CS 4758: Automated Semantic Mapping of Environment

Dongsu Lee, ECE, M.Eng., dl624@cornell.edu
Aperahama Parangi, CS, 2013, alp75@cornell.edu

***Abstract*— The purpose of this project is to program an Erratic robot so that it can automatically maneuver itself in a corridor and make a 3D map of the environment with interesting features labeled. The robot will take 2D pictures periodically as it maneuvers in a corridor, directed by a simple path planner which determines the robot's path from the depth images taken from a Kinect sensor, and from the 2D images, our algorithm will detect objects of interest and translate their 2D coordinates into 3D world coordinates. The algorithm is able to detect 3 kinds of objects; one is the exit signs, which are detected by using the color information and compactness, and the others are closed doors and fire extinguishers, which are detected by matching SIFT features from the template images of those objects. The resulting clouds of points which correspond to the locations of the detected objects will be filtered to produce well localized clusters, which will be added on top of the 3D map of the corridor. Since the pipeline that matches SIFT features and converts 2D coordinates in images to 3D world coordinates is complete, the user can easily add new types of objects of interest by adding the template images of the objects.**

I. Introduction

The purpose of this project is to program the Erratic robot so that it will automatically maneuver in a corridor and from the 2D images it took from its camera during the maneuvering, our program will extract the locations of exit signs, closed doors, and fire extinguishers and overlay the estimated 3D location of them on top of a 3D map of the environment. We believe that this information will help lifesaving efforts in a high risk environment tremendously.

The robot's path at a given time will be determined by a simple path planner which takes a depth image from a Kinect sensor attached to the robot, finds the optimal angle that gives the robot the path with the longest free space before it would hit an obstacle, and control the robot's speed according to the length of free space in front of the robot, so that it won't crash into a wall or an obstacle. While the robot moves according to the path planner's direction, it will take 2D pictures with the Kinect sensor periodically, and from those pictures, the algorithm will look for "evidences" that suggest the existences of the objects of interest. The algorithm will use the 2D location of the evidences in the photos, and the 3D location of the other points that constitute the 3D map of the environment to estimate the 3D location of the evidences in the 3D map. Then it will form point clouds with the evidences in the 3D space, and after filtering and processing, the algorithm will come up with the points that represent the location of the objects of interest. The algorithm will also display the level of confidence with each of the estimation of 3D position, by varying the size of the marker accordingly.

The 3D map of the environment will be obtained from a program named Bundler, which takes 2D photos of an environment and reconstruct the scene in 3D by extracting SIFT features from the photos and matching them, and it also stores the camera's estimated position and orientation based on the spatial relationship among the features. The information on the camera's position and orientation is stored in the output of Bundler, and will be used in order to estimate the 3D position of the objects of interest. The estimated world coordinates of the objects of interest are visualized along with the 3D map of the environment, using Processing, an open source programming language.

II. Issues and Implementation

A. Automatic Maneuvering of the Robot

Though the main goal of our project was to provide an extensible solution to the problem of semantically mapping environments, it was necessary to develop a platform specific navigation solution in order to acquire 2D images with which the output 3D map is built. To this end we developed a simple open space finder which operates using the Kinect sensor. Our technique takes the depth image from the Kinect and accumulates the values across each column. Then we perform a box blur on each column and turn the robot towards the column with the greatest average depth. The box blur is employed in order to enable the robot to avoid colliding into a corner by assigning a kind of a potential field to those columns near the corner. In addition, the program also drives the robot with the speed that is proportional to the average depth value of the path that the robot is taking, so that if the robot is very close to a wall or an obstacle, it will slow down so that it will have enough time to turn to the optimal path before it hits the obstacle. This simple solution allowed the robot to successfully navigate the test environment while avoiding obstacles and turning around corners.

