ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

HỌC KÌ 241

ĐỀ TÀI: XE ĐIỆN TỰ HÀNH

STT	MSSV	HỌ VÀ TÊN
1	2114105	Nguyễn Hoàng Mỹ
2	2111117	Đỗ Thanh Hà
3	2220024	Hà Hải Thiệu

CHƯƠNG 1: GIỚI THIỆU

1.1 Tổng quan

◆ Xe điện tự hành là một lĩnh vực quan trọng trong các ứng dụng về tự động hóa và robot. Đặc biệt, xe điện dò line là một trong những hệ thống đơn giản nhưng có nhiều tiềm năng nghiên cứu, thường được dùng trong các môi trường công nghiệp và học thuật để phát triển kỹ năng lập trình, cảm biến và điều khiển tự động. Xe dò line được ứng dụng trong các hệ thống sản xuất, kho bãi thông minh, giúp vận chuyển hàng hóa tự động mà không cần can thiệp của con người.

1.1.1 Tìm hiểu về cảm biến

➢ Có 3 loại cảm biến thường dùng trong các robot dò line: Camera, cảm biến quang điện trở và cảm biến hồng ngoại.

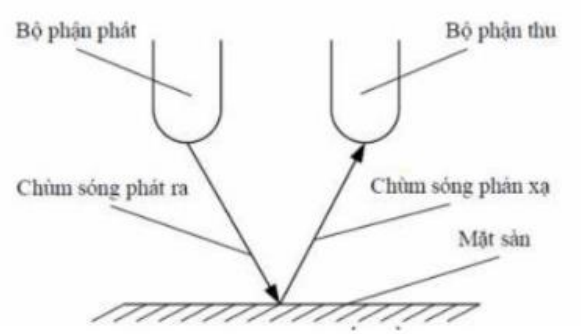
• Camera

Camera dùng để ghi lại hình ảnh của đường line, sau đó xử lý đưa ra thông tin sai lệch vị trí tương đối của line so với xe.

Phương pháp này cho độ chính xác cao, tuy nhiên do tốc độ xử lý ảnh cần nhiều thời gian, làm hạn chế tốc độ của xe, nên không phù hợp với xe đua.

• Cảm biến quang điện trở và cảm biến hồng ngoại

Cảm biến quang điện trở có điện trở thay đổi khi ánh sáng thay đổi. Trong bóng tối giá trị điện trở cao, nhưng khi được chiếu sáng giá trị điện trở giảm mạnh.

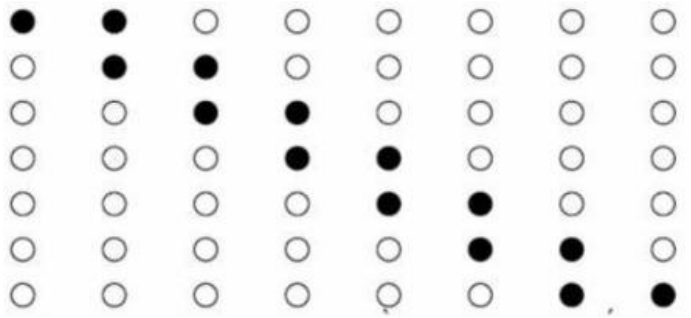


Hai phương pháp sử dụng cảm biến quang điện trở và cảm biến hồng ngoại có nguyên tắc hoạt động tương tự nhau, gồm 1 nguồn phát ánh sáng phản xạ xuống đất và 1 nguồn thu ánh sáng phản xạ từ đó xử lí tín hiệu và đưa ra vị trí của xe so với line. Tuy nhiên cảm biến hồng ngoại được ứng dụng nhiều hơn vì thời gian đáp ứng nhanh hơn.

❖ Cách xử lý tín hiệu của cảm biến:

◆ Sử dụng giải thuật so sánh

Sử dụng mạch lấy ngưỡng hoặc giải thuật lọc ngưỡng bằng lập trình để chuyển tín hiệu điện áp đọc từ cảm biến về thành mức cao hoặc mức thấp, từ đó suy ra vị trí của xe so với đường line.



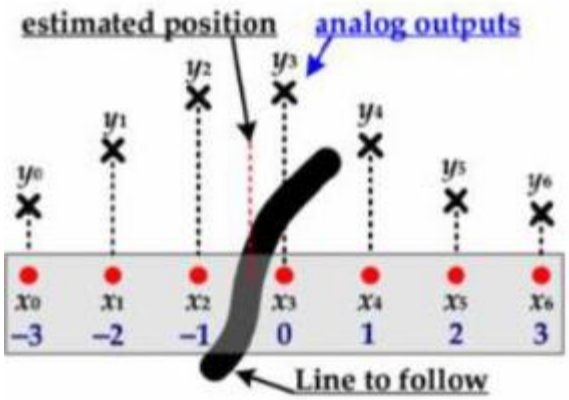
Hình 1 Mức so sánh của cảm biến ứng với các vị trí line khi xe di chuyển

◆ Sử dụng giải thuật xấp xỉ

Các giá trị đọc về qua phép xấp xỉ để tìm ra vị trí của đường line. Phương pháp này cho ra sai số phát hiện line nhỏ, có thể tới 2.6mm.

Công thức xấp xỉ:

$$x = \frac{\sum_{i=0}^{n} x_i y_i}{\sum_{i=0}^{n} y_i}$$

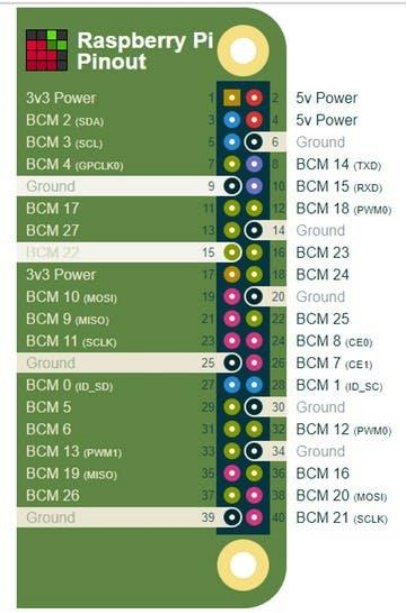


CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Giới thiệu chi tiết linh kiện

2.1.1 Raspberry Pi 4

Hình 11a: Raspberry Pi 4



Hình 2b: Sơ đồ chân Raspberry Pi 4

❖ **Thông số kỹ thuật**

Bộ xử lý (CPU): Model: Broadcom BCM2711
Kiến trúc: ARM Cortex-A72 (ARMv8) 64-bit SoC
Số lõi: 4 lõi
Tần số xung nhịp: 1.5 GHz

Bộ nhớ (RAM): 4 GB LPDDR4-3200 SDRAM

Đồ họa (GPU): Model: Broadcom VideoCore VI
Hỗ trợ: OpenGL ES 3.0, giải mã phần cứng video H.265 (4Kp60), H.264 (1080p60).

