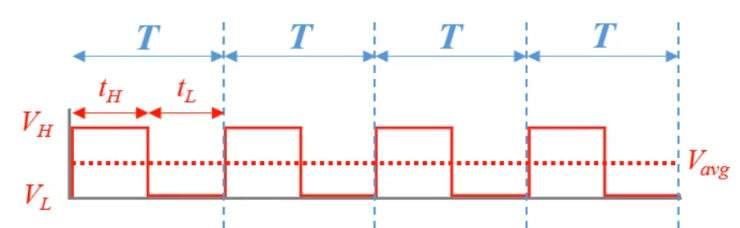
1. **Line detect and control the wheel speed/direction**

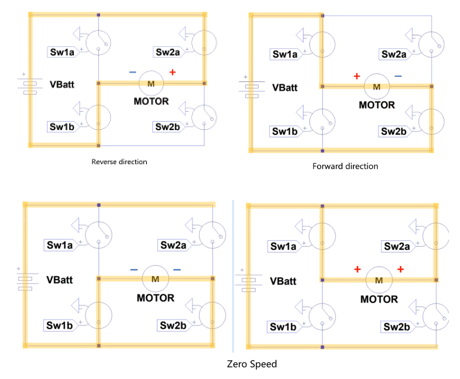
The infrared sensor has been chosen to detect the line. From the lab result, the voltage output will decrease when the line is closing the sensor. So the position of the line can be calculated by using sensors’ output.

For the wheel speed control, PWM (Pulse-width modulation) is used. PWM can be set by changing the duty cycle, as shown in Fig.1. Duty cycle D is the rate between the time of high state and the switching cycle T. And the final speed is the , which is the average speed in one cycle.



***Figure 1***

H-Bridge can change wheel direction. H-Bridge has four switches. These switches can change the direction of the motor which is shown in Fig.2. H-Bridge can change the motor direction to forward or reverse by changing the poles of the circuit. Alternatively, let the motor to zero speed by disconnect the motor.



***Figure 2***

To guide the Buggy back to the line and let the buggy turn, the feedback control system is used to detect the line error by a series of sensors. Change the single wheel speed to offset the error, and guide the Buggy to make a turn or back to the line.

1. **Proportional vs bang-bang controllers**

Bang-bang controller means that the Buggy will correct itself by a fixed configure when it deviates the expected measurement. Which is shown in Fig 3.a. Fig 3.b is an improved bang-bang controller with a “dead zone”. The motor will not affect when the measurement is in the dead zone. The dead zone can lessen the frequency of the speed changing. Bang-bang controller will cause the underdamped or unstable condition, which is shown in Fig.4 because this controller will switches abruptly between two states.