B. Exit Sign Detection

First of all, the part of our algorithm which detects the objects of interest from 2D images was developed using MATLAB. The objects that the algorithm looks for are exit signs, closed doors, and fire extinguishers, and the exit signs are detected using different method from the other two; the detection of the signs exploits the fact that the exit signs are bright red. Therefore, each pixel was tested by its color; a pixel was considered as a candidate if it had an R value higher than some threshold and G and B values smaller than some threshold. After such a pixel was found, region growing was performed with the pixel as the seed point, and the adjacent pixels that satisfy the same condition were put together as a region. After the regions were marked, another test was performed to decrease the error rate, and this time, the shape of the letters of the exit sign was used. Since the letters have long and thin form, the ratio between the number of all pixels in the region and the number of periphery pixels (compactness) were computed for each region, so that the regions that were "thin enough" were accepted as an evidence of an exit sign, and their center of mass coordinates were stored. The example of input and output images are shown by the figures 1 and 2. It can be seen that the letters of the exit sign in the output images are marked with green dots, while the red fire alarm below the fluorescent light is ignored.

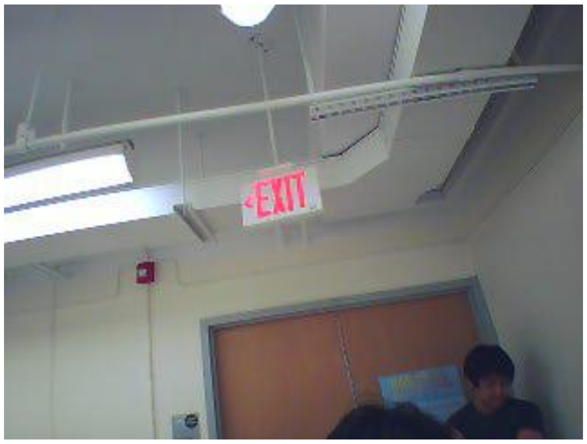

Figure 1. The Input Image

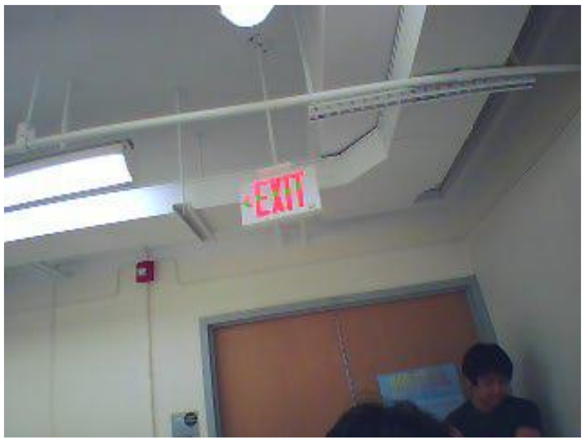

Figure 2. The Ouput Image

C. *Closed Doors and Fire Extinguisher Detection*

The other types of objects, which are the closed doors and the fire extinguishers, were detected by matching the SIFT features of the input images from the robot's camera to those of the template images. The applications that detects the SIFT features and matches the features from two images were supplied by David Lowe [3]. The template images were generated by taking pictures at the doors and the fire extinguishers with varying angle and cropping the region that contains the object. For fire extinguishers, 2 template images were taken, and for doors, 9 template images were made since there were a few types of doors in the environment where our algorithm will be tested on. The template images used for the project are shown by the figures below.

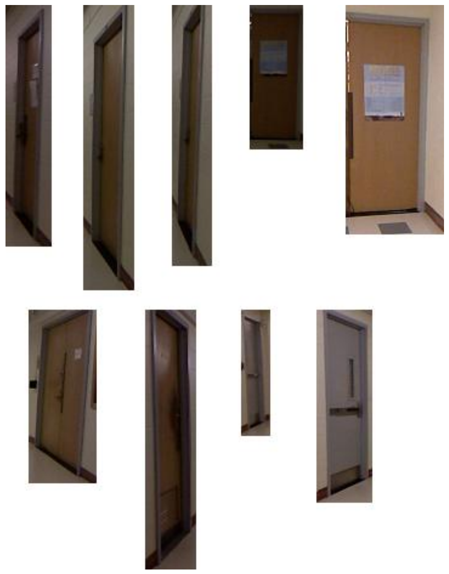

Figure 3. Template Images of Doors

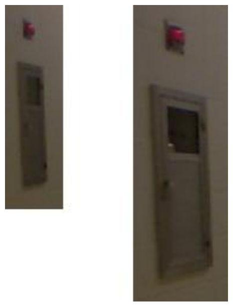

Figure 4. Template Images of Fire Extinguishers

It was found that the SIFT algorithm is able to detect matching features, even though the input images and the template images were taken from different angle, as shown by the figure 5, which was why it was deemed sufficient to use only a small number of templates. The user can easily add other types of objects that need to be detected by the system simply by adding the template images of the new types of object.

