

Tự làm xe tự hành

Các kiến thức cơ bản để bắt đầu học làm AI, robot

Quyển sổ tay dành cho các bạn bắt đầu tìm hiểu về thế giới robot và những khái niệm đầu tiên để xây dựng cho mình một chú robot tuyệt vời.

*"Để hiệu quả hơn khi sử dụng tài liệu này, mỗi phần lý thuyết tôi sẽ đề nghị các bạn "**Thử làm**" ngay lập tức một vấn đề, điều đó giúp các bạn hiểu rõ hơn các kiến thức khi thực hành".*

Lời mở đầu

Robot, A.I hay IoT là những thứ mà rất nhiều người đang nhắc đến trên truyền hình, mạng xã hội và báo chí. Các bạn sẽ luôn nghe và được nhắc lại nhiều lần những khái niệm thời kỳ 4.0, cách mạng công nghiệp. Thực sự ở thế kỷ 21, kỹ nguyên mà công nghệ sẽ làm thay đổi rất nhiều trong từng ngóc ngách của cuộc sống. Tôi tin rằng kỹ nguyên này Việt Nam sẽ có nhiều thế hệ học sinh tài năng có thể ứng dụng công nghệ, khoa học kỹ thuật để xây dựng đất nước và thế giới. Tất cả những xu hướng mà tôi nhắc đều đều cần những kiến thức rất căn bản đó là học giỏi các môn tự nhiên, đặc biệt là toán. Toán là môn học được đặt trọng tâm trong nền giáo dục Việt Nam, thực tiễn những học sinh Việt Nam thực sự dùng những kiến thức đó vào cuộc sống không nhiều. Những bạn học rất giỏi môn toán, sau khi rời ghế nhà trường, đi làm vẫn thực sự không biết những kiến thức ma trận, tổ hợp hay tích phân được dùng thực tế trong các bài toán thực tiễn nào.

Học sinh Việt Nam cần thực hành, cần liên kết những kiến thức trên sách vở vào thực tiễn, đó chính là mục tiêu tôi viết cuốn sách này. Cuốn sách này sẽ viết theo góc nhìn của tôi khi bắt đầu tìm hiểu về các vấn đề, các hướng gợi mở để các bạn tiếp tục tìm hiểu. Sau đó sẽ có các đề xuất giúp các bạn thực hành. Cuốn sổ tay này được viết năm 2020 khi tôi phát động một cuộc thi cho

các bạn cấp 3 trong cả nước về robot -Vietnam STEAM Challenge. Đây là cuộc thi tôi mong muốn các bạn học sinh sẽ bắt đầu thực hành sớm hơn, bắt đầu tự tay làm các sản phẩm của mình để thực sự ứng dụng kiến thức mà các bạn đã học.

Trong cuốn sách này tôi sẽ viết và sẽ liên tục có bản cập nhật hàng năm cho các bạn theo địa chỉ via.makerviet.org. Tôi mong muốn sau khi đọc xong cuốn sách này bạn sẽ là một maker thực sự. Đây sẽ là khởi đầu cho những sáng tạo của các bạn trong tương lai.

Thân gửi các bạn trẻ

Mục lục

Lời mở đầu	1
Nội dung cuốn sách	5
Phần 0: Robot và thiết bị tự hành ?	6
1. Robot trong thế kỷ 21	6
Phần 1: Dự án mã nguồn mở VIA.....	12
1. Các loại động cơ thông dụng làm robot.....	12
2. Các kết cấu hay và điển hình.....	15
3. Thiết kế cơ khí.....	20
4. Chế tạo và gia công.....	21
Phần 2: Điện tử, mạch máu và giác quan.....	30
1. Nguồn điện.....	31
2. Các loại linh kiện điện tử cơ bản	32
3. Các mạch điều khiển động cơ kinh điển:	33
4. Các loại cảm biến thông dụng	36
5. IC là gì ?	41
6. PCB	42
7. Arduino là gì?	46
8. Mạch Makerbot BANHMI là gì?	47

9. Thủ làm, điều khiển động cơ qua WIFI	50
 Phần 3: VIA và trí tuệ nhân tạo.....	52
Hệ thống trí tuệ nhân tạo cho xe tự lái	52
1. Nguyên lý hoạt động xe tự lái	52
Phát hiện làn đường	56
1. Làm quen với Python và Google Colab.....	56
2. Các thuật toán xử lý ảnh cơ bản.....	58
3. Phát hiện vạch kẻ đường bằng xử lý ảnh cơ bản.....	82
Lập trình điều khiển xe bám làn đường	89
Phát hiện biển báo giao thông.....	97
Triển khai hệ thống trí tuệ nhân tạo trên VIABot.....	107
1. Nạp firmware và kiểm tra phần cứng	107
2. Kiểm tra tín hiệu hình ảnh từ ESP32-CAM	108
 Phần 4: Mô phỏng trong VIA.....	113
1. A.I - Trí tuệ nhân tạo	113
 Phần 5: Thách thức VIA và làm sản phẩm.....	125
Phần 6: Tài nguyên và công cụ học tập	131
Máy in 3D	131
Arduino – Cộng đồng mã nguồn mở.....	137
Máy tính nhúng giá rẻ.....	137
Chế tạo sản phẩm IoT trong 30 phút.....	141

Tạp chí và các diễn đàn, website sáng tạo	153
Phần 7: Robot và định hướng nghề nghiệp	158
Kỹ sư trí tuệ nhân tạo – A.I	158
Kỹ sư phần mềm.....	159
Kỹ sư điện-điện tử	161
Kỹ sư cơ khí	162
Quản lý công nghệ	163
Index.....	165
2	165
3	165

Nội dung cuốn sách

Phiên bản 1.0 đầu tiên cuốn sách này tôi mong muốn sẽ gửi tới các bạn những kiến thức cơ bản nhất thiên hướng kỹ thuật, trong cuốn sách tiếp theo tôi mong muốn cập nhật thêm các góc nhìn về sản phẩm, công nghệ. Qua đó giúp các bạn nâng cao hiểu biết và có nhiều phương án giải quyết vấn đề để tạo cho mình các sản phẩm và giải pháp trong đời sống.

Phiên bản đầu tiên tôi viết liên tục trong 2 tuần và tổng hợp từ những bài viết chia sẻ trước của tôi nên còn rất nhiều thiếu sót. Mong các bạn đọc sẽ góp ý và gửi thư về địa chỉ

Quyển sách này sẽ được tôi cập nhật liên tục hàng năm, các bạn có thể tham khảo địa chỉ <http://via.makerviet.org/> để cập nhật thông tin mới nhất của cuốn sách.

Phần o: Robot và thiết bị tự hành ?

Robot là những cỗ máy do con người tạo ra và có khả năng di chuyển động để làm các nhiệm vụ cho con người. Từ xa xưa đã có những cỗ máy như "trâu gỗ, ngựa máy" trong truyện Tam quốc, hay chú ngựa thành Troy, hay những cỗ máy giúp con người di chuyển các tảng đá nặng hàng tấn... Qua từng cuộc cách mạng công nghiệp, robot lại dần hoàn thiện hơn để trở nên phổ biến. Thay vì cơ cấu đơn giản dựa vào lò xo hay động năng, robot có thể điều khiển chủ động bằng động cơ, các mạch điện tử phát triển, giúp mạch điều khiển robot gọn hơn, có nhiều cảm biến để phản hồi từ trạng thái của môi trường xung quanh. Thời kỳ máy tính internet bùng nổ giúp robot có khả năng linh hoạt và nhiều ứng dụng hơn và thời kỳ 4 thì robot sẽ trở nên thông minh hơn nhờ trí tuệ nhân tạo rất phát triển.

1. Robot trong thế kỷ 21

Thế kỷ 21 là thế kỷ của bùng nổ công nghệ trí tuệ nhân tạo và công nghệ robotic, tự động hóa. Trước kia robot bị rất nhiều giới hạn về khả năng tính toán và các bài toán về trí tuệ nhân tạo hiểu được môi trường, hiểu được con người. Trong giai đoạn đầu thế kỷ 21 robot có rất nhiều điều kiện để bùng nổ.

Tại sao robotic sẽ bùng nổ trong thế kỷ 21 ?

Nhưng năm đầu thế kỷ này trí tuệ nhân tạo phát triển bùng nổ và ứng dụng trong mọi lĩnh vực của cuộc sống. Trong 5 năm gần đây hàng loạt các trợ lý ảo khắp nơi ra đời với nhiều ngôn ngữ khác nhau: Siri – Apple, Alexa – Amazon, Cortana – Microsoft, Google Assistant, Bixby – Samsung,... Các bài toán nhận dạng hình ảnh, không gian, vật thể cũng đang rất phát triển. Robot dễ dàng di chuyển đi lại và hiểu được môi trường xung quanh nó. Robot không chỉ hoạt động cố định trong các nhà máy công xưởng thay con người, mà dần dần nó sẽ trở thành một người trợ lý, người bạn trong cuộc sống. Lực lượng nhân công giảm, dân số già đi, giá nhân công tăng lên. Nhu cầu thay thế robot ở những công việc đơn giản như lễ tân, bán hàng, bảo vệ, lao động chân tay,... sẽ tăng dần. Khi chi phí lao động tăng trên toàn cầu là cơ hội cho các công ty robotic đưa ra các sản phẩm và dịch vụ cho xã hội.

Cuộc sống càng hiện đại con người lười hơn. Nhiều robot chuyên biệt phục vụ cuộc sống từ robot lau nhà, hút bụi, gấp quần áo, nấu cơm, rửa bát,... Các nhu cầu chuyên biệt sẽ được các công ty giải quyết và đưa ra thị trường một cách nhanh chóng khi xuất hiện các nhu cầu và công nghệ đáp ứng được sản phẩm. Các robot thông dụng nhất trong cuộc sống giảm thời gian lao động của con người như hút bụi, gấp quần áo:

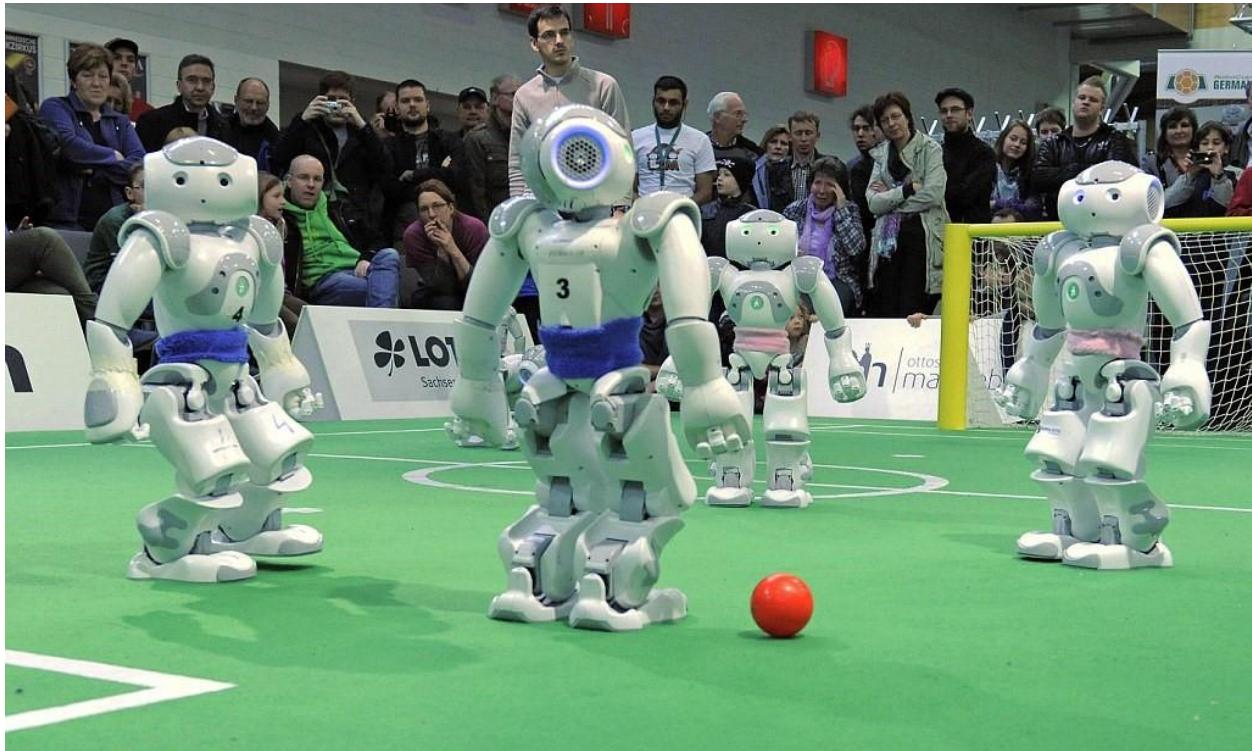


Mục tiêu của các nhà khoa học tại Robocup là 2050 robot sẽ đá bóng thắng con người. Ngày nay các trí tuệ nhân tạo đã thắng được con người trong các môn cờ vua, cờ vây (Alpha Go), nhưng môn suy nghĩ chưa có vận động, hoạt động độc lập. Bóng đá là một môn thể thao khó, cần vận động, chiến thuật và phối hợp, sự linh hoạt khéo léo. Robot có thể đạt được ngưỡng đó trong 30 năm tới sẽ là sự phấn đấu rất lớn của các nhà khoa học để có thể chiến thắng được con người. Đó cũng là tuyên ngôn khẳng định 30 năm nữa robot sẽ rất thông dụng và linh hoạt trong cuộc

sóng

chúng

ta.



Các hãng lớn đang đầu tư rất nhiều vào Robotic và cuộc chiến của tự hành như thế nào ?

Google



Amazon – Một dịch vụ mới trong 2019 được Amazon đưa ra là RoboMaker. Một nền tảng cloud xây dựng trên mã nguồn mở ROS. Giúp các công ty robot giảm cấu hình thiết bị tại robot, xây dựng môi trường nhận dạng, dịch vụ cần cấu hình mạnh trên cloud.

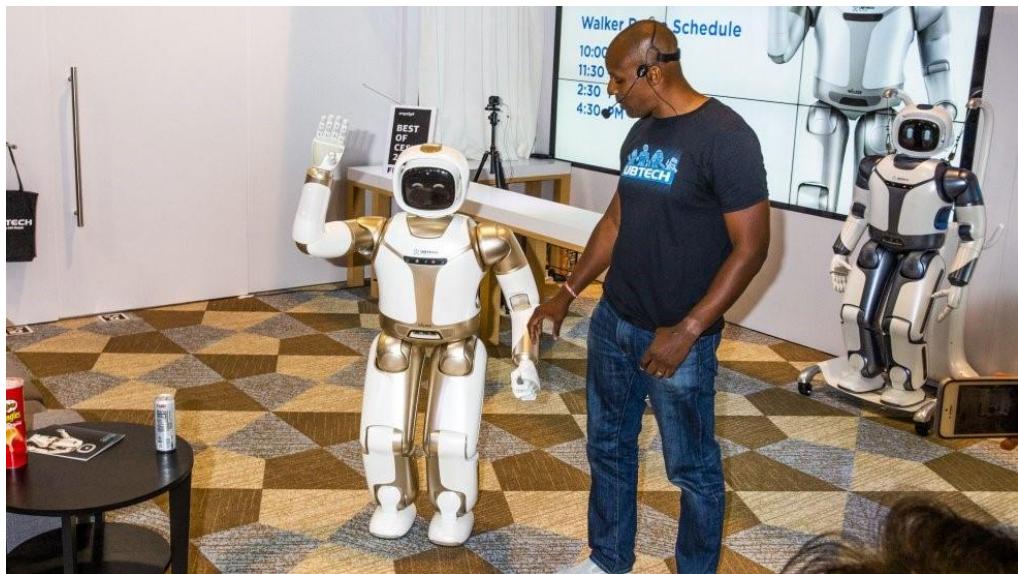
Softbank với dự án đầu tư mua NAO – Aldebaran, mới đây nhất họ lại mua công ty Boston Dynamics từ Google. Họ quyết tâm đưa robot thành một dịch vụ trong gia đình, ngài Son chủ tịch Softbank cho rằng cùng AI, Robotic sẽ là mũi nhọn tiên phong giúp Softbank thành tập đoàn toàn cầu.



Baidu với dự án mã nguồn mở Apollo xây dựng nền tảng xe tự hành cho cả thế giới. Họ có thể giúp một công ty không có kinh nghiệm về xe tự hành làm được xe tự hành. Tương lai Trung Quốc sẽ có rất nhiều nhà máy, công ty bán xe tự hành nhưng không có một kỹ sư trí tuệ nhân tạo nào.



Tencent với rất nhiều dự án đầu tư vào robotic. Đặc biệt trong tại CES 2019 công ty robot mà họ đầu tư vào đã cho ra một chú robot đáng người có tên là Walker như là Asimo.



Kỷ nguyên robot chắc chắn sẽ bùng nổ trong thế kỷ 21 và đi vào từng hộ gia đình. Robot sẽ trở thành người bạn, người cộng sự của chúng ta trong cuộc sống. Câu hỏi của chúng ta là 10 – 20 -30 năm robot sẽ trở nên rẻ và thông dụng như điện thoại thông minh ? Mong rằng các bạn

đọc sẽ tự có một dự đoán cho mình. Robot sẽ là sản phẩm tổng hợp của 4 cuộc cách công nghiệp và thay đổi căn bản cuộc sống của con người trong thế kỷ 21.

Thử làm: Kiếm thử hai động cơ nhỏ và tự làm cho mình một chú robot chuột mini từ con chuột máy tính . Robot sẽ tự động chuyển hướng khi gặp chướng ngại vật, giống các con robot hút bụi. Các bạn có thể tìm dự án PC mouse robot hoặc “làm robot từ chuột hỏng” hoặc theo link <https://www.instructables.com/id/PC-Mouse-Robot/>

Chia sẻ: Đây là dự án thú vị, tôi đã tự làm từ năm lớp 11, khi đó chưa có một kiến thức gì về điện tử và robot. Qua một chương trình truyền hình là Robocon, tôi đã khá bị thu hút bởi những chú robot chạy trên tivi. Tại sao những con robot có thể chạy được theo chiến thuật của những anh sinh viên trong từng trận đấu. Khi đó tôi có được đọc một bài trên echip làm robot chuột – thời đó công cụ tìm kiếm rất kém, không kiểu như các bạn giờ google là ra. Tôi rất khó khăn để tìm đủ linh kiện cho dự án chuột robot này. Nhưng thực tế sau khi làm xong dự án này tôi đã thấy thực sự hiểu hơn về một chú robot. Cách làm cho một con chuột máy tính di chuyển. Và đó là bắt đầu hành trình tôi tìm hiểu những chú robot sau này.

Phần 1: Dự án mã nguồn mở VIA

Cuộc cách mạng công nghiệp lần thứ 1 bắt đầu từ thế kỷ 18, xuất hiện ở nước Anh với hàng loạt các máy móc thay thế sức lao động của con người. Các máy dệt, máy kéo, tàu hỏa động cơ hơi nước, tàu thủy, ... đã lần lượt ra đời trong cuộc cách mạng này và giải phóng sức lao động của con người. Cách mạng công nghiệp lần thứ nhất diễn ra từ thế kỷ XVIII đến XIX ở châu Âu và Mỹ. Đó là thời kỳ mà hầu hết nông nghiệp, xã hội nông thôn đã trở thành công nghiệp và đô thị. Ngành công nghiệp sắt và dệt, cùng với sự phát triển của động cơ hơi nước, đóng vai trò trung tâm trong Cách mạng Công nghiệp. Đây là thành phần quan trọng nhất để phát triển một chú robot vững chắc và linh hoạt.

1. Các loại động cơ thông dụng làm robot.

Động cơ tạo nên sự khác biệt của robot với các sản phẩm như máy tính, điện thoại. Giúp robot chuyển động theo nhiệm vụ của chúng ta cần. Robot thường sẽ dùng các động cơ 1 chiều để dễ dàng di chuyển bằng PIN, robot không phù hợp với điện xoay chiều hay điện 3 pha.

a. Động cơ DC:

Động cơ DC là loại động cơ thông dụng nhất cho các dòng robot hiện nay. Điện áp thông dụng thường từ 3V – 24V. Các động cơ thường nhỏ từ ngón tay cái (dùng trong tay game để tạo rung) đến cơ bắp đùi. Dòng điện chạy từ 100 mA đến vài chục A. Tùy theo nhiệm vụ mà chúng ta chọn công suất phù hợp.

Động cơ tùy theo nhiệm vụ mà sẽ có các đặc tính và tốc độ khác nhau. Nhìn hình dưới các bạn có thể thấy một loại có hộp số và một loại thì không. Tùy kết cấu cơ khí mà ta chọn loại động cơ phù hợp, có nhưng cỡ cấu có giảm tốc và không có nhu cầu lớn về lực chúng ta có thể bỏ qua hộp số.



Động cơ dùng cho điều khiển để robot thường có hộp số và có thêm encoder (bộ đo tốc độ gắn phía sau động cơ). Phần này các bạn có thể tìm hiểu thêm ở chương 2.

Để điều khiển động cơ chúng ta cần có các mạch Driver để điều khiển tốc độ chứ không thể điều khiển trực tiếp từ microcontroler. Các bộ driver sẽ giúp chúng ta điều khiển tốc độ và đảo chiều động cơ. Một điểm rất đơn giản là để đảo chiều động cơ DC chúng ta chỉ việc thay đổi chiều dòng điện đi trong động cơ, đảo cực nguồn là xong, mạch driver sẽ giúp chúng ta chuyển này theo lệnh của phần mềm nhúng.

Thử làm: hãm từ động cơ ? mua một động cơ 1 chiều nối 2 dây động cơ vào nhau, các bạn thử tìm cách quay trực động cơ. Rất khó đúng không. Đây là cách mà khi lập trình động cơ chúng ta hay sử dụng để cho robot dừng, hay hệ thống như thang máy dừng mà không tốn năng lượng bên ngoài.

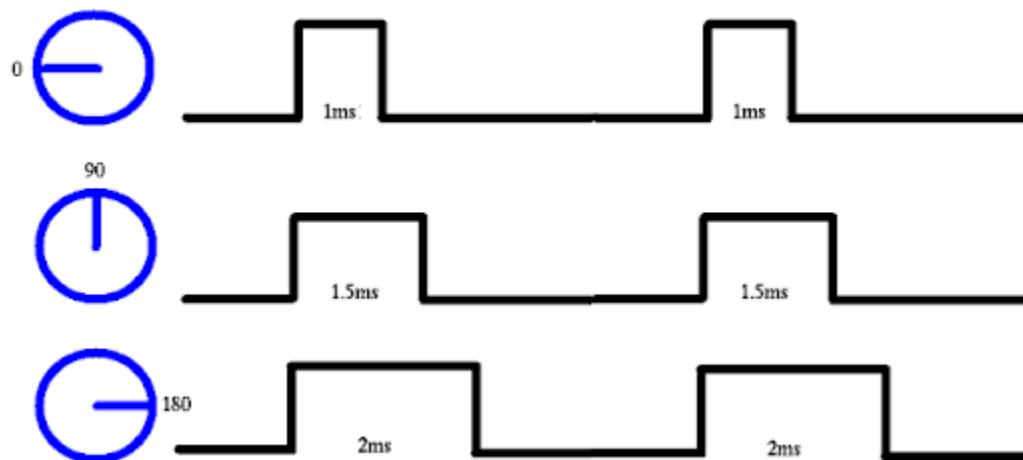
b. **Động cơ Servo**

Động cơ Servo là một động cơ DC có thêm phái hồi vị trí và có một mạch điều khiển bên trong. Động cơ servo là một loại động cơ điều theo chuẩn đã được quy ước cho tất cả các hãng. Các động cơ servo đều có cách điều khiển tương đồng trừ một số loại đặc biệt điều khiển qua bus truyền thông.

Động cơ servo sẽ có 3 chân, một chân nguồn VCC màu đỏ ở giữa, chân đất GND màu nâu và chân tín hiệu điều khiển – màu vàng. Động cơ servo được thiết kế để để quay các góc từ 0° – 180° theo nhu cầu sử dụng. Có một số loại servo điều khiển liên tục, khi đó nó có thể quay tục mà không điều khiển theo góc, mà chỉ điều khiển chiều quay, hay nói ngắn gọn là servo 360° .



Hình ảnh động cơ servo



Điều khiển servo bằng xung

Ứng dụng: Các động cơ này ứng dụng nhiều trong các máy bay mô hình, oto mô hình, các chi tiết cần điều khiển góc chính xác. Dân DIY rất thích loại động cơ này vì đơn giản trong điều khiển và chế tạo nhanh sản phẩm.

c. Động cơ bước

Động cơ bước - Step – motor là một loại động cơ một chiều được điều khiển khá đặc biệt. Thông thường động cơ này sẽ được điều khiển bằng 4 dây điều khiển và 2 dây nguồn đất VCC , GND. Loại động cơ này dùng ở các thiết bị cần điều khiển bước nhỏ, có công suất lớn như máy CNC, máy in 3D,...

d. Động cơ BLDC

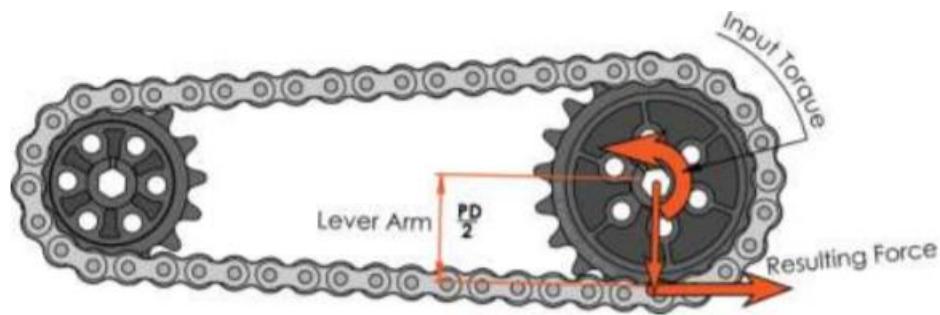
Động cơ BLDC là một loại động cơ DC khá phổ biến trong các robot có công suất lớn. Đây là động cơ không chổi than, hiệu năng hoạt động tốt hơn so với động cơ DC thông thường như phương pháp điều khiển lại phức tạp hơn rất nhiều. Nếu bạn nào đi sâu vào ngành tự động hóa sẽ được học rất sâu về cách điều khiển những loại thiết bị như thế này.

Ứng dụng: những máy bay drone, xe mô hình, robot tự cân bằng, xe điện hay dùng loại động cơ này.

2. Các kết cấu hay và điển hình

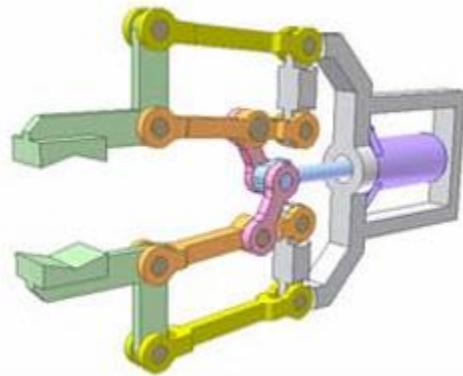
a. Kết cấu xích

Kết cấu xích là một cơ cấu được dùng nhiều trong chế tạo robot, đặc biệt những chi tiết cần yếu tố chắc chắn, tốc độ vừa phải. VD như cơ cấu nâng hàng trong các nhà máy. Trong quân sự, xe tăng được sử dụng với cơ cấu xích giúp di chuyển được nhiều địa hình phức tạp, tuy nhiên tốc độ không cao. Xe đạp là một ví dụ dễ hình dung nhất về kết cấu xích. Cơ cấu này giúp chúng ta thay đổi tốc độ xe, một chiếc xe địa hình có thể có tới 21 cấp tốc độ khác nhau dựa vào điều chỉnh cơ cấu xích, giúp người sử dụng có thể thay đổi xích theo địa hình dốc, đường phố khác nhau. Khi cần lên dốc, đĩa phía người đạp to và phía còn lại sẽ nhỏ, mình đạp nhanh nhưng tốc độ chậm, lực rất lớn, leo dốc rất tốt. Khi cần phóng nhanh, đĩa phía mình nhỏ, phía còn lại to, lực đạp sẽ rất nặng.



Tip: các bạn có thể dễ dàng kiểm cơ cấu xích cũ từ các bộ nhông xe máy thải ra ở cửa hàng sửa xe. Xích sẽ giúp liên kết rất tốt các cơ cấu cơ khí.

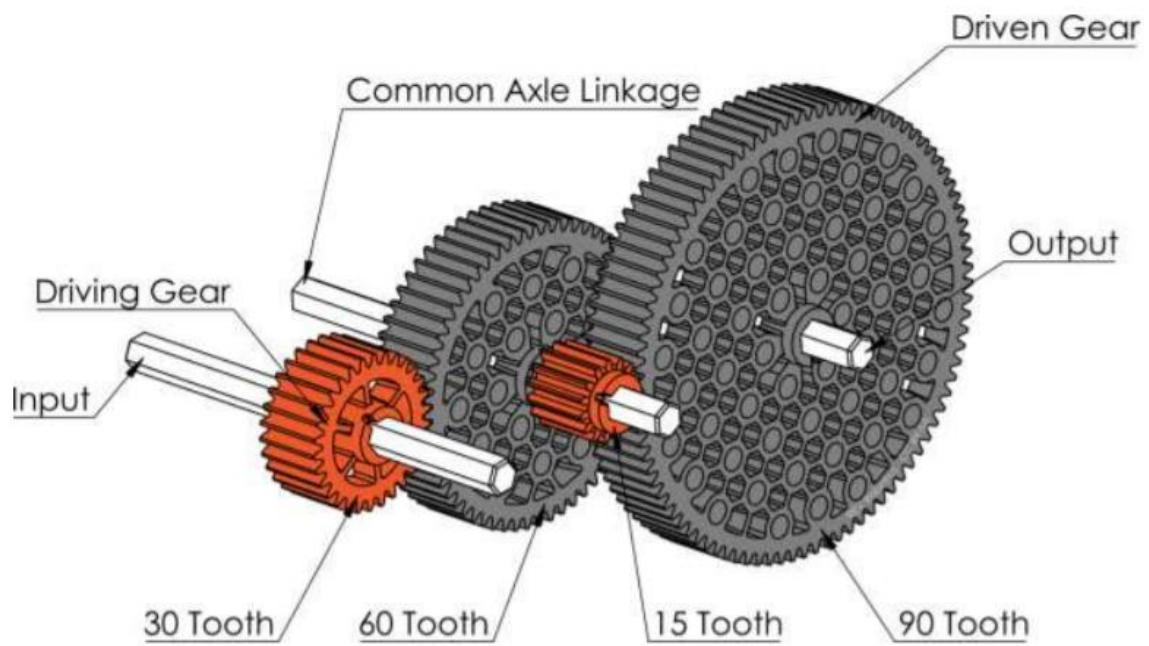
Cơ cấu trục vít: lợi dụng chuyển động quay để tạo ra các chuyển động tịnh tiến. Một mô hình cánh tay kẹp dùng trục vitme.



Ứng dụng: nhiều trong các máy CNC hay máy in 3D đều sử dụng trục vitme để di chuyển đầu gia công. Phương pháp này tạo độ chính xác cao và thiết kế gọn gàng.

b. Kết cấu bánh răng

Kết cấu bánh răng – gear – Đây là kết cấu điển hình giúp sáng tạo trong các sản phẩm Robot. Các robot thường phải chế tạo các bánh răng chuyên biệt để phù hợp với tải và công suất thiết bị. Việc tìm bánh răng phù hợp là rất khó khăn đối với việc tự chế tạo, hay làm lẻ. Thường chúng ta hay mua các động cơ có sẵn hộp số, có thông số phù hợp với chức năng. Tuy nhiên nhiều cơ cấu vẫn cần phải thêm những bánh răng bên ngoài như tay kẹp . Hay các chi tiết cơ khí liên động giữa các phần.



Tham khảo cách tính gear từ Rev Robotic bit.ly/2RwrOxE

Ứng dụng: thiết kế tay kẹp chữ M dùng bánh răng.



c. Các cơ cấu chấp hành khác

Động cơ là cơ cấu chính của các chú robot. Ngoài ra các hệ thống robot có sử dụng các cơ cấu khác, nhưng có 2 loại điển hình nhất tôi giới thiệu với các bạn dưới đây.

Khí nén là mô hình dùng áp lực của không khí để tạo ra các chuyển động cho robot. Thông thường các thiết bị cho khí nén khá đắt và yêu cầu độ an toàn cao. Thiết bị cần thiết là một bình chứa bụi được áp lực, hệ thống ống khí và các pittong để dẫn động từ lực tạo ra từ khí nén. Các bạn sẽ phải đầu tư máy bơm khí (giống máy bơm ở các cửa hàng xe đạp, xe máy). Thời xưa 2005 trong các robot chúng tôi dùng chai cocacola 1.5L để chế tạo ra các bình chứa khí (chịu được tầm 9-10kg). Các chi tiết cần nhanh trong một thời điểm cố định thì chúng tôi dùng giải pháp này. Đối với các hệ thống tự chế thì kết cấu này thường chỉ sử dụng một lần. Sau khi dùng xong thường phải kết nối máy bơm khí để tạo lại áp lực



Thử làm: tự làm một tên lửa nước sử dụng chai coca-cola 1.5l, Các bạn sẽ thử nén khí và kết hợp với nước để tạo áp lực giúp đẩy "tên lửa" của chúng ta lên. Đây là một thí nghiệm khá phổ biến tại các nước trên thế giới. Thường xuyên có cuộc thi về tên lửa nước cho học sinh và trẻ em trên khắp thế giới.Tìm kiếm wiki "tên lửa nước"



Lò xo: là một kết cấu cơ khí rất đơn giản, xuất hiện từ rất lâu, trước thời kỳ xuất hiện động cơ. Lò xo hay dây chun, đều là cách chúng ta tích lũy năng lượng vào để sử dụng lại một thời điểm. Trong kết cấu robot, chúng ta có thể dùng tay, gài cơ cấu có lò xo, hoặc dùng động cơ quay để giài (tuy nhiên phải chọn động cơ quay chậm, có hộp số tốt, vì lực kéo lò xo rất lớn). Tại các cuộc thi robot, hay các robot tự chế, các bạn thường dùng tay kéo lò xo và dùng một động cơ để nhả chốt. Từ đó giúp robot có thể thêm nhiều năng lượng nén khi hoạt động. Điển hình nhất là trò chơi bắn súng cao su – các bạn chỉ cần một đoạn dây cao su và một thanh gỗ, các bạn có thể tự chế cho mình một đồ chơi

Thử làm: Tự làm một khẩu súng lò xo, thay vì súng cao su, có thể tận dụng bút bi để tự chế bằng lén cò (ép lò xo) và bắn viên đạn nhỏ.



3. Thiết kế cơ khí

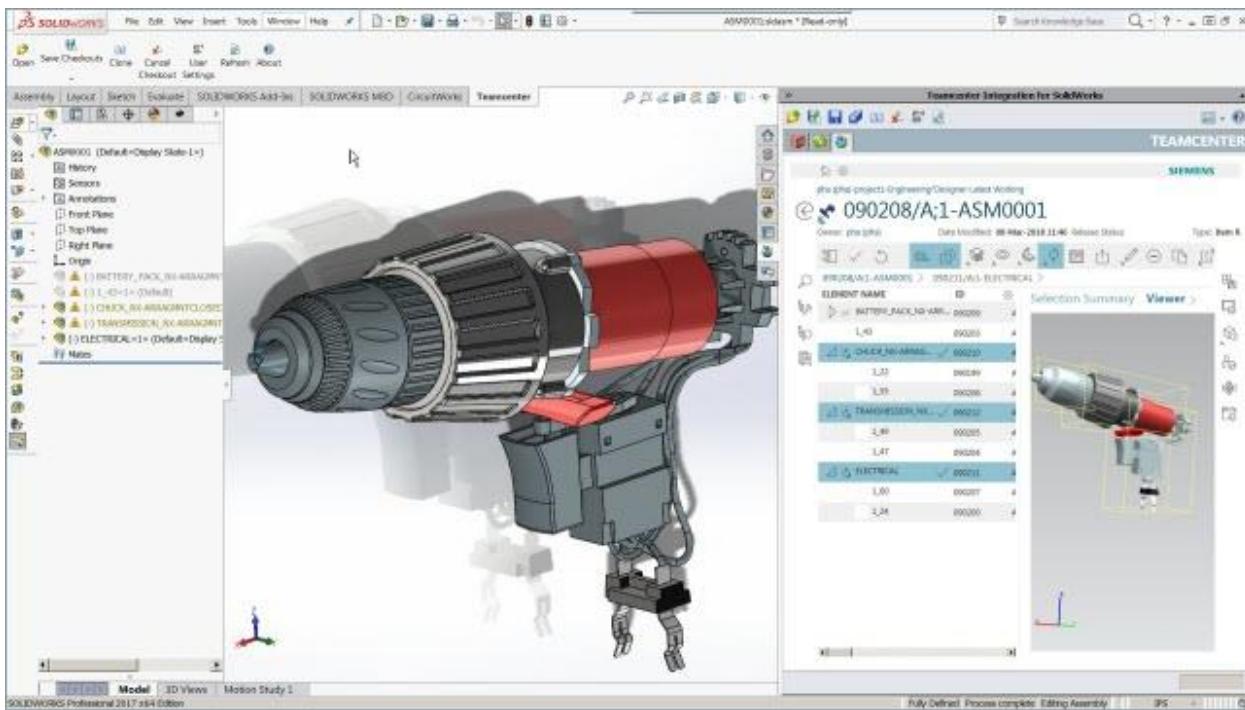
Để có thể tạo ra các chú robot thường chúng ta sẽ cần có một phần mềm để thiết kế các chi tiết để có thể tính toán trước các kết cấu hoặc gia công. Thường quá trình làm chúng ta sẽ vẽ bằng tay để trao đổi ý tưởng sau đó sẽ dùng các công cụ để vẽ lại. Tạo ra bản vẽ rất quan trọng trong quá trình làm robot để có thể gia công đặt hàng các chi tiết nhỏ. Nhưng quan trọng hơn là tính toán trong tâm và mô phỏng hoạt động của robot để đảm bảo tính khả thi của sản phẩm sau khi thực hiện. Đối với các bạn mới chế tạo robot, việc vẽ nháp rất quan trọng, có thể bỏ qua bước vẽ máy và thử chế tạo ngay giúp các bạn học thêm nhiều bài học trước khi làm trên máy.

a. Các công cụ vẽ phổ biến hiện nay

AutoCad – của hãng AutoDesk là phần mềm rất nổi tiếng cho vẽ 2D, gần như các bản vẽ thiết kế 2D đều sử dụng phần mềm này. Ngoài ra hãng cũng có một sản phẩm miễn phí cho giáo dục như Fusion 360. Hãng AutoDesk cũng có khá nhiều các sản phẩm cho STEM, giúp học sinh có thể dễ dàng tạo các thiết kế phức tạp từ bản vẽ nháp của mình thông qua phần mềm TinkerCad

Inventor lại một sản phẩm khác của AutoDesk dành cho vẽ, mô phỏng sản phẩm 3D và cạnh tranh trực tiếp với Solidwork trên toàn cầu.

SolidWork – Là phần mềm thiết kế 3D khá nổi tiếng và thông dụng. Đây là sản phẩm được dân cơ khí sử dụng rất nhiều để thiết kế.

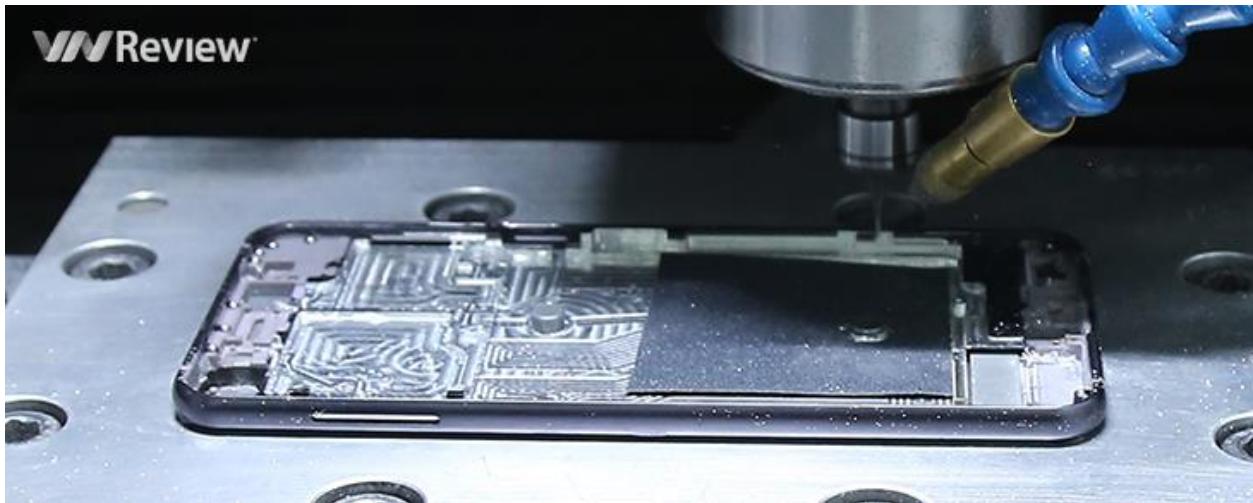


Ngoài **Solidwork** hay **Inventor** là các phần mềm vẽ và mô phỏng 3D thì có phần mềm như Catia cũng được lưu hành nhiều một số cộng đồng tại Việt Nam

4. Chế tạo và gia công

a. Chế tạo bằng máy công cụ

Sau khi các bạn thiết kế bằng các phần mềm 3D, các bạn sẽ phải xuất thành các tài liệu thiết kế 2D để mang tới các xưởng gia công. Thông thường tại các xưởng sẽ không có máy tính hay các công cụ phần mềm 3D cao cấp. Các bạn sẽ phải vẽ hình 2D có ghi chú rõ kích thước gia công. Các chi tiết đó sẽ phải tách lẻ ra để thợ có thể dùng các phôi có sẵn để chạy máy. Các máy gia công thông dụng như máy CNC, máy cắt laze,... Các vật liệu gia công chủ yếu là nhựa hay nhôm nguyên khối. Ảnh dưới là một chi tiết nhôm của vỏ điện thoại di động được gia công:



b. Chế tạo bằng máy in 3D

Máy in 3D là một cuộc cách mạng trong làm robot và sản xuất thử các khuôn mẫu.

Máy in 3D là một dạng máy công cụ giống các máy CNC truyền thống, giúp người dùng có thể dễ dàng tạo ra những sản phẩm được thiết kế từ những hình vẽ 3D. Ngày nay máy in 3D đang trở nên thông dụng trong các gia đình, các văn phòng trên thế giới. Chúng được áp dụng vào nhiều lĩnh vực khác nhau trong cuộc sống từ in mẫu thử cho sản phẩm, in đồ trang trí, in đồ dùng cá nhân (giá kệ điện thoại), ... thậm chí in cả thực phẩm.





IN nhà cho kỹ sư xây dựng, in bánh cho gia đình , in giày cho nhà thiết kế giày.

Máy in 3D có rất nhiều mã nguồn mở nhưng có 2 dòng thông dụng nhất

Dòng máy prusa dạng casatarian (trục đòn tay các x y)



z)



Dòng máy Delta (trục cánh tay song song)

Giá thành để tự chế các dòng máy in 3D này từ 4 - 8 triệu VND.

Ngoài ra còn khá nhiều dòng máy thương mại hóa tầm trung như Ultimaker, MakerBot giá từ 50 - 80 triệu đồng.



Cách để tạo ra một sản phẩm in 3D ?

Có các bước cơ bản như sau:

1. Dựng hình 3D bằng các phần mềm 3D

Phần mềm cơ khí chính xác: Inventor, Solidwork, NX9, NX10,....

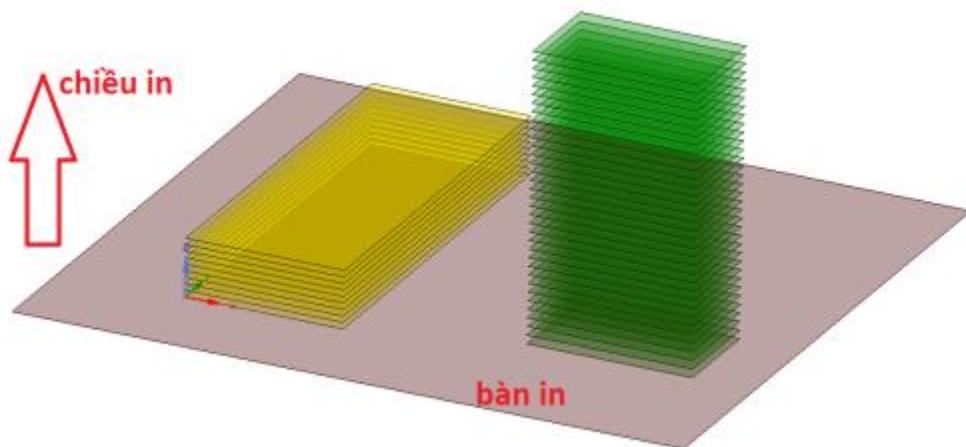
Phần mềm có tính chất nghệ thuật: 3Ds Max, SketchUp, Maya,.....

2. Xuất file 3D ra định dạng *.stl hoặc *.obj (nên để dưới dạng stl)
3. Đưa file *.stl hoặc *.obj vào phần mềm xử lý in 3D (Cura, SLic, Makerbot....)
4. Từ phần mềm xử lý in 3D save ra định dạng GCODE mà máy in 3D hiểu được
5. Chép file GCODE vào USB (thẻ nhớ) và cắm vào máy in để in 3D

Nếu bạn không có khả năng thiết kế có thể lấy các mẫu in 3D sẵn trên trang web Thingiverse
<https://www.thingiverse.com>.

Các bước trên khá đơn giản, tuy nhiên khi sử dụng máy in 3D các bạn sẽ phải lưu ý khá nhiều vấn đề cần lưu ý.

Hướng in từ dưới lên: Mức độ dính chặt lại phụ thuộc vào 3 yếu tố trong quá trình cài đặt phần mềm in 3D (nhiệt độ, bề dày lớp in, tốc độ). Điều này có thể can thiệp trong quá trình chạy máy in, tuy nhiên, bạn phải giữ trong đầu những ý niệm về "hướng in 3D".