Kết nối mạng: Ethernet: Gigabit Ethernet (băng thông thực 300 Mbps).
Wi-Fi: Wi-Fi 802.11ac (băng tần 2.4 GHz và 5 GHz).
Bluetooth: Bluetooth 5.0 BLE (Low Energy).

Lưu trữ: Thẻ nhớ microSD: Sử dụng cho hệ điều hành và lưu trữ dữ liệu.

Khả năng khởi động từ USB: Có thể khởi động từ ổ SSD hoặc USB flash.

Cổng kết nối: HDMI: 2 cổng micro-HDMI, hỗ trợ đầu ra 2 màn hình với độ phân giải 4K@30Hz hoặc 1 màn hình 4K@60Hz.

USB: 2 cổng USB 3.0, 2 cổng USB 2.0.

Cổng GPIO: 40 chân GPIO hỗ trợ nhiều giao thức (I2C, SPI, UART).

Cổng Camera và Display: MIPI DSI và MIPI CSI.

Nguồn: Nguồn cấp: 5V qua cổng USB-C (khuyên dùng nguồn 5V 3A).

Chân cấp nguồn GPIO: Có thể cấp nguồn qua chân GPIO (5V).

Kích thước: Kích thước: 85.6 mm × 56.5 mm

Trọng lượng: Khoảng 46 g

Hệ điều hành: Raspberry Pi OS (trước đây là Raspbian), hỗ trợ nhiều hệ điều hành như Ubuntu, và các hệ điều hành Linux khác.

❖ **Ứng dụng:** Raspberry Pi 4 là một máy tính nhỏ gọn nhưng rất mạnh mẽ, thường được sử dụng trong nhiều dự án công nghệ khác nhau, từ học tập đến ứng dụng chuyên nghiệp. ***Dự án IoT,*** *Mini Server, Desktop PC, Xử lý hình ảnh và video, Lập trình và giáo dục, Robot và hệ thống nhúng, Chơi game Retro, ...*

2.1.2 Module điều khiển L298N

Thông số kỹ thuật:

Driver: L298N tích hợp hai mạch cầu H

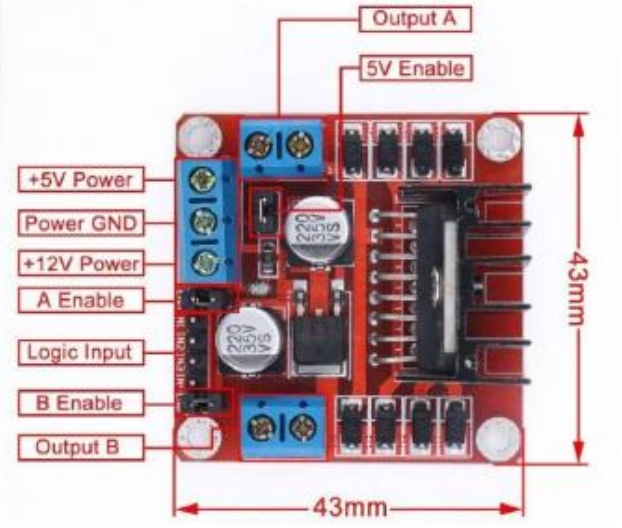
Điện áp điều khiển : +5V ~ +12 V

Dòng tối đa cho mỗi cầu H là :2A

Điện áp của tín hiệu điều khiển : +5 V ~ +7 V

Dòng của tín hiệu điều khiển : 0 ~ 36Ma

Công suất hao phí : 20W (khi nhiệt độ T = 75 °C)



Hình 3: Module điều khiển L298N

Nhiệt độ bảo quản : -25°C ~ +130

Tính năng:

IC L298 là một IC tích hợp nguyên khối gồm 2 mạch cầu H bên trong.Với điện áp làm tăng công suất nhỏ như động cơ DC loại vừa…Chức năng các chân của L298N:

• 4 chân INPUT: IN1, IN2, IN3, IN4 được nối lần lượt với các chân 5,7, 10, 12 của L298. Đây là các chân nhận tín hiệu điều khiển.

• 4 chân OUTUT: OUT1, OUT2, OUT3, OUT4 (tương ứng với các chân INPUT) được nối với các chân 2,3,13,14 của L298. Các chân này sẽ được nối với động cơ.

• Hai chân ENA và ENB dùng để điều khiển mạch cầu H trong L298.Nếu ở mức logic "1" (nối với nguồn 5V) cho phép mạch cầu H hoạt động, nếu ở mức logic "0" thì mạch cầu H không hoạt động.

• Với bài toán của mình ở trên, các bạn chỉ cần lưu ý đến cách điềukhiển chiều quay với L298:

Khi ENA = 0: Động cơ không quay với mọi đầu vào.

Khi ENA = 1:

INT1 = 1; INT2 = 0: Động cơ quay thuận.

INT1 = 0; INT2 = 1: Động cơ quay nghịch.

INT1 = INT2: Động cơ dùng ngay tức thì

2.1.3 Cảm biến hồng ngoại SCRT5000 và cảm biến siêu âm HC-SR04

◆Cảm biến siêu âm và cảm biến hồng ngoại là hai loại cảm biến phổ biến, cho phép hệ thống xe tự hành có thể nhận biết môi trường xung quanh và điều hướng trong không gian.

- **Cảm biến hồng ngoại SCRT5000**

◆ **Thông số kỹ thuật:**

Nguồn vào: 3.3V hoặc 5VDC.

Kết nối ba chân: OUT, GND và VCC:

Phạm vi phát hiện chướng ngại vật: 2cm đến 10cm

Hình 4a:Cảm biến hồng ngoại SCRT5000

Độ nhạy có thể điều chỉnh với chiết áp trên bo mạch.

Góc phát hiện: 35 độ

Thích hợp với tất cả vi điều khiển.

Kích thước: 3.1 cm x 1.5 cm

Khoảng cách so với đường đua để cảm biến hoạt động tốt là: 0,2 đến 15 mm

◆ **Tính năng:**

Cảm biến phát hiện vật cản/vật thể hồng ngoại đi kèm với chiết áp trên bo mạch để điều chỉnh độ nhạy. Vì module có đầu ra là tín hiệu kỹ thuật số nên dễ dàng kết nối với nhiều loại vi điều khiển khác nhau, ví dụ như Arduino/Genuino UNO,Mega, Leornado và Raspberry Pi. Cảm biến đi kèm một bộ phát và bộ thu hồng ngoại phía trước module. Bất cứ khi nào có vật thể chặn luồng hồng ngoại, nó sẽ phản xạ lại tia hồng ngoại và bộ thu sẽ nhận được tín hiệu qua mạch so sánh trên bo mạch. Tùy thuộc vào sự điều chỉnh, nó sẽ xuất ra logic Low tại chân đầu ra và đèn LED sẽ sáng lên để báo hiệu sự phát hiện. Cảm biến cũng có thể sử dụng chiết áp trên bo mạch để tăng độ nhạy và tăng phạm vi phát hiện. Nó tương thích với đầu vào 5V hoặc 3.3V.

