ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

TRẦN GIA TUẤN

ĐỒ ÁN TỐT NGHIỆP

THIẾT KẾ VÀ HIỆN THỰC HỆ THỐNG ĐIỀU KHIỂN CHO CÁNH TAY ROBOT 5 BẬC TỰ DO

DESIGN AND IMPLEMENT A CONTROL SYSTEM FOR A 5-DOF ROBOTIC MANIPULATOR

CỬ NHÂN NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

TP. HỒ CHÍ MINH, 2024

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG

TRẦN GIA TUẤN – 2012357

ĐỒ ÁN TỐT NGHIỆP

THIẾT KẾ VÀ HIỆN THỰC HỆ THỐNG ĐIỀU KHIỂN CHO CÁNH TAY ROBOT 5 BẬC TỰ DO

DESIGN AND IMPLEMENT A CONTROL SYSTEM FOR A 5-DOF ROBOTIC MANIPULATOR

CỬ NHÂN NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

GIẢNG VIÊN HƯỚNG DẪN

GS. TS. Hồ Phạm Huy Ánh

TP. HỒ CHÍ MINH, 2024

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI
TRƯỜNG ĐẠI HỌC BÁCH KHOA –ĐHQG -HCM

Cán bộ hướng dẫn Khóa luận tốt nghiệp : .....
(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Cán bộ chấm nhận xét 1 : .....
(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Cán bộ chấm nhận xét 2 : .....
(Ghi rõ họ, tên, học hàm, học vị và chữ ký)

Khóa luận tốt nghiệp được bảo vệ tại Trường Đại học Bách Khoa, ĐHQG Tp.HCM ngày . . . . . tháng . . . . năm . . . . .

Thành phần Hội đồng đánh giá khoá luận tốt nghiệp gồm:
(Ghi rõ họ, tên, học hàm, học vị của Hội đồng chấm bảo vệ khóa luận tốt nghiệp)
1. .....
2. .....
3. .....
4. .....
5. .....

Xác nhận của Chủ tịch Hội đồng đánh giá khóa luận tốt nghiệp và Chủ nhiệm Bộ môn sau khi luận văn đã được sửa chữa (nếu có).

CHỦ TỊCH HỘI ĐỒNG **CHỦ NHIỆM BỘ MÔN……**

TP. HỒ CHÍ MINH, 2024

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:

Thiết kế và hiện thực hệ thống điều khiển cho cánh tay robot 5 bậc tự do

Nhóm Sinh viên thực hiện:		**Cán bộ hướng dẫn:**
Trần Gia Tuấn	2012357	GS. TS. Hồ Phạm Huy Ánh

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:

<nhận xét về định dạng, cách thức viết báo cáo, phân bố nội dung, chương mục có hợp lý không, …>

2. Về nội dung luận văn:

<nhận xét về kiến thức, phương pháp mà sinh viên đã tìm hiểu, nghiên cứu, nhận xét ưu điểm và hạn chế>

3. Về tính ứng dụng:

<nhận xét về việc xây dựng ứng dụng demo, nhận xét ưu điểm và hạn chế>

4. Về thái độ làm việc của sinh viên:

<nhận xét về thái độ, ưu khuyết điểm của từng sinh viên tham gia>

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

<u>Điểm từng sinh viên:</u>

Trần Gia Tuấn ..... **/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:

Thiết kế và hiện thực hệ thống điều khiển cho cánh tay robot 5 bậc tự do

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

Trần Gia Tuấn 2012357

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:

<nhận xét về định dạng, cách thức viết báo cáo, phân bố nội dung, chương mục có hợp lý không, …>

6. Về nội dung luận văn:

<nhận xét về kiến thức, phương pháp mà sinh viên đã tìm hiểu, nghiên cứu, nhận xét ưu điểm và hạn chế>

7. Về tính ứng dụng:

<nhận xét về việc xây dựng ứng dụng demo, nhận xét ưu điểm và hạn chế>

8. Về thái độ làm việc của sinh viên:

<nhận xét về thái độ, ưu khuyết điểm của từng sinh viên tham gia>

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

<u>Điểm từng sinh viên:</u>

Trần Gia Tuấn ..... /10

Người nhận xét
(Ký tên và ghi rõ họ tên)

TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN: ĐIỀU KHIỂN TỰ ĐỘNG

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

TP. HCM, ngày….tháng…..năm……..

ĐỀ CƯƠNG CHI TIẾT

TÊN LUẬN VĂN: Thiết kế và hiện thực hệ thống điều khiển cho cánh tay robot 5 bậc tự do

Cán bộ hướng dẫn: GS. TS. Hồ Phạm Huy Ánh

Thời gian thực hiện: Từ ngày 01/09/2024 đến ngày 19/12/2024

Sinh viên thực hiện:

Trần Gia Tuấn – 2012357

Nội dung đề tài:

• Mục tiêu:

• Đối tượng và phạm vi nghiên cứu:

Kế hoạch thực hiện:

Công việc	**Thời gian**

Xác nhận của Cán bộ hướng dẫn (Ký tên và ghi rõ họ tên)	TP. HCM, ngày….tháng …..năm….. **Sinh viên** (Ký tên và ghi rõ họ tên)

DANH SÁCH HỘI ĐỒNG BẢO VỆ LUẬN VĂN

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

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

LỜI CẢM ƠN

Lời đầu tiên, tôi muốn bày tỏ lòng biết ơn chân thành và sâu sắc đến quý thầy cô Trường Đại học Bách Khoa – Đại học Quốc Gia Thành Phố Hồ Chí Minh nói chung cũng như Bộ môn Điều khiển và Tự động hóa nói riêng vì đã truyền đạt cho tôi những vốn kiến thức, kỹ năng, kinh nghiệm quý báu trong suốt chặng đường Đại học của tôi giúp tôi có thể hoàn thành đề tài đồ án tốt nghiệp này.

Tôi xin được dành lời cám ơn đặc biệt đến Thầy Hồ Phạm Huy Ánh – là giảng viên đã hướng dẫn tôi cả Đồ án 2 và Đồ án tốt nghiệp này. Tôi xin cám ơn những định hướng và những chỉ dạy về lĩnh vực Thầy đã dành cho tôi, giúp tôi vững tin thực hiện và hoàn thành đồ án của mình. Đây là một nhân duyên và cũng như may mắn khiến tôi có được sự hướng dẫn tuyệt vời của thầy, và tôi biết ơn điều đó. Tôi cũng xin cám ơn Thầy Nguyễn Thanh Tâm – giảng viên hướng dẫn tôi thực hiện Đồ án 1 đã cho tôi cái nhìn toàn diện về lĩnh vực để khám phá ra được đam mê thực sự của mình. Xin cám ơn Thầy Nguyễn Tiến Đạt và các Thầy trong Lab B3 – 108 đã như những người anh truyền đạt kinh nghiệm và kiến thức chuyên môn cho tôi. Cám ơn các Thầy đã mài giũa kỹ năng của tôi thông qua từng công việc nhỏ để có thể góp phần tạo nên tôi của ngày hôm nay.

Tôi xin gửi lời cảm ơn chân thành đến gia đình của mình, đã là chỗ dựa tinh thần, là hậu phương vững chắc trong suốt chặng đường học vấn của tôi. Xin cám ơn các đấng sinh thành vì những lời động viên và chăm sóc trong những lúc khó khăn khi tôi thực hiện đồ án. Lời cảm ơn cuối cùng tôi muốn dành cho bạn bè, những người đã ở bên cạnh tôi suốt những năm Đại học. Gửi lời cảm ơn đặc biệt đến Trịnh Cao Thắng vì đã luôn hỗ trợ hết mình như một người dẫn dắt, cám ơn Trần Phẩm Lương vì những hỗ trợ về mặt tinh thần và cám ơn Lê Nhứt Huy vì những giúp sức về thể chất.

Cuối cùng tôi muốn cám ơn tất cả mọi người một lần nữa vì đã tạo nên một Trần Gia Tuấn kiên trì trên chặng hành trình của mình như ngày hôm nay.

TPHCM, ngày tháng năm

Sinh viên

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC TỪ VIẾT TẮT

CPU: Central Processing Unit.

DH: Denavit – Hartenberg.

DMA: Direct Memory Access.

DOF: Degree Of Freedom.

FIFO: First In First Out.

FOV: Field Of View.

GNU GPL: GNU General Public License.

IDE: Integrated Development Environment.

LSPB: Linear Segment with Parabol Blend.

MSB: Most Significant Bit.

PID: Proportional Integral Derivative.

PWM: Pulse Width Modulation.

ROI: Regions of Interest.

RTOS: Real-Time Operating System.

SIFT: Scale-Invariant Feature Transform.

SoC: System on Chip.

SP: Set Point.

SURF: Speeded-Up Robust Features.

TTL: Transistor – transistor logic.

UART: Universal Asynchronous Receiver – Transmitter.

UV: Ultra Violet.

TÓM TẮT LUẬN VĂN BẰNG TIẾNG VIỆT

Trong thời đại hiện nay, với sự phát triển vượt bậc của khoa học và công nghệ, mọi ngành công nghiệp đang chuyển dịch theo xu hướng tự động hóa. Phần lớn các robot công nghiệp được đề cập trong các dây chuyền sản xuất thường là sự kết hợp của cánh tay robot và thị giác máy. Cánh tay robot có khả năng làm việc ổn định và chính xác cao bởi chúng sẽ không bị ảnh hưởng bởi cảm xúc như con người, hơn thế nữa, những cánh tay này có thể làm việc được dưới những môi trường khắc nghiệt và độc hại với sức khỏe con người như là phân loại hạt trong buồng UV,… Việc nghiên cứu và phát triển cánh tay robot ứng dụng thị giác máy trở thành một trong những vấn đề thiết yếu trong các nhà máy thông minh nhằm cải thiện tính đa dạng của lĩnh vực ứng dụng đồng thời tăng hiệu suất cũng như độ chính xác cho cánh tay robot.

Đồ án này tập trung vào vấn đề áp dụng lý thuyết điều khiển robot vào mô hình thực tế, kết hợp với các giải thuật xử lý ảnh để xử lý yêu cầu gắp vật theo vật mẫu bỏ vào vùng quy định. Trong suốt quá trình hiện thực đề tài, những sự điều chỉnh về mặt cơ khí của khung tay máy, về xung điều khiển động cơ cũng như hiệu chỉnh vị trí và kích thước vật khi được đọc từ camera sẽ luôn được tiến hành nhằm giúp tay máy có thể đến chính xác vị trí và gắp vật.

ABSTRACT

In the current era, with the rapid advancement of science and technology, all industries are shifting towards automation. Most industrial robots mentioned in production lines are often a combination of robotic manipulators and machine vision systems. Robotic manipulators are capable of working stably with high precision as they are unaffected by emotions like humans. Moreover, these arms can operate in harsh and hazardous environments that are detrimental to human health, such as sorting grains in UV chambers. Research and development of robotic arms integrated with machine vision have become essential in smart factories to improve the versatility of application fields, while also increasing the efficiency and precision of robotic arms.

This project focuses on applying robot control theory to a real-world model, integrating image processing algorithms to handle object-picking tasks based on given sample objects and designated areas. Throughout the implementation, adjustments in the robot arm's mechanical frame, motor control pulses, as well as object position and size calibration from the camera, will be continuously carried out. These adjustments aim to ensure the robotic arm can accurately reach the specified positions and grasp objects effectively.

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1 Đặt vấn đề

Trong những năm gần đây, việc áp dụng robot vào sản xuất công nghiệp đã và đang diễn ra mạnh mẽ, không chỉ nhằm gia tăng năng suất mà còn để giảm thiểu các rủi ro trong môi trường lao động nguy hiểm. Theo các báo cáo, robot công nghiệp được triển khai rộng rãi trong các ngành như lắp ráp điện tử, gia công cơ khí, và đặc biệt là đóng gói và kiểm tra sản phẩm. Thực tế này góp phần vào xu hướng thay thế lao động thủ công bằng robot, không chỉ để đáp ứng các yêu cầu chất lượng sản phẩm mà còn khắc phục tình trạng thiếu hụt lao động trong nhiều ngành công nghiệp.

“Sử dụng Robot công nghiệp, cánh tay robot công nghiệp trong sản xuất là một giải pháp tối ưu giúp doanh nghiệp nâng cao năng suất lao động, tăng khả năng cạnh tranh.”[1]

“Tại Việt Nam, thị trường Robot công nghiệp được cho là sẽ phát triển mạnh trong thời gian tới, cùng với việc các doanh nghiệp chú trọng vào cách mạng công nghiệp 4. Robot công nghiệp tăng trưởng nhiều nhất trong các ngành: Sản xuất ô tô (83%), Việt Nam hiện cũng được các chuyên gia đánh giá là thị trường tiềm năng của Robot công nghiệp với nhiều các nhà máy sản xuất sử dụng Robot như: Vinfast, Thaco...” [1]

“Thông thường, với một sản phẩm công nghiệp, các nhà sản xuất thường chế tạo ở nhiều kích cỡ theo nhu cầu đa dạng của thị trường cung ứng. Điều này dẫn đến việc phải điều chỉnh máy móc hỗ trợ nhiều lần, gây mất thời gian. Từ khi ứng dụng robot cộng tác, rào cản này đã không còn đáng kể.” [2]

“Do nhiệm vụ lắp ráp thường nằm ở cuối giai đoạn sản xuất và là một trong những khâu hoàn thiện sản phẩm nên việc đảm bảo chất lượng và tính nhất quán luôn được chú trọng. Đối với hai yêu cầu tiên quyết này, không ai có thể phủ nhận các

cobot thường làm tốt điều này hơn so với con người. Kết quả khả quan ở nhiều nhà máy khi ứng dụng robot cộng tác đã chứng minh nhận định trên là hoàn toàn chính xác.

Hơn thế nữa là khả năng ứng dụng của cánh tay robot cộng tác trong các nhiệm vụ lắp ráp đã cải thiện sức khỏe công nhân khá nhiều. Trước kia, những người lao động phải trực tiếp thực hiện hoàn toàn 100% thủ công, đến nay những phân đoạn phải tiêu hao sức lực đã được hạn chế đến mức tối đa.” [2]

““Khu vực nào của Trung Quốc phổ cập robot cao hơn, thì dòng người lao động nhập cư sẽ giảm xuống”, Osea Giuntella, giáo sư kinh tế ở Đại học Pittsburgh (Mỹ) và là tác giả chính của báo cáo nghiên cứu của Cục Nghiên cứu kinh tế quốc gia Mỹ (NBER) về phản ứng của lao động đối với tự động hóa ở Trung Quốc nhận định.”[-]

1.2 Tình hình nghiên cứu

Tại Việt Nam, các dự án liên quan đến chế tạo và ứng dụng robot ngày càng nhiều. Ví dụ, nhiều nhóm sinh viên và nhà nghiên cứu đã chế tạo thành công các robot phục vụ trong các môi trường độc hại hoặc đảm nhận các công việc phức tạp như lắp ráp và phân loại sản phẩm [ ] []. Nghiên cứu và ứng dụng cánh tay robot đã có những bước tiến khả quan, với một số dự án nổi bật khác như robot SM6 – cánh tay robot công nghiệp 6 bậc tự do do Viện Khoa học và Công nghệ Quân sự phát triển. Sản phẩm này được đánh giá là một bước đột phá trong việc nội địa hóa công nghệ, giảm giá thành sản phẩm và phù hợp với nhu cầu của doanh nghiệp vừa và nhỏ trong nước. Robot SM6 được thiết kế nhỏ gọn, dễ vận hành, và có khả năng cạnh tranh với các sản phẩm nhập khẩu [3].

Nghiên cứu và phát triển cánh tay robot công nghiệp đã đạt được nhiều tiến bộ, đặc biệt trong các ngành công nghiệp tự động hóa, ô tô, và y tế. Các cánh tay robot hiện đại ngày nay được trang bị trí tuệ nhân tạo và công nghệ cảm biến tiên tiến, cho

phép chúng thực hiện các nhiệm vụ với độ chính xác và hiệu suất cao. Ví dụ, robot của KUKA và ABB được ứng dụng rộng rãi trong hàn, lắp ráp, và kiểm tra sản phẩm. Các hệ thống điều khiển hiện đại sử dụng phương pháp học sâu, mạng nơ-ron nhân tạo, và thuật toán tối ưu để nâng cao hiệu suất robot.

Robot công nghiệp như "Pick and Place" được sử dụng phổ biến trong các ngành thực phẩm, dược phẩm, và hàng tiêu dùng. Các loại robot khác như robot hàn, robot cắt bằng tia nước, và robot đánh bóng cũng đã chứng minh được hiệu quả trong sản xuất tự động, giúp tăng năng suất, tiết kiệm chi phí, và nâng cao chất lượng sản phẩm [4].

