

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC BÁCH KHOA**  
**KHOA ĐIỆN – ĐIỆN TỬ**  
**BỘ MÔN ĐIỆN TỬ**

-----o0o-----



**LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC**

**HỆ THỐNG HỖ TRỢ LÁI TRÊN Ô TÔ**

**GVHD: TS. Nguyễn Lý Thiên Trường**  
**SVTH: Nguyễn Hữu Thuận**  
**MSSV: 1613428**

**TP. HỒ CHÍ MINH, THÁNG 9 NĂM 2020**



-----☆-----

-----☆-----

Số: \_\_\_\_\_ /BKĐT

Khoa: **Điện – Điện tử**

Bộ Môn: **Điện Tử**

## NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP

1. HỌ VÀ TÊN: NGUYỄN HỮU THUẬN MSSV: 1613428
2. NGÀNH: **ĐIỆN TỬ - VIỄN THÔNG** LỚP: DD16DV4
3. Đề tài: Hệ thống hỗ trợ lái trên ô tô
4. Nhiệm vụ:  
Nghiên cứu và thiết kế hệ thống hỗ trợ lái trên ô tô nhằm tạo sự an toàn và thoải mái cho tài xế khi lái xe. Từ những kiến thức đã nghiên cứu để thực hiện trên mô hình thực tế.
5. Ngày giao nhiệm vụ luận văn: 17/02/2020
6. Ngày hoàn thành nhiệm vụ: 07/09/2020
7. Họ và tên người hướng dẫn:  
TS. Nguyễn Lý Thiên Trường Phản hướng dẫn  
Hướng dẫn toàn bộ luận văn

Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.

Tp.HCM, ngày 07 tháng 09 năm 2020  
**CHỦ NHIỆM BỘ MÔN**

**NGƯỜI HƯỚNG DẪN CHÍNH**

TS. Trần Hoàng Linh

TS. Nguyễn Lý Thiên Trường

### **PHẦN DÀNH CHO KHOA, BỘ MÔN:**

Người duyệt (chấm sơ bộ): .....

Đơn vị: .....

Ngày bảo vệ: .....

Điểm tổng kết: .....

Nơi lưu trữ luận văn: .....

## **LỜI CẢM ƠN**

Đối với mỗi người chúng ta, ai cũng có thanh xuân và tuổi trẻ, mỗi người đều có những ước mơ hoài bão để thực hiện trong cuộc đời. Đối với tôi, trở thành một người kỹ sư là một trong những ước mơ mà tôi đã chọn. Để có cơ hội thực hiện được ước mơ đó tôi đã chọn Bách Khoa là ngôi trường để tôi xây dựng ước mơ. Được học tập tại Bách Khoa, tôi thấy mình rất may mắn, tôi học được rất nhiều thứ không những là kiến thức đơn thuần trong sách vở mà còn những kinh nghiệm mà thầy cô đã đúc kết được cũng như học tập được nhiều điều từ bạn bè. Đối với sinh viên thì luận văn tốt nghiệp chính là sản phẩm thể hiện lượng kiến thức đã tiếp thu được trong quá trình học tại giảng đường đại học và công sức đã bỏ ra để hoàn thành. Thông qua quá trình thực hiện luận văn, sinh viên sẽ trực tiếp đối mặt với những khó khăn trong thực tế mà trong khi học không bao giờ gặp, đó là điều kiện để sinh viên có thêm những kiến thức thực tế.

Em xin chân thành bày tỏ lòng biết ơn đến thầy **TS. Nguyễn Lý Thiên Trường** đã nhiệt tình hướng dẫn, góp ý, bổ sung những thiếu sót cho em trong quá trình thực hiện luận văn tốt nghiệp. Hơn nữa, những lời động viên của thầy đã giúp em vượt qua những khó khăn, trở ngại trong suốt quá trình thực hiện luận văn. Em xin chân thành cảm ơn đến toàn thể các thầy cô trong khoa cũng như trong nhà trường đã dạy dỗ và truyền đạt cho em những kiến thức lẫm kinh nghiệm cho em trong những năm qua. Lời cảm ơn sâu sắc nhất, con xin gửi đến ba mẹ đã động viên tinh thần con rất nhiều trong quá trình học tập cũng như làm luận văn tốt nghiệp. Trong quá trình nghiên cứu để thực hiện luận văn sẽ không tránh khỏi thiếu sót, em mong nhận được các ý kiến chỉ bảo của các quý thầy cô để em khắc phục và thực hiện tốt hơn trong công việc sau này. Cuối cùng, em xin kính chúc quý thầy cô luôn luôn dồi dào sức khỏe để hoàn thành tốt sự nghiệp giáo dục của mình, đào tạo ra nhiều sinh viên có ích cho đất nước.

Em xin chân thành cảm ơn!

Tp. Hồ Chí Minh, ngày 7 tháng 9 năm 2020.

**Sinh viên**

**Nguyễn Hữu Thuận**

## TÓM TẮT LUẬN VĂN

Luận văn này trình bày về hệ thống hỗ trợ lái trên ô tô. Về cơ bản, hệ thống hỗ trợ lái sử dụng camera, các cảm biến để theo dõi môi trường xung quanh, gửi tín hiệu về bộ điều khiển trung tâm, qua đó đưa ra cách xử lý phù hợp.

Chiếc xe sẽ tự động tính toán và giảm tốc theo xe phía trước, tự động đánh lái vô lăng để tự động đi đúng làn, hay đáng quan tâm hơn là tự động phanh khẩn cấp khi cần thiết để va chạm không xảy ra, hoặc hạn chế tối đa chấn thương đối với người ngồi trong xe cũng như hư hại xe xuống mức thấp nhất.

Nội dung trong luận văn này sẽ trình bày ba vấn đề chính cơ bản như sau:

Vấn đề 1: Nghiên cứu phương pháp nhận diện làn đường bằng xử lý ảnh dùng thư viện OpenCV thực hiện thử nghiệm trên bo mạch Raspberry Pi 4

Vấn đề 2: Phát hiện vật cản phía trước khi xe đang di chuyển trên đường từ đó hỗ trợ xe chuyển làn khi các điều kiện xung quanh cho phép nhờ vào các cảm biến được trang bị xung quanh xe.

Vấn đề 3: Phát hiện và nhận dạng các loại biển báo trên đường để đưa ra các hành động xử lý kịp thời cho các loại biển báo khác nhau cho phù hợp, dùng board Raspberry Pi 4 kết hợp với thư viện OpenCV, Tensorflow và Keras cùng với giải thuật học sâu.

Kết hợp các vấn đề chính đã nói trên ta sẽ có một hệ thống hỗ trợ lái trên ô tô đơn giản, xử lý được các trường hợp không quá phức tạp trên thực tế.

## MỤC LỤC

1. GIỚI THIỆU .....	1
1.1 Tổng quan .....	1
1.2 Tình hình nghiên cứu trong và ngoài nước .....	4
1.3 Nhiệm vụ luận văn.....	9
2. LÝ THUYẾT .....	11
2.1. Các phần cứng được sử dụng .....	11
2.1.1. Khung xe kim loại Racing Car RC1 .....	11
2.1.2. Raspberry Pi 4.....	15
2.1.3. Camera Raspberry Pi V1 5MP .....	17
2.1.4. Tiva C .....	18
2.1.5. Mạch điều khiển động cơ DC L298 .....	20
2.1.6. Động Cơ DC Servo GM25-370 DC Geared Motor.....	22
2.1.7. Động cơ RC Servo MG996 .....	24
2.1.8. Mạch giảm áp DC 5V 3A .....	24
2.1.9. Mạch giảm áp DC LM2596.....	25
2.1.10. Cảm Biến Khoảng Cách Hồng Ngoại Analog SHARP GP2Y0A21YK0F	26
2.1.11. Cảm biến thân nhiệt chuyển động PIR HC-SR501 .....	31
2.1.12. Cảm biến khoảng cách VL53L0X Laser Distance ToF Sensor GY-53L0.	33
2.1.13 Mạch giảm áp DC Mini 5V 3A cổng USB Charge Module.....	39
2.2. Các nguyên lý được áp dụng vào phần mềm .....	41
2.2.1. OpenCV .....	41
a. Hough Line Transform.....	41
a. Biến đổi hình ảnh .....	44
b. Làm mịn hình ảnh .....	45
c. Bộ lọc Gauss .....	46

d. Canny Edge Detection.....	48
2.2.2. FreeRTOS .....	50
2.2.3 Tensorflow .....	62
2.2.4. Thuật toán phát hiện đối tượng.....	64
2.2.5. Thuật toán phân loại đối tượng.....	65
2.2.6. Chuẩn giao tiếp I2C .....	66
3. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG .....	74
3.1 Khối cung cấp năng lượng .....	75
3.2 Khối nhận dạng làn đường .....	76
3.3 Khối nhận dạng biển báo .....	77
3.4 Khối cảnh báo .....	78
3.5 Khối cảm biến .....	79
3.6 Khối thực thi .....	81
3.7 Khối xử lý trung tâm .....	82
4. THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM .....	84
4.1. Nhận diện làn đường với Raspberry Pi .....	84
4.2. Phát hiện và nhận diện biển báo giao thông dùng Raspberry Pi.....	85
4.2.1. Tiền xử lý ảnh.....	85
4.2.1.1. Không gian màu HSV .....	85
4.2.1.2. Xử lý ảnh theo hình thái.....	87
4.2.1.3. Tính toán miền được kết nối .....	87
4.2.1.4. Phát hiện hình elip .....	87
4.2.2. Phân loại mạng thần kinh chuyển đổi LeNet .....	88
4.2.2.1. Lớp C1-Convolutional .....	89
4.2.2.2. Lớp S2-Pooling .....	89
4.2.2.3. Lớp C3-Convolutional .....	89
4.2.2.4. Lớp S4-Pooling .....	90
4.2.2.5. Lớp C5-Convolutional .....	90

4.2.2.6. F6 – Fully connected layer.....	91
4.2.2.7. OUTPUT - Fully connected layer.....	91
4.3. Tránh vật cản.....	94
4.4. Các task được thực hiện trên Tiva C .....	100
5. KẾT QUẢ THỰC HIỆN .....	106
5.1 Tổng quan về sản phẩm thực hiện.....	106
5.2 Chức năng nhận dạng làn đường.....	113
5.3 Chức năng nhận dạng biển báo giao thông .....	116
5.4 Chức năng chuyên làn .....	122
5.5 Các số liệu đánh giá kết quả đạt được.....	124
6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	126
6.1 Kết luận .....	126
6.2 Hướng phát triển.....	126
7. TÀI LIỆU THAM KHẢO .....	128

## DANH SÁCH HÌNH MINH HỌA

<i>Hình 1.1 Minh họa hệ thống hổ trợ lái trên ô tô .....</i>	1
<i>Hình 1.2 Minh họa hệ thống hổ trợ lái trên ô tô .....</i>	5
<i>Hình 1.3 Minh họa hệ thống hổ trợ lái trên ô tô .....</i>	6
<i>Hình 1.4 Minh họa hệ thống hổ trợ lái trên ô tô .....</i>	7
<i>Hình 2.1. Tổng quan khung xe kim loại RC1 .....</i>	11
<i>Hình 2.2. Khung xe kim loại RC1 .....</i>	11
<i>Hình 2.3. Khung xe kim loại RC1 .....</i>	12
<i>Hình 2.4. Hệ thống truyền động xe kim loại RC1 .....</i>	12
<i>Hình 2.5. Hệ thống truyền động xe kim loại RC1 .....</i>	12
<i>Hình 2.6. Servo xe kim loại RC1 .....</i>	13
<i>Hình 2.7. Servo xe kim loại RC1 .....</i>	13
<i>Hình 2.8. Khung xe kim loại RC1 .....</i>	13
<i>Hình 2.9. Bánh trước xe kim loại RC1 .....</i>	14
<i>Hình 2.10. Bánh trước xe kim loại RC1 .....</i>	14
<i>Hình 2.11. Hệ thống lái trên xe kim loại RC1 .....</i>	14
<i>Hình 2.12. Mạch Raspberry Pi 4 .....</i>	16
<i>Hình 2.13. Sơ đồ GPIO Raspberry Pi .....</i>	17
<i>Hình 2.14 Camera Raspberry Pi V1 5MP .....</i>	18
<i>Hình 2.15. Mạch Tiva C .....</i>	19
<i>Hình 2.16. Sơ đồ GPIO Tiva C .....</i>	19
<i>Hình 2.17. Sơ đồ mạch điều khiển động cơ DC L298 .....</i>	20
<i>Hình 2.18. Mạch điều khiển động cơ DC L298 .....</i>	21
<i>Hình 2.19. Cấu tạo động cơ DC Servo GM25-370 .....</i>	23

Hình 2.20. Động cơ RC Servo .....	24
Hình 2.21. Mạch giảm áp DC 5V 3A.....	25
Hình 2.22. Mạch giảm áp LM2596 .....	26
Hình 2.23. Cảm biến khoảng cách hồng ngoại analog Sharp .....	26
Hình 2.24. Sơ đồ khói GP2Y0A21YK0F .....	27
Hình 2.25. Biểu đồ thời gian GP2Y0A21YK0F .....	28
Hình 2.26. Biểu đồ khoảng cách và điện áp GP2Y0A21YK0F .....	29
Hình 2.27. Biểu đồ khoảng cách và điện áp GP2Y0A21YK0F .....	29
Hình 2.28 Mạch chuyển tín hiệu analog sang digital .....	30
Hình 2.29 Sơ đồ nguyên lý mạch chuyển tín hiệu tương tự sang tín hiệu số .....	30
Hình 2.30 Sơ đồ nguyên lý mạch chuyển tín hiệu tương tự sang tín hiệu số .....	31
Hình 2.31. Cảm biến thân nhiệt chuyển động HC-SR501 .....	32
Hình 2.32. Cảm biến thân nhiệt chuyển động HC-SR501 .....	32
Hình 2.33 Nguyên lý hoạt động cảm biến PIR .....	33
Hình 2.34. Cảm biến khoảng cách VL53L0X .....	34
Hình 2.35. Sơ đồ cảm biến khoảng cách VL53L0X.....	34
Hình 2.36. Cảm biến khoảng cách VL53L0X .....	35
Hình 2.37. Sơ đồ khói VL53L0X.....	35
Hình 2.38. Sơ đồ PINOUT VL53L0X .....	36
Hình 2.39. Sơ đồ schematic VL53L0X.....	36
Hình 2.40 Giao thức truyền dữ liệu.....	37
Hình 2.41 Địa chỉ thiết bị VL53L0X I2C: 0x29 .....	37
Hình 2.42 Định dạng dữ liệu VL53L0X (ghi).....	38
Hình 2.43 Định dạng dữ liệu VL53L0X (đọc) .....	38
Hình 2.44 Định dạng dữ liệu VL53L0X (ghi tuần tự) .....	39

Hình 2.45. Định dạng dữ liệu VL53L0X (đọc tuần tự).....	39
Hình 2.46. Mạch giảm áp 5V-3A .....	40
Hình 2.47. Minh họa biểu diễn đường thẳng trong hệ tọa độ cực .....	42
Hình 2.48. Mapping một đường thẳng từ không gian ảnh sang không gian Hough ....	43
Hình 2.49. Mapping một điểm từ không gian ảnh sang không gian Hough .....	43
Hình 2.50. Mapping nhiều điểm thẳng hàng từ không gian ảnh sang không gian Hough .....	44
Hình 2.51. Biểu diễn 2 đường thẳng trong không gian Hough .....	44
Hình 2.52 Chuyển ảnh RGB sang ảnh xám .....	45
Hình 2.53. Làm mịn ảnh .....	46
Hình 2.54. Bộ lọc Gauss .....	47
Hình 2.55. Non-maximum Suppression .....	49
Hình 2.56. Lọc ngưỡng .....	49
Hình 2.57. Các cấp độ yêu cầu real-time .....	50
Hình 2.58. Cấu tạo của một hệ điều hành thời gian thực (RTOS) .....	51
Hình 2.59. Nguyên lý làm việc của RTOS .....	52
Hình 2.60. Task trong RTOS .....	52
Hình 2.61. Round-robin .....	53
Hình 2.62. Priority base .....	54
Hình 2.63. Priority-based pre-emptive .....	54
Hình 2.64. Signal event .....	56
Hình 2.65. Message queue .....	57
Hình 2.66. Queue .....	58
Hình 2.67. Mail queue .....	58
Hình 2.68. Mail queue .....	59
Hình 2.69. Semaphore .....	60

Hình 2.70. Mutex .....	61
Hình 2.71. Một sơ đồ Venn mô tả học sâu .....	63
Hình 2.72 Bus vật lý I2C .....	67
Hình 2.73 Thiết bị chủ (Master) và tớ (Slave).....	68
Hình 2.74 Giao thức truyền dữ liệu I2C.....	68
Hình 2.75 Điều kiện bắt đầu I2C.....	69
Hình 2.76 Điều kiện kết thúc truyền dữ liệu I2C.....	69
Hình 2.77 Gửi dữ liệu đến thiết bị Slave .....	70
Hình 2.78 Gửi dữ liệu đến thiết bị Slave .....	71
Hình 2.79 Gửi dữ liệu đến thiết bị Slave .....	71
Hình 2.80 Gửi dữ liệu đến thiết bị Slave .....	72
Hình 2.81 Gửi dữ liệu đến thiết bị Slave .....	72
Hình 2.82 Khung dữ liệu Master gửi cho Slave .....	73
Hình 2.83 Khung dữ liệu Master nhận từ Slave .....	73
Hình 3.1. Sơ đồ khái tổng quát của hệ thống .....	74
Hình 3.2. Sơ đồ khái hệ thống cung cấp năng lượng .....	75
Hình 3.3 Sơ đồ khái nhận dạng làn đường.....	76
Hình 3.4 Sơ đồ khái nhận dạng biển báo .....	77
Hình 3.5. Sơ đồ khái hệ thống cảnh báo .....	78
Hình 3.6 Khái cảm biến.....	79
Hình 3.7 Khái thực thi .....	81
Hình 3.8. Sơ đồ khái giao tiếp các board mạch .....	82
Hình 4.1 Sơ đồ giải thuật nhận dạng làn đường .....	84
Hình 4.2. Các biển báo giao thông cơ bản.....	85
Hình 4.3. Hệ màu HSV .....	86

Hình 4.4. Mạng thần kinh tích chập LeNet .....	88
Hình 4.5. Sơ đồ giải thuật xác định và nhận dạng biển báo giao thông .....	92
Hình 4.6. Kết quả huấn luyện mạng CNN .....	94
Hình 4.7 Sơ đồ giải thuật tổng quát cho chức năng tránh vật cản .....	96
Hình 4.8. Lưu đồ giải thuật chương trình rẽ trái .....	97
Hình 4.9. Lưu đồ giải thuật chương trình rẽ phải .....	99
Hình 4.10. Lưu đồ giải thuật task 1 – Nhận tín hiệu từ Raspberry_1 .....	100
Hình 4.11. Lưu đồ giải thuật task 2 – Nhận tín hiệu từ Raspberry_2 .....	101
Hình 4.12. Lưu đồ giải thuật task 3 – Điều khiển servo RC .....	102
Hình 4.13. Lưu đồ giải thuật task 4 – Điều khiển động cơ DC .....	104
Hình 4.14. Lưu đồ giải thuật task 5 – Chương trình cảnh báo .....	105
Hình 5.1. Phía trước xe .....	106
Hình 5.2. Bên phải xe .....	107
Hình 5.3. Bên trái xe .....	107
Hình 5.4. Phía sau xe .....	108
Hình 5.5. Phía trên xe .....	109
Hình 5.6. Đoạn đường thử nghiệm .....	110
Hình 5.7. Các biển báo thử nghiệm .....	111
Hình 5.8. Các biển báo thử nghiệm .....	111
Hình 5.9. Các biển báo thử nghiệm .....	112
Hình 5.10 Kết quả nhận dạng làn đường – điều hướng đi thẳng .....	113
Hình 5.11 Kết quả nhận dạng làn đường điều hướng rẽ trái .....	114
Hình 5.12 Kết quả nhận dạng làn đường – điều hướng rẽ phải .....	115
Hình 5.13 Kết quả nhận dạng biển báo giới hạn tốc độ 80 km/h .....	116
Hình 5.14 Kết quả nhận dạng biển báo giới hạn tốc độ 30km/h .....	117

Hình 5.15 Kết quả nhận dạng biển báo giới hạn tốc độ 10km/h .....	118
Hình 5.16 Kết quả nhận diện biển báo dừng lại .....	119
Hình 5.17 Kết quả nhận dạng biển báo rẽ trái .....	120
Hình 5.18 Kết quả nhận diện biển báo rẽ phải .....	121
Hình 5.19 Khoảng cách được đo từ cảm biến lazer .....	122
Hình 5.20 Đèn báo hiệu rẽ trái khi xe chuẩn bị rẽ trái .....	122
Hình 5.21 Xe bật đèn rẽ trái khi gặp vật cản phía trước và bên trái .....	123
Hình 5.22 Xe dừng lại khi gặp vật cản và không thể rẽ trái .....	123

## **DANH SÁCH BẢNG SỐ LIỆU**

<i>Bảng 2.1 Thông số hoạt động của cảm biến</i> .....	28
<i>Bảng 2.2 Đặc tính quang – điện của cảm biến</i> .....	28
<i>Bảng 2.3 Điện áp cung cấp khuyến dùng</i> .....	28
<i>Bảng 2.4 Thông số kỹ thuật của cảm biến</i> .....	35
<i>Bảng 3.1 Giao tiếp GPIO giữa Tiva C và Raspberry_1</i> .....	77
<i>Bảng 3.2 Giao tiếp giữa GPIO giữa Tiva C và Raspberry_2</i> .....	78
<i>Bảng 3.3 Kết nối giữa Tiva C, LED và buzzer</i> .....	79
<i>Bảng 3.4 Kết nối giữa Tiva C và các cảm biến</i> .....	80
<i>Bảng 3.5 Kết nối giữa Tiva C và các động cơ</i> .....	82
<i>Bảng 3.6 Kết nối tổng quát của Tiva C</i> .....	83
<i>Bảng 4.1. Bảng tham số lớp C1-Convolutional</i> .....	89
<i>Bảng 4.2. Bảng tham số lớp S2-pooling</i> .....	89
<i>Bảng 4.3. Bảng tham số lớp C3-Convolutional</i> .....	89
<i>Bảng 4.4. Bảng tham số lớp S4-pooling</i> .....	90
<i>Bảng 4.5. Bảng tham số lớp C5-Convolutional</i> .....	90
<i>Bảng 4.6. Tín hiệu kết nối giữa Raspberry_2 và Tiva C</i> .....	94
<i>Bảng 5.1 Kết quả nhận dạng làn đường</i> .....	124
<i>Bảng 5.2 Kết quả nhận dạng biển báo</i> .....	124
<i>Bảng 5.3 Kết quả chuyển làn khi gấp vật cản</i> .....	124

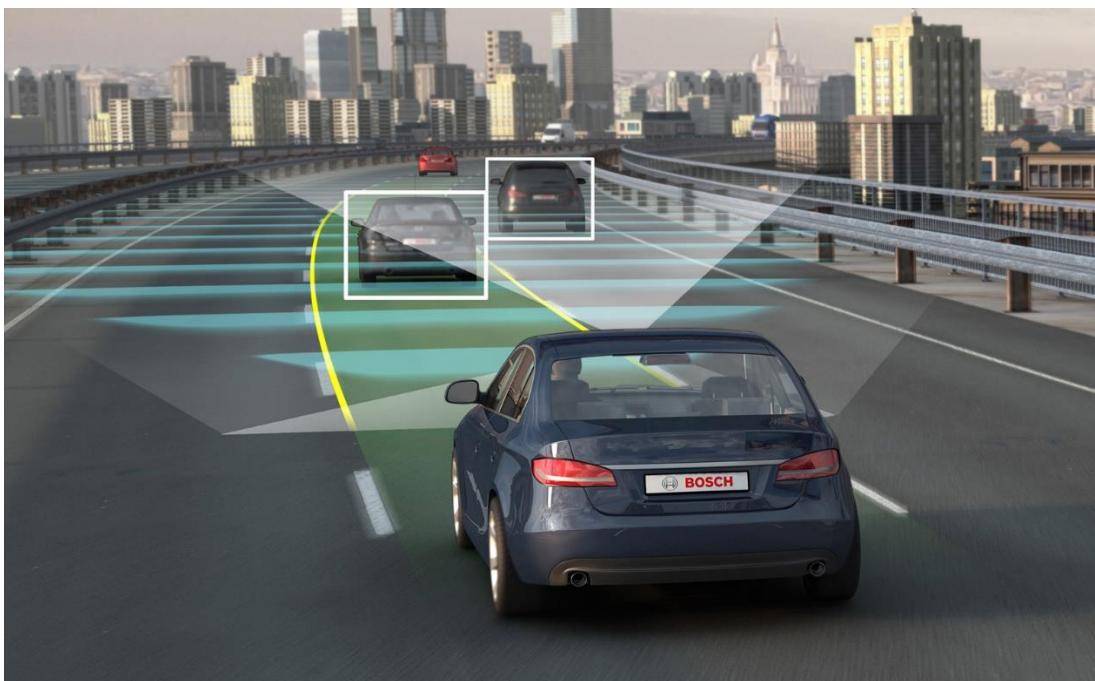
## 1. GIỚI THIỆU

### 1.1 Tổng quan

Về cơ bản, hệ thống hỗ trợ lái sử dụng camera, radar hay các cảm biến để theo dõi môi trường xung quanh, gửi tín hiệu về bộ điều khiển trung tâm, qua đó tự động đưa ra các xử lý phù hợp.

Chiếc xe sẽ tự động tính toán và giảm tốc theo xe phía trước, tự động đánh lái vô lăng để tự động đi đúng làn, hay đáng quan tâm hơn là tự động phanh khẩn cấp khi cần thiết để va chạm không xảy ra, hoặc hạn chế tổn thất.

Đối với việc phát triển hệ thống hỗ trợ lái xe thì phần mềm đóng vai trò quan trọng nhất. Nó được ví như não bộ, đảm bảo sự vận hành cho xe. Nếu trước đây, ô tô đặc trưng bởi động cơ, hộp số, bộ dẫn động, vô lăng điều khiển, xăng dầu... thì ngày nay, nó giống như một chiếc máy tính. Phần mềm và điện đã thay thế chức năng của các yếu tố cơ học, con người và nhiên liệu. Một chiếc xe hiện đại được điều khiển bởi 80% tới 100% hệ thống nhúng, 90% sáng tạo của xe hơi hiện nay nằm ở phần mềm, 100% xe sẽ kết nối với Cloud.<sup>[1]</sup>



Hình 1.1 Minh họa hệ thống hỗ trợ lái trên ô tô

Đối với việc phát triển hệ thống này thì phần mềm đóng vai trò quan trọng nhất. Các nghiên cứu về công nghệ trên xe tập trung vào 2 lĩnh vực chính: phát hiện làn đường và nhận dạng đối tượng.

*Phát hiện làn đường:* Vấn đề này đã được nghiên cứu trong nhiều thập kỷ qua. Phần lớn các hệ thống phát hiện làn đường đã được phát triển và ứng dụng trong nhiều loại xe sang.

*Nhận dạng đối tượng:* Đây là một thành phần quan trọng của hệ thống xe tự hành. Gần đây, công nghệ này đã có những bước tiến bộ lớn như nhận dạng được đối tượng tĩnh như xe đạp, người đi bộ, ô tô, biển báo giao thông... và đang tiến tới việc nhận dạng đối tượng động như xe/người đang di chuyển trên đường thật...

Là hướng nghiên cứu trong tương lai của ô tô, lái xe không người lái có tác động sâu sắc đến ngành công nghiệp ô tô và thậm chí cả ngành vận tải. Sự ra đời của những chiếc xe không người lái sẽ giải phóng bàn tay con người, giảm tần suất tai nạn giao thông và đảm bảo an toàn cho con người. Đồng thời, với sự tiến bộ đột phá và liên tục của các công nghệ cốt lõi như trí tuệ nhân tạo và phát hiện cảm biến, xe không người lái chắc chắn sẽ trở nên thông minh hơn, và cũng có thể nhận ra sự công nghiệp hóa của xe không người lái. Một chiếc xe không người lái có nghĩa là trang bị cho chiếc xe phần mềm thông minh và nhiều thiết bị cảm biến, bao gồm cảm biến trên xe, radar, GPS và camera, để có được thông tin về môi trường xung quanh của chiếc xe, và thực hiện xử lý và phân tích thông minh dựa trên thông tin thu được như não người ra quyết định, qua đó kiểm soát hướng và tốc độ của phương tiện, thực hiện việc lái xe tự chủ, đến đích an toàn và hiệu quả, và cuối cùng đã loại bỏ hoàn toàn tắc nghẽn giao thông và tai nạn giao thông, góp phần vượt trội vào bảo vệ môi trường. Nói một cách đơn giản, một chiếc xe không người lái đê cập đến một chiếc xe thông minh hoàn toàn tự động và không cần sự can thiệp của con người trong quá trình lái xe. Nền tảng hệ thống lái xe không người lái rất phức tạp, chủ yếu liên quan đến bốn lĩnh vực kỹ thuật: nhận thức môi trường, lập kế hoạch đường đi, điều khiển máy tính và kiểm soát quyết định. Do đó, hệ thống lái xe không người lái chủ yếu bao gồm hệ thống nhận thức môi trường, hệ thống điều khiển máy tính, hệ thống kiểm soát quyết định và hệ thống thông tin địa lý. Hệ thống nhận thức môi trường hoạt động như một con mắt tinh ranh của chiếc xe

không người lái, chủ yếu thông qua các cảm biến bên ngoài gắn trên chiếc xe không người lái.

Thiết bị có được thông tin môi trường bên ngoài, mô hình hóa nó, và truyền chính xác và nhanh chóng thông tin địa lý và thông tin chướng ngại vật của xe đến hệ thống điều khiển máy tính, vì vậy hệ thống này chủ yếu bao gồm hai phần: hệ thống định vị xe và hệ thống phát hiện chướng ngại vật. Thời kỳ đầu, công nghệ điều hướng từ tính được sử dụng để lấy thông tin thông qua các thiết bị định vị như định từ hoặc dây điện chôn trên đường, sau đó một camera quang học được sử dụng để phân tích và xử lý hình ảnh được chụp bởi camera để lấy thông tin đường, nhưng cách xử lý và chế độ hình ảnh này yêu cầu công nghệ nhận dạng rất cao, nếu không thì khó đảm bảo tính chính xác và hiệu suất thời gian thực của hệ thống. Hiện tại, chiếc xe tự lái do Google phát triển sử dụng công nghệ flipar. Nguyên tắc chính của công nghệ này là phát ra tia laser và nhận tín hiệu laser, phát hiện tín hiệu phản xạ từ chướng ngại vật mà laser gặp phải. Khoảng cách giữa chướng ngại vật và cảm biến cũng có thể có được định vị thông qua Google Maps. Theo thuật toán điều khiển của phần mềm hệ thống, các hướng dẫn chiến lược điều khiển sau đó được truyền đến bộ chấp hành để thực thi. Kiểu ra quyết định thông minh này đòi hỏi sự hỗ trợ của phần mềm nền mạnh mẽ để có thể xử lý chính xác và nhanh chóng một lượng lớn thông tin như lidar, GPS, điện áp pin, dòng điện làm việc để đáp ứng các yêu cầu nghiêm ngặt về thời gian thực. Nhận dạng biển báo giao thông thuộc về nhận dạng hình ảnh của hệ thống nhận thức môi trường. Phân tích hình ảnh được chụp bởi máy ảnh thông qua phát hiện hình elip và trích xuất các biển báo giao thông. Mạng thần kinh tích chập LeNet được sử dụng để phân loại các biển báo giao thông được trích xuất để xác định di chuyển tiếp theo của chiếc xe. Để xác minh hiệu quả thực tế của thuật toán, ta áp dụng mô hình cho chiếc xe được điều khiển thử nghiệm trên Raspberry Pi và Tiva C, nhận ra việc xây dựng hệ thống nhận dạng tín hiệu giao thông và mô phỏng các kịch bản ứng dụng có thể có của những chiếc xe không người lái trong tương lai.

## 1.2 Tình hình nghiên cứu trong và ngoài nước

Hiện đã có rất nhiều tập đoàn sản xuất xe hơi và công nghệ lớn trên thế giới đã tham gia vào cuộc chạy đua phát triển xe hơi công nghệ tự lái thông minh (gọi tắt là xe tự lái, xe tự hành) mà không cần đến bàn tay can thiệp của con người, trong đó có những tên tuổi nổi bật như Tesla, Daimler, Google,...

Theo CNN, những báo cáo sau các thử nghiệm cho thấy, xe tự lái có thể giảm tới 90% các vụ tai nạn như hiện nay, cứu được 30.000 người, giảm được 2,12 triệu thương tích mỗi năm. Xe tự lái chạy điện cũng giúp tiết kiệm được 134 tỷ gallon xăng/năm. Còn trong khảo sát của trang tin công nghệ Verge, trên 60% người trưởng thành tại Mỹ sẽ sử dụng xe tự lái, còn 32% sử dụng những dịch vụ mà xe tự lái mang lại.

Với xu hướng phát triển công nghệ hiện nay, các chuyên gia dự đoán, trong khoảng 15 đến 20 năm nữa, xe tự lái sẽ áp đảo các phương tiện đang thịnh hành hiện nay. Trước mắt, theo ông Elon Musk, Giám đốc điều hành hãng Tesla Motor, các mẫu xe tự lái của hãng đang thử nghiệm có thể tự vận hành lên đến 90% thao tác mà không cần sự can thiệp của con người. Bên cạnh đó, mẫu xe Cadillac thử nghiệm tự lái của GM có thể tự vận hành với vận tốc lên đến 70 dặm/giờ (khoảng 112 km/giờ).

Google là một trong những tập đoàn tích cực tham gia vào các dự án xe tự lái. Cá Google và Tesla cùng dự đoán rằng trong những năm tới đôi tay của con người sẽ được giải phóng hoàn toàn khi lên xe. Google còn đang tính toán làm sao cho sản phẩm của họ trở nên thông minh hơn trong tương lai như: Tự xử lý tình huống khi gặp phải ùn tắc, khi có người cần cấp cứu trên xe, lập trình đi qua các nút giao thông,... Ngoài ra, hãng còn lập một trung tâm xử lý thông tin tiếp nhận các phản hồi từ người dùng để khắc phục lỗi nhằm hoàn thiện sản phẩm. 48 chiếc xe tự lái của Google được cấp phép cũng gặp phải những tai nạn khi chạy thử tại bang California. Chính quyền bang Nevada cũng chính thức cấp phép cho xe tự lái của Google chạy trên đường. Chris Urmson, Giám đốc dự án chương trình xe tự lái của Google cho biết: “Trong 6 năm kể từ khi triển khai dự án này, chúng tôi gặp phải 11 vụ tai nạn nhỏ (hư hại nhẹ, không có chấn thương) trong 1,7 triệu dặm đường tự lái và có lái với những người lái xe an toàn ngồi

sau vô lăng. Tám trong số 11 vụ tai nạn xảy ra trên đường phố”. Urmson nhận xét, nhờ các bộ cảm biến, xe tự lái có khả năng “quan sát” 360 độ, có thể duy trì một khoảng cách an toàn với người đi đường và các phương tiện khác.



*Hình 1.2 Minh họa hệ thống hỗ trợ lái trên ô tô*

Xu thế xe tự lái đang khiến cho Tập đoàn sản xuất ô tô GM lo ngại. Đại diện GM cho biết, xe tự lái có thể trở thành sự cạnh tranh rất nghiêm trọng, trở thành đối thủ lớn của ngành công nghiệp sản xuất ô tô truyền thống. Gần đây nhất, sau khi thử nghiệm công nghệ tự lái trên những chiếc xe thông thường, Google giới thiệu mẫu xe tự lái hai chỗ không vô lăng, không phanh. Người đi chỉ cần nhập địa chỉ nơi đến, xe sẽ chạy với tốc độ 40 km/h mà không phải “động tay” trong suốt hành trình. Hiện GM, Toyota hay những hãng khác đang thử nghiệm công nghệ xe tự lái, để chuẩn bị cho xu thế cạnh tranh mới trong tương lai.

Việc số lượng xe tự động lái gia tăng cũng sẽ đồng nghĩa với giá thành giảm xuống, hiện nay một chiếc xe hơi thông minh được trang bị kết nối sẽ có mức giá rơi vào khoảng 55.000 USD, do thị trường của dòng xe này vẫn còn khá nhỏ bé.

Xuất hiện lần đầu tiên năm 1939 dưới sự tài trợ của General Motors tại Hội chợ thế giới, xe không người lái chủ yếu phục vụ mục đích nghiên cứu khoa học. Thế nhưng

những năm gần đây, các hãng xe lớn như GM, VW, Audi, BMW, Volvo hay Cadillac đã bắt đầu thử nghiệm hệ thống tự vận hành trên nhiều mẫu xe ứng dụng.

Không nằm ngoài xu hướng đó, VW hiện nay đang thử nghiệm hệ thống điều khiển tự động tạm thời (TAP) cho phép xe tự lái với tốc độ lên tới 128 km/giờ trên đường cao tốc. Nhà máy ô tô tự động đầu tiên của Trung Quốc kết hợp với Đại học Công nghệ Quốc phòng Quốc gia đã thử nghiệm mẫu xe không người lái Hongqi HQ3 trên quãng đường 280 km, với vận tốc 88 km/giờ trong điều kiện đường cao tốc tấp nập.

Ví dụ với nghiên cứu của Google, để một chiếc xe có thể tự điều khiển, cần tới sự kết hợp của một loạt các công nghệ kèm theo như bản đồ được lập trình sẵn, radar, cảm biến laser và camera. Mỗi chiếc xe đều trải qua thời kì thử nghiệm nghiêm ngặt và giai đoạn phát triển lâu dài để đảm bảo rằng tất cả thiết bị hoạt động nhịp nhàng đồng thời.

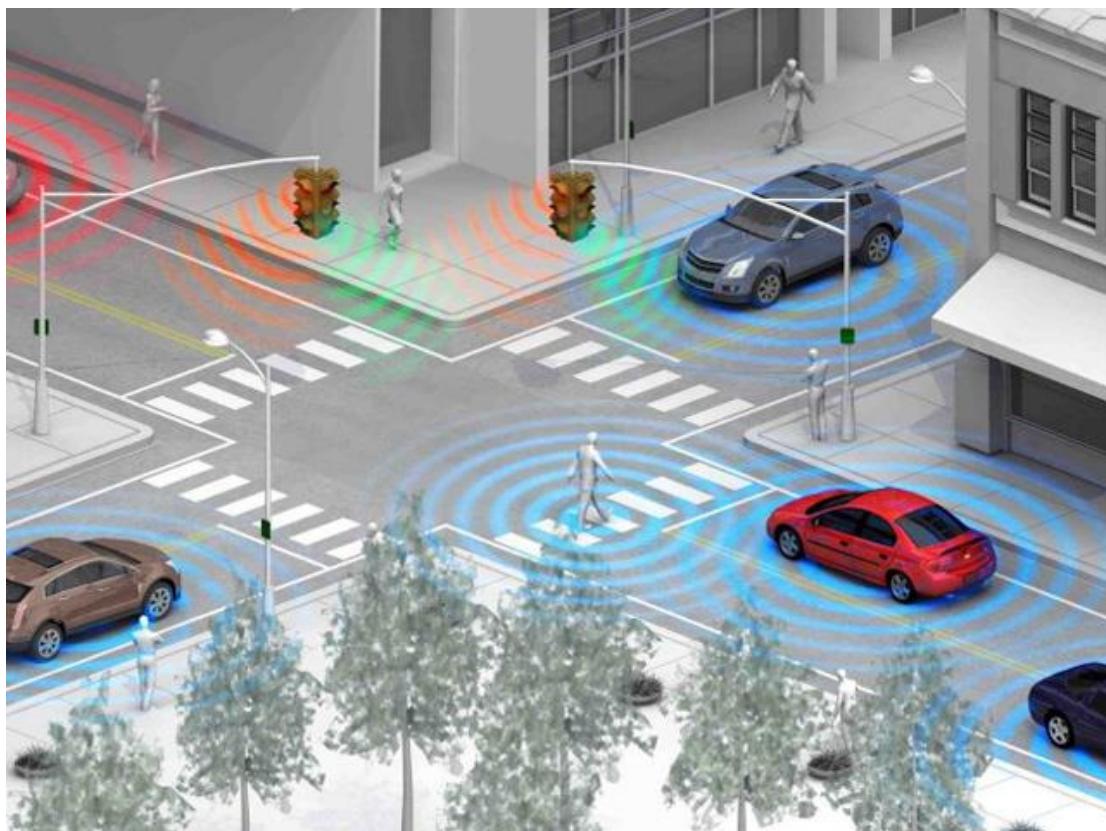


*Hình 1.3 Minh họa hệ thống hỗ trợ lái trên ô tô*

Trước khi áp dụng công nghệ tự động trên bất cứ tuyến đường nào, các kỹ sư phải tự lái xe và sử dụng công cụ như máy ảnh, cảm biến hay radar để ghi lại bản đồ kỹ thuật số thật chi tiết những đặc tính của lộ trình. Bằng cách lập bản đồ đánh dấu làn

đường cũng như các biển báo giao thông, phần mềm trong xe sẽ được làm quen trước với môi trường bên ngoài và đặc điểm đường lái.

Những chuyến đi ban đầu được thực hiện với sự giúp sức của hệ thống hỗ trợ lái xe để chuẩn bị sẵn sàng cho bước tiếp theo. Chiếc xe sẽ xử lý trên đường mà thiếu vắng hệ thống hỗ trợ, chỉ có camera, cảm biến laser và radar để giúp quyết định vị trí và tốc độ di chuyển của các xe khác. Phần mềm kiểm soát tăng giảm tốc độ cùng với camera gắn phía trước sẽ đọc và giải thích tín hiệu đèn giao thông hay các tín hiệu khác xuất hiện trên đường.



*Hình 1.4 Minh họa hệ thống hỗ trợ lái trên ô tô*

Tương tự với công nghệ “Super Cruise” của Cadillac, ý tưởng đằng sau khá đơn giản: trong điều khiển môi trường nhất định được đáp ứng, hệ thống sẽ cung cấp sự điều khiển, phanh và xác định làn đường hoàn toàn tự động. Khi được triển khai, hệ thống sẽ dựa trên thiết bị kiểm soát thích ứng hành trình để duy trì khoảng cách an toàn với xe phía trước, cộng với sự hỗ trợ của camera để phát hiện làn đường. Dữ liệu này cũng

được so sánh với dữ liệu GPS để theo dõi thay đổi bất ngờ trên đường như khúc cua hay vỉa hè.

Những ưu điểm của xe tự lái:

Chỉ cần xem xét khả năng kiểm soát mà máy tính có thể thực hiện dưới mui xe, không thể nghi ngờ gì về mức độ hiệu quả của xe tự hành. Hệ thống chống bó phanh tự động luôn tốt hơn ngưỡng phanh mà người lái thực hiện trên thiết bị tiêu chuẩn, tương tự với việc kiểm soát lực bám và kiểm soát sự ổn định. Công nghệ tiên tiến này đã giảm bớt gánh nặng đặt lên người điều khiển trên đường cao tốc hay khi tắc đường bởi tất cả đều được tự động hóa. Từ phát hiện điểm mù, cảnh báo làn đường khi khởi hành, quản lý gia nhập làn đường và thậm chí tự đậu xe...

Kết quả nghiên cứu cho thấy người điều khiển là nguyên nhân chính gây ra tai nạn giao thông. Với xe tự hành, sự phân tâm khi mệt mỏi hay say rượu sẽ không ảnh hưởng tới khả năng làm chủ trên đường bởi nhiệm vụ này được máy tính đảm nhận. Không thể kỳ vọng mức độ giảm thiểu tai nạn là 100% nhưng xe tự hành sẽ thay đổi đáng kể con số chục ngàn trường hợp tử vong xảy ra mỗi năm tại nhiều quốc gia.

Từ trước đến nay, điều khiển một chiếc xe chỉ dành cho người có thể vượt qua kì thi lấy bằng lái. Những người quá trẻ hoặc quá già, người khuyết tật hoặc có triệu chứng tâm lý không thể có được trải nghiệm này. Nhưng với xe tự hành, giao thông cá nhân sẽ mở ra tất cả các phân khúc đơn lẻ của xã hội. Việc làm chủ một thiết bị di chuyển hiện đại sẽ nằm trong tầm tay của tất cả mọi người.

Về mặt thẩm mỹ, xe tự hành sẽ thay đổi cách nhìn nhận một chiếc xe hộp, hoặc ít nhất về khía cạnh nội thất sẽ rất khác biệt. Vô-lăng, nút bấm, bàn đạp phanh, ga, cần sang số, đồng hồ tốc độ và nhiều cụm thiết bị khác bỗng trở nên thừa thãi. Những nguyên tắc cứng nhắc như ghế ngồi phải hướng về phía trước sẽ không còn hiệu lực. Nội thất xe sẽ được thiết kế phục vụ nhu cầu sinh hoạt và giải trí của người sử dụng. Chúng ta có thể ăn, ngủ, nghỉ, thậm chí tận hưởng cuộc sống ngay trong lúc xe vẫn đang chạy trên đường. Xét về tính ứng dụng, xe tự hành giống như một ngôi nhà lưu động và trở thành giải pháp cho hiện trạng đô thị chật chội ngày nay.

Rào cản từ các điểm yếu:

Với khá nhiều ưu điểm nhưng hệ thống này không thực sự hoàn hảo bởi sự phụ thuộc vào điều kiện môi trường. Tính hiệu quả chỉ phát huy tối đa khi môi trường xung quanh hội tụ đủ tiêu chuẩn nhất định. Thời tiết không thuận lợi sẽ ảnh hưởng tới độ chính xác cũng như an toàn của xe khi vận hành.

Tuy nhiên, rào cản lớn nhất không đến từ khía cạnh kỹ thuật mà là luật pháp. Ví dụ như khi tai nạn xảy ra, thật khó xác định lỗi thuộc về người ngồi trên xe hay không. Luật pháp dù có cải tiến vẫn chậm chạp hơn tốc độ phát triển của công nghệ. Một số tiểu bang Hoa Kỳ đã chú ý tới điều luật dành cho xe tự động và bán tự động.