◆ **Xác định khoảng cách giữa cảm biến so với đường đua**

Ta có khoảng cách giữa 2 cảm biến là $d = 3{,}5mm$

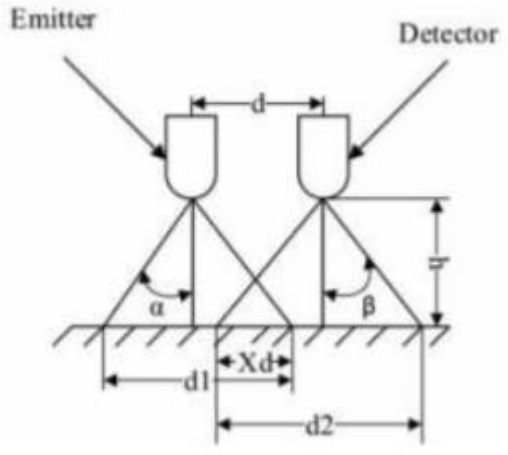
Các góc chiếu $\alpha = 16°, \beta = 30°$

Vùng giao thoa X_d

Ta có: $\frac{d_1}{2} = h\tan\alpha = h\tan 16°$

$\frac{d_2}{2} = h\tan\beta = h\tan 30°$

$\Rightarrow X_d \geq 0 \Leftrightarrow \frac{d_1 + d_2}{2} \geq d \Leftrightarrow h \geq 4{,}05mm$

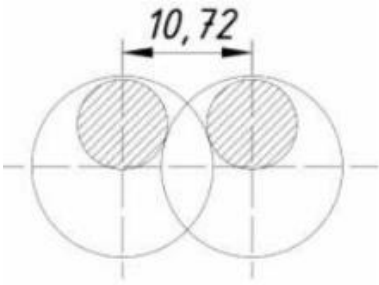


Để cảm biến hoạt động tốt, cần đảm bảo xuất hiện vùng giao thoa X_d

Vậy cần đặt cảm biến cách mặt đường đua một khoảng $h \geq 4{,}05$mm

◆ **Xác định khoảng cách giữa các cảm biến**

Ta chọn khoảng cách giữa 2 cảm biến liên tiếp sao cho không xảy ra nhiễu.



Ta nhận thấy rằng khoảng cách nhỏ nhất giữa 2 cảm biến là l để Collector 1 không nhận nhầm tia hồng ngoại của Emiter 2.

$l_{min} = 10{,}72\ mm \ \rightarrow \ Ta\ chọn\ \ l \geq 11\ mm$

➢ **Cảm biến siêu âm HC-SR04**

Hình 5: Module siêu âm HC-SR04

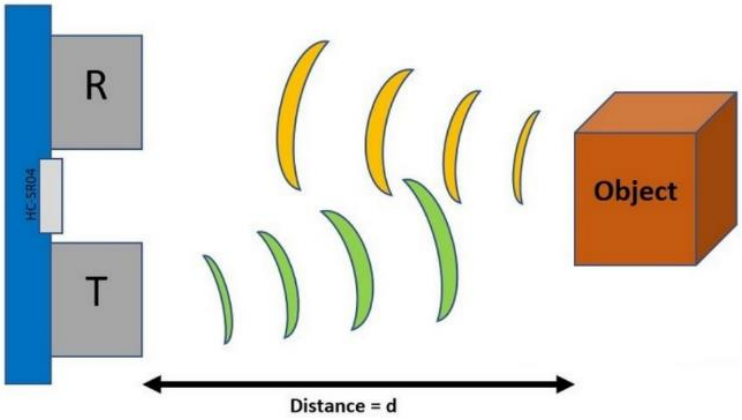
◆ **Thông số kỹ thuật:**

- Điện áp hoạt động: 5VDC
- Dòng tiêu thụ: 10~40mA
- Tín hiệu giao tiếp: TTL
- Góc quét:<15 độ
- Tần số phát sóng: 40Khz
- Chân tín hiệu: Echo, Trigger (thường dùng) và Out (ít dùng).
- Khoảng cách đo được: 2~450cm (khoảng cách xa nhất đạt được ở điều khiện lý tưởngvới không gian trống và bề mặt vật thể bằng phẳng, trong điều kiện bình thường cảm biến cho kết quả chính xác nhất ở khoảng cách <100cm).
- Sai số: 0.3cm (khoảng cách càng gần, bề mặt vật thể càng phẳng sai số càng nhỏ).
- Kích thước: 43mm x 20mm x 17mm

◆ **Tính năng và nguyên lý**

Cảm biến siêu âm HC-SR04 được sử dụng để nhận biết khoảng cách từ vật thể đến cảm biến nhờ sóng siêu âm, cảm biến có thời gian phản hồi nhanh, độ chính xác cao, phù hợp cho các ứng dụng phát hiện vật cản, đo khoảng cách bằng

sóng siêu âm. Cảm biến siêu âm HC-SR04 có hai cách sử dụng là sử dụng cặp chân Echo / Trigger hoặc chỉ sử dụng 1 chân Out để phát và nhận tín hiệu, cảm biến được sử dụng phổ biến với vô số bộ thư viện và Code mẫu với Arduino.



Hình 6 Nguyên lý hoạt động của HC-SR04

2.1.4 SG90 Động Cơ Servo 180 Độ

Thông số kỹ thuật:

Kích thước	21.5 x 11.8 x 22.7mm
Góc quay	180 độ
Trọng lượng	9g
Tốc độ không tải	0.12 giây / 60° (4.8V)
Mô-men xoắn	1.2-2.4 kg.cm (4.8V)
Nhiệt độ hoạt động	-30 → +60 ° C
Điện áp hoạt động	4.8 → 6 V dc

Hình 7 SG90 Động Cơ Servo 180 Độ

2.1.5 Động cơ DC giảm tốc V1 Dual Shaft Plastic Geared TT Motor

Thông số kỹ thuật:

Điện áp hoạt động : 3~9VDC

Dòng điện tiêu thụ: 110~140mA

Tỉ số truyền 1:48

Hình 8 Động cơ DC giảm tốc V1 Dual Shaft Plastic Geared TT Motor

125 vòng/ 1 phút tại 3VDC.

208 vòng/ 1 phút tại 5VDC.

Moment: 0.5KG.CM

Tỉ số truyền 1:120

50 vòng/ 1 phút tại 3VDC.

83 vòng/ 1 phút tại 5VDC.

