoft's Kinect sensor. In their work, stereo matching is thus avoided, and matching is only performed between images from different times. They utilized standard feature detectors, SIFT, but match between 3D points to calculate pose change directly.

4. Robot Setup and camera calibration

For this project, a Roomba iCreate 2 mobile robot controlled by a joystick is used. The vision system is Microsoft Kinect One, which is mounted on top of the robot along with LIDAR. All code for the Roomba system is in Python and a ROS package is developed for communication between different components in the system.

The camera parameters are calibrated using the camera calibration toolbox on Matlab software. The reprojection errors are below one pixel. The intrinsic parameters of the monocular camera include: focal length 528.4 pixel/m and principal point (323.2, 264.7) pixel.

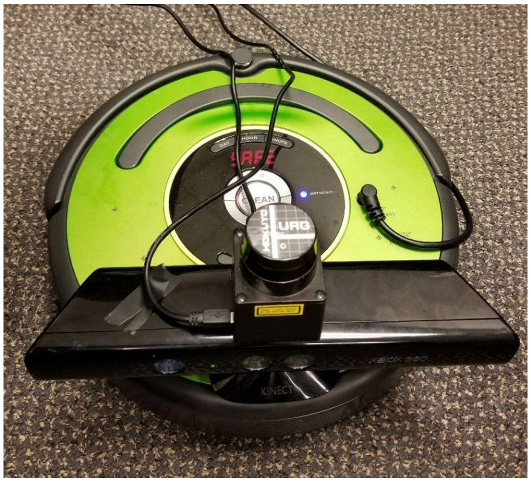

Figure 1. Robot and vision setup

4. Proposed approach

4.1. Monocular Visual Odometry

First, we did a literature review on visual odometry methods. Considering different types of cameras and input information, visual odometry can be separated into monocular visual odometry, binocular odometry, and VO with a RGBD camera. The differences will be discussed below.

The classical monocular VO method will only determine the rotation matrix R and the direction of movement between two matched frames. The distance of the translation is not found. Usually, the monocular method assumes some prior knowledge of the waypoint, either when it starts, or while it moves. The assumption can be either that the initial movement is purely translational without rotation, or started with some pre-known markers. Some methods also assume that all the detected features during the movement are on the ground plane. If no assumption is made, the monocular VO requires other odometry information, e.g. GPS signal or wheel odometry from a ground robot.

Binocular VO, on the other hand, does not require such prior knowledge while running. It requires higher calculation since an extra step is required to compute the depth information of features in two frames from two cameras.

VO with a RGB-D camera can get the depth of feature directly from the camera. The computational complexity for this method should be the lowest. The main disadvantage of using a RGB-D camera is that the IR sensor used to get depth information is only valid for indoor applications due to the high illumination that exists outdoor affecting the distance measurement. We chose the Kinect RGB-D camera to get depth information directly and only worry about the calculation between two continuous frames. Currently we implement the fovis library for Kinect [20]. The method implements the localization with the following steps:

1. Image preprocessing with Gaussian blur and Gaussian pyramid.
2. Feature extraction using the FAST detector.
3. Initial rotation estimation basically from a downsampled frame to roughly estimate rotation. This is used to help matching.
4. Feature matching, features from the FAST detector with a patch of 9x9 pixel.
5. Inlier detection, using a method similar to RANSAC.
6. Motion estimation, based off of a keyframe where small rotation or translation will not affect the keyframe and each new frame is matched to the keyframe.

We also have monocular VO and binocular VO partially implemented. For monocular VO, we developed a package in ROS using what we learned in the class. The steps are:

1. Capturing image from camera.
2. Undistorting the image.
3. Detecting feature using FAST algorithm.
4. Using RANSAC to compute the essential matrix.
5. Estimating the rotation matrix and direction vector.
6. Read in the scalar from the wheel odometry.

The binocular VO is partially done using the similar step but with an extra calculation of depth for features using a calibrated camera. The estimation of translational distance is what we need to do in the next step.

A package that can switch between the RGB-D VO and binocular VO is developed to enable the Kinect to work both indoor and outdoor.

