TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG

HOÀNG ANH HIỆP

LUẬN VĂN TỐT NGHIỆP

DELTA ROBOT

KỸ SƯ NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

TP. HỒ CHÍ MINH, 2015

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG

HOÀNG ANH HIỆP – 41101175

LUẬN VĂN TỐT NGHIỆP

DELTA ROBOT

KỸ SƯ NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

GIẢNG VIÊN HƯỚNG DẪN

NGUYỄN TUẤN AN

TP. HỒ CHÍ MINH, 2014

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

TP. HCM, ngày….tháng…..năm……..

NHẬN XÉT LUẬN VĂN TỐT NGHIỆP

CỦA CÁN BỘ HƯỚNG DẪN

Tên luận văn:

DELTA ROBOT

Nhóm Sinh viên thực hiện:		**Cán bộ hướng dẫn:**
HOÀNG ANH HIỆP	41101175	NGUYỄN TUẤN AN

Đánh giá Luận văn

Về cuốn báo cáo:

Số trang	_____	Số chương	_____
Số bảng số liệu	_____	Số hình vẽ	_____
Số tài liệu tham khảo	_____	Sản phẩm	_____

Một số nhận xét về hình thức cuốn báo cáo:

Về nội dung luận văn:

Về tính ứng dụng:

Về thái độ làm việc của sinh viên:

<u>Đánh giá chung:</u> Luận văn đạt/không đạt yêu cầu của một luận văn tốt nghiệp kỹ sư, xếp loại Giỏi/ Khá/ Trung bình

<u>Điểm từng sinh viên:</u>

HOÀNG ANH HIỆP**:……../10**

Người nhận xét

(Ký tên và ghi rõ họ tên)

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

TP. HCM, ngày….tháng…..năm……..

NHẬN XÉT LUẬN VĂN TỐT NGHIỆP

CỦA CÁN BỘ PHẢN BIỆN

Tên luận văn:

DELTA ROBOT

Nhóm Sinh viên thực hiện:		**Cán bộ phản biện:**
HOÀNG ANH HIỆP	41101175	……

Đánh giá Luận văn

Về cuốn báo cáo:

Số trang	_____	Số chương	_____
Số bảng số liệu	_____	Số hình vẽ	_____
Số tài liệu tham khảo	_____	Sản phẩm	_____

Một số nhận xét về hình thức cuốn báo cáo:

Về nội dung luận văn:

Về tính ứng dụng:

Về thái độ làm việc của sinh viên:

<u>Đánh giá chung:</u> Luận văn đạt/không đạt yêu cầu của một luận văn tốt nghiệp kỹ sư, xếp loại Giỏi/ Khá/ Trung bình

<u>Điểm từng sinh viên:</u>

HOÀNG ANH HIỆP:……/10

Người nhận xét

(Ký tên và ghi rõ họ tên)

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

TP. HCM, ngày….tháng…..năm……..

ĐỀ CƯƠNG CHI TIẾT

TÊN ĐỀ TÀI: DELTA ROBOT
Cán bộ hướng dẫn: NGUYỄN TUẤN AN
Thời gian thực hiện: Từ ngày……..đến ngày……
Sinh viên thực hiện: **HOÀNG ANH HIỆP - 41101175**
Nội dung đề tài: *THIẾT KẾ MÔ HÌNH DELTA ROBOT NHẰM MỤC ĐÍCH NGHIÊN CỨU VÀ HỌC TẬP . MỤC ĐỊCH THỰC HIỆN ĐỀ TÀI LÀ ĐỂ AM HIỂU HƠN VỀ ROBOT VÀ CÁC VẤN ĐỀ LIÊN QUAN ĐẾN VIỆC ĐIỀU KHIỂN ROBOT CŨNG NHƯ NGHIÊN CỨU SÂU HƠN VỀ HOẠT ĐỘNG CỦA VI XỬ LÝ STM32.*
Kế hoạch thực hiện: *NGHIÊN CỨU LÝ THUYẾT VỀ DELTA ROBOT, THIẾT KẾ MẠCH ĐIỀU KHIỂN, THIẾT KẾ CƠ KHÍ ROBOT, THIẾT KẾ CHƯƠNG TRÌNH ĐIỀU KHIỂN, HOÀN TẤT VÀ BÁO CÁO.*

Xác nhận của Cán bộ hướng dẫn	TP. HCM, ngày….tháng …..năm….. **Sinh viên**

DANH SÁCH HỘI ĐỒNG BẢO VỆ LUẬN VĂN

Hội đồng chấm luận văn tốt nghiệp, thành lập theo Quyết định số ……
ngày …….. của Hiệu trưởng Trường Đại học Bách khoa TP.HCM.

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

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

LỜI CẢM ƠN

Đề tài được hoàn thành với sự giúp đỡ ủng hộ rất lớn từ gia đình, thầy cô cũng như bạn bè thân thương. Đây là tình cảm tình nghĩa sâu đậm nhất mà không sao đền đáp hết được, dù trong hoàn cảnh tồi tệ nhất nhưng mọi người vẫn luôn động viên và tìm phường hướng giúp đỡ, thật sự đáng trân trọng.

Cảm ơn ba mẹ đã cùng con vượt qua những khó khăn, động viên con.

Cảm ơn các thầy cô trong bộ môn tự động trường đại học Bách Khoa đã hướng dẫn và giúp đỡ em rất nhiêu trong quá trình thực hiện luận văn.

Cảm ơn những người bạn chân thành đã hỗ trợ mình trong quá trình làm luận văn, chính nghĩa câu nói " học thầy không tày học bạn".

Qua đây em cũng nhận được nhiều ý kiến đóng góp tốt đẹp từ thầy cô trong hội đồng bảo vệ luận văn

Cuối cùng em xin chúc mọi người luôn thành công trong cuộc sống và luôn khỏe mạnh.

TÓM TẮT LUẬN VĂN

- ✓ Thực hiện đề tài " Thiết kế và điều khiển DELTA ROBOT"
- ✓ Thực hiện nghiên cứu đề tài về nguyên lý, khái niệm và điều khiển robot để di chuyển sản phẩm.
- ✓ Delta robot sẽ có giao diện điều khiển trên máy tính, có thể lập trình để điều khiển theo những điểm đã lấy trước và có thể điều khiển để sử dụng kèm với camera xử lý ảnh.
- ✓ Sử dụng vi điều khiển STM32 và phần mềm kiel c để nạp cho nó.
- ✓ Sử dụng visual studio 2010 để lập trình điều khiển từ máy tính.
- ✓ Sử dụng altium 14 để thế kế mạch điều khiển.

MỞ ĐẦU

Hiện nay, khoa học kĩ thuật đang rất phát triển nên nhu cầu nghiên cứu học tập cũng rất cao, đặc biệt trong các lĩnh vực như điện tử, tự động hóa…. Trong các lĩnh vực đó thì việc phát triển robot cũng như lập trình điều khiển ngày càng trở nên quan trọng trong cuộc sống. Việc tìm hiểu ứng dụng của vi xử lý và robot là mốt vnấ đề vô cùng lý thú cũng như nan giải, yêu cầu sự đầu tư mạnh dạn và phát triển lâu dài nhằm phát triển ra những sản phẩm đa dạng phong phú hổ trợ cuộc sống của con người.

Lựa chọn vi điều khiển STM32 cho vấn đề này là hợp lý với mô hình robot nhỏ phục vụ cho nhu cầu luận văn tốt nghiệp, cũng như sẽ có khả năng phục vụ cho nhu cầu thực tiễn nếu có them vài cải cách.

Quá trình làm việc với robot rủi ro rất lớn, đặc biệt là ở phần thiết kế cơ khi và mạch điều khiển. Tuy đã có rất nhiều cố gắng nhưng không thể tránh khỏi những sai sót, rủi ro trong quá trình thi công robot; mong quý thầy cô thông cảm và sự góp ý chân thành từ quý thầy cô cùng các bạn sinh viên.

Chương 1. TỔNG QUAN

1.1. Giới thiệu DELTA ROBOT

Thuật ngữ robot xuất hiện vào năm 1920 trong tác phẩm văn học của nhà văn có tên Karel Capek.

Thuật ngữ industrial robot (IR) xuất hiện đầu tiên do công ty mỹ AMF (American Machine and Foundry company) quảng cáo mô tả một thiết bị mang dáng dấp và có chức năng như tay người được điều khiển để thực hiện một số chức năng cụ thể. Từ đó nổi trội lên một xu thế phát triển robot mới phục vu đời sống con người.

Vào đầu thập niên 80, Reymond Clavel (giáo sư của EPFL) đã nảy ra một ý tưởng độc đáo là sử dụng cơ cấu hình bình hành để tạo ra một robot song song có ba bậc tự do tịnh tiến và một bậc tự do quay. Không như một số bài báo đã xuất bản đâu đó, ý tưởng này hoàn toàn là của Reymond Clavel chứ không phải bắt chước từ cơ cấu song song đã được Willard L. Polard đăng ký bản quyền vào năm 1942, và vào thời điểm đó Willard L. Polard cũng không hề biết đến giáo sư Clavel. Robot song song Delta đã được đánh giá là một trong những thiết kế robot song song thành công nhất với hàng trăm robot đang hoạt động trên toàn thế giới. Năm 1999, tiến sĩ Clavel đã nhận được giải thưởng Golden Robot Award được tài trợ bởi ABB Flexible Automation, để tôn vinh những hoạt động sáng tạo của ông về robot song song Delta.

Phần thân robot nằm trên vùng hoạt động và tất cả động cơ đều được lắp đặt trên đó. Từ phần thân robot , 3 cánh tay được đưa ra nối vào khớp cầu tạo thành hình tam giác đều. Việc xoay khớp Trên cùng sẽ truyền động làm dịch chuyển tịnh tiến các khớp cầu bên dưới theo 2 hướng.

Bởi vì tất cả động cơ đều nằm trên phần thân, nên các cánh tay đều được giải phóng khỏi khối lương của động cơ và có thể được làm từ các vật liệu nhẹ như nhựa hoặc nhôm hoặc sợi carbon

Việc sử dụng các bộ tác động gắn trên đế và các khâu có khối lượng nhẹ cho phép tấm dịch chuyển đạt được gia tốc lên đến 50 G trong phòng thí nghiệm và 12 G

trong các ứng dụng công nghiệp. chính điều này làm cho robot Delta trở thành một ứng cử viên sáng giá cho các hoạt động nâng – đặt đối với các đối tượng nhẹ (từ 10 gr đến 1 kg). Vùng làm việc của nó là sự giao nhau của 3 đường gờ tròn, nhưng robot Delta trên thị trường có thể hoạt động trong vùng làm việc hình trụ với đường kính là 1 m và có chiều cao là 0,2 m.

Hình 1.1.1: ABB delta robot

1.2. Tình hình phát triển robot

Lịch sử của robot Delta trên thương trường rất dài, phức tạp và vô cùng hấp dẫn. Mọi chuyện khởi nguồn từ năm 1983 khi mà hai anh em người Thụy Sĩ là marc-Olivier và Pascal Demaurex thành lập công ty Demaurex đóng tại Romanel-sur-Lausanne, Thụy Sĩ. Vào năm 1987, họ mua giấy phép sử dụng bản quyền robot Delta và đặt ra mục tiêu chính là thương mại hóa robot này vào ngành công nghiệp đóng gói. Sau vài năm, Demaurex đã thành công trong việc giữ vai trò trọng yếu trong thị trường mới mẻ đầy khó khăn này. Và họ cũng đã tiến hành một vài cải tiến sản phẩm của họ. Bốn phiên bản khác cũng đã được đưa ra thị trường với tên gọi là Pack-Placer, Line-Placer, Top-Placer và Presto. Đến thời điểm hiện nay, Demaurex tuyên bố đã bán được hơn 500 robot Delta trên toàn thế giới.

Hình 1.2.1: Robot hãng Demaurex đóng gói bánh quy.

Vào năm 1996, anh em nhà Demaurex đã mua bản quyền robot Delta từ EPFL. Tuy nhiên, trước khi thương vụ này diễn ra, EPFL củng đã bán 2 giấy phép sử dụng. Giấy phép đầu tiên là về loại robot có kích cỡ nhỏ (cánh tay và hình bình hành có chiếu dài nhỏ hơn 800 mm) được cấp cho Demaurex vào năm 1987. Giấy phép thứ hai về loại robot có kích cỡ lớn được bán cho AID và sau đó được bán lại cho DeeMed. Công ty này sau đó lại được mua lại bởi Elekta, một công ty Thụy Điển chuyên về lĩnh vực giải phẫu và sản xuất robot Delta dùng để nâng đỡ kính hiển vi có khối lượng lớn (20 kg), sản phẩm này có tên gọi là SurgiScope. Công nghệ robot Delta lại một lần nữa được bán cho Medtronic vào cuối năm 1999

Trước thương vụ SurgiScope, Elekta IGS đã tiến hành đàm phán về giấy phép sử dụng robot Delta của họ với hãng ABB. ABB cũng đã có được giấy phép sản xuất robot Delta với các kích cỡ lớn. Cùng thời gian này Demaurex công bố quyết định sản xuất robot có kích cỡ lớn hơn (khoảng từ 1.200 mm trở lên). Tuy vậy Demaurex có lẽ không sản xuất các robot có kích cỡ lớn hơn vì đã giao ước không thâm nhập vào thị trường robot có kích cỡ 1.200 mm trở lên mà chỉ độc quyền sản suất các loại có kích cỡ dưới 800 mm.

Demaurex cũng đã thỏa thuận với một công ty Đức tên là GROB-Werke về giấy phép sản xuất tên TRIAGLIDE 5g, cũng như với Mikron Technology Group về robot Triaglide, cả 2 robot này đều là loại robot Delta được trang bị động cơ tuyến

tính. Nhưng thú vị ở chỗ là có một sản phẩm đối chọi với robot Delta với hệ truyền động tuyến tính là máy xay 3 trục tọa độ Quickstep được cấp bản quyền cho hãng Krause & Mauser Group (bản quyền số US 6,161,992). Ngoài ra, Renault Automation Comau cũng đã giới thiệu tại EMO năm 1999 ở Pari về máy xay Urane SX, máy này có thiết kế tương tự như Quickstep.

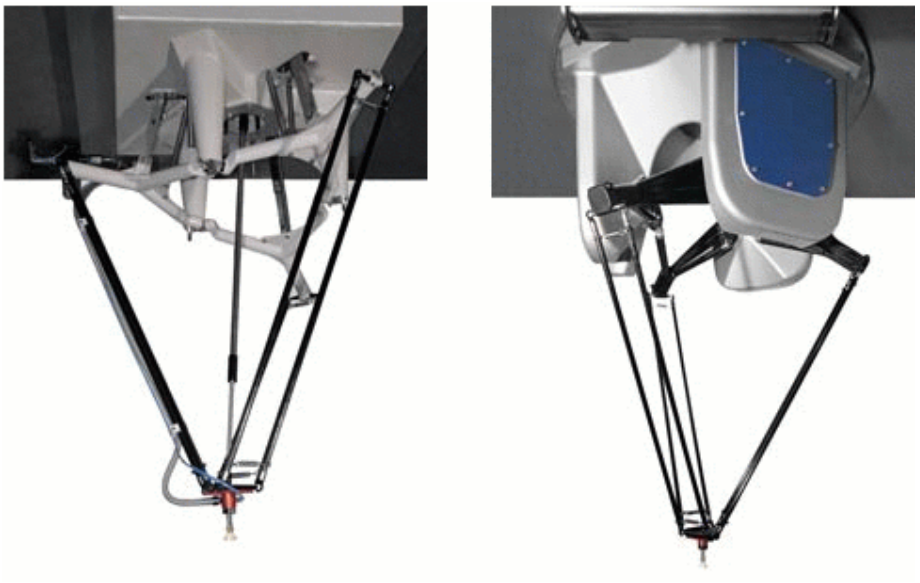
Demaurex cũng cấp giấy phép cho một công ty Nhật Bản có tên là Hitachi Seiki được quyền sản xuất robot Delta có kích cỡ nhỏ dùng để đóng gói (DELTA) và khoan (PA35). Trên thực tế, Hitachi Seiki là nhà đại diện của Demaurex tại Nhật Bản.

