

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH
KHOA ĐIỆN - ĐIỆN TỬ
BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG

VŨ QUỐC ANH – 1410149

LÊ ANH ĐỨC – 1410922

LUẬN VĂN TỐT NGHIỆP
CÁNH TAY ROBOT 5 BẬC TỰ DO
TÍCH HỢP THỊ GIÁC MÁY

KỸ SƯ NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

TP. HỒ CHÍ MINH, 2018

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG

VŨ QUỐC ANH – 1410149

LÊ ANH ĐỨC – 1410922

LUẬN VĂN TỐT NGHIỆP
CÁNH TAY ROBOT 5 BẬC TỰ DO
TÍCH HỢP THỊ GIÁC MÁY

5 D.O.F ROBOT MANIPULATOR WITH COMPUTER VISION

KỸ SƯ NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

GIẢNG VIÊN HƯỚNG DẪN
PHẠM VIỆT CƯỜNG

TP. HỒ CHÍ MINH, 2018

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:

**CÁNH TAY ROBOT 5 BẬC TỰ DO TÍCH HỢP THỊ GIÁC MÁY
5 D.O.F ROBOT MANIPULATOR WITH COMPUTER VISION**

Nhóm Sinh viên thực hiện:

Vũ Quốc Anh

1410149

Phạm Việt Cường

Lê Anh Đức

1410922

Phạm Việt Cường

Cán bộ hướng dẫn:

Đánh giá Luận văn

1. 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:

.....
.....
.....
.....
.....
.....
.....
.....

2. Về nội dung luận văn:

3. Về tính ứng dụng:

4. Về thái độ làm việc của sinh viên:

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

Điểm từng sinh viên:

Vũ Quốc Anh :/10

Lê Anh Đức :/10

Cán bộ hướng dẫn

(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:

**CÁNH TAY ROBOT 5 BẬC TỰ DO TÍCH HỢP THỊ GIÁC MÁY
5 D.O.F ROBOT MANIPULATOR WITH COMPUTER VISION**

Nhóm Sinh viên thực hiện:

Vũ Quốc Anh

1410149

Phạm Việt Cường

Lê Anh Đức

1410922

Phạm Việt Cường

Cán bộ hướng dẫn:

Vũ Quốc Anh

Phạm Việt Cường

Đánh giá Luận văn

5. 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:

.....
.....
.....
.....
.....
.....
.....
.....
.....

6. Về nội dung luận văn:

.....

.....
.....
.....
.....
.....
.....

7. Về tính ứng dụng:

.....
.....
.....
.....
.....
.....

8. Về thái độ làm việc của sinh viên:

.....
.....
.....
.....
.....
.....

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

Điểm từng sinh viên:

Vũ Quốc Anh :/10

Lê Anh Đức :/10

Người nhận xét

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn tới **tập thể giảng viên, cán bộ tại Trường Đại Học Bách Khoa Thành Phố Hồ Chí Minh** vì những kiến thức cần thiết, hữu ích cùng sự giúp đỡ, chỉ bảo trong quá trình chúng em học tập tại trường.

Chúng em xin được cảm ơn các Thầy, Cô trong bộ môn Điều khiển – Tự động, những người có ảnh hưởng trực tiếp nhất đến luận văn này: Thầy **Huỳnh Thái Hoàng** và Cô **Bùi Thanh Huyền**, những người giảng dạy chúng em những kiến thức nền tảng và nâng cao của lý thuyết điều khiển tự động; Thầy **Nguyễn Trọng Tài**, Thầy trang bị cho chúng em kiến thức nền tảng về Thị giác máy; Thầy **Trần Ngọc Huy** với kiến thức về điều khiển robot. Đây là những Thầy cô có ảnh hưởng lớn nhất đến luận văn của chúng em, và còn rất nhiều những Thầy cô khác trong tập thể cán bộ trường mà chúng em không thể kể hết. Cuối cùng, chúng em xin dành lời cảm ơn tới Thầy **Phạm Việt Cường**. Thầy vừa là người giảng dạy chúng em trong môn Trí tuệ nhân tạo cùng môn Thị giác máy, vừa là người trực tiếp hướng dẫn chúng em làm luận văn này. Một số kiến thức nâng cao mà Thầy đã dạy được chúng em áp dụng trong luận văn này. Thầy cũng là người hỗ trợ tài chính cho chúng em để biến lý thuyết thành hiện thực. Chúng em xin thể hiện sự biết ơn sâu sắc đến Thầy, xin cảm ơn Thầy !

Xin cảm ơn **các bạn lớp DD14TD** cùng **nhiều anh chị, bạn bè khác** đã đồng hành với chúng mình trong suốt 4 năm qua. Chính sự giúp đỡ cũng như gianh đua từ các bạn, là một phần không thể thiếu, tạo nên thời sinh viên năng động cùng những kỷ niệm khó quên, đẹp đẽ.

Xin cảm ơn anh **Nguyễn Tân Khoa K08**, anh **Trịnh Hoài Nam** là những người đi trước và chỉ dạy chúng em những vấn đề chuyên môn khi thực hiện luận văn. Các anh đều là cựu sinh viên của trường.

Chúng em xin dành lời cảm ơn cuối cùng tới **Gia đình, những người luôn thấu hiểu và hỗ trợ chúng em quá trình**.

BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG

TP. HCM, ngày tháng năm

ĐỀ CƯƠNG CHI TIẾT

**TÊN LUẬN VĂN: CÁNH TAY ROBOT 5 BẬC TỰ DO TÍCH HỢP THỊ GIÁC MÁY
- 5 D.O.F ROBOT MANIPULATOR WITH COMPUTER VISION**

Cán bộ hướng dẫn: Phạm Việt Cường

Thời gian thực hiện: Từ ngày 01/01/2018 đến ngày 10/06/2018

Sinh viên thực hiện:

VŨ QUỐC ANH - 1410149

LÊ ANH ĐỨC - 1410922

Nội dung đề tài:

Xây dựng cơ cấu mô hình cánh tay robot 5 bậc tự do có kết cấu cơ khí vững chắc và phù hợp với ứng dụng gấp, thả vật có khối lượng vừa phải với độ chính xác cao làm đối tượng công tác.

Hệ thống tích hợp camera Kinect, sử dụng các thuật toán xử lí và bám theo hình ảnh vật thể trong không gian và trả về tọa độ 3 chiều để mô hình cánh tay robot di chuyển, tạo thành vòng điều khiển kín để tương tác với vật.

Quá trình xây dựng bộ điều khiển vị trí của cánh tay robot bao gồm các bước:

- Xây dựng thông số, thiết kế mô hình.
- Xây dựng thông số ma trận D-H.
- Giải bài toán động học thuận và bài toán động học ngược.
- Giải thuật điều khiển vị trí của các khớp.

Đồng thời để tiến tới xây dựng và phát triển một mô hình robot 5 bậc tự do thực tế hoàn chỉnh trong thời gian thực hiện luận văn, nhóm thực hiện xây dựng bộ điều khiển điều khiển vị trí của động cơ DC bằng bộ điều khiển có khả năng thích nghi với sự thay đổi momen tải.

Nhóm sử dụng vi điều khiển STM32F407 làm bộ điều khiển trung tâm của cánh tay nhận vị trí của từng động cơ. Từng giải thuật cụ thể trên từng động cơ như sau:

- 5 động cơ đều sử dụng giải thuật STR MRAS.
- Động cơ điều khiển đầu gắp Gripper điều khiển vòng hở.

Ngoài ra nhóm cũng dự định thực hiện ứng dụng thị giác máy thuật toán thị giác máy xử lý ảnh khai thác phần cứng hệ thống camera Kinect để nhận dạng và bám theo vật thể, đồng thời xác định tọa độ 3 chiều của vật thể theo gian thực với độ chính xác cao nhất.

- Xây dựng giao diện người - máy bằng thư viện Qt để giám sát và ra chỉ thị điều khiển.

KÌ VỌNG KẾT QUẢ ĐẠT ĐƯỢC CỦA ĐỀ TÀI:

Xây dựng được mô hình thực tế cánh tay robot 5 bậc tự do hoàn chỉnh đạt được các tiêu chí sau:

- Kết cấu vững chắc phù hợp với ứng dụng.
- Bộ điều khiển cánh tay có chất lượng điều khiển tốt, sai số thấp theo điểm đặt.
- Hạn chế các tình trạng rung lắc của cánh tay trong quá trình vận hành.
- Hoạch định được quỹ đạo từng khớp.

Hệ thống thị giác máy có khả năng nhận diện vật chính xác, thuật toán xử lý với tốc độ cao cho phép bám theo vật và xác định tọa độ 3 chiều của vật.

Tích hợp hệ thống cánh tay robot và xử lý hình ảnh qua Kinect, xây dựng giao diện điều khiển trên máy tính và giao thức Protocol giao tiếp giữa 2 hệ thống hoàn chỉnh, được tiêu chí:

- Thuật toán xử lý ảnh xác định vật có độ chính xác cao.
- Phối hợp điều khiển cánh tay robot bám theo vật thể chính xác như tọa độ đã được gửi xuống từ máy tính.

Kế hoạch thực hiện:

Các công việc chính và kế hoạch làm việc:

Số thứ tự	Công việc	Hoàn thành trước	Thời gian thực hiện
1	Nghiên cứu đề tài luận văn, cơ sở lý thuyết và hướng tiếp cận	15/03/2018	3 tháng
2	Thiết kế mô hình cơ khí và lựa chọn động cơ	15/03/2018	1 tháng
3	Xây dựng bộ điều khiển vị trí cho động cơ DC	28/02/2018	1 tháng
4	Xây dựng bộ điều khiển tính toán động học thuận / ngược	28/02/2018	2 tháng
5	Thi công, lắp ráp cơ khí và hoàn thiện bộ điều khiển cánh tay robot	15/05/2018	2 tháng
6	Xác định tọa độ vật trong không gian bằng Kinect	20/04/2018	1 tuần
7	Xây dựng thuật toán nhận dạng và theo dõi vật thể	10/05/2018	2 tuần
8	Xây dựng giao diện người – máy bằng Qt	10/06/2018	2 tháng
9	Tích hợp hệ thống và kiểm tra, cải thiện hệ thống	10/06/2018	1 tháng
10	Viết báo cáo	10/06/2018	5 tháng

Phân công công việc:

Vũ Quốc Anh: Thực hiện các việc từ 6 đến 8.

Lê Anh Đức: Thực hiện các việc từ 2 đến 4.

Cả hai: Thực hiện các việc còn lại

Xác nhận của Cán bộ hướng dẫn (Ký tên và ghi rõ họ tên)	TP. HCM, ngày 10 tháng 6 năm 2018	
Phạm Việt Cường	Sinh viên Vũ Quốc Anh	Lê Anh Đức

DANH SÁCH HỘI ĐỒNG BẢO VỆ LUẬN

VĂN

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

1. - Chủ tịch.
2. - Thư ký.
3. - Ủy viên.
4. - Ủy viên.
5. - Ủy viên.

MỤC LỤC

NHẬN XÉT LUẬN VĂN TỐT NGHIỆP CỦA CÁN BỘ HƯỚNG DẪN.....	i
NHẬN XÉT LUẬN VĂN TỐT NGHIỆP CỦA CÁN BỘ PHẢN BIỆN	ii
LỜI CẢM ƠN	iii
ĐỀ CƯƠNG CHI TIẾT	1
KÌ VỌNG KẾT QUẢ ĐẠT ĐƯỢC CỦA ĐỀ TÀI:	2
MỤC LỤC	5
DANH MỤC BẢNG	8
DANH MỤC HÌNH ẢNH	9
CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN ĐỀ TÀI.....	3
1.1. GIỚI THIỆU TỔNG QUAN VỀ ROBOT CÔNG NGHIỆP:.....	3
1.1.1. Lịch sử phát triển robot công nghiệp:	3
1.1.2. Tình hình nghiên cứu và ứng dụng của robot công nghiệp:.....	5
1.2. MỤC ĐÍCH NGHIÊN CỨU:.....	7
1.3. MỤC TIÊU LUẬN VĂN:	8
1.4. PHƯƠNG PHÁP THỰC HIỆN:.....	9
CHƯƠNG 2: THIẾT KẾ CƠ KHÍ VÀ BỘ ĐIỀU KHIỂN CÁNH TAY ROBOT	10
2.1. ĐỊNH NGHĨA - KHÁI NIỆM CƠ BẢN TRONG ROBOT:	10
2.1.1. Bậc tự do – Degree of Freedom (DOF):.....	10
2.1.2. Hệ tọa độ - Hệ trục tọa độ:.....	11
2.1.3. Trường công tác - Workspace or Range of motion:	11
2.1.4. Cấu trúc chung của một robot công nghiệp:.....	13
2.1.5. Một số kết cấu robot phổ biến:	14
2.1.6. Một số vấn đề cần giải quyết:	16
2.2. MÔ HÌNH THIẾT KẾ CƠ KHÍ CÁNH TAY ROBOT:	17
2.2.1. Mô hình thiết cơ khí và thông số mô hình:	17
2.2.2. Các động cơ truyền động trên từng khớp và tỉ số truyền động cơ khí:	21
2.2.3. Cấu hình tay gấp:.....	22
2.2.4. Phân tích bài toán động học thuận – nghịch:.....	23

2.3.	THIẾT KẾ BỘ ĐIỀU KHIỂN ROBOT:	26
2.3.1.	Giới thiệu sơ lược về kit STM32F407VGT Discovery:	26
2.3.2.	Cấu hình chức năng cho kit điều khiển:	29
2.3.3.	Các khói chức năng công suất - cảm biến – nguồn – giao tiếp: ...	30
2.4.	GIẢI THUẬT ĐIỀU KHIỂN VỊ TRÍ ĐỘNG CƠ:.....	38
2.4.1.	Giải thuật STR MRAS cải tiến:	38
2.4.2.	Giải thuật qui hoạch quỹ đạo vị trí:	43
CHƯƠNG 3:	GIẢI THUẬT XỬ LÍ ẢNH XÁC ĐỊNH VỊ TRÍ VÀ NHẬN DẠNG BÁM THEO VẬT THỂ:.....	45
3.1.	TỔNG QUAN KHÓI XỬ LÍ ẢNH.....	45
3.1.1.	Mục đích khói xử lí ảnh:.....	45
3.1.2.	Phương pháp tiếp cận:	46
3.1.3.	Giới thiệu sơ lược về Kinect và thư viện Point Cloud Library – PCL:	46
3.1.4.	Tổng quan thuật toán xử lí hình ảnh:	53
3.2.	GIẢI THUẬT NHẬN DẠNG VÀ XÁC ĐỊNH VỊ TRÍ:.....	54
3.2.1.	Chọn keypoints:.....	54
3.2.2.	Đặc trưng SHOTs (Signature of Histogram of Orientation features):	56
3.2.3.	Tìm các keypoints tương đồng (Matching):	58
3.2.4.	Gom keypoints thành vật (Clustering – Correspondence grouping):	58
3.2.5.	Lọc kết quả (Hypothesis Verification):.....	59
3.3.	GIẢI THUẬT THEO DÕI VẬT THỂ:	60
3.4.	THIẾT LẬP - CÀI ĐẶT HỆ THỐNG CHO GIẢI THUẬT XỬ LÝ ẢNH	61
3.4.1.	Cấu hình hệ thống:	62
3.4.2.	Cài đặt thư viện và thử nghiệm:.....	62
3.4.3.	Thử nghiệm:	63
CHƯƠNG 4:	CHƯƠNG TRÌNH ĐIỀU KHIỂN TRUNG TÂM:	69
4.1.	SƠ ĐỒ VỊ TRÍ MÔ HÌNH LUẬN VĂN:	69
4.2.	SƠ ĐỒ KHÓI TỔNG QUAN CHƯƠNG TRÌNH ĐIỀU KHIỂN:	69

4.3. CHƯƠNG TRÌNH NHÚNG TRÊN VI ĐIỀU KHIỂN:	71
4.4. CHƯƠNG TRÌNH GIAO DIỆN NGƯỜI DÙNG VÀ XỬ LÍ ẢNH TRÊN MÁY TÍNH:.....	76
CHƯƠNG 5: KẾT QUẢ - KẾT LUẬN – HƯỚNG PHÁT TRIỂN:.....	80
5.1. NGHIỆM THU KẾT QUẢ:.....	80
5.1.1. ĐIỀU KHIỂN CÁNH TAY ROBOT ĐỘC LẬP	80
5.1.2. CHƯƠNG TRÌNH XỬ LÍ ẢNH.....	90
5.1.3. HỆ THỐNG TAY ROBOT KẾT HỢP XỬ LÍ ẢNH:	93
5.2. TỰ NHẬN XÉT VÀ ĐÁNH GIÁ:	96
5.2.1. CÁC MỤC TIÊU ĐÃ THỰC HIỆN ĐƯỢC:.....	96
5.2.2. NHỮNG KHUYẾT ĐIỂM CẦN KHẮC PHỤC:	96
5.3. HƯỚNG PHÁT TRIỂN ĐÈ TÀI LUẬN VĂN:.....	98
5.3.1. KHẢ NĂNG ÚNG DỤNG:.....	98
5.3.2. GỢI Ý PHƯƠNG HƯỚNG PHÁT TRIỂN:	98
TÀI LIỆU THAM KHẢO.....	100

DANH MỤC BẢNG

Bảng 1.1 Số lượng robot sử dụng trong cách ngành sản xuất.	6
Bảng 1.2 Số lượng robot công nghiệp sản xuất hàng năm.	6
Bảng 1.3 Top 9 công ty sản xuất robot công nghiệp hàng đầu.	7
Bảng 2.1 Bảng thông số Denavit-Hartenberg.....	24
Bảng 2.2 Bảng cấu hình chức năng AF.....	30
Bảng 3.1 Bảng thuộc tính PCL.....	52
Bảng 3.2 Bảng HEADER của PCD.....	53
Bảng 4.1 Protocol giao tiếp USART PC - VĐK.....	70
Bảng 4.2 Các lệnh gửi từ PC đến STM32F407VGT.	71
Bảng 4.3 Các lệnh gửi từ STM32F407VGT đến PC.	71
Bảng 4.4 Bảng giải thích chức năng nút nhấn trên giao diện.	77
Bảng 5.1 Tọa độ thực tế và sai số Euclidean khi điểm đặt tại [250; 200; 70]....	89
Bảng 5.2 Tọa độ thực tế và sai số Euclidean khi điểm đặt tại [150; -300; 20] ..	89
Bảng 5.1 Thời gian xử lý các hàm tiền xử lý dữ liệu hình ảnh.	91
Bảng 5.2 Thời gian xử lý các hàm xử lý nhận diện và theo dõi hình ảnh.....	91

DANH MỤC HÌNH ẢNH

Hình 2.1 Kí hiệu khớp xoay và khớp tịnh tiến trong các cơ hê.....	10
Hình 2.2 Hệ tọa độ cơ bản và hệ tọa độ suy rộng của robot.....	11
Hình 2.3 Mặt chiếu ngang Workspace.	13
Hình 2.4 Mặt chiếu đứng Workspace.	13
Hình 2.5 Cấu trúc chung của một robot công nghiệp.....	13
Hình 2.6 Robot dạng RRR (bên trái) và Robot dạng RRP (bên phải).....	14
Hình 2.7 Kết cấu robot SCARA – RRP.....	15
Hình 2.8 Robot ABB 6 DOF.....	15
Hình 2.9 Cơ cấu robot SCORBOT.....	16
Hình 2.10 Mô hình thiết kế toàn cảnh.	18
Hình 2.11 Phần chân đế robot.	19
Hình 2.12 Phần thân trên robot.	19
Hình 2.13 Toàn bộ phần thân trên và cánh tay robot.	20
Hình 2.14 Thông số kỹ thuật tay gấp gripper.	22
Hình 2.15 Đặt hệ trục tọa độ từng khâu của robot.	23
Hình 2.16 Sơ đồ chân STM32F407VGT.....	27
Hình 2.17 Cấu tạo Encoder tương đối.	30
Hình 2.18 Xác định chiều thông qua xung Encoder.	31
Hình 2.19 Incremental Encoder 200PPR.....	31
Hình 2.20 Cầu H Arduino IBT-2 BTS7960 43A	32
Hình 2.21 Cầu H L298.....	33
Hình 2.22 Bộ nguồn 24VDC, 20A, 480W.....	34
Hình 2.23 Bộ nguồn 12VDC 15A.	35
Hình 2.24 Bộ nguồn 12VDC, 5A.....	36
Hình 2.25 Bản vẽ schematic sơ đồ chân nối board điều khiển.....	37
Hình 2.26 Hình mạch thực tế hoàn chỉnh.	37
Hình 2.27 Bộ ước lượng bình phương tối thiểu đệ qui.	38
Hình 2.28 Bộ điều khiển theo mô hình chuẩn.	41
Hình 3.1 Thiết bị Kinect.	46
Hình 3.2 Cấu tạo bên trong của Kinect.	47
Hình 3.3 Cấu trúc khối xử lí và truyền nhận Kinect.	48
Hình 3.4 Phương pháp đo giá trị độ sâu Kinect.	48
Hình 3.5 Logo Point Cloud Library.	49
Hình 3.6 Giải thuật xử lý ảnh.....	53
Hình 3.7 Giải thuật nhận dạng và xác định vị trí vật thể.....	54
Hình 3.8 Minh họa voxel grid.....	55
Hình 3.9 Minh họa cấu trúc của SHOT	57
Hình 3.10 Sơ đồ hệ trục tọa độ gốc robot và trục tọa độ gốc tọa độ Kinect.	61

Hình 3.11 Chương trình mẫu pcl_openni_viewer.....	64
Hình 3.12 Thời gian xử lý các hàm tiền xử lý dữ liệu hình ảnh.....	66
Hình 3.13 Thời gian xử lý các hàm xử lý nhận diện và theo dõi hình ảnh.	67
Hình 3.14 Hệ trục tọa độ của Kinect trên OpenNI.....	67
Hình 4.1 Sơ đồ bố trí vị trí robot công nghiệp, camera Kinect và vật thể cần nhận diện và theo dõi.	69
Hình 4.2 Sơ đồ khái quát quan chương trình điều khiển.	70
Hình 4.3 Sơ đồ trạng thái vòng lặp main().	73
Hình 4.4 Sơ đồ trạng thái chương trình ngắt theo chu kỳ 10 ms.	73
Hình 4.5 Sơ đồ thuật toán qui hoạch quỹ đạo Path Planning.	74
Hình 4.6 Sơ đồ bộ điều khiển vị trí STR MRAS.	75
Hình 4.7 Giao diện người dùng trên máy tính.	76
Hình 4.8 Sơ đồ trạng thái chương trình điều khiển và xử lý ảnh trên máy tính..	78
Hình 5.1 Mô hình hoàn thiện cánh tay robot 5 DOF SCORBOT.....	80
Hình 5.2 Bộ điều khiển trung tâm cánh tay robot 5 DOF SCORBOT.	81
Hình 5.3 Đồ thị giám sát sai số trực Error! Bookmark not defined.	
Hình 5.4 Đồ thị giám sát sai số trực SHOULDER.....Error! Bookmark not defined.	
Hình 5.5 Đồ thị giám sát sai số trực ELBOW.... Error! Bookmark not defined.	
Hình 5.6 Đồ thị giám sát sai số khớp PITCH Error! Bookmark not defined.	
Hình 5.7 Đồ thị giám sát sai số khớp ROLL Error! Bookmark not defined.	
Hình 5.8 Đồ thị giám sát sai số độ dời tiến tới điểm 1	85
Hình 5.9 Đồ thị giám sát sai số độ dời tại điểm 2	86
Hình 5.10 Đồ thị giám sát sai số độ dời tại điểm 3	86
Hình 5.11 Đồ thị giám sát sai số độ dời tại điểm 4	87
Hình 5.12 Đồ thị giám sát sai số độ dời tại điểm 5	88
Hình 5.13 Giao diện người dùng trên máy tính.	90
Hình 5.14 Đồ thị biến thiên giá trị đo trực x theo thời gian.	92
Hình 5.15 Đồ thị biến thiên giá trị đo trực y theo thời gian.	92
Hình 5.16 Đồ thị biến thiên giá trị đo trực z theo thời gian.....	93
Hình 5.17 Sơ đồ bố trí hệ thống.	94
Hình 5.18 Sơ đồ bố trí thực tế của hệ thống.	94
Hình 5.19 Sai số độ dời khi vận hành tích hợp xử lý ảnh thời gian thực.....	95

TÓM TẮT LUẬN VĂN BẰNG TIẾNG VIỆT

Tóm tắt: Đề tài này xây dựng một hệ thống điều khiển cánh tay máy 5 bậc tự do hoàn chỉnh: từ thiết kế cơ khí đến thiết kế giao diện người dùng bằng Qt. Hệ thống ứng dụng thuật toán điều khiển động cơ Tự chỉnh theo mô hình chuẩn (STR-MRAS) và kết hợp với các thuật toán xử lý ảnh mới mẽ từ thư viện PCL và camera Kinect để tạo thành vòng điều khiển kín, tự động nhận dạng, theo dõi và tương tác với vật thể trong không gian làm việc.

Từ khóa: Cánh tay máy, STR-MRAS, Xử lý ảnh, PCL, Nhận dạng, Theo dõi, Vòng điều khiển kín.

ABSTRACT

Abstract: This project is meant to test all the acquired knowledge at university: building a complete control system of a 5 DOFs robot manipulator from scratch. The system utilizes Self-Tuning Regulator – Model Reference Adaptive System (STR-MRAS) to control DC motors, meanwhile integrating with some newest image processing algorithm from PCL and Kinect to form a closed-loop control system.

Keywords: Robot manipulator, STR-MRAS, Image processing, PCL, Closed-loop.

CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN ĐỀ TÀI

1.1. GIỚI THIỆU TỔNG QUAN VỀ ROBOT CÔNG NGHIỆP:

1.1.1. Lịch sử phát triển robot công nghiệp:

Robot là hệ thống cơ khí có chuyên động tương tự sinh vật và có trí thông minh và có trí thông minh, hoạt động theo sự điều khiển của con người. Robot về cơ bản được chia làm hai loại là robot công nghiệp và robot di động. Trong đề tài luận văn này, ta sẽ tìm hiểu sơ lược về robot công nghiệp (còn gọi là tay máy robot).

Theo khái niệm cơ bản nhất, robot công nghiệp (Industrial Robot hay viết tắt là IR) có thể được hiểu là những thiết bị tự động linh hoạt, thực hiện các chức năng lao động công nghiệp của con người dưới một hệ thống điều khiển theo những chương trình đã được lập trình sẵn.

Sơ lược quá trình lịch sử phát triển cánh tay robot công nghiệp trên thế giới:

- Vào năm 1921 Thuật ngữ “Robot” xuất phát từ tiếng Séc (Czech) “Robota” có nghĩa là công việc tạp dịch trong một vở kịch.
- Năm 1950 ở Mỹ thành lập viện nghiên cứu đầu tiên.
- Đầu năm 1960 công ty AMF cho ra đời sản phẩm đầu tiên có tên gọi là Versatran.
- Từ năm 1967, ở Anh, người ta đã bắt đầu nghiên cứu và chế tạo IR.
- Từ năm 1968, ở Châu Á, Nhật bắt đầu nghiên cứu những ứng dụng của IR, năm 1970, Robot đã được chú ý nhiều hơn và bắt đầu xuất hiện ở các nước Đức, Ý, Pháp...
- Nhát là vào những năm 1990 số lượng Robot công nghiệp đã gia tăng với nhiều tính năng vượt bậc.
- Đến nay, trên thế giới có khoảng trên 200 công ty sản xuất IR. Trong đó Mỹ và Nhật chiếm đa số.

Trong quá trình phát triển, theo chủng loại, mức độ điều khiển và khả năng nhận biết thông tin của tay máy – người máy, người máy trên thế giới có thể phân loại các IR thành các thé hệ sau:

- Thé hệ 1: Thé hệ có thể điều khiển theo chu trình dạng chương trình cứng không có khả năng nhận biết thông tin.
- Thé hệ 2: Thé hệ có điều khiển theo chu kỳ dạng chương trình mềm bước đầu có khả năng nhận biết thông tin.
- Thé hệ 3: Thé hệ có thể điều khiển dạng tinh khôn, có khả năng nhận biết thông tin và bước đầu có một số chức năng lí trí của con người.

Đối với tay máy công nghiệp đã có hơn 250 loại và trong đó có hơn 40% là tay máy có thể điều khiển đơn giản thuộc thé hệ thứ 1.

Ở Việt Nam, theo xu hướng công nghiệp hóa hiện đại hóa, các dây chuyền sản xuất ngày càng đổi hỏi các hệ thống tự động chính xác để tăng hiệu suất làm việc, vì thế trong 25 năm qua việc nghiên cứu và phát triển robot công nghiệp đã có những bước tiến đáng kể khi được các khoa cơ khí, chế tạo máy móc, điện tử tự động hóa và cả các công ty, viện nghiên cứu quan tâm nghiên cứu và chế tạo nhiều sản phẩm ấn tượng trên trường quốc tế.

Trong xuyên suốt quá trình hình thành và phát triển, robot công nghiệp được phân loại theo nhiều cách khác nhau theo từng đặc tính, trong đó phổ biến nhất là phân loại theo kết cấu, phương pháp điều khiển và phân loại theo ứng dụng.

- Phân loại theo kết cấu:
 - Robot chuỗi: Là một chuỗi động học hở với một khâu cố định gọi là đế và các khâu động, trong đó các khâu động được bố trí nối tiếp với nhau. Mỗi khâu động được liên kết hay nối động với một khâu khác nhờ các khớp liên kết.
 - Robot song song: Là một chuỗi động học kín, ở đó mỗi khâu luôn luôn được liên kết với ít nhất hai khâu khác.
 - Robot có khớp nối: Là robot có những khớp quay. Robot có khớp có thể có hai kết cấu nối với nhau rất đơn giản đến những hệ thống có tới hơn 10 kết cấu tương tác với nhau. Chúng có thể dùng để nhắc các chi tiết nhỏ với độ chính xác cực cao.

- Phân loại theo phương pháp điều khiển:
 - Điều khiển hở: Dùng truyền động bước mà quãng đường hoặc góc dịch chuyển tỷ lệ với xung điều khiển. Kiểu này đơn giản nhưng cho độ chính xác thấp.
 - Điều khiển kín: Điều khiển kiểu servo, sử dụng tín hiệu phản hồi vị trí để tăng độ chính xác điều khiển.
 - Kiểu điều khiển điểm-điểm: Phần công tác dịch chuyển từ điểm này đến điểm kia theo đường thẳng với tốc độ không cao, thường được dùng trên các Robot hàn điểm, vận chuyển, tán định và bắn định.
 - Điều khiển contour: Đảm bảo cho phần công tác dịch chuyển theo quỹ đạo bất kì, với tốc độ có thể điều khiển được. Có thể gấp kiểu điều khiển này trên các Robot hàn hồ quang và phun sơn.
- Phân loại theo ứng dụng:

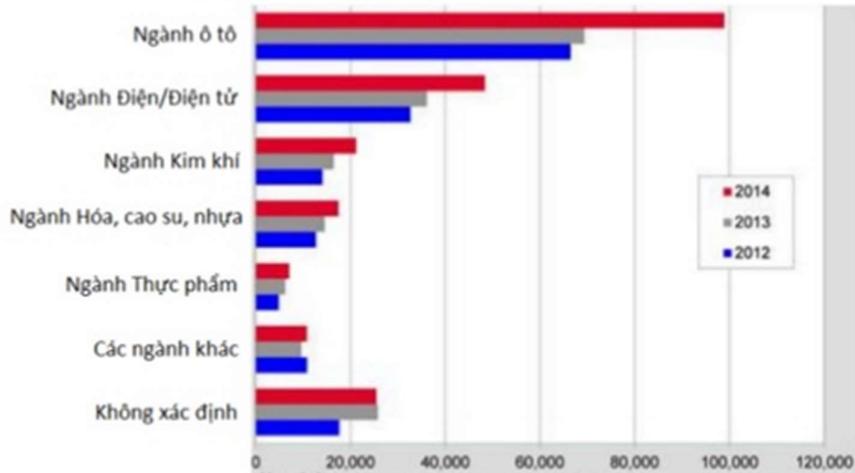
Dựa vào những ứng dụng của robot trong sản xuất ta có những loại robot sau: robot sơn, robot hàn, robot lắp ráp, robot chuyển phôi...

1.1.2. Tình hình nghiên cứu và ứng dụng của robot công nghiệp:

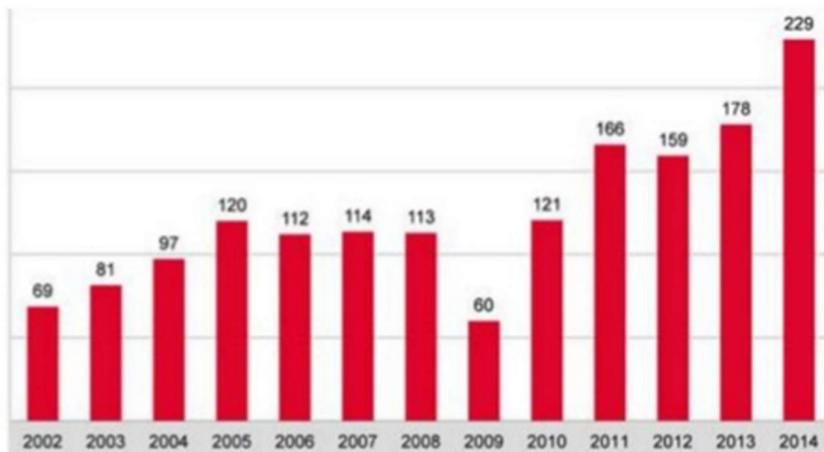
Robot giúp giải quyết các vấn đề thiếu hụt lao động có tay nghề cao, giảm lao động nặng nhọc cho công nhân, giảm giá thành sản phẩm, tăng năng suất trong nhiều ngành sản xuất công nghiệp, dịch vụ, y tế, chăm sóc sức khỏe, nông nghiệp, xây dựng, duy tu bảo dưỡng ...

Cụ thể, robot công nghiệp được dùng trong các lĩnh vực sau:

- Robot công nghiệp được sử dụng phổ biến nhất trong dây chuyền sản xuất tự động: lắp ráp, sơn, hàn, bóc dỡ hàng, trong công nghiệp ô tô, cơ khí, ... sử dụng trong dây chuyền sản xuất mềm dẻo kết hợp với máy CNC.
- Robot công nghiệp sử dụng nhiều nhất trong ngành sản xuất ô tô.



Bảng 1.1 Số lượng robot sử dụng trong cách ngành sản xuất.



Bảng 1.2 Số lượng robot công nghiệp sản xuất hằng năm.

Như bảng số liệu như trên, chúng ta thấy rõ ràng nhu cầu ứng dụng của robot công nghiệp ngày càng tăng dưới yêu cầu nâng cao năng suất của những chất lượng sản xuất, nhờ đó kéo theo ngành nghiên cứu và sản xuất robot công nghiệp phát triển mạnh mẽ. Các công ty lớn sản xuất robot công nghiệp thường là các công ty đa quốc gia với trụ sở đặt ở các quốc gia có nền tảng kĩ thuật khoa học tiên bội hàng đầu như Mỹ, Nhật, Đức.

Tên công ty	Trụ sở	Số lượng robot đã sản xuất (ngàn đơn vị)
Yaskawa Electric Corp	USA	300
ABB Robotics	Switzerland	250

Fanuc America	USA	250
Kawasaki Robotics	USA	110
Nachi Fujikoshi	Japan	100
Kuka AG	Germany	80
Denso Robotics	Japan	80
Epson	Japan	45
Adept Technology	USA	25

Bảng 1.3 Top 9 công ty sản xuất robot công nghiệp hàng đầu.

Gần đây sản phẩm của Universal Robotics được đánh giá rất cao nhờ thiết kế linh hoạt, gọn gàng, thực hiện được nhiều chuyển động phức tạp hơn so với các mẫu mô hình robot công nghiệp đời trước. Hay cũng không thể không nhắc tới các công ty robot đến từ Trung Quốc đang ngày càng phát triển cạnh tranh với các tập đoàn robot có chỗ đứng vững chắc trên thị trường.