To better illustrate our monocular visual odometry, the following steps are listed to show how Monocular VO works.

1. Read image from the camera
2. Undistort image based on calibration data
3. Find matched features in the current image from previous image
4. Calculate fundamental matrix
5. Get rotation matrix and translation vectors from fundamental matrix
6. Combine the wheel odometry from Roomba for scaling
7. Integrate R and t
8. Find new features when the number of available features drops below certain threshold

4.2. RGB-D SLAM

In previously discussed methods, drift appears as error is made in feature detection, rotational, and traditional matrix integration. Monocular and binocular vision only compares sequential images. This makes the errors cumulative as time continues. Though efforts can be made to reduce the rate of drift, error is eminent.

In contrast the SLAM methodology makes use of past exploration data to reduce drift. This is through a process called loop closure which will be described later. In the methodology described by Labbe and Michaud [21] a RGB-D sensor was used. As the past exploration data is utilized, a graph structure is needed to quickly access and search. The nodes contain visualization information such as RGBD images and SURF features. SURF features are preferred when compared to SIFT as the scale invariance may cause feature matches when an object is at different scales. The relative scale can be used to better determine translational movements. Edges within the graph are the odometry transformation. A graph increases search speeds by reducing the search area. Once a location within the graph is found, the process relative location search is reduced to neighboring nodes for small changes in translation and rotation.

A bayesian filter is used to evaluate a hypothesis over the stored nodes. Once a loop closure hypothesis reaches a predefined threshold, a loop closure is detected. The threshold testing is done using a visual dictionary of SURF features. With two potential matching images and RGB-D data, the 3D location of the visual words is calculated.

Using a very similar approach as in class, these two images are matched using RANSAC. If a computed fundamental matrix is found to have a sufficient amount of inliers, the loop closures are accepted and an edge between the two portion of the graph is added. For larger maps, a search over the entire graph is computationally complex and cannot be done in real time. Insead, a subset of images scattered throughout the graph is selected and searched through.

4.3. Comparison of different localization methods

The main method used as a baseline is the robot's wheeled odometry, which is available on the Roomba platform. An onboard LIDAR sensor is used for online comparison. Although LIDAR is expensive and lacks rich information on the environment, it can provide accurate position and orientation of the robots. In addition, GPS is used to obtain the global position knowledge for outdoor experiments. The GPS signal is not available in indoor and urban applications due to signal blockage and attenuation. Signal disruptions may even happen outdoor occasionally due to interference with the environment.

4.4. Map Generation

Each RGB-D image is a data rich image that can be easily converted to a RGB point cloud. This point cloud is based upon the camera frame. Through processes discussed previously, the camera global position and orientation is known. With this data, the each point can be transformed into the base reference frame. All of the images can be combined together to form a color map of the 3D space. In our case, the point clouds were stored in a database and incrementally added to RVIZ for visualization.

5. Results

The experiment was carried out for both indoor and outdoor environments verifying that the algorithm works properly. In the indoor experiment, we compared the RGB-D SLAM algorithm with LIDAR-based SLAM and monocular visual odometry. The outdoor test was to compare monocular visual odometry, wheel odometry, and GPS.

5.1. Indoor

The indoor test was setup in the MEB building. In order for the LIDAR to work, we built a small and customized environment also with feature rich walls. Figures 2-5 show snapshots of the experiments and final results. As we can see in the figure, the LIDAR SLAM can accurately build the grid map with the explored area shown in white, and the unexplored marked gray. The current LIDAR position is marked with the blue-green-red axis. Unfortunately, the LIDAR odometry is not shown. The odometry from RGB-D vision is plotted in orange arrows. It is clear that the RGB-D visual localization is slightly to the left of the LIDAR SLAM position. This is probably because, when the map was initially built, the visual odometry drifted and the loop closure correction had not taken effect yet.

