# LỜI CẢM ƠN

🙢🕮🙠

Lời đầu tiên, tôi xin gửi lời cảm ơn chân thành đến quý Thầy Cô Trường Đại Học Cần Thơ nói chung và những Thầy Cô tại khoa Công nghệ thông tin và truyền thông nói riêng đã tận tình giảng dạy và truyền đạt những kiến thức cơ bản, đại cương và những kiến thức cơ sở để tôi có nền tảng kiến thức vững chắc cho việc học tập và phát triển kiến thức chuyên ngành.

Tiếp đến, tôi xin gửi lời biết ơn sâu sắc tới Ths. Nguyễn Hữu Vân Long và Ts. Trần Công Án, các thầy đã tận tình hướng dẫn, định hướng, giúp đỡ và động viên tôi trong suốt thời gian nghiên cứu thực hiện luận văn này. Thầy đã giúp đỡ tôi trong việc tiếp cận với khoa học, công nghệ, giải pháp, những tri thức mới trong xã hội và đạt được thành công trong lĩnh vực nghiên cứu của mình.

Tôi xin gửi lời cám ơn chân thành và sâu sắc đến cố vấn học tập của lớp DI15Y9A1 Ts. Ngô Bá Hùng. Cám ơn Thầy trong suốt khoảng thời gian vừa qua đã luôn gắn bó hết mình với tập thể lớp. Thầy đã quan tâm, chỉ bảo cho lớp trong những ngày đầu bước chân vào giảng đường đại học. Thầy luôn động viên khích lệ tinh thần tập thể lớp nói chung; Thầy đã luôn tân tình giúp đỡ và chia sẻ với tập thể lớp những kiến thức, kinh nghiệm hữu ích trong quá trình học tập cũng như trong cuộc sống.

Cuối cùng chúng tôi xin gửi lời cảm ơn tới tất cả các thành viên trong gia đình, bạn bè những người luôn ở bên cạnh và chia sẻ cho chúng tôi để hoàn thành đề tài này.

Tôi xin chân thành cảm ơn!

TP.Cần Thơ, ngày tháng năm 2019

Sinh viên thực hiện

Thái Minh Anh

# MỤC LỤC

[LỜI CẢM ƠN i](#_Toc26471909)

[MỤC LỤC ii](#_Toc26471910)

[DANH MỤC BẢNG v](#_Toc26471911)

[DANH MỤC HÌNH vi](#_Toc26471912)

[KÝ HIỆU VÀ VIẾT TẮT vii](#_Toc26471913)

[TÓM TẮT ĐỀ TÀI viii](#_Toc26471914)

[ABSTRACT ix](#_Toc26471915)

[PHẦN GIỚI THIỆU 1](#_Toc26471916)

[1. Đặt vấn đề 1](#_Toc26471917)

[2. Lịch sử giải quyết vấn đề 1](#_Toc26471918)

[3. Mục tiêu đề tài 2](#_Toc26471919)

[4. Đối tượng và phạm vi nghiên cứu 2](#_Toc26471920)

[5. Nội dung nghiên cứu 3](#_Toc26471921)

[6. Bố cục luận văn 3](#_Toc26471922)

[PHẦN NỘI DUNG 4](#_Toc26471923)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 4](#_Toc26471924)

[1.1. Cháy nổ 4](#_Toc26471925)

[1.2 Arduino IDE (Integrated Development Environment) 4](#_Toc26471926)

[1.3 Raspbian 6](#_Toc26471927)

[1.4 Mô hình YOLOv3 7](#_Toc26471928)

[1.5 Các gói cài đặt và thư viện 7](#_Toc26471929)

[1.5.1 Tensorflow 7](#_Toc26471930)

[1.5.2 OpenCV 8](#_Toc26471931)

[1.5.3 Keras 9](#_Toc26471932)

[1.5.4 ImageAI 9](#_Toc26471933)

[1.6 Phần cứng thiết bị 9](#_Toc26471934)

[1.6.1 Raspberry Pi 3 B+ 9](#_Toc26471935)

[1.6.2 Camera 10](#_Toc26471936)

[1.6.3 Arduino Mega 2560 11](#_Toc26471937)

[1.6.4 Module điều khiển động cơ L298N. 13](#_Toc26471938)

[1.6.5 Động cơ giảm tốc V1. 14](#_Toc26471939)

[1.6.6 Động cơ Servo SG90. 15](#_Toc26471940)

[1.6.7 Motor bơm nước 15](#_Toc26471941)

[1.6.8 Còi báo động SFM-27 16](#_Toc26471942)

[1.6.9 Cảm biến phát hiện lửa 5 kênh. 17](#_Toc26471943)

[1.6.10 Mạch giảm áp DC LM2596 17](#_Toc26471944)

[1.6.11 Pin Lipo 18](#_Toc26471945)

[Tổng kết chương 1. 19](#_Toc26471946)

[CHƯƠNG 2: MÔ TẢ BÀI TOÁN 20](#_Toc26471947)

[2.1 Mô tả bài toán 20](#_Toc26471948)

[2.1.1 Mô tả tổng quan 20](#_Toc26471949)

[2.1.2 Yêu cầu giao tiếp 21](#_Toc26471950)

[Tổng kết chương 2. 22](#_Toc26471951)

[CHƯƠNG 3: CÀI ĐẶT GIẢI PHÁP 23](#_Toc26471952)

[3.1 Cài đặt 23](#_Toc26471953)

[3.2 Lưu đồ giải thuật 24](#_Toc26471954)

[3.2.1 Phát hiện lửa 24](#_Toc26471955)

[3.2.2 Dập tắt lửa 26](#_Toc26471956)

[3.3 Thiết kế mô hình 27](#_Toc26471957)

[3.3.1 Kết nối trên Arduino Mega 27](#_Toc26471958)

[3.3.2 Raspberry kết nối với Mega 33](#_Toc26471959)

[Tổng kết chương 3. 34](#_Toc26471960)

[CHƯƠNG 4: ĐÁNH GIÁ VÀ KIỂM THỬ 35](#_Toc26471961)

[4.1 Mục tiêu kiểm thử 35](#_Toc26471962)

[4.2 Các trường hợp đánh giá và kiểm thử 35](#_Toc26471963)

[4.2.1 Đánh giá thành phần của mô hình 35](#_Toc26471964)

[4.2.2 Kiểm thử chức năng 37](#_Toc26471965)

[4.3 Đánh giá kết quả kiểm thử 39](#_Toc26471966)

[Tổng kết chương 4. 39](#_Toc26471967)

[PHẦN KẾT LUẬN 40](#_Toc26471968)

[1. Kết quả đạt được 40](#_Toc26471969)

[2. Khó khăn 40](#_Toc26471970)

[3. Hướng phát triển 41](#_Toc26471971)

[TÀI LIỆU THAM KHẢO 42](#_Toc26471972)

[PHỤ LỤC 43](#_Toc26471973)

# DANH MỤC BẢNG

[Bảng 1. 1 Cổng giao tiếp với phần cứng 12](#_Toc26471672)

[Bảng 3. 1 Sơ đồ chân cắm của Mega và cảm biến lửa 27](#_Toc26471783)

[Bảng 3. 2 Sơ đồ chân cắm của Mega với L298N và động cơ giảm tốc 29](#_Toc26471784)

[Bảng 3. 3 Sơ đồ chân cắm của Mega, Motor nước và còi 30](#_Toc26471785)

[Bảng 4. 1 Ưu nhược điểm của phần cứng 37](#_Toc26471797)

[Bảng 4. 2 Kiểm thử các tính năng trên board Mega 38](#_Toc26471798)

[Bảng 4. 3 Kiểm thử các tính năng trên Raspberry Pi 39](#_Toc26471799)

# DANH MỤC HÌNH

[Hình 1. 1 Arduino IDE 6](#_Toc26471848)

[Hình 1. 2 Raspbian 6](file:///D:\hoc\luận%20văn\luanvanMA\IN\MinhAnh.docx#_Toc26471849)

[Hình 1. 3 Raspberry Pi 3 B+ 10](#_Toc26471850)

[Hình 1. 4 Camera 11](#_Toc26471851)

[Hình 1. 5 Arduino Mega 2560 13](#_Toc26471852)

[Hình 1. 6 Module L298N 14](#_Toc26471853)

[Hình 1. 7 Motor giảm tốc 15](#_Toc26471854)

[Hình 1. 8 Động cơ servo 15](#_Toc26471855)

[Hình 1. 9 Motor bơm nước 16](#_Toc26471856)

[Hình 1. 10 Còi báo 16](file:///D:\hoc\luận%20văn\luanvanMA\IN\MinhAnh.docx#_Toc26471857)

[Hình 1. 11 Cảm biến phát hiện lửa 5 kênh 17](#_Toc26471858)

[Hình 1. 12 LM2596 18](file:///D:\hoc\luận%20văn\luanvanMA\IN\MinhAnh.docx#_Toc26471859)

[Hình 1. 13 Pin Lipo 18](file:///D:\hoc\luận%20văn\luanvanMA\IN\MinhAnh.docx#_Toc26471860)

[Hình 2. 1 Khối xử lý 20](file:///D:\hoc\luận%20văn\luanvanMA\IN\MinhAnh.docx#_Toc26471861)

[Hình 3. 1 Thuật toán phát hiện lửa bằng cảm biến 24](file:///D:\hoc\luận%20văn\luanvanMA\IN\MinhAnh.docx#_Toc26471862)

[Hình 3. 2 Thuật toán phát hiện lửa bằng camera 25](#_Toc26471863)

[Hình 3. 3 Thuật toán phát hiện và dập lửa 26](file:///D:\hoc\luận%20văn\luanvanMA\IN\MinhAnh.docx#_Toc26471864)

[Hình 3. 4 Kết nối cảm biến phát hiện lửa 27](#_Toc26471865)

[Hình 3. 5 Kết nối động cơ bánh xe 28](#_Toc26471866)

[Hình 3. 6 Kết nối động cơ báo động và dập tắt lửa 30](#_Toc26471867)

[Hình 3. 7 Mô hình kết nối tổng quát 32](#_Toc26471868)

[Hình 3. 8 Phát hiện lửa trên Raspberry gửi sang Mega 33](#_Toc26471869)

# 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 |
| VCC | Voltage Controlled Clock |
| GND | Ground |
| ICSP | In Circuit Serial Programming |
| TX | Transmitter |
| RX | Receiver |
| YOLO | You only look once |

# 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 được. 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 mini tự hành**" để xây dựng đề tài luận văn của mình.

