Jorge Anton Garcia
Team D – CuBi
Teammates: Laavanye Bahl, Paulo Camasmie, Changsheng Shen (Bobby), Nithin Subbiah Meganathan

ILR06
Sept. 26, 2019

Individual progress

For this progress review I worked on adding functionality at the controls level, allowing cubi to reset odometry, testing drift, and creating a series of scripts to test different subsystems.

CuBi Technical

My first goal was to compare how much drift cubi experienced with and without the new large caster wheel. To make this process easier, I added functionality to our mobility code. Instead of always specifying the absolute position the user wants cubi to move to, cubi can be commanded to move forward x centimeters or rotate by x degrees. I also added the ability to tell cubi to move to a relative position from its current position. With this new functionality, I sent cubi the following commands:

- Move forward 20cm
- Rotate 90º
- Move forward 130cm
- Rotate 90º
- Move forward 30cm
- Go back to start

Cubi moved a total of 3.3m and rotated 360º and had an average drift of 5cm regardless of whether it had the larger caster wheel or not. However, whenever there was just a translation involved, cubi's localization accuracy significantly increase. When sending it a command to move forward 1.5m, it would only move 1.3m and think it had moved 1.3m. With the new caster wheel, it was within 1 cm.

I also created tests for different subsystem. Previously, we would have to run the entire state machine to test whether or not the manipulator worked or if cubi's controls had improved. Now I have created functions to test for drift, all manipulator movements, and resetting odometry.

On the technical-side one of our biggest risks is having poor localization. One way to mitigate this is to reset every time we drop off a toy. To find a toy, cubi will travel at most 8.4 meters (estimated total distance to explore an entire room of 4x5 meters). Given our drift is less than 5%, we can expect cubi to drift at most 42cm. At this distance we expect to be able to see the AR tag in front of the box, so we think it is the easiest viable option.

Several decisions needed to be made regarding when and how to reset odometry. We decided to do so only once per drop off as we calculated that given the drift, this would be enough to find the box each time. Other options we consider was to always reset it when we saw an AR tag or to do so after some amount of time passed. Then there was the question of how to go about resetting the AR tag. If Cubi resets odometry with values which are very noisy, it could be a huge risk to our success. It will affect its ability to find the box again and hence drop off toys. Therefore, I decided to have cubi stop, look at the AR tag, read two values and reset odometry if the values are similar. Finally, I decided to put the AR tag on the box. Even though it affects the

angles at which cubi can see it, it is the most general case and we won't have to be moving the AR tag in the case that we put it on the ground.

I created a state in which cubi would reset odometry after picking up a toy. It goes towards the box, points at it and calculates how much it has drifted. The drift was calculated because we know where the AR tag should be with respect to odom (where we start) and we have measurements of where the robot believes it is. I experienced issues updating odometry though.

Other smaller tasks I worked on were:

- Reduce our rotational accuracy by a factor of two. Before cubi could be anywhere plus or negative ten degrees from the desired orientation, now it is between five and negative five degrees.
- Debugged issue of controller not working with Bobby and Nithin
- Caught Paulo up with how the controls and mobility portions of cubi work
- Debugged a 5cm offset error we always had because odometry starts in between the wheels, but we calculate desired position from the front of the base. This issue will be resolved once we can reset odometry.

CuBi Project Management

Every week, I keep our task manager up to date. I am realizing we are having issues as everyone's tasks are not granular enough, so I am trying to help the team do so. This will improve accountability and track how much progress we have actually made.

Challenges

Technical

I had issues resetting odometry after I calculated the drift. I first tried to send a reset odom command using the rostopic used by the turtlebot to reset odometry. I was unable to get that working and even if it did, we would not be able to use it as we want to reset odometry from a distance. We do not want the current location of cubi be the new center of the world frame. I then tried broadcasting a tf from odom to one of the frames on the robot, but then that makes the robot stay still even if it moves. My next attempt will be to try to save the offset and transform all the commands by that offset. This has been implemented, but I was unable to test it.

Team Work

Nithin: He caught me up on how to install the realsense. This will be important as I am finishing the controls and mobility portion of cubi and begin working on computer vision aspects.

Paulo: He created a design to attach a new caster wheel. I tested how good it was to see if it improved drift. This is very important as it could solve some of our localization errors.

Laavanye: He worked on creating an occupancy grid with Nithin. Bobby and I will then use the occupancy grid to plan a path to explore the room.

Bobby: Worked on 2D SLAM using Hector SLAM. We started planning the global planning portion that will be necessary to efficiently explore the room.

Plans

I will fix the odometry reset and create our first base line test of the semester. I will also work with Bobby to design a new planning algorithm and I will install realsense to gain familiarity with it.