# LỜI CẢM ƠN

🙢🕮🙠

Lời đầu tiên, em xin gửi lời biết ơn sâu sắc tới Ts. Trần Công Án và Ths. Nguyễn Hữu Vân Long, các Thầy đã trực tiếp hướng dẫn, định hướng, đóng góp ý kiến, dạy bảo những kiến thức, chia sẽ kinh nghiệm cần thiết và quý giá để giúp đỡ em hoàn thành luận văn tốt nghiệp này.

Tiếp đến, em xin chân thành cảm ơn đến quý thầy cô Trường Đại học Cần Thơ nói chung, đặc biệt là quý thầy cô Khoa Công nghệ thông tin và Truyền thông nói riêng, những người đã luôn tận tâm giúp đở, dạy bảo và truyền đạt cho em những kiến thức quý báo về lý thuyết và thực hành trong suốt thời gian học tập ở trường.

Em cũng xin bày tỏ lòng biết ơn đến Ts. Ngô Bá Hùng là giáo viên cố vấn lớp DI15Y9A1. Cám ơn Thầy trong suốt khoảng thời gian vừa qua đã luôn gắn bó hết lòng với tập thể lớp. Thầy đã quan tâm, dạy bảo cho lớp trong những ngày đầu bước chân vào giảng đường đại học. Và cũng xin gửi lời cảm ơn tới gia đình, bạn bè đã động viên và luôn luôn bên cạnh trong những lúc khó khăn nhất.

Dù đã cố gắng rất nhiều để hoàn thành đề tài tốt nhất có thể và theo tiến độ dự kiến nhưng không thể tránh khỏi những thiếu sót, hạn chế về mặt kiến thức cũng như xử lý lỗi trong quá trình làm bài. Rất mong nhận được sự đóng góp ý kiến từ quý thầy cô và các bạn để đề tài có thể phát triển hoàn thiện hơn và được đưa vào sử dụng rộng rãi trong thực tế.

Với lòng biết ơn sâu sắc và đầy sự kính trọng, em xin chân thành cảm ơn!

Cần Thơ, ngày tháng năm 2019

Sinh viên thực hiện

Huỳnh Thị Yến Quyên

# MỤC LỤC

[LỜI CẢM ƠN i](#_Toc26475874)

[MỤC LỤC ii](#_Toc26475875)

[DANH MỤC BẢNG v](#_Toc26475876)

[DANH MỤC HÌNH v](#_Toc26475877)

[KÝ HIỆU VÀ VIẾT TẮT vii](#_Toc26475878)

[TÓM TẮT ĐỀ TÀI viii](#_Toc26475879)

[ABSTRACT ix](#_Toc26475880)

[PHẦN GIỚI THIỆU 1](#_Toc26475881)

[1. Đặt vấn đề 1](#_Toc26475882)

[2. Lịch sử giải quyết vấn đề 1](#_Toc26475883)

[3. Mục tiêu đề tài 2](#_Toc26475884)

[4. Đối tượng và phạm vi nghiên cứu 3](#_Toc26475885)

[5. Nội dung nghiên cứu 3](#_Toc26475886)

[6. Bố cục luận văn 4](#_Toc26475887)

[PHẦN NỘI DUNG 5](#_Toc26475888)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 5](#_Toc26475889)

[1.1 Arduino IDE 5](#_Toc26475890)

[1.2 Raspbian 6](#_Toc26475891)

[1.3 Các gói cài đặt và thư viện 6](#_Toc26475892)

[1.3.1 Yolo v3 6](#_Toc26475893)

[1.3.3 OpenCV 6](#_Toc26475894)

[1.3.4 TensorFlow 7](#_Toc26475895)

[1.3.5 Keras 7](#_Toc26475896)

[1.3.6 ImageAI 8](#_Toc26475897)

[1.4 Phần cứng thiết bị 8](#_Toc26475898)

[1.4.1 Arduino Mega 2560 8](#_Toc26475899)

[1.4.2 Cảm biến khoảng cách HC-SR04 10](#_Toc26475900)

[1.4.3 Động cơ RC Servo 9G SG90 11](#_Toc26475901)

[1.4.4 Module điều khiển động cơ L298N. 12](#_Toc26475902)

[1.4.5 Động cơ giảm tốc 13](#_Toc26475903)

[1.4.6 Raspberry Pi 3 B+ 14](#_Toc26475904)

[1.4.7 Camera 15](#_Toc26475905)

[1.4.8 Mạch giảm áp DC LM2596 16](#_Toc26475906)

[1.4.9 Pin Lipo 16](#_Toc26475907)

[1.4.10 Khung xe robot 4 bánh 17](#_Toc26475908)

[Tổng kết chương 1. 18](#_Toc26475909)

[CHƯƠNG 2: MÔ TẢ BÀI TOÁN 19](#_Toc26475910)

[2.1 Mô tả bài toán 19](#_Toc26475911)

[2.1.1 Mô tả tổng quan 19](#_Toc26475912)

[2.1.2 Yêu cầu giao tiếp 20](#_Toc26475913)

[2.2 Phân tích 20](#_Toc26475914)

[2.2.1 Các tính năng của mô hình 20](#_Toc26475915)

[2.2.2 Các yêu cầu phi chức năng 20](#_Toc26475916)

[2.3 Đánh giá 21](#_Toc26475917)

[2.3.1 Đánh giá phần cứng 21](#_Toc26475918)

[2.3.2 Đánh giá phần mềm 22](#_Toc26475919)

[2.3.3 Đánh giá chung 22](#_Toc26475920)

[Tổng kết chương 2. 23](#_Toc26475921)

[CHƯƠNG 3: THIẾT KẾ VÀ CÀI ĐẶT GIẢI PHÁP 24](#_Toc26475922)

[3.1 Thiết kế module 24](#_Toc26475923)

[3.1.1 Module tránh vật cản 24](#_Toc26475924)

[3.1.2 Module phát hiện lửa bằng camera 25](#_Toc26475925)

[3.2 Cài đặt phần mềm 26](#_Toc26475926)

[3.3 Lưu đồ thuật toán 27](#_Toc26475927)

[3.3.1 Tránh vật cản 27](#_Toc26475928)

[3.3.2 Phát hiện lửa bằng camera trên Raspberry Pi 3 B+ 29](#_Toc26475929)

[3.4 Sơ đồ kết nối và cài đặt mô hình 30](#_Toc26475930)

[3.4.1 Kết nối trên Arduino Mega 2560 30](#_Toc26475931)

[3.4.2 Kết nối trên Raspberry 37](#_Toc26475932)

[Tổng kết chương 3. 38](#_Toc26475933)

[CHƯƠNG 4: ĐÁNH GIÁ VÀ KIỂM THỬ 39](#_Toc26475934)

[4.1 Mục tiêu kiểm thử 39](#_Toc26475935)

[4.2 Quá trình kiểm thử 39](#_Toc26475936)

[4.2.1 Kiểm thử trên board Mega 2560 39](#_Toc26475937)

[4.2.2 Kiểm thử trên Raspberry Pi 3 B+ 40](#_Toc26475938)

[4.3 Các trường hợp kiểm thử 40](#_Toc26475939)

[4.4 Đánh giá kết quả kiểm thử 44](#_Toc26475940)

[Tổng kết chương 4. 44](#_Toc26475941)

[PHẦN KẾT LUẬN 45](#_Toc26475942)

[1. Kết quả đạt được 45](#_Toc26475943)

[2. Khó khăn 46](#_Toc26475944)

[3. Hướng phát triển 46](#_Toc26475945)

[TÀI LIỆU THAM KHẢO 48](#_Toc26475946)

[PHỤ LỤC 49](#_Toc26475947)

# DANH MỤC BẢNG

🙢🕮🙠

[Bảng 1. 1 Cổng giao tiếp phần cứng 8](#_Toc26517262)

[Bảng 1. 2 Sơ đồ nối chân giữa HC-SR04 với Arduino 11](#_Toc26517263)

[Bảng 2. 1 Ưu và nhược điểm phần cứng 22](#_Toc26477854)

[Bảng 3. 1 Sơ đồ chân cắm của Mega với Cảm biến khoảng cách và Servo 30](#_Toc26519034)

[Bảng 3. 2 Sơ đồ chân cắm Mega với L298N và bốn động cơ giảm tốc 32](#_Toc26519035)

[Bảng 4. 1 Kiểm thử các chức năng trên Mega 40](#_Toc26517275)

[Bảng 4. 2 Kiểm thử các tính năng trên Raspberry 40](#_Toc26517276)

# DANH MỤC HÌNH

🙢🕮🙠

[Hình 1. 1 Giao diện Arduino IDE 5](#_Toc26478172)

[Hình 1. 2 Hệ điều hành Raspbian 6](file:///C:\Users\htyqu\Desktop\LuanvanQuyen%20(1).docx#_Toc26478173)

[Hình 1. 3 Mega 2560 9](#_Toc26478174)

[Hình 1. 4 Sơ đồ hoạt động của Cảm biến HC-SR04 10](#_Toc26478175)

[Hình 1. 5 Cảm biến khoảng cách 11](#_Toc26478176)

[Hình 1. 6 Động cơ Servo 12](#_Toc26478177)

[Hình 1. 7 Module L298N 13](#_Toc26478178)

[Hình 1. 8 Động cơ giảm tốc 13](#_Toc26478179)

[Hình 1. 9 Raspberry Pi 3 B+ 14](#_Toc26478180)

[Hình 1. 10 Camera 15](#_Toc26478181)

[Hình 1. 11 Mạch giảm áp LM2596 16](file:///C:\Users\htyqu\Desktop\LuanvanQuyen%20(1).docx#_Toc26478182)

[Hình 1. 12 Pin Lipo 17](file:///C:\Users\htyqu\Desktop\LuanvanQuyen%20(1).docx#_Toc26478183)

[Hình 1. 13 Khung xe bốn bánh 17](file:///C:\Users\htyqu\Desktop\LuanvanQuyen%20(1).docx#_Toc26478184)

[Hình 2. 1 Khối xử lý 19](file:///C:\Users\htyqu\Desktop\LuanvanQuyen%20(1).docx#_Toc26476390)

[Hình 3. 1 Sơ đồ hoạt động của module tránh vật cản 24](file:///C:\Users\htyqu\Desktop\LuanvanQuyen%20(1).docx#_Toc26478210)

[Hình 3. 2 Sơ đồ hoạt động của module nhận dạng lửa bằng camera 25](#_Toc26478211)

[Hình 3. 3 Thuật toán tránh vật cản 27](#_Toc26478212)

[Hình 3. 4 Kết nối cảm biến khoảng cách 30](#_Toc26478213)

[Hình 3. 5 Kết nối động cơ bánh xe 32](#_Toc26478214)

[Hình 3. 6 Sơ đồ tránh vật cản 35](file:///C:\Users\htyqu\Desktop\LuanvanQuyen%20(1).docx#_Toc26478215)

[Hình 3. 7 Mô hình kết nối tổng quát trên Mega 36](#_Toc26478216)

[Hình 3. 8 Phát hiện lửa trên Raspberry gửi sang Mega 37](#_Toc26478217)

[Hình 4. 1 Xe gặp vật cản và khoảng cách trái, phải nhỏ hơn 10cm 41](#_Toc26478333)

[Hình 4. 2 Xe gặp vật cản và khoảng cách trái lớn hơn phải 41](#_Toc26478334)

[Hình 4. 3 Xe gặp vật cản và khoảng cách phải lớn hơn trái 42](#_Toc26478335)

[Hình 4. 4 Nhận dạng lửa từ Camera lần 1 42](#_Toc26478336)

[Hình 4. 5 Nhận dạng lửa từ Camera lần 2 43](#_Toc26478337)

[Hình 4. 6 Nhận dạng lửa từ Camera lần 3 43](#_Toc26478338)

[Hình 1 Mô hình xe cứu hỏa mini 45](#_Toc26478449)

[Hình 2 Các thiết bị làm khung xe 49](#_Toc26478450)

[Hình 3 Động cơ giảm tốc đã nối dây 49](#_Toc26478451)

[Hình 4 Khung xe đã lắp động cơ giảm tốc 50](#_Toc26478452)

[Hình 5 Khung xe đã hoàn thành 50](#_Toc26478453)

[Hình 6 Noobs 51](#_Toc26478454)

# KÝ HIỆU VÀ VIẾT TẮT

|  |  |
| --- | --- |
| **Ký hiệu và viết tắt** | **Viết đấy đủ** |
| IDE | Integrated Development Environment |
| CPU [ARM](https://vi.wikipedia.org/wiki/C%E1%BA%A5u_tr%C3%BAc_ARM) | Central Processing Unit Advanced RISC Machine |
| GPIO | General Purpose Input Output |
| USB | Universal Serial Bus |
| URL | Uniform Resource Locator |
| GUI | Graphical User Interface |
| DC | Direct Current |
| IC | Integrated Circuit |
| PPM | Parts Per Million |
| VNC | Virtual Network Computing |
| UART | Universal Asynchronous Receiver Transmitter |
| VCC | Voltage Controlled Clock |
| GND | Ground |
| ICSP | In Circuit Serial Programming |
| TX | Transmitter |
| RX | Receiver |
| XML | eXtensible Markup Language |
| OpenCV | Open Source Computer Vision Library |
| Yolo | You only look once |
| CNN | Convolutional Neural Network |
| RNN | Recurrent neural network |

# TÓM TẮT ĐỀ TÀI

🙢🕮🙠

Ngày nay, tai nạn hỏa hoạn ngày càng trở nên phổ biến và trở thành vấn đề luôn được xã hội đặc biệt quan tâm. Hỏa hoạn có thể xảy ra ở bất cứ đâu vào bất kỳ thời điểm nào, con người không thể theo dõi liên tục và có mặt kịp thời trong các tình huống xảy ra hỏa hoạn. Việc phát hiện ra hỏa hoạn có thể nói là rất quan trọng vì nó không những ảnh hưởng đến tài sản mà còn là ranh giới giữa sự sống và cái chết. Do đó, với hy vọng xây dựng một hệ thống thông minh giúp con người trong việc bảo vệ tính mạng, hạn chế rủi ro do hỏa hoạn gây ra. Chúng tôi đã chọn đề tài "Mô hình thử nghiệm xe cứu hỏa tự hành mini" để xây dựng đề tài luận văn của mình. Phân hệ “Phát hiện vật cản, thay đổi lộ trình và nhận dạng lửa bằng camera” được tôi lựa chọn để thực hiện.