Ngoài ra ở Việt Nam cũng đã có nhiều công ty doanh nghiệp thực hiện nghiên cứu, thiết kế và chế tạo các tay robot công nghiệp có thể đưa vào sản xuất như Robotics 3T.

Việc nghiên cứu xây dựng mô hình cánh tay robot độc lập nay đã được mở rộng hơn để tích hợp các công nghệ mới theo xu hướng phát triển như tích hợp cách thuật toán xử lý ảnh, thuật toán trí tuệ nhân tạo, các thuật toán điều khiển mới, các cơ cấu phát hiện và dừng chuyển động khi có va chạm...

1.2. MỤC ĐÍCH NGHIÊN CỨU:

Như đã thấy từ phần giới thiệu tổng quan về robot công nghiệp, đề tài xây dựng mô hình cánh tay robot công nghiệp không phải là một đề tài mới trong việc nghiên cứu và phát triển khi đã được nhiều tập đoàn, công ty, và cả các dự án nghiên cứu không chỉ ngoài mà cả trong nước, trong các trường đại học khai thác.

Song ứng dụng của robot công nghiệp trong thực tế vẫn rất rộng rãi và nhiều cơ hội trong thời đại phát triển bùng nổ về tự động hóa, các dây chuyền tự động và cả trong cách mạng công nghiệp 4.0 như ngày nay. Việc nghiên cứu và học về robot công nghiệp vẫn cần thiết để làm chủ hoàn toàn công nghệ.

Chính vì vậy, nhóm đánh giá đây vẫn là một đề tài rất thử thách đối với sinh viên để xây dựng một cánh tay robot hoàn chỉnh, đặc biệt trong quá trình thực hiện sẽ tìm hiểu thêm được những kiến thức về kết cấu cơ khí rất bổ ích trong thực tế vì việc nghiên cứu các thuật toán điều khiển cần được áp dụng thực tế vào những mô hình cụ thể, từng bước tiếp cận với thực tiễn, từng bước góp phần làm chủ công nghệ trong tương lai.

Không dừng lại chỉ ở mục đích chỉ xây dựng mô hình và bộ điều khiển cánh tay robot công nghiệp, để từng bước tiếp cận với các ứng dụng xử lý ảnh đang rất được quan tâm và phát triển trong thời gian gần đây, nhóm sẽ đồng thời xây dựng ứng dụng xử lý ảnh (Computer Vision) giải thuật tích hợp nhận diện và xử lý hình ảnh trong không gian 3 chiều, sử dụng camera xác định tọa độ vật trong không gian để truyền tọa độ để cánh tay thực thi lệnh di chuyển bám vật, gấp thả vật tự động, giúp tay vận hành linh động hơn không bị cố định theo một quỹ đạo định sẵn cụ thể. Đây là điểm mới trong ứng dụng về mục đích nghiên cứu trong luận văn mà nhóm sẽ thực hiện.

1.3. MỤC TIÊU LUẬN VĂN:

Trong quá trình nghiên cứu, phát triển và thực hiện luận văn, nhóm đề ra các mục tiêu cần thực hiện trong luận văn như sau:

- Tìm hiểu, phân tích cơ bản các cơ cấu cơ khí để thực hiện xây dựng mô hình cánh tay robot công nghiệp.
- Xây dựng bộ điều khiển trung tâm cho cánh tay robot công nghiệp hoàn chỉnh, giao tiếp được với máy tính. Bộ điều khiển bao gồm cả phần cứng, phần mềm và nguồn cung cấp.
- Các giải thuật điều khiển động cơ trên từng khớp đạt được chất lượng điều khiển tốt với cơ cấu cơ khí tương ứng.
- Xây dựng chương trình nhận dạng vật, bám theo chuyển động của vật và xác định tọa độ ba chiều của vật theo thời gian thực.
- Xây dựng giao diện điều khiển và giao tiếp giữa bộ điều khiển cánh tay robot và máy tính có thực thi giải thuật xử lý ảnh nêu trên. Hệ thống vận hành ổn định và chính xác.

1.4. PHƯƠNG PHÁP THỰC HIỆN:

Phương pháp thực hiện các mục tiêu của luận văn bám sát theo các mục tiêu của luận văn đã đề ra ở mục nêu trên:

- Trước tiên ta cần phân tích các mô hình robot công nghiệp hiện có để chọn và thiết kế mô hình cơ khí cánh tay robot phù hợp.
- Tiếp theo ta cần nghiên cứu phương pháp xây dựng hệ trực tọa độ để chuyển tọa độ robot sang các biến điều khiển trên từng động cơ (xoay hoặc tinh tiến), từ đó xây dựng bài giải các bài toán động học thuận và nghịch.
- Song song đó ta xây dựng bộ điều khiển vị trí động cơ cho từng khớp cánh tay robot. Thách thức lớn ở đây là bộ điều khiển cần điều khiển chính xác trong điều kiện mô men tải động cơ thay đổi liên tục.
- Xây dựng bộ điều khiển phần cứng để thu về các tín hiệu cảm biến, tính toán và xuất ra các tín hiệu điều khiển, cũng như cấp nguồn động cơ hoàn chỉnh để điều khiển được mô hình cánh tay robot.
- Nghiên cứu các thuật toán, thư viện, và thiết bị camera để lựa chọn giải thuật xử lý ảnh phù hợp. Từ đó phát triển và tối ưu tốc độ xử lý cho thuật toán xử lý hình ảnh đáp ứng yêu cầu của mục tiêu.
- Xây dựng bộ điều khiển xử lý ảnh trên máy tính khai thác phần cứng camera và giao tiếp tốt với bộ điều khiển cánh tay robot.

CHƯƠNG 2: THIẾT KẾ CƠ KHÍ VÀ BỘ ĐIỀU KHIỂN CÁNH TAY ROBOT

2.1. ĐỊNH NGHĨA - KHÁI NIỆM CƠ BẢN TRONG ROBOT:

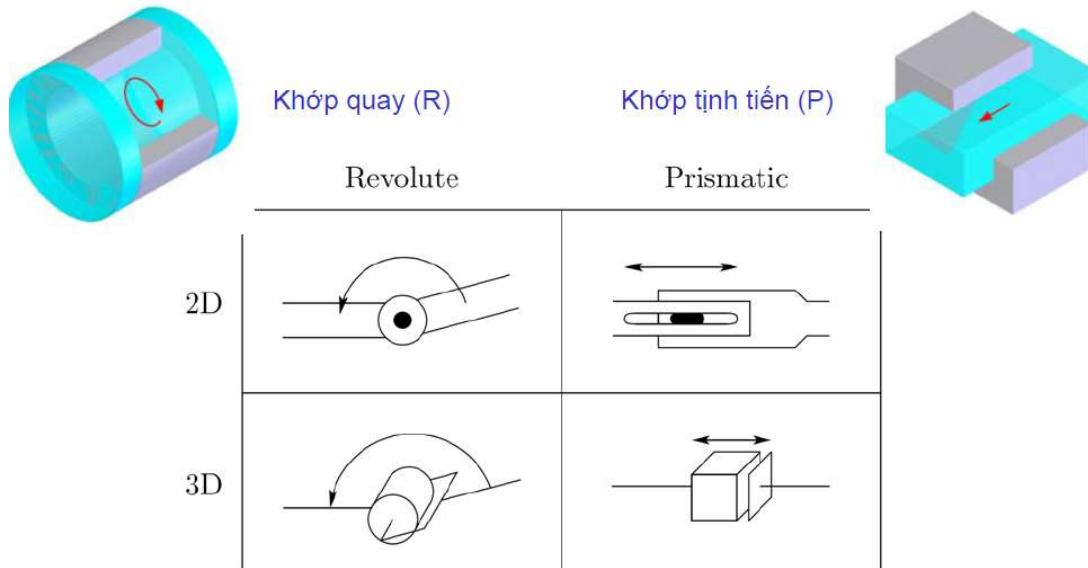
2.1.1. Bậc tự do – Degree of Freedom (DOF):

Bậc tự do là số khả năng chuyển động của một cơ cấu (chuyển động quay hoặc tịnh tiến). Để dịch chuyển được một vật thể trong không gian, cơ cấu chấp hành của robot phải đạt được một số bậc tự do. Nói chung cơ hệ của robot là một cơ cấu hở, do đó bậc tự do của nó có thể tính theo công thức như sau:

$$w = 6n - \sum_{i=0}^5 ip_i$$

Với: n là số khâu động; p_i là số khớp loại i ($i=1, 2..5$: số bậc bị hạn chế).

Đối với các cơ cấu có các khâu được nối với nhau bằng khớp quay hoặc tịnh tiến (khớp động loại 5) thì số bậc tự do bằng với số khâu động. Đối với hệ hở, số bậc tự do bằng tổng số bậc tự do của các khớp động.



Hình 2.1 Kí hiệu khớp xoay và khớp tịnh tiến trong các cơ hệ.

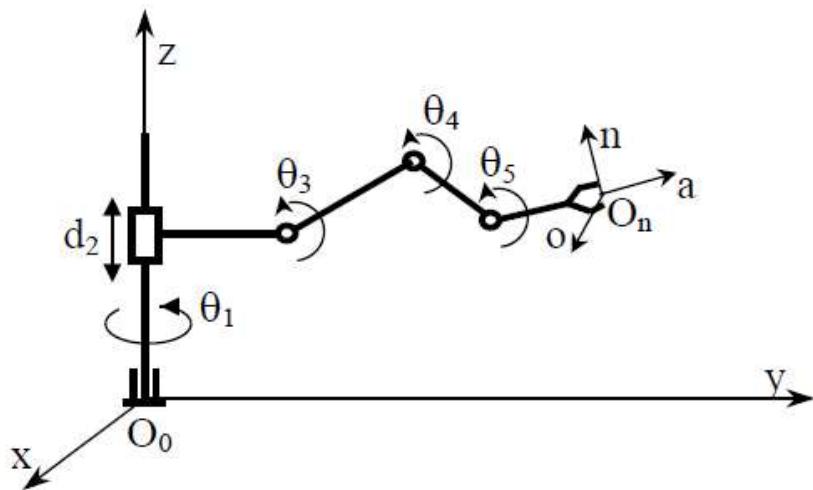
Như vậy với cấu hình robot như SCORBOT hoặc MITSUBISHI RV-5-AJ, đây là các loại robot có 5 bậc tự do tương ứng với 5 khớp xoay.

2.1.2. Hệ tọa độ - Hệ trục tọa độ:

Mỗi robot thường được cấu tạo bởi nhiều khâu (links) liên kết với nhau qua các khớp (joints) tạo thành một xích động học xuất phát từ một khâu cơ bản (base) đứng yên (thường được đánh số thứ tự 0). Các hệ tọa độ trung gian khác gắn với các khâu động gọi là hệ tọa độ suy rộng.

Trong từng thời điểm hoạt động, các tọa độ suy rộng xác định cấu hình của robot bằng các chuyển dịch dài đối với khớp tịnh tiến hay chuyển động góc đối với khớp xoay.

Ngoài ra có loại hệ trục tọa độ thường dùng cho là hệ trục tọa độ Decartes (XYZ), hệ trục tọa độ cầu hay hệ trục tọa độ trụ, trong phạm vi luận văn chỉ sử dụng hệ trục tọa độ Decartes với 3 trục OxOyOz.

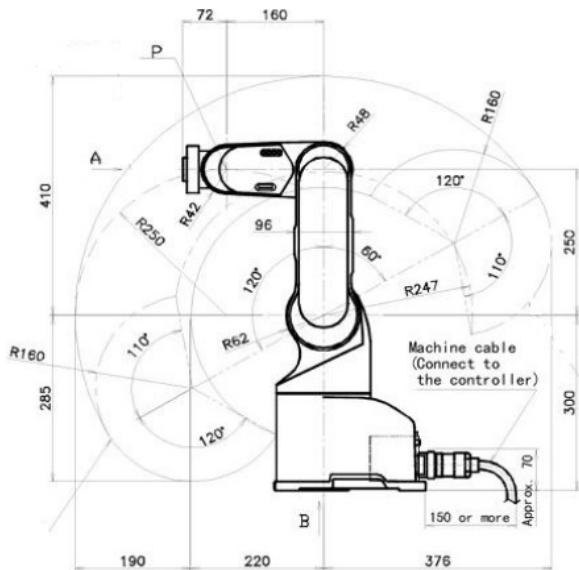


Hình 2.2 Hệ tọa độ cơ bản và hệ tọa độ suy rộng của robot.

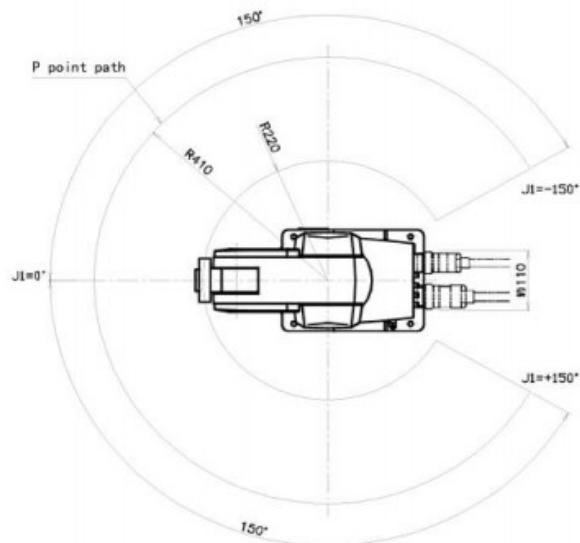
2.1.3. Trường công tác - Workspace or Range of motion:

Trường công tác của robot là toàn bộ phần thể tích được quét bởi khâu chấp hành cuối khi robot thực hiện tất cả các chuyển động có thể. Trường công tác bị ràng buộc bởi các thông số hình học của robot cũng như các ràng buộc cơ học của các khớp.

Trong các bảng Specification, người ta thường dùng hai hình chiếu đứng và hình chiếu ngang để mô tả trường công tác của robot. Hoặc trong các mô phỏng chuyển động 3D, người ta có thể mô tả trường công tác của robot bằng cách dựng giới hạn biên lớn nhất và nhỏ nhất mà điểm cuối robot có thể di chuyển tới.

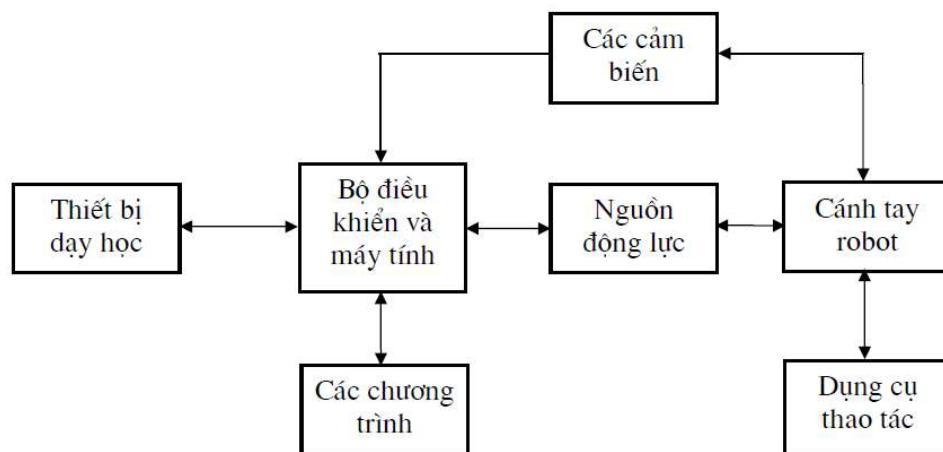


Hình 2.3 Mặt chiếu ngang Workspace.



Hình 2.4 Mặt chiếu đứng Workspace.

2.1.4. Cấu trúc chung của một robot công nghiệp:



Hình 2.5 Cấu trúc chung của một robot công nghiệp.

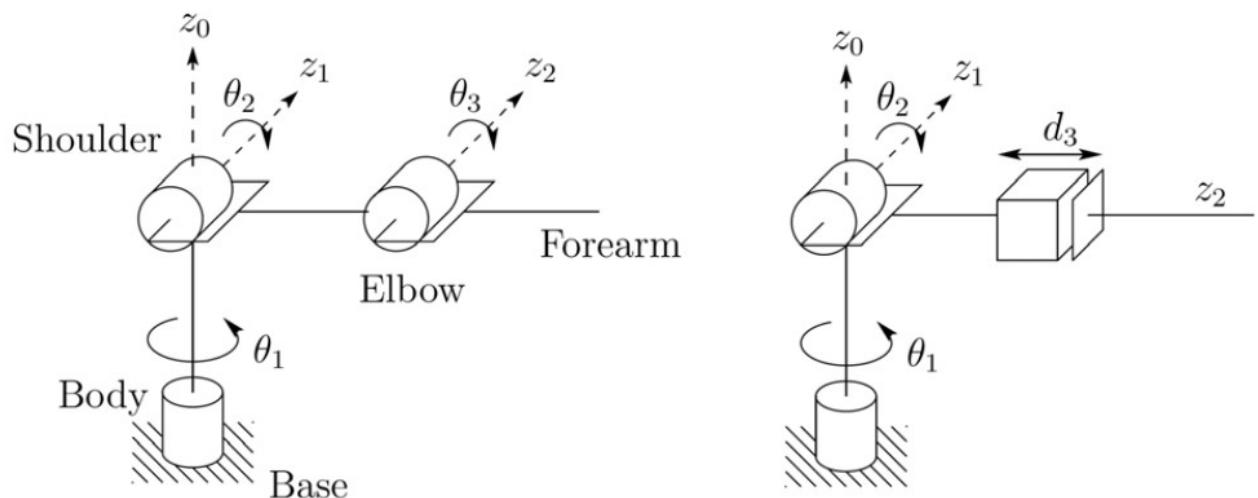
- Phần cánh tay: kết cấu cơ khí bao gồm các khâu liên kết với các khớp.
- Nguồn động lực: bao gồm động cơ điện, cơ cầu thủy lực, khí nén trực tiếp tác động làm cơ cầu cơ khi dịch chuyển.
- Cảm biến: thường dùng nhất là Encoder gắn với động cơ, các loại biến trở công nghiệp có sai số thấp hay công tắc hành trình.

- Dụng cụ thao tác (end-effector) là khâu chấp hành cuối, có thể là tay gấp, giác hút, đầu hàn, đầu phun sơn,...
- Bộ điều khiển và máy tính + Các chương trình: chương trình nhúng được thiết kế và áp dụng các giải thuật để điều khiển các khớp một cách chính xác nhất.
- Thiết bị dạy học: các tín hiệu đầu vào để điều khiển robot di chuyển, ví dụ tọa độ điểm đến hoặc các góc quay tương ứng trên từng khớp.

2.1.5. Một số kết cấu robot phổ biến:

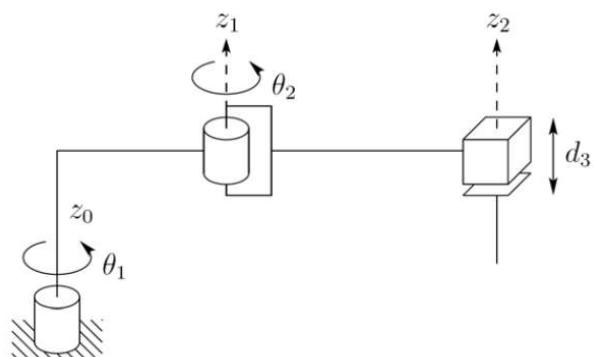
Người ta kí hiệu kết cấu robot theo thứ tự các khớp tịnh tiến (P – Prismatic) hoặc khớp xoay (R – Revolute).

Ví dụ như:



Hình 2.6 Robot dạng RRR (bên trái) và Robot dạng RRP (bên phải).

Ngoài ra còn có một số cơ cấu robot phổ biến: SCARA, SCORBOT, ABB 6 DOF,...

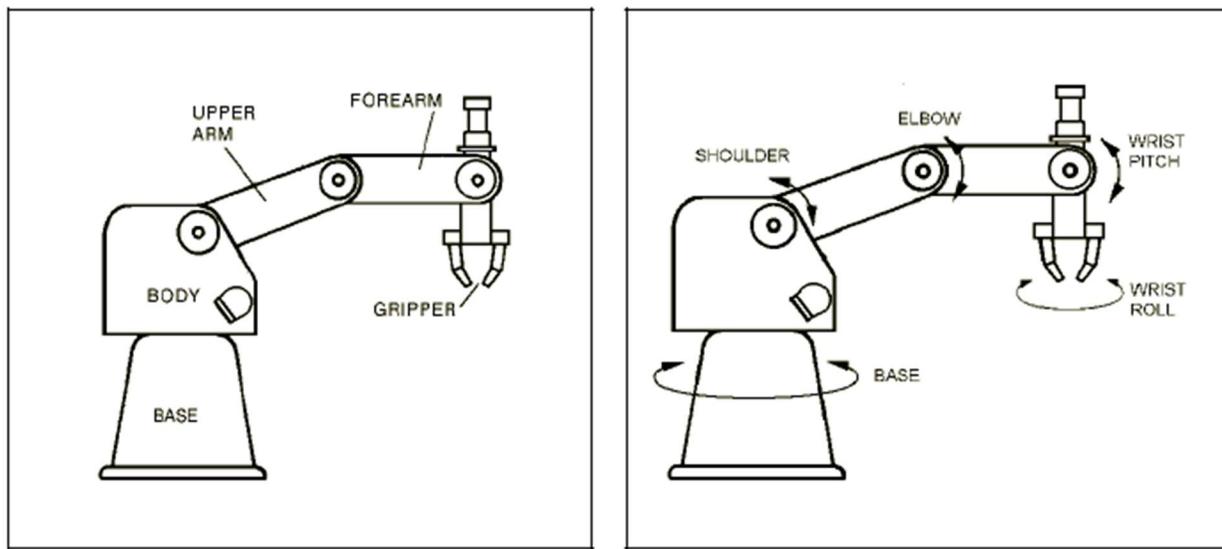


Hình 2.7 Kết cấu robot SCARA – RRP.



Hình 2.8 Robot ABB 6 DOF.

Cơ cấu robot 6 DOF RRRRRR là một cơ cấu được sử dụng nhiều trong robot công nghiệp với ứng dụng lắp ráp hay phun sơn ô tô do độ chính xác cao, linh hoạt.



Hình 2.9 Cơ cấu robot SCORBOT.

Cơ cấu robot SCORBOT 5 DOF, ít hơn 1 bậc tự do so với cơ cấu 6DOF, là cơ cấu robot linh hoạt, thiết kế cơ khí không quá phức tạp, sử dụng các động cơ DC điều khiển phù hợp với các ứng dụng nghiên cứu giải thuật điều khiển cũng như dễ dàng bảo trì thay thế. Sự linh động trong yêu cầu vận hành vẫn cao nhưng việc giải bài toán động học ngược đơn giản hơn so với cơ cấu 6 DOF.

Chính vì lí do đó nên SCORBOT thường được sử dụng trong các mô hình nghiên cứu và và phòng thí nghiệm.

2.1.6. Một số vấn đề cần giải quyết:

Trong việc xây dựng mô hình và bộ điều khiển cánh tay robot trong công nghiệp, có một số thách thức cần được giải quyết như sau:

- Động cơ điều khiển từng khớp sẽ có mô men tải thay đổi liên tục theo thời gian, các động cơ cần đảm bảo đủ sức nâng tải, đồng thời quá trình vận hành không được rung lắc.
- Các mô hình thường gấp đều có động cơ ở vị trí khớp vai cần chịu tải rất lớn, đòi hỏi khi không cấp điện khớp vẫn phải tự đứng yên không bị sập (hiện tượng back-driven).

- Chọn cơ cấu truyền động phù hợp để mô hình vận hành tốt trên từng khớp, điều khiển chính xác vị trí có hồi tiếp vòng kín.
- Hạn chế các hiện tượng khung gây rung lắc mô hình do các khớp di chuyển và dừng lại đột ngột quanh điểm đặt.

Sau khi nghiên cứu và tham khảo, nhóm đã quyết định thực hiện xây dựng mô hình robot theo cơ cấu SCORBOT cũng vì những lí do như sau:

- Mô hình SCORBOT phổ biến trong các mục đích nghiên cứu sinh viên do các cơ cấu truyền động không quá phức tạp, cơ cấu linh hoạt.
- Khi thiết kế mô hình có thể chọn nhiều loại động cơ khác nhau để điều khiển do không gian rộng. Ở đây nhóm sử dụng toàn bộ 5 động cơ DC có Encoder để tạo vòng điều khiển kín trên toàn bộ các động cơ ở từng khớp. Ứng dụng được các bộ điều khiển đã học, ở đây nhóm chọn thực hiện bộ điều khiển STR MRAS và hiệu chỉnh từ nguồn tham khảo có sẵn.
- Nhờ cơ cấu linh hoạt không quá phức tạp nên các linh kiện có thể dễ dàng quan sát và thay thế, phục vụ tốt cho mục đích nghiên cứu tiếp tục về sau.
- Vị trí động cơ khớp vai (Shoulder) nhóm dùng động cơ truyền động trực vít bánh răng đảm bảo không sập khi không cấp điện, các khớp còn lại truyền động bánh răng hoặc truyền động đai. Với truyền động đai cần có phương pháp dự phòng dây đai bị trùng.

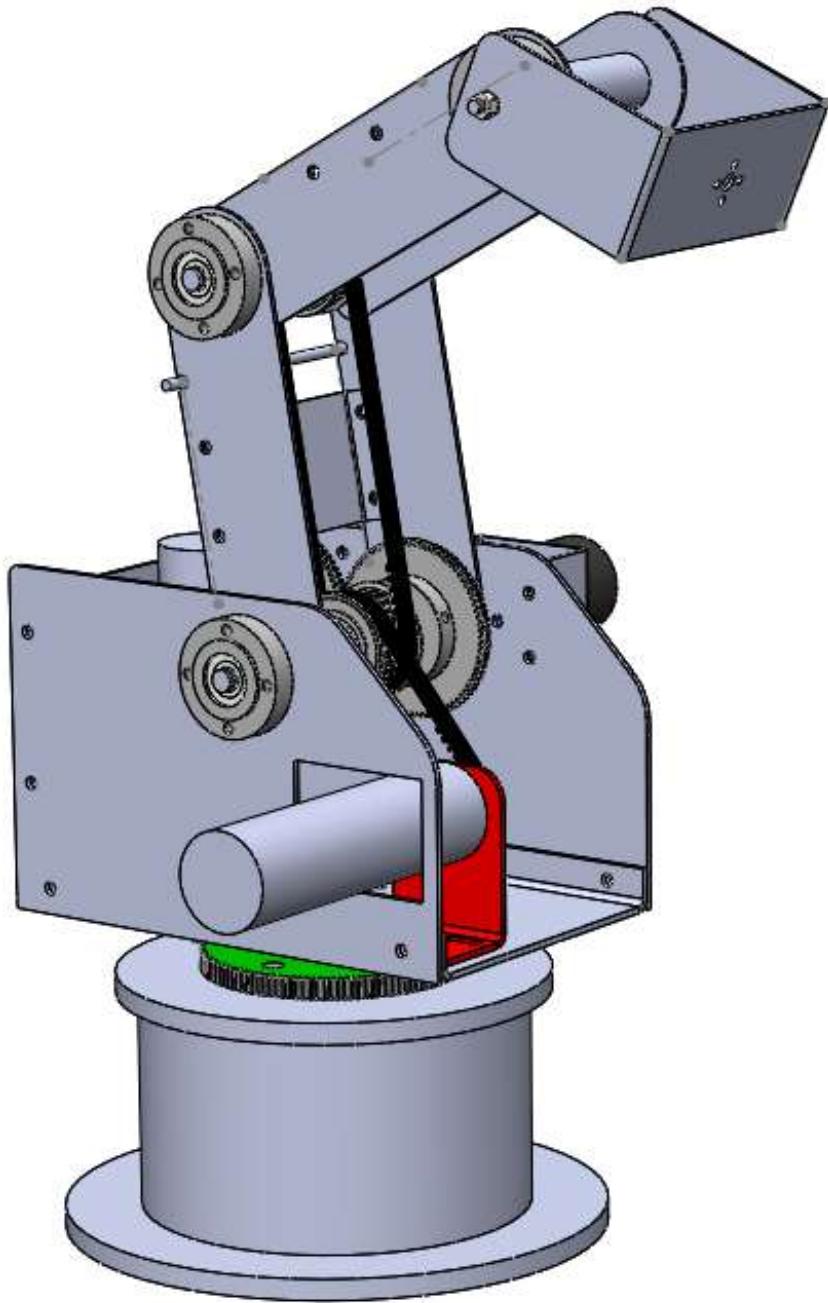
2.2. MÔ HÌNH THIẾT KẾ CƠ KHÍ CÁNH TAY ROBOT:

2.2.1. Mô hình thiết cơ khí và thông số mô hình:

Toàn bộ bản thiết kế sử dụng chương trình Solidworks 2017, tên các linh kiện cơ khí như ổ bi, bánh răng, bạc đạn,... được chọn theo các linh kiện tiêu chuẩn với chất lượng tốt, được liệt kê chi tiết trong bản thiết kế 3D trong chương trình Solidworks.

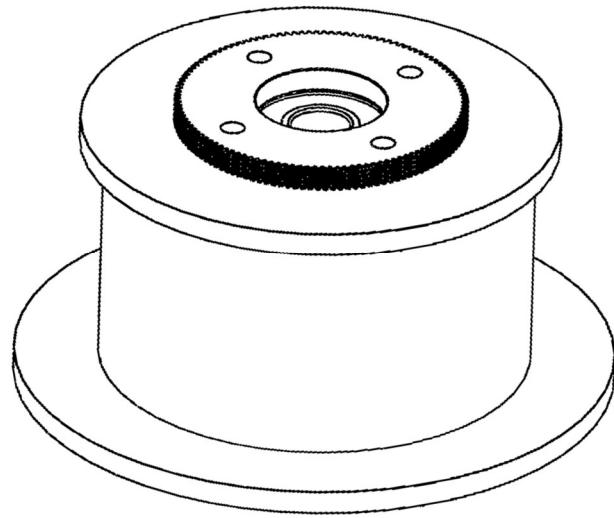
Ở đây nhóm chỉ thể hiện các góc nhìn tổng quan thiết kế của cánh tay robot, một số thông số chi tiết được ghi chi tiết ở các phần sau.

- Mô hình thiết kế tổng quan toàn cảnh:



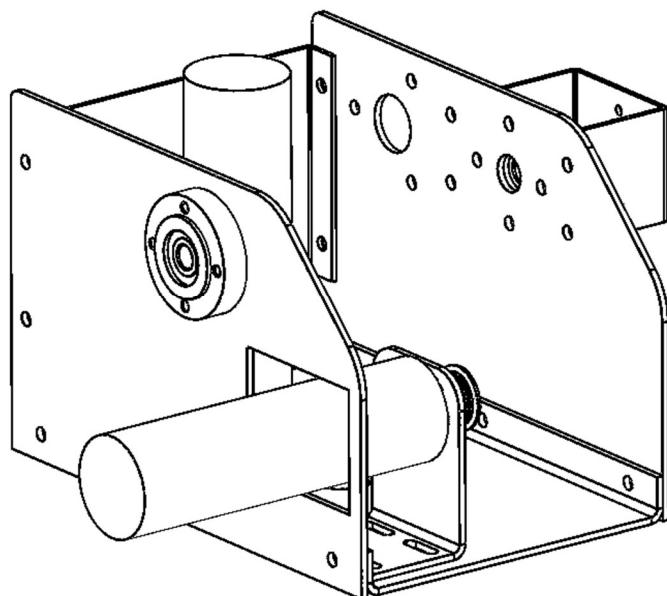
Hình 2.10 Mô hình thiết kế toàn cảnh.

- Phần chân đế Ground Base:



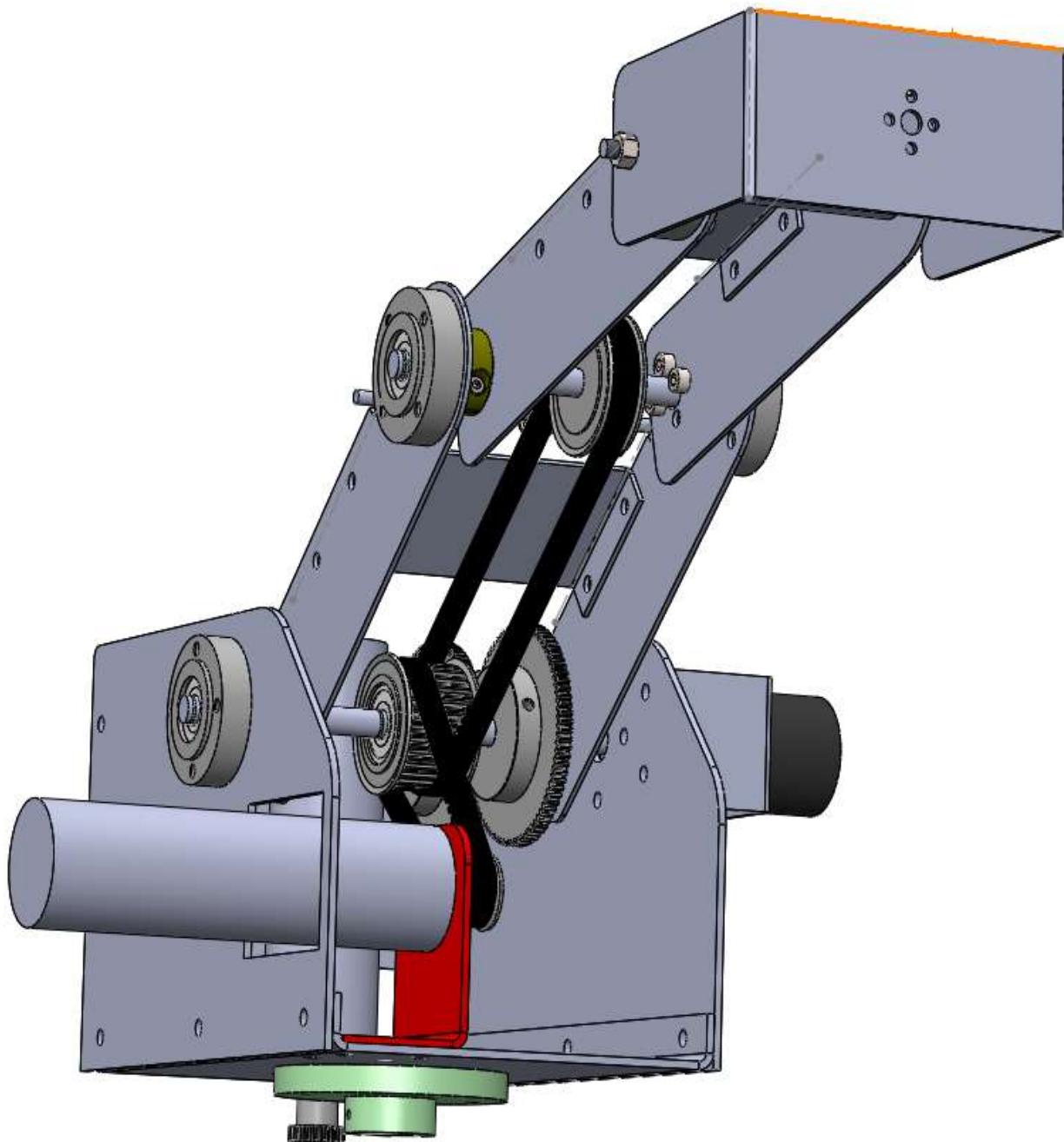
Hình 2.11 Phần chân đế robot.

- Phần thân trên:



Hình 2.12 Phần thân trên robot.

- Toàn bộ phần thân trên và cánh tay:



Hình 2.13 Toàn bộ phần thân trên và cánh tay robot.

Trong cơ cấu truyền động bằng đai sê có hiện tượng trùng dây đai truyền động trong quá trình vận hành lâu dài, khi đó chất lượng điều khiển sẽ giảm. Vì vậy nhóm

có thiết kế thêm các cơ cầu thanh cảng đai đối với ổ bi đôi lên khớp elbow và các lỗ trượt điều chỉnh vị trí của động cơ elbow để cảng đai từ động cơ lên ở bi đôi.

