CMPUT 412 Competition 4: Egg Hunt

Jae-yeon (Leo) Yoon

Sumeyye Adyin

Objectives

Generally, the objective of this competition was to combine gained knowledge throughout the course into one applicable program. The concepts include:

- State Machines (smach)
- Robot Movement
- Odometry
- Mapping and Localization via Simultaneous Localization and Mapping (SLAM)
- Map construction via GMapping
- Camera Calibration
- Feature Detection via ORB
- AR tag Detection
- Navigation via Adaptive Monte Carlo Localization

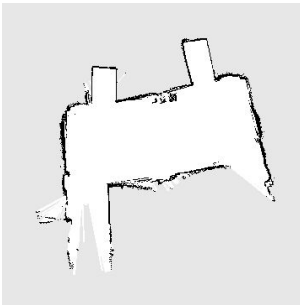
Mapping concepts were applied - with SLAM - to construct a map of a designated area on the second floor of the Computing Science Center at the University of Alberta. Through construction of a relatively accurate map, our robot was able to navigate near the walls of the area to potentially locate AR tags and/or University of Alberta emblems.

Background

The Robot Operating System (ROS) was the main software component used in our Turtlebot 2 machines. Documentation of ROS can be found at http://wiki.ros.org/ .

For global localization, the robot used both the map.pgm and map.yaml files to simulate its relative location in the real world environment. As stated in the objective, the .pgm and .yaml files were produced through SLAM vis GMapping. Documentation on GMapping concepts can be found at http://wiki.ros.org/gmapping .

In general, the GMapping algorithm distributes a Rao-Blackwellized particle filter in the simulated environment to predict both the location of the robot and the position of obstacles. Through information recieved from our visual sensor (Asus Xiton Pro Live Camera), the algorithm alters its previous predictions to generate a more accurate location in real-time.



Generated map of competition area

During the navigation portion, Adaptive Monte Carlo Localization (AMCL) - a probabilistic localization algorithm - was used to keep track of the robot's location within the generated map along with the real world environment.

Hypothesis

Our competition algorithm allows the Turtlebot 2 to navigate the given environment while detecting AR tags and University of Alberta emblems placed in various locations on the walls within five minutes.

Materials

- Turtlebot 2 Robot
- Asus Xtion Pro
- ROS
- Logitech USB Camera

Method

Localization

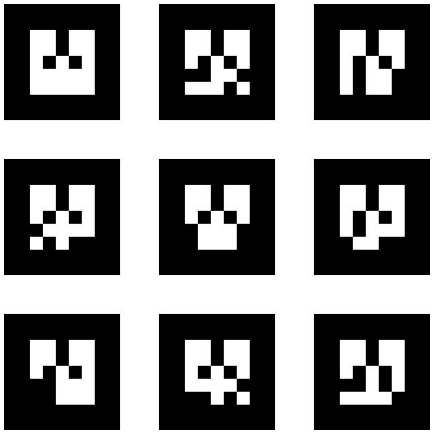
Initially, we run the *global_localization()* function provided by rospy to localize the robot within the environment - regardless of starting location. The *move_base/clear_costmaps* service is called to allow an external user to message *move_base* to clear obstacles in the costmaps generated by *move_base*.

Navigation

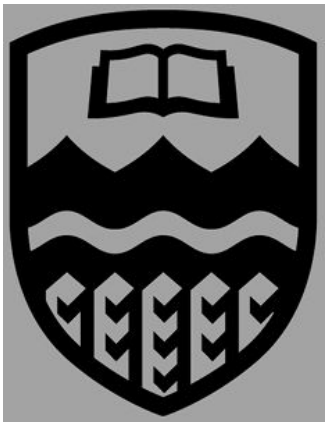
Navigation around the area was done via setting four waypoints on the corners of the competition zone. The waypoints were estimated through trial and error within rviz to obtain a desirable patrol route.

During movement, if the side USB camera was to pick up either an AR tag or University of Alberta emblem, the state would change to the docking procedure. The task of navigating the environment and avoiding potential obstacles was left to ROS's package, *turtlebot_navigation*.

Tag Detection



AR Tags



University Emblem

Originally, the multi-scale template matching technique (concept from https://www.pyimagesearch.com/2015/01/26/multi-scale-template-matching-using-python-opencv/) was going to be used to detect both University of Alberta emblems and AR tags. But, we soon realized that this method was only to be used for static images. Using this method would result in high amounts of false positives, as AR tags are simple in structure.

In the end, we decided to divide emblem and AR tag detection into separate approaches. The AR tag would use the provided *ar_tracker_alvar* package provided by ros. For emblem detection, ORB feature matching was used (concept from https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_orb/py_orb.html). Thresholds were adjusted to ensure elimination of false positives.

Docking

As specified by the competition, the robots were to dock within a 50 cm x 50 cm square before moving onto the next waypoint. Unfortunately, due to time constraint, this portion was incomplete for use during the competition. The idea was abandoned, and stopping in front of the tags was implemented instead.

Results

Unfortunately, our robot was unable to qualify according to competition specifications. Our robot was able to stop in front of AR tags and play a sound. Although, the distance from the tag was greater than 1 m; thus, was not accepted as successful tag detection. Although, our robot was able to localize itself as expected, and navigate around the area without any problems.

The videos of or attempts can be found here:

- First attempt: https://youtu.be/cbVr9zPsujQ
- Second attempt: https://youtu.be/2VMNP3ABf2Y

Discussion

The most difficult and time consuming issue to tackle during the competition was detection of both types of tags (University of Alberta emblems, and AR tags). Although the detection system worked at a stand still or slow sweeping motions, the Turtlebot's movement speed was too high for consistent detection. Adjusting the move_base's maximum velocity was attempted but due to poor implementation of the official package, we could not alter the variable during the competition. Adjusting the location of the camera may have changed the result of our stopping distance from the tags.

Conclusions

In the end, our goal was to further understand SLAM, feature detection, and navigation of a map via AMCL. From the results of the competition, we can see that there is definite room for improvement. Our robot was unable to meet specifications due to it not being able to stop within 1 m of Overall, the results of our program fails to support our hypothesis, due to its inability to reliably locate tags while moving and successfully dock within the docking zone.

Future improvements would include:

- Working docking feature
- Reliable feature detection during movement
- Better mapping/more accurate mapping