B37VB - Edinburgh - Robotics - Group 4 - Final Report

Contact details

Cailean Scott, H00436718, cs2091@hw.ac.uk

Lewis Gill, H00436489, lg2050@hw.ac.uk

Revision History

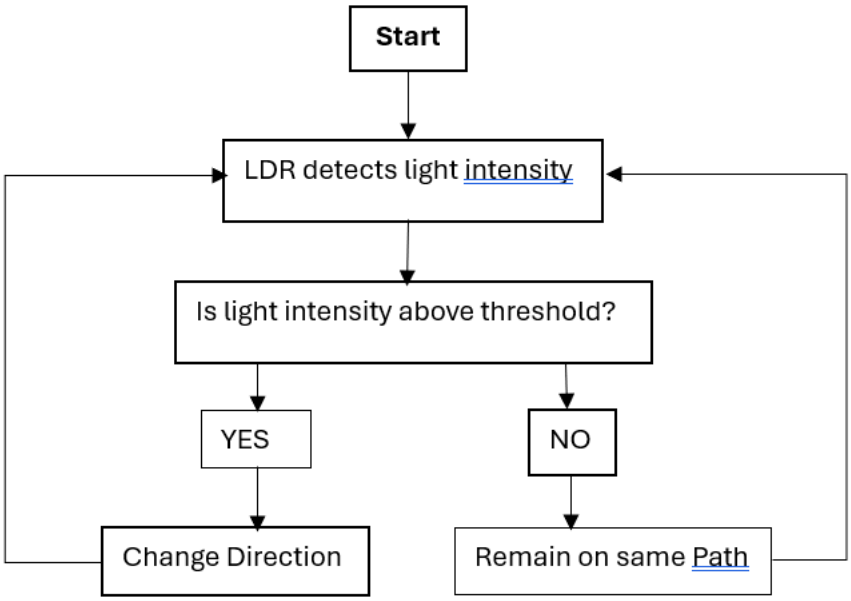
Date	Authors	Version #	Notes
8th of April , 2024	Lewis Gill & Cailean Scott	Version 1	Initial layout
10th of April, 2024	Lewis Gill & Cailean Scott	Version 2	Adding main bodies
14th of April, 2024	Lewis Gill & Cailean Scott	Version 3	Final changes

Contents

Introduction

Closed loop control is a form of system control that includes feedback. One example of a component used in closed loop control systems are light dependant resistors (LDRs) which are used to detect the intensity of light falling on it and, can be used in various practical circumstances for example what we did, using an LDR to control a moving buggy. The output from the LDR is fed into a microcontroller unit (MCU) which compares the actual light intensity as detected by the LDR to a pre-defined threshold value. Based on the comparison made by the MCU, the motors adjust the buggy's movement. For instance, if the light intensity increases, the controller commands the buggy to change direction. The buggy's movement is continuously adjusted based on the feedback received from the LDR. If the light intensity changes, the LDR detects it, the controller adjusts the buggy's movement accordingly, and this process repeats, creating a closed-loop feedback system.

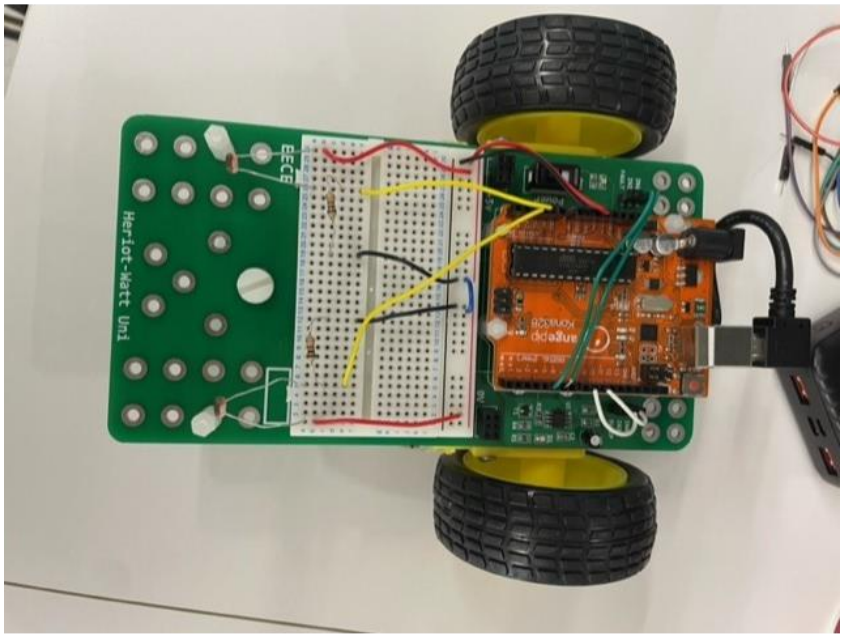
Figure 1 - Flowchart of using an LDR to control a buggy's direction