- Toàn bộ thông số thiết kế cánh tay robot:

Chiều dài tay máy	d_1	mm	272
	d_2		20
	d_3		174
	d_4		175
	d_5		230
Phạm vi hoạt động	J1	degree	180 (-90 đến 90)
	J2		75 (5 đến -70)
	J3		180 (90 đến -90)
	J4		180 (0 đến -180)
	J5		180 (-180 đến +180)

Trong chương trình điều khiển, cần lưu ý với mô hình robot như trên, động cơ ở vị trí ELBOW không điều khiển góc θ_3 mà điều khiển góc $\theta_2 + \theta_3$, quá trình xây dựng giải thuật phần mềm và bài toán động học thuận, nghịch cần lưu ý.

2.2.2. Các động cơ truyền động trên từng khớp và tỉ số truyền động cơ khí:

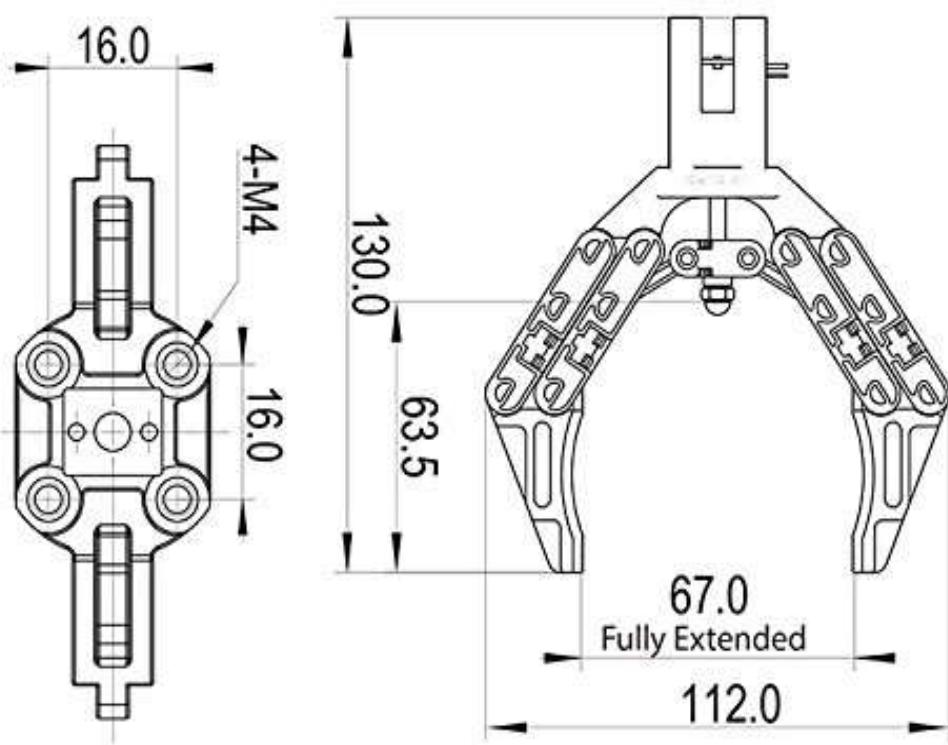
Vị trí khớp	Tên loại động cơ	Điện áp hoạt động	Tỉ số truyền hộp số	Tỉ số truyền cơ khí	Số xung Encoder
BASE	Planetary 24V 60W 70RPM	24V	1:139	1:6	1 PPR
SHOULDER	Worm-Geared DC Motor	12V	X	1:1	200 PPR
ELBOW	Planetary 24V 60W 30RPM	24V	1:264	1:2	13 PPR

PITCH	JGA25-370 19RPM	12V	1:226	X	11 PPR
ROLL	JGA25-370 300RPM	12Vs	1:34	X	11 PPR

Ở đây tại 3 vị trí BASE, SHOULDER, ELBOW, đặc biệt ở 2 vị trí SHOULDER và ELBOW phải chịu tải lớn nhất nên chọn loại động cơ có cơ cấu hộp số cũng như tỉ số truyền lớn, torque cao để trong quá trình không cấp điện các động cơ giữ nguyên vị trí không bị trượt.

2.2.3. Cấu hình tay gấp:

Tay gấp gripper dạng kẹp được làm bằng nhựa có lớp cao su ở mặt trong để tăng ma sát khi gấp vật, tránh trơn trượt vật, đồng thời tay gấp tích hợp sẵn động cơ DC 12V để dễ khiển gấp nhẹ.



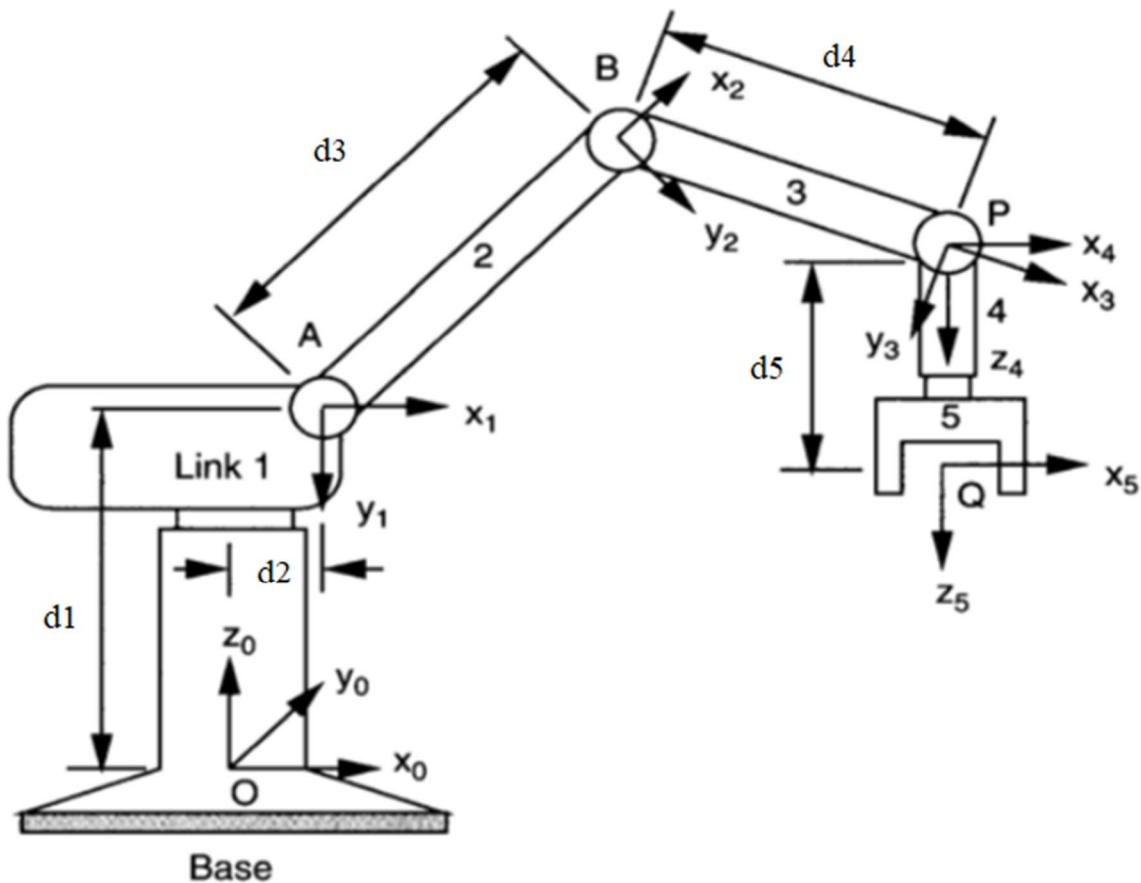
Hình 2.14 Thông số kỹ thuật tay gấp gripper.

2.2.4. Phân tích bài toán động học thuận – nghịch:

Quá trình xây dựng và giải bài toán động học thuận - nghịch có những qui ước và phương pháp để xây dựng được ma trận thông số Denavit-Hartenberg chi tiết sẽ không được nêu ra, tại đây chỉ nêu ra các kết quả chọn thông số cũng như hệ trục để ra các ma trận biến đổi. Phương pháp giải có thể tham khảo ở nhiều nguồn lý thuyết khác nhau.

2.2.4.1. Bài toán động học thuận

- Hệ trục tọa độ trên từng khâu của cánh tay robot như sau:



Hình 2.15 Đặt hệ trục tọa độ từng khâu của robot.

- Từ đó ta có bộ thông số Denavit-Hartenberg:

Khâu	α_i	θ_i	d_i	a_i
------	------------	------------	-------	-------

1	-90°	θ_1^*	d_1	d_2
2	0	θ_2^*	0	d_3
3	0	θ_3^*	0	d_4
4	-90°	θ_4^*	0	0
5	0	θ_5^*	d_5	0

Bảng 2.1 Bảng thông số Denavit-Hartenberg.

- Các ma trận biến đổi qua từng khâu:

$$A_1 = \begin{bmatrix} c_1 & 0 & -s_1 & d_2 c_1 \\ s_1 & 0 & c_1 & d_2 s_1 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}; A_2 = \begin{bmatrix} c_2 & -s_2 & 0 & d_3 c_2 \\ s_2 & c_2 & 0 & d_3 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix};$$

$$A_3 = \begin{bmatrix} c_3 & -s_2 & 0 & d_4 c_3 \\ s_3 & c_2 & 0 & d_4 s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; A_4 = \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; A_5 = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix};$$

- Ma trận biến đổi thuần nhất từ điểm cuối so với hệ trục tọa độ gốc:

$$T_5^0 = A_1 A_2 A_3 A_4 A_5$$

$$= \begin{bmatrix} s_1 s_5 + c_1 c_{234} c_5 & -c_1 c_{234} s_5 + s_1 c_5 & -c_1 s_{234} & c_1 (d_3 c_2 + d_4 c_{23} - d_5 s_{234} + d_2) \\ s_1 c_{234} c_5 - c_1 s_5 & -c_1 c_5 - s_1 c_{234} s_5 & -s_1 s_{234} & s_1 (d_3 c_2 + d_4 c_{23} - d_5 s_{234} + d_2) \\ -c_5 s_{234} & s_5 s_{234} & -c_{234} & d_1 - d_5 c_{234} - d_3 s_2 - d_4 s_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.2.4.2. Bài toán động học ngược

- Tính động học ngược robot từ ma trận động học thuận robot ứng với ma trận tọa độ và hướng của điểm cuối:

$$T_{end} = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} s_1s_5 + c_1c_{234}c_5 & -c_1c_{234}s_5 + s_1c_5 & -c_1s_{234} & c_1(d_3c_2 + d_4c_{23} - d_5s_{234} + d_2) \\ s_1c_{234}c_5 - c_1s_5 & -c_1c_5 - s_1c_{234}s_5 & -s_1s_{234} & s_1(d_3c_2 + d_4c_{23} - d_5s_{234} + d_2) \\ -c_5s_{234} & s_5s_{234} & -c_{234} & d_1 - d_5c_{234} - d_3s_2 - d_4s_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Tính $\theta 1$:

$$\theta_1 = atan2(p_y, p_x)$$

- Tính $\theta 5$:

$$\begin{cases} s_5 = n_x s_1 - n_y c_1 \\ c_5 = s_x s_1 + s_y c_1 \end{cases}$$

$$\text{Nếu } c_1^2 + s_1^2 = 1: \theta_5 = atan2(s_5, c_5)$$

- Tính $\theta 3$:

$$s_{234} = -c_1a_x - s_1a_y; c_{234} = -a_z$$

$$K_1 = p_x c_1 + p_y s_1 - d_2 + d_5 s_{234}; K_2 = -p_z + d_1 - d_5 c_{234}$$

$$c_3 = \frac{K_1^2 + K_2^2 - d_4^2 - d_3^2}{2d_3d_4}$$

$$\text{Nếu } c_3 \in [-1; 1]: \theta_3 = \cos(c_3)$$

Lấy chỉ số trường hợp $\theta_3 > 0$ để lấy cây hình khuỷu tay elbow gấp lên (Elbow up configuration), đạt độ cân bằng cao hơn.

- Tính $\theta 2$:

$$\begin{cases} c_2 = \frac{K_1d_4c_3 + K_1d_3 + K_2d_4s_3}{d_4^2 + d_3^2 + 2d_3d_4c_3} \\ s_2 = \frac{K_2d_4c_3 + K_2d_3 - K_1d_4s_3}{d_4^2 + d_3^2 + 2d_3d_4c_3} \end{cases}$$

$$\text{Nếu } c_2^2 + s_2^2 = 1: \theta_2 = atan2(s_2, c_2)$$

- Tính θ_4 :

$$\begin{cases} c_4 = s_{23}s_{234} + c_{23}c_{234} \\ s_4 = c_{23}s_{234} - s_{23}c_{234} \end{cases}$$

Nếu $c_4^2 + s_4^2 = 1$: $\theta_4 = \text{atan2}(s_4, c_4)$

2.3. THIẾT KẾ BỘ ĐIỀU KHIỂN ROBOT:

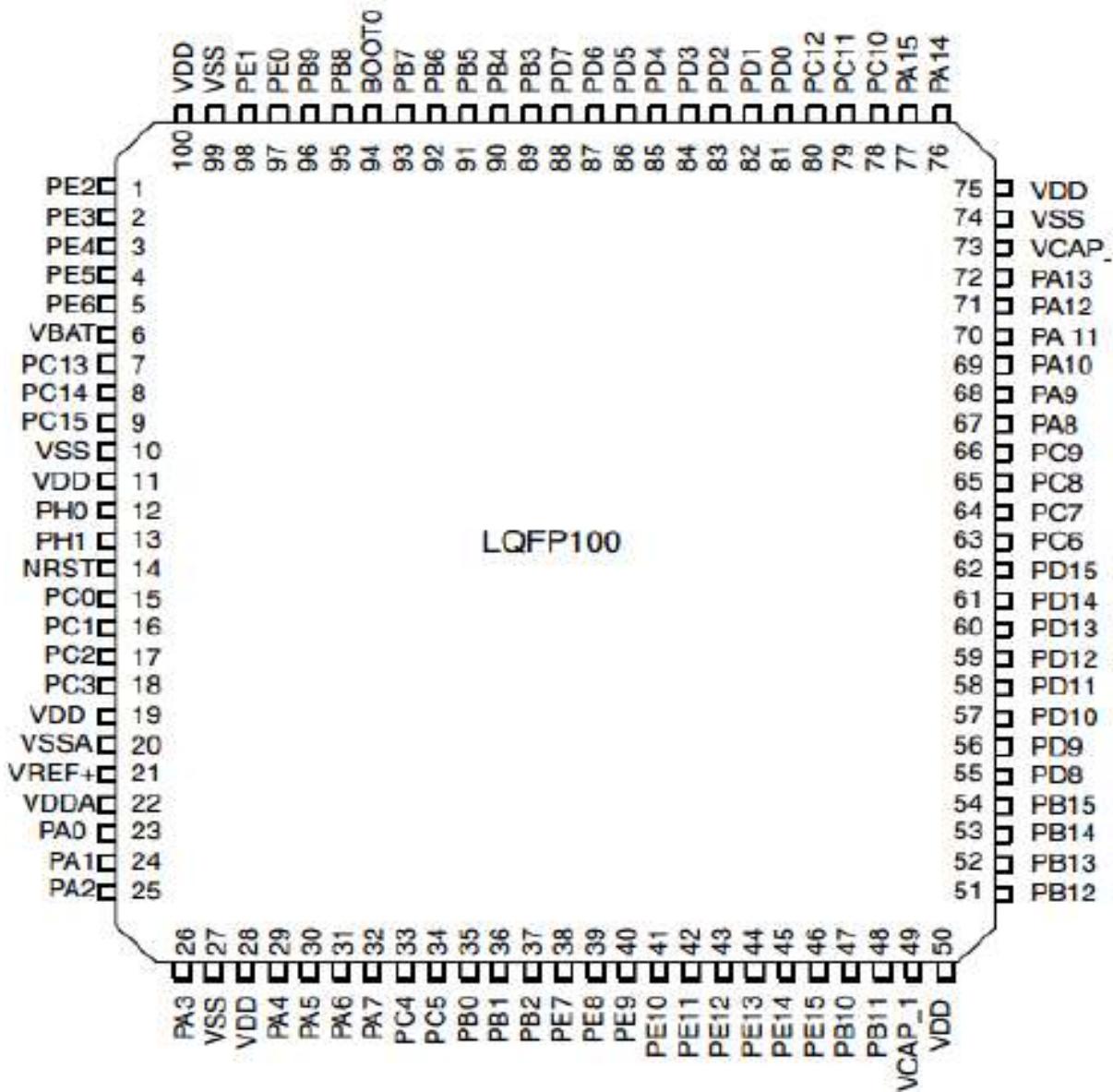
2.3.1. Giới thiệu sơ lược về kit STM32F407VGT Discovery:

Dòng Cortex-Mx được sản xuất cho các mục tiêu ứng dụng nhúng yêu cầu tốc độ vừa phải, nhiều nguồn ngắn và đáp ứng ngắn nhanh.

Các đặc điểm vượt trội của vi xử lý Cortex – M4:

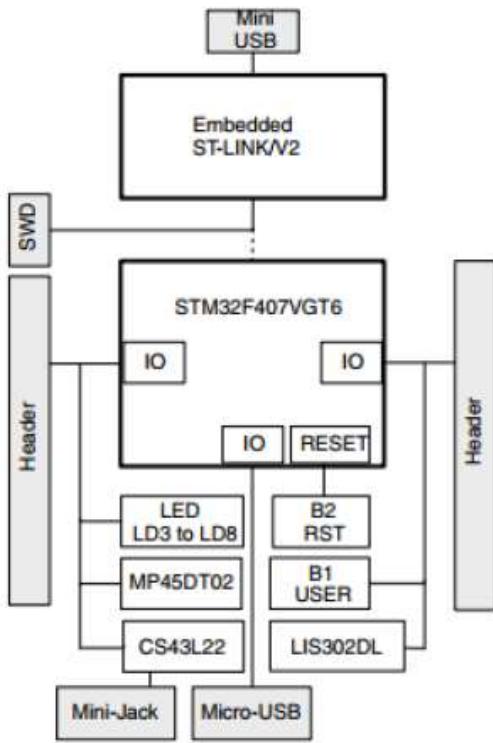
- o Bộ điều khiển ngắn lồng nhau NVIC (Nested Vector Interrupt Controller)
- o Phần tử xử lý số thực FPU (Floating Point Unit).

Vi điều khiển STM32F407 là vi điều khiển loại 32 bit của hãng ST Microelectronics với lõi vi xử lý ARM Cortex M4F với sơ đồ chân vi điều khiển STM32F407VGT:



Hình 2.16 Sơ đồ chân STM32F407VGT.

Board STM32F4 Discovery của ST dựa trên vi điều khiển STM32F407 có mạch nạp và debug trên board qua chuẩn SWD và nhiều ngoại vi để test các chức năng chính trên vi điều khiển.



Sơ đồ ngoại vi



Board STM32F4 DISCOVERY

Những module được sử dụng trong nội dung tiêu luận:

- **GPIO (GENERAL PURPOSE IN/OUT):** STM32F407 có 100 chân với 5 port PA, PB, PC, PD, PE chia nhau khoảng 80 chân GPIO.
- **NESTED VECTORED INTERRUPT CONTROLLER (NVIC):**

Đặc tính của ngắt STM32F4. Các NVIC và giao tiếp của lõi xử lý được kết hợp chặt chẽ, cho phép độ trễ gián đoạn xử lý thấp và xử lý hiệu quả các ngắt đến muộn.

Nhóm sử dụng ngắt để định chu kỳ đọc giá trị xung encoder; tính toán bộ điều khiển - xuất tín hiệu điều khiển điều rộng xung PWM. Ngoài ra còn sử dụng ngắt để truyền nhận thông tin từ máy tính.

- **TIMER:**

STM32F4 có tất cả 14 timer, 2 timer với mục đích nâng cao, 8 timer mở rộng, 2 timer cơ bản và 2 watchdog timers. Trong phạm vi tiêu luận, nhóm sử dụng 2 chức năng được tích hợp sẵn:

Điều rộn g xung PWM: Một timer khi được thiết lập có thể tạo ra 4 tín hiệu PWM với 4 duty cycle khác nhau.

Đọc xung của incremental encoder: đọc xung encoder từ 2 kênh A, B trên từng động cơ được kết nối với kit.

- **UASRT:**

Truyền thông nối tiếp giữa kit STM32F407VGT với các thiết bị ngoại vị khác, trong tiêu luận nhóm sẽ thực hiện kết nối với máy tính để đọc các tín hiệu giám sát và đưa các tín hiệu đặt đầu vào để điều khiển, cụ thể là tốc độ động cơ mong muốn.

2.3.2. Cấu hình chức năng cho kit điều khiển:

PIN	CHỨC NĂNG	ĐỐI TƯỢNG	Cấu hình AF	CHANNEL
PA15	ENC A	BASE	TIM2	CH1
PB3	ENC B	BASE	TIM2	CH2
PE6	PWM L	BASE	TIM9	CH2
PE5	PWM R	BASE	TIM9	CH1
PB6	ENC A	ELBOW	TIM4	CH1
PB7	ENC B	ELBOW	TIM4	CH2
PA7	PWM L	ELBOW	TIM14	CH1
PA6	PWM R	ELBOW	TIM13	CH1
PD10	DIR 1 GRAB	GRIPPER	GPIO_OUTPUT	—
PD8	DIR 2 DROP	GRIPPER	GPIO_OUTPUT	—
PB14	PWM	GRIPPER	TIM12	CH1
PA0	ENC A	PITCH	TIM5	CH1
PA1	ENC B	PITCH	TIM5	CH2
PC9	PWM L	PITCH	TIM8	CH4
PC8	PWM R	PITCH	TIM8	CH3
PA8	ENC A	ROLL	TIM1	CH1
PA9	ENC B	ROLL	TIM1	CH2

PC7	PWM L	ROLL	TIM8	CH2
PC6	PWM R	ROLL	TIM8	CH1
PB4	ENC A	SHOULDER	TIM3	CH1
PB5	ENC B	SHOULDER	TIM3	CH2
PB9	PWM R	SHOULDER	TIM11	CH1
PB8	PWM L	SHOULDER	TIM10	CH1
PD5	USART TX	Giao tiếp PC	USART2	
PD6	USART RX	Giao tiếp PC	USART2	
	GLOBAL_INTERRUPT	Ngắt lấy mẫu và tính toán theo chu kỳ 5 ms	TIM6	

Bảng 2.2 Bảng cấu hình chức năng AF.

Trong cấu hình chức năng của các chân tín hiệu, chân đọc Encoder của động cơ khớp PITCH là chân PA9 có tụ lọc C49 lọc tín hiệu cho chức năng khác. Tuy nhiên với chức năng đòi hỏi độ nhạy tín hiệu cao như đọc xung Encoder, hay truyền nhận giao tiếp UART, tụ lọc này làm giảm độ nhạy tín hiệu nên nhóm đã tháo tụ C49 ra khỏi kit STM32F4 để phục vụ chức năng đọc Encoder được sử dụng trong phạm vi luận văn.

2.3.3. Các khối chức năng công suất - cảm biến – nguồn – giao tiếp:

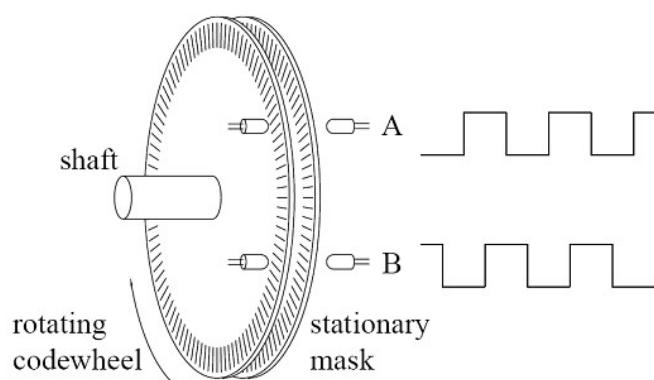
2.3.3.1. Khối cảm biến encoder và khối công suất:

- **Khối cảm biến:**

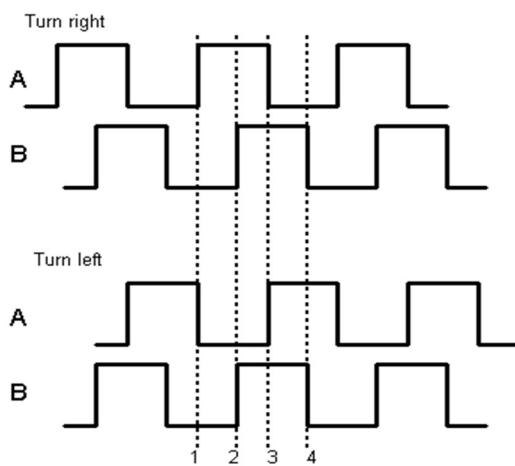
Các cảm biến trên các động cơ đều là loại Incremental Encoder được tích hợp trực tiếp trên động cơ, ngoại trừ động cơ SHOULDER cần được lắp Encoder thêm do chưa được tích hợp sẵn.

Sơ lược cấu tạo và nguyên lý hoạt động Incremental Encoder:

Incremental Encoder, hay còn được biết đến với cách gọi Encoder tương đối, là thiết bị cơ điện có khả năng chuyển đổi chuyển động tuyến tính (tròn hoặc thẳng) thành tín hiệu số hoặc xung,



từ đó có thể đưa tín hiệu vào các bộ điều khiển số để đo vị trí, góc lệch của trục quay đang được nối với thiết bị.



Hình 2.18 Xác định chiều thông qua xung Encoder.

Loại encoder tương đối phổ biến nhất hiện giờ là bộ mã hóa quang, cấu tạo gồm một đĩa quay, chất liệu phát quang và 2 thiết bị cảm biến quang để nhận tín hiệu 2 kênh A và B. Đĩa được gắn trên trục quay, có một phần trong suốt và một phần không trong suốt để tạo dãy mã hóa. Khi trục quay, thiết bị cảm quang sẽ lần lượt nhận và không nhận ánh sáng được xuyên qua phần trong suốt của đĩa, từ đó thành xung tín hiệu số. Với cấu tạo 2 bộ rãnh mã hóa trên đĩa đặt lệch nhau $\frac{1}{4}$ chu kỳ, ta dễ dàng xác định được cả vị trí và chiều quay của động cơ như hình bên.

Như vậy cấu tạo mạch Encoder tương đối cơ bản có 4 dây, 2 dây cấp nguồn thường là 5V và 2 dây tín hiệu kênh encoder A và B.

Trong luận văn, nhóm hoàn toàn sử dụng 5 Encoder tương đối với 4 Encoder được tích hợp sẵn trên động cơ và 1 Encoder lắp thêm cho trục khớp Shoulder. Xung được đọc lên vi điều khiển theo nguyên lý trên với cấu hình chức năng Encoder Interface có sẵn trên vi điều khiển.

Encoder tích hợp thêm với thông số 200PPR, nguồn 5VDC, đường kính trục 6mm, tốc độ 5.000.000 xung/phút.

- **Khối công suất:**

Để chuyên được tín hiệu điều khiển ra điện áp điều khiển chiều và tốc độ quay của động cơ DC, nhóm sử dụng phương pháp cơ bản và phổ biến nhất là phương pháp điều rộng xung với mạch cầu H.



Hình 2.19 Incremental Encoder 200PPR.

Mạch cầu H là mạch biến đổi điện áp một chiều. Đầu vào của mạch gồm điện áp DC dùng cung cấp cho động cơ và tín hiệu điều khiển hướng và độ rộng xung. Đầu ra của mạch là điện áp DC đã được thay đổi tương ứng với tín hiệu độ rộng xung đầu vào. Chi tiết lí thuyết hoạt động mạch cầu H có thể được tham khảo bằng nhiều nguồn, nhóm không phân tích chi tiết.

Trong phạm vi luận văn, nhóm sử dụng 2 loại module cầu H khác nhau, 1 loại công suất nhỏ dùng IC LM298 để điều khiển đầu gắp gripper và cấp nguồn 5V cho kit vi điều khiển, và cầu H Arduino IBT-2 dùng cho 5 động cơ khớp quay robot đòi hỏi công suất cao hơn và chịu dòng tốt.



Thông số kỹ thuật mạch cầu H IBT-2 BTS7960:

Nguồn điện áp đầu vào: 6 – 27V. Dòng điện tải mạch: 43A với tải trở hoặc 15A với tải cảm. Tín hiệu logic điều khiển 3.3V - 5V.

Tần số điều khiển tối đa 25KHz.

Chế độ tự động tắt khi điện áp thấp để tránh trường hợp động cơ có điện áp thấp nhưng không chuyển động gây dòng cao. Bảo vệ chống quá nhiệt thông qua cảm biến tích hợp.

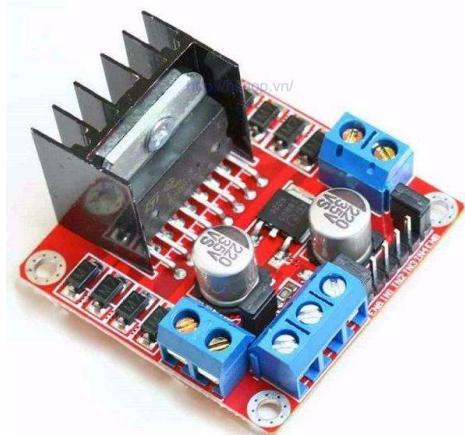
Hình 2.20 Cầu H Arduino IBT-2
BTS7960 43A

Sơ đồ chân logic điều khiển:

VCC	GND	2 dây cấp nguồn logic cho cầu H; VCC từ 5V – 3.3V; chân GND nối chung GND với VĐK.
R_IS	L_IS	Kết hợp điện trở để giới hạn dòng qua cầu H, chức năng này nhóm không sử dụng.
R_EN	L_EN	R_EN và L_EN là 2 chân enable/disable nửa cầu H trái/phải. Với mức logic 1 là Enable và 0 là Disable. Để điều khiển động cơ nhóm nối 2 chân này với mức logic 1.

RPWM	LPWM	<p>Chân cấp xung PWM cho chiều quay động cơ.</p> <p>$RPWM > 0$ và $LPWM = 0$ thì động cơ quay phải,</p> <p>$RPWM = 0$ và $LPWM > 0$ thì động cơ quay trái.</p> <p>Nhóm nối 2 chân với 2 kênh PWM phát xung độc lập.</p>
------	------	---

Thông số kỹ thuật mạch cầu H dùng IC L298:



Hình 2.21 Cầu H L298

IC chính: L298 – Dual Full Bridge Driver.

Điện áp đầu vào: 5 – 30VDC.

Công suất tối đa: 25W

Dòng tối đa cho mỗi cầu H là 2A.

Mức điện áp logic: LOW = 0.3 ~ 1.5V;
HIGH = 2.3V ~ Vss.

IC nguồn tích hợp 7805 giúp cấp nguồn 5VDC cho các module khác.

Sơ đồ chân:

VCC, GND	2 chân cấp nguồn đầu vào cho động cơ, đồng thời qua IC 7805 cấp nguồn cho chính cầu H và một ngõ ra 5VDC cấp nguồn cho module khác.
5V, GND	Nguồn đầu ra 5V cho module khác.
ENA, ENB	Enable/ Disable cầu H A hay B. Nối chân phát xung PWM vào chân này để điều khiển điện áp ngõ ra theo phương pháp điều rộng xung PWM.
IN1, IN2	Ngõ vào logic điều khiển chiều quay của động cơ cho cầu H A. IN1 = 1 và IN2 = 0: Quay chiều thuận; IN1 = 0 và IN2 = 1: Quay chiều nghịch;

IN3, IN4	Ngõ vào logic điều khiển chiều quay của động cơ cho cầu H B. Tương tự IN3 và IN4.
----------	--

2.3.3.2. Khối cấp nguồn:

Khối nguồn sử dụng:

- 1 bộ nguồn 24V – 20A tản nhiệt có quạt để cấp nguồn cho 2 động cơ Elbow và Base:



Hình 2.22 Bộ nguồn 24VDC, 20A, 480W.

- 1 bộ nguồn 12V – 15A có quạt để cấp nguồn cho động cơ Shoulder chịu tải nặng nhất và 2 động cơ PITCH, ROLL:



Hình 2.23 Bộ nguồn 12VDC 15A.

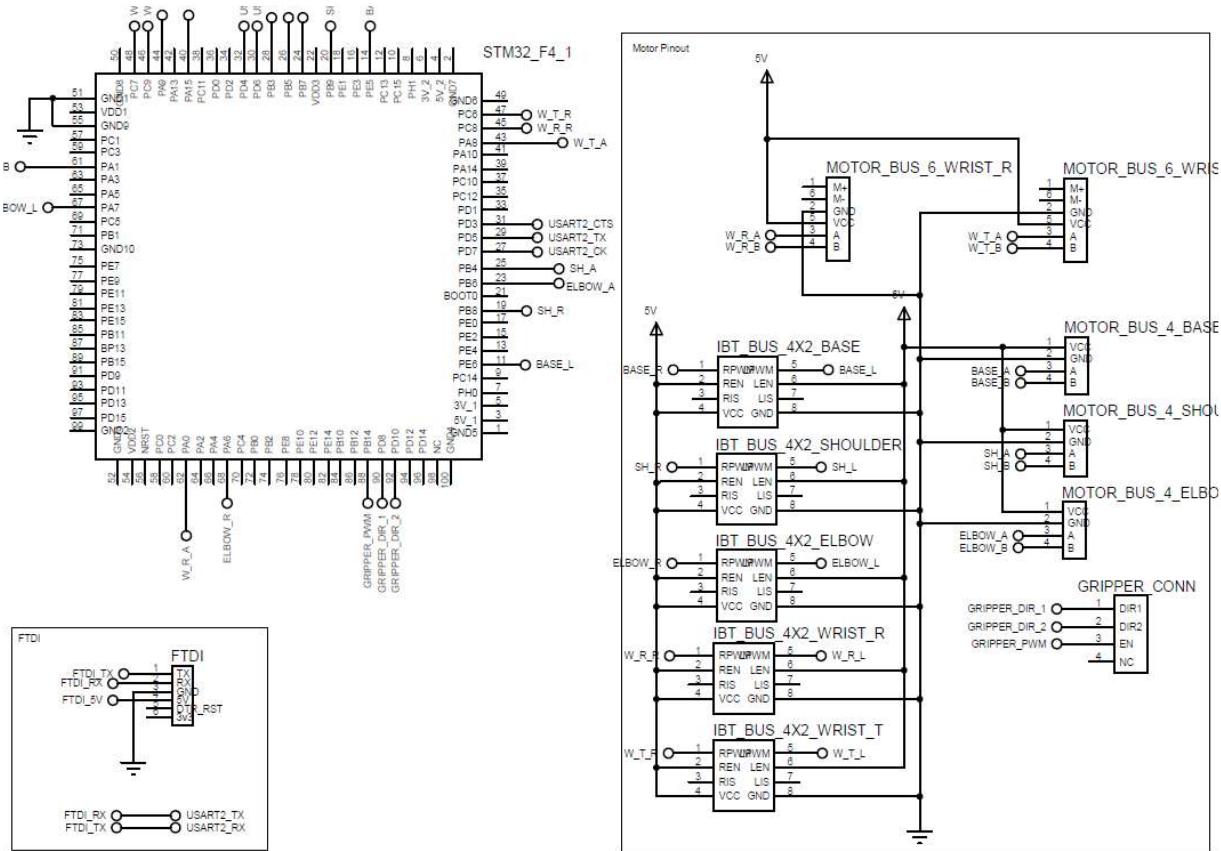
- 1 bộ nguồn 12V 5A để cấp nguồn cho điều khiển đầu gắp gripper và xuất nguồn 5V cho vi điều khiển và toàn bộ 5 Encoder trên từng khớp.



Hình 2.24 Bộ nguồn 12VDC, 5A.

