**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**Bùi Anh Khoa**

**KHÓA LUẬN TỐT NGHIỆP**

**MÔ HÌNH MÔ PHỎNG XE TỰ HÀNH**

**Automatic car simulation**

**TP. HỒ CHÍ MINH, 2019**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**Bùi Anh Khoa – 15520364**

**KHÓA LUẬN TỐT NGHIỆP**

**MÔ HÌNH MÔ PHỎNG XE TỰ HÀNH**

**Automatic car simulation**

**GIẢNG VIÊN HƯỚNG DẪN**

**Ths.**

**TP. HỒ CHÍ MINH, 2019**

DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số…………………… ngày ………………….. của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

* 1. …………………………………………. – Chủ tịch.
  2. ………………………………………… – Thư ký.
  3. …………………………………………. – Ủy viên.

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |
|  | *TP. HCM, ngày…..tháng…..năm……..* |

# NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP

**(CỦA CÁN BỘ HƯỚNG DẪN)**

**MÔ HÌNH MÔ PHỎNG XE TỰ HÀNH**

**Nhóm sinh viên thực hiện:** **Cán bộ hướng dẫn:**

Bùi Anh Khoa 15520364 ThS.

**Đánh giá khóa luận**

1. Về cuốn báo cáo:

1. Về nội dung nghiên cứu:

1. Về chương trình ứng dụng:

1. Về thái độ làm việc của sinh viên:

Đánh giá chung:

Điểm từng sinh viên:

Bùi Anh Khoa: ……/10

|  |  |
| --- | --- |
|  | **Người nhận xét** |

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |
|  | *TP. HCM, ngày…..tháng…..năm……..* |

# NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP

**(CỦA CÁN BỘ PHẢN BIỆN)**

**MÔ HÌNH MÔ PHỎNG XE TỰ HÀNH**

**Nhóm sinh viên thực hiện:** **Cán bộ phản biện:**

Bùi Anh Khoa 15520364

**Đánh giá khóa luận**

1. Về cuốn báo cáo:

1. Về nội dung nghiên cứu:

1. Về chương trình ứng dụng:

1. Về thái độ làm việc của sinh viên:

Đánh giá chung:

Điểm từng sinh viên:

Bùi Anh Khoa: ……/10

|  |  |
| --- | --- |
|  | **Người nhận xét** |

# LỜI CẢM ƠN

Để hoàn thành được khóa luận tốt nghiệp này, chúng em xin chân thành gửi lời cảm ơn đến quý thầy cô Trường Đại học Công nghệ Thông tin – Đại học Quốc gia Thành phố Hồ Chí Minh nói chung và quý thầy cô Khoa Kỹ thuật Máy tính nói riêng đã truyền đạt cho chúng em kiến thức và những kinh nghiệm quý báu trong suốt chặng đường 5 năm học vừa qua.

Xin gửi lời cảm ơn chân thành đến thầy Ths. , người đã dành những thời gian quý báu và kinh nghiệm của mình, trực tiếp hướng dẫn tận tình cho chúng em hoàn thành khóa luận tốt nghiệp.

Xin chân thành cảm ơn sự giúp đỡ của các bạn bè, anh chị, những người đã giúp đỡ chúng em tìm kiếm, thu thập thông tin trong suốt quá trình thực hiện khóa luận.

Một lần nữa xin chân thành cảm ơn đến tất cả những người đã giành thời gian, công sức giúp đỡ chúng em hoàn thành khóa luận tốt nghiệp. Trong quá trình thực hiện khóa luận, tất nhiên không tránh khỏi những sai lầm, thiếu sót, chúng em mong quý thầy cô, các bạn, anh chị bỏ qua và thứ lỗi.

Sinh viên thực hiện

**Bùi Anh Khoa**

Khoa Kỹ thuật Máy tính. Lớp KTMT2015

|  |  |
| --- | --- |
| ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  **TRƯỜNG ĐẠI HỌC**  **CÔNG NGHỆ THÔNG TIN** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc Lập - Tự Do - Hạnh Phúc** |
|  | *TP. HCM, ngày…..tháng…..năm……..* |

# ĐỀ CƯƠNG CHI TIẾT

**MÔ HÌNH MÔ PHỎNG XE TỰ HÀNH**

**Cán bộ hướng dẫn:** ThS.

**Thời gian thực hiện:** Từ ngày xx/xx/xxxx đến ngày xx/xx/xxxx

**Sinh viên thực hiện:**

Bùi Anh Khoa – 15520364

**Nội dung đề tài:**

* Lý do chọn đề tài:

Nắm được tầm quan trọng của tự động hóa trong đời sống và xe tự hành đang là một trong những để tài nóng hổi của thế giới với sự tham gia của nhiều hãng lớn cho nên nhóm đã quyết định theo hướng này để đón đầu công nghệ xử lý ảnh, máy học, AI, và board nhúng Nvida Jetson

* Mục tiêu của đề tài:

Nghiên cứu, thiết kế và xây dựng mô hình xe tự hành với khả năng đi vào đúng làn đường của mình và tuân thủ theo những biển báo.

* Nội dung chính:

Nghiên cứu xử lý nhận diện làn đường.

Tích hợp kết quả vào tính toán tốc độ và góc bẻ lái của xe RC.

Nghiên cứu xử lý phát hiện và nhận diện biển báo với phương pháp deep learning và machine learning.

Tích hợp kết quả vào nhận hiệu lệnh từ biển báo để điều khiển xe.

Nghiên cứu sử dụng physical của board Nvidia Jetson thông qua hệ điều hành Linux Ubuntu trên Arm

* Kết quả mong đợi:
* Nghiên cứu và áp dụng thành công xử lý ảnh, deep learning, machine learning và physical vào board Nvidia Jetson. Điều khiển xe RC chạy ổn định trên nhiều loại đường và biển báo, làm tiền đề để tương lai ứng dụng trên xe golf, xe ô tô.
* **Kế hoạch thực hiện:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tuần** | **Nội dung** | **Công việc** | **Người thực hiện** |
| **1** (14/08/2018 - 20/08/2018) | Tìm hiểu tổng quan về cảm biến cong và phương thức sử dụng | Đọc datasheet cũng như cách thức hoạt động trên trang sản xuất cung cấp.  Nghiên cứu sample và công thức tính toán | Hồ Mai Kim Chi, Nguyễn Minh Trí |
| **2** (21/08/2018 - 27/08/2018) | Tìm hiểu tổng quan về hệ thống tay mô phỏng có mặt trên thế giới và phương thức vận hành | Xem xét cấu tạo bộ cơ vận hành của hệ thống tay.  Xem xét tổng quan về thiết kế hệ thống và các mô hình thiết kế trên mạng |
| **3** (28/08/2018 - 03/09/2018) | Tìm hiểu STM32F103, module RF CC1101 và servo MG996R  Xác định cụ thể mô hình thiết kế 3D sẽ sử dụng cho cánh tay mô phỏng và momen lực của cánh tay | Tìm hiểu datasheet và nguyên lý hoạt động cũng như giao tiếp của STM32F103, module RF CC1101 và servo MG996R.  Xem xét các hàm và tác dụng lực cho cánh tay mô phỏng |
| **4** (04/09/2018 - 10/09/2018) |
| **5** (11/09/2018 - 17/09/2018) |
| **6** (18/09/2018 - 24/09/2018) | Tiến hành in ấn thiết kế 3D để áp dụng thử hệ thống | Xác định hệ thống ổn định và đạt đủ điều kiện cho yêu cầu | Hồ Mai Kim Chi |
| **7** (25/09/2018 - 01/10/2018) | Lắp ráp hệ thống | Hoàn thiện các board mạch điều khiển bao gồm mạch nguồn và mạch lắp ráp MCU và gắn servo | Hồ Mai Kim Chi |
| **8** (01/10/2018 - 04/10/2018) | Hoàn thiện mô hình bao gồm các board mạch ngoại vi và mô hình thiết kế 3D. | Hồ Mai Kim Chi, Nguyễn Minh Trí |
| **9** (04/10/2018 - 10/10/2018) | Thiết kế, xây dựng mô hình cánh tay thật, bắt đầu tiến hành lấy mẫu và xử lý mẫu dữ liệu từ Flex sensor | Hoàn thiện mạch gắn Flex sensor và lấy mẫu thô từ flex sensor | Hồ Mai Kim Chi |
| **10** (10/10/2018 - 15/10/2018) | Phân tích, chia nhóm và hoàn thiện mẫu dữ liệu để thực hiện gửi lệnh điều khiển cho cánh tay mô phỏng | Nguyễn Minh Trí |
| **11** (15/10/2018 - 21/10/2018) | Thực hiện lọc nhiễu cho flex sensor và xử lý cử động cho mô hình cánh tay mô phỏng | Nghiên cứu bộ lọc thônh thấp và sử dụng thuật toán để thực hiện lọc nhiễu cho flex sensor | Nguyễn Minh Trí |
| **12** (21/10/2018 - 27/10/2018) | Tiến hành tạo xung và điều khiển servo cho cánh tay giả để thực hiện co duỗi các ngón tay theo nhiều nấc | Hồ Mai Kim Chi |
| **13** (27/10/2018 - 05/11/2018) | Hoàn thiện giao tiếp truyền nhận tín hiệu điều khiển giữa thiết bị có gắn flex sensor và cánh tay giả | Thực hiện giao tiếp với module RF CC1101 và truyện nhận dữ liệu cho 2 bên | Hồ Mai Kim Chi |
| **14** (6/11/2018 - 11/11/2018) | Hoàn thiện giao tiếp và phương thức mã hóa cũng như bắt tay cho tín hiệu điều khiển giữa thiết bị gắn flex sensor và cánh tay mô phỏng | Nguyễn Minh Trí |
| **15** (11/11/2018 - 16/12/2018) | Thiết kế mô hình cánh tay mới phù hợp hơn. Thay thế thiết bị HM10 có tính ổn định và hiệu năng cao hơn. Sử dụng bộ nguồn di động cho cánh tay để nó linh hoạt hơn. | Nghiên cứu datasheet và phương thức hoạt động của HM10.  Thiết kế mạch nguồn mới, sử dụng nguồn từ pin Lipo để giúp cánh tay trở nên linh hoạt hơn | Hồ Mai Kim Chi |
| **16** (16/11/2018 - 22/11/2018) |  | Tiến hành thiết kế bàn tay mới linh hoạt hơn cho cánh tay mô phỏng | Nguyễn Minh Trí |
| **17**  (23/11/2018 - 26/11/2018) | Báo cáo tiến độ giữa kỳ (23/11/2018) | | |
| **18**  (26/11/2018 - 29/12/2018) | Nghiên cứu thực nghiệm so sánh dữ liệu và hiệu năng với việc điều khiển cánh tay mô phỏng giữa flex sensor và cảm biến MPU6050 | Nghiên cứu datasheet và phương thức hoạt động của MPU6050. | Nguyễn Minh Trí |
| **19**  (29/12/2018 - 05/01/2019) | Lắp ráp MPU6050 vào thiết bị điều khiển để lấy số liệu thô các trục | Hồ Mai Kim Chi |
| **20 - 22**  (05/01/2019 - 12/01/2019) | Tiến hành phân tích, ghi chép mẫu số liệu và so sánh hiệu năng của cả 2 bên | Sử dụng các công cụ ghi chép và mô phỏng để so sánh các dữ liệu trả về từ cả 2 loại cảm biến | Hồ Mai Kim Chi, Nguyễn Minh Trí |
| **22 - 23**  12/01/2019 - 16/01/2019) | Hoàn chỉnh nội dung báo cáo, quay video demo. | | Hồ Mai Kim Chi, Nguyễn Minh Trí |

|  |  |
| --- | --- |
| **Xác nhận của CBHD** | **TP. HCM, ngày….tháng …..năm…..**  **Sinh viên** |

# MỤC LỤC

[NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP 4](#_Toc26203378)

[NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP 5](#_Toc26203379)

[LỜI CẢM ƠN 6](#_Toc26203380)

[ĐỀ CƯƠNG CHI TIẾT 7](#_Toc26203381)

[MỤC LỤC 13](#_Toc26203382)

[CHƯƠNG 1. TỔNG QUAN 19](#_Toc26203383)

[1.1 Tình hình trong nước 19](#_Toc26203384)

[1.2 Tình hình ngoài nước 19](#_Toc26203385)

[1.3 Mục tiêu nghiên cứu 22](#_Toc26203386)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 23](#_Toc26203387)

[2.1 Mô hình xe RGT ROCK HAMMER 23](#_Toc26203388)