Biểu diễn 2 mẫu thiết kế giống nhau nhưng được sắp đặt theo 2 hướng in hoàn toàn khác. Các vân bên hông tượng trưng cho các lớp in 3D. Độ bền của mẫu in 3D màu vàng vượt trội so với mẫu màu xanh.

Để mẫu in 3D được đẹp, đúng kích thước, tiết kiệm vật liệu – thời gian in – chi phí, bạn cần đảm bảo các vấn đề sau:

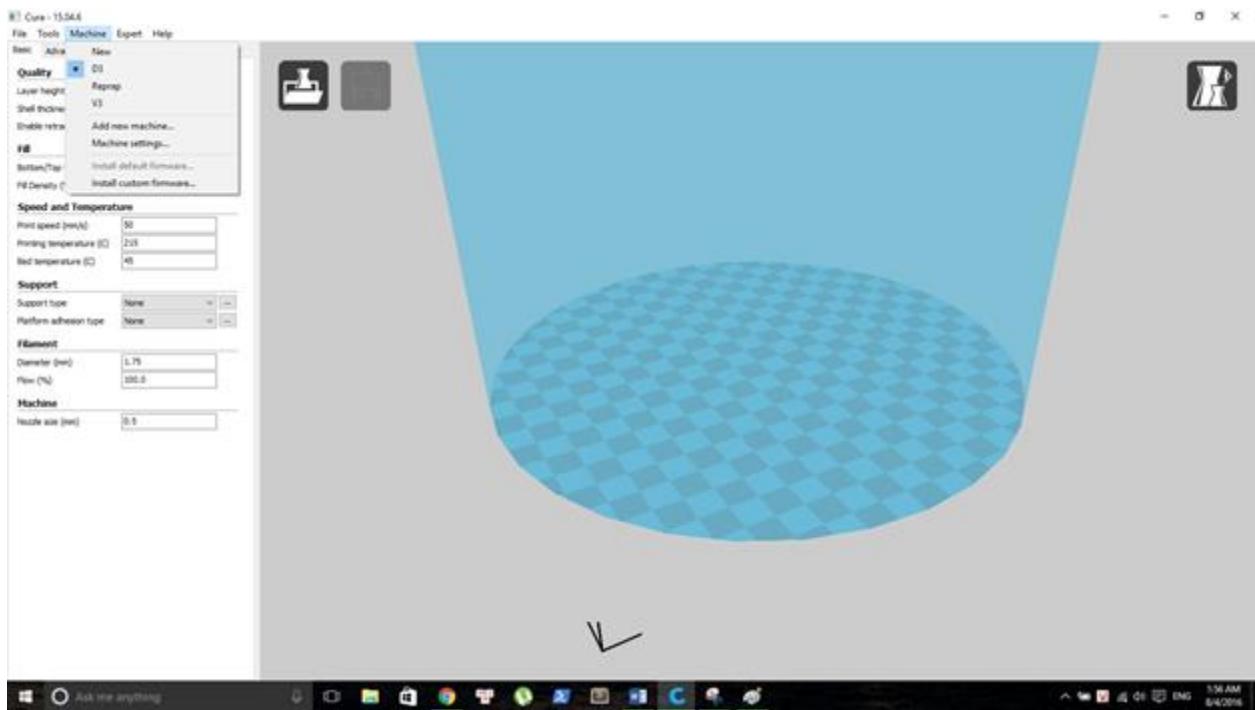
- Thiết kế mô hình theo kiểu "kim tự tháp" tức là dưới to trên nhỏ.
- Nên có một mặt để phẳng bên dưới mô hình.
- Hạn chế các bị trí mỏng hơn 1,2mm.

- Các phần quá bé trên mô hình 3D (0,1-1mm): mắt, mũi, tai, gờ, nút bấm,... rất khó hoặc không thể in 3D!
- Các phần nhô ra nên có góc nghiêng >45 độ so với phương ngang. Hạn chế phần nhô ra nằm ngang, hoặc phía dưới trống không (ví dụ như cây cầu)!
- Nên khống chế mô hình nằm vừa khổ in của máy in 3D, cũng đừng nên quá bé (không in được hoặc in ra xấu!)
- Các chi tiết có lắp ghép thì khoảng cách giữa 2 bề mặt nên để : Lắp lỏng $\geq 0,4$ mm; Lắp chặt $\leq 0,2$ mm.
- Chú ý tới độ phân giải của mô hình khi xuất ra file STL
- Chắc chắn về kích thước file STL là theo hệ inch hay mm!
- Mở lên xem lại file STL/OBJ vừa xuất ra. Hoặc dùng công cụ kiểm tra lỗi file 3D

Phần mềm tạo file chạy máy in 3D

Cura là phần mềm cài đặt thông số để in mô hình 3D và dễ sử dụng nhất. Cura là 1 phần mềm mã nguồn mở được phát triển bởi hãng Ultimaker. Giúp cho việc in 3D, tạo mẫu nhanh 3 chiều trở nên dễ dàng hơn, chất lượng sản phẩm tạo ra cũng tốt hơn. Cura đi kèm với 1 chương trình cài đặt thân thiện giúp bạn luôn luôn cập nhật phiên bản mới nhất với những tính năng mới.

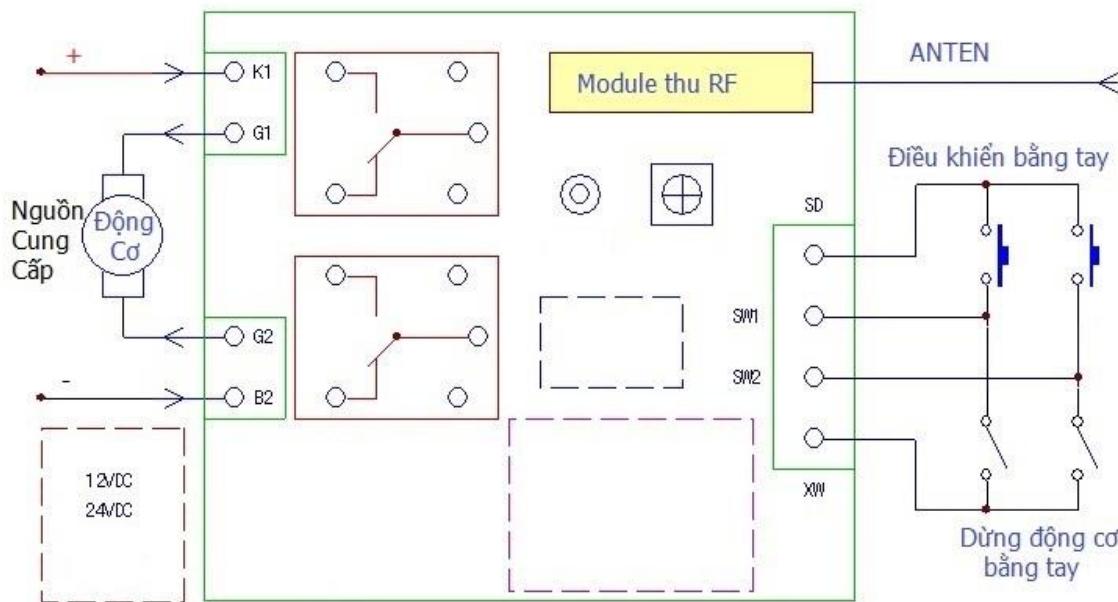
Cura hỗ trợ rất nhiều dòng máy in 3D theo công nghệ FDM Giao diện thân thiện, tính năng mạnh mẽ, thuật toán xử lý mô hình 3D nhanh và chuẩn xác, thao tác đơn giản, Cura là sự lựa chọn tốt nhất cho người mới bắt đầu sử dụng máy in 3D.



Máy in 3D ngày càng trở nên thông dụng, trong thế kỷ 21 có thể mỗi gia đình sẽ có một chiếc máy in 3D để in những vật dụng hàng ngày, những công cụ học tập cho trẻ em, đồ làm bếp cho mẹ, đồ làm việc sửa chữa cho bố. Hiện nay máy in 3D đang trong giai đoạn phát triển rất mạnh để cải tiến các vấn đề về tốc độ, vật liệu, độ tiện lợi và dễ dàng cho sử dụng.

Thử làm: Tự làm một chân đế robot điều khiển bằng tay có người ngồi lên được Dùng mạch driver là các role công suất cao Mạch này cấu tạo rất giống công tắc cầu thang. Tuy nhiên sẽ được điều khiển bằng tín hiệu điện. Bạn có mua các bộ thu phát RF, để điều khiển không dây, nối các tín hiệu điều khiển để bật tắt role . Tự tìm kiếm “điều khiển động bằng 2 role và rf”





Chia sẻ: Cơ khí là một môn không hề đơn giản như trò chơi xếp hình hòi nhỏ của chúng ta. Các bạn thích và đam mê nhưng bộ xếp hình, sáng tạo các hình khối thực sự phù hợp với ngành này. Khi tôi bước chân vào ngưỡng cửa đại học, việc đầu tiên là tìm hiểu làm thế nào để làm một chú robot. Tôi đã đi tìm hiểu và tìm được một số đồng đội cùng lớp tự động hóa để cùng thành lập một đội robot từ lúc không biết gì. Chúng tôi bắt đầu tự tìm hiểu để phát triển một cái đế robot. Thực sự không hề dễ dàng như chúng ta tưởng là chỉ cần mua động cơ về, gắn mạch điều khiển và tay cầm là xong. Mùa hè năm đầu tiên ở Bách Khoa, chúng tôi mò các ngóc ngách chợ trời để tìm các linh kiện phù hợp. Góp tiền mua nhôm, mua động cơ, đi cắt nhờ đồ. Làm thế nào để gá con động cơ vào khối nhôm ? Làm thế nào để gá bánh xe vào đầu động cơ ? Một loạt các vấn đề đặt ra, mọi thứ đều không tương thích với nhau, bánh xe thì không bắt vừa, động cơ thì tròn không vừa nhôm thì vuông,... Cuối cùng cả mùa hè đầu tiên với một cái xưởng cả lũ đã loay hoay tìm được các giải pháp để gá các cái đế của robot và điều khiển cho nó chạy, thậm chí là ngồi lên trên để lái. Thực sự rất vui với mô hình robot đầu tiên có sự góp sức cả một đội. Năm đó là hè 2004, thời mà linh kiện điện tử, vi điều khiển, robot là một thứ rất mới lạ với sinh viên Việt Nam.

Ngày nay để làm một chú robot như vậy khá đơn giản, các bạn có thể mua sẵn rất nhiều thứ khác với cách đây 16 năm chúng tôi phải tự chế và gia công nhờ rất nhiều chi tiết. Các bạn thử tự làm cho mình một chú robot to sẽ khác nhiều với chú robot nhỏ như các đồ chơi.

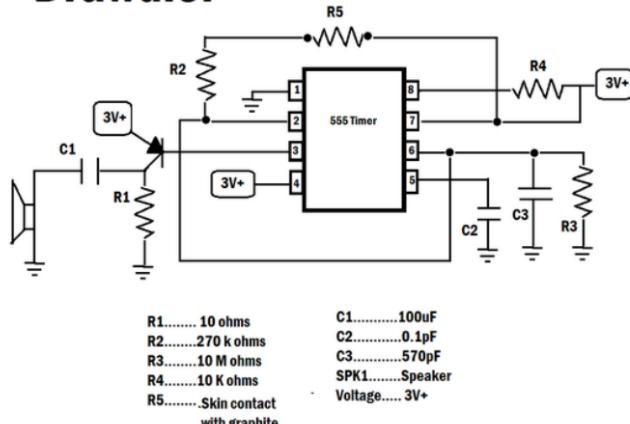
Phần 2: Điện tử, mạch máu và giác quan

Cuộc cách mạng công nghiệp lần thứ 2 trong tâm là các thiết bị điện tử, tự động hóa. Cuộc cách mạng này bắt đầu vào khoảng thập kỷ 1860, khi các tiến bộ kinh tế và kỹ thuật có được nhờ phát triển điện tín, điện thoại, đường sắt và việc áp dụng dây chuyền sản xuất hàng loạt. Đến cuối thế kỷ 19, động lực của Cách mạng công nghiệp lần 2 chủ yếu là động cơ đốt trong và máy móc sử dụng điện. Năm 1914, năm bắt đầu Thế chiến thứ nhất, giai đoạn thứ hai này kết thúc. Trong tâm cuộc cách mạng này là nước Mỹ, với các tập đoàn tiên phong là Ford, với ứng dụng công nghệ mới nhất về điện tử, tự động hóa, robot Ford đã vượt qua các ông lớn khác về xe hơi tại Mỹ để trở thành một doanh nghiệp toàn cầu cho tới ngày nay. Đây là thời kỳ đặt nền móng cho ngành robot đi vào trong các nhà máy.

Thử làm: bút vẽ biết chơi nhạc - "Drawdio" - Musical Pencil



Drawdio:



I.

1. Nguồn điện

Nguồn điện là thành phần quan trọng nhất của robot hay bất cứ mạch điện tử nào. Nguồn đảm bảo sự hoạt động ổn định của thiết bị. Rất nhiều nguyên nhân lỗi, sự cố do chất lượng nguồn không đảm bảo. Thông dụng chúng ta thường dùng các loại Pin AA, AAA cho các thiết bị điện tử gia dụng, đồ chơi, robot thường sẽ không dùng các nguồn năng lượng này được.

Thời kỳ trước đây để làm robot, chúng tôi thường mua acquy từ các loại dành cho oto và xe máy thông thường. Dòng có thể lên tới 1Ah -7Ah tùy mô hình ứng dụng. Các bạn có thể dễ dàng chọn và tìm mua. Phần lớn chỉ có ắc quy 12V, nếu có nhu cầu cao hơn như 24V thì phải đấu nối tiếp 2 ắc quy



Thời kỳ tiếp theo khi các xe oto mô hình, máy bay tự chế Drone ra đời, các loại pin mới xuất hiện và phổ biến hơn. Các dòng thiết bị công suất lớn thường dùng các dòng Pin Li-po để đảm

bảo công suất lớn trong cùng một thời điểm, dải điện áp đa dạng hơn. Các bạn sẽ có nghe khái niệm xS (2S,3S,4S,5S) tương ứng $2 \times 3.7V \sim 7.4V$ hay $4 \times 3.7V = 14.8V$.



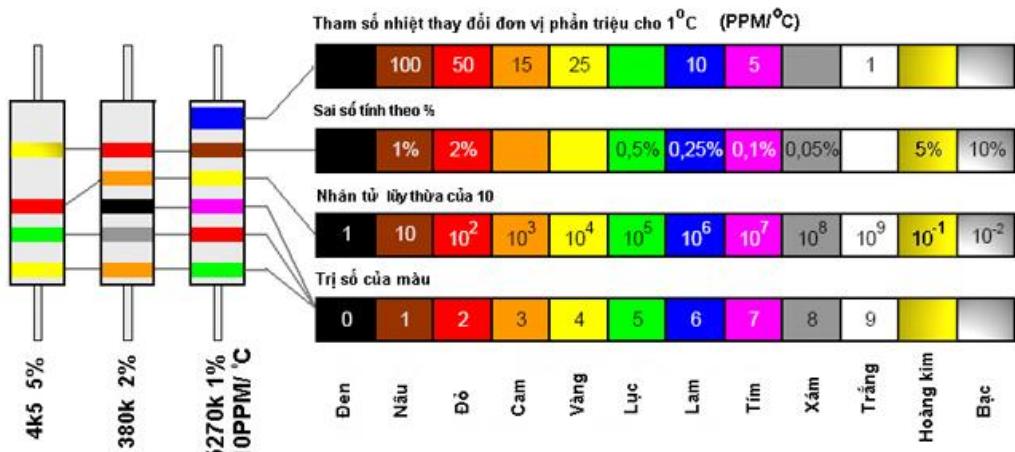
Đây gần như là loại pin tương đối phổ biến, với công suất lớn và kích thước nhỏ gọn, Tuy nhiên cũng khá nguy hiểm do dễ cháy nổ và bị phồng pin do điều kiện bảo quản và sử dụng. Các bạn phải lưu ý tránh để 2 đầu đỏ - đen chạm, với công suất lớn sẽ gây cháy nổ và dễ dàng hỏng các thiết bị điện tử.

2. Các loại linh kiện điện tử cơ bản

Linh kiện phân loại làm 2 loại dán và linh kiện cắm, ngày nay phần lớn các linh kiện trong các thiết bị điện tử đều là linh kiện dán. Các linh kiện cơ bản như điện trở (R -Ohm), tụ điện, diode, led (điốt phát sáng), transistor, cuộn cảm. Ngoài ra có các loại IC chức năng hay các đầu jump cắm.

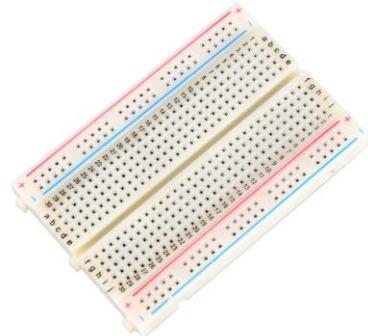
Các bạn nên hiểu cách đọc thông số của các loại linh kiện điện tử này.

Cách đọc thông số của linh kiện trớ cắm bằng mã màu :



Các bạn có thể tự tìm kiếm cách đọc giá trị trở dán hay cách đo giá trị các linh kiện này bằng đồng hồ điện tử. Tuy nhiên khi mua hoặc thiết kế mạch thường chung ta hay để một cuộn riêng và có ghi chú bên ngoài.

Transistor là một linh kiện khó hiểu với các bạn lúc bắt đầu làm điện tử, mặc dù kiến thức về nó được học rất nhiều trên trường. Các bạn hãy tưởng tượng đây đơn giản là một khóa điện tử, công tắc điện tử giúp nối thông dòng điện bằng các tín hiệu từ IC hoặc mạch xung nhịp. Khi cần chúng ta sẽ đóng hoặc mở. Nó được phân 2 loại PNP và NPN tùy theo các chúng ta mở trên hay mở dưới.



Bảng cắm linh kiện – breadboard: đây là cách gần như rất phổ biến cho các bạn bắt đầu học về điện tử. Thủ ghép các linh kiện điện tử thực tế trước khi thiết kế một PCB trên thực tế là cách dân điện tử cũng hay làm để thử nghiệm những sản phẩm mới. Bảng này được nối ngầm các dây ngầm phía dưới: 4 hàng phía ngoài, 2 hàng mỗi bên được nối dọc với nhau, thường sử dụng nối khối nguồn. 10 hàng ở giữa theo trực chính thì được nối ngang theo từng cột. Ở khe giữa thường cắm các IC chức năng hoặc vi điều khiển.

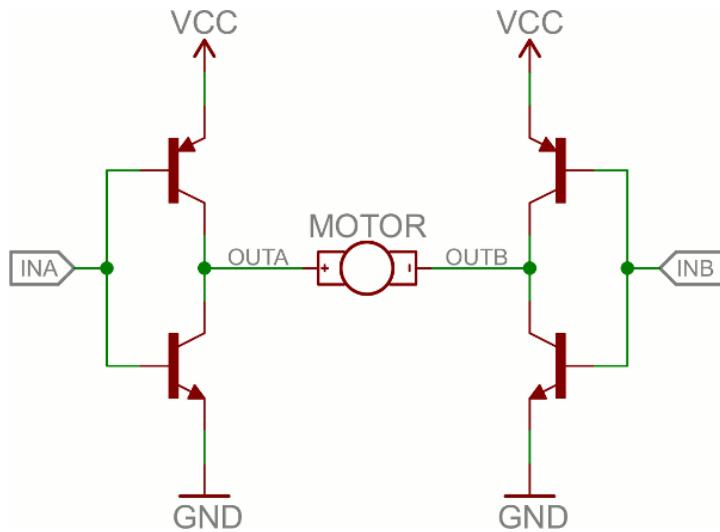
3. Các mạch điều khiển động cơ kinh điển:

Trong các linh kiện cơ bản, trong mạch điều khiển động cơ hay sử dụng các transistor công suất để điều khiển đóng mở động cơ. người ta có tên gọi khác là MOSFET (thuộc dòng FET) hoặc

dùng trực tiếp các mạch role để điều khiển động cơ. Tuy nhiên với các động cơ công suất lớn, sẽ rất hạn chế dùng Role bởi dòng điện cao, nếu lúc chuyển mạch cần thời gian delay lớn để dừng động cơ trước khi chuyển mạch. Ngày nay các công ty đã tích hợp rất nhiều linh kiện trong một IC công suất để có thể điều khiển động cơ trực tiếp

a. Mạch cầu H

Mạch cầu H: một mạch kinh điển được sử dụng 4 con FET để điều khiển tốc độ và chiều quay của một động cơ. Mạch này cần thiết kế và tính toán thông số phù hợp cho từng công suất khác nhau. Thời gian nhưng năm 2000 mạch này khá phổ biến trong một số nơi nghiên cứu về điều khiển động cơ và robocon. Tuy nhiên việc tính toán và xây dựng một mạch ổn định là một thách thức đối với người mới bắt đầu

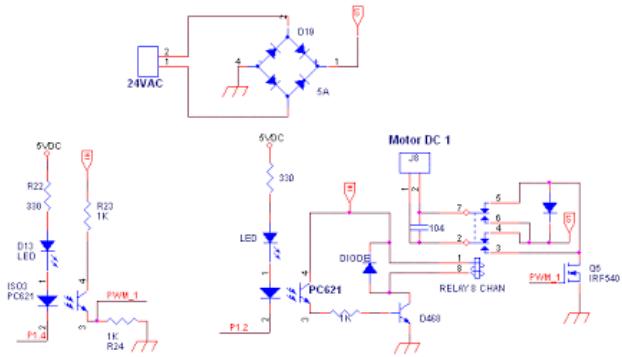


Logic mạch trên khá đơn giản, có 4 chế độ điều khiển, 2 chế độ hãm từ khi cùng đấu 2 đầu động cơ thông nguồn VCC và đất GND. Hai chế độ còn lại sẽ bật hai công tắc điện tử chéo nhau để đảo nguồn cung cấp cho động cơ.

b. Mạch IRF – Role

Mạch IRF – Role: Đây là mạch khá phổ biến cho những người mới bắt đầu làm việc với động cơ. Mạch dễ kiểm linh kiện, dễ chế tạo, tuy nhiên mạch này có nhược điểm là không chuyển sang chế độ hãm từ được. Mạch có một FET dùng IRF540N dùng để cấp nguồn GND cho động cơ và điều khiển tốc độ, đầu nguồn VCC sẽ được nối vào Rơ le. Động cơ sẽ được nối với chân chung

của role, role sẽ có nhiệm vụ đảo chiều động cơ bằng cách đảo cực nguồn VCC và GND. Có một điểm cần lưu ý khi sử dụng mạch này là không được bật tắt role trong lúc IRF đang điều khiển động cơ chạy



c. Mạch IC tích hợp

LM298 là mạch tích hợp điều khiển động cơ khá phổ biến trên thị trường Việt Nam, ngoài điều khiển động cơ DC, IC này có thể điều khiển được cả động cơ bước. Mạch này có thể điều khiển đồng thời 2 động cơ cùng một IC. Lưu ý gắn tản nhiệt ne



MC33887 là một IC dán được cộng đồng quốc tế dùng nhiều, tuy nhiên mạch này cần tích hợp nhiều chân điều khiển từ vi điều khiển. Ưu điểm lớn nhất của nó có phản hồi dòng, chế độ ngắn bảo vệ tự động khi quá dòng

M2 OUT2
M2 OUT1
GND
VIN
M1 OUT1
M1 OUT2



D2 (M2 PWM input/disable when low)
D1 (M2 disable)
IN2 (M2 input 2)
EN (M2 enable)
FS (M2 fault status)
IN1 (M2 input 1)
FB (M2 current feedback)
GND
FB (M1 current feedback)
IN1 (M1 input 1)
FS (M1 fault status)
EN (M1 enable)
IN2 (M1 input 2)
D1 (M1 disable)
D2 (M1 PWM input/disable when low)

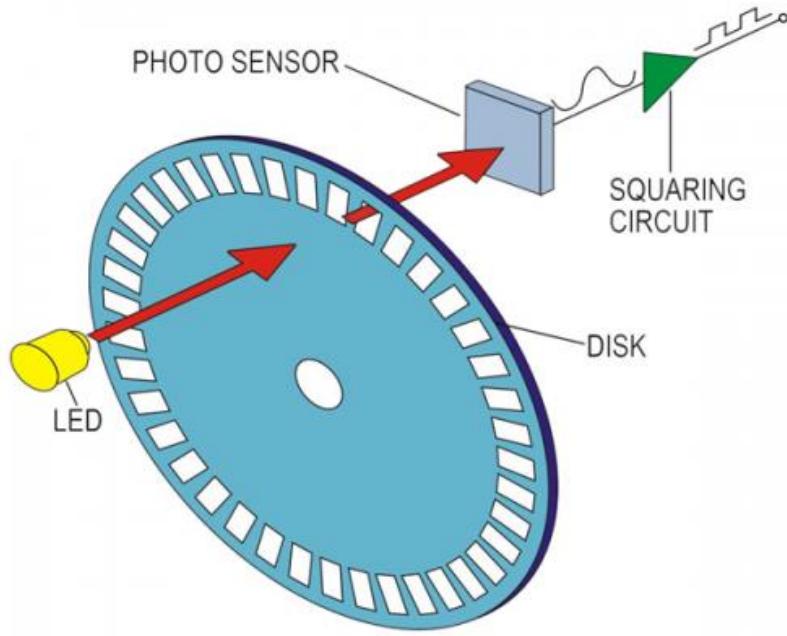
Thử làm: mua IC LM 298, và làm theo hướng dẫn tài liệu trên mạng để cắm vào breadboard điều khiển động cơ một chiều quay tiến lùi bằng cách đặt tín hiệu điện áp bằng tay.

4. Các loại cảm biến thông dụng

Giống như con người robot sẽ cần rất nhiều các loại cảm biến để có thể hoạt động và hiểu được môi trường bên ngoài. Robot khi hiểu môi trường sẽ có thể hoạt động một cách tự động theo các ứng dụng được lập trình.

a. Encoder

Encoder: là bộ đo tốc độ động cơ, một số động cơ một chiều có sẵn encoder, tuy nhiên phần lớn động cơ không có, tính năng này các bạn có thể gắn thêm bộ encoder vào trục của robot để đo độc lập. Endoder thường được thiết kế là một cặp mắt đọc qua một đĩa được đục lỗ cố định. Dựa vào việc biết thời gian động cơ quay hết một vòng đĩa, chúng ta có thể xác định được tốc độ động cơ. Nếu có 2 mắt đọc có thể xác định được chiều động

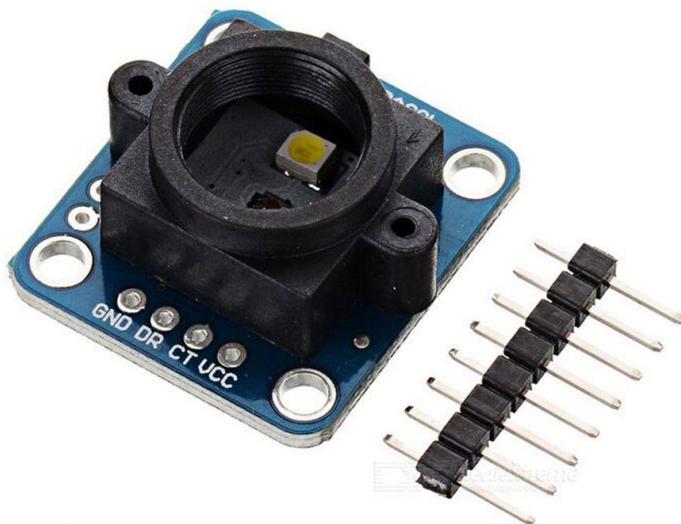


Cảm biến dò đường - quang trắc, ánh sáng: đây là cảm biến được dùng rất nhiều trong thực tế. Ngay như encoder cũng sử dụng cảm biến ánh sáng để xác định và đếm số vòng quay của động cơ, robot. Trong bộ 37 cảm biến, có 1 loại cảm biến được sử dụng để xác định được vạch trắng hoặc đen giúp robot chuyển động. Cảm biến gồm 3 chân VCC, GND và 1 chân tín hiệu báo có vạch trắng hoặc vạch đen. Một led phát hồng ngoại hoặc ánh sáng, một con thu ánh sáng hoặc hồng ngoại. Tùy vào sàn màu sắc sẽ phản chiếu lại giá trị khác nhau. Mạch có một IC so sánh cho quyết định là có phải vạch trắng hay vạch đen và trả lại giá trị logic. Được ứng dụng làm robot tự động chạy theo vạch trong các nhà máy.



b. Cảm biến màu sắc

Cảm biến màu sắc: có rất nhiều loại cảm biến có thể nhận dạng được màu sắc khác nhau. Các loại sẽ khác nhau về độ phân giải và cách thức giao tiếp. Ví dụ cảm biến dưới sử dụng IC TCS34725 để giao tiếp với các thiết bị khác bằng UART hoặc I2C. Cảm biến này có thể nhận dạng được chính xác giá trị RGB của từng loại màu sắc. Để hoạt động, các bạn phải cấp nguồn VCC và GND cho cảm biến và chọn cổng tương tác I2C hay UART để đọc giá trị.



c. Cảm biến chạm

Cảm biến chạm: đây là một loại cảm biến rất đơn giản về nguyên lý và chế tạo nhưng lại được dùng rất phổ biến trong các thiết kế robot cho kết thúc các hành trình. Cảm biến nay đơn giản là một công tắc, được thiết kế có các râu tiện trong từng trường hợp sử dụng. Các công tắc hành trình thường có 3 chân ký hiệu là NO – NC - COM tương ứng trạng thái thường mở - thường đóng – chân chung. Chân chung sẽ được nối xuống đất GND, chân NO hoặc NC thường được nối về mạch lập trình.



37 IN 1 Sensors kit for Arduino

	JoyStick XY		Flame		RGB LED		Heartbeat		Light Cup		Hall magnetic
	Relay		Linear Hall		SMD RGB		7Color flash		Tilt switch		TEMP 18B20
	Bigsound		Touch		Two-color		Laser emit		Ball switch		Analog temp
	Small sound		Digital temp		Two-color		Button		photoresistor		TR emission
	Tracking		Buzzer		Reed switch		Shock		temp and humidity		IR receiver
	Avoid		Passive buzzer		Mini Reed		Rotary encoders		Analog Hall		Tap module Light blocking

KIT cảm biến thông dụng 37 trong một hộp KIT, các bạn có thể dễ dàng mua trên mạng:

1. Module nút nhấn PS2
2. Module thu hồng ngoại
3. Module cảm biến Laser
4. Module cảm biến nhiệt độ và độ ẩm
5. Module cảm biến hồng ngoại
6. Module relay 5V
7. Module cảm biến vật cản
8. Module cảm biến nhịp tim
9. Module cảm biến âm thanh
10. Module cảm biến chạm

11. Module cảm biến khí ga
12. Module led RGB
13. Module cảm biến vạch
14. Module cảm biến từ trường
15. Module chỉnh góc quay
16. Module còi beep
17. Module công tắc thủy ngân
18. Module buzzer nhỏ
19. Module cảm biến nhiệt độ nhiệt điện trở RTC
20. Module ngắt ánh sáng
21. Module cảm biến nhiệt độ DHT11
22. Module Led 2 màu
23. Module công tắc thủy ngân KY-017
24. Module cảm biến từ trường
25. Module Led dán RGB
26. Module Led 2 màu mini
27. Module công tắc thủy ngân KY-020
28. Module Led 7 màu
29. Module nút nhấn
30. Module quang trở
31. Module cảm biến rung

- 32. Module công tắc rung
- 33. Module cảm biến nhiệt độ DS18B20
- 34. Module cảm biến từ trường tương tự
- 35. Module cảm biến âm thanh
- 36. Module công tắc từ
- 37. Module phát hồng ngoại

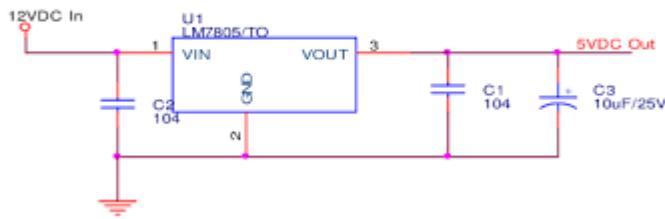
5. IC là gì ?

IC là gì ? IC đơn giản là mạch tích hợp của nhiều linh kiện transistor, điện trở được tích hợp trong một khối nhựa được đóng lại và có một số chân ra ngoài. Các IC thường được tích hợp các chức năng cụ thể: IC đọc chân, IC đi. VD như IC nguồn là 7805 mà bạn đã làm thử ở phần trên, hay IC so sánh LM324 dùng để so sánh giá trị cảm biến liên tục để đưa ra tín hiệu điều khiển. IC sẽ có nhiều hình dạng khác nhau. Nhưng có một điểm chung là IC nào cũng phải cấp nguồn VCC và GND cho nó. Thông thường các IC có nhiều chân sẽ có một dấu chấm nhỏ để bắt đầu đếm chân. Để sử dụng IC nào bạn cần phải đọc Datasheet - (tài liệu mô tả về thông số kỹ thuật, thiết kế mẫu và cách thức sử dụng cũng như các lưu ý sử dụng) để hiểu và làm việc



Thử làm: Một bộ nguồn 5V cung cấp cho mạch điện tử sử dụng IC nguồn từ điện áp 12V của acquy hoặc pin. Sử dụng IC 7805 và cắm lên bộ breadboard, dùng đồng hồ để đo điện áp.

Mạch nguồn



Các chuẩn giao tiếp cơ bản IC và MCU

UART: Giao tiếp bất đồng bộ qua 2 chân truyền (TX) và chân nhận (RX), mô hình này chỉ giao tiếp được 2 điểm, 2 thiết bị trực tiếp bằng cách nối chân Tx của thiết bị này với Rx thiết bị kia. Giao tiếp ngang hàng, thường giao tiếp giữa máy tính với máy tính nhúng, hoặc các vi điều khiển máy tính nhúng với nhau.

I2C: Giao tiếp đồng bộ, thường được thiết kế để các vi điều khiển giao tiếp với các cảm biến. Mô hình giao tiếp dạng bus có nhiều thiết bị trong cùng một mạng. Có thiết bị chủ, tớ.

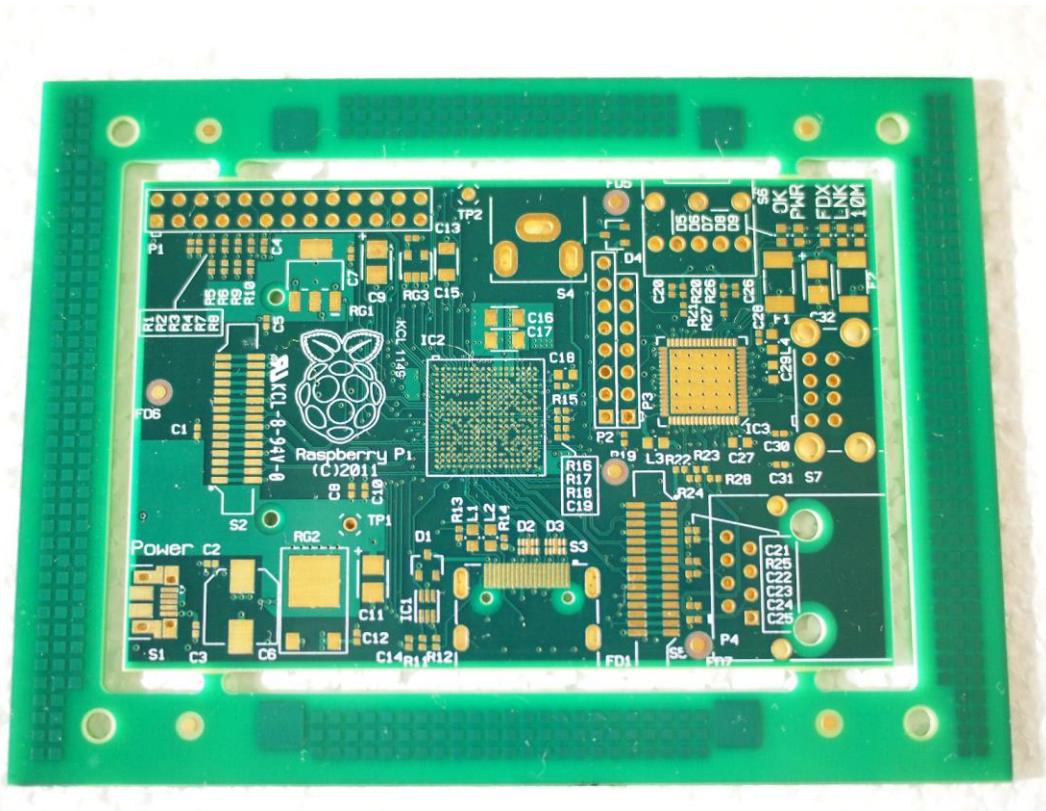
Other: tùy ứng dụng mà chúng ta sẽ dùng các chuẩn giao tiếp khác nhau với các thiết bị khác, các bạn có thể tự tìm hiểu thêm nhé USB, SPI, CAN, I2S,

Thử làm: mua một mạch Arduino, tìm cách nói chuyện với máy tính thử làm ứng dụng ở mục ví dụ của Arduino về giao tiếp 4. Communication. Qua ví dụ này bạn sẽ hiểu hơn về chuẩn giao tiếp truyền thông và giao tiếp thiết bị với máy tính.

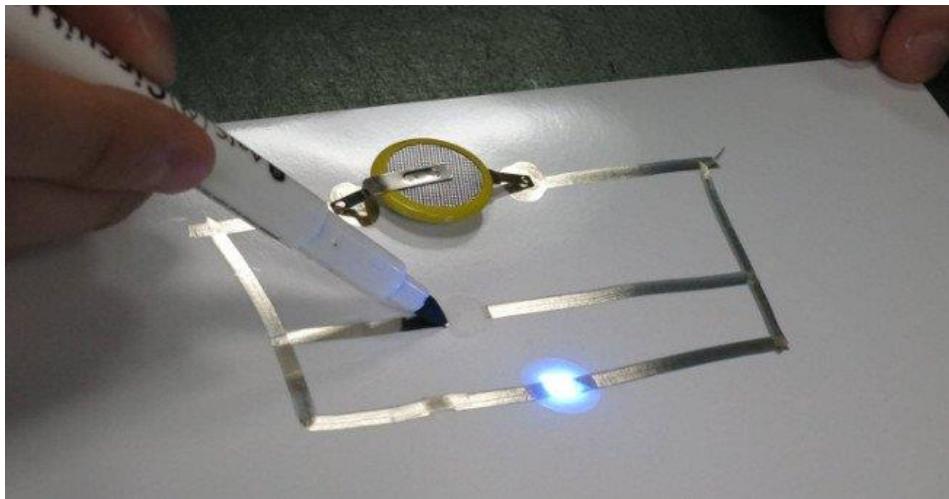
6. PCB

Printed circuit board (PCB): bảng mạch in, đôi khi gọi tắt là mạch in, là bảng mạch điện dùng nối các chân linh kiện với nhau trong các sản phẩm điện tử. PCB giúp một sản phẩm điện, điện tử trở nên gọn gàng, ổn định và dễ dàng chế tạo, sản xuất.

Chế tạo bảng mạch in là công đoạn quan trọng trong quá trình chế tạo bảng mạch điện tử. Trước đây việc làm bảng mạch in tách rời với công đoạn lập sơ đồ mạch điện. Ngày nay hệ thống thiết kế và sản xuất hỗ trợ bảng mạch (CAD-CAM) đảm bảo tự động liên hoàn từ thiết kế sơ đồ mạch điện đến lắp ráp, giảm nhẹ sự can thiệp của con người và cho ra sản phẩm giá thành hạ.



Thử làm: Vẽ một mạch in PCB bằng bút vẽ mạch, các bạn có thể biến một đồ vật trở thành một mạch điện tử nối các linh kiện. Thử vẽ lên áo một led và một pin 3V



Tự làm mạch in thủ công: sau khi thiết kế mạch in các bạn cần phải đặt mạch in hoặc tự làm mạch in. Cách tự làm mạch in cũng không quá phức tạp. Các bạn sẽ phải tự mua bảng đồng, thiết kế mạch, in mạch và là vào bảng đồng. Sau đó các bạn cho hóa chất (HCl hoặc bột Fe ăn mòn các chỗ không được che bởi mực in. Cuối cùng rửa sạch các bạn sẽ được một mạch điện theo thiết kế

Thử làm: Tìm trên mạng phương pháp làm mạch in thủ công, tự làm cho mình một mạch led trái tim!

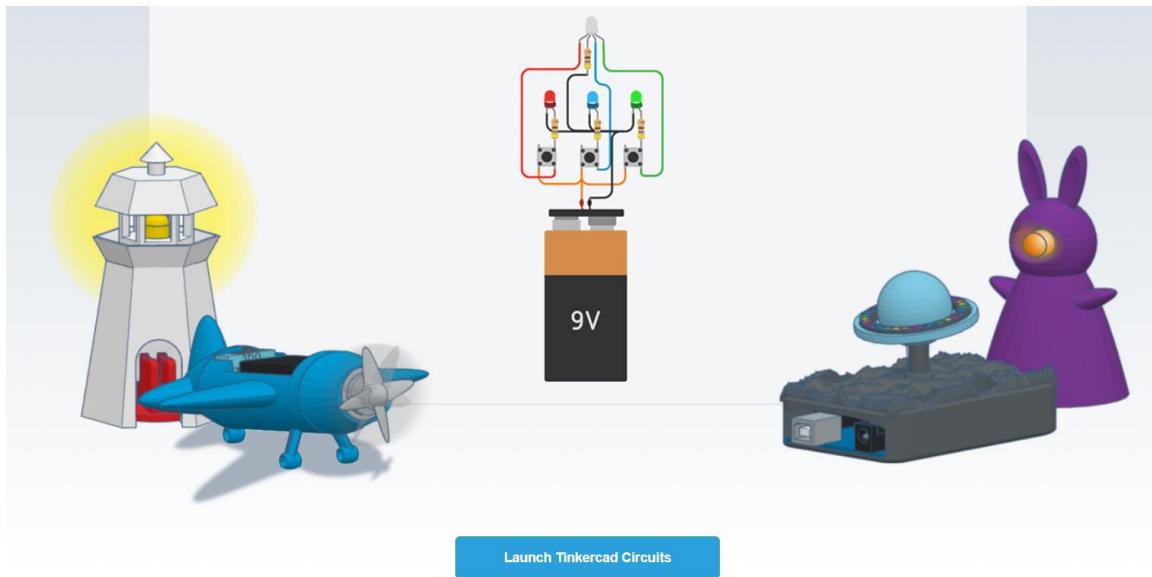
Thiết kế mạch in

Các phần mềm thiết kế mạch in nổi tiếng nhất là Orcad và Altium, các mạch khó nhiều lớp đều phải sử dụng 2 phần mềm này. Tuy nhiên chi phí bản quyền các phần mềm này lên tới hàng trăm triệu, rất khó để DIY tiếp cận.

Các phần mềm thiết kế mạch mã nguồn mở, như Eagle hay Kicad thì có điểm giới hạn về thư viện thiết kế và các công cụ hỗ trợ lúc làm mạch. Đặc biệt là các tính năng kiểm tra lỗi

Ngoài ra có một số phần mềm thiết kế mạch cho người mới bắt đầu, trẻ em, người không chuyên của AutoDesk <https://www.tinkercad.com/circuits>

Thử làm: Tự thiết kế một mạch in trên Tinkercad để làm một mạch điều khiển động cơ dùng L298 + khối nguồn 7805 với mạch điều khiển Arduino , nguồn vào 12V.



Kỹ năng hàn mạch –

Hàn mạch là một kỹ năng quan trọng để thiết kế một sản phẩm điện tử hay robot. Các mối hàn phải đảm bảo chất lượng để các thiết bị có thể hoạt động. Nay nay khi sản xuất hầu hết quy trình hàn đều được làm tự động bằng máy móc, tuy nhiên trước giai đoạn sản xuất đều cần các thử nghiệm để có thể kiểm tra tính năng của các mạch điện tử nên vẫn phải hàn bằng tay.