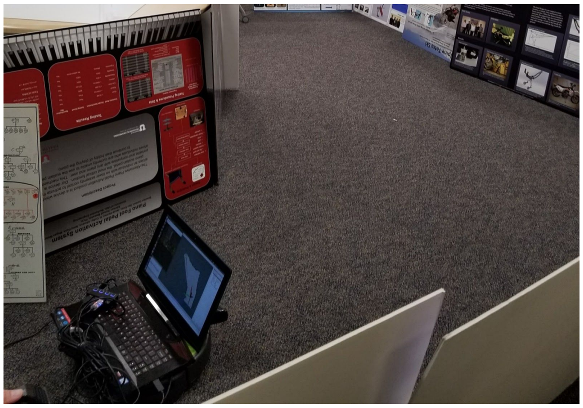

Figure 2. Environment and robot setup for indoor test.

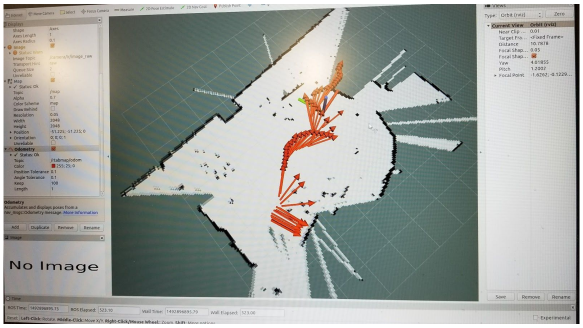

Figure 3. Indoor localization using graph-based SLAM with Kinect, LIDAR, and robot odometry.

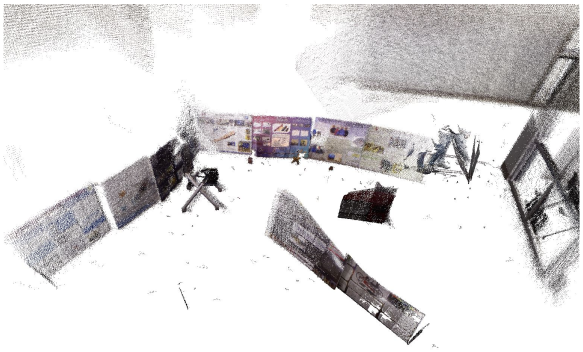

Figure 4. Generated 3D map of the indoor environment we made. Some posters we used as wall, the bumblebee action figure, water bottle, chair, and windows are visible in the generated point cloud.

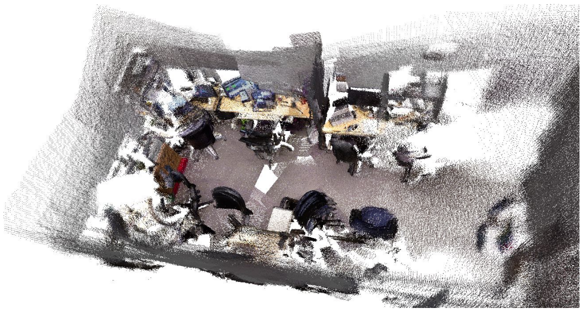

Figure 5. 3D map of our office generated by graph-based SLAM algorithm.

5.2. Outdoor

The algorithm using monocular vision and robot odometry was developed to be used outdoor when the Kinect’s IR sensor is saturated. We did the experiment outdoor (outside of the MEB building) where we thought we had enough features including trees and buildings. However, we achieved poor results. We believe it is due to two reasons. First, the algorithm heavily relies on robot odometry and the Roomba robot is not designed to be used outdoor on cement pavement. We saw a lot of slipping and the robot got stuck several times on rocks and cracks. Hence, we experienced poor odometry and as a result, poor localization. Second, the number of detected features was low which affected our localization accuracy.

We also got data from GPS, LIDAR, and graph-based SLAM for comparison. Results of the outdoor test are shown in Figures 6-10.