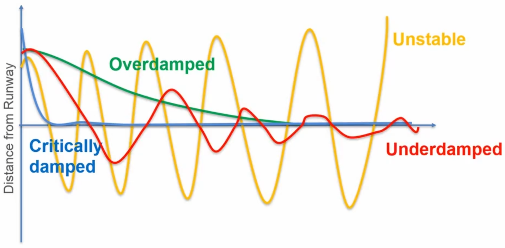
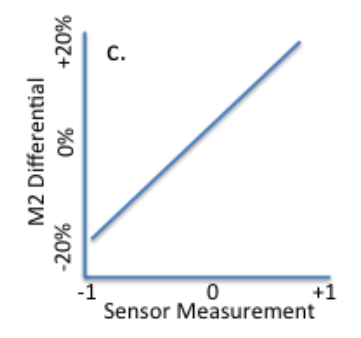
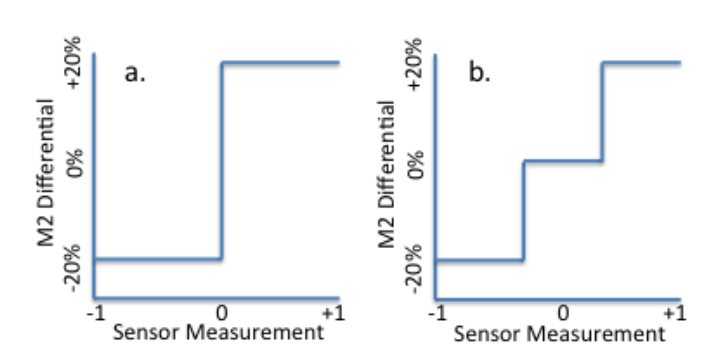
~~~~

Figure 3.a Figure 3.b Figure 3.c

Proportional controller means that the Buggy will correct itself by a proportional measurement, as shown in Fig.3.c. The proportional controller will occur underdamped when only P action has been activated. Overdamped will occurred when the constant of the controller has been set incorrectly. Critically damped will occurred when P, I, D actions are all used, and it will be an ideal damped figure.

***Figure 4***

Pros and Cons of the bang-bang and proportional controller have been listed in table 1 below.

***Table 1***

|  |  |  |
| --- | --- | --- |
|  | Pros | Cons |
| Bang-bang | Easy to achieve  Fast to execute | Need a long time to settle down  Always shaking with a given angle |
| Proportional | Correct the deviation dynamically  Fast to settle down | Need more compute  Three constants need to set |

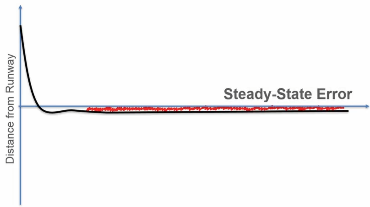
1. **Things to control**
2. Wheel speed: To get the optimal speed in a straight line and correct the direction when the Buggy is making a turn. Calculating wheel speed by using the PID algorithm, change wheel speed by setting PWM.
3. Wheel direction: Changed by H-bridge, can set the wheel direction to forward, reverse direction or zero speed.
4. Switching frequency: how long does the duty ratio is, which is related to the sampling frequency of the sensor.
5. **Algorithm choice**

The proportional controller feedback system has been chosen. The difference between two motors speed will proportional to how far away from the line the Buggy was.

For the proportional controller, the difference between the sensor measurement and the set-up value(which wishes to be 0) called error. Then an equation showed the relationship between the speed difference and the error.

Where u(t) is the manipulated variable, e is error, t is time and is termed the proportional gain. cannot be too low because it may need a long time to correct it. Also, it cannot be too high because it may over-active and will make massive speed changes even there is a small deviation.

There are some control problems need to be solved when the Buggy is running. For example, the loss of forces brings about the deviation of the entire Buggy. The forces loss ineluctable, like friction and motor loss. These losses will make the Buggy stay slightly to one side of the line, which is termed as a steady-state error, as shown in Fig.5. Integration part needs to be added to the controller to avoid the steady-state error.



***Figure 5***

Integration action can add up steady-state error and plus them into the algorithm. Integration has been used to integrate the error with time. The function of the PI controller has been written below.

Where is the integral gain. Where e(t) is the error in t seconds.

Also, when the Buggy needs steering, its wheel speed may decrease to have a more significant angle to steering. PI controller can handle with this, while the Buggy may rollover or motor overheat when the voltage of the motor changes drastically without a damper. So the derivation part needs to be added. Derivative action can provide damping to the controller by split the change of error into small pieces, which can make the Buggy moves more smoothly. The equation of the PID controller has been written below.

