https://docs.google.com/document/d/1rNny8oVHFjNhSoZElJKbUl6heKwAbwolAPBgzRRYa0A/edit?usp=sharing

General Methodology

This section is divided into subsections to present the process we followed to solve the problem.

Management

In order to maintain organization, we utilized a GitHub repository to make use of version control while making changes to codebase. Additionally, we wrote modular code in order to keep various components easily integratable and individualized during design.

Design

The overall design of the solution models a state machine where each state determines appropriate actions for specific parts of the track.

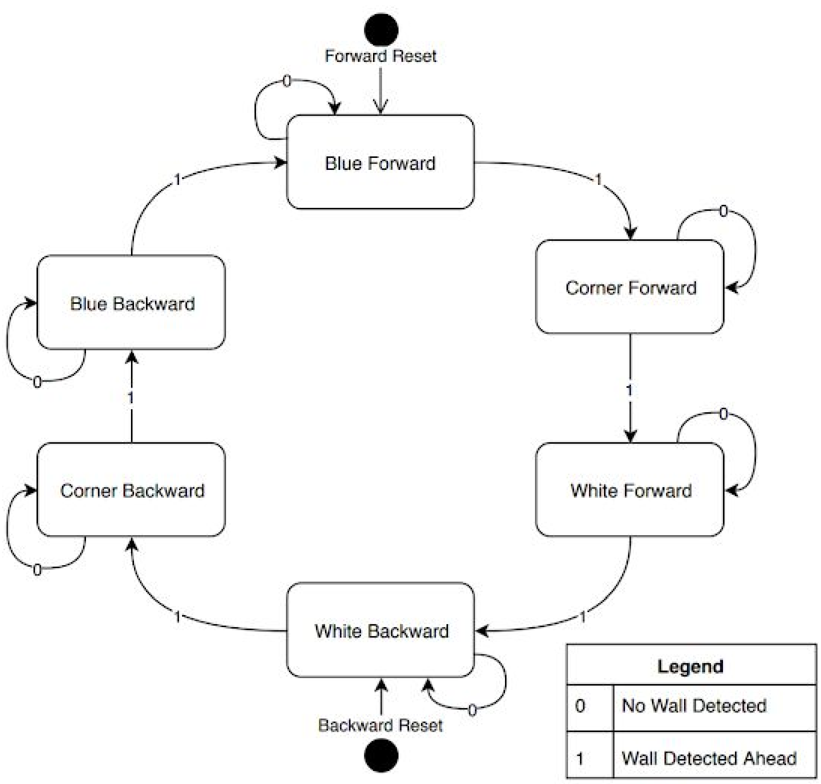

Figure 2. The states of this state machine correspond to different sections of the track.

Two main algorithms control the DE2Bot throughout the course. These algorithms keeps the bot in the specified region and reduces error between hardware and software components. The wall following algorithm utilizes a differential correction routine that uses sonar sensor input data to calculate an angle of correction (figure 3). The wall collision algorithm minimizes consequences of odometry drift by reversing the DE2Bot if it gets stuck on a wall and resets the odometry which allowed for a more accurate angle of correction.

