Project Report

Objective

The goal of this project is to design a behavior-based fire detection robot that is able to move from an unknown position in an indoor environment to look for a fire, raise an alarm, and extinguish it. The robot should use a set of behaviors, including *wander* (search) and *wall_following*. The walls of the rooms are approximately 15cm high and the fire location will be indicated by a burning candle (in a glass candle holder) on top of a differently colored area the size of half a floor tile.

Files

- *main.py*: main driver code that the robot would read.
- *behavior.py*: contains the wall_follow and wander functions for the robot to implement

Robot Design

The robot was designed with durability and compactness in mind. With all the walls the robot was going to hit, we had to make the robot sturdy enough to not fall apart but also compact so that it would not have a tough time making its way around the environment. To do this we decided to build a platform to hold the EV3 brick directly above the motors. This platform had many openings for the pegs to be placed, which allowed for the touch sensors to be attached directly to the side of the vehicle. The platform also left enough room underneath the vehicle to put the color sensor. The color sensor was placed about 1cm above the ground as that seems to be the most effective range where the sensor can accurately determine color.

We decided to attach two touch sensors to the front of the robot, with a bumper connecting them. This is because the main point of contact that the robot will make with the walls of the environment will be at the front. The bumper was to ensure that the touch sensors would be triggered when the robot contacted a wall. Without the bumper, the robot may drive the main body into a corner without triggering the touch sensors.

The ultrasonic sensor was attached to the left side of the robot near the tail end. Putting the ultrasonic sensor near the tail end ensured that the robot would have room to turn when it entered the *wander* function.

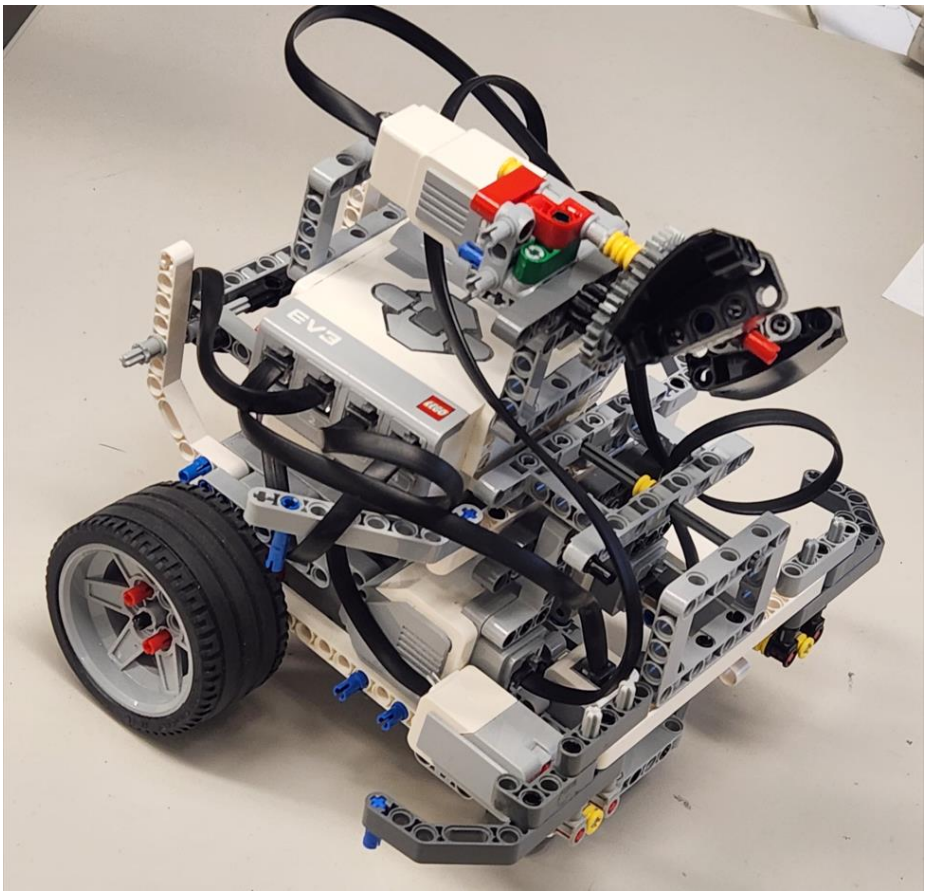

Figure 1: Final Design of the Robot

Wall Following Strategy

The robot's conditions to enter *wall following* are when it detects an applicable wall on the robot's left side. The conditions for a wall to be applicable is whether the wall is within 30cm according to the ultrasonic sensor attached to the left side of the robot. While the robot performs *wall following*, it will move forward along the way while maintaining 7cm away.

Responses to the touch sensors attached to the front have a higher priority than maintaining the set distance from the wall. In general, whenever the left bumper is hit, the robot will turn right. Likewise, whenever the right bumper is hit, the robot will turn left. Only when there is a wall on the left will the robot turn right after the right bumper is hit. In the scenario where the robot drives straight into a wall and both bumpers are triggered at the same time, the robot will turn left or right according to whether there is a wall on the left. If there is not, the robot will turn left, otherwise, the robot will turn right.

The robot will exit *wall following* and enter wander mode whenever the ultrasonic sensor no longer detects a nearby wall on the left. At that point, the robot will turn slightly to the left before entering *wander*.

Wandering Strategy

The idea was to have the wandering strategy be completely random, but issues came up when this idea was applied. We decided to filter the movement by having any necessary turns run for 0.3 - 0.4 seconds, randomly picking the right or left direction. We added *if statements* to deal with the various cases where the robot would hit or detect a wall. If a wall was detected to the left of the robot under the set distance threshold, the *wander* function would switch back to the *wall following*. When the color sensor detected the colored tile, it would stop and extinguish the fire. In the case that none of these events occurred, the robot goes straight for 0.5 seconds, turns for 0.3 - 0.4 seconds, and then repeats this until any of the cases mentioned above occurred.

Challenges Faced

There were three major challenges faced during this assignment: the design of the bumpers, the extent of randomness to implement to get to the goal, and the thresholds that maximize the robot's performance in various parts of the course.

When the edge of the bumper hit multiple walls consecutively, it would fall off. We went through four versions of the bumpers before deciding on the final one. These current bumpers are loosely supported by the main frame of the robot, providing support while also being loose enough for the touch sensors to still trigger.

The randomness of the robot's movement was also a polarizing factor within the group. If we made it too random it seemed like the robot would actively work against us to find the goal, but if we put too much structure it would not be able to navigate the entire environment, particularly in free spaces. To solve this problem, whenever in the *wander* function, we decided to give the robot the ability to turn randomly within a range. The robot can turn within a range of 0 to 60 degrees, either to the left or right of the robot, picked randomly using *randTime*. This allowed the robot to have the randomness needed to make it to the goal, while not being so random that the physical limitations of the vehicle would impede it.

The final, and biggest challenge was deciding the thresholds to use. Sometimes when going into a new room, the robot would hug the wall too tightly and continue in a giant loop. Similarly, the threshold to exit *wander* and move into *wall following* was too high, causing wander to rarely occur. By lowering the threshold to detect walls and follow walls, we managed to break this habit, allowing the robot to traverse the environment effectively.