[2.2 Nghiên cứu cấu trúc của những bộ phận trên xe 25](#_Toc26203389)

[2.3 Tìm hiểu thuật toán tìm đường đi sử dụng thư viện OpenCV 33](#_Toc26203390)

[2.3.1 Nhận diện màu sắc 33](#_Toc26203391)

[2.3.1.1 Hệ màu [11] 33](#_Toc26203392)

[2.3.2 Lọc màu đường và biển báo 37](#_Toc26203393)

[2.3.3 Thuật toán tìm góc lệch 37](#_Toc26203394)

[2.3.3.1 Cân chỉnh camera [12] 37](#_Toc26203395)

[2.3.3.2 Bẻ thẳng frame ảnh 38](#_Toc26203396)

[2.3.3.3 Biểu đồ những điểm thuộc làn đường 40](#_Toc26203397)

[2.3.3.4 Chia khung làn đường (Sliding window) 42](#_Toc26203398)

[2.3.3.5 Phương trình đường cong 43](#_Toc26203399)

[2.3.3.6 Tính toán dự đoán góc bẻ lái 44](#_Toc26203400)

[2.3.3.7 Tính toán khoảng cách lệch so với 2 bên làn đường 45](#_Toc26203401)

[2.3.3.8 Lưu lịch sử cho những frame kế tiếp để dự đoán 45](#_Toc26203402)

[2.3.4 Thuật toán tìm biển báo 45](#_Toc26203403)

[2.3.4.1 Nhận diện biển báo 45](#_Toc26203404)

[2.3.4.2 Những thành phần trong mạng neural 48](#_Toc26203405)

[2.3.4.3 Huấn luyện và Dự đoán 52](#_Toc26203406)

[2.4 Lập trình đa luồng 53](#_Toc26203407)

[2.4.1 Luồng (thread). Khác nhau giữa thread và process (tiến trình) 53](#_Toc26203408)

[2.4.2 Đa luồng (Multithreading) 54](#_Toc26203409)

[2.4.3 Đồng bộ hóa các thread trong python 55](#_Toc26203410)

[2.4.4 Ứng dụng đa luồng 55](#_Toc26203411)

[2.5 Board Nvidia Jetson Tx2 56](#_Toc26203412)

[2.5.1 Thông số kỹ thuật 56](#_Toc26203413)

[2.5.1.1 JETSON TX2 MODULE 56](#_Toc26203414)

[2.5.1.2 IO 56](#_Toc26203415)

[2.5.1.3 Power Option 56](#_Toc26203416)

[2.5.1.4 Jetson Camera Module 56](#_Toc26203417)

[2.5.1.5 Buttons 56](#_Toc26203418)

[CHƯƠNG 3. XÂY DỰNG MÔ HÌNH VÀ GIẢI PHÁP 57](#_Toc26203419)

[3.1 Mô hình xe với RGT Rock Hammer và Nvidia Jetson Tx2 Development board 57](#_Toc26203420)

[CHƯƠNG 4. KẾT QUẢ 58](#_Toc26203421)

[CHƯƠNG 5. TỔNG QUAN 59](#_Toc26203422)

[TÀI LIỆU THAM KHẢO 60](#_Toc26203423)

[PHỤ LỤC 62](#_Toc26203424)

**DANH MỤC HÌNH VẼ**

[Hình 1. Xe tự lái của Google [2] 18](#_Toc10407017)

[Hình 2. Công nghệ bên trong xe tự lái Tesla [3] 19](#_Toc10407018)

[Hình 3. Thiết kế bên trong của xe RGT Rock Hammer 22](#_Toc10407019)

[Hình 4. Thiết kế toàn thể của xe RGT Rock Hammer 23](#_Toc10407020)

[Hình 5. Minh họa hệ thống dẫn động 4 bánh 24](#_Toc10407021)

[Hình 6. Các loại hệ thống truyền động 4 bánh 25](#_Toc10407022)

[Hình 7. ESC của xe RGT Rock Hammer 26](#_Toc10407023)

[Hình 8. Động cơ có chổi than 27](#_Toc10407024)

[Hình 9. Động cơ không chổi than 28](#_Toc10407025)

[Hình 10. Tín hiệu điều khiển động cơ servo 29](#_Toc10407026)

[Hình 11. Phản hồi hệ thống động cơ 29](#_Toc10407027)

[Hình 12. Mạch vòng điều khiển động cơ servo 30](#_Toc10407028)

[Hình 13. Pin Lipo 30](#_Toc10407029)

[Hình 14. Phân biệt giữa 3 kênh R, G, B 33](#_Toc10407030)

[Hình 15. Phân biệt giữa 3 kênh L, A, B 34](#_Toc10407031)

[Hình 16. Phân biệt giữa 3 kênh H, S, V 34](#_Toc10407032)

[Hình 17. Vòng tròn hệ màu HSV 35](#_Toc10407033)

[Hình 18. Không gian 3 chiều của không gian màu HSV 35](#_Toc10407034)

[Hình 19. Ví dụ về radial distortion 36](#_Toc10407035)

[Hình 20. Cân chỉnh frame ảnh 37](#_Toc10407036)

[Hình 21. Kết quả sau khi cân chỉnh 37](#_Toc10407037)

[Hình 22. Mô hình về birdview 38](#_Toc10407038)

[Hình 23. Birdview với lane đường 39](#_Toc10407039)

[Hình 24. Ví dụ về histogram [14] 39](#_Toc10407040)

[Hình 25. Histogram của lane đường 40](#_Toc10407041)

[Hình 26. Flowchart sliding window 41](#_Toc10407042)

[Hình 27. Sliding window 42](#_Toc10407043)

[Hình 28. Skip sliding window 43](#_Toc10407044)

[Hình 29. Tính góc 43](#_Toc10407045)

[Hình 30. Kết quả tính độ lệch 44](#_Toc10407046)

[Hình 31. Chuyển ảnh màu thành xám 45](#_Toc10407047)

[Hình 32. Ví dụ giải thích Gaussian blur bằng toán học 45](#_Toc10407048)

[Hình 33. Gaussian blur 46](#_Toc10407049)

[Hình 34. Lọc màu xanh 46](#_Toc10407050)

[Hình 35. Threshold [15] 47](#_Toc10407051)

[Hình 36. Contour 47](#_Toc10407052)

[Hình 37. Các bước tạo ra map 2D 47](#_Toc10407053)

[Hình 38. Ví dụ về Conv2d [17] 48](#_Toc10407054)

[Hình 39. Ví dụ MaxPooling [18] 48](#_Toc10407055)

[Hình 40. Ví dụ MaxPooling trong thực tế [18] 49](#_Toc10407056)

[Hình 41. Quá trình biến đổi của flatten [19] 49](#_Toc10407057)

[Hình 42. Ví dụ Zero padding [20] 50](#_Toc10407058)

[Hình 43. Ví dụ Valid Padding [20] 51](#_Toc10407059)

TÓM TẮT KHÓA LUẬN

Cuộc cách mạng về khoa học công nghệ diễn ra từng ngày đang làm thay đổi toàn diện và sâu sắc cuộc sống cũng như quá trình sản xuất của con người. Trong xu thế phát triển đó, nền công nghệ non trẻ của Việt Nam, vốn đã tụt hậu khá xa, nếu không muốn tụt lại phía sau hơn nữa cần phải vừa tranh thủ thành tựu của các nước tiên tiến, vừa phải có sự đột phá, sáng tạo trong tư duy để theo kịp với công nghệ thế giới. Nhắc đến tự động hóa trong ngành ô tô, chắc hẳn ai cũng nghĩ tới xe tự hành.

Để ứng dụng mô phỏng cho xe tự hành, hãng Nvida đã cho ra board mạch Nvidia Jetson chuyên dùng cho xử lý ảnh, deep learning và machine learning. Board này sẽ được gắn trên những mô hình xe điều khiển cỡ lớn và tự động hoàn thành đường đi mà không có bất kì can thiệp nào từ con người trong lúc hoạt động. Hiện tại, khá ít công nghệ được public cho mảng này trên thế giới, vì thế hầu hết mọi người sẽ bắt đầu từ việc mô phỏng trong môi trường thực nghiệm và sau đó sẽ thử nghiệm trong môi trường thực tế và phát triển lên.

# TỔNG QUAN

## Tình hình trong nước

* Hiện tại, ở Việt Nam cũng đã có nhiều công ty như FPT, Ikorn, Bosch, … đang có xu hướng nghiên cứu và phát triển xe tự hành. Mô hình nghiên cứu được triển khai và có tính lan tỏa mạnh, thu hút nhiều sự quan tâm của giới nghiên cứu khoa học công nghệ, với ước vọng tạo ra và ứng dụng xe tự hành trong tương lai ở đời sống xã hội Việt Nam [1].
* Các nghiên cứu xe tự hành tại Việt Nam hiện nay đang tiến lên xe thực tế từ mô hình xe tự hành. Những mô hình demo này sử dụng cơ chế ngoại vi như servo để bẻ bánh lái và các động cơ điện 1 chiều để mô phỏng xe tự hành ngoài đời thực. Ứng dụng đang hiện thực hóa trên board chuyên biệt cho việc xử lý ảnh như board Nvidia và chỉ dừng lại ở phân đoạn nhận diện đường trắng liền, biển báo, vật cản.

## Tình hình ngoài nước

* Đã có nhiều hãng phát triển và thử nghiệm xe tự hành:
  + Google: 23 chiếc.
  + Tesla: 39 chiếc.
  + Apple: 55 chiếc.
  + …

Các hãng công nghệ lớn đã kể trên và nhiều hãng khác vẫn đang trong quá trình nghiên cứu và phát triển để ứng dụng xe tự hành vào đời sống thực tế. trong đó được tích hợp nhiều công nghệ như camera 360, stereo camera, vi xử lý, gpu tốc độ cao.



Hình 1. Xe tự lái của Google [2]



Hình 2. Công nghệ bên trong xe tự lái Tesla [3]

Xe tự hành AGV [4] bắt đầu được chế tạo để vận chuyển các phôi gia công vào những năm 70. Vấn đề định hướng của xe tự hành là một trong những vấn đề cốt lõi cần phải giải quyết. Bài toán định hướng hiện nay được phân làm 2 loại: định hướng trong nhà (indoor) và định hướng ngoài địa hình (outdoor). Quá trình định hướng của robot đi động gồm 4 bước: thu nhận cảnh quan môi trường, xác định vị trí, thiết kế quỹ đạo và tạo chuyển động. Với môi trường có cấu trúc, quá trình nhận biết cho phép tạo ra bản đồ hay mô hình không gian phục vụ cho bài toán định vị và thiết kế quỹ đạo robot. Đối với môi trường phi cấu trúc hay thay đổi, robot cần có khả năng tự học quan sát môi trường để xác định được hướng đi của mình. Do đó lĩnh vực xác định hướng đi cho robot di động là một lĩnh vực mà các phương pháp trí khôn nhân tạo  như quá trình nhận biết môi trường, suy diễn và tìm hướng đi tối ưu có thể được áp dụng. Vấn đề định vị và tạo bản đồ là những vấn đề nghiên cứu trọng tâm ở robot di động thời gian qua.  
Quá trình định vị là quá trình robot xác định được hiện nó ở đâu trong không gian hoạt động. Để đạt được mục tiêu này, cần sử dụng nhiều cảm biến thu nhận các dữ liệu liên quan đến trạng thái của robot và môi trường xung quanh. Các dữ liệu này thường bị nhiễu và có sai số tích lũy nên cần có các phương pháp lọc động và sử dụng các phương pháp phối hợp cảm biến để có được số liệu đo chính xác hơn. Phương pháp định vị  có thể  là cục bộ hay toàn cục. Giải pháp đơn giản nhất là định vị cục bộ khi robo thường xuyên cập nhật vị trí của nó so với điểm xuất phát. Ngược lại các phương pháp định vị toàn cục không đòi hỏi biết vị trí của điểm xuất phát. Để khắc phục độ bất định của các thông tin đo được từ các cảm biến ta cần sử dụng các phương pháp xác xuất. Các phương pháp định vị được sử dụng thường dựa trên nguyên lý lọc Bayes kết hợp với một thuật toán đệ quy để ước lượng được vị trí và hướng từ phương trình mô tả chuyển động của robot. Thời gian tính toán của lọc Bayes lâu nên nhiều nghiên cứu gần đây tập trung vào tìm các phương pháp đơn giản hóa để giảm khối lượng tính toán. Quá trình đơn giản hóa này dẫn đến nhiều thuật toán định vị khác nhau phân làm 2 loại tùy thuộc vào cách mô tả độ tin cậy của dữ liệu. Nếu dữ liệu được mô tả bằng các hàm phân bổ Gauss, ta có thể sử dụng phương pháp lọc Kalman. Nếu dữ liệu được mô tả bằng nhiều hàm phân bổ xác xuất khác nhau, ta có thể sử dụng các thuật toán định vị dựa trên quá trình Markov. Phương pháp định vị dựa trên phân bổ Gauss và lọc Kalman chỉ ứng dụng hiệu quả cho bài toán định vị cục bộ. Các phương pháp định vị Markov có thể là phương pháp  tô pô, phương pháp lưới và phương pháp sử dụng các mẫu rời rạc của các giá trị trạng thái. Khi các dữ liệu robot được mô tả bằng các mẫu rời rạc ngẫu nhiên ta có thể sử dụng các phương pháp lọc phần tử (particle filter) để xác định được vị trí của rô bốt tốt hơn.  
Do việc định vị và lập bản đồ cho robot có quan hệ mật thiết với nhau nên từ những năm 90 các nghiên cứu đã tập trung giải quyết hai vấn đề này đồng thời với tên chung là SLAM (Simultanous Localization And Mapping). Các nghiên cứu gần đây liên quan nhiều đến lập bản đồ cho môi trường động. Lọc Kalman có thể sử dụng được cho bài toán này nhưng không cho độ chính xác cao. Vấn đề tìm được một thuật toán lập bản đồ cho môi trường động còn là một thách thức lớn. Nhiều vấn đề còn bỏ ngỏ như phân biệt các đối tượng tĩnh, đối tượng chuyển động và mô tả chúng trên bản đồ.  
Ngoài các ứng dụng trong công nghiệp và quân sự,  robot di động cũng đã đi vào cuộc sống đời thường như robot hút bụi, lau nhà, cọ bể bơi, lau kính, robot bào tàng, robot dịch vụ văn phòng, bệnh viện và ở các nơi công cộng. Các robo này yêu cầu mức độ nhận thức nhất định, có khả năng tự định hướng, di chuyển trong môi trường biến động và có giao tiếp người-máy thân thiện. Ứng dụng robot di động trong giám sát, gác, cảnh báo về anh ninh cũng là một hướng được nhiều công ty bảo vệ an ninh sử dụng.