ABB Flexible Automation đã giới thiệu các robot Delta của mình vào năm 1999, đó là robot có tên IRB 340 FlaxPicker. Ba phân khúc thị trường mà họ hướng tới là các ngành công nghiệp thực phẩm, dược và điện tử. PlexPicker được trang bị hệ thống chân không được tích hợp luôn vào robot, có khả năng nhấc và nhà nhanh đối với các vật có khối lượng đến 1 kg. Robot được dẫn hướng bởi một thiết bị quan sát của hãng Cognex và được trang bị bộ điều khiển ABB S4C. Robot cũng có thể được trang bị một bộ điều khiển chuyển động và hệ thống quan sát của hãng Adept Technology. Vận tốc mà robot này đạt được khoảng 10 m/s và 3,6 deg/s (khoảng 150 lần nhấc trong một phút), và gia tốc lên đến 100 m/s2 và 1,2 rad/s2. Robot này có tới 2 phiên bản.

Sau hơn 15 năm giữ vai trò chủ đạo trên thị trường, Demaurex đột nhiên phải đối diện với quyết định của ABB gia nhập vào thị trường robot loại này. Demaurex đã thay đổi dòng sản phẩm từ các robot Delta rời rạc chuyển sang sản xuất các cụm robot hoàn chỉnh. Tuy nhiên, trong một nỗ lực nhằm bảo đảm sự ổn định lâu dài, hãng Demaurex nhỏ bé bắt đầu tìm kiếm đối tác. Và sau đó, Demaurex đã bắt tay hợp tác với một tập đoàn tại Thụy Sĩ là SIG vào cuối năm 1999.

SIG Group có 3 chi nhánh, mà chỉ với chi nhánh SIG Pack đã có hơn 2.000 nhân công - một công ty đủ lớn để hỗ trợ Demaurex nhiều mặt trong việc thâm nhập thị trường thế giới. Demaurex vẫn giữ tên SIG và thậm chí cả 2 nhà sáng lập và ban

lãnh đạo của SIG. Gần đây, 3 robot Delta khác cũng đã được SIG Pack Systems giới thiệu là C23, C33 do Demaurex sản xuất và CE33 do SIG Pack Systems sản xuất



Hình 1.2.2: SIG Pack Systems C33 và CE33.

Robot Delta không chỉ được ứng dụng trong công nghiệp mà còn được quan tâm nghiên cứu trong các phòng thí nghiệm. Rất nhiều biến thể của robot này đã được tạo ra, song hầu hết chúng đều gần với thiết kế gốc. Một trong những biến thể được tạo bởi đại học Genoa, trong thiết kế này cơ cấu hình bình hành được thay thế bằng các cơ cấu tương đương. Một phiên bản khác là robot NUWAR, được chế tạo tại đại họcWestern Australia. NUWAR được thiết kế để có thể đạt gia tốc 600m/s2 robot này khác biệt nhờ sự sắp xếp không đồng phẳng các trục của cơ cấu chấp hành. Phiên bản với ba động cơ tuyến tính cũng đã được tạo ra trong phòng thí nghiệm tạiFerdinand-von-Steinbeis Schule, ETH Zurich, và đại học Stuttgart.

Tất nhiên nơi robot Delta được nghiên cứu nhiều nhất và có nhiều biến thể nhất là tại nơi sinh ra chúng - EPFL.

Hiện nay trên thị trường robot thế giới robot Delta vẫn tỏ ra không có đối thủ cạnh tranh về tốc độ thực thi.

1.3. Nội dung và giới hạn đề tài

Yêu cầu của đề tài là thiết kế thành công và điều khiển được mô hình delta robot đơn giản, đồng thời để chứng tỏ tính thực thi của nó nên sẽ có them vài công cụ xử lý ảnh và bài tập gắp vật thể ví dụ để chứng minh.

Ở Việt Nam nói chung công nghệ chế tạo cơ khí chính xác còn chưa phát triển nên vấn đề tạo ra được phần cơ khí cho robot có thể thực thi được như sản phẩm trên thị trường là gần như không thể. Vì vậy nên ta tận dụng những gì có sẵn xung quanh để thiết kế thành công delta robot, trước hết là để ứng dụng làm phương tiện giảng dạy trong trường học và từ đó phát triển cao hơn để ứng dụng trong sản xuất đồng thời tiếp tục nghiên cứu và phát triển để cho ra những dòng sản phẩm vượt trội so với nước ngoài. Đề tài thiết kế và điều khiển delta robot không nằm ngoài những nhận định trên. Điều quan trọng hơn hết là các vấn đề lien quan đến việc tính toán thiết kế, chế tạo, nguyên lý hoạt động của robot phải được làm rõ. Nó sẽ là nguồn thong tin hữu ích cho những ai muốn tìm hiểu về lĩnh vực tự động hóa nói chung và lĩnh vực robot nói riêng.

Delta robot cũng khá đa dạng về nguồn gốc và chủng loại, đề tài được thực hiện trong điều kiện:

- ✓ Thời gian thực hiện đề tài chỉ trong một học kỳ
- ✓ Kinh nghiệm thực tế chưa có nhiều
- ✓ Tài liệu về lập trình robot còn hiếm
- ✓ Vật tư và linh kiện khó mua, tìm đồng thời cũng khó đồng bộ với nhau.
- ✓ Vì vậy em đã thực hiện đề tài nghiên cứu với đặc điểm chính sau đây:
- ✓ Thiết kế và xây dựng thành công mô hình delta robot
- ✓ Thiế kế và thi công toàn bộ mạch điện điều khiển
- ✓ Thiết kế và lập trình toàn bộ phần mềm từ lớp vi xử lý cho đến máy tính
- ✓ Xây dưng những chương trình ví dụ tiêu biểu

1.4. Phương pháp nghiên cứu

- ✓ Tìm hiểu về vi xử lý STM32 phần cứng và tập lệnh trong kiel c
- ✓ Thiết kế phần mạch điều khiển sử dụng vi xử lý STM2
- ✓ Lập trình điều khiển motor DC từ đơn giản đến phức tạp, tìm ra thong số tối ưu nhất có thể cho hoạt động của motor.
- ✓ Thực thi giao tiếp SPI và UART để test hoạt động của mạch
- ✓ Kiểm tra các phần còn lại của mạch đã hoạt động chính xác chưa

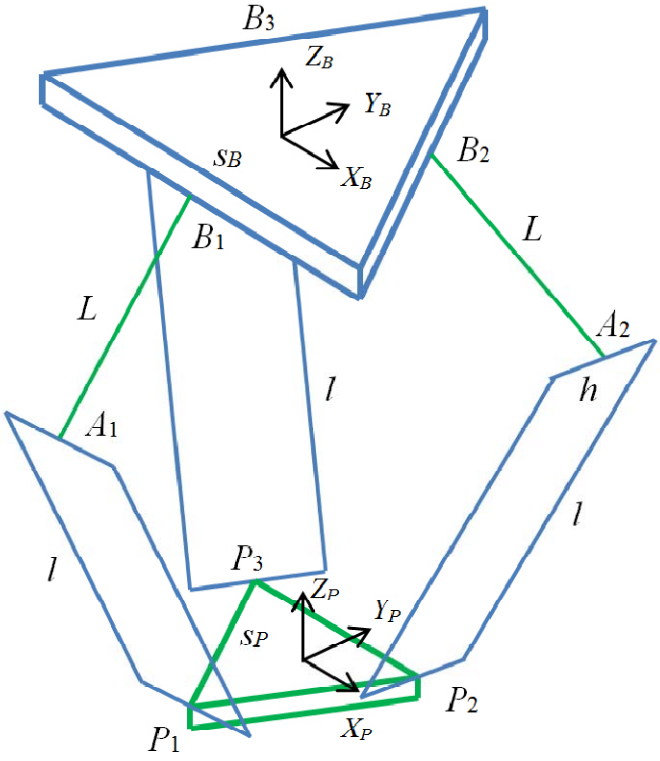
- ✓ Thiết kế phần cơ cho delta robot
- ✓ Láp ráp phần điện và phần cơ với nhau
- ✓ Hoàn thiện robot và lập trình giao diện điều khiển trên máy tính

Chương 2. CẤU TRÚC VÀ LÝ THUYẾT LIÊN QUAN ĐẾN HỆ THỐNG

2.1. Cấu trúc delta robot

Delta Robot gồm có:

- ✓ Một giá cố định. Trên giá cố định này người ta tạo ba thành phần khớp quay và có đường tâm quay kéo dài cắt nhau và tạo thành một tam giác đều, các đường trung trực của tam giác cắt nhau tại điểm O là tâm của giá cố định.
- ✓ Một bệ di động. Giá di động này chính là nơi mà ta sẽ gá đầu công tác lên đó.
- ✓ Mỗi chân được nối một đầu vào khớp phát động trên giá cố định (khớp quay) và đầu còn lại được nối vào bệ di động.
- ✓ Một nhánh của Delta Robot có kiểu R2S2S trong đó khớp R được chọn làm khớp phát động, khớp này có một thành phần khớp gắn cố định trên giá cố định và thành phần khớp còn lại gắn trên khâu phát động và nối với bệ di động thông qua hệ thống các khâu và khớp còn lại có kết cấu hình bình hành.

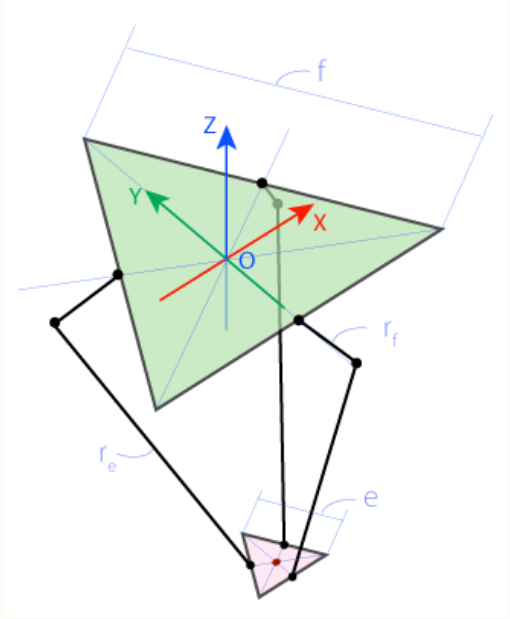


Hình 2.1.1: Cấu trúc delta robot

2.2. MÔ HÌNH ĐỘNG HỌC DELTA ROBOT

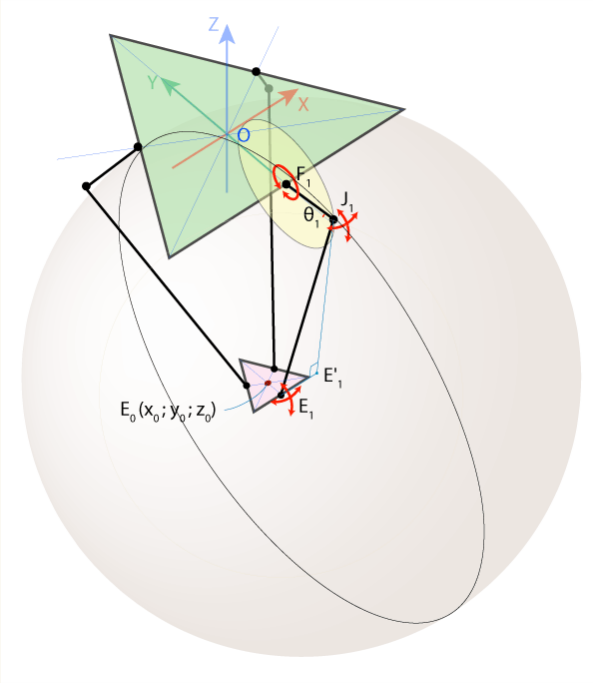
2.2.1. Động học ngược

- Đầu tiên ta đặt các giá trị hình học cho delta robot.
  - Cạnh tam giác đều của phần đế được đặt là f
  - Cạnh của tam giác đều phần end effector đặt là e
  - Độ dài của chân khớp trên là rf
  - Độ dài của chân khớp dưới là re



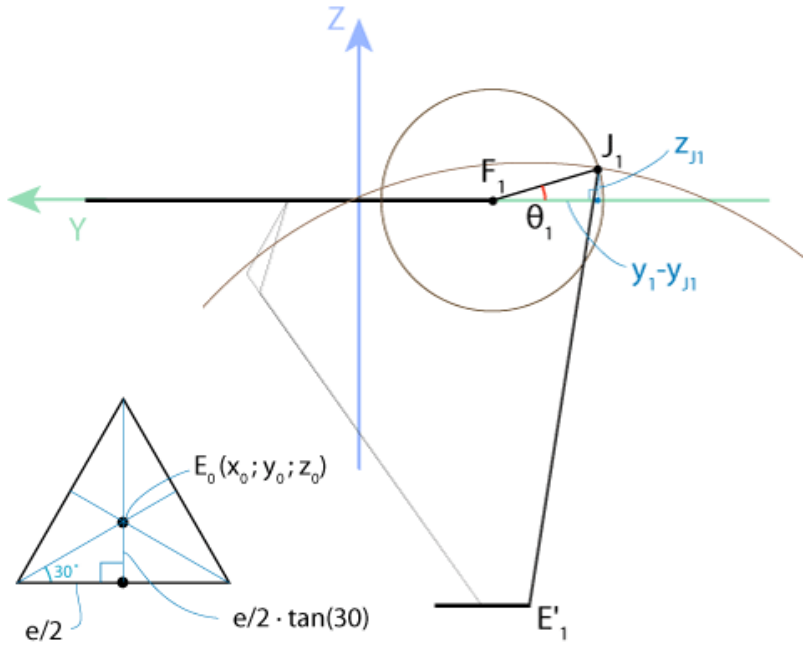
Hình 2.2.1: Sơ đồ delta robot

- Vì khớp F1J1 chỉ có thể di chuyển trong miền YZ (như hình dưới), nên tạo được 1 vùng hoạt động của tâm F1 với bán kính rf. Tương tự với E1 và J1 được gọi là universal joints, có nghĩa là E1J1 có thể di chuyển tự do dựa vào điểm tâm E1, tạo thành 1 vùng hoạt động hình cầu với tâm E1 và bán kính re.



