Criterion A: Planning

The Problem

My client is my father, who is one of the owners of C-Tonomy, LLC. C-Tonomy, LLC is a company that produces outdoor surveillance UGVs. Currently, their UGV, or unmanned ground vehicle, can patrol around a path and detect large objects around itself. According to my interview with my client, the problem is that "The LDAR is a planar scanner – it only sees big obstacles. It cannot detect something that is not big, like a boulder. The scans just go over it." The UGV is still in development; if it was on the market then the UGV would not be able to effectively follow a path automatically because it would run into anything shorter than its LIDAR. The UGV needs to be able to determine for itself what to do based off of sensor data on these smaller obstacles, as the UGV is autonomous. The problems associated with whatever solution that could be created is to make sure that the data is reliable and how much information can be obtained. It will be significantly more useful if the UGV can determine how large the obstacle is so that it can determine what path to take to avoid it. Whatever solution that is created, which will most likely be through a microcontroller, needs to make sure it isn't reading false positives, which can be difficult.

Proposed Product

My proposed product would be a system of time of flight sensors at varying angles to detect obstacles. The reason that this system would be used is that it is a cheap and effective way to solve the problem. Distance readings need to be obtained from around the UGV so that appropriate action around obstacles. Time of flight sensors are relatively cheap compared to other options, and from experience are more accurate than ultrasonic sensors. Exactly how many sensors and at what angles is dependent on the UGV, so the physical aspect of my proposed product will be created by me in collaboration with my client/adviser. There would need to be an algorithm to detect significant changes in the distance a sensor detects over multiple sensor readings so that it can avoid false positives on slopes and misreadings. The multiple sensors would help tell the robot where the obstacle is in relation to itself. The microcontroller would report where it thinks the obstacle is coming from and also how far away it thinks the obstacle is. An Arduino would work best because a more powerful microcontroller is not needed. An operating system is not needed, I just need to be able to process the data through an algorithm. If an Arduino is used, that limits the programming languages to Arduino C or Micropython. I have chosen to use Arduino C because that is the language my client is more familiar with Arduino C so updates/debugging by my client are more likely to happen. There is also a library that is provided for Arduino C that allows me to obtain sensor information from the time of flight sensors. There is no library with Micropython, which means it would much more effort in order to achieve the same results.

Success Criteria

- The system can read distance from the sensors
- An algorithm can determine whether it is more likely that a sensor is reading a false-positive or an actual obstacle to avoid.
- The system can determine approximately where obstacles are located with respect to the UGV.
- The system can communicate to the main computer about what actions to take if an obstacle is detected.

Word count

601