Trong nghiên cứu này, việc phát hiện, tự động việc tránh chướng ngại vật và nhận diện lửa bằng camera được tập trung để phân tích. Mô hình là một thiết bị thông minh có thể tự động cảm nhận chướng ngại vật phía trước nó và tránh chúng bằng cách tự xoay theo hướng khác. Mục đích của việc nghiên cứu này là thiết kế một mô hình xe sử dụng board mạch Arduino Mega 2560 kết nối với Raspberry Pi 3 B+, cảm biến và các động cơ. Để phát hiện chướng ngại vật sử dụng cảm biến khoảng cách siêu âm HC-SR04 gắn trên động cơ Servo và camera được sử dụng để phát hiện đám cháy trong phạm vi có thể. Ngoài ra, nó cũng được trang bị còi báo động để thông báo ngay lập tức khi đám cháy xảy ra. Xe chữa cháy tự động sẽ là giải pháp hoàn hảo để bảo vệ tài sản, tính mạng con người và môi trường xung quanh.

# ABSTRACT

🙢🕮🙠

Nowadays, fire accidents become more and more popular and also become a special concern to our society. Fire incidents can happen anywhere at any time, people cannot monitor continuously and be present in time of the fire situations when it occurs. The discovery of a flame sign can be said to be very important because it affects not only the properties but also the boundary between people’s survival and death. Therefore, in the hope of building an intelligent system that helps people in protecting their lives, limiting the risks caused by fires, we have chosen the topic on "Model self-propelled mini fire engines" to conduct my thesis. I selected the module "Detection obstacles, changing routes and detecting fire using camera" to implement.

  In this study, the detection and automatic obstacle avoidance and detecting fire using camera were concentrated for the analysis. Module is an intelligent device which can automatically sense the obstacle in front of it and avoid them by turning itself in another direction. The purpose of this research is to design a vehicle model using an Arduino Mega 2560 board connected to the Raspberry Pi 3 B+, sensors and motors. To detect obstacles using the ultrasonic distance sensor HC-SR04 ultra-sonic sensor mounted on a Servo motor and the camera is used to detect the flame in the setting area. Moreover, it is also equipped with a siren to notify immediately when a fire occurs. Fire engines will be the perfect solution to protect properties, human life and the surrounding environment.

# PHẦN GIỚI THIỆU

**1. Đặt vấn đề**

Khoa học công nghệ ở thế kỷ 21 hiện đang không ngừng phát triển. Đặc biệt là ở các lĩnh vực liên quan đến công nghệ thông tin. Sự ảnh hưởng của nó không chỉ phát triển trong nội bộ mà còn gây ra một sự ảnh hưởng mạnh mẽ đến các lĩnh vực khác như: giáo dục, y tế, chăm sóc sức khỏe, nông nghiệp, công nghiệp, hàng không, vũ trụ,…đặc biệt trong cả lĩnh vực phòng cháy chữa cháy.

Có rất nhiều nguyên nhân dẫn tới các thiệt hại vô cùng nghiêm trọng khi xảy ra cháy nổ, nhưng một trong những nguyên nhân chính là do phát hiện đám cháy muộn. Việc này dẫn tới nhiều hệ lụy như: đám cháy phát triển nhanh mất kiểm soát, đám cháy lan rộng ra xung quanh, lực lượng chức năng về phòng cháy chữa cháy (PCCC) tiếp nhận thông tin báo cháy chưa kịp thời, dẫn tới việc triển khai cứu hỏa, cứu nạn, cứu hộ bị chậm trễ…

Vì vậy, việc phát hiện cháy sớm sẽ góp phần giảm thiểu đến mức thấp nhất các thiệt hại về tính mạng, tài sản do cháy nổ gây ra. Công nghệ có thể góp phần thay thế con người công việc hết sức nguy hiểm này. Chính vì thế việc xây dựng mô hình xe cứu hỏa tự hành là phù hợp với nhu cầu thực tế và thúc đẩy sự sáng tạo và đáp ứng thời đại.

Do đó, trong phạm vi hiểu biết của mình, chúng tôi đã tìm hiểu và thực hiện đề tài: “Mô hình thử nghiệm xe cứu hỏa mini tự hành” với 2 phân hệ chính là “Phát hiện vật cản, thay đổi lộ trình và nhận dạng lửa bằng camera” và “Phát hiện và dập tắt lửa bằng cảm biến và camera”. Trong báo cáo này, tôi tập trung nghiên cứu phân hệ “ Phát hiện vật cản, thay đổi lộ trình và nhận dạng lửa bằng camera” với mong muốn xe có thể tự động tránh va chạm vật cản và kết hợp nhận dạng lửa.

**2. Lịch sử giải quyết vấn đề**

Hiện nay, những mô hình về xe tự hành kết hợp với các module khác như hút bụi, dò mìn, vận chuyển hàng hóa… hay đặc biệt là chữa cháy đang được xem là đề tài mà nhiều người quan tâm và nghiên cứu được xây dựng với nhiều dạng mô hình, nền tảng và những ngôn ngữ khác nhau. Nó là một trong những đề tài khá hay và rất phù hợp với việc học tập và nghiên cứu của mọi người hiện nay. Tại Việt Nam đang có nhiều mô hình robot chữa cháy do các bạn học sinh nghiên cứu như: "Robot chữa cháy điều khiển từ xa bằng sóng RF" của hai học sinh Lê Mai Hương và Trần Thanh Tuấn, học sinh lớp 9 năm học 2017-2018, trường Trung học cơ sở Lâm Mộng Quang, huyện Phú Lộc, Thừa Thiên Huế. "Robot cứu hỏa" của Võ Đặng Văn Thành và Phạm Hồng Thái, học sinh lớp 12, trường Trung học phổ thông Trần Quốc Tuấn, Quảng Ngãi…

Tại nhiều nước phát triển trên thế giới như Mỹ, Anh, Ấn Độ, Nhật Bản, Nga… cũng đã phát triển mô hình xe tự hành kết hợp cứu hỏa. Tại Mỹ, “Robot Thermite” là sản phẩm nghiên cứu của công ty công nghệ Howe and Howe giúp nhân viên cứu hỏa an toàn hơn khi làm nhiệm vụ; “Robot SAFFiR” cho các hoạt động phát hiện và dập tắt đám cháy trên tàu biển; tại Anh, Công ty công nghệ quốc phòng QinetiQ của Anh thiết kế 4 loại robot phục vụ cho mục đích dập tắt các vụ cháy liên quan đến khí acetylene công nghiệp… Tuy nhiên, những mô hình này đa số là được điều khiển từ xa và chỉ tập trung chủ yếu vào việc dập lửa. Với mong muốn phát triển mô hình có khả năng tự động phát hiện lửa và kết hợp dập tắt đám cháy tạm thời tại chỗ để hạn chế tối đa thiệt hại mà hỏa hoạn gây ra tôi đã xây dựng "Mô hình thử nghiệm xe cứu hỏa mini tự hành"để nó có thể tự động kịp thời phát hiện lửa và dập tắt đám cháy ngay lập tức. Vì vậy, hy vọng đề tài sẽ giúp ích trong việc học tập, tìm hiểu, xây dựng và ứng dụng vào thực tiễn.

**3. Mục tiêu đề tài**

Mục tiêu chung của đề tài “Mô hình thử nghiệm xe cứu hỏa mini tự hành” là tìm hiểu về Raspberry và Mega để xây dựng một số module phục vụ cho mô hình xe có thể tự động di chuyển, phát hiện và dập lửa bên trong tòa nhà. Cụ thể trong phân hệ này sẽ tập trung nghiên cứu về module “Phát hiện vật cản, thay đổi lộ trình và nhận dạng lửa bằng camera”.

Mục tiêu đặt ra của phân hệ này:

* Xe cần phát hiện và điều hướng tránh được vật cản trong quá trình di chuyển xử lý trên board Mega 2560.
* Xe cần thu được hình ảnh qua camera và xử lý nhận dạng có lửa trên Raspberry Pi 3 B+.
* Raspberry gửi tín hiệu cho Mega 2560 để bật còi báo và điều hướng di chuyển của xe về vị trí có lửa.

Với mong muốn xe cứu hỏa tự hành sẽ thay thế con người trong việc kiểm tra, tiếp nhận thông tin một cách kịp thời và giải quyết nhanh chóng những nguy cơ dẫn đến cháy nổ trong phạm vi có thể. Và nó sẽ giảm thiểu được sự bùng phát, lây lan của những đám cháy, hạn chế nhiều thiệt hại do hỏa hoạn gây ra đến tính mạng và tài sản con người.

**4. Đối tượng và phạm vi nghiên cứu**

Với phân hệ “Phát hiện vật cản, thay đổi lộ trình và nhận dạng lửa bằng camera**”** tập trung nghiên cứu về cách thức vận hành của xe trong môi trường văn phòng.

* Kết nối Cảm biến khoảng cách HC-SR04 với board Arduino Mega 2560, nhận tín hiệu từ cảm biến, xử lý điều hướng di chuyển của xe bằng module điều khiển động cơ L298N.
* Kết hợp động cơ Servo 9G SG90 với Cảm biến khoảng cách để tăng góc quan sát cho cảm biến.
* Nhận dạng lửa từ camera gắn trên Raspberry gửi tín hiệu nhận dạng lửa sang Mega để xử lý tín hiệu đồng thời điều khiển hoạt động và hướng di chuyển của động cơ.
* Cả Raspberry và Mega đều lập trình trên nhiều ngôn ngữ lập trình khác nhau như C, C++, Python... Trong đề tài sử dụng ngôn ngữ lập trình C trên nền tảng Arduino IDE cho Mega và ngôn ngữ Python để lập trình cho Raspberry.

**5. Nội dung nghiên cứu**

|  |  |
| --- | --- |
| Nội dung công việc | Người thực hiện |
| - Tìm hiểu và lập trình xử lý tránh vật cản trên Mega | Huỳnh Thị Yến Quyên |
| - Xây dựng mô hình lắp đặt phần cứng cho xe | Huỳnh Thị Yến Quyên |
| - Lập trình xử lý ảnh để phát hiện lửa bằng camera trên Raspberry | Huỳnh Thị Yến Quyên |
| - Thiết lập kết nối giữa Raspberry và Mega | Huỳnh Thị Yến Quyên |

**6. Bố cục luận văn**

Bố cục của luận văn bao gồm:

* Phần giới thiệu: trình bày sơ lược, lịch sử, mục tiêu, đối tượng và phạm vi nghiên cứu của đề tài.
* Phần nội dung được trình bày theo bố cục sau:

Chương 1: Trình bày cơ sở lý thuyết phân hệ “Phát hiện vật cản, thay đổi lộ trình và nhận dạng lửa bằng camera” gồm: giới thiệu tổng quát về định nghĩa, phần mềm và phần cứng.

Chương 2: Trình bày tổng quan về bài toán giúp người đọc hiểu rõ hơn các chức năng của module trong mô hình, phân tích và đánh giá mô hình.

Chương 3: Trình bày thiết kế, chức năng của mô hình, cách thức cài đặt và sơ đồ kết nối từng phần của module trong mô hình.

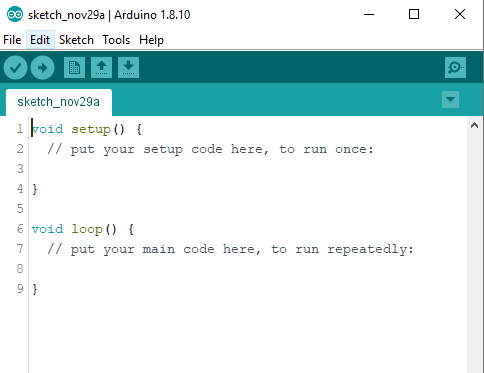
Chương 4: Mô tả quy trình kiểm thử và đánh giá kết quả của mô hình.

* Phần kết luận trình bày kết quả đạt được, thuận lợi, khó khăn và hướng phát triển của module trong mô hình.

# PHẦN NỘI DUNG

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

## 1.1 Arduino IDE

Arduino cung cấp đến môi trường lập trình tích hợp mã nguồn mở hỗ trợ người dùng viết code và tải nó lên bo mạch Arduino. Đây là môi trường đa nền tảng, hỗ trợ một loạt các bo mạch Arduino cùng rất nhiều tính năng độc đáo. Ứng dụng lập trình này có giao diện được sắp xếp hợp lý, phù hợp với cả những người dùng chuyên nghiệp lẫn không chuyên. Arduino có môi trường lập trình được viết bằng java, hiện đang được sử dụng cho các bo mạch Arduino và Genuido, được nhiều công ty trên thế giới sử dụng để lập trình cho các thiết bị của họ.  

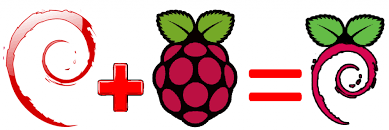
Hình 1. 1 Giao diện Arduino IDE

Arduino là môi trường phát triển tích hợp đa nền tảng, hỗ trợ cho một loạt các bo mạch Arduino như Arduino Uno, Nano, Mega, Esplora, Ethernet, Fio, Pro hay Pro Mini cũng như LilyPad Arduino. Phần mềm này cũng phù hợp cho những lập trình viên C và C++ là thay thế hoàn hảo cho các IDE khác.