Hình 2.2.2: Cấu trúc tính động học ngược

- Giao giữa hình cầu bán kính re và mặt phẫn YZ là 1 hình tròn tâm E'1 với bán kính E'1J1, với điểm E'1 là hình chiếu của điểm E1 lên mặt phẳng YZ.
- Khớp J1 trở thành diểm giao của 2 vòng tròn E'1 và F1 và khi đã tính được tọa độ J1 ta có thể suy ra được góc theta1
- Các phương trình tính toán trên mặt phẳn YZ như sau:



Hình 2.2.3: Phân tích hình học

$$E(x_0, y_0, z_0)$$

$$EE_1 = \frac{e}{2} tan30^0 = \frac{e}{2\sqrt{3}}$$

$$E_1\left(x_0, y_0 - \frac{e}{2\sqrt{3}}, z_0\right) \rightarrow E'_1\left(0, y_0 - \frac{e}{2\sqrt{3}}, z_0\right)$$

$$E_1E'_1 = x_0 \rightarrow E'_1J_1 = \sqrt{E_1J_1^2 - E_1E'^2_1} = \sqrt{r_e^2 - x_0^2}$$

$$f_0\left(0, -\frac{f}{2\sqrt{3}}, 0\right)$$

$$\begin{cases} (y_{J1} - y_{F1})^2 + (z_{J1} - z_{F1})^2 = r_f^2 \\ (y_{J1} - y'_E1)^2 + (z_{J1} - z'_E1)^2 = r_e^2 - x_0^2 \end{cases}$$

$$\rightarrow \begin{cases} \left(y_{J1} + \frac{f}{2\sqrt{3}}\right)^2 + {z_{J1}}^2 = r_f^2 \\ \left(y_{J1} - y_0 + \frac{e}{2\sqrt{3}}\right)^2 + (z_{J1} - z_0)^2 = r_e^2 - x_0^2 \end{cases}$$

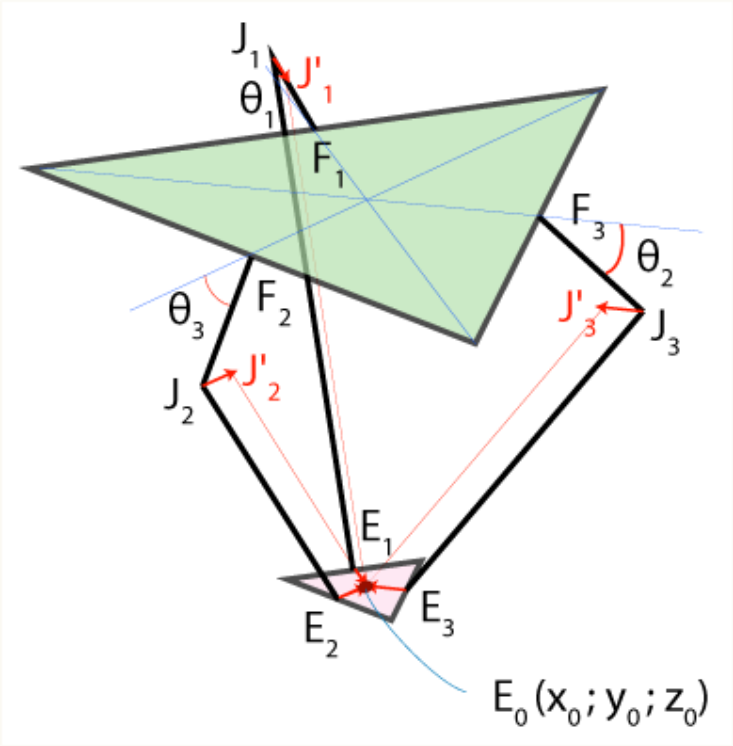
$$\rightarrow J_1(0, y_{J1}, z_{j1})$$

$$\boldsymbol{\theta_1 = artan(\frac{z_{J1}}{y_{f1} - y_{J1}})}$$

- Vì khớp F1J1 chỉ di chuyển theo trục YZ nên ta có thể loại bỏ hoàn toàn vai trò của tạo độ x trong phương trình.
- Tiếp theo để tính được theta2 và theta3 ta cũng dựa vào lợi thế đó cộng với cấu hình tam giác đều của delta robot.Đầu tiên ta xoay trục tọa độ của mặt phẳng XY trên trục Z 1 góc 120 độ ngược chiều kim đồng hồ
- Từ đó ta có hệ tọa độ mới X'Y'Z' và với hệ tọa độ mới ta có thể tính được theta2 với phương trình đã tính cho theta1. Công việc duy nhất ta phải làm khác là tính lại tọa độ X'0 và Y'0 cho điểm E0 mà có thể có được 1 cách dễ dàng bằng cách sử dụng ma trận xoay.
- Tương tự ta có thể tính được theta3 cũng với phương trình trên và xoay mặt XY 120 độ cùng chiều kim dồng hồ.

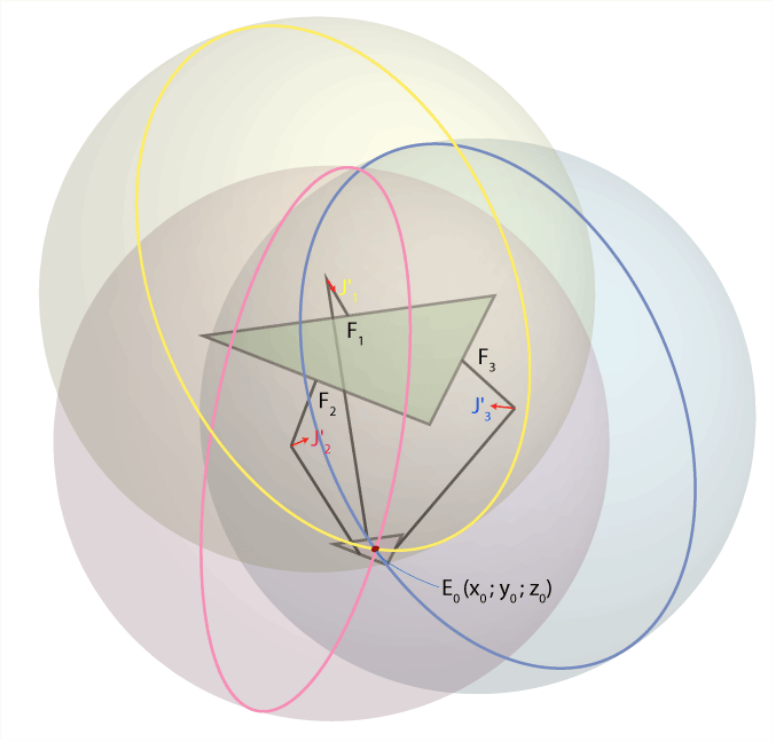
2.2.2. Động học thuận

- Với cấu trúc động học thuận thì ta đã biết trước được góc theta1,theta2, theta3 việc còn lại là tìm tọa độ x0,y0,z0 của E0.
- Khi đã biết được theta, ta có thể dễ dàng tìm được tọa độ J1,J2,J3
- Các khớp J1E1,J2E2,J3E3 có thể xoay 1 cách tự do quanh điểm J1,J2,J3 tạo nên hình cầu bán kính re



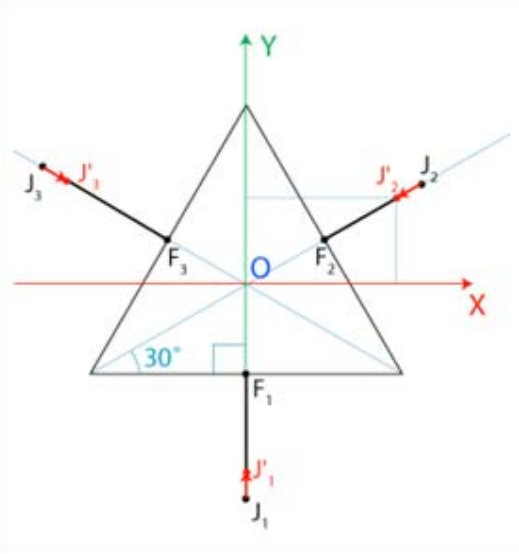
Hình 2.2.4: Cấu trúc động học thuận

- Giả sử ta di chuyển tâm của hình cầu tại điểm J1,J2,J3 tới điển J'1 J'2 j'3 sử dụng vectơ E1E0, E2E0,E3E0
- Sau khi di chuyển thì cả 3 hình cầu sẽ có chung 1 giao điểm tại E0 như hình sau:



Hình 2.2.5: Vùng không gian động học thuận

- Vì vậy để tính được tọa độ điểm E0 ta sẽ giải 3 phương trình như $(x-xj)^2 +(y-yj)^2 +(z-zj)^2 = re^2$. Với tọa độ tâm hình cầu xj,yj,zj và bán kính hình cầu re đã biết
- Đầu tiên ta phải tìm tọa độ J'1,J'2,J'3 như sau:



Hình 2.2.6: Cấu trúc tam giác trên

$$OF_1 = OF_2 = OF_3 = \frac{f}{2}tan30 = \frac{f}{2\sqrt{3}}$$

$$J_1J'_1 = J_2J'_2 = J_3J'_3 = \frac{e}{2}tan30 = \frac{e}{2\sqrt{3}}$$

$$F_1J_1 = r_f cos\theta_1, F_2J_{21} = r_f cos\theta_2, F_3J_3 = r_f cos\theta_3$$

$$J'_1(0; \frac{(-f+e)}{2\sqrt{3}} - r_f cos\theta_1; -r_f sin\theta_1)$$

$$J'_2(\left(\frac{(f-e)}{2\sqrt{3}} + r_f cos\theta_2\right)cos30; \left(\frac{(f-e)}{2\sqrt{3}} - r_f cos\theta_2\right)sin30; -r_f sin\theta_2)$$

$$J'_2(\left(\frac{(f-e)}{2\sqrt{3}} + r_f cos\theta_3\right)cos30; \left(\frac{(f-e)}{2\sqrt{3}} - r_f cos\theta_3\right)sin30; -r_f sin\theta_3)$$

- Từ phương trình trên ta có thể tính ra được tạo độ của J1, J2, J3 là (x1, y1, z1), (x2, y2, z2) và (x3, y3, z3)
- Thế vào phương trình giao điểm của 3 hình cầu ta được:

$$\begin{cases} x^2 + (y - y_1)^2 + (z - z_1)^2 = r_e^2 \\ (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = r_e^2 \\ (x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 = r_e^2 \end{cases}$$

$$\rightarrow \begin{cases} x^2 + y^2 + z^2 - 2y_1 y - 2z_1 z = r_e^2 - y_1^2 - z_1^2 \ (1) \\ x^2 + y^2 + z^2 - 2x_2 x - 2y_2 y - 2z_2 z = r_e^2 - y_2^2 - z_2^2 - x_2^2 \ (2) \\ x^2 + y^2 + z^2 - 2x_3 x - 2y_3 y - 2z_3 z = r_e^2 - y_3^2 - z_3^2 - x_3^2 \ (3) \end{cases}$$

$$w_i = x_i^2 + y_i^2 + z_i^2$$

$$\begin{cases} x_2 x + (y_1 - y_2)y + (z_1 - z_2)z = \dfrac{w_1 - w_2}{2} \ (4) = (1) - (2) \\ x_3 x + (y_1 - y_3)y + (z_1 - z_3)z = \dfrac{w_1 - w_3}{2} \ (5) = (1) - (3) \\ (x_2 - x_3)x + (y_2 - y_3)y + (z_2 - z_3)z = \dfrac{w_2 - w_3}{2} \ (6) = (2) - (3) \end{cases}$$

Từ (4) –(5):

$x = a_1 z + b_1$ (7) $\quad y = a_2 z + b_2$ (8)

$$a_1 = \frac{1}{d}[(z_2 - z_1)(y_3 - y_1) - (z_3 - z_1)(y_2 - y_1)]$$

$$a_2 = \frac{-1}{d}[(z_2 - z_1)x_3 - (z_3 - z_1)x_2]$$

$$b_1 = \frac{-1}{2d}[(w_2 - w_1)(y_3 - y_1) - (w_3 - w_1)(y_2 - y_1)]$$

$$b_2 = \frac{-1}{2d}[(w_2 - w_1)x_3 - (w_3 - w_1)x_2]$$

$$d = (y_2 - y_1)x_3 - (y_3 - y_1)x_2$$

Lấy (7)-(8) thay vào (1):

$$(a_1^2 + a_2^2 + 1)z_1 + 2(a_1 + a_2(b_2 - y_1) - z_1)z + (b_1^2 + (b_2 - y_1)^2 + z_1^2 - r_e^2) = 0$$

- Cuối cùng ta có thể tính được tọa độ Z0 dựa vào phương trình trên rồi từ đó suy ra tọa độ X0,Y0.

2.3. Mạch điều khiển delta robot

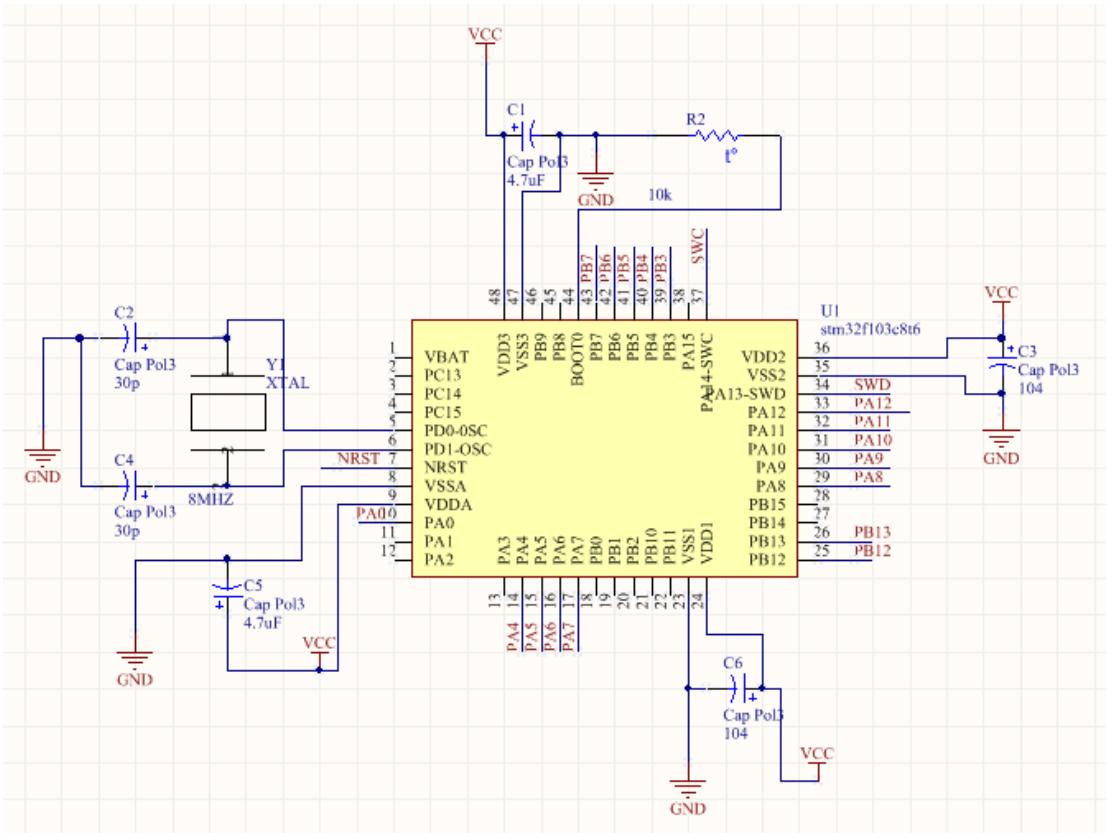
2.3.1. Sơ lược vi xử lý STM32F103C8T6

- Vi sử lý sử dụng con STM32F103C8T6 72MHz – chân 48 LQFP – 37 IO - 64KB flash - 20K RAM
- Sử dụng thạch anh ngoài 8MHz
- Nạp chương trình bằng STLink debuger
- Memories
  - 64 or 128 Kbytes of Flash memory
  - 20 Kbytes of SRAM
- Clock, reset and supply management
  - 2.0 to 3.6 V application supply and I/Os
  - POR, PDR, and programmable voltage detector (PVD)
  - 4-to-16 MHz crystal oscillator
  - Internal 8 MHz factory-trimmed RC
  - Internal 40 kHz RC
  - PLL for CPU clock
  - 32 kHz oscillator for RTC with calibration
  - 2 x 12-bit, 1 µs A/D converters (up to 16 channels)
  - Conversion range: 0 to 3.6 V
  - Dual-sample and hold capability
  - Temperature sensor
- DMA
  - 7-channel DMA controller Peripherals supported: timers, ADC, SPIs, I2Cs and USARTs
- Up to 80 fast I/O ports
- 7 timers
  - Three 16-bit timers, each with up to 4 IC/OC/PWM or pulse counter and quadrature (incremental) encoder input
  - 16-bit, motor control PWM timer with deadtime generation and emergency stop
  - 2 watchdog timers (Independent and Window)
  - SysTick timer 24-bit downcounter
- Up to 9 communication interfaces
  - Up to 2 x I2C interfaces (SMBus/PMBus)
  - Up to 3 USARTs (ISO 7816 interface, LIN, IrDA capability, modem control)
  - Up to 2 SPIs (18 Mbit/s)

