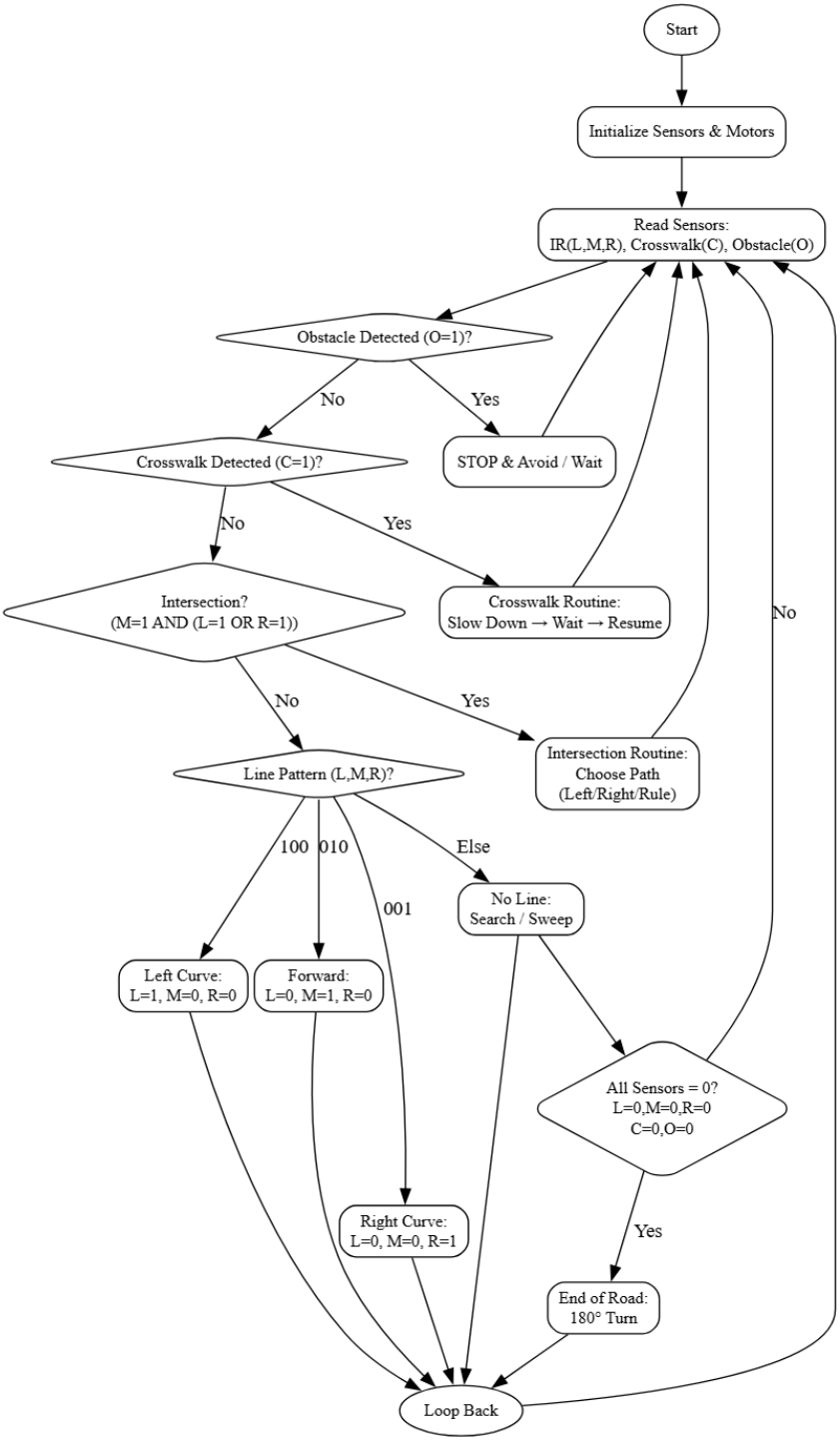
Theory Assignment
CSCI 455
Bao, Lorn, Kyle

1) Sketch your controlling mechanism for the autonomous robot using a state machine model in as much detail as possible.