Đặc điểm nổi bật: Chương trình đa nền tảng, quản lý board mạch, tạo Sketch đơn giản, tài liệu mô tả dự án, chia sẽ Sketch dễ dàng, thư viện đồ sộ…

## 1.2 Raspbian

Đây là hệ điều hành cơ bản, phổ biến nhất và do chính Raspberry Pi Foundation cung cấp. Là một [hệ điều hành máy tính](https://vi.wikipedia.org/wiki/H%E1%BB%87_%C4%91i%E1%BB%81u_h%C3%A0nh) dựa trên [Debian](https://vi.wikipedia.org/wiki/Debian) cho [Raspberry Pi](https://vi.wikipedia.org/wiki/Raspberry_Pi). Có một số phiên bản của Raspbian bao gồm Raspbian Stretch và Raspbian Jessie. Từ năm 2015, nó đã được Raspberry Pi Foundation chính thức cung cấp như là hệ điều hành chính cho gia đình máy tính bảng đơn Raspberry Pi. Raspbian được tạo ra bởi Mike Thompson và Peter Green như một dự án độc lập. Bản dựng ban đầu được hoàn thành vào tháng 6 năm 2012. Hệ điều hành vẫn đang được phát triển tích cực. Raspbian được tối ưu hóa cao cho các CPU [ARM](https://vi.wikipedia.org/wiki/C%E1%BA%A5u_tr%C3%BAc_ARM) hiệu suất thấp của dòng Raspberry Pi.

 Debian + Raspberry Pi = Raspbian

Hình 1. 2 Hệ điều hành Raspbian

## 1.3 Các gói cài đặt và thư viện

### 1.3.1 YOLOv3

Yolo là một trong những thuật toán nhận diện vật thể nhanh nhất thời điểm hiện tại, là một thư viện nhận dạng đối tượng trong ảnh hay video có tốc độ xử lý nhanh hơn rất nhiều thư viện hiện có, mức độ chính xác không phải tốt nhất nhưng ở mức tốt hợp lý trong trường hợp nhận dạng đối tượng trong video. Hiện tại Yolo có 3 phiên bản: Yolov1, Yolov2 và Yolov3.

Yolo v3 có tất cả những gì chúng ta cần để phát hiện đối tượng trong thời gian thực với việc phân loại chính xác và phân loại các đối tượng.

### 1.3.3 OpenCV

OpenCV là một thư viện mã nguồn mở, nó là miễn phí cho những ai bắt đầu tiếp cận với các học thuật. OpenCV được ứng dụng trong nhiều lĩnh vực như cho thị giác máy tính hay xử lý ảnh và máy học. Thư viện được lập trình trên các ngôn ngữ cấp cao: C++, C, Python, hay Java và hỗ trợ trên các nền tảng Window, Linux, Mac OS, iOS và Android. OpenCV đã được tạo ra tại Intel vào năm 1999 bởi Gary Bradsky, và ra mắt vào năm 2000. OpenCV có rất nhiều ứng dụng: nhận dạng ảnh, xử lý hình ảnh, phục hồi hình ảnh/video, thực tế ảo... Ở đề tài này thư viện OpenCV được chạy trên ngôn ngữ Python. OpenCV được dùng làm thư viện chính để xử lý hình ảnh đầu vào và nhận dạng ảnh.

Các tính năng của thư viện OpenCV:

* Đối với hình ảnh, có thể đọc và lưu hay ghi chúng.
* Về Video cũng tương tự như hình ảnh cũng có đọc và ghi.
* Xử lý hình ảnh có thể lọc nhiễu cho ảnh, hay chuyển đổi ảnh.
* Thực hiện nhận dạng đặc điểm của hình dạng trong ảnh.
* Phát hiện các đối tượng xác định được xác định trước như khuôn mặt, mắt,
* xe trong video hoặc hình ảnh.
* Phân tích video, ước lượng chuyển động của nó, trừ nền ra và theo dõi các đối tượng trong video.

### 1.3.4 TensorFlow

Là một thư viện phần mềm [mã nguồn mở](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m_ngu%E1%BB%93n_m%E1%BB%9F) dành cho [máy học](https://vi.wikipedia.org/wiki/M%C3%A1y_h%E1%BB%8Dc) trong nhiều loại hình tác vụ nhận thức và hiểu ngôn ngữ. TensorFlow là một thư viện được phát triển bởi Nhóm Google Brain để tăng tốc học máy và nghiên cứu mạng lưới thần kinh sâu. Nó được xây dựng để chạy trên nhiều CPU hoặc GPU và thậm chí cả hệ điều hành di động và nó có một số trình bao bọc trong một số ngôn ngữ như Python, C ++ hoặc Java.

Kiến trúc Tensorflow hoạt động trong ba phần:

* Tiền xử lý dữ liệu
* Xây dựng mô hình
* Đào tạo và ước tính mô hình

### 1.3.5 Keras

Keras là một thư viện được phát triển vào năm 2015 bởi François Chollet, là một kỹ sư nghiên cứu Deep learning tại Google. Nó là một Open source cho Neural network được viết bởi ngôn ngữ Python. Keras là một API bậc cao có thể sử dụng chung với các thư viện Deep learning nổi tiếng như Tensorflow (được phát triển bởi gg), CNTK(được phát triển bởi microsoft), Theano(người phát triển chính Yoshua Bengio). Keras có một số ưu điểm như :

* Dễ sử dụng, xây dựng model nhanh.
* Có thể run trên cả CPU và GPU
* Hỗ trợ xây dựng CNN, RNN và có thể kết hợp cả hai.

### 1.3.6 ImageAI

ImageAI là thư viện Computer Vision Python dễ sử dụng, cho phép các nhà phát triển dễ dàng tích hợp các tính năng Trí tuệ nhân tạo tiên tiến vào các ứng dụng và hệ thống mới và hiện có của họ. Nó được sử dụng bởi hàng ngàn nhà phát triển, sinh viên, nhà nghiên cứu, gia sư và chuyên gia trong các tổ chức doanh nghiệp trên toàn thế giới. Bạn sẽ tìm thấy các tính năng được hỗ trợ bên dưới, liên kết đến các tài liệu chính thức cũng như các bài viết trên ImageAI.

Tính năng và đặc điểm:

* Nhận dạng hình ảnh
* Phát hiện đối tượng
* Phát hiện và phân tích video
* Đào tạo, nhận dạng tùy chỉnh

## 1.4 Phần cứng thiết bị

### 1.4.1 Arduino Mega 2560

Arduino Mega 2560 là một vi điều khiển hoạt động dựa trên chip ATmega2560. Bao gồm:

* 54 chân digital (trong đó có 15 chân có thể được sủ dụng như những chân PWM là từ chân số 2 → 13 và chân 44 45 46).
* 6 ngắt ngoài: chân 2 (interrupt 0), chân 3 (interrupt 1), chân 18 (interrupt 5), chân 19 (interrupt 4), chân 20 (interrupt 3), and chân 21 (interrupt 2).
* 16 chân vào analog (từ A0 đến A15).
* 4 cổng Serial giao tiếp với phần cứng:

| **CỔNG SERIAL** | **CHÂN RX** | **CHÂN TX** |
| --- | --- | --- |
| Cổng 0 | 0 | 1 |
| Cổng 1 | 19 | 18 |
| Cổng 2 | 17 | 16 |
| Cổng 3 | 15 | 14 |

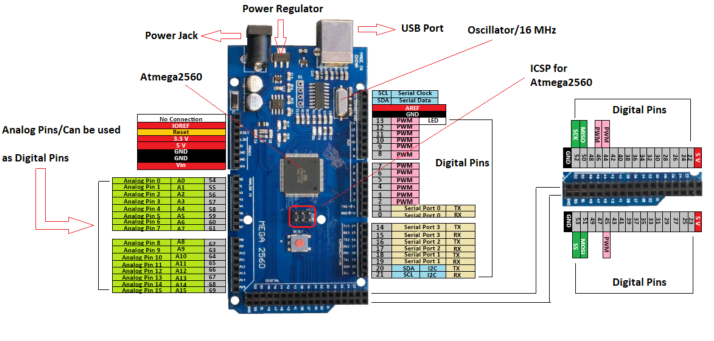
Bảng 1. 1 Cổng giao tiếp phần cứng

* 1 thạch anh với tần số dao động 16 MHz.
* 1 cổng kết nối USB.
* 1 jack cắm điện.
* 1 đầu ICSP.
* 1 nút reset.

Arduino Mega 2560 có thể sử dụng hầu hết các shiled dành cho các mạch Arduino Uno hay hoặc các mạch trước đây như Duemilanove hay Diecimila với cách cài đặt và nối chân tương tự như Arduino Uno.

**Thông số kỹ thuật**

* Chip xử lý: ATmega2560
* Điện áp hoạt động: 5V
* Điện áp vào (đề nghị): 7V-15V
* Điện áp vào (giới hạn): 6V-20V
* Cường độ dòng điện trên mỗi 3.3V pin: 50 mA
* Cường độ dòng điện trên mỗi I/O pin: 20 mA
* Flash Memory: 256 KB
* SRAM: 8 KB
* EEPROM: 4 KB
* Clock Speed: 16 MHz



Hình 1. 3 Mega 2560

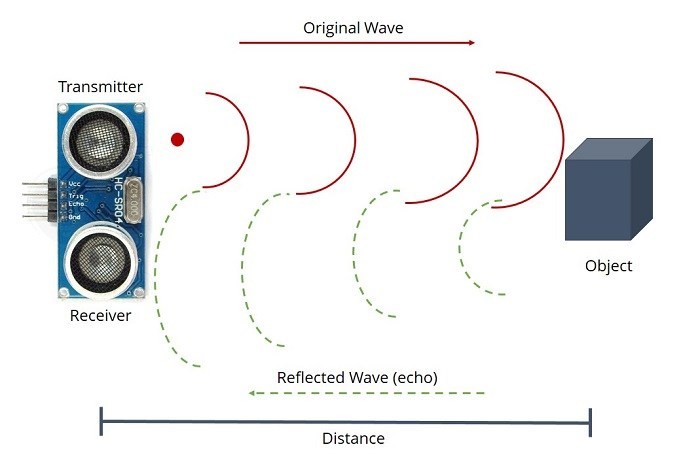
### 1.4.2 Cảm biến khoảng cách HC-SR04

Cảm biến khoảng cách HC-SR04 là cảm biến dùng để xác định khoảng cách trong phạm vi nhỏ bằng cách phát sóng siêu âm. Cảm biến với độ chính xác chính xác khá cao (với khoảng cách nhận biết nhỏ nhất 3mm) và độ ổn định cao trong quá trình sử dụng, đồng thời dễ dàng kết nối với các MCU (Arduino, DSP, AVR, PIC, ARM…)

Các ứng dụng thường thấy với cảm biến như: xe tránh vật cản, xác định khoảng cách của vật thể đến cảm biến, hay mô phỏng làm radar…

Nguyên lý hoạt động:

Để đo khoảng cách, ta sẽ phát một xung rất ngắn (5 microSeconds) từ chân Trig.Sau đó, cảm biến sẽ tạo ra một xung HIGH ở chân Echo cho đến khi nhận lại được sóng phản xạ ở pin này. Chiều rộng của xung sẽ bằng với thời gian sóng siêu âm được phát từ cảm biển và quay trở lại.

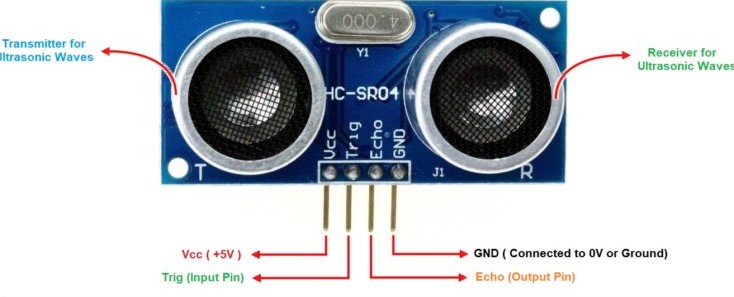


Hình 1. 4 Sơ đồ hoạt động của Cảm biến HC-SR04

**Công thức tính khoảng cách**

Khoảng cách = (thời gian \* vận tốc âm thanh (340 m/s))

Khoảng cách thực tế từ cảm biến đến vật cản chỉ là một nửa khoảng cách từ công thức trên ( vì trên là tính cả đi cả về).



Hình 1. 5 Cảm biến khoảng cách

**Thông số kỹ thuật**

* Điện áp: 5V DC
* Dòng hoạt động: < 2mA
* Mức cao: 5V
* Mức thấp: 0V
* Góc đo: 15 độ
* Khoảng cách: 2cm – 450cm (4.5m)
* Độ chính xác: 3mm
* Cảm biến HC-SR04 có 4 chân là: Vcc, Trig, Echo, GND.

|  |  |
| --- | --- |
| Vcc | 5V |
| Trig | Một chân Digital output |
| Echo | Một chân Digital input |
| GND | GND |

Bảng 1. 2 Sơ đồ nối chân giữa HC-SR04 với Arduino

### 1.4.3 Động cơ RC Servo 9G SG90

Động cơ RC Servo 9G là động phổ biến dùng trong các mô hình điều khiển nhỏ và đơn giản như cánh tay robot. Động cơ có tốc độ phản ứng nhanh, được tích hợp sẵn Driver điều khiển động cơ, dễ dàng điều khiển góc quay bằng phương pháp điều độ rộng xung PWM.