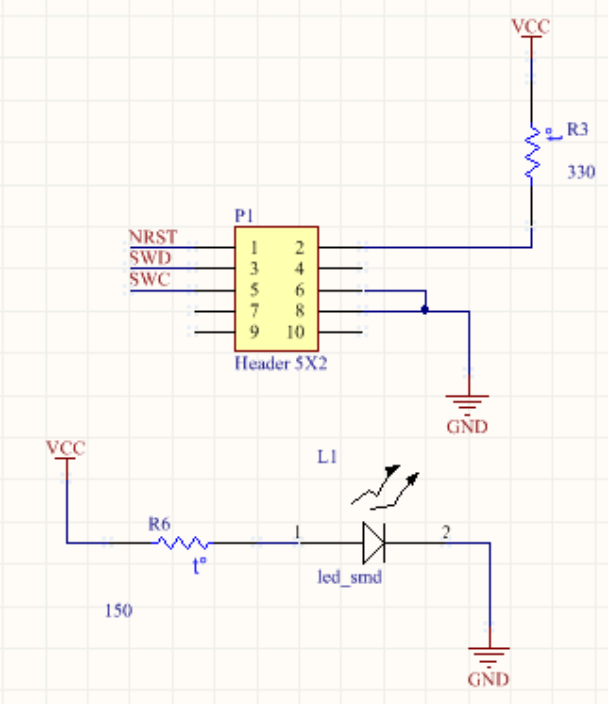
- CAN interface (2.0B Active)
- USB 2.0 full-speed interface

Hình 2.3.1: Vi xử lý STM32F103

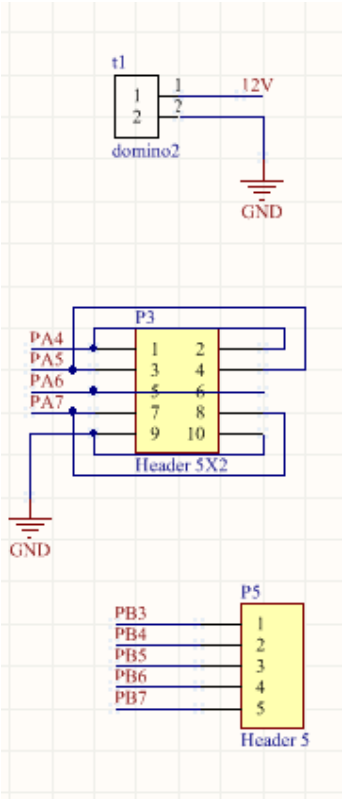
2.3.2. Mạch điều khiển SLAVE



Hình 2.3.2:sơ đồ chân ra vi xử lý

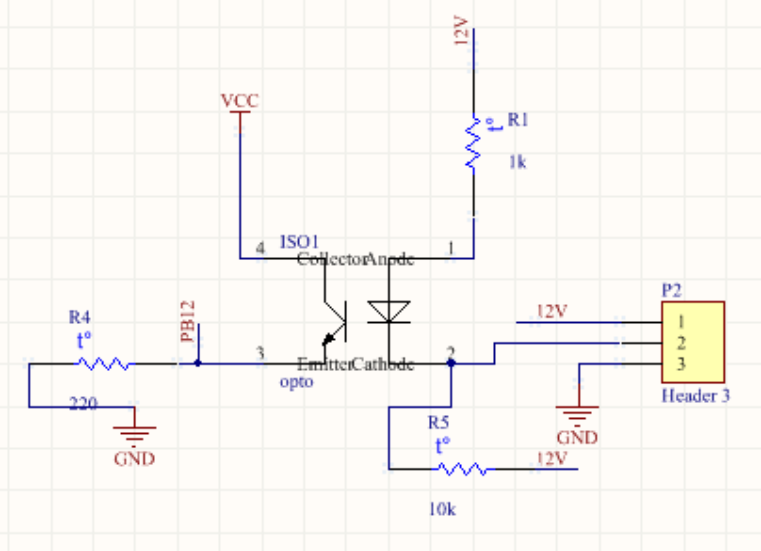


Hình 2.3.3: Sơ đồ jack cắm mạch nạp



Hình 2.3.4: Sơ đồ chân nối SPI

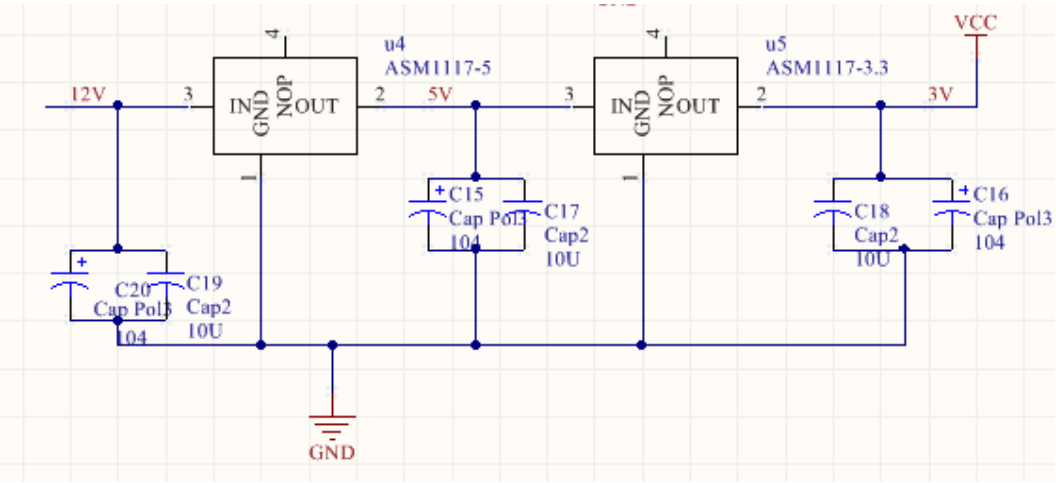
- Module giao tiếp cảm biến tiệm cận phát hiện kim loại:
  - ✓ Sử dụng cảm biến tiệm cận phát hiện kim loại LJ18A3 khoảng cách phát hiện 8MM
  - ✓ Sử dụng opto để cách ly tín hiệu cảm biến 12V và ngỏ vào vi xử lý 3.3V



Hình 2.3.5: Sơ đồ đọc cảm biến tiệm cận

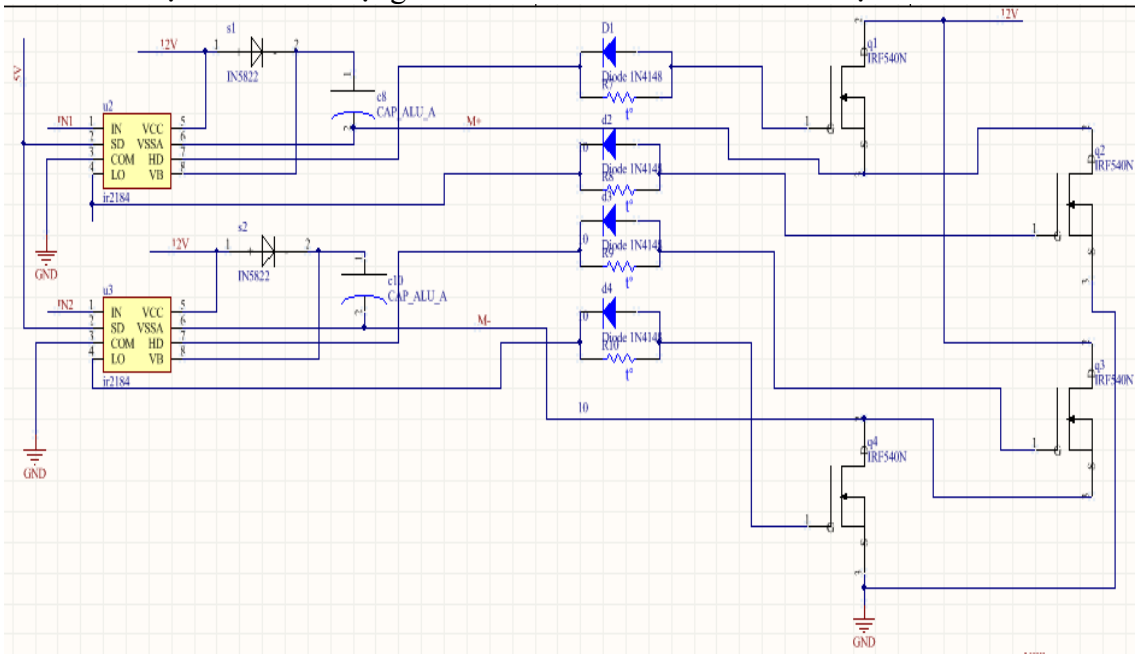
- Module ổn áp nguồn vào:

Sử dụng 2 IC ổn áp Asm1117-3.3 và ASM1117-5 để ra áp 3.3 và 5v cho toàn mạch



Hình 2.3.6: Sơ đồ ổn áp nguồn

- Module driver điều khiển động cơ:
  - ✓ Sử dụng IC cổng NOR để tính logic ngõ vào PWM và DIR
  - ✓ Sử dụng IC driver IR2184 để điều khiển 2 MOSFET kênh N trên cùng 1 nhánh
  - ✓ Sử dụng MOSFET kênh N IRF540
  - ✓ Ngoài ra còn có nhiều phần tử khác như diode schottky, diode bảo vệ ngược áp…
- Mạch cầu H sử dụng driver ir2184 và Mosfet irf540 hoặc irf3205



Hình 2.3.7: Sơ đồ mạch cầu H

Tụ bootstrap sử dụng là 1 tụ hóa 10uf và 1 tụ ceramic 104 và diode bootstrap sử dụng là schottky mbrs340t3.

Mạch bootstrap cho ir2184 bao gồm 1 diode và 1 tụ boostrap. Diode nên là loại có thời gian hồi phục nhanh và tụ bootstrap nếu sử dụng tụ electrolytic thì nên có 1 tụ ceramic thêm vào còn nếu là tụ Tantalum thì không cần.

Giải thích về mạch bootstrap: Khi chân IN =0 thì chân Lo được kích, làm cho con fet dưới dẫn, Vs nối xuống mass và tụ boot sẽ được nạp lên tới giá trị trên

chân Vb. Khi IN=1 thì điện áp đã được nạp trên tụ sẽ được dùng để tạo chênh lệch áp giữa 2 cực G và S của con fet ở trên, lúc đó Vb sẽ được nối vào Ho. Tụ bootstrap nên được chọn có giá trị điện dung đủ lớn để cung cấp cho toàn bộ thời gian khi fet trên được kích, nhưng nó cũng không nên lớn quá vì thời gian nạp vào tụ sẽ chậm. Vì vậy nên tần số càng thấp thì giá trụ tụ sẽ càng lớn và ngược lại.

Các điện trở và diode ở cực G của mỗi mosfet dùng để điều chỉnh thời gian lên và xuống của fet.

Điện trở thì có tác dụng hạn dòng ở cực gate lại.Vì giữa 2 cực G và S của fet có tính chất như 1 tụ điện, khi tụ nạp đầy thì fet đóng, và khi tụ xả hết thì fet mở. Điện trở ở cực gate sẽ làm dòng nạp vào tụ nhỏ lại và vì vậy fet sẽ đóng mở chậm hơn. Tại sao lại muốn đóng fet chậm hơn, vì khi chuyển mạch quá nhanh thì giữa 2 đầu bản cực sẽ sinh ra điện áp do quá độ khá lớn, có thể gấp đôi điện áp nguồn nên thường sẽ là fet chết đứng. Vậy sao không chọn điện trở này thật lớn, vì điện trở quá lớn khiến fet đóng chậm, mà trong thời gian lên đó thì fet chưa đóng hoàn toàn và bão hòa để đi đến trở kháng dẫn thấp nhất, trong lúc đó thì fet vẫn dẫn với trở kháng lớn sẽ làm fet nóng lên nhanh chóng và chết cháy. Vì vậy ở đây khuyến cáo sử dụng trở ở cực G trong khoảng từ 1->10 ohm

Do thêm điện trở để làm fet đóng mạch nhanh hơn, nhưng ta lại muốn fet mở mạch càng nhanh càng tốt nên vì vậy phải thêm con diode ngược ngạo ở đó để dẫn dòng xả ra của tụ con fet khi nó mở.

Còn điện trở giá trị 1k nối từ cực G sang cực S của mỗi con fet thực ra là để đảm bảo khi mình không kích cực G thì áp trên cực G sẽ bằng áp trên cực S, vì có 1 số trường hợp nhiễu gì gì đó làm cực G có 1 chút điện áp, khiến cho fet dẫn và làm nó cháy không rõ nguyên do.

2.3.3. Mạch điều khiển MASTER

Có nhiệm vụ nhận dữ liệu từ SLAVE và truyền lên cho máy tính thong qua UART đồng thời nhận lệnh từ máy tính truyền xuống lại cho slave.

Mạch có 3 luồng IO để kết nối SPI với 3 SLAVE đồng thời có gắn thiết bị uart – USB để giao tiếp với máy tính

MASTER còn làm nhiệm vụ đóng mở gripper là nam châm điện sử dụng điện 12VDC, để đóng ngắt nam châm này mạch sử dụng 1 opto và 1 con N channel MOSFET.

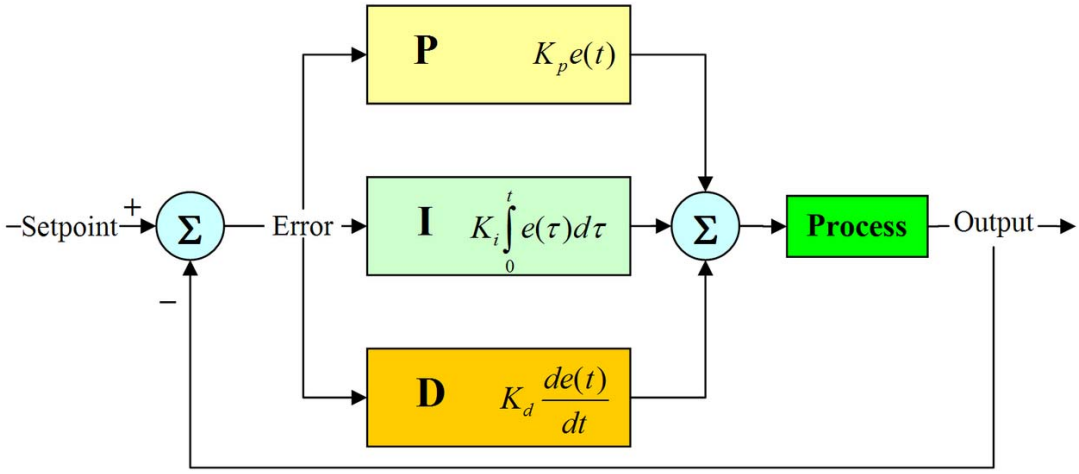
Mạch MASTER cũng sử dụng vi xử lý STM32F103C8T6 cùng thạch anh 8MHz

Mạch cũng có hệ thống ổn áp riêng cấp nguồn 5V và 3V cho mạch từ nguồn 12V đầu vào

2.4. Thiết kế giải thuật điều khiển cho motor DC

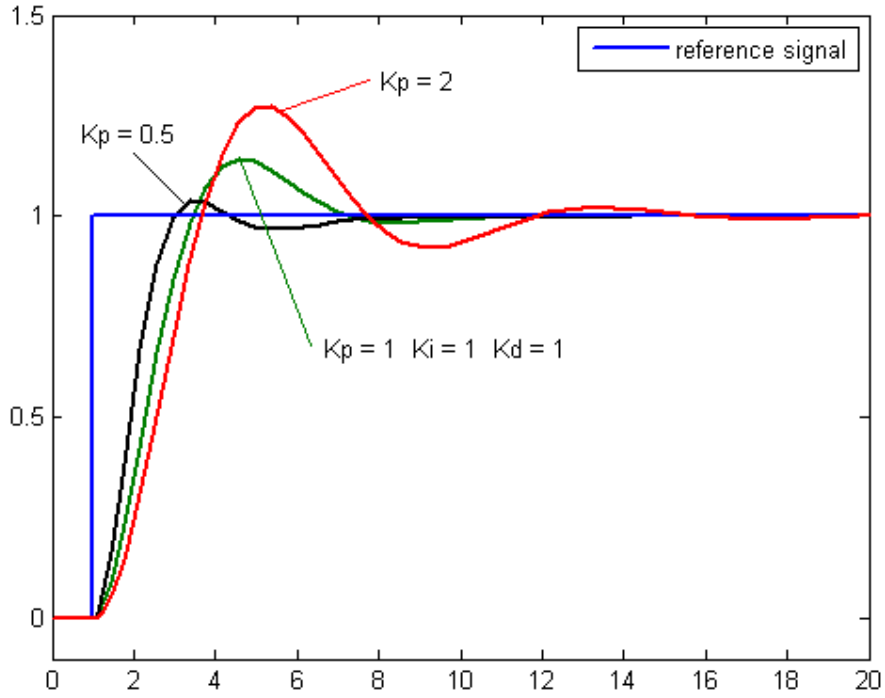
2.4.1. PID

Một **bộ điều khiển vi tích phân tỉ lệ** (**bộ điều khiển PID- Proportional Integral Derivative**) là một bộ điều khiển tổng quát được sử dụng rộng rãi trong các hệ thống công nghiệp – bộ điều khiển PID được sử dụng phổ biến nhất trong số các bộ điều khiển phản hồi. Một bộ điều khiển PID tính toán một 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. Trong trường hợp không có kiến thức cơ bản về quá trình, bộ điều khiển PID là bộ điều khiển tốt nhất. Tuy nhiên, để đạt được kết quả tốt nhất, các thông số PID sử dụng trong tính toán phải điều chỉnh theo tính chất của hệ thống-trong khi kiểu điều khiển là giống nhau, các thông số phải phụ thuộc vào đặc thù của hệ thống.