1.3 Lý do chọn đề tài

Vì các thực trạng được nêu ở mục 1.1, nghiên cứu và phát triển cánh tay robot công nghiệp trở thành một đề tài phổ biến tại các đồ án tốt nghiệp của sinh viên. Việc nghiên cứu và phát triển cánh tay robot 5 bậc tự do (5-DOF) để thực hiện các tác vụ như gắp nắp chai theo nắp mẫu bằng thuật toán SURF là một đề tài rất có ý nghĩa trong bối cảnh công nghiệp hiện nay. Đề tài giúp trao dồi đúng lĩnh vực đang theo đuổi cũng như vận dụng tốt và phát triển các kiến thực đã học tại Đại học.

Đề tài là sự kết hợp của nhiều kiến thức lĩnh vực của ngành Kỹ thuật Điều khiển và Tự động hóa như robot, nhúng, thị giác máy, đo lường điều khiển máy tính,… như một bài kiểm tra cuối cùng về toàn bộ những kiến thức liên quan đến chuyên ngành đối với sinh viên.

1.4 Mục tiêu của đề tài

Trong đề tài này, chúng ta cần điều khiển thành công cánh tay robot 5 bậc tự do với ứng dụng của giải thuật nhận diện đặc trưng, đề tài cần được làm rõ các nội dung sau:

- Thiết kế cánh tay cánh tay robot và lắp đặt hệ thống camera.

- Nghiên cứu và tìm hiểu về vi điều khiển STM32.
- Giải bài toán động học và hoạch định quỹ đạo đối với cánh tay robot.
- Tìm hiểu và lựa chọn bộ điều khiển động cơ.
- Tìm hiểu về máy tính nhúng Raspberry Pi.
- Nghiên cứu về SURF và các giải thuật xử lý ảnh liên quan.
- Lập trình các giải thuật xử lý ảnh để đọc tọa độ vật.
- Calib hệ tọa độ vật và cánh tay robot.
- Truyền nhận bằng UART.
- Nghiên cứu và tìm hiểu về FreeRTOS trên STM32.
- Kết hợp cánh tay robot và xử lý ảnh bằng chia luồng trên RTOS.

1.5 Cấu trúc của đề tài

Cấu trúc của đề tài gồm có:

- **Chương 1**: Giới thiệu đề tài.
- **Chương 2**: Tổng quan về mô hình cánh tay robot.
- **Chương 3**: Tính toán động học cho cánh tay robot.
- **Chương 4**: Giải thuật xử lý ảnh.
- **Chương 5**: Giải thuật điều khiển.
- **Chương 6**: Kết luận.

CHƯƠNG 2. TỔNG QUAN VỀ MÔ HÌNH CÁNH TAY ROBOT

Chương này giới thiệu các linh kiện và vật liệu chính trong mô hình cánh tay robot, bao gồm các khối nguồn, khối điều khiển và khối thị giác máy. Sự kết hợp này tạo nên cấu trúc bền vững và hệ thống điều khiển chính xác, đáp ứng yêu cầu về hiệu suất và tính linh hoạt cho robot.

2.1. Tổng quan

Hệ thống được chia làm hai phần chính cần thiết kế và điều khiển, gồm: phần cánh tay robot và phần thị giác máy.

Phần cánh tay robot được lắp ghép từ các động cơ servo với bộ não điều khiển là mạch vi điều khiển STM32 thông qua PCA 9685. Mỗi động cơ sẽ đảm bảo một vai trò cho từng góc quay của các khớp robot được lập trình phối hợp với nhau để chuyển động mượt mà. Nguồn sử dụng chính trong phần cánh tay cần được cân nhắc dựa trên nguồn sử dụng của các loại servo, mạch điều khiển và PCA 9685. Nguồn cũng cần đảm bảo đủ dòng cho mỗi servo trong cánh tay. Hệ thống cần được tinh gọn để không làm ảnh hưởng đến quy trình chuyển động của cánh tay robot.

Phần thị giác máy sẽ được điều khiển bởi máy tính nhúng là Raspberry Pi với các tác vụ xử lý ảnh nhằm cung cấp chính xác tọa độ cho mạch STM32. Raspberry Pi có giá thành khá cao và là một phần linh kiện quan trọng trong cả hệ thống, vì vậy nguồn cấp cho mạch phải đúng chính xác như các thông số cung cấp của nhà phát hành. Camera sử dụng sẽ được chọn lựa sao cho thích ứng với máy tính nhúng tốt nhất. Bên cạnh đó, phần này còn yêu cầu phải thiết kế lắp đặt camera ở nơi có thể quan sát cung cấp tọa độ cần thiết mà không bị ảnh hưởng bởi workspace của robot.

2.2. Khối cánh tay robot

Yêu cầu của bài toán đặt ra là thiết kế hệ thống điều khiển cánh tay robot có khả năng

di chuyển đến vị trí tương ứng của vật và gắp vật.

2.2.1. Nguồn

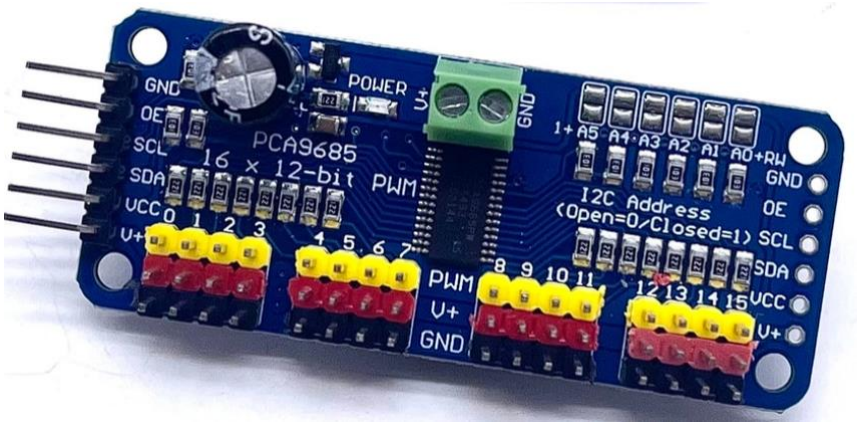
Để vận hành các động cơ servo cũng như các mạch điều khiển, ta cần một nguồn cấp 5V và dòng đủ lớn để điều khiển các servo, vì thế nguồn tổ ong 5V – 20A sẽ được sử dụng trong đồ án này để cấp nguồn cho các thành phần liên quan đến cánh tay robot.

Hình 2.1. Nguồn tổ ong 5 V - 20 A.

Điện áp đầu vào	180 ~ 240 VAC
Công suất	100 W
Dòng đầu ra tối đa	20 A
Nhiệt độ làm việc	10^{o} C ~ 60^{o} C
Kích thước	198 x 48 x 42 mm

Bảng 2.1. Thông số kỹ thuật nguồn tổ ong.

Để thuận tiện cho việc thiết kế hệ thống cũng như trong quá trình điều khiển, các động cơ servo được điều khiển thông qua PCA 9685. Mạch điều khiển 16 Chanel PWM PCA9685 được sử dụng để có thể xuất ra đồng thời 16 xung PWM từ 16 cổng khác nhau thông qua giao tiếp I2C sử dụng IC PCA 9685.



Hình 2.2. Mạch điều khiển 16 kênh PWM PCA9685.

IC chính	PCA 9685
Điện áp sử dụng	2.3 ~ 5.5 VDC
Số kênh PWM	16 kênh
Tần số	40 ~ 1000 Hz
Độ phân giải PWM	12 bit
Giao tiếp	I2C (chấp nhận mức Logic TTL 3 ~ 5 VDC)
Kích thước	62.5 x 25.4 x 3 mm

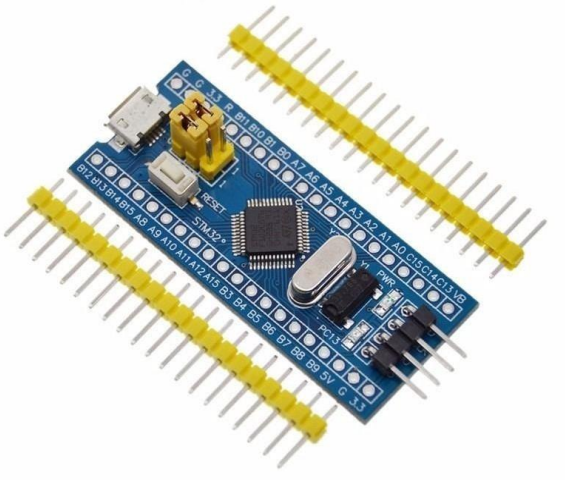
Bảng 2.2. Thông số kỹ thuật của PCA 9685.

2.2.2. Mạch điều khiển robot

Đối với khối điều khiển, ta cần các tác vụ giao tiếp I2C với PCA 9685 và giao tiếp UART với Raspberry Pi, vì thế đề xuất được đưa ra ở dự án này là mạch STM32F103C8T6 Blue Pill dùng làm bộ não chính của cánh tay robot.

Là một phiên bản nhỏ gọn trong các dòng chip phổ biến của ST nhưng vẫn giữ được các chức năng cơ bản, tương tự và cần thiết so với các loại vi điều khiển khác. STM32F103 thuộc họ F1 với lõi ARM 32-bit COTEX-M3 RISC cùng bộ dao động bên trong 4-16MHz, là một chip công nghệ flash CMOS.

Với số lượng các Timers, USART, và DMA được tích hợp trong vi điều khiển này cho phép nhận và truyền dữ liệu giữa lớp trên và lớp dưới một cách hợp lý.



Hình 2.3. Mạch vi điều khiển STM32F103C8T6.

Vi điều khiển	ARM 32-bit COTEX-M3 RISC
Điện áp hoạt động	2.0 ~ 3.6 VDC
Dòng điện nguồn	6 mA
Số lượng chân	47
Bộ nhớ flash	64/128 KB
Bộ dao động bên trong	4 → 16 MHz
Tốc độ CPU	72 MHz (tối đa)
Số Timer	3 Timers 16 bit hỗ trợ mode Input Capture/Output Compare/PWM, 1 Timer 16 bit hỗ trợ điều khiển động cơ, 2 Watchdog Timer, 1 Systick Timer 24 bit
Số bộ I2C	2
Số bộ USART	3
Số kênh DMA	7

Bảng 2.3. Thông số kĩ thuật của STM32F103C8T6.

2.2.3. Động cơ

Động cơ RC Servo TowerPro MG996R

Động cơ RC Servo TowerPro MG996R là loại động cơ servo đến từ hãng TowerPro với lực kéo, độ bền, độ chính xác và độ ổn định cao, động cơ có cấu tạo hộp số hoàn toàn bằng kim loại với trục chính bằng nhôm hợp kim có độ cứng cao 6061-T6 (màu bạc) giúp giảm trọng lượng, ngoài ra trục chính và các trục quay trong hộp số còn được bổ trợ bạc đạn và các vòng đệm kim loại giúp tăng độ bền và độ chính xác của động cơ khi hoạt động. *Động cơ RC Servo TowerPro MG996R* là dạng Digltal RC Servo với các đặc tính vượt trội và độ chính xác cao, thích hợp với các mô hình robot phức tạp.

Hình 2.4. Động cơ RC Servo MG996R.

Điện áp hoạt động	4.8 ~ 6.6 VDC
Lực kéo	9.4 kg/cm ở mức 4.8 VDC 11 kg/cm ở mức 6 VDC

Tốc độ quay	0.19s/60° (4.8 VDC không tải) 0.15s/60° (6 VDC không tải)
Kích thước	40 x 19.7 x 42.9 mm
Trọng lượng	55 g

Bảng 2.4. Thông số kĩ thuật của Động cơ RC Servo MG966.

Động cơ Digital RC Servo HiWonder HPS-2027

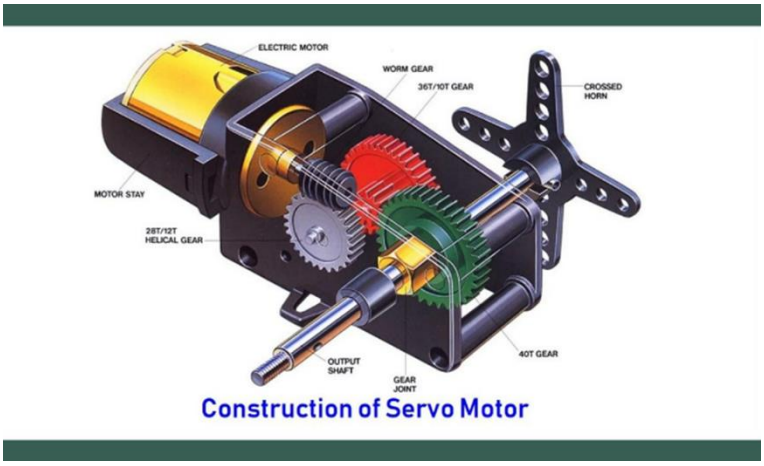
Vì khớp thứ 2 của cánh tay robot chịu moment quán tính lớn khiến cho *động cơ servo MG996R* không đủ khả năng ghì lại làm cho cánh tay bị ngã về phía trước dù đã cung cấp đủ dòng điện cần tiêu thụ. Việc ngã về phía trước sẽ dẫn đến sai lệch vị trí gắp của cánh tay hoặc hỏng cả hệ thống. Do đó, *động cơ servo HiWonder HPS-2027* được sử dụng như một sự thay thế cần thiết, giúp khớp thứ 2 của cánh tay robot có đủ khả năng chịu lực, giảm sai số do cơ khí và xung chạy của servo gây ra cho hệ thống.

Động cơ Digital RC Servo HiWonder HPS-2027 được sử dụng trong các Robot có cấu trúc phức tạp: Humanoid Robot, Biped Robot, Robot nhện,...vì các đặc tính vượt trội của Digital RC Servo so với các loại Analog RC Servo truyền thống *MG995, MG996*: lực kéo khỏe hơn, phản ứng nhanh hơn, chạy mượt hơn,... *Động cơ Digital RC Servo HiWonder HPS-2027* có hộp số bánh răng kim loại, tốc độ phản ứng nhanh với lực kéo khỏe theo thông số nhà sản xuất moment lên đến 20 kg.cm.

Hình 2.5. Động cơ Digital RC Servo HiWonder HPS-2027.

Điện áp hoạt động	6 ~7.4 VDC
Lực kéo	20 kg.cm tại 7.4 VDC
Tốc độ quay	0.2s / 60 ° tại 7.4 VDC
Kích thước	54.5 x 20 x 47.5mm
Trọng lượng	66 g
Tần số PWM	500 ~ 2500 us tương ứng 0^o ~ 270^o

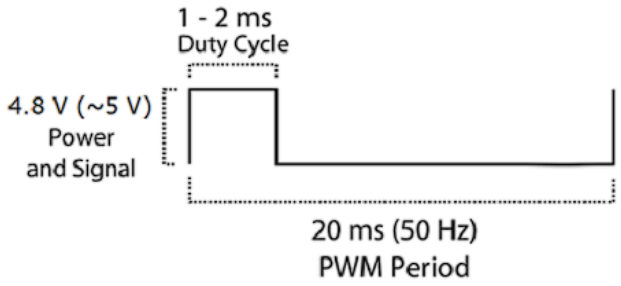
Bảng 2.5. Thông số kỹ thuật động cơ Digital RC Servo HiWonder HPS-2027.



Hình 2.6. Cấu tạo của 1 động cơ servo.

Động cơ servo được điều khiển bằng cách gửi một xung điện có độ rộng thay đổi hoặc điều chế độ rộng xung (PWM), thông qua dây điều khiển. Có một xung tối thiểu, một xung tối đa và tốc độ lặp lại. Một động cơ servo thường chỉ có thể quay 90 độ theo một trong hai hướng cho tổng chuyển động là 180 độ. Vị trí trung tính của động cơ được xác định là vị trí mà servo có cùng số vòng quay tiềm năng theo cả chiều kim đồng hồ hoặc ngược chiều kim đồng hồ. PWM được gửi đến động cơ xác định vị trí của trục và dựa trên thời gian của xung gửi qua dây điều khiển, motor sẽ chuyển sang vị trí mong muốn của người điều khiển.

Tín hiệu điều khiển servo là dạng tín hiệu tương tự như *Hình 2.7.*



Hình 2.7. Dạng tín hiệu điều khiển góc quay động cơ MG996R tại 50 Hz.

Theo mô tả của hình trên:

- Ứng với 1 ms sẽ là 0°.
- Ứng với 2 ms sẽ là 180°.

Đối với *động cơ Digital RC Servo HiWonder HPS-2027* thì sẽ là:

- Ứng với $500\,\mu s$ sẽ là 0^o.
- Ứng với $2500\,\mu s$ sẽ là 270^o.