<https://www.instructables.com/id/How-to-solder/>

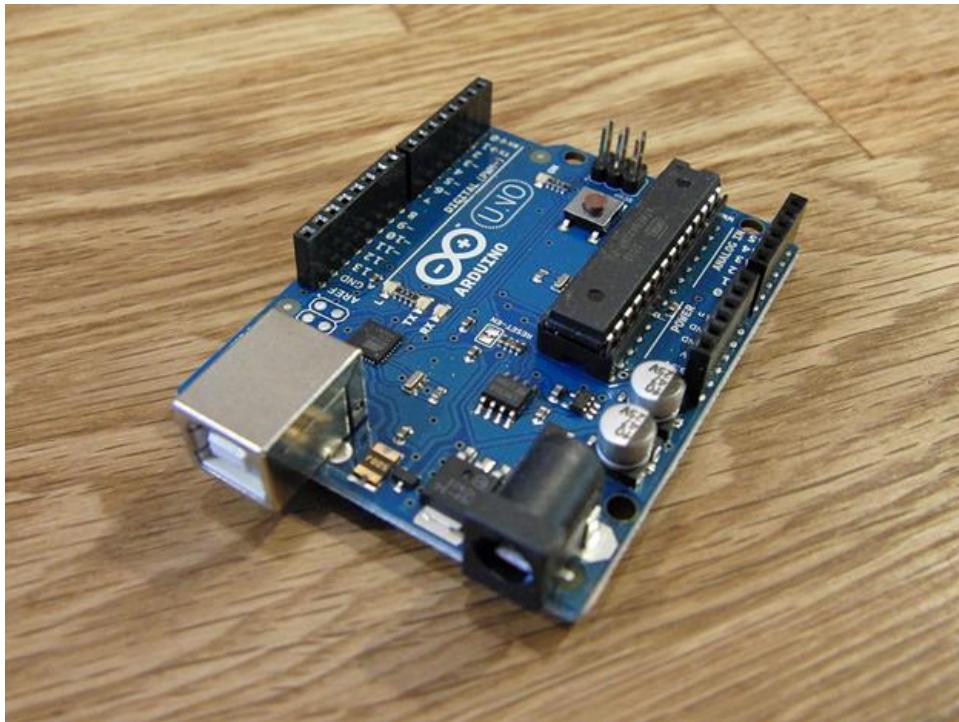
Thử làm: hàn một mạch đơn giản ví dụ như robot dò đường

Cửa hàng linh kiện điện tử cơ bản ở Việt Nam

- Minh Hà - <https://banlinhkiem.com/>
- Thiên Minh - <http://www.tme.vn/>
- IoT Maker - <https://iotmaker.vn/>

- Và rất nhiều các trang web bán linh kiện điện tử khác như KME, thegioiC, linhkien360,... Ngoài ra có 2 địa điểm bán đồ điện tử nổi tiếng từ rất lâu:
- Khu điện tử - Chợ trời - 33 Thịnh Yên, Phố Huế, Hai Bà Trưng, Hà Nội
- Khu điện tử - Chợ Nhật Tạo - Nhật Tảo, Phường 7, Quận 10, Hồ Chí Minh

7. Arduino là gì?

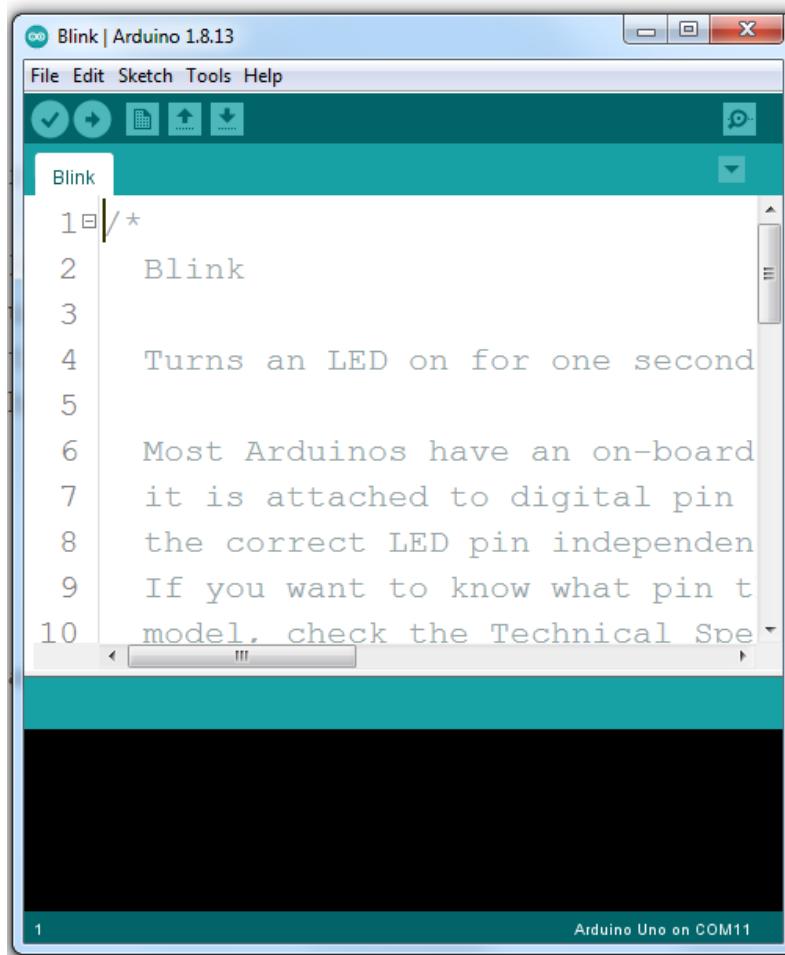


Arduino thường được biết tới với bảng mạch màu xanh nước biển đặc trưng, với logo là biểu tượng vô cực. Thường được sử dụng bởi rất nhiều đối tượng từ những bạn học sinh sinh viên, hay những người ít có kiến thức về điện tử cho tới những kỹ sư điện, điện tử, khoa học máy tính chuyên nghiệp, giúp họ biến ý tưởng của mình thành sự thật.

Arduino là một nền tảng mạch điều khiển mã nguồn mở, cả phần cứng và phần mềm. Được thiết kế với mục tiêu người dùng dễ tiếp cận, không yêu cầu nhiều về kiến thức điện tử hay lập trình và có giá thành thấp.

Phần cứng và của Arduino được kế trên nền tảng vi xử lý AVR Atmel 8bit, hoặc ARM Atmel 32-bit, với thiết kế đơn giản và mã nguồn ở, 1 trong số ít nền tảng phần cứng mã nguồn mở vào

thời điểm ra mắt, các mạch Arduino được phát triển cải tiến nhanh, và sản xuất rộng rãi với giá thành rẻ, dễ tiếp cận cho học sinh sinh viên

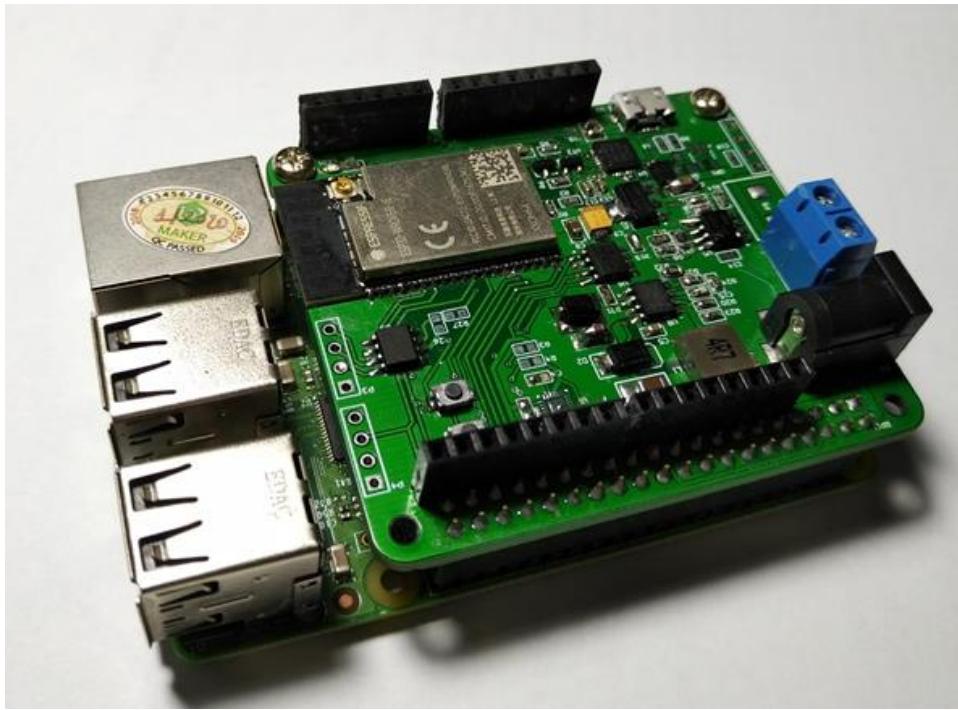


Phần mềm là đặc điểm kết nối hệ sinh thái của Arduino, khi cho phép người dùng sử dụng ngôn ngữ C++ bậc cao, đơn giản trên các mạch vi điều khiển. với cùng một chương trình có thể chạy trên nhiều mạch và nền tảng vi xử lý khác nhau. Hệ sinh thái phần mềm của Arduino phát triển rất mạnh, hỗ trợ nhiều vi điều khiển như STM32, ESP8266, ESP32, và kho thư viện mã nguồn mở đồ sộ. Phần cứng của mạch Makerbot Banhmi dựa trên vi điều khiển ESP32 nên việc sử dụng hệ sinh thái phần mềm của arduino sẽ giúp chúng ta dễ dàng và tiện lợi hơn khi bắt đầu

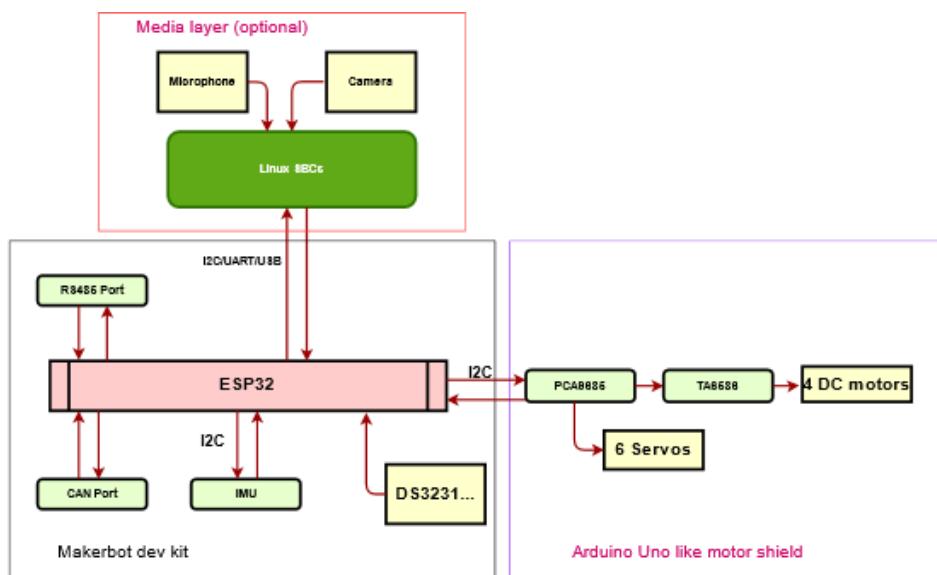
8. Mạch Makerbot BANHMI là gì?

Là một nền tảng Robot Mã Nguồn mở, hỗ trợ cho nhiều loại robot và xe tự hành. Với mục tiêu là một nền tảng đa năng, nhỏ gọn dễ tiếp cận với giá thành thấp .Hỗ trợ điều khiển nhiều

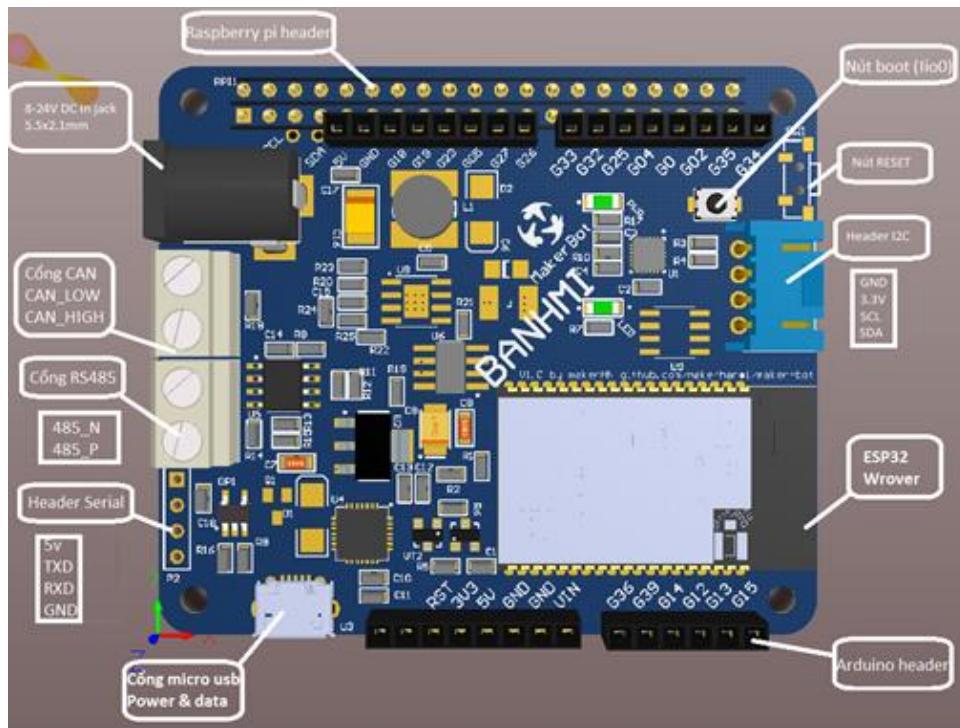
động cơ, kết nối giao tiếp với các mạch máy có khả năng điều khiển lên tới 10 động cơ độc lập (4 động cơ DC, và 6 động cơ servo). Mạch Makerbot BANHMI còn hỗ trợ các chuẩn kết nối không dây: WIFI, BLE giúp truyền dữ liệu và điều khiển. Trên mạch còn tích hợp cảm biến góc gia tốc 6 trục và chân cắm I2C, UART.



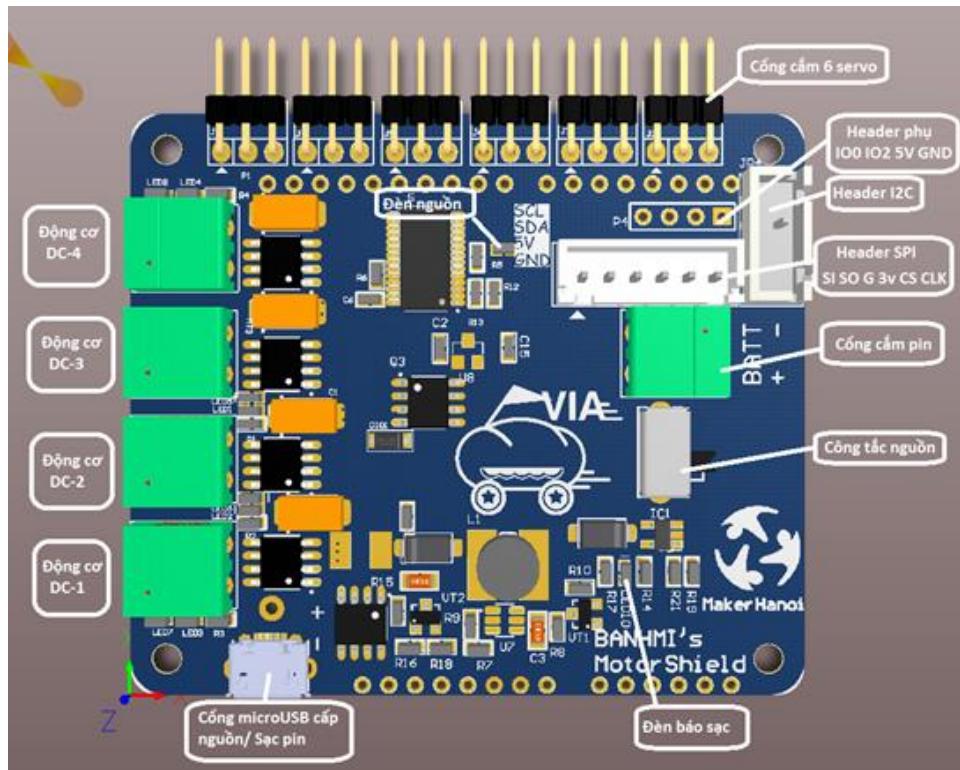
Thành phần chính của mạch Makerbot:



a. Mạch điều khiển (control board)



b. Mạch công suất (motor shield)



c. **Thử làm với MakerBot BANHMI**

Kết nối mạch Makerbot với máy tính

Kết nối MakerBot với máy tính qua cáp microUSB, Khi kết nối với máy tính, máy tính sẽ tự động tiến hành cài đặt driver cho mạch MakerBot, sau khi cài đặt driver xong trên máy tính sẽ xuất hiện thiết bị Silicon Labs CP210x USB to UART Bridge

Lưu ý:

- *Nên sử dụng dây cáp micoUSB có chất lượng tốt*
- *Không nên cắm mạch makerbot qua bộ chia USB (USB HUB)*
- *Nếu có thẻ, nên cắm mạch MakerBot vào cổng USB 3.0 để đảm bảo nguồn cung cấp năng lượng cho mạch MakerBot*

9. Thử làm, điều khiển động cơ qua WIFI

a. **Nạp chương trình cho mạch Makerbot**

Chương trình điều khiển động cơ DC qua WIFI có thể tìm thấy tại đây.

Để nạp chương trình các bạn cần sử dụng Platform IO hoặc Arduino IDE,

Lưu ý: đối với Arduino IDE cần phải cài đặt thêm board ESP32 chi tiết tại [đây](#)

Cách sử Dụng:

Khi cấp nguồn cho mạch MakerBot (không nhất thiết kết nối với máy tính, mạch MakerBot sẽ phát ra 1 WIFI access point có tên là ESPUI

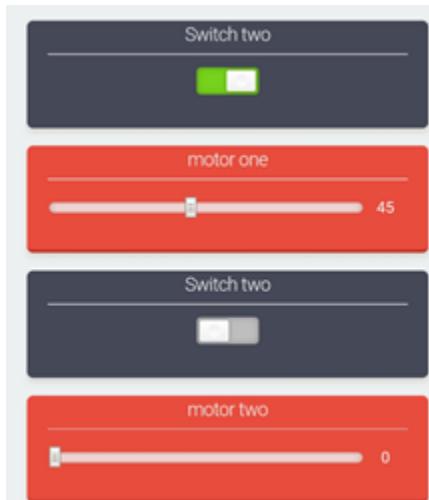
Khi kết nối với mạng WIFI này không yêu cầu mật khẩu.

Sau khi kết nối, mở trình duyệt nhập địa chỉ: **192.168.4.1**, giao diện điều khiển sẽ xuất hiện

Giao diện điều khiển bao gồm 4 công tắc và 4 thanh trượt:

- 4 thanh trượt có chức năng điều khiển tốc độ động cơ, khi thay đổi tốc độ động cơ độ sáng đèn báo hiệu động cơ cũng thay đổi tương ứng

- 4 công tắc có nhiệm vụ đảo chiều động cơ, khi thay đổi trạng thái công tắc và thanh trượt động cơ sẽ đổi chiều quay đồng thời đèn báo hiệu sẽ đổi màu tương ứng



Phần 3: VIA và trí tuệ nhân tạo

Hệ thống trí tuệ nhân tạo cho xe tự lái

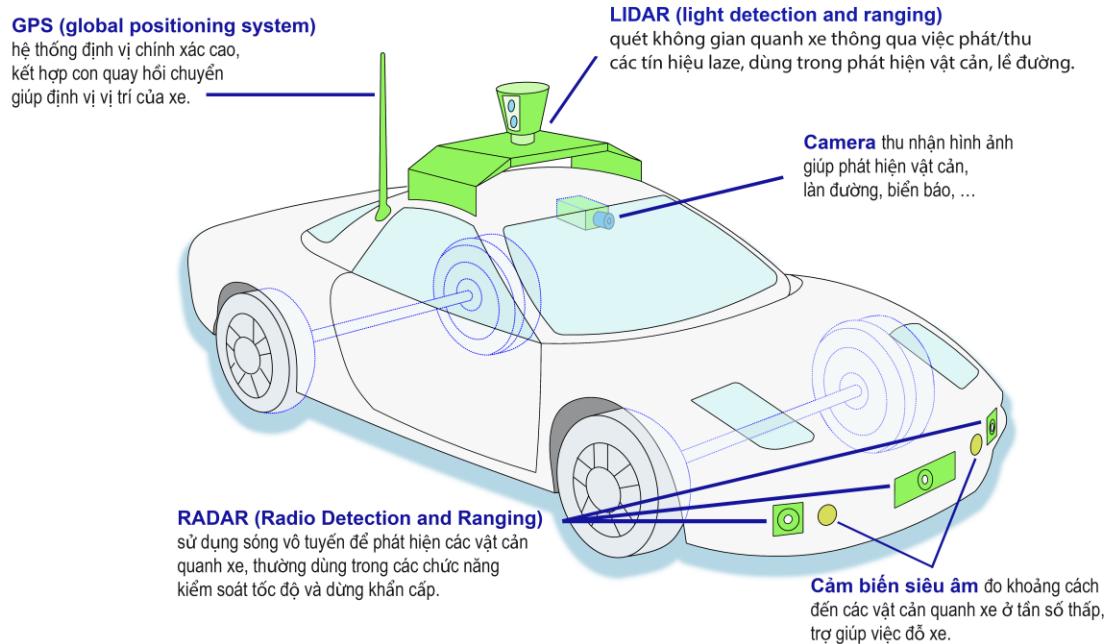
Hệ thống trí tuệ nhân tạo cho phép xe và robot tự hành có thể cảm nhận môi trường xung quanh, tính toán, đưa ra lộ trình phù hợp và điều khiển xe đi tự động, không cần nhiều can thiệp từ con người. Ở phần AI cho VIA, chúng ta sẽ làm quen với thiết kế của một hệ thống xe tự hành, bao gồm các thuật toán quan sát môi trường, tính toán và điều khiển xe.

1. Nguyên lý hoạt động xe tự lái

Để có được tính năng tự hành hiệu quả, an toàn, chiếc xe cần kết hợp thông tin của rất nhiều loại cảm biến khác nhau trên xe như camera, radar, lidar, cảm biến siêu âm, các loại cảm biến định vị, ... xử lý chúng bằng các thuật toán thông minh chạy trên một hệ thống máy tính công nghiệp trên xe. Từ thông tin phản hồi từ các cảm biến mà chiếc xe tự hành có thể nhận ra làn đường, vật cản, tín hiệu biến báo, và đèn giao thông, từ đó điều khiển bánh lái, hệ thống phanh, hệ thống đèn tín hiệu trên xe.

b. Các loại cảm biến

Để quan sát môi trường, một chiếc xe tự lái dùng rất nhiều cảm biến khác nhau.



Các loại cảm biến trên xe tự lái

- **Camera** là cảm biến có giá thành rẻ, có khả năng chụp ảnh tương tự như các loại máy ảnh, điện thoại khác. Ngoài các tính năng truyền thống như hỗ trợ lùi, quan sát điểm mù, camera trên ô tô tự hành thường được dùng để nhận dạng làn đường, vật cản, biển báo và đèn giao thông do có độ phân giải cao, dễ dàng phân biệt được các vật thông qua màu sắc, hình dạng của chúng.
- **Cảm biến siêu âm** cũng được dùng trên ô tô để cảnh báo va chạm, hỗ trợ lùi xe ở tốc độ thấp. Hiện tại loại cảm biến này đã được trang bị trên rất nhiều loại xe trên thị trường. Nguyên lý hoạt động của chúng là phát sóng siêu âm và thu lại sóng phản hồi, từ thời gian truyền sóng có thể tính toán được khoảng cách tới các vật cản.
- **RADAR** cũng là một loại cảm biến rất phổ biến cho xe tự lái, nhằm xác định vật cản, hỗ trợ tính năng kiểm soát tốc độ và dừng khẩn cấp. Radar trên ô tô thường là loại radar chủ

động, phát sóng điện từ và thu tín hiệu phản hồi để tính toán ra vị trí, vận tốc của các vật trước và sau xe. Ưu điểm của radar là giá thành và khả năng xác định được vận tốc của vật trực tiếp thông qua hiệu ứng Doppler.

- **LIDAR** là loại cảm biến được ứng dụng nhiều trong các hệ thống tự lái tiên tiến hiện nay. Nhờ việc phát và thu các chùm tia laze, lidar giúp xác định chính xác khoảng cách đến các vật xung quanh xe. Lidar có ưu điểm là có thể xác định khoảng cách chính xác và dễ dàng hơn camera, radar, đồng thời có độ phân giải tốt. Tuy vậy, giá thành của lidar thường tương đối cao nên chưa được ứng dụng rộng rãi trong ô tô.
- **Hệ thống định vị (GPS/GNSS/IMU)** cũng là một thành phần quan trọng trên các xe tự hành. Nó giúp chiếc xe biết được vị trí, hướng hiện tại. Nhờ đó chiếc xe có khả năng tìm đường chính xác. Chúng còn cung cấp các thông tin phản hồi về gia tốc, tốc độ cho xe, hỗ trợ các tính năng điều khiển.
- Ngoài ra, hệ thống ô tô còn bao gồm rất nhiều loại cảm biến khác nhau như **cảm biến vận tốc bánh, cảm biến mưa, cảm biến áp suất lốp, cảm biến mưa...** Để tận dụng tất cả các loại cảm biến trên xe, kết hợp chúng lại để tạo ra một phương tiện tự lái an toàn là nỗ lực rất lớn của các hãng xe hiện nay.

c. Hệ thống AI cho VIA

Hệ thống AI cho VIA được thiết kế dựa trên camera, gồm 2 module chính là module nhận dạng làn đường và module nhận dạng biển báo, từ đó ghép nối thêm phần điều khiển để giúp xe di chuyển trong các điều kiện giả lập và điều kiện sa hình khác nhau. Trong khóa học này, chúng ta sẽ cùng tìm hiểu các thành phần trong hệ thống AI cho dự án VIA, cùng nhau xây dựng một hệ thống AI đơn giản cho xe tự lái dựa trên camera. Mở rộng hơn, các bạn được gợi ý để phát triển thêm các tính năng khác như nhận dạng vật cản.

d. Thông tin thêm: Các cấp độ xe tự lái

Trên thực tế, một chiếc xe tự lái không có nghĩa là nó có thể thực hiện tất cả các thao tác lái xe một cách tự động hoàn toàn, không cần sự can thiệp từ con người. Có nhiều cấp độ xe tự lái khác nhau. Theo Hiệp hội Kỹ sư Ô tô (SAE International) và Cơ quan An toàn Giao thông Cao tốc

Quốc gia Mỹ (NHTSA), khả năng của xe ô tô tự lái được phân thành 6 cấp độ, từ cấp độ 0 đến cấp độ 5. Cụ thể như sau:

- **Cấp độ 0 - Hoàn toàn thủ công:** Đây là cấp độ mà phương tiện phụ thuộc hoàn toàn vào con người và người điều khiển xe sẽ chịu trách nhiệm cho mọi việc vận hành xe. Các tính năng tự lái không có mặt trong các xe cấp độ 0.
- **Cấp độ 1 - Hỗ trợ lái xe:** Ở cấp độ này, người lái xe và hệ thống tự động cùng chia sẻ quyền kiểm soát xe. Xe sẽ chỉ thực hiện duy nhất một nhiệm vụ như đánh lái, thay đổi – duy trì tốc độ hay phanh khẩn cấp tự động cảnh báo người lái xe khi có va chạm,...
- **Cấp độ 2 - Tự động một phần:** Xe được tích hợp những chức năng tự động tiên tiến có khả năng thực hiện nhiều hơn một nhiệm vụ, có thể kiểm soát cả chuyển hướng và điều khiển tốc độ, cảnh báo chệch làn đường và can thiệp điều chỉnh lại hướng. Tuy nhiên, người điều khiển vẫn phải giám sát việc lái xe và sẵn sàng can thiệp ngay lập tức bất cứ lúc nào nếu hệ thống tự động không phản hồi đúng cách.
- **Cấp độ 3 - Tự động hóa có điều kiện:** Xe cấp độ 3 có khả năng nhận định môi trường xung quanh và đưa ra quyết định cho chính mình, chẳng hạn như tăng tốc vượt qua xe đang di chuyển chậm; tự điều khiển từ điểm xuất phát đến đích mà không có sự tham gia của con người trong một số điều kiện nhất định. Nếu hệ thống không thể hoạt động một cách đáng tin cậy thì người lái xe phải luôn cảnh giác và sẵn sàng kiểm soát.
- **Cấp độ 4 - Tự động hóa hoàn toàn trong môi trường kiểm soát:** Chiếc xe không cần sự tương tác của con người trong hầu hết trường hợp, trong điều kiện là xe di chuyển trong một rào chắn địa lý (trong một vùng đã được xác định).
- **Cấp độ 5 - Tự động hóa hoàn toàn:** Đây là mục tiêu cuối cùng của một chiếc xe tự lái, có thể vận hành tự động trong mọi tình huống và điều kiện. Ở cấp độ này, xe sẽ hoàn toàn không có bất kỳ sự can thiệp nào của con người, thậm chí sẽ không có vô lăng hoặc bàn đạp tăng tốc, hay phanh thủ công từ phía người dùng.

Phát hiện làn đường

Trong bài toán phát triển xe tự hành với VIA, chúng ta sẽ dùng camera lắp phía trước để xác định và giúp xe bám theo làn đường. Trong phần này, các bạn sẽ được làm quen với ngôn ngữ lập trình Python, môi trường thí nghiệm Google Colab, các thuật toán xử lý ảnh cơ bản, đồng thời áp dụng những gì đã học để xây dựng chương trình phát hiện vạch kẻ đường đơn giản để điều khiển xe tự lái.

1. Làm quen với Python và Google Colab

a. Ngôn ngữ Python

Các bài học chúng ta sẽ học tiếp theo đây sẽ cần các bạn nắm trước các kiến thức cơ bản về lập trình Python như các khái niệm dữ liệu, biến, câu lệnh rẽ nhánh, cấu trúc lặp, hàm và kiểu dữ liệu danh sách (list). Vì vậy, trước khi bắt đầu bài học về phát hiện làn đường, các bạn cần tự tìm hiểu trước các kiến thức về ngôn ngữ Python thông qua các khóa học khác. Một số khóa học hay về Python có thể được kể đến như:

- Khóa học Python cơ bản – Howkteam – Miễn phí:
<https://www.howkteam.vn/course/lap-trinh-python-co-ban-37>
- Khóa học Python 3 của Codecademy – Khoảng 20 USD tháng, nhưng rất chất lượng, có môi trường học và thử nghiệm mã nguồn online.
<https://www.codecademy.com/learn/learn-python-3>

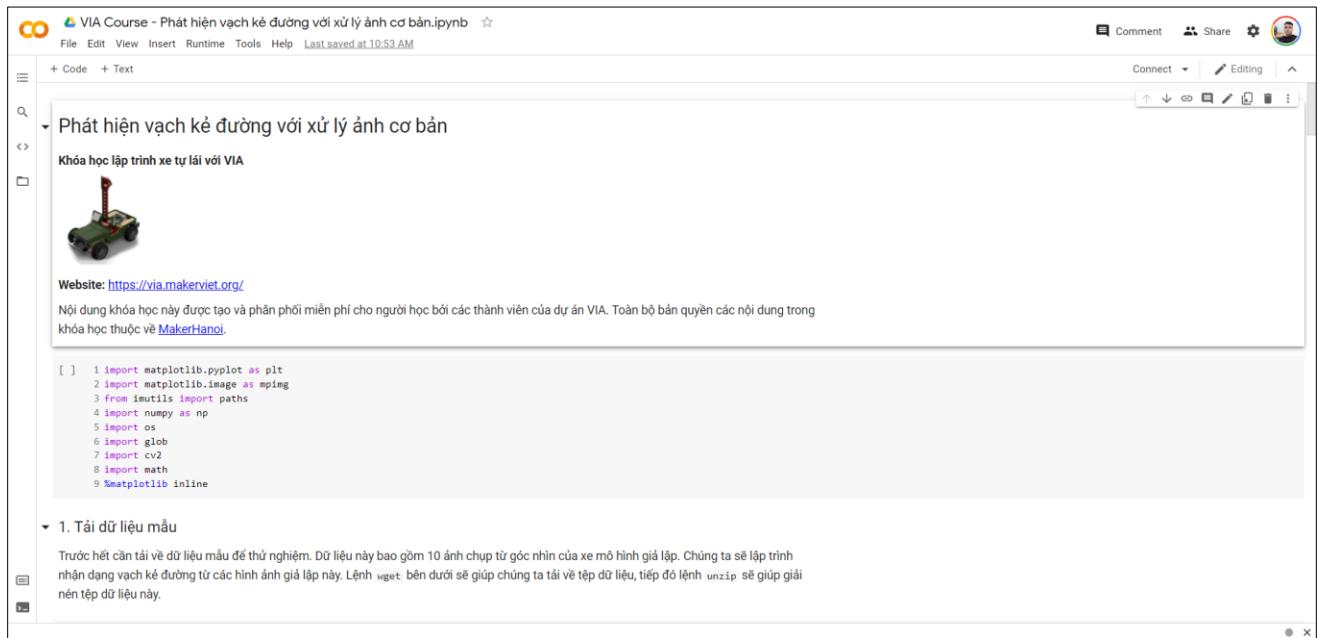
b. Môi trường thực hành Google Colab

Để việc thực hành diễn ra thuận lợi nhất, không gặp nhiều khó khăn về việc cài đặt môi trường, chúng ta sẽ sử dụng dịch vụ **Google Colab** để làm các bài thực hành về các thuật toán xử lý ảnh. Trước hết, các bạn cần tạo cho mình một tài khoản Google theo hướng dẫn tại liên kết sau:

[https://support.google.com/accounts/answer/27441?hl=vi.](https://support.google.com/accounts/answer/27441?hl=vi)

Tiếp đó, các bạn truy cập <https://colab.research.google.com/> để tạo và sử dụng các cuộn sổ Colab (Colab notebook), vừa có thể ghi chép, vừa có thể chạy thử nghiệm mã nguồn Python. Các bạn có thể xem một ví dụ sổ tay Colab tại đây:

<https://colab.research.google.com/github/makerhanoi/via-course-ai/blob/master/notebooks/01-So-tay-Colab.ipynb>.



The screenshot shows a Google Colab notebook titled "Phát hiện vạch kẻ đường với xử lý ảnh cơ bản". The notebook includes a header with the title and a sub-section "Khóa học lập trình xe tự lái với VIA". It features a small image of a green toy car. Below the image is a link to "Website: https://via.makerviet.org/". A note states that the content is free for educational purposes. The code cell contains Python imports for matplotlib, mpimg, imutils, numpy, os, glob, cv2, and math, followed by a call to plt.imshow. A section titled "1. Tải dữ liệu mẫu" provides instructions for downloading sample data from a website.

Hình ảnh một Colab notebook

c. Một số phím tắt quan trọng trong Colab

Cuốn sổ tay Colab được chia làm các phần nhỏ khác nhau, chứa mã nguồn hoặc nội dung ghi chép văn bản. Các phần này được gọi là các **cell**. Một cell có thể thuộc một trong hai loại, là cell mã nguồn hoặc cell văn bản. Các cell này có thể được chuyển đổi dễ dàng qua nhau. Tôi xin giới thiệu một số phím tắt quan trọng khi các bạn làm việc với Google Colab như sau. Các bạn nên nhớ các phím tắt này để thao tác dễ dàng hơn.

- Thực thi một cell: **Shift + Enter**. Khi ấn phím tắt này, nếu Colab sẽ thực hiện chạy các câu lệnh trong cell, nếu cell đó là cell mã nguồn, ngược lại sẽ thực hiện biên tập Markdown thành dạng văn bản đã định dạng.
- Tạo một cell mới bên dưới cell cũ: **Ctrl + M B** (các bạn ấn Ctrl + M sau đó ấn phím B trên bàn phím).
- Tạo một cell mới bên trên cell cũ: **Ctrl + M A**.
- Chuyển cell đang làm việc thành cell chứa mã nguồn: **Ctrl + M Y**.

- Chuyển cell đang làm việc thành cell chứa văn bản ghi chép: **Ctrl + M M**.
- Xóa cell hiện tại: **Ctrl + M D**.

2. Các thuật toán xử lý ảnh cơ bản

Xử lý ảnh bao gồm các thuật toán giúp tăng cường chất lượng ảnh, chuyển đổi ảnh hoặc trích xuất thông tin hữu ích từ hình ảnh. Các thuật toán xử lý ảnh có mặt trong rất nhiều các ứng dụng của đời sống như camera thông minh lắp ở các tòa nhà, xe tự lái, hay chính trong các ứng dụng chụp ảnh, sửa ảnh của điện thoại. Chúng ta hoàn toàn có thể xây dựng thuật toán cho các xe và robot tự hành bằng các thuật toán xử lý ảnh cơ bản mà chưa cần dùng đến các kỹ thuật nâng cao khác. Ở phần này, các bạn sẽ cùng tìm hiểu một vài thuật toán xử lý ảnh cơ bản và cách ứng dụng chúng với thư viện OpenCV, một thư viện rất phổ biến trong lĩnh vực thị giác máy tính.

a. *Làm quen với OpenCV*

Chúng tôi đã chuẩn bị sẵn tay Colab cho bài học về xử lý ảnh trên OpenCV tại liên kết bên dưới. Các bạn có thể sử dụng luôn cho các bài thực hành tại mục này.

<https://colab.research.google.com/github/makerhanoi/via-course-ai/blob/master/notebooks/02-OpenCV-co-ban.ipynb>

Trước tiên chúng ta cùng làm quen với thư viện xử lý ảnh OpenCV và các thao tác đọc, hiển thị ảnh trên Colab notebook. Đoạn mã nguồn sau sẽ giúp import các thư viện cần thiết. Thư viện OpenCV trong Python được sử dụng với tên **cv2**. Thông thường, khi chạy OpenCV trên môi trường Python của máy tính, chúng ta có thể sử dụng lệnh **cv2.imshow("Tên cửa sổ", <hình ảnh cần hiển thị>)** để hiển thị một ảnh. Tuy nhiên, với môi trường của sổ tay Colab, chúng ta không thể mở một cửa sổ mới, vì thế cần import thêm `cv2_imshow()` để hiển thị ảnh, thay thế cho `cv2.imshow()`.

```
# Import các thư viện cần thiết
from google.colab.patches import cv2_imshow # Dùng để hiển thị ảnh trên Colab
import cv2
import numpy as np
```

Tải và hiện ảnh từ URL: Với các ảnh được tìm thấy trên mạng, các bạn có thể dùng lệnh `wget` để tải về và hiện chúng lên với OpenCV. Ví dụ như sau:

```
# Tải hình ảnh và hiển thị trên notebook.
```

```
# Câu lệnh này sẽ thực hiện tải hình ảnh logo VIA từ đường dẫn https://via.makerviet.org/media/via-logo.,  
# lưu lại dưới tên `via-logo.jpg`. Để sao chép đường dẫn của một hình ảnh trên Chrome, các bạn ấn chuột phải,  
# chọn `Copy image address` hoặc `Sao chép URL`  
!wget https://via.makerviet.org/media/via-logo.jpg -O via-logo.jpg
```

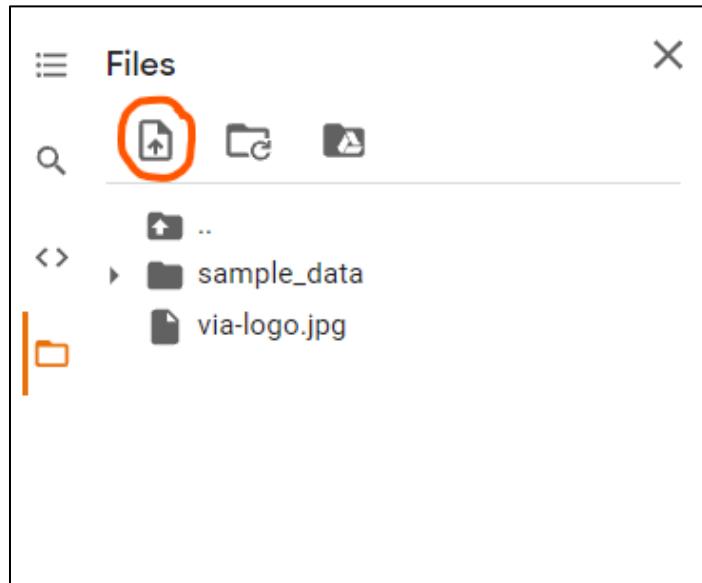
Tiếp đó, chúng ta mở hình ảnh với OpenCV và hiển thị trên notebook:

```
img = cv2.imread("via-logo.jpg") # Đọc ảnh via-logo.jpg  
cv2_imshow(img) # Hiện ảnh
```



Như vậy, hình ảnh `via-logo.jpg` đã được đọc lên với hàm `cv2.imread()` và hiển thị với lệnh `cv2_imshow()`. Bạn hãy thử tìm một hình ảnh khác trên Google, ấn chuột phải và chọn sao chép

đường dẫn của nó, sau đó thử tải và hiển thị lên với OpenCV để làm quen nhé. Bạn cũng có thể chọn tải lên một ảnh từ máy tính của mình với trình đơn **Files** ở bên tay trái của Colab.



b. Các kênh màu và chuyển đổi hệ màu

Không gian màu RGB

Các ảnh khi chúng ta đọc lên bằng OpenCV ở phần mã nguồn trước, nếu không có gì thay đổi, hoặc truyền các tham số đặc biệt vào hàm `cv2.imread()` thì sẽ được biểu diễn trong máy tính bằng sự kết hợp của 3 màu cơ bản: xanh lam, xanh lục, đỏ, theo thứ tự đó, hay còn gọi là không gian màu BGR (blue-green-red). Tại mỗi điểm ảnh, việc pha trộn 3 màu sắc cơ bản này sẽ tạo ra các màu sắc đa dạng khác nhau. Giá trị của mỗi màu (đỏ, lục, lam) thường được biểu diễn dưới dạng số nguyên, giá trị từ 0 (đen) đến 255 (giá trị cực đại của màu sắc đó). Khi đó, việc kết hợp 3 giá trị của các màu này sẽ thu được $255 \times 255 \times 255 = 16777216$ màu sắc khác nhau. Các bạn có thể thử chọn các màu sắc khác nhau và xem chúng được tạo từ 3 giá trị R (red), G (green) và B (blue) nào tại liên kết sau:

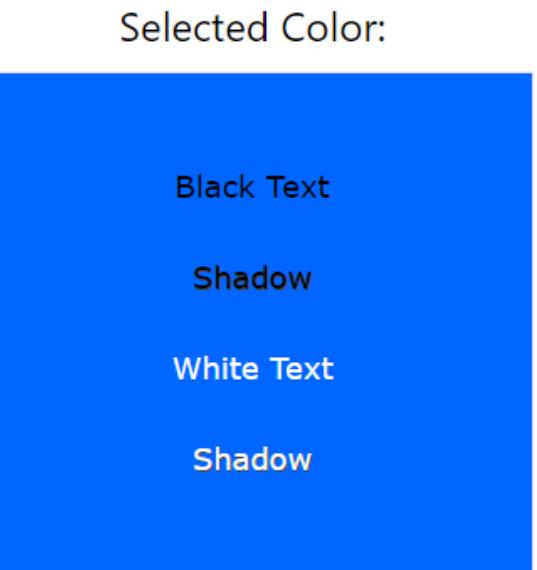
https://www.w3schools.com/colors/colors_picker.asp.



Or Enter a Color:

OK

Or Use HTML5:



Chúng ta sẽ thử tách hình ảnh lúc đầu và hiển thị màu sắc tại 3 kênh màu riêng biệt như sau.

```
# Tách các kênh màu
B, G, R = cv2.split(img)
# Hiển thị ở màu sắc của chúng
zeros = np.zeros(img.shape[:2], dtype="uint8")
R_merge = cv2.merge([zeros, zeros, R])
G_merge = cv2.merge([zeros, G, zeros])
B_merge = cv2.merge([B, zeros, zeros])
merge = np.hstack([R_merge, G_merge, B_merge])
cv2_imshow(merge)
```



Các bạn hãy thử nghiệm tải và tách 3 kênh màu của logo thư viện OpenCV tại đây nhé:

https://raw.githubusercontent.com/makerhanoi/via-course-ai/master/content/images/OpenCV_Logo.png

Ở thư viện OpenCV, không gian màu mặc định là BGR (lục, lam, đỏ), khác với rất nhiều hệ thống khác, cũng sử dụng hệ gồm 3 màu cơ bản này, nhưng theo thứ tự ngược lại là RGB. Để chuyển đổi giữa hai không gian màu này, ta sử dụng cú pháp như sau:

```
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)  
cv2_imshow(img_rgb)
```



Nếu bạn dùng lệnh `cv2_imshow()` để hiển thị hình ảnh thu được là `img_rgb`, bạn sẽ nhận thấy không gian màu đã bị thay đổi, và hình ảnh hiện lên không còn đúng màu sắc ban đầu nữa. Đó là vì các hàm `cv2_imshow()` hoặc `cv2.imshow()` sử dụng không gian màu BGR để hiển thị ảnh, khi truyền hình ảnh ở không gian màu khác, việc hiển thị có thể không đúng. Vì vậy, chúng ta nên chuyển đổi ảnh về hệ màu xám, hoặc không gian BGR trước khi hiển thị. Sau đây là câu lệnh chuyển đổi ảnh `img_rgb` ở không gian màu RGB thành `img_bgr` ở không gian màu BGR.

```
img_bgr = cv2.cvtColor(img_rgb, cv2.COLOR_RGB2BGR)  
cv2_imshow(img_bgr)
```



Không gian ảnh xám

Không gian ảnh xám là một định dạng ảnh khác cũng thường được sử dụng trong xử lý ảnh. Loại ảnh này khác ảnh BGR ở chỗ nó chỉ có một kênh màu duy nhất, biểu diễn mức xám của mỗi điểm ảnh trong bức ảnh. Sau đây là một ví dụ về việc chuyển hình ảnh BGR phía trên sang ảnh xám.

```
gray = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2GRAY)  
cv2_imshow(gray)
```



Không gian màu LAB

Không gian màu LAB được xây dựng dựa trên sự cảm nhận màu sắc của mắt người. Hệ màu này cũng có 3 kênh màu như BGR, nhưng gồm 3 thành phần khác, một trong số chúng là độ sáng (cường độ).

- L: Độ sáng (Cường độ).
- A: thành phần màu từ Xanh lục đến Đỏ tươi.
- B: thành phần màu từ Xanh lam đến Vàng.