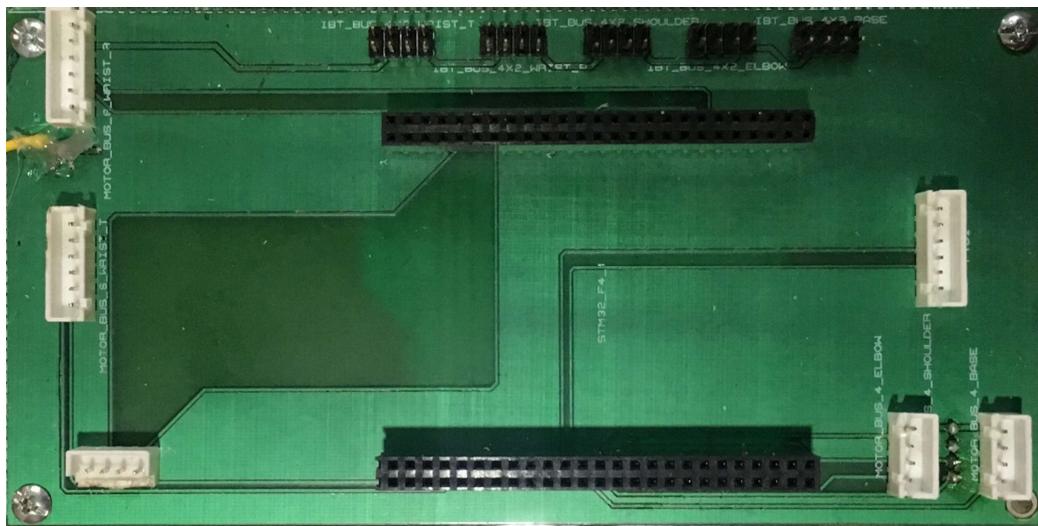
2.3.3.3. Board adapter giao tiếp giữa các khối:

- Board giao tiếp gồm:
 - 3 HEADER 4-PIN đọc ENCODER cho BASE, SHOULDER, ELBOW.
 - 2 HEADER 6-PIN nguồn áp và ENCODER cho động cơ PITCH và ROLL.
 - 1 HEADER 3-PIN tín hiệu điều khiển cho GRIPPER.
 - 5 HEADER 4x2-PIN tín hiệu điều khiển cho 5 khớp xoay.
 - 1 HEADER 6-PIN cho dây nối USART.
 - 2 dây cáp nguồn cho ENCODER và Vi điều khiển.
- Schematic:



Hình 2.25 Bản vẽ schematic sơ đồ chân nối board điều khiển.

- Hình mạch thực tế:



Hình 2.26 Hình mạch thực tế hoàn chỉnh.

2.4. GIẢI THUẬT ĐIỀU KHIỂN VỊ TRÍ ĐỘNG CƠ:

2.4.1. Giải thuật STR MRAS cải tiến:

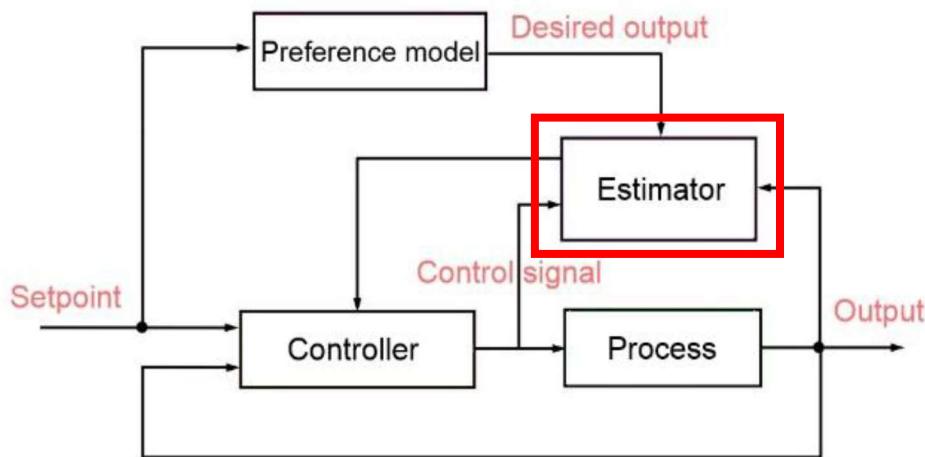
Giải thuật STR MRAS cải tiến [1] điều khiển vị trí của các khớp động cơ được nhóm sử dụng tham khảo từ một nguồn mở từ một đề tài luận văn của cựu sinh viên trường ĐH Bách Khoa TP HCM, chia sẻ trên trang web www.payitforward.com. Nhóm tham khảo và sử dụng bộ điều khiển ứng dụng trong bộ điều khiển robot.

- Bộ ước lượng bình phương tối thiểu đê qui:

Ý nghĩa của giải thuật này là xác định được vị trí ước lượng của động cơ $\hat{y}(k)$ thông qua các thông số thu về từ cảm biến và vị trí đặt trong ma trận $\varphi^T(k)$ và thông số ước lượng mô hình rời rạc trong ma trận $\hat{\theta}(k)$.

$$\hat{y} = \varphi^T(k)\hat{\theta}(k)$$

$$\left\{ \begin{array}{l} y(k) + a_1y(k-1) + a_2y(k-2) + a_3y(k-3) = b_1u(k-1) + b_2u(k-2) + b_3u(k-3) \\ \theta(k) = [a_1(k) \ a_2(k) \ a_3(k) \ b_1(k) \ b_2(k) \ b_3(k)]^T \\ \varphi(k) = [-y(k-1), -y(k-2), -y(k-3), u(k-1), u(k-2), u(k-3)]^T \end{array} \right.$$



Hình 2.27 Bộ ước lượng bình phương tối thiểu đê qui.

Giải thuật ước lượng trong trường hợp điều khiển vị trí:

Hàm truyền đổi tượng:

$$G(z) = \frac{b_1 z^2 + b_2 z + b_3}{z^3 + a_1 z^2 + a_2 z + a_3}$$

Bước 1: Khởi tạo thông số ma trận khởi đầu $\theta(0)$ và ma trận covariance $P(0)$.

$$P(0) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}; \quad \theta(0) = \text{random}(6,1);$$

Bước 2: Cập nhật ma trận trạng thái hệ thống:

$$\varphi(k) = [-y(k-1), -y(k-2), -y(k-3), u(k-1), u(k-2), u(k-3)]^T$$

Bước 3: Tính toán ma trận gain Kalmann $L(k)$ mới theo công thức:

$$L(k) = \frac{P(k-1)\varphi(k)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)}$$

Bước 4: Tính toán ma trận sai số ước lượng:

$$\epsilon(k) = y(k) - \varphi^T(k)\hat{\theta}(k-1)$$

Bước 5: Tính toán ma trận trạng thái hệ thống $\varphi(k)$ mới và lấy các hệ số ước lượng trạng thái động cơ:

$$\hat{\theta}(k) = \hat{\theta}(k-1) + L(k)\epsilon(k)$$

$$\hat{\theta}(k) = [\hat{a}_1(k) \hat{a}_2(k) \hat{a}_3(k) \hat{b}_1(k) \hat{b}_2(k) \hat{b}_3(k)]^T$$

Bước 6: Tính toán ma trận covariance P mới theo công thức:

$$P(k) = \frac{1}{\lambda} [P(k-1) - L(k)\varphi(k)P(k-1)]$$

Bước 7: Quay về bước 2, qua vòng lặp lấy mẫu và điều khiển tiếp theo.

- Thiết kế theo mô hình chuẩn:

Chuyển hàm truyền đổi tượng miền tần số sang miền rời rạc:

$$\frac{Y_m(s)}{U_c(s)} = \frac{10\omega_n^2}{(s - 10\omega_n)(s^2 + 2\xi\omega_n + \omega_n^2)}$$

$$\Rightarrow \frac{Y_m(z)}{U_c(z)} = \frac{b_{1m}z^2 + b_{2m}z + b_{3m}}{z^3 + a_{1m}z^2 + a_{2m}z + a_{3m}}$$

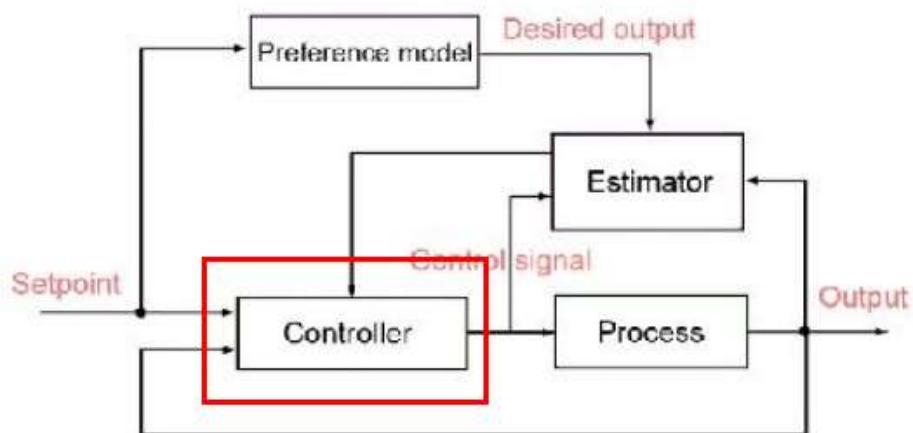
Từ mô hình trên, ta chọn mô hình chuẩn từ miền tần số liên tục rồi sử dụng Matlab để rời rạc hóa mô hình chọn làm mô hình chuẩn.

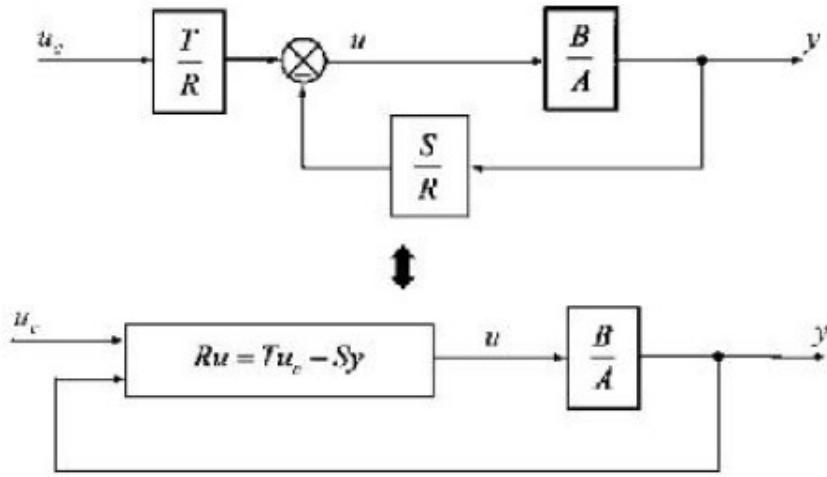
Nhóm chọn mô hình chuẩn với thông số $\omega = 400 \frac{rad}{s}$, $\delta = 0.826085$ để thỏa POT = 0%, $t_{qđ} < 10ms$ và thực hiện chương trình Matlab STR_MRAS_Position_Ref_Model.m đính kèm để rời rạc hóa mô hình chuẩn, chu kỳ 5 ms:

$$\frac{Y_m(s)}{U_c(s)} = \frac{5.12 * 10^9}{s^3 + 9322s^2 + 1.121 * 10^7s + 5.12 * 10^9}$$

$$\Rightarrow \frac{Y_m(z)}{U_c(z)} = \frac{0.9757z^2 + 0.7203z + 1.596 * 10^{-5}}{z^3 + 0.04638z^2 + 0.001349z - 1.387 * 10^{-2}}$$

Bộ điều khiển theo mô hình chuẩn:





Hình 2.28 Bộ điều khiển theo mô hình chuẩn.

Giải thuật điều khiển vị trí động cơ theo mô hình chuẩn:

Đối tượng:

$$Y(z) = \frac{B}{A} U(z) = \frac{b_1 z^2 + b_2 z + b_3}{z^3 + a_1 z^2 + a_2 z + a_3} U(z)$$

Mô hình chuẩn:

$$Y_m(z) = \frac{B}{A} U_c(z) = \frac{b_{1m} z^2 + b_{2m} z + b_{3m}}{z^3 + a_{1m} z^2 + a_{2m} z + a_{3m}} U_c(z)$$

Luật điều khiển sau khi thiết kế:

$$Ru = Tu_c - Sy \text{ trong đó } \begin{cases} S = (a_{1m} - a_1)z^2 + (a_{2m} - a_2)z + (a_{3m} - a_3) \\ R = b_1 z^2 + b_2 z + b_3 \\ T = b_{1m} z^2 + b_{2m} z + b_{3m} \end{cases}$$

$$\begin{aligned} \Rightarrow u &= - \left(\frac{b_2}{b_1} z^{-1} + \frac{b_3}{b_1} z^{-2} \right) u \\ &\quad + \left(\frac{b_{1m}}{b_1} + \frac{b_{2m}}{b_1} z^{-1} + \frac{b_{3m}}{b_1} z^{-2} \right) u_c \\ &\quad - \left[\left(\frac{a_{1m}}{b_1} - \frac{a_1}{b_1} \right) + \left(\frac{a_{2m}}{b_1} - \frac{a_2}{b_1} \right) z^{-1} + \left(\frac{a_{3m}}{b_1} - \frac{a_3}{b_1} \right) z^{-2} \right] y \end{aligned}$$

- Những cải tiến bộ ước lượng thông số được tác giả áp dụng bổ sung:
- Hệ số quên thay đổi theo thông số của hệ thống:

Hệ số quên tỉ lệ thuận với tính thích nghi. Hệ số quên càng lớn thì tính thích nghi càng cao, thích nghi theo càng nhiều trạng thái trước đó của động cơ. Tuy nhiên trạng thái hoạt động của động cơ mỗi lúc cần tính thích nghi nhanh chậm khác nhau. Đồng thời nếu $\lambda < 1$, khi trạng thái thông số hệ thống không thay đổi qua nhiều, ma trận P có xu hướng ít thay đổi sau mỗi chu kì lấy mẫu, ma trận P luôn được chia cho $\lambda < 1$ sẽ dẫn đến một lúc nào đó giá trị của P đạt đến vô cùng và hệ thống mất ổn định. Vì vậy cần có thuật toán thay đổi hệ số quên phù hợp.

Thuật toán cụ thể được đính kèm theo luận văn tham khảo không nêu trực tiếp tại báo cáo này.

- Giải thuật ước lượng có tính độ trễ động cơ: Trong thực tế, khi điều khiển động cơ ta phải tính đến cả độ trễ có động cơ, tác nhân này có tác động không nhỏ đến chất lượng điều khiển. Độ trễ này có nguyên nhân bởi phần cảm và phần ứng của động cơ, của thiết bị công suất cầu H, sự truyền nhận tín hiệu điều khiển từ động cơ đến vi điều khiển. Đồng thời nếu điều khiển vị trí không tính đến độ trễ mà vị trí qui hoạch thay đổi không đủ nhanh, động cơ sẽ bị khựng lại liên tục do đã đến vị trí trên từng thời điểm, gây rung lắc cho động cơ. Giải thuật ước lượng có tính độ trễ động cơ được tham khảo thực hiện theo tài liệu tham khảo, nhóm chọn ước lượng trễ 1 chu kỳ điều hiển để đảm bảo chất lượng điều khiển được tốt nhất. Cụ thể giải thuật được theo tài liệu tham khảo đính kèm.
- Giải thuật nhân chuỗi ma trận với số phép toán tối ưu: phương pháp cụ thể được đính kèm theo tài liệu tham khảo của luận văn, mục đích là chọn thứ tự nhân ma trận tối ưu để giảm tối đa số phép tính phải tính, từ đó tối ưu hiệu năng của thuật toán.
- Ngoài ra với yêu cầu điều khiển các khớp robot không được rung lắc (di chuyển qua lại liên tục quanh điểm đặt), nhóm có thực hiện áp dụng giải thuật

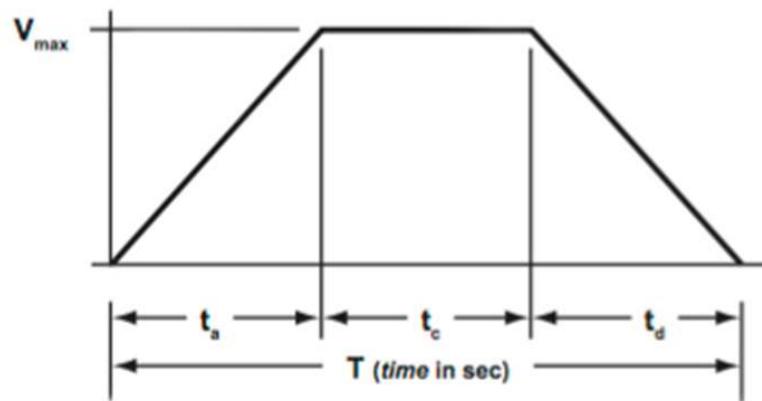
giảm tín hiệu điều khiển theo tỉ số lựa chọn khi sai số bé hơn một mức cho phép nhất định.

2.4.2. Giải thuật qui hoạch quỹ đạo vị trí:

Khi điều khiển động cơ DC ở các khớp cánh tay robot, động cơ sẽ có hiện tượng rung lắc hoặc chuyển động giật cục nếu có sự thay đổi trạng thái đột ngột từ đứng yên đến chuyển động vận tốc lớn hoặc ngược lại. Hiện tượng này làm quá trình chuyển động không đạt được chất lượng như mong muốn cũng như sẽ gây hại cho động cơ cũng như các phần tử chấp hành cơ khí.

Vì vậy nhóm thực hiện chọn giải thuật điều tốc hình thang Trapezoid Method để quá trình chuyển trạng thái chuyển động được mượt mà và hạn chế tối đa hiện tượng rung giật như đã nêu trên.

Với giải thuật này, vận tốc sẽ tăng từ từ đến trạng thái mong muốn cũng như giảm từ từ và dừng hẳn khi đến đích.



Ta có các biến t_0 là thời gian đạt được vị trí mong muốn, tốc độ tối đa là v_{max} , v_{01} là vận tốc góc hiện tại được đọc trực tiếp từ cảm biến.

Theo sơ đồ trên, nhóm thực hiện chọn tỉ số trên từng khoảng:

$$t_a = \frac{1}{5}t_0, t_c = \frac{3}{5}t_0, t_d = \frac{1}{5}t_0$$

Ứng với khoảng thời gian trên ta có vị trí qui hoạch trong từng khoảng như sau:

$$0 < t < \frac{t_0}{5}: x = x_0 + v_{01}t + \frac{5}{2}(v_{max} - v_{01}) * \frac{t^2}{t_0}$$

$$\frac{3t_0}{5} < t < \frac{4t_0}{5}: x = x_{01} + v_{max}(t - \frac{t_0}{5})$$

$$\frac{4t_0}{3} < t < t_0: x = x_{02} + v_{max} \left(t - \frac{4t_0}{5} \right) - \frac{1}{2} \left(\frac{5v_{max}}{t_0} \right) \left(t - \frac{4t_0}{5} \right)^2$$

Trong đó:

$$x_{01} = x_0 + v_{01} \left(\frac{t_0}{5} \right) + \frac{1}{2} (v_{max} - v_{01}) \left(\frac{t_0}{5} \right); x_{02} = x_{01} + v_{max} \left(\frac{3t_0}{5} \right)$$

Như vậy khi ta xác định được vị trí cần đưa đến x_f từ góc hiện tại x_0 với vận tốc góc tức thời là v_{01} , thời gian đạt được vị trí mong muốn là t_0 , ta tính được trên từng khốp:

$$v_{max} = \frac{x_f - x_0}{\frac{4}{5}t_0 - \frac{v_{01}}{8}}$$

Đồng thời các giá trị v_{max_i} thỏa $v_{max} < v_{limit}$, trong đó v_{limit} là vận tốc giới hạn cơ khí của từng khốp.

CHƯƠNG 3: GIẢI THUẬT XỬ LÍ ẢNH XÁC ĐỊNH VỊ TRÍ VÀ NHẬN DẠNG BÁM THEO VẬT THỂ:

3.1. TỔNG QUAN KHỐI XỬ LÍ ẢNH

3.1.1. Mục đích khối xử lý ảnh:

Sau khi đã xây dựng hoàn chỉnh mô hình thực tế cánh tay robot và bộ điều khiển, để điều khiển được robot ta phải chọn được vị trí đầu gắp cần đến, tức phải có được có tọa độ và hướng tiếp cận điểm đến này. Có nhiều cách khác nhau để thiết lập những điểm này, nhưng trong công nghiệp hoặc các hệ thống thường gặp, người ta thường làm theo những cách sau:

- Dựa theo qui trình sản xuất, kỹ sư điều khiển và kỹ sư qui trình sẽ xây dựng ra quy trình làm việc của robot, từ đó chọn ra những điểm cần đến theo thứ tự định trước, từ đó robot vận hành hoàn toàn theo quy trình sẵn (thông qua G-Code).
- Sử dụng tích hợp cảm biến khoảng cách trên băng chuyền để xác định tọa độ theo 1 phương, robot vẫn hoạt động theo quy trình có sẵn nhưng linh động hơn trong việc xác định tọa độ điểm cần gắp (chính xác theo 1 phương).
- Sử dụng camera xác định tọa độ vật trên một mặt phẳng cố định, thông thường chọn cố định độ cao mặt phẳng (z), xây dựng tọa độ theo cặp (x,y).

Với những ứng dụng trong công nghiệp cần độ chính xác cao trên một dây chuyền đã được thiết kế sẵn, những phương pháp từ đơn giản đến phức tạp này đều đã đáp ứng được yêu cầu.

Tuy nhiên để cánh tay được linh hoạt hơn và có thể nhận dạng cũng như tương tác với vật trong môi trường thực tế luôn thay đổi, nhóm xây dựng khối xử lý ảnh với mục tiêu cung cấp cho cánh tay robot vị trí của vật trong không gian, từ đó có thể tương tác với vật và làm các chức năng cần thiết theo yêu cầu của người sử dụng. Nhóm đặt mục tiêu thuật toán có thể xử lý theo thời gian thực, tạo tín hiệu hồi tiếp liên tục cho bộ điều khiển robot để tăng độ phản hồi của robot với môi trường thay đổi.

Với mục tiêu này, robot sẽ thông minh hơn và có thể được lập trình thực hiện những công việc thường nhật hơn với cuộc sống, như phục vụ bàn (gắp thả li nước), cánh tay robot gắn lên mobile robot nhặt rác tự động vệ sinh môi trường,...

3.1.2. Phương pháp tiếp cận:

Với mục đích và mục tiêu của khối xử lý ảnh được đặt ra như trên, nhóm đã nghiên cứu và chọn sử dụng camera Kinect là thiết bị chính để thu nhận hình ảnh từ môi trường và tích hợp thư viện Point Cloud Library (PCL) mà nguồn mở để xử lý hình ảnh.

- Kinect là thiết bị camera tích hợp 2 camera bên trong và đã được hiệu chỉnh chuẩn sử dụng trong môi trường trong nhà, có khả năng thu nhận ảnh màu và xác định khoảng vật trong không gian được hỗ trợ, phù hợp với mục tiêu xác định tọa độ vật 3D trong không gian.

- PCL là thư viện xử lý ảnh cùng mã nguồn mở, được xây dựng để chuyên sử dụng để xử lý các dữ liệu dạng Point Cloud thu được từ các loại camera đo chiều sâu không gian nói chung, đặc biệt là từ thiết bị Kinect. Hiện nay thư viện vẫn được tiếp tục bảo trì và phát triển, phù hợp với việc nghiên cứu ứng dụng và phát triển mở rộng sau này, không bị lỗi thời công nghệ.

3.1.3. Giới thiệu sơ lược về Kinect và thư viện Point Cloud Library – PCL:

3.1.3.1. Microsoft Kinect:

Giới thiệu tổng quan:

Tại triển lãm E3 2009, Kinect lần đầu được Microsoft giới thiệu dưới tên mã là Project Natal (Natal là tên một thành phố tại Brazil, nơi sinh ra của 1 giám đốc dự án này). Và sau đó vào tháng 11 năm 2010, Kinect đã được giới thiệu ra thị trường như một sản phẩm phụ kiện hoàn thiện hỗ trợ cho Xbox 360. Cái tên Kinect là sự kết hợp của 2 từ “Kinetic” (chuyển động) và “Connect” (kết nối).

Kinect là một thiết bị đầu vào, là cảm biến chuyển động của Microsoft sản xuất dành cho thiết bị giải trí Xbox 360 và máy tính cá nhân sử dụng hệ điều hành Windows. Thiết bị là một thiết bị camera add-on ngoại vi mở rộng được thiết kế ban đầu cho Xbox 360, cho phép và hỗ trợ người dùng điều khiển và tương tác với Xbox 360 không thông qua bộ điều khiển tay cầm truyền thống, mà

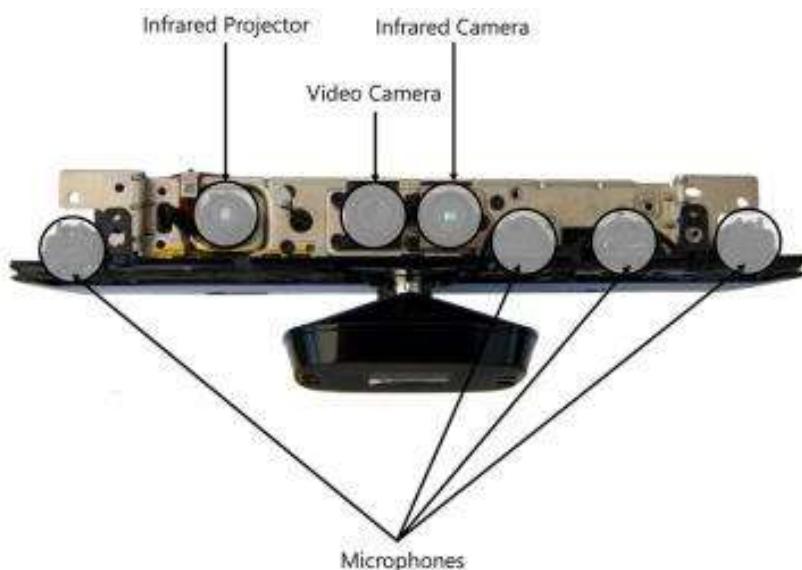


Hình 3.1 Thiết bị Kinect.

thông qua giao diện người dùng là cử chỉ và giọng nói, thân thiện và linh hoạt hơn với người dùng, nâng cao trải nghiệm người dùng trong lĩnh vực giải trí.

Đặc tính và cấu tạo kĩ thuật:

Thiết bị cảm biến Kinect là một thanh ngang kết nối với một trụ nhỏ thông qua trục cơ đứng. Kinect cấu tạo gồm: RGB camera, cảm biến độ sâu, dãy microphone và động cơ điều khiển góc ngẩng. Ở phạm vi luận văn chỉ xử lí hình ảnh nên ta chỉ tìm hiểu sâu hơn hệ thống cảm biến và camera.

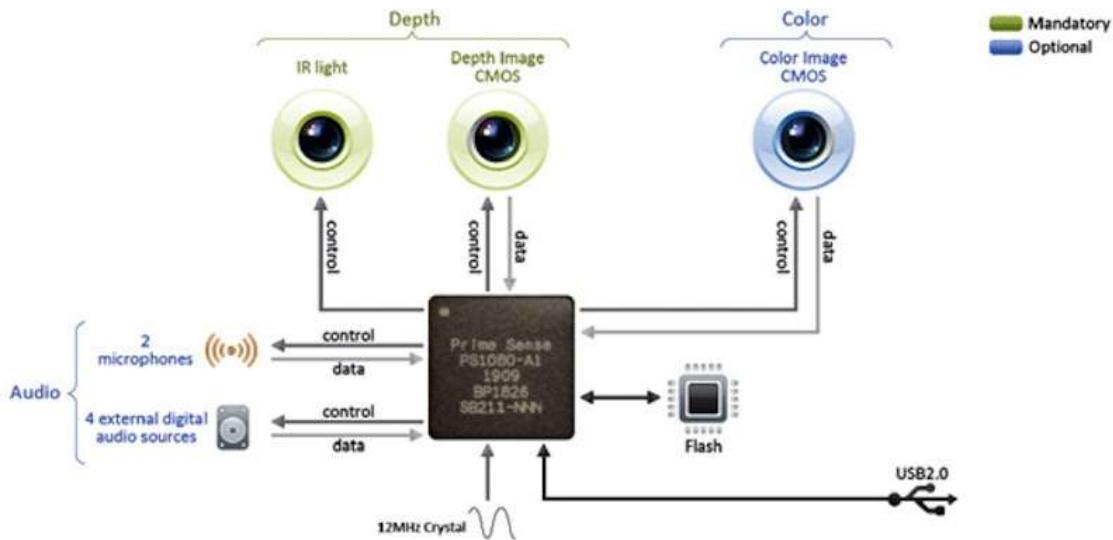


Hình 3.2 Cấu tạo bên trong của Kinect.

RGB camera như một camera thông thường với độ phân giải 640x480 cùng tối đa 30fps.

Cảm biến của Kinect bao gồm một bộ phát hồng ngoại (Infrared projector) cùng một camera hồng ngoại (depth image CMOS) với cỡ ảnh 320x240 với độ sâu 64536 mỗi pixel và tốc độ chụp 30fps, giúp Kinect hoạt động được dưới bất kì nguồn sáng đơn sắc nào. Tầm hoạt động tốt nhất của cảm biến là trong khoảng 1,2m đến 3,5m. Cặp cảm biến này sẽ phối hợp với nhau để tính ra giá trị độ sâu ảnh bằng công nghệ Light Coding và PrimeSense.

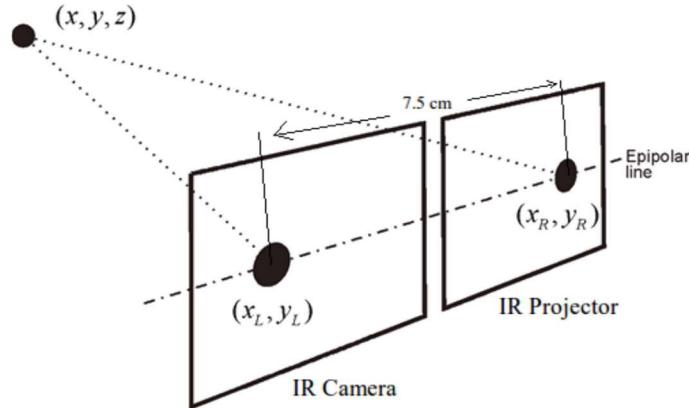
Tầm quan sát của Kinect là 57° theo trục ngang và 43° theo trục dọc.



Hình 3.3 Cấu trúc khối xử lí và truyền nhận Kinect.

Nguyên lý hoạt động:

Các tia hồng ngoại được chiếu qua bộ phát hồng ngoại đến đối tượng, sau đó camera hồng ngoại sẽ thu thập dữ liệu bị phản chiếu. 2. Camera RGB-D Chức năng chính của camera là nhận biết 3 màu cơ bản là đỏ, xanh lá cây và xanh da trời (Red-Green-Blue). Quá trình chụp bao gồm việc chụp một ảnh màu (RGB) và thực hiện một phép đo độ sâu (D). Cảm biến hình ảnh kết hợp với cảm biến chiều sâu nằm ở gần nhau, cho phép sáp nhập bản đồ, cho ra hình ảnh 3D. Thông tin ảnh RGB-D được lưu trữ. Với dữ liệu sâu thu được, nó sẽ tạo ra một bản đồ về cả màu sắc và chiều sâu của các vật thể và không gian.



Hình 3.4 Phương pháp đo giá trị độ sâu Kinect.

Cơ chế hình thành ảnh độ sâu của Kinect là nó sử dụng cặp gồm camera hồng ngoại (IR camera) bộ phát ánh sáng hồng ngoại (IR Projector) phát ra ánh sáng có cấu trúc (structured light) để tạo ra giá trị độ sâu bằng công nghệ Light Coding của

PrimeSense. Kỹ thuật Light Coding dùng nguồn sáng hồng ngoại của bộ phát ánh sáng hồng ngoại chiếu liên tục vào môi trường xung quanh kết hợp với việc sử dụng máy ảnh hồng ngoại chụp lại, sau đó tính toán để thu được ảnh độ sâu. Bằng việc so sánh giữa mẫu quan sát được và mẫu tham khảo biết trước, Kinect dự đoán về khoảng cách từ Kinect đến đối tượng. Kinect có thể cung cấp ảnh độ sâu trong điều kiện ánh sáng rất tối và ít chịu ảnh hưởng của chất lượng bề mặt vật thể. Hơn nữa, thiết bị Kinect có giá thành rẻ, nhỏ gọn và rất dễ sử dụng, độ phân giải và tốc độ thu nhận ảnh chấp nhận được (640 x 480 điểm ảnh). Công thức và cách xác định độ sâu cụ thể phức tạp không đề cập cụ thể tại đây.

3.1.3.2. Point Cloud Library:



Hình 3.5 Logo Point Cloud Library.

PCL là thư viện hỗ trợ cho n-D Point Cloud và cho việc xử lý ảnh trong không gian 3D. Thư viện được xây dựng với nhiều giải thuật như lọc (filtering), khôi phục bề mặt (surface reconstruction), phân vùng (segmentation), ước lượng đặc tính vật (feature estimation),... PCL có thể dùng trên nhiều platform như Linux, MacOS, Windows và Android. Để đơn giản cho việc phát triển, PCL được chia ra thành nhiều thư viện nhỏ và có thể biên dịch một cách riêng lẻ. Phiên bản mới nhất là PCL 1.3 đưa ra vào ngày 31 tháng 10 năm 2011. PCL hoàn toàn miễn phí cho việc nghiên cứu hay phát triển các sản phẩm thương mại hóa.

Có thể nói PCL là sự kết hợp của nhiều module nhỏ. Những module này thực chất cũng là các thư viện thực hiện các chức năng riêng lẻ trước khi được PCL đóng gói. Các thư viện cơ bản này là:

- Eigen: Hỗ trợ các phép toán tuyến tính, dùng vào hầu hết các tính toán toán học của PCL.
- FLANN: (Fast Library for Approximate Nearest Neighbors) Tìm kiếm nhanh các điểm lân cận trong không gian 3D.
- Boost: Giúp chia sẻ con trỏ trên tất cả các module và thuật toán trong PCL để tránh sao chép và trùng dữ liệu đã lấy về trong hệ thống.

- VTK: (Visualization Toolkit) Hỗ trợ nhiều platform trong việc thu về dữ liệu 3D, hỗ trợ hiển thị, ước lượng thể tích vật thể.
- CminPack: Thư viện mở giúp giải quyết phép toán tuyến tính và không tuyến tính.