**Thông số kỹ thuật**

* Điện áp hoạt động: 4.8-5VDC
* Tốc độ: 0.12 sec/ 60 degrees (4.8VDC)
* Lực kéo: 1.6KG.CM
* Kích thước: 21x12x22mm
* Trọng lượng: 9g.
* Nhiệt độ: 0 ℃ --55 ℃



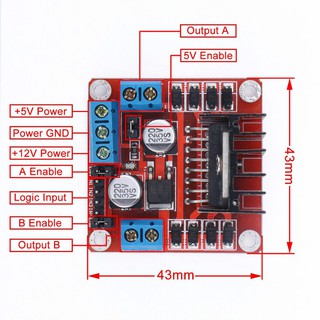
Hình 1. 6 Động cơ Servo

### 1.4.4 Module điều khiển động cơ L298N.

Có khả năng điều khiển 2 động cơ DC, dòng tối đa 2A mỗi động cơ, mạch tích hợp diod bảo vệ và IC nguồn 7805 giúp cấp nguồn 5VD cho các module khác (chỉ sử dụng 5V này nếu cập nguồn <12V).

**Thông số kỹ thuật**

* IC chính: L298 – Dual Full Bridge Driver
* Điện áp đầu vào: 5~30VDC
* Công suất tối đa: 25W 1 cầu (lưu ý công suất bằng dòng điện nhân điện áp nên áp cấp vào càng cao, dòng càng nhỏ, công suất có định 25W).
* Dòng tối đa cho mỗi cầu H là: 2A
* Mức điện áp logic: Low -0.3V~1.5V, High: 2.3V~Vss
* Kích thước: 43x43x27mm



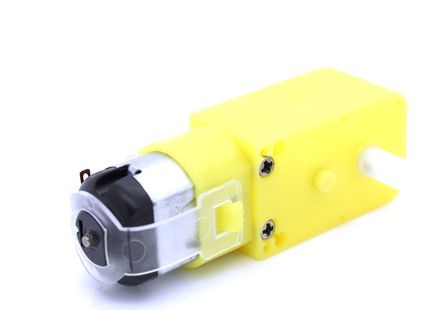
Hình 1. 7 Module L298N

### 1.4.5 Động cơ giảm tốc

Động cơ giảm tốc (Smart Car Robot Plastic Tire Wheel with DC 3-6v Gear Motor for arduino) là loại động cơ DC phổ thông được sử dụng nhiều nhất nhờ sự tiện dụng, giá thành rẻ. Thiết kế rất phù hợp cho các ứng dụng robot, kết hợp với các loại bánh xe thông dụng.

**Thông số kỹ thuật:**

* Điện áp hoạt động: 3 đến ­6VDC.
* Dòng điện tiêu thụ: 110 đến 140mA.
* Tỷ số truyền: 1:48, 1:120
* Số vòng/1phút:
* 125 vòng/ 1 phút tại 3VDC.
* 208 vòng/ 1 phút tại 5VDC
* Moment: 0.8KG.CM



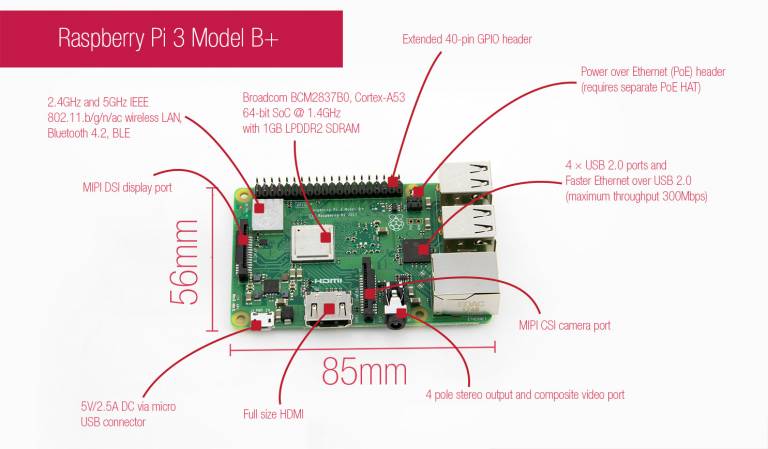
Hình 1. 8 Động cơ giảm tốc

### 1.4.6 Raspberry Pi 3 B+

Raspberry Pi 3 Model B+ là sản phẩm mới trong gia đình Raspberry Pi, nổi bật với chip 4 nhân 64-bit có tốc độ 1.4GHz – nhanh nhất từ trước đến nay. Phiên bản này hỗ trợ Wifi Dual-band 2.4GHz và 5GHz, Bluetooth 4.2/Bluetooth Low Energy, cổng Ethernet tốc độ cao (300Mbps) và Power over Ethernet (PoE) thông qua PoE HAT.

**Thông số kỹ thuật**

* Vi xử lý: Broadcom BCM2837B0, quad-core A53 (ARMv8) 64-bit SoC @1.4GHz
* RAM: 1GB LPDDR2 SDRAM
* Kết nối: 2.4GHz and 5GHz IEEE 802.11 b/g/n/ac wireless LAN, Bluetooth 4.2, BLE, Gigabit Ethernet over USB 2.0 (Tối đa 300Mbps).
* Hỗ trợ: 40-pin GPIO, 4 cổng USB2.0
* Video và âm thanh: 1 cổng full-sized HDMI, Cổng MIPI DSI Display, cổng MIPI CSI Camera, cổng stereo output và composite video 4 chân.
* Multimedia: H.264, MPEG-4 decode (1080p30), H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
* Lưu trữ: MicroSD
* Điện áp hoạt động: 5V/2.5A DC cổng microUSB, 5V DC trên chân GPIO, mPower over Ethernet (PoE) (yêu cầu thêm PoE HAT).



Hình 1. 9 Raspberry Pi 3 B+

### 1.4.7 Camera

Camera cho Raspberry Pi có thể được sử dụng để quay video độ phân giải cao, cũng như hình ảnh tĩnh. Nó dễ sử dụng cho người mới bắt đầu, nhưng cũng có thể cung cấp cho người dùng nâng cao.

Module này có camera lấy nét cố định 5MP hỗ trợ các chế độ video 1080p30, 720p60 và VGA90, cũng như chụp ảnh tĩnh. Nó gắn thông qua cáp FFC 15cm vào cổng CSI trên Raspberry Pi. Nó có thể được truy cập thông qua API MMAL, V4L và có rất nhiều thư viện của bên thứ ba được xây dựng cho nó, bao gồm thư viện Python Picamera.

**Thông số kỹ thuật**

* Độ phân giải: 2592 x 1944 (5 megapixel)
* Ống kính: f = 3.75mm, f/2.8
* Góc nhìn: 65 độ
* Phạm vi lấy nét: 0.69m đến vô cực (ở mức 1.38m)
* Hỗ trợ: 1080p @ 30 khung hình/giây với codec H.264 (AVC), 720p @ 60 fps và 640x480p @ 60/90 khung hình/giây
* Giao diện: CSI
* Kích thước: 25mm x 25mm x 10mm (0.98” x 0.98” x 0.39”)
* Trọng lượng: Khoảng 2.8g



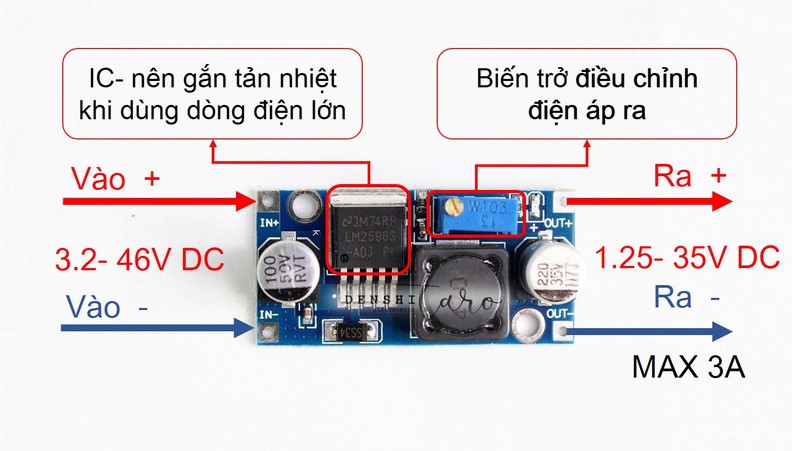
Hình 1. 10 Camera

### 1.4.8 Mạch giảm áp DC LM2596

Mạch giảm áp DC nhỏ gọn có khả năng giảm áp từ 30V xuống 1.5V mà vẫn đạt hiệu suất cao (92%) . Thích hợp cho các ứng dụng chia nguồn, hạ áp, cấp cho các thiết bị như camera, motor , robot,...

**Thông số kỹ thuật:**

* Điện áp đầu vào: Từ 3V đến 30V.
* Điện áp đầu ra: Điều chỉnh được trong khoảng 1.5V đến 30V.
* Dòng đáp ứng tối đa là 3A.
* Hiệu suất : 92% và công suất : 15W
* Kích thước: 45 (dài) \* 20 (rộng) \* 14 (cao) mm



Hình 1. 11 Mạch giảm áp LM2596

### 1.4.9 Pin Lipo

Pin Lipo 2200mAh 11.1V 25C 3S (loại pin sạc) có dung lượng khá phù hợp sử dụng cho các mô hình robot, xa điều khiển, máy bay điều khiển, máy bay bốn cánh, thuyền điều khiển...

**Thông số kỹ thuật:**

* Loại PIN: Lithium Polimer
* Dung lượng: 2200 mAh
* Điện áp 11.1V
* Cell: 3 Cell

Hình 1. 12 Pin Lipo

### 1.4.10 Khung xe robot 4 bánh

Khung xe sử dụng bốn động cơ giảm tốc DC chuyển hướng linh hoạt dựa trên xử lý tốc độ của động cơ, Khung xe có kích thước khá lớn, rất dễ để mở rộng, lắp đặt các module, mạch điện hỗ trợ.

Ứng dụng của xe robot bốn bánh : thích hợp cho những ứng dụng làm xe tự hành, tránh vật cản, lắp ghép với cánh cánh tay robot để điều khiển.

**Thông số kỹ thuật:**

* Bốn bánh xe robot bằng nhựa, lốp chất liệu cao su
* Một bộ ốc vít và ke để gắn động cơ vào khung xe
* Một bộ cọc đồng để ghép 2 tấm đế mica
* ****Phần khung sàn robot được làm bằng một miếng mica có độ dày 3mm, chiều rộng 15cm, có chiều dài 17cm.

Hình 1. 13 Khung xe bốn bánh

# Tổng kết chương 1.

Trong chương 1 đã trình bày sơ lược về lý thuyết, phần mềm và phần cứng, các thông số kỹ thuật và nguyên lý hoạt động của các thiết bị, các board mạch Mega và Raspberry. Trình bày cách làm việc cũng như thư viện cần cho viêc lập trình mô hình. Tiếp theo, trong chương 2 sẽ trình bày nội dung mô tả bài toán của đề tài dựa trên cơ sở lý thuyết của chương 1.

# CHƯƠNG 2: MÔ TẢ BÀI TOÁN

## 2.1 Mô tả bài toán

### 2.1.1 Mô tả tổng quan

* **Bối cảnh sản phẩm**

Xe tự tránh vật cản kết hợp nhận dạng lửa sẽ thay thế con người đi tuần tra tại những nguy cơ xảy ra cháy nổ cao như nhà máy, xưởng sản xuất, nhà kho… Trong nghiên cứu này tập trung vào việc vận hành mô hình thử nghiệm xe trong phạm vi văn phòng, nơi địa hình bằng phẳng.

* **Mô hình xử lý**

Hình 2. 1 Khối xử lý

**Mega**

( Xử lý trung tâm)

**Raspberry**

( Xử lý ảnh)

**Động cơ**

( Thực thi)

**Camera**

( Thu hình ảnh)

**Cảm biến**

( Đọc cảm biến)

* **Phần cứng yêu cầu các thiết bị:**
* Board Arduino Mega 2560
* Cảm biến khoảng cách HC-SR04
* Động cơ Servo S9 SG90
* Module điều khiển động cơ L298N
* Bốn motor động cơ giảm tốc
* Raspberry Pi 3 B+
* Raspberry Pi Camera Rev 1.3
* Breaboard và dây cắm
* Khung và bốn bánh xe
* Pin Lipo
* **Phần mềm yêu cầu**
* Arduino yêu cầu phiên bản mới để cập nhật thư viện mới.
* Raspberry Pi 3 B+ cài hệ điều hành Raspbian
* Cài đặt các gói thư viện Tensorflow ,  Keras , OpenCV vàImageAI.
* VNC Viewer
* **Các ràng buộc thực thi và thực tế**
* Cần cung cấp đủ nguồn 12V cho L298N điều khiển động cơ, 5V cho Mega và Raspberry, đặc biệt trên Raspberry cần dòng điện tối thiểu từ 1A trở lên.
* Cắm dây thông qua Breadboard phải đảm bảo chính xác về dây nguồn cũng như dây truyền dữ liệu, quan trọng cần chú ý tránh sai dây nguồn vì có thể làm hỏng board.
* Kết nối Mega vào máy tính bằng cổng USB phải chọn đúng COM và board mạch để lập trình.
* Kết nối giữa Mega và Raspberry phải chọn đúng COM và tốc độ baudrate.

### 2.1.2 Yêu cầu giao tiếp

* **Giao tiếp phần cứng**
* Mega và Raspberry giao tiếp với nhau bằng dây cáp USB
* Các cảm biến và động cơ kết nối trực tiếp với Mega thông qua các chân Analog, Digital.
* **Giao tiếp phần mềm**
* Trên hệ điều hành Raspbian, sử dụng hỗ trợ phần mềm Arduino IDE 1.8.10 để lập trình và upload chương trình vào board Mega 2560.
* Máy tính truy cập từ xa vào Raspberry thông qua công nghệ VNC Viewer

## 2.2 Phân tích

### 2.2.1 Các tính năng của mô hình

* Xe sẽ hoạt động tự động sau khi bấm nguồn
* Nhận dạng vật cản trên đường di chuyển
* Điều hướng xe tránh vật cản
* Nhận dạng lửa từ xa
* Điều hướng xe đến vị trí cháy

### 2.2.2 Các yêu cầu phi chức năng

* Khả năng sửa lỗi: việc upload chương trình trên Mega là khá dễ dàng nên khi gặp sự cố có thể dễ dàng khắc phục.
* Dễ sử dụng: mô hình không cần nhiều thao tác.
* Thiết bị dễ tìm, dễ thay thế.
* Cấp nguồn: cần cấp đủ nguồn, có tính di động, dễ sạc.