Những người theo chủ nghĩa yêu thích lái xe bảo thủ sẽ không bao giờ là khách hàng tiềm năng của xe tự hành. Theo quan điểm này, lái xe là một kỹ năng cần tới sự chăm chỉ, tập trung và khéo léo và tích lũy lâu dài mới có. Cảm giác làm chủ tay lái là một trải nghiệm thú vị mà không công nghệ kỹ thuật tiên tiến nào có thể thay thế được.

Dù tiêu cực hay tích cực, xe tự lái có thể là xu thế phát triển tất yếu của vòng quay kỹ thuật tiên tiến. Công chúng sẽ không mong đợi sự xuất hiện trong vài năm tới, thậm chí cả thập kỷ tiếp theo, nhưng sẽ từng bước tiếp cận với hình thức lái xe tự động để chuẩn bị cho sự bùng nổ công nghệ trong tương lai. [2]

### **1.3 Nhiệm vụ luận văn**

Nội dung 1: Tìm hiểu lý thuyết về nhận dạng làn đường qua xử lý ảnh và thực thi chương trình trên Raspberry Pi và Tiva C

- Tìm hiểu về Raspberry Pi 4 và camera dùng cho Raspberry Pi.
- Tìm hiểu về thư viện xử lý ảnh OpenCV và các hàm hỗ trợ trong việc xử lý ảnh.
- Tìm hiểu về các chức năng cơ bản của Tiva C và hệ điều hành FreeRTOS để tích hợp vào board mạch Tiva C.

Nội dung 2: Tìm hiểu lý thuyết nhận dạng biển báo giao thông cơ bản dựa vào thuật toán học sâu và thực thi chương trình trên Raspberry Pi

- Tìm hiểu về thuật toán LeNet dùng trong nhận dạng đối tượng để áp dụng vào trong nhận dạng biển báo giao thông.
- Tìm hiểu về các phương pháp tiền xử lý ảnh để phục vụ cho nhận dạng biển báo giao thông.

Nội dung 3: Thiết kế tính năng phát hiện vật cản phía trước khi xe đang di chuyển, thực hiện chuyển làn khi các điều kiện an toàn cho phép.

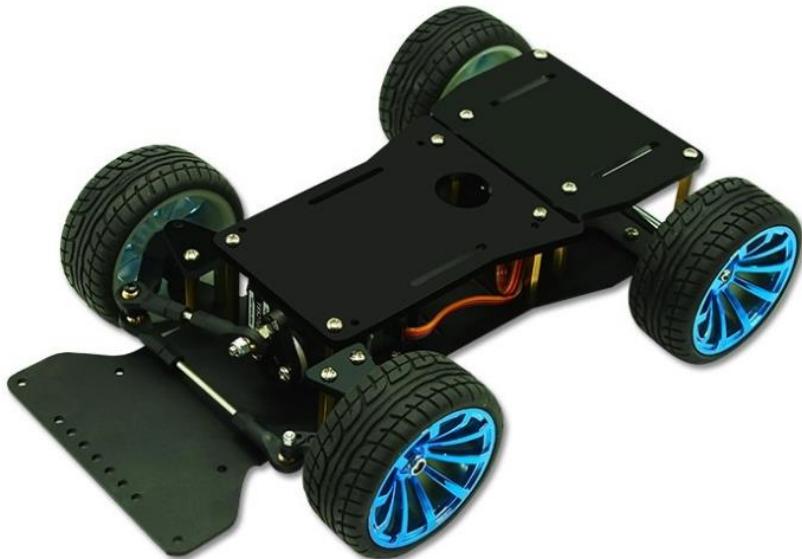
- Tìm hiểu về các cảm biến được sử dụng trong việc phát hiện vật cản, các cảm biến nhận biết điều kiện môi trường xung quanh trong việc hỗ trợ xe chuyển làn bao gồm các cảm biến hồng ngoại, cảm biến lazer, ...
- Tìm hiểu về điều khiển động cơ DC và RC servo bằng PWM.

## 2. LÝ THUYẾT

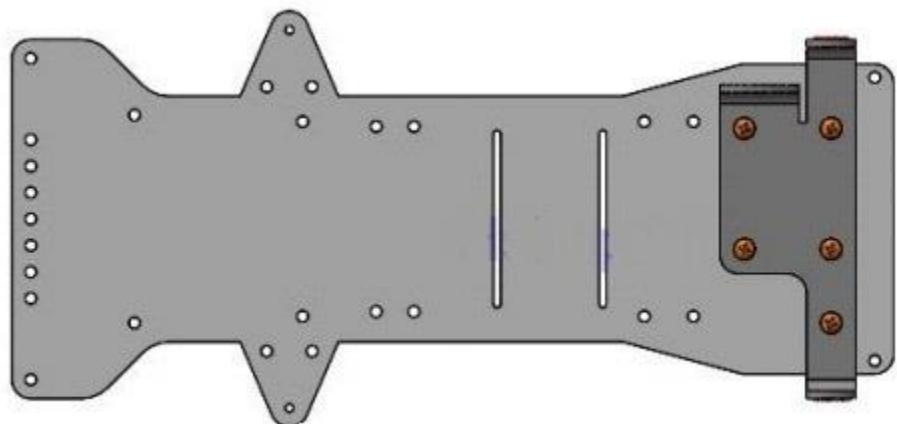
### 2.1. Các phần cứng được sử dụng

#### 2.1.1. Khung xe kim loại Racing Car RC1

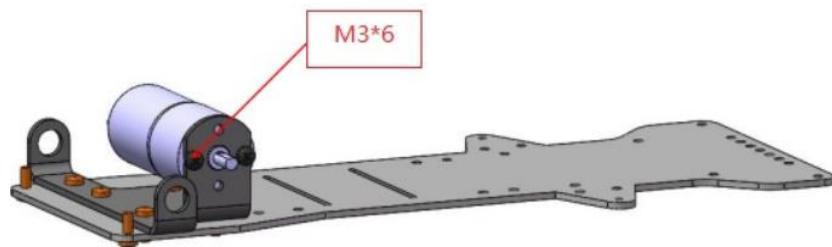
Khung xe robot kim loại Racing Car RC1 có phần thân xe làm bằng kim loại, cơ cấu chuyển hướng linh hoạt bằng RC Servo dễ sử dụng và phần dẫn động hai bánh sau được thiết kế với bánh răng dẫn động kim loại và bạc đạn làm tăng tỉ số truyền của động cơ giúp xe đạt được tốc độ và khả năng hoạt động ổn định tối đa.



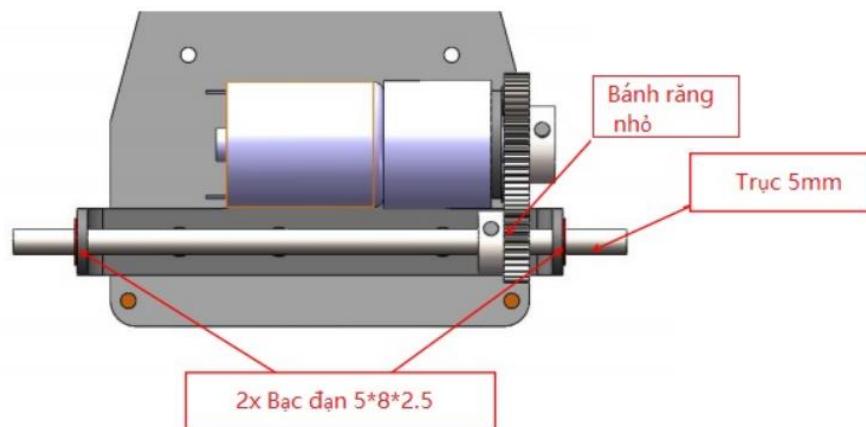
Hình 2.1. Tổng quan khung xe kim loại RC1



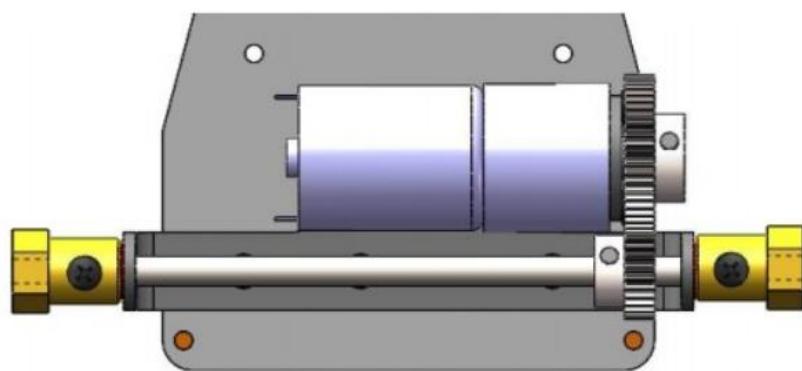
Hình 2.2. Khung xe kim loại RC1



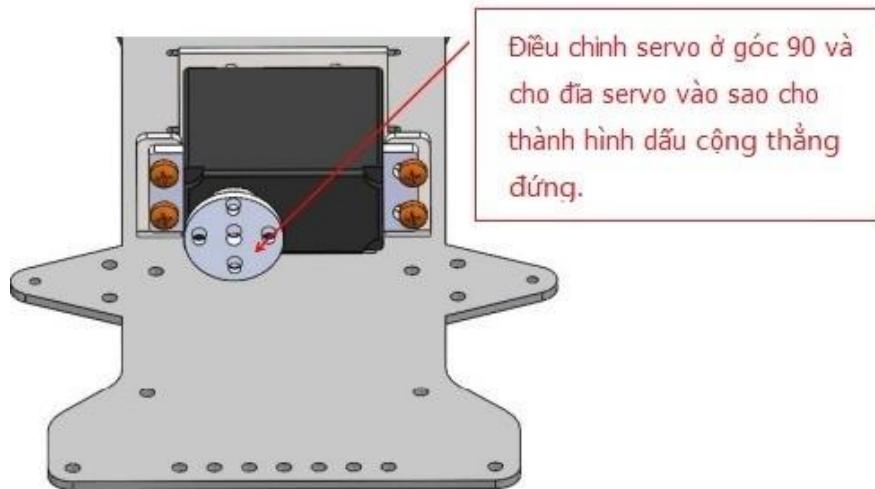
Hình 2.3. Khung xe kim loại RC1



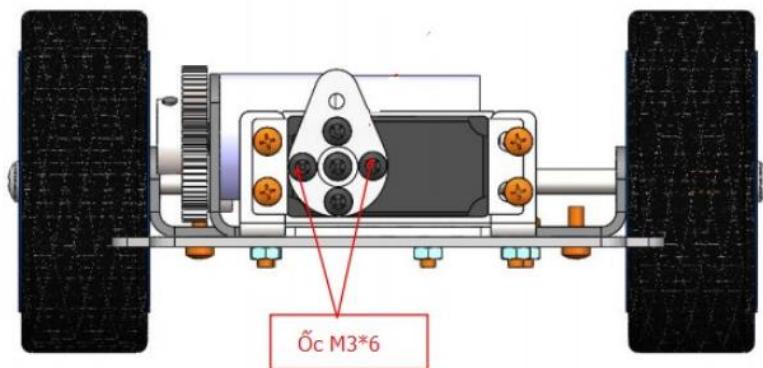
Hình 2.4. Hệ thống truyền động xe kim loại RC1



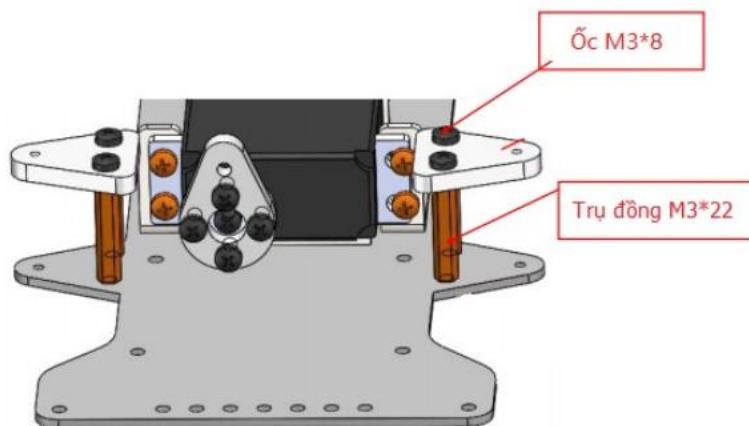
Hình 2.5. Hệ thống truyền động xe kim loại RC1



Hình 2.6. Servo xe kim loại RC1



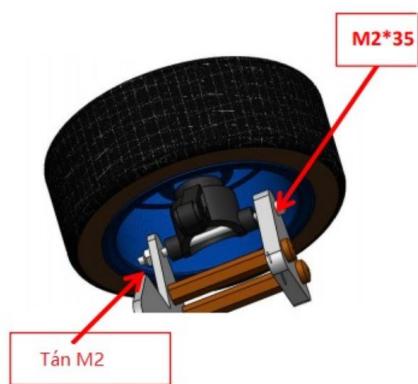
Hình 2.7. Servo xe kim loại RC1



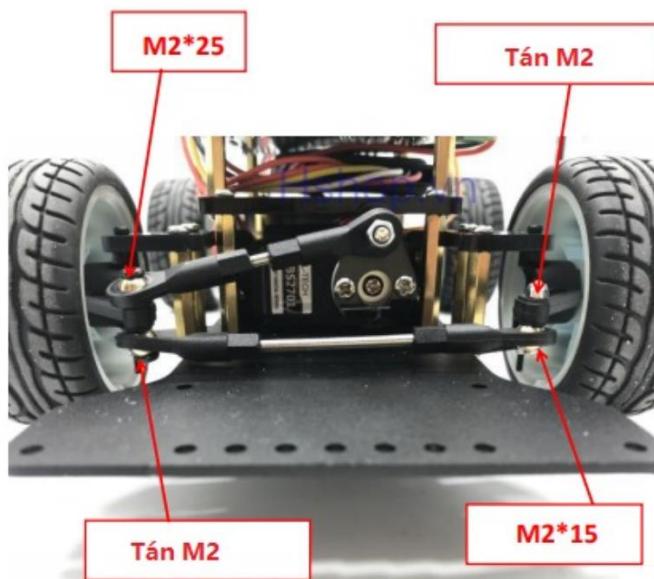
Hình 2.8. Khung xe kim loại RC1



Hình 2.9. Bánh trước xe kim loại RC1



Hình 2.10. Bánh trước xe kim loại RC1



Hình 2.11. Hệ thống lái trên xe kim loại RC1

### 2.1.2. Raspberry Pi 4

Raspberry Pi là máy tính nhúng chỉ có một board mạch kích thước chỉ bằng một thẻ tín dụng, được phát triển tại Anh bởi Raspberry Pi Foundation với mục đích ban đầu là thúc đẩy việc giảng dạy về khoa học máy tính cơ bản trong các trường học và các nước đang phát triển.

Raspberry Pi được sản xuất theo nhiều cấu hình khác nhau thông qua các thỏa thuận cấp phép sản xuất với Newark element14 (Premier Farnell), RS Components và Egoman. Các công ty này bán Raspberry Pi trực tuyến. Egoman sản xuất một phiên bản phân phối duy nhất tại Đài Loan, có thể được phân biệt với các bản Pi khác bởi màu đỏ của chúng và thiếu dấu FCC/CE. Phần cứng là như nhau đối với tất cả các nhà sản xuất.

Raspberry Pi ban đầu được dựa trên hệ thống trên một vi mạch (SoC) BCM2835 của Broadcom, bao gồm một vi xử lý ARM1176JZF-S 700 MHz, VideoCore IV GPU, và ban đầu được xuất xưởng với 256 MB RAM, sau đó được nâng cấp (model B và B+) lên đến 512 MB. Board này cũng có socket Secure Digital (SD) (model A và B) hoặc MicroSD (model A + và B +) dùng làm thiết bị khởi động và bộ lưu trữ liên tục.

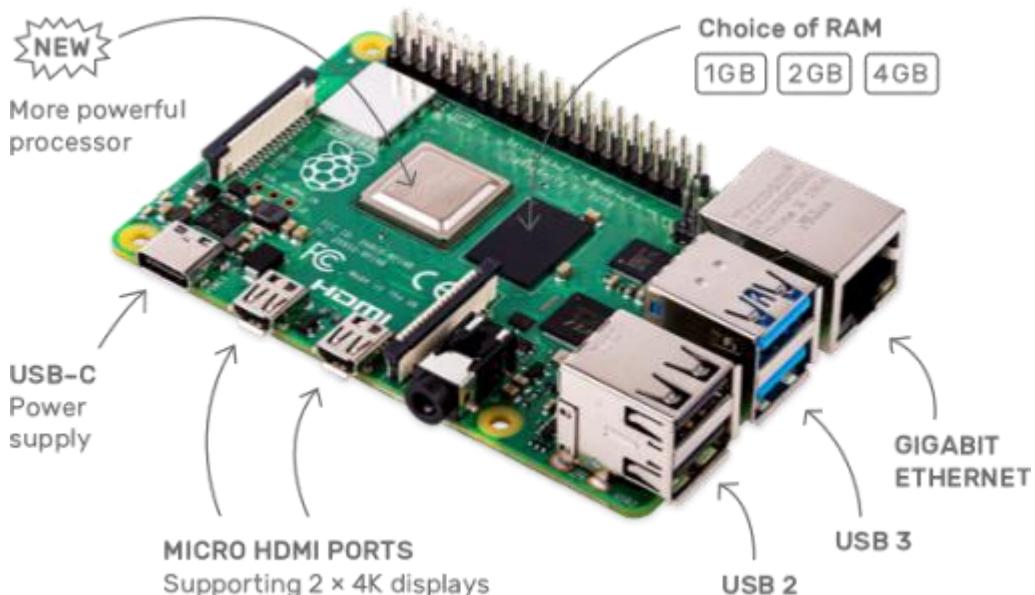
Trong năm 2014, Raspberry Pi Foundation đã phát hành mô-đun Compute, đóng gói một BCM2835 với 512 MB RAM và một flash chip eMMC vào một module để sử dụng như một phần của hệ thống nhúng.

Tổ chức này cung cấp Debian và Arch Linux ARM để người dùng tải về. Các công cụ có sẵn cho Python như là ngôn ngữ lập trình chính, hỗ trợ cho BBC BASIC (qua RISC OS image hoặc Brandy Basic clone cho Linux), C, C++, Java, Perl và Ruby.

Hầu hết các mạch Pi được sản xuất tại một nhà máy Sony tại Pencoed, Wales; một số được sản xuất tại Trung Quốc hoặc Nhật Bản.

- Thông số kỹ thuật của Raspberry Pi 4 Model B:
  - Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz.
  - Có 3 lựa chọn RAM: 1GB, 2GB hoặc 4GB LPDDR4-2400 SDRAM.

- Wifi chuẩn 2.4 GHz và 5.0 GHz IEEE 802.11ac. Bluetooth 5.0, BLE.
- Cổng mạng Gigabit Ethernet.
- 2 cổng USB 3.0 và 2 cổng USB 2.0.
- Chuẩn 40 chân GPIO, tương thích với các phiên bản trước.
- Hỗ trợ 2 cổng ra màn hình chuẩn Micro HDMI với độ phân giải lên tới 4K.
- Cổng MIPI DSI.
- Cổng MIPI CSI.
- Cổng AV 4 chân.
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode).
- OpenGL ES 3.0 graphics.
- Khe cắm Micro-SD cho hệ điều hành và lưu trữ.
- Nguồn điện DC 5V – 3A DC chuẩn USB-C.
- 5V DC via GPIO header (minimum 3A\*).
- Hỗ trợ Power over Ethernet (PoE) (yêu cầu có PoE HAT).



Hình 2.12. Mạch Raspberry Pi 4

Raspberry Pi 3 GPIO Header			
Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I <sup>2</sup> C)	DC Power 5v	04
05	GPIO03 (SCL1 , I <sup>2</sup> C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I <sup>2</sup> C ID EEPROM)	(I <sup>2</sup> C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

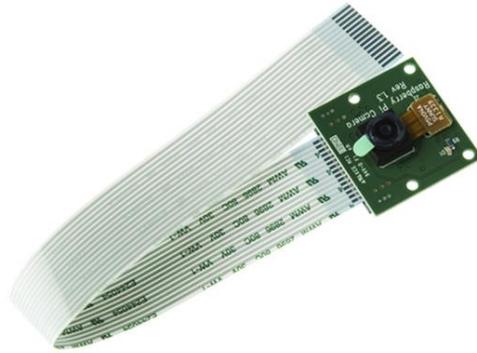
Hình 2.13. Sơ đồ GPIO Raspberry Pi

### 2.1.3. Camera Raspberry Pi V1 5MP

Camera Raspberry Pi V1 5MP là phiên bản đầu tiên của module camera cho Raspberry Pi với cảm biến OV5647 độ phân giải 5MP, sử dụng tương thích với tất cả các dòng Raspberry Pi từ trước đến nay, chất lượng hình ảnh tốt, độ phân giải cao và có khả năng quay phim ở chất lượng HD.

- Thông số kỹ thuật:
  - Module Camera V1 cho Raspberry Pi.
  - Cảm biến: OV5647.
  - Độ phân giải: 5MP.
  - Độ phân giải hình: 2592x1944 pixel.
  - Quay phim HD 1080P 30, 720P 60, VGA 640x480P 60.
  - Lens: Fixed Focus.

- Conector: Ribbon connector.
- Kích thước: 25x24x9mm



Hình 2.14 Camera Raspberry Pi V1 5MP

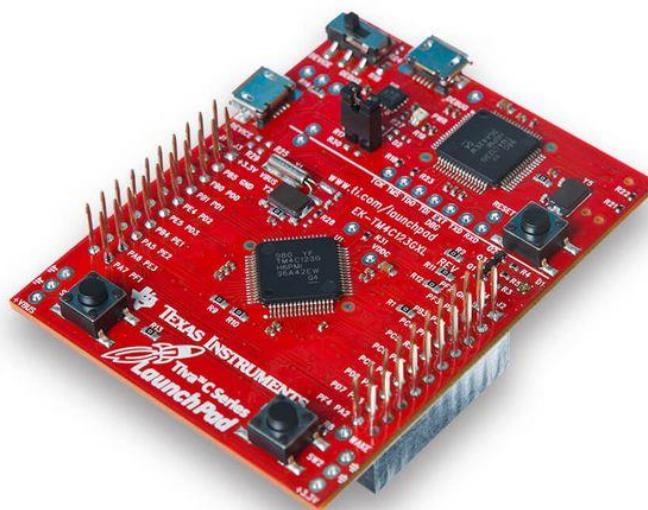
#### 2.1.4. Tiva C

Tiva Launchpad là phiên bản nâng cấp của kit Stellaris Launchpad hiện đã ngừng sản xuất, Tiva launchpad mang đầy đủ tính năng của kit Stellaris Launchpad đồng thời nâng cấp thêm 1 số ngoại vi và sửa những lỗi có trên phiên bản Stellaris Lauchpad cũ. Tiva C Series TM4C123G LaunchPad là 1 lựa chọn chi phí thấp cho vi điều khiển ARM® Cortex™ -M4 từ Texas Instruments.

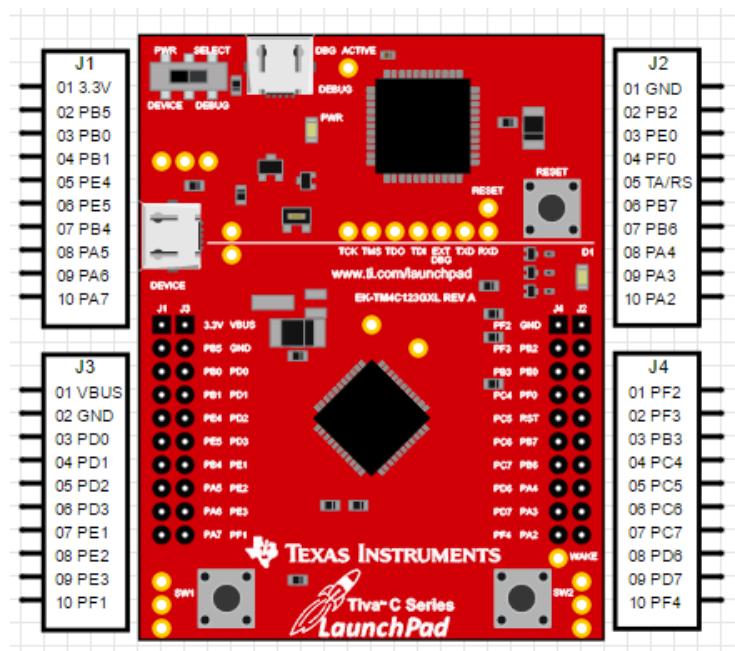
- Thông số kỹ thuật:
  - Kit bao gồm: Vi điều khiển Tiva C Series TM4C123GH6.
  - 2 cổng Micro USB: Dùng cắp nguồn và nạp chương trình.
  - USB host.
  - Led RGB.
  - Hai nút nhấn User.
  - Ngõ ra I/O được đưa ra chân sẵn.
  - Jumper debug ICDI.
  - Nút gạt chọn nguồn nuôi.
  - Chip nạp ICDI.
  - Nút nhấn Reset.

Được hỗ trợ bởi TivaWare™ cho phần mềm C Series, bao gồm Thư viện USB và thư viện các module ngoại vi hỗ trợ.

Tiva C Series TM4C123G LaunchPad BoosterPack XL ra chân mở rộng 40 chân.



Hình 2.15. Mạch Tiva C



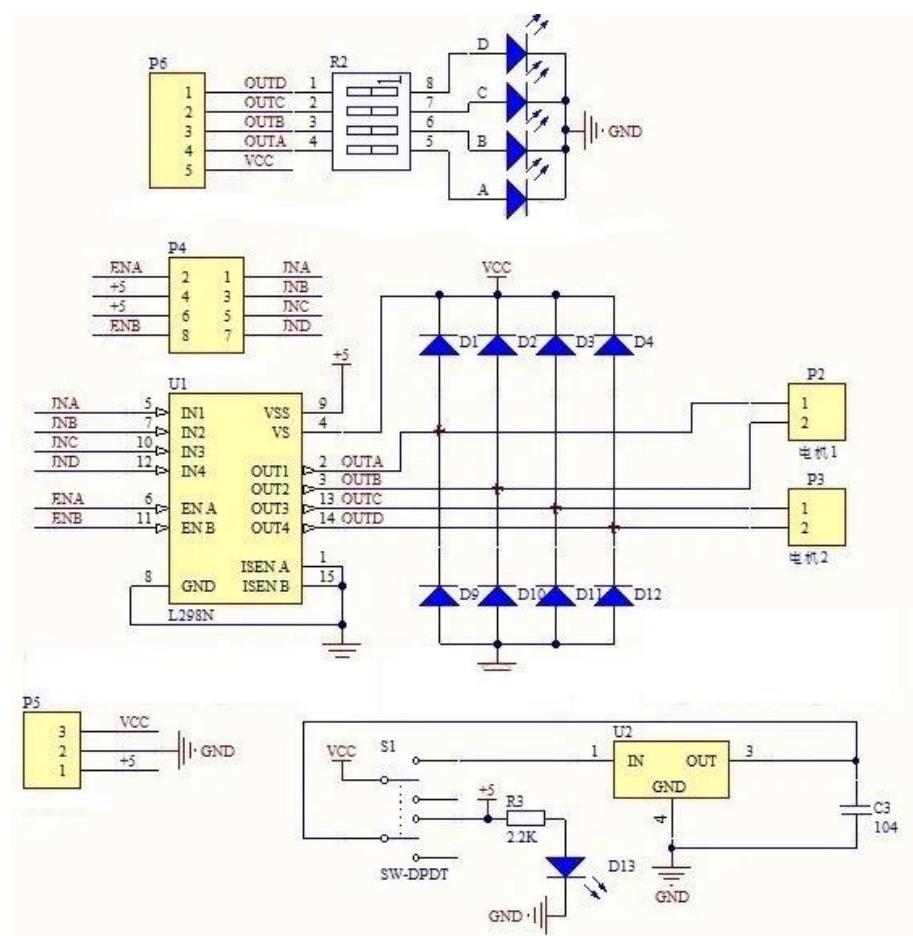
Hình 2.16. Sơ đồ GPIO Tiva C

### 2.1.5. Mạch điều khiển động cơ DC L298

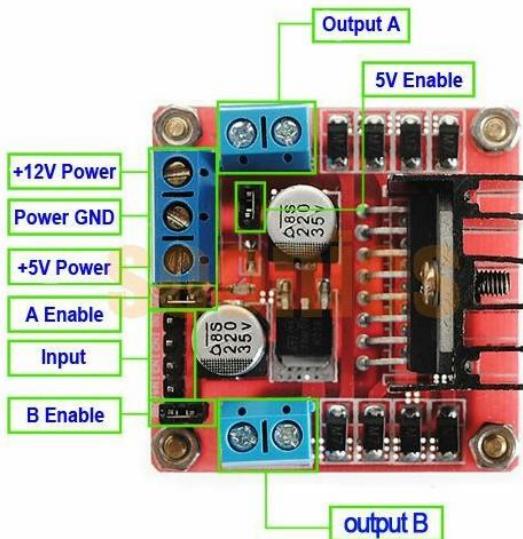
Mạch điều khiển động cơ DC L298 có khả năng điều khiển hai động cơ DC, dòng tối đa 2A mỗi động cơ, mạch tích hợp diode bảo vệ và IC nguồn 7805 giúp cấp nguồn 5VDC cho các module khác.

- **Thông số kỹ thuật:**

- IC chính: L298 - Dual Full Bridge Driver.
- Điện áp đầu vào: 5~30VDC.
- Công suất tối đa: 25W 1 cầu (công suất bằng tích dòng điện và điện áp nén áp cấp vào càng cao, dòng càng nhỏ, công suất cố định 25W).
- Dòng tối đa cho mỗi cầu H là: 2A.
- Mức điện áp logic: Low -0.3V~1.5V, High: 2.3V~Vss.



Hình 2.17. Sơ đồ mạch điều khiển động cơ DC L298



Hình 2.18. Mạch điều khiển động cơ DC L298

L298N gồm các chân 12V power, 5V power. Đây là hai chân cấp nguồn trực tiếp đến động cơ. Chúng ta có thể cấp nguồn 9-12V ở 12V. Bên cạnh đó có jumper 5V, nếu để như hình ở trên thì sẽ có nguồn 5V ra ở cổng 5V power, ngược lại thì không. Nếu để như hình thì ta chỉ cần cấp nguồn 12V ở 12V power là có 5V ở 5V power. Power GND chân này là GND của nguồn cấp cho động cơ. Hai Jump A enable và B enable dùng để kích hoạt hai kênh A và B. Gồm có bốn chân Input IN1, IN2, IN3, IN4. Chiều quay của động cơ được điều khiển bằng cách xuất các đầu ra HIGH hoặc LOW tại các chân IN<sub>x</sub>. Output A: nối với động cơ A. chú ý chân +, -. Nếu nối ngược thì động cơ sẽ chạy ngược. Và nếu nối động cơ bước, phải đấu nối các pha cho phù hợp. Board này gồm 2 phần điều khiển động cơ. Và có thể điều khiển cho 1 động cơ bước 6 dây hoặc 4 dây. Chiều quay của động cơ được điều khiển bằng cách xuất các đầu ra HIGH hoặc LOW tại các chân IN<sub>x</sub>. Với động cơ A: Logic HIGH ở IN1 và IN2 Logic LOW sẽ làm động cơ quay 1 hướng nếu đặt mức logic ngược lại sẽ làm động cơ quay theo hướng khác. Đây là cách động cơ chỉ quay hết công suất. Nếu muốn thay đổi tốc độ của nó, ta cần phải bằng các chân có hỗ trợ PWM. Giả sử, chân IN1 là chân OutA.1, chân IN2 là chân OutA.2. Cấp cực dương vào IN1, cực âm vào IN2 thì motor quay một chiều (chiều 1). Cấp cực âm vào IN1, cực dương vào IN2 thì motor quay chiều còn lại (chiều 2). Cực dương ở đây là điện thế 5V, cực âm ở đây là điện thế 0V. Hiện điện thế được tính là điện thế ở IN1 trừ hiệu điện thế IN2. Giả sử, hiệu điện thế 5V sẽ là mạnh nhất trong

việc điều khiển động cơ. Như vậy, chỉ cần hạ hiệu điện thế xuống là động cơ sẽ bị yếu đi. Nếu hiệu điện thế bé hơn 0 thì động cơ sẽ đảo chiều.

#### 2.1.6. Động Cơ DC Servo GM25-370 DC Geared Motor

Động Cơ DC Servo GM25-370 DC Geared Motor được tích hợp thêm Encoder hai kênh AB giúp đọc và điều khiển chính xác vị trí, chiều quay của động cơ trong các ứng dụng cần độ chính xác cao: điều khiển PID, Robot tự hành....

Động Cơ DC Servo GM25-370 DC Geared Motor có cấu tạo bằng kim loại cho độ bền và độ ổn định cao, được sử dụng trong các mô hình robot, xe, thuyền,..., hộp giảm tốc của động cơ có nhiều tỉ số truyền giúp ta dễ dàng lựa chọn giữa lực kéo và tốc độ (lực kéo càng lớn thì tốc độ càng chậm và ngược lại), động cơ sử dụng nguyên liệu chất lượng cao (lõi dây đồng nguyên chất, lá thép 407, nam châm từ tính mạnh,...) cho sức mạnh và độ bền vượt trội hơn các loại giá rẻ trên thị trường hiện nay (sử dụng lõi dây nhôm, nam châm từ tính yếu).

- **Thông số kỹ thuật:**
  - Điện áp sử dụng: 12VDC.
  - Đường kính: 25mm.
  - Encoder: Cảm biến từ trường Hall, có hai kênh AB lệch nhau giúp xác định chiều quay và vận tốc của động cơ, đĩa Encoder trả ra 11 xung/1 kênh/ 1 vòng (nếu đo tín hiệu đồng thời của cả hai kênh sẽ thu được tổng 22 xung / 1 vòng quay của Encoder).

Cách tính số xung của mỗi kênh trên 1 vòng quay của trực chính động cơ = Tỉ số truyền x số xung của Encoder, ví dụ tỉ số 150:1 thì số xung Encoder trả ra cho 1 vòng quay của trực chính động cơ sẽ là  $11 \times 150 = 1650$  xung / 1 kênh. Điện áp cấp cho Encoder hoạt động: 3.3~5VDC, lưu ý cấp quá áp hoặc ngược chiều sẽ làm cháy Encoder ngay lập tức.

- **Loại 12VDC 39RPM:**
  - Tỉ số truyền 217:1 (động cơ quay 217 vòng trực chính hộp giảm tốc quay 1 vòng).
  - Dòng không tải: 150mA.

- Dòng chịu đựng tối đa khi có tải: 750mA.
- Tốc độ không tải: 39RPM (39 vòng 1 phút).
- Tốc độ chịu đựng tối đa khi có tải: 32RPM (32 vòng 1 phút).
- Lực kéo Moment định mức: 8Kg.CM.
- Lực leo Moment tối đa: 9Kg.CM.
- Chiều dài hộp số L: 25mm.
- Số xung Encoder mỗi kênh trên 1 vòng quay trực chính:  $11 \times 217 = 2387$  xung.

Sơ đồ chân của động cơ:

1. M1 - Đỏ: Dây cấp nguồn cho động cơ.
2. GND - Đen: Dây cấp nguồn cho Encoder, 0VDC.
3. C1/A - Vàng: Kênh trả xung A
4. C2/B - Xanh lá: Kênh trả xung B
5. VCC - Xanh dương: Dây cấp nguồn cho Encoder 3.3~5VDC
6. M2 - Trắng: Dây cấp nguồn cho động cơ.



Hình 2.19. Cấu tạo động Cơ DC Servo GM25-370

### 2.1.7. Động cơ RC Servo MG996

Động cơ RC Servo MG996 là loại thường được sử dụng nhiều nhất trong các thiết kế robot hoặc dẫn hướng xe. Động cơ RC Servo MG996 có lực kéo mạnh, các khớp và bánh răng được làm hoàn toàn bằng kim loại nên có độ bền cao, động cơ được tích hợp sẵn driver điều khiển động cơ bên trong theo cơ chế phát xung - quay góc.

- Thông số kỹ thuật:
  - Chủng loại: Analog RC Servo.
  - Điện áp hoạt động: 4.8-6.6VDC.
  - Kích thước: 40mm x 20mm x 43mm.
  - Trọng lượng: 55g.
  - Lực kéo: 3.5 Kg-cm (180.5 ozin) at 4.8V-1.5A.
  - Tốc độ quay: 0.17sec / 60 degrees (4.8V không tải).



Hình 2.20. Động cơ RC Servo

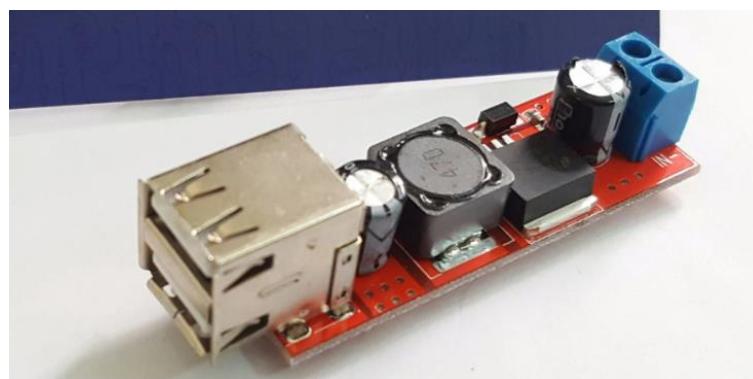
### 2.1.8. Mạch giảm áp DC 5V 3A

Mạch giảm áp DC 5V 3A 2 cổng USB Charge Module được thiết kế để có thể giảm áp từ các nguồn DC, ac-quy xuống 5VDC cổng đầu ra USB. Mạch giảm áp DC 5V 3A 2 cổng USB Charge Module có thiết kế nhỏ gọn, chất lượng tốt, đầu ra 2 cổng USB, nguồn DC đầu vào tối đa lên đến 36V, nguồn đầu ra luôn cố định 5V USB tối đa 3A, hiệu suất chuyển đổi năng lượng rất cao lên đến 95% (không bị hao nguồn, không

nóng).

\* Thông số kỹ thuật:

- Mạch giảm áp xung hiệu suất cao tần số 125Khz.
- Đầu vào tối đa: 6 - 36VDC.
- Đầu ra cố định: 5VDC - 3A (liên tục tối đa 10W).
- Đầu ra có dạng USB, 2 cổng USB.
- Hiệu suất chuyển đổi: 95%.
- Kích thước: 59x21x17mm.



*Hình 2.21. Mạch giảm áp DC 5V 3A*

#### 2.1.9. Mạch giảm áp DC LM2596

Mạch giảm áp DC LM2596 có hiển thị với ba led 7 đoạn hiển thị nguồn ngõ ra/vào tiện dụng cho quá trình sử dụng (hiển thị giá trị điện áp sai số trong khoảng  $\pm 0,1\text{V}$ ), mạch cho dòng điện ngõ ra lên đến 3A. Mạch còn có nút nhấn để chuyển đổi đo áp ngõ vào và ngõ ra.

- Thông số kỹ thuật:

- Dùng IC LM2596 với tần số lên đến 150Khz.
- Có nút nhấn chuyển chế độ hiển thị ngõ ra/vào.
- Điện áp đầu vào: Từ 4V đến 30V.
- Điện áp đầu ra: Điều chỉnh được trong khoảng 1.5V đến 29V.
- Dòng ngõ ra tối đa là 3A.
- Công suất: 15W.
- Kích thước: 66 x 35mm.



*Hình 2.22. Mạch giảm áp LM2596*

Ta cần cấp nguồn đầu vào cho mạch này tại header có ký hiệu VIN+ và VIN- tương ứng với VCC và GND ở đầu ra cũng được kết nối với header có kí hiệu là VOUT+ và VOUT-. Mạch được trang bị LED 7 đoạn để hiển thị điện áp vào và điện áp ra, ta có thể chọn giá trị điện áp vào hay ra để hiện thị bằng nút nhấn, để điều chỉnh điện áp ngõ ra, ta chỉnh biến trở trên mạch.

#### 2.1.10. Cảm Biến Khoảng Cách Hồng Ngoại Analog SHARP GP2Y0A21YK0F



*Hình 2.23. Cảm biến khoảng cách hồng ngoại analog Sharp*

Cảm biến GP2Y0A21YK0F sản xuất bởi hãng SHARP, được sử dụng để đo khoảng cách bằng tia hồng ngoại với dạng tín hiệu trả về là analog tương ứng theo khoảng cách nên có thể biết chính xác khoảng cách đến vật thể cần đo, cảm biến có thể đo được trong khoảng 10 ~ 80cm.

Cảm biến khoảng cách hồng ngoại Analog SHARP GP2Y0A21YK0F có độ ổn định cao, chống nhiễu tốt, kích thước nhỏ gọn, phù hợp với vô số ứng dụng khác nhau: robot dò đường, đo khoảng cách, tránh vật cản...

- Thông số kỹ thuật:
  - Model: GP2Y0A21YK0F
  - Điện áp sử dụng: 4.5 ~ 5.5VDC.
  - Dòng sử dụng trung bình: 30 mA.
  - Khoảng cách đo: 10 ~ 80cm.
  - Dạng tín hiệu trả về: analog voltage.

GP2Y0A21YK0F là một cảm biến đo khoảng cách, bao gồm PSD (đầu thu), IRED (diode phát hồng ngoại) và mạch xử lý tín hiệu. Sự đa dạng của độ phản xạ của vật thể, nhiệt độ môi trường và thời gian hoạt động không bị ảnh hưởng đến việc phát hiện khoảng cách vì áp dụng phương pháp tam giác. Thiết bị này xuất điện áp tương ứng với khoảng cách phát hiện. Vì vậy, cảm biến này cũng có thể được sử dụng như một cảm biến tiệm cận.

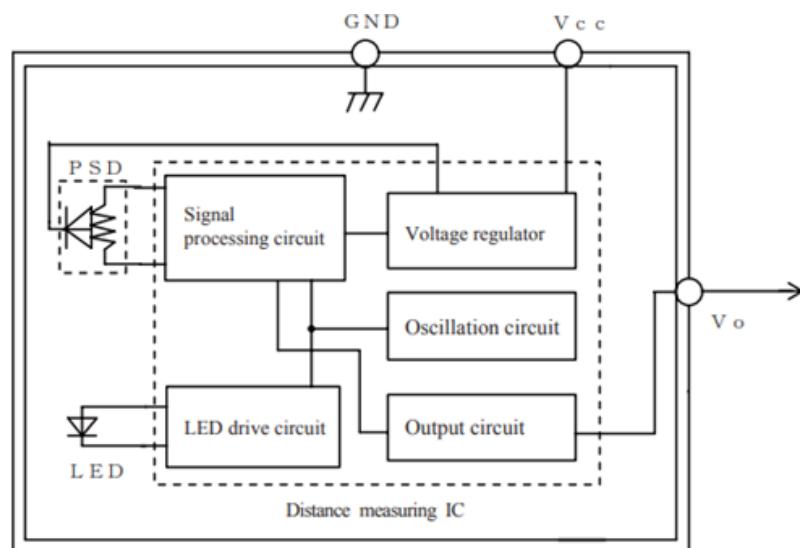
#### Ứng dụng:

- Công tắc cảm ứng (Thiết bị vệ sinh, Điều khiển chiếu sáng, v.v.).

- Robot lau chùi.

- Cảm biến để tiết kiệm năng lượng (ATM, Máy bán hàng tự động).