## Mục tiêu nghiên cứu

***Thiết kế một chiếc xe tự hành ở mức nhỏ có thể làm được những công việc sau:***

- Khả năng nhận diện và đi trên đường nét đứt màu vàng.

- Nhận diện biển báo: rẽ trái rẽ phải, dừng lại,…

***Công nghệ:***

- Nhận diện đường đi bằng OpenCV

- Nhận diện đường đi bằng máy học (Keras)

- Nhận diện biển báo bằng OpenCV

- Nhận diện biển báo bằng máy học (Keras)

- Phân loại biển báo bằng máy học (Keras)

- Hệ điều hành Linux (Ubuntu) trên board Nvidia TX2

# CƠ SỞ LÝ THUYẾT

## Mô hình xe RGT ROCK HAMMER

RGT Rock Hammer [5] là mẫu xe địa hình tỉ lệ 1/10 của hãng RGT Racing. Đây là dòng xe chuyên địa hình đầy sức mạnh, với thiết kế khung sườn kim loại bắt mắt và độ bền cao, hệ thống truyền động 4 bánh (4WD), servo nhông kim loại, bộ hộp số tạo lực kéo mạnh, bánh lớn, gầm cao giúp xe di chuyện ở những địa hình phức tạp (đá, cát, sình lầy hay cả nước...) với độ dốc lên tới 45 độ. Đây là mẫu xe phù hợp với những người thích thể loại cơ bắp, leo trèo khỏe, không cần tốc độ cao, thích chế độ các chi tiết trên xe cho giống xe thật.



Hình 3. Thiết kế bên trong của xe RGT Rock Hammer

Ngoài những thiết kế như trên, xe được bổ sung:

* Mạch được bọc kín hạn chế nước văng vào
* ESC chống nước 40A
* Servo chống nước E1501 với sức kéo 15kg
* Phuộc nhún

Thông số kỹ thuật đầy đủ:

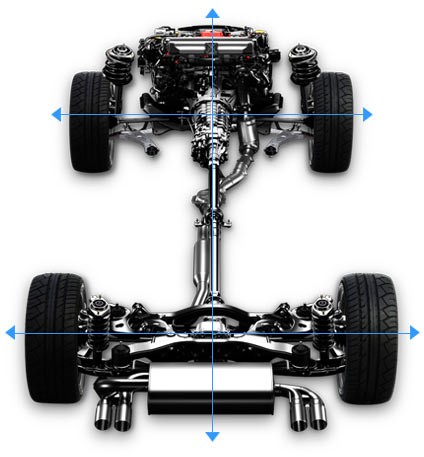
* Hãng: RGT Racing
* Tên sản phẩm: Rock Hammer
* Tỉ lệ: 1/10
* Kích thước: 470x270x215mm
* Chiều dài cơ sở: 350mm
* Khoảng sáng gầm xe: 80mm
* Bề rộng lốp: 52mm
* Đường kính lốp: 135mm
* Hex trục bánh: 12m
* Tần số điều khiển: 2.4GHz
* ESC: Hobbywing WP1040 40A, chống nước
* Motor: Brushed RC540-8020
* Servo: 15Kg
* Pin: Pin sạc 7.2V 2000 mAh
* Khoảng cách điều khiển: khoảng 100-120m
* Thời gian sạc: 3-4 tiếng
* Thời gian sử dụng: khoảng 15-20 phút
* Tốc độ: 25km/h



Hình 4. Thiết kế toàn thể của xe RGT Rock Hammer

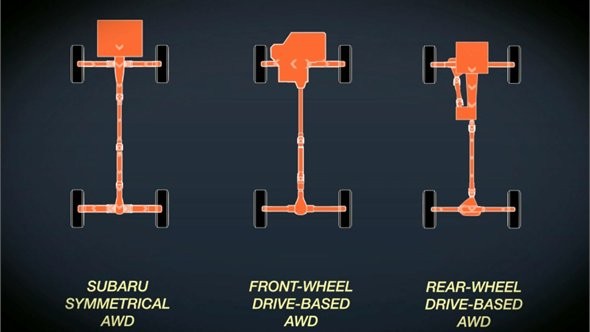
## Nghiên cứu cấu trúc của những bộ phận trên xe

* Hệ thống dẫn động 4 bánh toàn thời gian đối xứng (4WD) [6]:
  + Hệ thống dẫn động được cấu thành từ 3 thành phần: hệ truyền động, động cơ Boxer và hộp số, tất cả đều được bố trí một cách cân bằng và đối xứng qua trục dọc ở giữa thân xe



Hình 5. Minh họa hệ thống dẫn động 4 bánh

* + Sự kết hợp của cấu trúc truyền động đối xứng, động cơ Boxer dạng phẳng cùng với hộp số đặt theo trục dọc ở giữa mang đến sự phân bố trọng lượng hoàn hảo với trọng tâm của xe được hạn thấp tối đa. Về mặt vật lý, những điều này mang lại sự cân bằng tối ưu và độ đầm chắc mượt mà cho chiếc xe khi vận hành ở tốc độ cao.
  + Về mặt sức kéo: Hệ thống sẽ truyền sức kéo liên tục đến cả 4 bánh cùng một lúc, do đó chiếc xe sẽ có được lực kéo và khả năng tăng tốc tối ưu. Trong những điều kiện địa hình phức tạp hoặc có 1 bánh bị trượt, hệ thống sẽ tự động tính toán là ngắt mô-men xoắn ở bánh đó và truyền lực đến những bánh có độ bám tốt hơn, giúp xe không bị mất lái và vượt qua một cách nhanh chóng, an toàn.
  + Về mặt hiệu suất và độ bền: Hệ thống được bố trí thẳng hàng và đối xứng qua trục giữa một cách tối ưu. Nhờ vậy, nó có thể truyền lực kéo từ động cơ một cách liên tục và mượt mà đến cả 4 bánh xe. Do phải dùng động cơ với kiểu sắp xếp xy-lanh dạng chữ I hoặc chữ V, các hệ thống dẫn dộng 4 bánh AWD của những hãng khác phải có thêm nhiều thành phần dẫn động khác, tất nhiên là nó sẽ phức tạp hơn, trọng lượng nặng hơn và chi phí bảo trì cao hơn



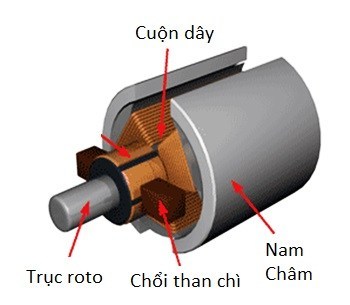
Hình 6. Các loại hệ thống truyền động 4 bánh

* Bộ điều khiên ESC (Electronic Speed Control) [7]
  + Trong ESC, bộ phận giới hạn điện áp cho pin lipo có tên là LVC (Low Voltage Cut-Off)
  + BEC: BEC là bộ nguồn ổn áp tạo điện áp làm việc (5 volt) cho vi điều khiển nằm trong ESC và cung cấp luôn cho RX cũng như các servo... BEC đi kèm ESC thường là loại ổn áp bù nối tiếp nên rất nóng (điện áp pin càng cao thì càng nóng) và rất hao pin, vì vậy thường người ta ko dùng BEC trong ESC mà dùng BEC rời gọi là UBEC
  + BEC: là nguồn ổn áp dạng đóng ngắt, sử dụng PWM (kỹ thuật điều biến bề rộng xung) để biến đổi điện thế DC to DC (1 chiều sang 1 chiều). Thường thì UBEC sử dụng dạng mạch step down để hạ áp (thường điện áp pin lớn hon điện áp làm việc).
  + PWM: kỹ thuật điều biến bề rộng xung, đây là phần nguyên lý của ESC, kỹ thuật này giúp ta điều chỉnh công suất cấp cho động cơ thông qua độ rộng xung. Giả dụ ta gắn 1 biến trở để điều khiển vận tốc quay của động cơ (như quạt máy chẳng hạn), phương pháp này có ưu điểm là dễ làm, nhưng mà biến trở sẽ tiêu hao 1 phần năng lượng và biến thành nhiệt năng. Ta thay biến trở bằng 1 nút nhấn, và bắt đầu nhấn, ta nhấn nhanh thì động cơ quay mượt, thời gian nhấn lâu hơn thời gian nhả thì động cơ quay nhanh. Đó là ý tưởng về PWM, ta thay 1 nút nhấn bằng 1 IC có khả năng phát xung và có thể điều chỉnh được 2 đại lượng (tần số và độ rộng xung). IC này ngày nay thường dùng các loại vi điều khiển mà điển hình là ATmega8 với khả năng điều chế PWM vượt trội và giá thành rẻ. Do đó, khi nói đến PWM trên ESC ta có thể tham khảo về mặt tần số với đơn vị là Hz (thường từ 6k-8kHz)
  + FET: transistor trường ứng, đây là khối công suất của mạch ESC, xung do vi điều khiển điều chế và phát ra không có khả năng cấp dòng cho động cơ vì đơn giản nó chỉ là tín hiệu PWM, vì vậy cần có các FET cấp dòng cho động cơ với sự điều khiển bằng xung PWM. Khả năng chịu dòng của FET cũng là khả năng chịu dòng của ESC