Để chuyển đổi từ không gian màu BGR sang LAB, chúng ta dùng lệnh sau:

```
img_lab = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2LAB)
```

Không gian màu HSV (HSB)

Một không gian màu khác cũng được ứng dụng rất nhiều trong xử lý ảnh là HSV, với 3 kênh:

- H: (Hue) Vùng màu
- S: (Saturation) Độ bão hòa màu
- B (hay V): (Bright hay Value) độ sáng

Vì giá trị màu của HSV (H) được tách ra thành một kênh riêng, chúng ta có thể sử dụng kênh này để lọc ra các vùng có màu sắc mong muốn, ứng dụng trong việc phát hiện đồ vật theo màu sắc. Để chuyển đổi từ không gian màu BGR sang HSV, ta dùng câu lệnh như dưới.

```
img_hsv = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2HSV)
```

Tựu chung lại, chúng ta có nhiều không gian màu khác nhau được sử dụng cho những mục đích khác nhau trong quá trình xử lý trên máy tính. Nhiều không gian màu đã được hỗ trợ bởi OpenCV. Việc chuyển đổi giữa các không gian màu này có thể dùng cú pháp tương tự nhau:

```
output_img = cv2.cvtColor(input_img, cv2.COLOR_<hệ màu cũ>2<hệ màu mới>)
```

Các mã chuyển đổi khác nhau có thể được tìm thấy tại tài liệu chính thức của OpenCV: https://docs.opencv.org/4.5.4/d8/d01/group_imgproc_color_conversions.html. Các bạn có thể tự thực hành đọc ảnh và chuyển đổi qua lại giữa các hệ màu trên một Colab notebook để làm quen thêm nhé.

c. Lọc ảnh / Làm mịn ảnh

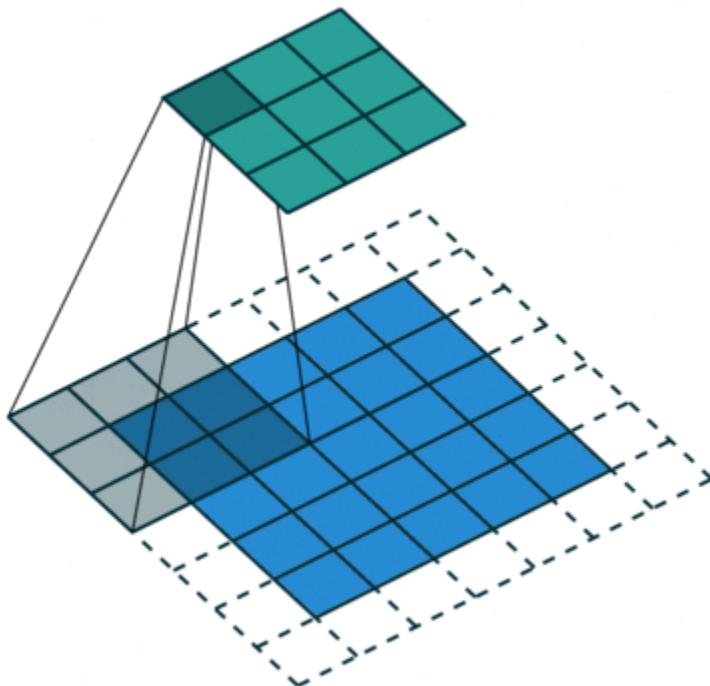
Lọc ảnh (hoặc làm mịn ảnh, làm mượt ảnh, lọc biên) là một bước rất quan trọng trong xử lý ảnh. Lọc ảnh thực tế có rất nhiều tác dụng như loại bỏ nhiễu, làm rõ biên đối tượng. Phần này sẽ giới thiệu đến các bạn một số nguyên lý chung và một vài bộ lọc hay dùng.

Sổ tay thực hành cho phần lọc ảnh:

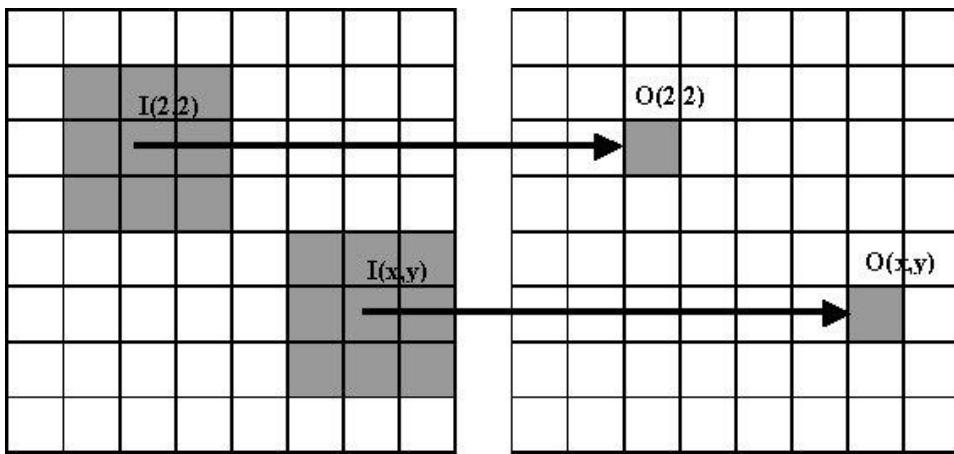
<https://colab.research.google.com/github/makerhanoi/via-course-ai/blob/master/notebooks/03-Loc-anh.ipynb>

Nguyên lý chung

Nguyên tắc chung của các phương pháp lọc là cho ma trận ảnh nhân với một ma trận lọc (kernel). Ma trận lọc lọc (kernel) còn có thể được gọi là cửa sổ chập (trong phép nhân chập), cửa sổ lọc, mặt nạ,... Việc nhân ảnh với ma trận lọc giống như việc trượt ma trận lọc theo hàng trên ảnh và nhân với từng vùng của ảnh, cộng các kết quả lại tạo thành kết quả của điểm ảnh trung tâm.



Minh họa việc nhân ma trận ảnh. Hình ảnh được lấy từ https://github.com/vdumoulin/conv_arithmetic. Hình ảnh động này xem tốt nhất trên web hoặc Microsoft Word.



Trong hình minh họa trên, chúng là có ảnh đầu vào I nằm bên trái, ảnh đầu ra O nằm bên phải. Khi ma trận lọc có kích thước 3×3 , mỗi điểm ảnh trên hình ảnh đầu ra O được tạo thành từ vùng $3 \times 3 = 9$ ô trên ảnh đầu vào I , với các hệ số khác nhau.

Trên thực tế, chúng ta sẽ thấy có 2 phép lọc ảnh là tương quan (correlation) và tích chập (convolution). Với phép tương quan, ma trận lọc sẽ được trượt đi và nhân với từng vùng của ảnh như trên. Tuy nhiên với phép tích chập, ma trận lọc sẽ được xoay 180 độ (theo cả chiều ngang và dọc) trước khi thực hiện nhân. 2 phép toán này là tương đương khi ma trận lọc đối xứng.

Làm mịn ảnh: Lọc trung bình

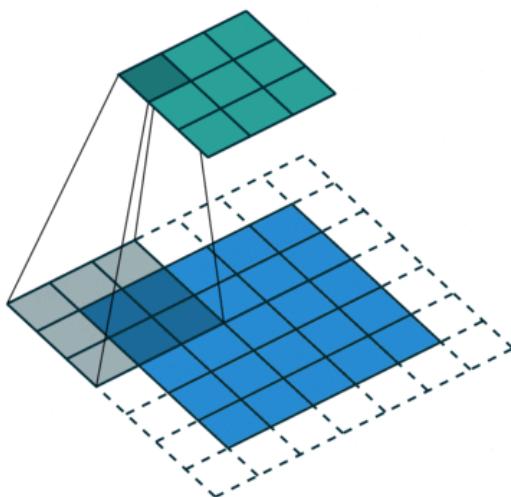
Trong nhiều trường hợp, hình ảnh đầu vào có nhiều nhiễu, chúng ta phải lọc, hay làm mịn để loại bỏ hoặc giảm nhiễu trước khi đưa vào các bước tiếp theo. Bộ lọc trung bình là một trong những bộ lọc được sử dụng để làm việc này. Ma trận lọc trung bình có dạng như sau:

$$K = \frac{1}{K_{width} \cdot K_{height}} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \\ \vdots & \ddots & \ddots & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \end{bmatrix}$$

Trong trường hợp ma trận lọc có kích thước 5×5 , nó sẽ có dạng như sau:

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

Có thể hiểu đơn giản, mỗi điểm ảnh trên ảnh đầu ra O sẽ là trung bình cộng của một vùng đúng bằng kích thước của ma trận lọc. Nếu ma trận lọc có kích thước 5×5 , việc áp dụng bộ lọc trung bình sẽ tương đương với việc tính toán trung bình của từng ô 5×5 trong ảnh và viết vào 1 ô trong ảnh mới. Sau đó, chúng ta lại xét vùng 5×5 tiếp theo ngay bên cạnh, có chồng lấn với vùng cũ. Ở hình động bên dưới, ảnh màu xanh lục chính là ảnh thu được khi trượt các cửa sổ 3×3 trên ảnh màu xanh lam. Trên thực tế, ảnh thu được có thể nhỏ hơn ảnh đầu vào. Để đảm bảo việc này không xảy ra, người ta thường mở rộng ảnh cũ bằng cách chèn các vùng màu vào xung quanh ảnh cũ. Việc này có thể giúp ảnh mới thu được có kích thước đúng bằng ảnh ban đầu.



Thư viện OpenCV cũng hỗ trợ việc lọc ảnh sử dụng bộ lọc trung bình hết sức dễ dàng với hàm `cv.blur()`. Đoạn mã nguồn bên dưới sẽ thực hiện đọc lên 2 hình ảnh. Hình thứ nhất là

`rose_gauss.jpg`, là hình một bông hoa hồng bị nhiễu Gaussian, loại nhiễu ngẫu nhiên gây ra bởi môi trường, chất lượng ống kính, cảm biến hoặc điều kiện ánh sáng không tốt khi chụp ảnh. Hình thứ 2 là `rose_salt_and_pepper.jpg`, cũng là hình ảnh bông hoa đó, nhưng bị nhiễu muối tiêu. Nhiều muối tiêu có đặc điểm phân bố rải rác thành các đốm nhỏ li ti như muối tiêu. Chúng ta sẽ dùng thử bộ lọc trung bình trong OpenCV để lọc nhiễu cho các ảnh này.

```
# Đọc ảnh
img = cv2.imread('rose_gauss.jpg')
img2 = cv2.imread('rose_salt_and_pepper.jpg')

# Áp dụng bộ lọc với ma trận lọc 5x5
blur = cv2.blur(img, (5,5))
blur2 = cv2.blur(img2, (5,5))

# Hiển thị hình ảnh
print("Ảnh ban đầu")
cv2_imshow(np.hstack([img, img2]))
print("Ảnh sau khi lọc")
cv2_imshow(np.hstack([blur, blur2]))
```

Ảnh ban đầu



Hình ảnh sau khi đưa qua bộ lọc trung bình



So sánh các ảnh trước và sau khi lọc, ta có thể thấy nhiễu đã được giảm phần nào nhờ bộ lọc trung bình.

Làm mịn ảnh: Lọc Gaussian

Bộ lọc trung bình có 1 vấn đề là giá trị 1 điểm ảnh có tọa độ (x_1, y_1) trong ảnh đầu ra sẽ phụ thuộc vào một vùng các điểm ảnh xung quanh đó trong ảnh đầu vào với trọng số bằng nhau. Như vậy bất kể các điểm ảnh ở xa hay ở gần vị trí trung tâm (x_1, y_1) đều sẽ đóng góp vào giá trị điểm ảnh mới với tỷ lệ như nhau, chỉ cần chúng nằm trong cửa sổ lọc. Ý tưởng của bộ lọc Gaussian là đánh trọng số lại tỷ lệ này, sao cho các điểm ảnh ở càng gần (x_1, y_1) sẽ chiếm tỷ lệ cao hơn khi tính toán, quyết định đến giá trị của điểm ảnh đầu ra nhiều hơn. Cách làm này cũng phù hợp hơn với logic. Trên thực tế, bộ lọc Gaussian được tạo ra nhờ phân phối chuẩn trong xác suất thống kê.

Ví dụ bộ lọc Gaussian 5×5 sẽ có dạng như sau:

```
kernel = cv2.getGaussianKernel(5, -1) # Tìm ma trận lọc Gaussian 1D
kernel_2D = kernel @ kernel.transpose() # Tìm ma trận lọc Gaussian 2D bằng
# cách nhân 2 ma trận 1D
print(kernel_2D)

[[0.00390625  0.015625   0.0234375  0.015625   0.00390625]
 [0.015625    0.0625     0.09375   0.0625     0.015625  ]
 [0.0234375   0.09375   0.140625  0.09375   0.0234375 ]
 [0.015625    0.0625     0.09375   0.0625     0.015625  ]
 [0.00390625  0.015625   0.0234375  0.015625   0.00390625]]
```

Hàm `cv2.getGaussianKernel()` được sử dụng để lấy ma trận Gaussian 1 chiều (1D). Tham số thứ nhất `ksize` là kích thước bộ lọc (ở đây là 5). Tham số thứ 2 là độ lệch chuẩn `sigma` của phân phối Gaussian. Với điều kiện kích thước bộ lọc đủ lớn, độ lệch chuẩn càng lớn thì điểm ảnh sinh ra càng phụ thuộc nhiều hơn vào các điểm ảnh ở xa, ngược lại, giá trị điểm ảnh thu được sẽ phụ thuộc nhiều hơn vào các điểm ảnh ở gần và coi nhẹ các điểm ở xa nó. Khi `sigma` được truyền vào -1 như trên, giá trị của nó sẽ được tính toán dựa trên ksize với công thức như sau:

$$\text{sigma} = 0.3 * ((\text{ksize}-1)*0.5 - 1) + 0.8$$

Bộ lọc Gaussian được cho là hoạt động hiệu quả nhất với nhiễu Gaussian. Các bạn có thể xem kết quả chạy bộ lọc Gaussian với hình ảnh hoa hồng bên dưới.

```
# Đọc ảnh
img = cv2.imread('rose_gauss.jpg')
img2 = cv2.imread('rose_salt_and_pepper.jpg')

# Áp dụng bộ lọc với ma trận lọc 5x5
gauss_blur = cv2.GaussianBlur(img, (5,5), 0)
gauss_blur2 = cv2.GaussianBlur(img2, (5,5), 0)

# Hiển thị hình ảnh
print("Ảnh ban đầu")
cv2_imshow(np.hstack([img, img2]))
print("Ảnh sau khi lọc")
cv2_imshow(np.hstack([gauss_blur, gauss_blur2]))
```

Ảnh ban đầu



Ảnh sau khi lọc



Có thể thấy, với cùng một kích thước bộ lọc, bộ lọc Gaussian sẽ giữ lại được các chi tiết tốt hơn bộ lọc trung bình.

Làm mịn ảnh: Bộ lọc trung vị

Phép lọc trung vị sẽ tính và lấy trung vị của các giá trị điểm ảnh trong vùng cửa sổ lọc để làm giá trị điểm ảnh đầu ra, có thể thực hiện với 2 bước như sau:

- Sắp xếp các giá trị điểm ảnh theo thứ tự tăng dần.
- Lấy giá trị điểm ảnh nằm giữa.

Trong OpenCV, đối với các ảnh có nhiều kênh màu, như ảnh RGB, HSV, thì các kênh màu được xử lý riêng biệt, sau đó sẽ gộp lại thành ảnh mới. Bộ lọc trung vị luôn lấy một giá trị điểm ảnh có sẵn trong ảnh gốc để đưa vào ảnh mới, thay vì tính toán để tạo ra một giá trị mới như các bộ lọc được giới thiệu trước đó. Bộ lọc này có khả năng loại bỏ nhiễu muối tiêu rất tốt. Trong OpenCV, ta có thể dùng hàm `cv2.medianBlur()` cho bộ lọc này. Dưới đây là ví dụ thực hiện.

```
# Đọc ảnh
img = cv2.imread('rose_gauss.jpg')
img2 = cv2.imread('rose_salt_and_pepper.jpg')

# Áp dụng bộ lọc với ma trận lọc 5x5
median_blur = cv2.medianBlur(img, 5)
median_blur2 = cv2.medianBlur(img2, 5)
```

```
# Hiển thị hình ảnh
print("Ảnh ban đầu")
cv2_imshow(np.hstack([img, img2]))
print("Ảnh sau khi lọc")
cv2_imshow(np.hstack([median_blur, median_blur2]))
```

Ảnh ban đầu



Ảnh sau khi lọc



Có thể thấy, các nhiễu muối tiêu đã được loại bỏ gần như hoàn toàn với bộ lọc trung vị, tốt hơn nhiều so với bộ lọc trung bình và bộ lọc Gaussian.

Làm mịn ảnh: Bộ lọc Bilateral

Quan sát lại các hình ảnh kết quả của 3 bộ lọc trước, các bạn có thể thấy ngoài việc lọc nhiễu, các bộ lọc gây ra tác dụng phụ là làm mờ các cạnh (các biên, các đường nét) trong ảnh. Bộ lọc trung bình tính toán giá trị điểm ảnh mới dựa trên các điểm ảnh cũ với trọng số như sau, không quan tâm đến màu sắc và vị trí tương đối của các điểm ảnh đó với nhau. Bộ lọc Gaussian đã có thể đặt các trọng số dựa theo vị trí: điểm ảnh nào càng ở gần sẽ quyết định nhiều hơn đến giá trị điểm ảnh mới. Tuy nhiên nó cũng chưa quan tâm đến các giá trị màu sắc, và việc điểm ảnh có nằm tại các cạnh trong ảnh không, vì thế cũng sẽ làm mờ các cạnh. Bộ lọc Bilateral sử dụng một bộ lọc Gaussian với khoảng cách đến điểm trung tâm, đảm bảo việc đánh trọng số theo khoảng cách. Đồng thời, nó sử dụng thêm một hàm Gaussian cho mức xám, đánh trọng số cho các mức xám để đảm bảo chỉ các điểm ảnh có mức xám tương đồng với điểm ảnh trung tâm sẽ đóng góp chính vào quá trình tính toán điểm ảnh mới. Tại các biên trong ảnh, mức xám có sự thay đổi rất rõ ràng, vì thế việc đánh trọng số cho theo sự thay đổi mức xám sẽ đảm bảo các vị trí điểm ảnh có mức xám khác nhau không ảnh hưởng đến nhau, vì thế sẽ không làm mờ các cạnh trong ảnh. Chúng ta có thể sử dụng bộ lọc Bilateral thông qua hàm `cv2.bilateralFilter()`, với các tham số:

- **d**: Bán kính xung quanh điểm ảnh sẽ được sử dụng để tính toán.
- **sigmaColor**: Giá trị lọc cho không gian màu sắc. Giá trị này càng lớn, chúng ta sẽ càng loại bỏ được các nhiễu màu sắc trong ảnh, có nghĩa là sẽ càng nhiều điểm ảnh có màu sắc tương đồng trong hình ảnh đầu ra.
- **sigmaSpace**: Giá trị lọc cho không gian khoảng cách. Giá trị này càng lớn, các điểm ảnh ở xa càng có cơ hội đóng góp nhiều hơn vào giá trị điểm ảnh mới, với điều kiện chúng đủ tương đồng về màu sắc (**sigmaColor**). Nếu ta truyền giá trị **d = 0** vào, **d** sẽ được tính toán tỷ lệ thuận với **sigmaSpace**.

Các giá trị sigma sẽ có tác động nhỏ nếu giá trị của chúng thấp (<10). Nếu các giá trị sigma được đặt cao (>150), bộ lọc sẽ có tác động rất mạnh, tạo ra ảnh mới giống như ảnh hoạt hình. Đọc thêm về bộ lọc Bilateral tại đây: <https://www.geeksforgeeks.org/python-bilateral-filtering/>.

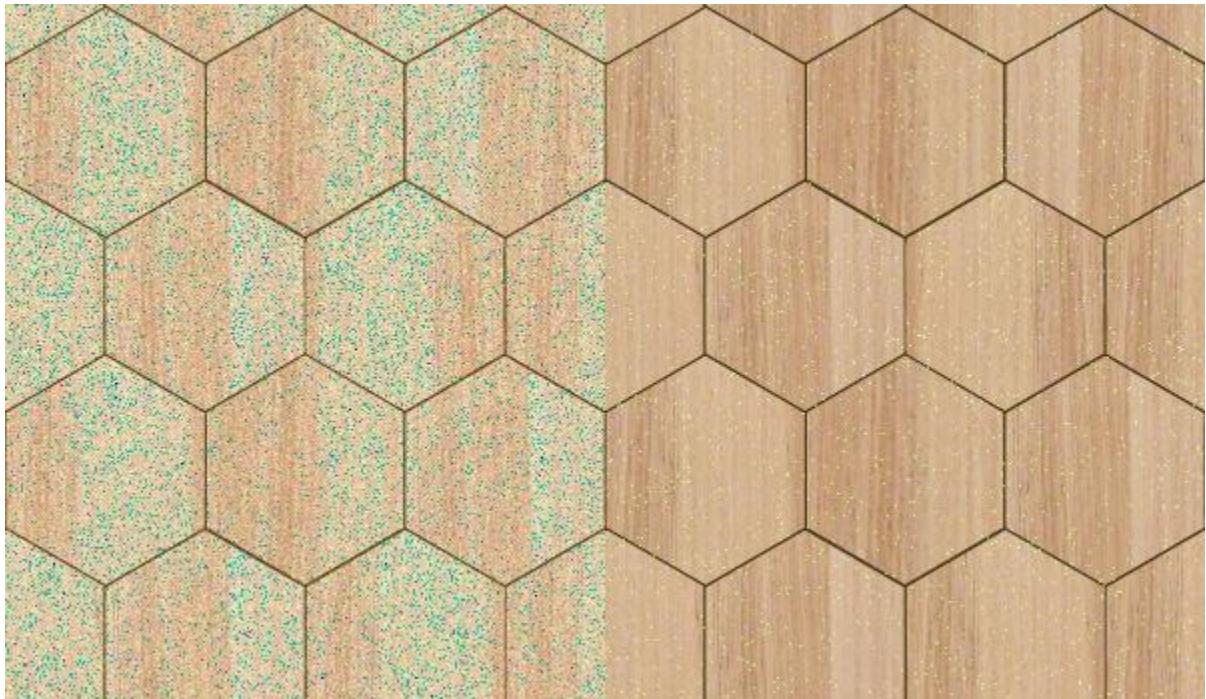
Sau đây là ví dụ về việc sử dụng bộ lọc Bilateral.

```
# Đọc ảnh
img = cv2.imread('wood_gauss.jpg')
img2 = cv2.imread('wood_salt_and_pepper.jpg')
```

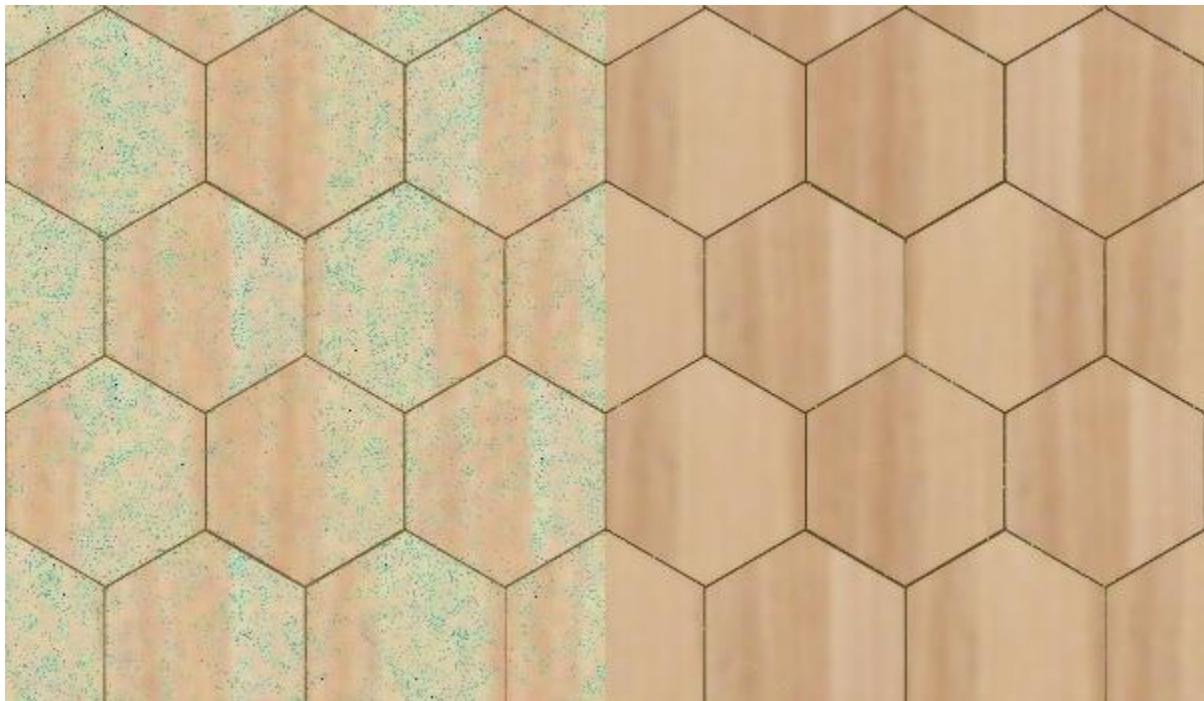
```
# Áp dụng bộ lọc Bilateral
d = 9
sigmaColor = 100
sigmaSpace = 50
bilateral.blur = cv2.bilateralFilter(img, d, sigmaColor, sigmaSpace)
bilateral.blur2 = cv2.bilateralFilter(img2, d, sigmaColor, sigmaSpace)

# Hiển thị hình ảnh
print("Ảnh ban đầu")
cv2_imshow(np.hstack([img, img2]))
print("Ảnh sau khi lọc")
cv2_imshow(np.hstack([bilateral.blur, bilateral.blur2]))
```

Ảnh ban đầu



Ảnh sau khi lọc



Hình ảnh sau khi đưa qua bộ lọc Bilateral





Hình ảnh sau khi đưa qua bộ lọc Gaussian – sử dụng để so sánh

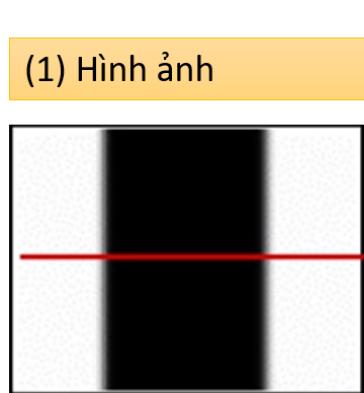
Có thể thấy nhiều trong ảnh được lọc khá tốt, trong khi các đường biên được bảo toàn rất tốt. So sánh với kết quả từ bộ lọc Gaussian 5x5 ở hình sau đó, ta thấy việc bảo toàn biên của bộ lọc Bilateral tốt hơn nhiều. Tuy vậy nhược điểm của bộ lọc Bilateral là tốc độ. Bộ lọc này chạy chậm hơn rất nhiều so với các bộ lọc trước đó trong trường hợp các giá trị sigma lớn. Vì thế, chúng ta cần tính đến yếu tố thời gian chạy khi có ý định sử dụng bộ lọc Bilateral cho các bài toán cần xử lý nhanh. Các bạn có thể thử thay đổi các giá trị **d**, **sigmaColor**, **sigmaSpace** để thấy được sự thay đổi về kết quả và thời gian xử lý.

Tìm cạnh trong ảnh: Bộ lọc Sobel

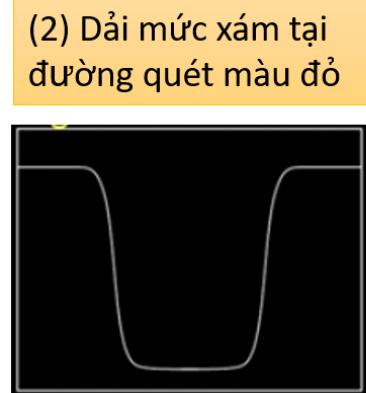
Các bộ lọc cũng được sử dụng để tìm cạnh, hay các đường biên, đường nét trong ảnh. Một trong các bộ lọc phổ biến nhất cho mục đích này là bộ lọc Sobel. Bộ lọc này hoạt động với ảnh xám.

Mối quen hệ giữa đạo hàm và các đường biên: Xét ví dụ sau: Ta có một hình ảnh (1) với hai biên đã được làm mờ. Hình (2) cho thấy mức xám tại đường quét màu đỏ của ảnh. Dễ dàng nhận thấy các đường biên ảnh chính là 2 vùng có sự thay đổi đột ngột về mức xám. Để xác định những sự

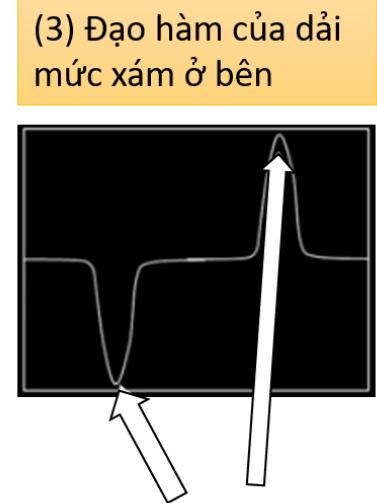
thay đổi này, ta sử dụng đạo hàm của dải mức xám và tìm các cực trị (địa phương) trên đó. Có thể thấy rõ mối liên hệ giữa các cực trị địa phương của đạo hàm với các biên trong ảnh.



(1) Hình ảnh



(2) Dải mức xám tại đường quét màu đỏ



(3) Đạo hàm của dải mức xám ở bên

Các cạnh trong ảnh ứng với các cực trị của đồ thị

Liên hệ giữa cực trị địa phương và các đường biên trong ảnh. Nguồn: Blog AICurious.io.

Vậy là các biên của ảnh sẽ có quan hệ với đạo hàm theo chiều x và đạo hàm theo chiều y của mức xám. Bộ lọc Sobel sử dụng phương pháp tính gradient (hay đạo hàm) của ảnh để làm nổi bật các đường biên của ảnh. Việc tính đạo hàm này thực chất là việc nhân tích chập các bộ lọc vào ảnh. Hàm Sobel trong OpenCV hỗ trợ tính đạo hàm bậc 1 đến bậc 3 với các kích thước bộ lọc khác nhau. Thông thường, các bộ lọc của Sobel cũng được kết hợp với phép lọc Gaussian để tạo ra ma trận lọc tổng hợp, ít nhiều có thể loại bỏ nhiễu trong ảnh. Người ta thường sử dụng 2 bộ cấu hình phổ biến cho bộ lọc này:

- Bộ lọc Sobel 3x3, đạo hàm bậc 1 theo chiều x – dùng để lọc các cạnh dọc:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

- Bộ lọc Sobel 3x3, đạo hàm bậc 1 theo chiều y – dùng để lọc các cạnh ngang:

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Việc tính toán Sobel được thực hiện với hàm `cv2.Sobel()` với các tham số chính như sau:

- **dx:** bậc đạo hàm theo chiều x.
- **dy:** bậc đạo hàm theo chiều y.
- **ksize:** kích thước cửa sổ lọc. Các giá trị cho phép: 1, 3, 5, hoặc 7.

Các bạn xem thêm các tham số khác tại tài liệu chính thức của OpenCV:

https://docs.opencv.org/4.5.4/d4/d86/group_imgproc_filter.html#gacea54f142e81b6758cb6f375ce782c8d

Sau đây là một ví dụ về việc sử dụng bộ lọc Sobel cho phát hiện cạnh.

```
# Đọc ảnh
img = cv2.imread('wood.png')

# Chuyển ảnh đã đọc sang ảnh xám
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Áp dụng bộ lọc Gaussian để loại bỏ bớt nhiễu.
# Các bạn có thể thử nghiệm các bộ lọc khác tại đây,
# như bộ lọc Median hoặc Bilateral.
img_gauss = cv2.GaussianBlur(gray, (5, 5), 0)

# Áp dụng Sobel để tìm cạnh dọc
sobel_ksize = 3
sobelx64f = cv2.Sobel(img_gauss, cv2.CV_64F, 1, 0, ksize=sobel_ksize)
abs_sobel64f = np.absolute(sobelx64f)
img_sobelx = np.uint8(abs_sobel64f)

# Áp dụng Sobel để tìm cạnh ngang
sobely64f = cv2.Sobel(img_gauss, cv2.CV_64F, 0, 1, ksize=sobel_ksize)
abs_sobel64f = np.absolute(sobely64f)
img_sobely = np.uint8(abs_sobel64f)

# Tính toán trung bình cạnh ngang - dọc
img_sobel = (img_sobelx + img_sobely) / 2
```

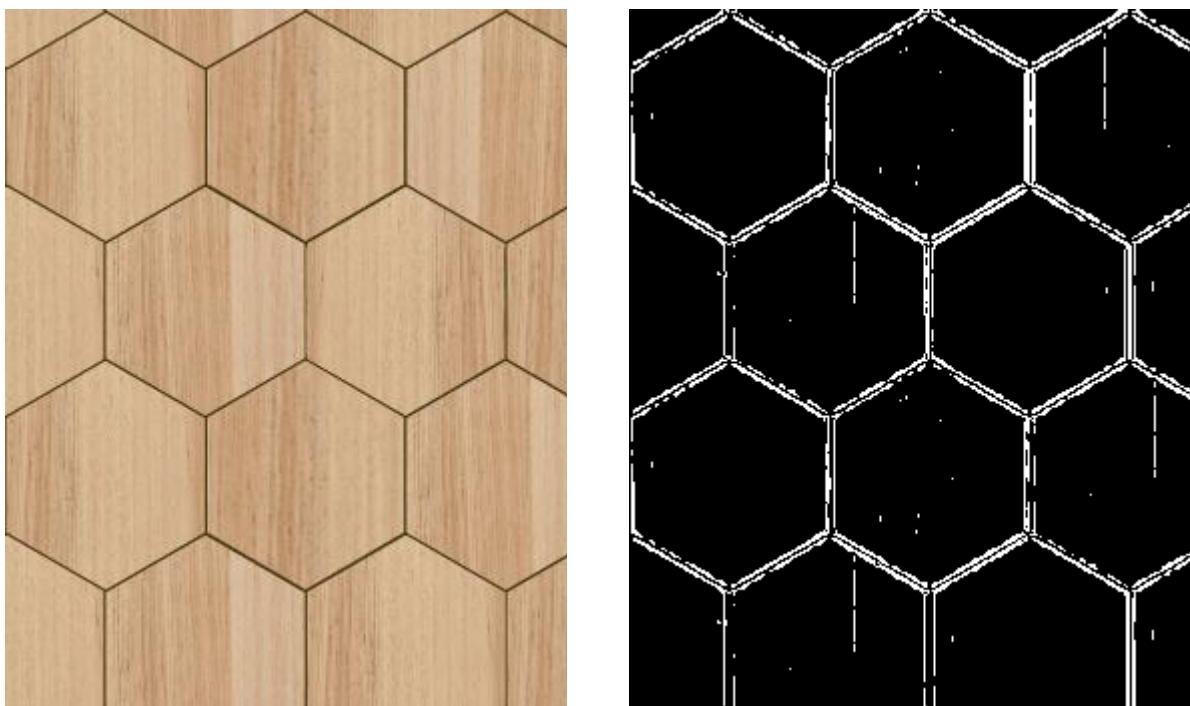
```

# Chuyển về ảnh nhị phân: Lọc ảnh để đặt các vùng có đạo hàm lớn - có sự c
huyển
# đổi màu sắc mạnh về màu trắng (255 - là cạnh) và các vị trí nằm dưới ngư
ỡng
# về màu đen (0 - không phải cạnh)
threshold = 60
img_sobel[img_sobel < threshold] = 0
img_sobel[img_sobel >= threshold] = 255

# Hiện ảnh đầu vào và ảnh kết quả
cv2_imshow(img)
cv2_imshow(img_sobel)

```

Hình ảnh kết quả:



Trong hình ảnh kết quả ta vẫn thấy các nhiễu. Các bạn có thể thử thay bộ lọc Gaussian bằng bộ lọc khác có khả năng lọc nhiễu tốt hơn và giữ lại tốt các đường biên trong ảnh.

Tìm cạnh trong ảnh: Bộ lọc Canny

Bộ lọc Canny là một bộ lọc rất mạnh mẽ trong thị giác máy tính, dùng cho mục đích phát hiện cạnh. Chúng ta có thể gọi bộ lọc này là thuật toán Canny, vì nó gồm nhiều bước:

- Lấy đạo hàm của ảnh theo chiều ngang và dọc.
- Tính cường độ và hướng của đạo hàm (gradient).
- Kết hợp các kết quả để loại bỏ các pixel thừa (Non-maximum suppression).
- Sử dụng các ngưỡng lọc trên (high) và dưới (low) để đưa ra các kết quả cuối cùng: Các cạnh nằm dưới ngưỡng dưới chắc chắn bị loại bỏ. Các cạnh nằm trên ngưỡng trên được gọi là các cạnh mạnh. Các cạnh nằm giữa hai ngưỡng được gọi là cạnh yếu. Một cạnh yếu sẽ chỉ được giữ lại khi nó kết nối với ít nhất 1 cạnh mạnh.

Canny cũng được hỗ trợ sẵn trong OpenCV, giúp các bạn có thể dễ dàng sử dụng. Dưới đây là mã nguồn ví dụ của việc lọc cạnh với bộ lọc Canny.

```
# Đọc ảnh
img = cv2.imread('wood.png')

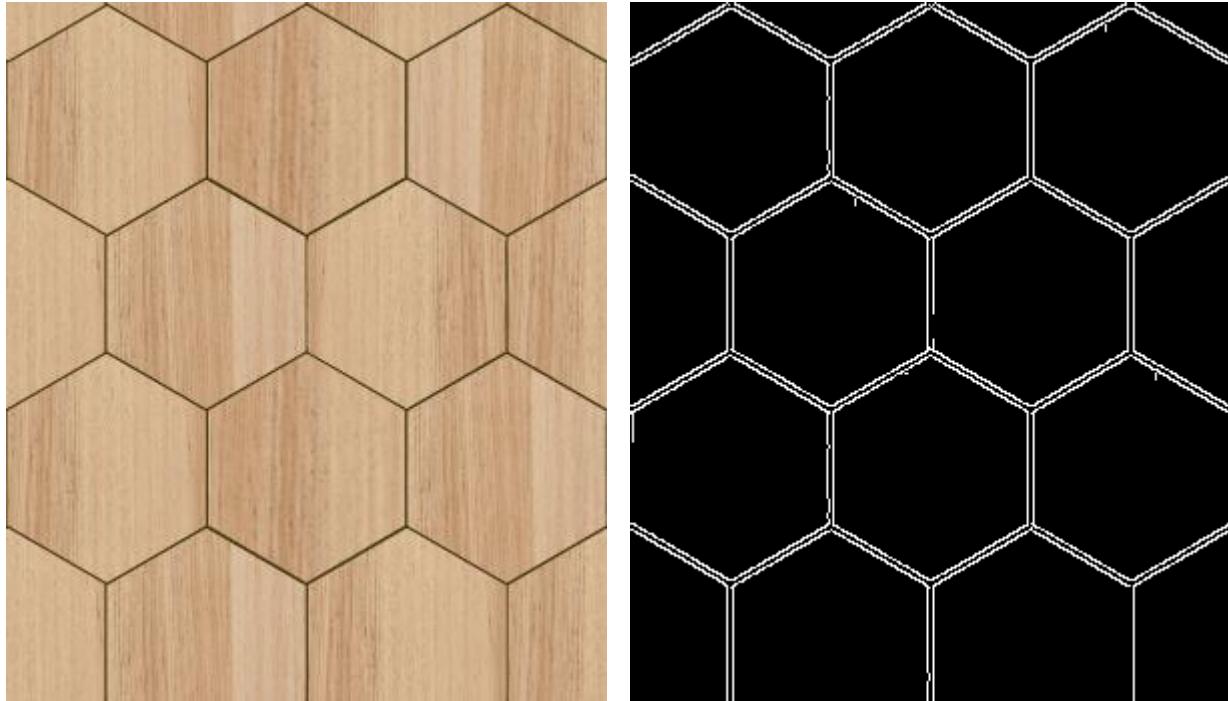
# Chuyển ảnh đã đọc sang ảnh xám
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Áp dụng bộ lọc Gaussian để loại bỏ bớt nhiễu.
# Các bạn có thể thử nghiệm các bộ lọc khác tại đây,
# như bộ lọc Median hoặc Bilateral.
img_gauss = cv2.GaussianBlur(gray, (5, 5), 0)

# Áp dụng bộ lọc Canny với 2 ngưỡng.
# Các bạn có thể điều chỉnh 2 ngưỡng này và xem sự thay đổi
# trong ảnh kết quả
thresh_low = 60
thresh_high = 180
img_canny = cv2.Canny(img_gauss, thresh_low, thresh_high)

# Hiển ảnh đầu vào và ảnh kết quả
cv2_imshow(img)
cv2_imshow(img_canny)
```

Hình ảnh kết quả:



d. Nâng cao: Thuật toán phát hiện đường thẳng với Hough line transform

Phát hiện đường thẳng với Hough line transform: <https://aicurious.io/posts/2019-10-24-hough-transform-phat-hien-duong-thang/>.

3. Phát hiện vạch kẻ đường bằng xử lý ảnh cơ bản

Sau phần 2, chúng ta đã được làm quen với các thuật toán xử lý ảnh cơ bản. Các thuật toán này hoàn toàn có thể được áp dụng để phát hiện lằn đường trong ảnh hết sức dễ dàng. Ở phần này, chúng ta sẽ cùng xây dựng thuật toán phát hiện vạch kẻ đường từ đầu bằng các kĩ thuật xử lý ảnh như chuyển đổi hệ màu, lọc nhiễu, phát hiện cạnh.

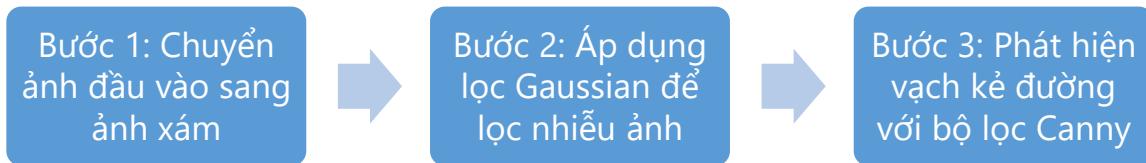
Sổ tay thực hành của bài học này:

<https://colab.research.google.com/github/makerhanoi/via-course-ai/blob/master/notebooks/04-Phat-hien-vach-ke-duong.ipynb>

a. Lọc vạch kẻ đường từ ảnh

Chúng ta sẽ cùng nhau làm quen với luồng xử lý đầu tiên để lọc vạch kẻ đường từ hình ảnh.

Việc xử lý gồm 3 bước chính như sau:



Toàn bộ các bước trên các bạn đã được làm quen qua từng phần nhỏ trong các thuật toán xử lý ảnh cơ bản, đặc biệt là trong phần thuật toán phát hiện cạnh Canny. Trên thực tế, người ta thường triển khai các thuật toán thành các hàm trong Python để tiện sử dụng. Thuật toán phát hiện vạch kẻ đường được viết lại như sau:

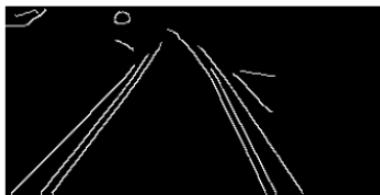
```
def find_lane_lines(img):
    """Phát hiện vạch kẻ đường
    Hàm này sẽ nhận vào một hình ảnh màu, ở hệ màu BGR,
    trả ra hình ảnh các vạch kẻ đường đã được lọc
    """

    # Chuyển ảnh đã đọc sang ảnh xám
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Áp dụng bộ lọc Gaussian để loại bỏ bớt nhiễu.
    # Các bạn có thể thử nghiệm các bộ lọc khác tại đây,
    # như bộ lọc Median hoặc Bilateral.
    img_gauss = cv2.GaussianBlur(gray, (11, 11), 0)

    # Áp dụng bộ lọc Canny với 2 ngưỡng.
    # Các bạn có thể điều chỉnh 2 ngưỡng này và xem sự thay đổi
    # trong ảnh kết quả
    thresh_low = 150
    thresh_high = 200
    img_canny = cv2.Canny(img_gauss, thresh_low, thresh_high)

    # Trả về kết quả vạch kẻ đường
    return img_canny
```



Kết quả phát hiện vạch kẻ đường từ notebook

b. Góc nhìn birdview

Việc nhìn đường từ hướng xiên với mặt đường rất khó để tính toán góc quay phù hợp nhất cho xe. Ta sẽ thay đổi góc nhìn của ảnh từ hướng xiên tới hướng vuông góc với mặt đường bằng Birdview transform.

- Đọc thêm về biến đổi góc nhìn với 4 điểm:

<https://www.pyimagesearch.com/2014/08/25/4-point-opencv-getperspective-transform-example/>

- Các phép biến đổi hình học:

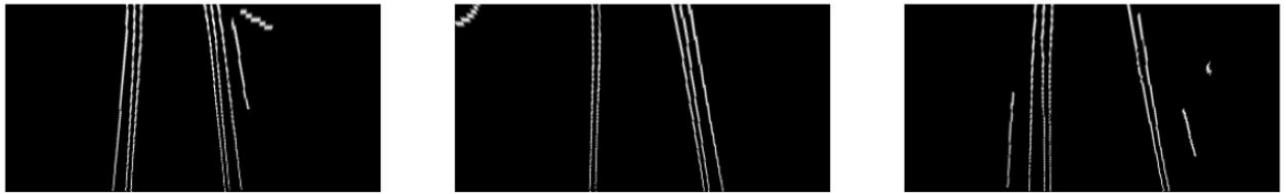
<https://phamdinhkhanh.github.io/2020/01/06/ImagePreprocessing.html#21-c%C3%A1c-bi%E1%BA%BFn-%C4%91%E1%BB%95i-h%C3%ACnh-h%E1%BB%8Dc>



Chúng ta lại cài đặt bước biến đổi birdview thành một hàm khác:

```
def birdview_transform(img):
    IMAGE_H = 160
    IMAGE_W = 320
    src = np.float32([[0, IMAGE_H], [320, IMAGE_H], [0, IMAGE_H//3], [IMAGE_W, IMAGE_H//3]])
    dst = np.float32([[90, IMAGE_H], [230, IMAGE_H], [-10, 0], [IMAGE_W+10, 0]])
    M = cv2.getPerspectiveTransform(src, dst) # The transformation matrix
    warped_img = cv2.warpPerspective(img, M, (IMAGE_W, IMAGE_H)) # Image warping
    return warped_img

# Áp dụng các biến đổi birdview với hình ảnh kết quả của bộ lọc Canny
birdview_images = [birdview_transform(img) for img in list_img_lines]
show_images(birdview_images, cmap="gray")
```



Kết quả sau khi biến đổi birdview

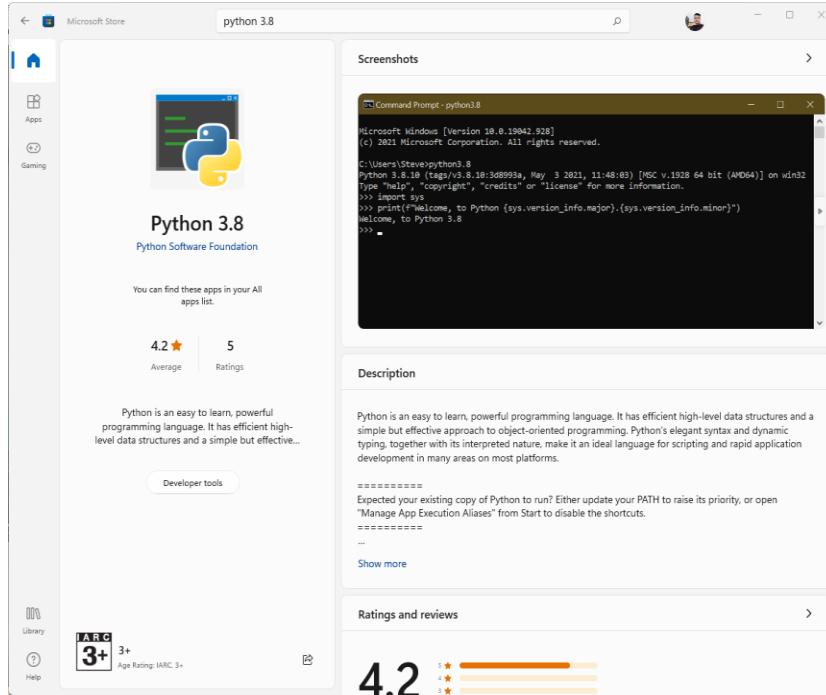
c. Chạy thử thuật toán trên giả lập

Sử dụng các trình giả lập giống như trò chơi đua xe để chạy thử các thuật toán là cách thức phổ biến mà các công ty phát triển xe tự lái thường sử dụng. Trong khóa học này, chúng ta sẽ sử dụng trình giả lập được phát triển bởi đội ngũ VIA để thử nghiệm các thuật toán. Giả lập có thể được sử dụng online tại địa chỉ sau:

<https://via-sim.makerviet.org/>

Cài đặt môi trường chạy Python: Để chạy thử thuật toán phát hiện vạch kẻ đường, chúng ta cần đưa toàn bộ các thuật toán xuống máy tính và chạy ở môi trường Python trên máy tính. Trước tiên, các bạn cần cài đặt Python cho máy tính của mình và trình quản lý gói PIP.

