一張含有 電子工程, 電子產品, 電氣線路, 機器 的圖片

自動產生的描述[Introduction]  
In this project, I partnered with another student from the ELEC1100 LA1 session and build a line-tracing robot car (RC) for the project demonstration. Which the car was required to trace the line of the department’s name ECE and make decision to turn the car automatically, when necessary, having us to figure out the solution by ourselves using Arduino code and flowcharts.

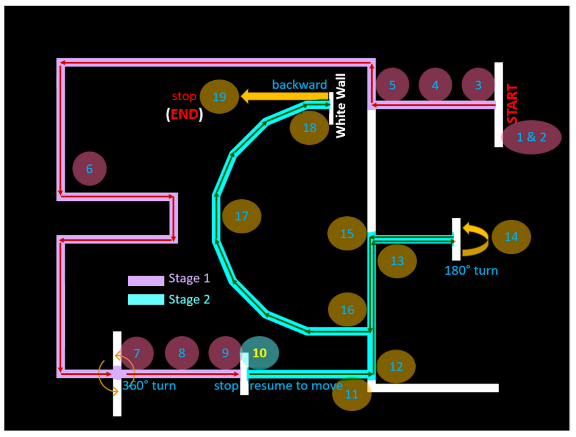
[LogicDesign]  
To trace the line successfully, we need to understand what kind of difficulties that we may encounter during the debug session. In here, based on the map shown in the Project Guide\_2024s, there are multiple potential difficulties that we may face including multiple T-junctions, incorrect decision making that causes a fail run and excessive inertia that causes the RC to go off line.

Figure 1: Point awarding scheme map from Project Guide\_2024s.

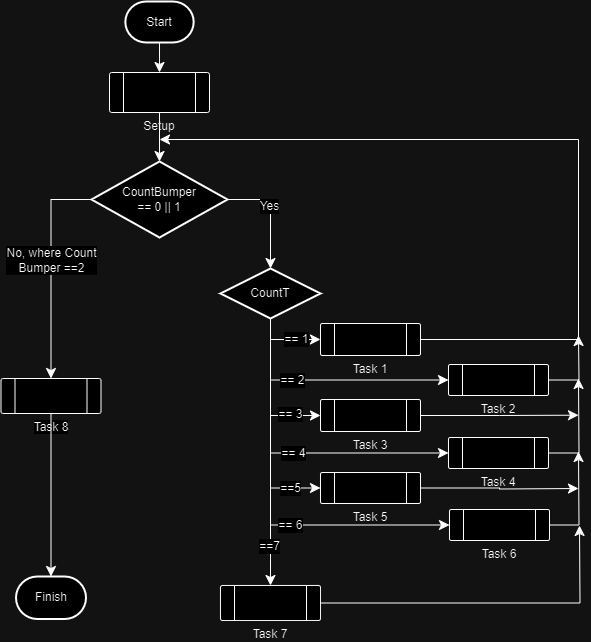
To mitigate this problem, we divided the main program into 2 parts, first based on the CountBumper value, which stands for the count of the triggering of the bumper sensor. If CountBumper is either 0 or 1, it will further compare the value of CountT, which stands for Count of T-Junction, and execute different tasks corresponding to values that stored inside CountT. If CountBumper value is 2, that means the RC car is at position/point 18 referring to Figure 1, the RC car will execute Task 8 and terminate the program for finishing.

Figure 2: Flowchart of the main program

Before we further explain the program in detail, there are a few procedures that needed to be explained as it associated with the corresponding tasks listed in Fig.2 .

一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述The first sub-program is forward(), this procedure subdue the problem of different speed of the motor, where the left motor needs to spin faster to mitigate the differences, HIGH is written to both digitalWrite of the direction pins of both motor for the RC to move forward, as shown in the code in the box.

Figure 3: code template for forward().

一張含有 文字, 螢幕擷取畫面, 字型, 圖表 的圖片

自動產生的描述The next 3 procedure includes turn\_right(), turn\_left() and self\_turn(), those 3 procedures allows the RC to turn in specific direction with corresponding speed, after turning to a certain angle using delay(delay\_time), here I called it delay duration, which the delay\_time is a referencing parameter from the upper level procedure, the RC will stop itself sharply in order to remove the excessive inertia. This can prevent the RC to go off the lane and give it spaces to adjust the direction and swap to trace\_line() properly. Figure 4 shows how these turning procedure works.

Figure 4: flowchart of turning procedures.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Table 1: Input-Output Table for the trace\_line function | | | | | | |
| L\_Sensor | R\_Sensor | L\_PWM | R\_PWM | L\_DIR | R\_DIR | Description |
| 1 | 1 | 215 | 195 | HIGH | HIGH | Move forward |
| 1 | 0 | 180 | 96 | HIGH | LOW | Faced too right, adjust to R |
| 0 | 1 | 96 | 180 | LOW | HIGH | Faced too left, adjust to L |
| 0 | 0 | 215 | 195 | HIGH | HIGH | Move forward |

The last sub-program is trace\_line(), this procedure allows the RC to trace the line properly and drive straight while checking the differences for the sensor value differences. The RC will move in a constant speed of forward with global speed (195) with motor speed difference mitigation. For turning, the speed of motor will depends on the input and adjust the trace\_spd (120) by MCU. The Input-Output table (with calculated PWM value), and the general description is shown above.