#### 4. Thiết bị giải trí (Robot, máy chơi trò chơi điện tử).



Hình 2.24. Sơ đồ khối GP2Y0A21YK0F

*Bảng 2.1 Thông số hoạt động của cảm biến*(T<sub>a</sub>=25°C, V<sub>CC</sub>=5V)

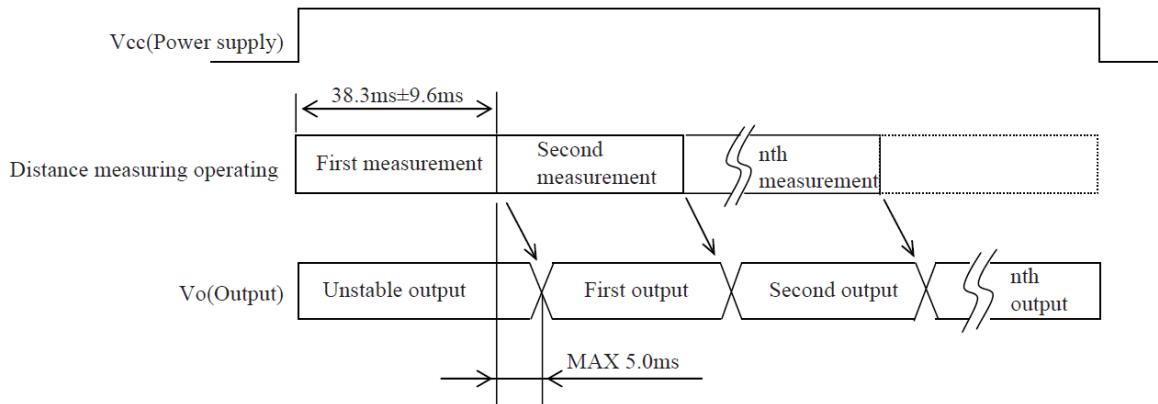
Parameter	Symbol	Rating	Unit
Supply voltage	V <sub>CC</sub>	-0.3 to +7	V
Output terminal voltage	V <sub>O</sub>	-0.3 to V <sub>CC</sub> +0.3	V
Operating temperature	T <sub>opr</sub>	-10 to +60	°C
Storage temperature	T <sub>stg</sub>	-40 to +70	°C

*Bảng 2.2 Đặc tính quang – điện của cảm biến*(T<sub>a</sub>=25°C, V<sub>CC</sub>=5V)

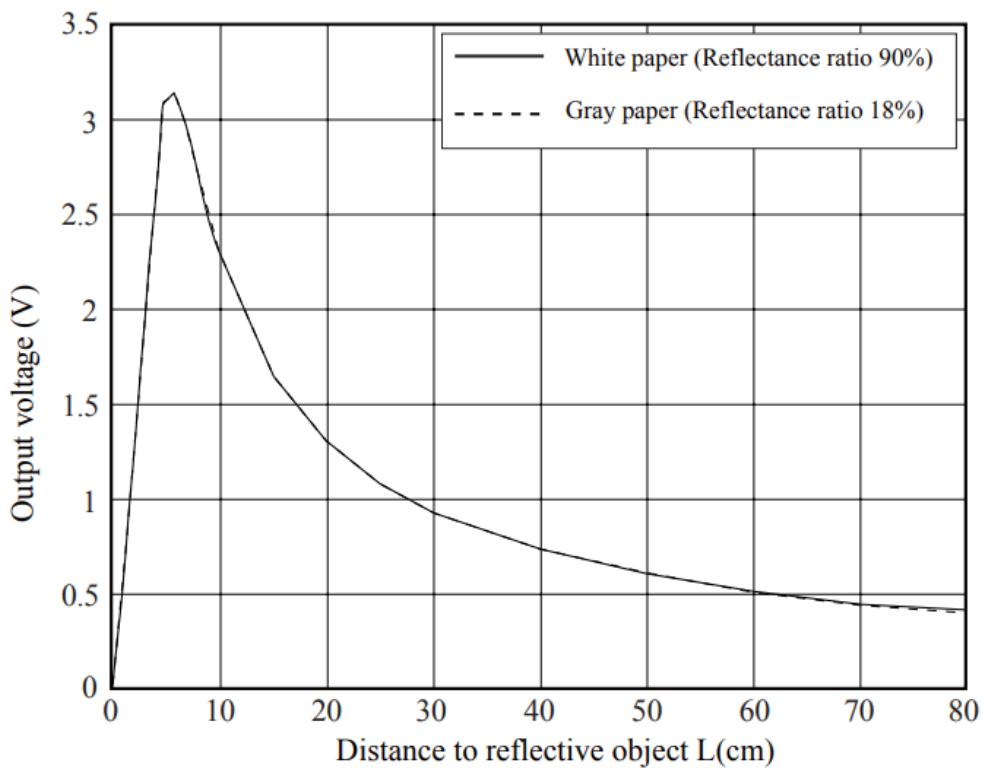
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Average supply current	I <sub>CC</sub>	L=80cm (Note 1)	—	30	40	mA
Distance measuring	ΔL	(Note 1)	10	—	80	cm
Output voltage	V <sub>O</sub>	L=80cm (Note 1)	0.25	0.4	0.55	V
Output voltage differential	ΔV <sub>O</sub>	Output voltage difference between L=10cm and L=80cm (Note 1)	1.65	1.9	2.15	V

*Bảng 2.3 Điện áp cung cấp khuyến dùng*

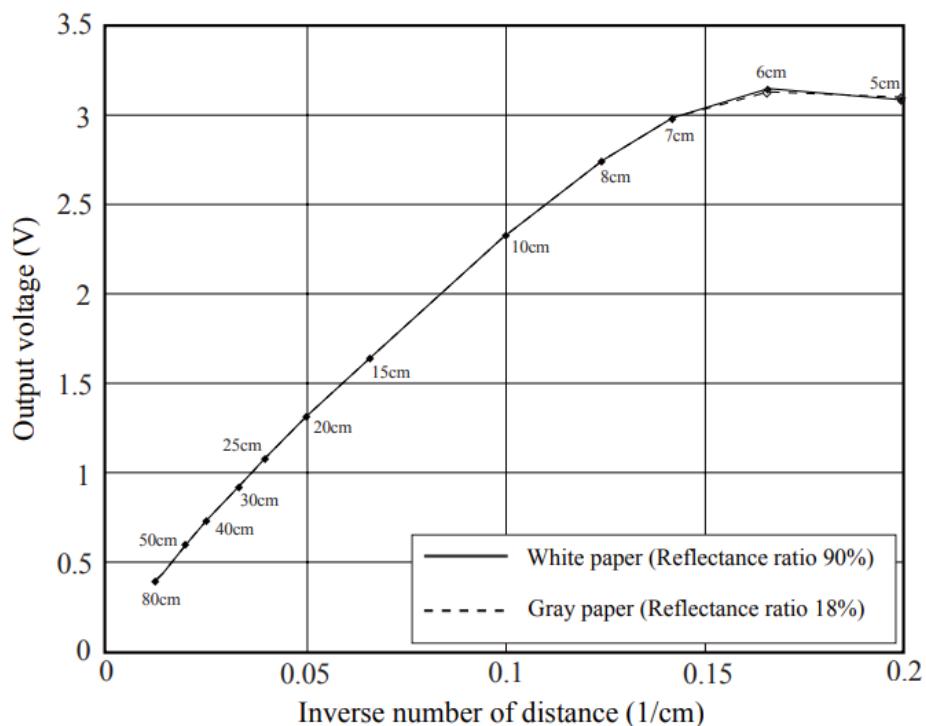
Parameter	Symbol	Rating	Unit
Supply voltage	V <sub>CC</sub>	4.5 to 5.5	V

*Hình 2.25. Biểu đồ thời gian GP2Y0A21YK0F*

Dựa vào biểu đồ đáp ứng thời gian như trên, ta có thể thấy chu kì trung bình để cảm biến hoạt động ổn định là 38.3ms và thời gian trễ ở ngõ ra so với ngõ vào tối đa là 5ms.

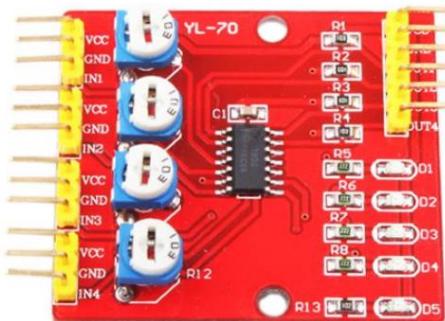


Hình 2.26. Biểu đồ khoảng cách và điện áp GP2Y0A21YK0F

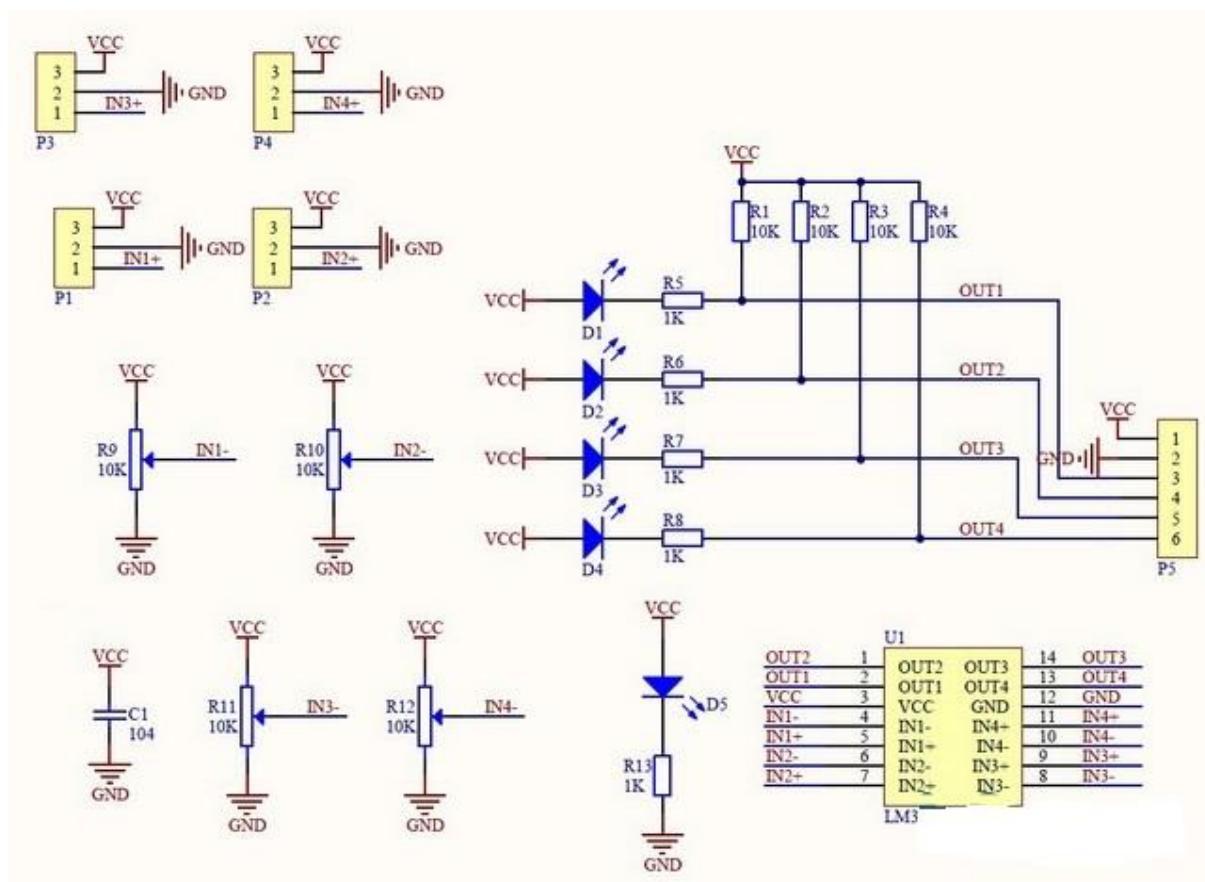


Hình 2.27. Biểu đồ khoảng cách và điện áp GP2Y0A21YK0F

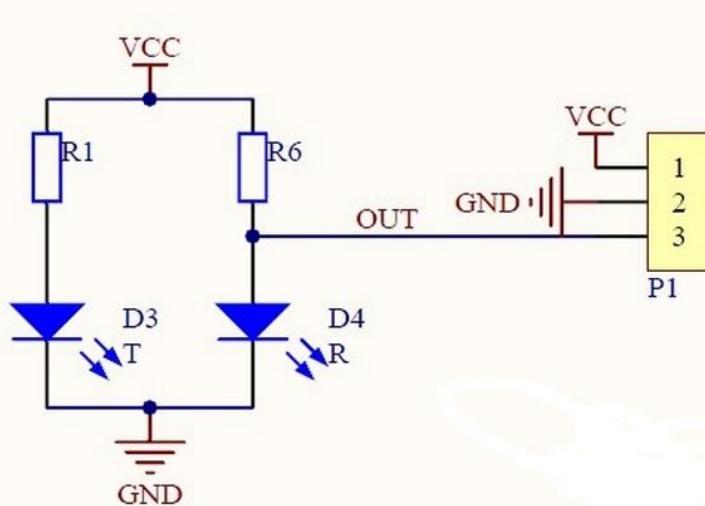
Ta có thể dựa vào một trong hai biểu đồ trên để xác định được điện áp trả về từ ngõ ra của cảm biến.



Hình 2.28 Mạch chuyển tín hiệu analog sang digital



Hình 2.29 Sơ đồ nguyên lý mạch chuyển tín hiệu tương tự sang tín hiệu số



*Hình 2.30 Sơ đồ nguyên lý mạch chuyển tín hiệu tương tự sang tín hiệu số*

Mạch chuyển đổi trên dùng để chuyển đổi tín hiệu analog trả về từ cảm biến hồng ngoại, ngõ ra của mạch này sẽ là tín hiệu 1 hoặc 0, tùy theo mức ngưỡng đã xác định trước, có thể điều chỉnh thông qua biến trở. Mạch này có 4 kênh đầu vào là analog, mỗi kênh sẽ có ba chân kết nối, hai chân cấp nguồn 5V(VCC và GND) cho cảm biến hoạt động, và một chân nối IN<sub>x</sub> nối với ngõ ra của cảm biến, IC trên mạch này sẽ so sánh điện áp mà cảm biến trả về với điện áp ngưỡng mà xuất ra tín hiệu 0 hoặc 1 tại ngõ ra OUT<sub>x</sub>. Ta có thể dựa vào các LED (D<sub>x</sub>) có trên mạch để xem sự hoạt động của mạch này, nếu điện áp trả về của cảm biến lớn hơn mức ngưỡng thì LED tại kênh đó sẽ tắt, ngược lại LED sẽ sáng.

#### 2.1.11. Cảm biến thân nhiệt chuyển động PIR HC-SR501

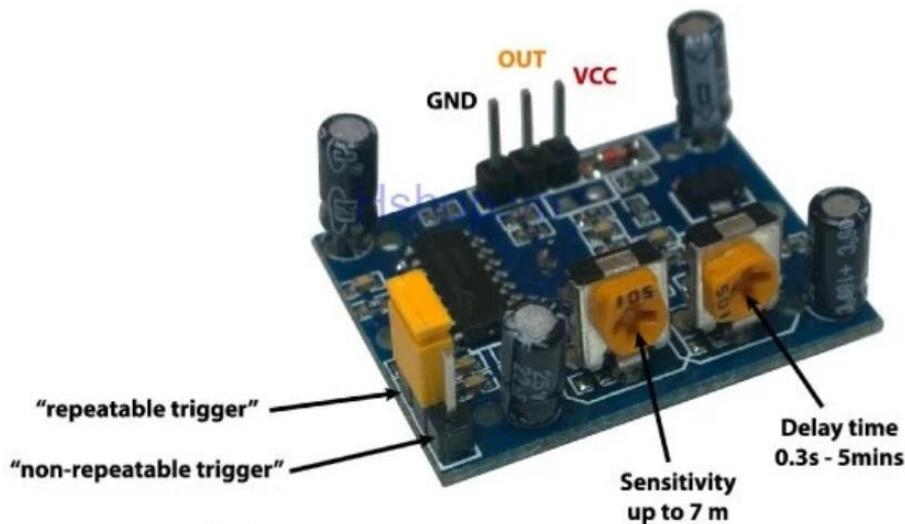
Cảm biến thân nhiệt chuyển động PIR (Passive infrared sensor) HC-SR501 được sử dụng để phát hiện chuyển động của các vật thể phát ra bức xạ hồng ngoại (con người, con vật, các vật phát nhiệt,...), cảm biến có thể chỉnh được độ nhạy để giới hạn khoảng cách bắt xa gần cũng như cường độ bức xạ của vật thể mong muốn, ngoài ra cảm biến còn có thể điều chỉnh thời gian kích trễ (giữ tín hiệu bao lâu sau khi kích hoạt) qua biến trờ tích hợp sẵn.

- Thông số kỹ thuật:
  - Phạm vi phát hiện: góc 360 độ hình nón, độ xa tối đa 6m.
  - Nhiệt độ hoạt động: 32-122 ° F (50 ° C).

- Điện áp hoạt động: DC 3.8V - 5V.
- Mức tiêu thụ dòng:  $\leq 50 \mu\text{A}$
- Thời gian báo: 30 giây có thể tùy chỉnh bằng biến trở.
- Độ nhạy có thể điều chỉnh bằng biến trở.



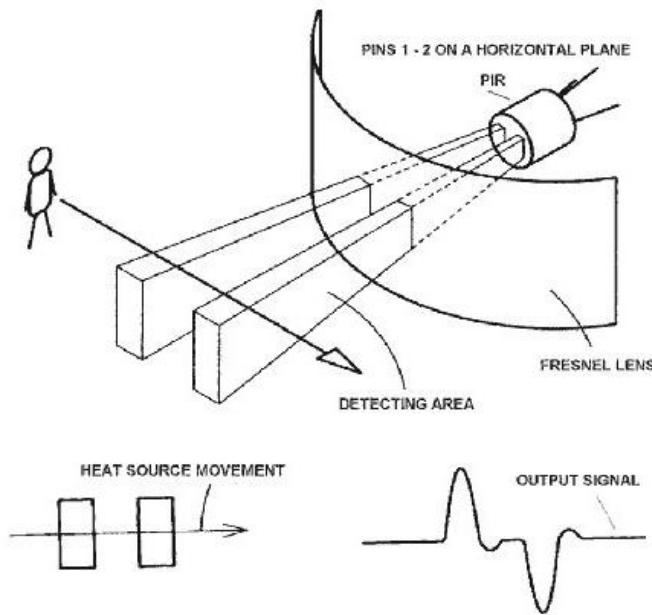
Hình 2.31. Cảm biến thân nhiệt chuyển động HC-SR501



Hình 2.32. Cảm biến thân nhiệt chuyển động HC-SR501

Nguyên tắc hoạt động:

Cơ chế hoạt động của cảm biến hồng ngoại PIR: là cảm biến thu tia hồng ngoại được phát ra từ các vật thể phát ra tia hồng ngoại như cơ thể con người (hay nguồn nhiệt bất kì).



*Hình 2.33 Nguyên lý hoạt động cảm biến PIR*

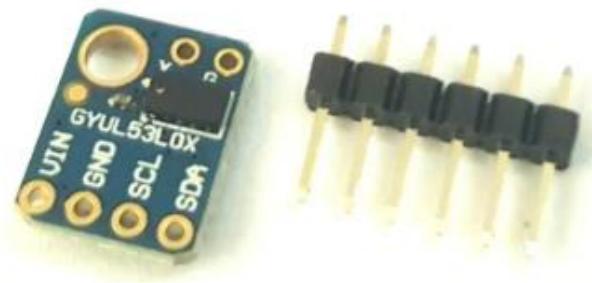
Các cảm biến PIR luôn có sensor (mắt cảm biến) với 2 đơn vị (element). Chắn trước mắt sensor là một lăng kính (thường làm bằng nhựa), chế tạo theo kiểu lăng kính fresnel. Lăng kính fresnel này có tác dụng chặn lại và phân thành nhiều vùng (zone) cho phép tia hồng ngoại đi vào mắt sensor. Hai đơn vị của mắt sensor có tác dụng phân thành 2 điện cực. Một cái là điện cực dương (+) và cái kia là âm (-). Khi 2 đơn vị này được tuần tự kích hoạt (cái này xong rồi mới đến cái kia) thì sẽ sinh ra một xung điện, xung điện này kích hoạt sensor. Ngoài ra, trên mạch HC – SR501 ta có thể điều chỉnh được khoanh cách phát hiện vật chuyển động và thời gian trễ bằng hai biến trở.

#### 2.1.12. Cảm biến khoảng cách VL53L0X Laser Distance ToF Sensor GY-53L0

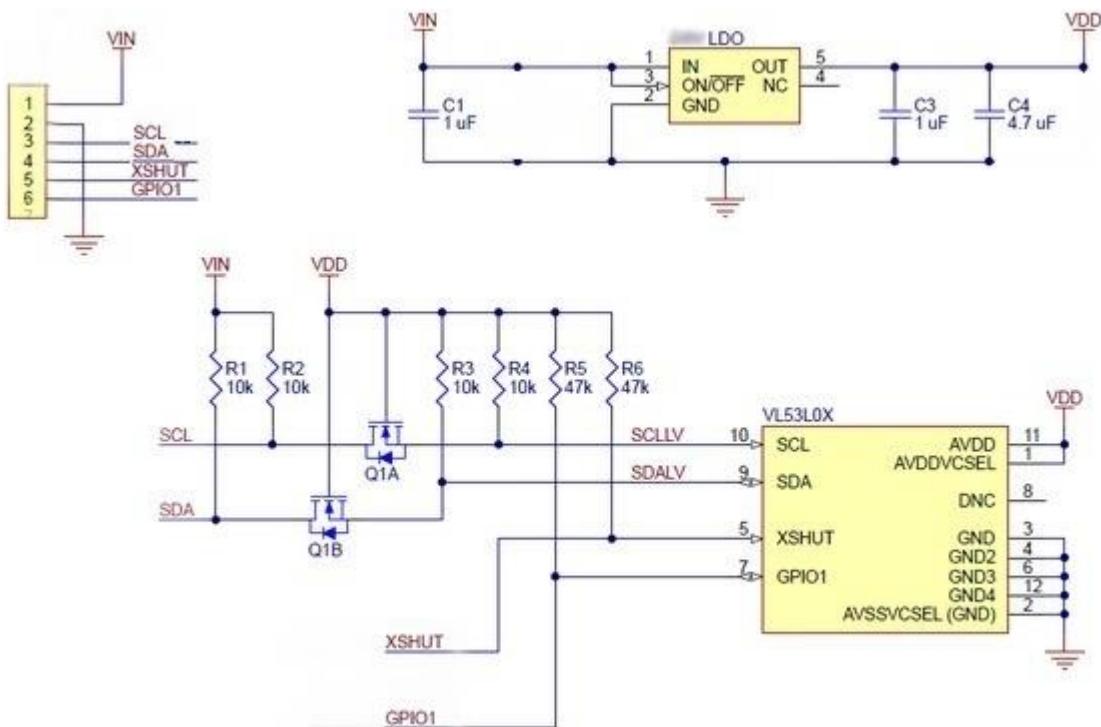
Cảm biến khoảng cách VL53L0X Laser Distance ToF Sensor GY-53L0 sử dụng tia Laser để đo khoảng cách đến vật thể với độ chính xác và độ ổn định cao, thích hợp cho các ứng dụng Robot tránh vật cản, đo khoảng cách, đặc biệt là các ứng dụng Radar bằng tia Laser (Lidar) trong các xe tự hành, Smart Car...

- Thông số kỹ thuật:
  - Model: VL53L0X Laser Distance ToF Sensor GY-53L0
  - Điện áp sử dụng: 2.8~5VDC
  - Dòng sử dụng trung bình: lúc hoạt động 20mA, lúc nghỉ 6uA.
  - Phương pháp đo khoảng cách: Tia Laser.

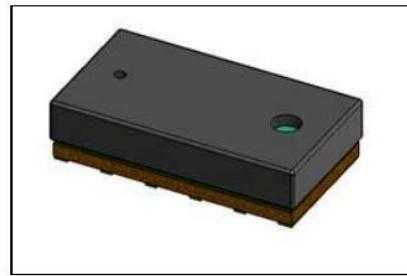
- Khoảng cách đo trung bình:
- Tối thiểu: 2cm.
- Trong nhà: Nền màu trắng: 200cm+, các màu khác: 80cm.
- Ngoài trời: Nền màu trắng: 80cm, các màu khác: 50cm.
- Dạng tín hiệu trả về: I2C mức TTL 3.3~5VDC.



Hình 2.34. Cảm biến khoảng cách VL53L0X



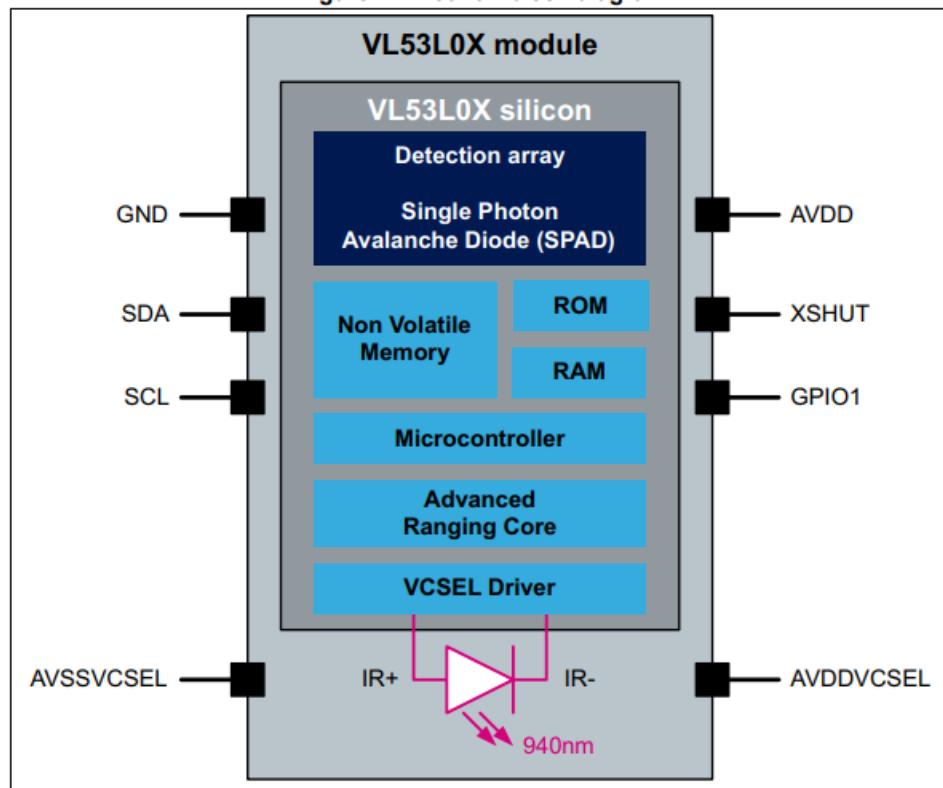
Hình 2.35. Sơ đồ cảm biến khoảng cách VL53L0X



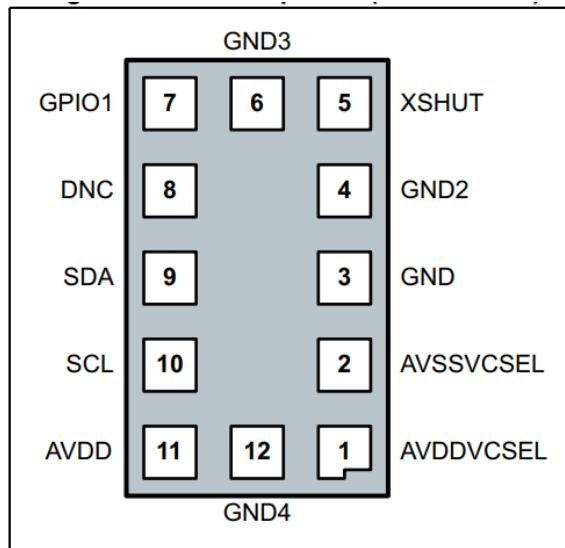
Hình 2.36. Cảm biến khoảng cách VL53L0X

Bảng 2.4 Thông số kỹ thuật của cảm biến

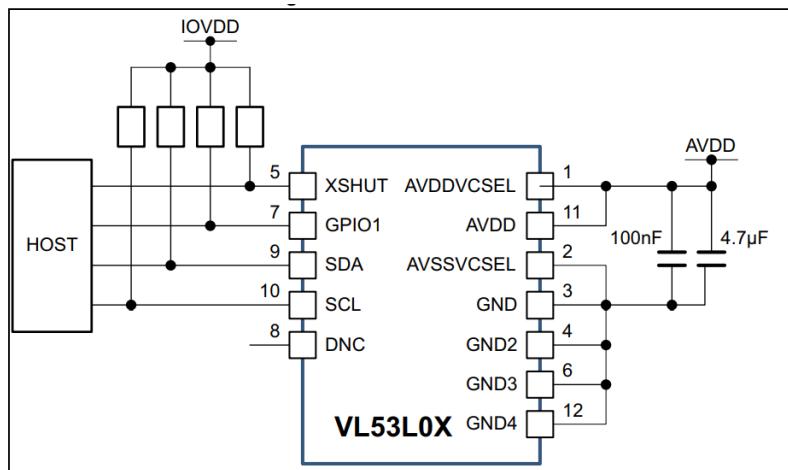
Feature	Detail
Package	Optical LGA12
Size	4.40 x 2.40 x 1.00 mm
Operating voltage	2.6 to 3.5 V
Operating temperature:	-20 to 70°C
Infrared emitter	940 nm
I <sup>2</sup> C	Up to 400 kHz (FAST mode) serial bus Address: 0x52



Hình 2.37. Sơ đồ khói VL53L0X



Hình 2.38. Sơ đồ PINOUT VL53L0X



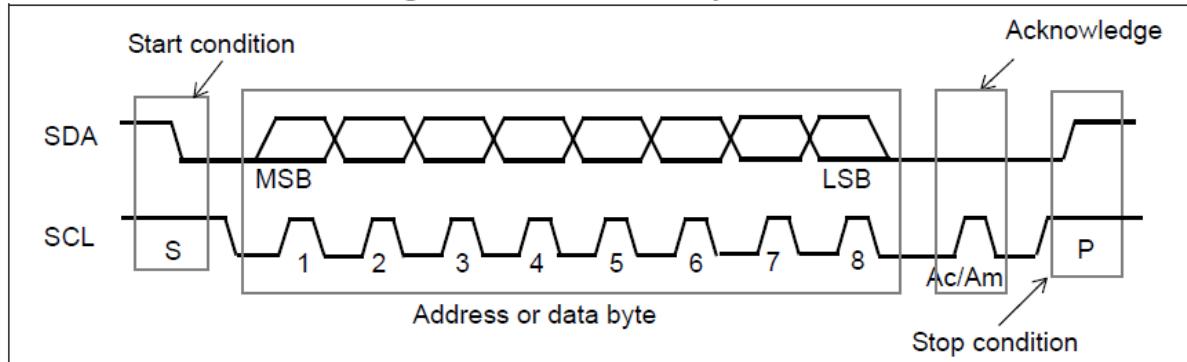
Hình 2.39. Sơ đồ schematic VL53L0X

Nguyên lý hoạt động:

Phần này chỉ định giao diện điều khiển. Giao diện I2C sử dụng hai tín hiệu: đường dữ liệu nối tiếp (SDA) và đường xung clock nối tiếp (SCL). Mỗi thiết bị kết nối với bus đang sử dụng một địa chỉ duy nhất và tồn tại một mối quan hệ chủ / tớ đơn giản.

Cả hai đường SDA và SCL đều được kết nối với điện áp cung cấp dương bằng cách sử dụng điện trở kéo lên trên mạch chủ. Các đường truyền chỉ được điều khiển tích cực ở mức thấp. Đường truyền ở mức cao xảy ra khi các đường dây thả nổi và điện trở kéo lên kéo các đường dây lên mức cao. Khi không có dữ liệu được truyền cả hai đường cao.

Việc tạo tín hiệu xung nhịp (SCL) được thực hiện bởi mạch chủ. Thiết bị chính bắt đầu truyền dữ liệu. Bus I2C trên VL53L0X có tốc độ tối đa là 400 kbytes/s và sử dụng địa chỉ thiết bị là 0x29.



Hình 2.40 Giao thức truyền dữ liệu

Thông tin được đóng gói trong các gói 8bit (1 byte) luôn sau là một bit xác nhận, Ac cho VL53L0X xác nhận và Am cho mạch chủ xác nhận (chủ bus chủ). Dữ liệu nội bộ được tạo ra bằng cách lấy mẫu SDA ở một cạnh lên của SCL. Dữ liệu bên ngoài phải ổn định trong thời kỳ SCL cao. Các ngoại lệ cho điều này là điều kiện bắt đầu (S) hoặc dừng (P) khi SDA giảm hoặc tăng tương ứng, trong khi SCL ở mức cao.

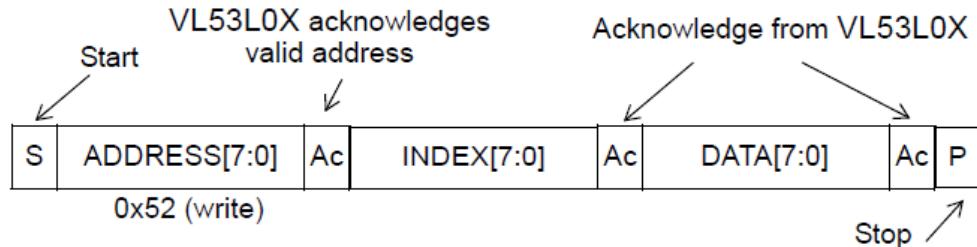
Một gói chứa một loạt byte trước điều kiện bắt đầu và sau là dừng hoặc bắt đầu lặp lại (một điều kiện bắt đầu khác nhưng không có điều kiện dừng trước đó) sau là một thông báo khác. Byte đầu tiên chứa địa chỉ vùng nhớ (0x29) và cũng chỉ định hướng dữ liệu. Nếu bit ít quan trọng nhất ở mức thấp (0x29) thì thông báo là bản ghi chính cho nó. Nếu lsb được đặt (0x53) thì thông báo là bản chính được đọc từ máy chủ.

MSBit	1	0	1	0	0	1	LSBit
0	1	0	1	0	0	1	R/W

Hình 2.41 Địa chỉ thiết bị VL53L0X I2C: 0x29

Tất cả các giao tiếp giao diện nối tiếp với mô-đun máy ảnh phải bắt đầu với điều kiện khởi động. Mô-đun VL53L0X xác nhận việc nhận một địa chỉ hợp lệ bằng cách điều khiển dây SDA ở mức thấp. Trạng thái của bit đọc / ghi (lsb của byte địa chỉ) được

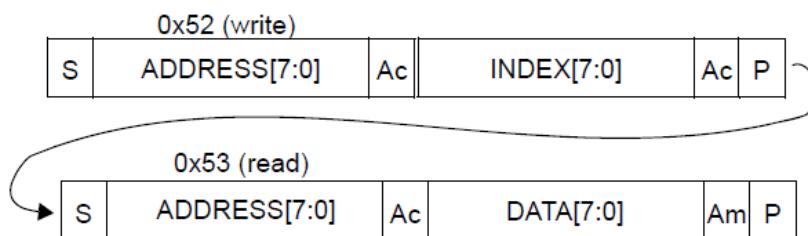
lưu trữ và byte dữ liệu tiếp theo, được lấy mẫu từ SDA, có thể được diễn giải. Trong một chuỗi ghi, byte thứ hai nhận được cung cấp một chỉ mục 8bit trả đến một trong các thanh ghi 8bit bên trong.



Hình 2.42 Định dạng dữ liệu VL53L0X (ghi)

Khi dữ liệu được nhận bởi phụ, nó được ghi từng bit vào một thanh ghi nối tiếp / song song. Sau khi mỗi byte dữ liệu được nhận bởi nô lệ, một xác nhận được tạo ra, dữ liệu sau đó được lưu trữ trong thanh ghi nội bộ được đánh địa chỉ bởi chỉ mục hiện tại.

Trong một thông báo đã đọc, nội dung của thanh ghi được định địa chỉ bởi chỉ mục hiện tại được đọc ra trong byte theo sau byte địa chỉ vùng nhớ. Nội dung của thanh ghi này được nạp song song vào thanh ghi nối tiếp / song song và được đưa ra khỏi thiết bị bởi cạnh xuống của SCL.

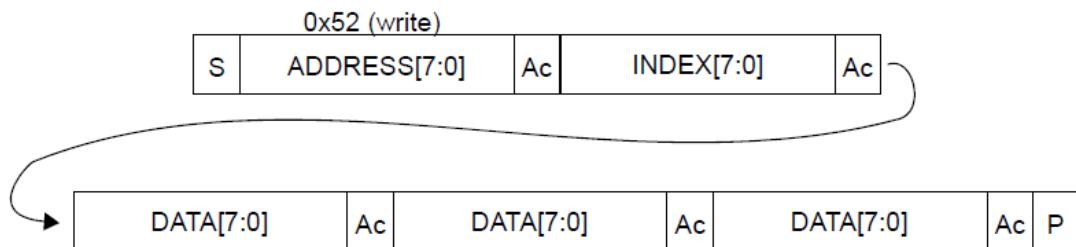


Hình 2.43 Định dạng dữ liệu VL53L0X (đọc)

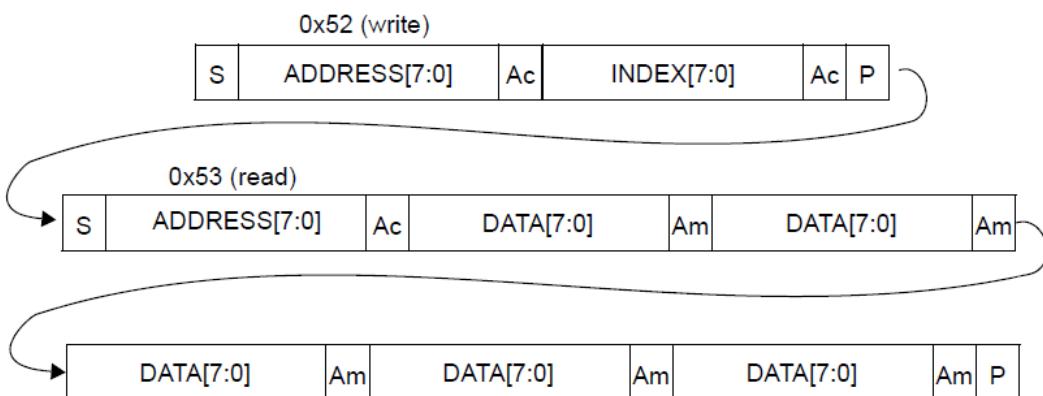
Vào cuối mỗi byte, trong cả chuỗi tin nhắn đọc và ghi, thiết bị nhận sẽ đưa ra một xác nhận (nghĩa là, VL53L0X để ghi và máy chủ để đọc).

Thông báo chỉ có thể được kết thúc bởi bus master, bằng cách đưa ra điều kiện dừng hoặc bằng một xác nhận phủ định (nghĩa là không kéo dòng SDA xuống thấp) sau khi đọc một byte hoàn chỉnh trong quá trình đọc.

Giao diện cũng hỗ trợ lập chỉ mục tự động tăng dần. Sau khi byte dữ liệu đầu tiên được chuyển, chỉ số sẽ tự động tăng thêm 1. Do đó, master có thể gửi các byte dữ liệu liên tục đến slave cho đến khi slave không cung cấp xác nhận hoặc master chấm dứt giao tiếp ghi với điều kiện dừng. Nếu tính năng tự động tăng dần được sử dụng, bản gốc không phải gửi chỉ mục địa chỉ kèm với các byte dữ liệu.



Hình 2.44 Định dạng dữ liệu VL53L0X (ghi tuần tự)



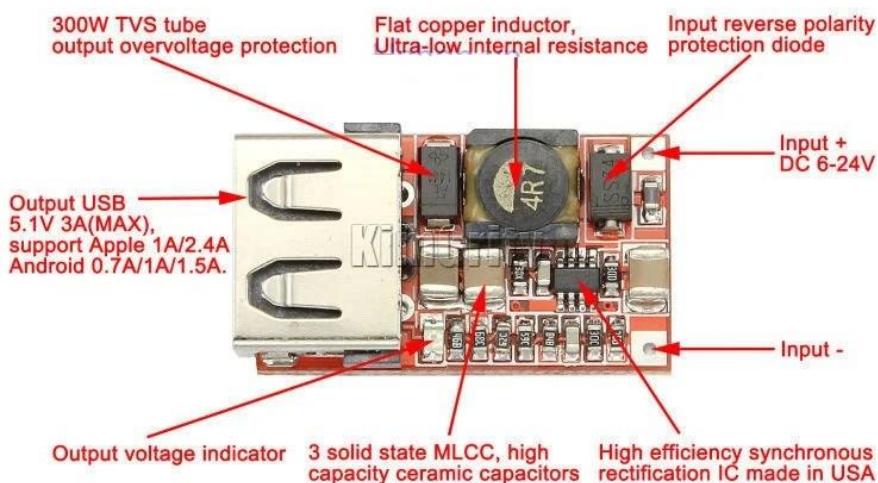
Hình 2.45 Định dạng dữ liệu VL53L0X (đọc tuần tự)

### 2.1.13 Mạch giảm áp DC Mini 5V 3A cổng USB Charge Module

Mạch giảm áp DC Mini 5V 3A cổng USB Charge Module có kích thước nhỏ gọn, đầu ra dạng cổng USB với công suất đầu ra tối đa lên đến 5VDC 3A, hiệu suất chuyển đổi cao 97.5% (theo thông số nhà SX), phù hợp với nhiều ứng dụng khác nhau như chế cổng sạc USB, cấp nguồn cho các thiết bị sử dụng cổng USB hoặc các thiết bị sử dụng 5VDC,...

- Thông số kỹ thuật:
  - Mạch giảm áp DC Mini 5V3A cổng USB.
  - Kiểu giảm áp: giảm áp xung không cách ly.
  - Điện áp đầu vào: 6~24VDC.

- Điện áp đầu ra: 5~5.2VDC.
- Dòng đầu ra tối đa: 3A.
- Dòng nghỉ: 0.85mA.
- Hiệu suất chuyển đổi: 97.5%
- Tần số chuyển đổi: 500Khz.
- Tích hợp Led báo nguồn.
- Tích hợp Diod chống cáp ngược nguồn.



Hình 2.46. Mạch giảm áp 5V-3A

Trên mạch ta sẽ có hai tiếp điểm cấp nguồn vào cho mạch là Input + và Input -, được cấp điện áp vào từ 6V đến 24V, ở đầu ra của mạch ta cần đầu cáp USB để có thể kết nối lấy nguồn từ mạch này, điện áp đầu ra cố định 5.1V và dòng điện ở ngõ ra lên đến 3A.

## 2.2. Các nguyên lý được áp dụng vào phần mềm

### 2.2.1. OpenCV

OpenCV (Open Computer Vision) là một thư viện mã nguồn mở hàng đầu cho xử lý về thị giác máy tính, machine learning, xử lý ảnh. OpenCV được viết bằng C/C++, vì vậy có tốc độ tính toán rất nhanh, có thể sử dụng với các ứng dụng liên quan đến thời gian thực. Opencv có các interface cho C/C++, Python Java vì vậy hỗ trợ được cho Window, Linux, MacOs lẫn Android, iOS OpenCV. [3]

OpenCV đang được sử dụng rộng rãi trong các ứng dụng bao gồm:

- Hình ảnh street view
- Kiểm tra và giám sát tự động
- Robot và xe hơi tự lái
- Phân tích hình ảnh y tế
- Tìm kiếm và phục hồi hình ảnh/video
- Phim - cấu trúc 3D từ chuyển động
- Nghệ thuật sắp đặt tương tác
- Chức năng OpenCV
- Image/video I/O, xử lý, hiển thị (core, imgproc, highgui)
- Phát hiện các vật thể (objdetect, features2d, nonfree)
- Geometry-based monocular or stereo computer vision (calib3d, stitching, videostab)
- Computational photography (photo, video, superres)
- Machine learning & clustering (ml, flann)
- CUDA acceleration (gpu)

#### a. Hough Line Transform

Hough Transform là thuật toán phát hiện đường thẳng khá hiệu quả trong xử lý ảnh. Áp dụng Hough Transform để phát hiện đường thẳng trong ảnh.

Ý tưởng chung của việc phát hiện đường thẳng trong thuật toán này là tạo mapping từ không gian ảnh (A) sang một không gian mới (B) mà mỗi đường thẳng trong không gian (A) sẽ ứng với một điểm trong không gian (B).

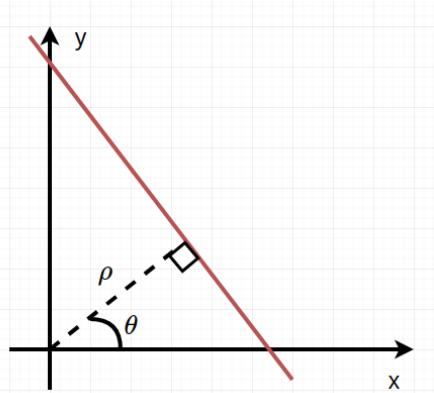
Phương trình đường thẳng trong không gian ảnh (A):

Như đã học ở cấp 2, phương trình đường thẳng cơ bản sẽ được biểu diễn theo 2 tham số a và b như sau:  $y = ax + b$

Tuy nhiên, với cách biểu diễn này, giá trị của góc nghiêng a trái dài từ  $-\infty$  đến  $+\infty$ . Có thể lấy ví dụ, để có được phương trình đường Oy ( $0x=0$ ) thì a phải tiến tới  $\infty$ . Thuật toán Hough Transform yêu cầu các giá trị a, b nằm trong một khoảng xác định (hay bị chặn trên dưới), ta phải sử dụng hệ tọa độ cực để biểu diễn phương trình đường thẳng. Cách biểu diễn này cũng nằm trong chương trình toán trung học:

$$\rho = x\cos(\theta) + y\sin(\theta).$$

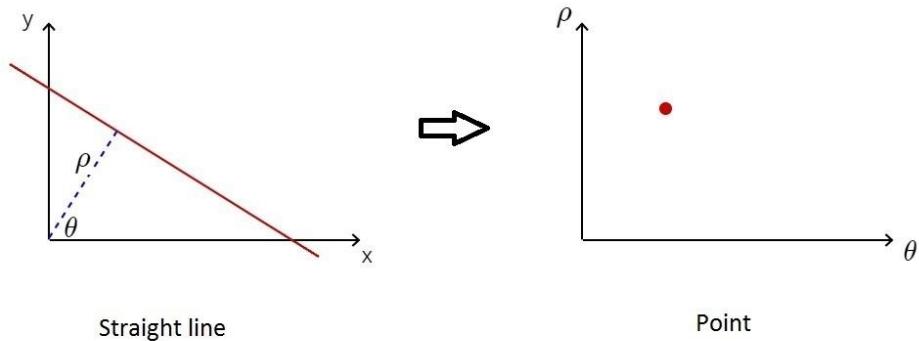
Xét thấy trong phương trình tọa độ cực, giá trị của góc  $\theta$  có thể bị chặn lại trong khoảng  $[0, \pi]$ . Trên thực tế, không gian ảnh là không gian hữu hạn (bị chặn lại bởi các cạnh của ảnh), do vậy giá trị  $\rho$  cũng bị chặn.



Hình 2.47. Minh họa biểu diễn đường thẳng trong hệ tọa độ cực

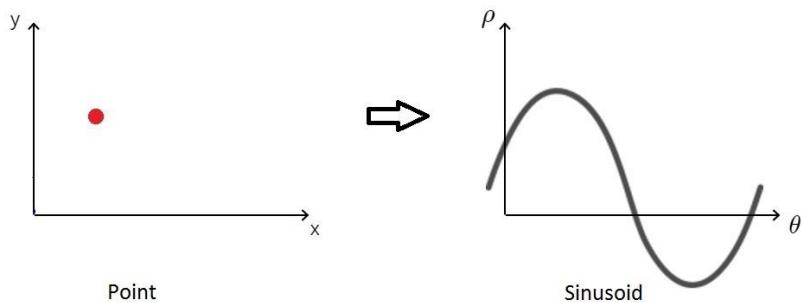
Mapping giữa không gian ảnh (A) và không gian Hough (B)

Từ một đường thẳng trong không gian ảnh (A) với 2 tham số  $\rho$  và  $\theta$ , chúng ta sẽ map sang không gian Hough (B) thành một điểm.