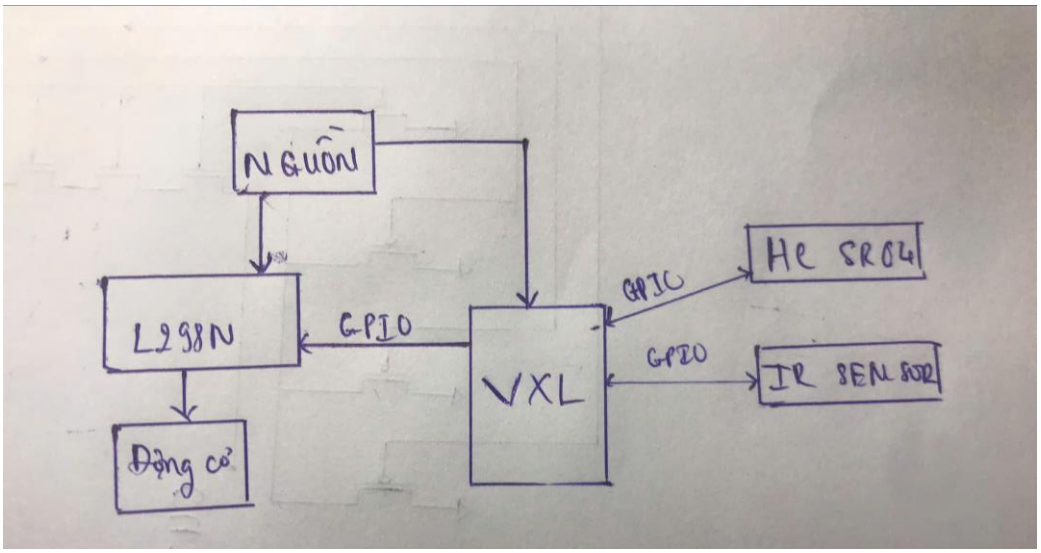
Moment: 1.0KG.CM

CHƯƠNG 3: THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

3.1 Các linh kiện được sử dụng

- Raspberry Pi 4 x1
- Module điều khiển L298N x1
- Cảm biến hồng ngoại SCRT5000 x4
- Cảm biến siêu âm HC-SR04 x1
- Động cơ DC x4
- Động cơ Servo SG90 x1
- Bộ khung xe 4 bánh x1
- Nguồn Pin 18650 (3,7V) x2
- Và các linh kiện khác

3.2 Sơ đồ khối tổng quát



• Raspberry Pi xử lý dữ liệu từ hai cảm biến và gửi tín hiệu điều khiển đến module L298N để điều chỉnh động cơ DC.

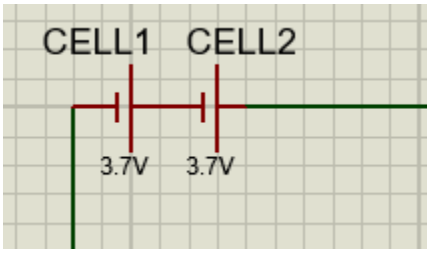
• Module L298N điều chỉnh hướng và tốc độ động cơ để đảm bảo xe chạy đúng đường và tránh vật cản.

• Cảm biến SCRT5000 sẽ liên tục theo dõi vị trí của xe so với đường kẻ và gửi tín hiệu đến Raspberry Pi.

• Cảm biến HC-SR04 đo khoảng cách với vật cản và gửi dữ liệu đến Raspberry Pi.

3.3 Nhiệm vụ và chức năng từng khối

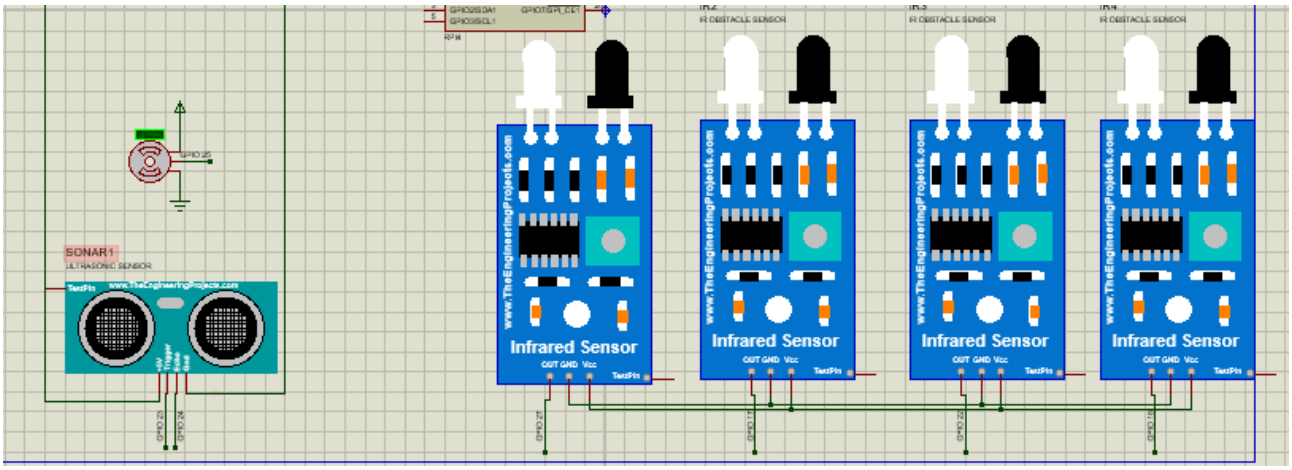
3.3.1 Khối nguồn



Hình 9 Khối nguồn mô phỏng bằng Proteus

- Nhiệm vụ: Khối Pin cung cấp nguồn điện cho toàn bộ hệ thống, bao gồm Raspberry Pi, module điều khiển L298N, cảm biến hồng ngoại SCRT5000.
- Chức năng:
    - Pin sẽ cung cấp năng lượng cần thiết để tất cả các thành phần trong hệ thống hoạt động liên tục mà không cần phải kết nối trực tiếp với nguồn điện ngoài.
    - Cấp nguồn riêng biệt cho các khối: Trong một số hệ thống, Pin có thể cấp nguồn riêng cho Raspberry Pi , module điều khiển để tránh nhiễu điện hoặc sụt áp, đảm bảo hệ thống hoạt động ổn định.

