CS189 - Prof. Radhika Nagpal May 2nd, 2018
Anisha Mittal & Daniel Soberanes

FETCH-Candy! Final Paper

Summary

FETCH-Candy! was a project intended to fully push some of the capabilities of our Turtlebots to the limit. Through the usage of EKF positioning, depth sensing cameras for obstacle avoidance, AR tag processing for localization, and all the other techniques implemented in the previous psets, the goal of this project was to emulate behavior that would be observed in a convenience store: a customer asks for a type of candy, and an employee moves to the appropriate location in the store, retrieves the item, and returns it to the customer. The only difference in our scenario was the fact that the employee was our Turtlebot, Athena. This was a very interesting and, while exhausting and stressful, fun and worthwhile experience. The results of our project varied and, in true robotics fashion, there were times during the process of making our final project where everything would work one second and then not work at all the next for no apparent reason. However, as we pushed through the challenges we were aware of how everything going wrong allowed us to worry about accounting for more and more things with our code, making the project more lasting and rewarding.

Our overall strategy was guided by the idea of dynamically setting the goal locations of each candy station. Rather than spending the time measuring the distances between each AR tag and establishing a set map for our robot (and, to be honest, under the impression that we weren't allowed to do so), we wanted Athena to have the ability to receive a tag number and then be able to reach her goal regardless of where the tag was located, even if the tag was later moved. We

felt that this strategy was more in line with the essence of the course, as a true autonomous system should be able to navigate scenarios where the environment is not always static but dynamically changing. So, in essence, our strategy consisted of having the robot start out facing a tag (and record its home location), travel to a set midpoint between the service stations and the candy stations (which would change depending on which home station the robot started at), spin around at that midpoint until the camera came into view of the requested tag, and then use trigonometry and the capabilities of the robot to set the global coordinates for the requested candy station and move towards it. Then, after acquiring the candy, Athena would utilize the previously stored home location to move back to another midpoint, search for the "home tag", and then move towards it in a similar manner.

While in the end we were happy with our decision of developing our project in this way, because it felt more interesting, extensible, and dynamic, it did have some setbacks hinging on it. The most standard one was that relying on the EKF as part of our method of determining the coordinates for the AR tags would lead to errors inherent in the standard drift that the accuracy of the robot's current position experienced. As we will discuss further in this paper, a huge part of the struggles and lacking side of our results was a direct consequence of the way in which we applied trigonometry to calculate the global coordinates of the AR tags. After much tinkering with them and trying to ensure that all the signs worked correctly, Athena was able to very closely reach the set points that she needed to. However, the error in calculation and the slight variations in Athena's movements and position calculations made it so, as a final result, most of the coordinates that she reached after seeing the AR tags were just a little off than what they should've been. Overall, however, we were satisfied with the results as Athena's movement through the "store" and interaction with other robots was quite satisfying.

Milestone 1 Strategy

Reaching Milestone 1 was probably the most time consuming part of our entire project. Our strategy, as mentioned above, consisted in dynamically calculating the global coordinates of the AR tags. In order to test this we first set up the robot in front of an AR tag and coded up the first version of our trigonometry to transpose the local coordinates that the robot received from the AR tag, in addition to the current position of the robot, to the global coordinates. After some initial work, we finally were able to get Athena to perfectly move onto the designated box. However, to our dismay once we set Athena off to an angle of the tag and tried to get her to go onto the box again, she happily turned in exactly the opposite direction and wandered off.

This resulted in several more hours of us attempting to modify the trigonometry and work out what was going amiss. Eventually we realized that we needed separate conditions for the calculation of the coordinates in the spirit of *if one angle is negative and the x coordinate is negative, calculate it in this way; else, calculate it in this other way*. Within the run method, we store the x and z values returned from processing the AR tag as x_ar and z_ar. We also store the distance from the robot to the AR tag using the distance formula between the current position and the relative AR tag position. Using the distance and x_ar, we can find the angle between the robot and the AR using arcsin. We called this angle alpha and the robot's orientation angle beta. After understanding the signs of the axes in each direction and testing the various cases (when the AR tag was to the right or left of the robot when the orientation was positive and negative), we applied trigonometry to the distance, beta, and alpha to calculate the global x and y coordinates of the AR tag. After a lot of working through issues with signs and more calculations, we were able to get Athena to almost consistently always travel onto the square she was supposed to go to very accurately.