Để tính xung cần cấp tương ứng với mỗi góc bất kỳ, ta sử dụng công thức như sau:

$$PulseWidth(ms) = Min\,PulseWidth + \left(\frac{Max\,PulseWidth - Min\,PulseWidth}{180} \times Góc \right).$$

2.3. Khối thị giác máy

Raspberry Pi chạy với hệ điều hành Raspberry Pi OS (trước đây gọi là Raspbian), một phiên bản của Debian Linux được tối ưu hóa cho phần cứng của Raspberry Pi. Khi chọn camera tương ứng, ta cũng cần tìm hiểu về thư viện điều khiển camera đó trên Raspberry. Có hai lựa chọn để có thể điều khiển Raspberry để ta cân nhắc khi mua linh kiện: remote control và direct control. Nếu chọn direct control, ta cần mua màn LCD để hiển thị cũng như cần chuột và bàn phím riêng để lập trình trên máy tính nhúng.

2.3.1. Nguồn

Đối với máy tính nhúng Raspberry Pi 4B, cần sử dụng nguồn riêng của hãng để tránh gây ra hư hỏng.

Hình 2. 8. Nguồn Raspberry Pi 4 -5.1 V – 3 A.

Điện áp đầu ra	5.1 VDC
Dòng tối thiểu	0 A
Dòng tối đa	3 A
Công suất tối đa	15.3 W
Điều tiết tải	± 5 %
Quy định dòng	± 2 %
Bảo vệ	• Bảo vệ ngắn mạch • Bảo vệ quá dòng • Bảo vệ quá nhiệt
Hiệu quả	Tối thiểu 81 % (dòng đầu ra từ 100 %, 75 %, 50 %, 25 %) tối thiểu 72 % ở 10 % tải
Chuẩn kết nối đầu ra	USB Type - C

Bảng 2.6. Thông số kĩ thuật của nguồn Raspberry Pi 4.

2.3.2. Camera

Camera đóng vai trò là đôi mắt cho cánh tay robot hoạt động. Đối với dự án này, thị giác sẽ thực hiện tác vụ đọc được tọa độ thực của vật trên bề mặt phẳng, từ đó sẽ truyền thông tin về khối điều khiển để xử lý.

Camera sẽ đóng vai trò vô cùng quan trọng trong việc điều hướng robot sao cho đi đúng đến vị trí vật, vì lẽ đó lựa chọn camera nào cần được cân nhắc trước khi đưa ra quyết định. Camera được lựa chọn ở đây sẽ là Raspberry Pi Camera module V3 75°.

Hình 2.9. Cảm biến USB Camera.

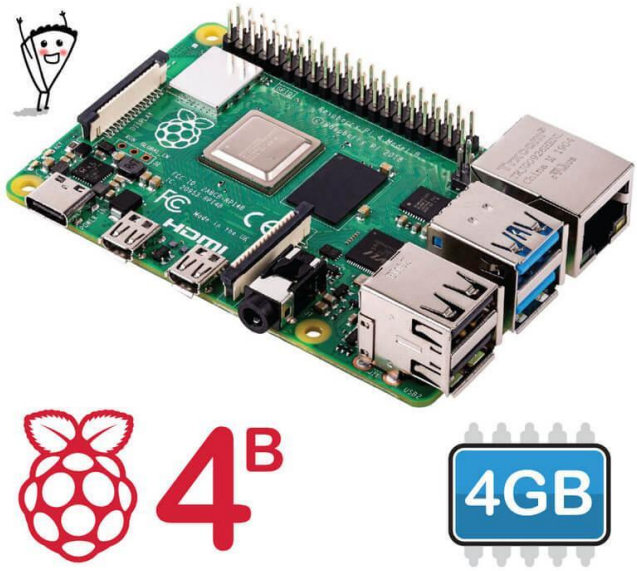
Độ phân giải	12 MP
Độ phân giải ảnh	4608 x 2592
Độ phân giải video	1080p@50fps
Cảm biến hình ảnh	IMX708
Góc nhìn	75°

Kích thước	25 x 23.862 mm

Bảng 2.7. Thông số kĩ thuật của USB Camera.

2.3.3. Máy tính nhúng

Để có thể thu nhận hình ảnh từ camera về xử lý để cho ra tọa độ vật, máy tính nhúng lựa chọn được đưa ra vì mạch điều khiển STM32F103C8T6 sẽ không đủ khả năng để xử lý công việc giao tiếp với camera. Máy tính nhúng lựa chọn sẽ là Raspberry Pi model 4B phiên bản 4GB.



Hình 2.10. Raspberry Pi 4 Model B – 4GB RAM.

RAM	4GB
SoC	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit @ 1.5GHz

Nguồn điện	5 VDC – 3 A chuẩn USB – C
Wifi	Chuẩn 2.4 GHz và 5.0 GHz IEE 802.11ac
Kích thước	85 x 56 mm

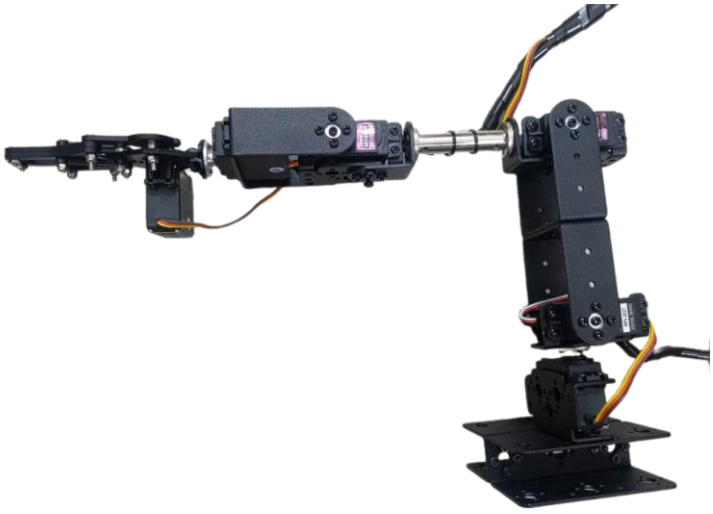
Bảng 2.8. Thông số kỹ thuật của Raspberry Pi 4 Model B.

2.4. Mô hình cánh tay và hệ thống

Hình 2.11. Mô hình cánh tay máy 5 bậc tự do.

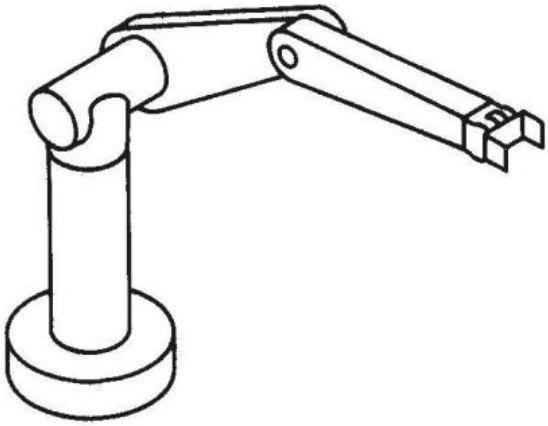
Tuy tay máy được vận hành bởi 6 động cơ servo, nhưng động cơ thứ 6 sẽ chịu trách nhiệm cho phần gripper của cánh tay. Các động servo cần được set một góc 90^o trước khi lắp vào mô hình, bởi vì ngoài động cơ HiWonder HPS-2027 có ngưỡng hoạt

động từ 0^o đến 270^o ra thì tất cả các động cơ còn lại đều có ngưỡng từ 0^o đến 180^o, vì vậy việc set 90^o sẽ giúp cánh tay robot không bị giới hạn khi di chuyển tới lui, tuy là giới hạn quay tới hoặc quay lùi sẽ bị giảm một nửa. Cánh tay sau khi set các góc như trên thì sẽ có vị trí bắt đầu của các khớp như *Hình 2.12.*

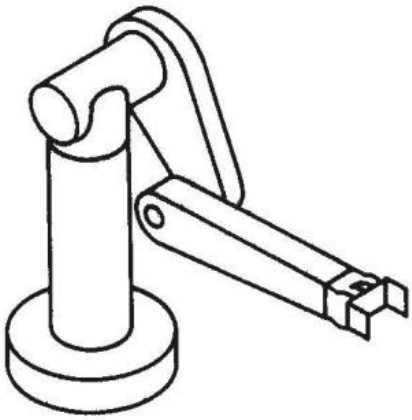


Hình 2.12. Vị trí sau khi preset các góc servo.

Theo như quan sát, khi ở vị trí bắt đầu như trên thì khi cánh tay duỗi thẳng một đường, động cơ thứ 3 sẽ đạt góc 180^o, vì vậy loại robot trong đồ án này phải là Above Arm chứ không phải Below Arm. Mỗi trường hợp cánh tay robot sẽ cho ra một cách giải động học ngược khác nhau, vì vậy việc xác định loại robot từ bây giờ sẽ giúp ích cho phần tính động học ở chương sau.



Hình 2. 13. a) Above Arm.



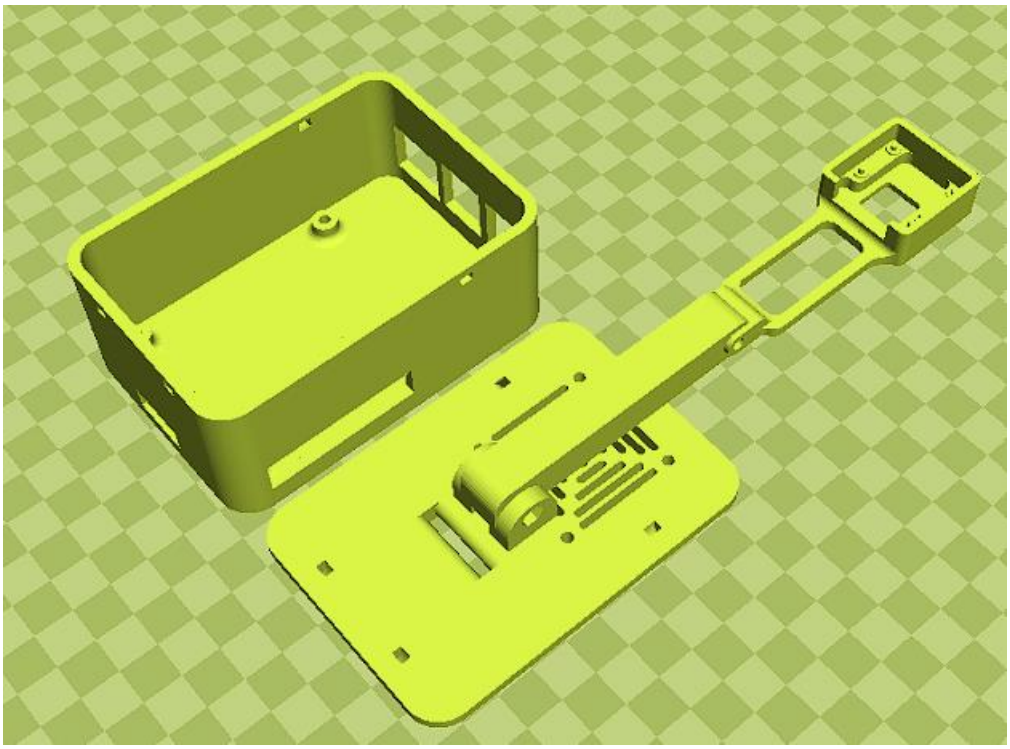
Hình 2. 14. b) Below Arm

Hình 2. 15. Loại cánh tay robot.

Trụ camera được chọn là Tripod thường dùng để đỡ máy chụp hình. Việc sử dụng như vậy vì Tripod có thể điều khiển độ cao cũng như góc nhìn camera. Khi sử dụng cần cố định độ cao của Tripod để dễ dàng calib hệ mỗi khi hoạt động. Chiều cao trục Tripod được set cố định cho trường hợp này là khoảng 23 cm ~ 25 cm, sai số do co giãn vì nhiệt hoặc va chạm vô tình làm thay đổi.

Hình 2.16. Tripod.

Vì camera sử dụng và Raspberry Pi không thể cố định với nhau cũng như case Raspberry Pi của hãng cũng không hỗ trợ nơi đặt camera cố định, nên cần thiết kế 1 case có khả năng vừa bảo vệ Raspberry Pi vừa giữ camera cố định trong quá trình vận hành hệ thống.



Hình 2.17. Case đựng camera và Raspberry Pi.

Hình 2.18. Case đựng camera và Raspberry Pi thực tế.

Độ cao từ mắt camera đến sàn quan sát là ~ 35 cm sẽ được hiệu chỉnh đúng như

vậy mỗi lần calib, vì chương trình chạy trong giải thuật xử lý ảnh đã được đặt hệ số scale đúng tương đương với độ cao 35 cm.

Hình 2.19. Hệ thống camera sau lắp đặt.

CHƯƠNG 3. TÍNH TOÁN ĐỘNG HỌC CHO CÁNH TAY ROBOT

Trong chương này, tính động học và hoạch định quỹ đạo của cánh tay robot sẽ được trình bày để làm rõ cách xác định vị trí, hướng của các khớp và lập trình quỹ đạo di chuyển. Bắt đầu với các khái niệm động học thuận và ngược, chương giúp cung cấp nền tảng tính toán chính xác chuyển động cho cánh tay robot. Ngoài ra, chương sẽ đề cập đến việc hoạch định quỹ đạo nhằm đảm bảo chuyển động mượt mà, giảm thiểu dao động và tối ưu thời gian di chuyển, qua đó nâng cao hiệu suất và độ chính xác của robot trong thao tác thực tiễn.

3.1. Tổng quan

Phần này cung cấp cái nhìn toàn diện về động học và hoạch định quỹ đạo trong hệ thống điều khiển cánh tay robot, đóng vai trò nền tảng trong việc thiết lập chuyển động chính xác và hiệu quả cho các ứng dụng thực tiễn. Đầu tiên, chương trình bày các nguyên lý động học thuận và động học ngược. Động học thuận là quá trình tính toán vị trí và hướng của hiệu dụng cụ từ các giá trị góc khớp đã biết, còn động học ngược cho phép xác định các góc khớp dựa trên vị trí mong muốn của hiệu dụng cụ. Hai kỹ thuật này bổ trợ lẫn nhau, giúp xây dựng mối quan hệ giữa không gian khớp và không gian tác vụ, đồng thời là cơ sở cho việc xác định chính xác các tọa độ và góc của mỗi khớp trong không gian ba chiều.

Tiếp theo, chương tập trung vào các phương pháp hoạch định quỹ đạo để đảm bảo cánh tay robot có thể di chuyển theo lộ trình liên tục và tối ưu. Hoạch định quỹ đạo không chỉ yêu cầu xác định lộ trình mà còn đòi hỏi tính toán vận tốc và gia tốc hợp lý tại các điểm chuyển tiếp, từ đó giảm thiểu các dao động đột ngột và duy trì sự ổn định trong chuyển động. Việc xây dựng quỹ đạo với các đoạn chuyển tiếp mượt mà giúp tối ưu thời gian di chuyển, đảm bảo các thao tác nhanh chóng và an toàn mà vẫn đạt độ chính xác cao. Bằng cách áp dụng các kỹ thuật động học và hoạch định quỹ đạo được

đề cập trong chương này, người thiết kế và điều khiển robot sẽ có cơ sở vững chắc để triển khai cánh tay robot trong các nhiệm vụ đòi hỏi hiệu suất cao và tính chính xác, như trong các hệ thống sản xuất tự động hay ứng dụng dịch vụ.

3.2. Cơ sở lý thuyết về cánh tay robot 5 bậc tự do

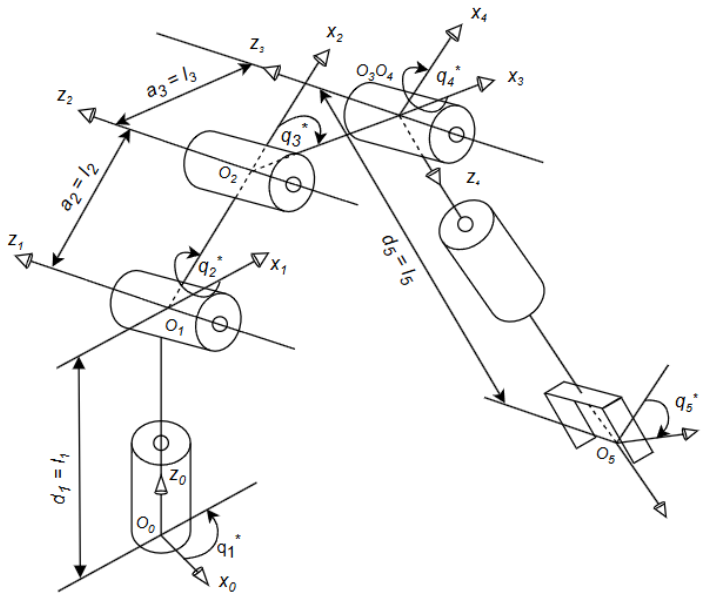
3.2.1. Đặt hệ trục tọa độ lên cánh tay robot

Để có thể đặt được trục, ta cần tuân thủ vài nguyên tắc cơ bản về đặt trục của phương pháp DH như sau:

- Trục z sẽ nằm trùng với trục xoay của động cơ.
- Trục x sẽ song song với pháp tuyến chung của $\vec{z}_n$ và $\vec{z}_{n-1}$ như sau

$$\vec{x}_n = \vec{z}_n \times \vec{z}_{n-1}.$$

- Hướng của $\vec{x}_n$ sẽ hướng từ $\vec{z}_{n-1}$ đến $\vec{z}_n$.
- Trục y được xác định bằng quy tắc bàn tay phải.