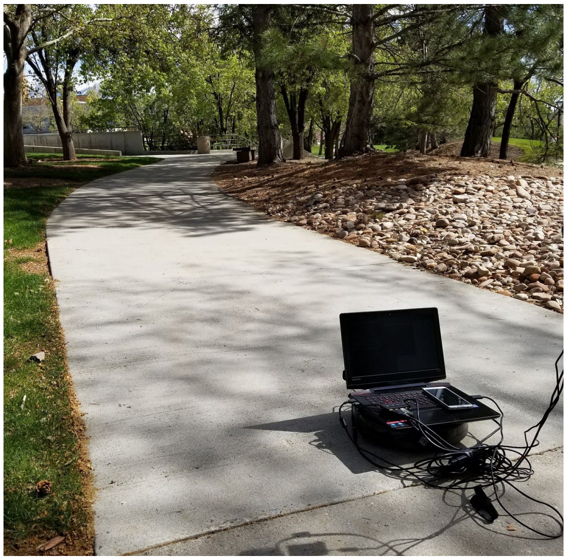

Figure 6. Outdoor localization setup using monocular vision, robot odometry, and GPS.

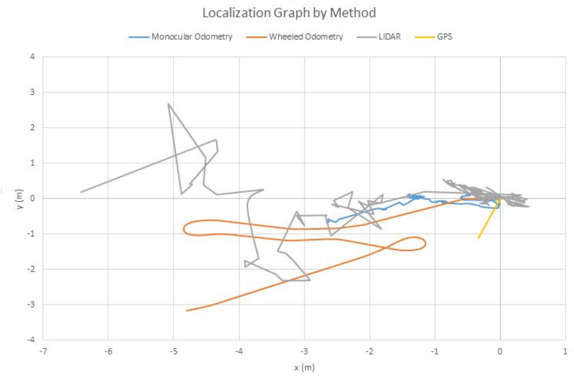

Figure 7. Result of outdoor localization and comparison between different methods Monocular Odometry, Wheeled Odometry, LIDAR, and GPS.

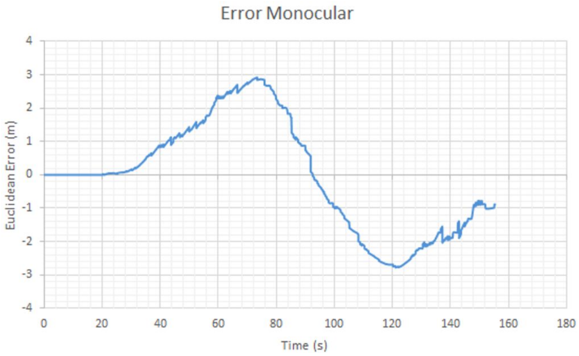

Figure 8. Error for localization using monocular vision.

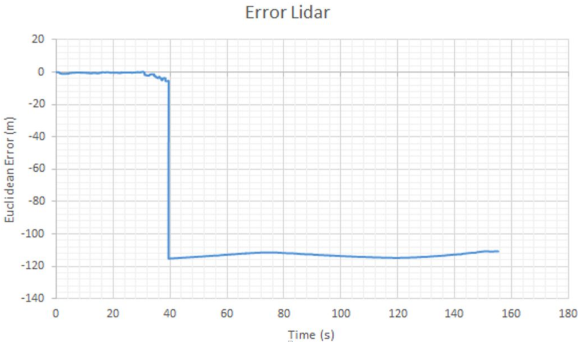

Figure 9. Error for localization using LIDAR.

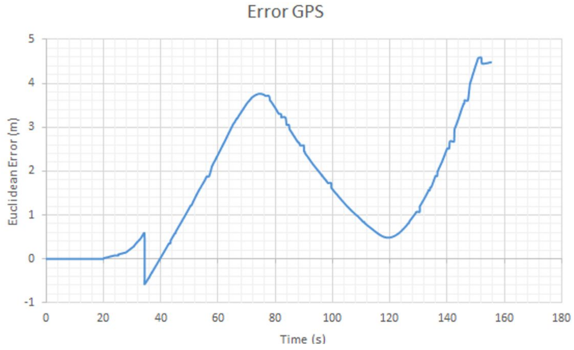

Figure 10. Error for localization using GPS.

6. Discussion and Future Work

In this project, we implemented different methods for visual localization of a mobile robot. As results showed, graph-based SLAM works better than all other approaches, including LIDAR, for indoor use.

