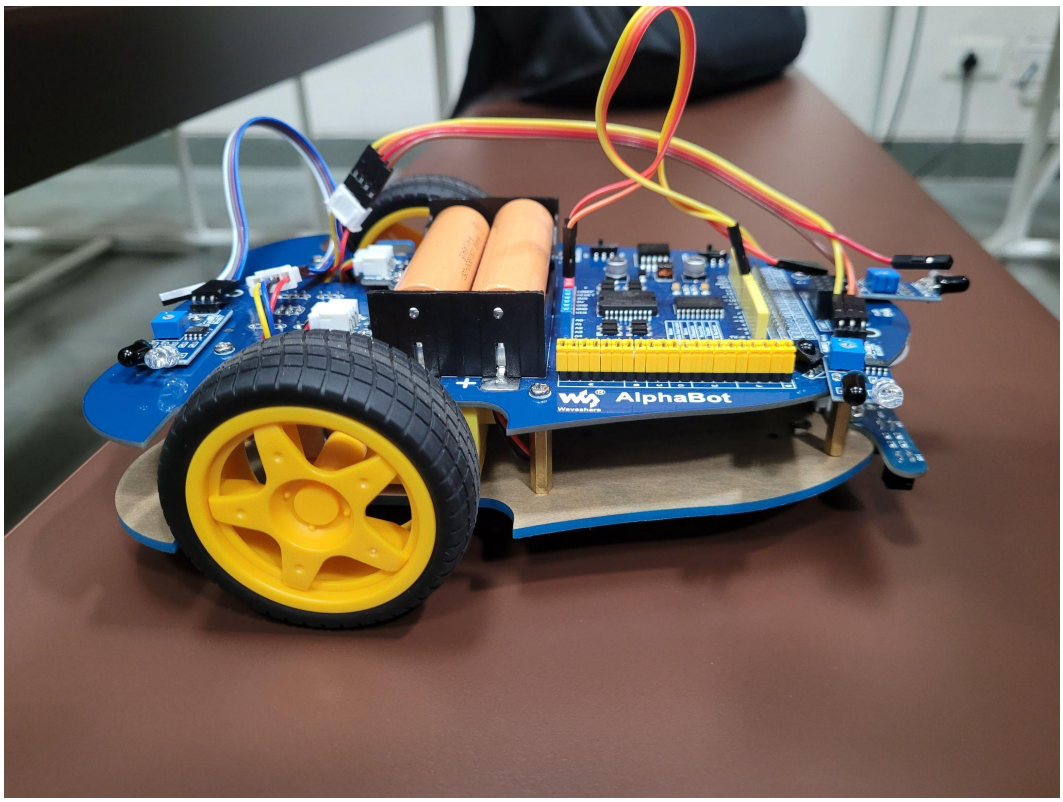
CS684: EMBEDDED SYSTEMS

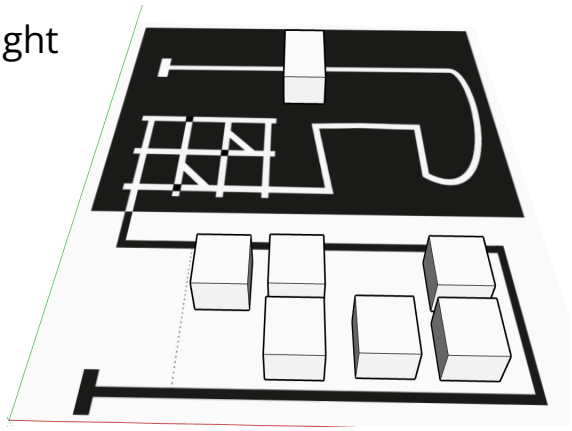
THE SUPERVISORS

Anmol Saraf, Tanmay Joshi, Shaun Zacharia



Overview

- To traverse the arena shown on the right
- Requirements:
  - Avoid the obstacle
  - The transition from white lines on a black background to black lines on white background
  - Park in a free space available, or go till the end