## 2.3 Đánh giá

### 2.3.1 Đánh giá phần cứng

|  |  |  |
| --- | --- | --- |
| **Tên thiết bị** | **Ưu điểm** | **Nhược điểm** |
| Arduino  Mega 2560 | - Có thể điều khiển nhiều loại động cơ  - Xử lý song song nhiều luồng dữ liệu số cũng như dữ liệu tương tự  - Số lượng chân cắm và dung lượng flash lớn có thể giải quyết vấn đề mô hình lớn | - Là một vi điều khiển  - Không xử lý được dữ liệu ảnh |
| Máy tính nhỏ  Raspberry Pi 3 B+ | - Là một máy tính thu nhỏ  - Có khả năng xử lý dữ liệu lớn một cách nhanh chóng | - Giá thành cao  - Không thể xuất và đọc analog  - Giới hạn việc đọc độ rộng xung, băm xung  - Chân GPIO chỉ xuất nhận tối đa 3.3V nên không có cảm biến trên Arduino |
| Cảm biến khoảng cách  HC-SR04 | - Có độ chính xác khá cao và ổn định trong quá trình sử dụng  - Dễ dàng kết nối với Arduino | - Thường chỉ áp dụng cho mô hình |
| Camera | - Nhận dạng được lửa từ xa | - Không thu được hình ảnh ở những góc khuất và những vị trí ngoài khung ảnh |
| Động cơ Servo | - Có tốc độ phản ứng nhanh  - Có tích hợp sẵn Driver điều khiển động cơ bên trong | - Chỉ làm các mô hình nhỏ  - Lực kéo không lớn |
| Động cơ giảm tốc | -Giá thành rẻ  - Dễ sử dụng | - Chỉ sử dụng cho các mô hình nhỏ |
| Module điều khiển động cơ L98N | - Có thể điều chỉnh tốc độ và chiều quay của động cơ.  - | - Không thể cung cấp tối đa nguồn điện cho động cơ |
| Mạch giảm áp LM2596 | - Có thể thay đổi hiệu điện thế ở nhiều mức khác nhau  - Chi phí thấp,độ ổn định cao | - Mạch không cách ly  - Phạm vi ứng dụng hạn chế |
| Pin Lipo | - Ít bị rò rỉ, hoạt động mạnh mẽ  - Trọng lượng nhẹ, bền  - Kích thước phù hợp với mô hình | - Giá thành khá cao  - Không tự động sạc |

Bảng 2. 1 Ưu và nhược điểm phần cứng

### 2.3.2 Đánh giá phần mềm

Các phần mềm được chọn là phiên bản mới và có bổ sung đầy đủ các thư viện, tính năng cần thiết:

* Hỗ trợ việc viết chương trình và upload chương trình lên các board một cách tốt nhất, ổn định nhất.
* Hỗ trợ việc truy cập từ xa vào Raspberry một cách thuận tiện dễ dàng.
* Hỗ trợ thiết kế các sơ đồ mạch điện tử một cách nhanh chóng và hiệu quả.

### 2.3.3 Đánh giá chung

Mô hình đang sử dụng những giải pháp tốt nhất có thể và những thiết bị phù hợp với khả năng và đáp ứng mục tiêu đặt ra cho mô hình. Tuy nhiên, mô hình vẫn chưa đạt hiệu quả tối ưu cần phát triển thêm ở một vài khía cạnh.

# Tổng kết chương 2.

Trong chương 2 đã trình bày sơ lược về những mô tả chi tiết cho bài toán, đưa ra các phân tích và đánh giá tổng thể về yêu cầu bài toán. Tiếp theo, trong chương 3 sẽ trình bày nội dung thiết kế và cài đặt giải pháp của đề tài trên từng phân hệ dựa trên mô tả bài toán từ chương 2.

# CHƯƠNG 3: THIẾT KẾ VÀ CÀI ĐẶT GIẢI PHÁP

🙢🕮🙠

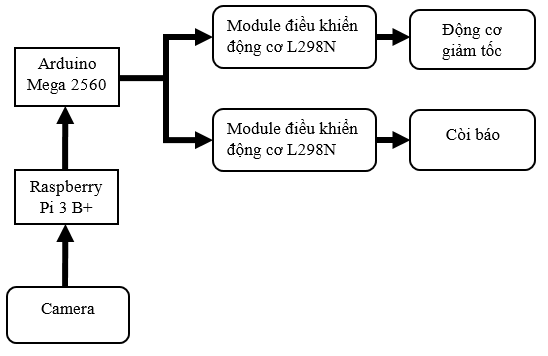
## 3.1 Thiết kế module

### 3.1.1 Module tránh vật cản

Hình 3. 1 Sơ đồ hoạt động của module tránh vật cản

* **Hoạt động của module:**
* Cảm biến khoảng cách HC-SR04 sử dụng sóng siêu âm ở tần số 40000Hz để xác định khoảng cách đến một vật thể.
* Cảm biến khoảng cách HC-SR04 gửi tín hiệu sang Arduino Mega 2560 để xử lý tín hiệu nhận được là khoảng cách đến vật cản ( đơn vị cm).
* Arduino Mega 2560 sẽ gửi tín hiệu điều khiển đến module điều khiển động cơ L298N và động cơ Servo 9G SG90.
* Động cơ Servo có nhiệm vụ xoay sang trái, phải, phía trước để tăng góc nhìn cho Cảm biến HC-SR04.
* Module điều khiển động cơ L298N sẽ điều khiển bốn động cơ tăng tốc, giảm tốc rẽ trái, rẽ phải, tiến, lùi để tránh vật cản.

### 3.1.2 Module phát hiện lửa bằng camera



Hình 3. 2 Sơ đồ hoạt động của module nhận dạng lửa bằng camera

**Hoạt động của module**

* Camera thu nhận hình ảnh trên lộ trình di chuyển.
* Truyền dữ liệu ảnh từ Camera đến Raspberry Pi 3 B+ để xử lý và nhận dạng lửa.
* Raspberry Pi 3 B+ thực hiện gửi tín hiệu sang Arduino Mega 2560 để xử lý tín hiệu nhận.
* Arduino Mega 2560 sẽ gửi tín hiệu xử lý đến hai module điều khiển động cơ L298N.
* Module điều khiển động cơ L298N (điều khiển bốn động cơ giảm tốc) sẽ điều khiển bốn động cơ tăng tốc, giảm tốc, rẽ trái, rẽ phải, tiến, lùi để tiến đến vị trí có lửa.
* Module điều khiển động cơ L298N (điều khiển còi báo) sẽ điều khiển bật còi báo.

## 3.2 Cài đặt phần mềm

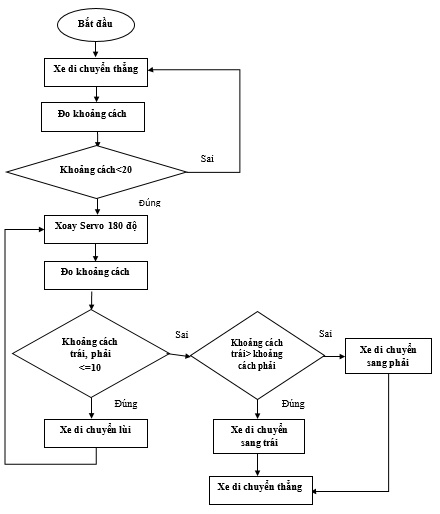
Để xây dựng mô hình cần cài đặt:

* Hệ điều hành Raspbian cài đặt trên Raspberry Pi 3 B+.
* Cài đặt VNC Viewer để truy cập từ xa vào Raspberry.
* Trình soạn thảo văn bản Arduino IDE 1.8.10.
* Sử dụng ngôn ngữ lập trình Python để lập trình trên Raspberry và ngôn ngữ C để lập trình cho Mega.
* Sử dụng tập dữ liệu nhận dạng hình ảnh trên Yolo
* Thư viện Rpi.GPIO có sẵn trên hệ điều hành Raspbian.
* Gói cài đặt Tensorflow , Keras , OpenCV vàImageAI.

*(Chi tiết về nội dung cài đặt xem ở phần phụ lục).*

## 3.3 Lưu đồ thuật toán

### 3.3.1 Tránh vật cản

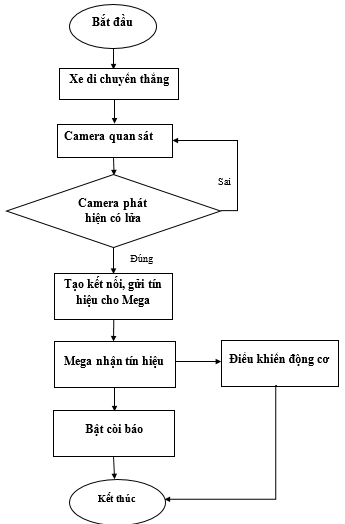


Hình 3. 3 Thuật toán tránh vật cản

Hình 3.3 cho thấy biểu đồ luồng hoạt động của xe tránh chướng ngại vật sử dụng cảm biến khoảng cách HC-SR04, động cơ Servo 9G SG90, module điều khiển động cơ L298N và bốn động cơ.

* Bắt đầu bật nguồn cho mô hình.
* Cảm biến HC-SR04 sẽ kiểm tra khoảng cách đến các vật cản. Nếu khoảng cách đo được nhỏ hơn 20cm (gặp vật cản).
* Xe sẽ ngừng di chuyển.
* Cảm biến HC-SR04 được gắn trên động cơ Servo sẽ quay 180 độ để đo khoảng cách trái và phải của xe.
* Nếu khoảng cách trái và khoảng cách phải đều nhỏ hơn 10cm thì xe sẽ tự động đi lùi lại và tiếp tục đo khoảng cách trái, phải.
* Nếu sai thì sẽ tiếp tục so sánh khoảng cách trái và khoảng cách phải nếu khoảng cách trái lớn hơn khoảng cách phải thì xe se rẽ trái ngược lại sẽ rẽ phải.
* Nếu như trên đường đi không gặp bất kỳ vật cản nào thì giản xe chỉ cần tiến về phía trước hoàn thành lộ trình của nó.

### 3.3.2 Phát hiện lửa bằng camera trên Raspberry Pi 3 B+



Đúng

Hình 3. 4 Thuật toán phát hiện lửa bằng Raspberry

Trên hình 3.4 cho thấy biểu đồ luồng hoạt động của xe nhận dạng lửa bằng camera xử lý trên Raspberry Pi 3 B+.

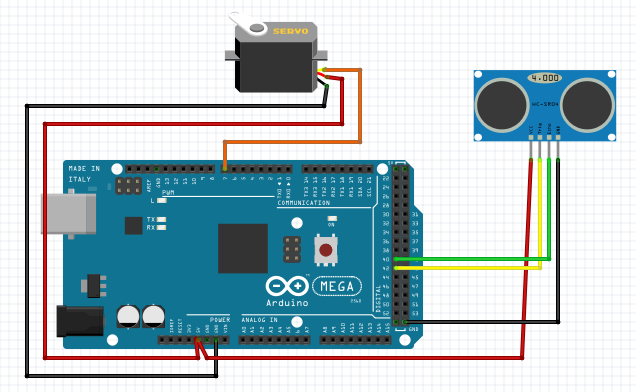
* Raspberry Pi 3 B+ đã được huấn luyện để có thể nhận dạng hình ảnh lửa trên camera sau đó xử lý.
* Camera quan sát thu thập dữ liệu ảnh.
* Nếu camera thu được hình ảnh Raspberry Pi 3 B+ nhận dạng lửa
* Raspberry Pi 3 B+ sẽ tạo một kết nối thông qua cổng truyền thông UART với Mega và gửi tín hiệu có lửa cho Mega.
* Mega 2560 nhận tín hiệu có lửa.
* Mega 2560 điều khiển bật còi báo, điều khiển động cơ.

Ngược lại, trong trường hợp Camera không phát hiện được lửa thì các hoạt động trên Mega vẫn diễn ra bình thường.

## 3.4 Sơ đồ kết nối và cài đặt mô hình

### 3.4.1 Kết nối trên Arduino Mega 2560

* **Giải pháp phát hiện vật cản**



Hình 3. 4 Kết nối cảm biến khoảng cách

|  |  |  |
| --- | --- | --- |
| **Mega** | **Cảm biến khoảng cách** | **Servo khoảng cách** |
| 5V | 5V | 5V |
| GND | GND | GND |
| 7 |  | servoKC |
| 40 | Echo |  |
| 42 | Trig |  |

Bảng 3. 1 Sơ đồ chân cắm của Mega với Cảm biến khoảng cách và Servo

* Đầu tiên, khai báo thư viện với động cơ Servo (với cảm biến HC-SR04 không khai báo thư viện):

#include <Servo.h>

* Tạo các biến cho pin kích hoạt chân Trig và Echo. Sau đó, chân Trig kích hoạt được kết nối với chân số 42 và chân echo được kết nối với chân số 40:

int trig = 42; // chân Trig của HC-SR04.

int echo = 40; // chân Echo của HC-SR04.