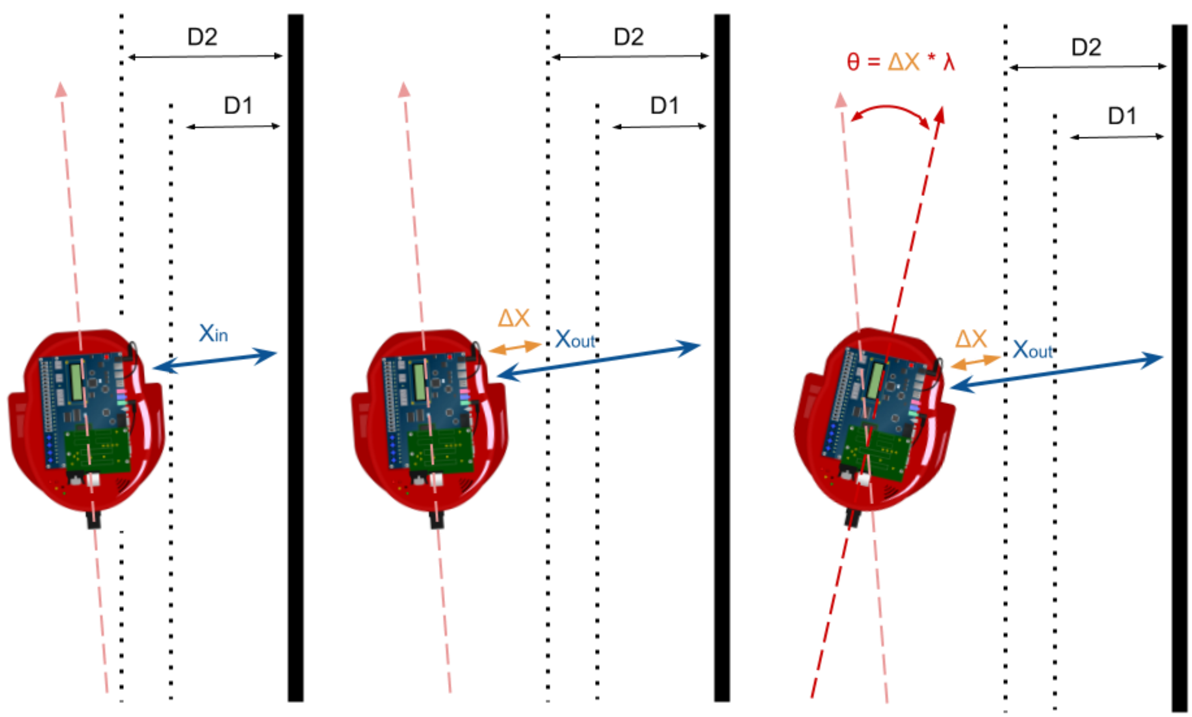

Figure 3. Differential correction keeps the bot within a certain distance threshold from the wall by using the distance from the threshold to determine the angle in which the bot turns towards or away from the wall.

Evolution

During initial development, the main wall following methods employs a hard-coded angle of correction. This more heavily relied on the odometry, and the bot eventually became negatively affected by odometry drift. As a result, we replaced this by differential correction to mitigate error accumulation.

Testing and Implementation

Throughout the testing and implementation process, the solution required optimization of constant values such as the speed and distances before turning. For the majority of the testing process, we focused on finding values that worked best. Furthermore, to improve differential correction performance, a limit on the angle of correction was imposed to prevent irrational sonar values from influencing the correction angle.

Technical Results

In the final testing phase, the DE2Bot consistently completed 6 legs every run while rarely crossing the yellow or colliding with the wall. However, during the demonstration, the robot averaged 4.5 legs per run, frequently crossed yellow and even had to be reset.

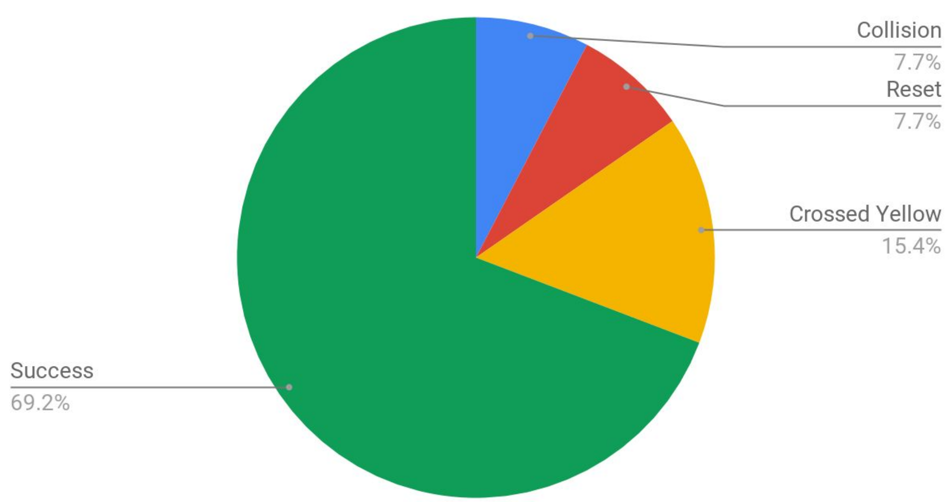

Figure 4. Performance results from the 13 legs completed on demo day.