Milestone 2 Strategy

For milestone two, our previous calculations once against went haywire. Our initial intent with this milestone was simply to add the code that would allow Athena to set her home location as her starting position, travel to her designated midpoint, and spin around until she was able to accurately see her goal AR tag and move towards it using the code that had been concocted for the first milestone. However, this turned out to be a much more challenging task than we had previously attributed to it.

First, to work the longer wandering motion of the robot into the code, we used proportional controls for both the linear movement and the angular movement. We fully implemented a state machine for Athena. She would initialize into an ‘order’ state where she would store her current location as her home, and then wait to receive a number for the candy-dispensing station she was required to go to. Then, she would move into an ‘orient midpoint’ state in which she would use a calculated angle difference and a destination orientation to rotate until it was within a certain angular threshold that determined that Athena was in fact facing her goal destination of the midpoint. Then, she would simply advance towards it using the linear proportional control to slow down as it approached its destination. Once Athena decided that she was in fact at the midpoint, she would enter the ‘midpoint’ state which allowed her to simply start slowly rotating in place. As she was rotating, the process_ar_tags method was looking for an AR tag in view that matched the ID that Athena had received when waiting for orders. Once this tag came into view, Athena would enter the ‘orient’ and consequent ‘approach’ states.

These worked very much the same way that ‘midpoint orient’ and ‘midpoint approach’ did, except now the given destination was the global coordinates that Athena calculated from the AR tag using the trigonometry that we programmed. This is when we started running into more issues with the trigonometry. Before this time we hadn’t really thought about the fact that Athena’s initial position was facing away from the rest of the AR tags, meaning that the initialization of our coordinate system was backwards. Because of this, the robot’s orientation (which we used to calculate the global coordinates) was off by an offset of around 180 degrees, since this accounted for the robot turning around before moving. This led to us spending a few more hours attempting to rewrite and fine-tune these new angles that we were working with, eventually opting to normalize the angle of orientation (beta) to be in a range of zero to 90 degrees. We then used a similar formula as in Milestone One with distance, beta, and alpha to find the global x and y coordinates of the AR tag. Once this was set, we were pretty happy with our results for the second milestone.

Milestone 3 Strategy

The third milestone turned out to be the one we were most confident in implementing. The previous code we had used to have Athena wander around without crashing into objects in a previous pset had worked really well every time we’d used it in the past, so we decided to integrate it into our code for the final project as well. As before, the depth of the view in front of Athena is processed by taking in the data from the camera and using a threshold in order to set a range of 0.6-0.8 meters that Athena can detect in front of her. If an object is detected, it is marked as being in the left, center, or right of the camera view, and this information is stored in self.cam_dir. Then, in run, Athena checks to see if process_depth_image has detected an obstacle, in which case she will turn away depending on the location of the obstacle. If the obstacle is to the right it will turn left and vice versa, and if the the obstacle is in the center of the field of vision it will recall the direction in which it previously was turning and do that again. If the robot gets stuck between two obstacles that are in close proximity (i.e. turning back and forth between them ten times or more) it will trigger its failsafe and perform a turn of about 90 degrees, again in the direction of the previous turn. If no obstacle is detected in front of Athena, then checks are performed for the bumpers. However, if no bumper has been triggered then Athena will proceed with the rest of the code previously implemented.