Tạo đối tượng để điều khiển Servo và kết nối với chân cắm số 7:

Servo ServoKC;

int servo\_KC = 7;

* Trong hàm setup(), khởi tạo cổng nối tiếp ở tốc độ baud là 9600 và đặt chân Trig làm đầu ra và chân echo làm đầu vào:

pinMode(echo, INPUT); // chân Echo sẽ nhận tín hiệu

pinMode(trig, OUTPUT); // chân Trig sẽ phát tín hiệu

Thiết lập cho Servo:

ServoKC.attach(servo\_KC);

* Trong hàm doKhoangCach(), kích hoạt cảm biến bằng cách gửi xung HIGH 10 microgiây. Nhưng, trước đó, cung cấp một xung LOW ngắn để đảm bảo bạn sẽ có được xung HIGH sạch:

digitalWrite(trig, LOW);

delayMicroseconds(5);

digitalWrite(trig, HIGH);

delayMicroseconds(10);

digitalWrite(trig, LOW);

Sau đó, có thể đọc tín hiệu từ cảm biến - một xung HIGH có thời lượng là thời gian tính bằng micrô giây từ khi gửi tín hiệu đến việc nhận tins hiệu của nó đến một đối tượng.

thoigian = pulseIn(echo, HIGH);

Cuối cùng, tính khoảng cách đến vật cản bằng công thức sau:

khoangcach = thoigian \* 0.034 / 2;

* Trong hàm quayCamBienTrai(), hàm quayCamBienPhai() bằng cách sử dụng hàm write (), chúng ta chỉ cần đặt vị trí của servo từ 0 đến 180 độ. Quay sang trái ở vị trí 180 độ và quay sang phải ở vị trí 0 độ.

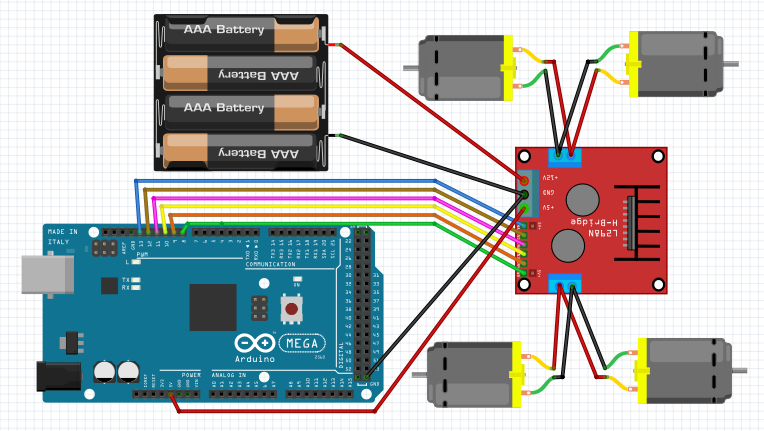
ServoKC.write([góc\_quay]);

Khi Servo quay trái, phải đồng thời Cảm biến HC-SR04 sẽ đo khoảng cách để xác định hướng di chuyển cho xe:

doKhoangCach();

Sơ đồ kết nối trên là Arduino Mega 2560 nhận tín hiệu đo khoảng cách từ cảm biến HC-SR04 và điều khiển động cơ Servo quay. Cảm biến HC-SR04 được thiết kế gắn trên động cơ servo với mục đích tăng góc nhìn cho cảm biến.

* **Giải pháp điều khiển động cơ**

****

Hình 3. 5 Kết nối động cơ bánh xe

|  |  |  |
| --- | --- | --- |
| **Mega** | **L298N1** | **Động cơ giảm tốc** |
| 5V | 5V |  |
| GND | GND |  |
| 8 | enB |  |
| 9 | IN4 | OUT4 |
| 10 | IN3 | OUT3 |
| 11 | IN2 | OUT2 |
| 12 | IN1 | OUT1 |
| 13 | enA |  |

Bảng 3. 2 Sơ đồ chân cắm Mega với L298N và bốn động cơ giảm tốc

* Xác định chân cho đầu vào của module trình điều khiển động cơ L298N.L298N có bốn chân đầu vào dữ liệu được sử dụng để điều khiển hướng và hai chân điều khiển tốc độ của động cơ được kết nối với nó.

int tien1 = 11;

int tien2 = 12;

int lui1 = 9;

int lui2 = 10;

int enA = 13;

int enB = 8;

* Trong hàm setup () , xác định hướng dữ liệu của các chân GPIO được sử dụng.

pinMode(tien1, OUTPUT);

pinMode(tien2, OUTPUT);

pinMode(lui1, OUTPUT);

pinMode(lui2, OUTPUT);

* Xây dựng hàm di chuyển cho động cơ gồm có 5 tình huống: tiến, lùi, trái, phải, dừng.

//hàm tien()

digitalWrite(tien1, LOW);

digitalWrite(tien2, HIGH);

digitalWrite(lui1, LOW);

digitalWrite(lui2, HIGH);

analogWrite(enA, 100);

analogWrite(enB, 100);

//hàm lùi()

digitalWrite(tien1, HIGH);

digitalWrite(tien2, LOW);

digitalWrite(lui1, HIGH);

digitalWrite(lui2, LOW);

analogWrite(enA, 80);

analogWrite(enB, 80);

//ham dung()

digitalWrite(tien1, LOW);

digitalWrite(tien2, LOW);

digitalWrite(lui1, LOW);

digitalWrite(lui2, LOW);

//hàm reTrai()

resetdongco();

digitalWrite(tien1, HIGH);// chân này không có PWM

digitalWrite(tien2, LOW);

digitalWrite(lui1, LOW);

digitalWrite(lui2, HIGH);

analogWrite(enA, 250);

analogWrite(enB, 190);

//hàm rePhai()

resetdongco();

digitalWrite(Lui1, HIGH);

digitalWrite(Lui2, LOW);

digitalWrite(Tien1, LOW);

digitalWrite(Tien2, HIGH);

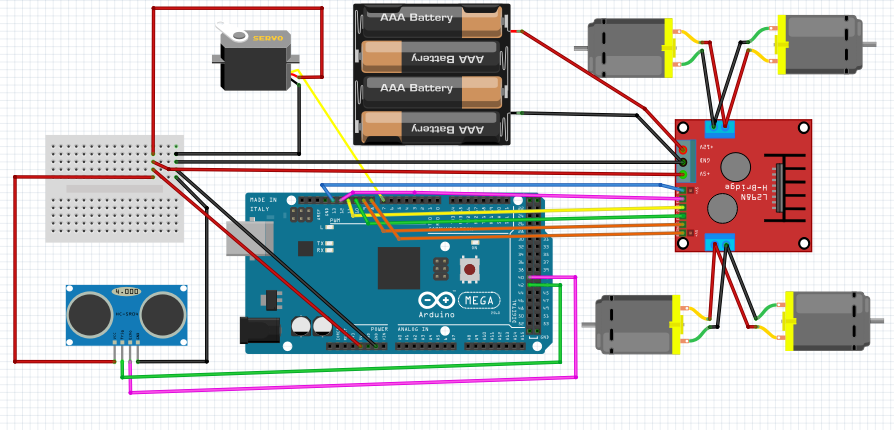
analogWrite(enB, 250);

analogWrite(enA, 190);

Sơ đồ kết nối trên dùng Arduino Mega 2560 kết nối với module điều khiển động cơ L298N để điều khiển bốn động cơ giảm tốc.

Nó có thể thay đổi tốc độ nhanh chậm của các động cơ, điều khiển chiều quay của nó để giúp xe dừng, đi thẳng, lùi, rẻ trái, rẻ phải theo mục đích yêu cầu.

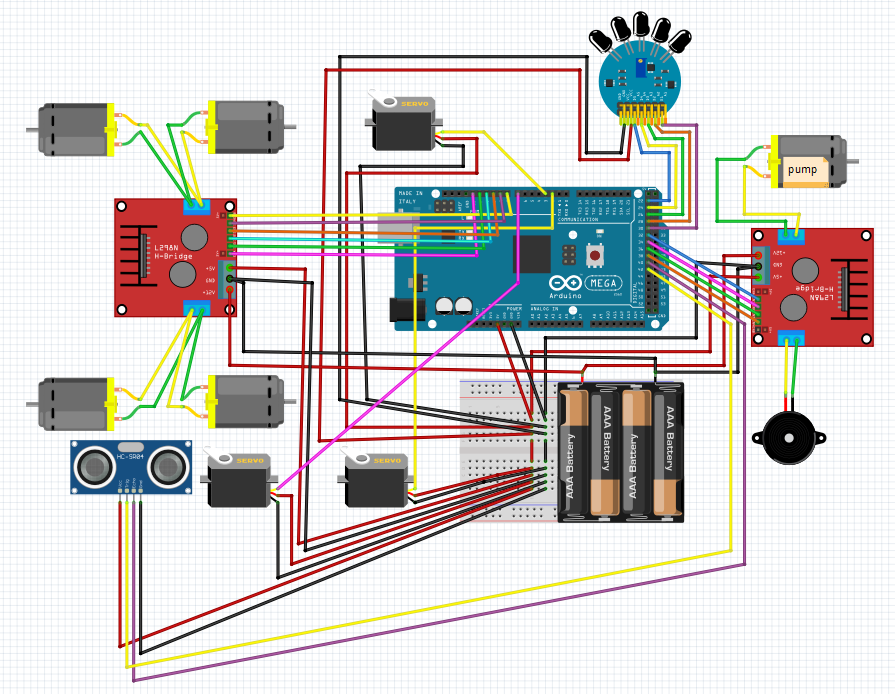
Cách thức thực hiện các lệnh điều khiển:

* Đi thẳng: Cả bốn động cơ đều tiến
* Đi lùi: Cả bốn động cơ đều lùi
* Rẽ phải: hai động cơ bên phải lùi, hai động cơ bên trái tiến
* Rẽ trái: hai động cơ bên trái lùi, hai động cơ bên phải tiến
* Dừng: Dừng bốn động cơ
* **Giải pháp tránh vật cản**

Hình 3. 6 Sơ đồ tránh vật cản

Xe sẽ hoạt động di chuyển về phía trước, cảm biến HC-SR04 sẽ làm nhiệm vụ kiểm tra xuyên suốt khoảng cách từ xe đến vật thể, đến phạm vi trong vòng 20cm thì sẽ dừng xe lại. Động cơ Servo quay trái để đo khoảng cách, lưu lại giá trị đã đo, tiếp tục quay sang phải để đo khoảng cách và lưu lại giá trị đã đo.

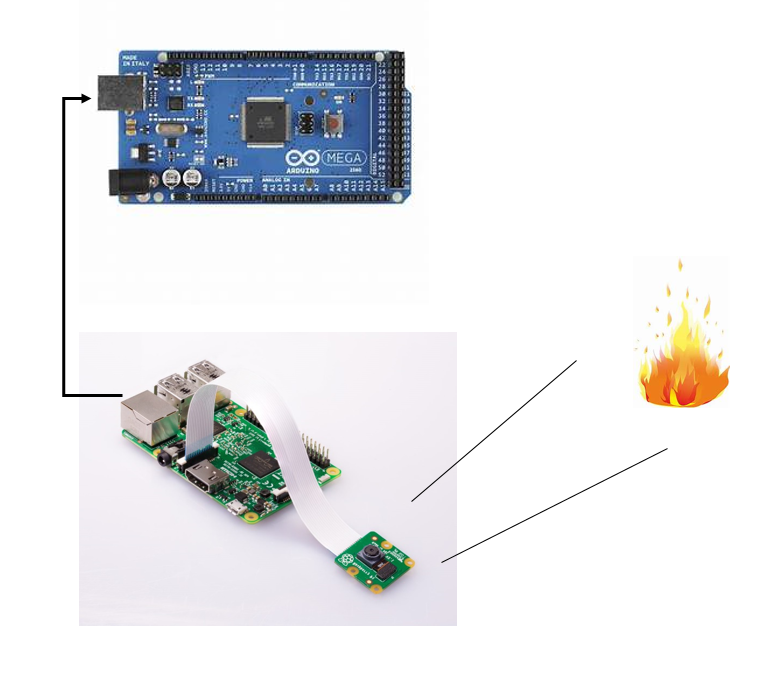
* Trường hợp 1: Nếu cả hai khoảng cách có giá trị nhỏ hơn hoặc bằng 10 thì thực hiện lui về sau.
* Trường hợp 2: Ngược lại cả hai khoảng cách có giá trị lớn hơn 10, nếu khoảng cách bên phải lớn hơn trái thì thực hiện rẽ phải.
* Trường hợp 2: Ngược lại cả hai khoảng cách có giá trị lớn hơn 10, nếu khoảng cách bên trái lớn hơn phải thì thực hiện rẽ trái.
* **Giải pháp trên cả hai phân hệ**

****

Hình 3. 7 Mô hình kết nối tổng quát trên Mega

Sơ đồ trên là kết nối phần cứng trên Mega 2560 để điểu khiển xe tránh vật cản và nhận biết lửa thông qua cảm biến đồng thời bật còi báo động và bơm nước khi có đám cháy xảy ra.

### 3.4.2 Kết nối trên Raspberry



Hình 3. 8 Phát hiện lửa trên Raspberry gửi sang Mega

* **Giải pháp nhận dạng lửa thông qua camera**

Để nhận dạng hình ảnh bằng thư viện OpenCV, Tensorflow, và Keras và ImageAI. Sau đây là các bước thực hiện:

* **Bước 1:** Chọn đối tượng cần nhận dạng

Trong đề tài này đối tượng nhận dạng cụ thể là lửa. Sử dụng tập dữ liệu ảnh trên Yolo. Tập dữ liệu được sử dụng gồm 502 hình ảnh được chia thành 412 hình ảnh để đào tạo và 90 hình ảnh để xác nhận.

* **Bước 2:** Cài đặt các gói thư viện cần thiết.
* OpenCV

pip3 install opencv-python

* Tensorflow

pip3 install –upgrade tensorflow

* Keras

Pip3 install keras

* ImageAI

pip3 install imageai --upgrade

* **Bước 3:** Dùng các pretrained YOLOv3 mô hình có sẵn.
* **Bước 4:** Chạy chương trình: python3 fire.py

**Mục đích nhận dạng hình ảnh qua camera:**

Nhằm giải quyết vấn đề phát hiện lửa từ xa và hạn chế sai xót do cảm biến lửa chưa phân biệt được lửa và ánh sáng mặt trời, nên sự kết hợp với nhận dạng ảnh qua camera sẽ là giải pháp giúp cho việc nhận dạng và phát hiện lửa trở nên chính xác và đáng tin cậy hơn.