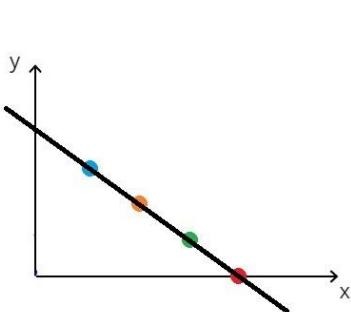
*Hình 2.48. Mapping một đường thẳng từ không gian ảnh sang không gian Hough*

Từ một điểm trong không gian ảnh, chúng ta lại có được một hình sin trong không gian Hough:

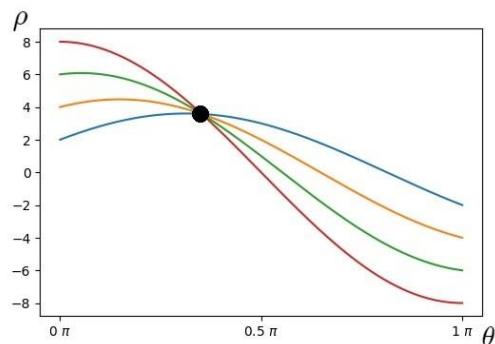


*Hình 2.49. Mapping một điểm từ không gian ảnh sang không gian Hough*

Các điểm nằm trên cùng một đường thẳng lại có biểu diễn là các hình sin giao nhau tại một điểm trong không gian Hough. Đây là nơi xuất phát ý tưởng của thuật toán Hough Transform. Chúng ta sẽ dựa vào các điểm giao nhau này để suy ngược lại phương trình đường thẳng trong không gian ảnh.



Points which form a line

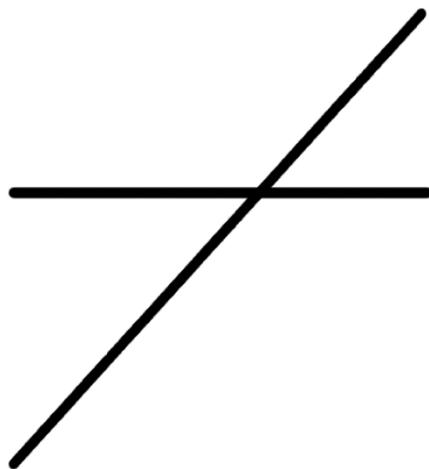


Bunch of sinusoids intersecting at one point

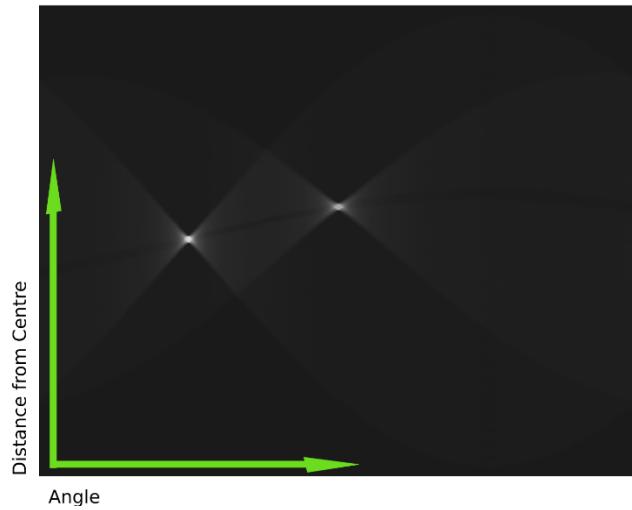
*Hình 2.50. Mapping nhiều điểm thẳng hàng từ không gian ảnh sang không gian Hough*

Mỗi đường thẳng khác nhau sẽ tạo thành một điểm sáng (nơi giao nhau của nhiều hình sin) trên không gian Hough. Dưới đây là sự biểu diễn 2 đường thẳng trong không gian Hough.<sup>[4]</sup>

Input Image



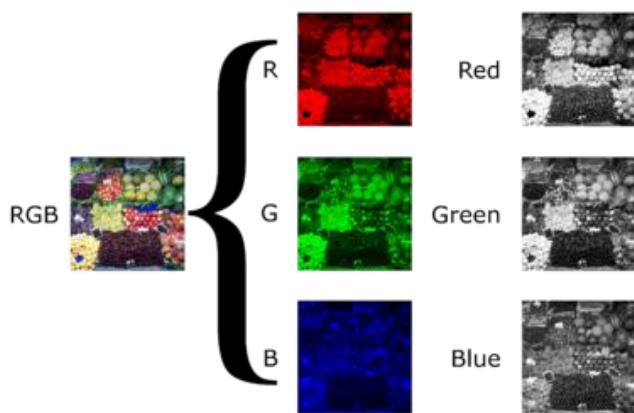
Rendering of Transform Results



*Hình 2.51. Biểu diễn 2 đường thẳng trong không gian Hough*

#### a. Biến đổi hình ảnh

Trong xử lý ảnh, việc chuyển đổi ảnh màu sang ảnh xám là công việc vô cùng phổ biến. Ảnh màu thực chất chỉ là tập hợp của những ma trận số có cùng kích thước. Khi muốn xử lý thông tin trên ảnh, sẽ dễ dàng hơn nếu ta chỉ xử lý dữ liệu trên một ma trận số thay vì nhiều ma trận số. Việc biến đổi ảnh màu về ảnh số (Grayscale converting) xuất hiện vì mục đích trên - biến đổi thông tin ảnh về một ma trận số hai chiều duy nhất.



*Hình 2.52 Chuyển ảnh RGB sang ảnh xám*

Giả sử, hình ảnh được lưu trữ dưới dạng RGB (Red-Green-Blue). Điều này có nghĩa là có ba ma trận xám tương ứng cho màu Red, Green, Blue. Công việc của chúng ta là tìm cách tổng hợp ba ma trận này về thành một ma trận duy nhất. Một trong số các công thức phổ biến để thực hiện việc đó là

$$Y = 0.2126R + 0.7152G + 0.0722B$$

Trong đó:

Y: ma trận xám cần tìm

R: ma trận xám đỏ của ảnh

G: ma trận xám lục của ảnh

B: ma trận xám lam của ảnh <sup>[5]</sup>

b. Làm mịn hình ảnh

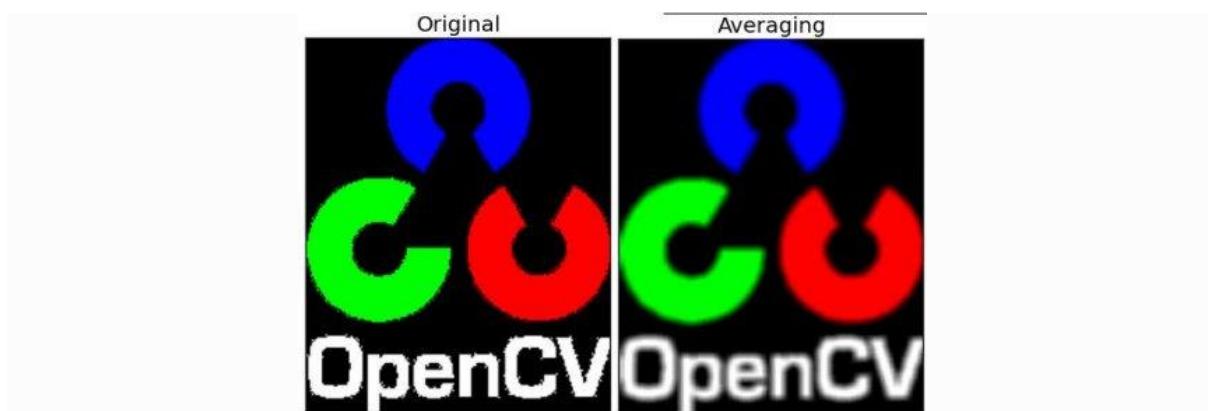
*Kết hợp 2D (Lọc ảnh)*

Đối với tín hiệu một chiều, hình ảnh cũng có thể được lọc bằng nhiều bộ lọc thông thấp (LPF), bộ lọc thông cao (HPF), v.v. LPF giúp loại bỏ nhiễu hoặc làm mờ hình ảnh. Bộ lọc HPF giúp tìm các cạnh trong ảnh.

OpenCV cung cấp một hàm, cv2.filter2D (), để kết hợp một kernel với một hình ảnh. Ví dụ, chúng ta sẽ thử một bộ lọc trung bình trên một hình ảnh. Một hạt nhân bộ lọc trung bình 5x5 có thể được định nghĩa như sau:

$$K = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Lọc với các kết quả hạt nhân ở trên được thực hiện như sau: với mỗi pixel, một cửa sổ  $5 \times 5$  được căn giữa vào pixel này, tất cả các pixel nằm trong cửa sổ này được tổng hợp và kết quả được chia cho 25. Điều này tương đương với tính toán trung bình của các giá trị pixel bên trong cửa sổ đó. Thao tác này được thực hiện cho tất cả các pixel trong ảnh để tạo ra ảnh được lọc đầu ra. Hãy thử mã này và kiểm tra kết quả:<sup>[6]</sup>



*Hình 2.53. Làm mịn ảnh*

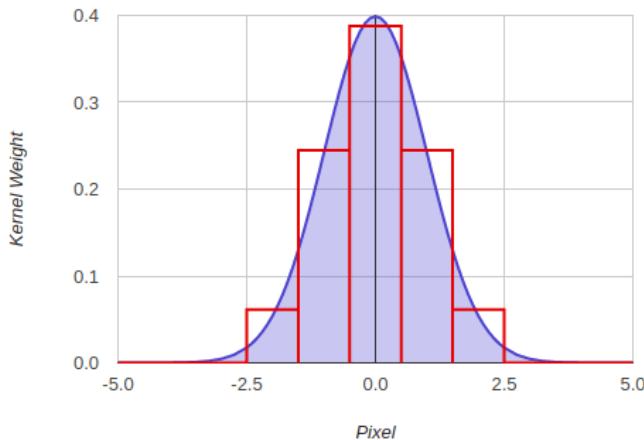
#### *Làm mờ hình ảnh (Làm mịn hình ảnh)*

Làm mờ hình ảnh đạt được bằng cách kết hợp hình ảnh với hạt nhân bộ lọc thông thấp. Nó rất hữu ích để loại bỏ nhiễu. Nó thực sự loại bỏ nội dung tần số cao (ví dụ: nhiễu, cạnh) khỏi hình ảnh dẫn đến các cạnh bị mờ khi áp dụng bộ lọc này. OpenCV cung cấp chủ yếu bốn loại kỹ thuật làm mờ.

##### c. Bộ lọc Gauss

Bộ lọc Gauss được coi là bộ lọc hữu ích nhất, được thực hiện bằng cách nhân chập ảnh đầu vào với một ma trận lọc Gauss sau đó cộng chúng lại để tạo thành ảnh đầu ra.

Ý tưởng chung là giá trị mỗi điểm ảnh sẽ phụ thuộc nhiều vào các điểm ảnh ở gần hơn là các điểm ảnh ở xa. Trọng số của sự phụ thuộc được lấy theo hàm Gauss (cũng được sử dụng trong quy luật phân phối chuẩn).



Hình 2.54. Bô lọc Gauss

Giả sử ảnh là một chiều. Điểm ảnh ở trung tâm sẽ có trọng số lớn nhất. Các điểm ảnh ở càng xa trung tâm sẽ có trọng số giảm dần khi khoảng cách từ chúng tới điểm trung tâm tăng lên. Như vậy điểm càng gần trung tâm sẽ càng đóng góp nhiều hơn vào giá trị điểm trung tâm.

Trên thực tế, việc lọc ảnh dựa trên hàm Gauss 2 chiều (ngang và dọc). Phân phối chuẩn 2 chiều có thể biểu diễn dưới dạng:

$$G_0(x, y) = Ae^{-\frac{-(x - \mu_x)^2}{2\sigma_x^2} - \frac{-(y - \mu_y)^2}{2\sigma_y^2}}$$

Trong đó  $\mu$  là trung bình (đỉnh),  $\sigma^2$  là phương sai của các biến số x và y.

Tham số  $\mu$  quyết định tác dụng của bộ lọc Gauss lên ảnh. Độ lớn của ma trận lọc (kernel) cần được lựa chọn cho đủ rộng.

Code thực tế lọc Gauss với Python - OpenCV: Trong OpenCV chúng ta sử dụng hàm sau để lọc Gauss: cv.GaussianBlur().<sup>[7]</sup>

#### *d. Canny Edge Detection*

Phát hiện cạnh Canny là một thuật toán phát hiện cạnh phổ biến. Nó được phát triển bởi John F. Canny vào năm 1986. Đây là một thuật toán gồm nhiều giai đoạn.

Trong hình ảnh, thường tồn tại các thành phần như: vùng trơn, góc / cạnh và nhiễu. Cạnh trong ảnh mang đặc trưng quan trọng, thường là thuộc đối tượng trong ảnh (object). Do đó, để phát hiện cạnh trong ảnh, giải thuật Canny là một trong những giải thuật phổ biến / nổi tiếng nhất trong xử lý ảnh.

#### *Giảm nhiễu*

Do phát hiện cạnh dễ bị nhiễu trong ảnh, bước đầu tiên là loại bỏ nhiễu trong ảnh bằng bộ lọc Gaussian 5x5. Kích thước 5x5 thường hoạt động tốt cho giải thuật Canny.

#### *Tính Gradient và hướng gradient:*

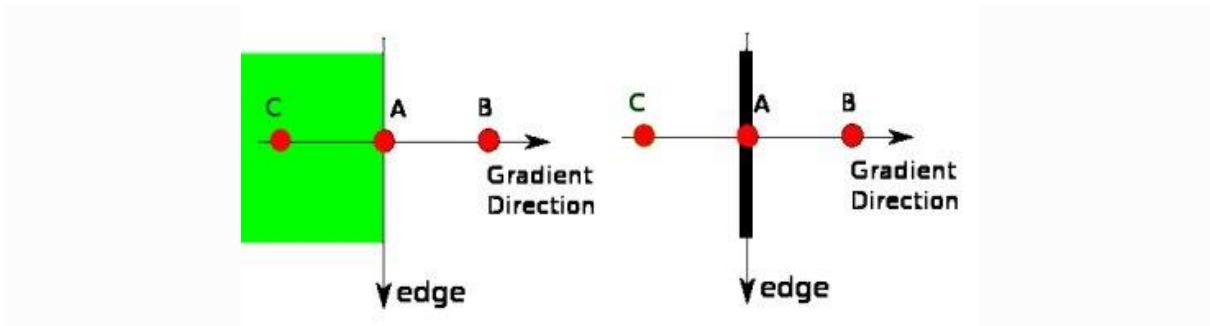
Ta dùng bộ lọc Sobel X và Sobel Y (3x3) để tính được ảnh đạo hàm G<sub>x</sub> và G<sub>y</sub>. Sau đó, ta tiếp tục tính ảnh Gradient và góc của Gradient theo công thức. Ảnh đạo hàm G<sub>x</sub> và G<sub>y</sub> là ma trận (ví dụ: 640x640), thì kết quả tính ảnh đạo hàm Edge Gradient cũng là một ma trận (640x640), mỗi pixel trên ma trận này thể hiện độ lớn của biến đổi mức sáng ở vị trí tương ứng trên ảnh gốc. Tương tự, ma trận Angle cũng có cùng kích thước (640x640), mỗi pixel trên Angle thể hiện góc, hay nói cách khác là hướng của cạnh.

$$\begin{aligned} \text{Edge Gradient } (G) &= \sqrt{G_x^2 + G_y^2} \\ \text{Angle } (\theta) &= \tan^{-1} \left( \frac{G_y}{G_x} \right) \end{aligned}$$

#### *Non-maximum Suppression (NMS)*

Loại bỏ các pixel ở vị trí không phải cực đại toàn cục. Ở bước này, ta dùng một filter 3x3 lặp lượt chạy qua các pixel trên ảnh gradient. Trong quá trình lọc, ta xem xét xem độ lớn gradient của pixel trung tâm có phải là cực đại (lớn nhất trong cục bộ - local maximum) so với các gradient ở các pixel xung quanh. Nếu là cực đại, ta sẽ ghi nhận sẽ giữ pixel đó lại. Còn nếu pixel tại đó không phải là cực đại lân cận, ta sẽ set độ lớn

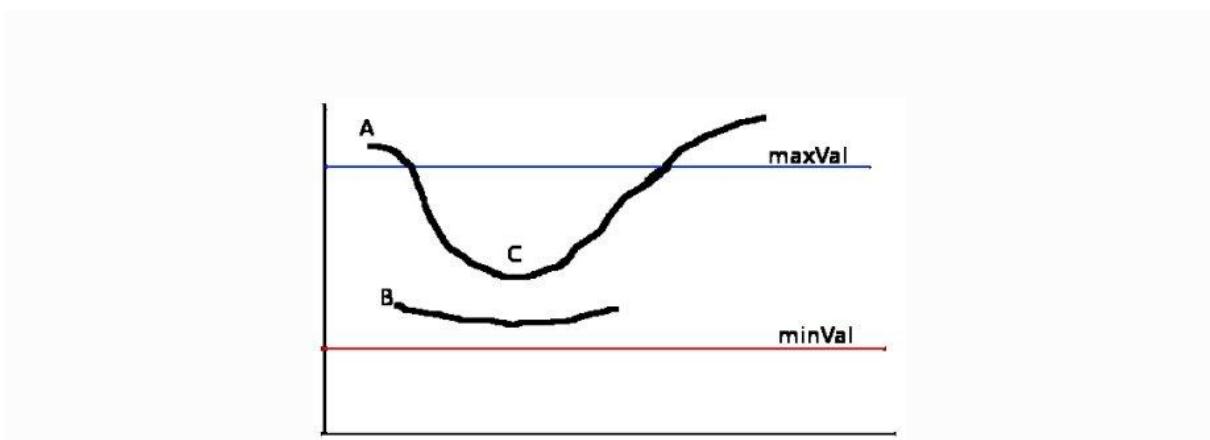
gradient của nó về zero. Ta chỉ so sánh pixel trung tâm với 2pixel lân cận theo **hướng gradient**. Ví dụ: nếu hướng gradient đang là 0 độ, ta sẽ so pixel trung tâm với pixel liền trái và liền phải nó. Trường hợp khác nếu hướng gradient là 45 độ, ta sẽ so sánh với 2pixel lân cận là góc trên bên phải và góc dưới bên trái của pixel trung tâm. Tương tự cho 2 trường hợp hướng gradient còn lại. Kết thúc bước này ta được một mặt nạ nhị phân.



Hình 2.55. Non-maximum Suppression

### Lọc ngưỡng

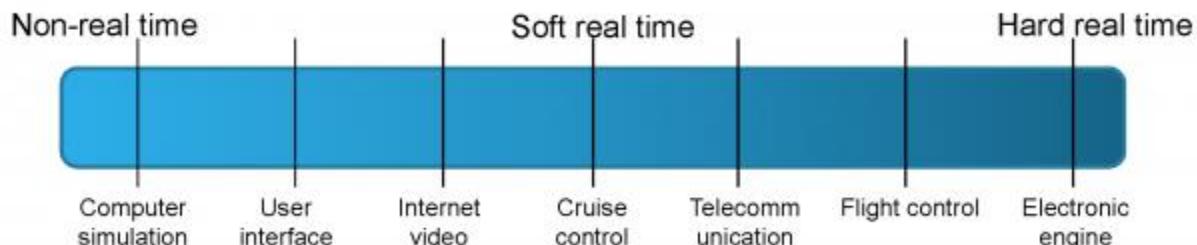
Ta sẽ xét các pixel dương trên mặt nạ nhị phân kết quả của bước trước. Nếu giá trị gradient vượt ngưỡng `max_val` thì pixel đó chắc chắn là cạnh. Các pixel có độ lớn gradient nhỏ hơn ngưỡng `min_val` sẽ bị loại bỏ. Còn các pixel nằm trong khoảng 2 ngưỡng trên sẽ được xem xét rằng nó có nằm liền kề với những pixel được coi là "chắc chắn là cạnh" hay không. Nếu liền kề thì ta giữ, còn không liền kề bất cứ pixel cạnh nào thì ta loại. Sau bước này ta có thể áp dụng thêm bước hậu xử lý loại bỏ nhiễu (tức những pixel cạnh rác hay cạnh ngắn) nếu muốn. Ảnh minh họa về ngưỡng lọc.<sup>[8]</sup>



Hình 2.56. Lọc ngưỡng

### 2.2.2. FreeRTOS

FreeRTOS là một hệ điều hành nhúng thời gian thực (Real Time Operating System) mã nguồn mở được phát triển bởi Real Time Engineers Ltd, sáng lập và sở hữu bởi Richard Barry. FreeRTOS được thiết kế phù hợp cho nhiều hệ nhúng nhỏ gọn vì nó chỉ triển khai rất ít các chức năng như: cơ chế quản lý bộ nhớ và tác vụ cơ bản, các hàm API quan trọng cho cơ chế đồng bộ. Nó không cung cấp sẵn các giao tiếp mạng, drivers, hay hệ thống quản lý tệp (file system) như những hệ điều hành nhúng cao cấp khác. Tuy vậy, FreeRTOS có nhiều ưu điểm, hỗ trợ nhiều kiến trúc vi điều khiển khác nhau, kích thước nhỏ gọn (4.3 Kbytes sau khi biên dịch trên Arduino), được viết bằng ngôn ngữ C và có thể sử dụng, phát triển với nhiều trình biên dịch C khác nhau (GCC, OpenWatcom, Keil, IAR, Eclipse, ...), cho phép không giới hạn các tác vụ chạy đồng thời, không hạn chế quyền ưu tiên thực thi, khả năng khai thác phần cứng. Ngoài ra, nó cũng cho phép triển khai các cơ chế điều độ giữa các tiến trình như: queues, counting semaphore, mutexes.

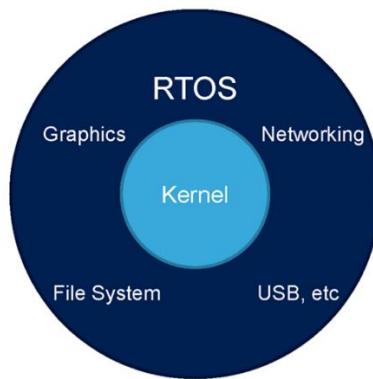


Hình 2.57. Các cấp độ yêu cầu real-time

Với mô phỏng máy tính sẽ là non-real time ở cấp độ cao nhất, tiếp đến là giao tiếp người dùng, truyền video thông qua internet, tới mức soft real time là hệ thống cruise control trên ô tô (hệ thống tự kiểm soát tốc độ), các ứng dụng về viễn thông, điều khiển trên máy bay, động cơ điện thì yêu cầu mức hard real time.

Một hệ điều hành có thể

- Cho phép nhiều chương trình chạy cùng 1 lúc (multi-tasking)
- Có quản lý tài nguyên về phần cứng và cung cấp các dịch vụ cho các chương trình khác.



Hình 2.58. Cấu tạo của một hệ điều hành thời gian thực (RTOS)

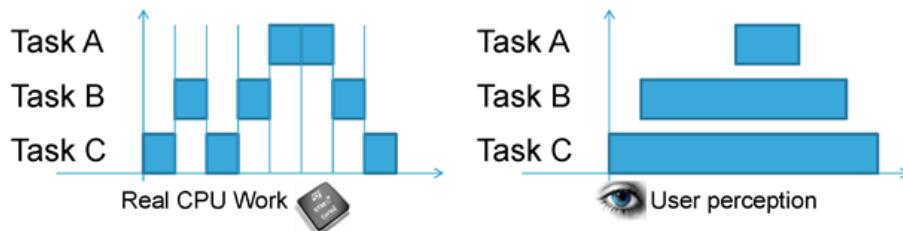
Các ứng dụng không cần dùng RTOS

- Ứng dụng đơn (ứng dụng chỉ có 1 chức năng)
- Ứng dụng có vòng lặp đơn giản
- Ứng dụng <32kB

Nếu ứng dụng của chúng ta mà kích thước chương trình lớn dần và độ phức tạp tăng lên thì RTOS sẽ rất hữu dụng trong trường hợp này, lúc đó RTOS sẽ chia các ứng dụng phức tạp thành các phần nhỏ hơn và dễ quản lý hơn.

Các công dụng của RTOS:

- Chia sẻ tài nguyên một cách đơn giản: cung cấp cơ chế để phân chia các yêu cầu về bộ nhớ và ngoại vi của MCU
- Dễ debug và phát triển: Mọi người trong nhóm có thể làm việc một cách độc lập, các lập trình viên thì có thể tránh được các tương tác với ngắn, timer, với phần cứng (cái này mình không khuyến khích lầm vì hiểu được phần cứng vẫn sẽ tốt hơn nhiều)
- Tăng tính linh động và dễ dàng bảo trì: thông qua API của RTOS...



Hình 2.59. Nguyên lý làm việc của RTOS

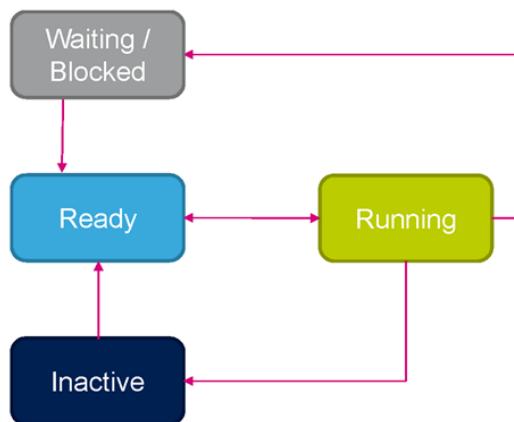
Kernel sẽ có nhiệm vụ quản lý nhiều task cùng chạy 1 lúc, mỗi task thường chạy mất vài ms. Tại lúc kết thúc task thường:

- Lưu trạng thái task.
- Thanh ghi CPU sẽ load trạng thái của task tiếp theo.
- Task tiếp theo cần khoảng vài ms để thực hiện.

Vì CPU thực hiện tác vụ rất nhanh nên dưới góc nhìn người dùng thì hầu như các task là được thực hiện 1 cách liên tục.

Task state:

Một task trong RTOS thường có các trạng thái như sau:



Hình 2.60. Task trong RTOS

RUNNING: đang thực thi

READY: sẵn sàng để thực hiện

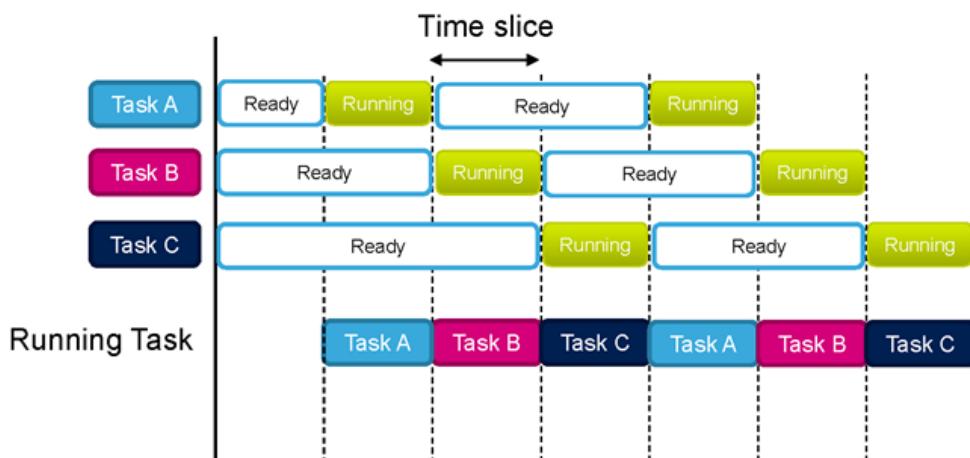
WAITING: chờ sự kiện

INACTIVE: không được kích hoạt

Scheduler:

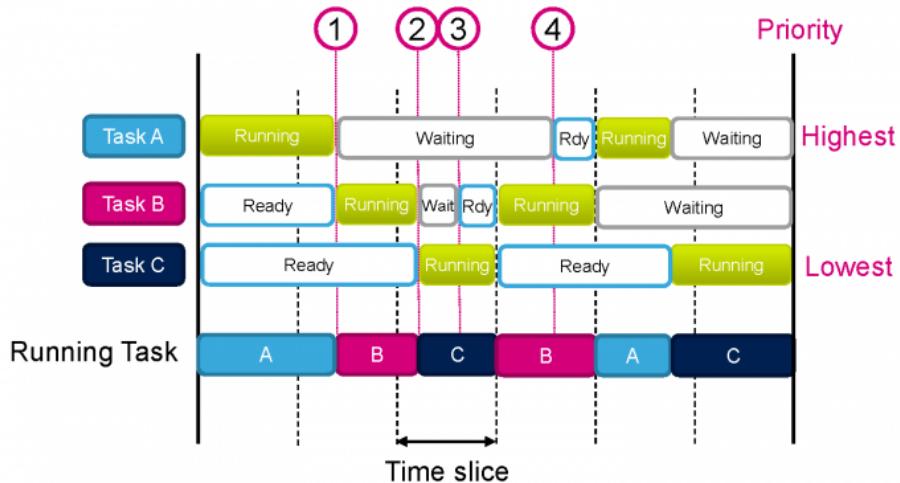
Đây là một thành phần của kernel quyết định task nào được thực thi. Có một số luật cho scheduling như:

- Cooperative: giống với lập trình thông thường, mỗi task chỉ có thể thực thi khi task đang chạy dừng lại, nhược điểm của nó là task này có thể dùng hết tất cả tài nguyên của CPU.
- Round-robin: mỗi task được thực hiện trong thời gian định trước (time slice) và không có ưu tiên.



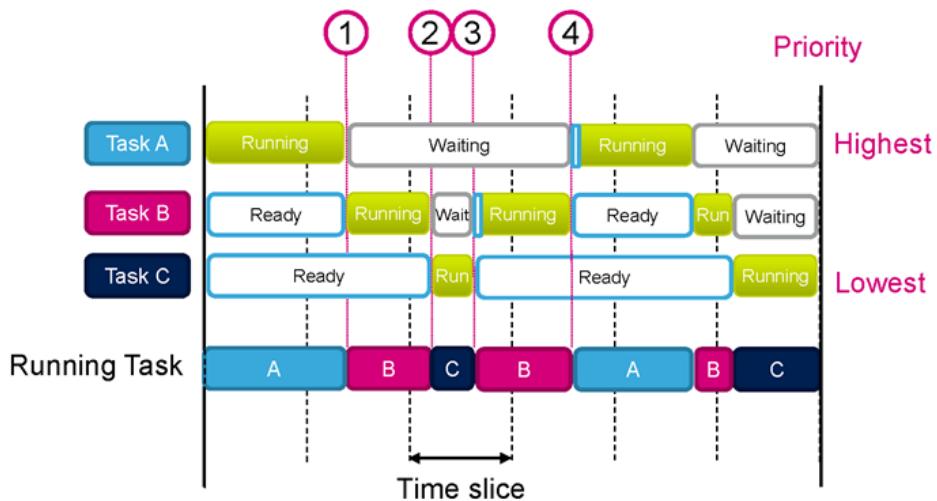
Hình 2.61. Round-robin

- Priority base: Task được phân quyền cao nhất sẽ được thực hiện trước, nếu các task có cùng quyền như nhau thì sẽ giống với round-robin, các task có mức ưu tiên thấp hơn sẽ được thực hiện cho đến cuối time slice.



Hình 2.62. Priority base

1. Task A chờ event
2. Task B chờ event
3. Task B event sẵn sàng
4. Task A event sẵn sàng
  - Priority-based pre-emptive: Các task có mức ưu tiên cao nhất luôn nhường các task có mức ưu tiên thấp hơn thực thi trước.



Hình 2.63. Priority-based pre-emptive

1. Task A chờ event.
2. Task B chờ event.
3. Task B event sẵn sàng.
4. Task A event sẵn sàng.

Một task là một chương trình, chương trình này chạy liên tục trong vòng lặp vô tận và không bao giờ dừng lại. Với mỗi task thì có niềm tin duy nhất là chỉ mình nó đang chạy và có thể sử dụng hết nguồn tài nguyên sẵn có của bộ xử lý (mặc dù là thực tế thì nó vẫn phải chia sẻ nguồn tài nguyên này với các task khác).

Một chương trình thường sẽ có nhiều task khác nhau. Ví dụ như máy bán đồ uống tự động sẽ có các thành phần sau:

- Task quản lý việc lựa chọn của người dùng.
- Task để kiểm tra đúng số tiền người dùng đã trả.
- Task để điều khiển động cơ/ cơ cấu cung cấp nước uống.

Kernel sẽ quản lý việc chuyển đổi giữa các task, nó sẽ lưu lại ngữ cảnh của task sắp bị hủy và khôi phục lại ngữ cảnh của task tiếp theo bằng cách:

- Kiểm tra thời gian thực thi đã được định nghĩa trước (time slice được tạo ra bởi ngắt systick).
- Khi có các sự kiện unblocking một task có quyền cao hơn xảy ra (signal, queue, semaphore...).
- Khi task gọi hàm Yield () để ép Kernel chuyển sang các task khác mà không phải chờ cho hết time slice.

Khi khởi động thì kernel sẽ tạo ra một task mặc định gọi là Idle Task.

Để tạo một task thì cần phải khai báo hàm định nghĩa task, sau đó tạo task và cấp phát bộ nhớ.

Kết nối Inter-task & Chia sẻ tài nguyên.

Các task cần phải kết nối và trao đổi dữ liệu với nhau để có thể chia sẻ tài nguyên, có một số khái niệm cần lưu ý:

Với Inter-task Communication:

- Signal Events – Đồng bộ các task.

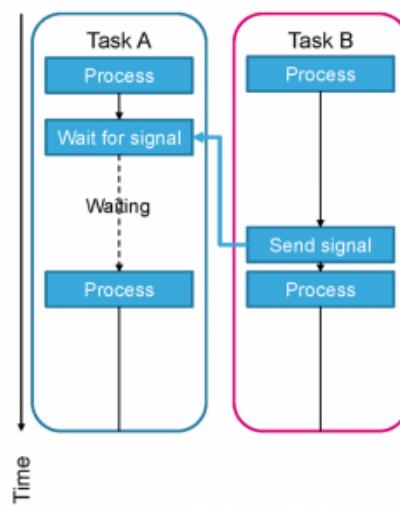
- Message queue – Trao đổi tin nhắn giữa các task trong hoạt động giống như FIFO.
- Mail queue – Trao đổi dữ liệu giữa các task sử dụng hằng đợi của khối bộ nhớ.

Với Resource Sharing:

- Semaphores – Truy xuất tài nguyên liên tục từ các task khác nhau.
- Mutex – Đóng bộ hóa truy cập tài nguyên sử dụng Mutual Exclusion.

Signal event:

Signal event được dùng để đồng bộ các task, ví dụ như bắt task phải thực thi tại một sự kiện nào đó được định sẵn.



Hình 2.64. Signal event

Ví dụ: Một cái máy giặt có 2 task là Task A điều khiển động cơ, Task B đọc mức nước từ cảm biến nước đầu vào.

- Task A cần phải chờ nước đầy trước khi khởi động động cơ. Việc này có thể thực hiện được bằng cách sử dụng signal event.
- Task A phải chờ signal event từ Task B trước khi khởi động động cơ.
- Khi phát hiện nước đã đạt tới mức yêu cầu thì Task B sẽ gửi tín hiệu tới Task A.

Với trường hợp này thì task sẽ đợi tín hiệu trước khi thực thi, nó sẽ nằm trong trạng thái là WAITING cho đến khi signal được set. Ngoài ra ta có thể set 1 hoặc nhiều signal trong bất kỳ các task nào khác.

Mỗi task có thể được gán tối đa là 32 signal event.

Ưu điểm của nó là thực hiện nhanh, sử dụng ít RAM hơn so với semaphore và message queue nhưng có nhược điểm lại chỉ được dùng khi một task nhận được signal.

Message queue là cơ chế cho phép các task có thể kết nối với nhau, nó là một FIFO buffer được định nghĩa bởi độ dài (số phần tử mà buffer có thể lưu trữ) và kích thước dữ liệu (kích thước của các thành phần trong buffer). Một ứng dụng tiêu biểu là buffer cho Serial I/O, buffer cho lệnh được gửi tới task.



Hình 2.65. Message queue

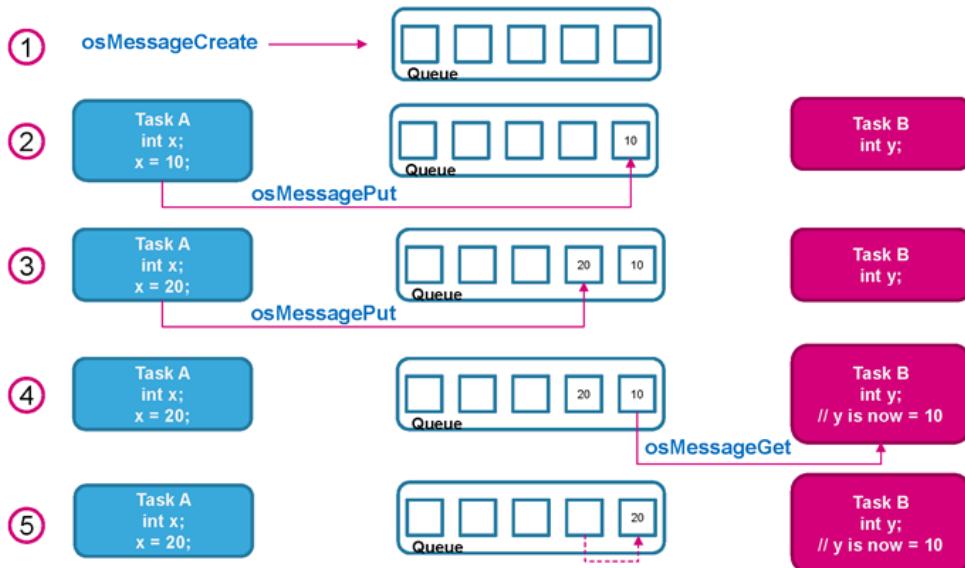
Task có thể ghi vào hàng đợi (queue):

- Task sẽ bị khóa (block) khi gửi dữ liệu tới một message queue đầy đủ.
- Task sẽ hết bị khóa (unblock) khi bộ nhớ trong message queue trống.
- Trường hợp nhiều task mà bị block thì task với mức ưu tiên cao nhất sẽ được unblock trước.

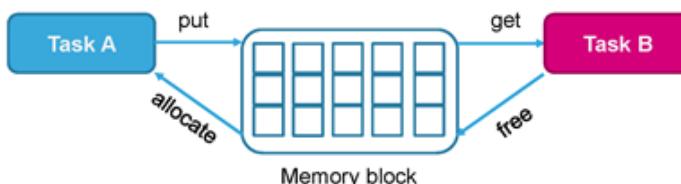
Task có thể đọc từ hàng đợi (queue)

- Task sẽ bị block nếu message queue trống.
- Task sẽ được unblock nếu có dữ liệu trong message queue.
- Tương tự ghi thì task được unblock dựa trên mức độ ưu tiên.

Ví dụ:

*Hình 2.66. Queue*

Giống như message queue nhưng dữ liệu sẽ được truyền dưới dạng khói (memory block) thay vì dạng đơn. Mỗi memory block thì cần phải cấp phát trước khi đưa dữ liệu vào và giải phóng sau khi đưa dữ liệu ra.

*Hình 2.67. Mail queue*

Gửi dữ liệu với mail queue:

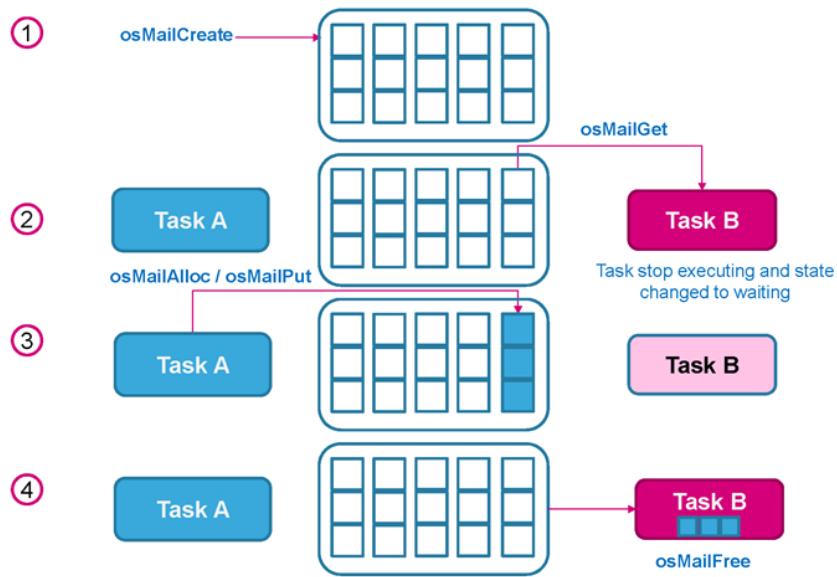
- Cấp phát bộ nhớ từ mail queue cho dữ liệu được đặt trong mail queue.
- Lưu dữ liệu cần gửi vào bộ nhớ đã được cấp phát.
- Đưa dữ liệu vào mail queue.

Nhận dữ liệu trong mail queue bởi task khác:

- Lấy dữ liệu từ mail queue, sẽ có một hàm để trả lại cấu trúc/ đối tượng
- Lấy con trỏ chứa dữ liệu

- Giải phóng bộ nhớ sau khi sử dụng dữ liệu

Ví dụ:



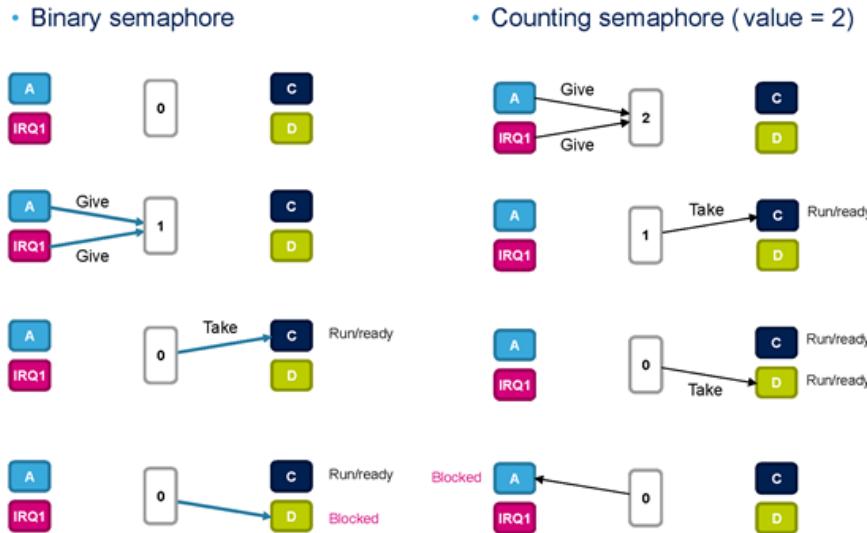
Hình 2.68. Mail queue

Được sử dụng để đồng bộ task với các sự kiện khác trong hệ thống. Có 2 loại:  
Binary semaphore:

- Trường hợp đặc biệt của counting semaphore.
- Có duy nhất 1 token.
- Chỉ có 1 hoạt động đồng bộ.

Counting semaphore:

- Có nhiều token
- Có nhiều hoạt động đồng bộ

*Hình 2.69. Semaphore*

Couting semaphore được dùng để:

#### Counting event

- Một event handler sẽ ‘give’ semaphore khi có event xảy ra (tăng giá trị đếm semaphore)
- Một task handler sẽ ‘take’ semaphore khi nó thực thi sự kiện (giảm giá trị đếm semaphore)
- Count value là khác nhau giữa số sự kiện xảy ra và số sự kiện được thực thi
- Trong trường hợp counting event thì semaphore được khởi tạo giá trị đếm bằng 0.

#### Resource management:

- Count value sẽ chỉ ra số resource sẵn có.
- Để điều khiển và kiểm soát được resource của task dựa trên count value của semaphore (giá trị giảm), nếu count value giảm xuống bằng 0 nghĩa là không có resource nào free.
- Khi một task finish với resource thì nó sẽ give semaphore trở lại để tăng count value của semaphore.

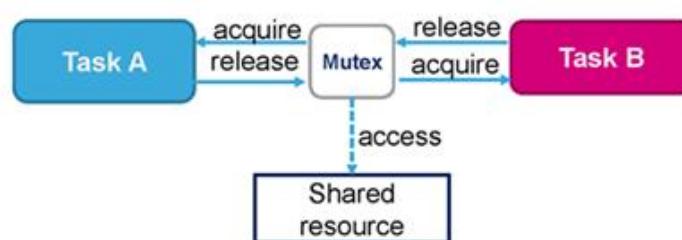
- Trong trường hợp resource management thì count value sẽ bằng với giá trị max của count value khi semaphore được tạo.

Mutex:

Sử dụng cho việc loại trừ (mutual exclusion), hoạt động như là một token để bảo vệ tài nguyên được chia sẻ. Một task nếu muốn truy cập vào tài nguyên chia sẻ:

- Cần yêu cầu (đợi) mutex trước khi truy cập vào tài nguyên chia sẻ
- Đưa ra token khi kết thúc với tài nguyên.

Tại mỗi một thời điểm thì chỉ có 1 task có được mutex. Những task khác muốn cùng mutex thì phải block cho đến khi task cũ thả mutex ra.



Hình 2.70. Mutex

Về cơ bản thì Mutex giống như binary semaphore nhưng được sử dụng cho việc loại trừ chứ không phải đồng bộ. Ngoài ra thì nó bao gồm cơ chế thừa kế mức độ ưu tiên (Priority inheritance mechanism) để giảm thiểu vấn đề đảo ngược ưu tiên, cơ chế này có thể hiểu đơn giản qua ví dụ sau:

- Task A (low priority) yêu cầu mutex.
- Task B (high priority) muốn yêu cầu cùng mutex trên.
- Mức độ ưu tiên của Task A sẽ được đưa tạm về Task B để cho phép Task A được thực thi.
- Task A sẽ thả mutex ra, mức độ ưu tiên sẽ được khôi phục lại và cho phép Task B tiếp tục thực thi.<sup>[9]</sup>

### 2.2.3 Tensorflow

Sự phát triển của trí tuệ nhân tạo dẫn đến việc tìm hiểu về máy học và học sâu đã trở thành xu thế hiện nay. Việc sử dụng các thư viện có sẵn để tính toán đã giúp việc tiếp cận các bài toán trở nên đơn giản hơn. Tensorflow là một thư viện phần mềm mã nguồn mở hỗ trợ mạnh mẽ các phép toán học để tính toán trong machine learning và deep learning.

TensorFlow là thư viện để xác định các biểu đồ tính toán trừu tượng, mục đích chung. Mặc dù chúng được sử dụng cho deep learning, nhưng chúng không phải là deep learning framework và thực tế được sử dụng cho rất nhiều ứng dụng khác ngoài học sâu.