Trong nghiên cứu này, việc phát hiện và tự động dập tắt đám cháy kết hợp với việc tránh chướng ngại vật được tập trung để phân tích. 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 và cảm biến phát hiện lửa, camera được sử dụng để phát hiện đám cháy trong phạm vi có thể. Bên cạnh đó, mô hình xe có thêm một vòi có khả năng phun nước và xoay theo nhiều hướng. 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.

  In this study, the detection and automatic extinguishment of fires combined with obstacle avoidance were concentrated for the analysis. 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 and a fire detection sensor, the camera is used to detect the flame in the setting area. In addition, the model has a water jet spray which is able to perform of sprinkling water and rotate towards the required direction. 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**

Cùng với sự phát triển của xã hội, bên cạnh những lợi ích cũng tiềm tàng những hiểm họa có thể xảy ra với con người mà trong đó phải kể đến hỏa hoạn. Hậu quả của nó gây ra là rất lớn và khó có thể lường trước được. Hỏa hoạn chẳng những ảnh hưởng trực tiếp đến tính mạng và tài sản của mọi người mà còn gây tâm lý bất an trong xã hội, ảnh hưởng xấu đến tình hình an ninh trật tự xã hội.

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, lực lượng chức năng về phòng cháy chữa cháy chưa kịp thời tiếp nhận thông tin báo, dẫn tới việc triển khai cứu hỏa, cứu nạn, cứu hộ bị chậm trễ… Việc này dẫn tới nhiều hệ lụy như: đám cháy phát triển nhanh mất kiểm soát, đám cháy lan rộng ra xung quanh.

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**”. Trong phân hệ “phát hiện và dập tắt lửa bằng cảm biến và camera” đã sử dụng sử dụng vi điều khiển trung tâm là module Arduino Mega và một máy tính nhỏ là Raspberry Pi 3 module B+, cảm biến phát hiện lửa, camera cùng các động cơ để di chuyển xe và dập tắt lửa.

1. **Lịch sử giải quyết vấn đề**

Hiện nay, Robot chữa cháy 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. 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 trên thế giới như Ấn Độ, Nhật Bản, Nga… cũng đã phát triển việc cứu hỏa bằng robot. 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 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.

1. **Mục tiêu đề tài**

Mục tiêu 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 phân hệ phục vụ cho mô hình xe có thể tự động di chuyển, phát hiện và dập lửa bên trong tòa nhà. Trong báo cáo này tập trung nghiên cứu về phát hiện và dập lửa thông qua cảm biến và camera.

Đề tài đặt ra mục tiêu:

* Nhận dạng hình ảnh từ xa qua camera để bật còi báo động khi có lửa.
* Cảm biến phát hiện lửa trong phạm vi có thể sẽ gửi tín hiệu cho Mega điều khiển sang hướng mà cảm biến lửa phát hiện.
* Kết hợp giữa việc phát hiện lửa bằng cảm biến và camera nhằm độ chính xác của việc nhận dạng lửa sau đó thực hiện quá trình dập tắt lửa.

Với mục tiêu trên xe chữa cháy 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. Chính vì thế, nó sẽ giảm thiểu được sự bùng phát của những đám cháy, hạn chế nhiều thiệt hại do hỏa hoạn gây ra đến tính mạng và tài sản con người.

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

Đề tài “**Mô hình thử nghiệm xe cứu hỏa mini tự hành**” với phân hệ “phát hiện và dập tắt lửa bằng camera” đã tiến hành thực hiện và nghiên cứu các nội dung sau:

* Tìm hiểu các mô hình phát hiện, cảnh báo và chữa cháy hiện có, những ưu điểm và hạn chế của nó. Từ đó lựa chọn giải pháp cho mô hình của mình.
* Tìm hiểu cơ sở lý thuyết liên quan đến đề tài: nguyên nhân cháy nổ, nhận dạng đám cháy, cách thức chữa cháy,...
* Thiết kế sơ đồ khối, lựa chọn linh kiện phù hợp cho mô hình
* Lắp ráp các board mạch, cảm biến tạo thành mô hình
* Tìm hiểu các giao thức truyền nhận giữa các module trong mô hình.
* Tìm hiểu về Arduino IDE và ngôn ngữ C – môi trường lập trình cho Arduino
* Tìm hiểu về Raspbian và ngôn ngữ Python – môi trường làm việc và lập trình cho Raspberry
* Tìm hiểu về cách giao tiếp giữa Arduino Mega và Raspberry thông qua UART
* Tìm hiểu và thực hiện việc nhận dạng lửa dựa trên mô hình nhận dạng lửa của Yolo
* Cài đặt và sử dụng VNC viewer cho Raspberry để điều khiển Raspberry phục vụ cho việc lập trình

1. **Nội dung nghiên cứu**

|  |  |
| --- | --- |
| Nội dung công việc | Người thực hiện |
| - Tìm hiểu và lập trình tránh vật cản bằng cảm biến khoảng cách trên Mega | Huỳnh Thị Yến Quyên |
| - Tìm hiểu và lập trình về phát hiện lửa bằng cảm biến phát hiện lửa và dập tắt lửa trên Mega | Thái Minh Anh |
| - Xây dựng mô hình lắp đặt phần cứng cho xe | Thái Minh Anh  Huỳnh Thị Yến Quyên |
| - Phát hiện lửa bằng camera trên Raspberry | Thái Minh Anh  Huỳnh Thị Yến Quyên |
| - Thiết lập kết nối và gửi tín hiệu giữa Raspberry và Mega | Thái Minh Anh  Huỳnh Thị Yến Quyên |

1. **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 mô hình xe cứu hỏa mini tự hành gồm: giới thiệu về hiểu biết cơ bản về lửa, phần mềm và phần cứng như là Raspberry Pi, Camera, Arduino Mega 2560, các cảm biến, các thiết bị điện.

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

Chương 3: Trình bày thiết kế, chức năng của mô hình, cách thức cài đặt từng phần các 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 các module trong mô hình

# PHẦN NỘI DUNG

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

## 1.1. Cháy nổ

**Khái niệm**

Cháy là phản ứng hoá học giữa chất cháy và oxy trong không khí có sự toả nhiệt và phát ánh sáng. Như vậy cháy có 3 dấu hiệu sau:

* Phản ứng hoá học.
* Toả nhiệt.
* Phát sáng.

**Những biện pháp phòng cháy chữa cháy cơ bản**

* Tạo ra môi trường không cháy hoặc khó cháy.
* Ngăn không cho lửa xuất hiện.
* Không cho vật cháy gần ngọn lửa.
* Giảm khối lượng chất cháy.
* Hạn chế diện tích cháy.

**Những biện pháp chữa cháy cơ bản**

Có nhiều cách để chữa cháy mỗi cách sẽ mang lại những hiệu quả khác nhau, và dập tắt các đám cháy lớn nhỏ khác nhau. Có thể các cách chữa cháy cơ bản khác nhau như:

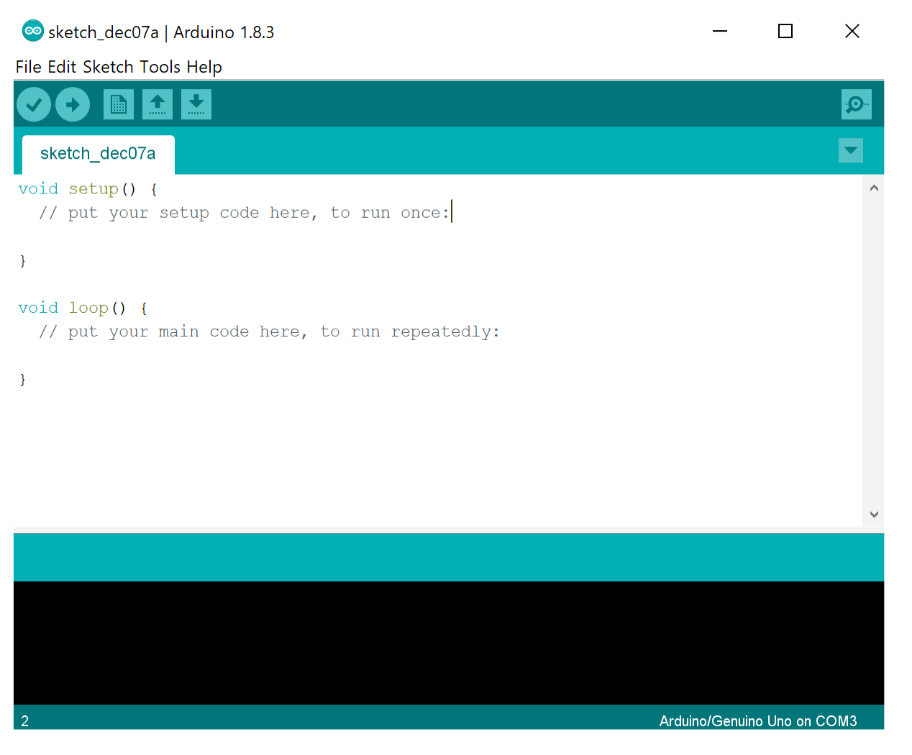
* Chữa cháy bằng nước
* Chữa cháy bằng cát
* Chữa cháy bằng khí trơ hoặc bằng bọt hòa không khí

## 1.2 Arduino IDE (Integrated Development Environment)

Là trình soạn thảo văn bản chạy trên các máy tính cá nhân, giúp việc lập trình và nạp vào board arduino. Các chương trình trên Arduino IDE thông thường sẽ viết bằng ngôn ngữ C, C++ hoặc Python.