The SIFT features of the input images that were matched to those of the template images were taken as "evidences" of the existence of objects of interest in the same way the center of mass locations of the exit sign regions were. All of those evidences, which are represented as (x, y) position in 2D images, are then processed by the pipeline that converts the 2D coordinates into 3D world coordinates, after producing the 3D map of the environment using Bundler. Bundler will take all of the 2D images taken during the robot's movement, stitch them using SIFT features of the images, estimate the locations of the camera when those images were taken, and will output a 3D point cloud which consists of the 3D location of those SIFT features that form a 3D map of the environment, where the locations of the objects of interest are going to be overlaid.

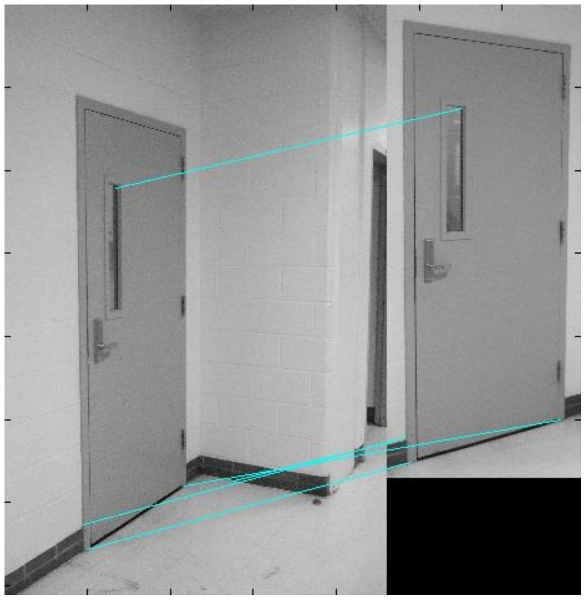

Figure 5. Matching SIFT features between two images

D. Estimating the World Coordinates of the Objects of Interest

Following the identification of the objects of interest and the formation of 3D map, we use a multistep technique to turn image space features into well localized world space features. First, for every object of interest feature we determine if there exists a Bundler feature within a certain radius in image space. We then throw out the feature if there does not exist such a point. The logic for this is that if a pixel is near a bundler feature pixel in image space, it is likely to be near in world space and therefore we can use the depth information of the Bundler feature.

After we filter out object features for which we have poor depth information, we then project the image space exit sign features out into world space using projection information given by Bundler for the nearest bundler feature point. This results in figure 6, in which we see several point clouds of varying density and localization.

Next we filter all the points in the cloud by calculating a density metric and then removing all points below a certain density threshold. This does a decent job of removing outlier points and therefore shrinking the clouds in the process. We calculate our density metric as

$$\sum_{i=1} \frac{1}{||x_0 - x_i||^3}$$

where x is the position of a sign feature in world space. This filtering results in figure 7. Few outliers remain and only areas of high density remain, as desired.

Then, in order to be able to report specific locations rather than just return point clusters, we coalesce each point cluster into one common point where that point's position is the average position of all points in the point cluster. We do this by running an algorithm that looks at pairs of points and if they are within a sufficiently small radius, marks them as visited and adds a new point to the list with a position that is the weighted average of the positions of the child points. In this way we collapse contiguous regions into single points.