- **Trên Windows 10/11:** Cài Python 3.8 và PIP từ cửa hàng của Microsoft tại địa chỉ sau: <https://www.microsoft.com/vi-vn/p/python-38/9msszt1n39l>. Chọn **Cài đặt / Mở** để vào ứng dụng cửa hàng và cài đặt Python hoàn toàn miễn phí.



Cài đặt Python trên cửa hàng Microsoft.

- **Cài đặt trên các hệ điều hành khác:**
 - macOS: <https://www.python.org/downloads/macos/>
 - Ubuntu: Mở Terminal và gõ:

```
sudo apt install python3-dev python3-pip
```

Các bạn cũng cần một trình chỉnh sửa mã nguồn hiệu quả. **Visual Studio Code** có thể là phần mềm khá phù hợp để các bạn làm việc này. Tải về và cài đặt tại địa chỉ sau: <https://code.visualstudio.com/>.

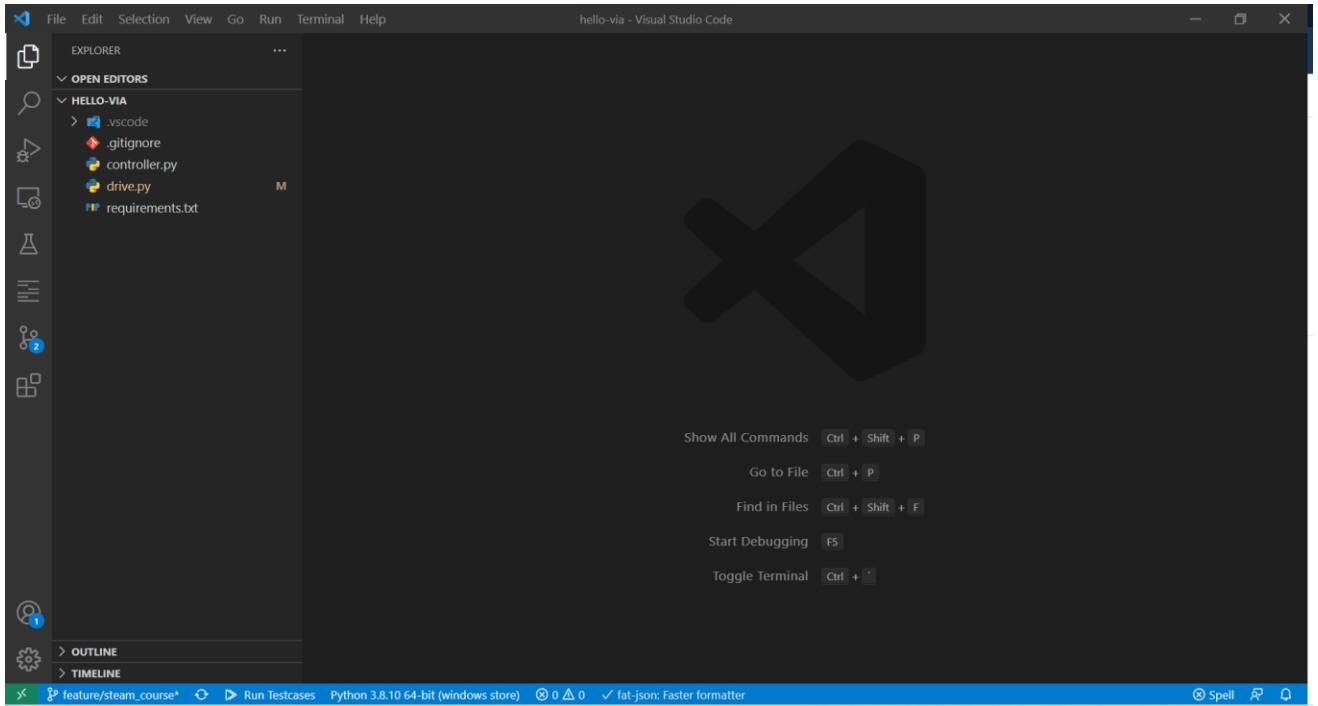
Tải và chạy thử mã nguồn:

Tải mã nguồn mẫu tại đây:

https://github.com/makerhanoi/via-course-ai/raw/master/storage/p1_lane_line_detection.zip

Toàn bộ mã nguồn xử lý ảnh chúng ta đã xây dựng đã được ghép lại, đồng thời bổ sung thêm một số tính năng phụ giúp các bạn kết nối, điều khiển xe trong giả lập và thử nghiệm mã nguồn một cách dễ dàng.

Sau khi tải mã nguồn về, các bạn mở toàn bộ thư mục chứa mã nguồn với trình chỉnh sửa **Visual Studio Code**. Giao diện hiển thị sẽ tương tự như hình dưới.



Để cài đặt các gói Python cần thiết, chọn menu **View > Terminal** sau đó gõ các lệnh sau vào vùng **Terminal** phía dưới màn hình.

```
pip install --upgrade pip  
pip install -r requirements.txt
```

Chạy mã nguồn lên với lệnh sau:

```
python drive.py
```

Tiếp đó tải lại cửa sổ giả lập bằng cách ấn **F5** hoặc **Ctrl + R**, tạo hoặc chọn bản đồ có sẵn. Các bạn sẽ thấy có cửa sổ kết quả xử lý ảnh hiện lên khi kết nối thành công. Để di chuyển xe, các bạn click chuột vào vùng cửa sổ của giả lập và dùng các phím **W, A, S, D** để điều khiển xe theo ý

muốn và xem các kết quả xử lý ảnh hiện lên nhé. Để xem các kết quả trung gian (các hình ảnh tạo ra trong quá trình xử lý), các bạn thêm các dòng sau vào mỗi bước muốn xem. Thay thế `img` bằng hình ảnh muốn hiển thị. Lưu ý “tên cửa sổ hình ảnh” cần khác nhau giữa các hình để mỗi hình ảnh được hiển thị trên một cửa sổ khác nhau trong quá trình chạy.

```
cv2.imshow("<tên cửa sổ hình ảnh>", img)  
cv2.waitKey(1)
```



Hình ảnh kết quả birdview và trình giả lập

d. Nâng cao: Phát hiện vạch kẻ đường sử dụng các mô hình học sâu

Các phương pháp xử lý ảnh có ưu điểm dễ hiểu và dễ cài đặt, tuy nhiên khá nhạy cảm với nhiều và các yếu tố môi trường như điều kiện ánh sáng, màu sắc. Vì thế, để đảm bảo hệ thống xe tự lái hoạt động tốt trong các điều kiện phức tạp của đường xá, người ta cần ứng dụng các kỹ thuật phức tạp hơn sử dụng các mô hình học máy, học sâu. Việc phát hiện làn đường có thể được thực hiện bằng các mạng học sâu như U-Net, E-Net, PiNet. Các bạn có thể tham khảo thêm về phát triển xe tự lái trên giả lập bằng các mô hình học sâu tại liên kết sau: <https://aicurious.io/posts/chung-toi-da-xay-dung-xe-tu-hanh-tren-gia-lap-the-nao/>.

Lập trình điều khiển xe bám làn đường

Trong phần này, chúng ta sẽ cùng làm quen với các thuật toán đơn giản để lập trình bám làn đường từ kết quả phát hiện vạch kẻ đường. Tiếp đó, chúng ta lại cùng quay lại với giả lập VIA Simulation để thử nghiệm các thuật toán của mình, điều khiển xe chạy theo làn đường với nhiều điều kiện khác nhau.

a. Duyệt ảnh birdview để tìm vạch kẻ đường bên trái và bên phải

Sau khi có được ảnh birdview, một phương pháp đơn giản giúp xác định vị trí xe so với làn đường là tìm 2 điểm, một điểm thuộc vạch kẻ đường bên trái và một điểm thuộc vạch kẻ đường bên phải. Sau cùng, ta xét vị trí của điểm giữa xe so với hai điểm đó để xác định độ lệch của xe so với làn đường và tính toán góc lái phù hợp.

Đoạn code bên dưới sẽ thực hiện xét một vạch kẻ ngang bức ảnh, cách mép trên ảnh một khoảng bằng 70% chiều cao ảnh. Tiếp đó, chúng ta tính vị trí tâm ảnh. Từ vị trí tâm này, ta duyệt sang 2 bên, tìm điểm ảnh có giá trị khác 0 đầu tiên, và coi đó là vị trí vạch kẻ đường bên trái và bên phải. Ta định nghĩa trước độ rộng đường lane_width = 100. Như vậy có thể dùng độ rộng này để tìm một trong hai điểm trái / phải khi chỉ nhìn thấy điểm còn lại.

```
def find_left_right_points(image, draw=False):  
  
    im_height, im_width = image.shape[:2]  
    if draw: viz_img = cv2.cvtColor(image, cv2.COLOR_GRAY2BGR)  
  
    # Vạch kẻ sử dụng để xác định tâm đường  
    interested_line_y = int(im_height * 0.7)  
    if draw: cv2.line(viz_img, (0, interested_line_y), (im_width, interested_line_y), (0, 0, 255), 2)  
    interested_line = image[interested_line_y, :]  
  
    # Xác định điểm bên trái và bên phải  
    left_point = -1  
    right_point = -1  
    lane_width = 100  
    center = im_width // 2  
  
    # Tìm điểm bên trái và bên phải bằng cách duyệt từ tâm ra
```

```

for x in range(center, 0, -1):
    if interested_line[x] > 0:
        left_point = x
        break
for x in range(center + 1, im_width):
    if interested_line[x] > 0:
        right_point = x
        break

# Dự đoán điểm bên phải khi chỉ nhìn thấy điểm bên trái
if left_point != -1 and right_point == -1:
    right_point = left_point + lane_width

# Dự đoán điểm bên trái khi chỉ thấy điểm bên phải
if right_point != -1 and left_point == -1:
    left_point = right_point - lane_width

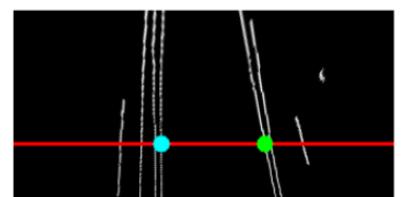
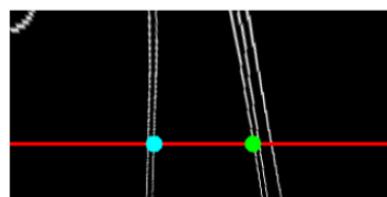
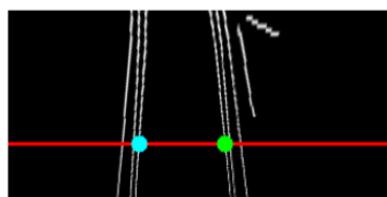
# Vẽ hai điểm trái / phải lên ảnh
if draw:
    if left_point != -1:
        viz_img = cv2.circle(viz_img, (left_point, interested_line_y),
7, (255,255,0), -1)
    if right_point != -1:
        viz_img = cv2.circle(viz_img, (right_point, interested_line_y),
7, (0,255,0), -1)

return left_point, right_point, viz_img

viz_images = []
for img in birdview_images:
    left_point, right_point, viz_img = find_left_right_points(img, draw=True)
    viz_images.append(viz_img)

show_images(viz_images)

```



Kết quả tìm kiếm hai điểm trái / phải

b. Tính toán góc lái và tốc độ

Chúng ta sẽ viết thêm hàm `calculate_control_signal()` để tính toán tốc độ và góc lái cho xe dựa trên ảnh đầu vào. Sau bước `find_lane_lines()` sử dụng Canny để tìm kiếm các vạch kẻ đường, hình ảnh sẽ được đưa qua `birdview_transform()` để thực hiện chuyển đổi birdview, lấy góc nhìn từ trên xuống. Tiếp đó là thuật toán tìm kiếm các điểm trái / phải của làn đường thông qua hàm `find_left_right_points()`. Từ hai điểm trái và phải, chúng ta hoàn toàn có thể tính toán được độ lệch của xe so với mặt đường (coi camera ở chính giữa xe, thì tâm ảnh `im_center` cũng là tâm của xe):

```
# Tính toán độ lệch giữa điểm giữa xe và làn đường
center_point = (right_point + left_point) // 2
center_diff = im_center - center_point
```

Góc lái sẽ được tính toán theo tỷ lệ với độ lệch. Ở đây là 1% độ lệch. Dấu – thể hiện góc lái sẽ theo hướng ngược lại với độ lệch của xe.

```
# Tính toán góc lái tỷ lệ với độ lệch
steering_angle = - float(center_diff * 0.01)
```

Sau đây là toàn bộ mã nguồn cho hàm `calculate_control_signal()`.

```
def calculate_control_signal(img):
    """Tính toán để tìm tốc độ và góc lái cho xe từ ảnh đầu vào
    """

    # Xử lý ảnh để tìm các điểm trái / phải
    img_lines = find_lane_lines(img)
    img_birdview = birdview_transform(img_lines)
    left_point, right_point, viz_img = find_left_right_points(img_birdview,
        draw=True)

    # Hiển thị kết quả xử lý ảnh
    cv2.imshow("Result", viz_img)
    cv2.waitKey(1)

    # Tính toán tốc độ và góc lái
    throttle = 0.5 # Tốc độ đang được đặt là 50% tốc độ cao nhất
    steering_angle = 0
    im_center = img.shape[1] // 2
    # Nếu tìm thấy điểm trái và điểm phải,
```

```

# các điểm này sẽ có giá trị khác -1
if left_point != -1 and right_point != -1:

    # Tính toán độ lệch giữa điểm giữa xe và làn đường
    center_point = (right_point + left_point) // 2
    center_diff = im_center - center_point

    # Tính toán góc lái tỷ lệ với độ lệch
    steering_angle = - float(center_diff * 0.01)

return throttle, steering_angle

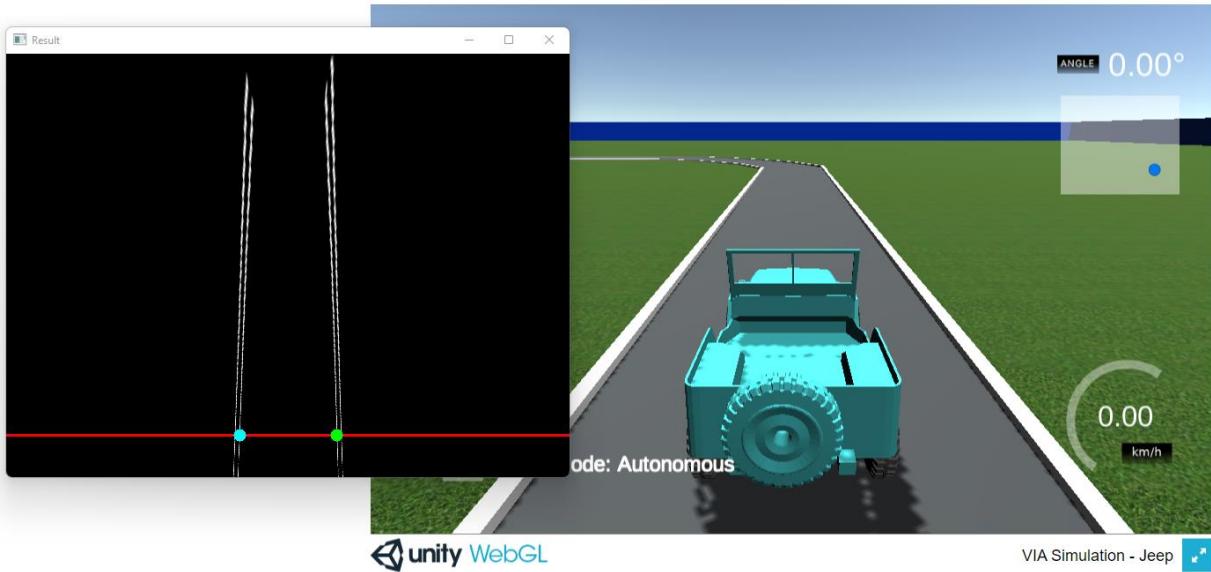
```

c. *Thử nghiệm mã nguồn trên giả lập*

Làm theo các bước như ở phần trước, chúng ta tải mã nguồn đã ghép từ liên kết sau, sau đó giải nén và chạy mã nguồn với giả lập của VIA để thử nghiệm việc điều khiển xe bám theo làn đường. Các bạn có thể chạy đổi hệ số góc lái so với độ lệch (hiện là 1%) và tốc độ tối đa để xem sự ảnh hưởng của chúng tới việc điều khiển xe nhé!

Mã nguồn phần này:

https://github.com/makerhanoi/via-course-ai/raw/master/storage/p2_control.zip



Kết quả xác định hai điểm trái, phải và điều khiển xe trên già lập VIA Simulation

d. Giới thiệu thuật toán PID

Chúng ta đã xây dựng được đầy đủ luồng chạy xác định làn đường và điều khiển xe dựa vào độ lệch của xe với tâm đường theo một tỷ lệ cố định. Trên thực tế, người ta thường dùng các bộ điều khiển phức tạp hơn, ví dụ như bộ điều khiển PID, nhằm giúp việc điều khiển mượt mà hơn dựa vào sự phản hồi. Bộ điều khiển PID sẽ tính toán giá trị "sai số" là hiệu số giữa giá trị đo thông số biến đổi và giá trị đặt mong muốn. Bộ điều khiển sẽ thực hiện giảm tối đa sai số bằng cách điều chỉnh giá trị điều khiển đầu vào. Việc điều chỉnh của PID dựa vào ba khâu: tỉ lệ, tích phân, đạo hàm, và phải được tinh chỉnh theo các ứng dụng điều khiển và môi trường khác nhau.

Các bạn có thể đọc thêm về bộ điều khiển PID, tìm kiếm các phiên bản cài đặt bằng Python, từ đó áp dụng vào mã nguồn có sẵn để thực hiện điều khiển xe.

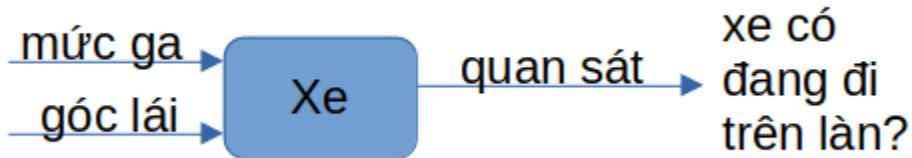
https://vi.wikipedia.org/wiki/B%E1%BB%99_%C4%91i%E1%BB%81u_khi%E1%BB%83n_PID

Cách cài đặt PID cho xe tự lái VIA

Khảo sát bài toán

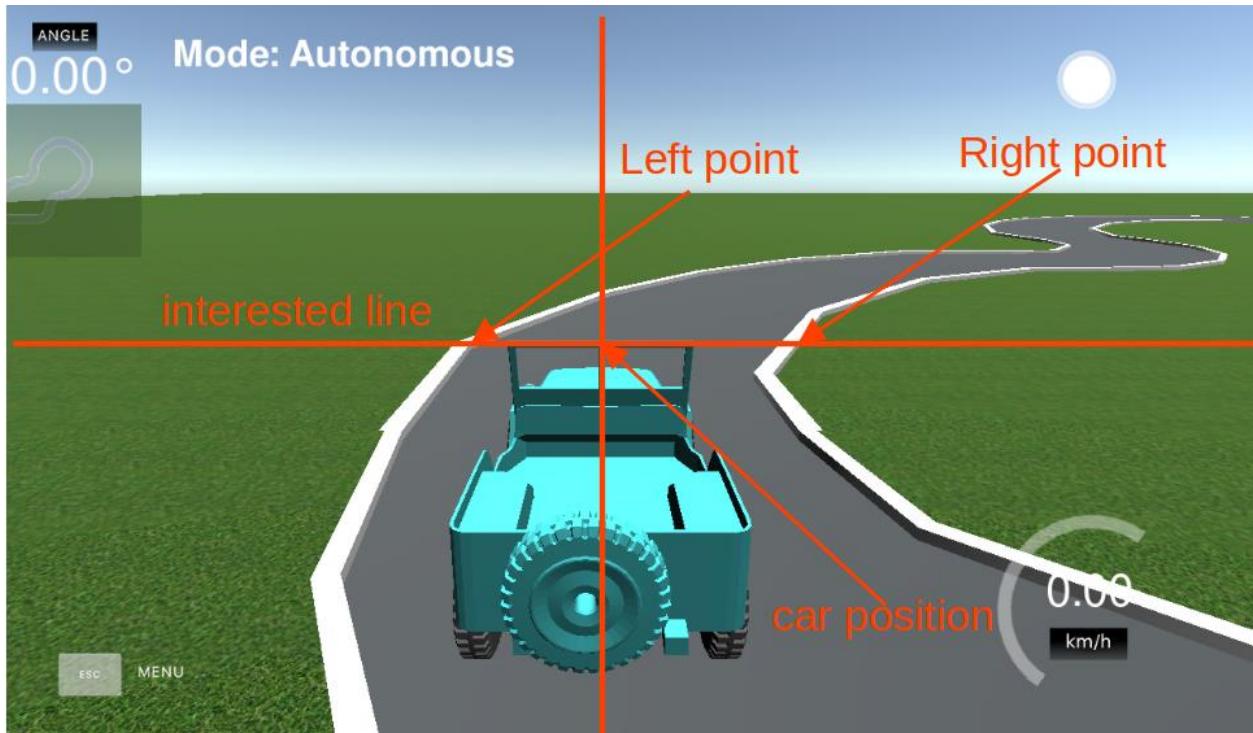
Trước tiên chúng ta cần tìm hiểu bài toán, mô hình hóa hệ thống cần điều khiển, các yếu tố đầu vào đầu ra và mục tiêu điều khiển.

Sau khi tạo map, rõ ràng, mục đích của điều khiển chính là giữ cho xe có một tốc độ ổn định mà không bị văng ra khỏi làn đường. Với điều kiện như vậy, hệ thống xe của chúng ta được phân tích như sau:



Hai đầu vào giúp ta điều khiển xe là mức ga (throttle) và góc đánh lái (steering angle), ở phía đầu ra, thông qua các thuật toán xử lý ảnh, ta sẽ có thông tin về làn đường (điểm tận cùng bên trái và bên phải của làn).

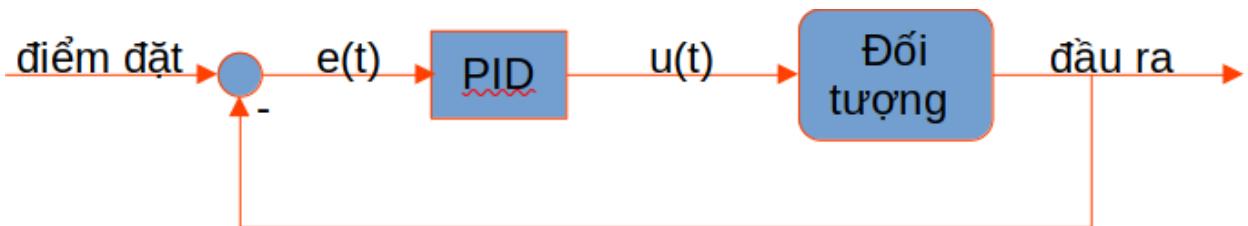
Vì vậy, bộ điều khiển xe cần tính toán liên tục các giá trị **throttle** và **steering angle** để giữ cho xe luôn trong làn đường, tức là điểm giữa của camera (chính là tâm xe) phải nằm ở giữa điểm trái và phải của làn.



Để xe có thể chạy trong làn và đạt vận tốc tốt nhất, có rất nhiều phương án điều khiển có thể áp dụng được, tuy nhiên cần cung cấp phương trình động học liên quan đến xe (mối liên hệ giữa đầu vào và đầu ra). Nếu đơn giản hóa bài toán, giả sử ta cố định vận tốc là hằng số (đặt throttle cố định) và chỉ điều khiển góc lái, khi đó có thể áp dụng một bộ điều khiển đơn giản để giữ cho xe luôn không lệch khỏi làn. Ví dụ như bộ điều khiển PID dưới đây.

Thiết kế bộ điều khiển PID

PID là chữ viết tắt của ba thành phần cơ bản có trong bộ điều khiển gồm khâu khuếch đại (P), khâu tích phân (I) và khâu vi phân (D).



$$u(t) = MV(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{d}{dt} e(t)$$

Trong bài toán này, ta cần điều khiển cho xe bám làn, vì vậy đầu ra sẽ thiết kế là sai lệch giữa tâm xe và tâm làn đường. Nếu sai lệch bằng 0 thì xe sẽ luôn đi giữa làn, vì vậy đặt

- Điểm đặt bằng 0.
- Sai lệch điều khiển $e = \text{điểm đặt} - \text{đầu ra}$.
- Tín hiệu điều khiển xe là u chính là góc lái.

Để bộ điều khiển đạt chất lượng tốt nhất, ta tiến hành chạy mô phỏng và hiệu chỉnh các tham số K_p, K_i và K_d . Video dưới đây hướng dẫn cách hiệu chỉnh các tham số này.

- Hướng dẫn về PID

https://drive.google.com/file/d/1lmrg7_Rm4gY9DS_pebnZbf7bVhKYqCw5/view?usp=sharing

- Hướng dẫn chỉnh sửa mã nguồn

https://drive.google.com/file/d/1TrAMmcKNjUhnkt0TPbTyrZfnBadN_ltX/view?usp=sharing

Phát hiện biển báo giao thông

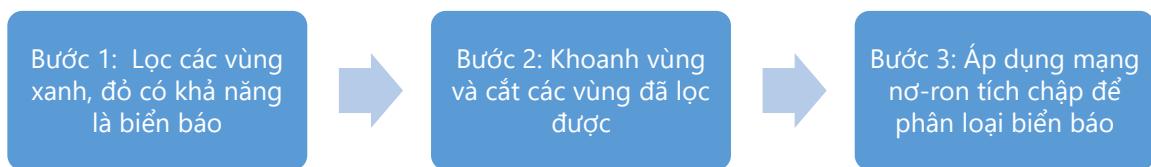
Ngoài việc đi đúng phần đường, làn đường của mình, các xe tự lái còn phải tuân thủ các hiệu lệnh giao thông như biển báo, đèn tín hiệu. Ở phần này, chúng ta sẽ ứng dụng các kỹ thuật xử lý ảnh và trí tuệ nhân tạo để phát hiện và phân loại các biển báo có trên đường và điều khiển xe theo chỉ dẫn của các biển báo đó.

Sổ tay Colab thực hành:

<https://colab.research.google.com/github/makerhanoi/via-course-ai/blob/master/notebooks/05-Phat-hien-bien-bao.ipynb>

a. Luồng phát hiện và phân loại biển báo cơ bản

Luồng phát hiện và phân loại biển báo cơ bản sẽ gồm 3 bước chính:



Ở bước thứ nhất, hình ảnh sẽ được đưa vào lọc màu để chọn ra toàn bộ các vùng có màu xanh hoặc màu đỏ, có thể là biển báo giao thông. Tiếp đó, các vùng này được cắt rời ra thành các vùng ảnh nhỏ tại bước 2 và đưa vào một mạng nơ-ron tích chập ở bước 3 để thực hiện phân loại.

b. Phát hiện biển báo thông qua màu sắc

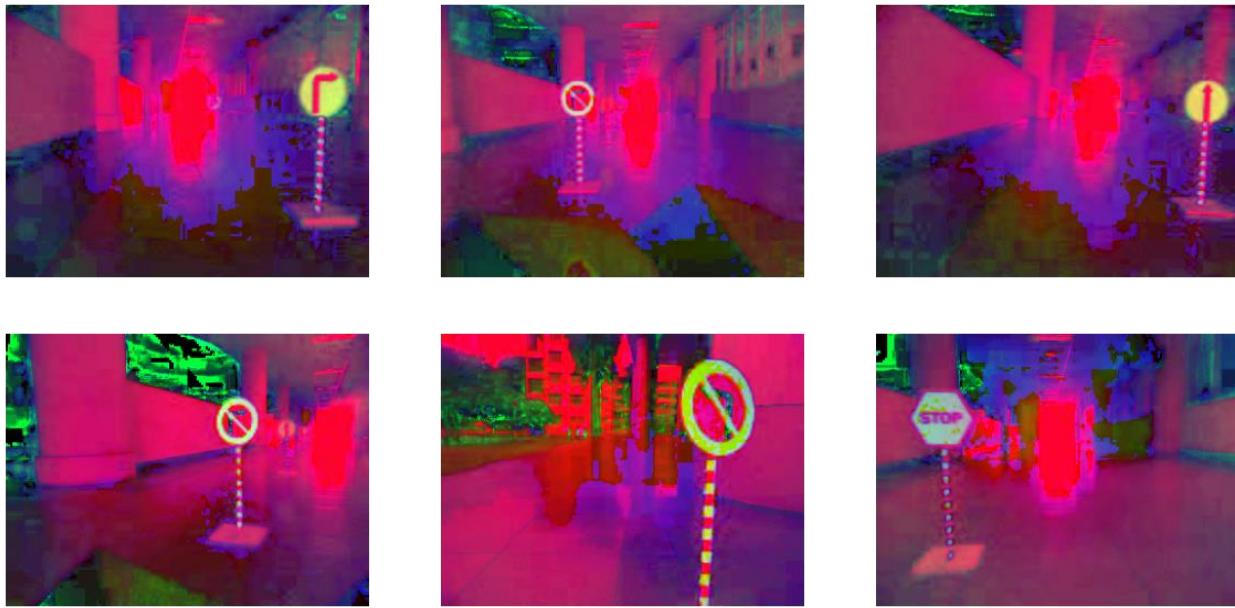
Như chúng ta đã biết, hệ màu HSV có khả năng phân biệt tốt các màu sắc, phù hợp với nhận thức của mắt người, để phát hiện các đối tượng biển báo thông qua màu sắc, ta đưa ảnh về hệ màu HSV và thực hiện lọc màu sắc theo các ngưỡng trong hệ màu đó.

Trước tiên, chúng ta sẽ tải về và sử dụng một vài hình ảnh mẫu để thử nghiệm các thuật toán. Các hình ảnh dưới đây được lấy tự bộ dữ liệu [VIA dataset](#). Các hình ảnh chúng ta sẽ thử nghiệm được đóng góp bởi các thành viên nhóm ICT K60, Đại học Bách khoa Hà Nội. Các hình ảnh cũng được chụp trong khuôn viên của trường.

```
!wget https://github.com/vietanhdev/sdc-tutorial/raw/master/content/images/traffic_sign_images.zip  
!unzip traffic_sign_images.zip
```



Để chuyển đổi các ảnh trên sang hệ màu HSV, trước khi thực hiện lọc theo màu sắc, ta sử dụng hàm `cv2.cvtColor()` đã được làm quen trong bài học xử lý ảnh cơ bản.



Sau khi chuyển ảnh sang hệ màu HSV, ta sẽ khảo sát và chọn ngưỡng màu để lọc ra các màu sắc cần thiết. Mỗi khoảng màu cần lọc sẽ bao gồm một ngưỡng dưới (lower) và một ngưỡng trên (upper).

Với các đối tượng có màu đỏ, ta lọc bằng 2 khoảng màu khác nhau:

```
lower1, upper1 = np.array([0,70,50]), np.array([10,255,255])
lower2, upper2 = np.array([170,70,50]), np.array([180,255,255])
```

Với các đối tượng có màu xanh dương:

```
lower3, upper3 = np.array([85,50,200]), np.array([135,250,250])
```

Thông qua hàm filter_signs_by_color() ta sẽ thu được một mặt nạ lọc, là một hình ảnh nhị phân có kích thước bằng kích thước ảnh ban đầu. Các điểm ảnh trong mặt nạ này sẽ có 2 giá trị là 0 (điểm ảnh tương ứng trong ảnh gốc không nằm trong khoảng màu cần lọc) và 255 (điểm ảnh tương ứng trong ảnh gốc nằm trong khoảng màu cần lọc).

```
def filter_signs_by_color(image):
    """Lọc các đối tượng màu đỏ và màu xanh dương - Có thể là biển báo.
    Ảnh đầu vào là ảnh màu BGR
    """
    # Chuyển ảnh sang hệ màu HSV
    image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
```

```

# Lọc màu đỏ cho biển báo cấm
lower1, upper1 = np.array([0, 70, 50]), np.array([10, 255, 255])
lower2, upper2 = np.array([170, 70, 50]), np.array([180, 255, 255])
mask_1 = cv2.inRange(image, lower1, upper1) # dải màu đỏ thứ nhất
mask_2 = cv2.inRange(image, lower2, upper2) # dải màu đỏ thứ hai
mask_r = cv2.bitwise_or(mask_1, mask_2) # kết hợp 2 kết quả từ 2 dải màu khác nhau

# Lọc màu xanh cho biển báo điều hướng
lower3, upper3 = np.array([85, 50, 200]), np.array([135, 250, 250])
mask_b = cv2.inRange(image, lower3, upper3)

# Kết hợp các kết quả
mask_final = cv2.bitwise_or(mask_r, mask_b)
return mask_final

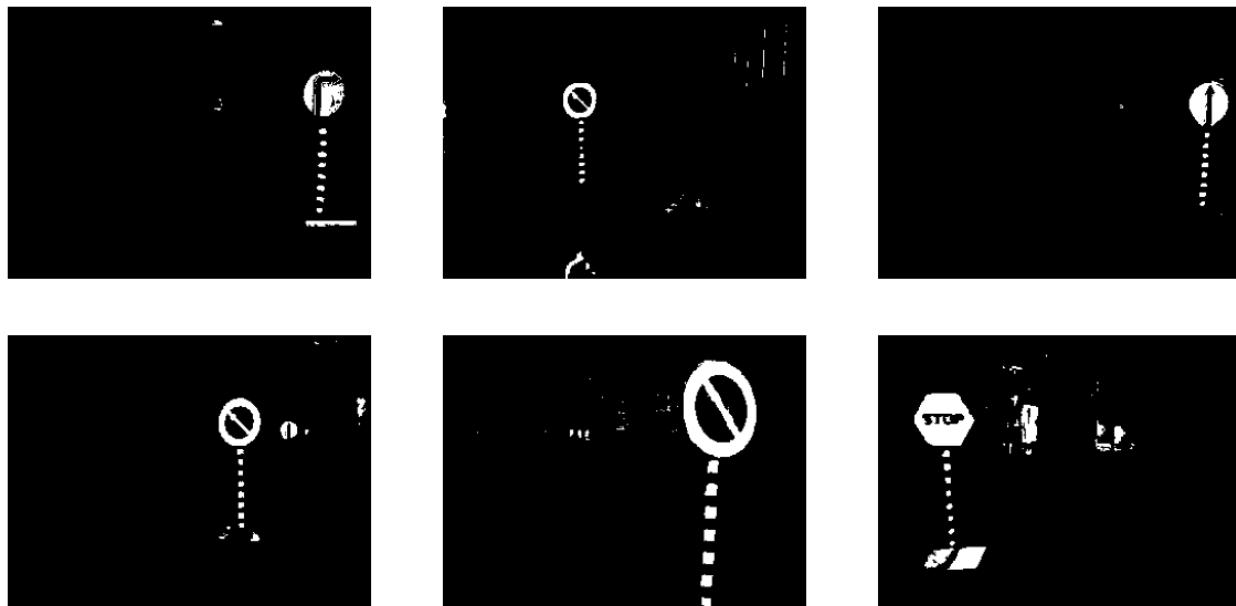
```

```

masks = [filter_signs_by_color(img) for img in bgr_images]
show_images(masks)

```

Kết quả:



Sau khi lọc các biển báo theo màu sắc, ta có thể nhìn thấy các nhiễu là các vùng có màu giống biển báo, là cột biển, hoặc các vùng nền. Ở bước tiếp theo, ta cần đi lọc các nhiễu này và tìm ra vùng chữ nhật bao quanh biển báo, thường được gọi là bounding box của biển báo.

```

def get_boxes_from_mask(mask):
    """Tìm kiếm hộp bao biển báo
    """
    bboxes = []

    nccomps = cv2.connectedComponentsWithStats(mask, 4, cv2.CV_32S)
    numLabels, labels, stats, centroids = nccomps
    im_height, im_width = mask.shape[:2]
    for i in range(numLabels):
        x = stats[i, cv2.CC_STAT_LEFT]
        y = stats[i, cv2.CC_STAT_TOP]
        w = stats[i, cv2.CC_STAT_WIDTH]
        h = stats[i, cv2.CC_STAT_HEIGHT]
        area = stats[i, cv2.CC_STAT_AREA]
        # Lọc các vật quá nhỏ, có thể là nhiễu
        if w < 20 or h < 20:
            continue
        # Lọc các vật quá lớn
        if w > 0.8 * im_width or h > 0.8 * im_height:
            continue
        # Loại bỏ các vật có tỷ lệ dài / rộng quá khác biệt
        if w / h > 2.0 or h / w > 2.0:
            continue
        bboxes.append([x, y, w, h])
    return bboxes

```

Ta tiến hành vẽ các kết quả lọc và tìm kiếm ở trên

```

results = []
for i, img in enumerate(bgr_images):
    mask = filter_signs_by_color(img) # lọc theo màu sắc
    bboxes = get_boxes_from_mask(mask) # tìm kiếm khung bao của các vật từ
    # mặt nạ màu sắc
    draw = img.copy() # Sao chép ảnh màu tương ứng để vẽ lên
    for bbox in bboxes:
        x, y, w, h = bbox
        # Vẽ khôi hộp bao quanh biển báo
        cv2.rectangle(draw, (x,y), (x+w,y+h), (0,255,255), 4) # vẽ hình chữ nhật bao quanh vật
    results.append(draw)
show_images(results)

```



c. Phân loại biển báo với mạng nơ-ron tích chập

Sau khi lọc các biển báo theo màu sắc, việc tiếp theo chúng ta cần làm là phân loại chúng thành các biển báo khác nhau (biển dừng - stop, rẽ trái - left, rẽ phải - right, cấm rẽ trái - no_left, cấm rẽ phải - no_right, đi thẳng - straight). Các vật không phải biển báo cũng được nhận ra và phân loại thành đối tượng không xác định (unknown).

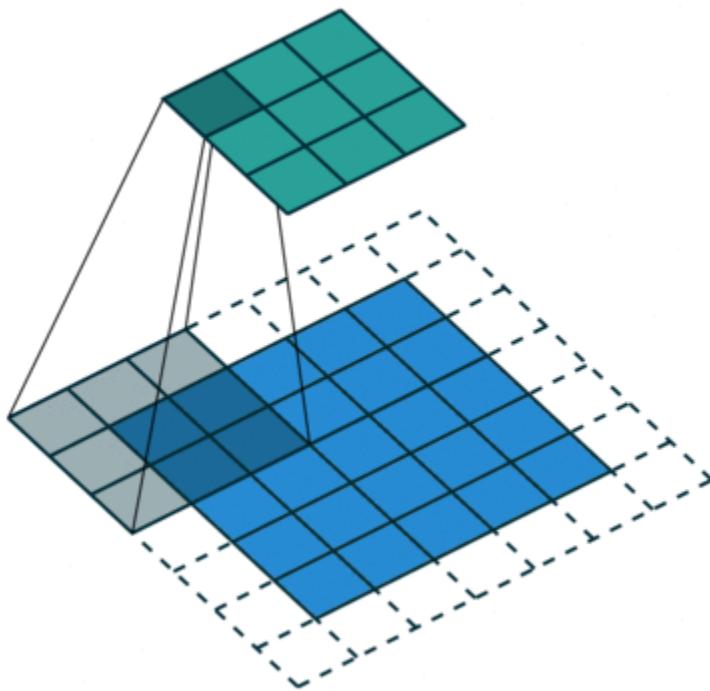
Tìm hiểu về mạng nơ-ron tích chập (Convolutional neural network - CNN)

Mạng nơ-ron tích chập (Convolutional neural network - CNN) là một trong những mô hình học máy tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh, thường dùng trong phân tích, nhận dạng hình ảnh với độ chính xác cao. Để tìm hiểu tại sao thuật toán này được sử dụng rộng rãi cho việc phân tích hình ảnh, chúng ta hãy cùng tìm hiểu về thuật toán này.

Ma trận lọc học từ dữ liệu

Trong bài học về các phép lọc ảnh, chúng ta đã được làm quen với các phép lọc sử dụng các bộ lọc tích chập. Chúng thực hiện nhân ma trận lọc với một cửa sổ trượt với toàn bộ ảnh. Các ma trận lọc khác nhau cho ta những ứng dụng khác nhau. Ví dụ với ma trận lọc trung bình, ma trận

Gaussian ta có các bộ lọc giúp làm mịn ảnh. Với ma trận lọc Sobel, ta có bộ lọc giúp tìm các đường biên trong ảnh. Bằng việc thay đổi các trọng số trong các bộ lọc này, chúng ta trích xuất được các thông tin khác nhau từ hình ảnh đầu vào. Với việc kết hợp mạng nơ-ron nhân tạo vào phép nhân tích chập, chúng ta tạo ra các bộ lọc có khả năng "học" các trọng số này từ một lượng lớn dữ liệu. Các đặc trưng ngữ nghĩa được trích xuất bởi các bộ lọc này vượt xa các đặc trưng trích xuất bởi các bộ lọc được thiết kế bằng tay trong nhiều trường hợp, từ đó mở ra ứng dụng mới trong các bài toán thị giác máy.



Minh họa việc nhân ma trận ảnh. Hình ảnh được lấy từ https://github.com/vdumoulin/conv_arithmetic

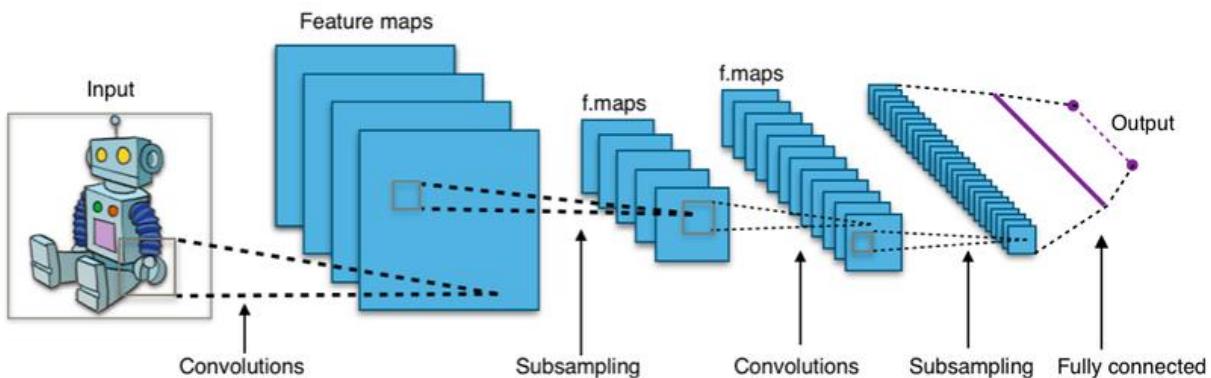
Cấu trúc của mạng nơ-ron tích chập

Một mạng là một tập hợp các lớp tích chập (convolution) chồng lên nhau, ở giữa có thêm các hàm kích hoạt phi tuyến như sigmoid, ReLU. Mỗi một lớp tích chập sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Thông tin khi đi vào các lớp về sau càng có tính tổng hợp cao, trích xuất được các ý nghĩa quan trọng của hình ảnh.

Trong quá trình huấn luyện một mạng nơ-ron tích chập, đưa hình ảnh đi qua toàn bộ mạng, sau đó cập nhật các trọng số (các tham số của các bộ lọc) dựa vào thuật toán lan truyền ngược - backpropagation. Các bạn có thể tham khảo về thuật toán này tại [Wikipedia](#). Thật may mắn, quá trình lan truyền ngược và cập nhật trọng số đều đã được cài đặt trong các thư viện học máy, học sâu phổ biến như Tensorflow hay PyTorch, từ đó giúp chúng ta huấn luyện được các mạng nơ-ron tích chập dễ dàng hơn.

Xây dựng mạng nơ-ron tích chập để phân loại biển báo

Chúng ta có thể ứng dụng các thiết kế của mạng nơ-ron tích chập để xây dựng một mô hình phân loại biển báo. Đầu vào của mạng phân loại là hình ảnh, đi qua nhiều lớp tích chập để trích xuất các đặc trưng phục vụ phân loại. Ta ghép thêm vào đó các lớp kết nối fully connected và cuối cùng là đưa ra kết quả dự đoán cho các biển báo, trả lời câu hỏi: đối tượng vừa đưa vào thuộc loại biển báo nào? Hình ảnh sau mô tả được kiến trúc mạng nơ-ron tích chập chúng ta sẽ xây dựng.