Giao diện của Arduino IDE gồm nhiều phần:

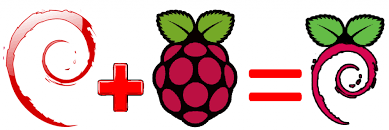
* Nút kiểm tra chương trình: Dùng để kiểm tra xem chương trình được viết có lỗi không. Nếu chương trình bị lỗi thì phần mềm Arduino IDE sẽ hiển thị thông tin lỗi ở vùng thông báo thông tin.
* Nút nạp chương trình xuống board Arduino: Dùng để nạp chương trình được viết xuống mạch Arduino. Trong quá trình nạp, chương trình sẽ được kiểm tra lỗi trước sau đó mới thực hiện nạp xuống mạch Arduino.
* Hiển thị màn hình giao tiếp với máy tính: Khi nhấp vào biểu tượng cái kính lúp thì phần giao tiếp với máy tính sẽ được mở ra. Phần này sẽ hiển thị các thông số mà người dùng muốn đưa lên màn hình. Muốn đưa lên màn hình phải có lệnh Serial.print() mới có thể đưa thông số cần hiển thị lên màn hình
* Vùng lập trình: Vùng này để người lập trình thực hiện việc lập trình cho chương trình của mình.
* Vùng thông báo thông tin: Có chức năng thông báo các thông tin lỗi của chương trình hoặc các vấn đề liên quan đến chương trình được lập.
* Một số menu thông dụng trên phần mềm Arduino IDE:
  + Có vài menu trong phần mềm IDE, tuy nhiên thông dụng nhất vẫn là menu File, ngoài những tính năng như mở một file mới hay lưu một file, phần menu này có một mục đáng chú ý là Example. Phần Example đưa ra các ví dụ sẵn để người lập trình có thể tham khảo, giảm bớt thời gian lập trình.
  + Một menu thường được sử dụng khác là menu Tools. Khi mới kết nối board Arduino với máy tính ta click vào **Tools->board** để chọn loại board sử dụng. Phần mềm chọn sẵn kiểu board là board **Arduino Uno**, nếu người dùng dùng kiểu board khác thì chọn kiểu board đang dùng.
  + Bên cạnh việc chọn board thì một phần quan trọng nữa là chọn cổng COM. Khi lần đầu gắn mạch Arduino vào máy tính, người sử dụng cần nhấn chọn cổng COM bằng cách vào **Tools -> Serial Port** (một số phiên bản dùng từ Port) sau đó nhấn chọn cổng COM. Những lần sau khi đưa chính board Arduino đó vào máy tính thì không cần chọn cổng COM, nếu đưa board Arduino khác vào máy thì cần phải chọn lại cổng COM, quy trình thực hiện cũng tương tự.



Hình 1. 1 Arduino IDE

## 1.3 Raspbian

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 Raspbian

## 1.4 Mô hình YOLOv3

YOLO 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 đang có ba phiên bản trong đáng chú ý là YOLOv3 với độ chính xác cao. Trong khi YOLOv2 chạy trên 45 FPS trên Titan X, phiên bản hiện tại YOLOv3 có xung nhịp khoảng 30 FPS. Điều này có liên quan đến sự gia tăng độ phức tạp của kiến ​​trúc cơ bản được gọi là Darknet.

YOLOv2 đã sử dụng kiến ​​trúc sâu tùy chỉnh darknet-19, mạng 19 lớp ban đầu được bổ sung thêm 11 lớp để phát hiện đối tượng. Với kiến ​​trúc 30 lớp, YOLOv2 thường phải vật lộn với các phát hiện đối tượng nhỏ. Điều này được quy cho việc mất các tính năng hạt mịn khi các lớp được ghép xuống đầu vào. Để khắc phục điều này, YOLOv2 đã sử dụng ánh xạ định danh, ghép các bản đồ tính năng từ lớp trước để chụp các tính năng cấp thấp.

Tuy nhiên, kiến ​​trúc của YOLOv2 vẫn còn thiếu một số yếu tố quan trọng nhất hiện đang là chủ yếu trong hầu hết các thuật toán hiện đại. Không có khối dư, không bỏ qua các kết nối và không lấy mẫu. YOLOv3 kết hợp tất cả những điều này.

Đầu tiên, YOLOv3 sử dụng một biến thể của Darknet, ban đầu có mạng 53 lớp được đào tạo trên Imagenet. Đối với nhiệm vụ phát hiện, có thêm 53 lớp được xếp chồng lên nhau, tạo cho chúng ta một lớp 106 kiến ​​trúc cơ bản tích chập hoàn toàn cho YOLOv3. Đây là lý do đằng sau sự chậm chạp của YOLOv3 so với YOLOv2.

## 1.5 Các gói cài đặt và thư viện

### 1.5.1 Tensorflow

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. Trong tesorflow có một vài khái niệm cơ bản sau:

* Tensor là cấu trúc dữ liệu trong tensorflow đại diện cho tất cả các loại dữ liệu. Nói cách khác, tất cả các kiểu dữ liệu khi đưa vào trong tensorflow thì đều được gọi là Tensor. Vậy nên có thể hiểu được Tensorflow là một thư viện mô tả, điều chỉnh dòng chảy của các Tensor. Tensor có 3 thuộc tính cơ bản là *rank*, *shape* và *type*.
* Rank là số bậc của tensor. Ví dụ Tensor = [1] thì có rank = 1, Tensor = [[3,4],[5,6]] thì sẽ có rank = 2. Việc phân rank này khá quan trọng vì nó đồng thời cũng giúp phân loại dữ liệu của Tensor. Khi các rank đặc biệt cụ thể, Tensor có những tên gọi riêng như sau:
  + Scalar: Khi Tensor có rank bằng 0
  + Vector: Vector là một Tensor rank 1. .
  + Matrix: Đây là một Tensor rank 2 hay mảng hai chiều theo khái niệm của Python
  + N-Tensor: Khi rank của Tensor tăng lên lớn hơn 2, chúng được gọi chung là N-Tensor.
* Shape là chiều của tensor. Ví dụ Tensor = [[[1,1,1],[178,62,74]]] sẽ có Shape = (1,2,3), Tensor = [[1,1,1],[178,62,74]] sẽ có Shape = (2,3).
* Type kiểu dữ liệu của các elements trong Tensor. Vì một Tensor chỉ có duy nhất một thuộc tính Type nên từ đó cũng suy ra là chỉ có duy nhất một kiểu Type duy nhất cho toàn bộ các elements có trong Tensor hiện tại.

Vì Tensorflow không phải là thư viện có sẵn trong python vì vậy khi sử dụng cần phải import bằng câu lệnh import tensorflow as tf.

### 1.5.2 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. Nó chính thức được ra mắt đầu tiên vào năm 1999. OpenCV đươc 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 có cộng đồng hơn 47 nghìn người dùng và số lượng download vượt quá 6 triệu lần.

Thư viện OpenCV bao gồm một số tính năng nổi bật như:

* Bộ công cụ hỗ trợ 2D và 3D
* Nhận diện khuôn mặt
* Nhận diện cử chỉ
* Nhận dạng chuyển động, đối tượng, hành vi,
* Tương tác giữa con người và máy tính
* Điều khiển Robot
* Hỗ trợ thực tế tăng cường

Thư viện OpenCV được sử dụng rộng rãi trong nhiều lĩnh vực.

* Giám sát tự động
* Tìm kiểm, phục hồi, xử lý ảnh
* Nhận dạng khuôn mặt, cử chỉ.
* Nhận dạng chữ viết, con số, ký tự.

### 1.5.3 Keras

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 google), 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 chạy trên cả cpu và gpu.
* Hỗ trợ xây dựng CNN, RNN và có thể kết hợp cả hai.

Cấu trúc của keras có thể chia ra thành 3 phần chính:

* Function để dựng bộ xương cho model.
* Function dùng để tiền dữ liệu.
* Function ở trong bộ xương của model.

Models trong keras. Để khởi tạo một model trong keras ta có thể dùng 2 cách:

* Cách 1: Thông qua Sequential.
* Cách 2: Dùng function API.

### 1.5.4 ImageAI

ImageAI là project phát hiện và nhận diện vật thể trong ảnh. Sử dụng thuật toán thị giác máy tính và deep learning cho ra kết quả chính xác cao. Project viết bằng Python và dễ dàng chạy trên nhiều hệ điều hành khác nhau.

## 1.6 Phần cứng thiết bị

### 1.6.1 Raspberry Pi 3 B+

Raspberry Pi 3 B+ là chiếc máy tính kích thước nhỏ được tích hợp nhiều phần cứng mạnh mẽ đủ khả năng chạy hệ điều hành và cài đặt được nhiều ứng dụng trên nó. Ban đầu, tổ chức Raspberry Pi Foundation phát triển dự án Raspberry với mục tiêu chính là giảng dạy máy tính cho trẻ em và tạo ra một công cụ giá rẻ (chỉ vài chục đô) để sinh viên nghiên cứu học tập. Tuy nhiên, sau khi xuất hiện Raspberry Pi được cộng đồng đánh giá cao về tính ứng dụng với phần cứng được hỗ trợ tốt, Pi đã nhanh chóng phát triển một cách rộng rãi. Pi phù hợp cho những ứng dụng cần khả năng xử lý mạnh mẽ, đa nhiệm hoặc giải trí và đặc biệt cần chi phí thấp. Hiện nay đã có hàng ngàn ứng dụng đa dạng được cài đặt trên Rasberry Pi.

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. 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. 3 Raspberry Pi 3 B+

### 1.6.2 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. 4 Camera

### 1.6.3 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 |

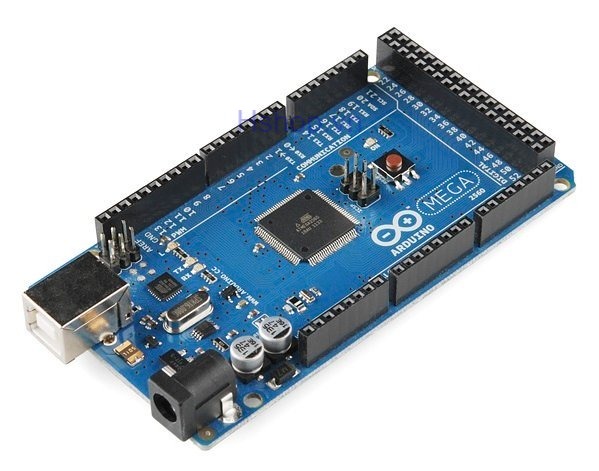
Bảng 1. 1 Cổng giao tiếp với phần cứng

* 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. 5 Arduino Mega 2560

### 1.6.4 Module điều khiển động cơ L298N.

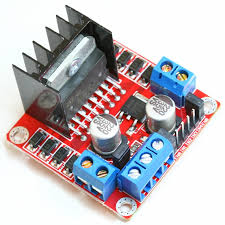
Module điều khiển động cơ (Motor Driver) sử dụng chip cầu H L298N giúp điều khiển tốc độ và chiều quay của động cơ DC một cách dễ dàng, ngoài ra module L298N còn điều khiển được một động cơ bước lưỡng cực. Mạch cầu H của IC L298N có thể hoạt động ở điện áp từ 5V đến 35V. Ngoài ra, Module L298N có tích hợp một IC nguồn 78M05 để tạo ra nguồn 5V để cung cấp cho các thiết bị khác.