3.3.2 Khối cảm biến



Hình 10 Khối cảm biến mô phỏng bằng Proteus

Cảm biến hồng ngoại SCRT5000

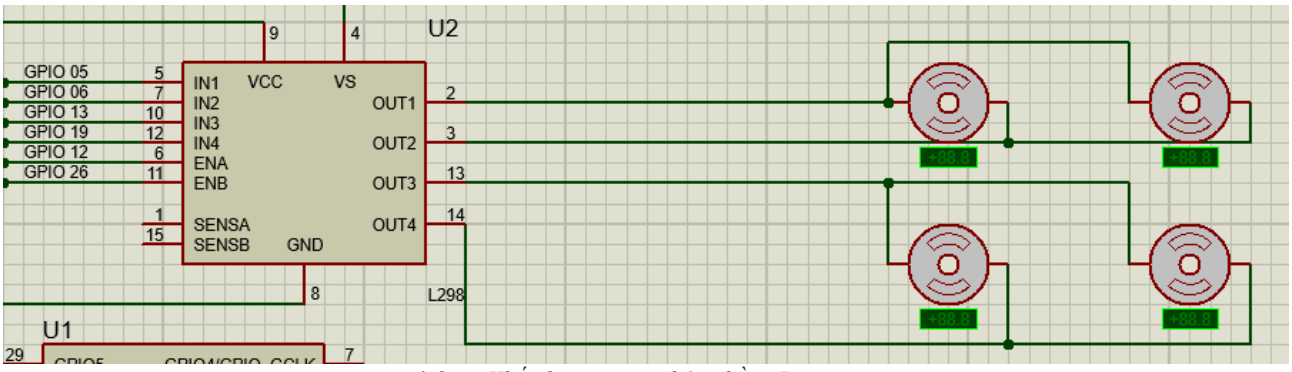
- **Nhiệm vụ**: Cảm biến hồng ngoại SCRT5000 chịu trách nhiệm phát hiện vạch kẻ đường, giúp xe tự động di chuyển theo đường kẻ.
- **Chức năng**:
    - **Phát hiện màu sắc bề mặt**: SCRT5000 có thể phân biệt giữa các vùng sáng (màu trắng) và vùng tối (màu đen). Thông thường, vạch kẻ đường là màu trắng và nền đường là màu đen.
    - **Gửi tín hiệu đến Raspberry Pi**: Khi cảm biến phát hiện sự thay đổi màu sắc (khi xe di chuyển lệch ra khỏi vạch kẻ), nó sẽ gửi tín hiệu đến Raspberry Pi để điều chỉnh lại hướng đi của xe.
    - **Dẫn đường cho xe**: Thông qua tín hiệu từ SCRT5000, Raspberry Pi sẽ điều chỉnh các động cơ để xe đi đúng theo vạch kẻ.

Cảm biến siêu âm HC-SR04

- **Nhiệm vụ**: Cảm biến HC-SR04 dùng để đo khoảng cách giữa xe và các vật cản phía trước, từ đó giúp xe tránh va chạm.
- **Chức năng**:
    - **Phát sóng siêu âm**: HC-SR04 phát ra một xung siêu âm tần số cao.

- **Đo thời gian phản xạ**: Khi sóng siêu âm va chạm với vật cản, nó sẽ phản xạ trở lại. Cảm biến đo thời gian sóng trở về để tính toán khoảng cách giữa xe và vật cản.
- **Gửi dữ liệu đến Raspberry Pi**: Dựa trên khoảng cách tính toán, cảm biến sẽ gửi tín hiệu đến Raspberry Pi. Nếu khoảng cách quá ngắn, Raspberry Pi có thể ra lệnh giảm tốc độ hoặc dừng xe để tránh va chạm.

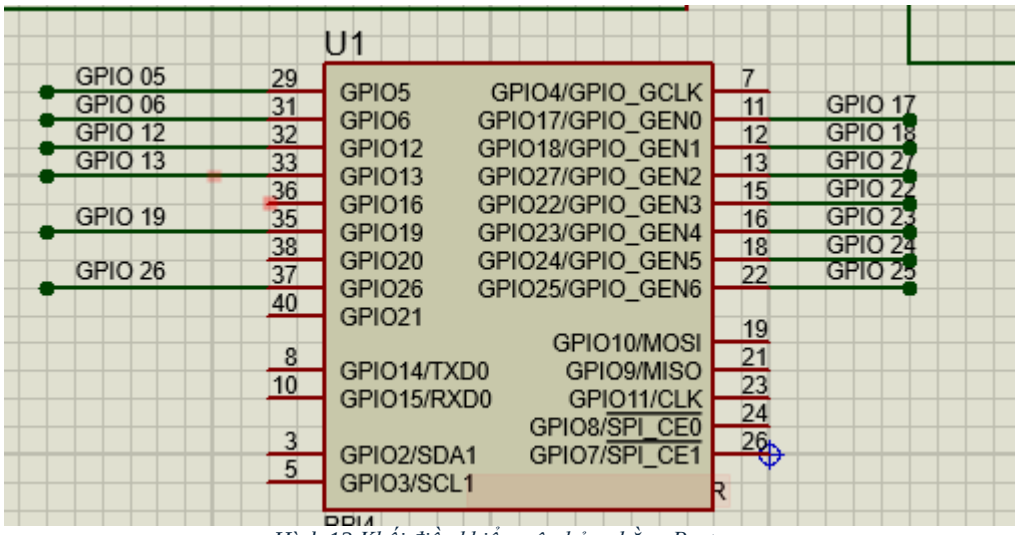
3.3.3 Khối động cơ



Hình 11 Khối động cơ mô phỏng bằng Proteus

- **Nhiệm vụ**: Module điều khiển L298N chịu trách nhiệm điều khiển động cơ DC theo tín hiệu từ Raspberry Pi.
- **Chức năng**:
  - **Điều khiển hướng quay động cơ**: L298N là một mạch cầu H, có khả năng điều khiển động cơ DC quay theo cả hai hướng (tiến/lùi) tùy theo tín hiệu từ Raspberry Pi.
  - **Điều chỉnh tốc độ động cơ**: L298N cho phép điều chỉnh tốc độ của động cơ DC bằng cách điều chế xung PWM (Pulse Width Modulation). Raspberry Pi gửi tín hiệu PWM đến L298N để tăng/giảm tốc độ của động cơ tùy theo yêu cầu.
  - **Cấp nguồn cho động cơ**: L298N cũng có chức năng cấp nguồn từ pin cho động cơ, đảm bảo động cơ hoạt động với mức điện áp phù hợp.