Hình 3.1. Hệ trục tọa độ từng khớp của cánh tay robot trong không gian 3 chiều.

3.2.2. Thiết lập bảng DH

Các thông số DH tương ứng với các khớp sẽ được xác định dựa trên quy tắc sau:

- θ_i là góc quay quanh trục z_{i-1} từ x_{i-1} sang x_i.
- d_i là khoảng cách từ điểm giao nhau của trục z_{i-1} với trục x_i đến gốc O_{i-1}.
- α_i là góc xoắn quay quanh trục x_i từ z_{i-1} sang z_i.
- a_i là khoảng cách dọc theo pháp tuyến chung giữa trục z_{i-1} và trục z_i.

Khâu	a_i	α_i	d_i	θ_i
1	0	$-\frac{\pi}{2}$	d_1	θ_1^*
2	a_2	0	0	θ_2^*
3	a_3	0	0	θ_3^*
4	0	$\frac{\pi}{2}$	0	θ_4^*
5	0	0	d_5	θ_5^*

Bảng 3.1. Bảng DH.

Để có được các thông số điền vào và hoàn thiện bảng DH, ta tiến hành đo đạc kính thước cho tay máy và có được kết quả như *Bảng 3.2.*

Thông số	d_1	a_2	a_3	d_5
Kích thước (cm)	9.4	10.5	13	18

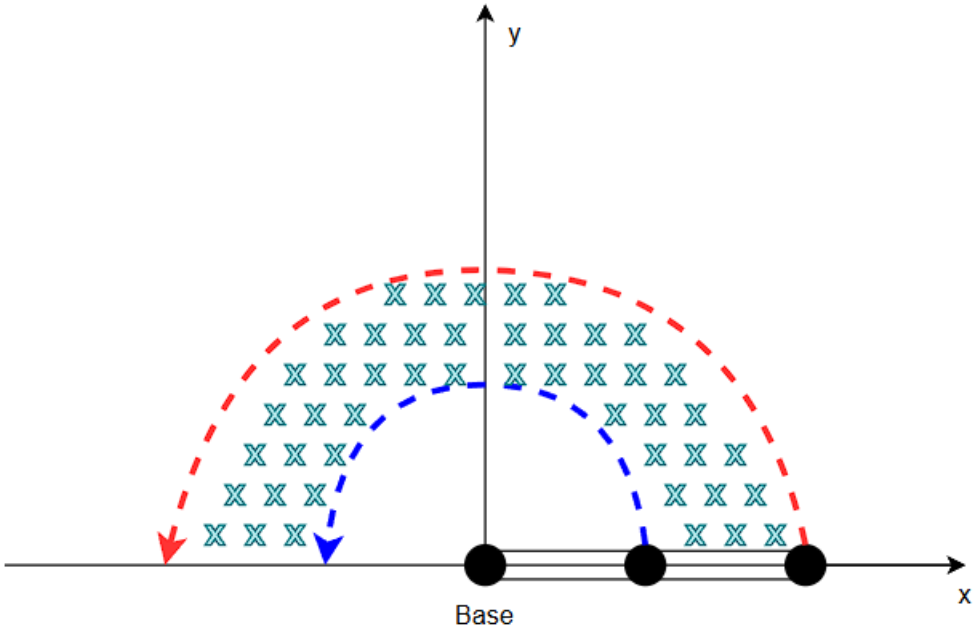
Bảng 3.2. Kích thước tay máy.

3.2.3. Không gian làm việc của cánh tay robot

Vùng giới hạn của robot khi chiếu lên từ trên xuống

Không gian làm việc của cánh tay robot được định nghĩa là một tập hợp các điểm nằm trong đó mà cánh tay robot có thể vươn tới được. Trong đồ án này, cánh tay robot được thiết lập để gắp theo chiều vuông góc với mặt sàn, vì vậy không gian sẽ khác với khi robot hoạt động tự do.

Khớp thứ nhất có khả năng quay được $[0^o;180^o]$, nên ta có được vùng hoạt động của cánh tay robot khi chiếu lên mặt phẳng Oxy như *Hình 3.2*.



Hình 3.2. Vùng hoạt động giới hạn của robot khi chiếu từ trên xuống.

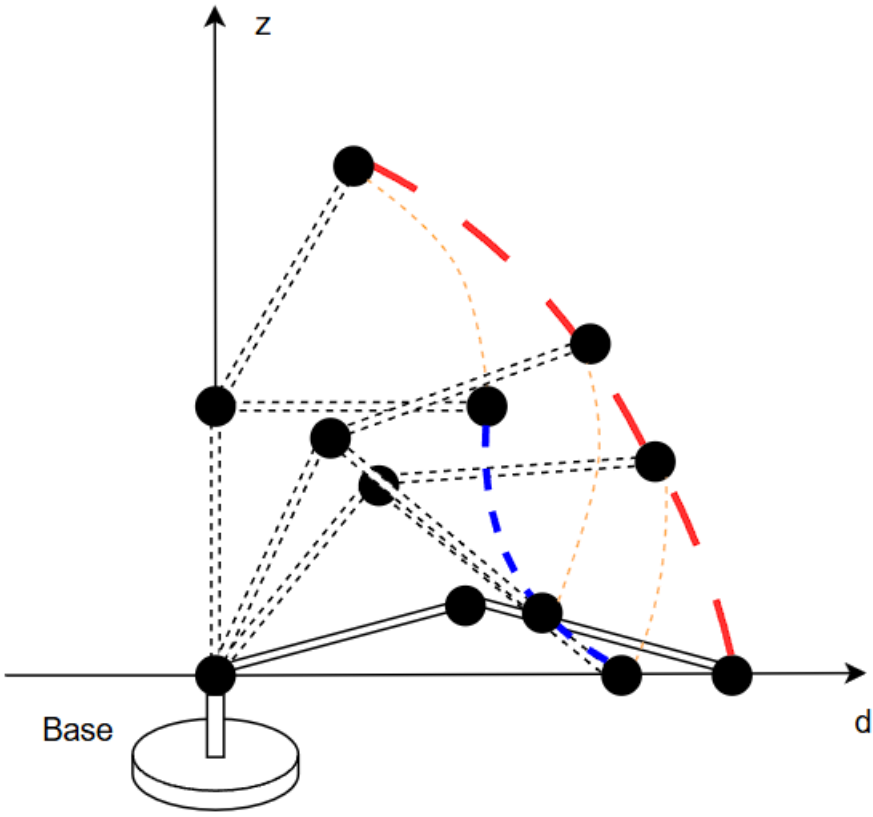
Vùng đánh dấu sẽ là vùng tồn tại cặp tọa độ (x, y) mà cánh tay robot có thể vươn tới được. Từ đó ta tiến hành đặt $D_{\min}$ là khoảng cách ngắn nhất từ gốc tọa độ mà cánh tay có thể vươn tới; $D_{\max}$ là khoảng cách xa nhất từ gốc tọa độ mà cánh tay có thể vươn tới.

Đặt $R = \sqrt{x^2 + y^2}$; $\theta_1 = arctan2(y, x)$.

Vậy $D_{min} < R < D_{max}$; $0° < \theta_1 < 180°$.

Mà theo như phương pháp gắp vật của đồ án này, cánh tay sẽ gắp thẳng đứng tương ứng với việc $D_{min} = a_2$; $D_{max} = a_2 + a_3$.

Vùng giới hạn của robot khi chiếu theo phương ngang từ trái sang phải



Hình 3.3. Vùng hoạt động giới hạn của robot khi chiếu ngang.

Đôi với phần tìm vùng hoạt động của robot khi chiếu lên phương ngang, ta cần quy đổi hệ tọa độ trong không gian Cartesian [x; y; z] sang hệ tọa độ [z; d] để tiến hành phân tích.

$$d = sign(y)\cdot\sqrt{x^2 + y^2}.$$

Theo phương pháp giải quyết của đồ án này cùng với xem xét vùng hoạt động của robot trên mặt phẳng Oxy, ta thấy được cánh tay robot sẽ được yêu cầu hoạt động trong vùng y > 0, mà góc thực tế của θ_2 sẽ được tính cùng chiều kim đồng hồ bắt đầu từ trục âm của y. Vì vậy khi xét với vùng hoạt động như trên, lý thuyết góc θ_2 sẽ chạy từ $[0°;90°]$ nhưng trên thực tế sẽ là từ $[90°;180°]$. Vậy ngưỡng giới hạn về góc của θ_2 là:

$$90° < \theta_2 < 180°.$$

Tiếp theo ta sẽ phân tích để tìm giới hạn góc θ_3. Phương pháp gắp vật được đề ra trong đồ án này là gắp thẳng đứng với phương vuông góc với mặt sàn gắp, mà ta có: $d_1 = 9.4cm; a_2 = 10.5cm; d_5 = 18cm \rightarrow d_1 + a_2 = 19.9cm \approx d_5 = 18cm.$

Vì vậy để khi gắp vật, cánh tay không va chạm với sàn hoặc không hất vật đi trước khi kịp gắp, ta sẽ đặt cận dưới cho góc θ_3 là 90º. Cận trên sẽ không bị ràng buộc bởi các điều kiện bài toán, tuy nhiên để cho cánh tay robot có thể vận hành trơn tru, ta sẽ đặt cận trên là 150º. Vậy ngưỡng giới hạn về góc của θ_3 là $90° < \theta_2 < 150°$.

Khi có điều kiện hai góc, ta tiến hành vẽ ra vùng hoạt động của cánh tay robot. Đầu tiên ta cần xét tại vị trí đặc biệt $z = 0$. Khi $z = 0$, ta có được $d = 19cm$, mà $a_2 + a_3 = 10.5 + 13 = 23.5cm > 19cm$, suy ra cánh tay khi đạt $z = 0$ sẽ không thể duỗi thẳng mà sẽ hơi gấp lại, dựa trên phương pháp tính động học nghịch được đề cập ở *mục 3.2.5,* ta có thể tính được bảng sau:

x	y	z	θ_1	θ_2	θ_3	θ_4	θ_5
0	19	0	90^o	124.85^o	124.7758^o	179.9258^o	0^o

Bảng 3.3. Các góc khớp tại z = 0.

Từ *Bảng 3.3* dẫn đến ta sẽ cần cập nhật lại giá trị cận trên của θ_2 là:

$$90^o < \theta_2 < 180^o.$$

Để vẽ vùng hoạt động của cánh tay robot, ta sẽ sẽ cần xác định hai đường cong gồm biên ngoài và biên trong. Tại mỗi giá trị θ_2, ta sẽ vẽ được 1 đường cung là giới hạn hoạt động của θ_3 (đường - - - - trong *Hình 3.3*). Ứng với mỗi cung, ta cần xác định 2 điểm, điểm ứng với z lớn nhất của cung và điểm ứng với z nhỏ nhất của cung. Hai điểm này tương đương với hai ngưỡng giới hạn về góc của θ_3. Để vẽ biên ngoài ta sẽ nối tất cả các điểm có z lớn nhất ứng với mỗi giá trị θ_2. Để vẽ biên trong ta sẽ nối tất cả các điểm có z nhỏ nhất ứng với mỗi giá trị θ_2.

Mỗi biên sau khi nối lại sẽ tạo 2 đường cong biên ngoài (đường - - - - trong *Hình 3.3*) và biên trong (đường - - - - trong *Hình 3.3*).

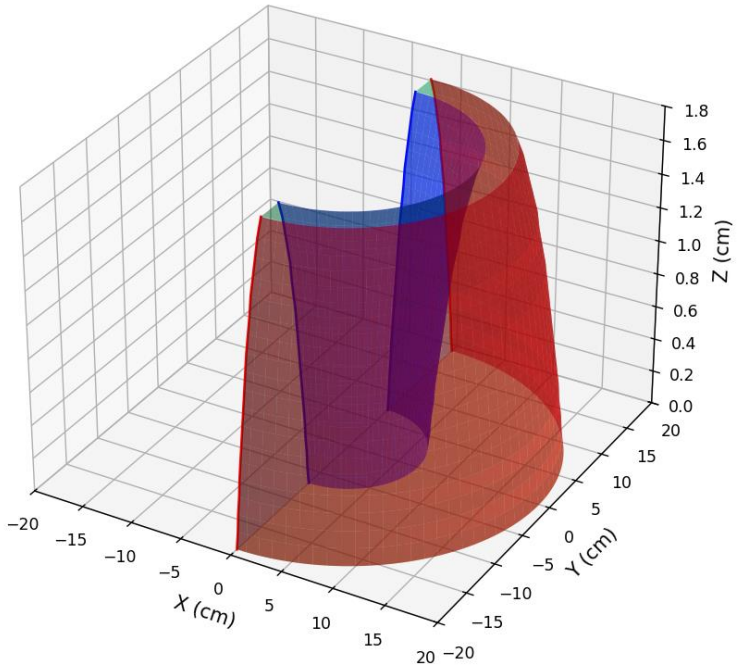
Tuy nhiên điểm kết thúc của biên ngoài sẽ không tuân theo quy luật đã được đề cập bên trên mà nó sẽ là điểm tại [z = 0; d = 19].

Sau khi quét qua các góc thỏa điều kiện tồn tại như đã kiểm tra, ta thu được *Bảng 3.4,* cho ta biết được tại mỗi tầng giá trị z ta sẽ thu được các điều kiện về d giúp tồn tại các nghiệm về góc.

d_{max}	d_{min}	z
19	7	0
18.8	7.2	0.1

18.7	7.3	0.2
18.5	7.5	0.3
18.4	7.6	0.4
18.2	7.8	0.5
18	8	0.6
17.8	8.2	0.7
17.6	8.4	0.8
17.4	8.6	0.9
17.2	8.8	1
17	9	1.1
16.7	9.3	1.2
16.4	9.6	1.3
16.2	9.8	1.4
15.8	10.2	1.5
15.4	10.6	1.6
15	11	1.7
14.4	11.6	1.8
13	0	1.9

Bảng 3.4. Giá trị d và z.



Hình 3.4. Không gian làm việc của robot khi gắp thẳng đứng.

3.2.4. Tính toán động học thuận của cánh tay robot

Để tính toán động học thuận của cánh tay robot, ta cần nhớ công thức để tính ma trận chuyển tiếp như sau

$$A_i = \begin{vmatrix} c_i & -c_{\alpha_i} \cdot s_i & s_{\alpha_i} \cdot s_i & a_i \cdot c_i \\ s_i & c_{\alpha_i} \cdot c_i & -s_{\alpha_i} \cdot c_i & a_i \cdot s_i \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

Trong đó: $c_i = \cos(\theta_i), s_i = \sin(\theta_i), c_{\alpha_i} = \cos(\alpha_i), s_{\alpha_i} = \sin(\alpha_i)$.

Dựa vào các thông số trên bảng DH cho cánh tay robot 5 bậc tự do:

$$A_1=\begin{vmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{vmatrix},\qquad A_2=\begin{vmatrix} c_2 & -s_2 & 0 & a_2.c_2 \\ s_2 & c_2 & 0 & a_2.s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix},\qquad A_3=\begin{vmatrix} c_3 & -s_3 & 0 & a_3.c_3 \\ s_3 & c_3 & 0 & a_3.s_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

,

$$A_4=\begin{vmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix},\qquad A_5=\begin{vmatrix} c_5 & -s_5 & 0 & 0 \\ s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

$$T_5=A_1.A_2.A_3.A_4.A_5,$$

$$T_5=\begin{vmatrix} c_1.c_{234}.c_5+s_1.s_5 & -c_1.c_{234}.s_5+s_1.c_5 & -c_1.s_{234} & c_1.\left(-d_5.s_{234}+a_3.c_{23}+a_2.c_2\right) \\ c_1.c_{234}.c_5-s_1.s_5 & -s_1.c_{234}.s_5-c_1.s_5 & -s_1.s_{234} & s_1.\left(-d_5.s_{234}+a_3.c_{23}+a_2.c_2\right) \\ -s_{234}.c_5 & s_{234}.s_5 & -c_{234} & d_1-a_2.s_2-a_3.s_{23}-d_5.c_{234} \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

Trong đó: $c_{ijk}=\cos\left(q_i+q_j+q_k\right), s_{ijk}=\sin\left(q_i+q_j+q_k\right).$

3.2.5. Tính toán động học nghịch của cánh tay robot

Đối với bài toán động học nghịch (vấn đề cần quan tâm trong đồ án này), yêu cầu đặt ra sẽ là ta cần tìm góc của các khớp sao để thỏa mãn với vị trí và hướng xoay được cho bởi ma trận:

$$H=\begin{vmatrix} n_x & O_x & a_x & p_x \\ n_y & O_y & a_y & p_y \\ n_z & O_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{vmatrix}.$$

Để giải quyết vấn đề trên ta sẽ có hai hướng tiếp cận: phương pháp đại số và phương pháp hình học.