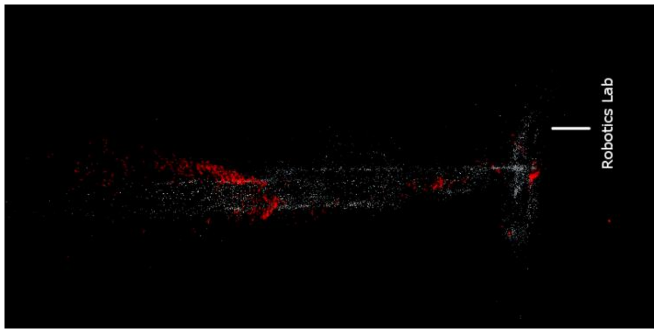

Figure 6. Point cloud of features, before filtering

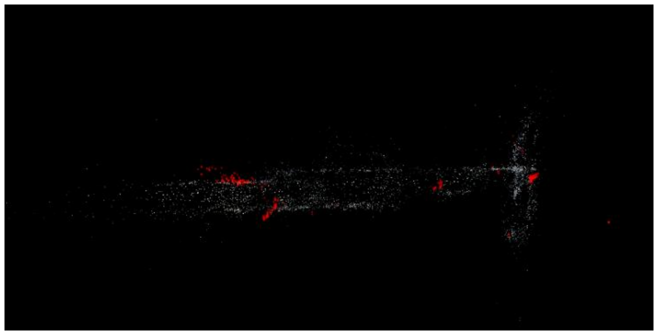

Figure 7. Point cloud, after filtering

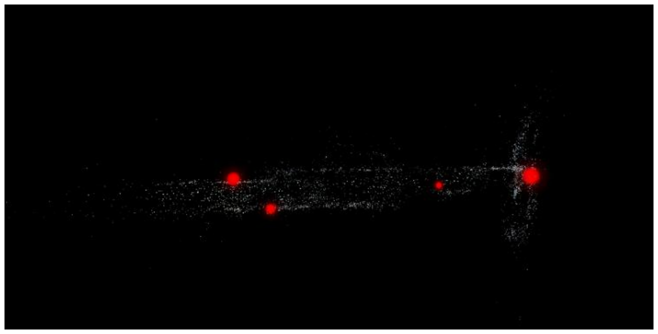

Figure 8. Localized exit signs from the clusters

Finally we remove clusters with a sufficiently low weight on the grounds that those clusters are outliers or we do not have enough evidence to support reporting them as discrete objects. The result of this technique in total is shown in figure 8. The size of each point represents the number of points from figure 7 that was merged to form the point, which shows our algorithm's confidence on the estimations of the location. The same process is repeated for different types of objects to get the complete map with the semantic information.

III. Experiments

Our dataset is a set of pictures taken from the third floor of Upson near the robotics lab and the point cloud produced by bundler's processing of those images. While the hallway is clear of obstacles and relatively devoid of distracting data, we aimed to make our techniques as robust as possible in order to function in cluttered environments.

Collecting input image data was primarily performed with the onboard camera of the AR.Drone, with additional data collected using the Erratic robot platform. While using the AR.Drone, data was collected using manual controls while holding the robot. In the case of the Erratic robot, the robot was set to maneuver in any direction around its environment while avoiding obstacles and collecting images of the environment taken from the depth camera at a rate of 5hz.

First of all, it was confirmed that the simple path planner for the robot was performing good enough to enable the robot to just roam around the corridor while it is taking pictures, while avoiding obstacles. However, it was found that due to the narrow angle of view of the Kinect sensor, the robot had difficulties when the obstacles were not visible by the sensor, although they were very close to the robot. Also, the Kinect sensor was mounted too high on the robot, so it had difficulties when the obstacles were very close to the base of the robot. Aside these limitations regarding hardware, the robot was able to find its optimal path for taking pictures of the environment.

As for the perception part of our project, which is the main component of the project, the output 3D map of the environment, with the locations of the objects of interest overlaid, is shown by the figure 9. The features are labeled as indicated by legend and the yellow line of dots shows the path of the robot, which is estimated by Bundler.

The original plan was to superpose this map to the floor plan of this building, but however, it was not possible to get the plan, so the output map had to be validated in an ad-hoc manner. Here we used Dice Similarity Coefficient for our accuracy measure, by matching the ground truth and the output 3D map visually in the corridor where the test image data set was taken. For instance in the case of exit signs, three out of the four identified exit signs were seen to correspond to real exit signs in the test environment and one was determined to be a partly obscured red poster, hence giving the DSC of 0.75. The doors gave 0.66, and finally the fire extinguishers gave 0.4.

Generally speaking, object recognition is a hard problem and robust recognition in unknown environments is difficult. Therefore we believe that our accuracy is primarily a result of the difficulty of building effective classifiers. However in spite of this, we believe we achieved good results in converting inaccurate classifier data into localized 3D positional data, resulting in a nice 3D map that the user can refer to when exploring the area, which is the main objective of this project. In general we correctly estimated the position (determined by inspection) of features and inaccuracy primarily stemmed from false positives from environmental noise.