一張含有 螢幕擷取畫面, 文字, 圖表, 設計 的圖片

自動產生的描述Given the procedures listed above, we divided the procedure into multiple parts, based on the value of CountBumper and countT, we associate them with various of tasks. In figure 5, we will define those sensors correspondingly with different names and the dimension/position of the IR sensor that settled on the IR sensor mount.

<Task\_explaination>  
For each completion of tasks, the countT value will accumulate itself by adding 1 onto it, so that the RC can continue to the next task, and Figure 6 shows the logic flowchart of each tasks.

In task 1, the robot will begin at the starting point listed in figure 1, the robot will initiate by spinning the motor. After initiating, the RC will stop at the start point. From there, the RC will check whether the bumper sensor is triggered. If yes, the RC will start the trial using forward() for 350 ms. 350 ms is used here to prevent the RC turns immediately after the trial start. While the RC had not reach the T-junction yet, it will keep tracing the line based on the value listed on Table 1. After the RC reached the T-junction by detecting if both left and right far sensor is triggered, it will turn right and move forward.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table 2: Input-Output Table for Task 2 | | | | |
| LF\_Sensor | L\_Sensor | R\_Sensor | RF\_Sensor | Procedure to be executed (from P.2) |
| 1 | 1 | 1 | 1 | Self turn for 360° (1000 ms delay) |
| 1 | 1 | 0 | 0 | Turn left |
| 0 | 0 | 1 | 1 | Turn right |
| 0 | 0 | 0 | 0 | Trace line |

In task 2, the RC is now at the first L-junction after point 5 from figure 1. It will start ckecking the value of different sensors, as shown in Table 2. For each L-junction turning, it will have a specific global variable L\_junction to count how many L-junction had passed for later references. For each turning, the RC car will sharp stop itself before continuing tracing the line, as the inertia generated from turning may causes the RC to go off the lane. When it reached the T-junction after passing 7 L-junctions, it will turn itself for 360°+ to complete the task. Excessive degree and trace\_line() for 150 ms is turned and used to mitigate the inconsistent position of center of mass after turning.

In task 3, the RC had turned 360° and reached position 9 from figure 1, it will check whether it reached the T-junction again. If so, it will sharp stop the RC in order to remove the inertia for 1 second before continuing, similar to what happens after triggering the bumper sensor at task 1. Otherwise, it will keep tracing the line until the RC reached the T-junction again.

In task 4, the RC is now at position 12 from figure 1. From there, the RC it will check whether it reached the T-junction again. If so, it will turn left and move forward for 100 ms. Otherwise, it will continue to trace line until it reached the T-junction again.

In task 5, the RC is now at position 16 from figure 1. From there, the RC will ignore the case of turning left, where only both left sensors were triggered. However, it will check whether both right sensors were triggered. If so, it will turn right and move forward to position 14, where there would be an T-junction awaits. When both far sensors were triggered, it will turn itself for 180° and trace line for 250 ms. If both cases requirement were not meeted, it will continue to trace line until it meets any case requirements.

In task 6, the RC is now back at position 15 from figure 1. From there, it will continue to check if both far sensors were detected. If so, the RC will turn left. Otherwise, it will continue to trace the lane until it reached the T-junction.

In task 7, the RC is now at position 16 from figure 1. Here, the RC will check if both right and right far sensor were triggered. If so, the RC will turn right and adjust the global variable in\_C into true, which indicates the RC will now start going through the letter C in figure 1. If the RC is now in the letter C, it will continue to trace the lane until the bumper sensor got triggered. When the bumper sensor got triggered, it will accumulate countT and countBumper by 1 to finish task 7.