Sensors and placement

- L, M, R, C are infrared sensors mounted in a row at the front of the hex bug:
    - L (Left), M (Middle), R (Right) are directly above the road so they see the black line.
    - C (Crosswalk) is shifted slightly to the side, outside the road, so we can add rule to detect the crosswalk
- O is an ultrasonic sensor mounted in front of the robot to detect obstacles.

Each sensor gives a binary reading every control cycle:

- 1 = line (for infrared sensors) or object in range (for ultrasonic)
- 0 = no line / no object

So at any time the controller sees a pattern like (L,M,R,C,O).

Explanation

How does the robot follow the predefined path?

The black line on the road defines the path. The (L,M,R) pattern (100 / 010 / 001) directly controls the steering: left, straight, or right, keeping the robot on the line.

How does it navigate to the destination?

At intersections, the “Intersection Routine” chooses which branch to take (e.g. we will apply the rule here). By following this rule at every intersection, the robot can reach a target destination along the predefined network of lines.

How does it avoid obstacles?

The ultrasonic sensor O continuously checks for objects in front. When O==1, the controller switches to STOP & Avoid / Wait, overriding line following until the path is clear.

How does it handle sensor inaccuracy?

- It uses multiple infrared sensors across the width, so decisions are based on patterns, not a single sensor.
- A search/sweep state handles temporary loss of the line instead of immediately assuming the road ended.
- In implementation, we can also require the same reading for several cycles before changing state (simple debouncing) to ignore short spikes/noise.

How does the system know where the robot is?

The robot’s “location” is expressed in terms of road features detected by the sensors:

- When (L,M,R) shows one of the line patterns → it is on the road segment.
- C==1 → it is at a crosswalk.
- M==1 and (L or R) ==1 → it is at an intersection.
- All sensors 0 → it is beyond the road (end of track).

Together with the current state of the machine (normal follow, intersection routine, crosswalk, etc.), this tells us what kind of place on the path the robot is currently on.

2) What are the possible challenges that you may need to solve?

- Our state machine is still a high level representation of the problem we are trying to solve. In reality the sensor readings are not a simple “obstacle detected”. Instead they will be noisy and hard to interpret signals that we will have to translate into discrete events.
- We have imperfect control and positioning for our robot. Smoothly following a line requires the ability to make continuous small adjustments while in motion. The spider is quite clunky. It moves slowly, turns too quickly, and slides all over the place.
- Handling uncertainty robustly over long periods of time. For the 10 inch course we had in the initial lab it was fine to have a line following procedure that was rough. However, when the course takes several minutes there is no room for gradual deviations.
- Actually mounting all of these sensors on the robot in a way that allows us to sense the environment will be difficult. We need sensor input from outside the lane, in front of the robot, as well as on and outside the guiding lines. This exceeds what we can reach by just mounting sensors on the bare robot.
- Doing the right thing at the right time. Each of the requirements for the assignment are simple in isolation but hard to do in aggregate, especially autonomously. Even little things like seeing the wrong sensor input in the middle of another routine, and executing an action because of that. For some situations the readings for a sensor might even be the same (such as 3 and 4 way intersections) and require additional action to differentiate.

3) What are possible solutions to solve these problems?

We have the low level techniques to solve each of these issues. With a combination of smoothing, taking into account the design of the sensor, and careful control, each of the tasks are achievable.

- To get cogent output from each of our sensors we can debounce our infrared sensors and apply a rolling average to the ultrasonic sensor.
- To mount the sensors we can build extensions onto the robot so that they do not get tangled in the legs and can reach out to the edge of the lane.
- For the most part positioning can be described by either being on the road or off the road. Unless we go off the road on purpose this should be sufficient.

- To tell situations apart we can't just look at our sensor readings at a moment in time. Instead we have to keep track of the surrounding context. Are we in a turn? Did we just take action?
- Avoiding executing multiple situations will require careful specification and enumeration of the possible sensor inputs as well as making sure events are blocking and exclusive. Most of this will be done in code. This will mostly just require careful thinking and ironing out of edge cases. The state machine will help with this but will not be the end of it.
- To deal with the clunky control of the robot we can make sure that the robot is never turning and moving at the same time. During our initial tests we found that this led to robot changing positions quickly and unpredictably. However a solution to this would be preferable.