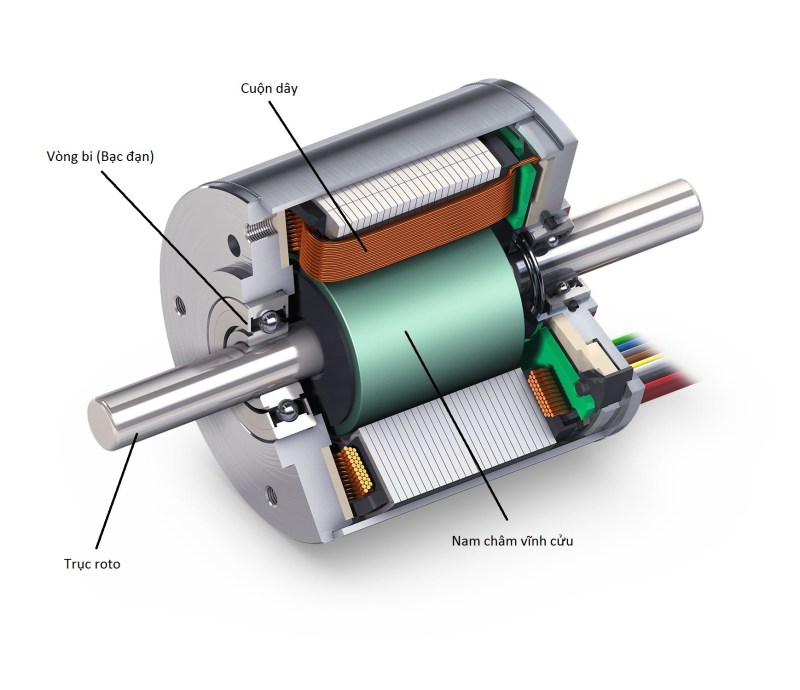
Hình 7. ESC của xe RGT Rock Hammer

* Động cơ (có/không) chổi than [8]
  + Chổi than là một vật liệu dẫn điện làm từ carbon có tác dụng tiếp điện, duy trì kết nối điện giữa bộ phận tĩnh và các phần chuyển động của động cơ điện DC hoặc AC được sử dụng trong công nghiệp sản xuất sử dụng động cơ dây quấn



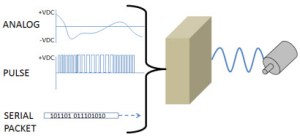
Hình 8. Động cơ có chổi than

* + Động cơ chổi than là loại mô-tơ dùng chổi than chì. Động cơ chổi than sử dụng trong xe điện RC là loại Mô-tơ dùng điện một pha, gồm có hai dây đỏ và đen. Ưu điểm của Mô-tơ chổi than là giá xe điều khiển từ xa sản xuất thấp cho nên chúng ta có thể thấy những chiếc xe điện RC giá thành thấp có gắn động cơ chổi than.
  + Nhược điểm của loại động cơ chổi than được làm bằng chổi than chì nên sau một khoảng thời gian sử dụng sẽ làm mòn chổi than, tuổi thọ kém, mô-tơ tiêu thụ điện lớn, công suất yếu hơn các loại mô-tơ không chổi than có cùng kích cỡ. Vì vậy những chiếc xe mô hình RC chạy loại mô-tơ này thường không có tốc độ cao.
  + Nhược điểm của đông cơ chổi than khá lớn và gây bất tiện, vì vậy việc sáng tạo ra động cơ không chổi than là một bước đột phá lớn, như việc sáng tạo ra pin Li-Po.

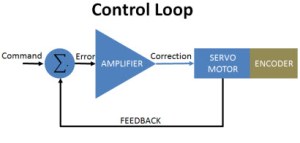


Hình 9. Động cơ không chổi than

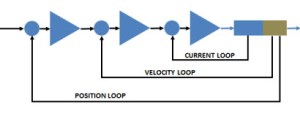
* + Động cơ DC không chổi than có các ưu điểm của động cơ đồng bộ nam châm vĩnh cửu như: tỷ lệ momen/quán tính lớn, tỷ lệ công suất trên khối lượng cao.
  + Do máy được kích từ bằng nam châm vĩnh cửu nên trên rotor hiệu suất động cơ cao hơn.
  + Động cơ kích từ nam châm vĩnh cửu không cần chổi than và vành trượt nên không tốn chi phí bảo trì chổi than. Ta cũng có thể thay đổi đặc tính động cơ bằng cách thay đổi đặc tính của nam châm kích từ và cách bố trí nam châm trên rotor.
  + Một số đặc tính nổi bật của động cơ không chổi than khi hoạt động:
    - Tỷ lệ công suất/khối lượng máy điện cao.
    - Tỷ lệ momen/quán tính lớn (có thể tăng tốc nhanh).
    - Vận hành nhẹ nhàng (dao động của momen nhỏ) thậm chí ở tốc độ thấp (để đạt được điều khiển vị trí một cách chính xác).
    - Mômen điều khiển được ở vị trí bằng không.
    - Vận hành ở tốc độ cao.
    - Có thể tăng tốc và giảm tốc trong thời gian ngắn.
    - Hiệu suất cao.
    - Kết cấu gọn.
* Động cơ Servo [9]
  + Động cơ Servo là một bộ phận của hệ thống điều khiển chuyển động của máy móc. Một trong các bộ phận không thể thiếu giúp Động cơ Servo có thể hoạt động đó chính là Driver servo. Tương tự như driver của máy tính. Động cơ Servo cung cấp lực chuyển động cần thiết cho các thiết bị máy móc khi vận hành
  + Ở ngành công nghiệp, đa số các động cơ Servo sử dụng động cơ một chiều không chổi than.
  + Rotor của động cơ là một nam châm vĩnh cửu có từ trường mạnh và Stator của động cơ được cuốn các cuộn dây riêng biệt, được cấp nguồn theo một trình tự thích hợp để quay rotor.
  + Nếu thời điểm và dòng điện cấp tới các cuộn dây là chuẩn xác thì chuyển động quay của rotor phụ thuộc vào tần số và pha, phân cực và dòng điện chạy trong cuộn dây stator.
  + Động cơ servo được hình thành bởi những hệ thống hồi tiếp vòng kín. Tín hiệu ra của động cơ được nối với một mạch điều khiển. Khi động cơ vận hành thì vận tốc và vị trí sẽ được hồi tiếp về mạch điều khiển này. Khi đó bầt kỳ lý do nào ngăn cản chuyển động quay của động cơ, cơ cấu hồi tiếp sẽ nhận thấy tín hiệu ra chưa đạt được vị trí mong muốn. Mạch điều khiển tiếp tục chỉnh sai lệch cho động cơ đạt được điểm chính xác nhất.



Hình 10. Tín hiệu điều khiển động cơ servo



Hình 11. Phản hồi hệ thống động cơ



Hình 12. Mạch vòng điều khiển động cơ servo

* Pin Lipo [10]
  + Pin Lipo (viết tắt từ Lithium Polymer) là một loại pin sạc sử dụng chất điện phân polymer khô. Sự ra đời của pin Lipo là một trong những yếu tố khiến cho mô hình điện phát triển nhanh chóng, đặc biệt là mô hình máy bay.
  + Những ưu điểm chính khiến Pin LiPo được dân chơi mô hình ưu tiên lựa chọn so với các loại pin sạc khác (NiCad, NiMH) là:
    - Pin RC LiPo nhỏ, nhẹ và có thể làm ở mọi hình dáng kích thước.
    - Pin RC LiPo có dung lượng cao trong khi kích thước, khối lượng nhỏ hơn các loại pin khác.
    - Pin RC LiPo có dòng xả cao đảm bảo đủ cung cấp năng lượng cho các động cơ có công suất cao.
  + Tóm lại, pin Lipo có tỉ lệ năng lượng lưu trữ/đơn vị khối lượng cao và có thể chế tạo ở kích thước, hình dạng đa dạng phù hợp với đa số mô hình RC.
  + Tuy nhiên, pin LIPO cũng có một vài nhược điểm so với các loại pin khác:
    - Pin LiPo vẫn còn đắt tiền so với pin NiCad và NiMH. Tuy nhiên, với sự phát triển của công nghệ, giá thành pin Lipo sẽ giảm dần.
    - Tuổi thọ Pin LiPo không cao, chỉ khoảng 300-400 lần sạc (và sẽ thấp hơn nhiều nếu không được chăm sóc đúng cách). Tuy nhiên nếu sử dụng đúng cách, tuổi thọ pin Lipo có thể lên tới 1000 lần sạc/xả.
    - Pin Lipo dễ bắt lửa và cháy nổ nếu không bảo quản, sử dụng đúng cách
    - Pin LiPo yêu cầu cao về việc tuân thủ các quy tắc khi sử dụng, khai thác để đảm bảo tuổi thọ và an toàn.
  + Thông số kỹ thuật pin lipo:



Hình 13. Pin Lipo

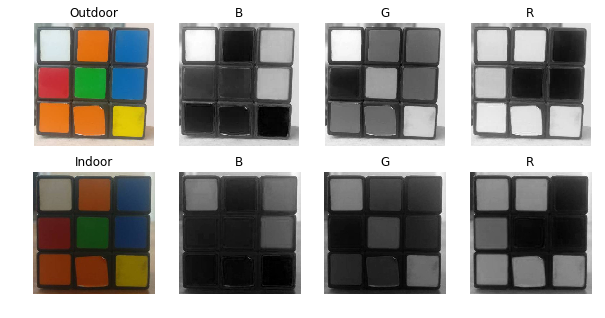
* + - Các thông số quan trọng cần biết khi mua pin Lipo gồm có: Điện áp, dung lượng, dòng xả.
    - Điện áp:
      * Một cục pin Lipo được ghép (song song hoặc nối tiếp) từ một hay nhiều cell pin. Mỗi cell pin lipo có điện áp 3,7V (khác với cell pin NiCad&NiMH thường có điện áp 1,2V). Tùy theo mô hình lựa chọn cần điện áp lớn hay nhỏ sẽ dùng pin được ghép bởi 1 cell, 2 cell hay nhiều hơn. Để tăng điện áp của pin lipo, các cell pin được ghép nối tiếp với nhau và được kí hiệu bởi chữ S. Ví dụ:
        + Pin 1 cell (1S): Điện áp 3,7V
        + Pin 2 cell (2S): Điện áp 7,4V
        + Pin 3 cell (3S): Điện áp 11,1V
        + Pin 4 cell (4S): Điện áp 14,8V
        + Pin 5 cell (5S): Điện áp 18,5V
        + Pin 6 cell (6S): Điện áp 22,2V
        + Pin 8 cell (8S): Điện áp 29,6V
        + Pin 10 cell (10S): Điện áp 37V
        + Pin 12 cell (12S): Điện áp 44,4V
        + Để tăng dung lượng pin mà vẫn giữ nguyên điện áp, các cell pin sẽ được mắc song song và được kí hiệu bởi chữ P. Ví dụ pin lipo có kí hiệu 2S2P gồm 2 cục pin 2S được mắc song song với nhau.
      * Trên máy bay mô hình, thường dùng động cơ 3 pha không chổi quét. Động cơ này có thông số quan trọng là Kv, tương ứng với số vòng quay được ứng với 1V điện áp. Ví dụ: Động cơ 1000kv trong dải từ 10V đến 25V nghĩa là động cơ này sẽ quay với vận tốc 10.000 vòng/phút khi điện áp pin là 10V và lên đến 25.000 vòng/phút khi điện áp là 25V. Chính vì vậy, điện áp pin rất quan trong khi chế tạo máy bay mô hình.
    - Dung lượng:
      * Dung lượng pin (Capacity – C) là lượng năng lượng lưu trữ trong pin khi nạp đầy, có đơn vị là mAh (mini ampe giờ). Dung lượng pin tương ứng với dòng tối đa mà pin xả (tiêu hao, tính theo mA) để pin chạy được 1 giờ. Ví dụ, một viên LiPo có dung lượng 1000mAh sẽ được xả hoàn toàn trong một giờ (dùng trong 1 giờ) với dòng tải 1000mA trên nó. Vẫn cùng viên pin này nhưng với dòng tải 500mAh thì dùng được khoảng 2 giờ, nhưng nếu tăng dòng tải lên 15000 mA thì thời gian để tiêu hao pin sẽ chỉ được khoảng 4 phút.
      * Như vậy, cùng một mô hình, dung lượng pin sẽ quyết định thời gian chạy trước khi hết pin. Đối với máy bay mô hình, dung lượng pin sẽ không tỉ lệ thuận với thời gian bay vì pin dung lượng lớn thì khối lượng pin sẽ lớn và dòng tiêu thụ sẽ cao hơn. Vì vậy cần cân đối của việc tăng dung lượng pin để tăng thời gian bay của mô hình.
    - Dòng xả:
      * Thông số quan trọng thứ 3 là dòng xả của pin. Dòng xả hiểu đơn giản là một viên pin có khả năng cung cấp dòng lớn một cách an toàn như thế nào. Dòng xả liên tục (continuous discharge) an toàn trên pin lipo được tính theo bội số của dung lượng pin (C). Một viên pin với một dòng xả 10C có nghĩa là bạn có thể xả một cách an toàn với tốc độ gấp 10 lần so với dung lượng của viên pin đó.
      * Ví dụ: Pin lipo có dung lượng 2000mAh; dòng xả 20C tức là dòng xả liên tục an toàn của pin là 20 x 2000mA = 40000mA = 40A.
      * Ngoài dòng xả liên tục (Continuous Discharge Rate), pin có thể xả với dòng cao hơn trong một thời gian rất ngắn (khoảng vài giây) để gia tăng công suất (tiếng anh là Burst Discharge).
      * Thông thường, Pin có dòng xả càng cao thì càng đắt. Vì vậy, nếu bạn không có nhiều tiền, không nên chọn pin có dòng xả quá cao so với nhu cầu sử dụng.
      * Vậy làm sao để bạn biết dòng xả bao nhiêu là phù hợp khi mua pin RC Lipo? Câu trả lời dễ dàng nhất sẽ là chọn pin có dòng xả C lớn nhất mà bạn có thể … Nếu tiền không phải là vấn đề. Tuy nhiên, nếu bạn mới chơi, nên chọn pin có dòng xả vừa phải. Để chọn dòng xả pin, bạn phải ước lượng xem, động cơ bạn sử dụng khi chạy với 100% ga thì dòng tiêu thụ là bao nhiêu. Ví dụ là 30A thì bạn nên chọn pin có dòng xả liên tục <40A (ví dụ pin 2000mAh thì chọn dòng xả <20C).