Hình 2.4.1: PID controller

Khâu tỉ lệ



Hình 2.4.2: Đồ thị PV theo thời gian K_p (K_i và K_d là hằng số)

Khâu tỉ lệ (đôi khi còn được gọi là *độ lợi*) làm thay đổi giá trị đầu ra, tỉ lệ với giá trị sai số hiện tại. Đáp ứng tỉ lệ có thể được điều chỉnh bằng cách nhân sai số đó với một hằng số K_p, được gọi là độ lợi tỉ lệ.

Khâu tỉ lệ được cho bởi:

$$P_{\text{out}} = K_p\, e(t)$$

trong đó

P_{out}: thừa số tỉ lệ của đầu ra

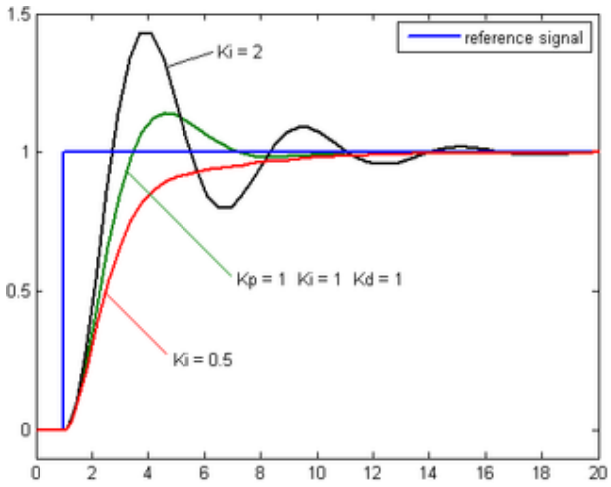
K_p: Độ lợi tỉ lệ, thông số điều chỉnh

e: sai số $= SP - PV$

t: thời gian hay thời gian tức thời (hiện tại)

Độ lợi của khâu tỉ lệ lớn là do thay đổi lớn ở đầu ra mà sai số thay đổi nhỏ. Nếu độ lợi của khâu tỉ lệ quá cao, hệ thống sẽ không ổn định. Ngược lại, độ lợi nhỏ là do đáp ứng đầu ra nhỏ trong khi sai số đầu vào lớn, và làm cho bộ điều khiển kém nhạy, hoặc đáp ứng chậm. Nếu độ lợi của khâu tỉ lệ quá thấp, tác động điều khiển có thể sẽ quá bé khi đáp ứng với các nhiễu của hệ thống.

Khâu tích phân



Hình 2.4.3: Đồ thị PV theo thời gian K_i (K_p và K_d không đổi)

Đồ thị PV theo thời gian, tương ứng với 3 giá trị K_i (K_p và K_d không đổi)

Phân phối của khâu tích phân (đôi khi còn gọi là *reset*) tỉ lệ thuận với cả biên độ sai số lẫn quảng thời gian xảy ra sai số. Tổng sai số tức thời theo thời gian (tích phân sai số) cho ta tích lũy bù đã được hiệu chỉnh trước đó. Tích lũy sai số sau đó được nhân với độ lợi tích phân và cộng với tín hiệu đầu ra của bộ điều khiển. Biên độ phân phối của khâu tích phân trên tất cả tác động điều chỉnh được xác định bởi độ lợi tích phân, K_i.

Thừa số tích phân được cho bởi:

$$I_{\text{out}} = K_i \int_0^t e(\tau)\, d\tau$$

trong đó

I_{out}: thừa số tích phân của đầu ra

K_i: độ lợi tích phân, 1 thông số điều chỉnh

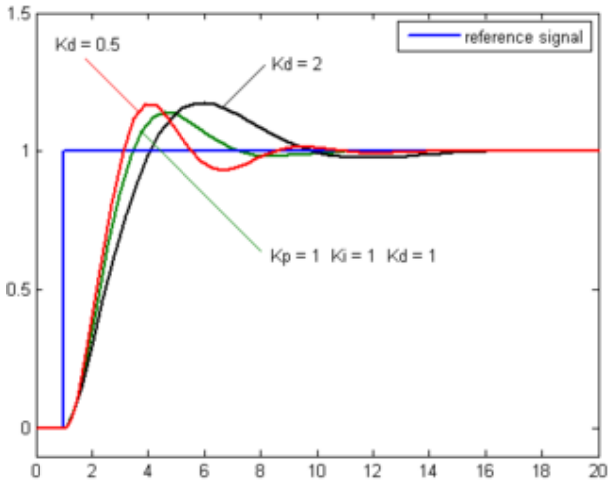
e: sai số $= SP - PV$

t: thời gian hoặc thời gian tức thời (hiện tại)

τ: một biến tích phân trung gian

Khâu tích phân (khi cộng thêm khâu tỉ lệ) sẽ tăng tốc chuyển động của quá trình tới điểm đặt và khử số dư sai số ổn định với một tỉ lệ chỉ phụ thuộc vào bộ điều khiển. Tuy nhiên, vì khâu tích phân là đáp ứng của sai số tích lũy trong quá khứ, nó có thể khiến giá trị hiện tại vọt lố qua giá trị đặt (ngang qua điểm đặt và tạo ra một độ lệch với các hướng khác). Để tìm hiểu thêm các đặc điểm của việc điều chỉnh độ lợi tích phân và độ ổn của bộ điều khiển.

Khâu vi phân



Hình 2.4.4: Đồ thị PV theo thời gian K_d (K_p and K_i không đổi)

Đồ thị PV theo thời gian, với 3 giá trị K_d (K_p and K_i không đổi)

Tốc độ thay đổi của sai số qua trình được tính toán bằng cách xác định độ dốc của sai số theo thời gian (tức là đạo hàm bậc một theo thời gian) và nhân tốc độ này với độ lợi tỉ lệ K_d. Biên độ của phân phối khâu vi phân (đôi khi được gọi là *tốc độ*) trên tất cả các hành vi điều khiển được giới hạn bởi độ lợi vi phân, K_d.

Thừa số vi phân được cho bởi:

$$D_{\text{out}} = K_d \frac{d}{dt} e(t)$$

trong đó

D_{out}: thừa số vi phân của đầu ra

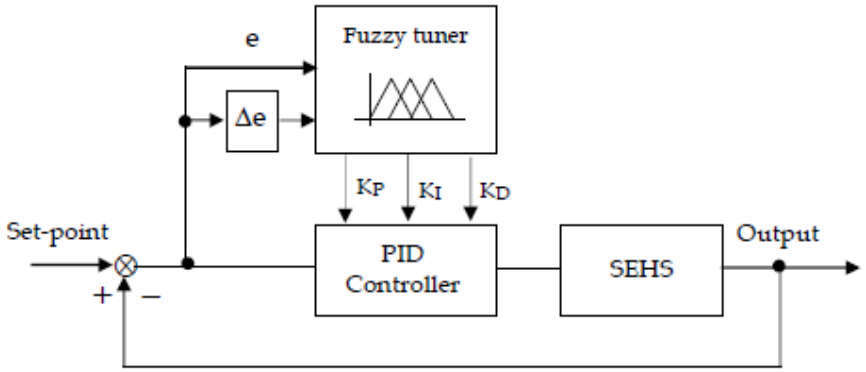
K_d: Độ lợi vi phân, một thông số điều chỉnh

e: Sai số $= SP - PV$

t: thời gian hoặc thời gian tức thời (hiện tại)

Khâu vi phân làm chậm tốc độ thay đổi của đầu ra bộ điều khiển và đặc tính này là đang chú ý nhất để đạt tới điểm đặt của bộ điều khiển. Từ đó, điều khiển vi phân được sử dụng để làm giảm biên độ vọt lố được tạo ra bởi thành phần tích phân và tăng cường độ ổn định của bộ điều khiển hỗn hợp. Tuy nhiên, phép vi phân của một tín hiệu sẽ khuếch đại nhiễu và do đó khâu này sẽ nhạy hơn đối với nhiễu trong sai số, và có thể khiến quá trình trở nên không ổn định nếu nhiễu và độ lợi vi phân đủ lớn. Do đó một xấp xỉ của bộ vi sai với băng thông giới hạn thường được sử dụng hơn. Chẳng hạn như mạch bù sớm pha.

2.4.2. PID Fuzzy self tunning



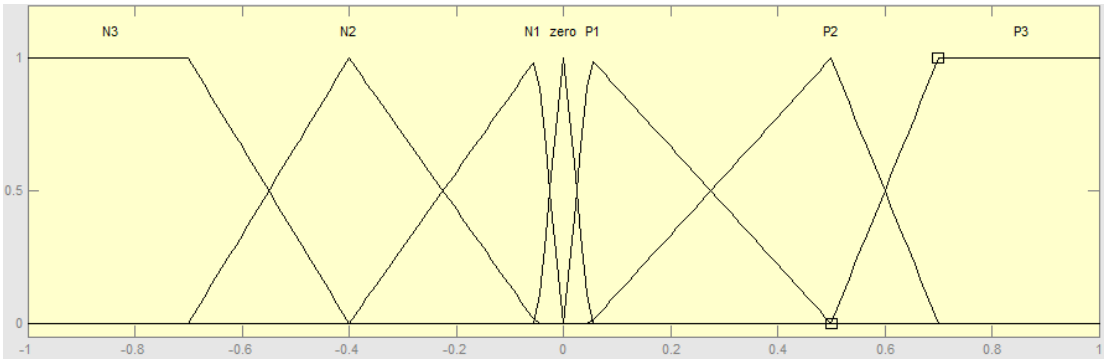
Hình 2.4.5: PID fuzzy seft tunning

Phương pháp này sử dụng 2 bộ điều khiển chạy song song nhau, bộ điều khiển PD Fuzzy sẽ lấy thông số nhiệt độ và độ thay đổi nhiệt độ để cho ra thông số Kp, Ki, Kd tương ứng, từ đó bộ điều khiển PID sẽ tiếp tục điều khiển đối tượng.

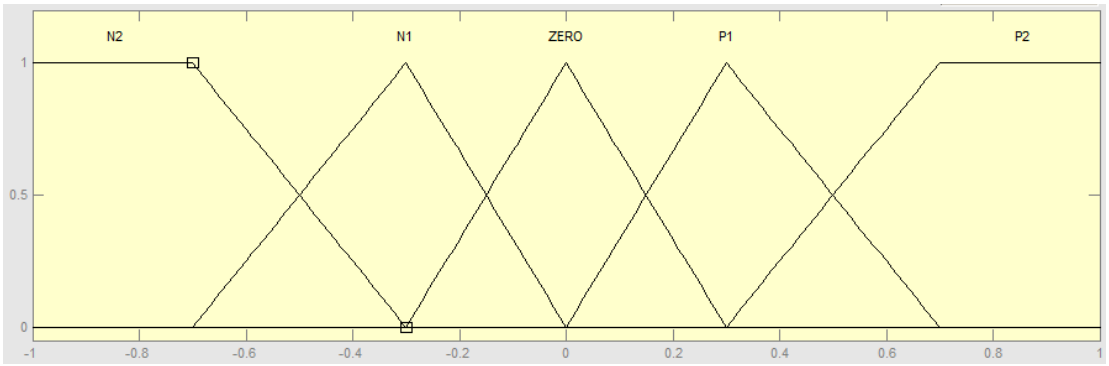
Do đặc điểm tùy chỉnh các thông số PID nên bộ điều khiển này có khả năng thích ứng hoàn hảo, cả khả năng đáp ứng thời gian lẫn khả năng chịu tải, nhiễu từ môi trường đều rất tốt, nhưng nó vẫn còn 1 khuyết điểm là các luật fuzzy điều khiển tùy chỉnh bằng kinh nghiệm của người thiết kế, bằng phương pháp

thử sai là chủ yếu nên các luật được thiết kế cho bộ điều khiển chưa phải là tối ưu.

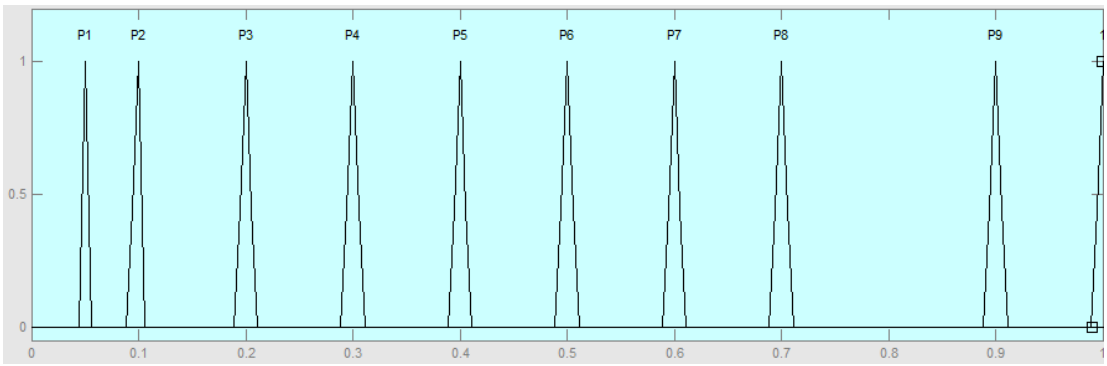
Phương pháp này cho khả năng điều khiển tối ưu nhất nhưng nên kết hợp với 1 phương pháp tối ưu hóa các bộ luật cho fuzzy như Nelder Mead chẳng hạn, có thể sẽ cho ra bộ điều khiện tự động hoàn toàn và có thông số hạt động chắn chắn nhất, đáng tin cậy được.