Thư viện có tất cả 14 modules:

PCL_Common	Chứa cấu trúc dữ liệu và phương thức được sử dụng bởi phần lớn các thư viện trong PCL Cấu trúc dữ liệu cốt lõi là các class pointCloud, các loại dữ liệu biểu diễn điểm, bề mặt, giá trị màu, mô tả tính năng... Ví dụ: PCL::PointXYZ; PCL::PointXY; PCL::PointXYZRGB;
Module Features	Chứa các cấu trúc dữ liệu và cơ chế tính toán, ước lượng 3D từ các dữ liệu điểm PCD. 3D Features biểu diễn chính xác điểm 3D hoặc vị trí trong không gian để mô tả phần hình khối dựa vào thông tin có được xung quanh điểm. Vùng dữ liệu được chọn lân cận điểm truy vấn thường gọi là K-neighborhood.
Module Filters	Chứa các kỹ thuật loại bỏ nhiễu.
Module Geometry	Chứa tất cả các cấu trúc dữ liệu và giải thuật để tính toán hình học.
PCL_IO:	Chứa các hàm và các lớp để đọc và ghi dữ liệu dạng PCD, có thể thu thập dữ liệu từ nhiều nguồn khác nhau (Trong đồ án này dùng Kinect).
PCL_Kdtree	Thư viện cung cấp cấu trúc dữ liệu Kd_tree, sử dụng FLANN giúp nhanh chóng tìm kiếm vùng gần nhất (nearest neighbors searches). Kd-tree là một cấu trúc dữ liệu phân để vùng không gian lưu trữ tập K-dimension điểm dưới dạng cây do đó dễ dàng phân loại và tìm kiếm. Có thể sử dụng để tìm sự tương ứng giữa các nhóm điểm, đặc tả tính năng, định nghĩa các vùng lân cận xung quanh điểm hoặc các điểm.
PCL_Keypoint	Là thư viện chứa thực thi của 2 thuật toán nhận dạng “Point cloud keypoint”. Key Point (hay interest point) là các điểm trong ảnh hoặc trong point cloud mà có tính chất ổn định, riêng

	biệt và có thể dễ dàng phát hiện ra. Thông thường số lượng Key Point nhỏ hơn tổng số điểm trong cloud.
PCL_Octree	<p>Chứa các thuật toán hiệu quả để tạo nên một cấu trúc dữ liệu phân cấp từ dữ liệu point cloud. Nó cho phép phân vùng không gian, downsampling (giảm số lượng các mẫu do đó tăng tốc độ tính toán) và thực hiện các phép toán tìm kiếm trong tập dữ liệu PointCloud. Mỗi nút Octree có 8 nút con hoặc không có nút con nào. Nút gốc (màu đỏ hình dưới) được biểu diễn trong 1 hình lập phương bao toàn bộ các điểm con. Tại mỗi cấp của cây, không quan trọng được chia thành 2 do đó tăng độ phân giải cho điểm ảnh không gian 3 chiều.</p> <p>Thư viện này cũng cung cấp các chương trình tìm kiếm lân cận hiệu quả.</p>
PCL_registration	<p>Kết hợp các bộ dữ liệu vào một mô hình chung, thông nhất thường được thực hiện bằng một kỹ thuật gọi là registration.</p> <p>Ý tưởng chính là xác định các điểm tương ứng trong bộ dữ liệu và tìm một chuyển đổi khoảng cách tối thiểu các điểm tương ứng.</p>
PCL_sample_consensus	Thư viện pcl_sample_consensus có khả năng tách các nhóm điểm có cùng tính chất (Sample Consensus hay SAC) giống như thuật toán RANSAC (Tìm kiếm đường thẳng trong tập hợp các điểm). Các nhóm điểm có thể là các mặt phẳng, mặt cầu, trụ. Thư viện này rất thích hợp trong các ứng dụng dò tìm các đối tượng như tường, cửa, các vật trên bàn...
PCL_Search:	<p>Cung cấp các phương pháp tìm kiếm lân cận (nearest neighbors) bằng cách sử dụng các cấu trúc dữ liệu khác nhau, bao gồm:</p> <ul style="list-style-type: none"> - Kd_tree (từ thư viện PCL_Kdtree) - Octrees (từ thư viện PCL_Octrees) - Brute force (Thuật toán) - Các tìm kiếm đặc biệt cho các bộ dữ liệu có tổ chức.
PCL_Segmentation	Chứa các thuật toán để phân chia Point Cloud thành các nhóm riêng biệt. Các thuật toán này thích hợp nhất khi xử lý các point Cloud bao gồm các vùng không gian bị chồng lấp. Trong trường hợp như vậy, các clustering thường chia nhỏ để sau đó có thể xử lý độc lập.

PCL_surface	<p>Là thư viện thích hợp cho việc xây dựng lại các bề mặt từ dữ liệu quét 3D. Các đối tượng chính gồm vỏ, bề mặt lướt, bề mặt nhẵn hay bình thường. Khi có nhiều có thể làm mịn và lấy mẫu lại.</p> <p>Chia lưới (meshing) là một cách tổng quát để tạo ra các bề mặt điểm. Hiện nay có 2 thuật toán là a very fast triangulation of the original points và a slower meshing that does smoothing and hole filling as well.</p> <p>Có thể dùng thư viện để tạo ra một thân lồi hoặc lõm thích hợp cho đại diện bề mặt đơn giản hóa hoặc chỉ ra các ranh giới.</p>
PCL_visualization	<p>Thư viện được tạo ra có thể nhanh chóng hiển thị các kết quả thuật toán trên dữ liệu 3D. Thư viện cung cấp:</p> <ul style="list-style-type: none"> - Các phương pháp dựng hình và thiết lập thuộc tính ảnh, màu sắc, kích thước cho bất kỳ bộ dữ liệu nào có kiểu “PCL::PointCloud<T>” - Vẽ các hình 3D cơ bản từ bộ điểm hoặc phương trình tham số. - Vẽ các biểu đồ.

Cấu trúc dữ liệu cơ bản của PCL:

- Point Cloud: là kiểu dữ liệu cơ bản trong PCL, Một PointCloud là 1 lớp C++ bao gồm:

Width (int)	Xác định chiều dài tập dữ liệu bằng số lượng điểm. Có nghĩa là có thể xác định tổng số các điểm trong cloud cho bộ dữ liệu có tổ chức hoặc xác định chiều rộng của một tập dữ liệu có tổ chức.
Height (int)	Tương tự Width nhưng là đối với cột trong ma trận điểm. Nếu height = 1 thì dữ liệu không được tổ chức.
Points (int)	Chứa các mảng dữ liệu lưu trữ tất cả các điểm có kiểu PointT.
Is_dense (bool)	Trả về giá trị logic, nếu các giá trị trong points hữu hạn thì trả về TRUE, ngược lại là FALSE.

Bảng 3.1 Bảng thuộc tính PCL.

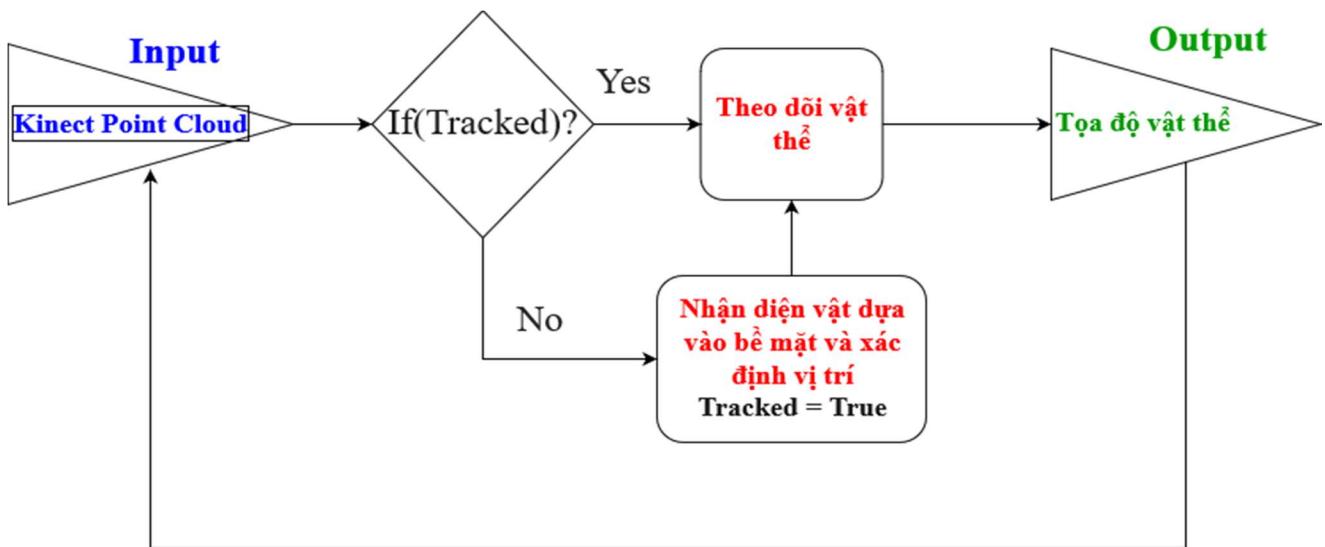
- Định dạng PCD: là một định dạng dùng để lưu trữ dữ liệu 3D Point Cloud. Bao gồm phần HEADER và phần dữ liệu. Mỗi HEADER xác định tính chất, thuộc tính của dữ liệu mà nó lưu trữ. HEADER này được mã hóa

bằng ASCII. Phần còn lại là DATA chưa dữ liệu cụ thể từng điểm, mỗi điểm có thể được mã hóa theo binanry hay ASCII.

VERSION	Xác định phiên bản định dạng PCD.
FIELD	Xác định tên các chiều và các trường của mỗi điểm.
SIZE	Xác định kích thước các chiều tính theo byte.
TYPE	Qui định kiểu của mỗi chiều, qui định bằng kí tự.
WIDTH	Xác định chiều rộng tập dữ liệu tính theo điểm.
HEIGHT	Tương tự WIDTH nhưng tính theo chiều dài.
POINT	Chứa giá trị tổng số điểm ảnh.

Bảng 3.2 Bảng HEADER của PCD.

3.1.4. Tổng quan thuật toán xử lý hình ảnh:



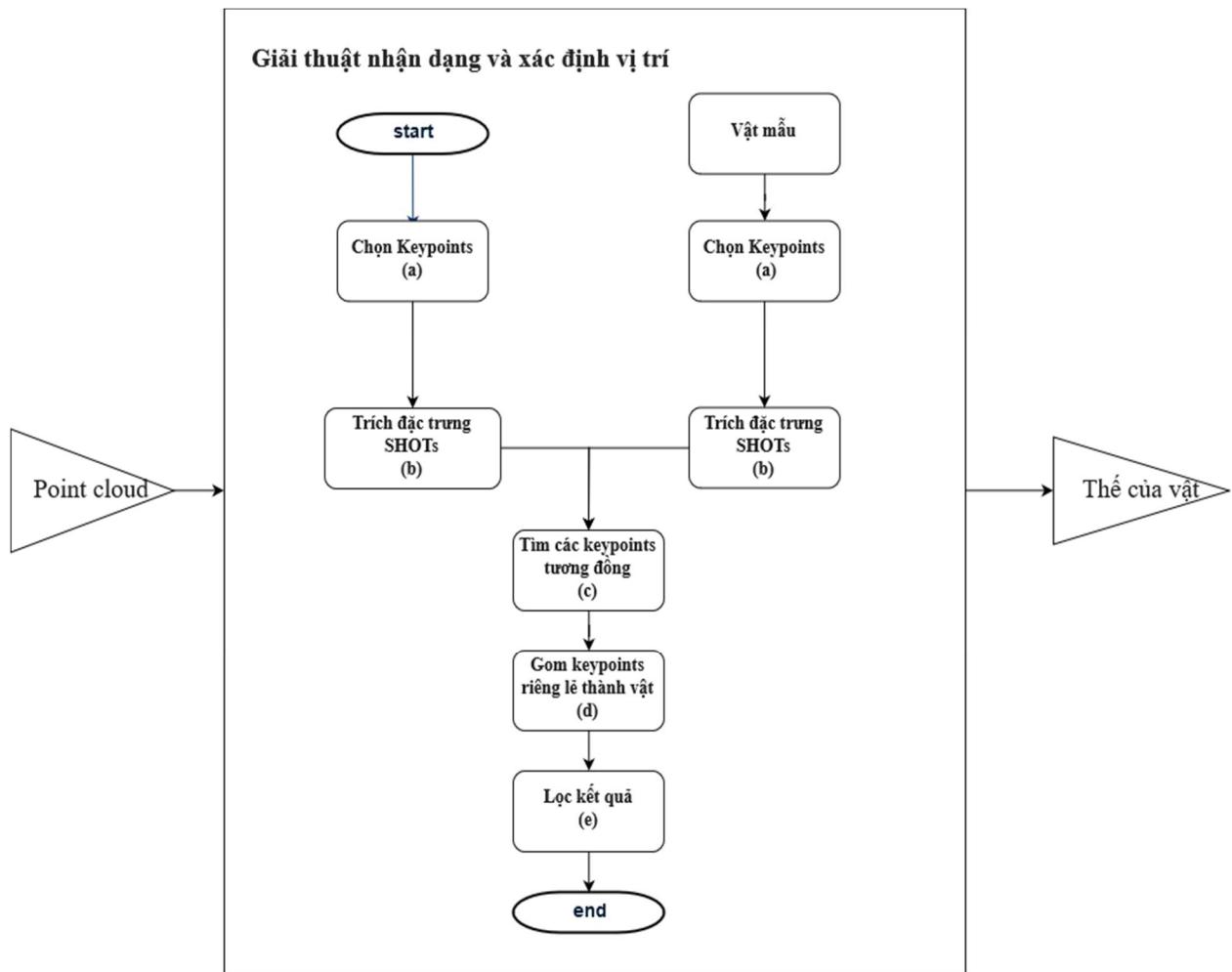
Hình 3.6 Giải thuật xử lý ảnh

Do giải thuật nhận dạng khá phức tạp và tính toán nặng, thuật toán chỉ nhận data từ Kinect, sử dụng giải thuật nhận diện để xác định vật và vị trí tương ứng một lần duy nhất rồi sau đó áp dụng giải thuật theo dõi để bám theo vật.

Giải thuật nhận dạng và xác định vị trí, giải thuật theo dõi sẽ được trình bày kỹ trong các mục 2 và 3, nhóm giới thiệu sơ về input là dữ liệu pointcloud dạng XYZRGB thu được từ Kinect, output là tọa độ [x,y,z] của vật so với cánh tay robot.

3.2. GIẢI THUẬT NHẬN DẠNG VÀ XÁC ĐỊNH VỊ TRÍ:

Giải thuật nhận dạng được áp dụng có cấu trúc như giải thuật nhận dạng hình ảnh đơn giản kinh điển, khác biệt lớn nhất là áp dụng loại đặc trưng riêng cho dữ liệu dạng pointcloud (đặc trưng SHOT [2]) và các thuật toán lọc kết quả.



Hình 3.7 Giải thuật nhận dạng và xác định vị trí vật thể

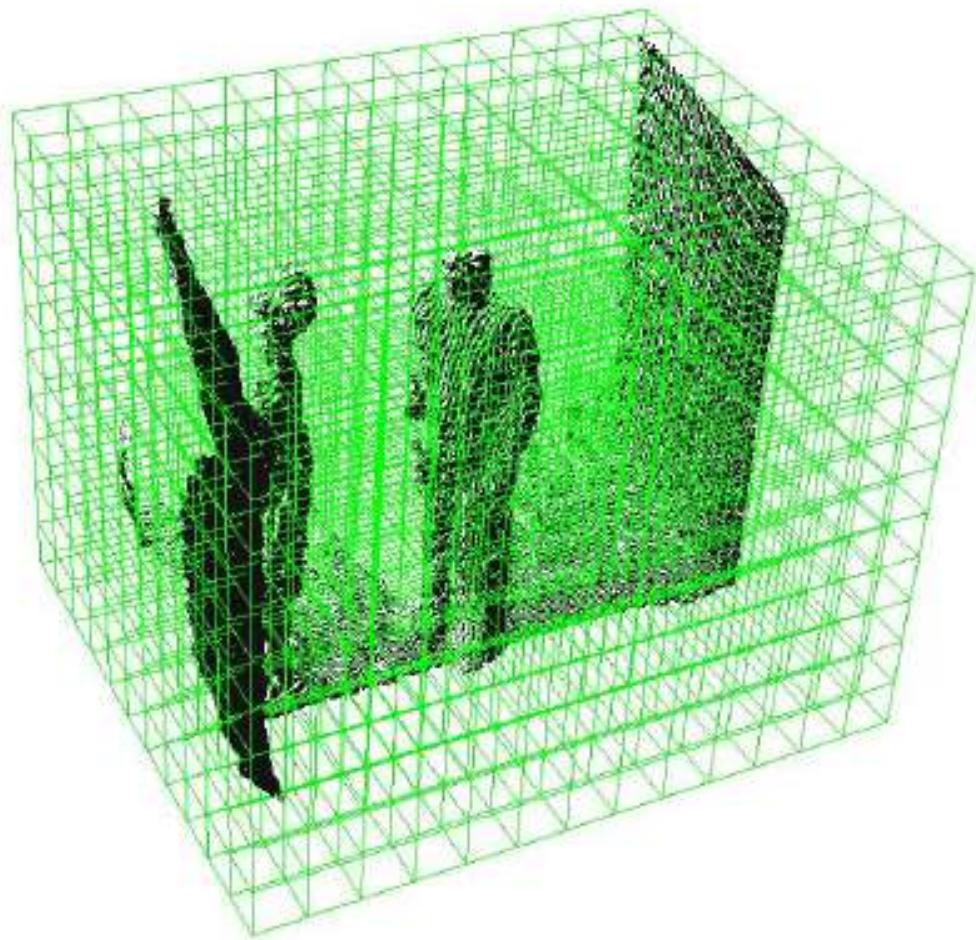
3.2.1. Chọn keypoints:

Mục đích của việc tìm keypoints là thu giản khói lượng dữ liệu đầu vào, giảm thời gian xử lý. Cùng với đó các keypoints có thể cần thỏa các tiêu chí chính: ổn định, đặc biệt. Nói cách khác, mục đích là có thể tìm ra các keypoints tương ứng với keypoints trong Cơ Sở Dữ Liệu (CSDL) khi thay đổi góc quay của camera đối với vật, hay vật ở vị trí khác để so sánh.

PCL hỗ trợ nhiều lớp (class) tìm keypoints, có những phương pháp tách keypoints đi liền với phương pháp tách đặc trưng của nó như: SIFTkeypoint → SIFTdescriptor, NARFkeypoint → NARF descriptor. Với đề tài luận văn này, nhóm chỉ sử dụng phương pháp Uniform sampling và tách đặc trưng SHOT sẽ trình bày rõ ở phần sau, mục đích chính là giảm tối thiểu thời gian xử lý.

Sơ lược về Uniform sampling: chia pointcloud đầu vào thành từng khối cubic nhỏ đều nhau (thành từng voxel grid, xem hình dưới), sau đó tính trọng tâm (centroid) từng khối, lấy trọng tâm làm điểm đại diện cho cả khối.

Nói đơn giản, keypoints nhóm sử dụng chỉ đơn thuần là dữ liệu đầu vào với độ phân giải thấp hơn.



Hình 3.8 Minh họa voxel grid

3.2.2. Đặc trưng SHOTs (Signature of Histogram of Orientation features):

SHOT [2] (viết tắt của Signature of Histogram of Orientation) là đặc trưng có khả năng nhận dạng bề mặt. Từ khả năng này, SHOT có nhiều ứng dụng trong lĩnh vực thị giác máy: số hóa mô hình 3D, ghép bề mặt và xác định đồ vật qua bề mặt.

Trước khi trình bày kỹ về đặc trưng *SHOT*, trích dẫn theo lời của các tác giả trong [2], có thể phân loại các đặc trưng cho công việc nhận dạng bề mặt trên dữ liệu PointCloud 3D thành 2 nhóm: đặc trưng theo Signature (nhóm tạm dịch là đặc trưng theo điểm nhấn riêng) và đặc trưng theo Histogram.

Loại đặc trưng đầu tiên, theo Signature, thường được tạo ra bằng mối liên hệ giữa mặt phẳng tham chiếu cục bộ (local reference frame) của keypoint và từng điểm trong lân cận của nó, cụ thể như các mối liên hệ về hình học. Mặt phẳng tham chiếu cục bộ của keypoint có thể hiểu nôm na là mặt phẳng xấp xỉ cho bề mặt của vật tại keypoint đó.

Loại đặc trưng theo Histogram, theo tên gọi của nó, mô tả vùng lân cận quanh keypoint qua một hay nhiều histogram.

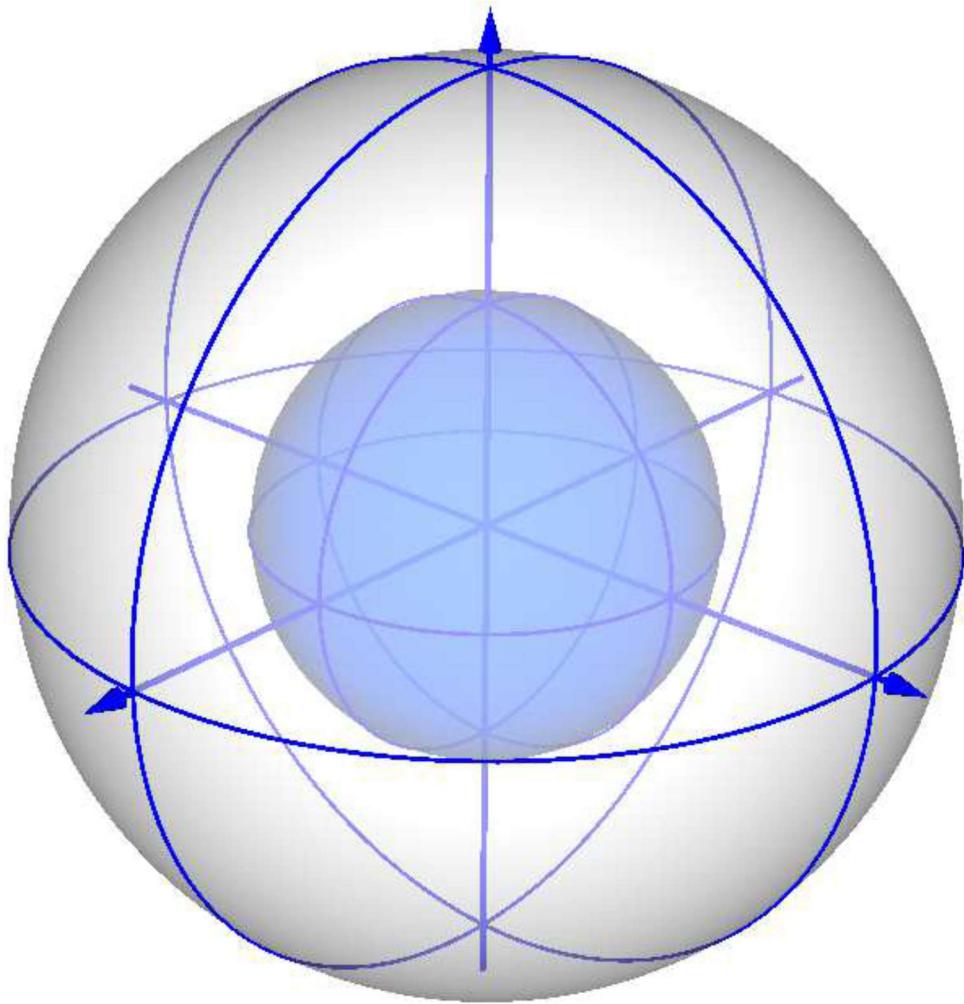
Nhìn chung, đặc trưng theo Signature mang nhiều thông tin về lân cận quanh keypoint, đồng thời lại dễ bị ảnh hưởng bởi những điểm nhiễu. Đặc trưng theo Histogram đổi lại ít mang tính chi tiết, nhưng ổn định hơn do thông tin được thể hiện một cách tổng hợp [2].

Đối với SHOT, tác giả đặt tên là Signature of Histogram Orientation, thể hiện đặc trưng này vừa mang tính cụ thể, vừa áp dụng histogram để tăng độ ổn định cho thuật toán.

Sự khác biệt của đặc trưng SHOT:

Để nói sơ về cách tính đặc trưng SHOT, nhóm muốn nói trước tới kết quả, hay hình thức biểu diễn của đặc trưng, là một mảng gồm 352 phần tử, mô tả围绕 keypoint. Cụ thể, bọc quanh mỗi keypoint là 2 hình cầu đồng tâm, bán kính khác nhau. Mỗi hình cầu chia làm 16 phần theo 4 đường kinh tuyến và 1 đường vĩ tuyến (hình minh họa phía dưới, tuy nhiên hình minh họa với mục đích thể hiện ý tưởng,

chỉ vẽ 2 đường kinh tuyế̂n). Mỗi phần này lại được thể hiện bằng 1 histogram gồm 11 ô. Vậy ta có $11 \times 2 \times 16 = 352$ phần tử đặc trưng.



Hình 3.9 Minh họa cấu trúc của SHOT

Histogram ở đây thể hiện sự phân bố góc của vector pháp tuyến của các điểm lân cận xung quanh từng keypoint. Chính xác hơn là tại mỗi keypoint, SHOT sẽ xác định một mặt phẳng tham chiếu cục bộ theo [3] và [4].¹ Tiếp đó, SHOT tính $\cos \theta_i$ với θ_i là góc giữa vector pháp tuyến tại từng điểm lân cận p_i của keypoint và trực z của mặt phẳng tham chiếu của keypoint. Việc sử dụng hàm $\cos \theta_i$ đơn giản vì tính

¹ Tuy nhiên, việc xấp xỉ mặt phẳng trong SHOT được hiệu chỉnh thêm trọng số để tăng độ ổn định khi có sự xuất hiện của nhiều vật trong hình (vân đê cluttering), cùng với đó là thêm các bước xác định hướng cho các trục x, y của mặt phẳng, khi thuật toán gốc chỉ xác định hướng của trục z [2].

toán nhanh ($\cos \theta_i = z_{keypoint} \cdot n_{pi}$) và giá trị của $\cos \theta_i$ cũng dễ dàng thể hiện góc θ_i trong không gian.

Sau khi tính được toàn bộ Histogram cho toàn không gian dữ liệu, như đã đề cập, mỗi keypoint được đại diện bởi 32 phần không gian, và mỗi không gian được mã hóa bằng các bộ histogram đã tính. Việc này giúp SHOT lưu giữ thông tin cục bộ của không gian quanh từng keypoint (tính Signature) nhưng đồng thời cũng mang tính ổn định cao.

Có thể nhận xét, đặc trưng SHOT mang hơi hướng giống đặc trưng SIFT [5] khi mà các histogram sử dụng đều thể hiện thông tin của nhiều vùng lân cận nhỏ quanh các keypoint, vừa lưu giữ chính xác thông tin của keypoint, vừa ổn định với nhiều hay các không lý tưởng.

PCL có hỗ trợ giải thuật SHOT với header <[pcl/features/shot.h](#)>

3.2.3. Tìm các keypoints tương đồng (Matching):

Bước này chỉ đơn thuần là xây dựng KdTree với 352 chiều cho vật mẫu, so sánh các keypoint với đặc trưng từ dữ liệu đầu vào và keypoint với đặc trưng tính sẵn trong cơ sở dữ liệu. Cuối cùng là lưu lại những cặp keypoint có sai biệt theo khoảng cách Euclidean dưới 1 mức định sẵn.

PCL có hỗ trợ class template `pcl::KdTreeFLANN` cùng header <[pcl/kdtree/kdtree_flann.h](#)> rất tiện lợi và hiệu quả.

3.2.4. Gom keypoints thành vật (Clustering – Correspondence grouping):

Nhóm áp dụng thuật toán Geometric Consistency Grouping (GCG) được trình bày trong [6], mục đích là nhóm các keypoints gần nhau và tạo ra những cấu trúc nhiều khả năng là vật mẫu (tạm gọi những cấu trúc này là những giả thiết). Theo đó, giả sử tồn tại phép biến đổi bảo toàn cấu trúc (rigid transformation) để biến vật mẫu thành một hay nhiều giả thiết tạo được. Thuật toán sẽ tìm ra phép biến đổi cùng với giả thiết có cấu trúc hình học đồng bộ nhất với cấu trúc vật mẫu. Cụ thể, GCG dựa trên thuật toán Iterative Geometric Consistency (IGC) [7]. Bắt đầu từ 1 keypoint, IGC tuần tự liên kết các keypoint lân cận khác với yêu cầu thỏa mãn điều kiện:

$$\left| \left\| p_i^m - p_j^m \right\|_2 - \left\| p_i^s - p_j^s \right\|_2 \right| < \varepsilon$$

Với p_i^m, p_j^m là 2 keypoint lân cận của vật mẫu, p_i^s, p_j^s là 2 keypoint tương ứng trong dữ liệu đầu vào và $\left\| p_i^m - p_j^m \right\|_2$ là chuẩn 2 hay khoảng cách Euclidean giữa tọa độ 2 keypoint. Cuối cùng ε là độ sai lệch tối đa cho phép.

Cùng với điều kiện trên, GCG thêm một điều kiện

$$\left| n_i^m \cdot n_j^m - n_i^s \cdot n_j^s \right| < \varepsilon_n$$

Cách ký hiệu cũng giống như trên, với n là vector pháp tuyến tại keypoint tương ứng, ε_n là góc lệch tối đa cho phép.

3.2.5. Lọc kết quả (Hypothesis Verification):

Các giả thiết được tạo ra từ phần d, sẽ được đưa qua bước kiểm định cuối cùng.

Được trình bày như trong [6], mục đích của phần này để tăng tối đa số lượng vật thể nhận dạng đúng (True Positive) đồng thời giảm thiểu tối đa giả định sai (False Positive)

Sơ lược, hàm đánh giá cho mức độ phù hợp của một giả thiết, được tạo nên từ 4 tiêu chí:

- Scene fitting
- Model outliers

Các giả thiết được “ướm” thử lại vào dữ liệu đầu vào. Các giả thiết ở đây được hiểu là những phép biến đổi bảo toàn cấu trúc, theo đó cấu trúc vật mẫu được tịnh tiến và xoay vào dữ liệu thu được từ camera, sau đó đánh giá độ phù hợp của giả thiết. Thước đo được sử dụng là khoảng cách Mahalanobis [8] theo vector đặc trưng của từng điểm trong vật mẫu đối với dữ liệu đầu vào trong không gian lân cận [6].

- Multiple assignment

Những giả thiết tiếp đó được xem xét có phù hợp với không gian vật lý hay không, ví dụ nếu 2 giả thiết về 2 vật khác nhau lại có cùng 1 vị trí sẽ bị giảm khả

năng có là True Positive, ngược lại 2 giả thiết cùng 1 vị trí và là cùng 1 vật sẽ được gộp lại và có khả năng là True Positive.

- Clutter

Trong nhiều trường hợp, vật cần nhận diện có thể bị che lấp và để lẫn lộn chung với các vật khác, dẫn đến việc nhận dạng sai đối với những giả thiết không phù hợp (False Positive). Vì lý do đó, [6] sử dụng một ảnh thể hiện dữ liệu đầu vào theo trường độ sâu (ảnh range - chỉ thể hiện khoảng cách dữ liệu tới camera), theo [9] và [10] xác định những vùng bị che lấp và loại bỏ những giả thiết không phù hợp.

Sau khi đánh giá 4 tiêu chí trên, hàm đánh giá cho ra một mảng dạng bool cùng số phần tử với mảng chứa giả thiết, trong đó những giả thiết phù hợp có giá trị là 1.

Thuật toán nhóm áp dụng hiện tại chỉ gồm 2 tiêu chí đầu.

Tiêu chí thứ 3 đòi hỏi nhiều giả thiết tạo ra bởi nhiều loại đặc trưng khác nhau như SHOT [2], SIFT [5], OUR-CVFH [11] (được trình bày trong [6])

Tiêu chí thứ 4 đòi hỏi phải tạo ra một ảnh range để xác định những phần không gian bị che lấp theo [9] và [10].

Cả 2 tiêu chí này đều đòi hỏi nhiều thời gian nghiên cứu và cũng có thể khiến thời gian xử lý của toàn bộ giải thuật tăng lên nhiều lần. Do đó nhóm sẽ để ngỏ 2 phần này và hy vọng giải thuật có thể được tiếp tục phát triển bởi những người khác sau này.

3.3. GIẢI THUẬT THEO DÕI VẬT THỂ:

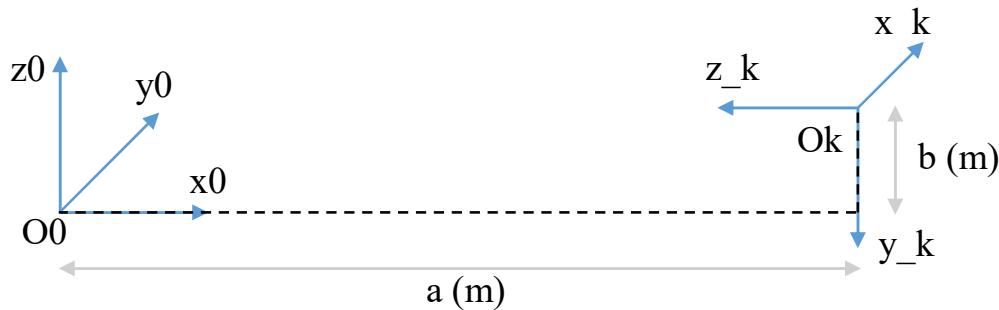
Thuật toán nhóm sử dụng là Particle filter [11] với không gian trạng thái (state space) dùng để theo dõi là màu HSV và vector pháp tuyến của điểm.

PCL có sẵn class ParticleFilterTracker nên có thể dễ dàng áp dụng thuật toán.

Đầu ra của thuật toán theo dõi vật thể sẽ là bộ tọa độ 3 chiều của vật đối với trục gốc của Kinect theo dạng [x, y, z].

Tuy nhiên để truyền xuống bộ điều khiển cánh tay robot và áp dụng thuật toán động học ngược tính ra các góc điều khiển của robot, ta cần chuyển tọa độ từ so với trục gốc của Kinect chuyển sang trục gốc của mô hình cánh tay robot.

Nhóm xây dựng mô hình đặt vị trí camera Kinect và mô hình cánh tay robot như sơ đồ sau:



Hình 3.10 Sơ đồ hệ trục tọa độ gốc robot và trục tọa độ gốc tọa độ Kinect.

Từ sơ đồ trên, ta lập ma trận chuyển đổi tọa độ từ tọa độ vật so với hệ trục tọa độ Kinect $O_kx_ky_kz_k$ sang tọa độ vật so với trục gốc của robot $O_0x_0y_0z_0$.

$$\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 & a \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & b \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} k_x \\ k_y \\ k_z \\ 1 \end{bmatrix}$$

Trong đó $[p_x \ p_y \ p_z]^T$ là bộ tọa độ so với trục gốc robot, $[k_x \ k_y \ k_z]^T$ là bộ tọa độ so với trục gốc của Kinect như sơ đồ trên.

Cách xây dựng ma trận chuyển đổi được đề cập trong kiến thức cơ bản các phép chuyển tọa độ trong xử lí và kĩ thuật robot, nhóm không nhắc lại lí thuyết tại báo cáo luận văn.

3.4. THIẾT LẬP - CÀI ĐẶT HỆ THỐNG CHO GIẢI THUẬT XỬ LÝ ẢNH

Phần này nhằm hỗ trợ những bạn khác dễ dàng cài đặt và chạy ứng dụng của nhóm đã viết, đồng thời nhóm đưa ra cấu hình hệ thống và tốc độ xử lý hiện tại để các bạn có thể có những quyết định phù hợp nếu muốn phát triển tiếp.

3.4.1. Cấu hình hệ thống:

Nhóm sử dụng một laptop cho công việc xử lý ảnh với cấu hình như sau:

- Hệ điều hành: Ubutuntu 16.04 LTS 64-bit
- Bộ nhớ: 12 GiB
- Vi xử lý: Intel® Core™ i5-5200U CPU @ 2.20Ghz x 4

Cấu hình này cho phép chạy thuật toán nhận dạng và theo dõi trong thời gian chấp nhận được.

3.4.2. Cài đặt thư viện và thử nghiệm:

Để có thể sử dụng được chương trình, cần cài đặt 3 thư viện chính là PCL và OpenNI và Qt 5.9.

PCL như đã đề cập là thư viện hỗ trợ các thuật toán xử lý pointcloud. Thư viện OpenNI chỉ đơn giản là driver để máy tính có thể lấy dữ liệu từ Kinect.

PCL có thể dễ dàng cài đặt qua hướng dẫn ở trang chủ của PCL ([12]). Đối với OpenNI, cách cài đặt có phần phức tạp hơn do cần nhiều thư viện hỗ trợ, tuy nhiên đều có sẵn hướng dẫn trên trang Github ([13]).

Tuy nhiên, cách đơn giản nhất để cài đồng thời cả PCL và OpenNI là cài thư viện Robot Operating System (ROS). Đây là thư viện mã nguồn mở chuyên sử dụng để mô phỏng, thiết kế và điều khiển robot cùng nhiều tiện ích được tích hợp sẵn, trong đó có xử lý ảnh. Để cài ROS với hệ điều hành Ubuntu [16.04 LTS](#), chỉ cần thực hiện tuần tự các lệnh sau trong terminal [14]:

- Lấy đường dẫn tải về từ kho package của ROS, thêm vào đường dẫn mặc định của Ubuntu:

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

- Cài đặt key để tải về.

```
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116  
sudo apt-get update
```

- Tải về ROS phiên bản kinectic-kame.

```
sudo apt-get install ros-kinetic-desktop-full
```

3.4.3. Thử nghiệm:

3.4.3.1. Kiểm tra kết nối

Sau khi cài đặt PCL và OpenNI, thử kết nối Kinect với máy tính² (đồng thời kết nối nguồn cho Kinect) và kiểm tra kết nối bằng cách nhập lệnh:

```
lsusb
```

Nếu không có gì bất thường, terminal sẽ trả về kết quả dạng như sau:

```
Bus 001 Device 005: ID 045e:02b0 Microsoft Corp. Xbox NUI Motor  
Bus 001 Device 006: ID 045e:02ad Microsoft Corp. Xbox NUI Audio  
Bus 001 Device 007: ID 045e:02ae Microsoft Corp. Xbox NUI Camera
```

3.4.3.2. Chạy thử chương trình đơn giản

Chạy lệnh để cài kho các chương trình demo của PCL

```
sudo apt-get install pcl-tools
```

pcl-tools chứa một vài chương trình mẫu đã biên dịch sẵn. Để chạy thử, mở terminal và chạy các chương trình bắt đầu với từ khóa `pcl_*`. (ví dụ `pcl_openni_viewer`)

² Lưu ý: Kinect chỉ có thể hoạt động với cổng USB 2.0.