## Tìm hiểu thuật toán tìm đường đi sử dụng thư viện OpenCV

### Nhận diện màu sắc

#### Hệ màu [11]

Hệ màu RGB: Màu RGB thu được từ sự kết hợp tuyến tính (Linear combination) của 3 giá trị Red, Green và Blue. Ba kênh màu là sự tương quan của lượng ánh sáng chiếu vào bề mặt.



Hình 14. Phân biệt giữa 3 kênh R, G, B

Ta rút ra được những đặc điểm từ hệ màu RGB như sau:

* + Không đồng nhất đáng kể giữa 3 kênh
  + Trộn lẫn giữa sắc độ và độ chói

Hệ màu LAB: Gồm 3 thành phần L – Lightness (Intensity), a – thành phần màu từ xanh lá 🡪 đỏ tươi (Green to Magenta), b – thành phần màu từ xanh dương 🡪 vàng (Blue to Yellow).

Khác với hệ RGB – 3 kênh riêng biệt được mã hóa dựa trên cường độ sáng. Còn với LAB, L được tách biệt là mã hóa về độ sáng và 2 kênh còn lại dùng cho mã hóa màu.

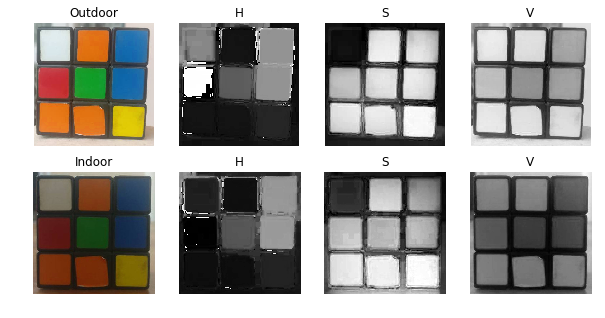


Hình 15. Phân biệt giữa 3 kênh L, A, B

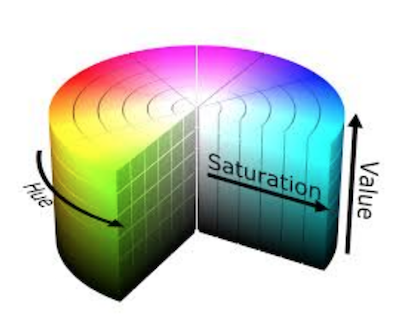
Ta rút ra những đặc điểm từ hệ màu LAB:

* + Không gian màu gần gũi và khá giống với cách nhận diện của con người
  + Khi hiển thị và chụp ảnh sẽ độc lập với thiết bị
  + Biển đổi được qua hệ màu RGB qua một phương trình biến đổi phức tạp

Hệ màu HSV: Có 3 thành phần: H – Hue (Bước sóng cực đại), S – Saturation (Độ tinh khiết, sắc thái màu sắc), V – Value (Cường độ). Chỉ sử dụng 1 kênh duy nhất để biểu diễn màu là Hue.



Hình 16. Phân biệt giữa 3 kênh H, S, V



Hình 17. Vòng tròn hệ màu HSV



Hình 18. Không gian 3 chiều của không gian màu HSV

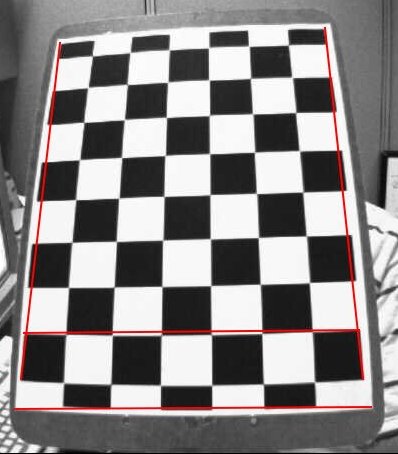
### Lọc màu đường và biển báo

### Thuật toán tìm góc lệch

#### Cân chỉnh camera [12]

Những chiếc camera giá rẻ trên thị trường sẽ cho ra những hình ảnh bị méo mó, biến dạng. Có 2 loại biến dạng chính: radial distortion và tangential distortion

Với radial distortion, những đường vẽ thẳng nhưng trên hình thực thì bị cong



Hình 19. Ví dụ về radial distortion

Để giải quyết vấn đề này, ta biến đổi qua công thức:

x_{corrected} = x( 1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \\
y_{corrected} = y( 1 + k_1 r^2 + k_2 r^4 + k_3 r^6)

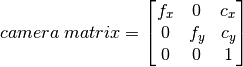
Đối với tangential ta có công thức biến đổi sau:

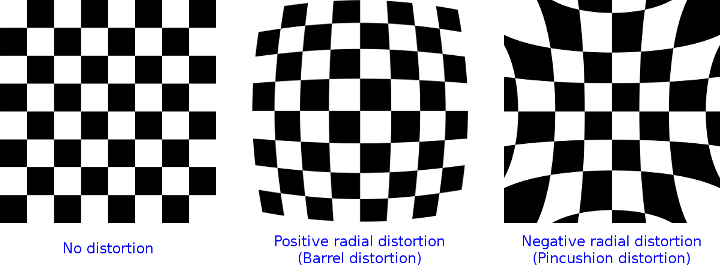
x_{corrected} = x + [ 2p_1xy + p_2(r^2+2x^2)] \\
y_{corrected} = y + [ p_1(r^2+ 2y^2)+ 2p_2xy]

Ta cần tìm 5 biến (hệ số biến dạng):

Distortion \; coefficients=(k_1 \hspace{10pt} k_2 \hspace{10pt} p_1 \hspace{10pt} p_2 \hspace{10pt} k_3)

Ngoài ra những biến ở các công thức trên được tìm và lưu sử dụng cho từng camera là 1 ma trận 3x3:





Hình 20. Cân chỉnh frame ảnh



Hình 21. Kết quả sau khi cân chỉnh

#### Bẻ thẳng frame ảnh

* + - Hàm getPerspectiveTransform: [13]hàm tính ra ma trận 3x3 của một ma trận biến đổi



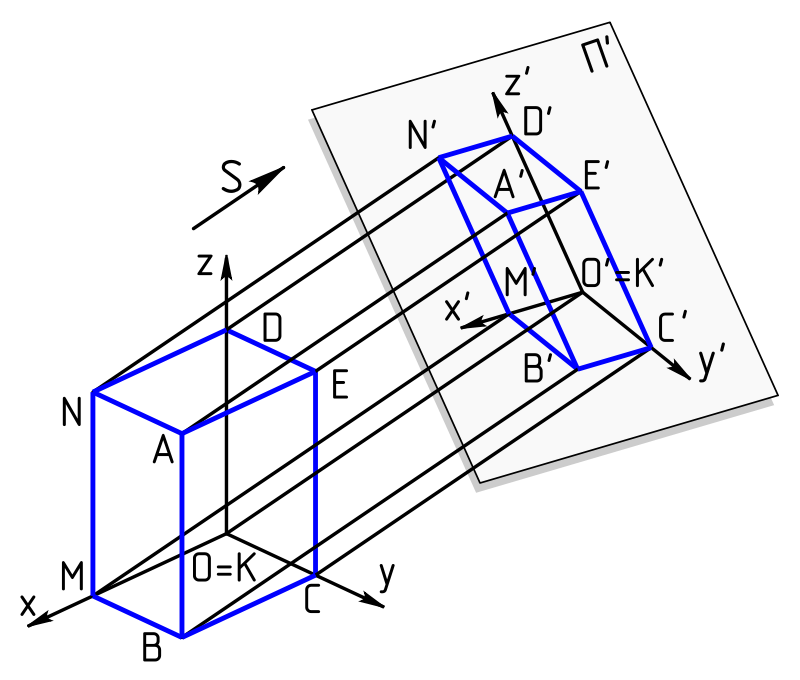
Với dst(i)=(x'_i,y'_i),
src(i)=(x_i, y_i),
i=0,1,2,3

* + - Hàm warpPerspective: hàm áp dụng biến đổi perspective lên ảnh lấy được từ hàm getPerspectiveTransform.

Công thức sử dụng để biến đổi:

\texttt{dst} (x,y) =  \texttt{src} \left ( \frac{M_{11} x + M_{12} y + M_{13}}{M_{31} x + M_{32} y + M_{33}} ,
     \frac{M_{21} x + M_{22} y + M_{23}}{M_{31} x + M_{32} y + M_{33}} \right )

* + - Sử dụng 2 hàm này, ta được một bức ảnh 2 line đường đã bẻ thẳng, còn được gọi là birdview
    - Birdview là tầm nhìn vật thể từ trên cao nhìn xuống như cách nhìn của loài chim, thường được dùng để vẽ bản thiết kế, thi công, bản đồ.



Hình 22. Mô hình về birdview



Hình 23. Birdview với lane đường

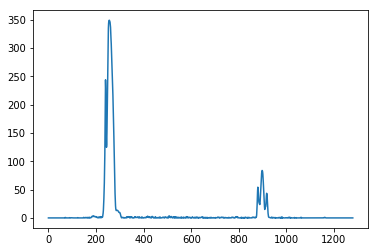
#### Biểu đồ những điểm thuộc làn đường

* + - Histogram là biểu đồ biểu diễn phân bố các điểm thỏa 1 yêu cầu với 1 lượng dữ liệu liên tục theo thời gian



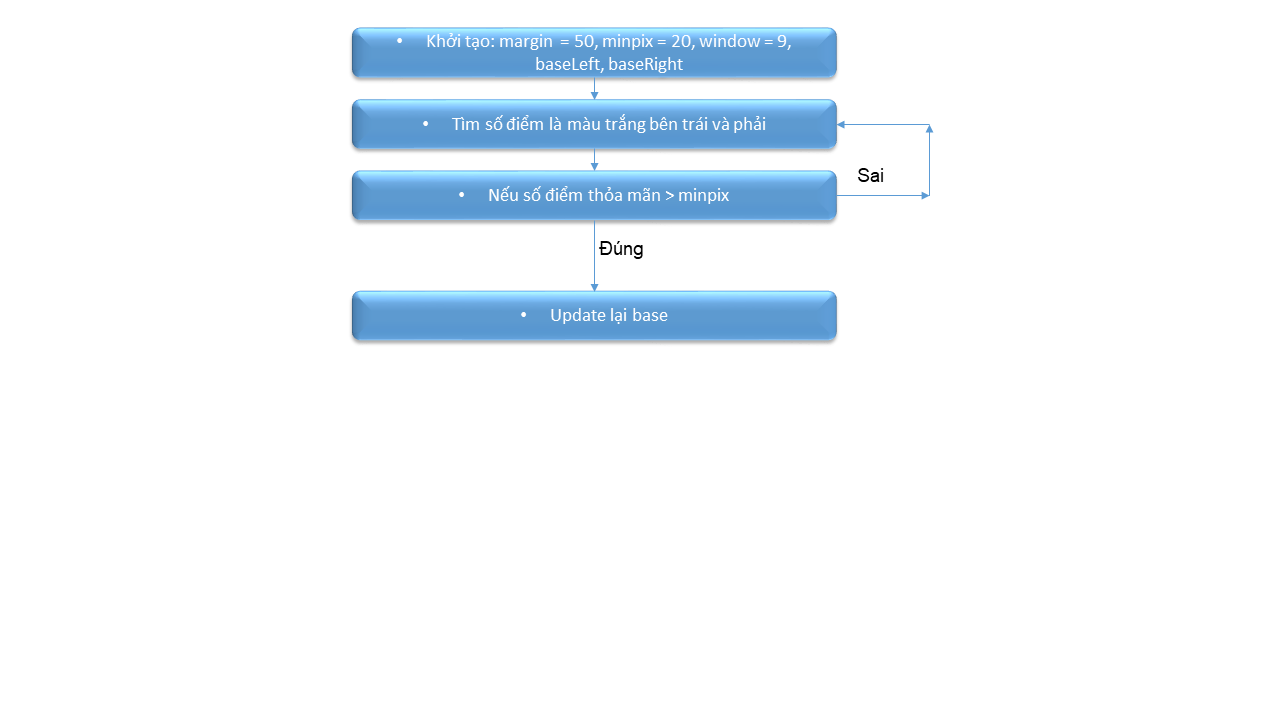
Hình 24. Ví dụ về histogram [14]