**Thông số kỹ thuật**

* Driver: L298N tích hợp hai mạch cầu H.
* Điện áp điều khiển: +5 V ~ +12 V
* Dòng tối đa cho mỗi cầu H là: 2A (=>2A cho mỗi motor)
* Điện áp của tín hiệu điều khiển: +5 V ~ +7 V
* Dòng của tín hiệu điều khiển: 0 ~ 36mA
* Công suất hao phí: 20W (khi nhiệt độ T = 75 ℃)
* Nhiệt độ bảo quản: -25 ℃ ~ +130 ℃
* Kích thước: 43x43x27mm

**Sơ đồ chân tín hiệu module L298N**

* 12V power, 5V power. Đây là 2 chân cấp nguồn trực tiếp đến động cơ.
  + Có thể cấp nguồn 9-12V ở 12V.
  + Chỉ cần cấp nguồn 12V vô ở 12V power là có 5V ở 5V power, từ đó cấp cho Arduino
* Power GND chân này là GND của nguồn cấp cho Động cơ.
* Nếu kết nối với Arduino thì nhớ nối với GND của Arduino
* Hai Jump A enable và B enable, để điều chỉnh xung băm cho hoạt động của động cơ
* Bốn chân Input IN1, IN2, IN3, IN4 và bốn chân OUT1, OUT2, OUT3, OUT4

****

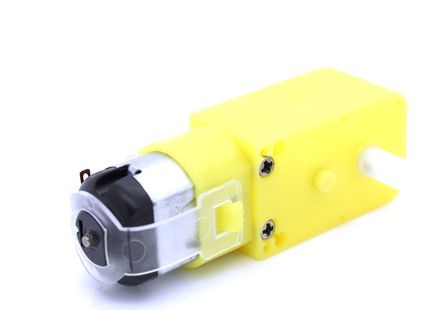
Hình 1. 6 Module L298N

### 1.6.5 Động cơ giảm tốc V1.

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.
* Tỉ 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. 7 Motor giảm tốc

### 1.6.6 Động cơ Servo SG90.

Động cơ RC Servo SG90 là động phổ biến dùng trong các mô hình điều khiển nhỏ và đơn giản như 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:**

* Khối lượng : 9g.
* Kích thước: 22.2x11.8.32 mm.
* Momen xoắn: 1.8kg/cm.
* Tốc độ hoạt động: 60 độ trong 0.3 giây.
* Điện áp hoạt động: 4.8V(~5V).
* Nhiệt độ hoạt động: 0 ºC – 55 ºC.



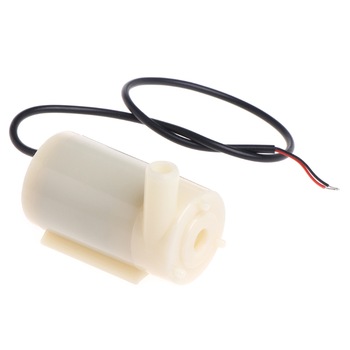
Hình 1. 8 Động cơ servo

### 1.6.7 Motor bơm nước

Động cơ bơm chìm Mini 5VDC có kích thước rất nhỏ gọn, sử dụng điện áp 3~5VDC, vì thuộc dạng bơm chìm nên động cơ có khả năng chống nước và hoạt động khi ngâm chìm trong nước, ứng dụng để bơm nước, dung dịch trong các thiết kế nhỏ, mô hình tưới cây, hồ cá...

**Thông số kỹ thuật**

* Điện áp sử dụng: 3~5VDC.
* Dòng điện sử dụng: 100~200mA.
* Lưu lượng bơm: 1.2~1.6L / 1 phút.
* Đường kính ngoài ống dẫn: 7.5mm
* Kích thước: 34 x 43 mm
* Trọng lượng: 28g



Hình 1. 9 Motor bơm nước

### 1.6.8 Còi báo động SFM-27

**Thông số kỹ thuật**

* Model: SFM-27
* Điện áp hoạt động: 12V
* Dải điện áp: 220VAC
* Dòng hoạt động: ≤ 30mA
* Cường độ âm thanh: ≥85dB
* Tần số âm thanh: 3000 ± 500Hz
* Kích thước: Đường kính: 30mm, Cao: 15MM
* Khoảng cách 2 lỗ: 40mm

Hình 1. 10 Còi báo

### 1.6.9 Cảm biến phát hiện lửa 5 kênh.

Sử dụng module này để phát hiện lửa ở phạm vi rộng (>120 độ). Nó nhận diện đám cháy bằng các sử dụng 5 cảm biến độc lập được bố trí cách nhau 30 độ. Module đầu ra bao gồm cả tín hiệu tương tự (giá trị thay đổi theo cường độ của ngọn lửa được cảm nhận) và cả tín hiệu số phù hợp với các ứng dụng máy vi tính. Chiết áp của board mạch có thể điều chỉnh mức độ nhạy đầu ra kỹ thuật số. 5 đèn led được tích hợp sẽ hỗ trợ gỡ lỗi và kiểm tra hệ thống.

**Nguyên tắc làm việc**

Phát hiện dãy lửa trong phạm vi 700-1100nm, sóng hồng ngoại gần sóng ngắn (SW-NIR) và kích hoạt đầu ra thông qua tín hiệu điện khi được cảm nhận.

**Thông số kỹ thuật**

* Điện áp cung cấp: Đầu ra 3.3 – 9V chia 5
* Điều chỉnh kỹ thuật số tương tự: Chiếc áp trên board mạch để điều chỉnh độ nhạy
* Tần số: 760nm đến 1100nm
* Khu vực phát hiện: >120 độ ở khoảng 80cm



Hình 1. 11 Cảm biến phát hiện lửa 5 kênh

### 1.6.10 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%
* Công suất : 15W
* Kích thước: 45 (dài) \* 20 (rộng) \* 14 (cao) mm

Hình 1. 12 LM2596

### 1.6.11 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. 13 Pin Lipo

# Tổng kết chương 1.

Trong chương 1 đã trình bày sơ lược về các khái niệm, 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 và các 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**

Hiện tại xe chữa cháy được xây dựng với dạng mô hình thử nghiệm phát hiện và dập tắt lửa những ngọn lửa nhỏ trong môi trường văn phòng, nơi có điều kiện ánh sáng không quá cao thích hợp cho hoạt động của cảm biến lửa để việc nhận dạng có thể đạt được kết quả tốt nhất có thể. Với mong muốn đây sẽ là nền tảng để có thể phát triển mô hình thay thế con người đi tuần tra tại những nguy cơ xảy ra cháy nổ cao như nhà máy, xưởng sản xuất, nhà kho,…

* **Các tính năng của mô hình**
* Phát hiện lửa từ xa bằng cảm biến phát hiện lửa và camera
* Điều hướng xe đến vị trí cháy
* Thông báo và tự động dập lửa
* **Thiết kế sơ đồ khối**

**Pin**

(Cấp nguồn)

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 phát hiện lửa**

(Đọc cảm biến)

**Chức năng của các thành phần trong sơ đồ khối**

* Mega (khối xử lý trung tâm): Nhận giá trị từ các cảm biến, Raspberry và xử lý giá trị nhận được, đồng thời gửi tín hiệu thực thi các động cơ.
* Cảm biến phát hiện lửa: Phát hiện có lửa bằng sự tăng cao của tia hồng ngoại trong môi trường từ đó gửi tín hiệu về khối xử lý trung tâm.
* Động cơ: Nhận tín hiệu yêu cầu thực thi của khối xử lý trung tâm và thực hiện nó.
* Pin: Cung cấp nguồn cho khối xử lý trung tâm, động cơ và Raspberry.
* Raspberry: Nhận hình ảnh từ camera thực hiện xử lý sau đó gửi sang cho khối xử lý trung tâm.
* Camera: Thu hình ảnh trong khi hoạt động, gửi hình ảnh cho Raspberry.
* **Phần cứng yêu cầu các thiết bị**
* Raspberry Pi 3 B+
* Camera
* Arduino Mega 2560
* Hai module L298N
* Bốn motor động cơ giảm tốc
* Hai động cơ servo
* Motor bơm nước
* Còi báo
* Cảm biến: Phát hiện lửa
* Breaboard và dây cắm
* Khung và bốn bánh xe
* Bình chứa nước và ống phun
* Pin
* **Phần mềm yêu cầu**
* Arduino yêu cầu phiên bản 1.6.0 trở lên
* Raspberry pi 3 B+ cài hệ điều hành Raspbian
* Các gói thư viện OpenCV, Tensorflow, Keras, ImageAI, Servo
* **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.
* Máy tính truy cập từ xa vào Raspberry thông qua công nghệ VNC Viewer trong môi trường mạng LAN hoặc mạng Internet.

# 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 về yêu cầu bài toán. Tiếp theo, trong chương 3 sẽ trình bày nội dung cài đặt giải pháp của đề tài trên từng phân hệ dựa trên mô tả bài toán từ chương 2.

# CHƯƠNG 3: CÀI ĐẶT GIẢI PHÁP

## 3.1 Cài đặt

Để xây dựng mô hình cần cài đặt:

* Hệ điều hành Raspbian cài đặt trên Raspberry Pi 3 B+.
* Trình soạn thảo văn bản Arduino IDE.
* Phần mềm truy cập máy tính từ xa VNC viewer để điều khiển Raspberry
* Tự động chạy chương trình trên Raspberry khi khởi động
* Sử dụng ngôn ngữ lập trình Python để lập trình trên Raspberry và ngôn ngữ C để lập trình cho Mega.
* Sử dụng mô hình nhận dạng hình ảnh trên YOLO
* Thư viện Rpi.GPIO có sẵn trên hệ điều hành Raspbian.
* Thư viện Tensorflow, Keras, ImageAI và OpenCV và Servo.