Hình 3.11 Chương trình mẫu pcl_openni_viewer

3.4.3.1. Chạy thử chương trình xử lý ảnh

Chương trình xử lý ảnh được đặt tên là MyEyes. Để chạy, mở terminal và di chuyển đến folder chứa MyEyes, thực hiện lệnh ./MyEyes.

Lưu ý có thể đường dẫn đến thư viện Qt ở mỗi máy tính là khác nhau, dẫn đến việc báo các lỗi như:

/usr/lib.x86_64-linux-gnu/libQt5Gui.so.5 version ‘Qt 5’ not found

/usr/lib.x86_64-linux-gnu/libQt5Gui.so.5 version ‘Qt 5’ not found

Libcui18n.so.56 not found

Việc cần làm là thực hiện lệnh sau để kết nối các file của thư viện Qt ứng với đường dẫn phù hợp

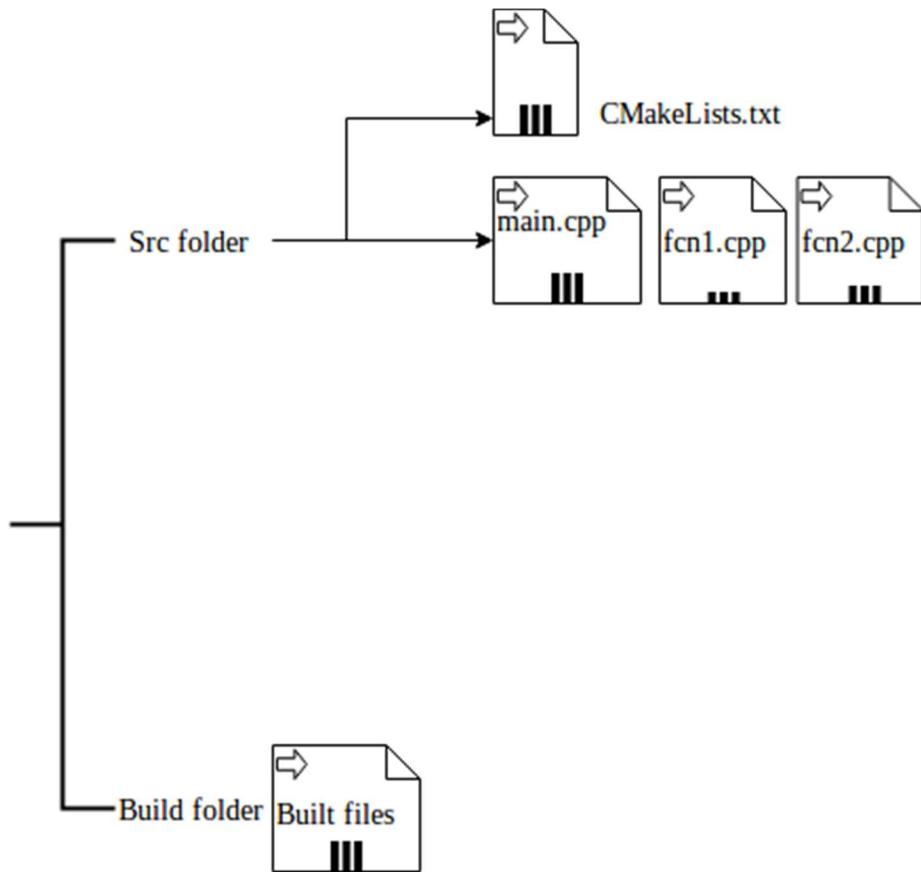
```
sudo ln -sf <đường dẫn tới nơi cài Qt>/gcc_64/lib/libQt<abc>.so.5  
/usr/lib.x86_64-linux-gnu/libQt5Gui.so.5 version
```

Nếu không có lỗi gì xảy ra, chương trình sẽ mở lên với giao diện:

<insert picture here>

3.4.3.2. Cách biên dịch chương trình

PCL sử dụng CMAKE để biên dịch. Theo đó, để biên dịch một chương trình PCL từ code, nên cấu trúc hệ thống file như sau:



Trong đó, ở folder Src, ngoài các file chứa mã nguồn còn có thêm sự xuất hiện của file CMakeLists.txt chứa các chỉ dẫn để biên dịch chương trình, lấy ví dụ cụ thể:

```
cmake_minimum_required(VERSION 2.8 FATAL_ERROR)
project(openni_grabber)          // Tên chương trình
find_package(PCL 1.7 REQUIRED)    // Kiểm thư viện cần thiết
include_directories(${PCL_INCLUDE_DIRS}) // Thêm vào các
file header
link_directories(${PCL_LIBRARY_DIRS}) // Đường dẫn thư viện
cho linker
add_definitions(${PCL_DEFINITIONS}) // Thêm các preprocessor của
PCL
```

```

add_executable (openni_grabber openni_grabber.cpp) // Đường dẫn tới
các file // source code
target_link_libraries (openni_grabber ${PCL_LIBRARIES})

```

Để biên dịch chương trình, sử dụng terminal di chuyển tới folder Build, sau đó thực hiện các lệnh:

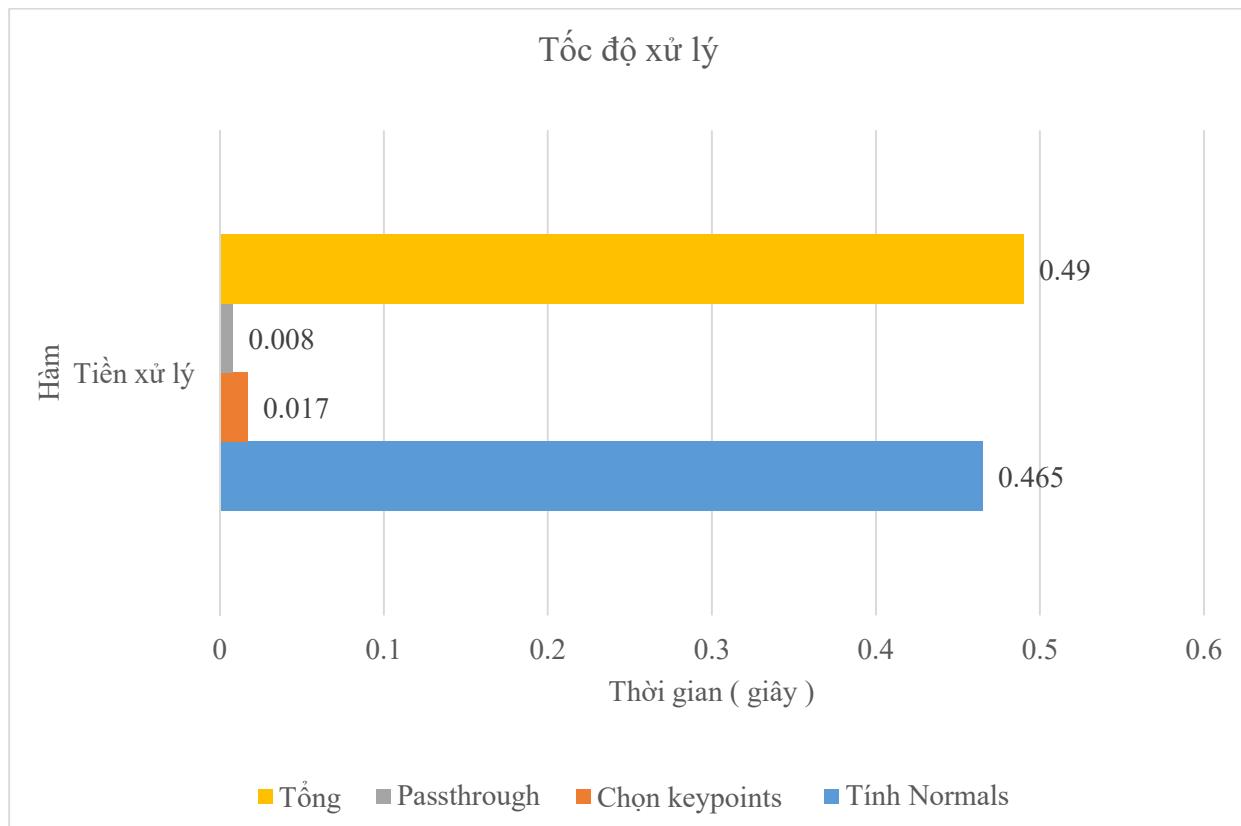
```

cmake <đường dẫn tới folder src>
make

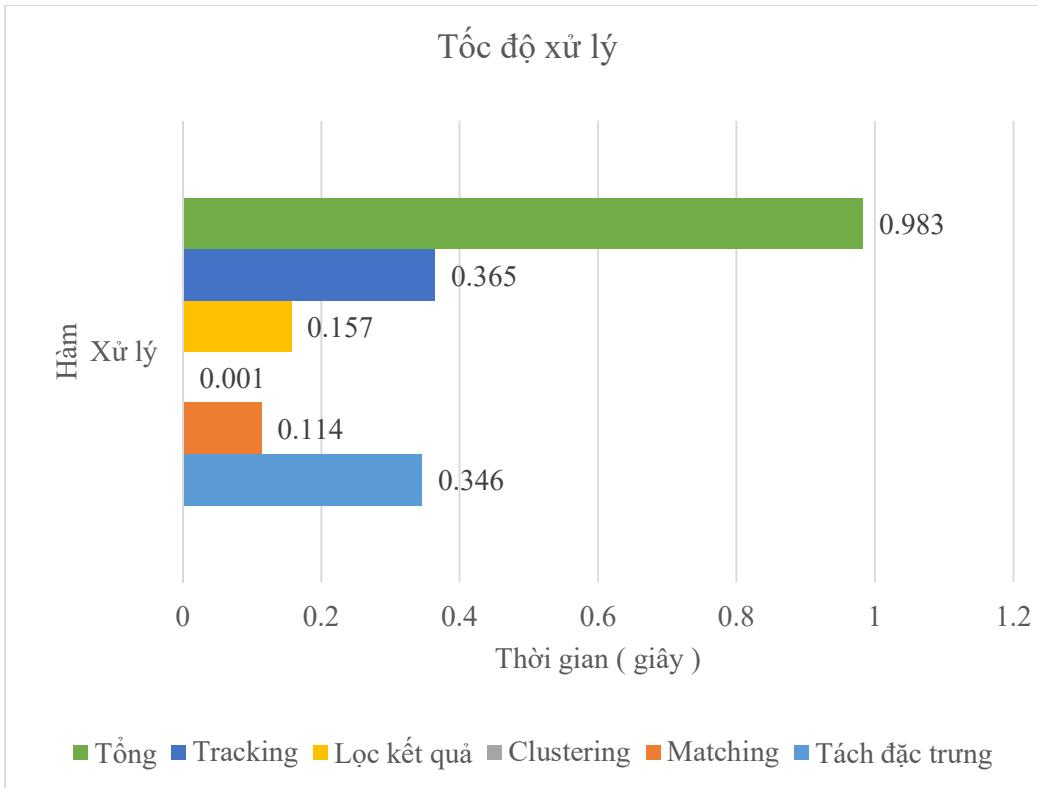
```

3.4.3.3. Tốc độ xử lý

Với cấu hình đề cập ở phần a, nhóm đo thời gian xử lý một số hàm thiết yếu của thuật toán:



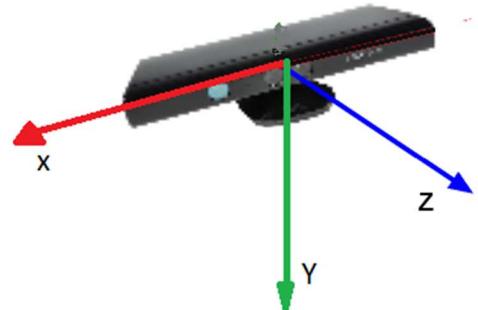
Hình 3.12 Thời gian xử lý các hàm tiền xử lý dữ liệu hình ảnh.



Hình 3.13 Thời gian xử lý các hàm xử lý nhận diện và theo dõi hình ảnh.

Lưu ý khi sử dụng thuật toán PCL và OpenNI để tính toán và trả về tọa độ vật trong không gian, vị trí hệ trục tọa độ gốc của Kinect được xác định như sau:

Theo đó, gốc tọa độ nằm ở đỉnh của camera giữa, cách mặt đất 0.08m. Trục z hướng sang phía trước, trục x hướng sang trái và trục y hướng lên trên.

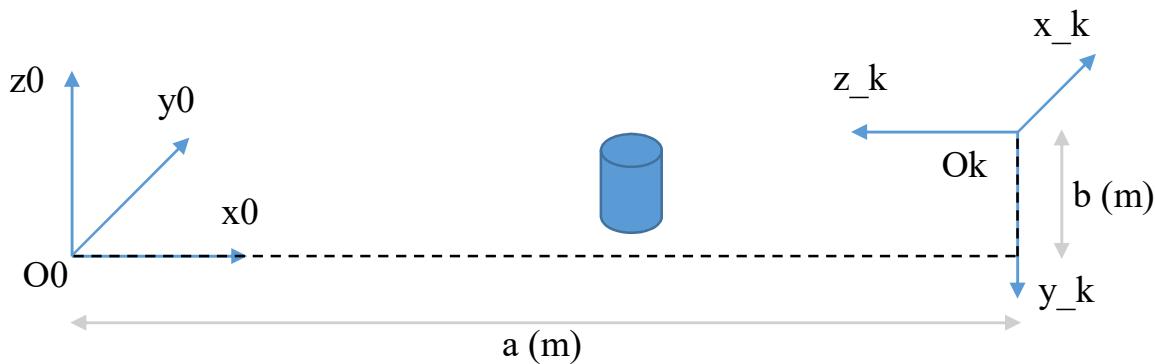


Hình 3.14 Hệ trục tọa độ của Kinect trên OpenNI

CHƯƠNG 4: CHƯƠNG TRÌNH ĐIỀU KHIỂN TRUNG TÂM:

4.1. SƠ ĐỒ VỊ TRÍ MÔ HÌNH LUẬN VĂN:

Như đã đề cập trong mục 3.3. Giải thuật theo dõi vật thể, sơ đồ vị trí của cánh tay robot công nghiệp và vị trí đặt camera Kinect để xác định tọa độ vật và bám theo vật thể như sau:

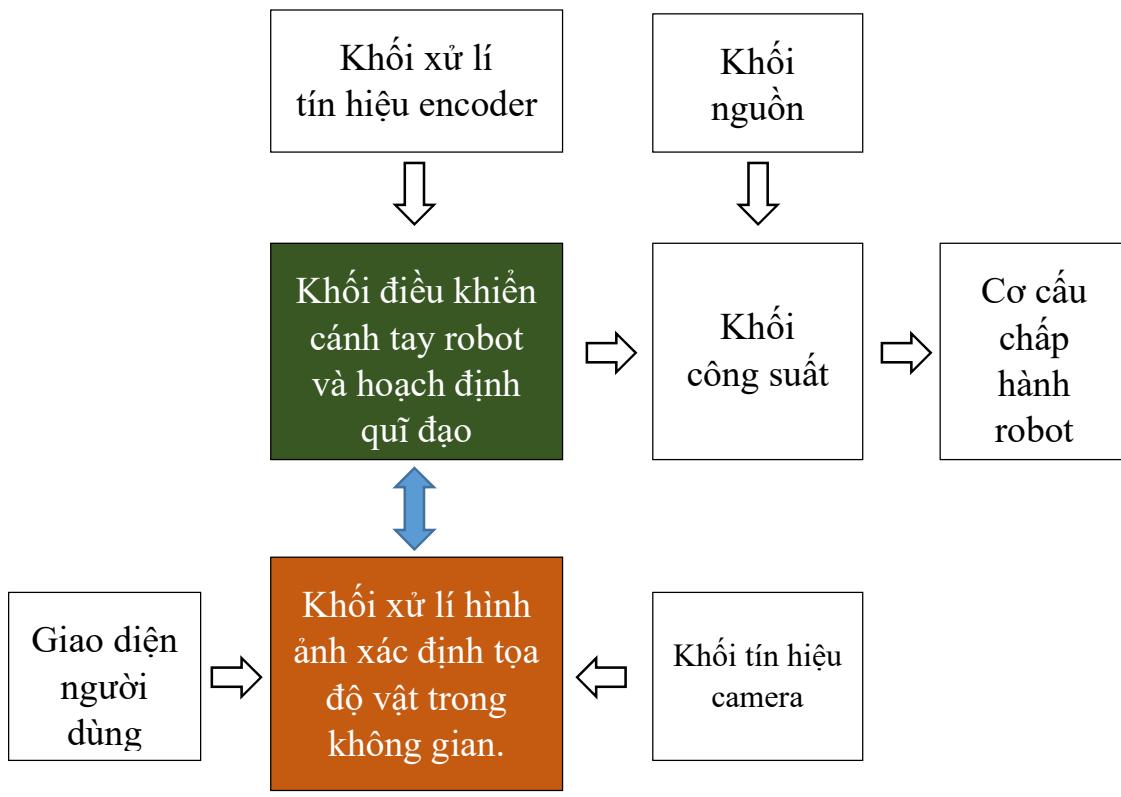


Hình 4.1 Sơ đồ bố trí vị trí robot công nghiệp, camera Kinect và vật thể cần nhận diện và theo dõi.

Trong giao diện chương trình điều khiển trên máy tính sẽ cho phép nhập vào các khoảng cách a và b tương ứng để thuật toán trả về vị trí chính xác với vị trí đặt các thiết bị trong thực tế.

4.2. SƠ ĐỒ KHỐI TỔNG QUAN CHƯƠNG TRÌNH ĐIỀU KHIỂN:

- Toàn bộ chương trình điều khiển được chia làm 2 phần chính:
 - Bộ điều khiển cánh tay robot được nhúng hoàn toàn trên vi điều khiển.
 - Chương trình xử lý ảnh – giao diện người dùng trên máy tính.



Hình 4.2 Sơ đồ khái niệm quan chung trinh điều khiển.

- 2 khối được giao tiếp với nhau bằng phương thức USART và được xây dựng protocol giao điểm đảm bảo các lệnh yêu và và dữ liệu giao tiếp được chính xác:

HEADER	DATA LENGTH	DATA (n bytes)		CRC
1 BYTE	1 BYTE	DATA[0]: Command	DATA[1..n-1]: Data Value	1 BYTE
Device ID	Độ dài chuỗi Data	Lệnh thực thi	Dữ liệu đi kèm (nếu có)	Check CRC

Bảng 4.1 Protocol giao tiếp USART PC - VDK.

Command	Instructions	Data
MOVETOXYZ	Update destination coordinate. → Calculate inverse kinematic. → If success, update desired positions for each joints. Otherwise send back the status “out of range” and stay the position.	1 float X coordinate 1 float Y coordinate 1 float Z coordinate
DROP	Do the drop action.	None
GRAB	Do the grab action.	None
STOP	Set all the PWM to zero and stop controller of each joints.	None

Bảng 4.2 Các lệnh gửi từ PC đến STM32F407VGT.

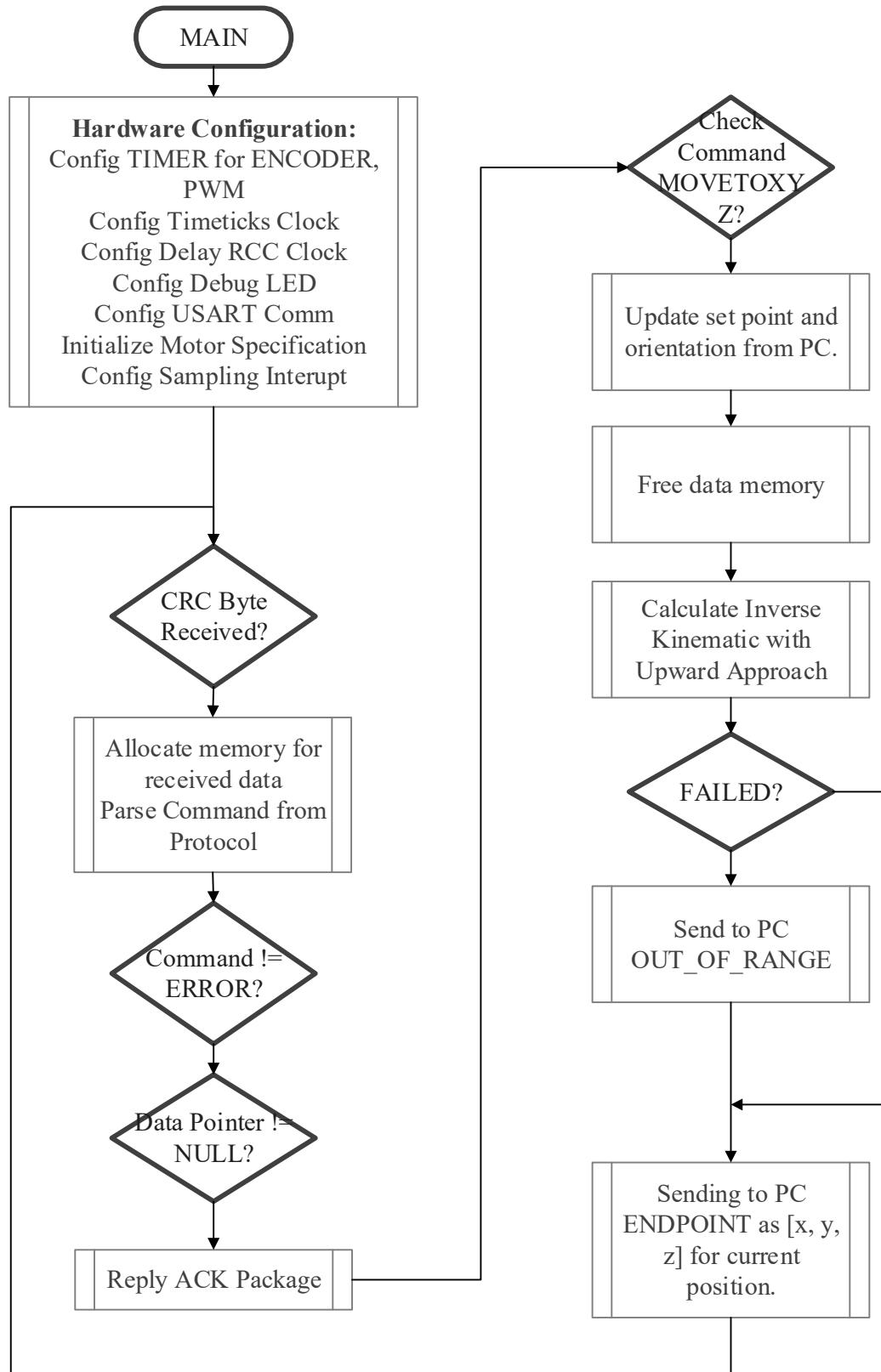
Command	Instructions	Data
OUT_OF_RANGE	Show the error in UI: Out of range.	None.
END_POINT	Send back the Coordinate of End-point by calculate the forward kinematics from measured positions of each joints.	1 float X coordinate 1 float Y coordinate 1 float Z coordinate

Bảng 4.3 Các lệnh gửi từ STM32F407VGT đến PC.

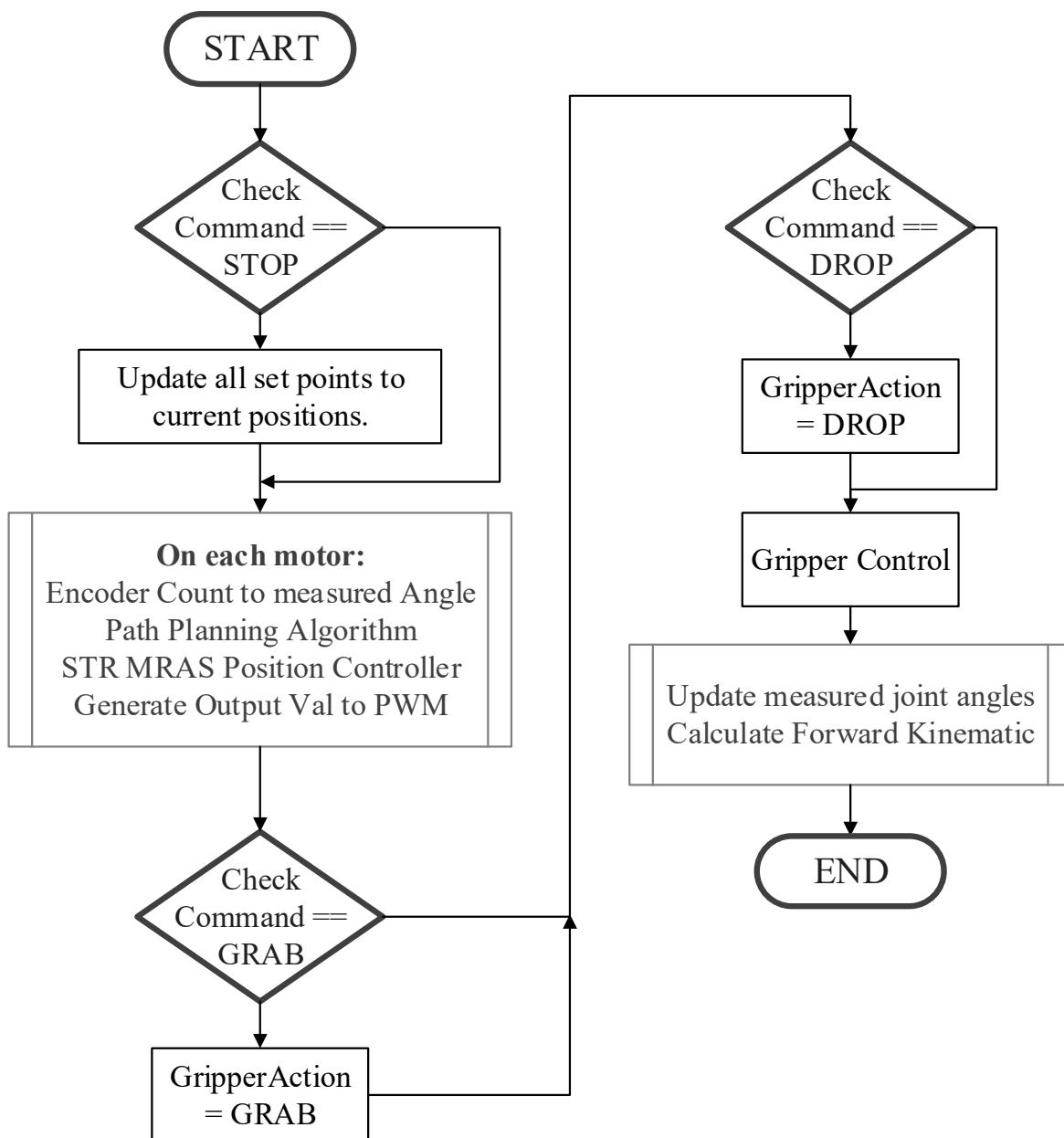
4.3. CHƯƠNG TRÌNH NHÚNG TRÊN VI ĐIỀU KHIỂN:

Như đã đề cập ở chương 2, mục 2.2.1, trong chương trình điều khiển, cần lưu ý với mô hình robot như trên, động cơ ở vị trí ELBOW không điều khiển góc θ_3 mà điều khiển góc $\theta_{23} = \theta_2 + \theta_3$, quá trình xây dựng giải thuật phần mềm và bài toán động học thuận, nghịch cần điều chỉnh cho phù hợp.

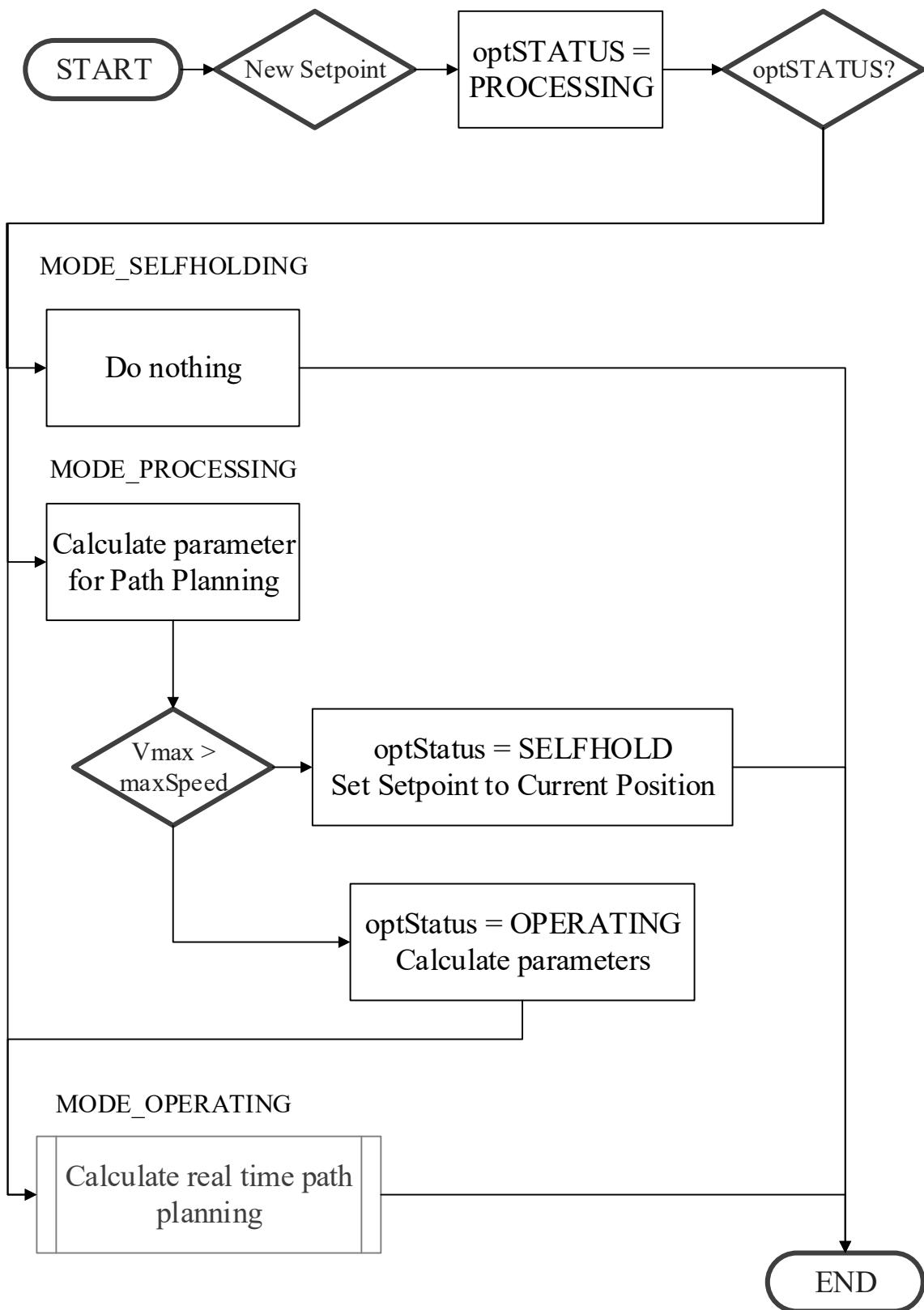
Chương trình xử lí bộ điều khiển cánh tay robot được xây dựng qua sơ đồ trạng thái như sau, các hàm và quá trình xử lí chi tiết có thể tìm hiểu sâu hơn trong source code chương trình, nhóm không đưa flow chart các bước đó để giao sơ đồ trạng thái tổng quan và dễ hiểu nhất cho người đọc:



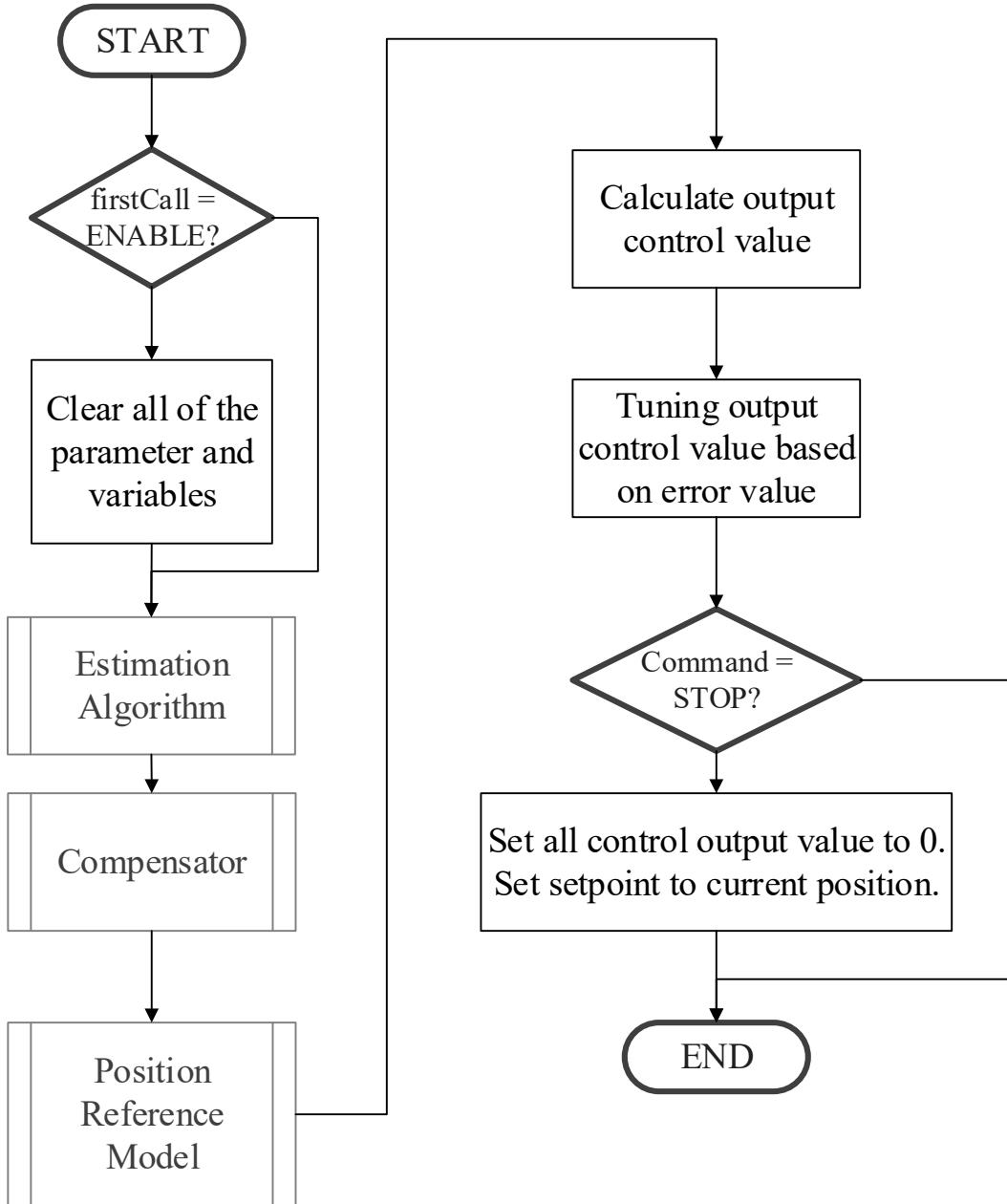
Hình 4.3 Sơ đồ trạng thái vòng lặp main().



Hình 4.4 Sơ đồ trạng thái chương trình ngắn theo chu kỳ 10 ms.