Where is the derivative gain.

There are different combinations for PID controllers. Pros and Cons of these controllers are listed in table 2 below.

***Table 2***

|  |  |  |
| --- | --- | --- |
|  | Pros | Cons |
| P | Easy to execute | Need a long time to settle down  May have a steady-state error |
| PD | Response fast | May have a steady-state error |
| PI | No steady-state error | Need a long time to settle down |
| PID | No steady-state error  Stable  Response fast |  |

In order to get a precise and stable system, the proportional controller can only solve a part of control problems. Add the integral and derivative part to the controller are essential. PID control feedback system can have precise and stable control. Also, it can smooth the Buggy and correct the steady-state error.

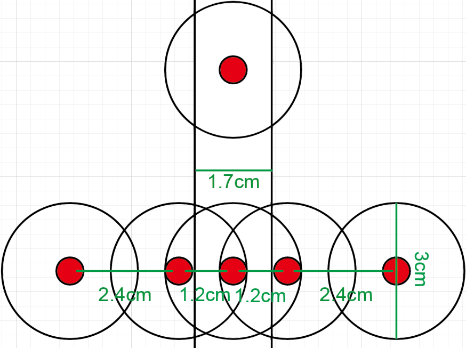
1. **Factors affecting the algorithm**

When sensors are designing and setting up, some factors may affect the algorithm. From the lab result, the gap between the sensor and the white line will directly influence the voltage output of the sensor. When the different gap of the sensor has been chosen, three constants which are set in the algorithm (P, I, D) should be reset.

When the Buggy is testing and debugging, it is essential to show that which sensor is read by the system. Groupmates can easily find which sensor is read by the system if there is a LED show the operational status of each sensor. It can let debugging quickly and more uncomplicated to improve the algorithm.

In the control system, sensors should be read separately in order to get each sensors’ voltage output correctly. So each sensor should occupy an interface on the motor drive board.

1. **Sensor implementation**

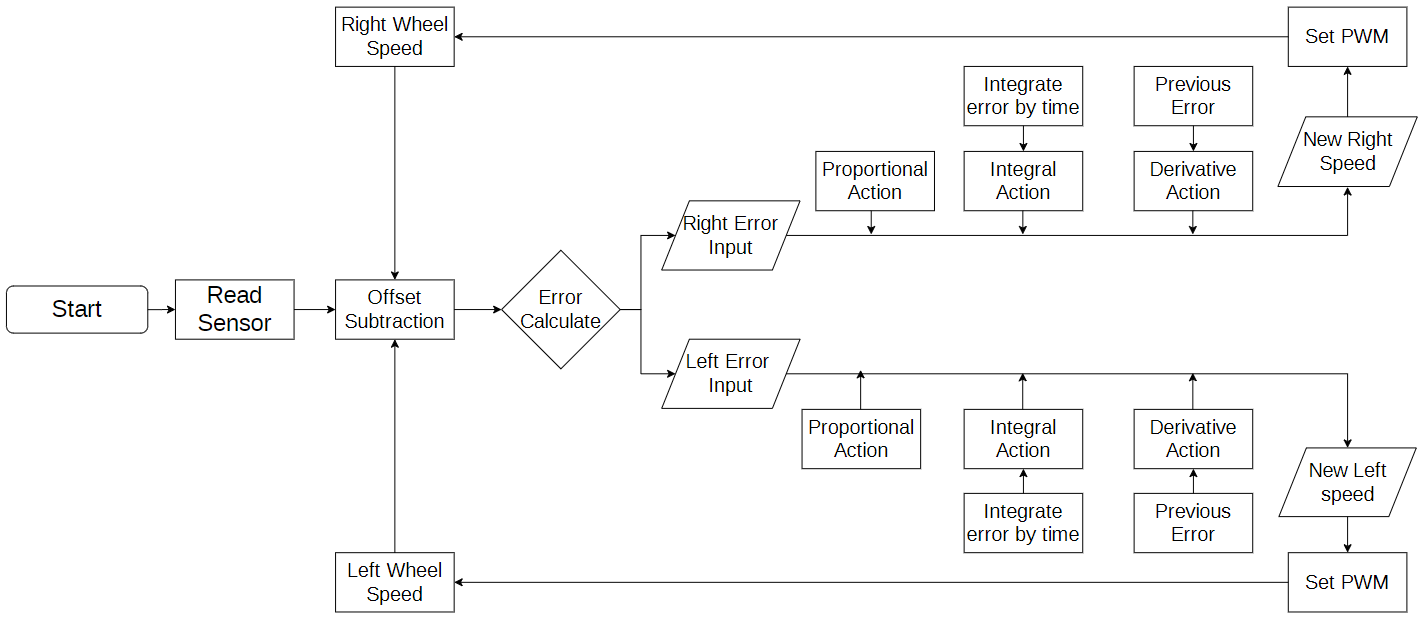
For sensor implementation part, TCRT5000 sensor has been used in line detecting. TCRT5000 contains an infrared emitter and receiver. Six TCRT5000 sensors all in analogue type are planned to implement in the Buggy, as shown in Fig.6. Five sensors are placed in one line, while the gap between each sensor is slightly different. For the three sensors located in the centre of the arrangement, they are planned to detect the slight shaking of the Buggy. So the gap will be 1.2 cm. In this case, the effective detecting area (a circle with a radius of 1.5cm) of the three sensors will overlap. When the Buggy slightly deviates from the white line, it will be detected by two or three sensors and back to the microcontroller. The microcontroller can accurately determine the error and correct the deviation by changing the PWM.

The two sensors on both sides play a role when there is a vast deviation between the Buggy and the white line, like turning or unpredictable accident. The gap between these two side sensors is 2.4 cm, while there is still a part of the effective detecting range that overlaps. It ensures that the control system will continue working when a vast deviation occurred.

***Figure 6***

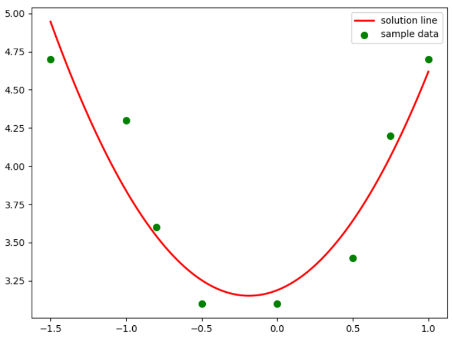
Another sensor will be placed in the top centre. Its function is to assist the centre sensors in correcting the set-point. Also, it can detect the line breaks before the centre sensors to assist the control system to operate.