Thử nghiệm với mô hình huấn luyện sẵn

Chúng tôi đã thực hiện huấn luyện sẵn một mô hình phân loại biển báo bằng kiến trúc mạng nơ-ron tích chập **LeNet**, xây dựng bởi nhà khoa học **Yann LeCun** năm 1989. Chúng ta sẽ tải về và thử nghiệm mô hình này trước khi tự xây dựng và huấn luyện một mô hình.

```
# Nạp mô hình bằng OpenCV
model = cv2.dnn.readNetFromONNX("traffic_sign_classifier_lenet_v2.onnx")

# Hàm phát hiện biển báo
```

```

def detect_traffic_signs(img, model, draw=None):
    """Phát hiện biển báo
    """

    # Các lớp biển báo
    classes = ['unknown', 'left', 'no_left', 'right',
               'no_right', 'straight', 'stop']

    # Phát hiện biển báo theo màu sắc
    mask = filter_signs_by_color(img)
    bboxes = get_boxes_from_mask(mask)

    # Tiền xử lý
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    img = img.astype(np.float32)
    img = img / 255.0

    # Phân loại biển báo dùng CNN
    signs = []
    for bbox in bboxes:
        # Cắt vùng cần phân loại
        x, y, w, h = bbox
        sub_image = img[y:y+h, x:x+w]

        if sub_image.shape[0] < 20 or sub_image.shape[1] < 20:
            continue

        # Tiền xử lý
        sub_image = cv2.resize(sub_image, (32, 32))
        sub_image = np.expand_dims(sub_image, axis=0)

        # Sử dụng CNN để phân loại biển báo
        model.setInput(sub_image)
        preds = model.forward()
        preds = preds[0]
        cls = preds.argmax()
        score = preds[cls]

        # Loại bỏ các vật không phải biển báo - thuộc lớp unknown
        if cls == 0:
            continue

        # Loại bỏ các vật có độ tin cậy thấp
        if score < 0.9:

```

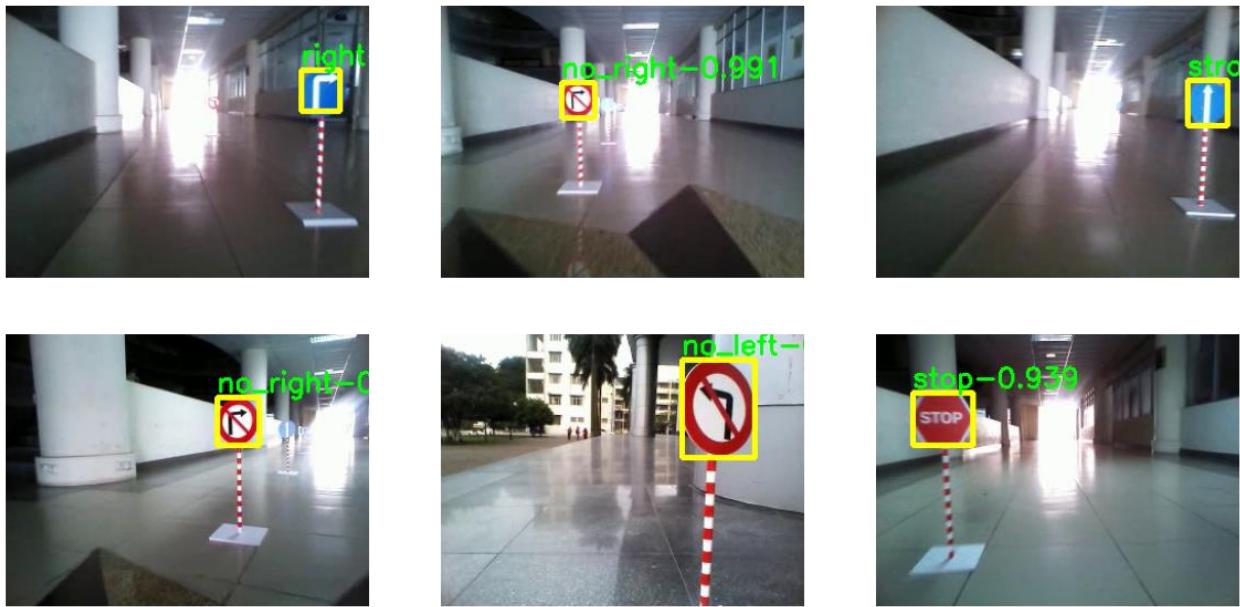
continue

```
    signs.append([classes[cls], x, y, w, h])

    # Vẽ các kết quả
    if draw is not None:
        text = classes[cls] + ' ' + str(round(score, 2))
        cv2.rectangle(draw, (x, y), (x+w, y+h), (0, 255, 255), 4)
        cv2.putText(draw, text, (x, y-5),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 0), 2)

    return signs

results = []
for i, img in enumerate(bgr_images):
    draw = img.copy()
    signs = detect_traffic_signs(img, model, draw=draw)
    results.append(draw)
show_images(results)
```



Huấn luyện mô hình LeNet cho phân loại biển báo

Các bước huấn luyện mô hình phân loại biển báo đã được trình bày chi tiết trong sổ tay Colab của bài học này. Các bạn có thể thử nghiệm thiết kế và huấn luyện các mô hình phân loại biển báo ngay trong môi trường Colab tại sổ tay dưới đây:

<https://colab.research.google.com/github/makerhanoi/via-course-ai/blob/master/notebooks/06-Huan-luyen-phan-loai-bien-bao.ipynb>

Triển khai hệ thống trí tuệ nhân tạo trên VIABot

Việc triển khai AI trên phần cứng Makerbot dựa vào repo mã nguồn sau:

<https://github.com/makerhanoi/via-course-makerbot>

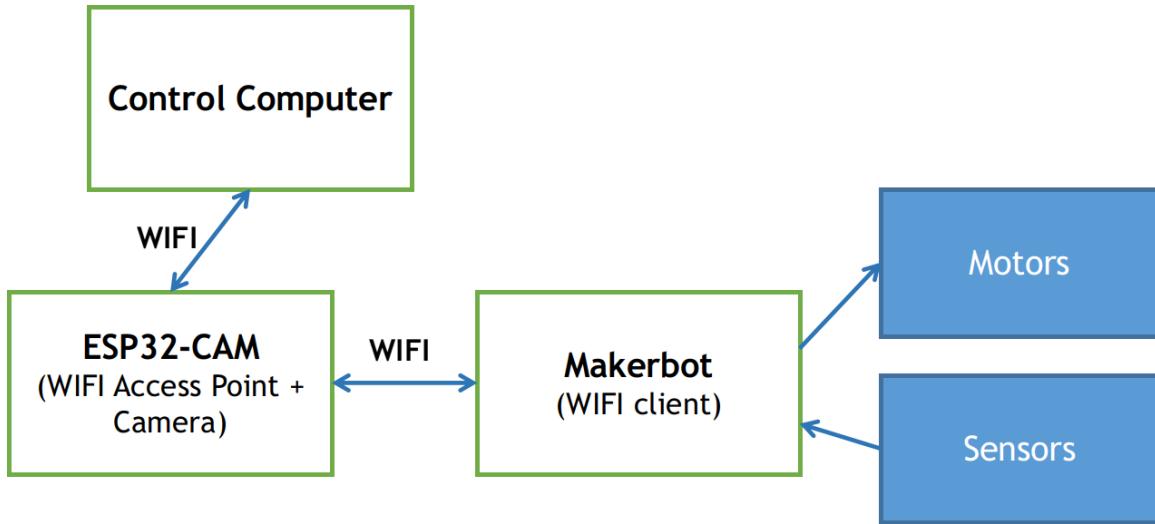
1. Nạp firmware và kiểm tra phần cứng

Thiết kế phần cứng

Để chạy được các ví dụ demo, các bạn cần có các linh kiện phần cứng sau:

- Mạch Makerbot để điều khiển động cơ, đọc cảm biến: <https://via.makerviet.org/vi/docs/hardware/design/>. Mạch này đã được BTC tặng cho mỗi đội tham gia.
- Module ESP32-CAM như một phần mở rộng của mạch Makerbot để đọc camera: <https://bit.ly/3fFHPOe>.
- Khung xe robot tròn hoặc một khung xe tương đương có lắp động cơ 5V: <https://bit.ly/3plS8u8>.
- Các linh kiện khác: Pin, sạc, dây nối.

Ở thiết kế cơ bản, chúng ta sẽ dùng ESP32-CAM để làm bộ phát WIFI. Máy tính điều khiển - Control Computer có thể được kết nối vào WIFI này, nhận hình ảnh từ camera. Thông qua WIFI của mạch ESP32-CAM, máy tính điều khiển cũng có thể kết nối tới mạch Makerbot để đọc các giá trị cảm biến và điều khiển động cơ. Mạch Makerbot sẽ hoạt động như một thiết bị client, bắt WIFI của ESP32-CAM, nhận lệnh từ máy tính truyền xuống và đồng thời truyền lại các giá trị cảm biến mà nó đọc được.



Thông tin kết nối WIFI của mạch ESP32-Cam

- SSID: VIA-MakerBot-01
- Mật khẩu: makerbotisfun

Các bạn có thể thay đổi thông tin này bằng cách sửa code và nạp lại firmware.

Nạp firmware

Trước tiên, chúng ta cần nạp firmware cho mạch Makerbot và module ESP32-CAM.

Nạp firmware với PlatformIO: Chúng tôi khuyến khích sử dụng PlatformIO được cài đặt trên trình chỉnh sửa code là Visual Studio Code. Sau đó, các bạn có thể nạp firmware bằng cách dùng PlatformIO mở các folder sau, biên dịch và nạp code lên các mạch. Hướng dẫn cài đặt và sử dụng PlatformIO để nạp firmware các bạn có thể truy cập tại [đây](#).

Các bạn cần cài đặt thêm Driver CH340 khi sử dụng với Windows theo hướng dẫn tại đây: <https://www.arduined.eu/ch340-windows-10-driver-download/>.

- Firmware cho mạch Makerbot: [firmware/makerbot fw](#).
- Firmware cho mạch ESP32-CAM: [firmware/esp32_cam fw](#).

Hiện tại code firmware chưa hỗ trợ Arduino IDE

2. Kiểm tra tín hiệu hình ảnh từ ESP32-CAM

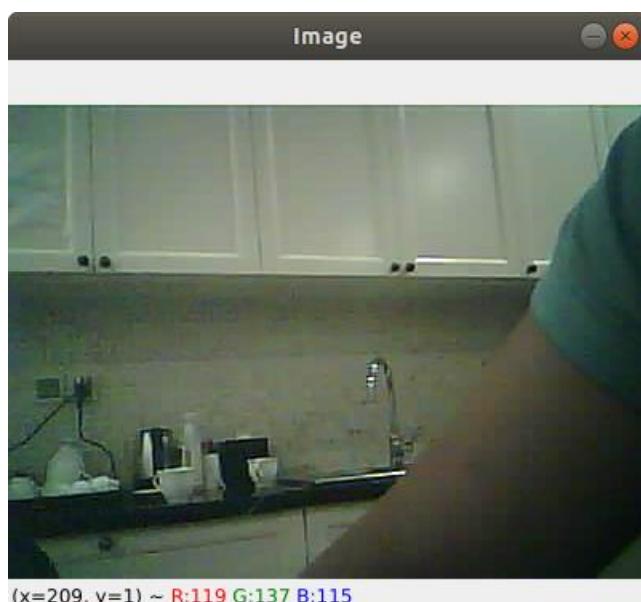
Kết nối vào WIFI VIA-MakerBot-01 và mở trình duyệt web, truy cập: <http://192.168.4.1> để xem hình ảnh thu được từ camera.

a. Đọc camera từ ESP32-CAM bằng Python:

Để đọc camera từ Python, máy tính của các bạn cần được cài đặt Python và package OpenCV. Trước tiên hãy cài đặt Python với trình quản lý gói Pip, sau đó dùng Pip để cài OpenCV: `pip install opencv-python`. BTC khuyến khích các bạn tìm hiểu về Anaconda / Miniconda để quản lý môi trường cho Python. Cách cài đặt Anaconda có thể được tìm thấy tại [đây](#).

Chạy code đọc camera từ ESP32: Các bạn kết nối vào WIFI của mạch ESP32-CAM, sau đó chạy chương trình tại [examples/read_esp32_cam/read_cam.py](#).

```
python read_cam.py
```



Hình ảnh thu được từ camera ESP32-CAM

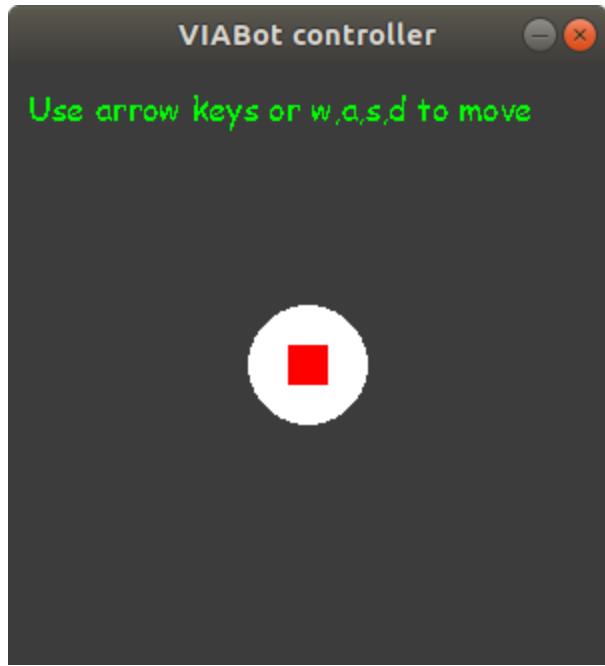
b. Ví dụ về điều khiển bằng bàn phím

Cài đặt môi trường Python từ
tệp `examples/keyboard_control/requirements.txt` bằng cách gõ

```
pip install -r requirements.txt.
```

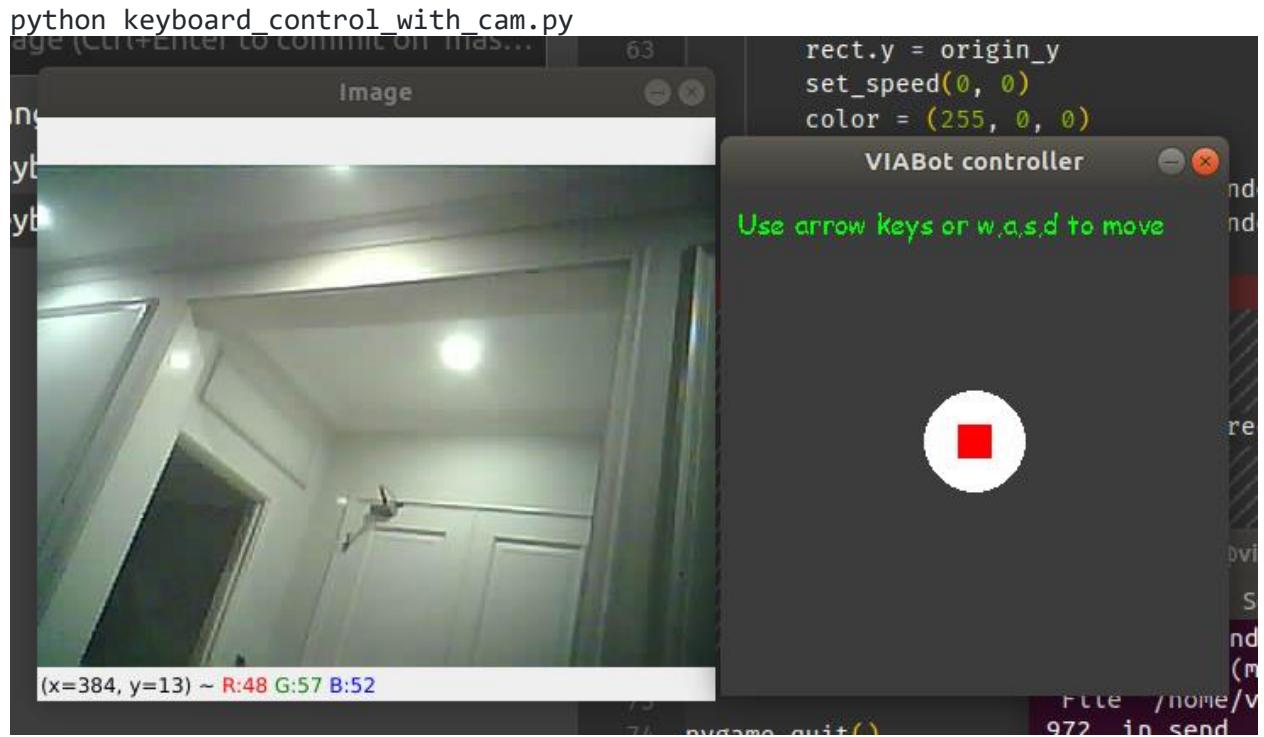
Chạy ví dụ điều khiển bằng bàn phím cho VIABot:

```
python keyboard_control.py
```



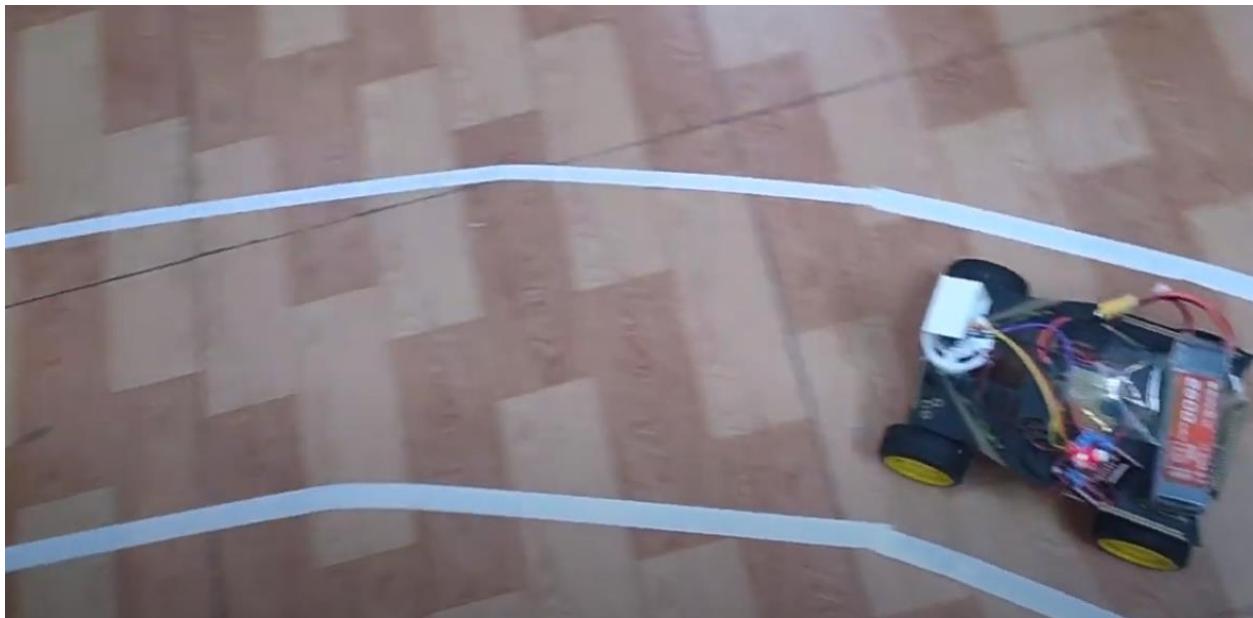
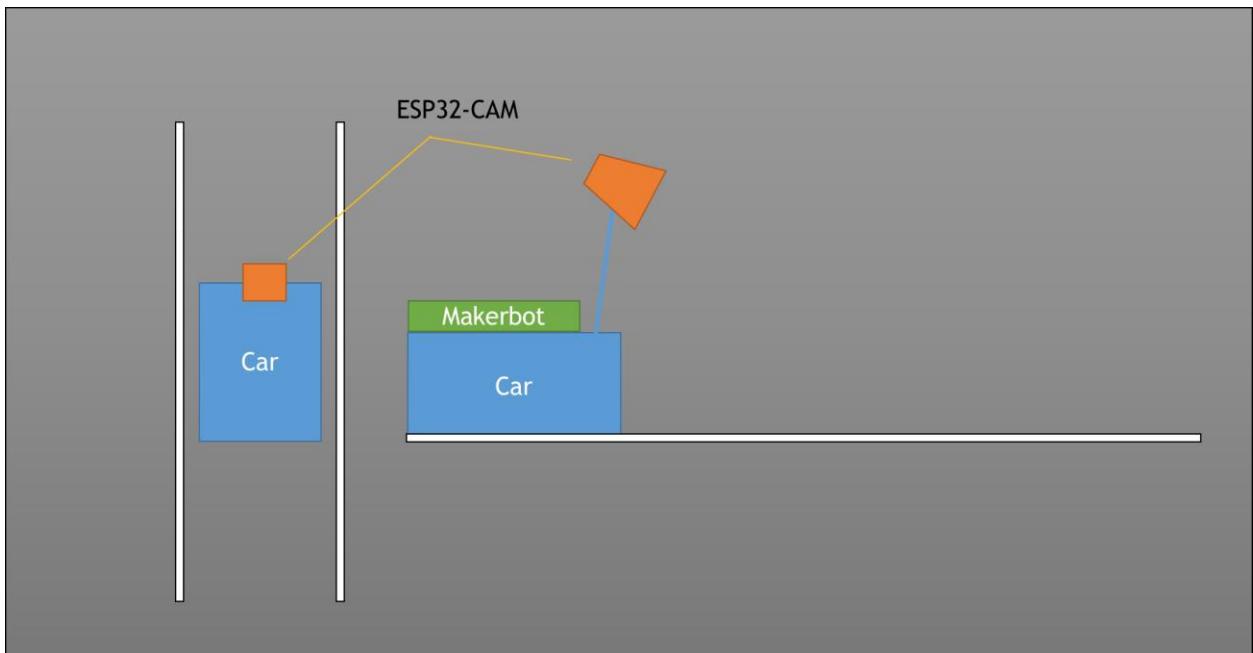
Một cửa sổ điều khiển hiện lên như trên. Dùng các phím mũi tên để thử điều khiển các motor của VIABot.

Điều khiển kết hợp xem camera từ mạch ESP32-Cam:



c. Ví dụ về điều khiển xe tự lái đi theo vạch kẻ đường

Dán vạch kẻ đường cho VIABot như hình:



Mở terminal trong folder examples/auto_drive và chạy ví dụ:
python drive.py

Các cửa sổ thể hiện sự phân tích hình ảnh sẽ hiện lên khi kết nối thành công với ESP32-CAM.



Chỉnh góc xoay của camera để nhìn thấy vạch kẻ đường. Các bạn nên để camera ở trên cao khoảng 15-20 cm, chúc đầu xuống một chút để đạt góc nhìn tốt nhất. Tiếp đó chỉnh các tham số và thuật toán tính toán trong hàm calculate_control_signal() của tệp [examples/auto_drive/controller.py](#).

Phần 4: Mô phỏng trong VIA

Cuộc cách mạng công nghiệp lần thứ 4 gắn liền với trí tuệ nhân tạo, IoT và robot. Cuộc cách mạng trước là tiền đề rất lớn để phát triển cuộc cách mạng 4, thông tin và dữ liệu là 2 tài sản được tích lũy từ quá trình số hóa, ứng dụng hóa các dịch vụ trong cuộc sống. Thời kỳ này khi các siêu máy tính bùng nổ giải phóng giới hạn cho các thuật toán có thể dễ dàng xây dựng mô hình một cách nhanh chóng. Các máy tính cấu hình tốt chi phí hợp lý giúp bùng nổ các thiết bị thông minh giúp lan tỏa các thiết bị để tiếp tục thu thập thông tin giúp máy tính càng ngày càng hiểu con người hơn. Robot đã được giải phóng "đôi mắt, cái tai" của mình, các thuật toán A.I đã giúp có thể nhìn và hiểu thế giới, có thể nghe và nói chuyện tương tác với con người từ đó ra những quyết định phù hợp. Đây là những yếu tố cuối cùng để robot ra được cuộc sống.

Tôi đã trải nghiệm rất nhiều các dự án trong thời kỳ này trong 8 năm

1. A.I - Trí tuệ nhân tạo

Trí tuệ nhân tạo là một lĩnh vực đang thu hút nhiều sự quan tâm của cộng đồng nghiên cứu trên thế giới. Nhiều ứng dụng trong đó sử dụng trí tuệ nhân tạo đã và đang được phát triển mạnh mẽ thời gian gần đây. Vậy trí tuệ nhân tạo (AI) là gì, dưới góc nhìn đơn giản, AI có thể hiểu là *chương trình giúp máy tính trở nên thông minh hơn*. Do đó các nghiên cứu trong lĩnh vực AI tập trung phát triển các phương pháp hướng đến mục đích làm cho máy móc trở nên hữu ích hơn và có trí tuệ để tự đưa ra quyết định trong một số tình huống.

Đặc điểm chính của AI là "*tìm kiếm heuristic*". Thực tế là "nếu bạn không thể nói cho máy tính biết cách tốt nhất để làm một cái gì đó, thì hãy lập trình để nó thử nhiều cách tiếp cận." (Baraiko (1982, tr. 450). Tuy nhiên, số lượng các cách tiếp cận (giải pháp) thường rất lớn. Do đó, thay vì duyệt qua tất cả các cách tiếp cận thì lời giải thường được tìm kiếm một cách "*heuristic*": tìm kiếm dựa trên một số quy tắc xây dựng từ kinh nghiệm thực tế. Một khái niệm khác của AI là "miền tri thức". Trí tuệ phụ thuộc vào kiến thức và kiến thức cần tích luỹ và được sử dụng khi cần thiết trong quá trình tìm kiếm. Các chương trình AI thường tách biệt tri thức với cơ chế điều khiển quá trình tìm kiếm.

a. Các thành phần cơ bản của trí tuệ nhân tạo

Tìm kiếm Heuristic

Phần lớn công việc thời kì đầu của AI là tập trung tạo ra chương trình tìm kiếm giải pháp cho bài toán. Mỗi khi đưa ra quyết định, bất kì sự thay đổi nào về tình huống sẽ lại mở ra những cơ hội mới cho các quyết định tiếp theo. Do đó tồn tại khái niệm về điểm rẽ nhánh và để giải quyết vấn đề đó trong AI, khái niệm về cây được sử dụng: bắt đầu từ điều kiện khởi tạo, và phân nhánh mỗi khi đưa ra quyết định. Khi thực hiện duyệt cây đi từ trên xuống, nhiều quyết định khác nhau được tạo ra, do đó số lượng nhánh cây có thể trở nên rất lớn khi vấn đề càng phức tạp. Do đó, cần có phương pháp hiệu quả để tìm kiếm trên cây.

Ban đầu, các phương pháp tìm kiếm trên cây thường là các phương pháp "mù" và việc tìm kiếm/duyệt chỉ cần đảm bảo một giải pháp không được thử nhiều lần. Tuy nhiên, đối với các vấn đề phức tạp hơn thì cách tiếp cận duyệt đơn giản không đảm bảo hiệu quả và phương pháp tìm kiếm "heuristic" ra đời nhằm hỗ trợ/định hướng quá trình lựa chọn các nhánh có khả năng nhất. Một ví dụ đơn giản về tìm kiếm "heuristic" là xác định hướng đi giúp người lái xe đi từ A đến B vào buổi tối là "đi theo hướng mặt trời lặn". Điều này có thể không tạo ra con đường tối ưu nhất từ A đến B, nhưng nó sẽ giúp người lái xe hướng tới mục tiêu nhanh hơn.

Biểu diễn tri thức

Thực tế kho tri thức con người đã xây dựng nên rất lớn. Do đó để có tri thức giúp giải quyết một vấn đề cụ thể, trước hết AI cần quan tâm đến việc mô hình hóa lượng kiến thức khổng lồ này và có phương pháp cho phép quá trình truy vấn tri thức mã hóa được dễ dàng và nhanh chóng. Biểu diễn tri thức có thể được coi là một trong những lĩnh vực nghiên cứu sôi động nhất trong AI.

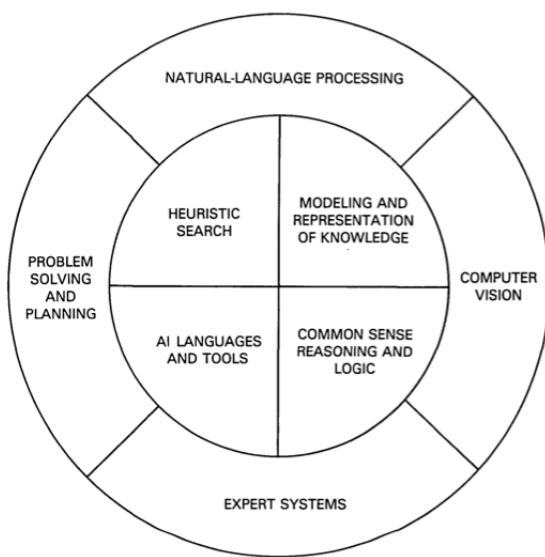
Cảm tính và logic

Cảm tính là một đặc tính của con người, và việc tìm ra phương pháp mô hình hóa "cảm tính" của con người để sử dụng được trong AI có thể xem điều khó khăn nhất và chắc chắn sẽ không thể có lời giải trong tương lai gần. Một lĩnh vực khác cũng rất quan trọng trong AI đó là logic: suy luận ra một tri thức mới dựa trên một loạt sự kiện/sự thật; đưa ra chứng minh cho một bài toán

dựa trên tập hợp các giả thiết v.v. Logic trong tính toán được sử dụng nhiều trong AI nhằm hỗ trợ tìm kiếm, giải quyết vấn đề phức tạp.

Các lĩnh vực ứng dụng chính của AI

Dựa trên các thành phần cơ bản của AI, Nilsson đã xác định một số lĩnh vực chính của AI bao gồm 1) xử lý ngôn ngữ tự nhiên, 2) Nhìn máy 3) Hệ chuyên gia 4) Giải quyết vấn đề và lập kế hoạch



Chia sẻ: Trí tuệ nhân tạo là những phần mềm do con người xây dựng dựa trên những "kinh nghiệm" của chúng ta khi tương tác với môi trường thực tế. Những bộ máy AI hiện nay chưa có khả năng phán đoán nhưng tình huống mang tính chất "cảm tính", phán đoán từ nhiều luồng thông tin. AI ngày nay thường tách thành các chủ đề cụ thể, độc lập để xây dựng thuật toán, mô hình và thu thập các dữ liệu phù hợp để giải quyết một vài bài toán cụ thể. Hiện nay chưa có đơn vị nào có thể phát hành một ứng dụng AI có khả năng tự học mà không cần sự huấn luyện của con người trong mọi lĩnh vực như các bộ phim khoa học viễn tưởng nói về Robot.

Tôi đã tham gia các dự án nghiên cứu A.I từ khoảng 8 năm trước, đó cũng là thời điểm các doanh nghiệp tại Việt Nam bắt đầu đầu tư vào lĩnh vực này để tìm kiếm hướng ứng dụng vào sản phẩm. Các bài toán A.I đang được giải rất cơ bản như :

Tổng hợp tiếng nói – TTS -Text to speech/ Speech Synthesizere: từ những dữ liệu thu âm của người bình thường thông qua hệ thống A.I có thể tạo ra một câu nói bất kỳ của người đó.

Thử làm: Tự tạo ra một đoạn file audio từ công cụ AI tiếng Việt hoặc tiếng Anh, sâu hơn có thể tự làm một bộ tổng hợp giọng nói của mình qua các công cụ mã nguồn mở cho tiếng Anh (tự thu, tự huấn luyện). Trang web khá phổ biến để các bạn vào thử <https://fpt.ai/tts>, <https://translate.google.com/> (có nút loa khi các bạn dùng)

Nhận dạng tiếng nói – ASR - Automatic Speech Recognition: cần phải thu thập rất nhiều giọng của nhiều người, xây dựng mô hình từ đó A.I sẽ nhận dạng được người nói và chuyển thể ra văn bản. Bài toán nhận dạng giọng nói có nhiều cấp độ, nhiều doanh nghiệp hiện nay đã dùng công nghệ này để xác thực tài khoản (như Google Home), định danh cá nhân trên các hệ thống phần mềm. A.I có thể biết bạn là ai, bạn là nam hay nữ, bạn nói tiếng địa phương nào, bạn đứng xa hay đứng gần, bạn có nói dối không,... tất cả chỉ qua âm thanh được thu thập.

Thử làm: Dùng ứng dụng Siri, Google Voice Search, Google Home các bạn sẽ thấy khả năng nhận dạng rất nhiều ngôn ngữ. Trên các dòng tivi Android, tính năng tìm kiếm tiếng Việt đã trở nên rất thông dụng

Nhận dạng hình ảnh (Computer Vision) bài toán này sẽ lại tiếp tục được chia nhỏ ra thành rất nhiều bài toán con nhỏ hơn như nhận dạng khuôn mặt, nhận dạng biển báo giao thông, nhận dạng oto, xe máy, nhận dạng biển xe máy, oto, nhận dạng thông tin từ Chứng minh thư, hộ chiếu,... Mỗi bài toán lại là một thách thức khác nhau, các đặc tính khác nhau, trong các điều kiện môi trường khác nhau.

Thử làm: tìm một số điện thoại, máy tính đori mới đều có tính năng FaceID, thử học và nhận dạng bạn với người khác để mở khóa máy tính hoặc điện thoại

Xử lý ngôn ngữ tự nhiên (NLP/NLU natural language processing /natural-language understanding) là những công nghệ AI để huấn luyện các văn bản và biểu diễn chúng thành các thông tin có ý nghĩa với máy, phần mềm khác. Ví dụ: Hôm nay thời tiết Hà Nội như thế nào ? Hệ thống A.I sẽ phải hiểu đây là câu hỏi, ý định (intent) là hỏi về thời tiết, đối tượng/thực thể (entity là Hà Nội – là địa danh). Từ đó biểu diễn sang một ngôn ngữ mà các hệ thống phần mềm khác có thể hiểu được. Nay NLP đã được ứng dụng trong rất nhiều thực tế: như các hệ thống chatbot, loa thông minh -Alexa, Google Home,....

Thử làm: Tự làm chat-bot tiếng Việt trả lời tự động trên page của các bạn trên nền tảng <https://console.fpt.ai/>

Ngoài ra còn rất nhiều các bài toán thực tiễn khác đã được ứng dụng các công nghệ về giọng nói, chữ viết, ngôn ngữ, hình ảnh, dữ liệu đang được huấn luyện để trở thành A.I. như những ví dụ trên của tôi.

Vậy con người xây dựng A.I như thế nào ? Bước đầu tiên bạn xác định bài toán cụ thể A.I cần giải quyết. VD tôi chọn bài toán A.I phát hiện tên người, biết đây là tên người và tên Nam hay nữ. Nếu bài toán là phát hiện tên thành phố, nếu không dùng AI bạn có thể tìm 64 tỉnh thành và so sánh với dữ liệu này, đó chưa gọi là A.I. hoặc coi là hình thái sơ khai của A.I – lập trình theo rule-base – theo luật biết trước. Bài toán này tôi sẽ phải thu thập một tập tên người đủ lớn, chọn đặc tính nhận dạng, cho vào các bộ huấn luyện dùng các thuật toán từ đó ra một mô hình xác nhận. Sau khi có mô hình, mỗi lần sử dụng tôi chỉ việc gọi để xác nhận tên Nguyễn Văn A có phải là tên người, xác suất là 95%, hay Đồng Văn Quất là tên người, xác xuất là 65% do tên, họ ít phổ biến. Đơn giản thuật toán trong bài toán này sẽ là thống kê theo cặp các từ cạnh nhau từ những dữ liệu đã được huấn luyện là tên người để ra mô hình ứng dụng.

Các mô hình, thuật toán trong A.I đều được xây dựng từ những môn toán mà chúng ta đang học trên ghế nhà trường như xác suất, giải tích, ma trận, vector, đạo hàm, vi phân, tích phân. Đó chính là ứng dụng của toán học trong thời kỳ này và đây cũng là môn học được quan tâm nhất ở Việt Nam. Trong một thập kỷ tới, hầu hết các ứng dụng, công việc có tính chất lập lại đều có thể dùng A.I để giải quyết như kế toán, văn phòng, tổng đài viên, nhân viên bán vé, phiên dịch viên,

...Đây là cuộc cách mạng mà con người sẽ chuyển dịch sang nghề mới, các nghề nhảm chán sẽ được máy móc và A.I

1. IoT – Internet of things

IoT: Internet of Things là thời kỳ mà các thiết bị xung quanh chúng ta đều trở nên thông minh và được kết nối tới những dịch vụ trên mạng - Gọi một cách đơn giản là vạn vật kết nối. Những thiết bị có sử dụng IoT thường được lập trình và có khả năng tự động làm việc, giúp cho cuộc sống của chúng ta thông minh hơn, tiết kiệm thời gian và chi phí trong cuộc sống hàng ngày. IoT là sự phát triển của A.I từ các ứng dụng trên máy tính, điện thoại tới các thiết bị khác xung quanh cuộc sống của con người.

Các thiết bị IoT thu thập các dữ liệu dạng hình ảnh, âm thanh, dữ liệu cảm biến để đưa ra cảnh báo hoặc quyết định. Ví dụ đơn giản nhất: một chiếc camera ip bình thường với kết nối Internet đã có 10-20 năm chưa được coi là thiết bị IoT, nhưng các thiết bị camera cloud mới, có dịch vụ nhận dạng người lạ, tiếng khóc, vật thể di chuyển được coi là thiết bị IoT.

Nếu như chiếc camera bình thường luôn cần một người bảo vệ giám sát màn hình 24/24 chống trộm thì với loại camera cloud mới được sử dụng IoT thường tích hợp dịch vụ tự động phát hiện bất thường, người lạ giúp chúng ta không cần phải giám sát 24/24.

Tại sao các doanh nghiệp đang đầu tư vào IoT ?

Mỗi cuộc cách mạng công nghiệp đều là cuộc cách mạng về sức lao động, tăng hiệu suất làm việc của các doanh nghiệp và xã hội. Số lượng người làm những công việc tay chân và những công việc có tính lặp lại giảm đi đáng kể theo những cuộc cách mạng này.

Thời kỳ công nghiệp 4.0 của chúng ta là thời kỳ của trí tuệ nhân tạo và thiết bị thông minh, trong mọi ngành nghề đều có thể ứng dụng để giảm sức lao động của con người, tăng chất lượng và năng suất doanh nghiệp.

Một ví dụ đơn giản về tác dụng của IoT, Amazon thay thế toàn bộ nhân viên trong kho hàng thương mại điện tử bằng những chiếc kệ biết di chuyển giúp tăng thời gian giao hàng. Nhiều doanh nghiệp café đã thay nhân viên pha chế bằng những máy pha tự động.

Những chiếc máy này có kết nối mạng và chúng có thể nhớ luôn khẩu vị của từng khách hàng và hoàn toàn tự động. Những công nghệ này giúp các doanh nghiệp mở rộng kinh doanh dễ dàng hơn, đặc biệt nâng cao chất lượng và giảm chi phí lao động. Bởi vậy IoT đang là ngành mũi nhọn được nhiều doanh nghiệp trong nhiều lĩnh vực đầu tư vào cho hoạt động sản xuất.

Ứng dụng IoT trong các ngành nghề khác nhau ?

Ứng dụng IoT khá rộng, ngoài các ứng dụng nhiều người thấy như nhà thông minh, thành phố thông minh thì IoT được ứng dụng tại rất nhiều ngành khác nhau như nông nghiệp, thể thao, y tế...

Có thể bạn chưa biết nhưng tại các trang trại tiên tiến, vật nuôi được gắn cảm biến theo dõi sức khỏe cân nặng, nhiệt độ... và sử dụng hệ thống cho ăn tự động phù hợp.

Trong thể thao, người ta gắn cảm biến để theo dõi vào giày, áo, bóng của vận động viên để phân tích các thông số. Đối với y tế, nhiều bệnh mãn tính đã được hỗ trợ các thiết bị IoT để kết nối thông số của bệnh nhân tới bác sĩ, các dịch vụ tự động phát hiện tình trạng nguy hiểm của bệnh nhân. Thậm chí nhiều nhà bếp cũng thay các cánh tay Robot được chế biến món ăn theo các công thức trên đám mây cho chuỗi cửa hàng.

Thử làm: Tự làm sản phẩm IoT trong 30 phút với các nền tảng mã nguồn mở sử dụng Esp8266 – Arduino, phần mềm di động Blynk. Làm ứng dụng đo nhiệt độ, độ ẩm trong phòng của bạn.

Chia sẻ: Cá nhân tôi phụ trách đội sản phẩm làm các giải pháp IoT từ năm 2014. Đó là thời mà các công ty nói đến IoT và đầu tư rất nhiều vào startup IoT như Google vào Nest hơn 1 tỷ USD, hay Samsung đầu tư vào Smartthing 200 triệu USD. Trong giai đoạn này từ năm 2016 bùng lên một xu hướng mới đó là loa thông minh, kết hợp AI và IoT tạo thành những sản phẩm kiểu mới. Amazon là ông lớn đi tiên phong trong lĩnh vực này sau khi rất thành công với dịch vụ cloud của mình tại thời kỳ trước. Từ 2016 đến 2019 là sự bùng nổ của các thiết bị giao tiếp giọng nói kết hợp với các thiết bị IoT trong gia đình như Alexa – Amazon, Google Home -Google, Home Pod – Apple đang trở thành trung tâm của gia đình. Trong thời gian ngắn nữa, tôi tin các ứng dụng giao tiếp giọng nói sẽ trở nên phổ biến trong từng gia đình, từng văn phòng. Thay vì phát triển các

ứng dụng trên mobile, thì các lập trình viên sẽ phát triển các ứng dụng giọng nói (VUI). Mọi thiết bị IoT xung quanh chúng ta có thể nói chuyện được không chỉ có loa. Giai đoạn 2021 -2024 sẽ là thời gian bùng nổ các thiết bị IoT trong các doanh nghiệp như các thiết bị chấm công nhận dạng khuôn mặt, các hệ thống thiết bị IoT hỗ trợ logistic như xe không người lái trong nhà máy, xe giao hàng trong chung cư, dịch vụ robot hút bụi công nghiệp thay người, IoT hóa trong các chuỗi cung ứng thực phẩm để kiểm soát chất lượng. IoT là một xu hướng sau thời kỳ A.I, nhờ có AI giúp các thiết bị trở nên thông minh và ngày càng phổ biến. IoT sẽ là cách tay nối dài của các dịch vụ A.I và Cloud, nó sẽ bùng nổ rất nhanh bởi chi phí chip càng ngày càng rẻ và tốc độ phát triển nhanh các sản phẩm mới.

2. Robot thời 4.0

Robot trong thời kỳ 4.0 đang có những tiến bộ vượt bậc so với các thời kỳ trước khi được nâng cấp thêm thuật toán và cấu hình máy tính. Robot không chỉ làm những nhiệm vụ cố định và lặp lại nữa, robot đã bắt đầu có thể linh hoạt hơn làm trong những điều kiện phức tạp và có khả năng tự ra quyết định trong một môi trường nhất định.

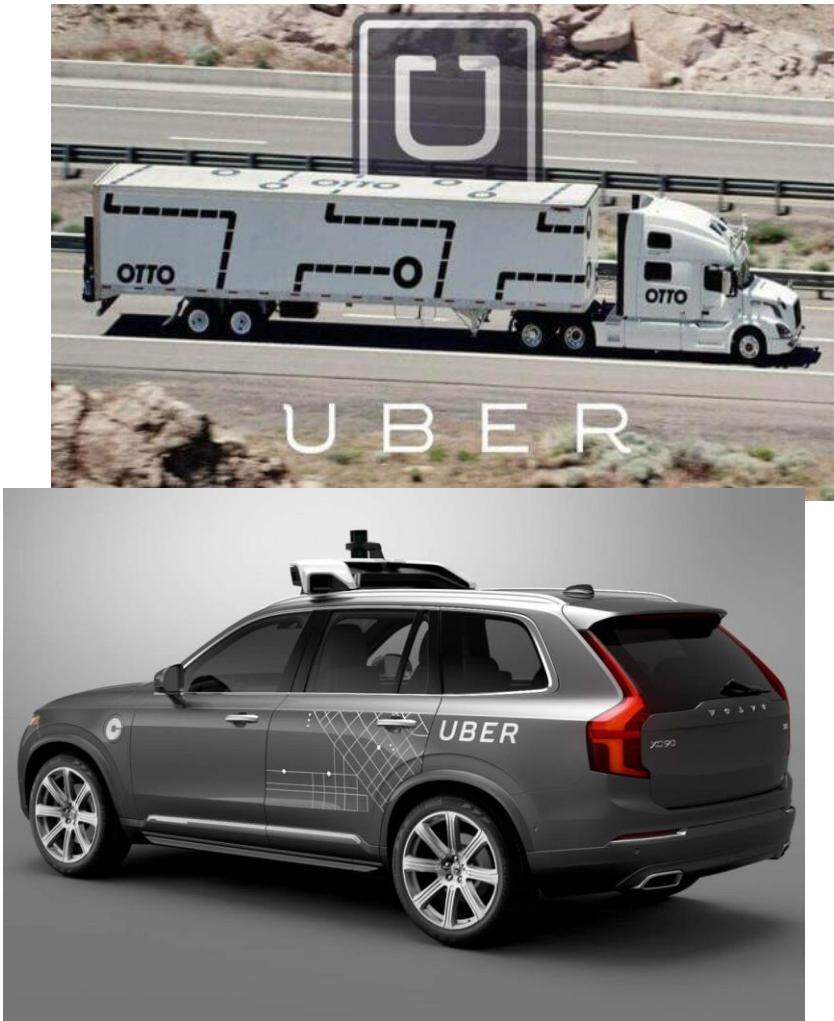
Xe không người lái trên đường, máy in 3D, cánh tay máy thông minh, robot tự hành trong nhà máy, máy bay không người lái phun thuốc trừ sâu ... Trong khoảng 5-10 năm gần đây các bạn sẽ thấy liên tục các bài viết nói về những xu hướng, sản phẩm dịch vụ mới này ở các nước phát triển. Vậy robot đã ứng dụng AI vào những sản phẩm đó như thế nào.