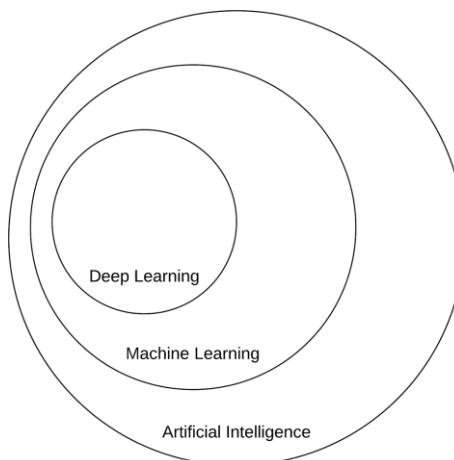
Mặt khác, Keras là một deep learning framework cung cấp API được thiết kế tốt để tạo điều kiện xây dựng mạng lưới thần kinh sâu một cách dễ dàng. Keras sử dụng phụ trợ tính toán TensorFlow, cho phép nó tận dụng các công cụ tính toán mạnh mẽ này. Hãy nghĩ về một phụ trợ tính toán như một động cơ chạy trong xe. Ta có thể thay thế các bộ phận trong động cơ của mình, tối ưu hóa các bộ phận khác hoặc thay thế hoàn toàn động cơ (miễn là động cơ tuân thủ một bộ thông số kỹ thuật tất yếu). Việc sử dụng Keras cung cấp cho ta tính di động trên các công cụ và khả năng chọn công cụ tốt nhất cho dự án. Nghĩ về điều này ở một góc độ khác, sử dụng TensorFlow để xây dựng một mạng lưới thần kinh sâu sẽ giống như sử dụng NumPy một cách nghiêm ngặt để xây dựng một bộ phân loại máy học. Tuy nhiên, nó có lợi hơn khi sử dụng một thư viện dành riêng cho máy học, chẳng hạn như scikit-learn, thay vì phát minh lại bánh xe với NumPy.

Những lợi ích của một công cụ tính toán cơ bản mạnh mẽ. API giúp ta dễ dàng xây dựng mạng lưới học tập sâu hơn của riêng mình. Hơn nữa, vì Keras sẽ được thêm vào thư viện TensorFlow cốt lõi tại Google, chúng ta luôn có thể tích hợp code TensorFlow trực tiếp vào các mô hình Keras. Theo nhiều cách, chúng ta đang trở nên tốt nhất của cả hai thế giới bằng cách sử dụng Keras.

Các phương pháp học sâu là các phương pháp học biểu diễn với nhiều cấp độ biểu diễn, thu được bằng cách biểu diễn các mô-đun đơn giản nhưng phi tuyến mà mỗi

biến đổi biểu diễn ở một cấp độ (bắt đầu bằng đầu vào thô) thành biểu diễn ở mức cao hơn, trừu tượng hơn một chút. Khía cạnh quan trọng của học sâu là các lớp này không được thiết kế bởi các kỹ sư của con người: chúng được học từ dữ liệu bằng cách sử dụng quy trình học tập đa năng - Yann LeCun, Yoshua Bengio và Geoffrey Hinton, Nature 2015.

Học sâu là một trường con của máy học, tổng quan hơn nó là một trường con của trí tuệ nhân tạo (AI). Mục tiêu trọng tâm của AI là cung cấp một tập hợp các thuật toán và kỹ thuật có thể được sử dụng để giải quyết các vấn đề mà con người thực hiện bằng trực giác và gần tự động, nhưng nếu không thì rất khó khăn cho máy tính. Một ví dụ ưu điểm về một loại vấn đề AI như vậy là diễn giải và hiểu nội dung của hình ảnh - nhiệm vụ này là điều mà con người có thể làm với rất ít nỗ lực, nhưng nó đã được chứng minh là cực kỳ khó khăn để máy móc thực hiện. Trong khi AI là hiện thân của một nhóm công việc lớn, đa dạng liên quan đến lý luận máy tự động (suy luận, lập kế hoạch, chẩn đoán, v.v.), thì trường con máy học có xu hướng đặc biệt quan tâm đến nhận dạng mẫu và học từ dữ liệu. Mạng nơ ron nhân tạo (ANN) là một lớp các thuật toán học máy học từ dữ liệu và chuyên về nhận dạng mẫu, lấy cảm hứng từ cấu trúc và chức năng của não. Học sâu thuộc về các thuật toán ANN và trong hầu hết các trường hợp, hai thuật ngữ có thể được sử dụng thay thế cho nhau. Trên thực tế, ta có thể ngạc nhiên khi biết rằng lĩnh vực học sâu đã tồn tại hơn 60 năm, với những tên gọi và hiện thân khác nhau dựa trên xu hướng nghiên cứu, phần cứng và bộ dữ liệu có sẵn và các tùy chọn phổ biến của các nhà nghiên cứu nổi tiếng vào thời điểm đó. [10]



Hình 2.71. Một sơ đồ Venn mô tả học sâu

#### 2.2.4. Thuật toán phát hiện đối tượng

Phát hiện mục tiêu đề cập đến việc phát hiện vị trí của một đối tượng trong ảnh để có được tọa độ và diện tích chính xác của đối tượng trong ảnh. Với sự phát triển của khoa học, nội dung và thuật toán của vấn đề phát hiện mục tiêu cũng đã trải qua quá trình tiến hóa liên tục: ban đầu việc phát hiện các hình hình học đơn giản như đường thẳng và hình elip, một số thuật toán truyền thống như biến đổi Hough và khớp đường cong có thể giải quyết vấn đề tốt hơn. Ngay sau đó để đạt được phát hiện hình ảnh dựa trên nguyên tắc các thuộc tính của hình ảnh, nhiều nhà khoa học khám phá việc xây dựng các tính năng cơ bản khác nhau của hình ảnh. Classical SIFT, FAST và SURF là tất cả các tính năng tiêu biểu được đề xuất trong giai đoạn này và sau đó mở rộng thêm, chẳng hạn như phát hiện khuôn mặt người, mắt người, v.v., cũng có thể được giải quyết bằng các phương pháp truyền thống như tính năng Haar + Cascade; phát hiện mục tiêu trong lĩnh vực học sâu thường đề cập đến việc phát hiện và phân loại các đối tượng trong hình ảnh để thu được các loại và hộp giới hạn 2D hoặc 3D. Các thuật toán thường được sử dụng bao gồm R-CNN, Fast R-CNN, YOLO, SSD và các thuật toán khác.

Trong lĩnh vực học sâu, đối tượng của vấn đề phát hiện đối tượng nói chung là những thứ có hình dạng tương đối cố định, tích hợp phát hiện và phân loại vị trí. R-CNN đã tiên phong phương pháp phát hiện mục tiêu dựa trên các vùng ứng của đối tượng. Nguyên tắc là trước tiên lọc ra một số vùng của đối tượng, sau đó sử dụng CNN để xử lý từng vùng của đối tượng để thu được độ lệch của kết quả phân loại từ hộp giới hạn. Học sâu là bước đột phá thành công đầu tiên trong lĩnh vực phát hiện đối tượng. Thuật toán fast R-CNN được cải thiện trên cơ sở thuật toán R-CNN. Cải tiến chính là cải thiện việc lựa chọn các vùng của đối tượng từ hình ảnh gốc đến ROI (Vùng trung tâm) trong bản đồ đặc tính. Điều này cải thiện đáng kể tốc độ huấn luyện. R-CNN nhanh hơn được cải thiện trên cơ sở Fast R-CNN. Cải tiến chính là đề xuất một mạng RPN để có được ROI, thay thế cho tính chọn lọc chuyên sâu tính toán trước đó Phương pháp tìm kiếm cải thiện hơn nữa tốc độ xử lý của thuật toán. Thuật toán YOLO khác với thuật toán trên. Đây là thuật toán học sâu dựa trên học tập từ đầu đến cuối. Nguyên tắc là trước tiên chia hình ảnh thành nhiều lối, sau đó phân tích các đối tượng trong mỗi lối.

Dự đoán thông tin có ưu điểm là tốc độ phát hiện nhanh và tốc độ phát hiện sai nền thấp. Nhược điểm là lỗi định vị đối tượng lớn. SSD dựa trên YOLO và được cải thiện bằng cách tích hợp ý tưởng đè cử khu vực. Nó cải thiện độ chính xác định vị của các vật thể nhỏ trong khi duy trì phát hiện nhanh.

#### 2.2.5. Thuật toán phân loại đối tượng

Vấn đề của phân loại hình ảnh là đưa ra một hình ảnh (chủ yếu) chỉ chứa một đối tượng duy nhất của một loại và lấy danh mục của các đối tượng mà nó chứa. Các thuật toán truyền thống trong vấn đề này bao gồm suy luận Bayes, máy vectơ hỗ trợ, K lân cận gần nhất và các phương pháp khác, nhưng do số lượng tính toán lớn, chúng đã không đáp ứng được yêu cầu thực tế hiện tại. Hiện nay, phương pháp được sử dụng phổ biến hơn dựa trên mạng thần kinh tích chập.

LeNet là một thuật toán CNN cổ điển, ban đầu đạt được thành công lớn trong nhận dạng chữ số viết tay; AlexNet cũng là một thuật toán CNN cổ điển, đặc điểm của nó là nó bao gồm hai mạng giống nhau song song, có thể chạy trên hai GPU cùng lúc; VGG cũng là một thuật toán CNN cổ điển, độ sâu mạng của nó sâu hơn nhiều so với trước đây và nó đã đạt được kết quả tốt hơn; GoogLeNet đã tiên phong trong việc bổ sung chế độ Inception, Phản ứng tổng hợp bản đồ được xử lý bởi các hạt tích chập có kích thước khác nhau; ResNet để xuất học tập dư, đã thay đổi mạng.

Tóm tắt và triển vọng dựa trên thuật toán phát hiện hình elip và mạng nơ ron tích chập LeNet, thuật toán nhận dạng và kiểm soát biển báo giao thông thời gian thực tốc độ cao. Thuật toán đã đạt được độ chính xác rất cao và khả năng mở rộng tốt trong bộ dữ liệu thử nghiệm và vị trí thử nghiệm thực tế, và có thể đáp ứng các yêu cầu về độ tin cậy và thời gian thực của các hệ thống lái xe không người lái trong tương lai. Tương lai của ô tô không chỉ giới hạn ở một phương tiện giao thông, mà còn phát triển theo hướng một thế hệ thiết bị đầu cuối Internet mới. Xe không người lái tích hợp nhận thức, ra quyết định, điều khiển và phản hồi vào một hệ thống, cho phép tách xe khỏi người lái và đảm bảo khả năng cơ động và an toàn khi lái xe. Sự xuất hiện của lái xe không người lái sẽ thay đổi căn bản các phương pháp điều khiển của ô tô truyền thống, và sẽ đảm bảo rất nhiều cho sự an toàn và hiệu quả giao thông của hệ thống giao thông. Với sự

phát triển liên tục của dữ liệu lớn, Internet of Things và điện toán đám mây, không cần lái xe.

Hiệu suất của xe sẽ hoàn hảo hơn. Là một thị trường mới nổi, lái xe không người lái đòi hỏi phải nghiên cứu sâu về công nghệ và kinh nghiệm. Vẫn còn một chặng đường dài trước khi một hệ thống không người lái được chế tạo và hoàn thiện cho đến khi một chiếc xe không người lái quy mô lớn bắt đầu hoạt động.

Nó đã trở thành xu hướng phát triển của ngành công nghiệp ô tô. Điều chắc chắn là trong tương lai, những chiếc xe không người lái sẽ đi vào cuộc sống của chúng ta, thay đổi và cải thiện cách đi của chúng ta đến một mức độ lớn và giúp nhân loại trong các lĩnh vực bảo vệ môi trường, giao thông, kinh tế, ...

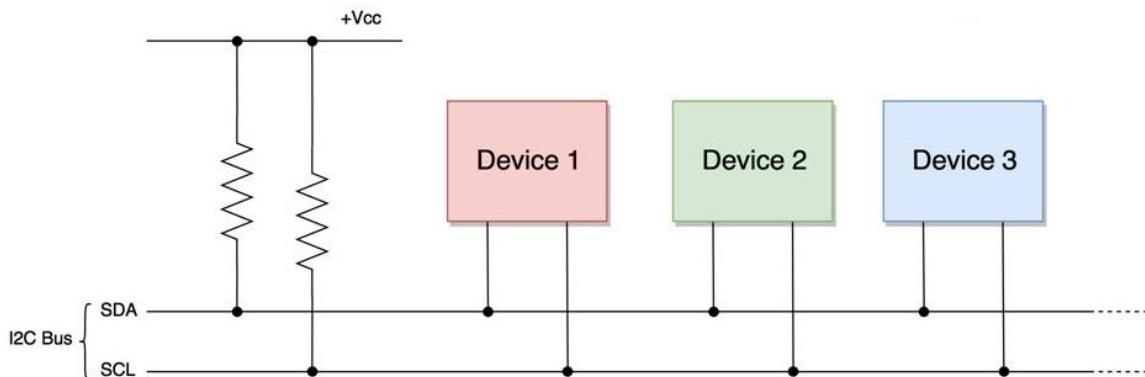
#### 2.2.6. Chuẩn giao tiếp I2C

I2C là tên viết tắt của cụm từ tiếng anh “Inter-Integrated Circuit”. Nó là một giao thức giao tiếp được phát triển bởi Philips Semiconductors để truyền dữ liệu giữa một bộ xử lý trung tâm với nhiều IC trên cùng một board mạch chỉ sử dụng hai đường truyền tín hiệu. Do tính đơn giản của nó nên loại giao thức này được sử dụng rộng rãi cho giao tiếp giữa vi điều khiển và mảng cảm biến, các thiết bị hiển thị, thiết bị IoT, EEPROMs, ...

Đây là một loại giao thức giao tiếp nối tiếp đồng bộ. Nó có nghĩa là các bit dữ liệu được truyền từng bit một theo các khoảng thời gian đều đặn được thiết lập bởi một tín hiệu đồng hồ tham chiếu. Chỉ cần có hai đường bus (dây) chung để điều khiển bất kỳ thiết bị / IC nào trên mạng I2C. Không cần thỏa thuận trước về tốc độ truyền dữ liệu như trong giao tiếp UART. Vì vậy, tốc độ truyền dữ liệu có thể được điều chỉnh bất cứ khi nào cần thiết. Sử dụng hệ thống địa chỉ 7 bit để xác định một thiết bị / IC cụ thể trên bus I2C. Các mạng I2C dễ dàng mở rộng. Các thiết bị mới có thể được kết nối đơn giản với hai đường bus chung I2C.

Bus I2C (dây giao tiếp) chỉ gồm hai dây và được đặt tên là Serial Clock Line (SCL) và Serial Data Line (SDA). Dữ liệu được truyền đi được gửi qua dây SDA và

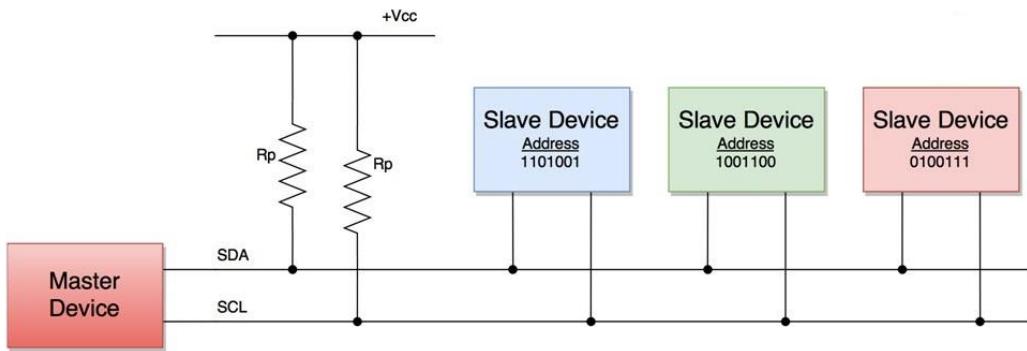
được đồng bộ với tín hiệu đồng hồ (clock) từ SCL. Tất cả các thiết bị / IC trên mạng I2C được kết nối với cùng đường SCL và SDA như sau:



Hình 2.72 Bus vật lý I2C

Cả hai đường bus I2C (SDA, SCL) đều hoạt động như các bộ lái cực máng hở (open drain). Nó có nghĩa là bất kỳ thiết bị / IC trên mạng I2C có thể lái SDA và SCL xuống mức thấp, nhưng không thể lái chúng lên mức cao. Vì vậy, một điện trở kéo lên (khoảng  $1\text{ k}\Omega$  đến  $4,7\text{ k}\Omega$ ) được sử dụng cho mỗi đường bus, để giữ cho chúng ở mức cao (ở điện áp dương) theo mặc định. Lý do sử dụng một hệ thống cực máng hở (open drain) là để không xảy ra hiện tượng ngắn mạch, điều này có thể xảy ra khi một thiết bị cố gắng kéo đường dây lên cao và một số thiết bị khác cố gắng kéo đường dây xuống thấp.

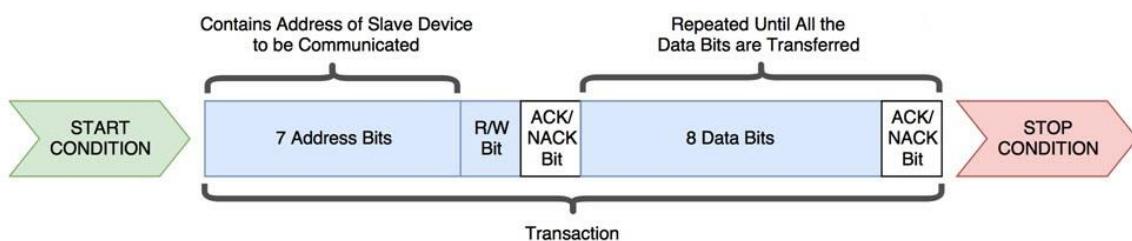
Các thiết bị kết nối với bus I2C được phân loại hoặc là thiết bị Chủ (Master) hoặc là thiết bị Tớ (Slave). Ở bất cứ thời điểm nào thì chỉ có duy nhất một thiết bị Master ở trạng thái hoạt động trên bus I2C. Nó điều khiển đường tín hiệu đồng hồ SCL và quyết định hoạt động nào sẽ được thực hiện trên đường dữ liệu SDA. Tất cả các thiết bị đáp ứng các hướng dẫn từ thiết bị Master này đều là Slave. Để phân biệt giữa nhiều thiết bị Slave được kết nối với cùng một bus I2C, mỗi thiết bị Slave được gán một địa chỉ vật lý 7-bit cố định. Khi một thiết bị Master muốn truyền dữ liệu đến hoặc nhận dữ liệu từ một thiết bị Slave, nó xác định địa chỉ thiết bị Slave cụ thể này trên đường SDA và sau đó tiến hành truyền dữ liệu. Vì vậy, giao tiếp có hiệu quả diễn ra giữa thiết bị Master và một thiết bị Slave cụ thể. Tất cả các thiết bị Slave khác không phản hồi trừ khi địa chỉ của chúng được chỉ định bởi thiết bị Master trên dòng SDA.



Hình 2.73 Thiết bị chủ (Master) và tớ (Slave)

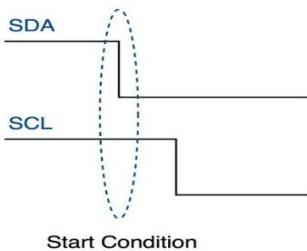
Giao thức sau đây (tập hợp các quy tắc) được theo sau bởi thiết bị Master và các thiết bị Slave để truyền dữ liệu giữa chúng.

Dữ liệu được truyền giữa thiết bị Master và các thiết bị Slave thông qua một đường dữ liệu SDA duy nhất, thông qua các chuỗi có cấu trúc gồm các số 0 và 1 (bit). Mỗi chuỗi số 0 và 1 được gọi là giao dịch (transaction) và dữ liệu trong mỗi giao dịch có cấu trúc như sau:



Hình 2.74 Giao thức truyền dữ liệu I2C

Bất cứ khi nào một thiết bị chủ / IC quyết định bắt đầu một giao dịch, nó sẽ chuyển mạch SDA từ mức điện áp cao xuống mức điện áp thấp trước khi đường SCL chuyển từ cao xuống thấp. Khi điều kiện bắt đầu được gửi bởi thiết bị Master, tất cả các thiết bị Slave đều hoạt động ngay cả khi chúng ở chế độ ngủ (sleep mode) và đợi bit địa chỉ.



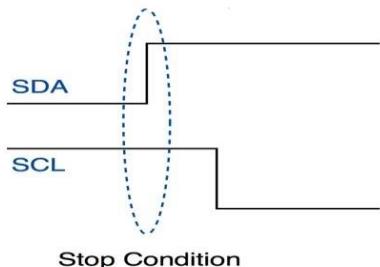
Hình 2.75 Điều kiện bắt đầu I2C

Nó bao gồm 7 bit và được lắp đầy với địa chỉ của thiết bị Slave đến / từ đó thiết bị Master cần gửi / nhận dữ liệu. Tất cả các thiết bị Slave trên bus I2C so sánh các bit địa chỉ này với địa chỉ của chúng. Bit read/write xác định hướng truyền dữ liệu. Nếu thiết bị Master / IC cần gửi dữ liệu đến thiết bị Slave, bit này được thiết lập là ‘0’. Nếu IC Master cần nhận dữ liệu từ thiết bị Slave, bit này được thiết lập là ‘1’.

ACK / NACK là viết tắt của Acknowledged/Not-Acknowledged. Nếu địa chỉ vật lý của bất kỳ thiết bị Slave nào trùng với địa chỉ được thiết bị Master phát, giá trị của bit này được set là ‘0’ bởi thiết bị Slave. Ngược lại, nó vẫn ở mức logic ‘1’ (mặc định).

Khối dữ liệu bao gồm 8 bit và chúng được thiết lập bởi bên gửi, với các bit dữ liệu cần truyền tới bên nhận. Khối này được sau bởi một bit ACK / NACK và được set thành ‘0’ bởi bên nhận nếu nó nhận thành công dữ liệu. Ngược lại, nó vẫn ở mức logic ‘1’. Sự kết hợp của khối dữ liệu sau bởi bit ACK / NACK được lặp lại cho đến quá trình truyền dữ liệu được hoàn tất.

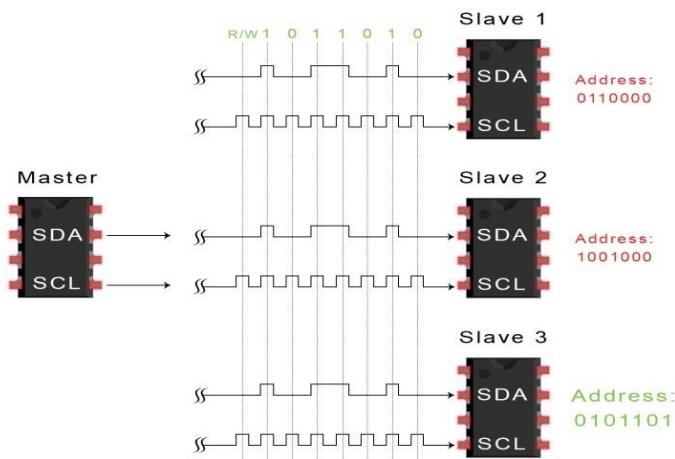
Sau khi các khung dữ liệu cần thiết được truyền qua đường SDA, thiết bị Master chuyển đường SDA từ mức điện áp thấp sang mức điện áp cao trước khi đường SCL chuyển từ cao xuống thấp.



Hình 2.76 Điều kiện kết thúc truyền dữ liệu I2C

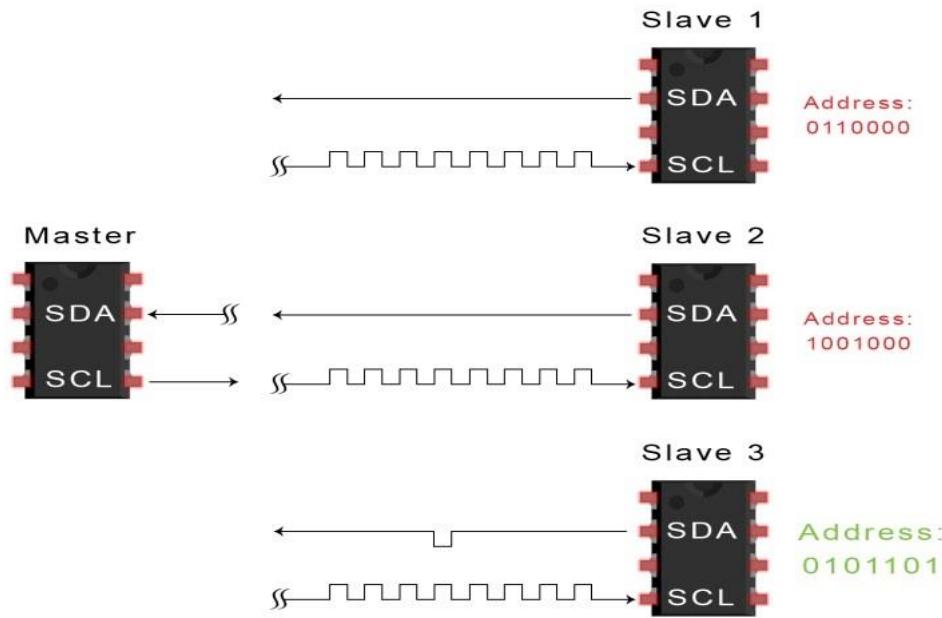
Giao tiếp I2C được bắt đầu bởi thiết bị Master hoặc để gửi dữ liệu đến thiết bị Slave hoặc nhận dữ liệu từ thiết bị đó.

Trình tự hoạt động sau đây diễn ra khi một thiết bị Master gửi dữ liệu đến một thiết bị Slave cụ thể thông qua bus I2C. Thiết bị Master gửi điều kiện bắt đầu đến tất cả các thiết bị Slave. Thiết bị Master gửi 7 bit địa chỉ của thiết bị Slave mà thiết bị Master muốn giao tiếp cùng với bit Read/Write.



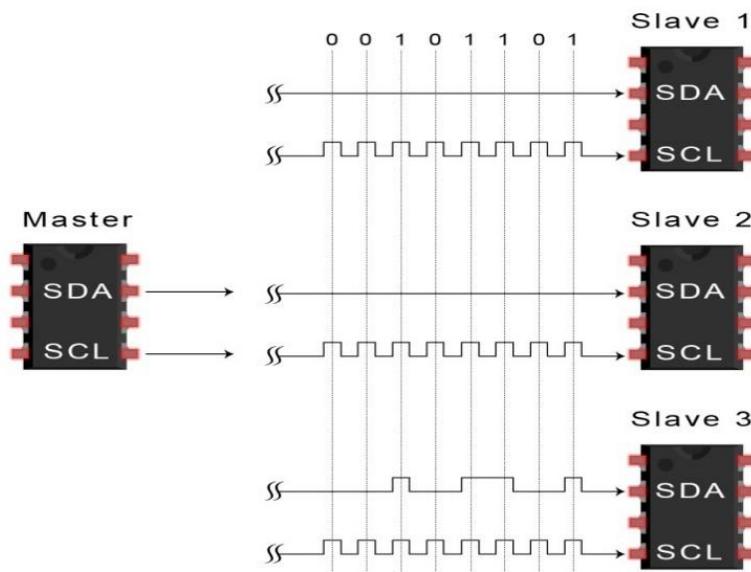
Hình 2.77 Gửi dữ liệu đến thiết bị Slave

Mỗi thiết bị Slave so sánh địa chỉ được gửi từ thiết bị Master đến địa chỉ riêng của nó. Nếu địa chỉ trùng khớp, thiết bị Slave gửi về một bit ACK bằng cách kéo đường SDA xuống thấp và bit ACK / NACK được thiết lập là ‘0’. Nếu địa chỉ từ thiết bị Master không khớp với địa chỉ riêng của thiết bị Slave thì đường SDA ở mức cao và bit ACK / NACK sẽ ở mức ‘1’ (mặc định).



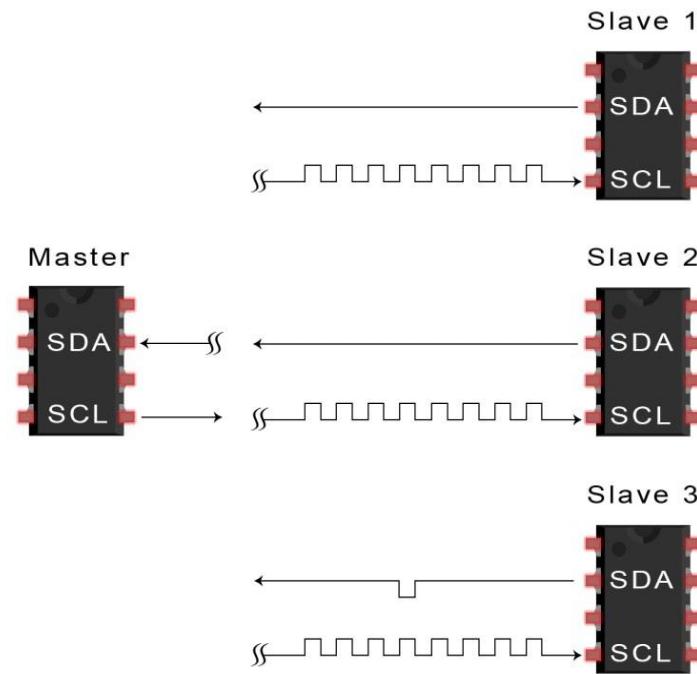
Hình 2.78 Gửi dữ liệu đến thiết bị Slave

Thiết bị Master gửi hoặc nhận khung dữ liệu. Nếu thiết bị Master muốn gửi dữ liệu đến thiết bị Slave, bit Read / Write là mức điện áp thấp. Nếu thiết bị Master đang nhận dữ liệu từ thiết bị Slave, bit này là mức điện áp cao.



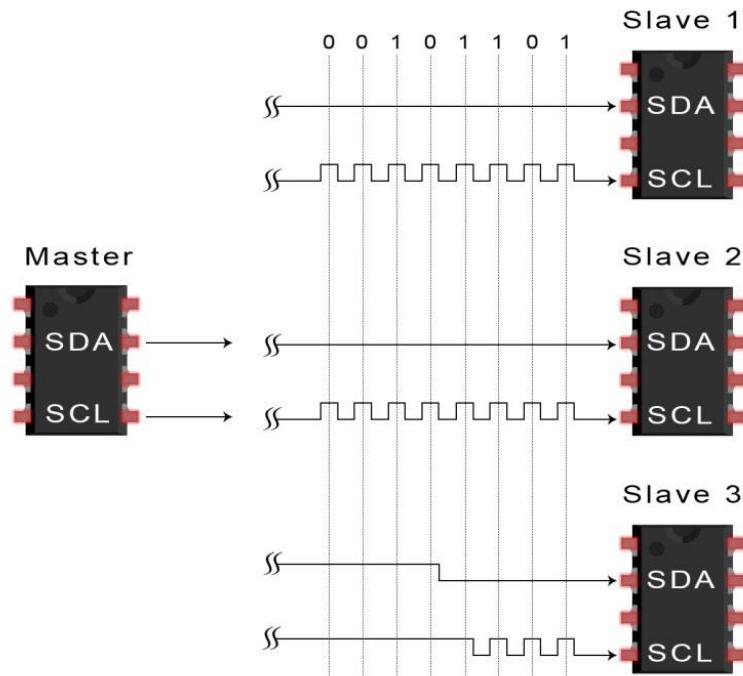
Hình 2.79 Gửi dữ liệu đến thiết bị Slave

Nếu khung dữ liệu được thiết bị Slave nhận được thành công, nó sẽ thiết lập bit ACK / NACK thành ‘0’, báo hiệu cho thiết bị Master tiếp tục

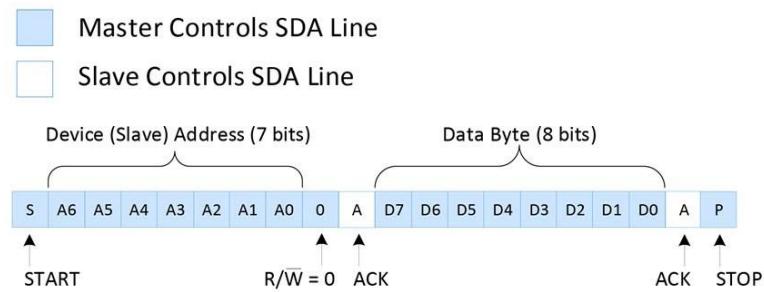


Hình 2.80 Gửi dữ liệu đến thiết bị Slave

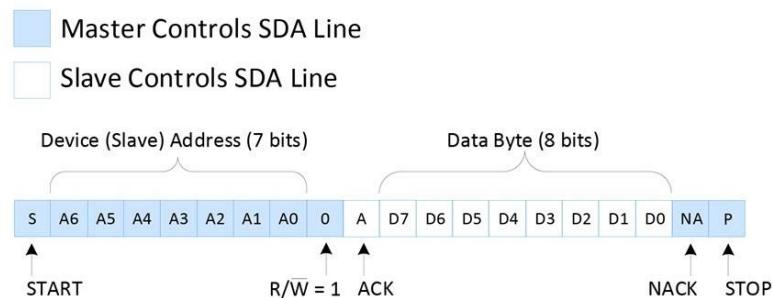
Sau khi tất cả dữ liệu được gửi đến thiết bị Slave, thiết bị Master gửi điều kiện dừng để báo hiệu cho tất cả các thiết bị Slave biết rằng việc truyền dữ liệu đã kết thúc [12].



Hình 2.81 Gửi dữ liệu đến thiết bị Slave



Hình 2.82 Khung dữ liệu Master gửi cho Slave



Hình 2.83 Khung dữ liệu Master nhận từ Slave

### 3. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

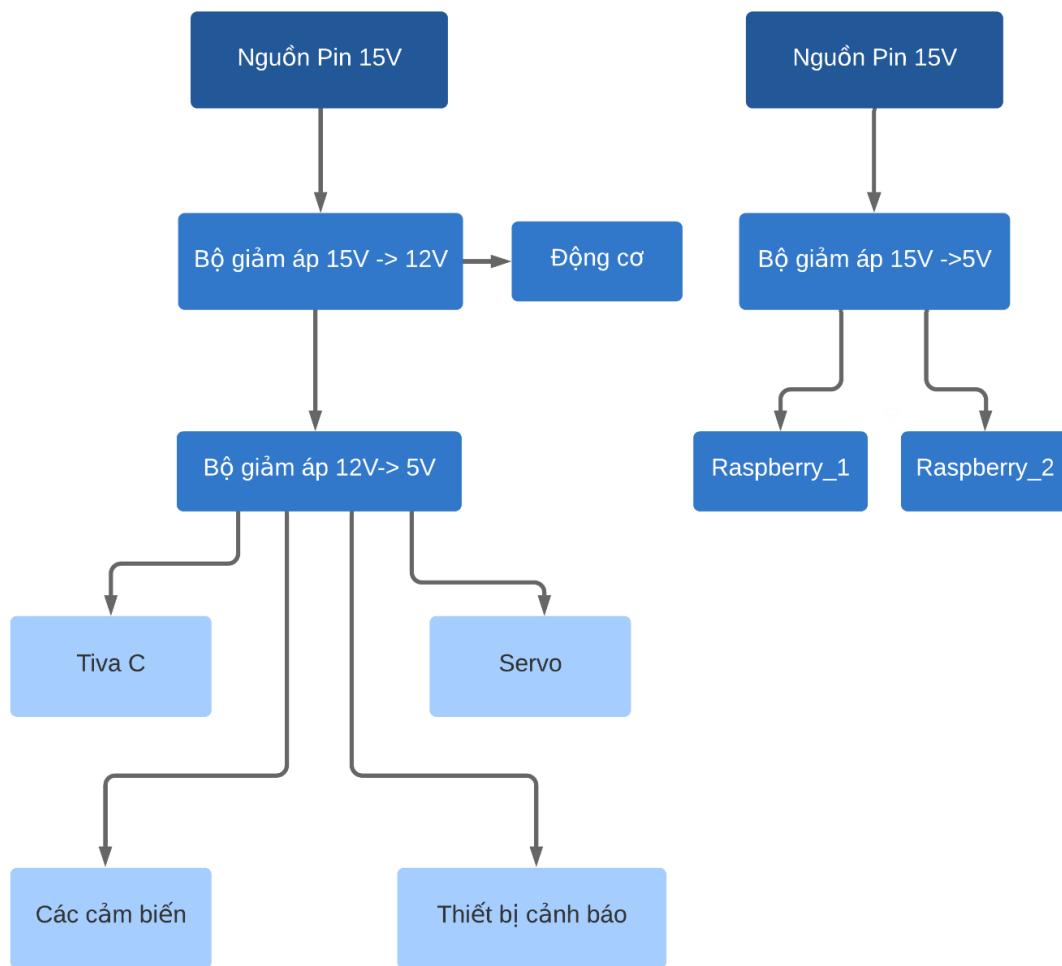


Hình 3.1. Sơ đồ khái niệm tổng quát của hệ thống

Hệ thống được thiết kế gồm có các khối cơ bản như hình 3.1. Khối cung cấp nguồn là khối cơ bản nhất trong tất cả các hệ thống điện nói chung và điện tử nói riêng, khối này gồm các pin cung cấp năng lượng điện cho hệ thống có thể hoạt động được. Khối nhận dạng làn đường thực hiện chức năng nhận dạng vạch kẻ trắng trên đường nhờ vào camera, sau đó khung hình được đưa vào Raspberry Pi để xử lý và gửi tín hiệu cho vi điều khiển trung tâm. Khối nhận dạng biển báo giao thông thực hiện chức năng chính là xác định được biển báo trong khung hình và nhận dạng biển báo đó là loại biển báo gì từ đó gửi tín hiệu xuống vi điều khiển trung tâm. Khối cảm biến bao gồm các cảm biến được gắn ở xung quanh xe như cảm biến lazer để đo khoảng cách với các vật ở phía trước xe, cảm biến hồng ngoại xung quanh xe và cảm biến hồng ngoại chuyển động đặt ở phía sau xe. Khối cảnh báo có chức năng chính là phát ra tín hiệu cảnh báo bao gồm tín hiệu đèn và tín hiệu còi trong các trường hợp cần thiết cho các phương tiện

khi tham gia giao thông được biết. Khối quan trọng nhất là khối xử lý trung tâm, chức chính là xử lý các tín hiệu nhận từ các vi điều khiển khác cũng như các tín hiệu từ cảm biến để đưa ra hướng xử lý cho phù hợp. Khối cuối cùng cũng không kém phần quan trọng đó chính là khói thực thi, khói này nhận tín hiệu điều khiển từ khói xử lý trung tâm để thực thi trong từng tình huống cụ thể.

### 3.1 Khối cung cấp năng lượng

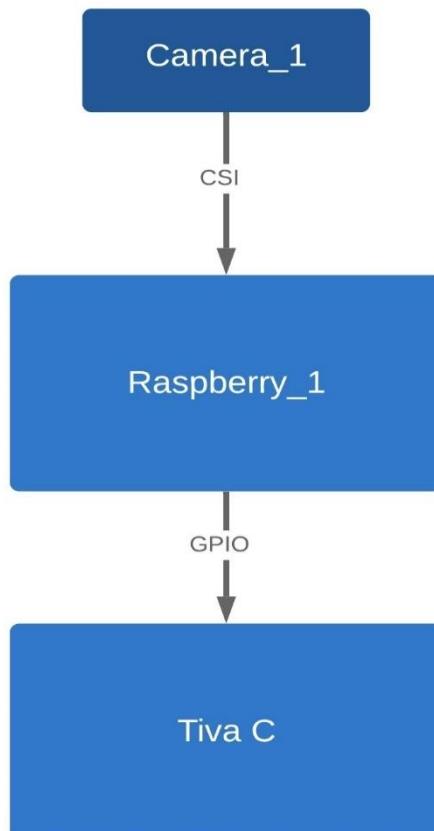


*Hình 3.2. Sơ đồ khái niệm hệ thống cung cấp năng lượng*

Khối cung cấp năng lượng cho hệ thống này gồm hai nguồn năng lượng độc lập nhau, có dung lượng như nhau, hai nguồn này đều sử dụng 4 pin Lipo 7.4V 1400mAh với dòng xả 30C, bốn pin này được nối song song với nhau thành hai cặp pin 7.4V và hai cặp pin này được mắc nối tiếp với nhau thành khối pin 14.8V (nếu trong trạng thái xạc đầy thì khói nguồn này sẽ có điện áp lớn hơn 15V). Khối pin thứ nhất sẽ được đưa qua mạch giảm áp từ 15V xuống 12V, với mức điện áp này thì sẽ được cung cấp cho

động cơ DC hoạt động. Sau đó từ nguồn 12V có sẵn thì ta sẽ giảm áp một lần nữa xuống mức điện áp 5V. Với mức điện áp ổn định là 5V, ta sẽ cung cấp cho mạch Tiva C, động cơ Servo RC, các cảm biến hồng ngoại, cảm biến laser, đèn và còi cảnh báo. Nguồn thứ hai cũng có cấu trúc pin tương tự như nguồn pin thứ nhất, nhưng điều đặc biệt ở đây chính là nguồn pin này chỉ cung cấp năng lượng cho hai board mạch Raspberry Pi, do yêu cầu khắt khe về nguồn cung cấp cho hai board mạch này, mỗi board được yêu cầu điện áp 5V và dòng cấp từ 3A, và sự ổn định của dòng cung cấp, do đó ta cần cấp nguồn riêng cho hai board mạch này. Với khói pin được thiết kế tương tự như khói pin thứ nhất nên ta cũng có điện áp ra từ nguồn pin này là 15V, để có điện áp 5V cung cấp cho board Raspberry Pi thì ta cần dùng dùng hai mạch giảm áp 15V xuống 5V để cung cấp điện áp cho mỗi mạch.

### 3.2 Khối nhận dạng làn đường



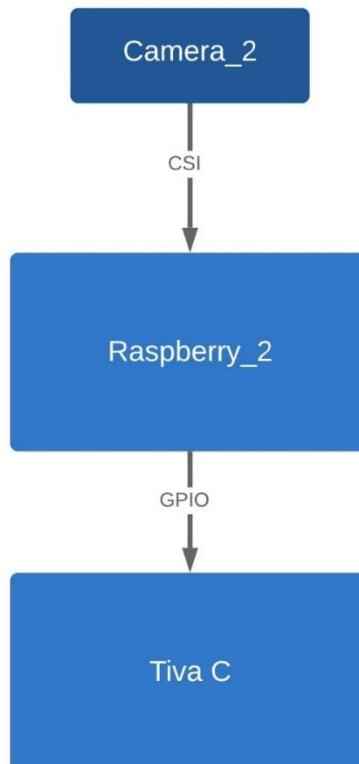
Hình 3.3 Sơ đồ khói nhận dạng làn đường

Khối nhận dạng làn đường bao gồm hai phần cơ bản gồm camera và Raspberry Pi. Camera được sử dụng là camera chuyên dùng cho Raspberry Pi có độ phân giải là 5 MP, module camera này có tác dụng chính là ghi lại hình ảnh từ môi trường bên ngoài và truyền tín hiệu về cho board mạch chính Raspberry để xử lý hình ảnh. Trên board mạch Raspberry có trang bị cổng CSI nên có thể giao tiếp với module camera. Khi đã nhận được thông tin hình ảnh truyền về từ camera thì Raspberry sẽ tiến hành xử lý và xuất tín hiệu sang Tiva C thông qua các chân GPIO.

*Bảng 3.1 Giao tiếp GPIO giữa Tiva C và Raspberry\_1*

Tiva C	Raspberry_1
A2	GPIO_25
A3	GPIO_8
A4	GPIO_7
A5	GPIO_1

### 3.3 Khối nhận dạng biển báo



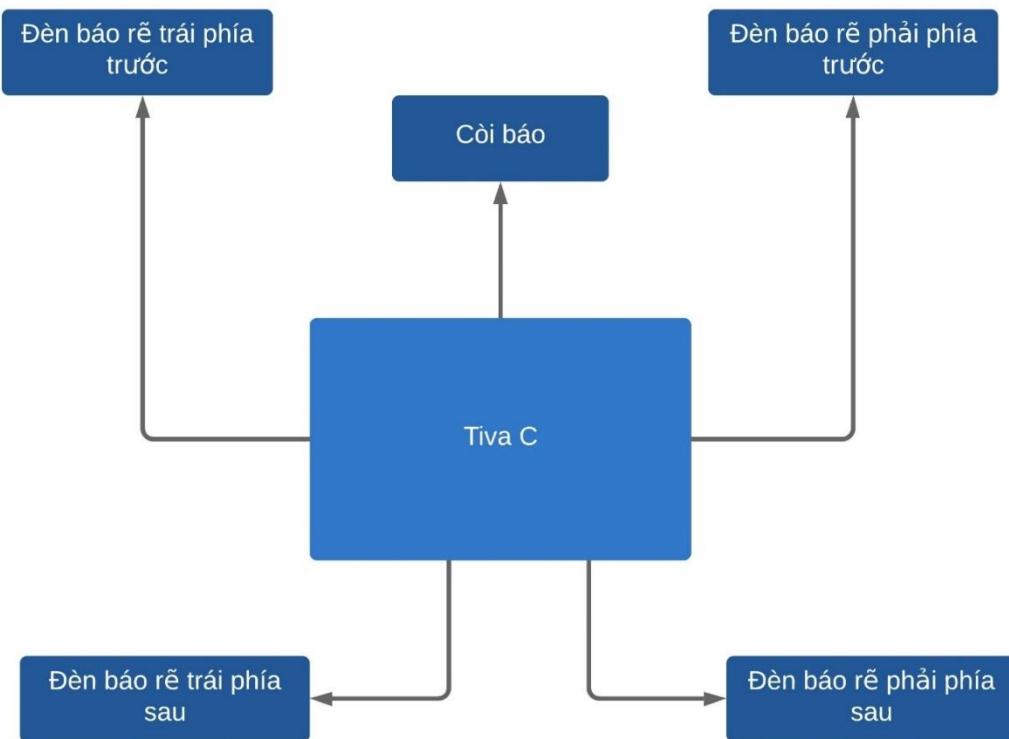
*Hình 3.4 Sơ đồ khái niệm nhận dạng biển báo*

Cũng tương tự như khối nhận dạng làn đường đã nói trên, khối nhận dạng biển báo giao thông cũng bao gồm Raspberry Pi và module camera. Camera truyền hình ảnh thu được về cho Raspberry, sau đó Raspberry xử lý các hình ảnh nhận được và truyền tín hiệu về cho Tiva C.