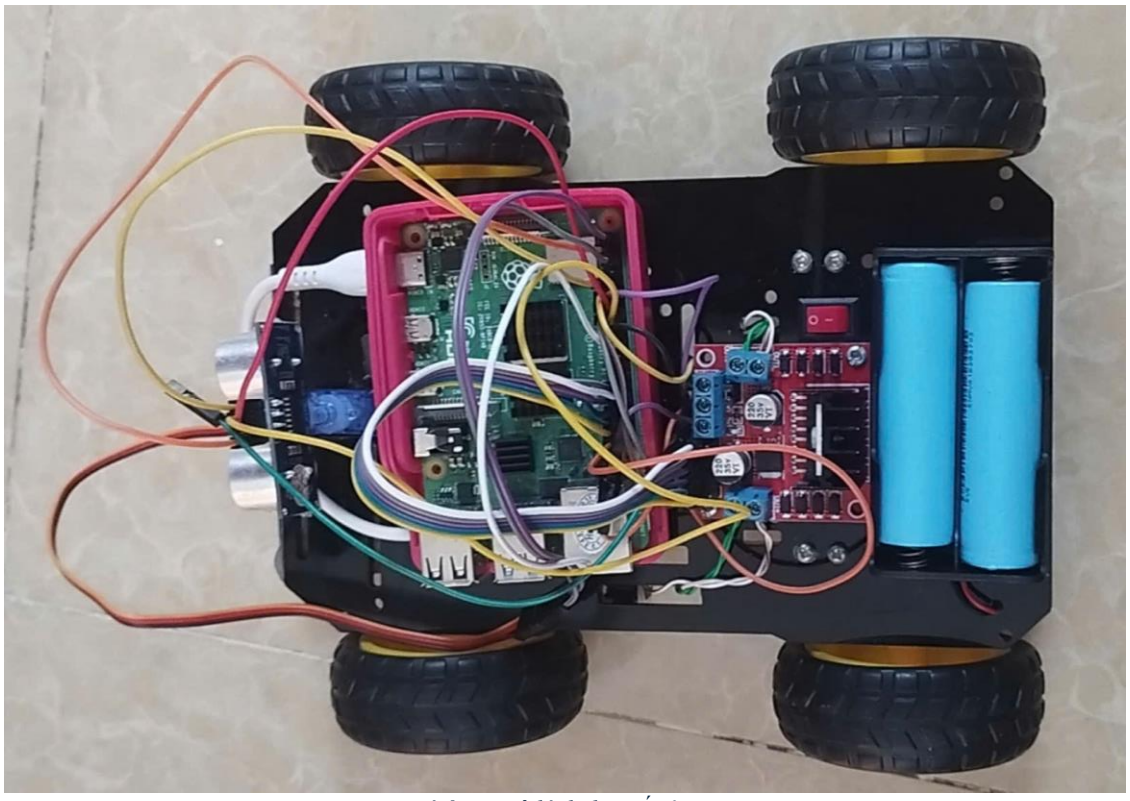
3.3.4 Khối điều khiển



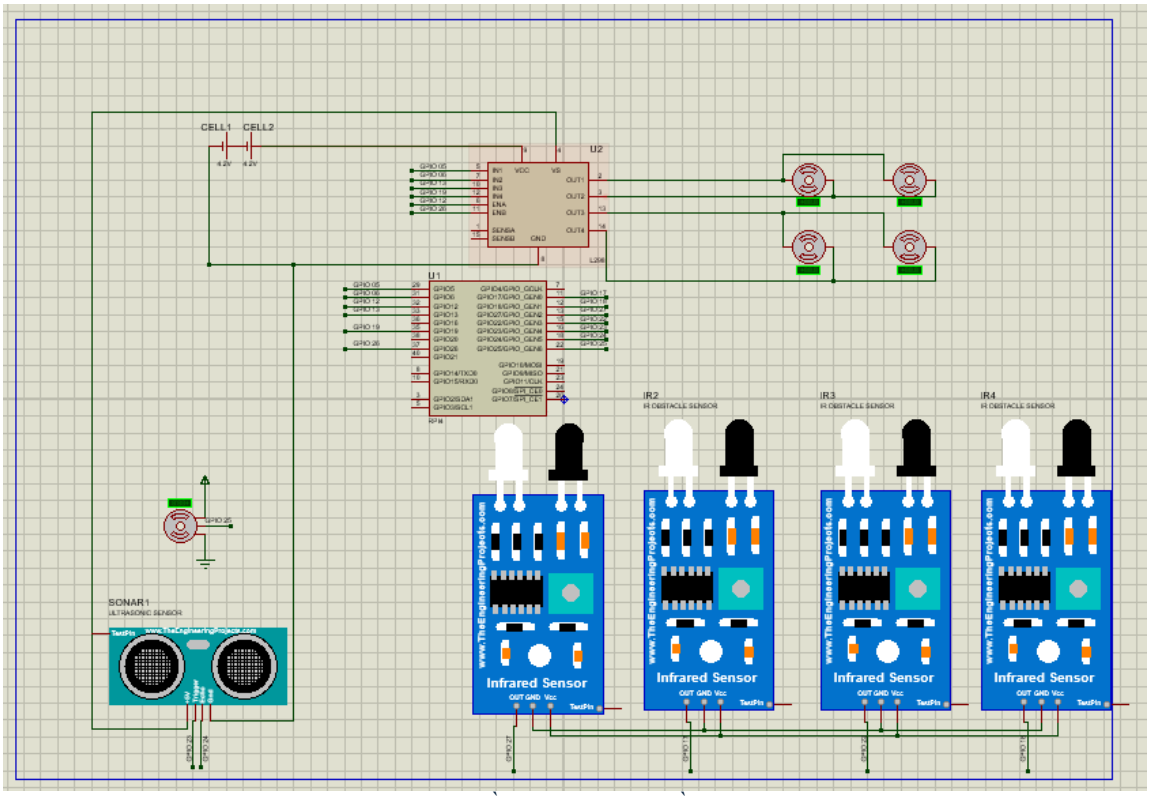
Hình 12 Khối điều khiển mô phỏng bằng Proteus

- **Nhiệm vụ**: Raspberry Pi 4 đóng vai trò như bộ điều khiển trung tâm của hệ thống. Nó nhận dữ liệu từ các cảm biến, xử lý dữ liệu và đưa ra các quyết định điều khiển cho động cơ.
- **Chức năng**:
    - **Xử lý dữ liệu cảm biến**: Raspberry Pi đọc tín hiệu từ cảm biến hồng ngoại (để nhận biết vạch kẻ đường) và cảm biến siêu âm (để nhận biết khoảng cách vật cản).
    - **Ra lệnh điều khiển động cơ**: Sau khi xử lý dữ liệu từ các cảm biến, Raspberry Pi tính toán và gửi tín hiệu điều khiển tốc độ, hướng quay của động cơ thông qua module điều khiển L298N.
    - **Giao tiếp I/O**: Raspberry Pi sử dụng các cổng GPIO để nhận dữ liệu từ cảm biến và truyền tín hiệu điều khiển đến module L298N.

3.4 Sơ đồ toàn mạch kết nối các linh kiện



Hình 13 Mô hình thực tế của xe



Hình 14 Sơ đồ mạch mô phỏng bằng Proteus

- Mô tả:

- Pin cung cấp nguồn cho Module điều khiển L298N thông qua chân VCC
- Pin cung cấp nguồn cho Raspberry Pi qua cổng micro-USB
- Module điều khiển L298N cung cấp nguồn và điều khiển Module siêu âm HC-SR04 qua chân VS
- Module điều khiển L298N điều khiển các động cơ DC qua các chân OUT1, OUT2. OUT3, OUT4.
- Raspberry Pi 4 sử dụng các cổng GPIO lần lượt 05,06,13,19,12,26 để nhận dữ liệu từ cảm biến và truyền tín hiệu điều khiển đến module L298N lần lượt kết nối với IN1, IN2, IN3, IN4,ENA, ENB.
- Raspberry Pi 4 đọc tín hiệu từ cảm biến hồng ngoại thông qua liên kết các cổng GPIO 17,18,27,22
- Raspberry Pi 4 đọc tín hiệu từ cảm biến siêu âm thông qua liên kết các cổng GPIO 23 và 24 và điều khiển động cơ quay thông qua cổng GPIO 25.