Đầu tiên phải kể đến đó là Amazon, công ty này đã mua lại công ty Kiva Systems năm 2012 với giá gần 1 tỷ USD, ngay sau đó 2014 Amazon đã thay thế 10.000 robot tự lấy hàng trong kho để nâng cao chất lượng dịch vụ của mình. Dịch vụ giao hàng nhanh trong 2h và 1 ngày của Amazon được hỗ trợ rất nhiều bởi những chú robot này.



Tesla là một ông lớn tiếp theo trong ngành oto, đây là hãng đầu tiên đưa ra dòng xe thương mại có tính năng chạy tự động trên cao tốc, thử nghiệm trên đường phố đô thị. Đây là hãng đầu tiên nói rất nhiều về xe tự hành và tạo lên một làn sóng ứng dụng xe tự hành cho các phương tiện giao thông trong tương lai.

Uber là một hãng xe nổi tiếng, một startup kỳ lân thời 4.0 cũng đầu tư rất mạnh trong mảng robotic này. Ngay năm 2016 công ty cũng bỏ ra hơn 600 triệu USD để mua OTTO một công ty phát triển về robot tự hành trong nhà máy. Uber đã biến các dịch vụ sản phẩm của OTTO ra những phương tiện giao thông như xe tải, xe taxi, xe giao hàng nhỏ.... Tương lai gần các bạn có thể tưởng tượng thế giới xe có hệ thống logistic không người lái, hệ thống taxi, xe buýt tự động hoàn toàn.



Robot trong thời kỳ này rất khác với các thời kỳ trước, mã nguồn mở rất phát triển, giúp các startup phát triển rất nhanh các dự án robot của mình. Các nền tảng mã nguồn mở ROS trở nên bùng nổ và phát triển mạnh trong những năm gần đây.

Supported robots



A lot more on <http://www.ros.org/wiki/Robots>

17

Thử làm: Tự làm cho mình một chiếc xe tự hành từ máy tính Pi hoặc máy tính jetson dựa vào mã nguồn mở <https://github.com/fpt-corp/DiRa>. Đây là các dự án khó, cần các bạn nắm hết các kiến thức cơ bản mới có thể bắt đầu làm dự án này.

Chia sẻ: Tôi ngoài tham gia làm các sản phẩm IoT cũng xây dựng các cuộc thi về Robot cho các bạn trẻ. Mục tiêu giúp các bạn học nhanh kiến thức mới thông qua thực hành tại các cuộc thi. Với tôi các cuộc thi sẽ giúp các bạn trưởng thành hơn kiến thức, khả năng logic và hệ thống hóa lại những thứ mình được học. Đặc biệt qua đó sẽ giúp các bạn kỹ năng giải quyết vấn đề, kỹ năng làm việc nhóm – đó là những thứ không thể thiếu trong thế kỷ 21 cho các bạn.

Cuộc thi cho cấp 1-2: mini First tôi đã tổ chức được 3 mùa cho các bạn nhỏ cấp 1 và cấp 2 tại Hà Nội. Năm ngoái Maker Hanoi và Risupia đã cùng phối hợp tổ chức, hai mùa đầu tiên chúng tôi phối hợp cùng đơn vị NIIT. Cuộc thi không chỉ là sân chơi, mà là nơi giúp bạn nhỏ khám phá các sở thích của mình. Từ đó giúp các bạn cấp 1-cấp 2 có thể định hình hơn về nghề nghiệp trong

tương lai. Cuộc thi này tôi dự kiến vẫn sẽ tổ chức hàng năm lấy chủ đề của cuộc thi FIRST Global toàn cầu để cho các bạn nhỏ cùng giải quyết.

Cuộc thi cho cấp 3: Vietnam STEAM Challenge - VSC là một cuộc thi mới được phát động mùa đầu tiên vào năm 2020. Cuộc thi đã có 35 đội tham gia có thành viên từ Hà Giang tới đất mũi Cà Mau. Tôi mong qua cuộc thi sẽ giúp các em định hướng tốt nghề nghiệp trong tương lai. Giúp các em chọn lựa ngành học của mình trước ngưỡng cửa đại học. Đặc biệt tôi mong muốn các em sẽ là các đại sứ để mang các kiến thức được học, được tìm hiểu thông qua cuộc thi chia sẻ tới các bạn nhỏ trong vùng. Tôi mong đây sẽ là những người bạn, người đồng nghiệp của tôi trong tương lai để cùng hành trình "Xóa mù công nghệ" cho các bạn trẻ trong cả nước. Giúp các bạn trẻ tận dụng được kiến thức về công nghệ và khoa học để ứng dụng trong cuộc sống và các bài toán lao động sản xuất tại địa phương. <https://www.facebook.com/VietnamSTEAMChallenge/>

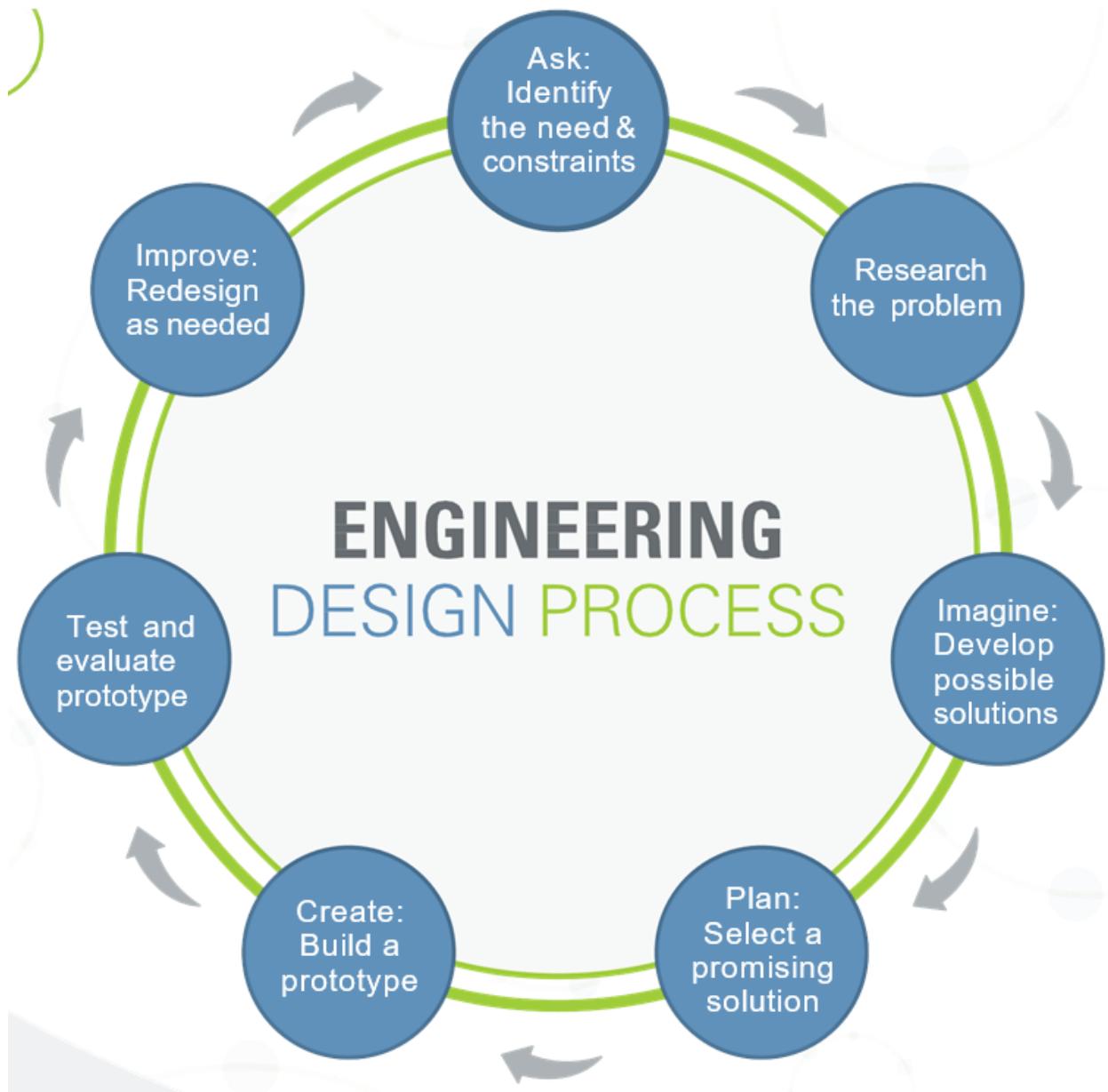
Cuộc thi cho đại học: Digital Race là một cuộc thi cho các bạn sinh viên cả nước về các công nghệ mới nhất trên thế giới. Mong muốn chúng tôi muốn trao cơ hội sớm nhất cho các sinh viên tài năng tại các trường đại học. Giúp các bạn gắn kiến thức vào thực tiễn. Cuộc thi đã tổ chức được 4 năm với chủ đề xe không người lái. Đây là cuộc thi tổng hợp được các kiến thức của thời kỳ 4.0 là A.I và robot.



Phần 5: Thách thức VIA và làm sản phẩm

Một chú Robot gồm 3 phần là điện tử, cơ khí và lập trình: Điện tử giống như mạch máu trong cơ thể giúp kết nối các bộ phận, tín hiệu của robot tới bộ não của robot, trái tim của Robot chính là phần nguồn, tạo năng lượng giúp robot hoạt động. Cơ khí là các cơ cấu chấp hành giúp robot di chuyển, thực hiện các chuyển động theo thiết kế; cơ khí là các bộ phận giống như cơ bắp của con người, người làm cơ khí cần kỹ năng không gian, liên kết động tốt. Lập trình được ví như bộ não của Robot, là mảng công việc cần tư duy logic tốt kết hợp các tín hiệu điện từ cảm biến (sensor) để điều khiển các kết cấu cơ khí chuyển động nhịp nhàng theo nhiệm vụ đề ra.

Quy trình chế tạo một chú robot gồm 3 bước chính: thiết kế ý tưởng cơ khí và thực hiện chế tạo, đi dây các thiết bị điện tử như cảm biến, lập trình cho robot. Tuy nhiên các bước này sẽ được lắp đi lắp lại theo từng tính năng của Robot và yêu cầu của sản phẩm. Người làm thường có các bản thiết kế nháp cả robot hoặc từng tính năng. Các bạn sẽ chế tạo cơ khí và điện tử cho chức năng chuyên biệt đó và lập trình thử nghiệm riêng lẻ. Khi các chức năng cơ bản hoàn thành thì sẽ được ghép vào một sản phẩm hoàn chỉnh để thử nghiệm, xác định các vấn đề của giải pháp hiện tại và tiếp tục cải tiến. Khó khăn nhất với các bạn trẻ mới bắt đầu là lĩnh vực điện tử công suất và ngôn ngữ lập trình. Trong các cuộc thi trên thế giới hiện nay, hai giai đoạn này thường được đơn giản hóa bằng các block, KIT để đóng gói một số công nghệ giúp các bạn trẻ rút ngắn thời gian khi làm robot.

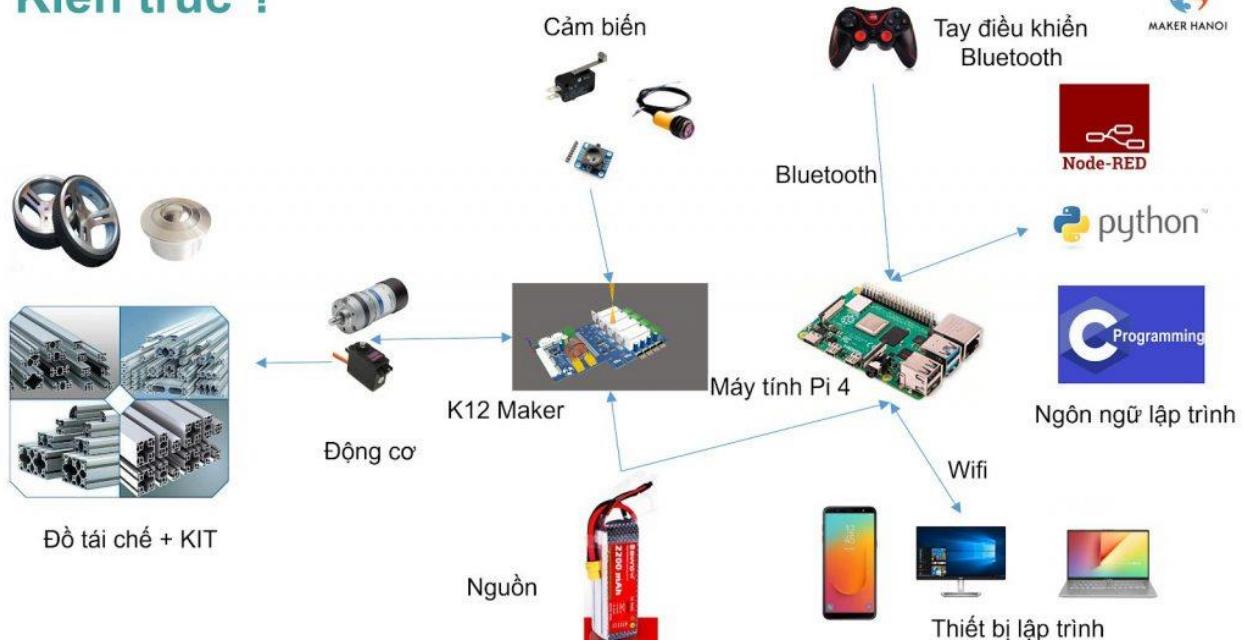


Thử làm: tạo một chú mini robot thông qua bộ K12 Maker hoặc những đồ tự chế

Tham gia cuộc thi VietNam STEAM Challenge – VSC các bạn sẽ có một bộ KIT điện tử, cơ khí để có thể tự dựng cho mình một chú robot hoàn thiện. Sau đó sẽ dùng các công cụ lập trình kéo thả Node-Red hoặc ngôn ngữ lập trình như C/Python để giúp chú robot hoạt động.



Kiến trúc ?



Chia sẻ: Đây là bài chia sẻ của Linh, một học sinh tham gia đội Robot Việt Nam tham gia giải đấu robot toàn cầu mùa 2019 chia sẻ. Tôi muốn các bạn có một góc nhìn khác của một bạn mới bắt đầu làm robot trong thời gian rất ngắn.

Sau 12 năm học cấp 3 chú trọng nhiều về lý thuyết, mình đã có cơ hội được trải nghiệm với engineering ở team Vietnam tham gia First Global Challenge (FGC) 2019 tại Dubai. Suốt gần 3 tháng làm việc và học hỏi cùng với team là một kỷ niệm vô cùng đáng nhớ, ở đó có cả niềm phấn khích lẫn những hối tiếc, nhưng hơn tất cả, FGC đã giúp mình lớn lên, mở rộng tầm hiểu biết, nuôi dưỡng đam mê với STEM của mình.

Ngay sau khi trở về từ cuộc thi Olympic Vật Lý quốc tế, mình đã được chú Lê Ngọc Tuấn (mentor của team Vietnam) giới thiệu đến với FGC. Ngày đầu tiên đến maker Hanoi mình đã rất choáng ngợp. Các bạn ở đây đã làm quen với nhau và có kinh nghiệm với robotics gần một năm trước khi mình tới. Khi nghe mọi người cùng nhau thảo luận về những cơ cấu, động cơ mình cảm thấy như đang bị bỏ lại phía sau vì đó mình còn chưa hiểu được bánh răng hay dây xích dùng để làm gì. Suốt những tuần đầu mình chỉ dám lắng nghe và làm những phần việc đơn giản mà không nói lên những ý tưởng trong đầu. Việc làm quen lúc đầu của mình khá chậm và phải mất một thời gian thì mới bắt kịp được các bạn trong đội. Tuy rằng đến giai đoạn sau mình đã được phân làm engineering lead phụ trách việc lắp ráp robot, nhưng mình vẫn tiếc rằng giá như ở thời gian đầu đó, mình mạnh dạn đưa ra ý kiến, dám sửa sai và học hỏi từ mọi người thì công việc sẽ trở nên hiệu quả hơn rất nhiều. Mình nhận ra rằng STEM là nơi để cùng nhau khám phá, không ai có thể làm mọi việc hoàn hảo ngay từ lần đầu, chỉ cần chúng ta có một tinh thần cởi mở, chắc chắn chúng ta sẽ học được rất nhiều.

Ở maker mình được học rất nhiều kỹ năng khác nhau: cơ khí, lập trình và thiết kế. Mình chủ yếu làm về phần cơ khí. Trước đây, mình đã học khá nhiều về phần Cơ học, các nguyên lý truyền động, nhưng khi bắt tay vào làm thì mình mới nhận ra thực tế rất khác với lý thuyết. Tùy vào mỗi chức năng như là để chuyển động nhanh hay lực nâng tốt mà phải chọn tỉ lệ bánh răng hợp lý. Tất cả những điều này đều không thể đọc từ sách vở nào mà mình cần phải làm thử rồi rút kinh nghiệm cho mỗi trường hợp. Điều kiện thực tế thay đổi cũng làm thay đổi sai số cho mỗi lần thực hiện nhiệm vụ của robot. Trước đây mình nghĩ rằng chỉ cần giấy bút thì có thể tính toán được độ cao, tốc độ bắn tối ưu, nhưng thực tế còn phụ thuộc vào cả năng lượng của pin, động cơ, lực ép do mỗi lần chỉnh là khác nhau. Cả team đã có những lần bị mắc vào một vấn đề quá lâu mà không thể chỉnh được nên đã phải thay đổi toàn bộ kết cấu. Nhìn lại những lần như vậy, mình nghĩ rằng

việc thay đổi là rất quan trọng; việc cố chỉnh sửa một sản phẩm không hoàn hảo có thể còn tốn thời gian hơn việc lắp ráp một cơ cấu mới. Ở phần lập trình tuy mình không làm nhiều nhưng cũng làm quen với ngôn ngữ lập trình Blockly gồm những blocks để người lập trình có thể lắp ráp rất tiện dụng. Việc học lập trình có thể làm cho một người mới như mình khá sợ vì có cảm giác như phải học một ngôn ngữ hoàn toàn mới. Nhưng khi tìm hiểu dần dần thì mình thấy lập trình chính là tư duy logic. Chính vì thế những người yêu thích sự logic trong toán học hay cả ngôn ngữ đều có thể tự tin học lập trình.

Mình còn nhận ra rằng tính ngăn nắp, tỉ mẩn và kỷ luật là rất quan trọng trong engineering. Có những lần vì làm những chỗ nối lỏng lẻo mà cơ cấu không thể hoạt động như mình mong muốn. Mình đã phải để tháo và lắp lại; trong khi đó nếu ngay từ đầu mình cẩn thận hơn trong quá trình lắp ráp thì đã tiết kiệm được rất nhiều thời gian và công sức. Trong thời gian gấp rút để hoàn thành robot, bọn mình đã để maker lộn xộn vì thế mà những linh kiện nhỏ như ốc vít rất khó tìm. Lẽ ra, cuối mỗi buổi làm việc team nên dành thời gian dọn dẹp và sắp xếp các dụng cụ thì công việc sẽ suôn sẻ hơn. Ngoài ra các bạn trong team chủ yếu đều đang học cấp 3 nên thời gian sắp xếp được cho công việc khá ít, vì thế mà bọn mình bị dồn công việc về gần cuối dẫn đến mệt mỏi. Các bạn bắt đầu làm việc nhóm nên xếp lịch họp dãn ra hơn, nhưng mỗi buổi làm việc phải thật tập trung để tránh tình trạng bị quá tải khi gần đến kì thi.

Nếu có cơ hội để bắt đầu một dự án robot mới, mình nghĩ điều quan trọng nhất cần làm là dám thử sai. Ngay từ đầu mình sẽ không quá quan trọng là mục tiêu phải giành bao nhiêu điểm mà làm được 1 cơ cấu hoạt động được trước. Khi hoàn thành, mình sẽ nghĩ đến cách cải tiến để nó hoạt động tốt hơn, nếu bị bế tắc đến hơn 3 tuần mà vẫn không có hiệu quả hơn thì sẽ thử với thiết kế mới. Đồng thời với việc lắp cơ cấu mới thì cũng cần nghĩ đến thiết kế tổng thể của cả robot xem nên lắp đặt các cơ cấu ở đâu. Một tháng trước kì thi là thời gian để hoàn thiện robot, cho dù cơ cấu mình làm vẫn chưa được như mong muốn thì cũng nên dừng lại với thiết kế có hiệu quả cao nhất để ghép robot hoàn chỉnh. Một điều thiếu sót của team bọn mình là chưa cho chạy thử robot nhiều, đến khi thi mới phát hiện có những lỗi do lỏng dây,... Cho nên mình nghĩ nên dành ít nhất là 2 tuần cuối chạy thử robot, làm quen với cách lái, phát hiện những lỗi cơ bản như lỏng dây, lỏng ốc. Sau khi kiểm tra kỹ như vậy thì khi thi dù kết quả như thế nào cũng sẽ

không gây nên hối tiếc nhiều vì chúng ta cũng đã cố hết sức. Mỗi thành viên nên lắng nghe ý kiến của bạn mình một cách trọn vẹn nhất vì chúng ta không thể tưởng tượng được thiết kế đó có khả thi hay không mà chỉ bắt tay vào làm thì mới biết được.

Bên cạnh việc làm quen với engineering, mình còn có những trải nghiệm vui vẻ với những người bạn cùng niềm đam mê STEM. Ngoài việc lắp ráp robot, chúng mình còn làm các video quảng bá Việt Nam đến với thế giới. Team Vietnam có 5 thành viên và mỗi bạn đều có những đóng góp nổi bật cho team. Khi đến Dubai, mình rất ngưỡng mộ các bạn trẻ từ khắp nơi trên thế giới đến cuộc thi với tinh thần của kỳ thi First: tiên phong, hòa nhập, luôn sẵn sàng giúp đỡ. Team bọn mình cũng giúp đỡ và nhận được sự giúp đỡ của các team khác. Mình còn học hỏi được các cơ cấu rất thú vị từ các đội. Đôi lúc mình có chút tiếc nuối vì có những ý tưởng đã từng lóe lên trong đầu nhưng mình vẫn chưa thực hiện vì sợ quá phức tạp và không hiệu quả nhưng có đội đã xây dựng được ý tưởng đó rất thành công. Cả quá trình chuẩn bị và kì thi là một dấu mốc đáng nhớ của mình: được khám phá thế giới STEM, có được những bài học quý giá để phát triển bản thân. Mình luôn hy vọng rằng hoạt động STEM sẽ được đem đến cho tất cả các bạn trẻ Việt Nam vì như người sáng lập của First đã nói: "Kids don't build robots, but robots builds kids" (Những đứa trẻ không xây robots mà chính robots phát triển các em). Những hoạt động STEM sớm sẽ chuẩn bị cho các bạn trẻ những kỹ năng, bài học để bạn sẵn sàng trở thành công dân toàn cầu trong thời đại mới, cho dù sau này bạn theo đuổi bất kỳ lĩnh vực nào.

Chúc các bạn sẽ có những kỷ niệm đáng nhớ với những bước chân đầu tiên đến với STEM!

Phần 6: Tài nguyên và công cụ học tập

Ngày nay các mã nguồn mở và các công cụ, tài liệu rất nhiều để phát triển cho các dự án, đặc biệt là các dự án robot rất phức tạp. Học kiến thức từ cộng đồng mạng, các dự án là một cách giúp các bạn nhỏ tiến bộ rất nhanh. Trong bài viết này tác giả sẽ giới thiệu những trang web, cộng đồng mà tác giả biết và đã từng sử dụng để tự học tập cho trẻ em và người lớn. Thậm chí, các bạn nhỏ từ 5-6 tuổi đã có thể tự lập trình hay chế tạo cho mình những chú robot từ những nguồn tài liệu này.

Máy in 3D

Máy in 3D là một dạng máy công cụ giống các máy CNC truyền thống, giúp người dùng có thể dễ dàng tạo ra những sản phẩm được thiết kế từ những hình vẽ 3D. Ngày nay máy in 3D đang trở nên thông dụng trong các gia đình, các văn phòng trên thế giới. Chúng được áp dụng vào nhiều lĩnh vực khác nhau trong cuộc sống từ in mẫu thử cho sản phẩm, in đồ trang trí, in đồ dùng cá nhân (giá kệ điện thoại), ... thậm chí in cả thực phẩm.





IN nhà cho kỹ sư xây dựng, in bánh cho gia đình , in giày cho nhà thiết kế giày.

Máy in 3D có rất nhiều mã nguồn mở nhưng có 2 dòng thông dụng nhất

- Dòng máy prusa dạng casatarian (trục đòn tay các x y z)



- Dòng máy Delta (trục cánh tay song song)



Giá thành để tự chế các dòng máy in 3D này từ 4 8 triệu VND.

Ngoài ra còn khá nhiều dòng máy thương mại hóa tầm trung như Ultimaker, MakerBot giá từ 50 - 80 triệu đồng.



Cách để tạo ra một sản phẩm in 3D ?

Có các bước cơ bản như sau:

1. Dựng hình 3D bằng các phần mềm 3D

Phần mềm cơ khí chính xác: Inventor, Solidwork, NX9, NX10,....

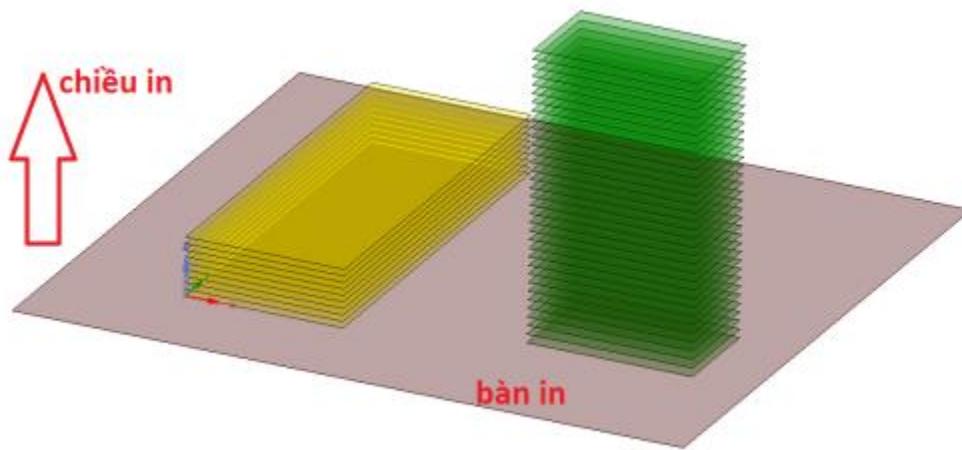
Phần mềm có tính chất nghệ thuật: 3Ds Max, SketchUp, Maya,.....

2. Xuất file 3D ra định dạng *.stl hoặc *.obj (nên để dưới dạng stl)
3. Đưa file *.stl hoặc *.obj vào phần mềm xử lý in 3D (Cura, SLic, Makerbot....)
4. Từ phần mềm xử lý in 3D save ra định dạng GCODE mà máy in 3D hiểu được
5. Chép file GCODE vào USB (thẻ nhớ) và cắm vào máy in để in 3D

Nếu bạn không có khả năng thiết kế có thể lấy các mẫu in 3D sẵn trên trang web <https://www.thingiverse.com>

Các bước trên khá đơn giản, tuy nhiên khi sử dụng máy in 3D các bạn sẽ phải lưu ý khá nhiều vấn đề cần lưu ý.

Hướng in từ dưới lên: Mức độ dính chặt lại phụ thuộc vào 3 yếu tố trong quá trình cài đặt phần mềm in 3D (nhiệt độ, bề dày lớp in, tốc độ). Điều này có thể can thiệp trong quá trình chạy máy in, tuy nhiên, bạn phải giữ trong đầu những ý niệm về “hướng in 3D”.



Biểu diễn 2 mẫu thiết kế giống nhau nhưng được sắp đặt theo 2 hướng in hoàn toàn khác. Các ván bên hông tượng trưng cho các lớp in 3D. Độ bền của mẫu in 3D màu vàng vượt trội so với mẫu màu xanh.

Để mẫu in 3D được đẹp, đúng kích thước, tiết kiệm vật liệu – thời gian in – chi phí, bạn cần đảm bảo các vấn đề sau:

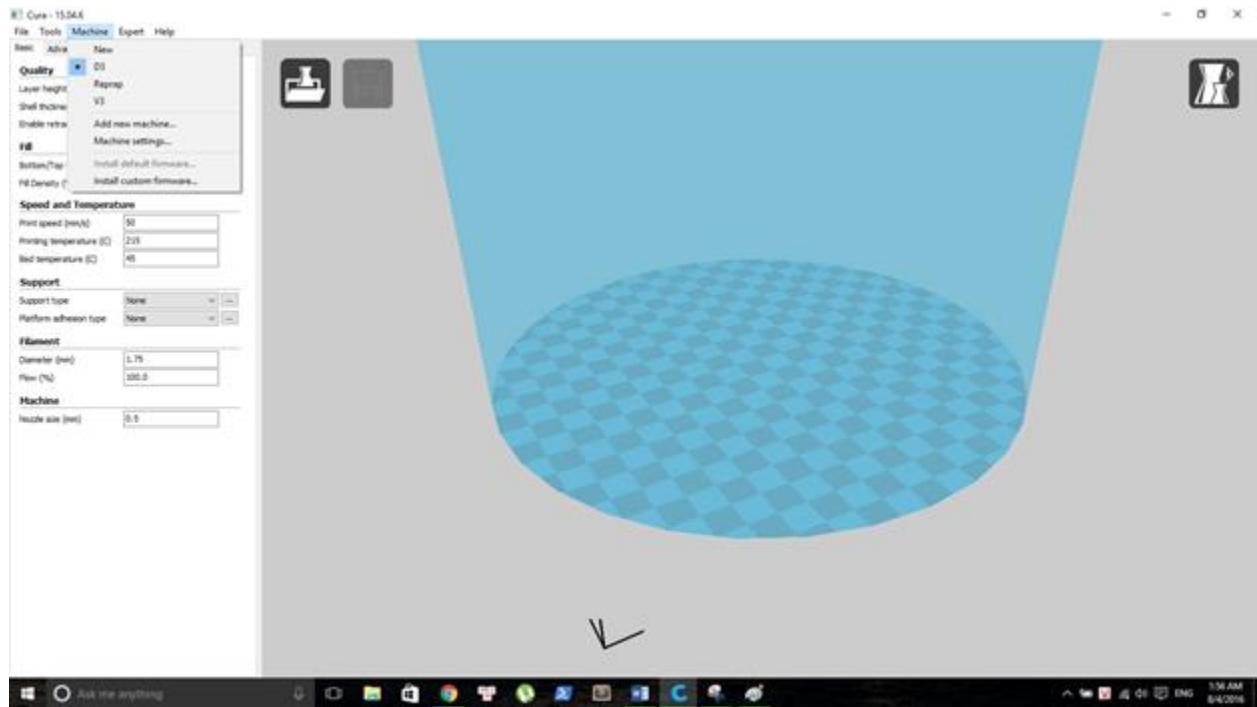
- Thiết kế mô hình theo kiểu “kim tự tháp” tức là dưới to trên nhỏ.
- Nên có một mặt để phẳng bên dưới mô hình.
- Hạn chế các vị trí mỏng hơn 1,2mm.
- Các phần quá bé trên mô hình 3D (0,1-1mm): mắt, mũi, tai, gờ, nút bấm,... rất khó hoặc không thể in 3D!
- Các phần nhô ra nên có góc nghiêng >45 độ so với phương ngang. Hạn chế phần nhô ra nằm ngang, hoặc phía dưới trống không (ví dụ như cây cầu)!

- Nên không chế mô hình nằm vừa khít trong khung in của máy in 3D, cũng đừng nên quá bé (không in được hoặc in ra xấu!)
- Các chi tiết có lắp ghép thì khoảng cách giữa 2 bề mặt nên để: Lắp lỏng $\geq 0,4\text{mm}$; Lắp chật $\leq 0,2\text{mm}$.
- Chú ý tới độ phân giải của mô hình khi xuất ra file STL
- Chắc chắn về kích thước file STL là theo hệ inch hay mm!
- Mở lên xem lại file STL/OBJ vừa xuất ra. Hoặc dùng công cụ kiểm tra lỗi file 3D

Phần mềm tạo file chạy máy in 3D

Cura là phần mềm cài đặt thông số để in mô hình 3D và dễ sử dụng nhất. Cura là 1 phần mềm mã nguồn mở được phát triển bởi hãng Ultimaker. Giúp cho việc in 3D, tạo mẫu nhanh 3 chiều trở nên dễ dàng hơn, chất lượng sản phẩm tạo ra cũng tốt hơn. Cura đi kèm với 1 chương trình cài đặt thân thiện giúp bạn luôn luôn cập nhật phiên bản mới nhất với những tính năng mới.

Cura hỗ trợ rất nhiều dòng máy in 3D theo công nghệ FDM Giao diện thân thiện, tính năng mạnh mẽ, thuật toán xử lý mô hình 3D nhanh và chuẩn xác, thao tác đơn giản, Cura là sự lựa chọn tốt nhất cho người mới bắt đầu sử dụng máy in 3D.



Máy in 3D ngày càng trở nên thông dụng, trong thế kỷ 21 có thể mỗi gia đình sẽ có một chiếc máy in 3D để in những vật dụng hàng ngày, những công cụ học tập cho trẻ em, đồ làm bếp cho mẹ, đồ làm việc sửa chữa cho bố. Hiện nay máy in 3D đang trong giai đoạn phát triển rất mạnh để cải tiến các vấn đề về tốc độ, vật liệu, độ tiện lợi và dễ dàng cho sử dụng.

Arduino – Cộng đồng mã nguồn mở

Là một cộng đồng mở về phần cứng và phần mềm, bắt nguồn từ một Lab tại một trường đại học của Italy. Mục tiêu hướng tới của nền tảng này là giúp cho học sinh và người mới làm quen với phần cứng, lập trình nhúng một cách dễ dàng và nhanh nhất.

Nền tảng Arduino ban đầu chỉ hỗ trợ dòng chip 8 bit AVR của Atmel, ngày nay nó gần như hỗ trợ mọi dòng chip mới và thông dụng.

Ngày nay mọi ứng dụng IoT hay các sản phẩm mã mở đều được chia sẻ trên mạng. Các dự án DIY, maker đều sử dụng nền tảng này để chia sẻ.

Một người mới có thể bắt đầu làm điện tử và lập trình nhúng rất nhanh, chỉ cần các bạn chịu tìm kiếm.

Thử làm: Mua một bộ mạch Arduino Uno và bắt đầu dự án đầu tiên, lập trình đèn nhấp nháy!

Máy tính nhúng giá rẻ

Một chiếc máy tính \$5 – Pi Zero, một chiếc nút thông minh Amazon Dash nối wifi \$5, một máy tính nhúng C.H.I.P có giá \$9 gồm cả wifi, bluetooth, flash 4Gb, ram 512, chip 1Ghz. Thậm chí tại Ấn Độ, một chiếc điện thoại thông minh chỉ có giá xấp xỉ 33.000 VND...

Hệ sinh thái sản phẩm IoT là một miếng bánh lớn trong tương lai không xa. Các hãng sản xuất đang hướng tới việc đưa ra thị trường các sản phẩm IoT có giá thậm chí còn thấp hơn cả giá sản xuất để xây dựng thương hiệu, tạo ecosystem riêng và xâm nhập thị trường người tiêu dùng.

Hệ sinh thái IoT và bí mật phía sau các sản phẩm giá rẻ:

C.H.I.P là startup nổi lên năm 2015 với một sản phẩm máy tính nhúng giá rẻ. Chỉ với \$9 bạn có thể có một máy tính nhúng đầy đủ các tính năng từ RAM, FLASH, WIFI, Bluetooth, CPU,...

Bắt đầu từ tháng 7/2016, Công ty Next Thing mới chính thức gửi những sản phẩm giá rẻ tới tay người tiêu dùng nhưng chỉ trong 1 tháng hãng đã mang về 2,000,000 USD từ những người sẵn sàng mua trước sản phẩm để dùng thử. Bạn có thể dễ dàng đặt mua sản phẩm C.H.I.P trên kickstarter hay trang web chính thức của hãng.

Là một trong những đối thủ đáng gờm, nổi tiếng một thời gian dài với dòng máy tính nhúng Pi raspberry trong giới những người làm DIY chỉ với giá 35\$, năm 2015 hãng cũng phải tuyên bố ra đời dòng sản phẩm mới Pi Zero với giá \$5 và chính thức bán ra rộng rãi vào năm 2016 để có thể cạnh tranh với C.H.I.P.

Ngoài Pi, C.H.I.P còn có rất nhiều tên tuổi máy tính nhúng cũ như Ordroid, Beaglebone, Jetson,... các tên tuổi mới như Pine64, Samsung Artik,...

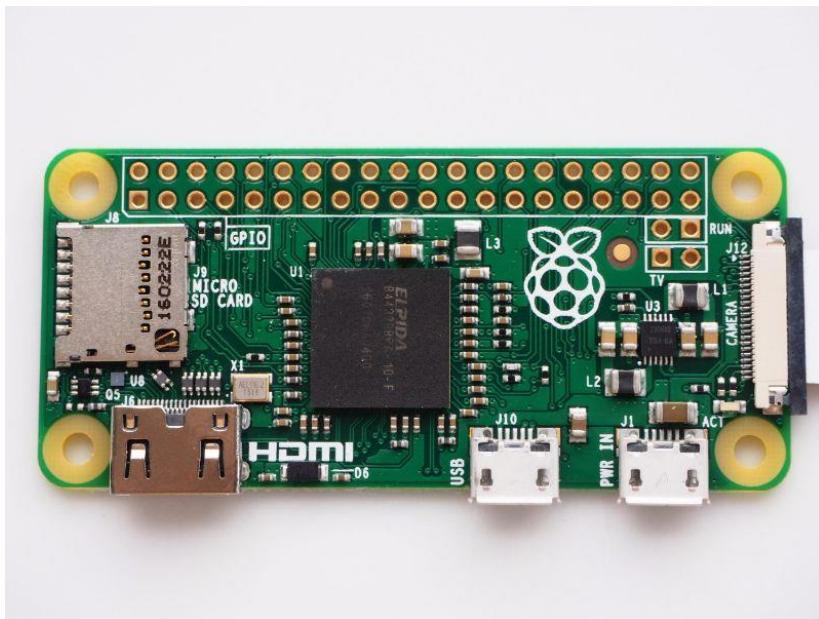
Nguồn lực về người và thời gian là một trong những yếu tố quan trọng khiến cho các công ty vừa và nhỏ khó có thể thiết kế hoàn thiện và sản xuất máy tính nhúng giá rẻ. Trong khi đó, các hãng máy tính nhúng đều có tham vọng chiếm được thị phần dòng sản phẩm IoT. Để làm được điều này, các hãng đều đang nỗ lực không ngừng để đưa ra các sản phẩm mới với cấu hình cao, khỏe, có khả năng xử lý độc lập, tích hợp được với các sản phẩm IoT mới với nhiều thư viện, công cụ hỗ trợ các lập trình viên và đặc biệt là vẫn RẺ. Với tính cạnh tranh khốc liệt mà chúng ta có thể dự đoán được, có vẻ như tương lai sẽ chỉ còn 2-3 hãng có khả năng tồn tại như cuộc chiến giữa Intel và Arm trong thời mobile, PC. Cho đến thời điểm hiện nay, các hãng đang cố gắng hết mình để có thể đưa đến tay người tiêu dùng các sản phẩm với tính năng cao và giá thành rẻ nhất.

Cấu hình cơ bản một số dòng máy tính nhúng

Pi Zero – \$5

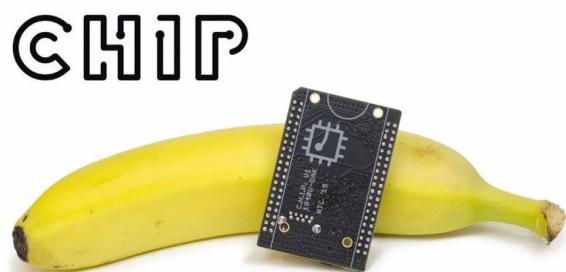
- 1Ghz, Single-core CPU
- 512MB RAM

- Mini HDMI and USB On-The-Go ports
- Micro USB power



C.H.I.P – \$9

- 1Ghz, Single-core CPU
- 512MB RAM
- 4GB of High speed Storage
- Bluetooth 4.0
- Wifi B/N/G build-in



PINE64 – \$15

- 1.2 Ghz Quad-Core ARM Cortex
- 512MB/ 1GB/ 2GB RAM
- Dual core Mali 400 MP2 Graphics
- Video 4K H.265
- Bluetooth 4.0 and 802.11BGN connectivity optional.



Giá trị của một mạch máy tính nhúng?

Nếu bạn làm người làm và hiểu các thiết bị phần cứng thì hiện nay giá một module Bluetooth Smart khoảng \$4 – \$6, một module wifi giá cũng dao động từ \$5 – \$10. Một con chip Bluetooth của hãng Nordic nếu mua lẻ giá là \$5 và mua trên 1000 con giá giảm khoảng 50% (\$2.6).

Trong khi một máy tính nhúng có rất nhiều linh kiện đắt tiền hơn như vi điều khiển trung tâm ARM, RAM, FLASH.

Nếu bạn tự làm và thiết kế một máy tính nhúng, tiền linh kiện có thể lên tới trên \$100, như vậy giá một máy tính nhúng sẽ giảm được 10-20 lần.

Có thể nói, để có thể sống sót trong môi trường cạnh tranh khốc liệt giữa các sản phẩm IoT, các doanh nghiệp cần phải có kế hoạch dài hạn về chính sách giá và cách thức xâm nhập thị trường người tiêu dùng. Doanh nghiệp cần có một mức giá phù hợp để có thể thu hút được khách hàng cho riêng mình, mức giá người viết nghĩ sẽ dễ dàng được người tiêu dùng chấp nhận vào khoảng từ 5\$ đến 9\$.

Ngày nay, các hãng công nghệ, các nhà đầu tư đang nỗ lực hết mình để chiếm lấy thị phần trong hệ sinh thái sản phẩm IoT. Hiện nay, họ đang lên kế hoạch để giảm thiểu chi phí sản xuất một cách tối đa cho hàng triệu sản phẩm. Không chỉ giới hạn bản thân với việc chỉ bán phần cứng, tương lai không xa, các doanh nghiệp này sẽ cung cấp thêm các dịch vụ trên chính những sản phẩm của họ. Cùng với các start-up, họ sẽ hỗ trợ nhau để đưa ra những sản phẩm và dịch vụ mới với nhiều tính năng và công dụng ưu việt. Người viết cho rằng thế giới sản phẩm IoT trong tương lai sẽ hợp tác win-win để đáp ứng các nhu cầu rất nhỏ của con người.

Chế tạo sản phẩm IoT trong 30 phút

Với mã nguồn mở và các thiết bị có sẵn, bạn có nghĩ mình đủ khả năng xây dựng một ứng dụng cho các sản phẩm IoT. Để làm một sản phẩm IoT demo cho ý tưởng của bạn không hề đơn giản, bạn cần rất nhiều kỹ năng: Phần cứng thiết kế mạch, nắm rõ mạng thiết kế kết nối Internet, hiểu về server, có mobile app. Bài viết dưới đây sẽ hướng dẫn các bạn tiếp cận 3 nền tảng để làm nhanh sản phẩm IoT trong vòng 30 phút cho sản phẩm MVP.

Hiện nay, có rất nhiều cộng đồng và nền tảng phát triển hỗ trợ làm sản phẩm IoT, trong bài này người viết sẽ đi sâu vào phân tích và sử dụng những nền tảng đại diện cho từng nhóm công nghệ.

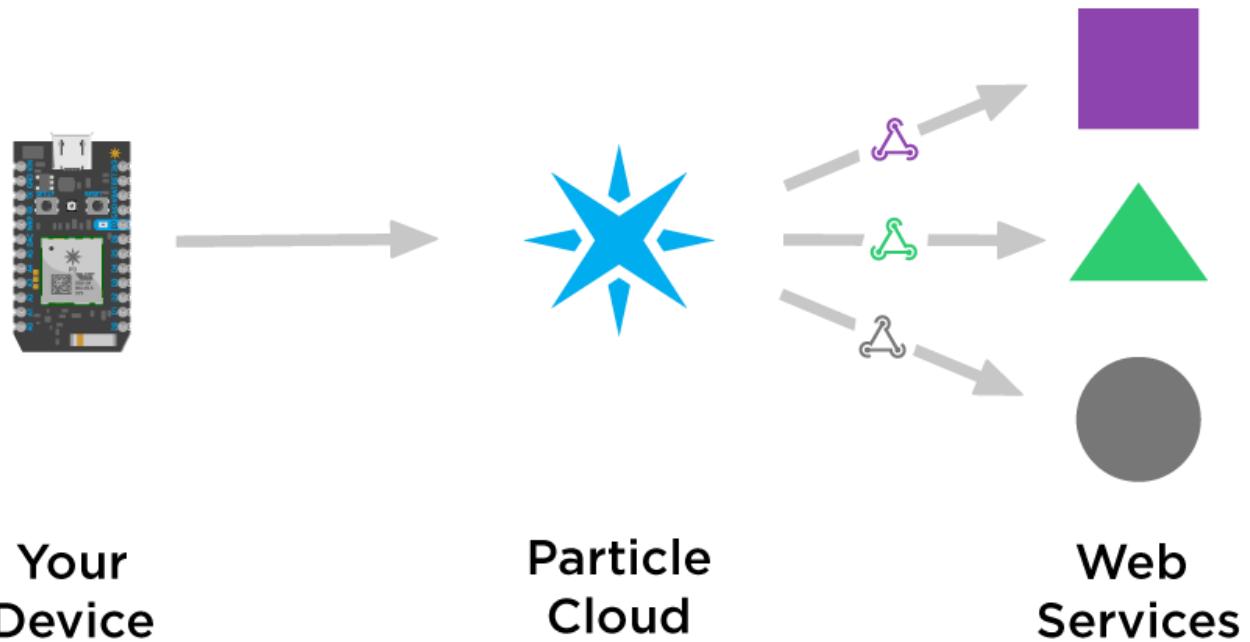
- **Social:** Cộng đồng mã nguồn mở dùng ngôn ngữ Arduino – Top 3 ngôn ngữ lập trình thông dụng.
- **Mobile:** Blynk một ứng dụng trên hai nền tảng iOS và Android giúp bạn làm nhanh một ứng dụng trong vòng 5 phút để giám sát và điều khiển các thiết bị IoT.
- **Analyze:** SpeakThing nền tảng mã nguồn mở IoT cho phép bạn hiển thị và phân tích dữ liệu.
- **Cloud:** Particle Dashboard hỗ trợ lập trình IDE online, lập trình một thiết bị phần cứng như lập trình web với Restful
- **Thing:** Photon và SparkCore là một mạch được thiết kế tích hợp có thể lập trình bằng ngôn ngữ Arduino và kết nối qua mạng. Toàn bộ nền tảng này là OpenSource của hãng Particle. Bạn có thể sử dụng Esp8266 để thay cho Photon. Mua rất sẵn tại Việt Nam.