*Bảng 3.2 Giao tiếp giữa GPIO giữa Tiva C và Raspberry\_2*

Tiva C	Raspberry_1
A2	GPIO_25
A3	GPIO_8
A4	GPIO_7
A5	GPIO_1

### 3.4 Khối cảnh báo



*Hình 3.5. Sơ đồ khái niệm hệ thống cảnh báo*

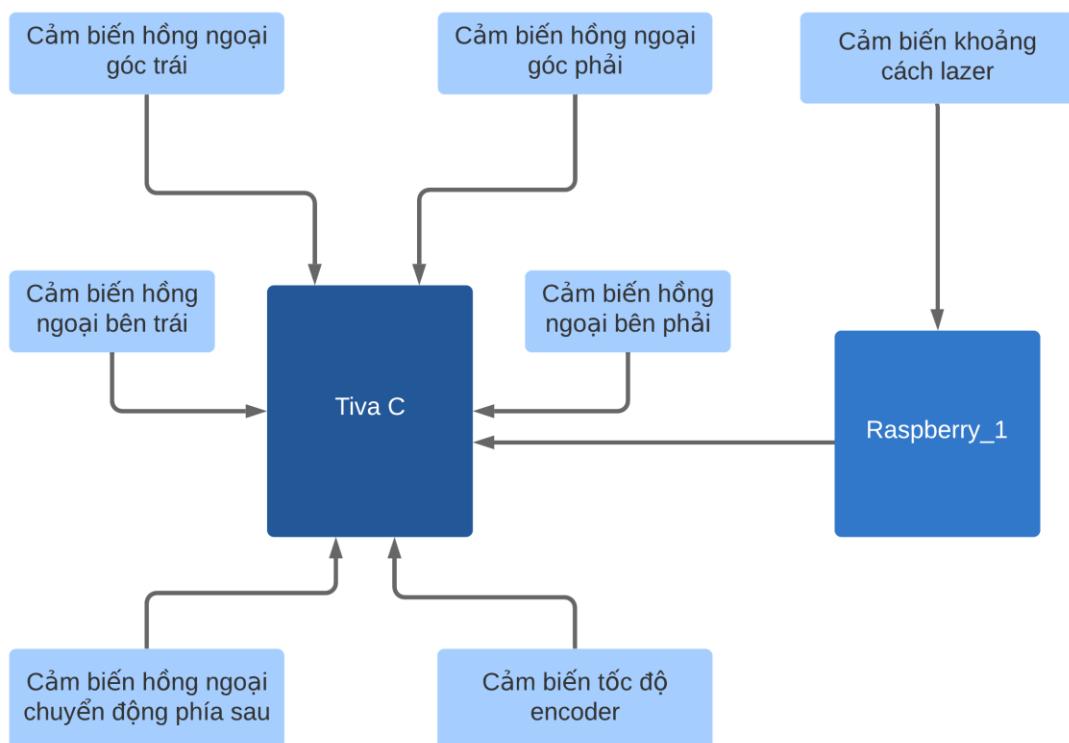
Khối cảnh báo có chức năng quan trọng trong việc đưa ra các tín hiệu cho các phương tiện tham gia giao thông được biết từ đó hạn chế được rủi ro không đáng có khi di chuyển trên đường. Khối này gồm hai phần chính đó là còi và đèn cảnh cáo. Còi được sử dụng ở đây là buzzer, thiết bị phát ra âm thanh cảnh báo thông dụng trong các thiết

bị điện, để người dùng chú ý đến điều bất thường mà thiết bị đang gặp phải. Đèn cảnh báo được thiết kế gồm bốn dãy đèn LED được đặt tại bốn góc của xe. Các đèn và còi báo này được điều khiển bởi Tiva C.

Bảng 3.3 Kết nối giữa Tiva C, LED và buzzer

Tiva C	Kết nối với ngoại vi
A6	LED_8
A7	Buzzer
B4	LEDS_LEFT_CONTROL
B5	LEDS_RIGHT_CONTROL
C4	LED_1
C5	LED_2
C6	LED_3
C7	LED_4
D2	LED_5
D3	LED_6
D6	LED_7
E1	LED_5
E2	LED_6

### 3.5 Khối cảm biến



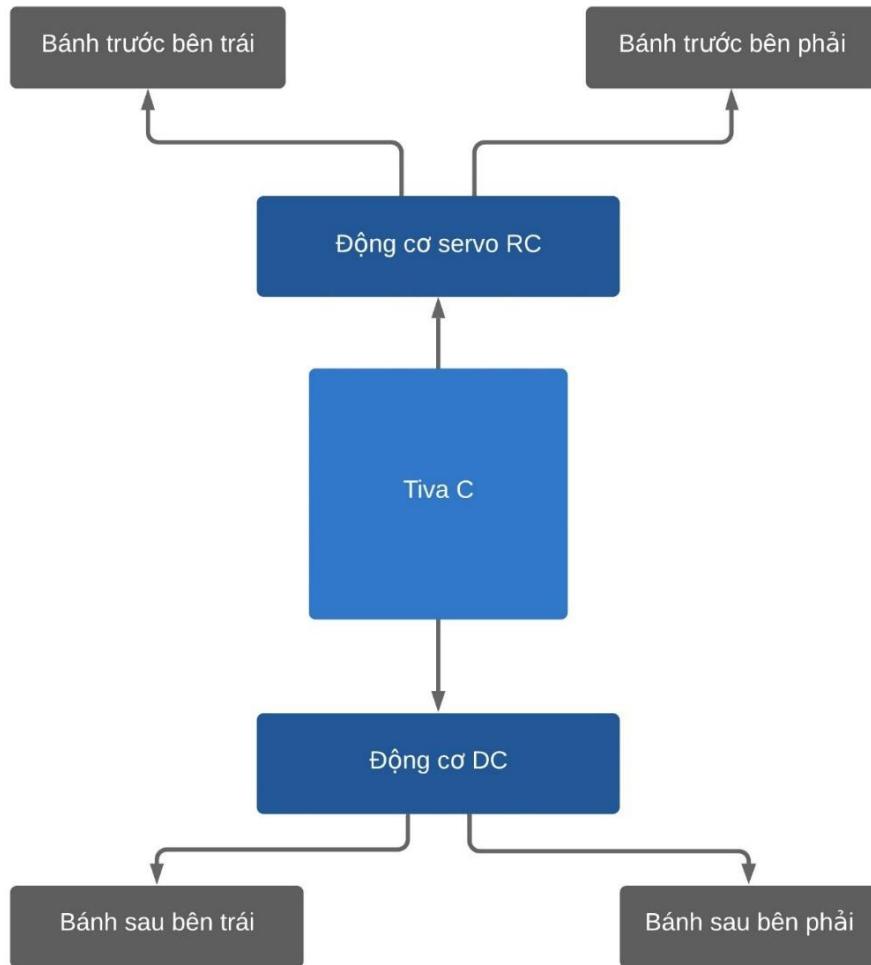
Hình 3.6 Khối cảm biến

Khối cảm biến được thiết kế gồm có các cảm biến khoảng cách hồng ngoại, cảm biến khoảng cách lazer, cảm biến tốc độ encoder và cảm biến hồng ngoại vật chuyển động. Cảm biến khoảng cách lazer được đặt ở trước đầu xe dùng để xác định khoảng cách với các vật thể ở phía trước, tín hiệu của cảm biến này được đọc ở board mạch Raspberry. Ở phía trước góc đầu xe được trang bị cảm biến khoảng cách hồng ngoại, hai bên hông xe cũng được trang bị một cảm biến tương tự như vậy ở mỗi bên. Bốn cảm biến hồng ngoại này dùng để hỗ trợ xe trong quá trình chuyển làn khi xe gặp vật cản. Phía sau xe cũng được gắn một cảm biến hồng ngoại vật chuyển động nhằm hỗ trợ hệ thống phát hiện vật chuyển động ở phía sau. Xe còn được trang bị một cảm biến tốc độ encoder nhằm hỗ trợ xe trong việc chuyển làn, được gắn chung với trục quay của động cơ DC.

Bảng 3.4 Kết nối giữa Tiva C và các cảm biến

Tiva C	Kết nối với ngoại vi
B0	Cảm biến hồng ngoại góc trái
B1	Cảm biến hồng ngoại cạnh trái
B2	Cảm biến hồng ngoại góc phải
B3	Cảm biến hồng ngoại cạnh phải
E0	Cảm biến tốc độ Encoder
F2	Cảm biến hồng ngoại vật chuyển động

### 3.6 Khối thực thi



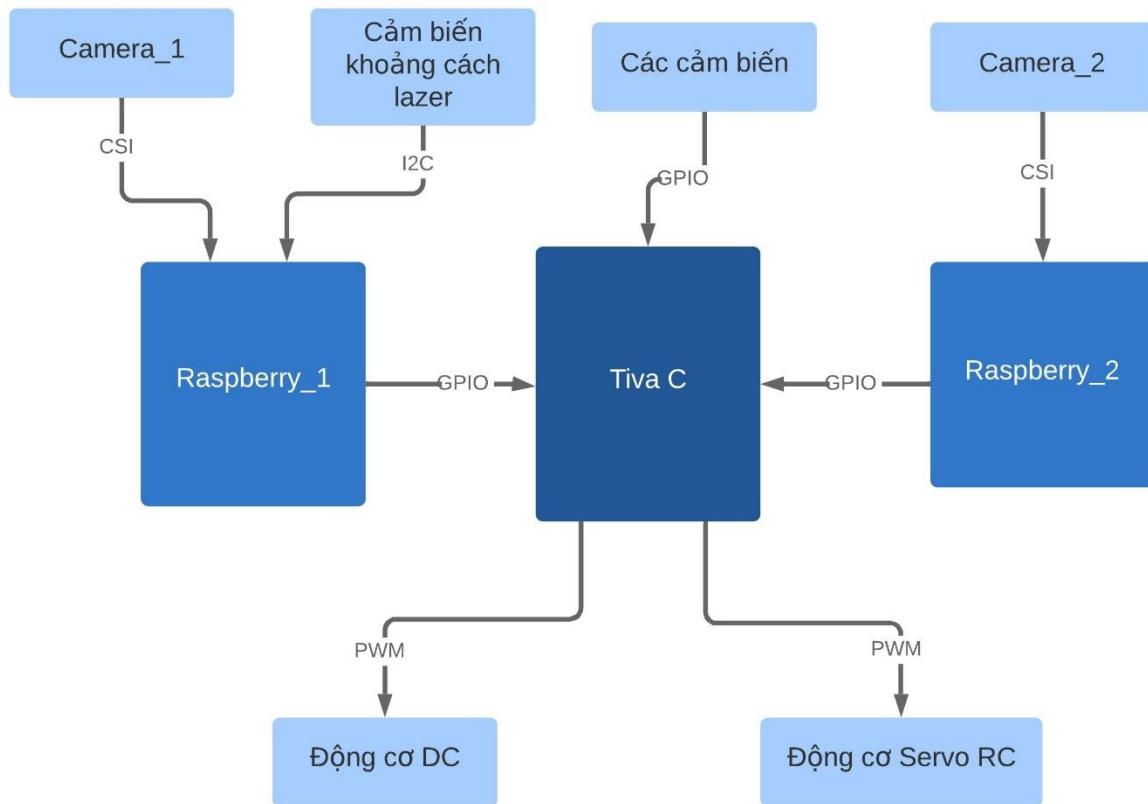
*Hình 3.7 Khối thực thi*

Sau khi các tín hiệu nhận được từ hai Raspberry và các cảm biến được tổng hợp và xử lý trên Tiva C, tín hiệu điều khiển sẽ được gửi xuống khối thực thi (bao gồm động cơ DC và servo RC). Hai bánh sau được gắn cố định vào trục và được điều khiển tốc độ bởi động cơ DC. Hai bánh này tạo lực đẩy để xe tiến về phía trước, khi điều khiển tốc độ của xe ta chỉ cần điều khiển tốc độ động cơ DC với tốc độ mà ta mong muốn. Hai bánh trước được thả nổi không được gắn động cơ chuyển động. Hai bánh này được gắn vào trục điều khiển góc lái bởi động cơ servo RC. Vì thế, để điều khiển góc lái của xe ta chỉ cần điều khiển góc servo RC thì góc lái của xe được điều khiển theo. Tóm lại, để điều khiển hệ thống chuyển động của xe thì ta cần điều khiển động cơ DC và servo RC.

*Bảng 3.5 Kết nối giữa Tiva C và các động cơ*

Tiva C	Kết nối với ngoại vi
D0	Điều khiển PWM (Servo)
D1	Điều khiển PWM (Động cơ)

### 3.7 Khối xử lý trung tâm

*Hình 3.8. Sơ đồ khái niệm giao tiếp các board mạch*

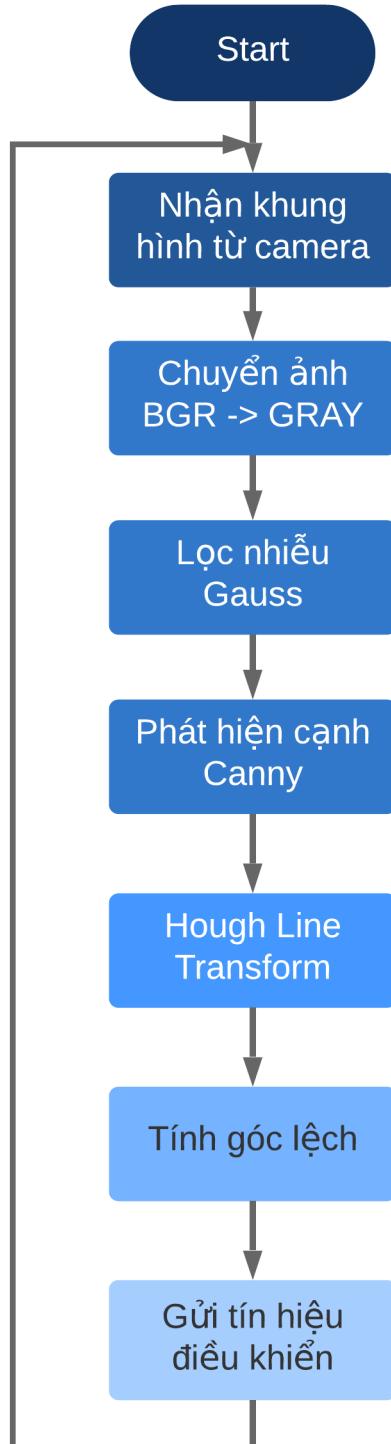
Khối xử lý trung tâm có vai trò đặc biệt quan trọng trong các hệ thống. Tiva C được chọn là khối xử lý trung tâm cho hệ thống này. Tiva C có nhiệm vụ tổng hợp các tín hiệu nhận được từ các cảm biến cũng như tín hiệu nhận được từ hai board Raspberry. Từ các tín hiệu từ môi trường xung quanh thì Tiva C sẽ thực hiện phân tích dữ liệu để thực hiện truyền tín hiệu điều khiển đến khối thực thi cũng như là khống cảnh báo trong các điều kiện cần thiết. Tiva C sẽ nhận tín hiệu qua giao tiếp với GPIO trên board và điều khiển các động cơ bằng xung PWM.

*Bảng 3.6 Kết nối tổng quát của Tiva C*

Tiva C	Raspberry_1	Raspberry_2	Kết nối với ngoại vi	Chức năng
A2	GPIO_25			Nhận tín hiệu từ Raspberry
A3	GPIO_8			Nhận tín hiệu từ Raspberry
A4	GPIO_7			Nhận dạng làn đường
A5	GPIO_1			Nhận dạng làn đường
A6	`		LED_8	
A7			Buzzer	
B0			Cảm biến hồng ngoại trái	
B1			Cảm biến hồng ngoại trái	
B2			Cảm biến hồng ngoại phải	
B3			Cảm biến hồng ngoại phải	
B4			Điều khiển LED trái	
B5			Điều khiển LED phải	
B6				D0
B7				D1
C4			LED_1	
C5			LED_2	
C6			LED_3	
C7			LED_4	
D0			Điều khiển PWM (Servo)	
D1			Điều khiển PWM (Động cơ)	
D2			LED_5	
D3			LED_6	
D4	x			
D5	x			
D6			LED_7	
D7	x			
E0			Encoder	
E1			LED_5	
E2			LED_6	
E3		GPIO_8		Nhận dạng biển báo
E4		GPIO_7		Nhận dạng biển báo
E5		GPIO_1		Nhận dạng biển báo
F0				
F1				
F2			Cảm biến PIR	
F3				
F4				

## 4. THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM

### 4.1. Nhận diện làn đường với Raspberry Pi



Hình 4.1 Sơ đồ giải thuật nhận dạng làn đường

Với chương trình xử lý ảnh - nhận dạng làn đường, camera sẽ truyền các khung hình từ môi trường cho mạch Raspberry, tại đây dữ liệu sẽ được xử lý, các ảnh này được biểu diễn dưới dạng ma trận hai chiều. Khung hình khi nhận được là khung ảnh màu có dạng BRG, để thuận tiện cho việc xử lý thì ma trận ảnh màu sẽ chuyển thành ma trận ảnh xám. Sau khi chuyển đổi xong thì ảnh xám sẽ được áp dụng các giải thuật để lọc nhiễu phát hiện cạnh, chuyển sang không gian Hough để phát hiện và nhận dạng được được vạch kẻ đường, đó chính là sự khác nhau, tương phản về màu sắc giữa nền đường và vạch kẻ đường, dựa vào các đường thẳng là các vạch kẻ đường, ta sẽ xác định được góc lệch của xe so với làn đường, khi độ lệch vượt ngưỡng cho phép thì Raspberry\_1 sẽ gửi tín hiệu điều hướng cho Tiva C để thực hiện điều khiển góc lái sao cho đúng.

#### 4.2. Phát hiện và nhận diện biển báo giao thông dùng Raspberry Pi

##### 4.2.1. Tiền xử lý ảnh

Để cải thiện độ chính xác của các tác vụ phân loại hình ảnh và đáp ứng các yêu cầu thời gian thực của hệ thống, cần thực hiện các thao tác cấp thấp trên các khung hình được chụp bởi máy ảnh, xóa thông tin dư thừa không cần thiết và làm nổi bật thông tin tính năng quan trọng.

###### 4.2.1.1. Không gian màu HSV



*Hình 4.2. Các biển báo giao thông cơ bản*

Có thể thấy rằng màu sắc của các biển báo giao thông sau đây chủ yếu là màu đỏ và màu xanh. Chúng ta có thể trích xuất dải màu quan tâm trong một không gian màu cụ thể. Các dấu hiệu giao thông cần được nhận biết bởi vì trong không gian màu RGB

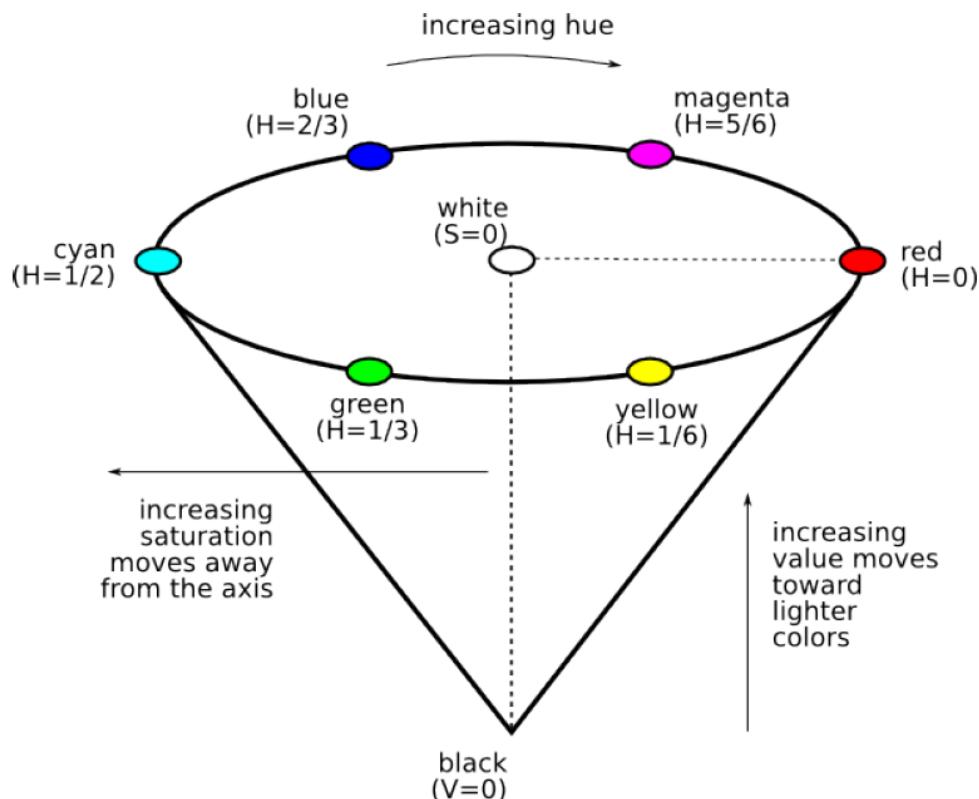
truyền thống, các giá trị của ba thành phần màu được xếp chồng lên nhau, không dễ dàng trích xuất màu của một màu sắc nhất định, mô hình HSV tốt hơn cho nhiệm vụ này. Mô hình HSV là ánh xạ phi tuyến tính của mô hình RGB và mối quan hệ chuyển đổi của chúng như sau:

$$V \leftarrow \max(R, G, B) \quad [3.1.1 - 1]$$

$$S \leftarrow \begin{cases} \frac{V - \min(R, G, B)}{V} & \text{if } V \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad [3.1.1 - 2]$$

$$H \leftarrow \begin{cases} 60(G - B)/(V - \min(R, G, B)) & \text{if } V = R \\ 120 + 60(B - R)/(V - \min(R, G, B)) & \text{if } V = G \\ 240 + 60(R - G)/(V - \min(R, G, B)) & \text{if } V = B \end{cases} \quad [3.1.1 - 3]$$

Theo công thức, chúng ta có thể lấy sơ đồ mô hình HSV như hình bên dưới. Chúng ta có thể thấy rằng thành phần màu (H) phân tách màu sắc tốt trên mặt phẳng góc. Chúng ta chỉ cần trích xuất các thành phần H trong một vài phạm vi và tương ứng Giới hạn các thành phần S và V để có được màu chúng ta muốn tách.



Hình 4.3. Hệ màu HSV

#### 4.2.1.2. Xử lý ảnh theo hình thái

Như chúng ta biết, phép toán giãn nở sẽ phóng to các thành phần của hình ảnh và phép toán co sẽ thu nhỏ các thành phần của hình ảnh. Hai hoạt động hình thái quan trọng khác có thể được kết hợp bằng phép toán co và giãn nở: phép toán mở và phép toán đóng. Thao tác mở thường làm mịn đường viền của vật thể, phá vỡ cổ hẹp và loại bỏ các phần nhô ra nhỏ. Thao tác đóng cũng có xu hướng làm trơn một phần đường viền, nhưng ngược lại với thao tác mở, nó thường hợp nhất các khe nứt hẹp và các rãnh dài, loại bỏ các lỗ nhỏ và lắp đầy các khoảng trống trong đường viền. Hoạt động mở của phần tử cấu trúc B trên tập A được biểu thị là  $A \circ B$ , được định nghĩa như sau:

$$A \circ B = (A \ominus B) \oplus B$$

Do đó, thao tác mở của B đến A là sự co của B đến A, và sau đó B được sử dụng để mở rộng kết quả. Hoạt động đóng của phần tử cấu trúc B trên tập A được biểu thị là  $A \cdot B$  và được định nghĩa như sau:

$$A \cdot B = (A \oplus B) B$$

Đó là tính hai mặt với hoạt động mở. Hoạt động đóng của B thành A là sự mở rộng của B thành A, và sau đó kết quả của B được sử dụng để co.

#### 4.2.1.3. Tính toán miền được kết nối

Do các yếu tố cấu trúc cho mỗi hoạt động hình thái là chắc chắn, nên việc xử lý hình thái một mình thường không đạt được kết quả mong đợi hoặc có thể khiến hình ảnh bị mất một phần thông tin. Ở đây, ta xem xét sử dụng phương pháp lấy các miền được kết nối để xóa các khu vực có khu vực được kết nối (nghĩa là tổng số pixel có trong khu vực được kết nối) nhỏ hơn một ngưỡng nhất định. Trong một hình ảnh, đơn vị nhỏ nhất là một pixel. Có 8 pixel liền kề xung quanh mỗi pixel. Có hai mối quan hệ kề nhau: 4 liền kề và 8 liền kề. Liền kề với tổng cộng 4 điểm, cụ thể là lên, xuống, trái và phải. Có tổng cộng 8 điểm liền kề, bao gồm các điểm trên đường chéo.

#### 4.2.1.4. Phát hiện hình elip

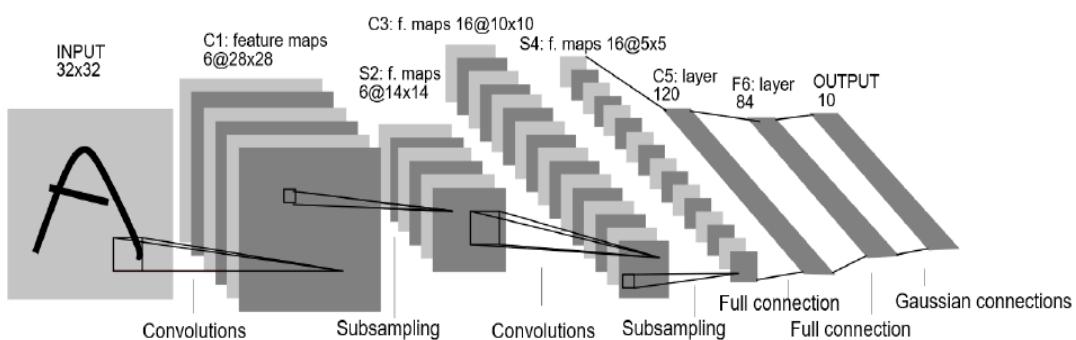
Vì các biển báo giao thông cần được nhận dạng đều là hình tròn (biển báo STOP là gần đúng), chúng ta có thể sử dụng hình dạng như một tính năng chính để trích xuất

các biến báo giao thông và cắt chúng. Do thuật toán phát hiện hình elip cụ thể quá phức tạp, nên ta cần xem xét sử dụng phương pháp trích xuất đường viền của đối tượng. Quá trình chung như sau:

- Phát hiện và đánh dấu tất cả các đường viền trong biểu đồ nhị phân, trả về tọa độ và mức của từng điểm đường viền.
- Kiểm tra các điểm đường viền cùng cấp và kiểm tra xem chúng có tạo thành một vòng tròn hay không, nghĩa là chúng có cách đều nhau từ một điểm trung tâm nhất định không.
- Trong số tất cả các hình elip được tìm thấy, chọn hình elip lớn nhất làm kết quả phát hiện hiện tại.
- Cắt hình elip ra khỏi khung hình để thực hiện phân loại biển báo.

#### 4.2.2. Phân loại mạng thần kinh chuyển đổi LeNet

Mạng thần kinh chuyển đổi LeNet lần đầu tiên được LeCun đề xuất vào năm 1998. Đây là một mạng lưới thần kinh tích chập rất hiệu quả để nhận dạng ký tự viết tay. Mạng thần kinh tích chập LeNet có 7 lớp, không chứa đầu vào và mỗi lớp chứa các tham số có thể huấn luyện, mỗi lớp có nhiều mạng đặc trưng và mỗi mạng đặc trưng sử dụng bộ lọc tích chập để trích xuất một tính năng của đầu vào. Nhiều tế bào thần kinh. Đầu tiên, kích thước hình ảnh đầu vào được chuẩn hóa đồng đều thành  $32 \times 32$ . Tiếp theo, mạng 7 lớp sẽ được giới thiệu chi tiết.<sup>[11]</sup>



Hình 4.4. Mạng thần kinh tích chập LeNet

#### 4.2.2.1. Lớp C1-Convolutional

Sử dụng 6 kernel tích chập có kích thước  $5 \times 5$  để thực hiện thao tác tích chập đầu tiên trên mạng đặc trưng đầu vào và nhận 6 mạng đặc trưng đầu ra với kích thước  $28 \times 28$ . Kích thước của hạt nhân chập là  $5 \times 5$ , do đó có tổng cộng  $6 \times (5 \times 5 + 1) = 156$  tham số, trong đó  $+1$  có nghĩa là hạt nhân có độ lệch. Đối với lớp chập C1, mỗi pixel được kết nối với  $5 \times 5$  pixel và 1 offset trong hình ảnh đầu vào, do đó có tổng cộng  $156 \times 28 \times 28 = 122304$  kết nối.

*Bảng 4.1. Bảng tham số lớp C1-Convolutional*

Tên	Tham số
Mạng đặc trưng đầu vào	32x32
Kích thước hạt nhân kết hợp	5x5
Số hạt nhân chập	6
Mạng đặc trưng đầu ra	28x28

#### 4.2.2.2. Lớp S2-Pooling

Sau tích chập đầu tiên, có một hoạt động gộp, sử dụng lõi  $2 \times 2$  để gộp chung, do đó thu được 6 ma trận đặc trưng đầu ra  $14 \times 14$ . Lớp gộp S2 là tổng các pixel trong vùng  $2 \times 2$  trong C1, nhân nó với hệ số trọng số và thêm phần bù, sau đó ánh xạ lại kết quả. Có  $5 \times 14 \times 14 \times 6 = 5880$  kết nối cùng một lúc.

*Bảng 4.2. Bảng tham số lớp S2-pooling*

Tên	Tham số
Mạng đặc trưng đầu vào	28x28
Khu vực lấy mẫu	2x2
Số lượng mẫu	6
Mạng đặc trưng đầu ra	14x14

#### 4.2.2.3. Lớp C3-Convolutional

*Bảng 4.3. Bảng tham số lớp C3-Convolutional*

Tên	Tham số
Mạng đặc trưng đầu vào	$14 \times 14$
Kích thước hạt nhân kết hợp	$5 \times 5$
Số hạt nhân chập	16
Mạng đặc trưng đầu ra	$10 \times 10$

Mỗi mạng đặc trưng đầu vào trong lớp chập C3 được kết nối với tất cả 6 hoặc một số mạng đặc trưng trong S2, chỉ ra rằng mạng đặc trưng của lớp này là sự kết hợp

khác nhau của các mạng đặc trưng được trích xuất từ lớp trước. 6 mạng đặc trưng đầu tiên của C3 lấy 3 tập hợp mạng đặc trưng liền kề trong S2 làm đầu vào. 6 mạng đặc trưng tiếp theo lấy 4 tập hợp mạng đặc trưng liền kề trong S2 làm đầu vào. 3 tiếp theo lấy 4 tập hợp mạng đặc trưng không liền kề làm đầu vào. Cái cuối cùng lấy tất cả các mạng đặc trưng trong S2 làm đầu vào. Hoạt động này có hai ưu điểm. Một là nó có thể giảm các tham số và số lượng huấn luyện. Thứ hai là phương pháp kết hợp và kết nối không đối xứng này có lợi để trích xuất nhiều tính năng kết hợp.

#### 4.2.2.4. Lớp S4-Pooling

S4 là lớp gộp, kích thước cửa sổ vẫn là  $2 \times 2$ , tổng cộng 16 mạng đặc trưng đầu vào và 16 mạng  $10 \times 10$  trong lớp C3 được gộp với  $2 \times 2$  làm đơn vị để có được 16 mạng đặc trưng  $5 \times 5$ . Có  $5 \times 5 \times 5 \times 16 = 2000$  kết nối. Phương thức kết nối tương tự như lớp S2.

*Bảng 4.4. Bảng tham số lớp S4-pooling*

Tên	Tham số
Mạng đặc trưng đầu vào	10x10
Khu vực lấy mẫu	2x2
Số lượng mẫu	16
Mạng đặc trưng đầu ra	5x5

#### 4.2.2.5. Lớp C5-Convolutional

Lớp C5 là một lớp chập. Do kích thước của 16 biểu đồ trong lớp S4 là  $5 \times 5$ , có cùng kích thước với hạt nhân chập, nên kích thước của mạng được hình thành sau khi tích chập là  $1 \times 1$ . 120 kết quả tích chập được hình thành ở đây. Mỗi mạng được kết nối với 16 hình ảnh ở cấp trên. Vì vậy, có  $(5 \times 5 \times 16 + 1) \times 120 = 48120$  tham số, và cũng có 48020 kết nối.

*Bảng 4.5. Bảng tham số lớp C5-Convolutional*

Tên	Tham số
Mạng đặc trưng đầu vào	5x5
Kích thước hạt nhân kết hợp	5x5
Số hạt nhân chập	120
Mạng đặc trưng đầu ra	1x1

#### 4.2.2.6. F6 – Fully connected layer

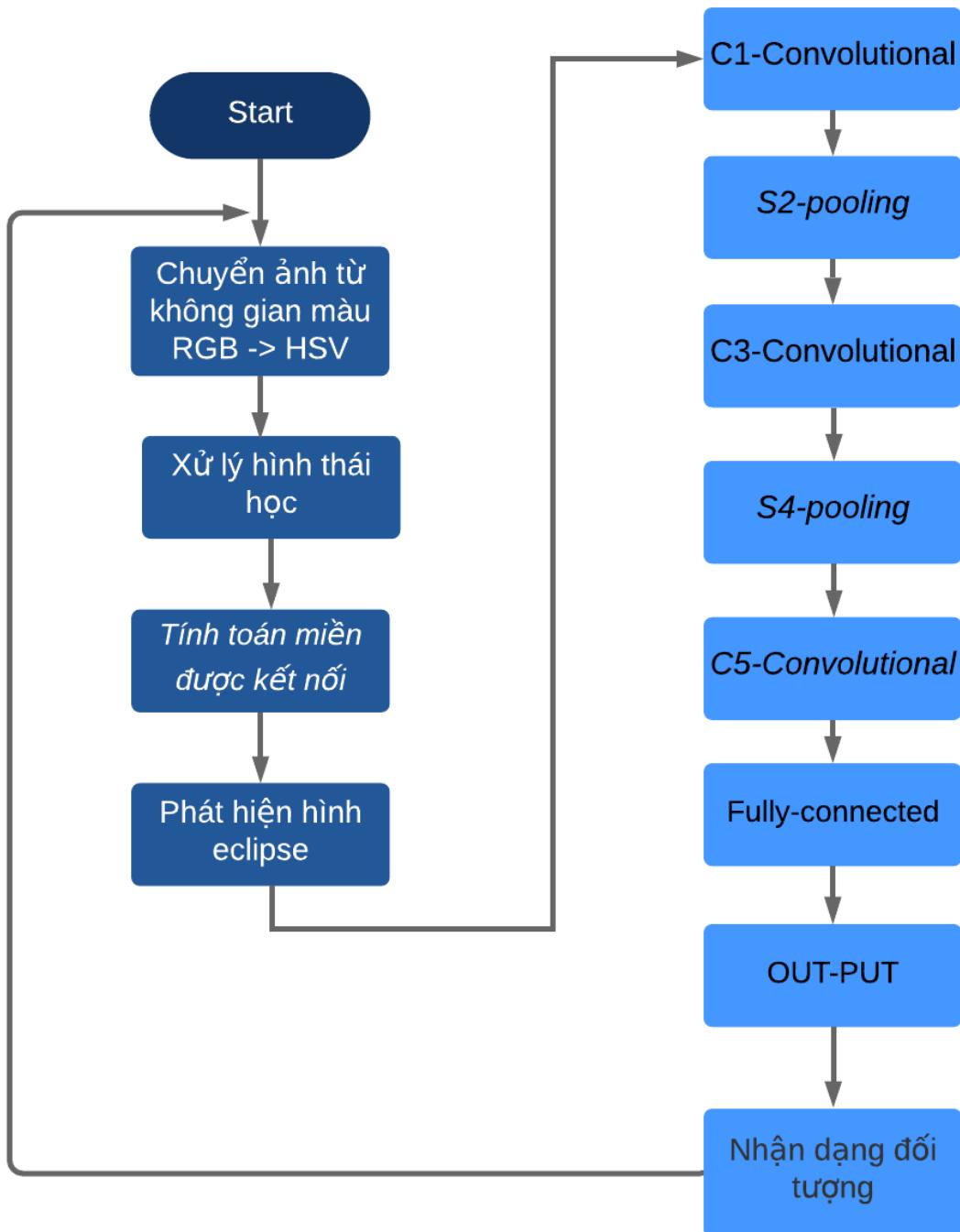
Đầu vào của lớp được kết nối đầy đủ là đầu ra vecto 120 chiều của lớp chập C5. Trong lớp được kết nối đầy đủ, tính toán cuối cùng là tích giữa vecto đầu vào và vecto trọng số, cộng với phần bù và kết quả là đầu ra thông qua hàm sigmoid. Các thông số có thể huấn luyện là:

$$84 \times (120 + 1) = 10164.$$

#### 4.2.2.7. OUTPUT - Fully connected layer

Lớp OUTPUT cũng là lớp được kết nối đầy đủ, có tổng cộng 7 nút, tương ứng với 7 kết quả nhận dạng biển báo giao thông - tốc độ 10, tốc độ 30, tốc độ 80, rẽ trái, rẽ phải, dừng và các nút khác. Phương thức kết nối mạng của chức năng cơ sở xuyên tâm (RBF) được thông qua. Giả sử  $x$  là đầu vào của lớp trước và  $y$  là đầu ra của RBF, phương thức tính toán của đầu ra RBF là:

$$y_i = \sum_j (x_j - w_{ij})^2$$



Hình 4.5. Sơ đồ giải thuật xác định và nhận dạng biển báo giao thông

Dựa vào sơ đồ giải thuật trên, ta có thể nhận thấy chương trình này có hai phần cơ bản gồm phần nhận biển báo và phân loại biển báo. Về phần nhận dạng biển báo, hình ảnh khi nhận về từ camera sẽ được Raspberry xử lý. Đầu tiên, hình ảnh màu RGB sẽ được chuyển ảnh màu HSV, sau đó chương trình sẽ xác định và tách biển báo trong ảnh ra để thực hiện việc phân loại. Sau khi đã xác định được biển báo trong khung hình

và tách biển báo thì biển báo đó sẽ được đưa vào chương trình thực hiện nhận dạng loại biển báo, hình ảnh biển báo sẽ được đưa qua các bộ lọc để xử lý ảnh. Ở giai đoạn cuối cùng, tùy theo xác suất nhận được đối với sáu loại biển báo đã được huấn luyện trước, xác suất gần với loại biển báo nào thì hệ thống sẽ đưa ra kết quả là loại biển báo đó.

#### Độ chính xác của mô hình

Đối với nhiệm vụ nhận dạng biển báo giao thông, chúng ta đã huấn luyện mô hình trên bộ dữ liệu đã được xây dựng. Vì nhiệm vụ phân loại tương đối đơn giản, để tăng tốc độ tính toán, sau khi đọc các hình ảnh, chúng ta sẽ cắt chúng thành kích thước  $32 * 32$ .

Cài đặt tham số huấn luyện:

- Tỷ lệ học tập learning\_rate = 0: 001
- Kích thước hàng loạt batch\_size = 32
- Số lần lặp epochs = 50
- Lựa chọn chiến lược tối ưu Adam

Trong cùng một môi trường tính toán như trên, mô hình hội tụ nhanh chóng và thời gian thực hiện tính toán là ít hơn 3 phút. Hàm mất và đường cong thay đổi tỷ lệ chính xác của tập huấn luyện và tập xác minh trong quá trình huấn luyện được thể hiện trong hình dưới đây:



*Hình 4.6. Kết quả huấn luyện mạng CNN*

Kết quả cuối cùng là: giá trị tổn thất của tập huấn luyện là 0,0494, tỷ lệ chính xác của tập huấn luyện là 0,9832, giá trị tổn thất của tập chứng minh là 0,1068 và tỷ lệ chính xác của tập xác minh là 0,9765.

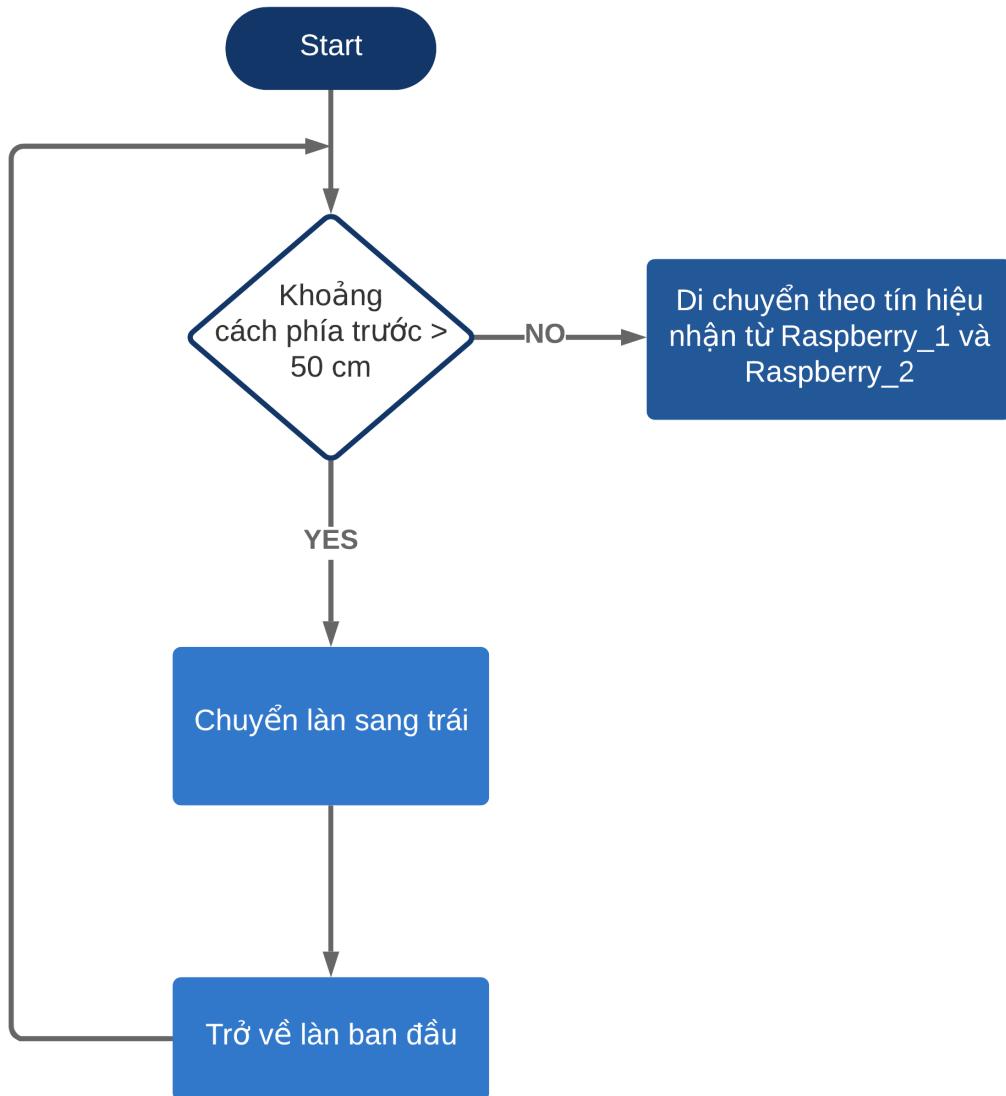
*Bảng 4.6. Tín hiệu kết nối giữa Raspberry\_2 và Tiva C*

E3	E4	E5	Tiva C
GPIO_8	GPIO_7	GPIO_1	Raspberry_2
0	0	0	Speed_10
0	0	1	Speed_30
0	1	0	Speed_80
0	1	1	Turn_Left
1	0	0	Turn_Right
1	0	1	Stop
1	1	0	Do_nothing
1	1	1	None

### 4.3. Tránh vật cản

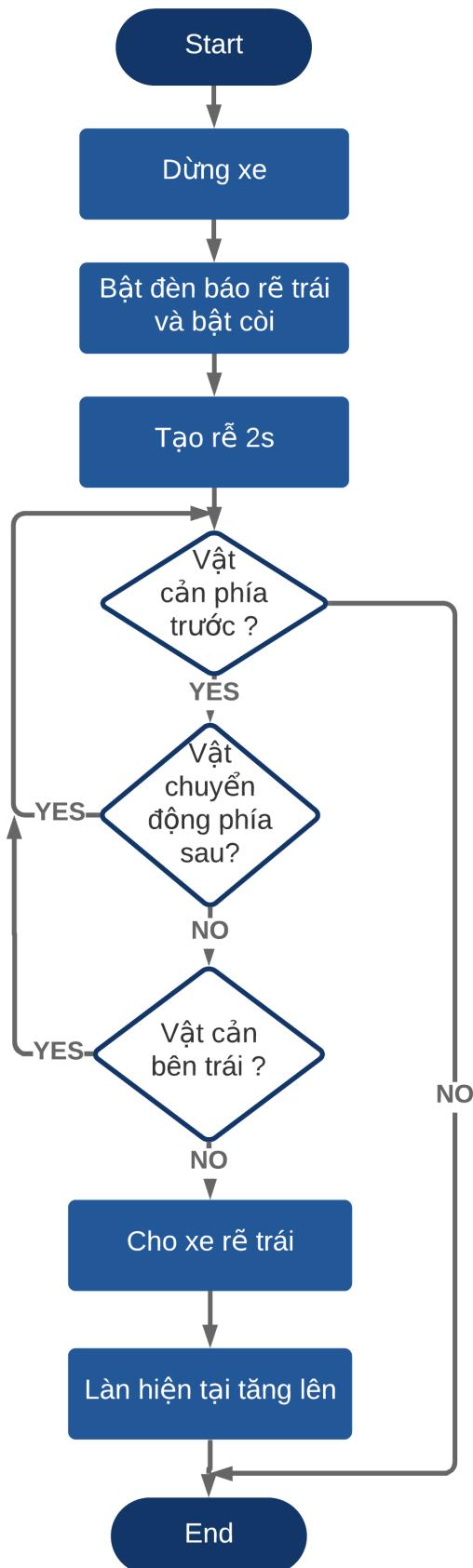
Khi di chuyển trên đường, đôi khi ta sẽ gặp các chướng ngại ở trên làn đường mà chúng ta đang di chuyển, điều đó sẽ cần trở quá trình di chuyển của xe chúng ta. Với hệ thống này thì xe hoàn toàn có thể tự dừng lại khi phát hiện vật cản phía trước, bật đèn và còi để báo hiệu cho các phương tiện khác khi đang cùng di chuyển trên

đường. Khi điều kiện xung quanh đã đủ an toàn thì hệ thống sẽ cho phép xe tự động chuyển sang làn bên trái để tiếp tục di chuyển. Các điều kiện được thiết lập cho hệ thống bao gồm vật cản phía trước vẫn còn, chưa di chuyển nhờ cảm biến lazer mà ta có thể phát hiện được điều đó, bên trái xe không có vật cản hay xe đang di chuyển nhờ vào hai cảm biến hồng ngoại được đặt bên cạnh trái của xe và phía sau không còn xe đang di chuyển đến phía trước nhờ vào cảm biến hồng ngoại phát hiện vật chuyển động được đặt ở phía sau xe. Trong trường hợp chuyển làn qua bên trái thì xe sẽ bật còi và đèn báo rẽ trái để các phương tiện đang lưu thông trên đường được biết. Nếu trong thời gian chờ để được sang làn bên trái mà xe phát hiện vật cản phía trước không còn nữa thì xe sẽ tiếp tục di chuyển ở làn hiện tại và không chuyển sang làn bên trái. Khi đã vượt chướng ngại vật và đang di chuyển ở làn bên trái, sau một khoảng thời gian nhất định thì xe sẽ bật đèn xin được rẽ sang làn lúc đầu trước đó, nếu phát hiện không có vật cản thì xe sẽ rẽ phải về làn lúc đầu mà xe đang đi.