Phương pháp đại số

Phương pháp này sẽ chủ yếu là đồng nhất hệ số khi ta cho hai ma trận bằng nhau để giải:

$$T_5 = H.$$

Có thể thấy để giải được ra các góc, ta phải giải 1 hệ 5 phương trình 5 ẩn vô cùng phức tạp. Điều này dẫn đến 1 giải pháp hay hơn là giải bằng hình học.

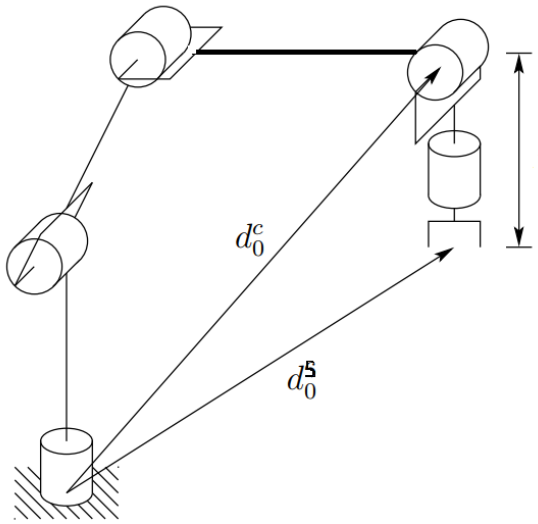
Phương pháp hình học

Trước tiên ta cần biết 1 điều, chỉ có 3 khớp đầu tiên sẽ có trách nhiệm với tọa độ vị trí điểm cuối, còn 2 khớp còn lại sẽ ảnh hướng đến hướng xoay. Vì thế ta sẽ áp dụng phương pháp Kinematic Decoupling để tách ra làm 2 phần để giải.

- Giải động học nghịch cho vị trí (tìm góc của 3 khớp đầu tiên)

Từ ma trận H ta có được các tọa độ x, y, z của vật tương ứng là p_x, p_y, p_z.

Vì tọa độ mà 3 khớp đầu tiên khi xoay mang lại sẽ là tọa độ của khớp thứ 4 nên khi giải ta sẽ lấy tọa độ thực biến đổi để về tọa độ khớp thứ 4 rồi mới giải



Hình 3.5. Kinematic Decoupling.

$$\boldsymbol{x} = \boldsymbol{p}_x - \boldsymbol{d}_5 \cdot \boldsymbol{r}_{13},$$
$$\boldsymbol{y} = \boldsymbol{p}_y - \boldsymbol{d}_5 \cdot \boldsymbol{r}_{23},$$
$$\boldsymbol{z} = \boldsymbol{p}_z - \boldsymbol{d}_5 \cdot \boldsymbol{r}_{33}.$$

Trong đó: r_{ij} là phần tử hàng i cột j trong ma trận xoay.

Tuy nhiên đối với dự án này, vì là gắp thẳng đứng nên x và y của vật cũng chính là x và y của khớp thứ 4. Để tìm z, ta cần xác định r_{33}, khi gắp vuông góc với mặt sàn, trục z của end-effector sẽ hướng cùng phương ngược chiều với trục z của hệ tọa độ gốc nên ta có $r_{33} = -1$. Vì vậy, ta có thể kết luận:

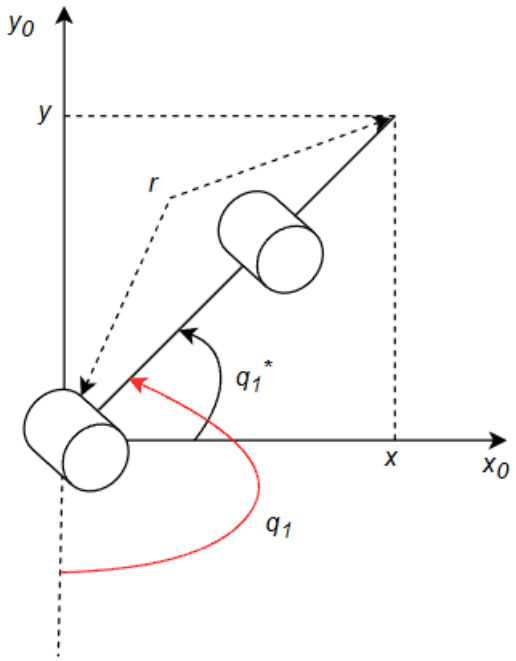
$$\begin{cases} \boldsymbol{x} = \boldsymbol{p}_{\mathbf{x}} \\ \boldsymbol{y} = \boldsymbol{p}_{\mathbf{y}} \\ \boldsymbol{z} = \boldsymbol{p}_{\mathbf{z}} + \boldsymbol{d}_5 \end{cases}$$

(*) được dùng để ký hiệu các biến khi vẽ hình và thực đặt trục tọa độ theo lý thuyết.

Trên thực tế, mỗi động cơ servo trước khi lắp vào mô hình cần phải được preset một góc 90 độ. Tùy vào các đặt trục so với thực tế, ta có thể quy đổi các góc từ lý thuyết sang thực tế như *Bảng 3.5*.

θ_1	θ_2	θ_3	θ_4	θ_5
θ_1^* + 90º	180º - θ_2^*	180º - θ_3^*	180º - θ_4^*	θ_5^* + 90º

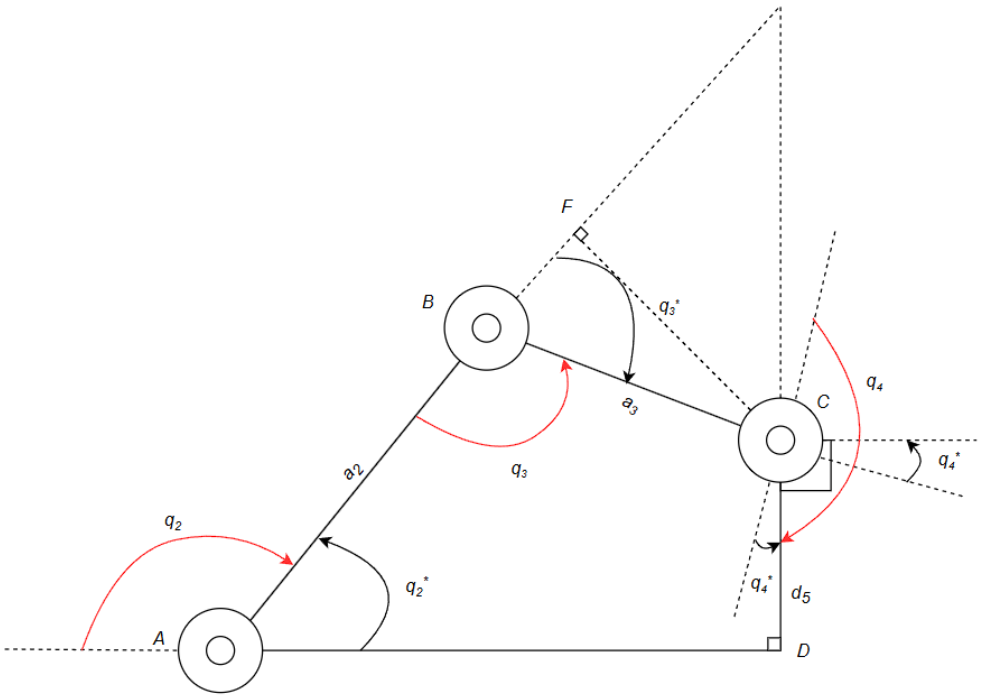
Bảng 3.5. Bảng quy đổi tọa độ thực.



Hình 3.6. Góc chiếu thẳng đứng từ trên xuống của cánh tay robot.

Từ hình trên ta tìm được góc thứ nhất của cánh tay robot bằng phương trình

$$\theta_1 = arctan2(y, x),$$



Hình 3.7. Góc nhìn ngang của cánh tay robot.

Từ *Hình 3.7*, ta phân tích để giải ra các góc khớp như sau:

$$r = \sqrt{x^2 + y^2}\,,$$

$$s = z - d_1\,,$$

$$\cos\left(ABC\right) = \frac{r^2 + s^2 - {a_2}^2 - {a_3}^2}{-2.a_2.a_3} := D\,,$$

$$\rightarrow \theta_3 = arctan2\left(\sqrt{1-D^2}, D\right),$$

$$BAD = \pi - \theta_2\,,$$

$$BAD = arctan2\left(s, r\right) + arctan2(a_3.sin\left(\pi - \theta_3\right), a_2 + a_3.cos\left(\pi - \theta_3\right))\,,$$

$$\rightarrow \theta_2 = \pi - arctan2\left(s, r\right) - arctan2(a_3.sin\left(\pi - \theta_3\right), a_2 + a_3.cos\left(\pi - \theta_3\right))\,,$$

$$\theta_4 = 2\pi - \frac{\pi}{2} - \theta_3 - BAD,$$

- Giải động học nghịch cho hướng (tìm góc của 2 khớp cuối)

Dựa trên các góc đã tìm được thông qua động học nghịch vị trí, ta thay vào để tìm ma trận R_3^0

$$R_3^0 = R_1^0 \cdot R_2^1 \cdot R_3^2 = \begin{bmatrix} c_1 & 0 & -s_1 \\ s_1 & 0 & c_1 \\ 0 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} c_2 & -s_2 & 0 \\ s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_3 & -s_3 & 0 \\ s_3 & c_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_1 \cdot c_{23} & -c_1 \cdot s_{23} & -s1 \\ s_1 \cdot c_{23} & -s_1 \cdot s_{23} & c1 \\ -s_{23} & -c_{23} & 0 \end{bmatrix}.$$

Ma trận R_5^3 sẽ được tính như sau:

$$R_5^3 = R_4^3 \cdot R_5^4 = \begin{bmatrix} c_4 & 0 & -s_4 \\ s_4 & 0 & c_4 \\ 0 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} c_5 & -s_5 & 0 \\ s_5 & c_5 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_4 \cdot c_5 & -c_4 \cdot s_5 & -s_4 \\ c_5 \cdot s_4 & -s_4 \cdot s_5 & c_4 \\ -s_5 & -c_5 & 0 \end{bmatrix}.$$

Để giải hai khớp cuối của cánh tay robot ta dựa trên phương trình sau:

$$R_5^3 = \left(R_3^0\right)^T \cdot R.$$

$$\text{Với } R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}.$$

Từ phương trình trên, ta tiến hành tính các biến:

$$-s_4 = s_{23},$$

$$c_4 = -c_{23},$$

$$-s_5 = -s_1,$$

$$-c_5 = c_1.$$

Sau khi có các biến trên, ta có thể tìm được θ_4^* và θ_5^* :

$$\theta_4* = arctan2(s_4, c_4),$$

$$\theta_5* = arctan2(s_5, c_5).$$

Có các góc trên theo lý thuyết, ta dựa vào Bảng 3.5 để quy đổi ra các góc thực.

Tuy nhiên giải pháp này đảm bảo về tính tổng quát nhưng không tối ưu với trường hợp đặc biệt là bài toán của đề tài này. Nhận thấy ma trận góc xoay R sẽ luôn không đổi vì hướng gắp sẽ cố định, ta có thể tìm mối liên kết giữa θ_4 và các biến θ_2 ; θ_3 dựa vào *Hình 3.7,* ta có thể tính ra các góc hướng như sau:

$$\theta_4 = \frac{\pi}{2} - \theta_3 + \theta_2,$$

$$\theta_5 = 0 + \frac{\pi}{2}.$$

Vì vật cần gắp là nắp chai hình tròn không có hướng xoay trong mặt phẳng Oxy nên ta có thể cho cố định góc θ_5 như trên.

3.2.6. Hoạch định quỹ đạo

Hoạch định quỹ đạo là thiết kế quỹ đạo chuyển động cho cánh tay robot. Để xác định được đường đi mong muốn của cánh tay robot theo thời gian, quỹ đạo có thể được tính toán và thiết kế trong hệ tọa độ truyền thống Oxyz (Cartesian Space) hoặc là trong không gian chứa các biến khớp (Configuration Space).

Trong thiết kế quỹ đạo cho cánh tay robot dựa trên các ứng dụng của chúng, ta thường gặp hai trường hợp sau:

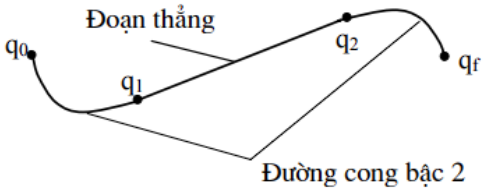
- Trường hợp 1: Đối với các ứng dụng chỉ quan tâm chủ yếu đến vị trí và hướng tại đích đến, điển hình là các công việc gắp thả của cánh tay

robot,… Khi đó, các điểm đầu và cuối sẽ là những điểm quan trọng của hành trình, còn dạng đường đi sẽ chỉ là vấn đề thứ yếu.

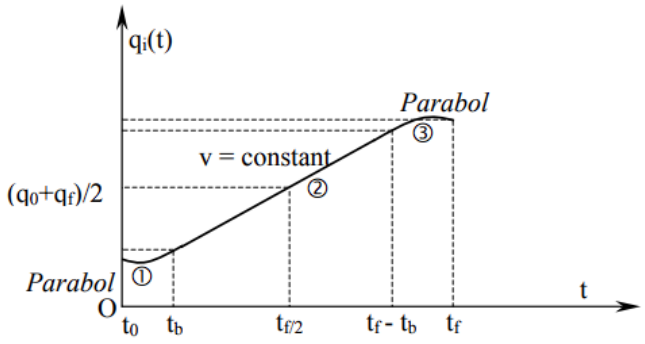
- Trường hợp 2: Đối với các ứng dụng quan tâm đến hành trình của cánh tay robot, phải luôn xác định đúng các điểm trên quỹ đạo xác định theo thời gian thực, có thể kể đến là sơn, hàn, cắt kim loại,…Vấn đề thiết kế quỹ đạo trong các trường hợp này là rất quan trọng. Quỹ đạo sẽ là đường sơn, vết hàn hay nét cắt trên sản phẩm, do đó nó quyết định trực tiếp đến chất lượng đầu ra của sản phẩm.

Đối với đồ án này, ta chỉ cần quan tâm đến ứng dụng của trường hợp 1. Quỹ đạo sẽ được thiết kế trong không gian chứa các biến khớp (Configuration Space), tức là xác định quy luật chuyển động của các biến khớp sao quỹ đạo của biến khớp sau khi hoàn thành chuyển động giống với quỹ đạo đã xác định.

Quỹ đạo được chọn để thiết kế ở đây là quỹ đạo LSPB: phối hợp đa thức bậc 2 với đa thức bậc 1.



Hình 3.8. Quỹ đạo LSPB.



Hình 3.9. Quỹ đạo LSPB của góc theo thời gian.

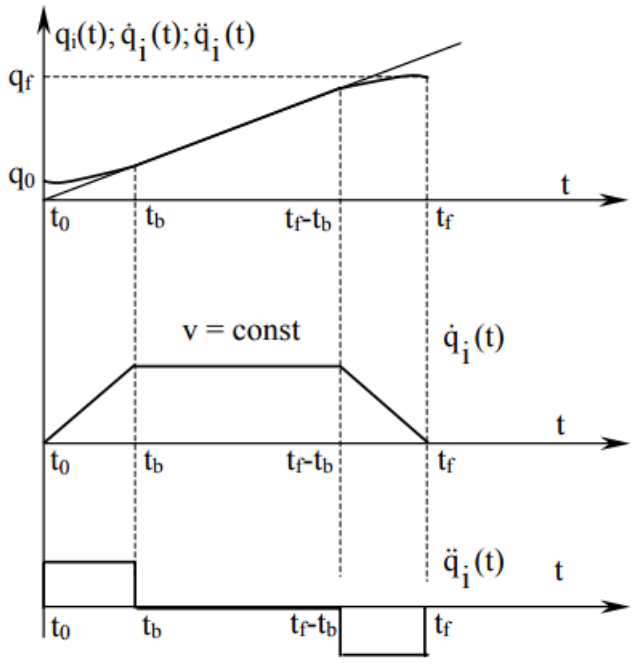
LSPB sẽ quy hoạch quỹ đạo góc theo ba giai đoạn:

Giai đoạn 1: Góc quay sẽ thay đổi nhanh dần, có quỹ đạo là đa thức bậc 2, dạng là đường parabol, vận tóc góc sẽ tăng đều tuyến tính theo gia tốc góc, gia tốc góc sẽ là + a_{max}.

Giai đoạn 2: Góc quay sẽ thay đổi tuyến tính với vận tốc góc không đổi, vì vận tóc góc không đổi nên gia tốc góc sẽ bằng 0.