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

## 3.2 Lưu đồ giải thuật

### 3.2.1 Phát hiện lửa

* Lưu đồ giải thuật phát hiện lửa bằng cảm biến trên Mega

Hình 3. 1 Thuật toán phát hiện lửa bằng cảm biến

Đúng

Sai

Cảm biến lửa = HIGH

Xe di chuyển thẳng

Xe rẽ sang hướng có lửa

Bật cảm biến lửa

Quá trình hoạt động của xe bắt đầu khi các thiết bị phần cứng được kích hoạt. Do cảm biến lửa 5 kênh có góc đo tương đối rộng, mỗi kênh có một góc đo giá trị riêng nên nếu cảm biến lửa xác định được tia hồng ngoại ở mức cao thì sẽ gửi tín hiệu có lửa về trung tâm xử lý để điều khiển hướng đi của xe. Trong trường hợp còn lại khi không có lửa hoặc đã dập tắt lửa thì xe vẫn di chuyển thẳng như bình thường.

* Lưu đồ giải thuật phát hiện lửa bằng camera trên Raspberry Pi

Đúng

Khởi động Raspberry

Bật chế độ camera

Sai

Camera phát hiện có lửa

Tạo kết nối với Mega

Gửi tín hiệu cho mega

Hình 3. 2 Thuật toán phát hiện lửa bằng camera

Trên hình 3.2 cho thấy biểu đồ luồng hoạt động của xe tự chữa cháy là nhận dạng hình ảnh lửa của camera phần này chủ yếu là hoạt động của Raspberry trong mô hình.

Raspberry Pi đã được cài đặt để có thể nhận dạng hình ảnh lửa trên camera sau đó gửi một chuỗi mang thông tin có lửa về trung tâm xử lý hoạt động của xe. Nếu camera quay được hình ảnh lửa thì Raspberry sẽ tạo một kết nối thông qua cổng truyền thông UART với arduino Mega để gửi tín hiệu có lửa cho Mega thực hiện báo động và di chuyển gần đến vị trí lửa. Ngược lại, trong trường hợp camera không phát hiện được lửa thì các hoạt động trên Mega vẫn diễn ra bình thường.

### 3.2.2 Dập tắt lửa

Hình 3. 3 Thuật toán phát hiện và dập lửa

Cảm biến lửa = HIGH

Khởi động motor bơm nước

Khởi động Raspberry

Bật chế độ camera

Bật cảm biến lửa

Dừng hoạt động của xe

Đúng

Đúng

Sai

Camera phát hiện có lửa

Xe di chuyển thẳng

Bật còi báo động

Xe rẽ sang hướng có lửa

Sai

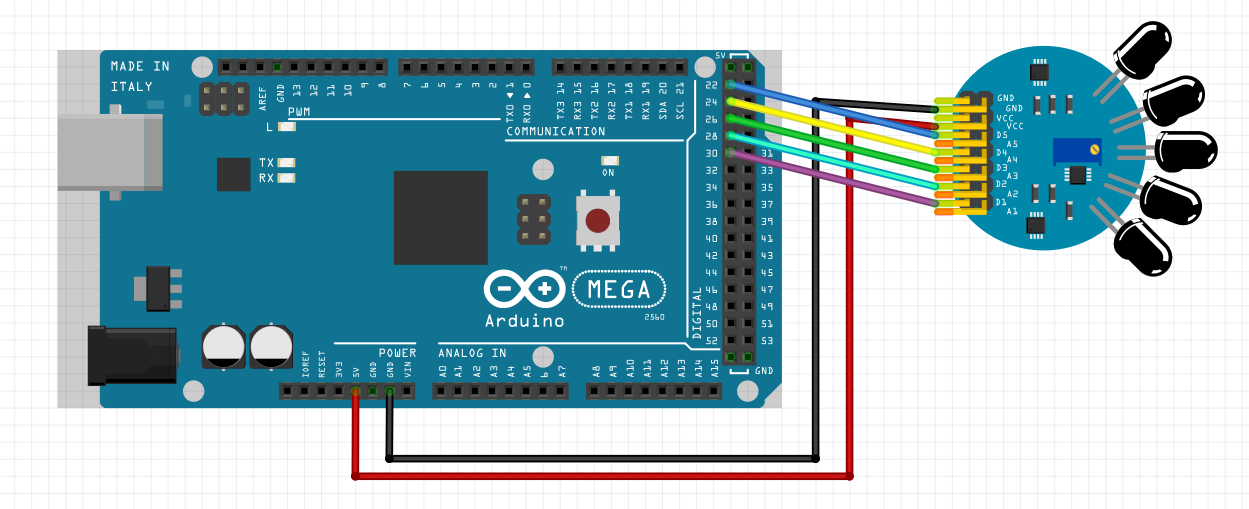
Trên hình 3.3 cho thấy biểu đồ luồng hoạt động của xe tự chữa cháy. Trong giải thuật này chủ yếu là hoạt động điều khiển hoạt động bằng trung tâm xử lý Arduino Mega.

Khi được cấp nguồn xe bắt đầu hoạt động nếu camera quay được hình ảnh lửa và đã được Raspberry kết nối với arduino Mega thông qua cổng truyền thông UART để gửi tín hiệu có lửa cho Mega thực hiện báo động và di chuyển gần đến vị trí lửa. Kết hợp với điều kiện nhận dạng lửa bằng cảm biến để biết được là đang gần vị trí lửa lúc đó Mega sẽ thực hiện dừng xe và bật motor bơm nước dập tắt lửa. Ngược lại, trong trường hợp không phát hiện được lửa thì các hoạt động trên Mega vẫn diễn ra bình thường.

## 3.3 Thiết kế mô hình

### 3.3.1 Kết nối trên Arduino Mega

* **Giải pháp phát lửa bằng cảm biến phát hiện lửa**



Hình 3. 4 Kết nối cảm biến phát hiện lửa

|  |  |
| --- | --- |
| **Mega** | **Cảm biến lửa** |
| 5V | 5V |
| GND | GND |
| 30 | D1 |
| 28 | D2 |
| 26 | D3 |
| 24 | D4 |
| 22 | D5 |

Bảng 3. 1 Sơ đồ chân cắm của Mega và cảm biến lửa

Phần khai báo

Khai báo các chân Digital của cảm biến lửa:

int [Tên\_biến] = [ Chân\_Digital];

**Phần cài đặt**

Thiết lập chân Digital là input

pinMode([Tên\_biến], INPUT);

**Phần thực thi**

​if (digitalRead([ Một trong các chân Digital]) == 1) {

​ // Thực thi các hoạt động khi có lửa

​}

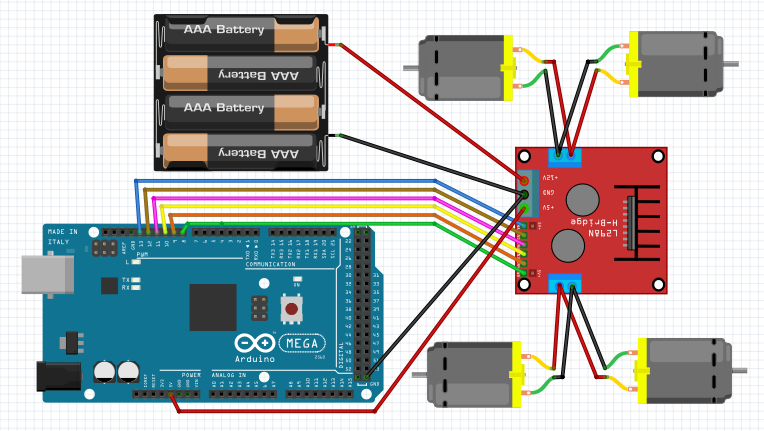
else

​ // Thực thi các hoạt động bình thường

}

Trên hình 3.5 là sơ đồ kết nối của cảm biến phát hiện lửa. Trên cảm biến phát hiện lửa ngoài chân VCC và 5V còn có các chân Analog và Digital trên từng con led của cảm biến lửa. Các chân Digital của cảm biến lửa kết nối với các chân số trên board Mega để cảm biến liên tục đo các giá trị của tia hồng ngoại. Nếu đưa cảm biến đến gần ngọn lửa thì giá trị sẽ tăng dần.

* **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 của Mega với L298N và động cơ giảm tốc

**Phần khai báo**

Khai báo chân cắm:

int [tên\_biến] = [chân\_digital] ;

**Phần cài đặt**

Thiết lập chân sẽ phát tín hiệu:

pinMode([tên\_biến], OUTPUT);

**Phần thực thi**

Điều khiển hoạt động của động cơ giảm tốc:

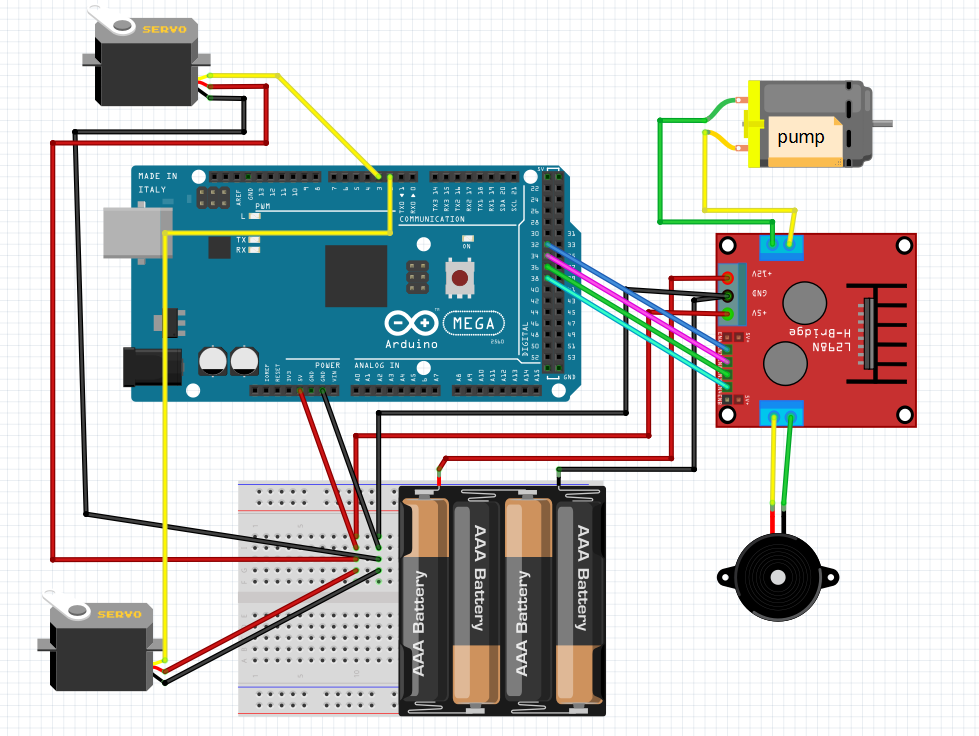
digitalWrite([tên\_biến], HIGH);

Sơ đồ kết nối trên dùng Mega kết nối L298N để điều khiển bốn động cơ giảm tốc. Các động cơ được điều khiển để thay đổi tốc độ nhanh chậm, và các chiều quay như tiến, lùi, rẻ trái, rẻ phải theo yêu cầu.