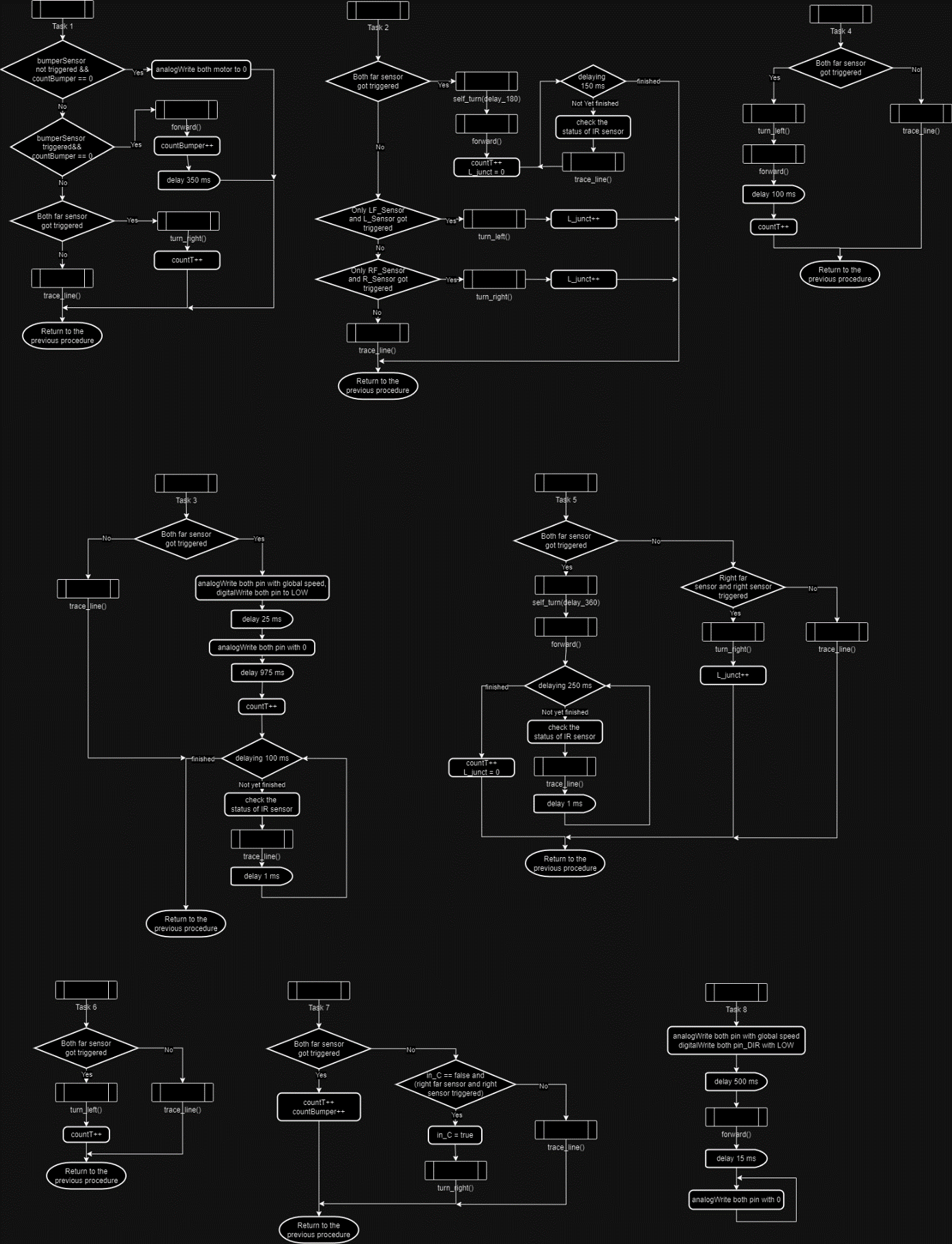
In task 8, the RC had finished tracing the letter C and is now at position 18. From there, it will move backward to the finish position for 500 ms. After that, the RC will be stopped by writing 0 PWM to both motor PWMs, finishing the entire trial.

Figure 6: flowchart of different tasks.

[Results\_&\_Conclusion]  
Looking backwards, I had discovered that my skill sets were improved from the ELEC1100 courses, which I had found that there are multiple parts that I had done quite well, as well as poorly.

For the parts that I did well, I found that the hardware construction of the RC was done quite well, that we had made the RC to drive in nearly a perfect straight line by setting both middle sensors from figure 5 close to each other. However, regarding the places that I had done poorly, even though I had come up multiple concepts for the coding, I could not adapt the real life situation, where offsets were included as a factor while completing the trial. Therefore, my project groupmate and them coded properly based on my concepts.

But of course, the project trial may went wrong sometimes. The most notable one for me is that I had tuned up the speed of RC far too much that it hit a wall and broke the 6 hole bars, but we came up a solution by placing the IR mount as close as the body of the car, which hides the 6 hole bars completely. Not only that solves that issue, but also strengthen the crash durability of the RC car as well as getting the center of mass close to the motor, which helps us to fine tune the turning speed properly.

Even though we got 18 marks from the trial, which is considered a nearly perfect run, there are still a few parts that I had considered them a mistake. First of all, if I could improve the source code, further simplify it into reusable codes and try to make use of those inertia, it may saves our time debugging it and trying the trials. Besides, the most notable mistake should be accidentally adjusted the position of IR sensors, that mistake caused every trial from final demo to be a fail run. If I could start that over, I would rather not to have my itchy hands closes to those sensors, and just restart the trial right away, which potentially can get ourselves to a true perfect run.

Overall, I see this project to be successful, not only both of us had come up a proper solution to complete the tasks given from the demo for ELEC1100 course, but also having our mindset, skillset and friendship to be refined. I had learnt that good communication in a project leads to a great height, not only personal, but also for everyone that had participated in the project. If there’s some more chances for us to participate in, I would had myself signed up to refine myself even further.