CHƯƠNG 4: THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM

4.1 Yêu cầu đặt ra

Phần mềm phải điều khiển một xe tự động sử dụng Raspberry Pi 4, có khả năng:

- **Theo dõi đường**: Sử dụng các cảm biến hồng ngoại để nhận diện đường và duy trì xe đi đúng làn.
- **Tránh vật cản**: Sử dụng cảm biến siêu âm để phát hiện vật cản trước mặt và điều chỉnh hướng đi để tránh vật cản.
- **Điều khiển động cơ**: Điều khiển các bánh xe bằng PWM để kiểm soát tốc độ và hướng di chuyển.
- **Điều khiển servo**: Xoay servo để kiểm tra khoảng cách bên trái và phải khi phát hiện vật cản.

Các yêu cầu chi tiết và kỹ thuật:

Phát hiện đường:

- Sử dụng 4 cảm biến hồng ngoại (SENSOR_1, SENSOR_2, SENSOR_3, SENSOR_4).
- Phải phát hiện đúng và nhanh các tình huống xe lệch khỏi đường và điều chỉnh lại hướng.
- Tần số đọc cảm biến hồng ngoại: **1 kHz**.

Phát hiện vật cản:

- Sử dụng cảm biến siêu âm HC-SR04 với **độ chính xác 2 mm**.
- Khoảng cách an toàn để phát hiện vật cản: **dưới 20 cm**.

Điều khiển động cơ:

- Điều khiển tốc độ động cơ bằng PWM với tần số **1 kHz**.

- Tốc độ xe có thể điều chỉnh từ **40% đến 50%** công suất tối đa tùy vào tình huống.

 Điều khiển servo:

- Góc xoay servo từ **0 đến 180 độ** với độ chính xác **±1 độ**.

4.2 Phân tích yêu cầu để thực hiện chương trình

Phân tích yêu cầu:

- Xe cần phải di chuyển tự động theo dõi đường và tránh vật cản, do đó có hai chế độ chính:
    1. **Theo dõi đường**: Sử dụng 4 cảm biến hồng ngoại để điều chỉnh hướng xe đi thẳng hoặc rẽ trái/phải.
    2. **Tránh vật cản**: Sử dụng cảm biến siêu âm để đo khoảng cách phía trước và điều khiển xe tránh chướng ngại vật.

Phương pháp thực hiện:

- **Chương trình chính** sẽ là một vòng lặp vô hạn, liên tục:
    - Kiểm tra giá trị từ cảm biến hồng ngoại để điều khiển hướng xe.
    - Phát hiện vật cản bằng cảm biến siêu âm, sau đó kiểm tra khoảng cách bên trái và phải bằng servo để tìm lối đi an toàn.
- **Hành động cụ thể**:
    - Khi phát hiện vật cản gần hơn 20 cm, xe dừng lại, xoay servo để kiểm tra lối đi an toàn bên trái và bên phải, sau đó quyết định hướng đi.
    - Khi xe lệch khỏi đường, điều chỉnh lại bằng cách tăng tốc một bánh để rẽ về đúng hướng.

4.3 Lưu đồ giải thuật tổng quát

Mô tả tổng quát:

Bắt đầu: Khởi tạo GPIO và thiết lập thông số PWM, servo, cảm biến.

Vòng lặp chính:

- Đọc cảm biến hồng ngoại và điều chỉnh hướng xe dựa vào trạng thái của các cảm biến.
- Đọc cảm biến siêu âm để phát hiện vật cản trước mặt.
- Nếu phát hiện vật cản, dừng xe và sử dụng servo để kiểm tra khoảng cách hai bên, sau đó quyết định hướng di chuyển.

Giải thích chức năng:

- **Theo dõi đường**: Điều chỉnh xe dựa vào thông tin từ các cảm biến hồng ngoại.
- **Tránh vật cản**: Dừng xe khi gặp vật cản, xoay servo kiểm tra và tránh vật cản.

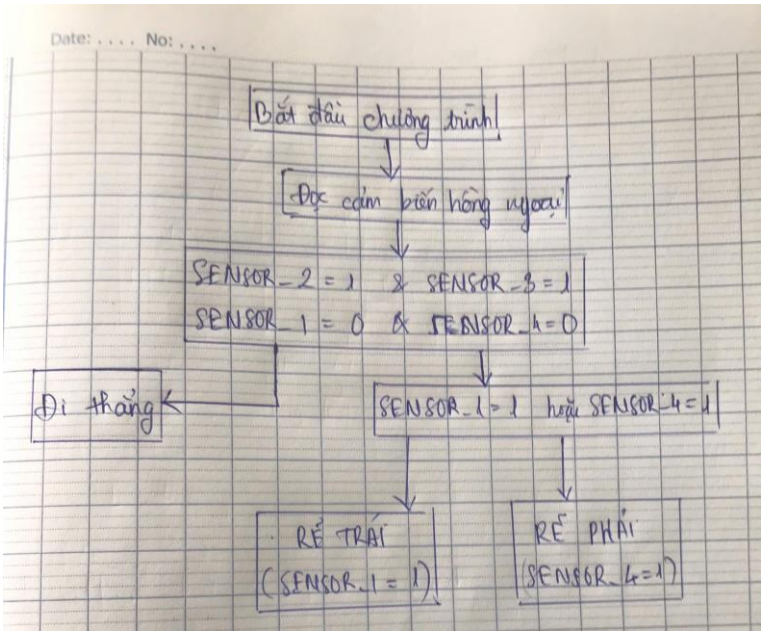
4.4 Lưu đồ giải thuật chi tiết

4.4.1 Lưu đồ giải thuật cho theo dõi đường

◆ Bắt đầu chương trình.

◆ Đọc giá trị từ 4 cảm biến hồng ngoại:

- Nếu cảm biến 2 và 3 = 1, còn 1 và 4 = 0: Xe đi thẳng.
- Nếu **SENSOR_1** = 1: Xe lệch về bên phải, rẽ trái để điều chỉnh lại.
- Nếu **SENSOR_4** = 1: Xe lệch về bên trái, rẽ phải để điều chỉnh lại.