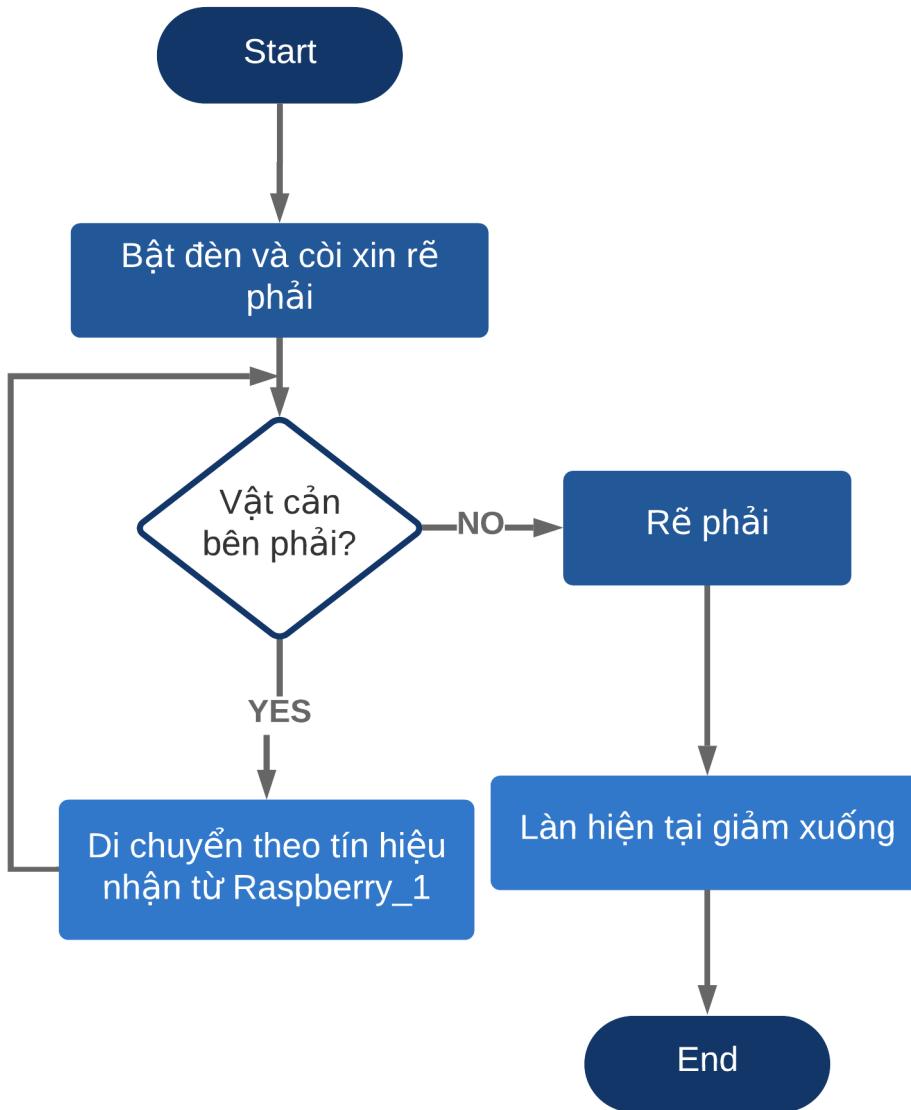
*Hình 4.7 Sơ đồ giải thuật tổng quát cho chức năng tránh vật cản*

Khi cảm biến lazer gửi dữ liệu về cho Raspberry\_1 mà khoảng cách so với vật phía trước nhỏ hơn 50 cm thì Raspberry\_1 sẽ gửi tín hiệu về cho Tiva C, khi đó chương trình điều khiển xe sẽ gọi chương trình con thực hiện lệnh rẽ trái để tránh vật cản phía trước. Sau khi đã chuyển sang làn bên trái để di chuyển thì hệ thống sẽ thực hiện chương trình để đưa xe trở về làn ban đầu.



Hình 4.8. Lưu đồ giải thuật chương trình rẽ trái

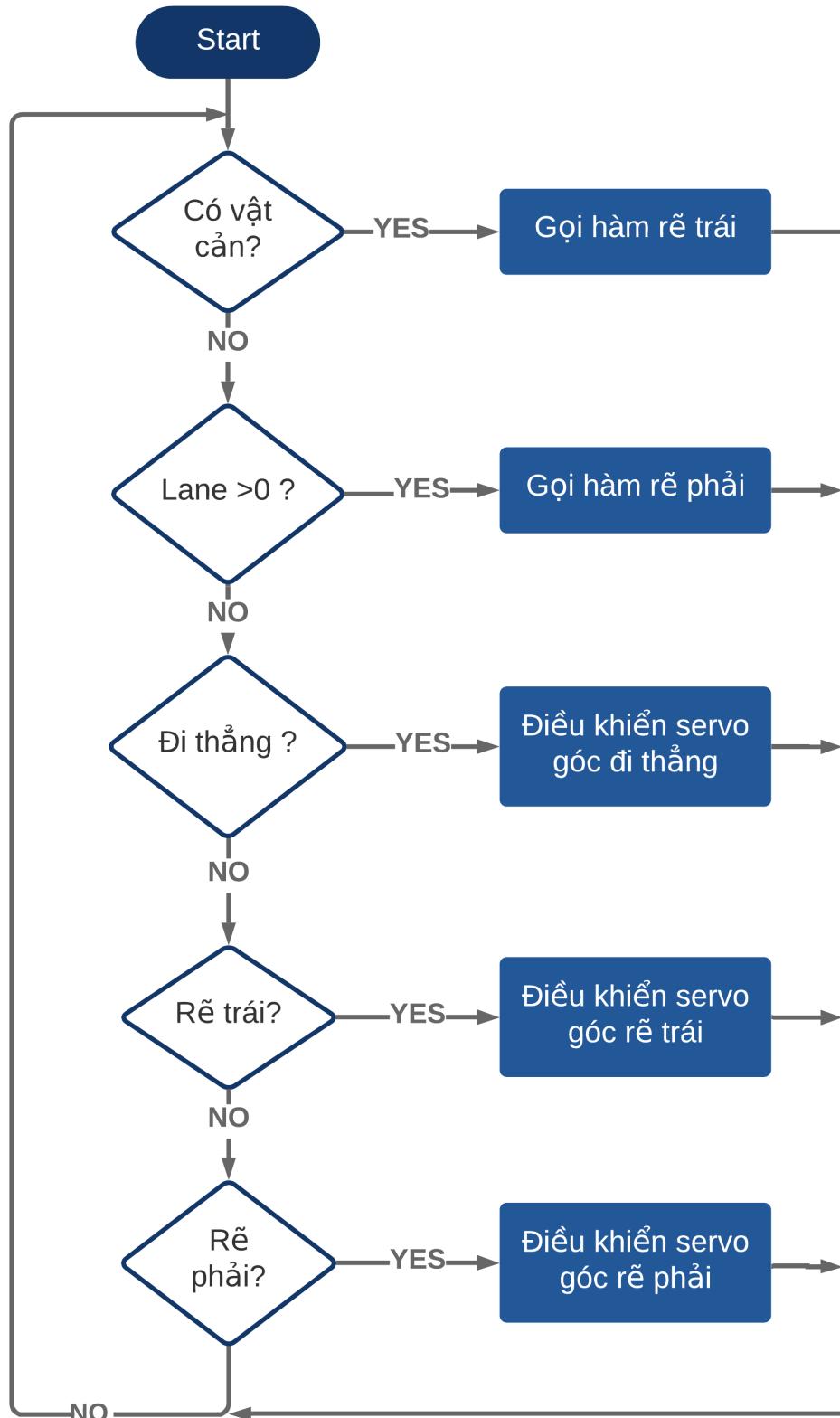
Khi chương trình chính gọi chương trình con để hỗ trợ xe chuyển làn thì ta sẽ có lưu đồ giải thuật cho chương trình con này như trên. Đầu tiên, xe dừng lại bật đèn báo rẽ trái và còi cho các phương tiện cùng di chuyển trên đường được biết trong thời gian 2 giây. Sau đó chương trình này sẽ kiểm tra còn vật cản phía trước hay không bằng tín hiệu nhận được từ cảm biến khoảng cách lazer, nếu không còn vật cản thì xe tiếp tục di chuyển như bình thường, nếu còn vật cản thì xe sẽ kiểm tra các điều kiện về môi trường xung quanh để đảm bảo an toàn cho việc chuyển làn. Các điều kiện được kiểm tra bao gồm khoảng cách bên trái xe và góc trái phía trước xe, bằng hai cảm biến khoảng cách hồng ngoại, kiểm tra vật chuyển động phía sau bằng cảm biến hồng ngoại vật chuyển động, nếu kết quả nhận về đều đạt yêu cầu thì xe sẽ thực hiện rẽ trái để chuyển làn. Trong khi chuyển làn, nhờ vào cảm vào cảm biến tốc độ encoder để xác định được quãng đường đã di chuyển và đưa xe về trạng thái bình thường đúng lúc. Sau khi chuyển làn xong thì xe sẽ tắt hết các đèn báo rẽ và còi sau đó xe di chuyển bình thường. Sau khi đã rẽ trái xong thì xe đang ở làn bên trái so với làn trước đó xe đã di chuyển, biến nhớ làn của xe sẽ tăng lên một đơn vị.



*Hình 4.9. Lưu đồ giải thuật chương trình rẽ phải*

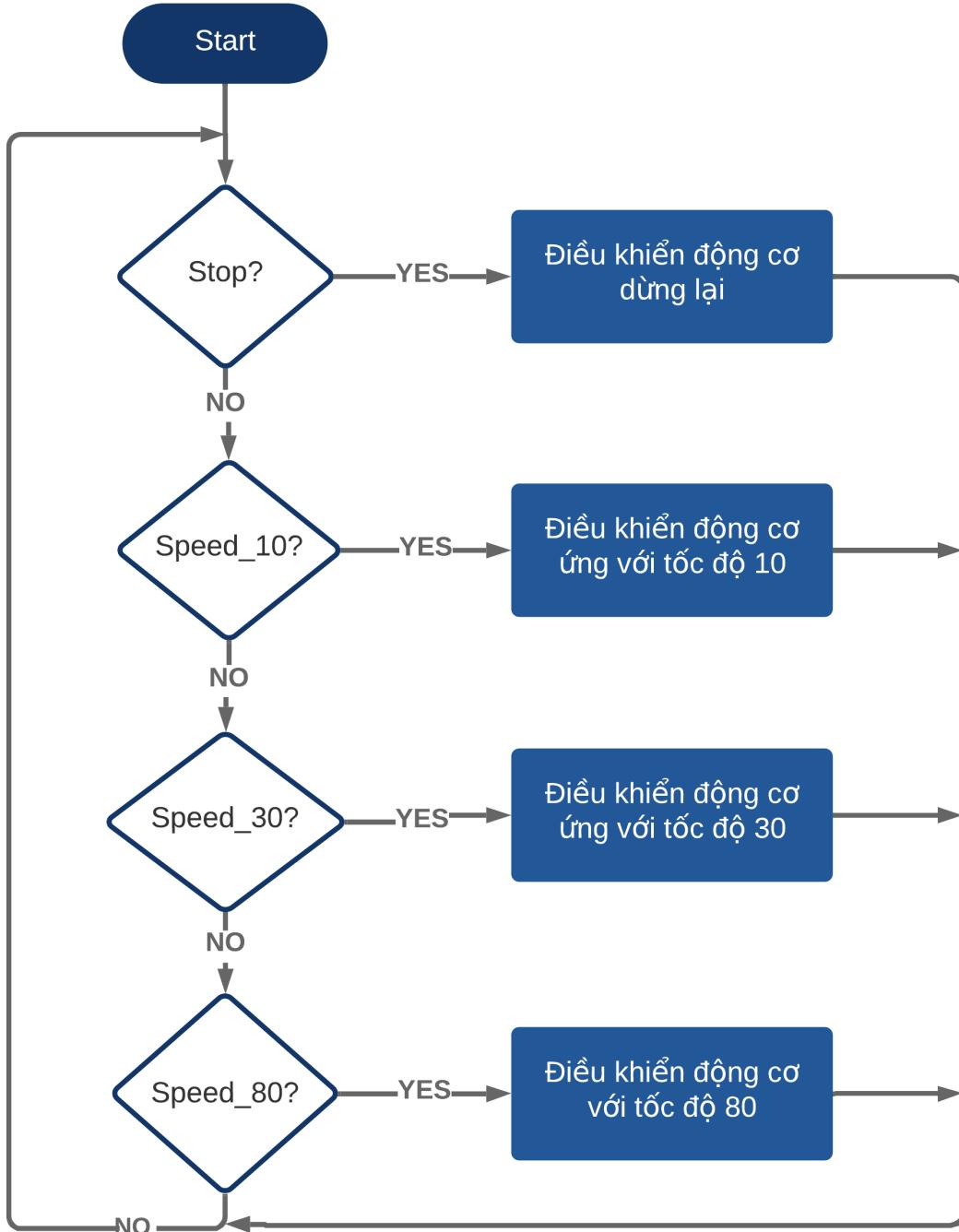
Sau khi thực hiện rẽ trái và biến nhó vị trí làn đã tăng lên so với làn trước đó. Xe sẽ tiếp tục di chuyển trong 10 giây và thực hiện chuyển làn về làn ban đầu. Trước khi chuyển làn thì xe sẽ bật đèn xin rẽ phải và còi cho các phương tiện được biết. Khi kiểm tra các cảm biến hồng ngoại bên phải (cảm biến bên phải xe và cảm biến góc phải phía trước xe), nếu tín hiệu trả về từ các cảm biến này đều đạt yêu cầu về khoảng cách thì xe tiến hành rẽ phải để đưa xe về làn như ban đầu. Trong khi chuyển làn, nhờ vào cản vào cảm biến tốc độ encoder để xác định được quãng đường đã di chuyển và đưa xe về trạng thái bình thường đúng lúc. Sau đó xe tắt hết đèn xin rẽ và còi, biến ghi nhớ vị trí làn sẽ giảm đi một đơn vị. Sau khi thực hiện rẽ phải thì xe di chuyển bình thường.

## 4.4. Các task được thực hiện trên Tiva C



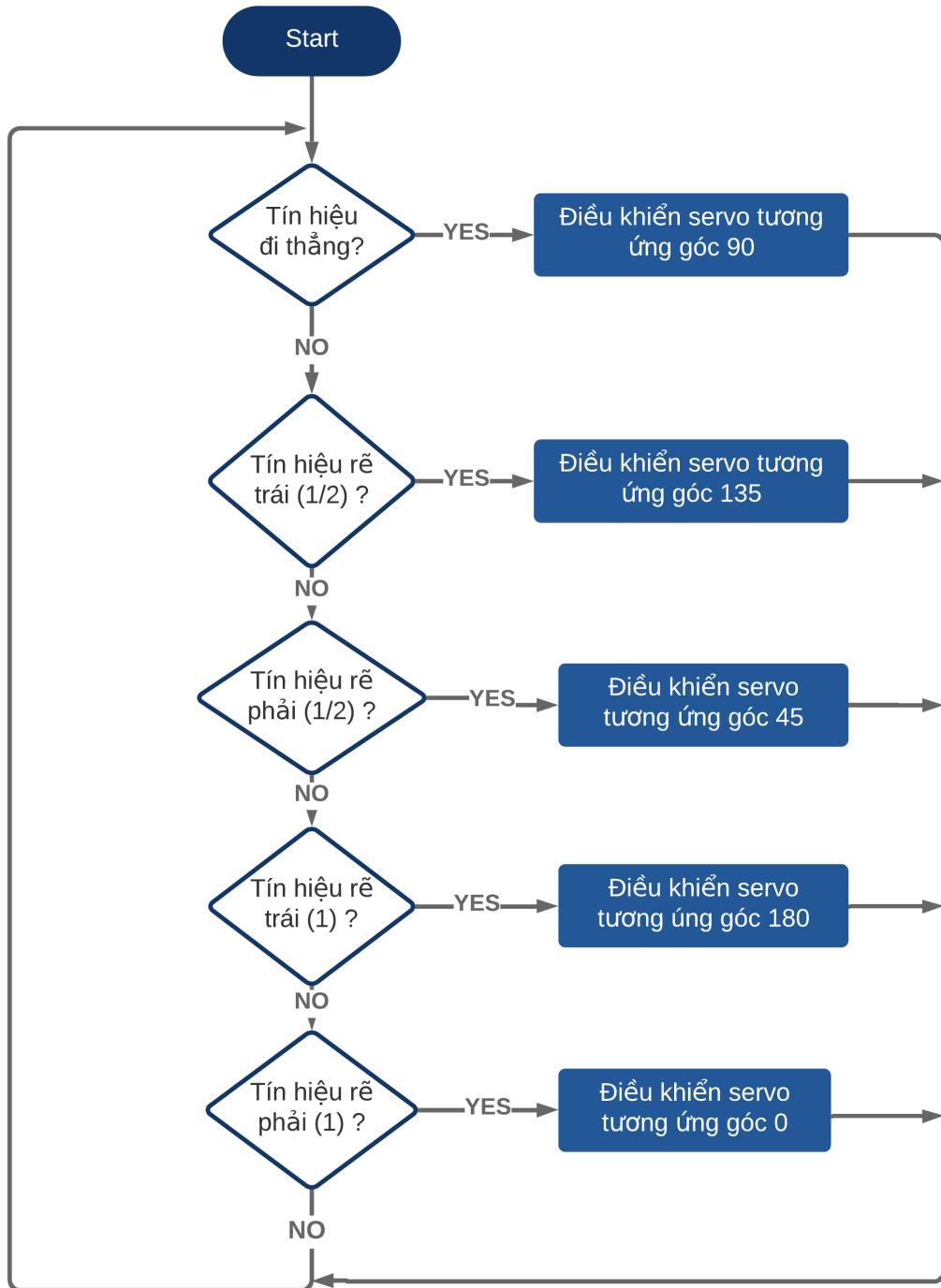
Hình 4.10. Lưu đồ giải thuật task 1 – Nhận tín hiệu từ Raspberry\_1

Chương trình này sẽ ưu tiên kiểm tra vật cản phía trước, nếu có thì gọi chương trình rẽ trái để chuyển làn, nếu không đi kiểm tra điều kiện xe có di chuyển đúng làn ban đầu hay không, nếu không thì thực hiện chuyển làn, để xe đi đúng làn ban đầu, nếu xe đã đi đúng thì tiếp tục điều khiển xe theo tín hiệu nhận được từ Raspberry\_1 (rẽ trái, rẽ phải, đi thẳng).



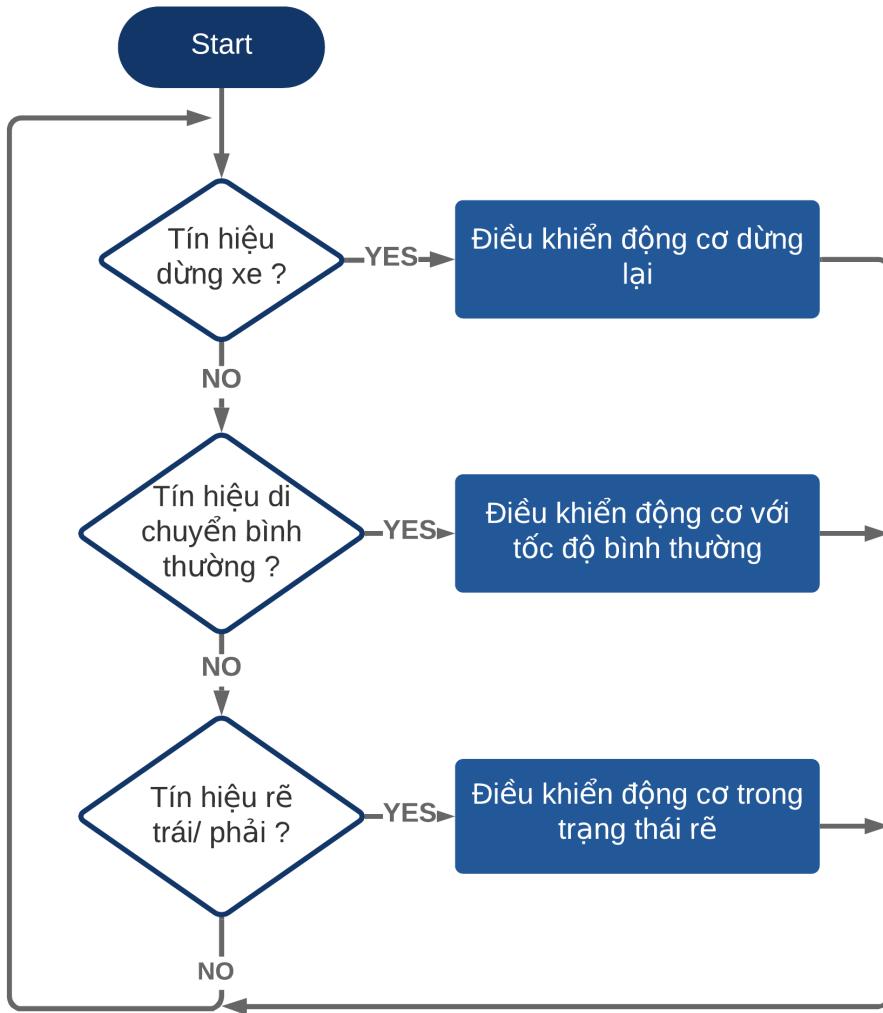
Hình 4.11. Lưu đồ giải thuật task 2 – Nhận tín hiệu từ Raspberry\_2

Với lưu đồ giải thuật như trên ta có thể thấy task nhận tín hiệu từ Raspberry\_2 sẽ dựa và tín hiệu nhận được và các hành động cần thiết. Nếu tín hiệu nhận được là stop thì chương trình sẽ dừng lại. Tương tự như vậy cho các loại biển báo giới hạn tốc độ, Tiva C sẽ tự điều chỉnh xung PWM điều khiển động cơ DC tỉ lệ với tín hiệu về giới hạn tốc độ mà nó nhận được từ Raspberry\_2. Từ đó, xe sẽ di chuyển theo tốc độ mong muốn.



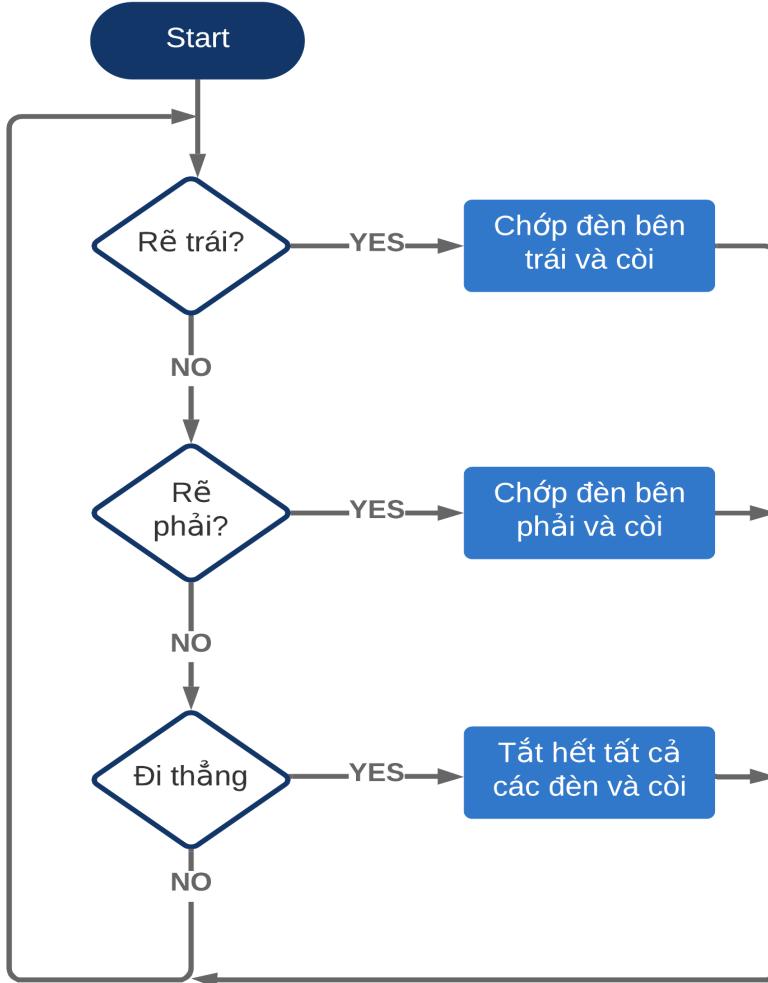
Hình 4.12. Lưu đồ giải thuật task 3 – Điều khiển servo RC

Task điều khiển Servo RC trên Tiva C sẽ xuất ra xung PWM điều khiển góc qua servo, thông qua hệ thống truyền động sẽ làm thay đổi góc của hai bánh xe trước từ đó giúp xe đi đúng làn đường hay thực hiện được chức năng chuyển làn như ta mong muốn. Khi nhận được tín hiệu từ các Raspberry và cảm biến, chương trình chính sẽ gán biến trạng tùy theo tín hiệu nhận được. Đối với tín hiệu đi thẳng thì chương trình sẽ xuất xung điều khiển góc servo là  $90^\circ$  từ làm hai bánh trước ở vị trí hướng thẳng giúp xe đi thẳng. Tín hiệu rẽ trái (1/2) và tín hiệu rẽ phải (1/2) là tín hiệu nhận được khi xe đang di chuyển mà bị lệch sang phải hoặc lệch sang trái. Khi đó, chương trình sẽ xuất ra xung PWM điều khiển servo quay góc  $135^\circ$  hoặc  $45^\circ$  tương ứng để đưa xe di chuyển về đúng làn. Còn tín hiệu rẽ phải (1) hay rẽ trái (1) được sử dụng khi xe được chuyển làn, khi đó góc qua servo được điều khiển ở vị trí cực đại hoặc cực tiểu để điều khiển xe chuyển làn một cách nhanh chóng, tương ứng với góc  $180^\circ$  và  $0^\circ$  ở servo.



Hình 4.13. Lưu đồ giải thuật task 4 – Điều khiển động cơ DC

Tương tự như task điều khiển servo RC, điều khiển động cơ DC cũng nhận tín hiệu từ Raspberry và cảm biến để truyền xung PWM cho động cơ DC. Việc điều khiển này cũng phụ thuộc vào biến trạng thái, biến này sẽ được cài đặt giá trị phụ thuộc vào dữ liệu các cảm biến và Raspberry trả về. Biến này sẽ có ba trạng thái cơ bản sau, đầu tiên là trạng thái dừng xe, trong trường hợp có vật cản phía trước hay gặp biến báo dừng thì động cơ DC sẽ được điều khiển dừng lại. Trạng thái tốc độ bị giới hạn thì phụ thuộc vào loại biến báo (10km/h, 30km/h, 80km/h) thì ta có các tỉ lệ về tốc độ để điều khiển cho động cơ DC tương ứng. Trạng thái cuối cùng là trạng thái chuyển làn, ở trạng thái này thì động cơ DC được điều khiển với xung PWM khá lớn, do xe chuyển làn góc của lớn dẫn đến lực ma sát của bánh xe lớn nên cần lực mạnh để chuyển động được xe.

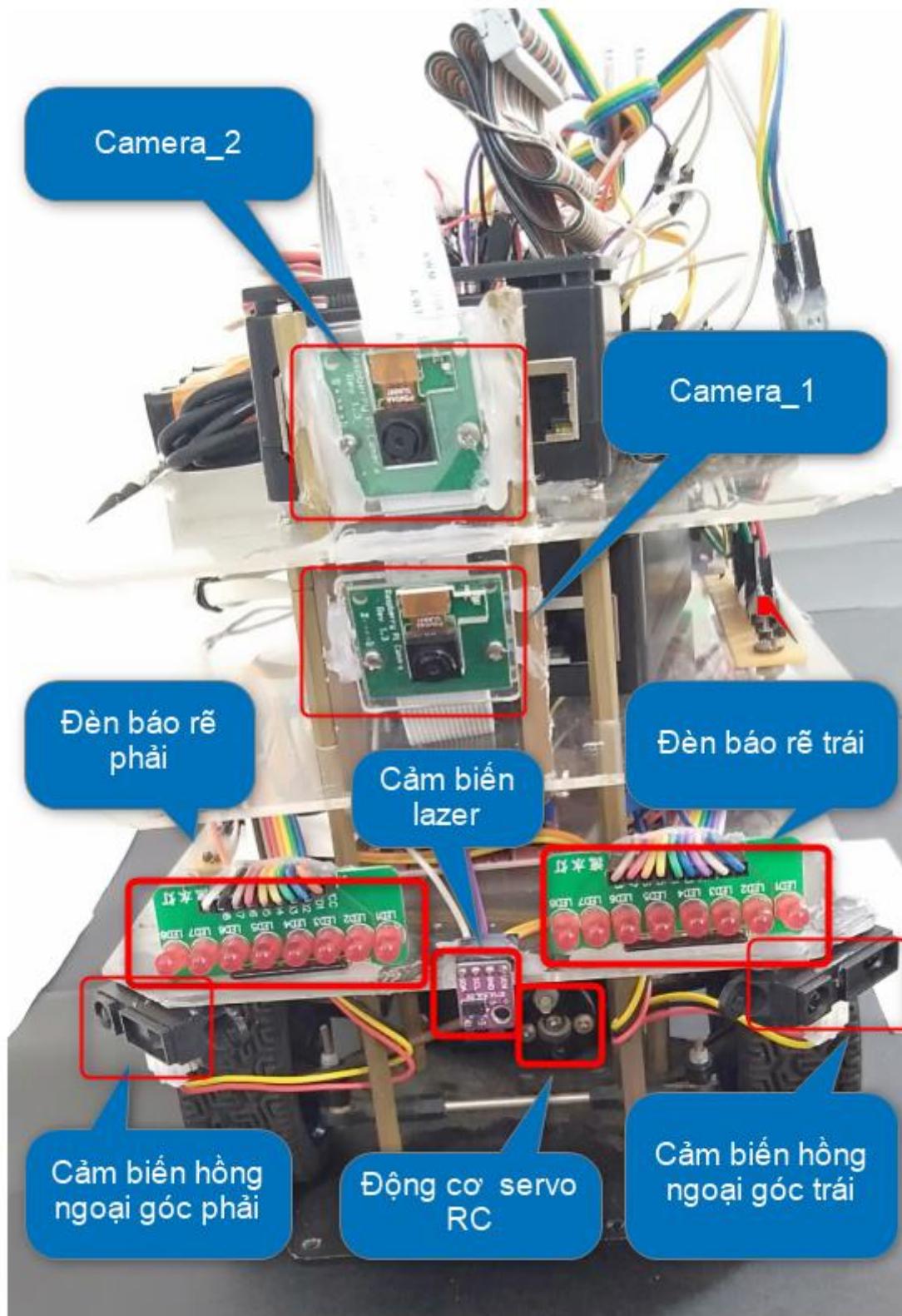


*Hình 4.14. Lưu đồ giải thuật task 5 – Chương trình cảnh báo*

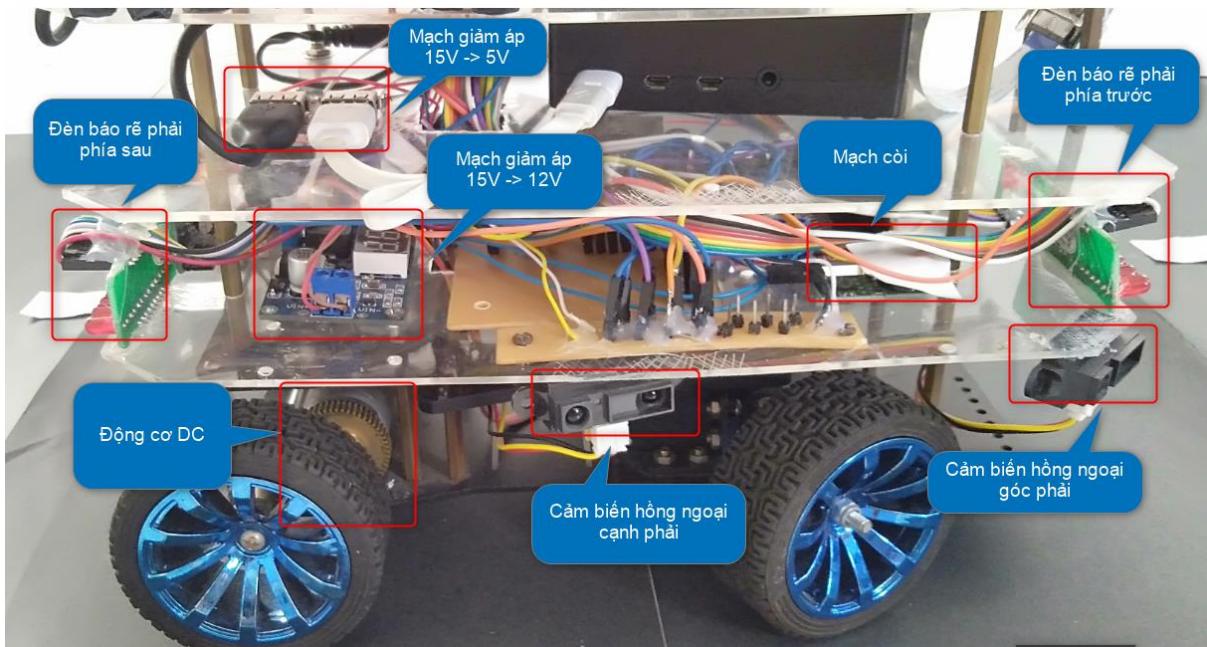
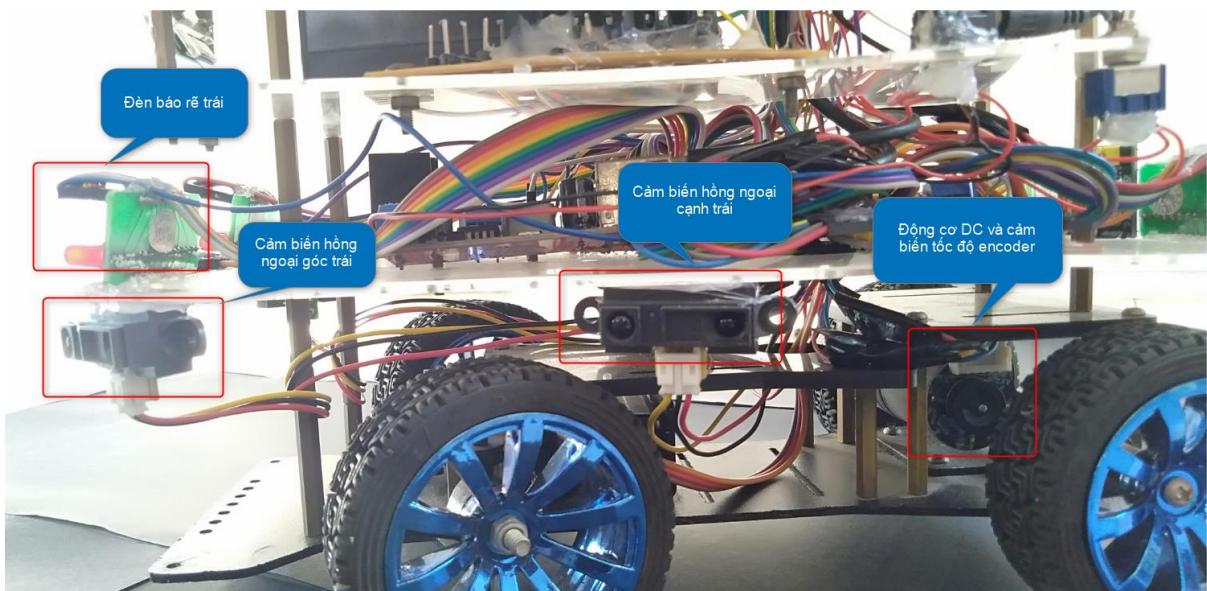
Trong chương trình cảnh báo thì đèn LED và còi là hai thành phần chính sẽ thực thi cho chức năng cảnh báo. Trong khi chương trình thực hiện việc chuyển làn sang trái hay đợi để được rẽ trái thì còi báo sẽ kêu và cụm đèn LED trái (gồm 8 LED đỏ) sẽ chạy hiệu ứng từ phải sang trái cả cụm trước và cụm sau. Cũng tương tự đối với chuyển làn sang phải hay đợi để chuyển làn sang phải thì còi kêu và cụm LED bên phải cũng sẽ chạy hiệu ứng từ trái sang phải cả cụm LED phía trước và phía sau. Trong trường hợp đặt biệt, khi gấp biển báo stop thì xe sẽ dừng lại đồng thời bật tín hiệu báo như tín hiệu báo cả rẽ trái và rẽ phải đã nêu ở trên.

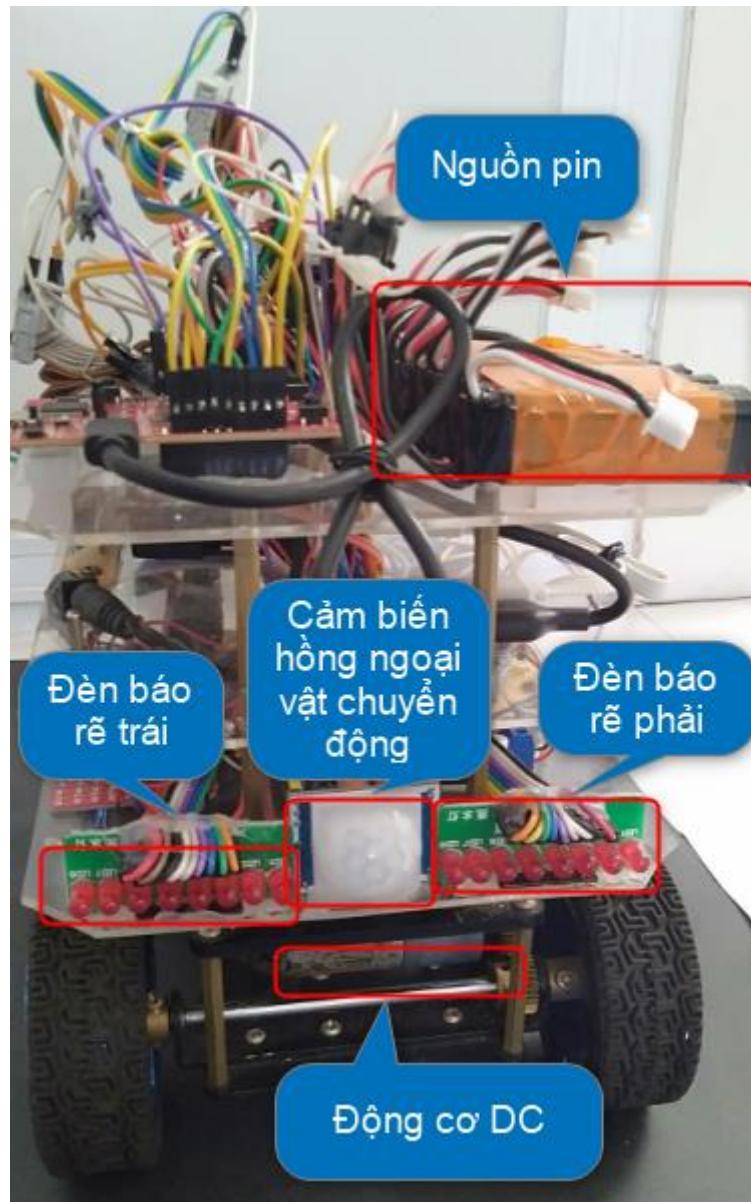
## 5. KẾT QUẢ THỰC HIỆN

### 5.1 Tổng quan về sản phẩm thực hiện

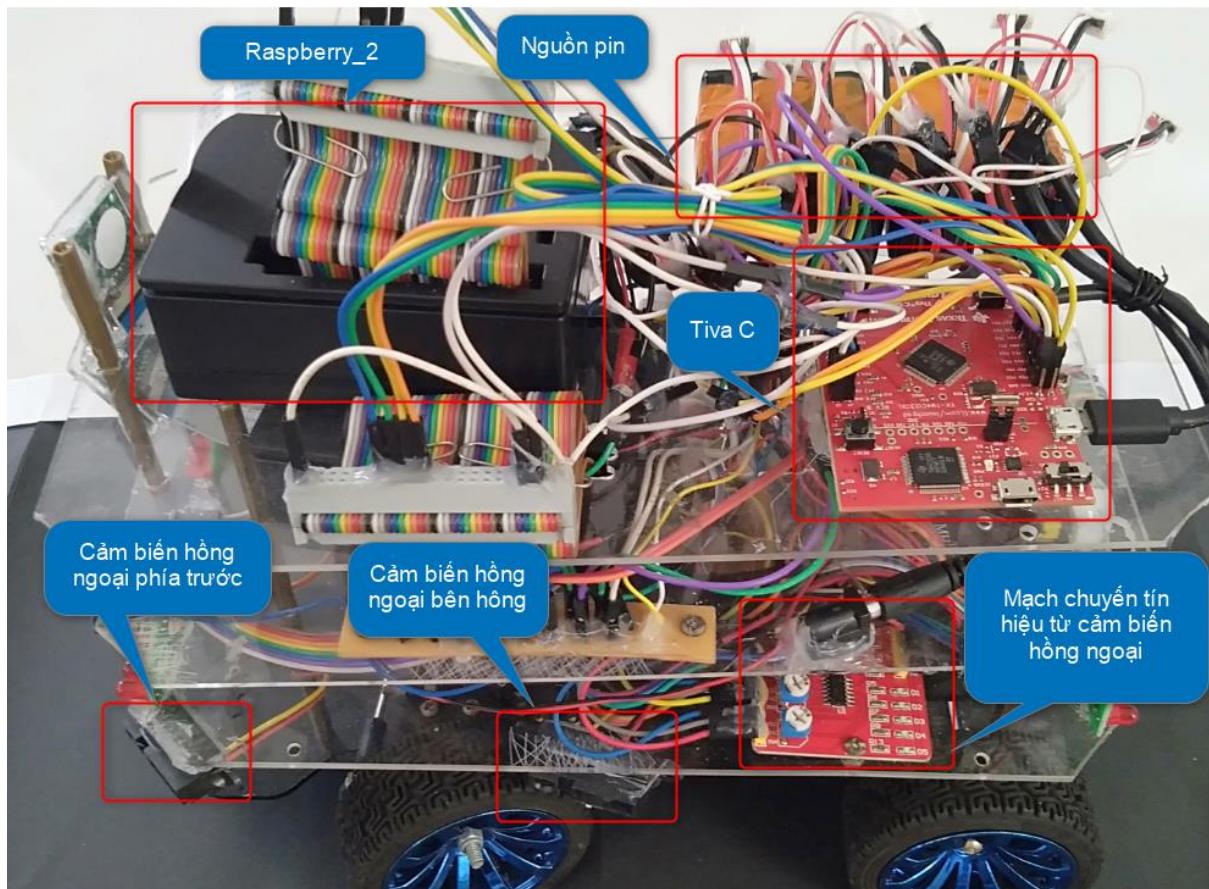


Hình 5.1. Phía trước xe

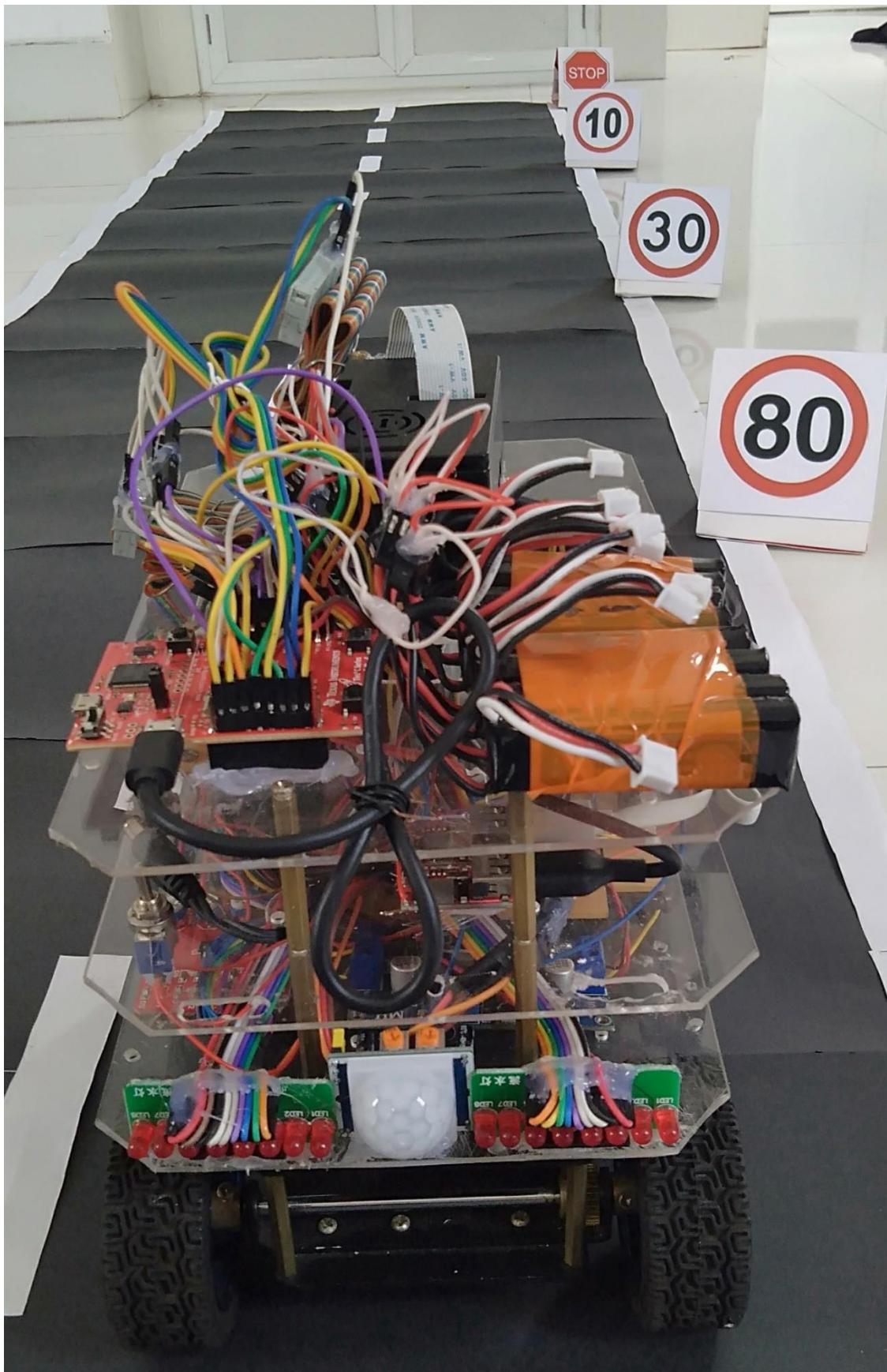
*Hình 5.2. Bên phải xe**Hình 5.3. Bên trái xe*