Nguồn được sử dụng cho động cơ, các cảm biến và board là nguồn pin Lipo 3S 11.1V được cấp vào L298N và board giảm áp để hạ xuống mức 5V cấp cho Mega và Raspberry.

**Cách 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 điều khiển động cơ bơm nước và còi**

****

Hình 3. 6 Kết nối động cơ báo động và dập tắt lửa

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mega** | **L298N2** |  | **Motor bơm nước** | **Còi báo động** | **Servo nước dọc** | **servo nước ngang** |
| 5V | 5V |  |  |  | 5V |  |
| GND | GND |  |  |  | GND |  |
| 3 |  |  |  |  | servonuocdoc |  |
| 4 |  |  |  |  |  | servonuocngang |
| 32 | IN1 |  | OUT1 |  |  |  |
| 34 | IN2 |  | OUT2 |  |  |  |
| 36 | IN3 |  |  | OUT1 |  |  |
| 38 | IN4 |  |  | OUT2 |  |  |

Bảng 3. 3 Sơ đồ chân cắm của Mega, Motor nước và còi

**Phần khai báo**

Khai báo thư viện: Servo.h

Tạo đối tượng để điều khiển Servo

Servo [tên\_đối\_tượng];

Khai báo chân Digital cho servo

int [Tên\_biến] = [ Chân\_Digital];

Khai báo các chân IN của động cơ L298N:

int [Tên\_biến] = [ Chân\_IN];

**Phần cài đặt**

Cài đặt Servo

[Tên\_đối\_tượng].attach([Tên\_biến]);

Thiết lập chân IN của các động cơ là output

pinMode([Tên\_biến], OUTPUT);

**Phần thực thi**

Điều khiển xoay của Servo:

[Tên\_đối\_tượng].write([Vị\_trí\_quay]);

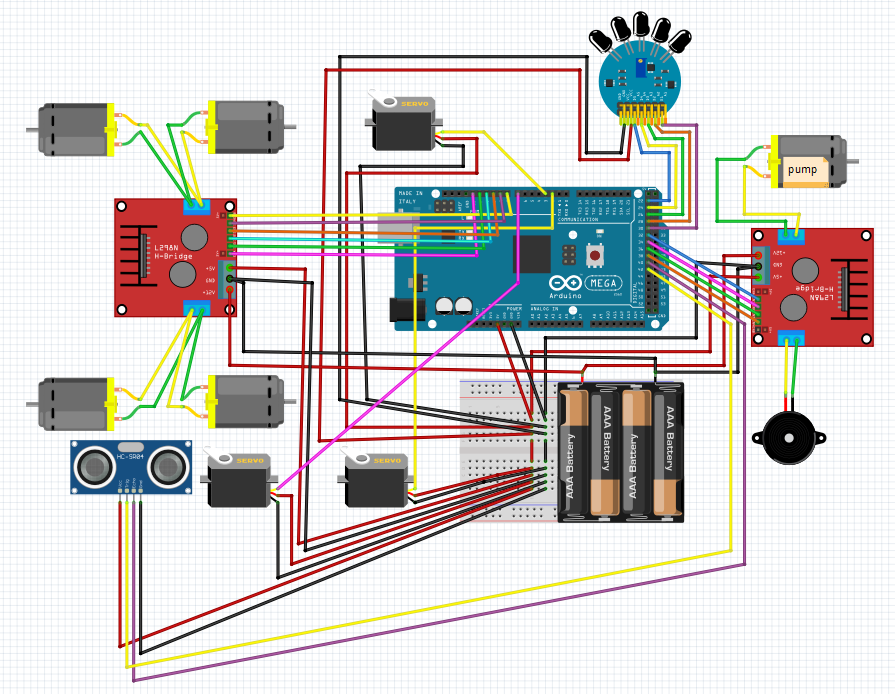
Điều khiển hoạt động của động cơ:

digitalWrite([Tên\_biến], HIGH);

Trong sơ đồ kết nối trên dùng Mega kết nối L298N để điều khiển motor bơm nước và còi báo động giống như việc kết nối của động cơ giảm tốc các thiết bị này cũng cần được cấp cho 1 nguồn với đầu vào 12V và đầu ra cho Mega là 5V.

Motor bơm nước và còi sẽ thực hiện phun nước và báo động khi nhận được tín hiệu có lửa. Hai servo trong sơ đồ được dùng để điều chỉnh hướng phun nước ngang và dọc vào ngọn lửa để dập tắt lửa.

* **Giải pháp toàn mô hình**

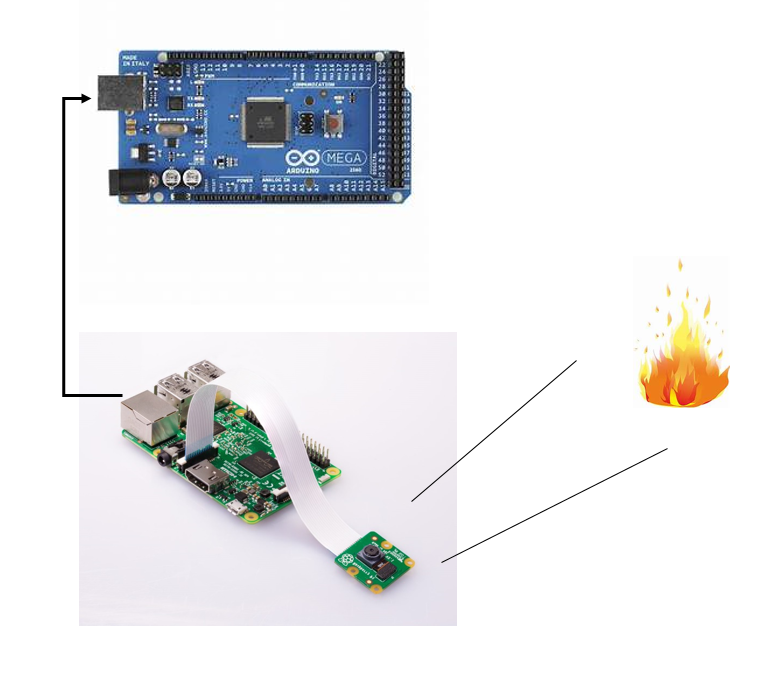
****

Hình 3. 7 Mô hình kết nối tổng quát

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

### 3.3.2 Raspberry kết nối với Mega

* **Giải pháp kết nối Mega và Raspberry**



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

**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){

//thực hiện nhận tín hiệu

}

**Mục đích của việc kết nối:**

Raspberry Pi có khả năng như một chiếc máy tính, có wifi, bluetooth, có thể send và get 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 IOTs 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).

**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 trở nên chính xác và đáng tin cậy hơn.

# 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 và dập lửa thông qua cảm biến và 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 và thành phần 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 Các trường hợp đánh giá và kiểm thử

### 4.2.1 Đánh giá thành phần của mô hình

* **Đánh giá phần cứng**

|  |  |  |
| --- | --- | --- |
| **Tên thiết bị** | **Ưu điểm** | **Nhược điểm** |
| 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 | - 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 phát hiện lửa | -  Tốc độ đáp ứng nhanh và độ nhạy cao  - Có biến trở để tùy chỉnh độ nhạy cảm biến  - Nhỏ, gọn dễ dàng áp dụng trong mô hình | - Dễ bị nhiễu bởi ánh sáng có tia hồng ngoại lớn  - Khoảng cách nhận biết nhỏ |
| Camera | - Bắt được lửa từ xa | - Không thu được hình ảnh ở những góc khuất và những vị trí ngoài khung hình |
| Module L298N | - Có thể tùy chỉnh tốc độ và chiều quay của động cơ  - Giá thành tương đối rẻ  - Hiệu điện thế đầu ra thích hợp để sử dụng cho Arduino | - Không thể cung cấp tốc độ tối đa nguồn điện đầu ra cho động cơ DC |
| Động cơ giảm tốc | - Giá thành rẻ  - Dễ sử dụng | - Chỉ sử dụng cho các mô hình nhỏ |
| Động cơ Servo | - Tốc độ phản ứng nhanh  - Có tích hợp sẳn driver điều khiển động cơ bên trong | - Chỉ sử dụng cho các mô hình nhỏ  - Lực kéo không lớn |
| Motor bơm nước | - Có khả năng chống nước và hoạt động khi ngâm chìm trong nước  - Thiết kế nhỏ, gọn thích hợp cho mô hình  - Không cần phải tốn nhiều năng lượng để hút nước vào bơm. Áp suất nước tự đẩy nước vào buồng máy bơm, do đó “tiết kiệm” rất nhiều năng lượng tiêu thụ | - Động cơ dễ bị hư hổng và ăn mòn do ngâm trong nước thời gian dài |
| Còi báo | - Báo động ngay khi có lửa  - Giá rẻ thích hợp sử dụng trong robot | - Dễ bị hư hỏng do điều kiện môi trường |
| Mạch giảm áp DC LM2596 | - Có thể thay đổi hiệu điện thế ở nhiều mức khác nhau  - Chi phí thấp, độ ổn định cao | - Phạm vi ứng dụng còn hạn chế |
| Pin Lipo | -  Ít bị rò rỉ, hoạt động mạnh mẽ  - Trọng lượng nhẹ, bền, chất điện phân tốt ít khi bị ăn mòn.  - Kích thước phù hợp với mô hình | - Giá thành khá cao  - Không thể tự động sạc |

Bảng 4. 1 Ưu nhược điểm của phần cứng

* **Đánh giá phần mềm**

Các phần mềm được chọn là phiên bản mới và có bổ sung đầy đủ các thư viện, hỗ trợ việc viết chương trình và upload chương trình lên các board một cách tốt nhất, ổn định nhất.