The line sensor board will have some interfaces to connect the motor drive board. VDD and GND as power supply and the ground will be connected to CN4 connector of the motor drive board. Six sensors need to be read separately. So they are planned to be connected to A0-A5 interface of the CN8 connecter on the motor drive board.

1. **Plan for algorithm implementation**

***Figure 7***

The flowchart of the PID control algorithm has been shown in Fig.7. It shows the process of the algorithm.

First, the distance between sensors and white line from sensors’ voltage output should be calculated. The distance can be termed as ‘error value ’. From the sensor implementation part, five sensors are planned to allocated to a series with a specific gap between each other. A python-based algorithm is found in CSDN website[1], which using the fitting algorithm to fit a binomial equation by several points. This fitting algorithm is planned to use to calculate the distance between the sensor and the white line. There only lab result published by the EEE department, so only eight values can be used in this algorithm. The fitting figure is shown in figure 8, and the equation has shown below.

Where is the distance between the sensor and white line, is the voltage output of the sensor. By calculating , two results should be outputted. Determine which result should be used by judging the position of the sensor during polling. More data for sensors can be got in the future experiment, so a more precise equation can be calculated by fitting algorithm.

Then, the calculated error value should be put into the PID algorithm below.

Error value should be negative for the left side sensors. So for the left wheel speed, there should be a negative sign before the P action. D action is a damper, so its sign should be opposite to the P action. Mention that three gain values, , and should be tried in the future test and give the optimal solution.

In the implementation, the integral and derivate function may slow down the response time for the microcontroller. So the integration part can be changed to

Where is the sampling error at kth sampling instant. And the derivative part can be changed to

New speed values should be calculated by the algorithm, which are and . These speed values show the difference between the actual speed and the new speed. Use speed values to set the PWM of the motor. Then, the left and right wheel speed change to the expected value. Finally, use two new wheel speed as inputs to start a new loop.

1. **Solutions of line breaks and direct sunlight**

During the testing and the final race, there are several problems that need to be considered—for example, line breaks and direct sunlight. To cope with line breaks, a timer is tried to be added into the program. The timer is ordered to countdown the time that the Buggy needs to pass through the break. From the slides in ESP week 1, line breaks are up to 6mm. So the time can be calculated by 6mm divided by the speed of the Buggy. Then, using wait command and ISR function to let the Buggy keep running when a line break occurred.

To cope with the direct sunlight, it can be implemented by software. Turn off the infrared emitter to detect the environment infrared intensity by the receiver. Then, use the measured value as the offset. When the algorithm is reading the output value of the sensor, subtract the offset to get the correct value.