* **Giải pháp kết nối Mega và Raspberry**
* Phần khai báo:

Thư viện time, serial cho Raspberry

* Phần cài đặt:

Serial.begin(9600); // Thiết lập serial trên Mega

serial.Serial(); //Thiết lập serial trên Raspberry

* Phần thực thi:

//trên Raspberry

int write([dữ\_liệu]); //Gửi dữ liệu

//trên Mega

If( Serial.available()>0){

Serial.readString();//thực hiện nhận tín hiệu

}

**Mục đích của việc kết nối:**

Raspberry Pi 3 B+ có khả năng như một chiếc máy tính, có wifi, bluetooth, có thể gửi và nhận mail… nhưng còn có một số hạn chế như không thể xuất và đọc Analog, việc đọc độ rộng xung cũng bị giới hạn, băm xung khó khăn,…Và chân GPIO của Raspberry Pi chỉ xuất nhận tối đa 3.3V nên khi dùng các cảm biến 5V trở lên sẽ dễ gây hư hỏng mạch. Thế nên, để phát triển mạnh mẽ hơn về các dự án IoT đồng thời tránh hư hỏng cho mạch Raspberry Pi, nên cần phải kết hợp giữa Arduino và Raspberry Pi. Để có thể làm điều đó, có rất nhiều cách như giao tiếp I2C, giao tiếp bằng UART, giao tiếp thông qua Bluetooth, thông qua Wifi… Ở nghiên cứu này thực hiện cách giao tiếp đơn giản nhất là qua UART – Serial (cổng USB).

# Tổng kết chương 3.

Trong chương 3 đã trình bày về cách thức thiết kế và cài đặt giải pháp để thực hiện bài toán trên phân hệ: “Phát hiện vật cản, thay đổi lộ trình và nhận dạng lửa bằng camera”. Tiếp theo, trong chương 4 sẽ trình bày nội dung đánh giá và kiểm thử của đề tài dựa trên phần mô hình đã thiết kế và cài đặt của chương 3.

# CHƯƠNG 4: ĐÁNH GIÁ VÀ KIỂM THỬ

## 4.1 Mục tiêu kiểm thử

Kiểm thử nhằm mục đích tìm ra lỗi các tính năng trong quá trình vận hành của mô hình, kiểm tra sự ổn định của mô hình sau một khoảng thời gian sử dụng. Kiểm tra sự khả thi khi ứng dụng mô hình vào thực tế. Việc kiểm thử không khẳng định rằng các chức năng của sản phẩm hoạt động đúng trong mọi điều kiện, mà nó chỉ khẳng định rằng nó có thể hoạt động đúng trong một vài điều kiện cụ thể.

## 4.2 Kiểm thử chức năng

Quá trình chuẩn bị kiểm thử:

* Bước 1: Lắp nguồn cho thiết bị.
* Bước 2: Kết nối thiết bị với các phần cứng khác.
* Bước 3: Bật nguồn, tiến hành kiểm thử.

### 4.2.1 Kiểm thử trên board Mega 2560

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Mô tả dữ liệu**  **kiểm thử** | **Kết quả mong đợi** | **Kết quả thực tế** |
| 1 | Cảm biến khoảng cách kiểm tra vật cản: Có vật cản | - Xe dừng  - Kiểm tra khoảng cách trái và phải  - Xe rẽ sang hướng có khoảng từ xe đến vật cản xa nhất | - Xe dừng  - Kiểm tra khoảng cách trái và phải  - Xe rẽ sang hướng có khoảng từ xe đến vật cản xa nhất |
| 2 | Cảm biến khoảng cách kiểm tra vật cản: Không có vật cản | Xe đi thẳng | Xe đi thẳng |
| 3 | Nhận và xử lý tín hiệu có lửa và tọa độ vị trí của lửa từ Raspberry | - Nhận tín hiệu kết nối từ Raspberry  - Nhận dữ liệu từ Raspberry bao gồm: thông tin có lửa  - Điều khiển hướng đi của xe đến vị trí lửa  - Thông báo bằng cách bật còi báo động | - Nhận tín hiệu kết nối từ Raspberry  - Nhận dữ liệu từ Raspberry bao gồm: thông tin có lửa  - Điều khiển hướng đi của xe đến vị trí lửa  - Thông báo bằng cách bật còi báo động |

Bảng 4. 1 Kiểm thử các chức năng trên Mega

### 4.2.2 Kiểm thử trên Raspberry Pi 3 B+

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Mô tả dữ liệu**  **kiểm thử** | **Kết quả mong đợi** | **Kết quả thực tế** |
| 1 | Raspberry kết nối với Mega | -Nhận tín hiệu kết nối  - Gửi dữ liệu cho Mega | -Nhận tín hiệu kết nối  - Gửi dữ liệu cho Mega |
| 2 | Raspberry nhận dạng hình ảnh qua Camera: Thấy lửa | - Nhận dạng được lửa  - Xe dừng  - Kết nối với Mega gửi tín hiệu có lửa và tọa độ | - Nhận dạng được lửa  - Xe dừng  - Kết nối với Mega gửi tín hiệu có lửa và tọa độ |
| 3 | Raspberry nhận dạng hình ảnh qua Camera: Không thấy lửa | - Hoạt động bình thường | -Hoạt động bình thường |

Bảng 4. 2 Kiểm thử các tính năng trên Raspberry

## 4.3 Các trường hợp kiểm thử

* **Xe gặp vật cản**

Bật nguồn cho xe hoạt động di chuyển về phía trước, cảm biến HC-SR04 sẽ làm nhiệm vụ kiểm tra xuyên suốt khoảng cách từ xe đến vật thể, đến phạm vi trong vòng 20cm thì sẽ dừng xe lại. Động cơ Servo quay trái để đo khoảng cách, lưu lại giá trị đã đo, tiếp tục quay sang phải để đo khoảng cách và lưu lại giá trị đã đo.

* Trường hợp 1: Nếu cả hai khoảng cách có giá trị nhỏ hơn hoặc bằng 10 thì thực hiện lui về sau.



Hình 4. 1 Xe gặp vật cản và khoảng cách trái, phải nhỏ hơn 10cm

* Kết quả: Xe lùi về sau, tiếp tục đo khoảng cách và di chuyển với tỷ lệ đúng là tương đối, do khoảng cách quá gần đôi khi xe xử lý chậm dẫn đến va chạm.
* Trường hợp 2: Ngược lại cả hai khoảng cách có giá trị lớn hơn 10, nếu khoảng cách bên phải lớn hơn trái thì thực hiện rẽ phải.



Hình 4. 2 Xe gặp vật cản và khoảng cách trái lớn hơn phải

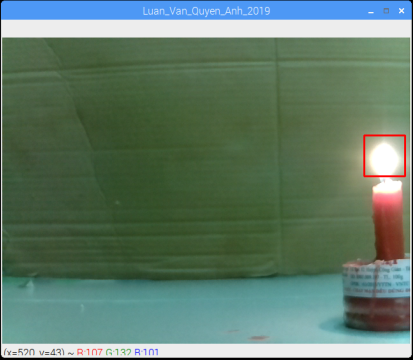
* Kết quả: Xe quay sang trái với tỷ lệ đúng cao, ít khả năng va chạm.
* Trường hợp 2: Ngược lại cả hai khoảng cách có giá trị lớn hơn 10, nếu khoảng cách bên trái lớn hơn phải thì thực hiện rẽ trái.



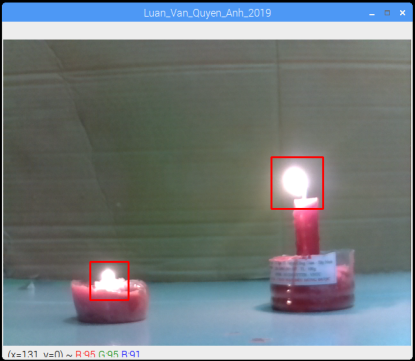
Hình 4. 3 Xe gặp vật cản và khoảng cách phải lớn hơn trái

=>Kết quả: Xe quay sang trái với tỷ lệ đúng cao, ít khả năng va chạm.

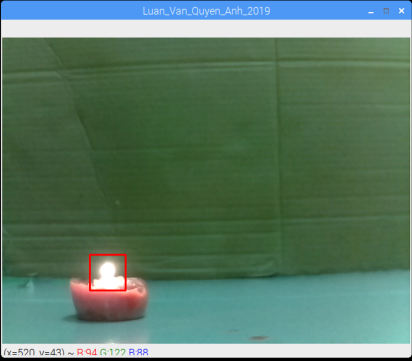
* Camera nhận dạng lửa



Hình 4. 4 Nhận dạng lửa từ Camera lần 1



Hình 4. 5 Nhận dạng lửa từ Camera lần 2



Hình 4. 6 Nhận dạng lửa từ Camera lần 3

=>Kết quả: nhận dạng được lửa khá ổn, nhưng vẫn còn nhận dạng sai khi quá xa, khi có tia sáng trắng chiếu qua, khi có gió thổi, khi nền là màu giống lửa.

## 4.4 Đánh giá kết quả kiểm thử

Do quá trình xây dựng hệ thống còn hạn chế về mặt thời gian và kiến thức hiểu biết nên việc kiểm thử chưa thể thực hiện với số lần đáng kể, vì thế vẫn chưa kiểm tra được những lỗi tiềm ẩn trong mô hình. Kiểm thử trong điều kiện về môi trường thực tế chưa nhiều và lý tưởng nên mô hình chưa có độ tương thích ở nhiều môi trường khác nhau. Chi phí đầu tư cho sản phẩm còn hạn chế nên còn một vài tính năng chưa thể kiểm thử được.

Mặc dù còn gặp nhiều khó khăn trong việc kiểm thử, nhưng kết quả kiểm thử cũng cho thấy mô hình hoạt động tương đối ổn định, ít xảy ra các tình trạng lỗi, các tính năng đơn giản hoạt động tốt.

# Tổng kết chương 4.

Trong chương 4 đã trình bày tổng quát các trường hợp kiểm thử trong quá trình vận hành của mô hình xe cứu hỏa mini tự hành và đưa ra những đánh giá kết quả từ quy trình kiểm thử trên.

# PHẦN KẾT LUẬN

## 1. Kết quả đạt được

* Về mặt lý thuyết:
* Hoàn thiện việc nghiên cứu các kỹ thuật sử dụng và điều khiển các thiết bị cảm biến và các board mạch.
* Hoàn thiện các kỹ thuật chuyển đổi giao tiếp giữa 2 board mạch Raspberry Pi 3 B+ và Mega 2560 và lắp nguồn hoạt động cho toàn mô hình.
* Hoàn thiện các kiến thức về xử lý ảnh để nhận dạng đối tượng trên Raspberry
* Về mặt thực tiễn:
* Hoàn thành các module đã đề ra là phát hiện vật cản và thay đổi đường đi; phát hiện và nhận dạng lửa thông qua Raspberry.
* Mô hình hoạt động tương đối ổn định.



Hình 1 Mô hình xe cứu hỏa mini

* Về bản thân:
* Tích lũy thêm lượng kiến thức mới không hề nhỏ.
* Rèn luyện kỹ năng lập trình C với Mega 2560 và Python với Raspberry trên nền tảng hệ điều hành Raspbian.
* Nâng cao khả năng tìm kiếm thông tin, cũng cố các kiến thức đã học.

## 2. Khó khăn

Do thời gian có hạn, kiến thức còn hạn hẹp, điều kiện tài chính chưa đủ đầu tư thiết bị và kinh nghiệm làm việc với các thiết bị điện tử còn khá hạn chế. Do đó, việc xây dựng **“Mô hình thử nghiệm xe cứu hỏa mini tự hành”** vẫn còn gặp nhiều vấn đề khó khăn và hạn chế chưa khắc phục được:

* Việc phát hiện lửa còn hạn chế không thể phát hiện lửa tại nhưng nơi khuất hoặc phía sau xe mà cảm biến lửa và camera không nhìn thấy được.
* Không thể dập tắt được đám cháy quá lớn
* Tránh vật cản làm thay đổi đường đi của xe gây ra khó khăn trong việc đi tuần
* Cảm biến khoảng cách và cảm biến phát hiện lửa hồng ngoại hoạt động không ổn định khi có ánh sáng mạnh chiếu trực tiếp vào.
* Việc quản lý năng lượng đến các module chưa được hiệu quả. Dễ xảy ra việc tốn năng lượng khi xe trong quá trình hoạt động do phải điều khiển nhiều động cơ cùng lúc.
* Mô hình xe không được an toàn do các thiết bị không được bảo vệ bởi vật liệu chống cháy.
* Chưa áp dụng và phát triển những công nghệ mới vào đề tài.

**3. Hướng phát triển**

Bên cạnh những phần đã làm được, vẫn còn nhiều chức năng mà đề tài chưa khai thác hết được. Mô hình có thể bổ sung thêm một vài tính năng như:

* Đo nhiệt độ kết hợp camera để cảnh báo nguy cơ cháy nổ sắp xảy ra, gửi thông tin cho người giám sát đồng thời hỗ trợ việc kiểm tra các thiết bị, vật dụng, tình trạng nơi xe đi qua mà không cần phải mất công sức đi đến từng nơi kiểm tra.
* Ngoài ra để an toàn cho các thiết bị trên xe cần dùng vật liệu cách nhiệt và có hệ thống nước hạ nhiệt quanh xe. Với camera có thể thay thế bởi cụm camera hồng ngoại để sử dụng trong môi trường ánh sáng yếu, nhận biết chỗ nóng đồng thời nhận biết lửa xuyên khói, bụi.
* Có thể nhận dạng có người trong đám cháy để kịp thời thông báo cứu người.
* Cải thiện đường đi cho xe, cho phép xe hoạt động trên nhiều đường đi khác nhau, xe sẽ tự động sạc khi nguồn năng lượng yếu, có giải pháp tiết kiệm năng lượng.
* Xây dựng cơ sở dữ liệu để lưu trữ lịch sử hoạt động cũng như thông tin về hỏa hoạn. Kết hợp xây dựng WebApp để dễ dàng quan sát, kịp thời nhận thông tin và điều khiển hoạt động của xe khi cần thiết.

