Jarred Wilson
#001277133
NIP1 — NIP1 TASK 2: DISASTER RELIEF ROBOT

The environment simulated with the Coppellia Robotic BubbleRob is that of a post tornado disaster area. The bot is confined to an area surrounded by rubble. Inside of that area, is a structure that is designed to resemble a damaged house with only three walls remaining. Surrounding the damaged house, are three other clusters of objects that are intended to help simulate the presence of debris on the ground that cannot be traversed through by the robot. This could be parts of a building, cars, trees, or many other objects thrown in the wind by a tornado.

The robot has the ability to aid a disaster relief team in several ways. One of the ways that this robot can aid a disaster recovery team is by gathering preliminary data on the surrounding area and providing clear routes to areas. Another example of how this robot will be able to help a disaster recovery team is by using the included sensor arrays to be able to find and locate those in need and report their location to the recovery team. This will be able to speed up the average time of recovery to give aid to those in need.

Additional proximity sensors were added to the front of the robot. During testing, it was clear that a single sensor on the front led to unfavorable results due to collisions just outside of the robot's sensor. This would result in the robot hitting an object in its path without knowing it or being able to account for it. Another change that was added to the robot was a disc shaped proximity sensor that is used to detect the presence of a person among the debris. This allows the robot to report the locations of individuals more quickly and precisely. The last modification that I made to the bot was the reduction in the reversal of the bot after a collision. Frequently while running the simulation, the bot would reverse after a collision, but would collide with objects behind it. This would result in the bot falling forward and not being able to move.

While configuring the environment and the bot, there were several instances where, I believe, that I was optimizing the bot to perform better in the environment. Behaviorally, I modified the degree that the bot would reverse and added its ability to to report when a sphere entered into the view of the proximity sensor. The bot can detect collisions and act upon that information. It could also be made to store data of its previous path as well as points where the bot collided with an object. The points of collision can be marked as to represent that there is an object, and can be used to make a pseudo map of the environment. This bot does not have to have a complete understanding of the environment. Because of its array of sensors and programmed behaviors, the bot does not have to be certain of the environment and will react accordingly.

The robot has several advantages in this scenario. It provides a way for a rescue team to traverse and collect environmental data in a disaster zone without having to unnecessarily risk their safety. In theory, you could deploy many of these robots more quickly than you could a team of people. While reducing the risk to the rescue team, the bot can quickly and effectively find clear paths through debris or notify the team on the location of people in need.

Disadvantages to this robot would include the possibility of the bot to miss individuals in an environment and result in the delay in deploying aid. Additionally the bot could become damaged or trapped by the environment and no longer be able to provide a rescue team with valuable information. The robot currently lacks a sufficient array of sensors and behaviors to eliminate negative outcomes. In practical use, there is no way to guarantee that a change to the environment could not result in a negative outcome for the bot.

We measure the success of the robot by its ability to quickly find victims and being to maintain operation time. While this simulation does not provide necessary tools to formulate the sufficient metrics, It could be modified to allow for testing these factors. The robot is not successful at its job if it misses victims or takes too long to find them. We must supervise and adjust the simulation accordingly to fine tune the results. The robot performs well if it finds all of the victims more quickly than the average time it takes a human task force to complete the same task. The robot solves the problem if it can perform as well or better than the human counterpart and if it successfully mitigates risks for the disaster recovery team (Russell & Norvig, 2010).

Testing the robot in a real world scenario can prove to be very difficult. There are many unknown variables that can make iterative testing a time consuming process. In order to effectively test this robot, computer simulations provide a quick and iterative testing process, but are far more limited than real world testing. As a result, it would be far more resource efficient to perform supervised learning initially. After the bot is performing well other methods such as reinforcement learning can be performed to help fine tune the bots behavior prior to testing the bot in the field.

Jarred Wilson
#001277133
NIP1 — NIP1 TASK 2: DISASTER RELIEF ROBOT

The prototype could have sensors added to it to better gather information on its environment. The bot could also store information about the environment to better map out and provide information to the rescue team. It also could be programmed behaviorally to adapt to the environment better. In AI, reinforcement learning is an effective method for training. In reinforcement training, the bot is given positive reinforcement for meeting the desired outcomes of the simulation (Russell & Norvig, 2010). Using this method, the bot can be taught behaviors that encourage its behavior in a way that meets the desired outcome.

Jarred Wilson
#001277133
NIP1 — NIP1 TASK 2: DISASTER RELIEF ROBOT

Works Cited

Russell, S. J. & Norvig, P. (2010). Artificial intelligence: A modern approach (3rd ed.)
Upper Saddle River, NJ: Pearson.
ISBN 9780136042594