Hình 2.4.6: Ngõ vào E



Hình 2.4.7: Ngõ vào DE



Hình 2.4.8: Ngõ ra U

Kp=1+30*U		E						
		N3	N2	N1	ZERO	P1	P2	P3
DE	N2	P9	P7	P6	P2	P2	P3	P5
	N1	P8	P6	P5	P1	P3	P4	P6
	ZERO	P7	P5	P4	P0	P4	P5	P7
	P1	P6	P4	P3	P1	P5	P6	P8
	P2	P5	P3	P2	P2	P6	P7	P9

Bảng 2.4.1: bảng luật cho kp

Ki= U		E						
		N3	N2	N1	ZERO	P1	P2	P3
DE	N2	P9	P7	P7	P2	P3	P3	P5
	N1	P8	P6	P6	P1	P4	P4	P6
	ZERO	P7	P5	P5	P0	P5	P5	P7
	P1	P6	P4	P4	P1	P6	P6	P8
	P2	P5	P3	P3	P2	P7	P7	P9

Bảng 2.4.2: bảng luật cho Ki

2.5. Thiết kế giao thức truyền thông

2.5.1. Giao thức SPI

SPI (Serial Peripheral Bus) là một chuẩn truyền thông nối tiếp tốc độ cao do hang Motorola đề xuất. Đây là kiểu truyền thông Master-Slave, trong đó có 1 chip Master điều phối quá trình tuyền thông và các chip Slaves được điều khiển bởi Master vì thế truyền thông chỉ xảy ra giữa Master và Slave. SPI là một cách truyền song công (full duplex)

nghĩa là tại cùng một thời điểm quá trình truyền và nhận có thể xảy ra đồng thời. SPI đôi khi được gọi là chuẩn truyền thông "4 dây" vì có 4 đường giao tiếp trong chuẩn này đó là SCK (Serial Clock), MISO (Master Input Slave Output),

MOSI (Master Ouput Slave Input) và SS (Slave Select). Hình 1 thể hiện một kết SPI giữa một chip Master và 3 chip

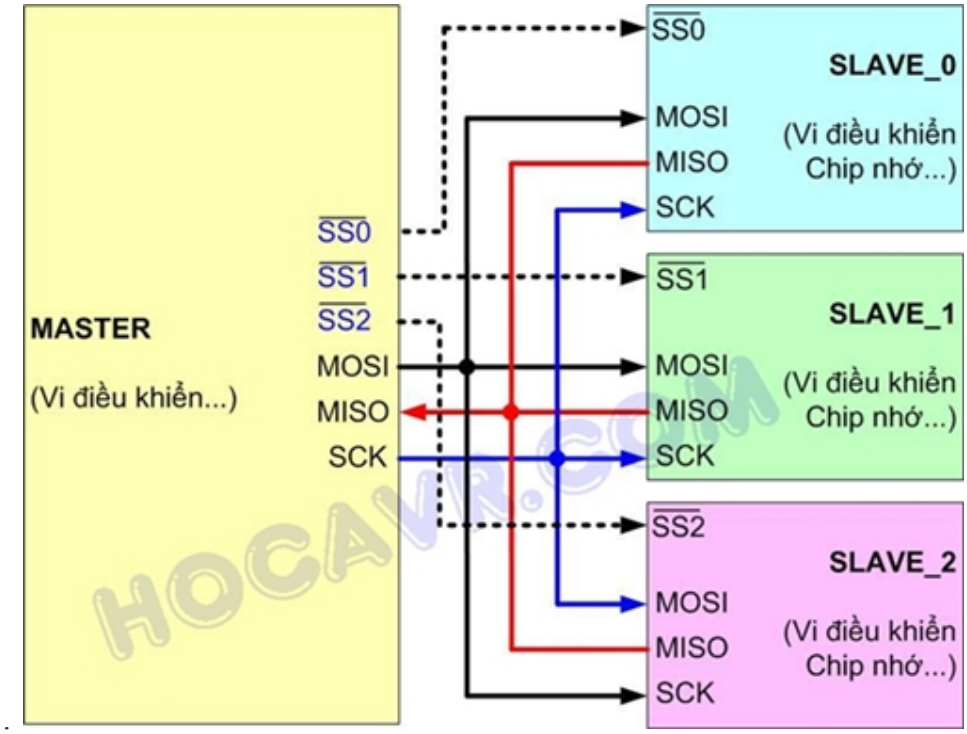
Slave thông qua 4 đường.

SCK: Xung giữ nhịp cho giao tiếp SPI, vì SPI là chuẩn truyền đồng bộ nên cần 1 đường giữ nhịp, mỗi nhịp trên chân SCK báo 1 bit dữ liệu đến hoặc đi. Đây là điểm khác biệt với truyền thông không đồng bộ mà chúng ta đã biết trong chuẩn UART. Sự tồn tại của chân SCK giúp quá trình tuyền ít bị lỗi và vì thế tốc độ truyền của SPI có thể đạt rất cao. Xung nhịp chỉ được tạo ra bởi chip Master.

MISO– Master Input / Slave Output: nếu là chip Master thì đây là đường Input còn nếu là chip Slave thì MISO lại là Output. MISO của Master và các Slaves được nối trực tiếp với nhau..

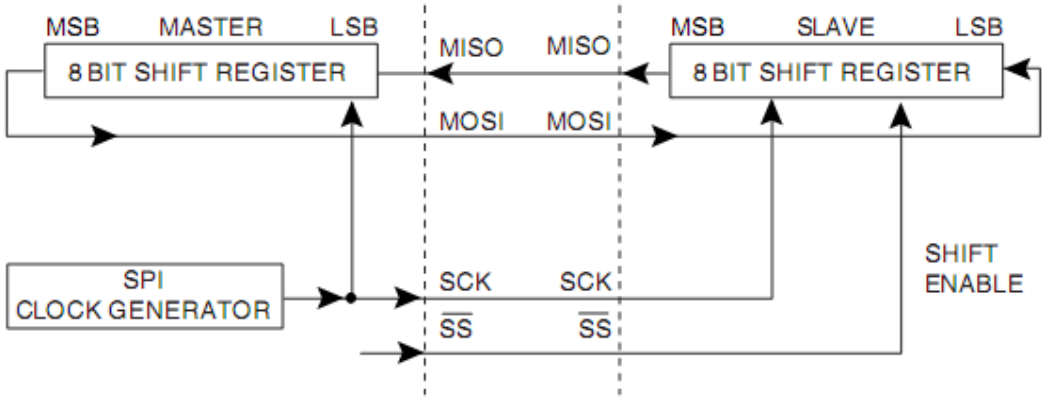
MOSI – Master Output / Slave Input: nếu là chip Master thì đây là đường Output còn nếu là chip Slave thì MOSI là Input. MOSI của Master và các Slaves được nối trực tiếp với nhau.

SS – Slave Select: SS là đường chọn Slave cần giap tiếp, trên các chip Slave đường SS sẽ ở mức cao khi không làm việc. Nếu chip Master kéo đường SS của một Slave nào đó xuống mức thấp thì việc giao tiếp sẽ xảy ra giữa Master và Slave đó. Chỉ có 1 đường SS trên mỗi Slave nhưng có thể có nhiều đường điều khiển SS trên Master, tùy thuộc vào thiết kế của người dùng.



Hình 2.5.1: Giao diện SPI

Hoạt động: mỗi chip Master hay Slave có một thanh ghi dữ liệu 8 bits. Cứ mỗi xung nhịp do Master tạo ra trên đường giữ nhịp SCK, một bit trong thanh ghi dữ liệu của Master được truyền qua Slave trên đường MOSI, đồng thời một bit trong thanh ghi dữ liệu của chip Slave cũng được truyền qua Master trên đường MISO. Do 2 gói dữ liệu trên 2 chip được gởi qua lại đồng thời nên quá trình truyền dữ liệu này được gọi là "song công". Hình 2 mô tả quá trình truyền 1 gói dữ liệu thực hiện bởi module SPI trong AVR, bên trái là chip Master và bên phải là Slave.



Hình 2.5.2: Thanh ghi dịch SPI

Cực của xung giữ nhịp, phase và các chế độ hoạt động: cực của xung giữ nhịp (Clock Polarity) được gọi tắt là CPOL là khái niệm dùng chỉ trạng thái của chân SCK ở trạng thái nghỉ. Ở trạng thái nghỉ (Idle), chân SCK có thể được giữ ở mức cao (CPOL=1) hoặc thấp (CPOL=0). Phase (CPHA) dùng để chỉ cách mà dữ liệu được lấy mẫu (sample) theo xung giữ nhịp. Dữ liệu có thể được lấy mẫu ở cạnh lên của SCK (CPHA=0) hoặc cạnh xuống (CPHA=1). Sự kết hợp của SPOL và CPHA làm nên 4 chế độ hoạt động của SPI. Nhìn chung việc chọn 1 trong 4 chế độ này không ảnh hưởng đến chất lượng truyền thông mà chỉ cốt sao cho có sự tương thích giữa Master và Slave.

Start Funcion	Data 0	Data 1	Data 2	Data 3	Function Code
Giá trị cho biết byte bắt đầu của khung truyền có giá trị bằng 250	Dữ liệu được truyền xuống, 1 biến float được tách thành 4 bytes				Byte cho biết chức năng mà slave sẽ thực hiện trong câu lệnh.

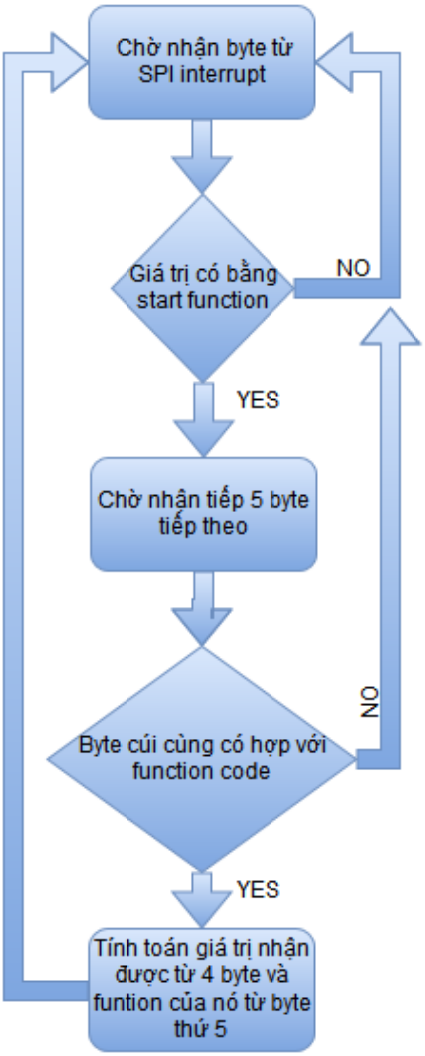
Bảng 2.5.1: Khung dữ liệu Master - Slave

Start Funcion	Data 0	Data 1	Data 2	Data 3	Function Code
Giá trị cho biết byte bắt đầu của khung truyền có giá trị bằng 255	Dữ liệu được truyền xuống, 1 biến float được tách thành 4 bytes				Byte cho biết đây là frame của slave nào.

Bảng 2.5.2: Khung dữ liệu Slave - Master

Việc thực hiện nhận dữ liệu qua SPI đều thông qua interupt, trước đó có thực nghiệm bằng DMA nhưng vì 1 MASTER phải giao tiếp với 3 SLAVE nên việc nhận dữ liệu chính xác là khá khó.

Sơ đồ chung cho việc xử lý nhận dữ liệu ở SPI



Hình 2.5.3: Flow chart nhận SPI

Việc gửi dữ liệu đi hoàn toàn do MASTER làm chủ, khi cần đọc giá trị từ SLAVE thì MASTER sẽ gửi xuống 1 frame với function code bằng 0 và khi nhận được thì SLAVE sẽ gửi lại dữ liệu cần thiết.Tương tự khi mún đặt giá trị cho motor chạy thì MASTER gửi frame với data là giá trị của góc muốn quay và function code là 2 hoặc là thời gian đặt cho hàm dốc với function code là 1.

Việc gửi dữ liệu ở MASTER thực hiện bằng thủ công, gửi từng byte sau đó chờ cờ hoàn tất gửi dữ liệu để tiếp tục gửi byte tiếp theo. Trong khi đó thì việc gửi dữ

liệu ở SLAVE sử dụng DMA, bình thường DMA sẽ bị ngưng kích hoạt, khi nhận được 1 frame truyền yêu cầu gửi dữ liệu lên từ MASTER thì SLAVE sẽ kích hoạt DMA lại, sau khi có interrupt đã truyền tải xong của DMA thì nó sẽ được ngưng kích hoạt lại.

2.5.2. Giao thức UART

Giao thức UART đã quá quen thuộc trong quá trình học tập tại trường nên trong đề tài luận văn này sẽ không nhắc lại lý thuyết về giao thức này nữa.

Sự dụng Mạch chuyển USB UART PL2303

Sử dụng chip PL2303HX chuyển đổi USB - UART dễ dàng kết nối với máy tính.

Module dễ dễ dàng cho việc nghiên cứu các module khác bằng cách gửi lệnh trực tiếp từ máy tính và phân tích dữ liệu nhận được lên màn hình máy tính mà không cần thông qua chương trình của vi điều khiển.

Thông số kỹ thuật:

Điện áp 5V cấp trực tiếp từ cổng USB.

Ngõ ra dạng UART gồm 2 chân TX, RX.

Với 3 led trên board: led báo nguồn, led RX, led TX.

Kích thước: 15 x 31 mm.

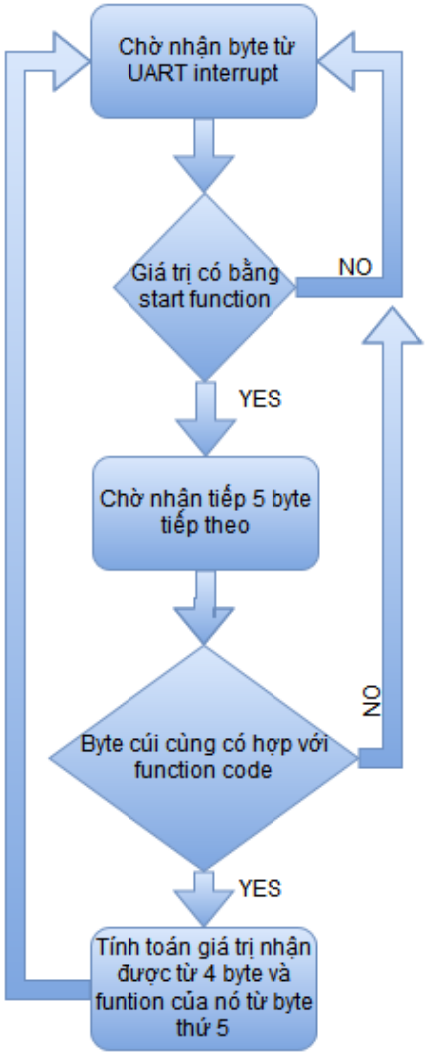
Start Funcion	Data 0	Data 1	Data 2	Data 3	Function Code
Giá trị cho biết byte bắt đầu của khung truyền có giá trị bằng 200	Dữ liệu được truyền xuống, 1 biến float được tách thành 4 bytes				Byte cho biết chức năng mà slave sẽ thực hiện trong câu lệnh.

Bảng 2.5.3: Khung dữ liệu máy tính - Master

Start Funcion	Data 0	Data 1	Data 2	Data 3	Function Code
Giá trị cho biết byte bắt đầu của khung truyền có giá trị bằng 100	Dữ liệu được truyền xuống, 1 biến float được tách thành 4 bytes				Byte cho biết đây là dữ liệu của slave nào.