As future work, we can try to make the code work faster by customizing it for special applications and reduce the number of ROS topics. We also should try to fix the problem of low number of detected features in monocular vision code for outdoor localization. Finally, we should finish the automatic transition between indoor and outdoor environments in our ROS package.

7. References

[1] J.S. Gutmann and D. Fox, “An Experimental Comparison of Localization Methods Continued,” in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems, 2002, vol. 1, pp. 454 – 459

[2] S. Se, D. Lowe, and J. Little. "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks." *The international Journal of robotics Research* 21.8 (2002): 735-758.

[3] A. Babinecj, L. Jurišica, P. Hubinský, and F. Duchoň. "Visual Localization of Mobile Robot Using Artificial Markers." *Procedia Engineering* 96 (2014): 1-9.

[4] D.Scaramuzza, and F. Fraundorfer. "Visual odometry [tutorial]." *IEEE robotics & automation magazine 18.4 (2011): 80-92.*

[5] *K. Konolige, M. Agrawal, and J. Sol, “Large scale visual odometry for rough terrain,” in Proc. Int. Symp. Robotics Research, 2007.*

[6] A. I. Mourikis and S. Roumeliotis, “A multi-state constraint kalman filter for vision-aided inertial navigation,” *in Proc. IEEE Int. Conf. Robotics and Automation, 2007, pp. 3565–3572.*

[7] E. Jones and S. Soatto, “Visual-inertial navigation, mapping and localization: A scalable real-time causal approach,” *Int. J. Robot. Res., vol. 30, no. 4, pp. 407–430, 2010.*

[8] D. Nistér, O. Naroditsky, and J. Bergen, “Visual odometry”. In *Computer Vision and Pattern Recognition*, *CVPR 2004.*

[9] Y. Matsumoto, M. Inaba, and H. Inoue. Visual Navigation Using View Sequenced Route Representation*. In Proc. of IEEE Int’l Conf. on Robotics and Automation (ICRA), volume 1, pages 83–88, 1996.*

[10] E. Kiriy and M. Buehler, “Three-State Extended Kalman Filter for Mobile Robot Localization,” Tech. Rep., McGill University, Montreal, Canada, 2002.

[11] I. Rekleitis, “Cooperative Localization and Multi-robot Exploration” , Ph.D. thesis, School of Computer Science, McGill University, Montreal, Canada, 2003.

[12] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, “Monte Carlo Localization for Mobile Robots,” in Proc. IEEE Int. Conf. Robotics and Automation , May 1999, vol. 2, pp. 1322 –1328.

[13] D. Fox, W. Burgard, and S. Thrun, “Markov Localization for Mobile Robots in Dynamic Environments,” Journal of Artificial Intelligence Research , vol. 11, no. 3, pp. 391–427, 1999

[14] N. Ganganath, and H. Leung. "Mobile robot localization using odometry and kinect sensor.", *IEEE International Conference on Emerging Signal Processing Applications (ESPA),* 2012.

[15] S. Se, D. Lowe, and J. Little. "Local and global localization for mobile robots using visual landmarks." *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001.

[16] V. Ayala, J. Hayet, F. Lerasle, and M.l Devy. "Visual localization of a mobile robot in indoor environments using planar landmarks." *IEEE/RSJ International Conference on Intelligent Robots and Systems,* 2000.

[17] A. J. Davison, et al. "MonoSLAM: Real-time single camera SLAM." *IEEE transactions on pattern analysis and machine intelligence* 29.6, 2007.

[18] G. Klein, and D. Murray. "Parallel tracking and mapping for small AR workspaces." *6th IEEE and ACM International Symposium on Mixed and Augmented Reality(ISMAR)*, 2007.

[19] M. Fiala, and A. Ufkes. "Visual odometry using 3-dimensional video input." *Canadian Conference on Computer and Robot Vision (CRV),* 2011.

[20] Huang, Albert S., et al. "Visual odometry and mapping for autonomous flight using an RGB-D camera." *Robotics Research*. Springer International Publishing, 2017. 235-252.

[21] Labbe, Mathieu, and François Michaud. "Online global loop closure detection for large-scale multi-session graph-based slam." *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.