* + - Biểu đồ phân bố tần số (Biểu đồ phân bố mật độ, biểu đồ cột) dùng để đo tần số xuất hiện của một vấn đề nào đó, cho ta thấy rõ hình ảnh sự thay đổi, biến động của một tập dữ liệu. Đây là một khái niệm rất phổ biến, được sử dụng rộng rãi từ lĩnh vực kỹ thuật tới kinh tế.
    - Biểu đồ phân bố tần số (Biểu đồ phân bố mật độ, biểu đồ cột) dùng để đo tần số xuất hiện của một vấn đề nào đó, cho ta thấy rõ hình ảnh sự thay đổi, biến động của một tập dữ liệu. Đây là một khái niệm rất phổ biến, được sử dụng rộng rãi từ lĩnh vực kỹ thuật tới kinh tế.
    - Biểu đồ phân bố tần số (Biểu đồ phân bố mật độ, biểu đồ cột) dùng để đo tần số xuất hiện của một vấn đề nào đó, cho ta thấy rõ hình ảnh sự thay đổi, biến động của một tập dữ liệu. Đây là một khái niệm rất phổ biến, được sử dụng rộng rãi từ lĩnh vực kỹ thuật tới kinh tế.
    - Trục tung (Oy) biểu diễn số lượng điểm ảnh (Pixel) của mức xám.
    - Trục hoành (Ox) biểu diễn mức xám.
    - Giá trị lớn nhất của trục hoành chính là số lượng điểm ảnh (Pixel) có trong một bức ảnh.
    - Với ảnh màu như RGB thì có tới 3 biểu đồ Histogram thể hiện từng kênh màu.
    - Một biều đồ tốt à biểu đồ có số lượng điểm ảnh nhiều nhất ở vùng giữa (Độ sáng trung bình) và ít dần ra 2 vùng sáng tối (Ngọn núi).
    - Dựa vào biều đồ Histogram mà bạn có thể biết được hình ảnh sáng tối như thế nào.
    - Áp dụng cho các xử lý ảnh cao cấp khác.
    - Ảnh tối là ảnh có tập trung quá nhiều điểm ảnh bên vùng tối.
    - Ảnh sáng là ảnh có tập trung quá nhiều điểm ảnh bên vùng sáng.
    - Ảnh có độ tương phản cao là ảnh có điểm ảnh tập trung nhiều ở 2 vùng sáng và tối và tập trung ít ở vùng giữa.
    - Ảnh có độ tương phản thấp là ảnh có điểm ảnh tập trung nhiều ở vùng giữa và tập trung rất ít ở vùng hai vùng sáng và tối.



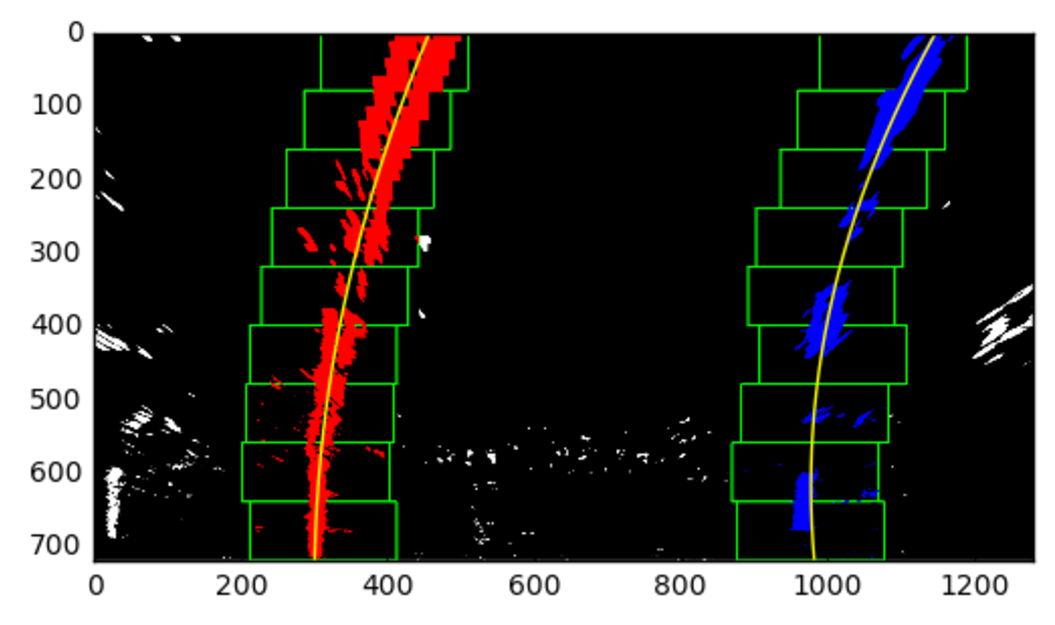
Hình 25. Histogram của lane đường

#### Chia khung làn đường (Sliding window)



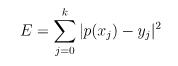
Hình 26. Flowchart sliding window

* + - Chọn margin = 50: một window sẽ rộng 50 pixels
    - Chọn 9 window
    - Đầu tiên ta chọn điểm gốc của làn trái và làn phải bằng cách lấy max của histogram
    - Đếm số điểm màu trắng (tương ứng với phần tử lane đường) với chiều rộng của khung bằng margin và tổng hợp vào một mảng.
    - Nếu mật độ của những điểm màu trắng vượt mức minpix thì cập nhật lại điểm gốc
    - Qua những khung đó, ta vẽ được đường cong



Hình 27. Sliding window

#### Phương trình đường cong

* + - Hàm numpy.polyfit: Tổng hợp 1 mảng điểm thành một đa thức với công thức: p(x) = p[0] \* x\*\*deg + ... + p[deg] với góc tìm được từ những điểm (x, y) 🡺 Hàm trả về 1 vector hệ số p. Từ đó ta vẽ được đường cong qua phương trình
    - 
    - Phân tích ta được thuật toán trên code như sau:

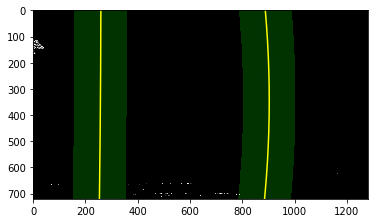
x[0]\*\*n \* p[0] + ... + x[0] \* p[n-1] + p[n] = y[0]

x[1]\*\*n \* p[0] + ... + x[1] \* p[n-1] + p[n] = y[1]

...

x[k]\*\*n \* p[0] + ... + x[k] \* p[n-1] + p[n] = y[k]

* + - Sau khi tính 2 phương trình ta vẽ được qua hàm fill poly



Hình 28. Skip sliding window

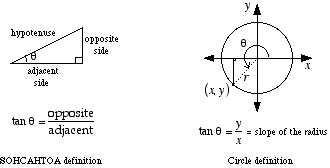
#### Tính toán dự đoán góc bẻ lái

Đo khoảng cách từ camera tới trung bình tâm của 2 bánh sau = 37cm

Đo khoảng cách từ camera tới điểm trên ảnh ở thực tế = 29cm

Tính khoảng cách từ điểm đích tới trung bình tâm của 2 bánh sau theo pixel = (29cm / 150) \* (29cm + 37cm) = 341 pixel. 150 ở đây là mức pixel trên ảnh tính từ gốc.

Tính góc lệch bằng hàm tan trong lượng giác

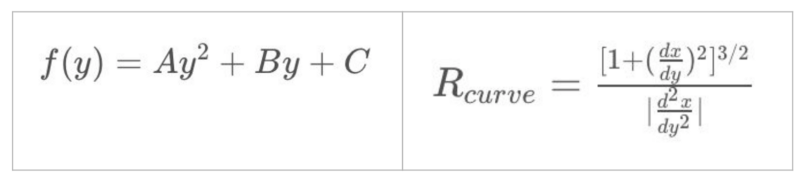


Hình 29. Tính góc

Cân chỉnh góc bằng một hệ số nhân với hệ số tìm được do thực nghiệm trên sa hình thực tế, ở đây chọn 1.5 để góc chính xác với sa hình thực tế.

#### Tính toán khoảng cách lệch so với 2 bên làn đường

* + - Sau khi tìm được 2 đường cong, ta tìm độ lệch (curvation) so với 2 làn đường qua công thức:





Hình 30. Kết quả tính độ lệch

#### Lưu lịch sử cho những frame kế tiếp để dự đoán

* Mỗi lần xử lý 1 frame ảnh, so sánh curvation
* Nếu khác nhau 🡺 Lấy kết quả của frame mới để xử lý và cho xe chạy
* Nếu giống nhau 🡺 Lấy kết quả của frame cũ để xử lý và cho xe chạy
* Sau mỗi lần tính toán góc lệch, tâm điểm thành công thì lưu vào lịch sử.

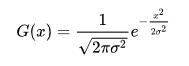
### Thuật toán tìm biển báo

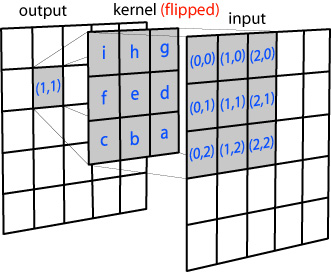
#### Nhận diện biển báo

* + - Chuyển màu thành ảnh xám (RGB 🡪 GRAY):
      * Công thức: **Y = 0.2126R + 0.7152G + 0.0722B**
      * Trong OpenCV dùng hàm: cvtColor(img, grayscale\_img, CV\_BGR2GRAY);

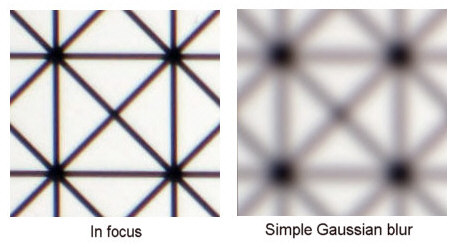


Hình 31. Chuyển ảnh màu thành xám

* + - Làm mờ ảnh bằng Gaussian
      * Sử dụng hàm cv2.GaussianBlur()
      * Đây là một dàng biến đổi Gaussian với công thức trong không gian 1 chiều: 
      * Trong không gian 2 chiều: 
      * Để dùng Gaussian blur, ta cần một kernel [N x N] và mỗi pixel sẽ nhân với kernel này