Experimental Method

Our task was to operate a closed loop control system to control two drive motors in order to manage the movement of our buggy. The control system consisted of the hardware running on the ATMega328P MCU (microcontroller unit) and the H-bridge integrated circuit is used to provide power to the two motors. We decided to go for a design which includes two LDR's, one on each side of the buggy. This meant that when we put our light source up to one the motor on that side would slow down, hence meaning the buggy would follow our light source. We started by wiring our microcontroller to the breadboard attached on the buggy however we saw that even with the provided code our buggy did not operate as intended, it would not react to our light source and was going in circles as one motor was operating faster then the other. Upon inspection of our wiring, we noticed that we had two wires going into the ground meaning the circuit was shorted. We then did our adjustments to resolve this issue and the motor did then operate as intended, it went in a straight line and would follow our light source when present. We noticed that despite it working well it seemed to be quite slow in responding to the light source however we changed this in the code by reducing the response time. We saw an immediate difference in that the buggy would change direction far more promptly in response to our light source.

Figure 2 - Buggy with completed wiring



Results

Table 1

Speed no.#	MaxSpeed	SlowSpeed
PWM value	250	100

Table 2

Time no.#	Time 1	Time 2	Time 3
Time speed is set for (ms)	100	250	500

Difference in motors for turning = 40 PWM

Analysis

Table 1 shows the PWM values of the slow speed and max speed which are choses as the buggy has range of speed from 0-255, so both values had to be within this range and choosing these round numbers made life easier in the long run. Table 2 shows the amount of time the buggy goes at one speed for on each wheel. For example, using time 1, every 100ms the speed of the wheels will adjust to follow the light source therefore either increasing or decreasing its PWM Value on each wheel to change direction. We used three different times (100ms, 250ms and 500ms) to see the potential change in sensitivity of the buggy, by adjusting this time we discovered that for the less time the more sensitive the buggy becomes. The difference in motor speed was set to a PWM value of 40, this value is important as it turns the buggy left and right. We found that at this value the buggy didn't turn to sharply that it was hard to steer but also would turn enough that it could turn at a 90-degree angle in a small distance travelled which makes the buggy more than agile enough for use.

Conclusion

In conclusion, during this experiment we used a light tracking algorithm to steer a buggy, during this we had to make decisions on speeds and times to figure out an ideal speed and sensitivity of steering. We discovered that using smaller time speed is set for the more sensitive the steering became but as we used a slower moving buggy, we didn't need our steering to be very sensitive as it became more difficult to steer and less smooth whilst turning therefore, we finalised our time at 500ms (time 3 on table 2). During this part of the experiment, we learnt the importance of micro adjustments as these changes in the time severely changed the way the buggy drove and was controlled. During the wiring of the buggy, we discovered the importance of being accurate as we put a wire that should have connected to the motor into a ground which cause a shortage, this caused us to almost rewire our buggy completely. We also learnt it's important to keep wires neat and colour coded this meant that we could clearly see which wires went where for example black wires for our ground, and having wires the correct length made the wiring easier to understand and easier to physically wire.