4.4.2 Lưu đồ giải thuật cho tránh vật cản

- Phát hiện có vật cản trước mặt.
- Dừng xe.
- Xoay servo về góc trái (45 độ)

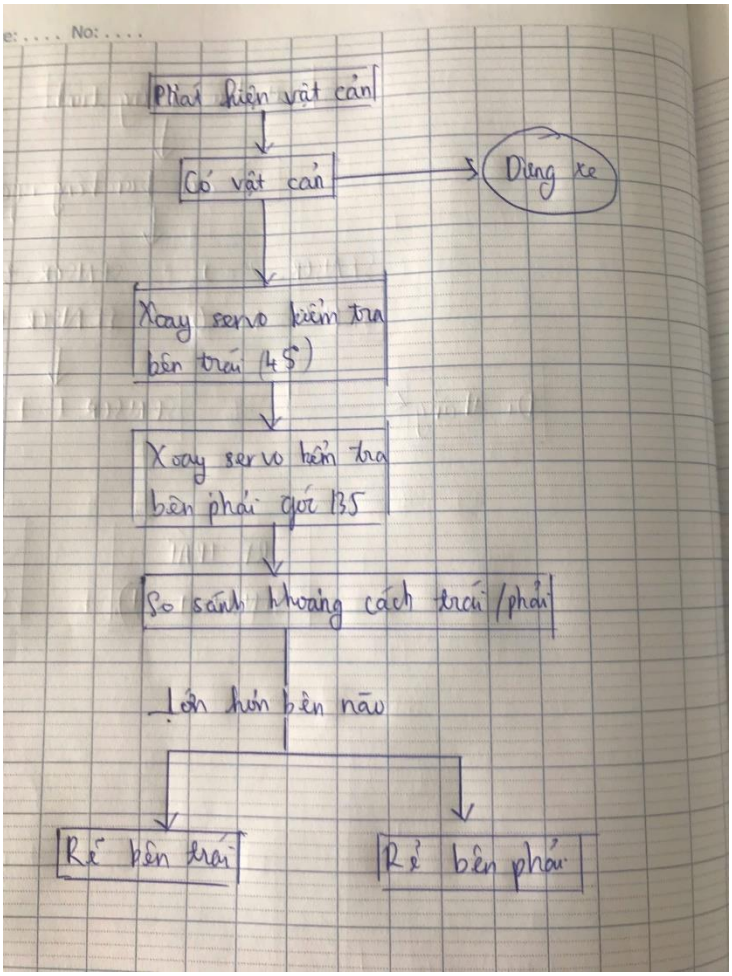
để kiểm tra khoảng cách.

- Xoay servo về góc phải (135 độ)

để kiểm tra khoảng cách.

- So sánh khoảng cách bên trái và bên phải:
  - Nếu khoảng cách bên nào lớn hơn,

    rẽ về hướng đó để tránh vật cản.



4.5 Khối chương trình chính điều khiển

- Khối điều khiển chuyển động của xe:
  - *xe_tien():* Điều khiển xe di chuyển thẳng về phía trước.
  - *xe_lui():* Điều khiển xe di chuyển lùi về phía sau.
  - *xe_re_phai():* Điều khiển xe rẽ phải.
  - *xe_re_trai():* Điều khiển xe rẽ trái.
  - *dung_xe():* Dừng xe lại ngay lập tức.

  Các hàm này điều khiển động cơ thông qua các chân GPIO, kết hợp điều chỉnh tốc độ và hướng thông qua PWM (điều chế độ rộng xung).

  Áp dụng:

```
def xe_tien():
    pwm_left.ChangeDutyCycle(40)
    GPIO.output(IN1, GPIO.LOW)
    GPIO.output(IN2, GPIO.HIGH)
    pwm_right.ChangeDutyCycle(40)
    GPIO.output(IN3, GPIO.LOW)
    GPIO.output(IN4, GPIO.HIGH)
```

- Khối điều khiển servo
  - *goc_servo(goc):* Điều khiển góc quay của động cơ servo từ 0 đến 180 độ.

  Hàm này được sử dụng để quay servo gắn với cảm biến siêu âm nhằm quét xung quanh và kiểm tra khoảng cách vật cản.

  Áp dụng:

```
def goc_servo(goc):
    if goc < 0:
        goc = 0
    elif goc > 180:
        goc = 180
    chu_ky = 2 + (goc / 18)
```

```
    chu_ky = max(2, min(12, chu_ky))
    GPIO.output(25, True)
    pwm.ChangeDutyCycle(chu_ky)
    time.sleep(0.5)
    GPIO.output(25, False)
    pwm.ChangeDutyCycle(0)
```

- Khối đo khoảng cách bằng cảm biến siêu âm

• *do_khoang_cach():* Đo khoảng cách từ xe đến vật cản bằng cảm biến siêu âm HC-SR04.

Hàm này sử dụng tín hiệu từ các chân *TRIG* và *ECHO* để tính toán thời gian xung âm thanh di chuyển, sau đó tính ra khoảng cách.

Áp dụng:

```
def do_khoang_cach():
    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)
    while GPIO.input(ECHO) == 0:
        bat_dau = time.time()

    while GPIO.input(ECHO) == 1:
        ket_thuc = time.time()
    thoi_gian = ket_thuc - bat_dau
    khoang_cach = (thoi_gian * 34300) / 2
    return khoang_cach
```

- Khối tránh vật cản

• *phat_hien_vat():* Kiểm tra xem có vật cản phía trước không, và nếu có, sẽ quyết định rẽ trái hoặc phải dựa trên khoảng cách đo được từ cảm biến siêu âm.

Hàm này sử dụng cảm biến siêu âm và động cơ servo để xác định khoảng cách đến vật cản và điều chỉnh hướng đi để tránh vật cản.

• *xe_tranh_vat_ben_trai()* và *xe_tranh_vat_ben_phai():* Thực hiện việc tránh vật cản bằng cách rẽ và tiếp tục di chuyển theo hướng đã điều chỉnh.

- Khối theo dõi đường

• *do_line():* Đọc dữ liệu từ các cảm biến hồng ngoại để điều khiển xe bám theo đường đã vạch sẵn. Khối này quyết định hướng đi của xe dựa trên tín hiệu nhận được từ các cảm biến. Tùy thuộc vào trạng thái của các cảm biến, xe sẽ rẽ trái, phải hoặc đi thẳng để giữ xe trên đúng lộ trình.

Áp dụng:

```
def do_line():
    sensor_1_state = GPIO.input(SENSOR_1)
    sensor_2_state = GPIO.input(SENSOR_2)
    sensor_3_state = GPIO.input(SENSOR_3)
    sensor_4_state = GPIO.input(SENSOR_4)
```

- Vòng lặp chính

• *try và while True:* Đây là vòng lặp chính của chương trình, kết hợp việc phát hiện vật cản và theo dõi đường.

Trong vòng lặp này, chương trình sẽ liên tục kiểm tra các cảm biến hồng ngoại và cảm biến siêu âm để điều chỉnh hướng đi của xe theo tình huống thực tế.

Áp dụng:

```
try:
    time.sleep(1)
    goc_servo(90)
    time.sleep(1)
    while True:
        sensor_1_state = GPIO.input(SENSOR_1)
        sensor_2_state = GPIO.input(SENSOR_2)
        sensor_3_state = GPIO.input(SENSOR_3)
        sensor_4_state = GPIO.input(SENSOR_4)
```

```
    if sensor_1_state == 1 or sensor_2_state == 1 or sensor_3_state == 1 or
sensor_4_state == 1:
        phat_hien_vat()
    do_line()
```