* **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.

### 4.2.2 Kiểm thử chức năng

* **Kiểm thử trên board Mega**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Mô tả dữ liệu**  **kiểm thử** | **Kết quả mong đợi** | **Kết quả thực tế** |
| 1 | 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 được tín hiệu có lửa được gửi từ Raspberry  - 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 được tín hiệu có lửa được gửi từ Raspberry  - Điều khiển hướng đi của  xe đến vị trí lửa  - Thông báo bằng cách bật còi báo động |
| 2 | Cảm biến phát hiện lửa nhận dạng tia hồng ngoại kiểm tra lửa: Có lửa | - Phát hiện được lửa  - Điều khiển hướng đi của xe đến vị trí có lửa | - Nhận dạng được lửa, nhưng việc nhận dạng đôi khi cũng có sai xót tùy thuộc vào vị trí và thời gian nhận dạng trong ngày. Tại những vị trí có điều kiện môi trường thích hợp ít bị ảnh hướng bởi tia hồng ngoại của ánh sáng mặt trời thì mức độ chính xác cao, ngược lại tại thời điểm ánh sáng quá cao thì độ chính xác tương đối thấp  - Điều khiển hướng đi của xe khi cảm biến lửa nhận dạng đúng |
| 3 | Cảm biến phát hiện lửa nhận dạng tia hồng ngoại kiểm tra lửa: Không có lửa | Xe đi thẳng | Xe đi thẳng |
| 4 | Mega nhận được thông tin có lửa từ Raspberry đồng thời cảm biến lửa nhận dạng được có lửa | - Điều khiển hướng đi của xe  - Dừng xe  - Bơm nước dập tắt lửa | - Điều khiển hướng đi của xe tương đối chính xác theo hướng mà cảm biến nhận dạng  - Dừng xe  - Bơm nước dập tắt lửa. tuy nhiên đôi khi ngọn lửa quá lớn xe không đi nước để dập tắt hoàn toàn đám cháy |

Bảng 4. 2 Kiểm thử các tính năng trên board Mega

* **Kiểm thử trên Raspberry Pi**

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

Bảng 4. 3 Kiểm thử các tính năng trên Raspberry Pi

## 4.3 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 hệ thống 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

## 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.
* Xây dựng mô hình xe với các thiết bị cảm biến và 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 mục tiêu đã đề ra là phát hiện và dập lửa thông qua cảm biến phát hiện lửa trên Mega và camera trên Raspberry.
* Mô hình hoạt động tương đối ổn định.
* Về bản thân:
* Tích lũy thêm lượng kiến thức mới không hề nhỏ.
* Học hỏi thêm được những công nghệ, kỹ thuật mới.
* 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.

## 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
* Không vận chuyển được lượng nước quá nhiều trên xe
* Tránh vật cản làm thay đổi đường đi của xe gây ra khó khăn trong việc đi tuần và kiểm tra lửa
* 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.

## 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 văn phòng nơi xe đi qua mà không cần phải đi đến từng nơi.
* 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.
* Nhận dạng có người trong đám cháy để kịp thời thông báo cứu người
* Có thể kết hợp với việc đo mực nước trong bình và tự cung cấp nước khi sắp hết nước nhằm đảm bảo cho công tác chữa cháy.
* Cải thiện đường đi cho xe, cho phép xe hoạt động trên nhiều đường đi khác nhau, xe sẽ tự sạc khi nguồn năng lượng yếu.
* 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 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.

# TÀI LIỆU THAM KHẢO

🙢🕮🙠

**Website tham khả****o**

[1] Arduino Based Autonomous Fire Fighting Robot:

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

[2] Design and Implementation of Flame Sensor and Obstacle Detection for Automatic Fire Fighting Robot:

<http://ijsetr.com/uploads/561423IJSETR16468-48.pdf?fbclid=IwAR3KFkUpj0-3_B2jTWuc327e3FAFexTQD8qsMrVOEtt0Aoo__Bu2yxH5rfs>

[3] FireNet: A Specialized Lightweight Fire & Smoke Detection Model for Real-Time IoT Applications:

<https://arxiv.org/pdf/1905.11922.pdf>

[4] Giao tiếp giữa Arduino và Raspberry Pi (UART):

<https://mechasolution.vn/Blog/bai-10-giao-tiep-arduino-va-raspberry-pi-uart>

[5] Giới thiệu về Arduino mega

<https://www.stdio.vn/articles/printing/542/gioi-thieu-ve-arduino-mega-2560>

[6] Sử dụng cảm biến khoảng cách

<http://arduino.vn/bai-viet/233-su-dung-cam-bien-khoang-cach-hc-sr04?fbclid=IwAR3Q_doWAC1N8WUkiW9bBKxPViOT-KFjnkn9VgXmNi8W8fdQ9mQxh7jH_0I>

[7] Cảm biến lửa – module cảm biến với arduino

<http://arduino.vn/bai-viet/1032-cam-bien-lua-mot-mo-dun-cam-bien-don-gian-voiardunio?fbclid=IwAR1JND4Usygsz8YLDE4BFxwVvOg3WpwKopoIpL1GDMmTiEKL57vTc_IUTEo>

[8] Camera

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

[9] Điều khiển động cơ L298N

<http://arduino.vn/bai-viet/893-cach-dung-module-dieu-khien-dong-co-l298n-cau-h-de-dieu-khien-dong-co-dc?fbclid=IwAR1B2Env7FMFJ1sk0MuRrLxMp4q0mfSSiKkf0fusxeMxWlmM9RkIWuYMGBo>

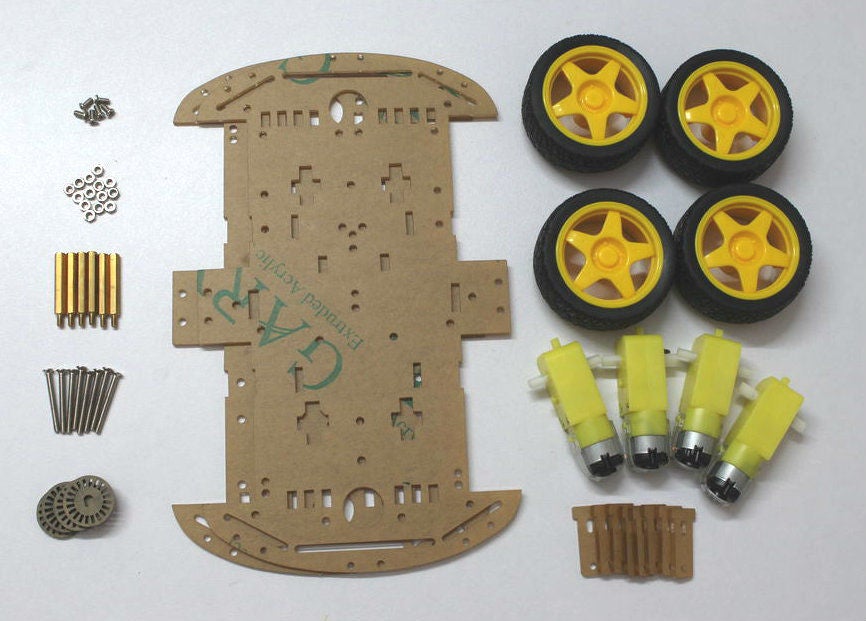
[10] Cài đặt Frameworks để làm Deep Learning.

<https://dlapplications.github.io/2018-06-18-install-TF-with-Anaconda/>

# PHỤ LỤC

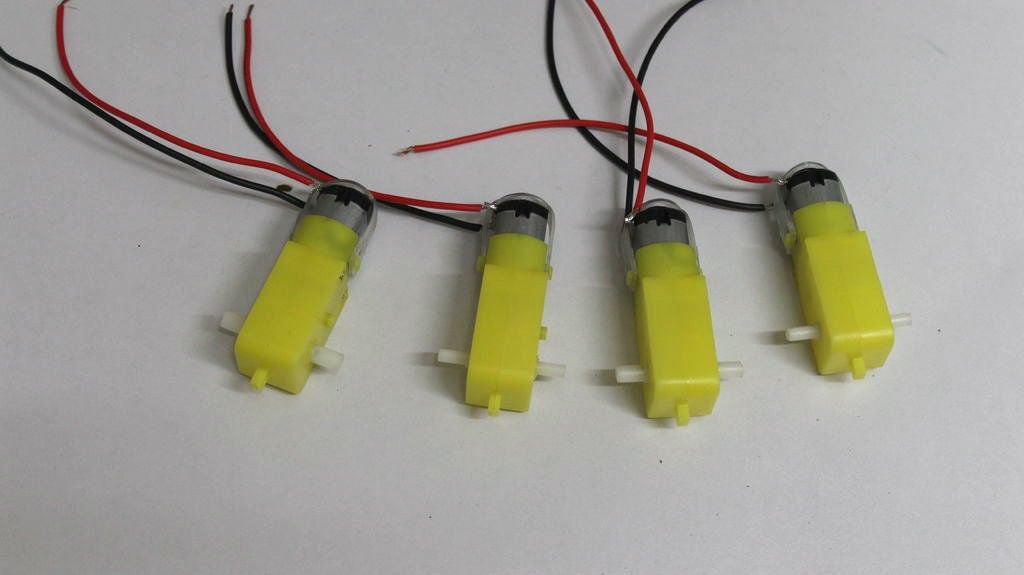
**1. Hướng dẫn lắp đặt khung xe**

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

****

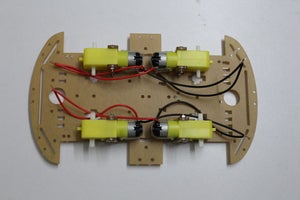
**Hình 1: Thiết bị làm khung xe**

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

****

**Hình 2: Động cơ giảm tốc đã nối dây**

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

****

**Hình 3: Khung xe đã lắp động cơ giảm tốc**

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

****

**Hình 4: Khung xe đã hoàn thành**

**2. Hướng dẫn cài đặt Arduino IDE**

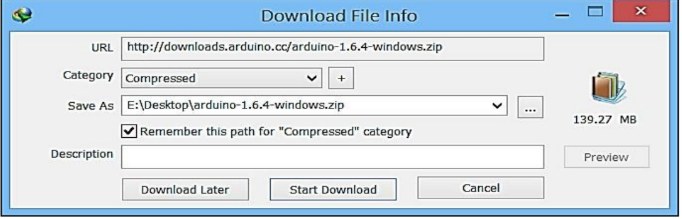
  Cần cài đặt Java Runtime Environment (JRE) trước khi cài Arduino IDE

**Bước 1**: Bước 1: Truy cập địa chỉ http://arduino.cc/en/Main/Software/... Đây là nơi lưu trữ cũng như cập nhật các bản IDE của Arduino. Bấm vào mục Windows ZIP file for non admin install như hình minh họa.