Photon

Photon là một mạch tích hợp bao gồm một chip lõi ARM và một chip Wifi để kết nối Internet. Phiên bản đầu tiên của nó có tên là Spark Core, được phát triển bởi hãng Particle vào năm 2013 từ dự án trên Kickstart.



Photon được tích hợp sẵn các thư viện dùng cho việc kết nối mạng, nó dễ dàng được lập trình bằng Arduino chỉ trong mấy phút. Ví dụ, chỉ cần hai dòng lệnh chúng ta có thể biến Photon thành một server REST API thông qua Particle Cloud để thu thập dữ liệu hoặc điều khiển các thiết bị khác. Ngoài ra, Particle Cloud đã hỗ trợ Webhook và có thể kết nối tới dịch vụ cloud của các hãng công nghệ lớn như Microsoft, Amazon và IFTTT – “**if this, then that**”.

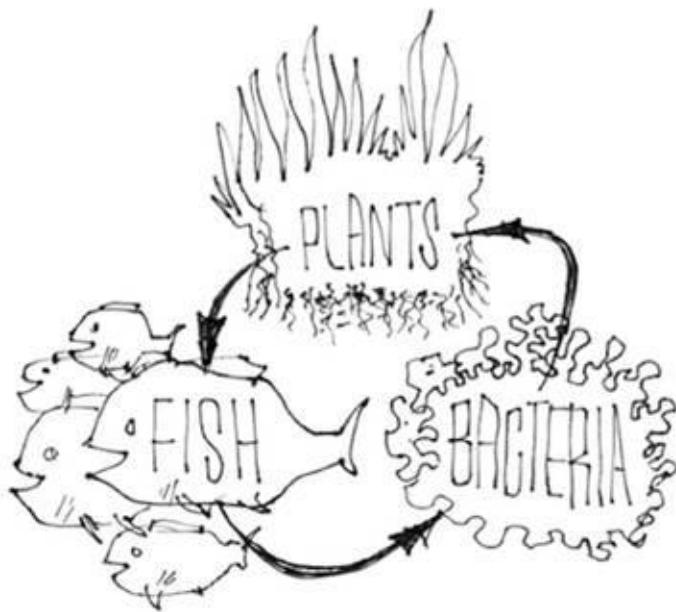


Ở giai đoạn đầu tiên trong tiến trình khởi nghiệp của mình, nhiều doanh nghiệp trên thế giới đã dùng Photon để phát triển Prototype. Thậm chí sau khi đã có những nền tảng vững chắc họ vẫn tiếp tục làm việc với Particle để biến nó thành sản phẩm thương mại, hỗ trợ mở rộng quy mô quản lý sản phẩm. Trong đó, sản phẩm tiêu biểu mà người viết muốn nhắc đến ở đây là máy pha cà phê Keurig. Sản phẩm này chỉ làm thử trong một ngày, sau một tháng đội ngũ sản xuất đã hoàn thành sản phẩm và chỉ 6 tháng sau, sản phẩm đã được trình làng trực tiếp trên thị trường.

Michael Cunningham, CIO, Keurig

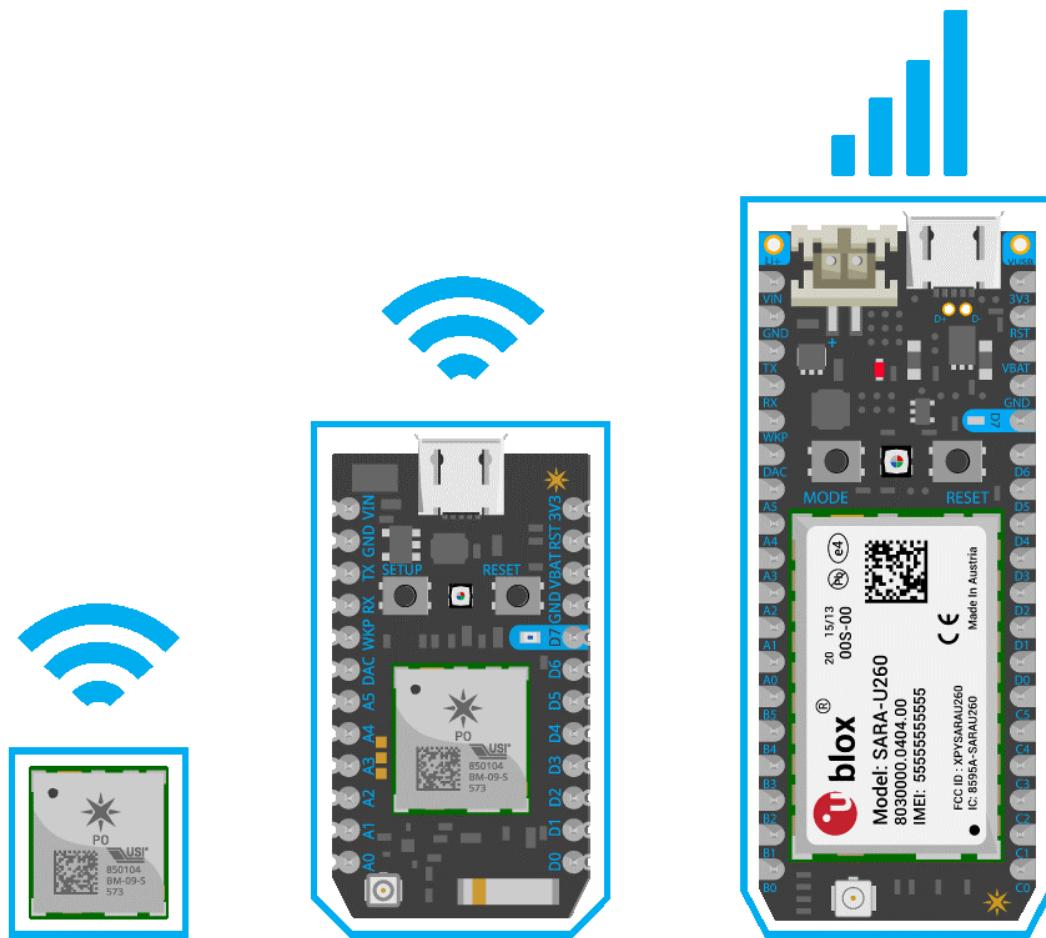


Hệ sinh thái vườn thông minh trong nhà của Grove Lab (<https://grovelabs.io>) cũng là một trong những sản phẩm tiêu biểu khi chỉ với duy nhất một kỹ sư nhúng, họ đã hoàn thành dự án này.



Các sản phẩm của Particle đều được OpenSource trên GITHUB tại địa chỉ <https://github.com/spark>

Hiện nay hãng có 3 dòng sản phẩm chính là module wifi Photon, KIT Photon, Electron 2G/3G, dòng sản phẩm đầu tiên Spark Core đã ngừng bán.

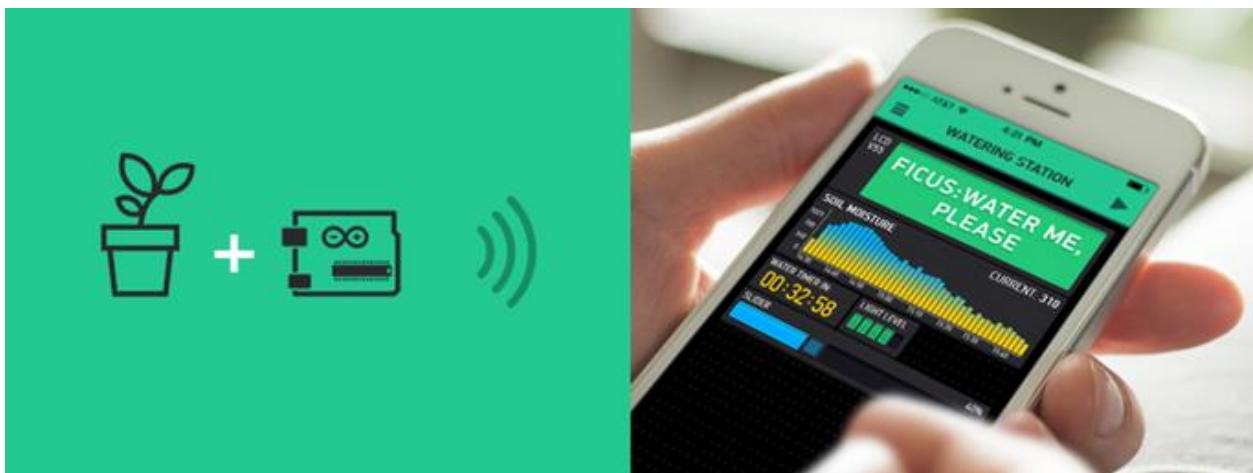


Blynk

Blynk là một ứng dụng trên iOS và Android hỗ trợ lập trình viên viết các ứng dụng di động cho thiết bị thông minh – IoT chỉ trong vài phút. Ứng dụng này dễ dàng kết nối với các mạch tích hợp và nền tảng thông dụng như Arduino, Raspberry Pi, Esp8266, Particle (Photon/ SparkCore) thông qua Internet. Với Blynk Cloud, người dùng có thể đồng bộ dữ liệu với ứng dụng di động từ thư viện ở các nền tảng khác nhau. Sản phẩm này có giao diện tương đối dễ dùng, thao tác bằng cách kéo thả và hiện đang miễn phí với dự án thử nghiệm, dùng thử. Riêng đối với một số thiết bị, ứng dụng Blynk đã bắt đầu có hình thức thu phí.



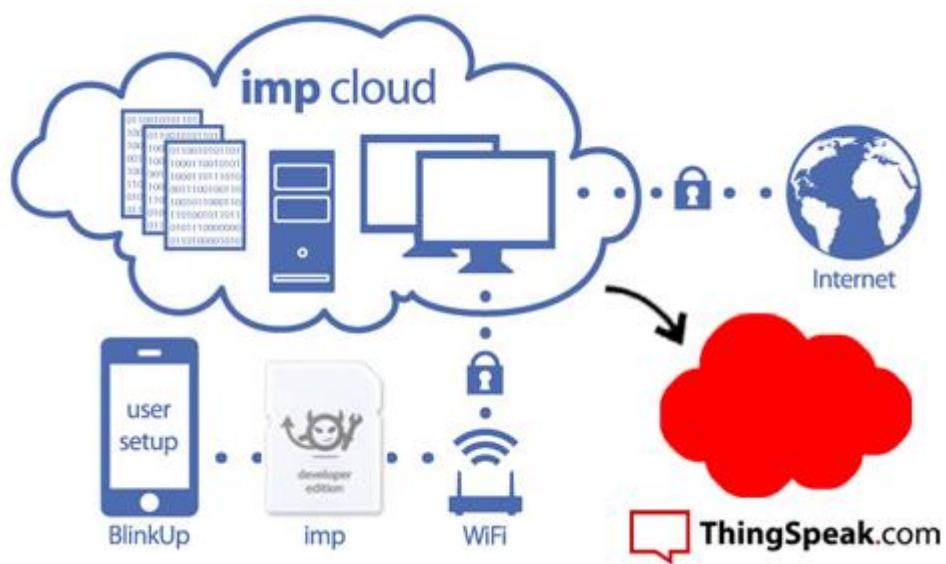
Các ứng dụng tiêu biểu của Blynk: tay điều khiển Drone, giám sát và ứng dụng trồng cây thông minh qua Arduino. Chi tiết tham khảo tại: <http://www.blynk.cc/>



ThingSpeak

ThingSpeak là một mã nguồn mở cho các ứng dụng của "Internet of Things". Mã nguồn này hỗ trợ các API lưu trữ, lấy dữ liệu từ các thiết bị, sản phẩm sử dụng HTTP qua Internet hoặc thông qua một Local Area Network. Như một HUB đợi các thông tin cảm biến từ thiết bị và có nhiệm vụ lưu trữ và xử lý dữ liệu, với ThingSpeak, bạn có thể tạo ra các ứng dụng phân tích dữ liệu, lưu trữ dữ liệu, quản lý dữ liệu một cách đơn giản.

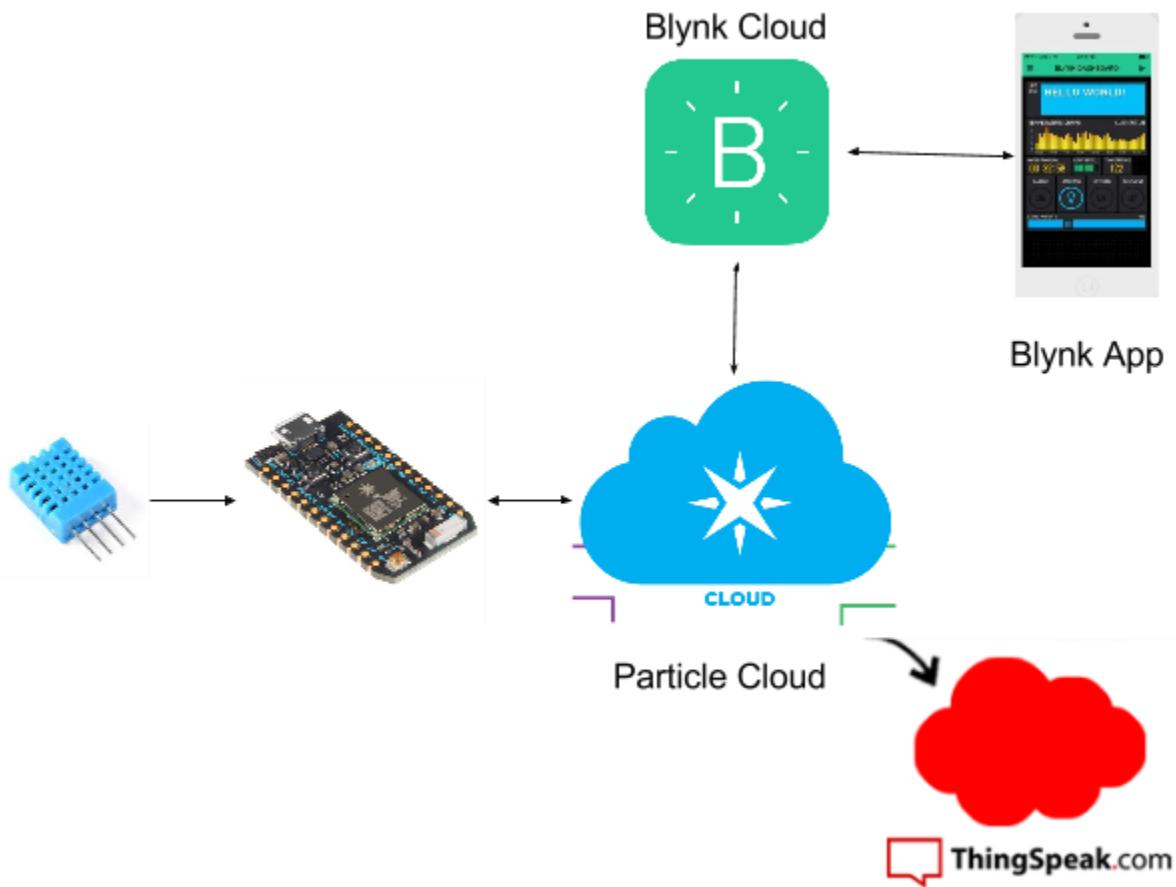
ThinkSpeak được phát triển bởi ioBridge và được opensource trên GITHUB
<https://github.com/iobridge/thingspeak>



Kết hợp 3 nền tảng trên để xây dựng nhanh một sản phẩm IoT

Xây dựng ứng dụng theo dõi nhiệt độ, độ ẩm phòng sử dụng cảm biến DHT11 /DHT22

Mô hình sản phẩm



Bước 1:

Lập tài khoản tại 3 dịch vụ

<https://thingspeak.com>

<https://build.particle.io>

www.blynk.cc

Bước 2:

Setup wifi và kết nối Photon vào tài khoản Particle Cloud.

<https://docs.particle.io/guide/getting-started/start/photon/>

Bước 3:

Cắm cảm biến DHT11/DHT22 vào Photon và kiểm open source lập trình giao tiếp DHT11/DHT22

<http://www.geothread.net/the-internet-of-things-with-photon-temperature-and-humidity-logging/>

Biên dịch và nạp code online.

Bước 4:

Tải ứng dụng Blynk trên điện thoại, login tạo một dự án, chọn mạch tích hợp là Photon.

Lấy Token của dự án và gửi về mail của bạn.



Thêm code vào code của Photon để đẩy dữ liệu và giao tiếp với Blynk Cloud.

Chi tiết hướng dẫn <https://www.hackster.io/gusgonnet/temperature-humidity-monitor-with-blynk-7faa51>

Bước 5:

Đăng nhập vào ThingSpeak, tạo một Channel của dự án.

Tiến hành lấy thông tin để đưa vào dự án trên Photon

```
String writeAPIKey = "A7SDN4JU76WMPIJ4";
```

```
String channelID = "14295";
```

Chi tiết hướng dẫn:

<https://www.hackster.io/makers-ns/particle-photon-thingspeak-e063d5>

Kết luận:

Vậy là sau 5 bước, với các account có sẵn và một số kĩ năng căn bản, chỉ mất 30 phút bạn đã có thể làm một ứng dụng demo đơn giản cho ý tưởng của mình. Bắt tay vào làm ngay và luôn để kiểm tra tính khả thi cho ý tưởng của mình, tại sao không?

Tuy nhiên, người viết có thể khẳng định rằng làm một sản phẩm IoT thực sự không hề đơn giản. Để có một sản phẩm IoT chạy độc lập và liên tục trong một thời gian dài, đội làm sản phẩm cần chăm chút kĩ lưỡng từng tính năng, tối ưu các vấn đề về năng lượng và ổn định của sản phẩm. Với việc dùng các nền tảng có sẵn, bạn sẽ hoàn toàn phụ thuộc vào kế hoạch và khả năng của bên thứ 3 và khó có thể làm chủ các sản phẩm của mình.

Với những hướng dẫn chi tiết ở trên, người viết tin rằng bạn có thể làm sản phẩm demo trong 30 phút nhưng để hoàn thiện sản phẩm từ những tính năng nhỏ nhất bạn có thể sẽ phải mất cả năm. Để có được một sản phẩm IoT ưng ý tung ra thị trường, bạn phải đầu tư vào phần cứng, phần mềm, sản xuất thử nghiệm và theo dõi ý kiến của khách hàng.

Tạp chí và các diễn đàn, website sáng tạo

Học kiến thức từ cộng đồng mạng, từ các dự án mở ra một cách giúp các bạn tiến bộ rất nhanh. Trong mục này tôi sẽ giới thiệu các bạn những trang web, những cộng đồng mà tôi biết và đã từng sử dụng để tự học tập. Tất cả các link sẽ được tôi cập nhật trên blog tại địa chỉ www.robotbook.makerviet.org

- 1.** Tạp chí, sự kiện, tổ chức hay về sáng tạo
- 2. Make Magazine:** Một tạp chí hàng đầu về những sáng tạo mới, các chế tạo DIY cho mọi lứa tuổi. Tạp chí có bản giấy in và phiên bản online, các bạn có thể mua và đăng ký hàng tháng. – <https://makezine.com>
- 3. Makers Faire:** Một sự kiện lớn toàn cầu được tổ chức hàng năm tại nhiều địa điểm. Đây là nơi mà mọi người có thể tới xem hay đem các sáng tạo, sản phẩm DIY của mình để chia sẻ với mọi người. <https://makerfaire.com/>
- 4. Instructables:** Một tạp chí online nổi tiếng về DIY, gần như tất cả các dự án mở tự chế đều có và được chia sẻ trên trang web theo phong cách làm từng bước một. <https://www.instructables.com/>
- 5. Fab Central -** Website chính thức của tổ chức hơn một ngàn Fablab toàn cầu, trang web của đơn vị sáng lập ra mô hình này MIT Center for Bits and Atoms . Fablab là không gian sáng tạo offline cho các bạn trẻ, người đi làm về phần cứng, chế tác,...
<http://fab.cba.mit.edu/>
- 6.** Tại Việt Nam chúng ta có rất nhiều FabLab tuy nhiên có 4 FabLab được hình thành đầu tiên xây dựng cộng đồng Maker tại Việt Nam là:
 - 7.** FabLab SaiGon - <https://fablabsaigon.org/>
 - 8.** FabLab Đà Nẵng

- 9.** FabLab Hà Nội
- 10.** Maker Hanoi <http://makerviet.org/>
- 11.** Trang web hay về dự án DIY
- 12.** **DIY KIDS** khóa học online cộng đồng chia sẻ những sản phẩm tự chế của các bạn nhỏ <https://diy.org/>
- 13.** **Howtoons:** cộng đồng hướng dẫn sáng tạo, làm sản phẩm dựa vào truyện tranh. Ngoài truyện, trang web còn có những video thể hiện kết quả rất thú vị cho các bạn trẻ yêu khoa học và truyện tranh. <https://howtoons.com/>
- 14.** **Make Magazine Projects –** Dự án điện tử thú vị của tạp chí makezine, các bài viết hay về robot, máy tính nhúng, điện tử, khoa học hay sáng tạo nghệ thuật đều được chia sẻ tại blog này. <https://makezine.com/projects/>
- 15.** **Sparkfun Tutorials -** Trang web chuyên các đồ điện tử cho dân DIY, trang có riêng một blog để hướng dẫn cho những người mới bắt đầu như Arduino, điện tử, dụng cụ điện tử cơ bản,... <https://learn.sparkfun.com/tutorials>
- 16.** **Adafruit Learning System -** Trang web chuyên các đồ điện tử cho dân DIY, khá tương đồng với sparkfun và có các chia sẻ hệ thống hơn về các máy tính nhúng Arduino, Pi, Microbit,...<https://learn.adafruit.com/>
- 17.** **Makerbot 3D Printing Education –** Khóa học thú vị về STEM và chế tạo, làm các thiết kế đơn giản cho các bạn nhỏ. <https://www.makerbot.com/education>
- 18.** **Tinkering Studio –** Trải nghiệm, dự án và sản phẩm thú vị cũng như các hoạt động của bảo tàng khoa học Exploratorium tại California – Mỹ <https://www.exploratorium.edu/tinkering>
- 19.** Các cộng đồng mạch điện tử nổi tiếng

- 20. Raspberry Pi** – Cộng đồng máy tính nhúng giá rẻ lớn nhất thế giới hiện nay, rất nhiều các mã nguồn mở được xây dựng cho các dự án DIY. Có chức năng đầy đủ như một chiếc máy tính laptop hay PC (cần lắp thêm màn hình và bàn phím) với giá chỉ \$35.
<https://www.raspberrypi.org/>
- 21. Arduino** : là một cộng đồng mã nguồn mở phần cứng và phần mềm cho các thiết bị lập trình nhúng, thiết bị điện tử. Cộng đồng này đã xóa bỏ giới hạn giữa dân phần cứng và phần mềm. Người làm công nghệ thông tin cũng có thể làm phần cứng và ngược lại người làm phần cứng cũng có thể dễ dàng sử dụng những phần mềm mở.
<https://www.arduino.cc/>
- 22. Adafruit:** Trang web chuyên các đồ điện tử cho dân DIY, các bạn có thể dễ dàng mua các linh kiện hay các mạch nổi tiếng tại đây, ngoài ra Adafruit cũng có những mạch thiết kế riêng của mình <https://www.adafruit.com/>
- 23. Sparkfun** Trang web chuyên các đồ điện tử cho dân DIY gần tương đồng như Adafruit, trang web này thì có thêm nhiều khóa học thú vị cho cộng đồng trên blog
<https://www.sparkfun.com/>
- 24. Seeedstudio** Trang web chuyên tạo ra các sản phẩm KIT mới thú vị cho người dùng, đặc biệt trang web này còn đưa ra quy trình lean factory, giúp các DIY, maker có thể chế tạo sản phẩm của mình từ ý tưởng đến khuôn mẫu, pcb.
<https://www.seeedstudio.com/>
- 25. MaKey Makey:** Là một bộ mạch sáng tạo cho mọi độ tuổi, chỉ cần cắm và chạy, các bạn nhỏ có thể tự chế chiếc đàn từ các loại hoa quả, hay một cái đàn ghita bằng giấy có thể hoạt động được,...Nó đơn giản là một thiết bị biến mọi thứ trở thành bàn phím để nối với máy tính. <https://makeymakey.com/>
- 26. Lilypad:** Là một bộ mạch sáng tạo cho các bạn thích làm đồ thời trang, các module linh hoạt để có thể may vào các bộ quần áo tạo thành các sản phẩm vô cùng sáng tạo.
<https://www.arduino.cc/en/Main/ArduinoBoardLilyPad/>

- 27.** Các phần mềm sáng tạo
- 28.** **Code.org :** trang web dạy lập trình cho trẻ em nổi tiếng toàn cầu, chỉ trong vòng 1 giờ, một bạn nhỏ 5 tuổi có thể kéo thả cho mình một sản phẩm điều khiển nhân vật hoạt hình và làm quen với logic của máy tính. <https://code.org/>
- 29.** **Scratch:** là một nền tảng, công cụ biên dịch lập trình kéo thả nổi tiếng dành cho trẻ em được các kỹ sư MIT xây dựng và phát hành mã nguồn mở. Hiện nay Scratch có rất nhiều các phiên bản khác nhau, còn phiên bản chính thức thì đã được online hóa. Học sinh có thể làm và chia sẻ các dự án trên website này. <https://scratch.mit.edu/>
- 30.** **S4A Scratch for Arduino:** là một phần mềm biến thể từ Scratch giúp kết nối mạch Arduino tới công cụ lập trình scratch, nhược điểm của phần mềm này là khi lập trình và chạy ứng dụng luôn phải nối với máy tính và S4A. <http://s4a.cat/>
- 31.** **mBlock:** là một phần mềm biến thể từ Scratch, tuy nhiên họ xây dựng thành hệ sinh thái để các đơn vị khác cũng có thể đưa mạch của mình và dùng chung nền tảng của họ. Hiện nay đã có phiên bản online để có thể lập trình các sản phẩm robot, arduino một cách dễ dàng. <https://www.mblock.cc/en-us/>
- 32.** **MakeCode – Microsoft** là một nền tảng công cụ lập trình online cho trẻ em do Microsoft phát triển. Hiện nay Makecode được tích hợp khá nhiều các sản phẩm phần cứng thú vị như Lego, Microbit – BBC, Minecraft,... <https://www.microsoft.com/en-us/makecode>
- 33.** **Blocky** là một nền tảng mã nguồn mở của Google phát hành để biến ngôn ngữ kéo thả thành bất kỳ các ngôn ngữ lập trình chuyên nghiệp nào, nền tảng này giúp rất nhiều startup khởi nghiệp về giáo dục ứng dụng vào lập trình những sản phẩm cho trẻ em như microbit, ozbot, ... <https://developers.google.com/blockly>
- 34.** **MIT App Inventor** là một phần mềm hỗ trợ các bạn trẻ lập trình mobile bằng các công cụ kéo thả. Đặc biệt các bạn chỉ cần lên web cũng có thể nhanh chóng tạo cho mình

một ứng dụng trên điện thoại di động. Bản đầu tiên của nền tảng này chỉ hỗ trợ Android chưa có phiên bản iOS <https://appinventor.mit.edu/>

35. Các công cụ, diễn đàn về chế tạo

36. **Thingiverse:** Đây là trang web chia sẻ các mẫu thiết kế mở cho máy cắt, máy CNC, máy in 3D. Gần như tất cả các mẫu in 3D mở đều có trên trang web này. Bạn muốn in một thứ gì hãy lên Thingiverse tìm mẫu in hoặc ý tưởng có sẵn. <https://www.thingiverse.com/>

37. **SketchUp:** Phần mềm khá dễ sử dụng cho người mới bắt đầu vẽ và thiết kế 3D. Phần mềm này miễn phí trong giai đoạn đầu và hiện nay đã thu phí sử dụng. Ưu điểm có rất nhiều thiết kế sẵn từ cộng đồng sử dụng phần mềm này. <https://www.sketchup.com/>

38. **Tinkercad - Autodesk's 123D:** Phần mềm miễn phí do hãng AutoDesk phát hành, khá dễ dàng sử dụng, ban đầu dự án có tên là 123dapp sau này chuyển thành Tinkercad và phát triển thành một cộng đồng STEAM thiết kế 3D, sản phẩm sáng tạo cho trẻ em và những người không chuyên. <https://www.tinkercad.com/>

39. **Blender :** Phần mềm này được sử dụng rất phổ biến để thiết kế ý tưởng, nhân vật hoạt hình 3D, lập trình viên trò chơi, ứng dụng. <https://www.blender.org/>

40. **Shapeways:** trang web cung cấp dịch vụ in 3D, bạn chỉ cần thiết kế, gửi mẫu, shapeways sẽ gửi sản phẩm cho bạn. Đây là trang web khá phổ biến cho cộng đồng maker và DIY ở nước ngoài. <https://www.shapeways.com/>

41. **Ponoko:** trang web cung cấp dịch vụ chế tác, khoan, cắt, phay, in 3D để tạo ra bất cứ thứ gì bạn tạo ra. "personal factory". Dịch vụ tại đây hỗ trợ nhiều vật liệu khác nhau từ gỗ, mica, kim loại, gốm hay nhựa in 3D. <https://www.ponoko.com/>

42.

Phần 7: Robot và định hướng nghề nghiệp

Robot là sự tổng hợp của 4 thời kỳ cách mạng công nghiệp, sau 4 cuộc cách mạng này, cuối thế kỷ 21 chính là thời kỳ của robot. Để làm được một chú robot hoàn chỉnh dễ dàng tương tác với con người thì cần rất nhiều kỹ sư có những kỹ năng khác nhau. Đây cũng là các nghề nghiệp đang rất thiếu trên thị trường lao động hiện nay. Nhóm kỹ sư trí tuệ nhân tạo, nhóm kỹ sư phần mềm, nhóm kỹ sư điện tử, nhóm kỹ sư cơ khí. Trong 4 nhóm chính đó chúng ta lại phân chia tiếp ra các công việc cụ thể khác nhau. Trong tương lai gần nhóm kỹ sư trí tuệ nhân tạo sẽ bổ sung thêm nhóm chuyên gia từ cơ khí và điện-điện tử lên, hiện nay cá nhân tôi vẫn xếp ở mục cụ thể thì phù hợp. Hiện nay thì hai nhóm kỹ sư A.I và nhóm kỹ sư phần mềm đang có nhu cầu nhiều nhất, đến thời kỳ robot bùng nổ thì 2 nhóm kỹ sư tiếp theo sẽ cần thêm nhiều hơn.

Kỹ sư trí tuệ nhân tạo – A.I

- 1.** Đây là nhóm kỹ sư đang có mức lương cao nhất hiện nay do nhu cầu các công ty trong thời kỳ này đang tăng. Nhóm kỹ sư và chuyên gia này không chỉ ứng dụng trong robot mà sẽ cần trong mọi ngành nghề của cuộc sống. Robot cần nhìn, cần hiểu môi trường xung quanh có gì để di chuyển và ra quyết định. Robot cần nghe thấy mọi người nói gì để phản hồi cho phù hợp. Robot cần hiểu nhu cầu con người qua những lời nói và hành động. Đó là những nhiệm vụ mà nhóm kỹ sư này phải làm.
- 2.** Chuyên gia nghiên cứu về trí tuệ nhân tạo: giải toán, kiên trì, thích làm việc với các công thức. Các kỹ sư này sẽ đi sâu nghiên cứu về các thuật toán cho từng bài toán rất cụ thể trong lĩnh vực như nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên, nhận dạng, xử lý tiếng nói, ...Nhóm này sẽ đưa ra các thuật toán đã được chứng minh trên các mô hình và tập

dữ liệu cụ thể để nhóm kỹ sư khác ứng dụng đưa vào sản phẩm. Ví dụ thuật toán nhận dạng khuôn mặt, thuật toán phân loại nam, nữ, thuật toán nhận dạng chữ viết tay,...

- 3.** Kỹ sư phát triển ứng dụng trí tuệ nhân tạo: nhóm này thích ứng dụng, thích học toán, dùng những thuật toán của nhóm trên để đưa vào các sản phẩm cụ thể, hay hệ thống dịch vụ cho sản phẩm. Nhóm này cần hiểu và đọc được các công thức và thành thạo các ngôn ngữ lập trình, nền tảng phù hợp cho A.I để tối ưu và đưa vào thực tế.
- 4.** Chuyên gia xây dựng dữ liệu cho trí tuệ nhân tạo: Quá trình làm của nhóm 1 chỉ chứng minh trên một tập dữ liệu nhỏ, cụ thể. Khi đưa sản phẩm vào thực tế chúng ta cần rất nhiều dữ liệu, để phần mềm A.I thông minh thì cũng cần học và chuẩn hóa các dữ liệu cho máy có thể hiển thị được.
- 5.** Phân tích hành vi người dùng và kiểm thử ứng dụng trí tuệ nhân tạo: Ngày nay con người mới bắt đầu làm quen với các hệ thống A.I Để các sản phẩm này tương tác tốt trong các ứng dụng, phải có nhóm chuyên phân tích các phản ứng của con người trước khi đưa ứng dụng ra thị trường. Nhóm này sẽ đảm bảo kiểm tra các phản ứng của người dùng sẽ là phù hợp để các ứng dụng A.I có thể phản hồi

Kỹ sư phần mềm

Đây là nhóm kỹ sư phát triển các ứng dụng dịch vụ trên các nền tảng hệ điều hành, nền tảng Cloud và trình duyệt web như các hệ điều hành (Window, Mac, Linux), nền tảng Cloud (AWS, Azure, Google Cloud), trình duyệt (Chrome, Edge, Cốc Cốc). Các nền tảng hay hệ điều hành này đều có những công cụ hay bộ phát triển, hướng dẫn cụ thể để các lập trình viên xây dựng ứng dụng. Robot cần nhóm kỹ sư này để thiết kế hệ thống ứng dụng trực tiếp trên robot, trên cloud cũng như các nền tảng khác có thể tương tác với con người.

- 1.** Kỹ sư phát triển phần mềm dịch vụ trên cloud, server – Backend Dev: phát triển các ứng dụng phía sau các bạn frontend để cho ứng dụng dịch vụ web hoạt động. Các bạn có tính logic cao sẽ phù hợp với nhóm kỹ sư này.

- 2.** Kỹ sư phát triển giao diện trên nền tảng web, dịch vụ online – Fontend Dev: tạo ra các giao diện các trang web chúng ta hay sử dụng, phù hợp các bạn thích cái đẹp, biết dùng các công cụ thiết kế, biết chọn màu, bố cục phù hợp và xây dựng giao diện từ các ngôn ngữ nền tảng cơ bản.
- 3.** Kỹ sư phát triển ứng dụng trên điện thoại di động Mobile Dev: nhóm kỹ sư này sẽ phát triển phần mềm trên các nền tảng điện thoại như Android hay iOS, hiện nay có những nền tảng ngôn ngữ lập trình nhanh phù hợp nhiều nền tảng. Nhóm này cũng sẽ nhiều bạn đi chuyên vào lĩnh vực thiết kế lập trình game trên mobile, trên màn hình của robot.
- 4.** Kỹ sư phát triển ứng dụng VUI – tương tác bằng giọng nói. Voice User Interface, các ứng dụng ngày nay của Alexa – Amazon hay Google Home đang bùng nổ trên thế giới. Nhưng ứng dụng không cần màn hình, không cần giao diện. Chỉ cần nói: "Alexa, gọi cho tôi cái pizza." Đang sau đó là những ứng dụng VUI được kết nối với những dịch vụ cloud trí tuệ nhân tạo. Đó chính là nhiệm vụ của nhóm kỹ sư này khi xây dựng ứng dụng tương tác cho robot. Đây có thể nói sẽ là cách tương tác chính
- 5.** Chuyên gia kiểm thử phần mềm – Tester: Những bạn này sẽ đảm bảo là những sản phẩm chạy tốt không bị lỗi. Trước khi ra mắt sản phẩm, các chức năng đều được test rất kỹ để tránh sự cố khi người dùng sử dụng. Công việc này phù hợp cho các bạn thích sử dụng phần mềm, nhưng cần có tính chi tiết, cẩn thận.
- 6.** Kỹ sư tích hợp quản trị hệ thống Ops Dev: Ngày nay khi các dịch vụ cloud lênh ngói, các dịch vụ được các tập đoàn công nghệ lớn xây dựng, các đội sản phẩm sẽ giảm số người rất nhiều. OpsDev sẽ phụ trách quản lý tài nguyên, dịch vụ hệ thống server cho các sản phẩm chạy trơn tru trên các nền tảng cloud khác nhau.
- 7.** Kỹ sư đa năng Full Stack phần mềm: nhóm kỹ năng này rất được săn đón tại startup, nhóm sản phẩm mới phát triển. Các kỹ sư có khả năng làm tất cả các việc trên để giúp xây dựng sản phẩm của mình.

Kỹ sư điện-điện tử

Đây là nhóm kỹ sư có kiến thức về lập trình các thiết bị nhúng (tủ lạnh, điều hòa, tivi, xe máy, ô tô,...). Hiểu biết về các linh kiện điện tử, cảm biến, động cơ, bảng mạch - PCB, công cụ lập trình cho vi điều khiển nhúng, máy tính nhúng, hiểu các hệ điều hành nhúng thời gian thực. Nhóm này phần lớn hiện nay dùng các ngôn ngữ cơ bản như C/C++. Nhóm này đảm bảo cho robot cảm nhận với môi trường qua các giác quan khác và điều khiển để robot tương tác với môi trường. Ví dụ như chú robot bước chân lên cầu thang. Nhóm này cần xác định robot đã bước chân chưa, nền sàn cứng hay mềm, làm thế nào để bước được chân lên cầu thang.

- 1.** Chuyên gia nghiên cứu thuật toán điều khiển: động cơ là phần không thể thiết của robot. Cách điều khiển động cơ hoạt động với các môi trường khác nhau như thế nào cho phù hợp do nhóm chuyên gia này quyết định. Có rất nhiều bài toán chuyên sâu vẫn đang được nhóm này giải quyết. Nhóm này cần học toán tốt và thích làm việc với các công thức.
- 2.** Kỹ sư thiết kế mạch PCB: bất cứ sản phẩm điện tử gia dụng, hay đồ đặc cắm điện của chúng ta đều có một bảng mạch điện tử. Nhóm kỹ sư này sẽ phải thiết kế ra bảng mạch điện tử giúp gắn các linh kiện điện tử lên trước khi sản phẩm được lập trình. Chuyên sâu nhất mảng này là thiết kế những mạch nhiều lớp cho máy tính, điện thoại hay những mạch công suất lớn cho động cơ khổng lồ như máy phát điện sức gió.
- 3.** Kỹ sư lập trình ứng dụng nhúng – firm-ware: cần làm nhiều các ngôn ngữ bậc thấp như C/C++, cần am hiểu về mạch và thiết bị điện tử cũng như các chuẩn giao tiếp cơ bản của máy móc.
- 4.** Kỹ sư phát triển mid-ware – driver sản phẩm phức tạp như ô tô, tivi, điện thoại thường có một tầng mid-ware trước khi xây dựng ứng dụng. Các tầng này sẽ làm những phần mềm giao tiếp với các chức năng cụ thể như cảm biến, động cơ, led, Thường các middleware sẽ được phát triển trên các OS nổi tiếng như linux hoặc cách OS khác nhỏ hơn cho nền tảng nhúng.

- 5.** Kỹ sư vận hành máy móc hệ thống tự động, robot: am hiểu về kỹ thuật điện cơ bản, hiểu biết để vận hành sản phẩm, khi có sự cố có thể thay thế một số chi tiết dựa trên các bản vẽ kỹ thuật.
- 6.** Kỹ sư full-stack điện : nắm được tất cả các nhóm kỹ năng trên và tự thiết kế lập trình được một sản phẩm hoàn chỉnh cho từng nhu cầu.

Kỹ sư cơ khí

Nhóm kỹ sư này cần giỏi sử dụng các phần mềm vẽ 3D, hiểu về kết cấu, các loại vật liệu, khả năng tính toán mô phỏng sản phẩm, hiểu biết về các máy công cụ. Hiện nay nhóm kỹ sư này tham gia thiết kế vỏ các sản phẩm trong cuộc sống như tivi, điều hòa, máy giặt, máy in, ô tô, tủ lạnh, ... Các bộ phận bên ngoài, khung xương của các thiết bị đều được nhóm kỹ sư này chế tạo và sản xuất. Robot phức tạp hơn các chi tiết cố định như đồ gia dụng, một con robot thì nhiều chi tiết hơn, người làm cơ khí phải tưởng tượng khi hoạt động thực tế sẽ như thế nào, có vấn đề gì xảy ra. Thiết kế có hài hòa về kiểu dáng và trọng tâm. Mô phỏng giả lập các tương tác đó trong các môi trường ảo trước khi đưa ra bên ngoài.

- 1.** Chuyên gia nghiên cứu mô phỏng, động học: Giỏi về toán, thích hình học không gian, vật lý, nhóm kỹ sư này giúp xây dựng các hệ tính toán hệ thống mô phỏng giả lập robot trước khi sản xuất, trong quá trình thiết kế.
- 2.** Kỹ sư thiết kế sản phẩm: Có khả năng tưởng tượng tốt, thích hình không gian, dùng tốt các công cụ vẽ 3D, thích sáng tạo. Vẽ và thiết kế các chi tiết của sản phẩm và robot
- 3.** Kỹ sư mô phỏng hệ thống: dựa vào các thuật toán của các chuyên gia nghiên cứu ứng dụng tính toán cho từng sản phẩm.
- 4.** Kỹ sư chế tạo gia công sản phẩm: Hiểu được các máy công cụ, biến các thiết kế thành các bản vẽ có thể sản xuất hàng loạt được.

- 5.** Kỹ sư vận hành cách máy công cụ: nắm được kiến thức chung, hiểu được nguyên lý hoạt động của máy, có thể xử lý sự cố khi robot hoặc thiết bị có vấn đề.
- 6.** Kỹ sư cơ khí full-stack: nắm được tất cả các nhiệm vụ trên, tự làm và thiết kế cũng như sản xuất. Trong tương lai khi các công cụ hỗ trợ cơ khí phát triển thì nhóm kỹ sư này sẽ cần rất nhiều.

Quản lý công nghệ

Nhóm này được phát triển từ những kỹ sư full-stack trong các lĩnh vực trên nhưng có góc nhìn tổng quan, định hướng, có sự am hiểu hệ thống, có khả năng tương tác, giao tiếp tốt với mọi người. Đối với người làm quản lý công nghệ trong robot có thể cần biết tới 4 nhóm kỹ năng trên so với người làm quản lý công nghệ thông thường hiện nay.

- 1.** Kiến trúc sư hệ thống - Software/Hardware Architect: những người làm nhiều các sản phẩm, có cái nhìn tổng quan cho từng bài toán, đưa ra các giải pháp công nghệ cho từng bài toán. Từ đó thành bản thiết kế để nhóm các kỹ sư cùng bắt tay vào làm.
- 2.** Quản lý sản phẩm – Product Manager: nhóm người này có cái nhìn tổng quan về hệ thống như một kiến trúc sư, nhưng lại có góc nhìn chiến lược cho sản phẩm, hiểu thị trường và khách hàng để định hướng phát triển cho sản phẩm. Nhóm này cần kỹ năng giao tiếp tốt để làm việc với khách hàng, lãnh đạo và đội ngũ kỹ sư.
- 3.** Giám đốc công nghệ - CTO: Nhưng người có định hướng công nghệ rất tốt, đi lên từ những người quản lý sản phẩm công nghệ. Vị trí này giúp định hướng công nghệ cho một công ty, một tập đoàn phù hợp với năng lượng, thị trường và con người của từng doanh nghiệp. Đây là mức cao nhất có một chuyên gia công nghệ và có kỹ năng mềm phù hợp.

Trong tương lai gần các công cụ càng phát triển, quy trình phát triển phần mềm, phần cứng ngày càng trở nên đơn giản. Sẽ có thời điểm tất cả các ngành đều có những nền tảng để chung ta phát triển sản phẩm rất nhanh. Nhu cầu xuất hiện các kỹ sư full-stack am hiểu phần cứng, phần

mềm, thậm chí full-stack trong 4 nhóm kỹ sư để phát triển nhanh các sản phẩm. Kỹ sư robot là nhóm kỹ sư full-stack đó, các bạn sẽ phải nắm được 4 kỹ năng của 4 cuộc cách mạng công nghệ để xây dựng cho mình những chú robot cho tương lai.

Index

2.

3D Printing, 154

A.I., 113

Adafruit, 155

Analyze, 142

Arduino, 137, 155

ASR, 116

AutoCad, 20

AutoDesk, 20

Blender, 157

Blocky, 156

bread-broad, 33

Cloud, 142

Computer Vision, 116

Datasheet, 41

DIY, 15

Driver, 13

encoder, 13

entity, 117

Fablab, 153

intent, 117

Inventor, 21

IoT, 118

Lilypad, 155

loa thông minh, 117

Make Magazine;, 153

MakeCode, 156

Maker Hanoi, 154

Makerbot, 154

Makers Faire;, 153

MaKey Makey, 155

máy in 3D, 25

Máy in 3D, 131

microcontroler, 13

MIT App Inventor, 156

NLP/NLU, 117

PCB, 42

Mạch in, 42

Pi Zero, 138

Ponoko, 157

Raspberry Pi, 155

Robot, 120

Scratch, 156

Seeedstudio, 155

Shapeways, 157

SketchUp, 157

Social, 142

SolidWork, 21

Sparkfun, 155

Thingiverse, 25, 157

Tinkercad, 157

TinkerCad, 20

Tinkering Studio, 154

Trí tuệ nhân tạo. A.I

TTS, 116

3.