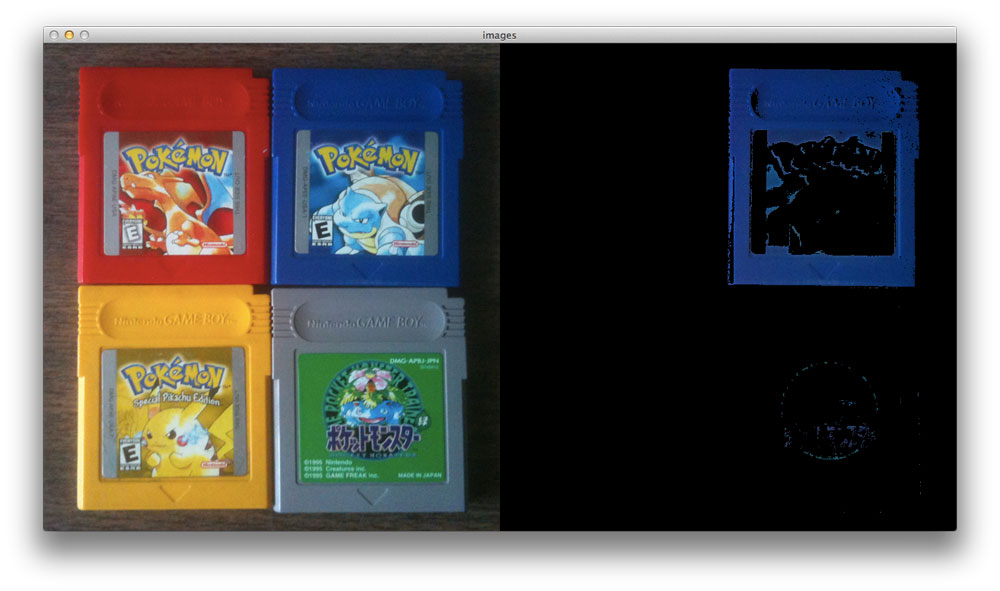


Hình 32. Ví dụ giải thích Gaussian blur bằng toán học



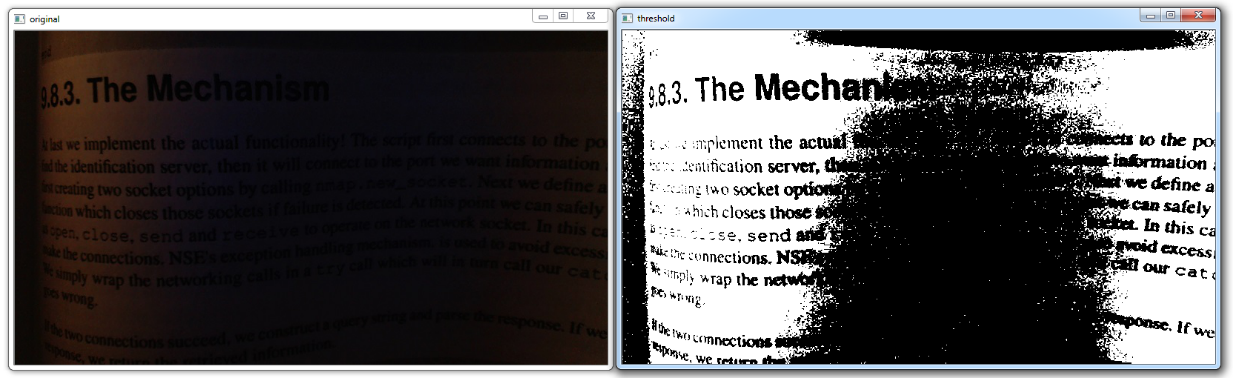
Hình 33. Gaussian blur

* + - Lọc màu xanh: Sử dụng hàm inrange của Opencv trong không gian màu HSV



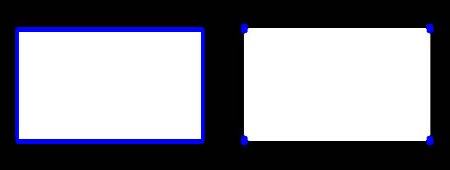
Hình 34. Lọc màu xanh

* + - Lọc trắng đen bằng ngưỡng 🡺 Đưa ảnh về dạng nhị phân (0 và 255): sử dụng hàm Threshold của OpenCV



Hình 35. Threshold [15]

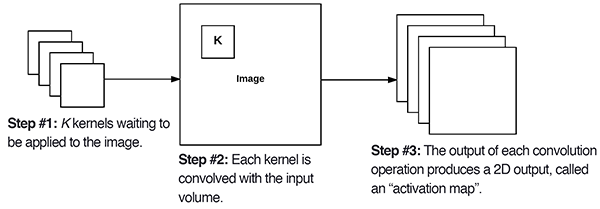
* + - Tìm contours bằng hàm find contours:



Hình 36. Contour

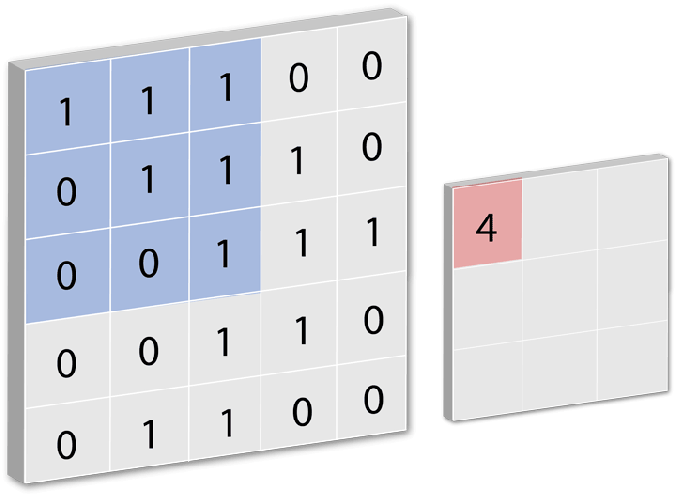
#### Những thành phần trong mạng neural

* + - Conv2D [16]
      * filters



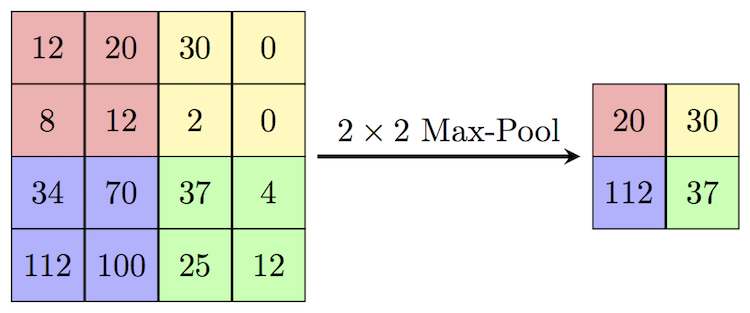
Hình 37. Các bước tạo ra map 2D

* + - * Conv2D dùng để trích xuất những feature có cấp cao hơn bằng cách thay thế cho từng pixel với giá trị tổng hợp từ những pixel trong ma trận NxN cho trước.

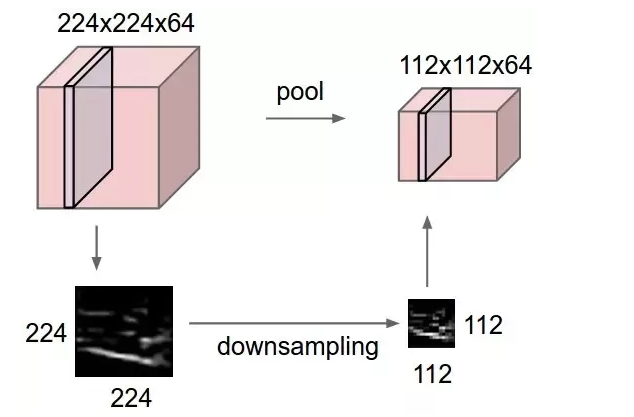


Hình 38. Ví dụ về Conv2d [17]

* + - MaxPooling
      * Quy trình hạ bậc, lấy 1 mẫu làm đại diện cho các phần tử đầu vào
      * MaxPooling dùng để giảm thiểu phép tính bằng cách giảm số phần tử trên mảng và thu nhỏ mảng ban đầu

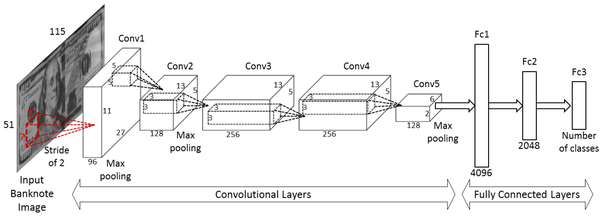


Hình 39. Ví dụ MaxPooling [18]



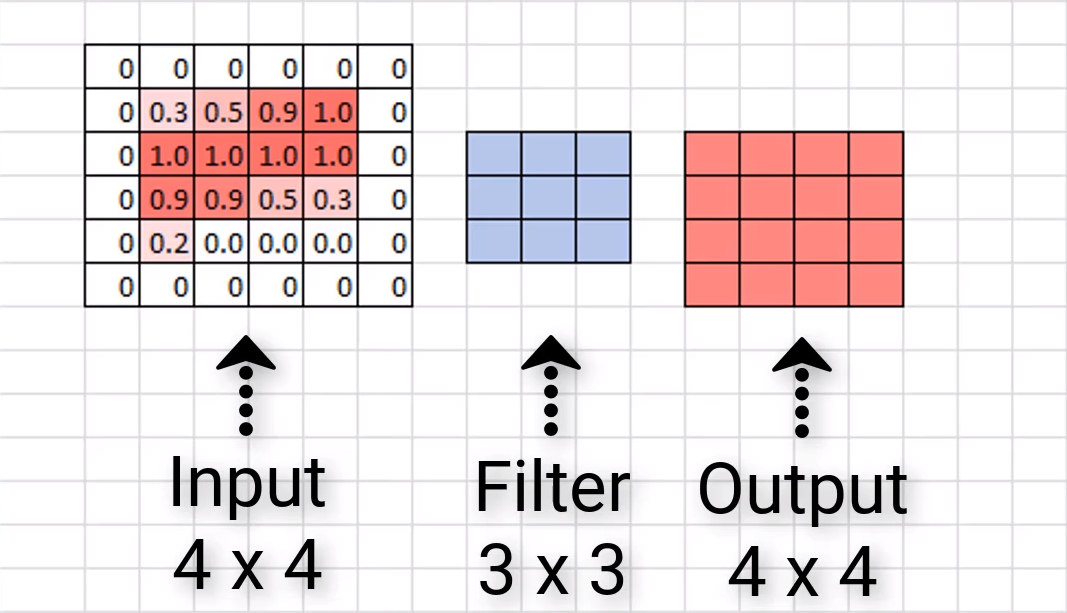
Hình 40. Ví dụ MaxPooling trong thực tế [18]

* + - Dropout
      * Dropout là kĩ thuật loại bỏ ngẫu nhiên những neural trong mạng để giảm thiểu overfitting. Điều này có nghĩa là những output của neural trước sẽ bỏ qua không sử dụng cũng như weight cũng sẽ không được update vào mạng hiện tại đang sử dụng dropout
    - Flatten
      * Những layer fully connected được định nghĩa dưới dạng vector. Nhưng mạng lưới neural cho ra hàng loạt bộ lọc (filter) dưới hình dạng lưới (grid). Flatten có nhiệm vụ biến đổi những bộ lọc này thành dạng vector để sử dụng tích chập và backpropagate (hồi tiếp).



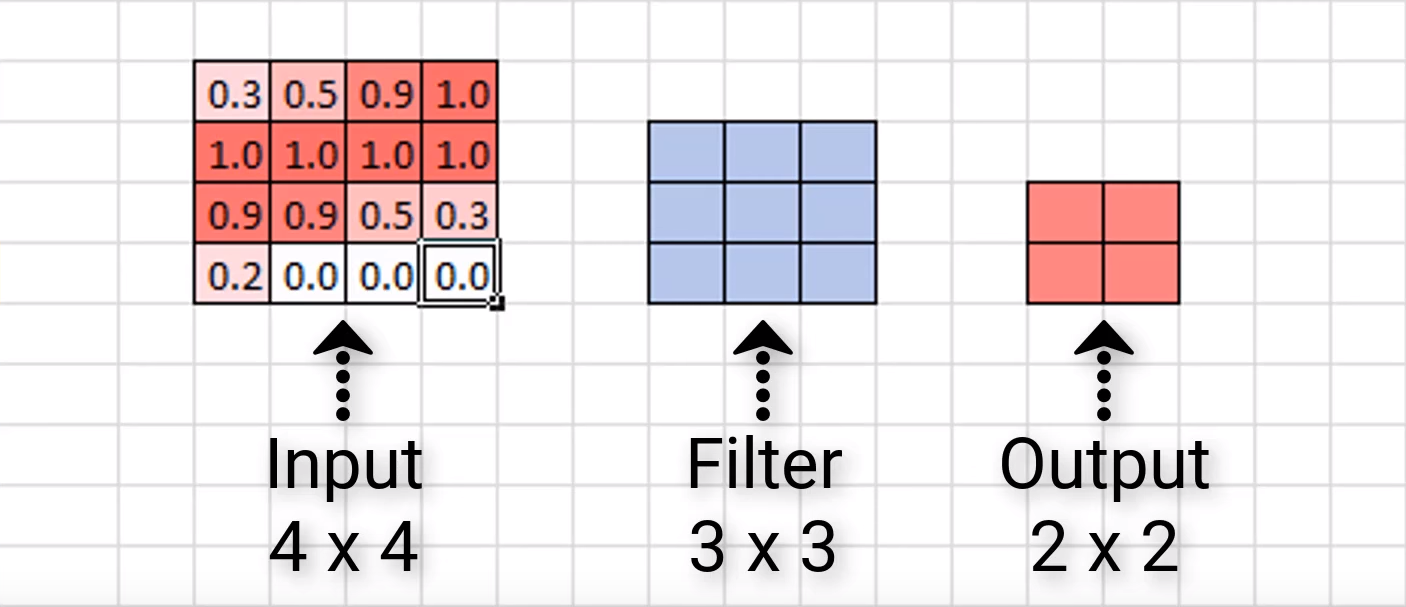
Hình 41. Quá trình biến đổi của flatten [19]