Giai đoạn 3: Góc quay sẽ thay đổi chậm dần, có quỹ đạo là đa thức bậc 2, dạng là đường parabol, vận tóc góc sẽ giảm đều và tuyến tính theo gia tốc góc, gia tốc góc sẽ là - a_{max}.

Các đặc tính chuyển động theo quỹ đạo LSPB của khớp q_i như sau:



Hình 3.10. Đặc tính quỹ đạo LSPB.

Bài toán đặt ra với các tham số đầu vào cho trước như sau:

q_0 : Góc hiện tại.

q_f : Góc tới mục tiêu.

t_f : Thời gian đạt đến góc mục tiêu.

$v_{\max}$: Tốc độ góc tối đa đạt được.

$a_{\max}$: Gia tốc góc tối đa.

Phương trình quỹ đạo góc theo thời gian như sau:

$$q(t) = \begin{cases} q_0 + \frac{1}{2} \cdot a \cdot t^2, & 0 < t \le t_b \\ q(t_b) + v_{\max} \cdot (t - t_b), & t_b < t \le t_b + t_{const} \\ q(t_b) + v_{max} \cdot (t_{const}) + v_{max} \cdot (t - t_2) - \frac{1}{2} \cdot a_{max} \cdot (t - t_2)^2, & t_b + t_{const} < t \le t_f \end{cases}$$

Phương trình vận tóc góc theo thời gian như sau:

$$v(t) = \begin{cases} at, & 0 < t \le t_b \\ v_{max}, & t_b < t \le t_b + t_{const} \\ v_{max} - a \cdot (t - t_2), & t_b + t_{const} < t \le t_f \end{cases}$$

CHƯƠNG 4. GIẢI THUẬT XỬ LÝ ẢNH

Trong chương này, ta sẽ đi sâu vào những phương pháp để xử lý ảnh đầu vào để có thể cung cấp đầu ra là tọa độ vật cần gắp. Mục tiêu của chương là xử lý từng frame ảnh theo thời gian thực và dùng giải thuật để khớp các đặc trưng với vật mẫu. Bên cạnh đó chương cũng sẽ đưa ra các giải pháp tinh chỉnh hệ tọa độ vật sao cho đúng với hệ tọa độ thực của cánh tay robot.

4.1. Thư viện và nền tảng lập trình

Phần mềm được phát triển dựa trên OpenCV, một thư viện mã nguồn mở mạnh mẽ dành cho xử lý ảnh và thị giác máy, bằng ngôn ngữ lập trình Python. Việc xây dựng và biên dịch mã nguồn được thực hiện trên trình soạn thảo Geany trong môi trường hệ điều hành Linux.

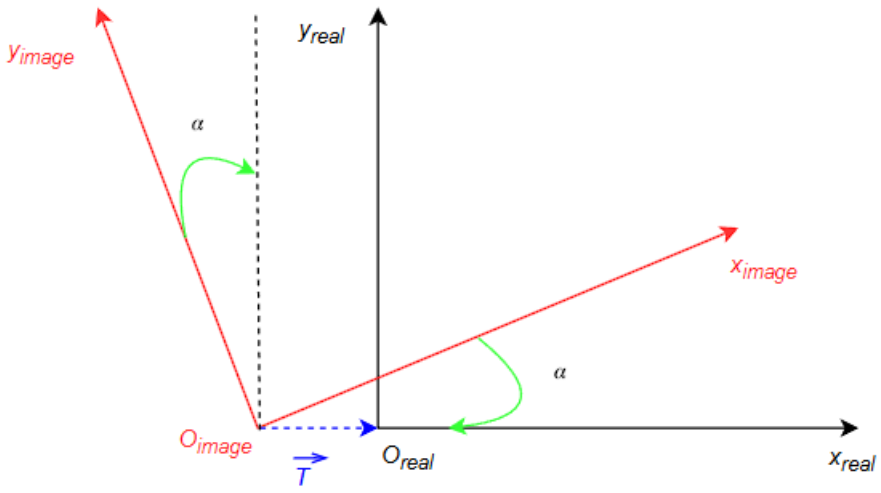
Geany là chương trình soạn thảo văn bản đa hệ sử dụng nền tảng GTK+ dựa trên Scintilla với Môi trường phát triển tích hợp (IDE) cơ bản. Nó được thiết kế giới hạn phụ thuộc vào các gói riêng lẻ để có thời gian nạp ngắn. Nó có mặt rộng rãi trên các hệ điều hành, như Windows, Linux, BSD và Solaris. Các ngôn ngữ lập trình được hỗ trợ (theo tài liệu) là C, Java, PHP, HTML, Python, Perl và Pascal.Geany là một trong những trình soạn thảo đầy đủ chức năng cho Linux, vì các trình soạn thảo văn bản cho Linux thường có mức đơn giản tối đa. Nó gần giống với các trình soạn thảo văn bản cho Windows như NoteTab hoặc ConTEXT. Geany là phần mềm tự do được cấp theo giấy phép GNU GPL.

Python là một ngôn ngữ lập trình cấp cao, đa mục đích. Triết lý thiết kế của nó nhấn mạnh khả năng đọc mã với việc sử dụng thụt lề có ý nghĩa. Python có kiểu động và tự động thu gom rác. Nó hỗ trợ nhiều mô hình lập trình, bao gồm lập trình có cấu trúc (đặc biệt là thủ tục), lập trình hướng đối tượng và lập trình hàm. Python thường được mô tả là ngôn ngữ "đầy đủ công cụ" nhờ thư viện chuẩn phong phú của nó.

4.2. Hiệu chỉnh hệ tọa độ camera với hệ tọa độ tay máy

Vì khi xử lý các thuật toán và đọc vật về bằng camera, tọa độ sẽ có đơn vị pixel tương ứng với vị trí của vật trên frame ảnh. Do đó để tay máy có thể hiểu được và di chuyển đến chính xác vị trí vật, ta cần calib hệ tọa độ camera sao cho quy đổi về đúng với hệ tọa độ thực của tay máy.

$$\begin{bmatrix} x_{real} \\ y_{real} \end{bmatrix} = S \cdot \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} \cdot \begin{bmatrix} x_{image} \\ y_{image} \end{bmatrix} + \begin{bmatrix} t_{11} \\ t_{21} \end{bmatrix}.$$

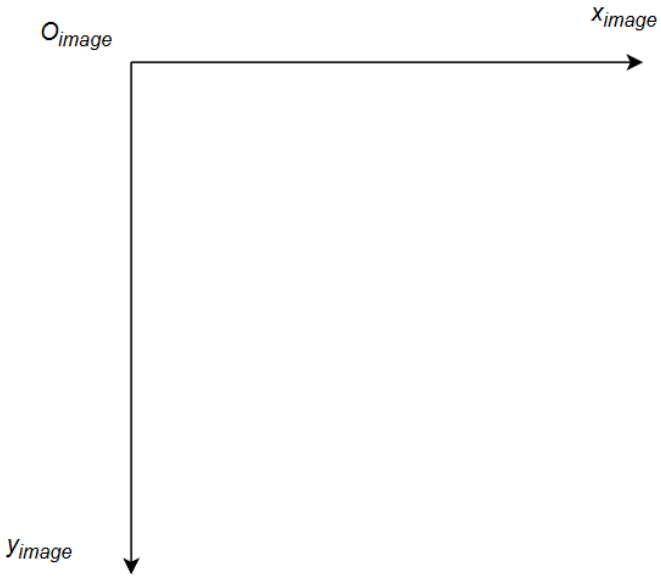


Hình 4.1. Hệ trục tọa độ ảnh và hệ trục tọa độ thực.

Hệ số S được tính bằng tỉ lệ giữa chiều rộng FOV và kích thước chiều rộng của frame ảnh dưới đơn vị pixel.

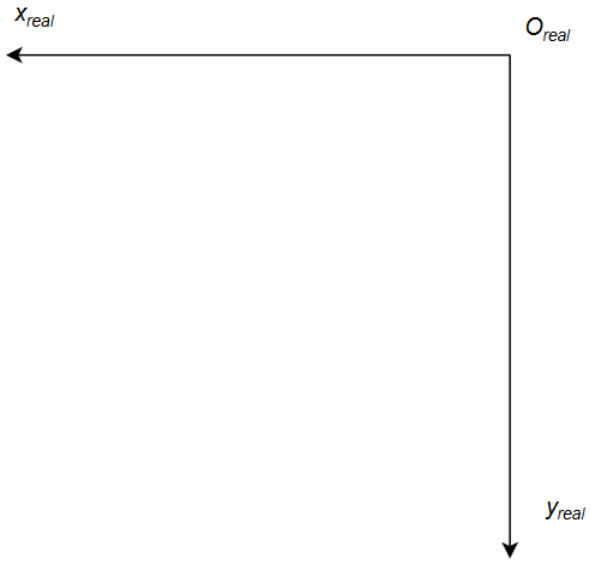
$$S = \frac{width(FOV)}{width(resolution)}.$$

Ma trận xoay R sẽ mô tả hệ tọa độ thực tế sẽ lệch như thế nào so với hệ tọa độ trong ảnh về góc. Đối với đồ án này, vì python khi xử lý tọa độ của khung ảnh hệ tọa độ ảnh sẽ có hình như sau:



Hình 4.2. Hệ tọa độ ảnh.

Hệ tọa độ thực tế sẽ như sau:



Hình 4.3. Hệ tọa độ thực.

Vì vậy ta sẽ tính ra được ma trận xoay R là: $\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$.

Ma trận dịch chuyển T sẽ mô tả độ dịch chuyển của gốc tọa độ so với gốc tọa độ trên ảnh.

4.3. Khớp vật với vật mẫu bằng giải thuật SURF

Thuật toán feature – matching SURF được dùng để phát hiện ra vật cần gắp trong đề tài này. Giải thuật tập trung vào việc khai thác và nhận diện các đặc trưng của một bức ảnh từ đó có thể tìm và match với những đặc trưng đang hiển thị theo thời gian thực, qua đó sẽ khoanh vùng được vật cần gắp phục vụ cho các bước tiếp sau đó.

SURF (Speeded Up Robust Features) là thuật toán được đề xuất bởi Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF là một phiên bản nâng cấp so với SIFT, với các ưu điểm là nhanh hơn và ổn định hơn khi nhận diện các đặc trưng của vật dưới góc nhìn khác hoặc bị ảnh hưởng bởi chuyển dịch và thay đổi.

SURF cũng tuân thủ theo các bước cơ bản như SIFT nhưng có đôi phần khác biệt khi ta đào sâu vào từng bước. SURF bao gồm ba bước chính:

- *Phát hiện điểm đặc trưng*

  Điểm đặc trưng thường được phát hiện dựa trên điểm đó có phải cực trị địa phương hay không. Trong SIFT, người ta dùng DoG để phát hiện ra các cực trị địa phương. Trong SURF, thay vì sử dụng Difference of Gaussian (DoG) như trong SIFT, thuật toán này dùng Fast-Hessian – một phương pháp nhanh hơn dựa trên determinant của ma trận Hessian để phát hiện điểm đặc trưng. Tại mỗi điểm ảnh, ta sẽ được cung cấp tọa độ p = (x, y), ma trận Hessian tại điểm p với mức scale σ:

$$H(p,\sigma)=\begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{yx}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix}.$$

Sau đó, ta sẽ tiến hành tính định thức ma trận để có thể đánh giá điểm tại đó:

$$\det(H) = L_{xx}.L_{yy} - (0.9.L_{xy})^2.$$

Hệ số 0.9 được thêm vào nhằm điều chỉnh ảnh hưởng của đạo hàm hỗn hợp L_{xy}, giúp giảm thiểu nhiễu từ các vùng không phải là đặc trưng rõ rệt.

- $det(H) > 0$ và $L_{xx} > 0$ → Cực trị là cực tiểu địa phương.
- $det(H) > 0$ và $L_{xx} < 0$ → Cực trị là cực đại địa phương.
- $det(H) < 0$ → Điểm đang xét là điểm yên ngựa.
- $det(H) = 0$ → Không thể kết luận điều gì.

Để tính toán ma trận Hessian tại một điểm trong ảnh, ta cần phải sử dụng các bộ lọc Gaussian bậc hai (second-order Gaussian derivatives) để tính toán các đạo hàm này. Tuy nhiên, việc tính toán các đạo hàm bậc hai trực tiếp có thể rất tốn thời gian. Thay vì sử dụng các bộ lọc Gaussian phức tạp, SURF sử dụng box filters để xấp xỉ các bộ lọc Gaussian bậc hai. Ảnh tích phân giúp tính toán các box filters này một cách nhanh chóng.

Trong SURF, mỗi điểm đặc trưng được mô tả bằng một vector đặc trưng dựa trên các đặc trưng Haar-like (tương tự như trong các thuật toán nhận diện khuôn mặt của Viola-Jones). Đặc trưng Haar-like là các phép toán tích phân trên các vùng ảnh hình chữ nhật. Với ảnh tích phân, ta có thể tính tổng của pixel trong một vùng hình chữ nhật chỉ bằng việc tham chiếu tới các

giá trị tại bốn góc của vùng đó. Điều này cho phép tính toán các đặc trưng Haar-like một cách nhanh chóng và hiệu quả mà không cần phải tính toán lại tổng của các pixel trong từng vùng cho mỗi điểm đặc trưng.

Công thức ảnh tích phân (integral image) sẽ có dạng như sau:

$$S(x,y) = \sum_{i=0}^{x}\sum_{j=0}^{y} I(i,j).$$

Trong SURF, không gian tỷ lệ được xây dựng bằng cách sử dụng box filters thay vì pyramid ảnh. Các lớp tỷ lệ được tạo ra bằng cách áp dụng bộ lọc có kích thước tăng dần (9×9, 15×15, 21×21, v.v.). Tỷ lệ bộ lọc được ước tính bằng công thức:

$$\sigma_{approx} = current\ filter\ size \times \left(\frac{base\ filter\ scale}{base\ filter\ size} \right).$$

- *Mô tả vùng lân cận cục bộ*

Mục tiêu của một descriptor (đặc trưng) là cung cấp một mô tả duy nhất và mạnh mẽ cho một đặc trưng ảnh, ví dụ, bằng cách mô tả phân bố cường độ của các điểm ảnh trong vùng lân cận của điểm quan tâm. Các descriptor thường được tính toán theo cách cục bộ, nghĩa là mỗi điểm quan tâm đã được xác định trước đó sẽ có một mô tả riêng.

Kích thước của descriptor có ảnh hưởng trực tiếp đến độ phức tạp tính toán và độ chính xác khi so khớp điểm. Một descriptor ngắn có thể giúp tăng tính mạnh mẽ trước sự thay đổi hình ảnh nhưng có thể không đủ phân biệt, dẫn đến nhiều trường hợp sai.

- *Xác định phương hướng*: Để đạt được tính bất biến khi xoay, phương hướng của điểm quan tâm cần phải được xác định. Các phản hồi của sóng Haar theo cả hướng x và y trong một vùng tròn có bán kính 6s xung quanh

điểm quan tâm được tính toán, trong đó s là tỉ lệ mà điểm quan tâm được phát hiện. Các phản hồi này được trọng số bằng một hàm Gaussian và được vẽ trên không gian hai chiều. Phương hướng chủ yếu được xác định bằng cách tính tổng tất cả các phản hồi trong cửa sổ hướng trượt có kích thước π/3. Vecto phương hướng dài nhất sẽ xác định phương hướng của điểm quan tâm.

- *Descriptor dựa trên tổng phản hồi sóng Haar*: Để mô tả vùng xung quanh điểm, một vùng vuông được cắt ra, trung tâm là điểm quan tâm và hướng theo phương hướng đã xác định trước đó. Kích thước cửa sổ này là 20s. Vùng quan tâm được chia thành các vùng con vuông 4x4, và tại mỗi vùng con, các phản hồi sóng Haar được tính tại các điểm mẫu đều 5x5. Các phản hồi này được trọng số bằng Gaussian để tăng tính mạnh mẽ đối với biến dạng, nhiễu và dịch chuyển.

- *Khớp đặc trưng*

Bằng cách so sánh các descriptors có được từ các bức ảnh khác nhau, các cặp giống nhau sẽ được ghép lại để hoàn tất việc khớp đặc trưng.