Hình 4.5 Sơ đồ thuật toán qui hoạch quỹ đạo Path Planning.



Hình 4.6 Sơ đồ bộ điều khiển vị trí STR MRAS.

Trong chương trình nhóm đã xây dựng lại một số thư viện phục vụ cho việc tính toán và kế thừa chương trình, danh sách các thư viện như sau:

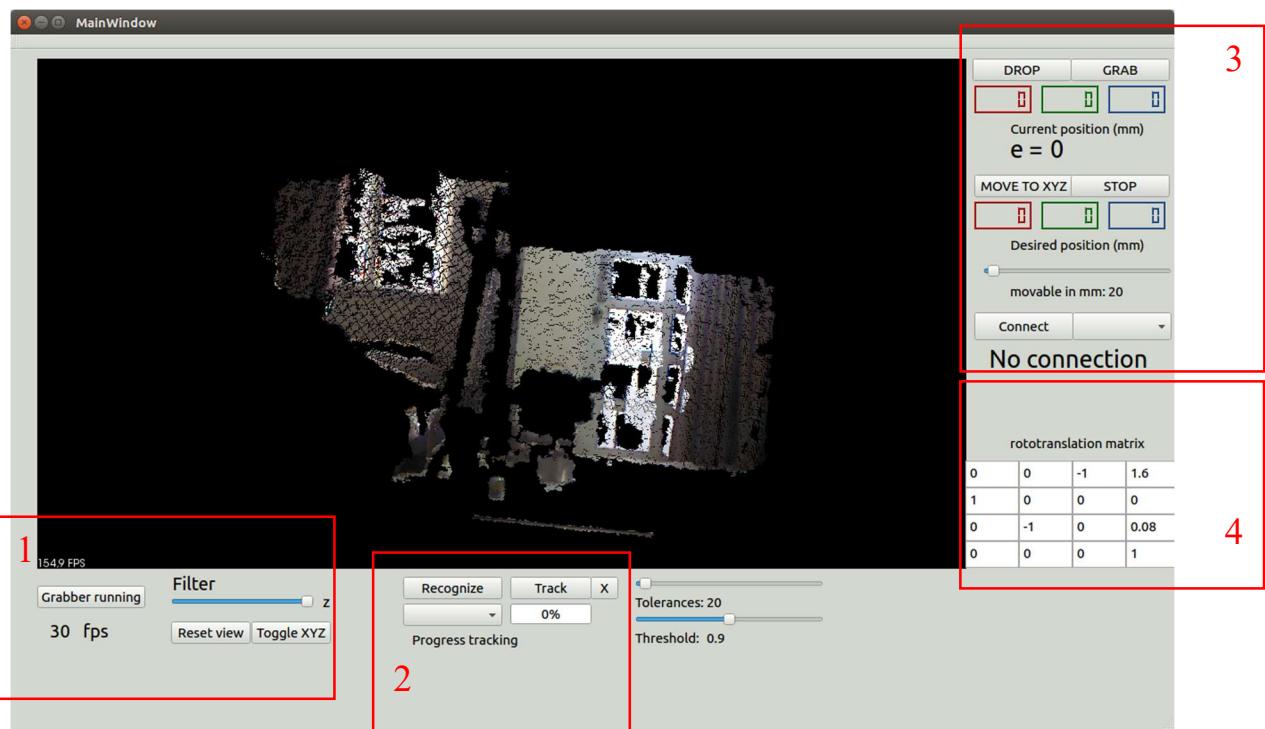
- Matrix: phục vụ việc tính toán công thức nhân chia ma trận 6x6.
- Controller: các bộ điều khiển STR MRAS và thuật toán điều tốc Path Planning.

- Ultis: Các hàm tách nối các biến phục vụ cho việc truyền nhận dữ liệu trên từng byte.
- User_uart: Xây dựng các hàm xử lý truyền nhận UART bằng Protocol giao tiếp.
- Configuration: Cấu hình chức năng của chương trình.

Ngoài ra nhóm cũng xây dựng các đoạn chương trình debug thêm để tinh chỉnh hệ thống thông qua chương trình STM32 Studio của hãng ST, có hỗ trợ giám sát, đọc và thay đổi thông số các biến toàn cục của chương trình.

4.4. CHƯƠNG TRÌNH GIAO DIỆN NGƯỜI DÙNG VÀ XỬ LÝ ẢNH TRÊN MÁY TÍNH:

Giao diện người dùng được xây dựng bằng phần mềm QtCreator 5.9 trên Ubuntu, hình ảnh giao diện như sau:



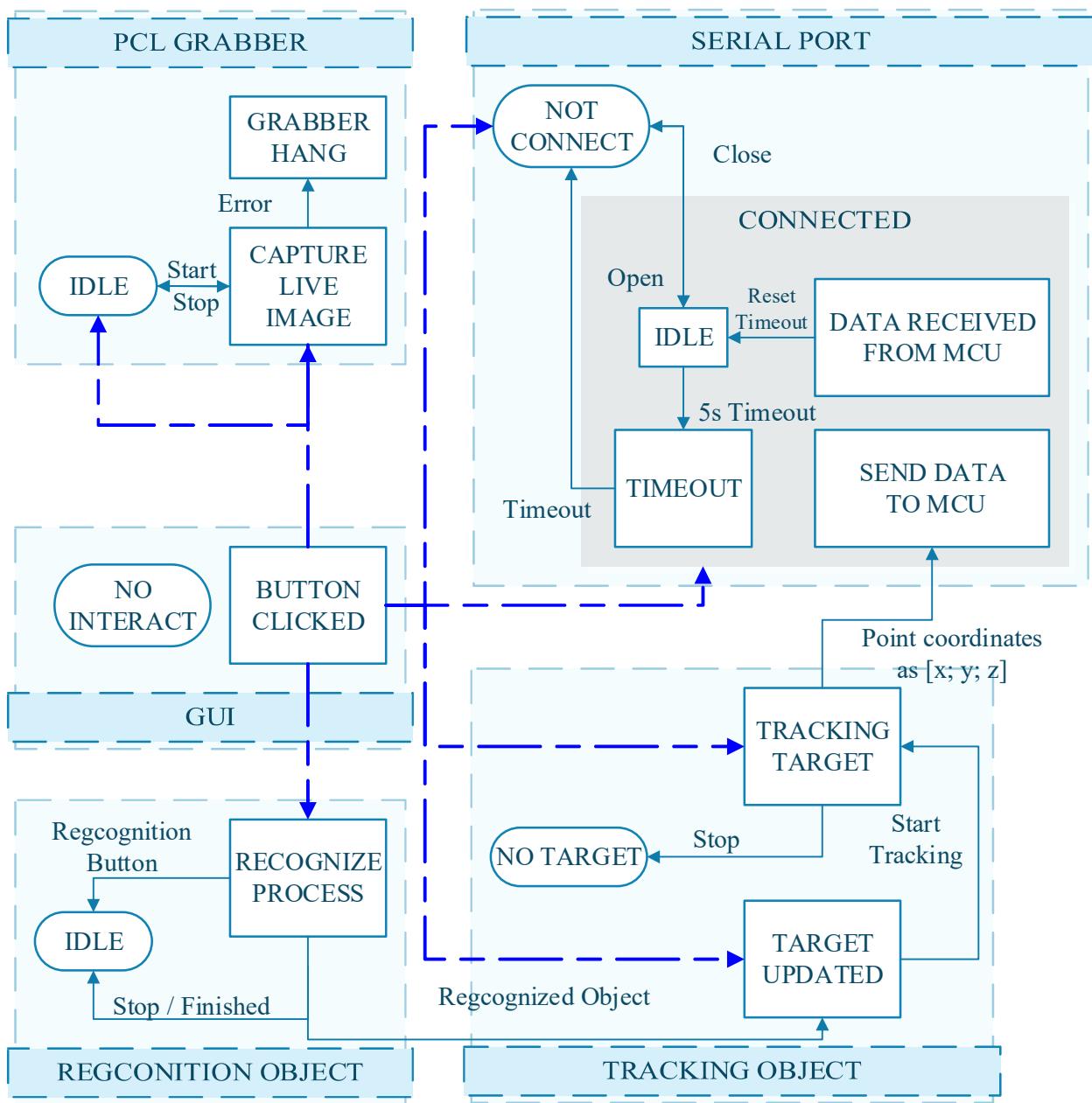
Hình 4.7 Giao diện người dùng trên máy tính.

Khu vực	Nút nhấn	Giải thích chức năng
---------	----------	----------------------

1	Reset Grabber	Tái khởi động nhận dữ liệu từ Kinect
	Reset View	Reset góc nhìn ban đầu
	Toggle XYZ	Bật/Tắt hệ trục gốc của Kinect
	Filter z	Hiệu chỉnh chiều sâu lấy dữ liệu hình ảnh, đơn vị là meter
2	Regconize	Hiện cửa sổ chọn model cần nhận diện.
	Track	Sau khi học nhận diện hoàn tất 100%, chọn track để bắt đầu bám vật, ấn [x] để dừng.
	List box	Trong trường hợp nhận diện được nhiều vật giống nhau trong khung hình, chọn vật thể cần nhận diện tại khung này.
3	DROP	Gửi lệnh mở tay gấp.
	GRAB	Gửi lệnh thu tay gấp.
	MOVETOXYZ	Khi không bám theo vật thể, ta có thể gửi tọa độ cần tiếp cận xuống.
	STOP	Gửi lệnh dừng hệ thống robot tức thời.
	CONNECT	Kết nối giao USB USART
4	Input matrix	Nhập ma trận chuyển đổi tọa độ hệ trục của Kinect sang hệ trục gốc tay robot.

Bảng 4.4 Bảng giải thích chức năng nút nhấn trên giao diện.

Chương trình xử lí ảnh nhận diện, bám theo vật để trả về tọa độ không gian ba chiều của vật được xây dựng như sơ đồ trạng thái sau (đường mũi tên liền nét thể hiện tín hiệu, đường mũi tên đứt nét thể hiện đường tín hiệu kích hoạt sự kiện từ các nút nháy button trên giao diện):



Hình 4.8 Sơ đồ trạng thái chương trình điều khiển và xử lí ảnh trên máy tính.

CHƯƠNG 5: KẾT QUẢ - KẾT LUẬN – HƯỚNG PHÁT TRIỂN:

5.1. NGHIỆM THU KẾT QUẢ:

5.1.1. ĐIỀU KHIỂN CÁNH TAY ROBOT ĐỘC LẬP

Sau quá trình nghiên cứu, gia công mô hình cánh tay robot và xây dựng bộ điều khiển cánh tay robot, nhóm đã hoàn thiện mô hình cánh tay robot chắc chắn, điều khiển các góc điều khiển trên từng khớp đáp ứng nhanh, sai số trên từng khớp là nhỏ hơn 1 độ theo giá trị đọc của Encoder.

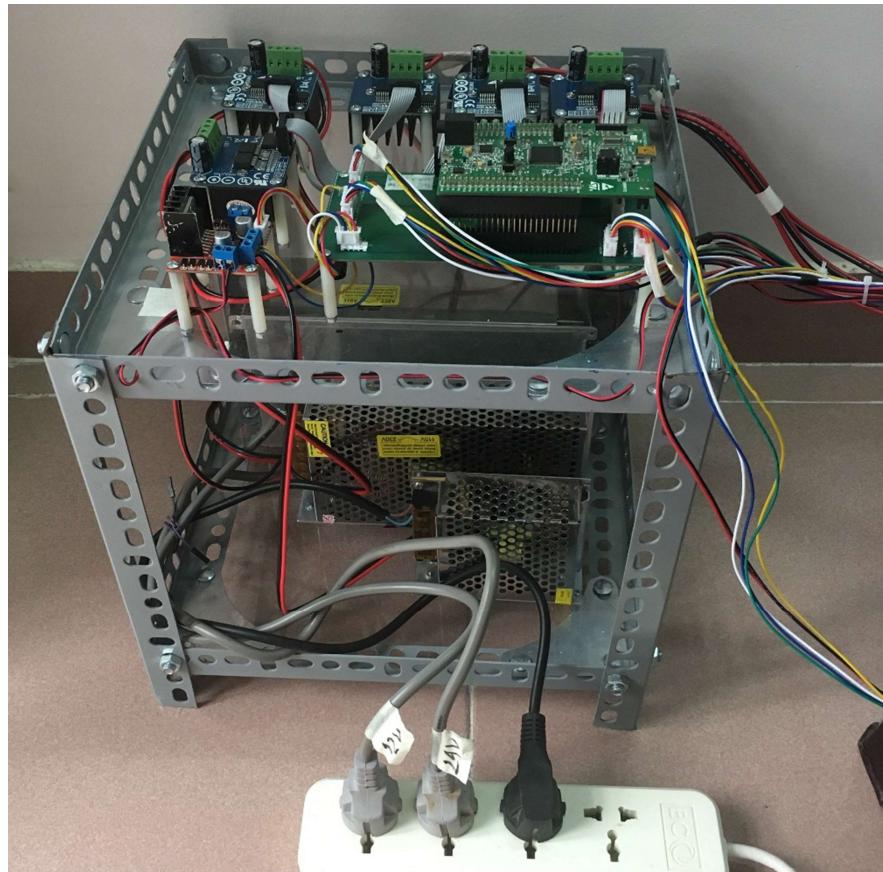
Mô hình cánh tay được hoàn thiện gia công hoàn toàn bằng sắt, ngoại trừ chân đế làm bằng nhôm, các module của bộ điều khiển đều chạy ổn định, có hiện không quá nhiệt hay nhiễu gây mất ổn định hệ thống.

Hình ảnh mô hình cánh tay robot:



Hình 5.1 Mô hình hoàn thiện cánh tay robot 5 DOF SCORBOT.

Hình ảnh bộ điều khiển trung tâm:



Hình 5.2 Bộ điều khiển trung tâm cánh tay robot 5 DOF SCORBOT.

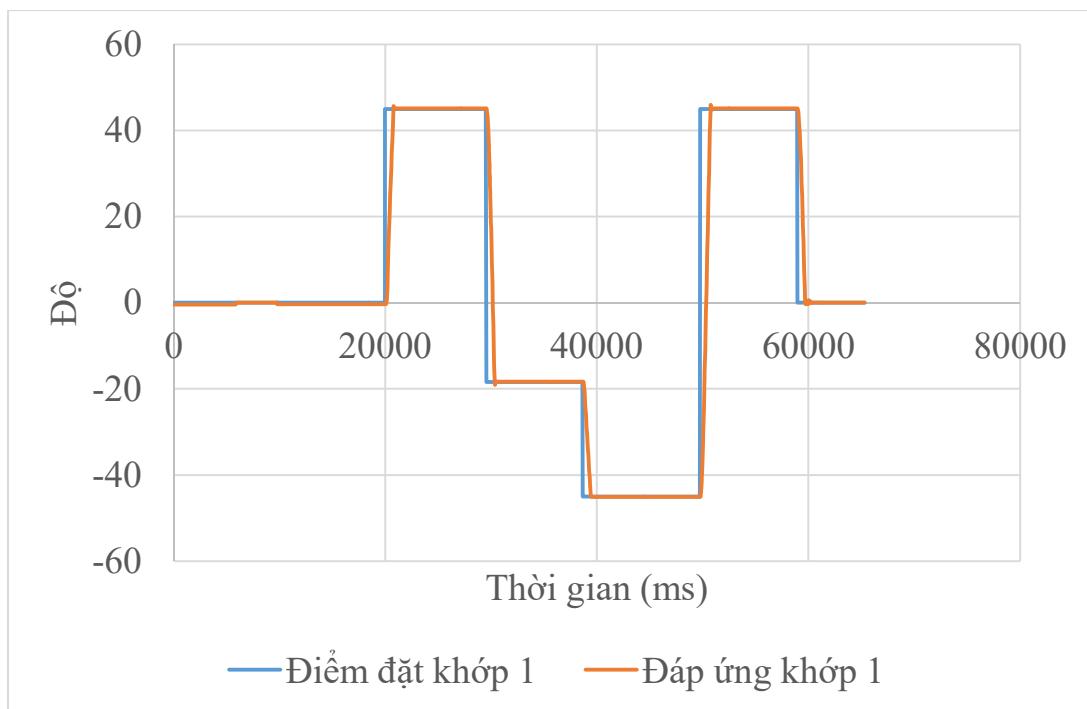
Bộ điều khiển trung tâm bao gồm 2 tầng:

- Tầng trên chứa adapter nối kit STM32F407VGT Discovery với các header nối các dây tín hiệu Encoder từ robot, các tín hiệu điều khiển từ vi điều khiển ra các cầu H là các khối công suất. Các khối công suất được nối với nguồn và các dây nguồn của các động cơ.
- Tầng dưới chứa 3 bộ nguồn chính cấp cho khối điều khiển và khối công suất đã được nêu trong mục 2.3.2.

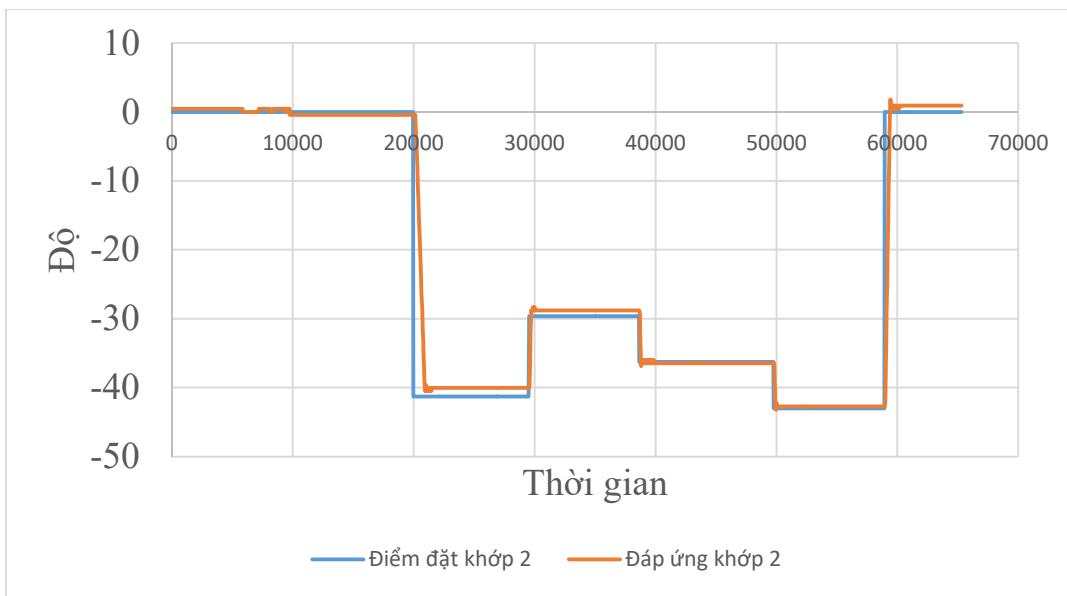
Trong quá trình điều khiển, mô hình thực hiện tốt các góc quay trên từng khớp, khi điều khiển đồng thời, mô hình có xảy ra rung lắc nhẹ do quán tính, đã được giảm thiểu tối đa bằng thuật toán điều tốc, qui hoạch quỹ đạo nhưng chưa khắc phục triệt để.

Tuy nhiên do bộ điều khiển STR của từng động cơ có các giá trị ước lượng chưa lưu lại được sau mỗi lần tắt chương trình, vì mỗi lần bắt đầu khởi động bộ điều khiển từ đầu, cần cho chạy độc lập từng khớp qua chương trình debug STM32 Studio để bộ điều khiển ước lược chính xác các thông số trước khi đưa vào điều khiển tích hợp với thuật toán xử lý ảnh.

- Đáp ứng góc điều khiển trên từng khớp trong quá trình điều khiển số liệu đo trong trường hợp chạy chương trình thực tế:



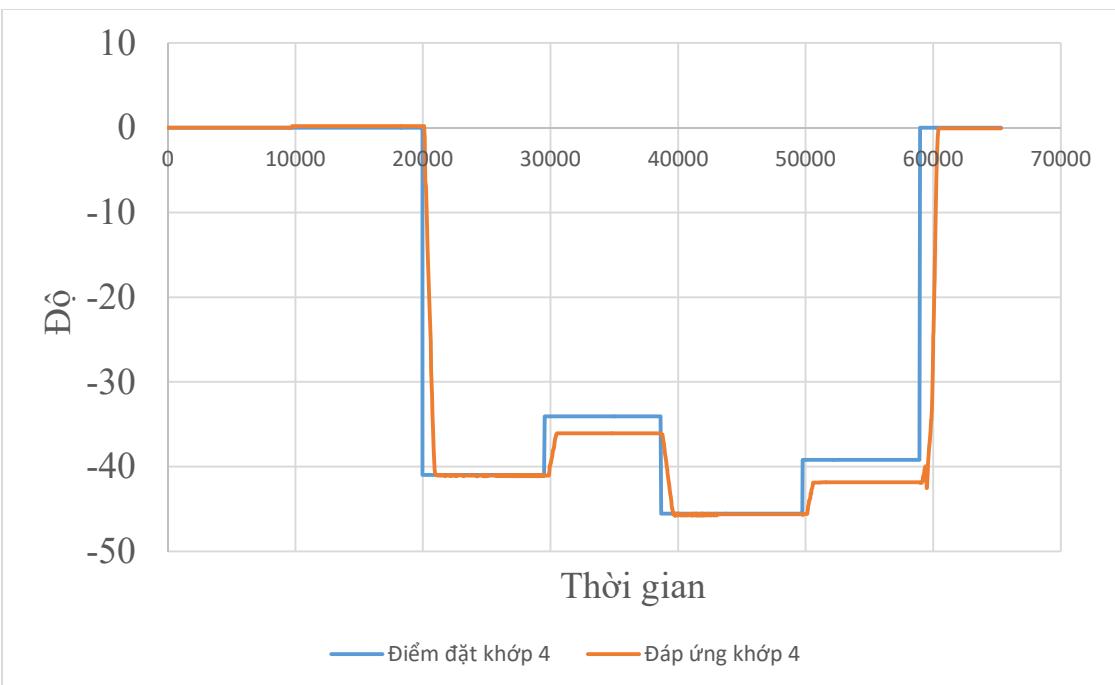
Hình 5.3 Đồ thị đáp ứng khớp BASE



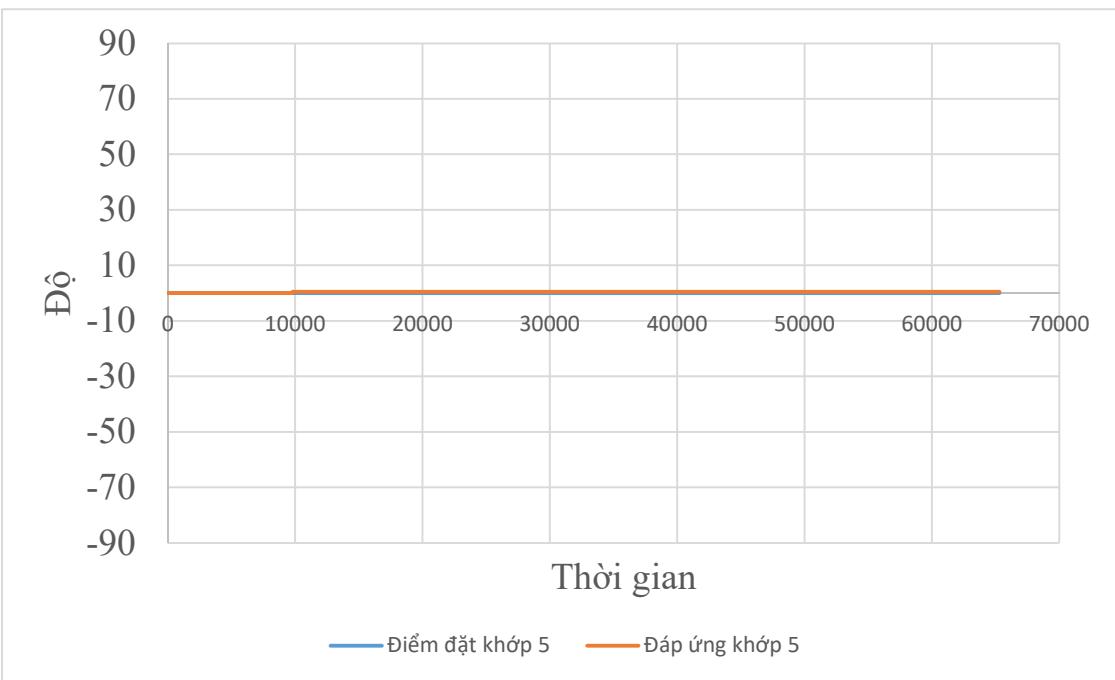
Hình 5.4 Đồ thị đáp ứng trực SHOULDER



Hình 5.5 Đồ thị đáp ứng trực ELBOW



Hình 5.6 Đồ thị đáp ứng khớp PITCH

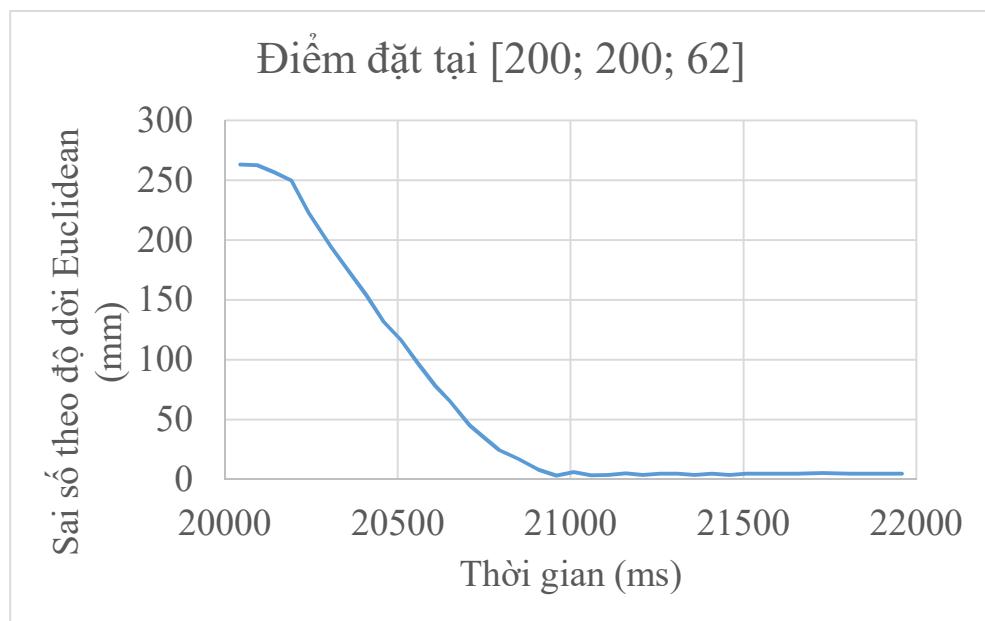


Hình 5.7 Đồ thị đáp ứng khớp ROLL

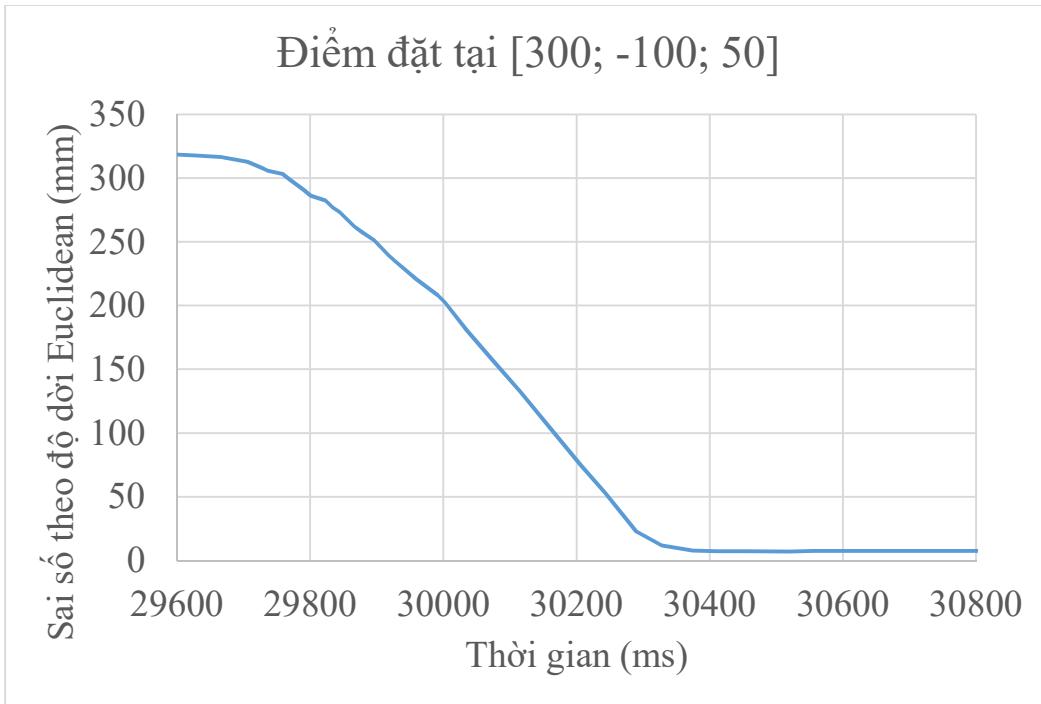
Nhận xét: Sai số trên các khớp đều được duy trì ở mức tối đa +/- 2 độ khi tới điểm đặt. Đáp ứng tốt cho mục tiêu điều khiển vị trí.

Để đánh giá chất lượng điều khiển vị trí robot thực hiện được so với vị trí điểm đặt, nhóm dùng sai số độ dời (hay còn gọi là sai số theo khoảng cách Euclidean). Sau đây là các bảng thông số giám sát tại các điểm đặt khác nhau khi nhóm lần lượt đi qua các điểm [200; 200; 62], [300; -100; 50], [200; -200; 40], [200; 200; 70], điểm bắt đầu tại [369; 0; 62] tức điểm HOME của robot:

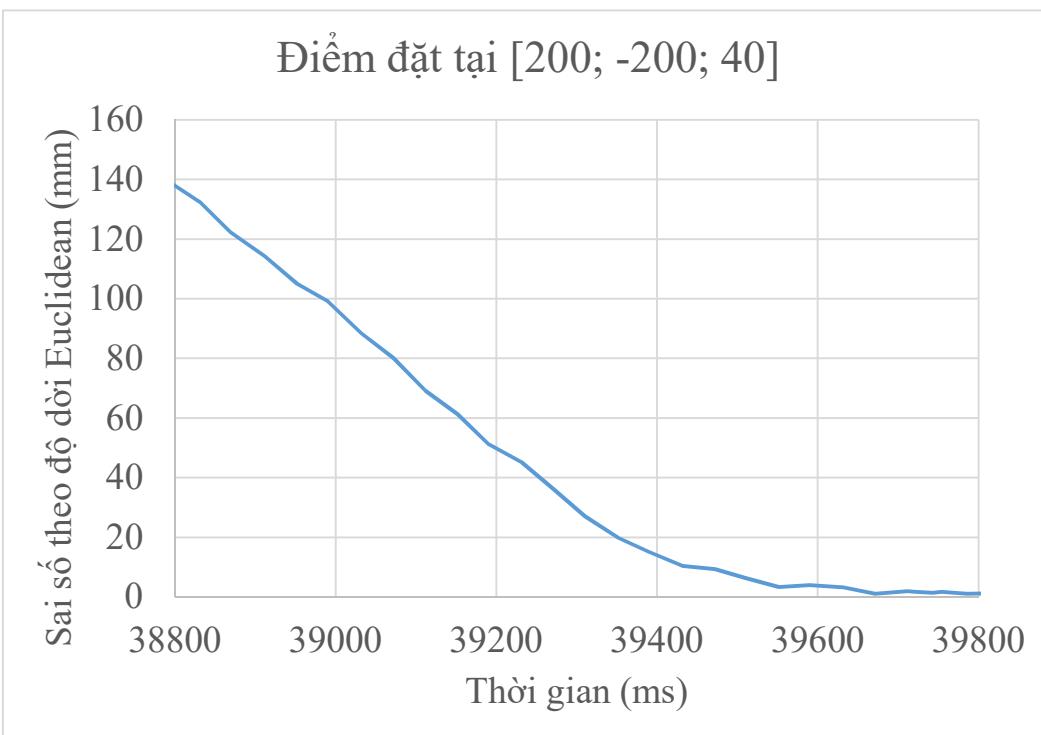
:



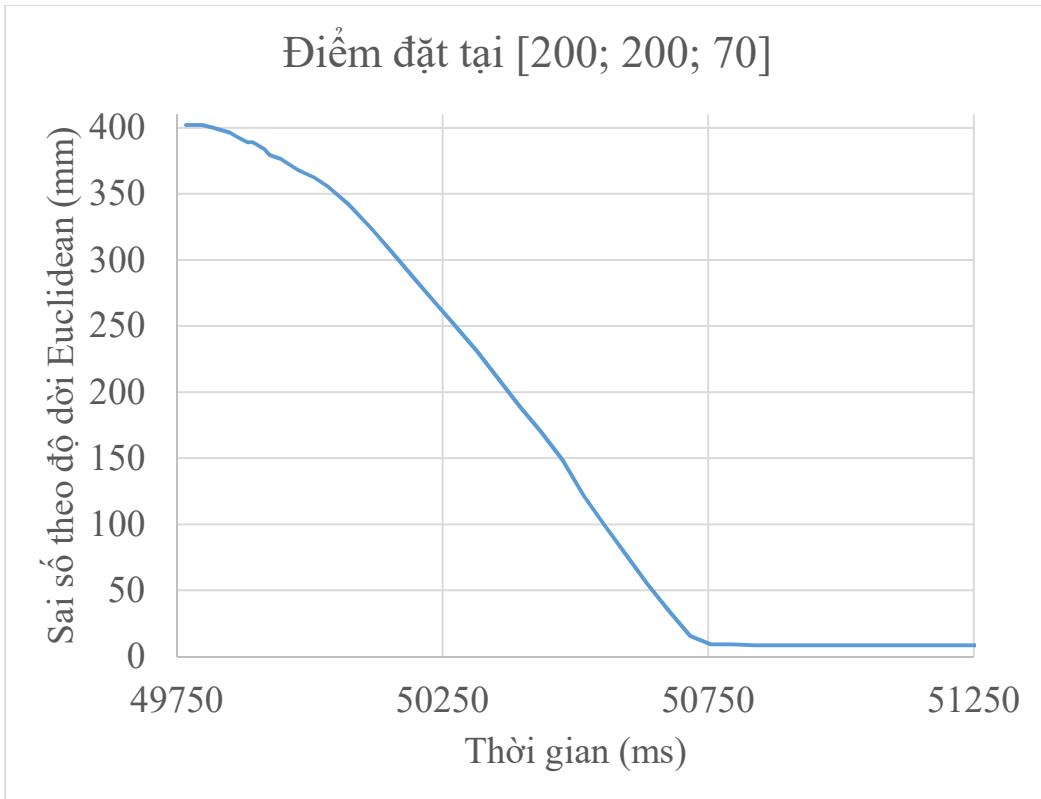
Hình 5.8 Đồ thị giám sát sai số độ dời tiến tới điểm 1



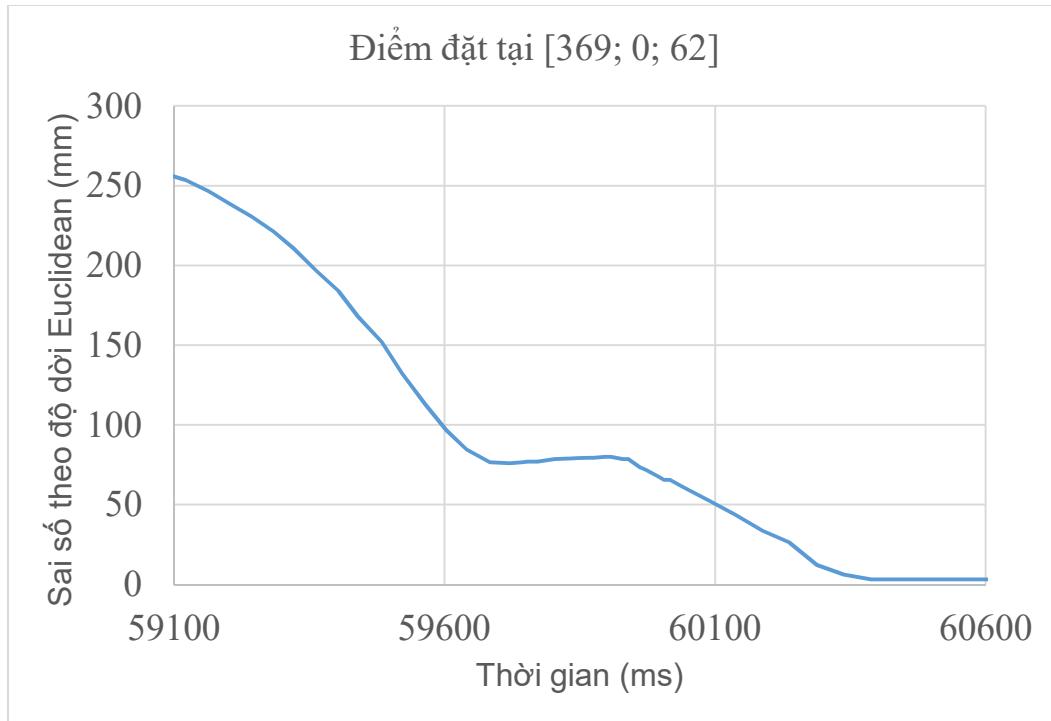
Hình 5.9 Đồ thị giám sát sai số độ dời tại điểm 2



Hình 5.10 Đồ thị giám sát sai số độ dời tại điểm 3



Hình 5.11 Đồ thị giám sát sai số độ dời tại điểm 4



Hình 5.12 Đồ thị giám sát sai số độ dời tại điểm 5

Nhận xét: Khi tiến tới tới điểm đặt được yêu cầu, sai số độ dời tối đa ở mức chỉ khoảng 5mm, đạt độ chính xác tốt để có thể hoạt động tích hợp với chương trình xử lí ảnh.