* + - Dense
      * Lớp dense đại diện cho một ma trận phép nhân vector. Những giá trị của ma trận là những thông số dùng để huấn luyện và những những giá trị này cũng được update khi backpropagate
      * uT.W,W∈Rn×m
      * Output sẽ cho ra một vector m chiều và lớp dense sẽ thay đổi số chiều của vector này. Theo cách nói toán học, nó sẽ áp dụng rotation (xoay), scaling (thay đổi kích thước), translation transform (biến đổi bằng cách dịch chuyển vị trí).
    - Padding
      * Zero padding
        + Zero padding là kĩ thuật cho phép giữ lại kích thước input (input size)
        + Zero padding xảy ra khi ta cho tất cả các phần tử viền của ma trận bằng 0



Hình 42. Ví dụ Zero padding [20]

* + - * Valid padding
        + Kích thước input sẽ bị co lại
        + Với kích thước n x n, ta sẽ giải quyết với 1 bộ lọc f x f và kết quả thu được (n – f + 1) x (n – f + 1)
        + Ví dụ với ma trận 4 x 4 : (n – f + 1) = (4 – 3) + 1 = 2



Hình 43. Ví dụ Valid Padding [20]

#### Huấn luyện và Dự đoán

* Compile:
  + Thành phần này có nhiệm vụ tạo thành một object Python, object này sẽ xây dụng mạng CNN
  + Object này được xây dựng dựa trên tính toán đồ thị với một định dạng đúng với keras backend đang sử dụng.
  + Complie có những thành phần như loss (hàm mất mát), optimizer (hàm tối ưu), …
* Batch size [21]
  + Là 1 siêu tham số định nghĩa số lượng mẫu cần làm việc trước khi cập nhật những thông số trong model
  + Batch cũng có thể hiểu là vòng lặp for qua 1 hoặc nhiều mẫu và dự đoán. Cuối chuỗi hoạt động của batch, lớp dự đoán sẽ được so sánh với output mong muốn và tính toán ra sai số.
* Epoch
  + Là 1 siêu tham số định nghĩa số lần sẽ huấn luyện
  + 1 epoch bao gồm nhiều batch
* Verbose
  + Dùng để hiện thị quá trình huấn luyện
    - 0: chế độ im lặng, không hiện gì
    - 1: chế độ có thanh tiến độ
    - 2: chế độ chỉ hiện số epoch

## Lập trình đa luồng

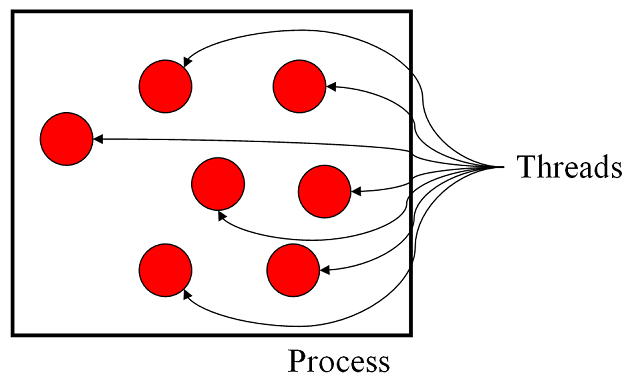
### Luồng (thread). Khác nhau giữa thread và process (tiến trình)

Thread là một đơn vị cơ bản trong CPU. Một luồng sẽ chia sẻ với các luồng khác trong cùng process về thông tin data, các dữ liệu của mình. Việc tạo ra thread giúp cho các chương trình có thể chạy được nhiều công việc cùng một lúc.

Process là quá trình hoạt động của một ứng dụng. Tiến trình (process)chứa đựng thông tin tài nguyên, trạng thái thực hiện của chương trình.

Thread là một bước điều hành bên trong một process. Luồng (thread) là một khối các câu lệnh (instructions) độc lập trong một tiến trình và có thể được lập lịch bởi hệ điều hành. Hay nói một cách đơn giản, Thread là các hàm hay thủ tục chạy độc lập đối với chương trình chính. Một process dĩ nhiên có thể chứa nhiều thread bên trong nó. Điểm quan trọng nhất cần chú ý là một thread có thể làm bất cứ nhiệm vụ gì một process có thể làm.

Một điểm khác biệt nữa đó là nhiều thread nằm trong cùng một process dùng một không gian bộ nhớ giống nhau, trong khi process thì không. Điều này cho phép các thread đọc và viết cùng một kiểu cấu trúc và dữ liệu, giao tiếp dễ dàng giữa các thread với nhau. Giao thức giữa các process, hay còn gọi là IPC (inter-process communication) thì tương đối phức tạp bởi các dữ liệu có tính tập trung sâu hơn. Ngoài các tài nguyên riêng của mình (các biến cục bộ trong hàm), các luồng chia sẻ tài nguyên chung của tiến trình. Việc thay đổi tài nguyên chung (ví dụ, đóng file, gán giá trị mới cho biến) từ một thread sẽ được nhìn thấy bởi tất cả các thread khác. Vì vậy, lập trình viên cần phải thực hiện đồng bộ việc truy cập tài nguyên chung giữa các luồng.



Hình 44. Sự khác nhau giữa thread và process [22]

### Đa luồng (Multithreading)

Một chương trình đa luồng chứa hai hoặc nhiều phần mà có thể chạy đồng thời và mỗi phần có thể xử lý tác vụ khác nhau tại cùng một thời điểm, để sử dụng tốt nhất các nguồn có sẵn, đặc biệt khi máy tính của bạn có nhiều CPU.

Python cung cấp thread Module và threading Module để bạn có thể bắt đầu một thread mới cũng như một số tác vụ khác trong khi lập trình đa luồng. Mỗi một Thread đều có vòng đời chung là bắt đầu, chạy và kết thúc. Một Thread có thể bị ngắt (interrupt), hoặc tạm thời bị dừng (sleeping) trong khi các Thread khác đang chạy – được gọi là yielding.

Một ví dụ đơn giản chỉ sử dụng một thread, có truyền tham số. Để chạy thread, ta dùng method start(). Target sẽ là function myThread, là nhiệm vụ mà thread phải hoàn thành. [22]

### Đồng bộ hóa các thread trong python

Trong lập trình đa luồng, các threads chia sẻ chung tài nguyên của tiến trình, vì vậy có những thời điểm nhiều luồng sẽ đồng thời thay đổi dữ liệu chung. Do đó, ta cần những cơ chể để đảm bảo rằng, tại một thời điểm chỉ có duy nhất một luồng được phép truy cập vào dữ liệu chung, nếu các luồng khác muốn truy cập vào đoạn dữ liệu này thì cần phải đợi cho thread trước đó hoàn thành công việc của mình.

Python cung cấp threading Module, mà bao gồm một kỹ thuật locking cho phép bạn đồng bộ hóa các Thread một cách dễ dàng. Một lock mới được tạo bởi gọi phương thức Lock().

Phương thức acquire(blocking) của đối tượng lock mới này được sử dụng để ép các Thread chạy một cách đồng bộ. Tham số blocking tùy ý cho bạn khả năng điều khiển để xem một Thread có cần đợi để đạt được lock hay không.

Nếu tham số blocking được thiết lập là 0, tức là Thread ngay lập tức trả về một giá trị 0 nếu không thu được lock và trả về giá trị 1 nếu thu được lock. Nếu blocking được thiết lập là 1, thì Thread cần đợi cho đến khi lock được giải phóng.

Phương thức release() của đối tượng lock được sử dụng để giải phóng lock khi nó không cần nữa.

### Ứng dụng đa luồng

Đa luồng có rất công dụng vô cùng hữu ích thích hợp cho những tác vụ chạy ngầm không cần quan tâm chính xác thời gian hoàn thành, đặc điểm lớn nhất của nó vẫn là chạy song song nhiều luồng cùng 1 lúc, có thể kể đến tác dụng hữu hiệu nhất trong các ứng dụng web, và dự án của bản thân là ghi log. Ví dụ dự án có tác vụ gồm nhiều ưu đãi, người dùng muốn lưu ưu đãi đó cho dùng lần sau thì cần ghi log 2 sự kiện gồm view log và save log

## Board Nvidia Jetson Tx2

### Thông số kỹ thuật

#### JETSON TX2 MODULE

* NVIDIA Pascal™ Architecture GPU
* 2 Denver 64-bit CPUs + Quad-Core A57 Complex
* 8 GB L128 bit DDR4 Memory
* 32 GB eMMC 5.1 Flash Storage
* Connectivity to 802.11ac Wi-Fi and Bluetooth-Enabled Devices
* 10/100/1000BASE-T Ethernet

#### IO

* USB 3.0 Type A
* USB 2.0 Micro AB (supports recovery and host mode)
* HDMI
* M.2 Key E
* PCI-E x4
* Gigabit Ethernet
* Full-Size SD
* SATA Data and Power
* GPIOs, I2C, I2S, SPI, CAN\*
* TTL UART with flow control
* Display Expansion Header\*
* Camera Expansion Header\*
  + \*I/O expansion headers: refer to product documentation for header specification.

#### Power Option

* External 19V AC Adapter

#### Jetson Camera Module

* 5 MP Fixed Focus MIPI CSI Camera

#### Buttons

* Power On/Off
* Reset
* Force Recovery
* User-Defined

# XÂY DỰNG MÔ HÌNH VÀ GIẢI PHÁP

## Mô hình xe với RGT Rock Hammer và Nvidia Jetson Tx2 Development board

# KẾT QUẢ

# TỔNG QUAN

TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | "xe tu hanh," fpt, 2019. [Online]. Available: https://vnexpress.net/so-hoa/hanh-trinh-dau-tien-tren-xe-tu-hanh-cua-fpt-3664552.html. [Accessed 2019]. |
| [2] | "techcrunch," 2019. [Online]. Available: https://techcrunch.com/2016/04/26/google-uber-lyft-join-automakers-in-self-driving-car-lobby/. [Accessed 2019]. |
| [3] | "theverge," 2019. [Online]. Available: https://www.theverge.com/2017/9/20/16341478/tesla-amd-chip-self-driving-car. [Accessed 2019]. |
| [4] | "vicotech," 2019. [Online]. Available: https://vicotech.com.vn/he-thong-xe-tu-hanh-agv. [Accessed 2019]. |
| [5] | "vietrc," RGT Racing, 2019. [Online]. Available: http://www.vietrc.com/rgt-rock-hammer. [Accessed 2019]. |
| [6] | "danhgiaxe," [Online]. Available: https://www.danhgiaxe.com/he-thong-dan-dong-4-banh-toan-thoi-gian-doi-xung-cua-subaru-7780. [Accessed 2019]. |
| [7] | "myrcsaigon," [Online]. Available: http://myrcsaigon.com/can-nam-ro-ve-esc-bec-va-ubec/. [Accessed 2019]. |
| [8] | "xomrc," [Online]. Available: http://xomrc.com/kinh-nghiem-mua-hang/dong-co-khong-choi-than-la-gi-bai-viet-nay-se-giup-ban-hieu-ro-hon-ve-don-co-khong-choi-than/. [Accessed 2019]. |
| [9] | "solarstore," [Online]. Available: https://solarstore.vn/khai-niem-dong-co-servo-la-gi/. [Accessed 2019]. |
| [10] | "bkaero," [Online]. Available: https://bkaero.vn/nhung-kien-thuc-can-biet-ve-pin-lipo/. [Accessed 2019]. |
| [11] | "learnopencv," [Online]. Available: https://www.learnopencv.com/color-spaces-in-opencv-cpp-python/. [Accessed 2019]. |
| [12] | "opencv-python-tutroals," [Online]. Available: https://opencv-python-tutroals.readthedocs.io/en/latest/py\_tutorials/py\_calib3d/py\_calibration/py\_calibration.html. [Accessed 2019]. |
| [13] | "opencv," [Online]. Available: https://docs.opencv.org/2.4/modules/imgproc/doc/geometric\_transformations.html. |
| [14] | "laerd," [Online]. Available: https://statistics.laerd.com/statistical-guides/understanding-histograms.php. [Accessed 2019]. |
| [15] | "opencv," 2019. [Online]. Available: https://opencv.org/. |

PHỤ LỤC