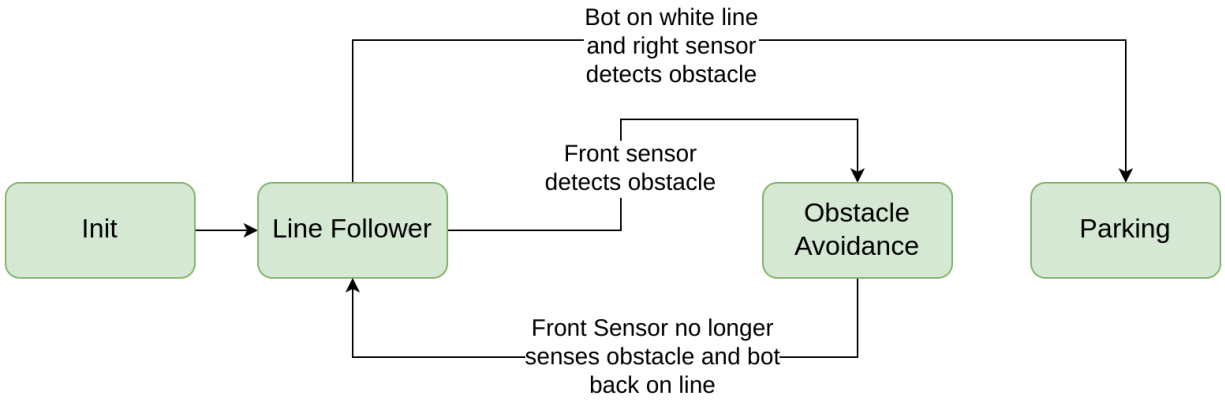


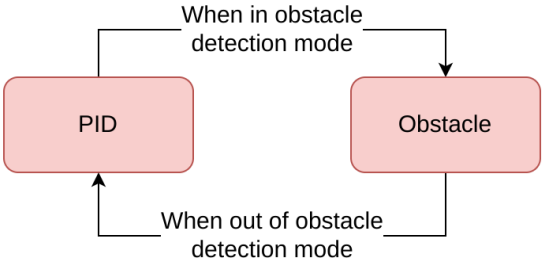
Algorithm

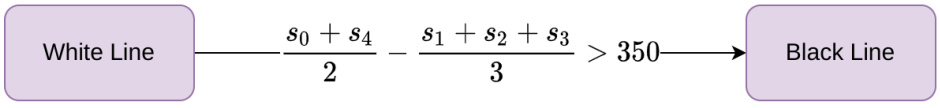
The hardware used currently for Obstacle Avoidance and Line Following are-

- IR Proximity Sensor (3): {0, 1}
- DC Motors (2)
- White Line Sensor Array (5): 0-1023

Heptagon was used for designing the bot's reactive kernel. We have used mealy machine automata for modelling the states and sub-states the bot can be in. After extracting the C code from Heptagon, we have used further modifications and integration into the C code to make it a feasible supervisor for the bot. The bot was programmed using Arduino IDE.

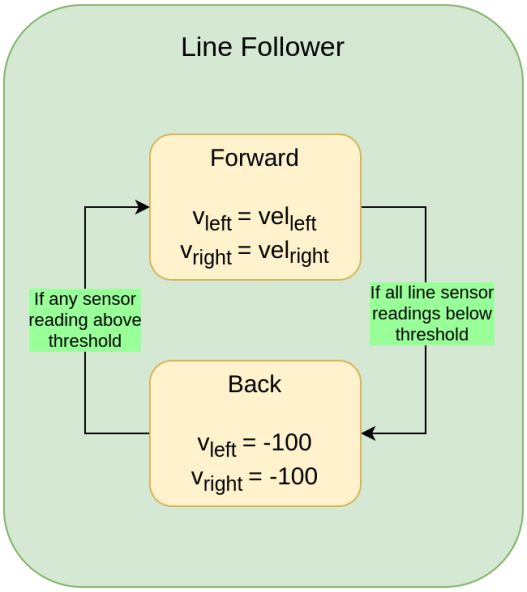






Line Follower and PID

- Basic design: two PID controllers
- Input to PID controller #1:
  - the middle tracker sensor value
  - constant
- Input to PID controller #2: weighted difference of tracker sensor values (Left-Right)
- One wheel gets the sum of the two PID outputs
- The other wheel gets the difference between the two PID outputs