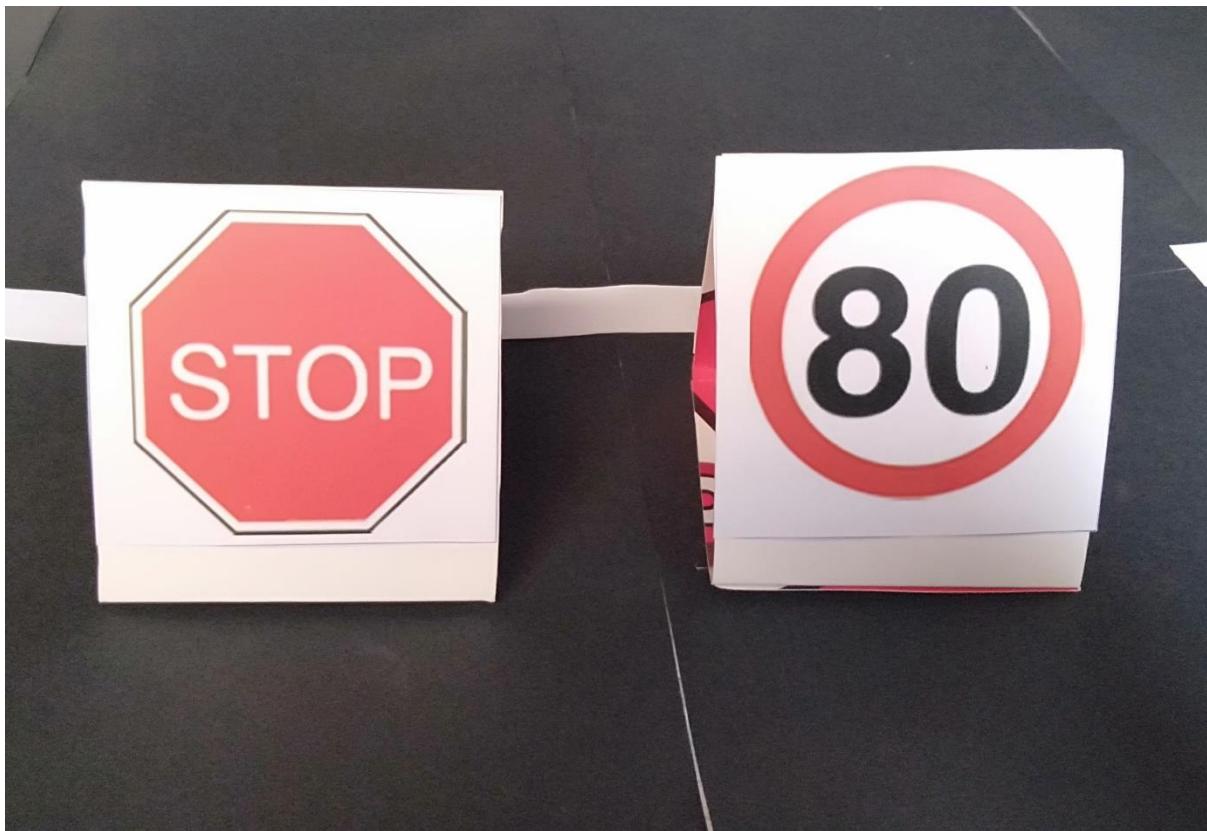
Hình 5.4. Phía sau xe



Hình 5.5. Phía trên xe



Hình 5.6. Đoạn đường thử nghiệm



Hình 5.7. Các biển báo thử nghiệm



Hình 5.8. Các biển báo thử nghiệm



Hình 5.9. Các biển báo thử nghiệm

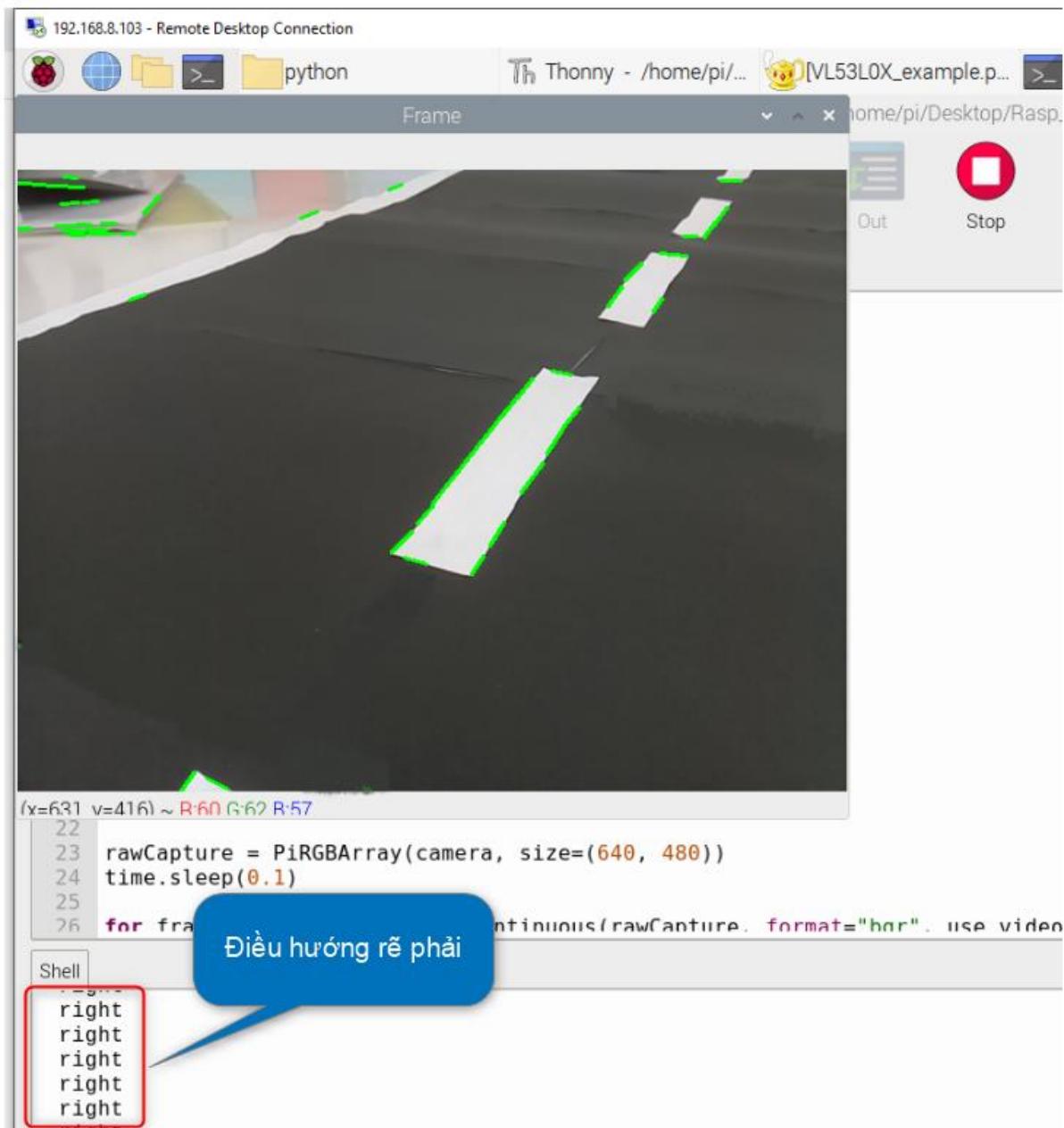
## 5.2 Chức năng nhận dạng làn đường



Hình 5.10 Kết quả nhận dạng làn đường – điều hướng đi thẳng

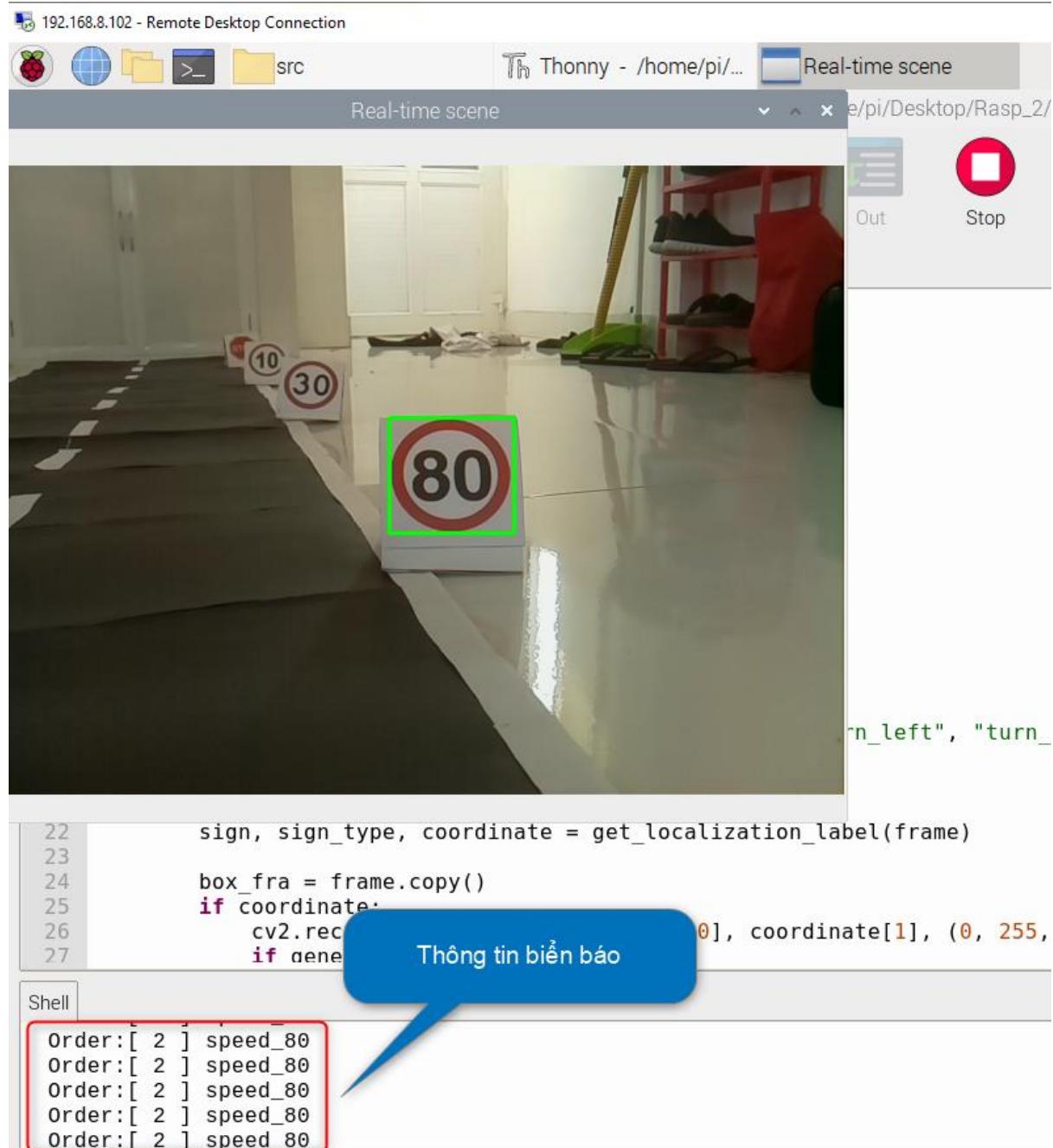


Hình 5.11 Kết quả nhận dạng làn đường điều hướng rẽ trái

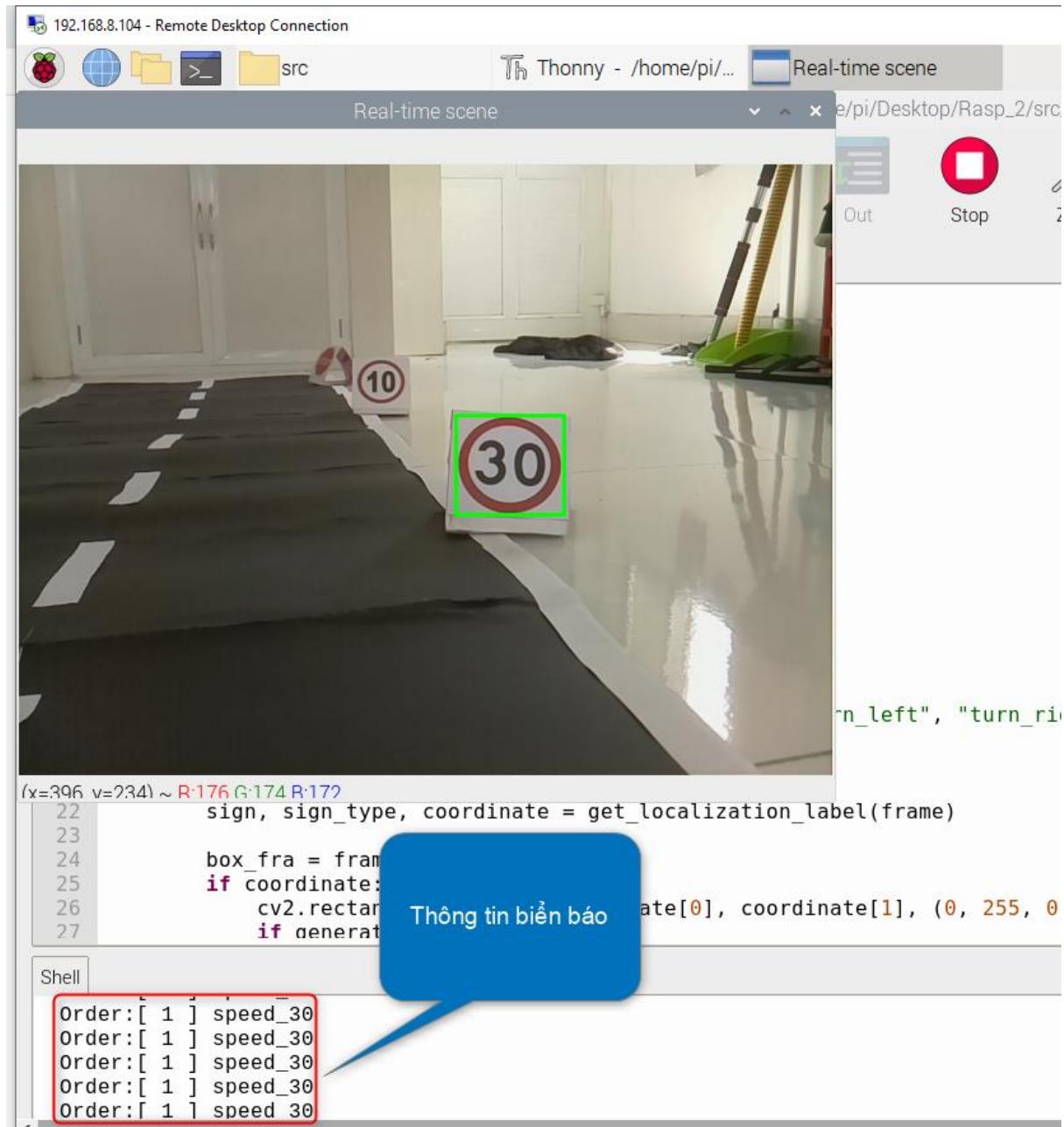


Hình 5.12 Kết quả nhận dạng làn đường – điều hướng rẽ phải

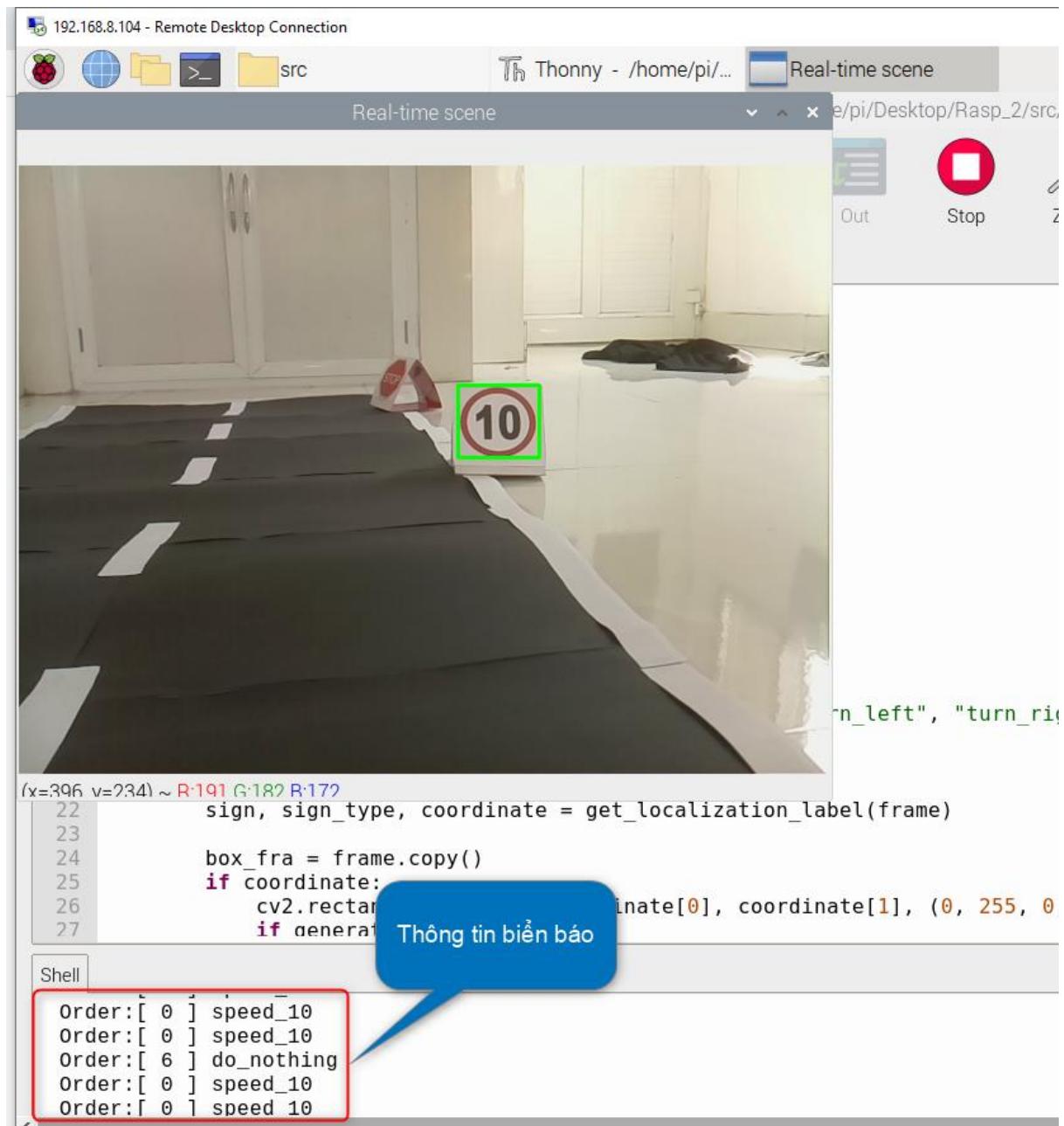
### **5.3 Chức năng nhận dạng biển báo giao thông**



Hình 5.13 Kết quả nhận dạng biển báo giới hạn tốc độ 80 km/h

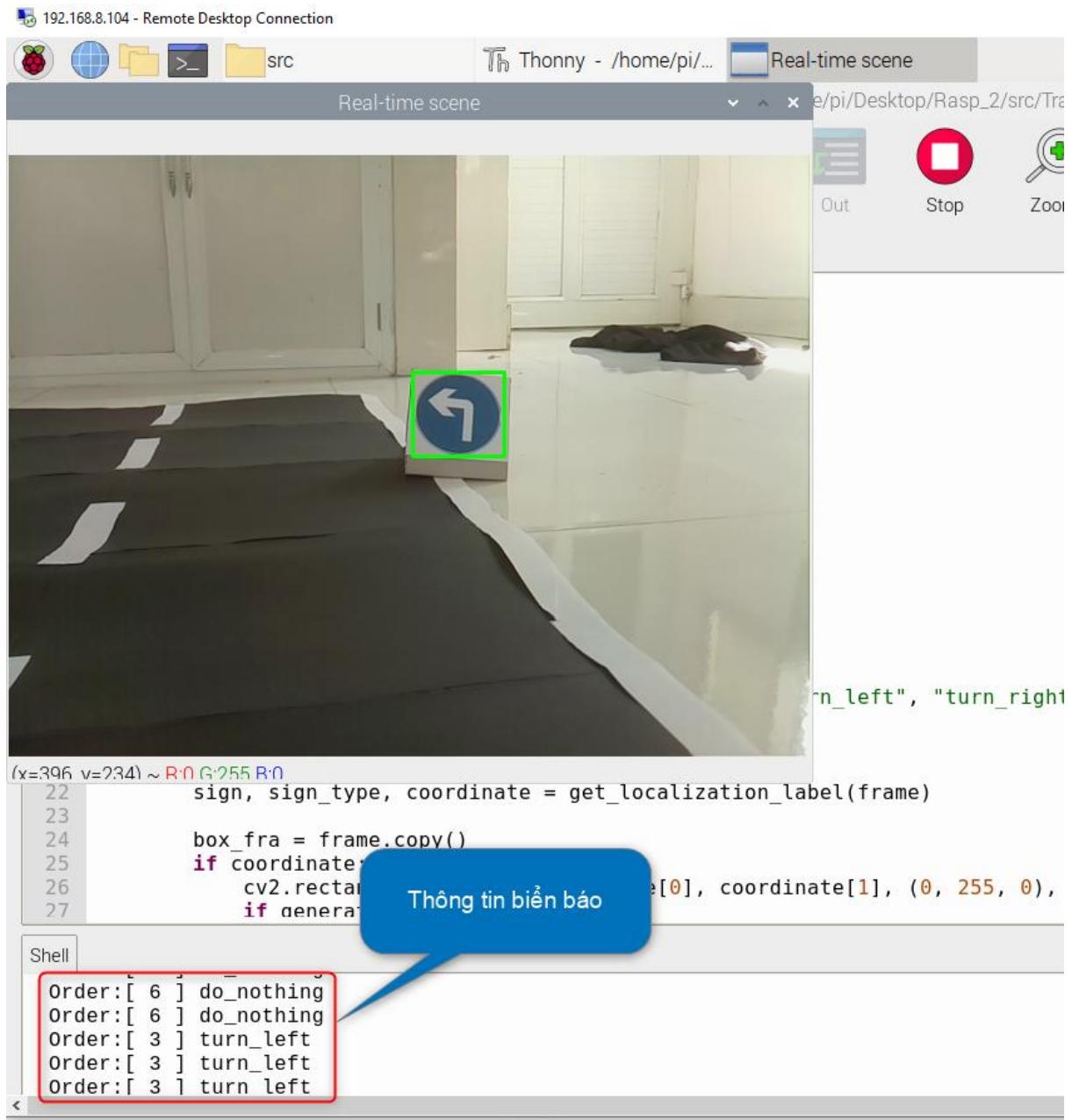


Hình 5.14 Kết quả nhận dạng biển báo giới hạn tốc độ 30km/h

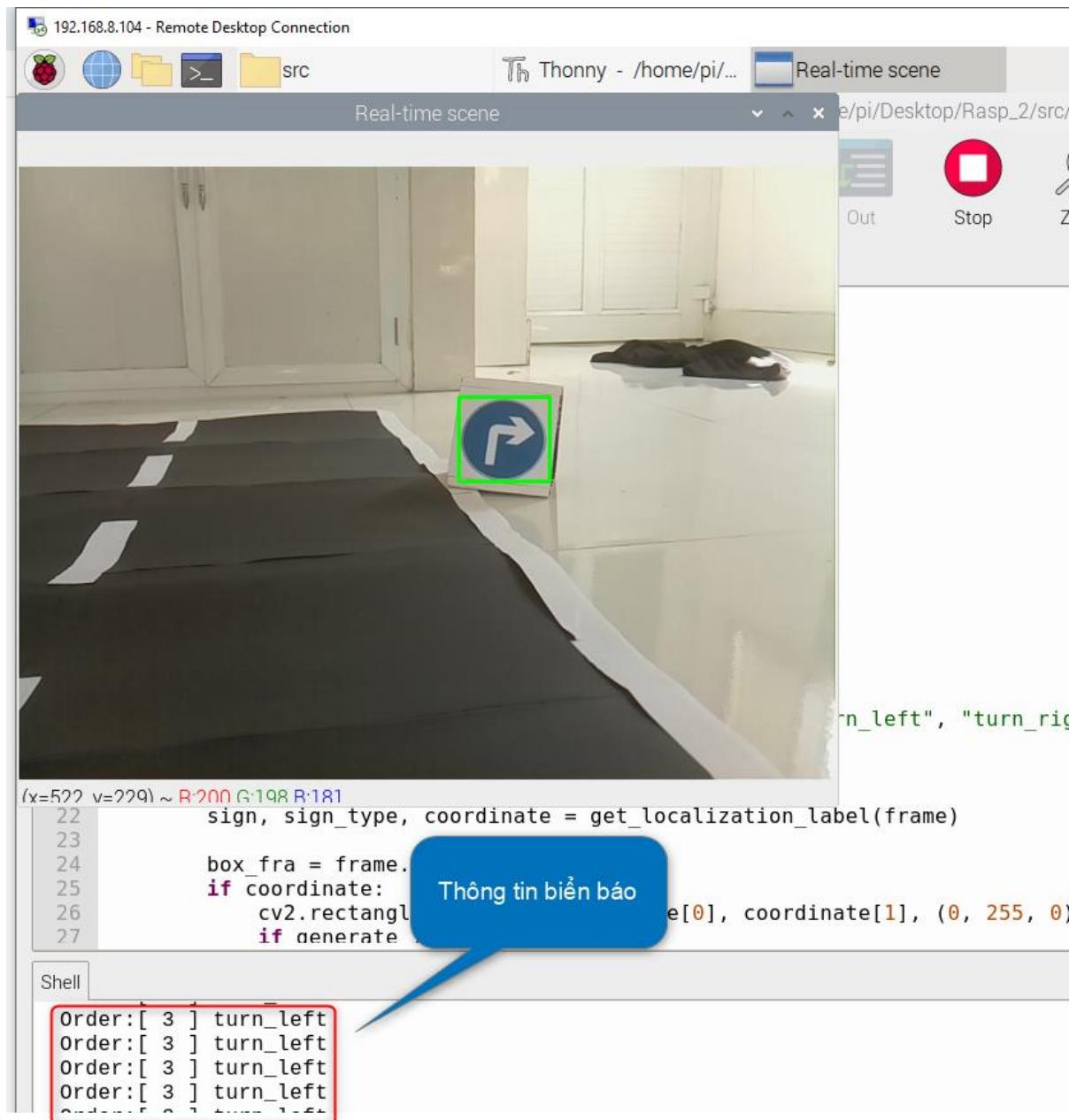


Hình 5.15 Kết quả nhận dạng biển báo giới hạn tốc độ 10km/h



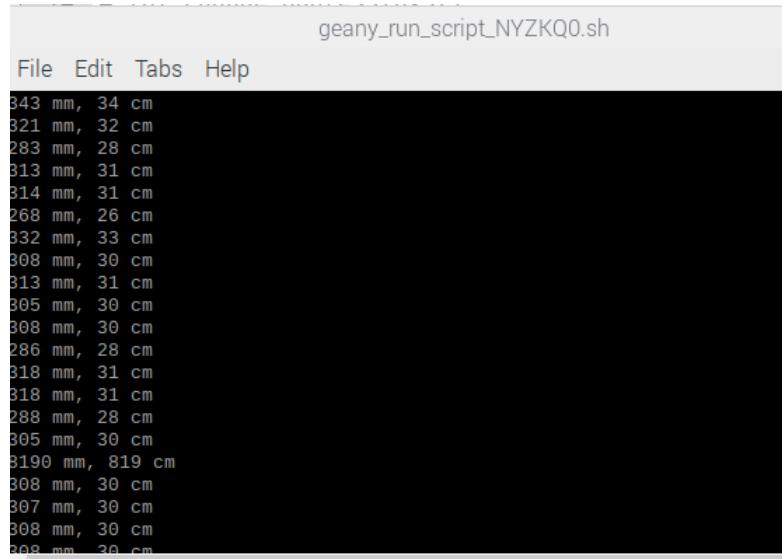


Hình 5.17 Kết quả nhận dạng biển báo rẽ trái



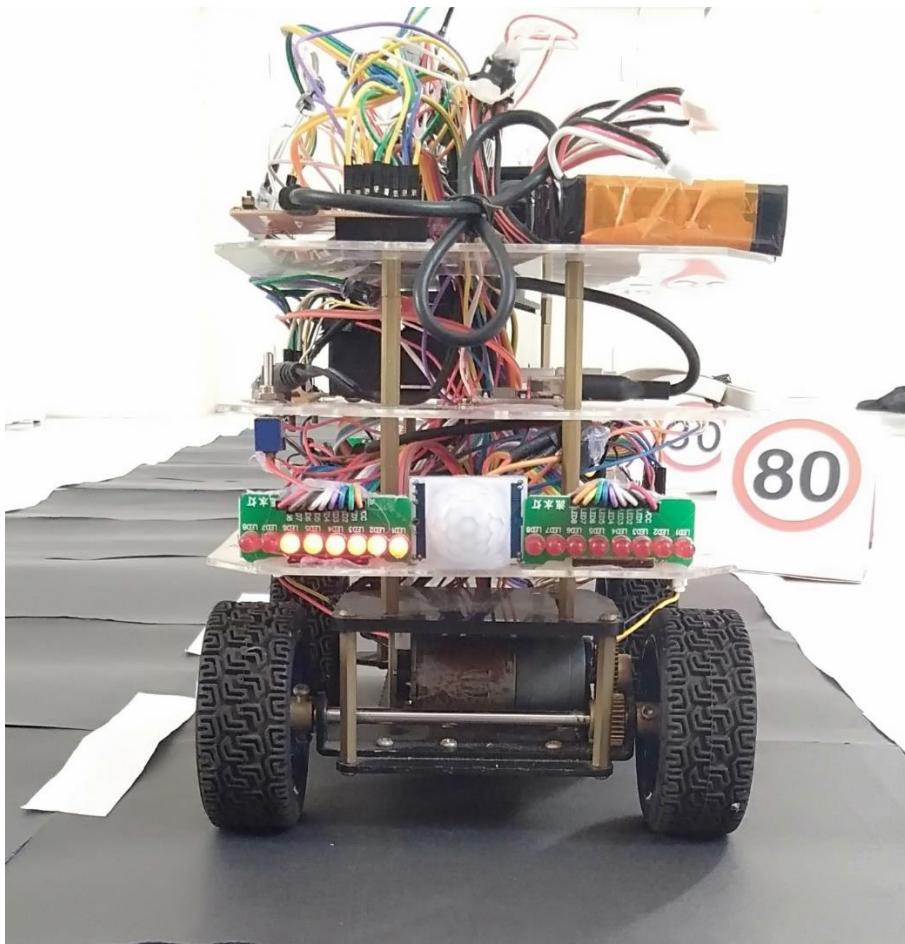
Hình 5.18 Kết quả nhận diện biển báo rẽ phải

#### 5.4 Chức năng chuyển làn

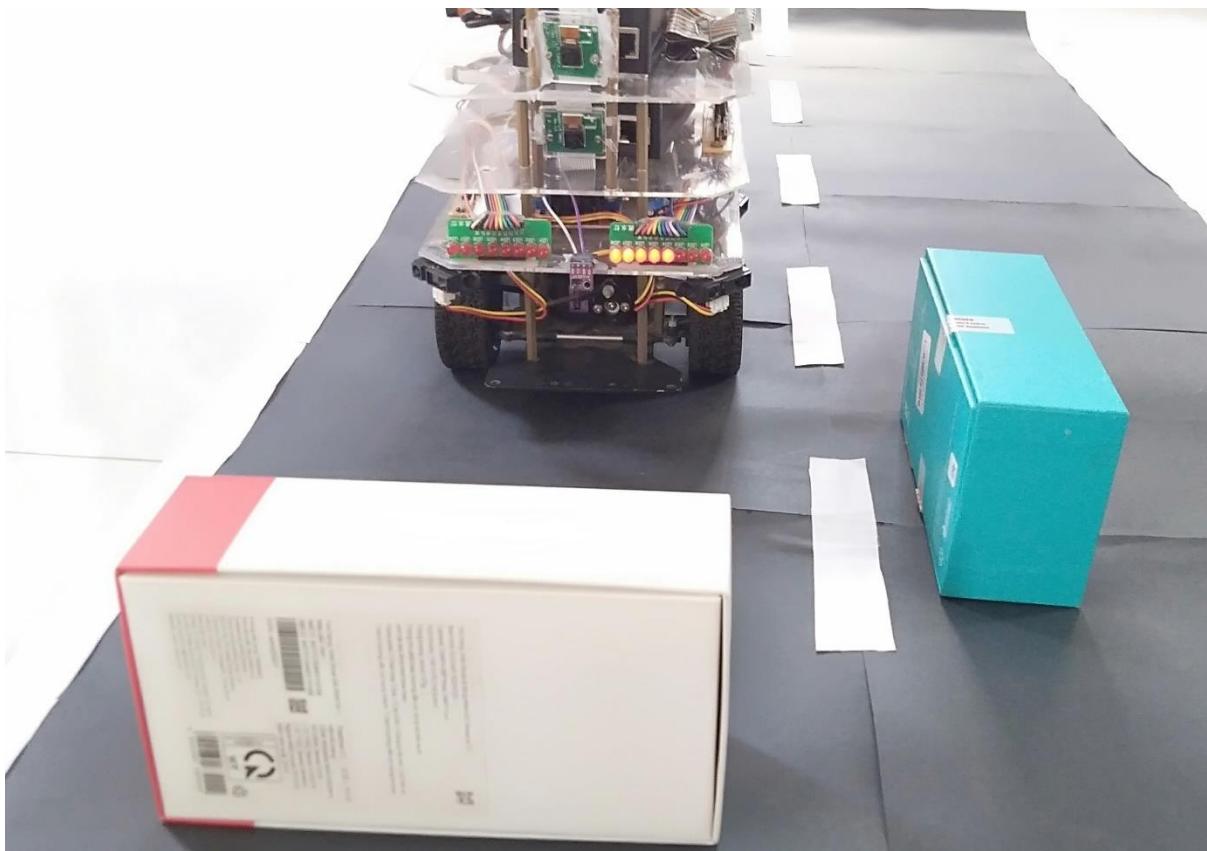


```
geany_run_script_NYZKQ0.sh
File Edit Tabs Help
343 mm, 34 cm
321 mm, 32 cm
283 mm, 28 cm
313 mm, 31 cm
314 mm, 31 cm
268 mm, 26 cm
332 mm, 33 cm
308 mm, 30 cm
313 mm, 31 cm
305 mm, 30 cm
308 mm, 30 cm
286 mm, 28 cm
318 mm, 31 cm
318 mm, 31 cm
288 mm, 28 cm
305 mm, 30 cm
3190 mm, 819 cm
308 mm, 30 cm
307 mm, 30 cm
308 mm, 30 cm
309 mm, 30 cm
```

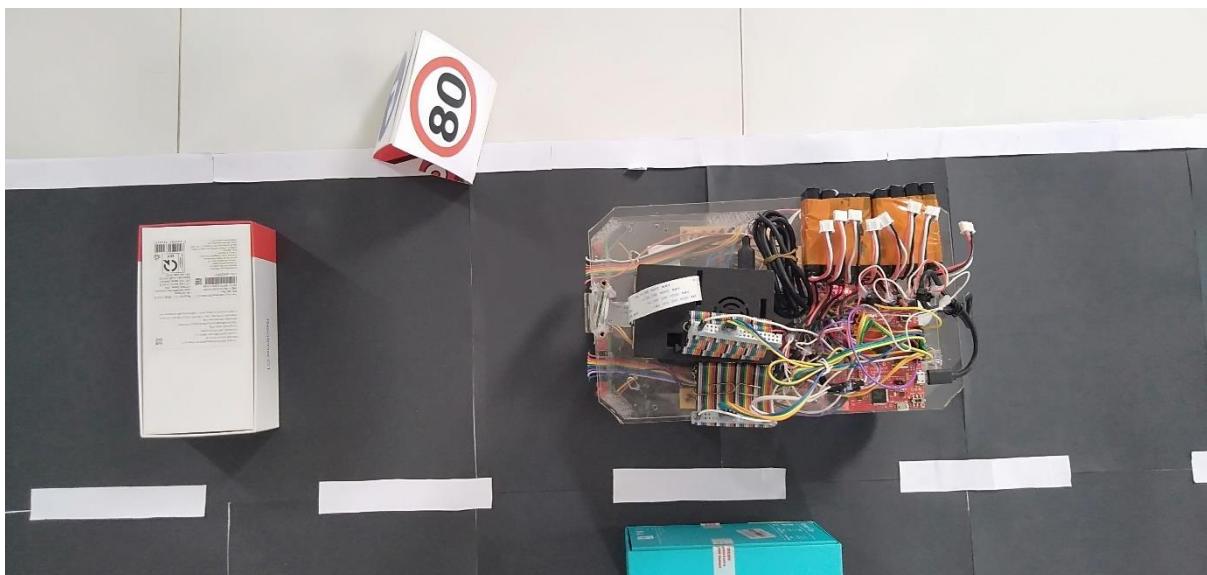
Hình 5.19 Khoảng cách được đo từ cảm biến lazer



Hình 5.20 Đèn báo hiệu rẽ trái khi xe chuẩn bị rẽ trái



Hình 5.21 Xe bật đèn rẽ trái khi gặp vật cản phía trước và bên trái



Hình 5.22 Xe dừng lại khi gặp vật cản và không thể rẽ trái

### 5.5 Các số liệu đánh giá kết quả đạt được

Bảng 5.1 Kết quả nhận dạng làn đường

Tốc độ	Số lần thử nghiệm	Số lần chạy chính xác	Tỉ lệ chạy chính xác
Thấp (~0.5m/s)	100	95	95%
Trung bình (~0.8m/s)	100	63	63%
Cao (~1.2m/s)	100	45	45%

Bảng 5.2 Kết quả nhận dạng biển báo

Tên biển báo	Số lần thử nghiệm	Số lần nhận dạng chính xác	Tỉ lệ nhận dạng chính xác
Biển báo giới hạn tốc độ 80 km/h	100	90	90%
Biển báo giới hạn tốc độ 30 km/h	100	87	87%
Biển báo giới hạn tốc độ 10 km/h	100	95	95%
Biển báo dừng	100	93	93%
Biển báo rẽ trái	100	86	86%
Biển báo rẽ phải	100	65	65%

Bảng 5.3 Kết quả chuyển làn khi gấp vật cản

Tốc độ	Số lần thử nghiệm	Số lần thực hiện chính xác	Tỉ lệ thực hiện chính xác
Thấp (~0.5m/s)	100	97	97%
Trung bình (~0.8m/s)	100	85	85%
Cao (~1.2m/s)	100	72	72%

Nhận xét: Với kết quả thu được sau khi hoàn thành mô hình và chạy thực tế ta có thể thấy rằng trong dải tốc độ thấp và trung bình thì xe hoạt động tương đối tốt nhưng trong khi chạy thử nghiệm với tốc độ cao thì kết quả không được như ý muốn. Nguyên nhân chính dẫn đến khuyết điểm như vậy chính là khả năng đáp ứng của động cơ servo RC, do độ trễ đáp ứng của động cơ này tương đối lớn so với tốc độ mà ta đang thử nghiệm. Khi xe chạy với tốc độ cao mà đáp ứng chậm trong việc lái hai bánh trước dẫn đến xe sẽ nhanh chóng bị lệch khỏi làn đường. Bên cạnh đó, trong khi chạy với tốc độ

cao sẽ dẫn đến quán tính của xe lớn sẽ phần nào ảnh hưởng đến khoảng cách của xe với vật cản khi xe bất ngờ gặp vật cản trên làn đang chạy. Ngoài ra, mô hình này không được trang bị hệ thống phanh nên ít nhiều có ảnh hưởng đến khả năng dừng lại của xe trong những trường hợp cần thiết. Về phần nhận dạng biển báo giao thông, đa số hệ thống nhận diện tốt nhưng với biển báo rẽ trái và rẽ thì tỉ lệ nhận diện vẫn chưa được cao như các biển báo còn lại. Nguyên nhân chính dẫn đến kết quả như vậy là do các hình ảnh ban đầu để huấn luyện cho mô hình chưa được rõ ràng và thuật toán nhận dạng chưa được tối ưu.

## 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 6.1 Kết luận

Ưu điểm:

- + Chạy đúng làn đường.
- + Điều chỉnh góc lái tốt trong dải tốc độ thấp và trung bình.
- + Nhận dạng được một số loại biển báo cơ bản.
- + Tư điều chỉnh được tốc độ trong trường hợp phát hiện các biển báo giới hạn tốc độ.
- + Tránh được vật cản khi xe đang di chuyển.
- + Tự động chuyển làn trong trường hợp các điều kiện môi trường xung quanh cho phép.
- + Kích hoạt hệ thống cảnh báo trong trường hợp không được phép di chuyển do gặp biển stop.
- + Phát tín hiệu xin chuyển làn trong khi tránh vật cản phía trước và tiếp tục hành trình khi không còn vật cản.

Khuyết điểm:

- + Góc lái chưa linh hoạt.
- + Chưa nhận dạng được hết tất cả các biển báo ngoài thực tế hiện nay.
- + Khi chạy ở tốc độ cao thì độ chính xác chưa cao như mong đợi.

### 6.2 Hướng phát triển

Có thể được sử dụng tại các đoạn đường có mật độ giao thông thấp, tình hình giao thông không phức tạp.

Gắn thêm pin năng lượng mặt trời để cung cấp thêm năng lượng cho xe lúc đang vận hành.

Gắn thêm định vị GPS và kết nối internet trên xe để xe có thể gửi dữ liệu về hệ thống cloud cũng như giao tiếp giữa các xe gần nhau.

Tích hợp ứng dụng Google Maps thêm trên xe, dùng ứng dụng này để vẽ ra lộ trình để xe chạy đến đích đến đã chỉ ra trên bản đồ.

Chúng ta có thể có được thông tin về góc của xe bằng cách thêm nhiều cảm biến và trực tiếp điều khiển góc quay.

Phương pháp phát hiện có thể được thay đổi để cho phép thuật toán xử lý nhiều hình dạng biển báo đường hơn.

Có thể sử dụng nhiều phương pháp tiền xử lý hơn để giải quyết ảnh hưởng của các điều kiện ánh sáng khác nhau.

Phát triển chương trình nhận dạng trong trường hợp các biển báo bị che khuất một phần.

Tối ưu phần cứng và phần mềm để hệ thống có thể hoạt động tốt trong nhiều điều kiện ánh sáng khác nhau như trời nắng gắt, trời mưa hay ban đêm,...

## 7. TÀI LIỆU THAM KHẢO

- [1] <https://zingnews.vn/he-thong-ho-tro-lai-tren-oto-khong-the-thay-the-tai-xe-post1010678.html>
- [2] <https://www.thegioididong.com/tin-tuc/xu-huong-xe-tu-lai-thong-minh-tuong-lai-cua-nganh-cong-nghe-oto-la-day--667481>
- [3] <https://techmaster.vn/posts/33943/opencv-vacac-ung-dung-cua-no-hiennay>
- [4] [https://opencv-python-tutorial.readthedocs.io/en/latest/py\\_tutorials/py\\_imgproc/py\\_houghlines/py\\_houghlines.html](https://opencv-python-tutorial.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_houghlines/py_houghlines.html)
- [5] [https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous\\_transformations.html](https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html)
- [6] [https://opencv-python-tutorial.readthedocs.io/en/latest/py\\_tutorials/py\\_imgproc/py\\_filtering/py\\_filtering.html](https://opencv-python-tutorial.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_filtering/py_filtering.html)
- [7] [https://docs.opencv.org/master/d4/d13/tutorial\\_py\\_filtering.html](https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html)
- [8] [https://opencv-python-tutorial.readthedocs.io/en/latest/py\\_tutorials/py\\_imgproc/py\\_canny/py\\_canny.html](https://opencv-python-tutorial.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_canny/py_canny.html)
- [9] <https://hocarm.org/rtos-co-ban-phan-1/>
- [10] <https://vi.wikipedia.org/wiki/TensorFlow>
- [11] <https://topdev.vn/blog/thuat-toan-cnn-convolutional-neural-network/>
- [12] <http://dammedientu.vn/gioi-thieu-chuan-giao-tiep-i2c/>