One addition that was necessary for the project was to ensure that if Athena detected another robot currently at the station she was supposed to go to, she would wait until that robot left the station and then move forward rather than try to go around the robot. This was implemented so that, if Athena was in the state in which she would be making her final approach onto the AR tag, she would have a final check for depth in front of her. If there was an obstacle, she recognized this as a robot and waited until the obstacle was gone. Otherwise, she initiated her final approach. In this final approach state we decided to disable the depth sensing so that Athena wouldn’t be deterred from the destination due to sensing the wall.

With this implemented, all that was left was for us to determine a different midpoint based on which ID tag Athena detected as its “home” upon initialization. Once this was implemented, then the state-based program took care of everything else. Each state had its own set of specifications that allowed for depth to only be sensed at certain times, for obstacles to be avoided versus waited for, for rotation and approaching and homing to all be compartmentalized which all in all resulted in a simpler way for us to treat each stage of Athena’s development.

Final Demonstration

While we were very satisfied with the final tests and results of our robot, we knew that our final demonstration had the potential to be better than it was. As mentioned before, even after hours of tinkering with the math and making sure that we were able to correctly acquire and transpose the exact coordinates necessary for Athena to reach the candy dispensers, there were still some kinks that prevented our calculations to be exact. Sometimes the distance was just a little larger in magnitude than it was supposed to be, which resulted in Athena attempting to go behind the walls of the playpen area. Other times, she would stop just short of acquiring the candy. All in all these were frustrating situations since we were aware that perhaps if we’d gone with a static map approach would have been handled in a much easier way. However, we were really happy that we had committed to challenging ourselves with our approach despite some of its shortcomings. We also attempted to reduce the impact of some of these small deviations by adding offsets to different scenarios, for example *if you are approaching tag number 7 make sure to go half a meter further than you normally would*. However we weren’t too excited about doing so since it was a last-minute after thought and more closely approached simply hardcoding the locations of each AR tag, so we only did this for one of the starting locations.

Overall, we were able to capture Athena correctly collecting candy, returning it to the home station, venturing out for more, and correctly avoiding other robots and obstacles correctly. We know there was still work left to be done in terms of perfecting our results, but overall in the time we had with the resources we were given, we were very satisfied with Athena's performance and happy that we decided to stick with her throughout all our hardships.

Ethics

We are very aware than in an ever-changing environment of growth and innovation, it is easy to always side with the progress and forget the parts of the situation that this progress is replacing. However, we firmly believe that a lot of the developments in robotics such as, for example, robots taking charge of managing a store, fetching products, and tending to customers, would not be a situation of total overtake and absorption of the other employees by the robots. A robot's job should be to reduce the amount of time that is required of humans in menial and repetitive tasks, such as the ones mentioned above. Something we would hope our project would allow would be for people to have more freedom to devote their time to fulfilling and rewarding jobs. While customer service can at times be considered awful due to unpleasant interactions, it is also arguably the only part of a store clerk's labor that doesn't require simple, boring, repetitive motions of fetching items, charging customers, and the likes. If robots were at least partly implemented into a system, then humans would be free to welcome the customers into the store and help them out in a more personable and enriching way than would be possible if they were being held back by the weight of deadening jobs.

Beyond that, initiating an implementation of robotic help in such services would then allow and even encourage the future development of more complex robots, which would in turn open up the possibility for these workers to instead of being bogged down in the menial tasks receive training and acquire incredible skills to produce and maintain these robots, skills that would be far more fulfilling and more than that could be further applied to so many other walks of life than simply the simple store scenario that a worker was previously involved in. The introduction of robots can allow for the introduction of a universal basic income. It allows for people to pursue opportunities equally and focus on jobs that enhance finding excellence in skills and more artisanal crafts as well as contributing to society. Humans can be allowed to make a social contribution, experience community, and realize their many-sided potential following these solutions. Rather than allowing for robots to completely erase the necessity for humans, the introduction of artificial system also introduces new scenarios in which a more resonant harmony and essential symbiosis between human and machine can enhance human experience beyond comprehension.

PROJECT VIDEO LINK: https://youtu.be/lzL4TK4DVnw