Nhóm thực hiện khả năng thực hiện thao tác lặp lại của mô hình robot bằng cách cho robot chạy luân phiên đến 2 điểm bất kĩ trong 20 lần, nhóm chọn 2 điểm có tọa độ ở [150; -300; 20] và [250; 200; 70], giá trị thực tế được tính thông qua bài toán động học thuận, ta có 2 bảng số liệu sau:

Tọa độ x	Tọa độ y	Tọa độ z	Sai số độ dời
248.90	199.70	69.12	1.44
251.60	201.09	70.31	1.96
249.07	200.61	73.65	2.21
250.58	198.73	73.65	2.27
250.91	202.09	69.42	2.35
250.37	200.11	72.55	2.58
250.91	202.09	69.42	2.67
250.58	198.73	73.65	2.89
249.07	200.61	73.65	3.01

250.27	203.91	69.15	3.42
252.74	202.00	68.00	3.94
251.66	199.59	71.38	3.95
251.98	202.95	68.00	4.08
250.84	202.03	70.24	4.11
251.60	203.43	68.00	4.28
249.11	205.32	69.15	4.47
251.54	203.38	67.89	4.54
250.04	206.09	68.25	4.56
248.99	204.44	69.42	4.59
251.81	205.17	69.20	4.73

Bảng 5.1 Tọa độ thực tế và sai số Euclidean khi điểm đặt tại [250; 200; 70]

Tọa độ x	Tọa độ y	Tọa độ z	Sai số độ dời
152.80	-302.98	16.64	5.29
151.24	-301.31	17.50	3.08
147.05	-298.63	19.15	3.36
149.21	-302.28	19.63	2.45
151.31	-302.67	19.07	3.12
151.09	-300.09	19.40	1.25
148.68	-300.59	21.43	2.03
152.11	-301.42	22.80	3.78
150.65	-302.83	21.52	3.28
152.55	-302.74	21.49	4.02
151.77	-302.18	21.90	3.39
151.81	-297.27	17.58	4.07
148.89	-302.14	22.93	3.80
147.37	-301.36	17.85	3.66
149.28	-301.39	21.84	2.42
151.95	-297.39	21.00	3.41
147.78	-298.83	20.07	2.51
149.68	-299.33	19.67	0.81
149.84	-301.65	18.80	2.04
149.42	-302.70	17.21	3.93

Bảng 5.2 Tọa độ thực tế và sai số Euclidean khi điểm đặt tại [150; -300; 20]

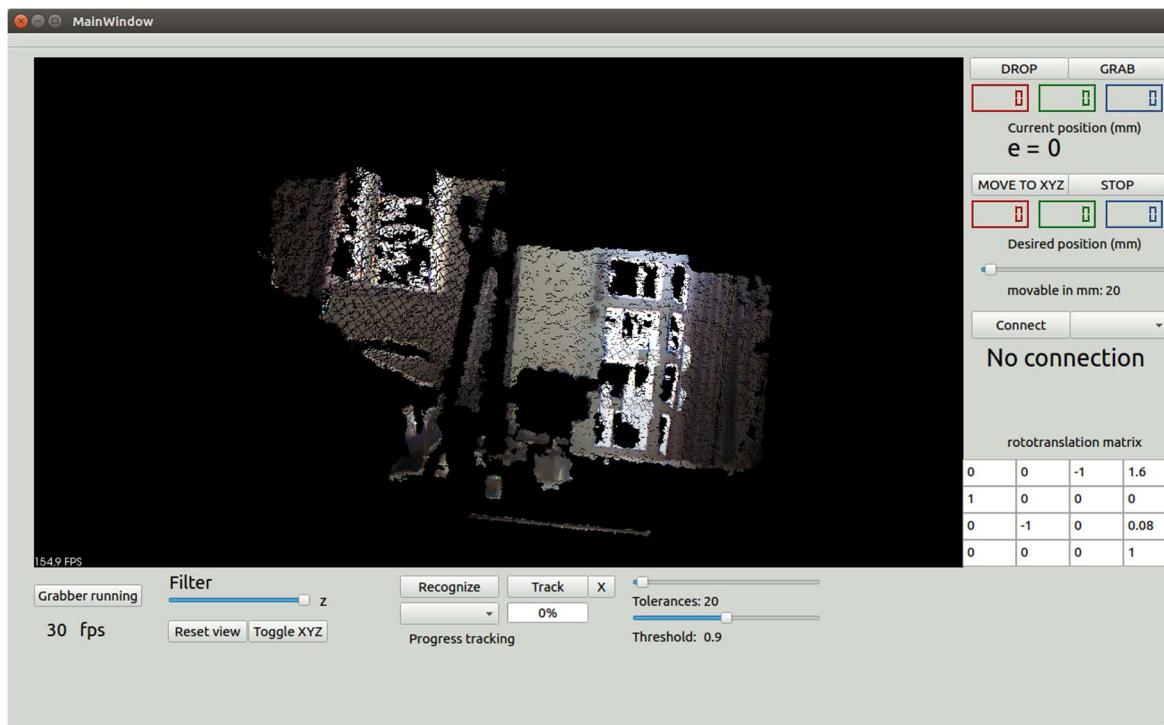
Nhận xét: Khi tiến tới tới điểm đặt được yêu cầu, và lặp lại tại 2 vị trí 20 lần, sai số độ dời tối đa ở mức chỉ khoảng 5mm, tuy nhiên đôi khi vẫn có trường hợp sai

số nằm trong khoảng 1cm đến 2cm, nhưng ít khi xảy ra, vì vậy nhóm tự đánh giá robot đạt độ chính xác tốt để có thể hoạt động tốt với các thao tác cần sự lặp lại nhiều lần

5.1.2. CHƯƠNG TRÌNH XỬ LÍ ẢNH

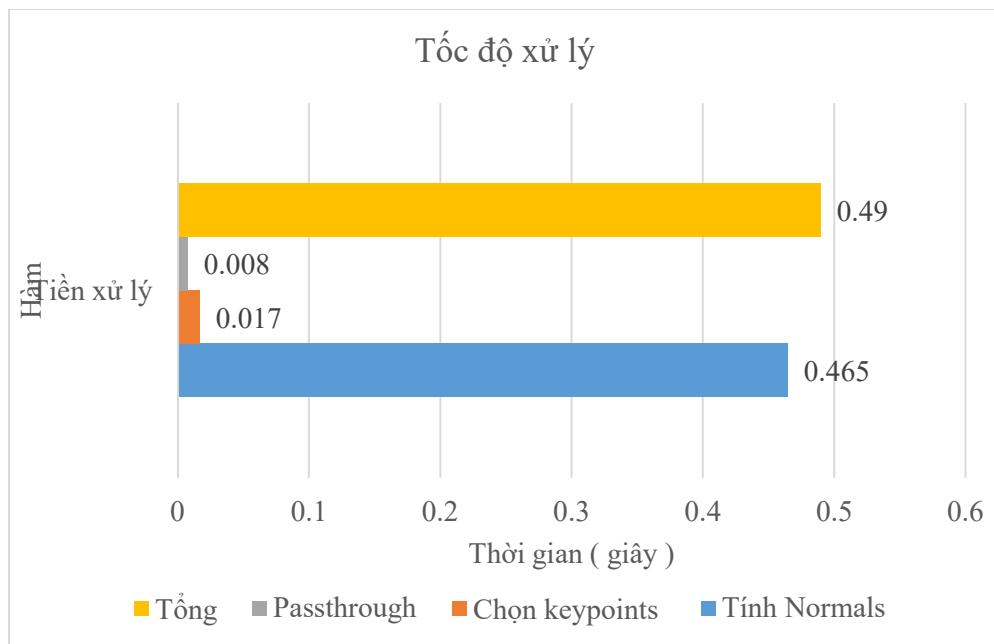
Chương trình nhận dạng và bám theo vật thể 3 chiều xử lý tốt, đáp ứng khá nhanh, chính xác, phù hợp với mục tiêu đề ra ban đầu của luận văn.

Giao diện trực quan, dễ quan sát và thao tác:

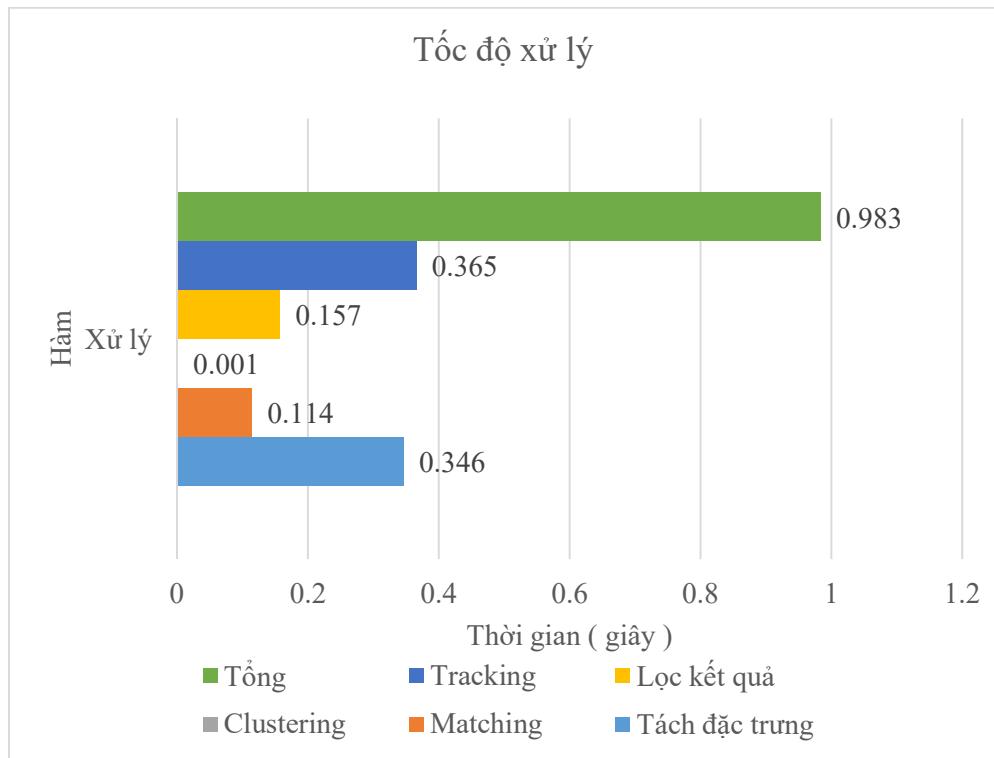


Hình 5.13 Giao diện người dùng trên máy tính.

Thu thập dữ liệu thời gian xử lý các thuật toán về xử lý ảnh, các kết quả tối ưu về tốc độ xử lý ở sản phẩm của nhóm như sau:



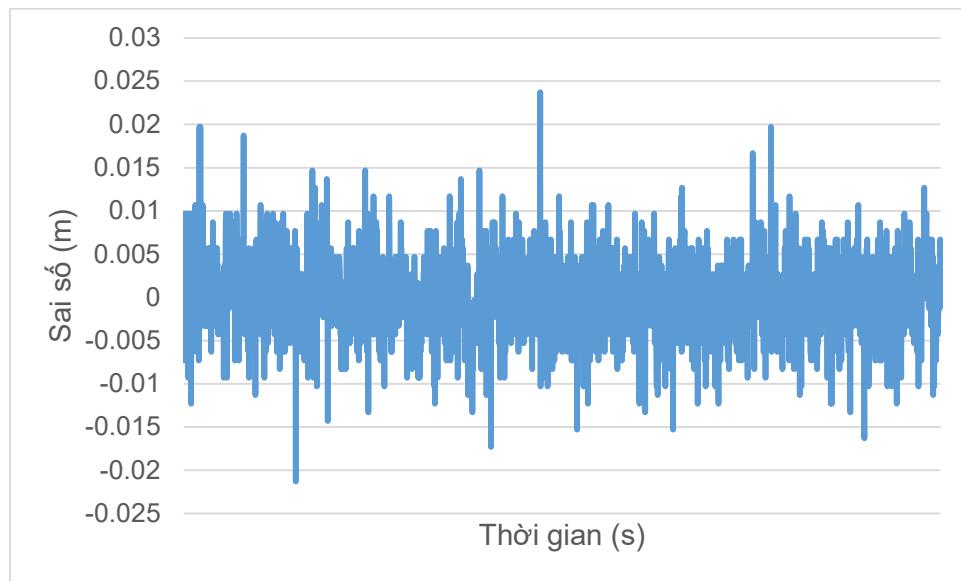
Bảng 5.3 Thời gian xử lý các hàm tiền xử lý dữ liệu hình ảnh.



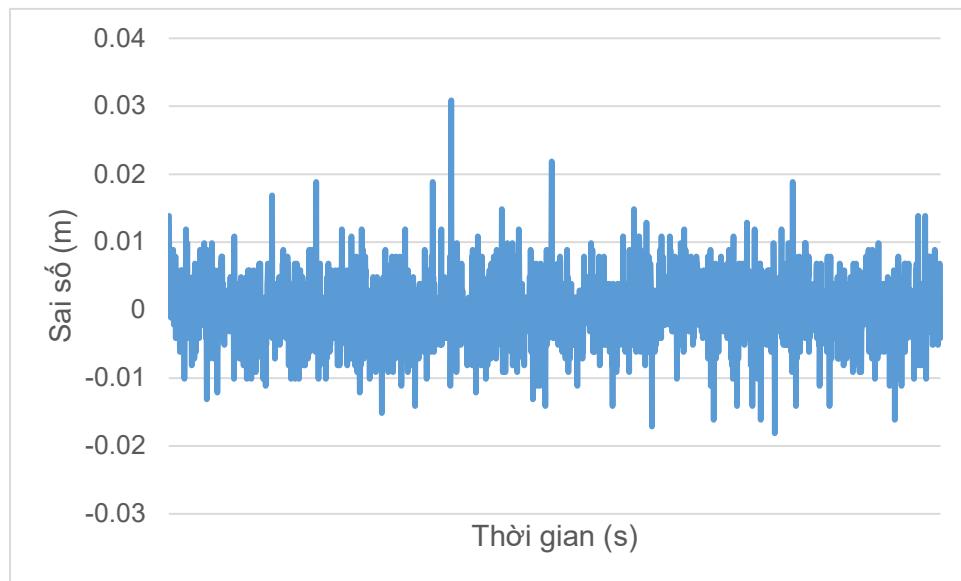
Bảng 5.4 Thời gian xử lý các hàm xử lý nhận diện và theo dõi hình ảnh.

Để kiểm tra và tự đánh giá độ ổn định của thuật toán xác định tọa độ vật thể trong không gian, nhóm thực hiện để Kinect và chương trình xử lí ảnh xác định tọa

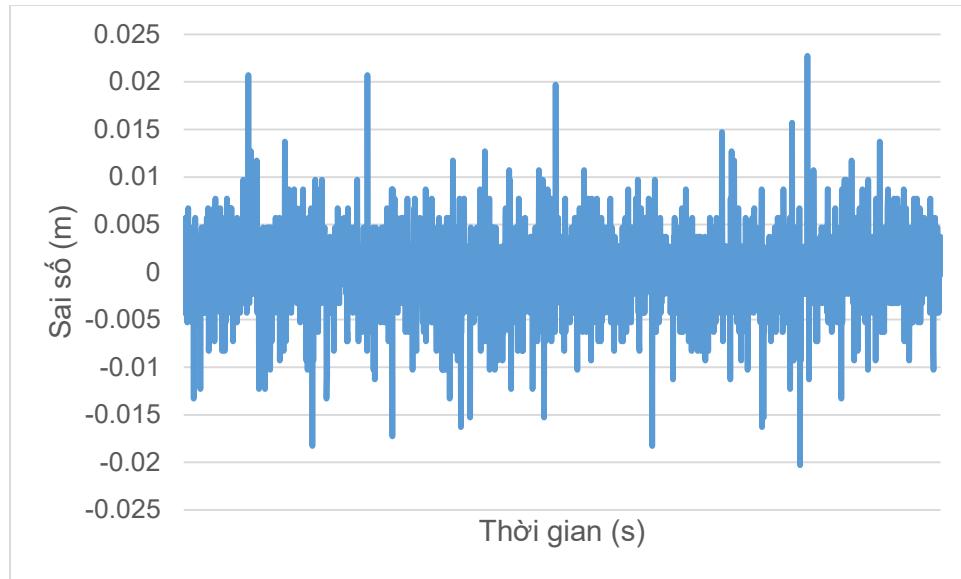
độ vật đang đứng yên trong 80s, sau đó xử lí số liệu được các độ thi sai số tọa độ theo các trục như sau:



Hình 5.14 Đồ thị biến thiên giá trị đo trục x theo thời gian.



Hình 5.15 Đồ thị biến thiên giá trị đo trục y theo thời gian.



Hình 5.16 Đồ thị biến thiên giá trị đo trục z theo thời gian.

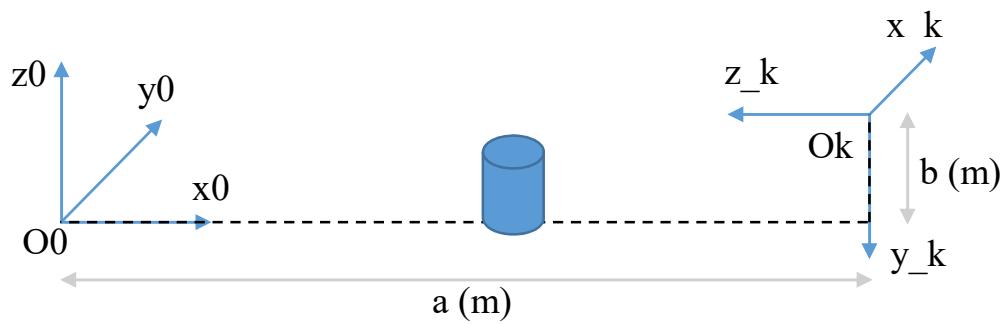
Từ các số liệu trên, nhóm tính toán được độ lệch chuẩn theo độ dời Euclidean của thuật toán theo dõi và xác định tọa độ vị trí vật của hệ thống camera Kinect là 7.567 mm.

Nhận xét:

Đây là giá trị sẽ ảnh hưởng đến chất lượng hệ thống tích hợp. Cơ cấu robot khi di chuyển cũng có một khoảng sai số độ dời Euclidean nhất định khoảng 5mm, cộng thêm phần sai số độ dời từ camera Kinect sẽ tạo thành sai số điểm đặt với vị trí thực đạt được của robot.

5.1.3. HỆ THỐNG TAY ROBOT KẾT HỢP XỬ LÝ ẢNH:

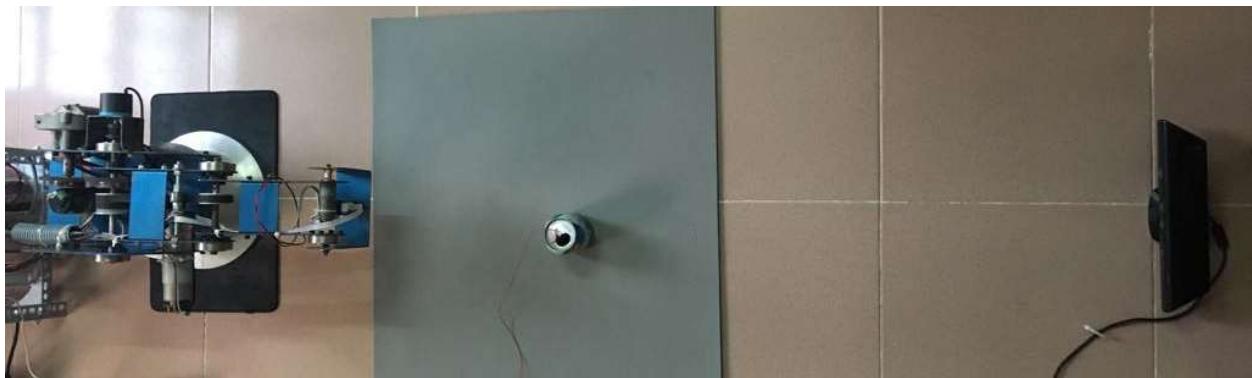
Như đã đề cập ở các phần trên , nhóm thực hiện sơ đồ bố trí hệ thống vị trí của robot và Kinect như sau:



Hình 5.17 Sơ đồ bố trí hệ thống.

Với sơ đồ bố trí trên, nhóm xác định các thông số $a = 1.6$ m, $b = 0.08$ m.

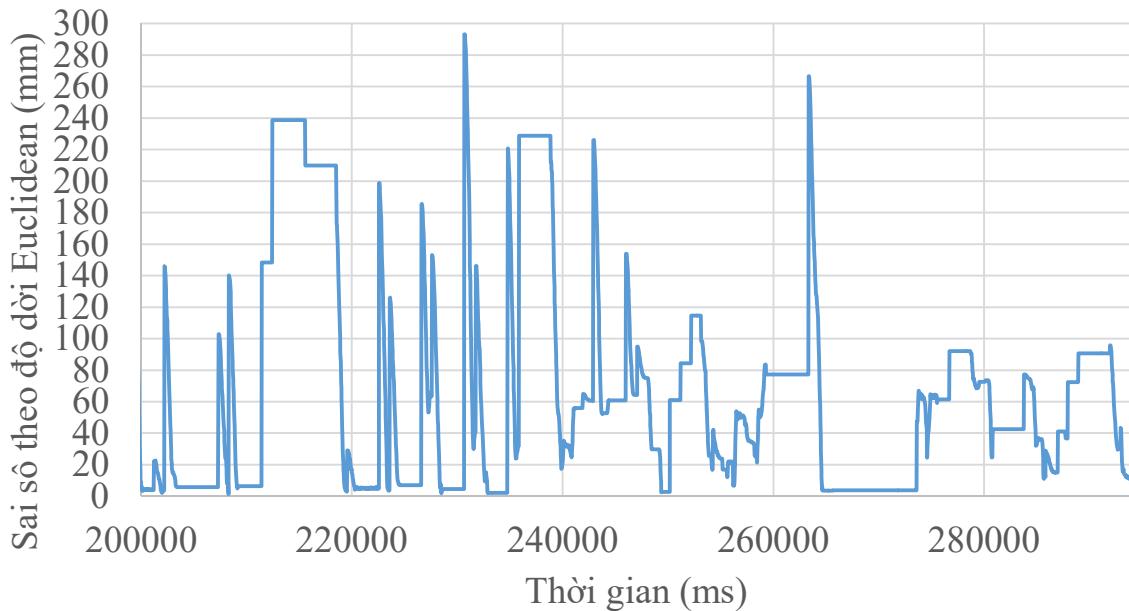
Hình ảnh sơ đồ bố trí thực tế của hệ thống:



Hình 5.18 Sơ đồ bố trí thực tế của hệ thống.

Các kết quả thu nhận được khi thực hiện ra lệnh robot bám theo vật thể theo thời gian thực, đánh giá các kết quả sai sót điểm đạt được của robot so với vị trí được gửi trực tiếp từ máy tính.

Sai số độ dời khi chạy chương trình tích hợp xử lí ảnh



Hình 5.19 Sai số độ dời khi vận hành tích hợp xử lí ảnh thời gian thực

Lưu ý nhóm khi thực hiện tích hợp hệ thống đã tháo cơ cấu tay gấp do quá trình chuyển động của robot chỉ dựa trên điểm đến, chưa xây dựng quỹ đạo để tiếp cận, vì vậy nếu lắp cả tay gấp robot dễ va chạm với vật cần gấp, khó thể hiện độ chính xác của vị trí robot so với vật thể cần bám theo.

Nhận xét:

Chương trình hoạt động tích hợp ổn định, quá trình truyền nhận lệnh từ máy tính đến bộ điều khiển và gửi dữ liệu vị trí từ bộ điều khiển về máy tính không xảy ra lỗi, thông tin truyền nhận chính xác.

Dựa theo đồ thị hình 5.15, ta có thể thấy được những thời điểm sai số Euclidean nhỏ hơn 10mm là khi robot đã đạt được vị trí của vật thể. Như vậy về cơ bản, hệ thống vận hành đã hoạt động tốt, robot đã có thể bám theo vật thể, khi đến vật thể sai số nhỏ hơn 10mm (theo hình 5.15).

Tuy nhiên đáp ứng bám theo vật thể chưa cao, khi vật bị thay đổi vị trí quá nhanh mà robot chưa di chuyển tới, các cơ cấu khớp điều khiển sẽ bị rung lắc do phải điều chỉnh góc nhỏ liên tục.

5.2. TƯ NHẬN XÉT VÀ ĐÁNH GIÁ:

5.2.1. CÁC MỤC TIÊU ĐÃ THỰC HIỆN ĐƯỢC:

- Nghiên cứu một số kiến thức cơ bản về cơ khí, mô hình, các cơ cấu truyền động, dựng mô hình bằng Solidworks và nắm các đặc tính ưu nhược điểm của mô hình được xây dựng. Hoàn thiện mô hình cơ khí cánh tay robot vững chắc, đáp ứng các tiêu chí để điều khiển đã đề ra. Nhóm triển khai thêm các cơ cấu cảng đai truyền động để tối ưu hơn về chất lượng điều khiển.
- Nghiên cứu board phát triển STM32F407VGT Discovery bằng ngôn ngữ C, tổ chức qua chương trình KeilC, sử dụng thư viện StdPeriph để triển khai các câu lệnh phần cứng được trực quan, không quá sâu vào câu lệnh thanh ghi nhưng cũng không lạm dụng quá đà vào các thư viện cấp cao hơn như HAL.
- Hoàn thiện bộ điều khiển cánh tay robot về cả phần cứng và phần mềm, các bộ điều khiển được tối ưu về thuật toán để điều khiển được tốt nhất mà nhóm có thể thực hiện được, tổ chức chương trình trực quan, dễ đọc và phát triển mở rộng.
- Kết quả điều khiển từng khớp của robot chính xác và ổn định theo quỹ đạo đã được qui hoạch.
- Xây dựng giao diện bằng C++ bằng phần mềm Qt 5.9 trên nền Ubuntu Linux, kết hợp thuật toán nhận diện vật, bám theo vật và trả về các tọa độ 3 chiều theo thời gian bằng các thư viện cho Kinect hiệu quả, thực tế chương trình hoạt động ổn định.
- Xây dựng protocol giao tiếp giữa chương trình xử lý ảnh trên máy tính và bộ điều khiển cánh tay robot ổn định, có tình kế thừa và có thể mở rộng thêm các chức năng sau này.

5.2.2. NHỮNG KHUYẾT ĐIỂM CẦN KHẮC PHỤC:

- Mô hình cơ khí còn một số chi tiết truyền động chưa đạt được kì vọng mong muôn:
 - 2 bánh răng truyền động cho khớp Shoulder có độ rõ nhất định, khi không cấp áp nhưng bị tác động vẫn có thể tự lệch +/- 2 độ, đặc biệt

thiếu ổn định ở vị trí khớp Shoulder hướng thẳng đứng, nhóm khắc phục bằng phần mềm không cho khớp quay lên vị trí đó.

- Các ổ bị, bánh răng cố định vào các trục sau thời gian hoạt động có thể bị rò rỉ so với trục, cần được siết cố định lại thường xuyên.
 - Khối lượng phân bổ trên các khớp không đều, các cơ cấu truyền động chưa thực sự tốt nên trong quá trình vận hành cả hệ thống vẫn bị rung nhẹ do quán tính, thuật toán qui hoạch quỹ đạo chỉ khắc phục 1 phần.
 - Cơ cấu tay gấp đôi khi bị đứng cấp áp không chạy, cần kích lại bằng tay. Khi mô phỏng việc gấp thả vật thực tế khi có tay gấp chưa thực hiện được do quá trình chuyển động của robot chỉ dựa trên điểm đến, chưa xây dựng quỹ đạo để tiếp cận.
- Thuật toán điều khiển qui hoạch quỹ đạo mới chỉ giới hạn tốc độ trên của khớp nhưng chưa giới hạn tốc độ dưới, vì khi các quỹ đạo quá chậm (ở đây là các góc nhỏ), các khớp chạy rồi dừng gây rung lắc nhẹ, cần có thuật toán xử lý vấn đề này.
- Tuy nhóm đã áp dụng thành công thuật toán điều khiển vị trí từng khớp STR MRAS, nhưng trong vùng sai số nhỏ bé hơn 2 độ vẫn khó điều khiển, vẫn có trường hợp không điều khiển được chính xác điểm đặt. Động cơ khớp vai Shoulder có sự thay đổi mô men tải quá lớn, dù đã áp dụng thuật toán STR MRAS nhưng khi điều khiển thay đổi góc nhỏ hơn 20 độ sai số lớn, sai số từ 2 đến 5 độ, khi thay đổi góc lớn hơn 20 độ thì đáp ứng tốt.
- Đồng thời do bộ điều khiển trong quá trình thích nghi rất nhạy với nhiễu, giải thuật cần phát triển bộ giám sát lỗi để cài đặt lại thông số khi có lỗi xảy ra.
- Thuật toán xử lý ảnh cần xác định thêm hướng tiếp cận vật thể để độ linh hoạt được cao hơn, trong luận văn nhóm chỉ cho tiếp cận vật từ trên xuống. Sử dụng thêm các bộ lọc để giá trị đầu ra tọa độ của vật ổn định, hiện nay tọa độ các trục vẫn sai số lớn nhất +/- 1cm.

5.3. HƯỚNG PHÁT TRIỂN ĐỀ TÀI LUẬN VĂN:

5.3.1. KHẢ NĂNG ÚNG DỤNG:

- Mô hình tay robot và chương trình xử lí ảnh thông qua Kinect có thể sử dụng tiếp tục trong mục đích nghiên cứu và phát triển, đặc biệt là phát triển các thuật toán điều khiển để chất lượng điều khiển ngày càng được cải thiện tốt hơn.
- Sử dụng mô hình như một đối tượng thực thi cho các thuật toán xử lí ảnh, trí tuệ nhân tạo khác.

5.3.2. GỢI Ý PHƯƠNG HƯỚNG PHÁT TRIỂN:

- Tối ưu các thuật toán qui hoạch quỹ đạo để hệ thống vận hành mượt mà và ổn định hơn. Sử dụng bộ điều khiển 2 vòng cả vận tốc và vị trí để đạt chất lượng điều khiển trọn tru hơn, khắc phục triệt để dừng đột ngột.
- Có thể cân nhắc thay đổi sử dụng động cơ Stepper hoặc BLDC, cấp điện áp điều khiển có độ đáp ứng tốt hơn so với động cơ DC ở vị trí khớp vai Shoulder.
- Tích hợp thêm bộ nhớ để lưu các giá trị trạng thái của bộ ước lượng trạng thái cũng như thêm các công tắc hành trình để giới hạn các góc quay của khớp, dễ dàng hơn trong việc calib hệ thống, giảm sai số hệ thống bị cộng dồn. Việc lưu được các giá trị này giúp sau mỗi lần khởi động, thuật toán không phải ước lượng lại từ đầu gây rung lắc hoặc lỗi ước lượng sai trong quá trình thích ứng ban đầu khi vận hành.
- Tối ưu tốc độ xử lí các thuật toán để có thể giảm thời gian lấy mẫu, ngắt chạy chương trình, khi đó chất lượng điều khiển sẽ tốt hơn. Sử dụng RTOS để tối ưu thực thi tác vụ của bộ điều khiển cánh tay robot. Đáp ứng với khoảng thay đổi nhỏ trên từng khớp.
- Phát triển thuật toán hoạch định quỹ đạo điểm cuối đi theo đường cao, đường thẳng, quỹ đạo được định trước, để làm được việc này cần tối ưu thuật toán điều khiển trên từng khớp để điều khiển được sự thay đổi góc nhỏ.
- Tối ưu thuật toán xác định tọa độ vật thông qua các bộ lọc (ví dụ như Kalman hay Particle).

TÀI LIỆU THAM KHẢO

- [1] T. K. Nguyen, "Design a Self-Tuning-Regulator for DC Motor's Velocity and Position Control," Department of Automation and Control Engineering - VNU, HCMUT, Ho Chi Minh, 2013.
- [2] S. Salti, F. Tombari, and L. Di Stefano, "SHOT: Unique Signatures of Histograms for Surface and Texture Description," *"Computer Vision and Image Understanding"*, vol. 125, pp. 251-264, 2014.
- [3] Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., "Surface reconstruction from unorganized points," in *Proc. of the Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, New York, NY, USA, 1992.
- [4] Mitra, N.J., Nguyen, A., Guibas, L., "Estimating surface normals in noisy point cloud data," *International Journal of Computational Geometry and Applications*, no. 14(4–5), p. 261–276, 2004.
- [5] Lowe, D.G., "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision (IJCV)*, vol. 60, p. 91–110, 2004.
- [6] Aitor Aldoma, Federico Tombari, Luigi Di Stefano, Markus Vincze, "A Global Hypothesis Verification Framework for 3D Object Recognition in Clutter," *IEEE PAMI*, vol. 38, no. 7, pp. 1383 - 1396, 2015 .
- [7] B. Taati and M. Greenspan, "Local shape descriptor selection for," *CVIU*, 2011.
- [8] Mahalanobis distance, "Wiki," [Online]. Available: https://en.wikipedia.org/wiki/Mahalanobis_distance.
- [9] A. Mian, M. Bennamoun, and R. Owens, "3d model-based object recognition and segmentation in cluttered scenes," *TPAMI*, 2006.
- [10] C. Papazov and D. Burschka, "An efficient ransac for 3d object recognition in noisy and occluded scenes," *ACCV*, 2010.

- [11] A. Aldoma, F. Tombari, R. B. Rusu, and M. Vincze, "Our-cvfh: Oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6dof pose estimation," *Joint DAGM-OAGM Pattern Recognition Symposium*, 2012.
- [12] PointCloudLibrary Download, "PCL," [Online]. Available:
<http://pointclouds.org/downloads/>.
- [13] OpenNI repository, "Github," [Online]. Available:
<https://github.com/OpenNI/OpenNI>.
- [14] ROS installation guide, "ROS," [Online]. Available:
<http://wiki.ros.org/kinetic/Installation/Ubuntu>.
- [15] Johnson, A., Hebert, M., "Using spin images for efficient object recognition in cluttered 3D," *PAMI 21*, pp. 433-449, 1999).
- [16] Sebastian THRUN, Wolfram BURGARD, Dieter FOX, "The Particle filter," in *Probabilistic robotics*, The MIT Press, 2005, pp. 77-89.