Bảng 2.5.4: Khung dữ liệu Master – Máy tính

Cũng tương tự như SPI, việc trao đổi giữa máy tính và MASTER theo phương thức UART như sau:

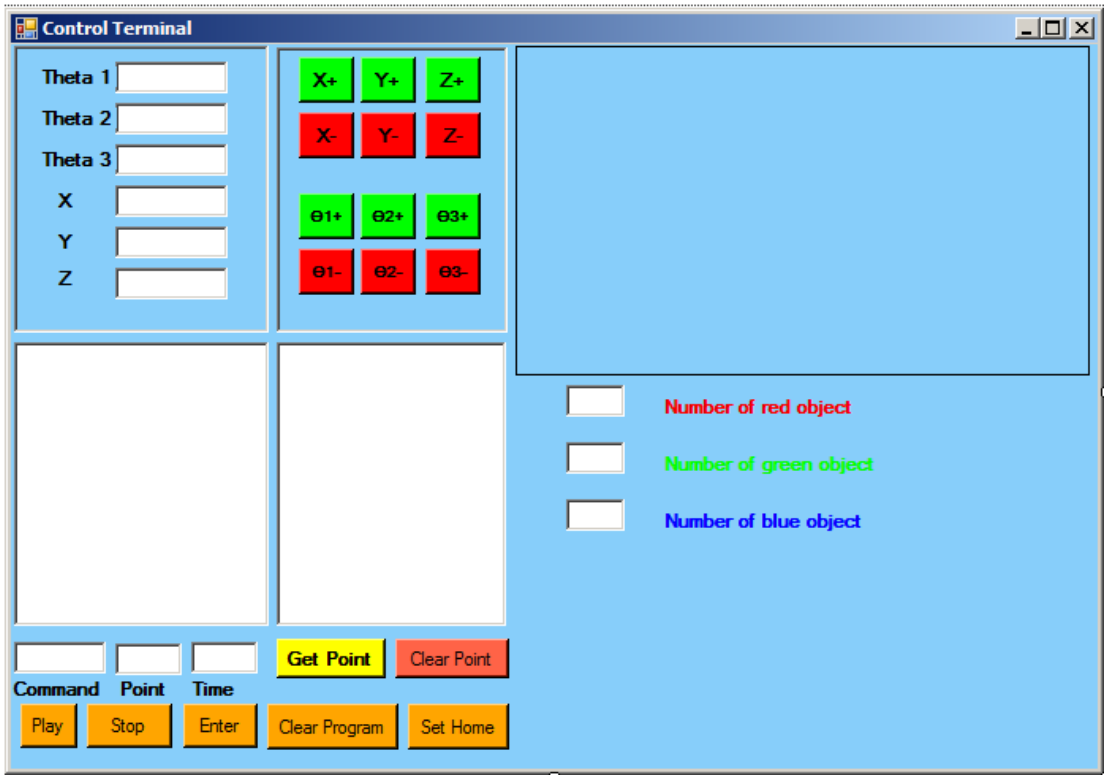


Hình 2.5.4: flow chart nhận UART

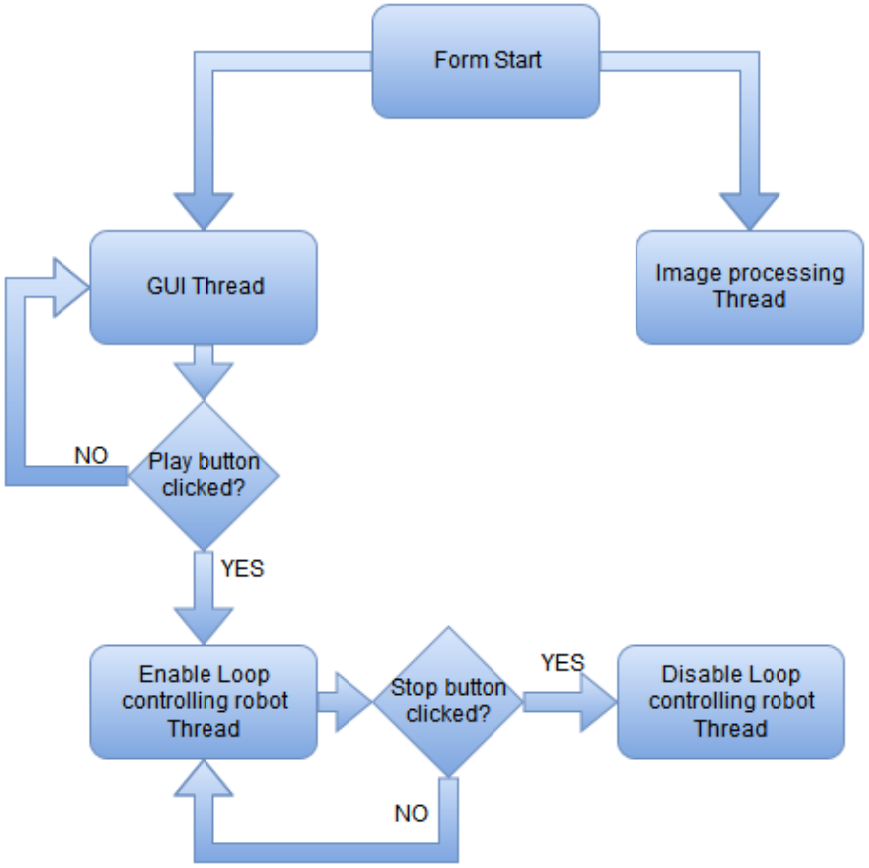
Việc gửi dữ liệu ở cả máy tính và MASTER đều là thủ công và độc lập. Từng byte sẽ được đưa vào buffer và gửi đi khi có lệnh. Nhưng việc nhận dữ liệu ở máy tính là theo sơ đồ trên trong khi nhận dự liệu ở MASTER lại sử dụng DMA, vì ở đây giao tiếp đơn giản, chỉ có 2 thiết bị đầu cuối liên lạc với nhau nên việc đồng bộ với nhau thông qua DMA là khả thi.

2.6. Chương trình điều khiển trên máy tính

Giao diện lập trình trên Windows Form Aplication bằng ngôn ngữ C#



Hình 2.6.1: giao diện GUI máy tính



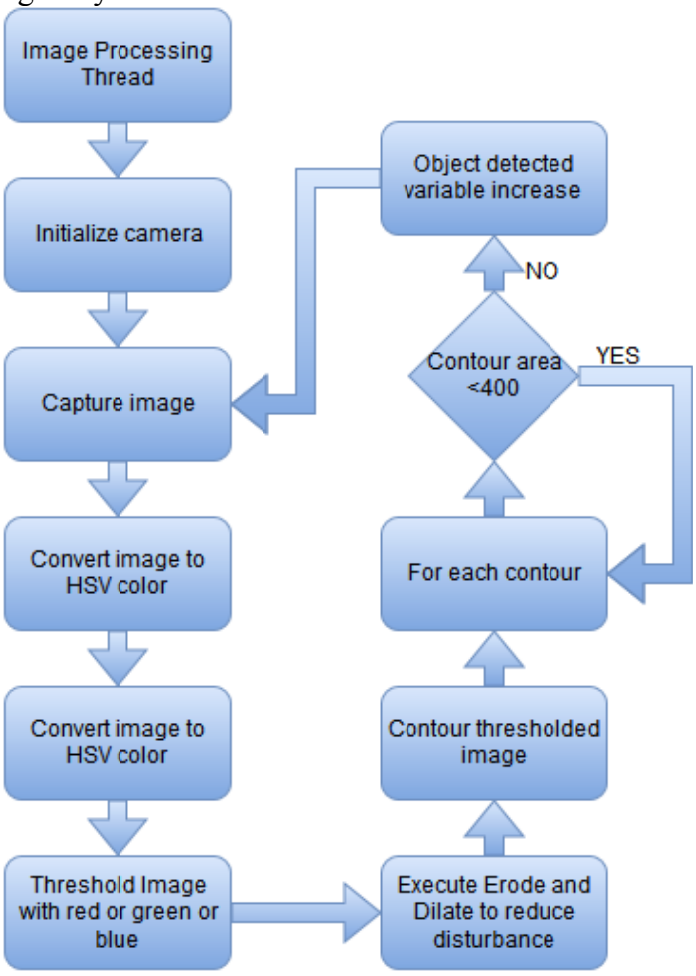
Hình 2.6.2: Flow chart chương trình máy tính

- Cấu trúc luồng giao diện GUI:
  - GUI thread là luồng xử lý chính của winfow form. Nó đảm nhiệm toàn bộ xử lý tác vụ cho window form timer cũng như tài nguyên trên giao diện màn hình. Các tài nguyên được GUI thread đàm nhiệm như sau
  - Timer 1: Giữ nhiệm vụ cập nhật các giá trị nhận được từ các interrupt khác như là góc quay của robot, tọa độ xyz của nó, và dữ liệu đã xử lý được của luồng xử lý ảnh.
  - Timer 2: Là thời gian lấy mẫu để tính toán ra các giá trị góc quay cho robot khi nhấn các nút tăng giảm vị trí của robot, đồng thời ngay sau đó truyền toàn bộ dữ liệu hoặc lệnh này xuống cho robot thực hiện thông qua UART.
  - Khối nút nhấn XYZ và theta: dùng để ngắt xem người dùng đang muốn thay đổi vị trí của robot như thế nào và bật cờ tương ứng lên cho bộ phận timer2 xử lý.
  - Khối điều khiển lập trình cho robot gồm có:
  - Bộ phận trích lấy điểm làm việc của robot gồm có 2 nút nhấn get point và clear point. Nút nhấn get point sẽ đưa tọa độ hiện tại của robot vào một bộ nhớ riêng của nó và hiển thị ra khung text point bên trên cho người dùng

kiểm tra lại. Tương tự nút clear point dùng để xóa toàn bộ vùng bộ nhớ mà get point đã tạo ra.

- Bộ phận kiểm soát hoạt động của robot và lập trình robot gồm có :
- Nút nhấn play và stop: đây là cặp nút nhấn dùng để khởi động chu trình làm việc của robot theo chương trình đã định sẵn trong bộ nhớ bằng cách tạo ra một phân luồng mới và cho nó chạy độc lập trên đó.
- Nút nhấn enter: dùng để ghi nhớ lại toàn bộ câu lệnh đã được nhập vào trong các ô text lệnh bên trên vào trong một bộ nhớ lưu trữ để sử dụng cho chu trình chạy của robot sau này đồng thời hiện thị các lệnh đã được nhập lên text box bên trên nó.
- Nút nhấn clear program : dùng để xóa toàn bộ câu lệnh trong bộ nhớ đã đc lưu cho robot chạy.
- Nút nhấn set home: dùng để đưa robot về vị trí home tức là góc quay đặt của cả 3 động cơ là bằng 0.

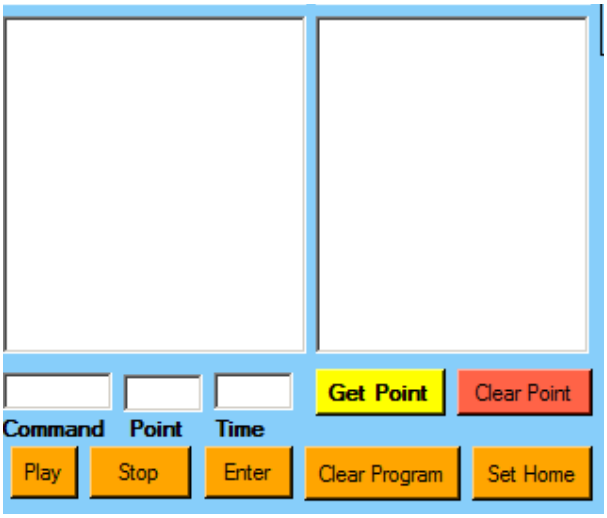
- Cấu trúc luồng xử lý ảnh



Hình 2.6.3: Flow chart xử lý ảnh

- Chương trình xử lý ảnh còn đơn giản, do đây không phải là trọng tâm của luận văn. Chương trình chỉ kết hợp với robot để cho thấy được hoạt động thực tiễn. Tuy đơn giản nhưng cũng cho kết quả xử lý khá chính xác và đáng tin cậy, đặc biệt khi hoạt động trong một môi trường cho trước.
- Hoạt động xử lý ảnh chủ yếu của chương trình là lấy threshold và contour hình ảnh, điểm lợi thế của phương pháp này là cho kết quả nhanh chóng. Theo chương trình đang chạy thì nhận dạng cả ba màu đỏ lục lam trong cũng một hình cho kết quả tầm 24-29 frame per second. Tuy nhiên khi môi trường thay đổi như độ sáng tối xung quanh thì độ nhạy của chương trình giảm xuống đáng kể. Để khắc phục vấn đề này mà vẫn sử dụng phương pháp này thì phải áp dụng thêm một số giải thuật để cân bằng sáng và loại nhiễu trước khi đưa vào xử lý như trên.

Tập lệnh lập trình cho robot và cách sử dụng:

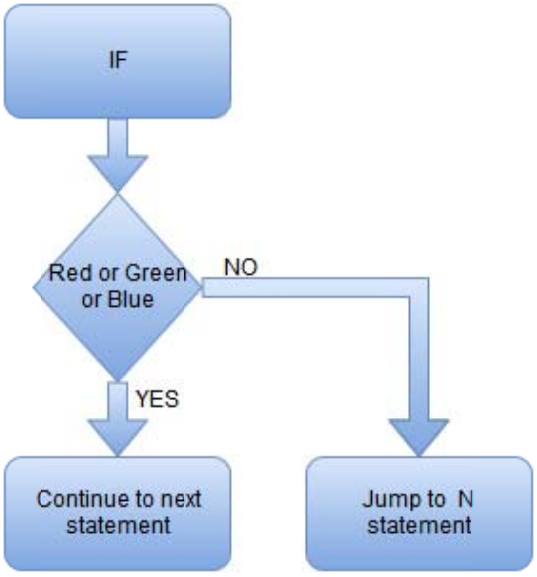


Hình 2.6.4: giao diện nhập lệnh

- Chương trình tập lệnh điều khiển robot khá đơn giản chỉ với 4 lệnh cơ bản:
- Lệnh goto: Thực thi việc cho robot di chuyển tới 1 điểm đã được lưu trước bởi lệnh getpoint. Vị trí cần đến sẽ ghi vào ô text box Point và thời gian di chuyển cho động tác này sẽ được ghi vào ô text box Time. Ví dụ : [goto 1 1000] là cho robot di chuyển đến vị trí P1 với thời gian 1s.
- Lệnh wait: cho chương trình vào trạng thái chờ trong một khoảng thời gian đặt trước trong ô Time, khi này thì ô text box point sẽ không có nghĩa. Ví dụ [wait 1 2000] thì chương trình sẽ dừng trong 2s.
- Lệnh gripper: khi thực thi lệnh chương trình sẽ gửi yêu cầu xuống Master để đóng ngắt MOSFET nhằm cho dòng chạy qua solenoid hút vật lên và thả vật xuống. Lệnh thực hiện hút vật khi giá trị tại ô point text box bằng 1 và thả vật khi giá trị tại đó bằng 0, ngoài ra thời gian đặt tại ô time cũng là thời gian

delay sau khi hút thả vật. Ví dụ [gripper 1 1000] sẽ cho lệnh hút vật lên và chờ 1s sau đó và lệnh [gripper 0 2000] sẽ thả vật xuống và chờ 2s sau đó.

- Lệnh if: Đây là lệnh điều kiện đặt biệt chỉ sử dụng cho chương trình có xử lý ảnh. Cấu trúc lệnh if như sau:



Hình 2.6.5: Flow chart xử lý điều khiển robot

- Lệnh if sẽ xem xét người dùng muốn chọn vật thể có màu gì, và màu vật thể sẽ được ghi nhận tại ô point text box. Nếu điều kiện đúng, tức là vật thể xác nhận là có thì lệnh tiếp theo sau if sẽ được thực thi. Nếu điều kiện sai thì lệnh tiếp theo thứ N sẽ được thực thi, và N là con số được lấy từ trong ô time text box.