**Hình 5: Nhấp vào Windows Zip file for non admin install.**

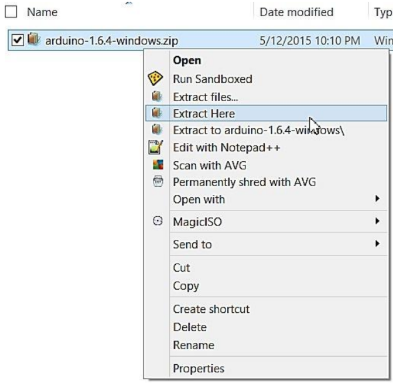
Bạn sẽ được chuyển đến một trang mời quyền góp tiền để phát triển phần mềm cho Arduino, tiếp tục bấm JUST DOWNLOAD để bắt đầu tải.

**Hình 6: Nhấp vào JUST DOWNLOAD để tải phần mềm.**



**Hình 7: Chọn nơi lưu file và bấm Start Download.**

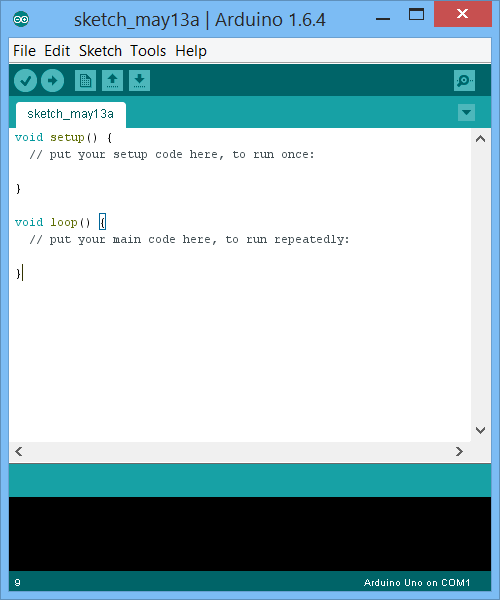
**Bước 2:**Sau khi download xong, các bạn bấm chuột phải vào file vừa download arduino-1.6.4-windows.zip và chọn “Extract here” để giải nén.



**Hình 7:** **Giải nén file vừa tải.**

**Bước 3:** Copy thư mục arduino-1.6.4 vừa giải nén đến nơi lưu trữ.

**Bước 4:** 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-1.6.4\ để khởi động Arduino IDE

[](http://k1.arduino.vn/img/2015/05/13/0/1398_12320-1431518163-0-2015-05-13-18h55-51.png)

**Hình 6: Giao diện IDE**

**3. 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:**

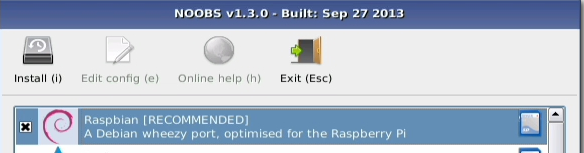
Nếu thẻ SD của bạn mới tinh thì không nhất thiết phải làm bước này. Phần mềm [SD Formatter 4.0](https://www.sdcard.org/downloads/formatter_4/) được khuyên dùng cho việc Format thẻ.

**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. Bạn sẽ thấy một khung lựa chọn tương tự như thế này:



**Hình 7: Giao diện 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: raspberry

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

**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-essential 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,…

* 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 libxvidcore-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 python2.7-dev python3-dev

Lấy mã nguồn OpenCV

* cd ~
* wget -0 opencv.zip <https://github.com/Itseez/opencv/archive/3.0.0.zip>
* unzip 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 \
  + - * -DCMAKE\_INSTALL\_PREFIX=/usr/local \
      * INSTALL\_C\_EXAMPLES=ON \
      * INSTALL\_PYTHON\_EXAMPLES=ON \
      * OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_contrib-3.0.0/modules \
      * BUILD\_EXAMPLES=ON ..

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
* **Tensorflow**

Cài đặt TensorFlow trên Raspberry Pi chạy Raspbian rất dễ dàng. Đầu tiên, cài đặt các thư viện tiên quyết với lệnh sau:

* sudo apt install libatlas-base-dev

Sau đó, cài đặt TensorFlow bằng pip với lệnh sau:

* pip3 install tensorflow
* **Keras**

Chạy câu lệnh sau:

* pip3 install keras
* **ImageAI**

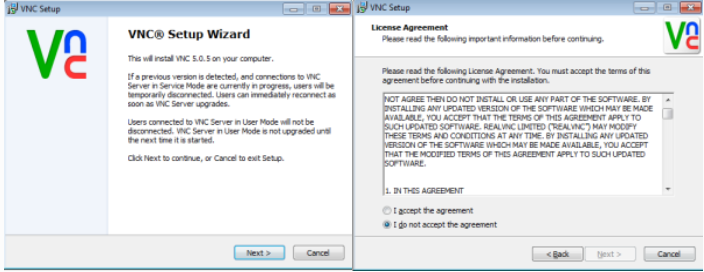
Chạy câu lệnh sau để cài đặt imageAI

* pip3 install imageai --upgrade

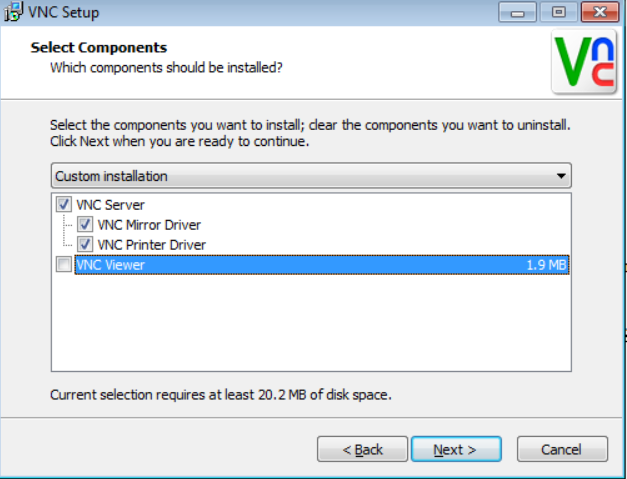
**5. Hướng dẫn cài VNC Viewer**

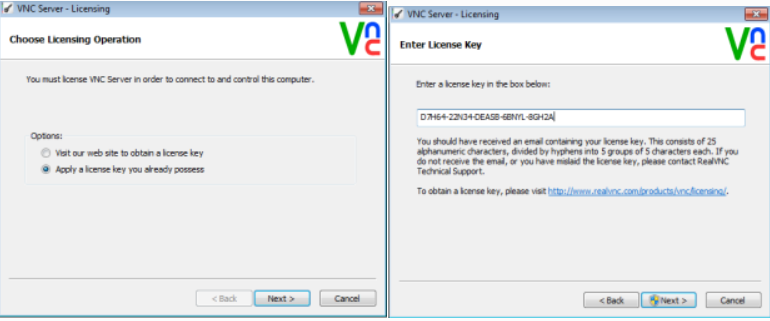
Để cài đặt cần thực hiện tải các gói phần mềm được cung cấp miễn phí tại trang chủ của VNC <http://www.realvnc.com/download/vnc/>

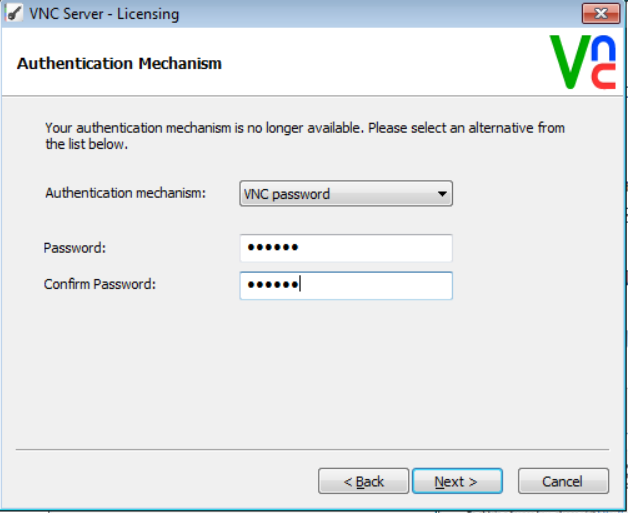
* Tiến hành cài đặt:
* Kích đúp chuột vào tập tin VNC-5.0.5-Windows.exe vừa được tải về.

****

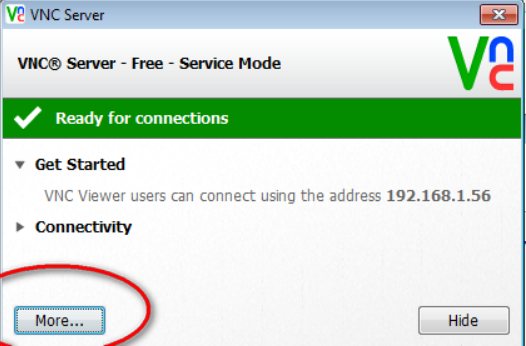
* Kích chọn RUN để thực hiện cài đặt.
* Kích chọn NEXT và chọn I accept the agreement \ NEXT



* Sau khi kết thúc quá trình cài đặt VNC Server yêu cầu chúng ta kết nối Internet để đăng ký License hoặc sử dụng License: K74D2-326FR-CUSD2-R27B2-HR2YA
* Kích chọn Unencrypted connections are acceptable và chọn NEXT
* Nhập mật khẩu cho VNC Server (ít nhất là 1 ký tự) \ NEXT



* **Tiến hành cấu hình – Options:**



**6. Hướng dẫn tự động chạy chương trình trên Raspberry khi khởi động**

**Bước 1: Set permission cho file đó**

* sudo chmod +x file.py

**Bước 2: Sửa file *rc.local***

* sudo nano /etc/rc.local

Thêm vào dòng lệnh sau phía trên dòng**exit 0**

* sudo python /path/file.py > /dev/null 2>&1 &