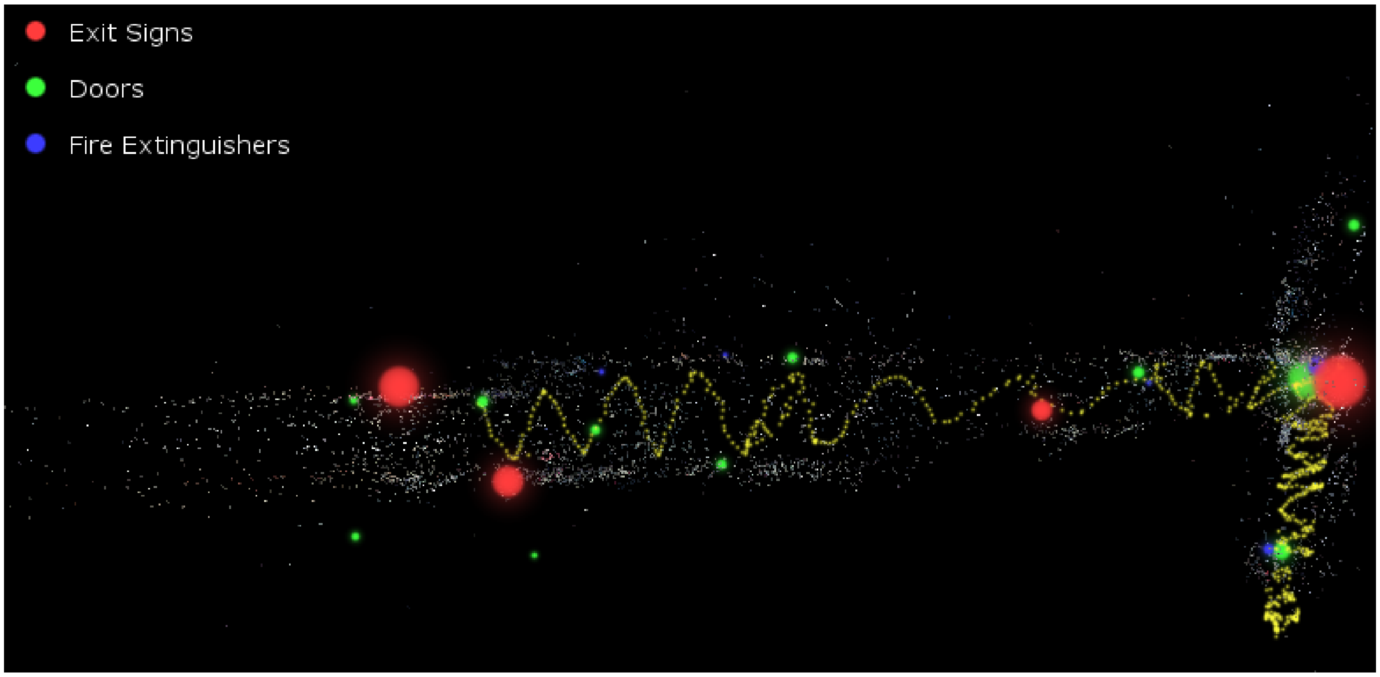

Figure 9. Output 3D Map from the Algorithm

IV. CONCLUSION

For this project, the Erratic robot was programmed to automatically maneuver in an indoor environment, and from the 2D images the robot took during the maneuvering, our perception part of the algorithm detected the objects of interest, which consist of exit signs, closed doors, and fire extinguishers, and their location in the 2D images were converted to a 3D map. The robot was able to move in a corridor without bumping into an obstacle or a wall, and although the localization of the objects did not yield such a good result due to the limited performance of the feature detector, the algorithm was able to come up with the 3D world coordinates of those objects, and combined with the map of the environment given by Bundler, the program was able to yield a good 3D map that can be valuable for the further exploration of the environment.

REFERENCES

[1] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo Tourism: Exploring Photo Collections in 3D. SIGGRAPH Conf. Proc., 2006.

[2] Noah Snavely, Steven M. Seitz, Richard Szeliski. Modeling the World from Internet Photo Collections. International Journal of Computer Vision (to appear), 2007.

[3] Lowe, David. "Keypoint detector." David Lowe. University of British Columbia, July 2005. Web. <http://www.cs.ubc.ca/~lowe/keypoints/>.