Chương 3. KẾT QUẢ THỬ NGHIỆM VÀ ĐÁNH GIÁ

3.1. Kết quả thử nghiệm

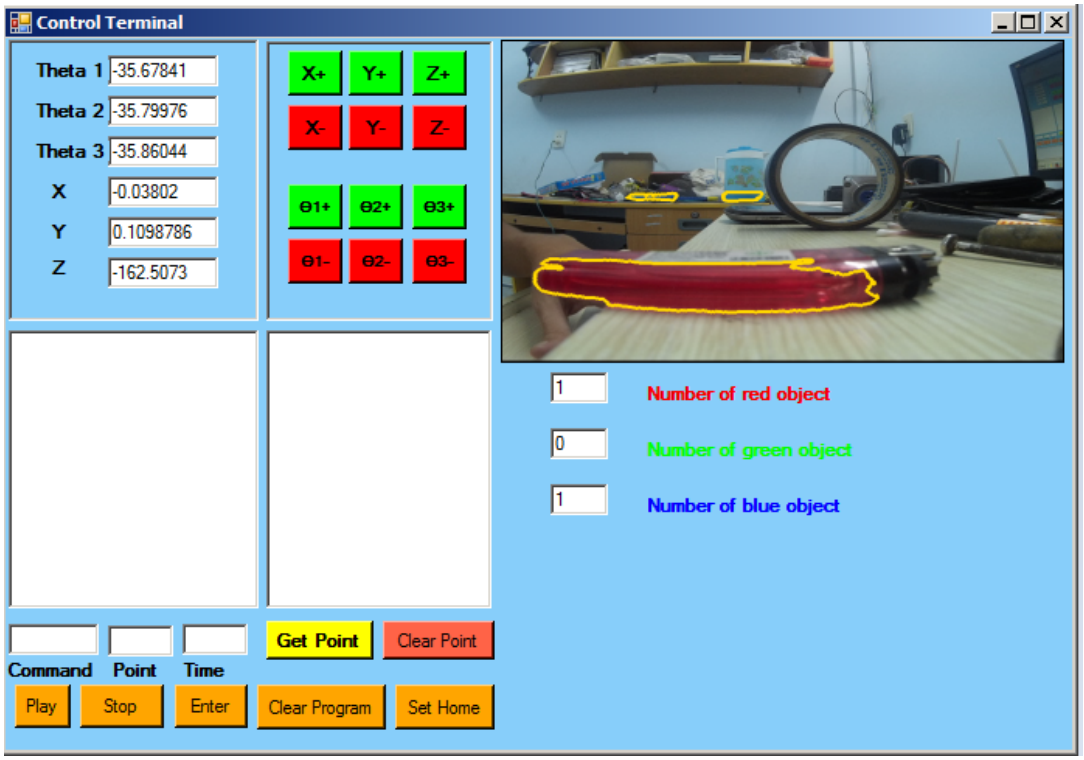
Sau một thời gian nghiên cứu và tìm hiểu để ra được mô hình robot như bây giờ thì phần lớn thời gian của sinh viên là dành cho việc thiết kế mô hình và phần cơ khí. Tuy cấu trúc có vẻ đơn giản nhưng lại vô cùng khó khăn vì việc tìm mua phụ kiện cho robot, ví dụ như khớp cầu và các thanh ren nói với nó là khá khó.

Thời gian mất nhiều nhất là ở phần cơ và chính phần cơ đó cũng là điểm yếu nhất của mô hình robot này. Vì khớp cầu có góc xoay chỉ 30 độ nên vùng làm việc của robot cũng bị giảm xuống đáng kể,vấn đề nữa là các thông số tay đòn trên dưới của robot chỉ là thông số được tính tỉ lệ lại so với con delta robot của ABB nên phải nói mô hình phần cơ khí ở đây chỉ gọi là bắt chước chứ chưa thực sự là tính toán vùng làm việc hoặc mô hình động lực học để ra được thông số.

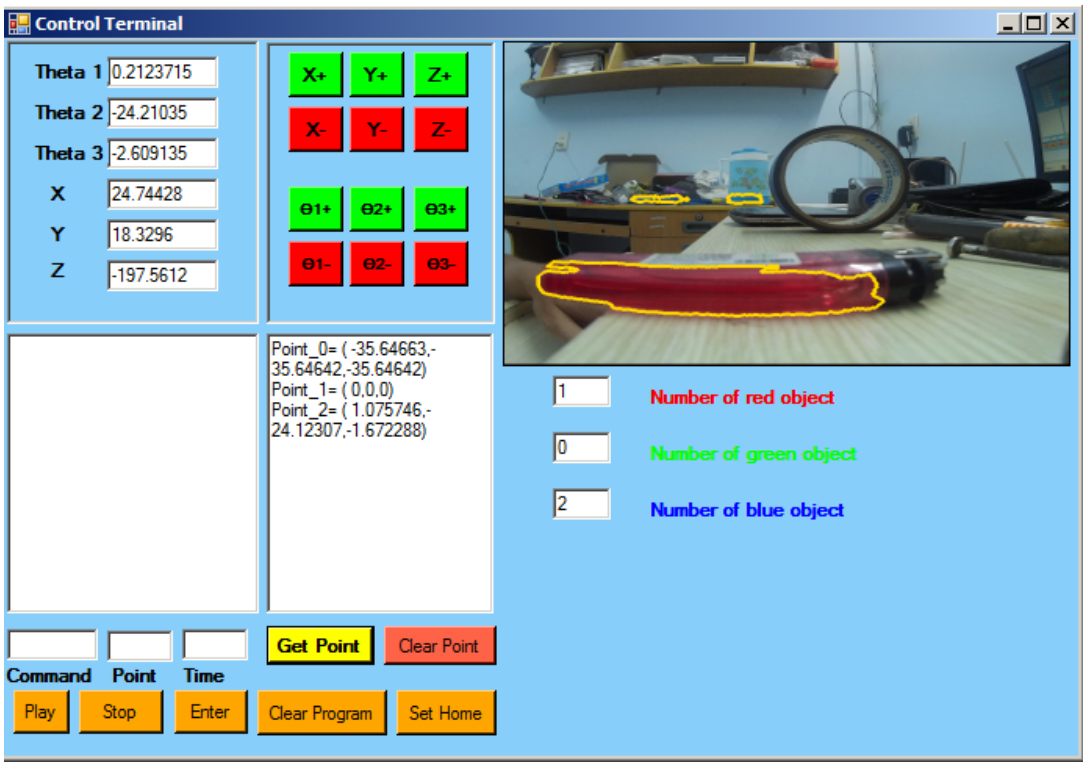
Về vấn đề điều khiển thì cũng có một số hạn chế, ban đầu motor được điều chỉnh và sử dụng bộ điều khiển PID fuzzy seft tunning cho giá trị hoạt động ổn định và đáp ứng tốt nhưng sau khi đã ráp motor vào khung robot thì bộ điều khiển lại tỏ vẻ mất ổn định và gây ra vài sự cố, vì vậy để giải quyết vấn đề này sinh viên đã quay trở lại sử dụng bổ điều khiển PID thông thường với thông số được chỉnh dưa theo kinh nghiệm.

Sau một thời gian ráp lên khung robot tuy có vài sự cố nhưng robot đã vận hành được nhưng còn khá nhìu lỗi chưa giải quyết được, chủ yếu là không còn thời gian để sửa lỗi nữa.

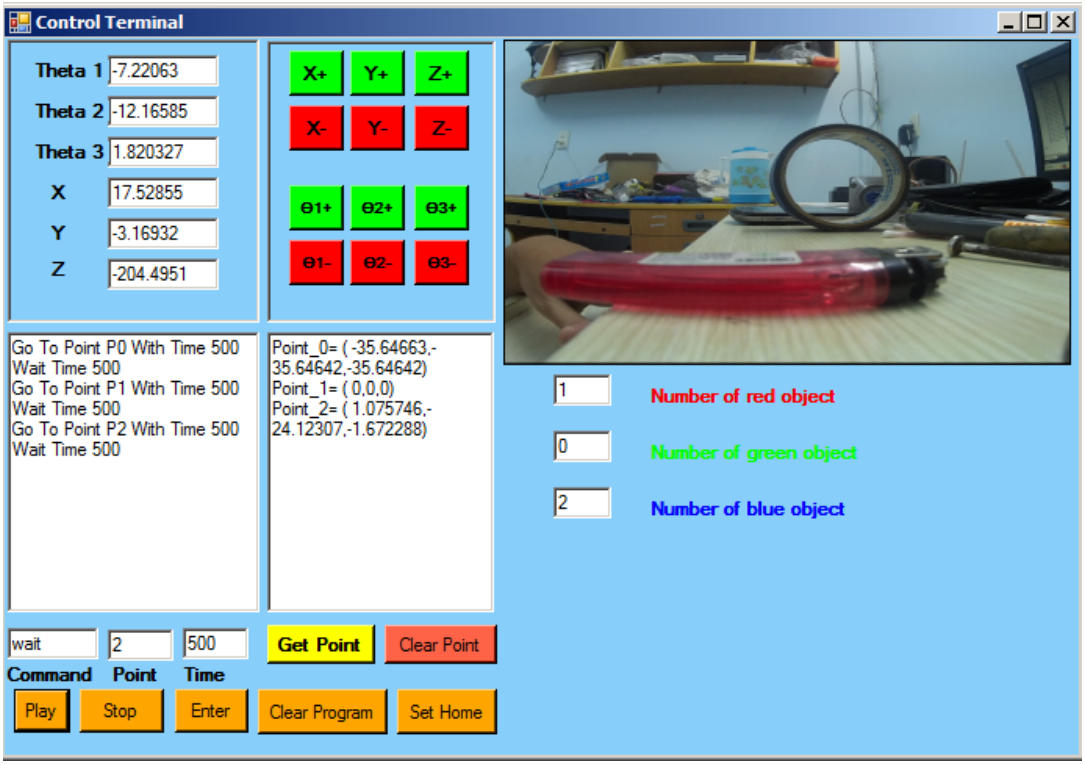
Vì robot được sử dụng tín hiệu điều khiển là hàm dốc nên đáp ứng của nó rất tốt, đầu cuối robot chạy khá êm mặc dù khung robot khá yếu ớt.



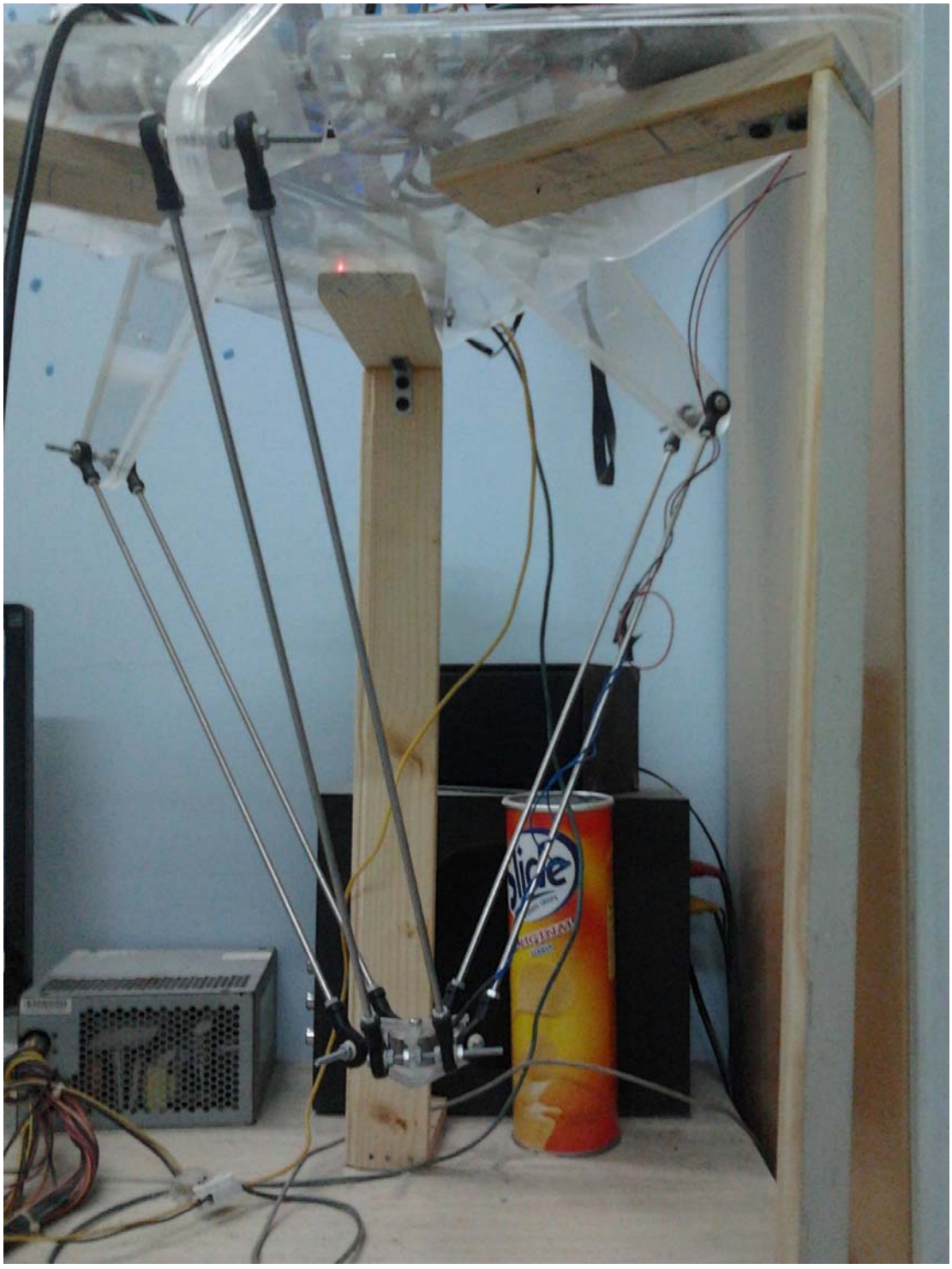
Hình 3.1.1: giao diện hoạt động



Hình 3.1.2: Giao diện hoạt động



Hình 3.1.3: giao diện hoạt động



Hình 3.1.4: Mô hình robot thực tế

Hình 3.1.5: Mô hình robot thực tế

Hình 3.1.6: Mô hình robot thực tế

3.2. Hướng phát triển tương lai

Đối với đề tài này thì đây chỉ là mô hình cho thấy tính khả dụng của delta robot và hoàn toàn chưa thể đảm nhận một công việc cụ thể trong thực thế.

Trong thực tế delta robot được sử dụng motor ac servo cho độ chính xác và tốc độ cao khi điều khiển đồng thời hệ cơ khí của robot cũng được thiết kế chính xác để robot có thể thực hiện được ngay cả những tác vụ tinh vi nhất như là gắp những linh kiện điện tử để hàn lên mạch.

Delta robot hiện nay ở nước ta có thể ít thấy vì bản thân robot này là để thay thế nguồn nhân lực lao động của con người, nhưng vì nước ta nguồn nhân lực còn nhìu nên phát triển delta robot trong thời gian hiện tại là chưa mấy lạc quan

Trong tương lai chắc chắn việc phát triển loại robot này ở nước ta sẽ rất phát triển vì những yêu cầu nghiêm ngặt về chất lượng của sản phẩm trên thị trường quốc tế khi mà Việt Nam ngày càng đi sâu vào thị trường này. Đặc biết là những ngành công nghiệp thực phẩm, sản xuất mạch điện tử hay phổ biến nhất là đóng gói thành phẩm,… tất cả đều là những ngành mà delta robot đã và đang hướng tới.

TÀI LIỆU THAM KHẢO

[1] Huỳnh Thái Hoàng, *Slide bài giảng lý thuyết điều khiển nâng cao*., 2012.

[2] Huỳnh Thái Hoàng, *Slide bài giảng nhập môn điều khiển thông minh*., 2014.

[3] A. A. Lasheen, A. M. El-Garhy, and E. M. Saad and S.M.Eid, *TUNING PID CONTROLLERS USING HYBRID GENETIC AND NELDER-MEAD ALGORITHM*.: Journal of Engineering Sciences, Assiut University, 2010.

[4] Robert L. Williams II, *The Delta Parallel Robot: Kinematics Solutions*., 2015.

[5] Huỳnh Thái Hoàng, *slide bài giảng lý thuyết điều khiển tự động*., 2006.

[6] STmicroelectronics, *STM32F103xx reference manual*.

[7] STmicroelectronics, *STM32f103xx hardware manual*.

[8] International Rectifier, *Ir2184 datasheet*.

[9] International Rectifier, *HV Floating MOS Gate Drivers*.

[10] Wikipedia, *PID controler*.

[11] OpenCV, *OpenCV user manual*.

[12] hocavr, *giao tiếp SPI*.