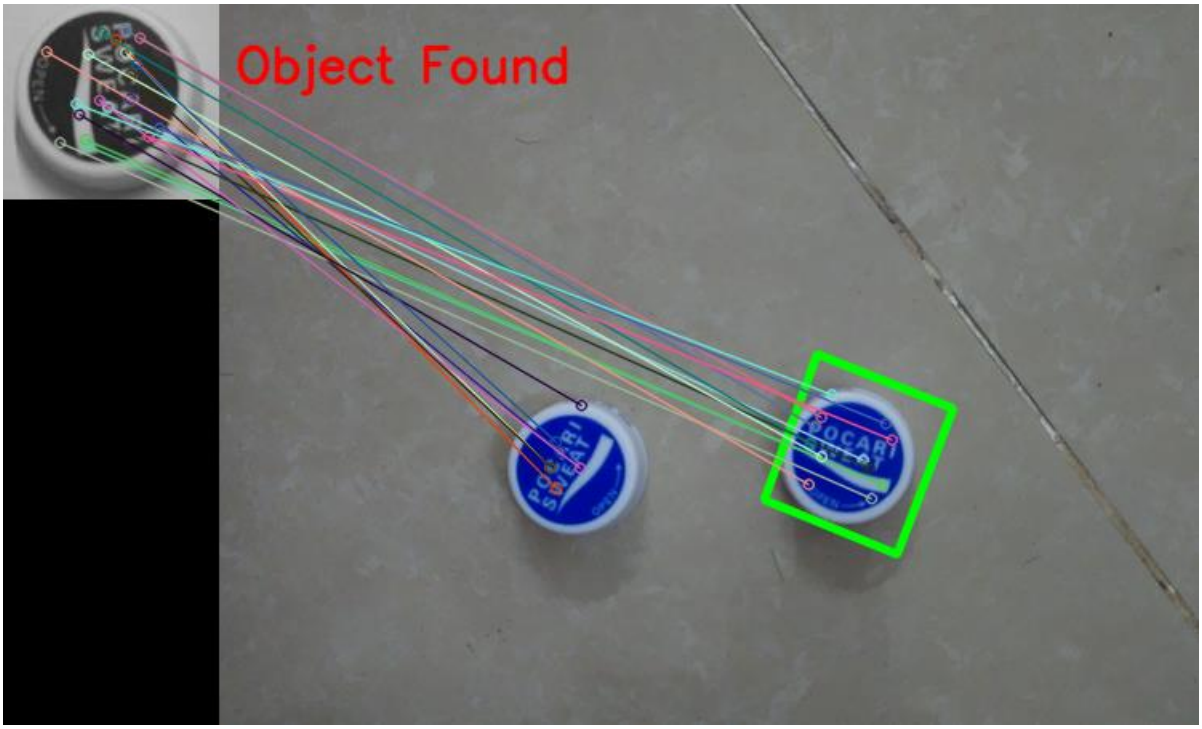
SURF (Speeded Up Robust Features) và SIFT (Scale-Invariant Feature Transform) là hai thuật toán phổ biến trong xử lý ảnh để phát hiện và mô tả các đặc trưng, nhưng SURF có một số ưu điểm so với SIFT:

- *Tốc độ xử lý nhanh hơn*: SURF được thiết kế để tối ưu hóa tốc độ so với SIFT. SURF sử dụng bộ lọc Haar wavelet trong tính toán các đặc trưng, giúp giảm đáng kể thời gian xử lý, đặc biệt là trong các ứng dụng thời gian thực.

- *Khả năng phát hiện đặc trưng mạnh mẽ hơn*: SURF cho phép phát hiện các đặc trưng có độ phân giải thấp hơn so với SIFT, giúp phát hiện được

nhiều đặc trưng hơn trong một ảnh. Điều này đặc biệt hữu ích khi xử lý ảnh có độ phân giải thấp hoặc hình ảnh bị mờ.

- *Khả năng chống nhiễu tốt*: SURF có khả năng chống nhiễu tốt hơn SIFT trong nhiều trường hợp, đặc biệt là khi có các thay đổi nhỏ về độ sáng hoặc độ tương phản.
- *Khả năng sử dụng vector mô tả ngắn hơn*: SURF tạo ra các vector đặc trưng ngắn hơn so với SIFT (64 chiều thay vì 128 chiều), giúp tiết kiệm tài nguyên khi lưu trữ và tăng tốc độ tính toán khoảng cách giữa các đặc trưng.



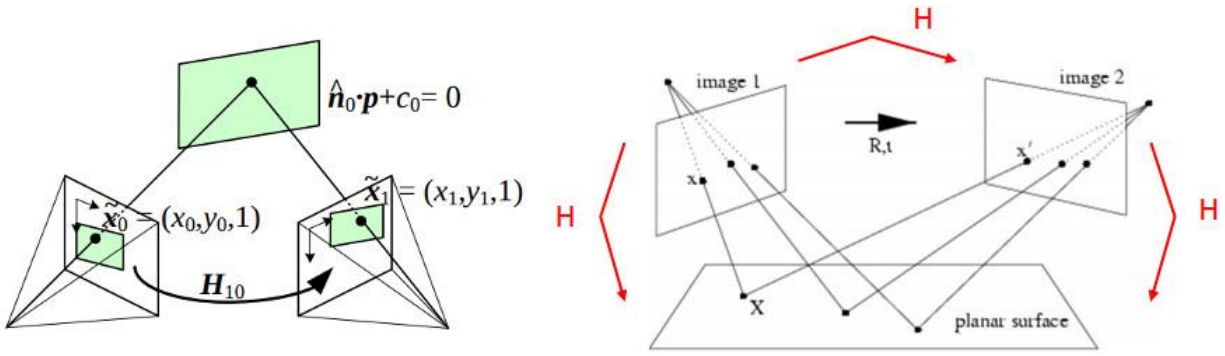
Hình 4.4. Trường hợp chạy thử thuật toán SURF.

SURF đã được dùng để tìm ra các đặc trưng của nắp chai và khớp các đặc trưng lại với nhau. Khi số đặc trưng tốt (good_matches) lớn hơn hoặc bằng một số lượng cố định, ta có thể xác định được vùng tập hợp các đặc trưng đó sẽ có khả năng cao là vật

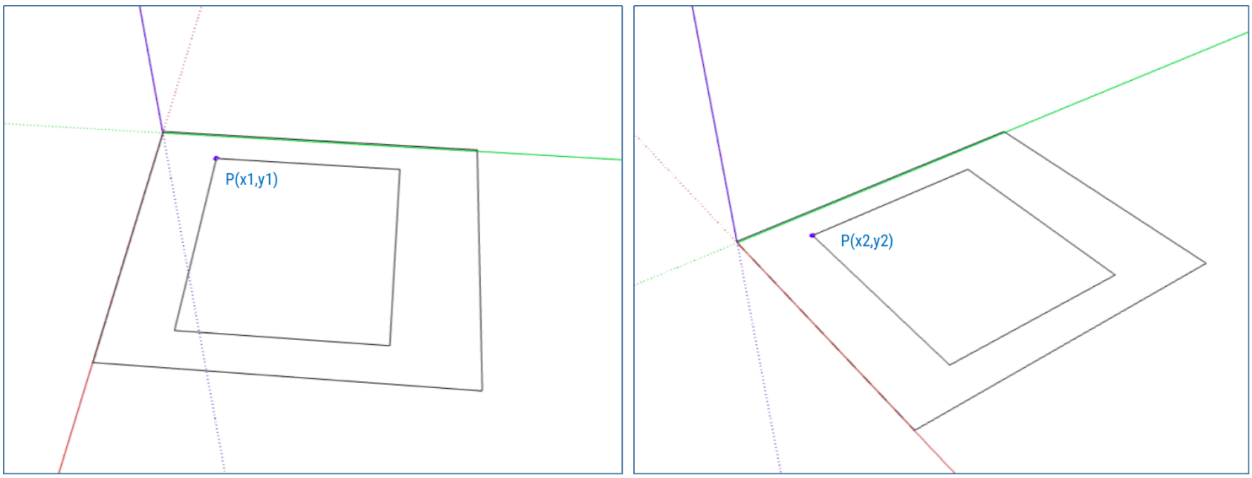
cần tìm, sau đó sẽ dùng phương pháp Homography được nêu ở mục 4.4 để có thể xác định đúng hình dạng cũng như vị trí vật.

4.4. Tìm homography

Vì trong khung hình video, các nắp chai không có vị trí hay hướng cố định, đồng thời ánh sáng ngoại cảnh cũng có thể tác động đến độ nhận diện, nên chúng ta sẽ áp dụng phương pháp ánh xạ hình ảnh từ ảnh mẫu sang ảnh trong khung hình để xác định vị trí, hướng, hoặc góc nhìn của camera so với vật. Để giải quyết vấn đề này, chúng ta có thể sử dụng homography, một phép biến đổi hình học mạnh mẽ giúp ánh xạ các điểm trong ảnh mẫu sang ảnh trong khung hình. Homography cho phép xác định mối quan hệ giữa các điểm trong không gian 2D của ảnh mẫu và ảnh thực tế, giúp chúng ta nhận diện được vị trí, góc quay và hướng của các nắp chai, bất chấp sự thay đổi về góc nhìn hay điều kiện ánh sáng. Bằng cách tính toán ma trận homography từ các điểm tương ứng giữa ảnh mẫu và ảnh trong khung hình, chúng ta có thể xác định chính xác vị trí của đối tượng và điều chỉnh góc nhìn của camera để cải thiện hiệu quả nhận diện.



Hình 4.5. Minh họa hoạt động của Homography.



Hình 4.6. Ví dụ cho ánh xạ của ảnh khi thay đổi góc nhìn.

Mục đích của homography là:

- *Ánh xạ điểm giữa hai ảnh*: Khi chúng ta có hai ảnh của cùng một đối tượng từ các góc nhìn khác nhau, homography giúp ánh xạ các điểm tương ứng giữa ảnh mẫu (reference image) và ảnh thực tế (input image).
- *Xử lý ảnh thay đổi phối cảnh*: Khi camera di chuyển hoặc đối tượng thay đổi góc nhìn, homography giúp chúng ta hiểu được sự thay đổi đó và điều chỉnh ảnh sao cho phù hợp.
- *Chuyển đổi mặt phẳng*: Mặt phẳng 2D trong ảnh mẫu có thể bị biến dạng khi nhìn từ một góc độ khác, và ma trận homography giúp chuyển các điểm trên mặt phẳng này sang một mặt phẳng khác trong ảnh.

Ma trận homography là một ma trận 3x3 có khả năng ánh xạ điểm từ một ảnh này sang ảnh khác. Để minh họa quá trình này, giả sử chúng ta có hai điểm trong hai ảnh khác nhau, điểm $P_1 = (x_1, y_1)$ trong ảnh 1 và điểm $P_2 = (x_2, y_2)$ trong ảnh 2, mối quan hệ giữa chúng có thể được mô tả bởi ma trận homography H.

Công thức ánh xạ điểm giữa hai ảnh được mô tả như sau:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = H \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}.$$

Trong đó:

$P_1 = (x_1, y_1)$ là tọa độ điểm trong ảnh mẫu.

$P_2 = (x_2, y_2)$ là tọa độ điểm tương ứng trong ảnh thực tế.

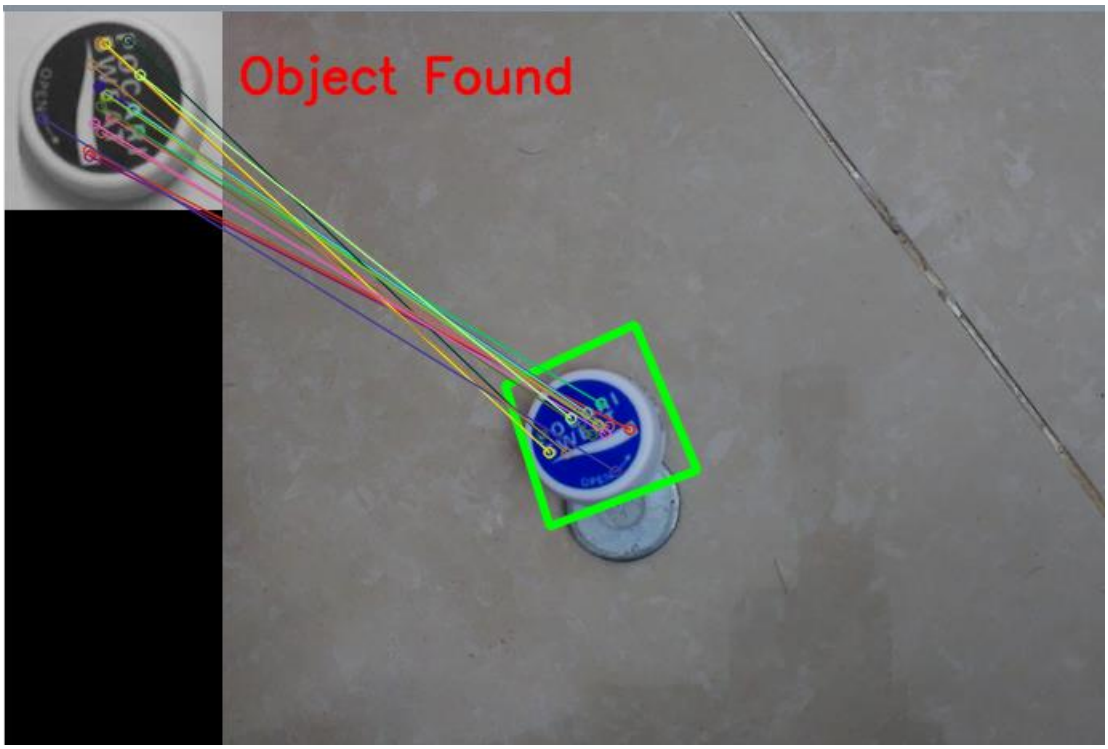
H là ma trận homography 3×3, với các phần tử như sau:

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}.$$

Điểm P_1 và P_2 sẽ có được từ trích xuất đặc trưng ảnh mẫu và ảnh lấy được từ mỗi frame của live video.

Dựa trên các điểm tương ứng, ta có thể tạo ra các hệ phương trình để tính các phần tử trong ma trận homography *H*. Quá trình này có thể được giải quyết bằng phương pháp least squares hoặc RANSAC (để loại bỏ các điểm nhiễu và xác định homography chính xác).

Sau khi có ma trận *H*, ta có thể sử dụng nó để chuyển đổi một điểm từ ảnh mẫu sang ảnh thực tế bằng cách áp dụng công thức ánh xạ ở trên. Quá trình này cho phép ta tìm kiếm các điểm tương ứng giữa các ảnh hoặc biến đổi ảnh mẫu sao cho khớp với ảnh thực tế.



Hình 4.7. Chạy thử chương trình áp dụng Homography khi vật bị chênh.

Ảnh trên cung cấp ví dụ cụ thể về sử dụng Homography để tìm ảnh ánh xạ. Nắp chai trong khung hình camera được chêm lên bởi một mảnh kim loại làm nắp chai không được đặt song song với mặt đất, tuy nhiên với ánh xạ thông qua ma trận Homography, nắp chai vẫn được khoanh vùng và trả về kết quả đúng như mong đợi.

4.5. Đọc bán kính và tọa độ vật

Sau khi chọn đúng được vật cần gắp, thông tin cần được đọc tiếp theo là tọa độ và bán kính của vật. Đây là giai đoạn cần thiết nhằm loại bỏ bớt những thông tin không quan trọng và nhiễu. Việc xử lý ở giai đoạn này sẽ đưa về dạng thông tin sơ cấp của hình ảnh với ảnh nhị phân và các đường biên nhằm phục vụ dễ dàng cho việc trích xuất tọa độ vật.

Giai đoạn này sẽ bao gồm các công việc như: loại bỏ nhiễu, tăng độ tương phản, trích xuất cạnh.

Box filter có thể giúp giảm nhiễu trong ảnh nhưng không hoàn toàn loại bỏ được. Đây là một loại bộ lọc trung bình, trong đó giá trị của mỗi điểm ảnh trong ảnh được tính bằng trung bình cộng của các giá trị của các điểm ảnh xung quanh nó trong một cửa sổ hình vuông hoặc hình chữ nhật (tùy vào kích thước của cửa sổ).

Kết quả của box filter là làm mượt ảnh và giảm các biến động đột ngột của độ sáng, giúp giảm bớt nhiễu ngẫu nhiên (như nhiễu salt-and-pepper). Tuy nhiên, nó không thể loại bỏ nhiễu một cách hoàn hảo và có thể làm mất chi tiết của ảnh, vì mỗi điểm ảnh mới là trung bình của các điểm ảnh xung quanh, dẫn đến làm mờ các cạnh của các đối tượng trong ảnh.

Điều chỉnh độ sáng là thay đổi đồng thời giá trị các mức xám của mỗi kênh màu trong ảnh lên một khỏng nhất định, hiển thị rõ các thông tin có mức sáng thấp đồng thời làm mờ các thông tin có mức sáng cao – là các thông tin không cần thiết trong phạm vị đề tài. Phép biến đổi độ sang được biểu diễn ở công thức

$$g(x,y) = \alpha \cdot f(x,y) + \beta .$$

Trong đó:

f, g là mức xám trước và sau phép biến đổi.

x, y là tọa độ pixel trên ảnh.

α, β là thông số của phép biến đổi.

Điều chỉnh độ tương phản là giãn cách sự phân bố mức xám của hai vùng là đối tượng và nền, nhằm tạo thành một histogram phân bố rõ ràng thành hai thái cực vùng mức xám, làm cho đối tượng nổi bật, dễ phát hiện và xử lý. Phép điều chỉnh độ tương phản được biển diễn ở công thức:

$$g(x,y)=\left(\frac{f(x,y)}{255}\right)^{\gamma}\cdot 255.$$

Trong đó:

f, g là mức xám trước và sau phép biến đổi.

x, y là tọa độ pixel trên ảnh.