Đó là những gì có thể phát triển trong tương lai. Còn giải quyết cho hệ thống ở hiện tại là khắc phục hoàn toàn các hạn chế như đã nêu trên, hoàn thiện bước đầu trong việc vận hành mô hình xe cứu hỏa tự hành.

# TÀI LIỆU THAM KHẢO

🙢🕮🙠

**Website tham khả****o**

[1] Nevon Projects, “Arduino based autonomous fire fighting robot”,

<https://nevonprojects.com/arduino-based-autonomous-fire-fighting-robot/>

[2]Instructables, “Firefighter Robot”,

<https://www.instructables.com/id/Firefighter-Robot/>

[3] Mechasolution, “Bài 10 Giao tiếp Arduino và Rasperry Pi (UART), <https://mechasolution.vn/Blog/bai-10-giao-tiep-arduino-va-raspberry-pi-uart>, 2018

[4] IoT Maker, “Raspberry pi 3 B+”,

<https://iotmaker.vn/raspberry-pi-model-b-plus.html>

[5] Điện tử VIETNIC, “Sơ đồ chân Arduino Mega2560”, <https://www.vietnic.vn/arduino-mega2560>

[6] “Introduction to HC-SR04 (Ultrasonic Sensor)”,

<https://www.theengineeringprojects.com/2018/10/introduction-to-hc-sr04-ultrasonic-sensor.html>,10/2018

[7] “Introduction to HC-SR04 (Ultrasonic Sensor)”, 10/2018,

<https://www.theengineeringprojects.com/2018/10/introduction-to-hc-sr04-ultrasonic-sensor.html>

[8] Mlab embedded Solutions, “Rpi camera V13”

<http://pivietnam.com.vn/rpi-camera-v13.html>

[9] IoT LabChannel, “ Mạch điều khiển động cơ L298N”, <https://iotlabchannel.com/trang-chu/mach-dieu-khien-dong-co-dc-l298n/?v=e14da64a5617>

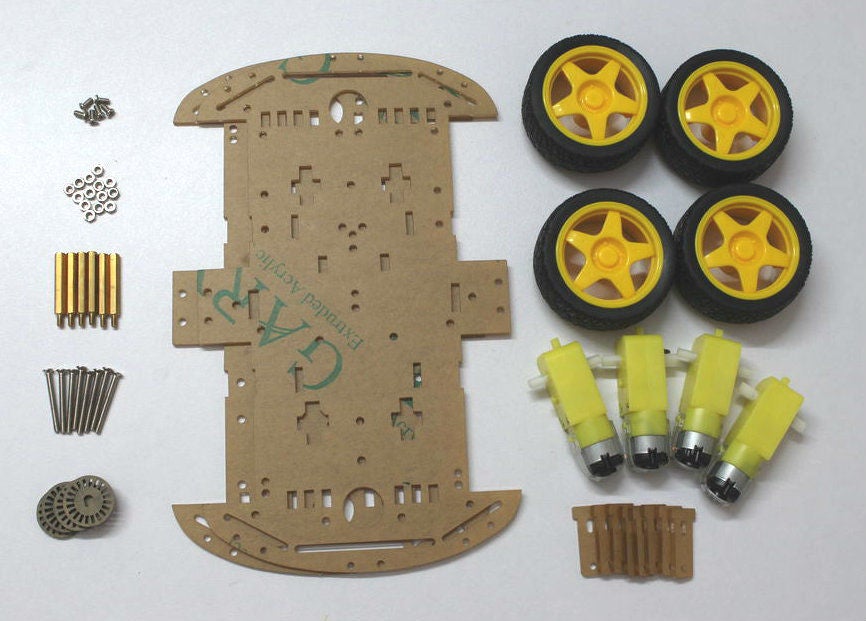
[10] Denshitaro, “Mạch hạ áp DC-DC từ 3-40V xuống 1.25-35V 3A MAX LM2596 D00-013”,

<https://www.denshitaro.vn/products/mach-ha-ap-dc-dc-tu-3-40v-xuong-1-25-35v-3a-max-lm2596-d00-013>

# PHỤ LỤC

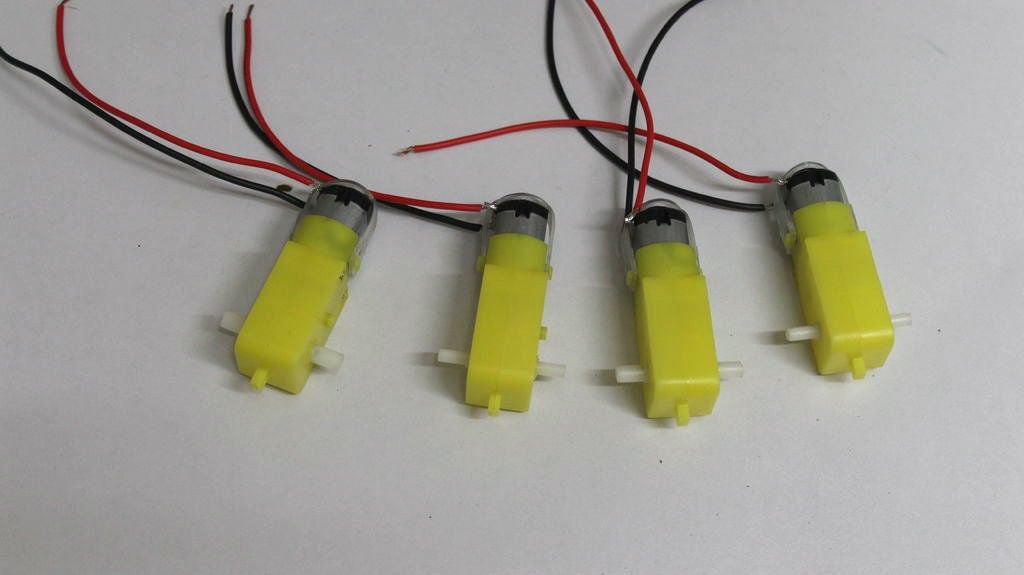
**1. Hướng dẫn lắp đặt khung xe**

**Bước 1: Chuẩn bị các công cụ và thiết bị cần thiết**

****

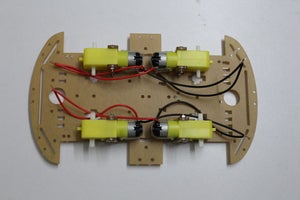
Hình 2 Các thiết bị làm khung xe

**Bước 2: Hàn dây vào động cơ giảm tốc**

****

Hình 3 Động cơ giảm tốc đã nối dây

**Bước 3: Lắp bốn động cơ giảm tốc vào khung xe**

****

Hình 4 Khung xe đã lắp động cơ giảm tốc

**Bước 4: Lắp ráp bốn bánh xe vào khung**

****

Hình 5 Khung xe đã hoàn thành

**2. Hướng dẫn cài đặt hệ điều hành Raspbian**

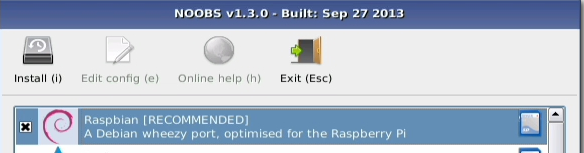
* Sử dụng gói NOOBS, đây là cách đơn giản dành cho người mới bắt đầu.

**Bước 1:**[**Download**](https://www.raspberrypi.org/downloads/noobs/)**gói NOOBS:**  
Đây là gói cài đặt đã có sẵn Raspbian được cung cấp bởi nhà phát hành Raspberry Pi.

**Bước 2: Format thẻ SD của bạn:**  
Nếu thẻ SD của bạn mới tinh thì không nhất thiết phải làm bước này.  
Phần mềm [SD Formatter 4.0](https://www.sdcard.org/downloads/formatter_4/) được khuyên dùng cho việc Format thẻ.

**Bước 3: Giải  nén gói NOOBS vào thẻ SD:**  
Sau khi việc download gói NOOBS và format thẻ hoàn tất.  
Bạn chỉ cần giải nén gói NOOBS vào thẻ SD.

**Bước 4: Gắn thẻ vào Raspberry Pi và khởi động:**  
Đây là lần khởi động đầu tiên của Raspbian.  
Khung lựa chọn tương tự như thế này:



**Hình 6 Noobs**

Bạn chỉ cần chọn Raspbian và nhấn vào biểu tượng **Install** (hoặc nhấn phím **i**) và đợi việc cài đặt hoàn thành.

* Bước đầu khởi động

Sau khi Raspbian được cài đặt xong , Raspberry Pi sẽ boot lần đầu tiên.  
Màn hình lệnh gọi là prompt sẽ hiện ra yêu cầu người dùng đăng nhập. Tên và mật khẩu dùng để đăng nhập như sau:

* login: pi
* Password: ras pberry

Sau đó nếu muốn chuyển sang giao diện Desktop ta dùng lệnh startx :

* pi@raspberrypi ~ $ startx

Muốn khởi động lại:

* pi@raspberrypi ~ $ sudo reboot

Muốn tắt Raspberry Pi:

* pi@raspberrypi ~ $ sudo shutdown -h now

hoặc

* pi@raspberrypi ~ $ sudo halt

**3. Hướng dẫn cài Arduino IDE**

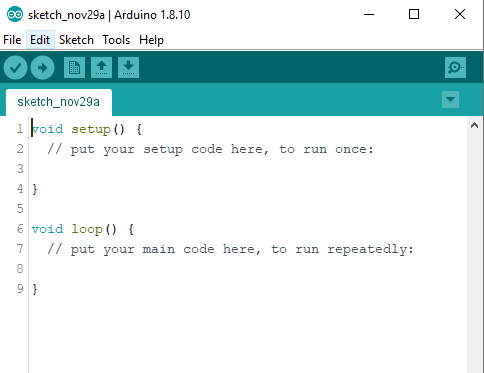
**Bước 1**: Truy cập địa chỉ [http://arduino.cc/en/Main/Software/](https://arduino.cc/en/Main/Software/) . Đây là nơi lưu trữ cũng như cập nhật các bản IDE của Arduino, sau đó download về máy.

****

**Hình 6 Giao diện download Arduino IDE**

**Bước 2:** Sau khi download xong, các bạn bấm chuột phải vào file vừa download **arduino-1.8.10-windows.zip**và chọn “**Extract here**” để giải nén.

**Bước 3:** Chạy file [http://k3.arduino.vn/img/2015/05/13/0/1342_81220-1431518023-0-2015-05-13-18h53-25.png](http://k3.arduino.vn/img/2015/05/13/0/1342_81220-1431518023-0-2015-05-13-18h53-25.png) trong thư mục **arduino-1.8.10**để khởi động **Arduino IDE**

****

**Hình 7 Giao diện sau khi cài đặt**

**4. Hướng dẫn cài thư viện**

* **OpenCv**

Đầu tiên cần phải cập nhật, nâng cấp các gói hiện có, tiếp theo là cập nhật phần mềm RaspberryPi.

$sudo apt-get update

$sudo apt-get upgrade

$sudo rpi update  
Khởi động lại Raspberry Pi sau khi cập nhật phần mềm:

$sudo reboot  
Tiếp theo là cài đặt công cụ phát triển:

$sudo apt-get install build-esential git cmake pkg-config   
Cài đặt gói ảnh I/O cho phép chúng ta tải các định dạng tệp hình ảnh như JPEG, PNG, TIFF, v..v

$sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev  
Cài đặt gói video I/O cho phép chúng ta tải các định dạng tệp video khác nhau cũng như làm việc với các luồng video:

$sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev

$sudo apt-get install libavcodec-dev libx264-dev  
 Cài đặt thư viện phát triển GTK (công cụ tạo giao diện đồ họa) để chúng ta có thể biên soạn modun phụcủa OpenCV, cho phép chúng ta hiển thị hình ảnh trên màn hình và xây dựng các giao diện GUI đơn giản:

$sudo apt-get install libgtk2.0-dev  
Cuối cùng, chúng ta cần cài đặt các tập tin Python 2.7 và Python 3 để có thể biên dịch OpenCV + Python (ngôn ngữ biên dịch) mặc định.

$sudo apt-get install python3-dev  
Lấy mã nguồn OpenCV

$cd ~

$wget -0 opencv.zip <https://github.com/Itseez/opencv/archive/3.0.0.zip>

$upzip opencv.zip  
Để cài đặt đầy đủ OpenCV 3(bao gồm các tính năng như SIFT (nhận dạng điểm đặc trưng) và SURF (nhận dạng đối tượng)) ta phải chắc chắn việc lấy kho lưu trữ của OpenCV.  
Tiếp theo là thiết lập để xây dựng opencv:

$cd ~/opencv-3.0.0/

$mkdir build

$cd build

$cmake –D CMAKE\_BUILD\_TYPE=RELEASE \  
Biên dịch OpenCV:

$make –j4  
-j4 làviết tắt của số lõi để sử dụng khi biên soạn OpenCV.Vì chúng ta đang sử dụng Raspberry Pi 2, chúng ta sẽ tận dụng tất cả bốn lõi của bộ vi xử lý để thu thập nhanh hơn.  
Giả sử OpenCV biên soạn không bị lỗi và cài đặt nó vào hệ thống bằng lệnh:

$sudo make install

$sudo ldconfig