# Adaptive Maze Learning Robot

Keaton Kelberlau
Faculty sponsor: Dr. Nathan Backman

Buena Vista University

## Abstract

This robot is a line-following robot that uses Infrared sensors along with two servos motors to implement a PID control algorithm. This control algorithm allows my robot to follow the line and adjust the servos as needed and will search through the maze until it finds the end designated by a black square. Once the robot has searched the maze and found the end, it will then back track through this mapped maze and fill out a list of turns. After it has a list of turns to get back to the start , it will go back through the maze from the end to the beginning with this optimal path found.

## Objectives of Robot

1. The robot must follow the line and does this using PID algorithm
2. Once robot meets an intersection or junction must properly identify this junction
3. After robot has moved through maze and found end must go to the beginning of maze
4. It will do this by filling out a tree of junctions, so it knows where it is in the maze
5. It can then back track through the tree and make a list of turns back to the beginning

## Hardware

- 5 channel Infrared sensor
- 2 continuous servo
- Teensy 4.1 micro-controller
- Rocker switch
- Variable bucc converter for voltage conversion
- 3d printed chassis
- 9.6-volt battery

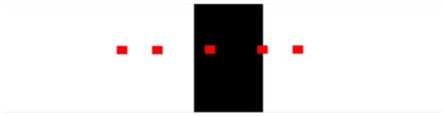

*Figure 1. IR sensors over the line*

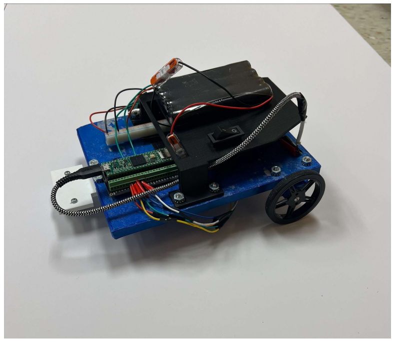

*Figure 2. This is the maze following robot that Uses 5 IR sensors*

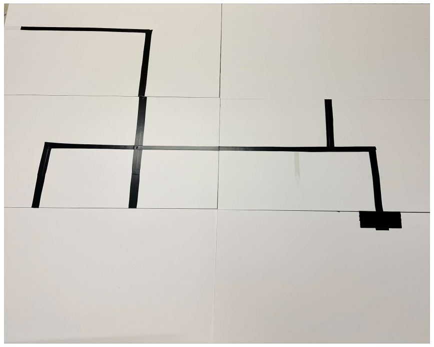

*Figure 3. One configuration of the maze that the Robot can complete*

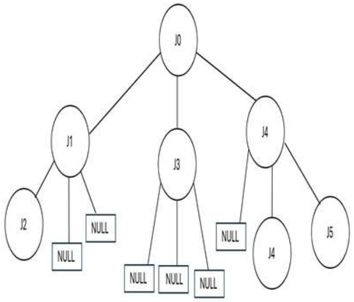

*Figure 4. A tree data structure to map the maze*

## PID line following

To use PID control algorithm you must give the algorithm a setpoint it will try and maintain. The setpoint in my project would be having the middle line sensor stay over the center of the line. Once it has this set point it will adjust the system with the math below. The K values are the gain values which adjust each output to produce a meaningful value for my motor's speed.

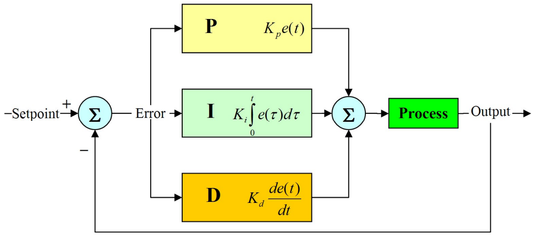

*Figure 5. Diagram of PID process*

## Moving through the maze

Since the maze has no cycles, we can represent it as a tree and use DFS to find the end of the maze. This allows us to define a junction that will have a parent a right, left, and forward junction that it points to. This means there is only six possible types of junctions to identify which are represented below. As the robot follows the path it checks if there is any turns. Once it detects a potential turn it then goes into junction detection mode which can identify one of the six junctions and make the corresponding node for my tree. Having a properly made tree allows my robot to know where it is in the maze and navigate through it based off this mapping

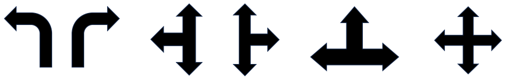

*Figure 6. Types of possible junctions*