Jeremy Greenwood: ID #000917613
Instructor Jim Ashe
Mentor Rebekah McBride
C951 – NIP1 Task 2: Disaster Relief Robot

SCENARIO

For this project, I wanted to explore the assessment of the effects of a structural fire. A structural fire is any fire "originating in and burning any part or all of any building, shelter, or other structure." (National Park Service) The effects of a structural fire can render a building unsafe for humans to maneuver during search and rescue operations. This creates a dangerous situation for first responders and rescue workers. It presents an opportunity to explore increasing the margin of safety through the use of autonomous robots.

SOLUTION

It would be impractical to assume that an autonomous robot would have ready access to a building's floor plan in an emergency situation. As such, the robot will have to freely explore the damaged areas without following a known path. To achieve this, I have used an algorithm known as "wall-following," useful for allowing one to escape a maze. (Dalton & Dalton)

"Wall-following" is a Depth First Search in its most basic form. The person within the maze selects a wall to follow. They then place a hand on the selected wall and travel through the maze, never removing their hand until they have exited the maze. Each room explored may be considered a node. Rooms with only one entry/exit may be considered leaf nodes. Each node is explored until the solution (an exit) is found. Because the search space is finite, the method is guaranteed to find an exit from the maze (a solution) if one exists. However, every corridor within the maze (or every room in the structure) will be searched. Thus, the heuristic is not optimal.

ENVIRONMENT

For this scenario, I used Coppelia Robotic's V-REP program to model a simple, single story floor plan. The floor plan consists of one 4x4 meter room, one 1x2 meter room, and four 3x4 meter rooms. In addition, there is a 4x6 meter entryway and a 5x2 meter hallway. Coppelia Robotics provides a robotic agent, bubbleRob, which I used to operate within the environment.

From the robot's perspective, the environment is:

Environment Characteristics	Explanation
Single-Agent	the robot bubbleRob is the only agent in operation
Fully Observable	sensors will detect all aspects relevant to the robot's choice of action(s)
Stochastic	the environment state is not dependent on actions of the agent
Sequential	the robot's actions are predicated on its prior spatial relation to the wall
Static	the environment does not change while the agent is deliberating
Continuous	the robot views the building as one long wall
Known	the rules governing the environment are known beforehand

Table 1. Properties of the target environment

STRENGTHS & WEAKNESSES

The single biggest strength of the bubbleRob robot is its configurability. V-REP provides multiple capabilities which can easily extend bubbleRob's functionality. In its initial configuration, the bubbleRob robot was not suited to adequately meet the rigors of this scenario. It lacked the ability to adequately execute the wall-following algorithm and was incapable of identifying a search target. To correct these limitations, I added additional sensors for each task.

In testing, a further limitation of bubbleRob's configuration became apparent. Steering was achieved by altering the drive velocity for each wheel individually. The use of 2 drive wheels and a rear pivot creates a sort of tripod configuration. This means that the robot is incapable of executing a zero-radius turn. In my implementation, near-zero-radius turns are necessary to execute a 90º turn in a confined space. I found the only solution to be a combined maneuver of reversing and turning. This allowed the robot to execute the required maneuver in a manner sufficient to attain the primary goal.

The most significant limitation was a result of using the wall-following algorithm. Left turns are simplistic, in that they are a result of running into an obstruction. Once the forward sensor encounters an obstruction, the robot will execute the aforementioned approximation of a zero-radius turn to the left. However, right turns are a result of losing "sight" of the right wall. As such, there is no sensor data to orient the turn. To compensate, a careful balance of sensor settings and movement responses is calibrated such that a turn to the right will place the right sensor back in range.

MOVEMENT

This environment presents multiple obstacles to autonomous robots. I chose to focus on two – walls and doorways – represented with the following conditions:

	Forward Sensor	Right Sensor	Movement Response
1	-	Wall detected	forward( )
2	Object detected	<input is ignored>	left_turn( )
3	-	-	right_turn( )

Table 2. Movement Responses to sensor input

The lack of reasoning abilities in the robot made developing the movement processes somewhat challenging. I could not simply instruct the robot to turn when it approaches a right-angle intersection. The simple programming of bubbleRob does not present any way to represent a right-angle. I attempted an approximation of this understanding utilizing a combination of the forward- and right- sensors. Ultimately, this proved unsuccessful.

In order to traverse an area using the wall-following algorithm, it is necessary for the robot to be able to maintain a parallel trajectory to a wall. This coincides with the first condition in Table 2. If the distance between the robot and the wall exceeds 0.15m or dips below 0.13m, the robot must correct its motion to maintain the distance. I implemented this with the responses in Table 3.

	Right Sensor Distance	Movement Response
1	< 0.13 meters	adjust_left( )
2	> 0.15 meters	adjust_right( )

Table 3. Additional Movement Responses specific to Right Sensor input

The second condition in Table 2 occurs when the robot approaches a corner. When this occurs, the robot executes a 90º turn to the left, altering the robot's orientation to align with the wall adjoining the corner. In the event that the sensor detects another forward obstruction, a series of 90º turns will execute until it does not detect an obstruction. When the sensor no longer detects a forward obstruction, the robot will revert to the default condition and resume moving parallel to the new wall.

The third condition in Table 2 occurs when the robot reaches an opening, representative of a doorway. When the robot's right sensor moves out of range of the wall, a right turn will be initiated, rotating the sensor back into range. The robot will continue forward, again taking the right sensor out of range. This creates an alternating series of right/forward/right motions that repeats until it can revert to either the first or second condition. This loosely equates to a right turn with a parabolic trajectory.

TESTING

The primary goal was for the robot to successfully navigate the test environment in such a manner that it would traverse the entire wall space and return to the start point. I approached this by monitoring for the 3 responses indicated in Table 2: forward( ), left_turn( ), and right_turn( ). The testing consisted of observing the robot in the simulated environment. When the robot was capable of executing the 3 maneuvers at all appropriate instances, the test was passed. Each maneuver was tested separately as a unit test. Additionally, the execution of all maneuvers in situ was tested. Regression testing was used to ensure that changes in one response did not negatively impact the operation of the other maneuvers.

A secondary goal was for the robot to locate the Target object and stop all movement. Testing for this, again, consisted of observing the robot in the simulated environment. When the robot was capable of executing the desired behavior (stopping all movement) as it came within range of the Target, the test was passed. This necessitated the use of a 3rd sensor, located on the left side of the robot. This sensor is dedicated exclusively to searching for the target. This is roughly analogous to looking for injured victims or structural damage and deformities, depending on the calibration and capabilities of the sensors used.

FURTHER DEVELOPMENT

The utility of the robot is ultimately predicated on its ability to actually find survivors of catastrophe, detect dangers to rescue workers, or evaluate environmental hazards. In its current form as a goal-based agent, bubbleRob lacks reasoning ability. It is incapable of achieving any of these goals autonomously. It can not even differentiate between a closed door and a wall.

One way to facilitate the development of this needed reasoning ability is to utilize machine learning. The use of Reinforcement Learning could help teach the robot to discern between closed doors and wall space, stable construction and impending collapse, or injured humans and inanimate objects. Clearly, this is an area that deserves the highest priority in development.

BIBLIOGRAPHY

National Park Service. "Fire Terminology." *National Park Service | US Forest Service,* https://www.fs.fed.us/nwacfire/home/terminology.html Accessed 20 Feb. 2020.

Dalton, Ruth & Dalton, Nick. “How to Escape a Maze – According to Maths.” *THE CONVERSATION*, 26 Jan. 2017, https://theconversation.com/how-to-escape-a-maze-according-to-maths-71582. Accessed 20 Feb. 2020.