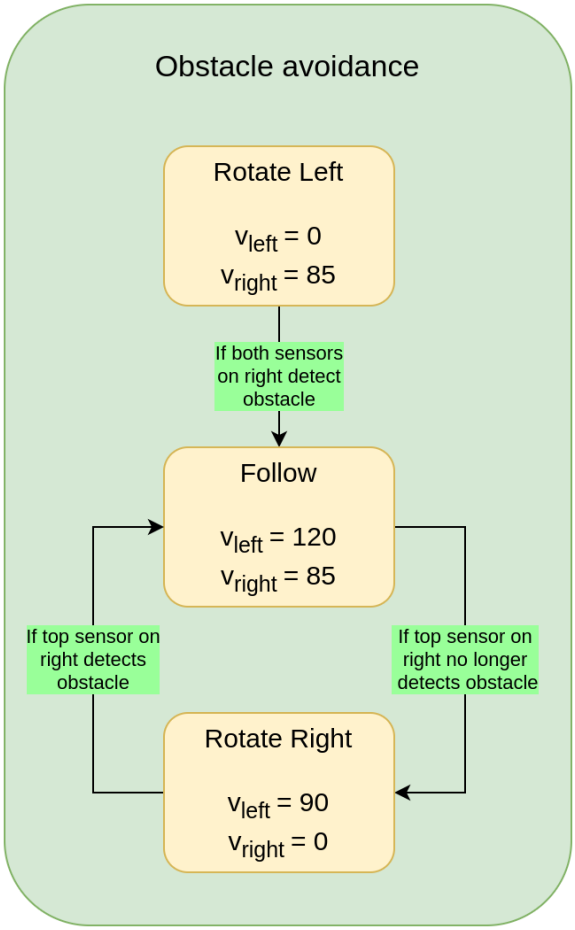
PID

$$\text{pid}_{\text{forward}} = 80$$
$$\text{pid}_{\text{left}} = 0 \rightarrow k_{p2} \cdot (s_0 - s_4) + k_{p3} \cdot (s_1 - s_3) + k_{d2} \cdot (s_0 - pre(s_0) - s_4 + pre(s_4)) + k_{d3} \cdot (s_1 - pre(s_1) - s_3 + pre(s_3))$$

$$\text{vel}_{\text{left}} = max(min(\text{pid}_{\text{forward}} - \text{pid}_{\text{left}}, 150), 55)$$
$$\text{vel}_{\text{right}} = max(min(\text{pid}_{\text{forward}} + \text{pid}_{\text{left}}, 150), 55)$$

Obstacle Avoidance

- While doing PID-based line following, if the IR sensor in the front detects an obstacle, the bot shifts to an obstacle-following mode.
- The bot turns to the left till both the sensors on the side detect the obstacle.
- Then the bot keeps turning right and going straight whenever the top sensor on the right does not detect an obstacle.
- It does this till it reaches the white line again and goes back to Line the following mode.



White to Black Transition and Parking

White to Blackline transition

- The bot transitions to a black line from the white line black background region when the mean of the extreme TR sensor values is greater than the mean of the middle three values by a certain threshold.
- After a lot of tinkering and tuning, this threshold was set to 350 for appropriate functioning

White Line $\xrightarrow{\frac{s_0 + s_4}{2} - \frac{s_1 + s_2 + s_3}{3} > 350}$ Black Line

Parking

- When in the line the following mode, the bot shifts to the parking mode if it is on a white line and also detects an obstacle to the right
- It will continue moving on the line till it no longer detects an obstacle to the right, and makes a right turn and stops, thereby parking itself

Demonstration

The video of the bot performing the Valet Parking tasks is depicted in the video below:

https://www.youtube.com/watch?v=hOUl5oRU8VU

Challenges

- As the battery discharged, the hardcoded values did not work effectively, so we had to change velocities for our implementation to work correctly continuously.
- Proximity sensors needed to be tuned multiple times to give consistent output overruns.
- The black part needed to be spray painted so our sensors could differentiate between white and black.
- Velocity values after PID had to be capped between 55 and 150 (can range from 0-255) so the bot did not unnecessarily stop or go too fast
- White and black regions needed different thresholds to get similar performance, found out through trial and error

Acknowledgements

We thank the professors Kavu Arya & Paritosh K Pandya for teaching us the required skills to complete the project. We would also like to thank the e-Yantra lab for providing us with multiple bots and helping us with the technical issues. We also want to thank fellow batchmates who helped with the fixing of the arena and brainstorming ideas.