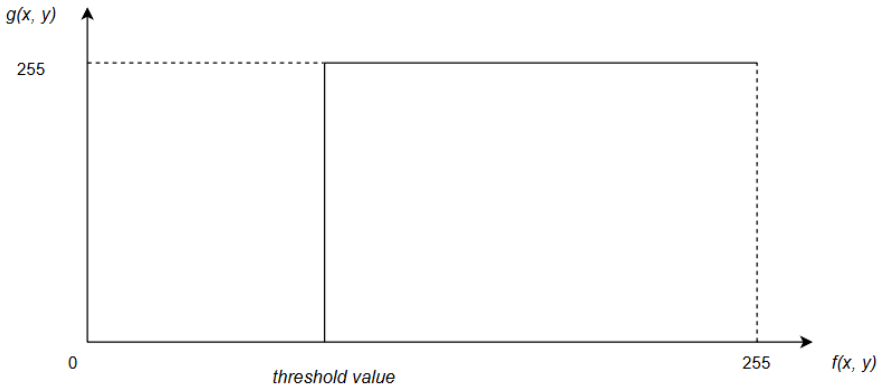
γ là thông số của phép biến đổi.

a) *b)* *c)*

Hình 4.8. a) Ảnh vật mẫu. b) Ảnh sau khi dùng Boxfilter. c) Ảnh sau khi được tăng độ tương phản.

Lấy ngưỡng ảnh là phương pháp trích xuất đặc trưng cơ bản của ảnh với yêu cầu đầu ra là cô lập và xác định được đối tượng cần trích xuất. Lấy ngưỡng ảnh có nhiều hình thức như lấy ngưỡng tĩnh, lấy ngưỡng thích nghi, lấy ngưỡng Otsu theo vùng,…

Trong phạm vi đề tài sử dụng lấy ngưỡng tĩnh với mức xám lấy ngưỡng là 75 do sau khi xử lý với các bước trước đó, ta đã thấy hầu như tách biệt được phần nền với vật do độ sáng được đẩy lên cao, các chi tiết cần quan tâm được làm bật lên trên nền trắng, do đó chỉ cần lấy ngưỡng tĩnh với mức ngưỡng trên đã có thể cô lập vật.



Hình 4.9. Ảnh lấy ngưỡng tĩnh.

Kết quả sau khi lấy ngưỡng:

Hình 4.10. a) Ảnh vật mẫu. b) Ảnh sau khi xử lý ảnh để chuẩn bị lấy đường bao.

Contours (đường viền) trong xử lý ảnh là các đường cong liên tục dùng để biểu diễn biên của các vật thể trong ảnh, thường được xác định dựa trên sự thay đổi mức xám từ nền sang viền. Điều này giúp đưa ra các ý tưởng về phát hiện đường viền dựa trên đạo hàm. Contours giúp xác định hình dạng, kích thước, và vị trí của các đối tượng, rất hữu ích trong các ứng dụng như phát hiện đối tượng, phân đoạn ảnh và theo dõi chuyển động.

Hàm `cv2.findContours` trong OpenCV được sử dụng để tìm contours trong một ảnh nhị phân. Hàm này nhận vào ảnh nhị phân và trả về danh sách các contours tìm được, mỗi contour là một tập hợp các điểm biên liên tiếp của một đối tượng trong ảnh. Cú pháp của hàm:

```
contours, hierarchy = cv2.findContours(image, mode, method)
```

Trong đó:

`image`: Ảnh nhị phân đầu vào (ảnh trắng đen), thường là ảnh kết quả sau khi áp dụng threshold hoặc edge detection.

`mode`: Chế độ lấy contours, ví dụ `cv2.RETR_EXTERNAL` (lấy các contours ngoài cùng) hoặc `cv2.RETR_TREE` (lấy toàn bộ contours theo cấu trúc phân cấp).

`method`: Phương pháp xấp xỉ contours, như `cv2.CHAIN_APPROX_SIMPLE` (giảm bớt các điểm dư thừa) hoặc `cv2.CHAIN_APPROX_NONE` (lưu toàn bộ điểm trên contour).

Khi có được các tọa độ của các điểm trên contour, trọng tâm của vật được tính bằng cách lấy trung bình tọa độ tất cả các pixel của vật như công thức:

$$center(x,y) = \frac{1}{n} \cdot \sum_{i=0}^{n-1} pixel_i(x,y).$$

Trong đó: n là tổng số pixel được tổng hợp thuộc đường viền.

Bán kính của vật được tính bằng trung bình khoảng cách từ trọng tâm đến các điểm ngoài đường bao của vật như công thức:

$$r = \frac{1}{n} \sum_{i=0}^{n-1} norm\left(center(x,y) - contour(x,y)\right).$$

Hình 4.11. Xác định đường bao của vật.

```
[0:48:07.119601545] [3284]  INFO Camera camera_manager.cpp:325 libcamera v0.3.2+
27-7330f29b
[0:48:07.151820312] [3306]  WARN RPiSdn sdn.cpp:40 Using legacy SDN tuning - ple
ase consider moving SDN inside rpi.denoise
[0:48:07.154275652] [3306]  INFO RPI vc4.cpp:447 Registered camera /base/soc/i2c
0mux/i2c@1/imx708@1a to Unicam device /dev/media3 and ISP device /dev/media0
[0:48:07.154508799] [3306]  INFO RPI pipeline_base.cpp:1126 Using configuration
file '/usr/share/libcamera/pipeline/rpi/vc4/rpi_apps.yaml'
[0:48:07.162329242] [3284]  INFO Camera camera.cpp:1197 configuring streams: (0)
 640x480-XRGB8888 (1) 1536x864-SBGGR10_CSI2P
[0:48:07.162875184] [3306]  INFO RPI vc4.cpp:622 Sensor: /base/soc/i2c0mux/i2c@1
/imx708@1a - Selected sensor format: 1536x864-SBGGR10_1X10 - Selected unicam for
mat: 1536x864-pBAA
Nhấn 'q' để thoát.
12.1
64
Gửi tọa độ x: 12.1, y: 8.2
```

Hình 4.12. Đọc tọa độ từ Hình 4.11.

CHƯƠNG 5: GIẢI THUẬT ĐIỀU KHIỂN

Trong chương này, chúng ta sẽ đi sâu vào thiết kế bộ điều khiển cho từng servo từ các góc quay đã được tính toán thông qua động học robot. Bên cạnh đó sẽ là giải thuật chung điều khiển toàn bộ hệ thống.

5.1. Tổng quan

Bộ điều khiển được lựa chọn cho servo trong nghiên cứu này là bộ điều khiển PID, nhằm cải thiện hiệu suất vận hành bằng cách giảm thiểu độ giật và rung động của động cơ trong quá trình di chuyển từ vị trí ban đầu đến vị trí đích. Việc điều khiển các động cơ được thực hiện thông qua mạch điều khiển PWM 16 kênh PCA 9685, giúp giảm tải cho vi điều khiển STM32 và tăng cường khả năng quản lý các động cơ.

Đối với phần cứng xử lý thuật toán điều khiển robot, vi điều khiển STM32 được sử dụng như "bộ não" trung tâm, đảm nhận vai trò điều khiển chuyển động của cánh tay robot. Phần mềm điều khiển được phát triển bằng ngôn ngữ lập trình C và biên dịch trên nền tảng STM32CubeIDE.

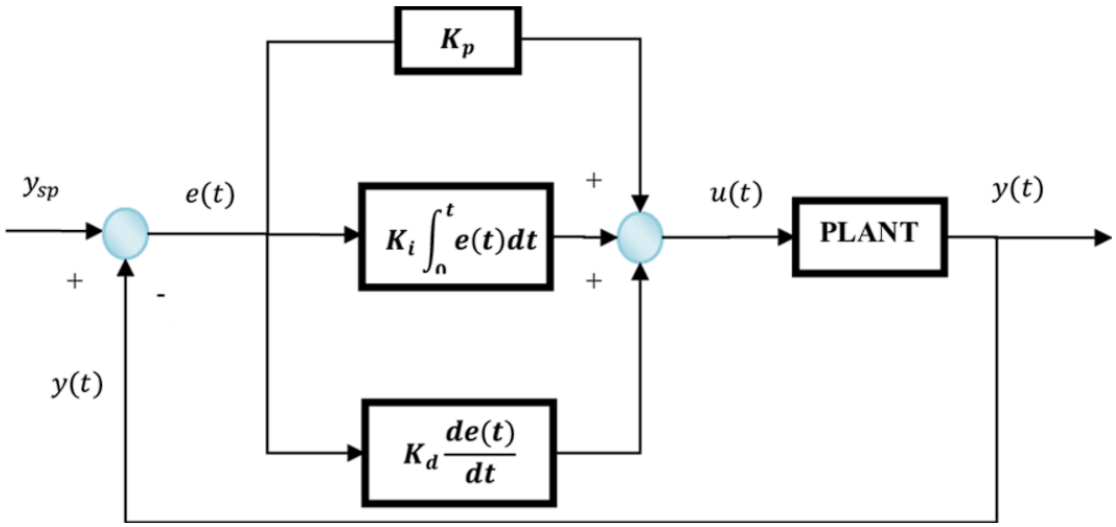
Hệ thống giao tiếp giữa STM32 và Raspberry Pi, chịu trách nhiệm trao đổi thông tin tọa độ và tín hiệu điều khiển, được thực hiện thông qua giao thức UART. Dữ liệu truyền nhận được tổ chức dựa trên một cấu trúc khung (frame) được thiết kế sẵn, đảm bảo tín hiệu được nhận đầy đủ và chính xác trước khi khởi động các hành động điều khiển. Các tọa độ được truyền ở dạng số thực, đòi hỏi xử lý mã hóa và giải mã dấu thập phân để duy trì độ chính xác của thông tin vị trí.

Do việc xử lý dữ liệu truyền nhận và điều khiển cánh tay robot đòi hỏi thời gian thực thi lớn, hiệu suất tổng thể của hệ thống có thể bị ảnh hưởng. Để giải quyết vấn đề này, nghiên cứu áp dụng hệ điều hành thời gian thực (RTOS) để quản lý luồng xử lý. RTOS là hệ điều hành được thiết kế đặc biệt cho các hệ thống nhúng nhỏ gọn, đảm bảo các tác vụ được thực thi trong khoảng thời gian xác định với độ trễ thấp và độ tin cậy cao. Nhờ khả năng quản lý đa nhiệm, phân bổ tài nguyên và thời gian CPU hiệu quả, RTOS giúp tối ưu hóa việc điều phối hai luồng xử lý trong nghiên cứu này, đáp ứng các yêu cầu thời gian thực mà không tiêu tốn quá nhiều tài nguyên phần cứng.

5.2. Bộ điều khiển PID

PID (Proportional-Integral-Derivative) là một thuật toán điều khiển phổ biến được sử dụng rộng rãi trong các hệ thống tự động hóa và điều khiển. Bộ điều khiển PID hoạt động dựa trên ba thành phần chính: Proportional (P) giúp giảm sai lệch tức thời giữa giá trị mong muốn và giá trị thực, Integral (I) tích lũy sai lệch để loại bỏ sai số lâu dài, và Derivative (D) dự đoán và giảm thiểu dao động bằng cách xem xét tốc độ thay đổi của sai lệch. Sự kết hợp linh hoạt giữa ba thành phần này giúp PID trở

thành một công cụ mạnh mẽ trong việc đảm bảo hệ thống hoạt động ổn định, nhanh chóng đạt được trạng thái cân bằng và đáp ứng tốt với các thay đổi đột ngột. Trong chương này, PID sẽ được ứng dụng để điều chỉnh hiệu suất của hệ thống, đảm bảo tính chính xác và ổn định trong quá trình vận hành.



Hình 5.1. Bộ điều khiển PID.

Trong đó:

PLANT hay đối tượng cần điều khiển ở đồ án này là động cơ servo.

SP hay giá trị đặt sẽ là xung PWM mục tiêu có được thông qua bài toán LSPB (được đề cập ở *Mục 3.2.6*).

Output hay $y(t)$ sẽ là xung PWM cấp cho động cơ tại thời điểm đang xét.

Công thức sẽ cung cấp sai số tại thời điểm t dùng để hiệu chỉnh thông qua bộ PID:

$$e(t) = y_{sp} - y(t).$$

Khâu tỉ lệ (P)

- Vai trò:
    - K_P là hệ số khuếch đại tỷ lệ, phản ánh mức độ phản ứng của hệ thống với sai lệch $e(t)$.
    - Giá trị K_P càng lớn thì phản ứng của hệ thống càng nhanh, giúp giảm thời gian ổn định và độ lệch ban đầu.
- Ảnh hưởng của K_P:
    - K_P thấp: Hệ thống phản ứng chậm, sai số ổn định (steady-state error) lớn.
    - K_P cao: Tăng tốc độ phản ứng nhưng có thể gây dao động hoặc mất ổn định nếu quá lớn.
- Lưu ý khi điều chỉnh K_P:

  Giá trị K_P chỉ giảm sai số tạm thời (transient error) mà không loại bỏ sai số duy trì (steady-state error).
- Công thức:

$$u_P(t) = K_P.e(t).$$

Khâu tích phân (I)

- Vai trò:
    - K_I giúp tích lũy sai lệch theo thời gian, đảm bảo sai số duy trì được loại bỏ hoàn toàn.
    - Thành phần này rất quan trọng trong các hệ thống yêu cầu độ chính xác cao.

- Ảnh hưởng của K_I:
    - K_I thấp: Quá trình loại bỏ sai số duy trì diễn ra chậm.
    - K_I cao: Có thể loại bỏ sai số nhanh hơn, nhưng dễ dẫn đến hiện tượng dao động và mất ổn định (wind-up effect).
- Lưu ý khi điều chỉnh K_I:

    Cần cẩn thận để tránh "hiện tượng bão hòa tích phân" (integral wind-up), khi giá trị tích phân tăng quá lớn, gây trễ lớn trong hệ thống.
- Công thức:

$$u_I(t) = K_I.\int_0^t e(\tau).d\tau.$$

Khâu vi phân (D):

- Vai trò:
    - K_D giúp dự đoán xu hướng thay đổi của sai lệch, nhờ đó giảm hiện tượng dao động và cải thiện độ ổn định.
    - Thành phần này đặc biệt hữu ích trong các hệ thống có đáp ứng nhanh hoặc chịu tác động mạnh từ nhiễu.
- Ảnh hưởng của K_D:
    - K_D thấp: Hệ thống có thể dao động mạnh hoặc chậm ổn định.
    - K_D cao: Tăng độ ổn định nhưng nhạy cảm với nhiễu, có thể gây hiện tượng "phóng đại nhiễu" (noise amplification).
- Lưu ý khi điều chỉnh K_D:

Chỉ nên sử dụng K_D khi hệ thống yêu cầu phản ứng mượt mà và giảm dao động, vì giá trị K_D lớn dễ bị nhiễu làm sai lệch.

- Công thức:

$$u_D(t) = K_D.\frac{de(t)}{dt}.$$

Đầu ra của bộ PID

$$u(t) = u_P(t) + u_I(t) + u_D(t)$$

$$\rightarrow u(t) = K_P.e(t) + K_I.\int_0^t e(\tau).d\tau + K_D.\frac{de(t)}{dt}.$$

Việc hiệu chỉnh các thông số của bộ PID sẽ quyết định chất lượng đầu ra của hệ thống, vì vậy cần làm rõ mối quan hệ giữa các thông số cũng như cách thay đổi bộ thông số đó sao cho hệ thống nhanh chóng đạt đến hiệu quả mong muốn.

Mối quan hệ giữa K_P, K_I và K_D

- Cân bằng hệ thống:
  - K_P quyết định mức độ phản ứng tức thời.
  - K_I quyết định khả năng loại bỏ sai số ổn định.
  - K_D quyết định sự mượt mà và ổn định của đáp ứng.
- Tác động tổng hợp:
  Mỗi tham số có vai trò riêng, nhưng sự kết hợp giữa chúng mới tạo nên hiệu quả của PID:
  - Nếu chỉ dùng K_P, hệ thống có thể nhanh nhưng thiếu chính xác.

- Nếu chỉ dùng K_I hệ thống loại bỏ sai số nhưng phản ứng chậm và dễ dao động.
- Nếu chỉ dùng K_D, hệ thống giảm dao động nhưng không loại bỏ được sai số duy trì.

- Phương pháp điều chỉnh các thông số PID:

Có một số phương pháp có thể đề cập như: phương pháp thủ công, phương pháp Ziegler-Nichols, phương pháp tối ưu hóa tự động. Tuy nhiên đối với đồ án này, việc xử lý bộ điều khiển động cơ servo không phải mục tiêu chính, vì thế phương pháp thủ công sẽ được lựa chọn nhằm giảm tải khối lượng công việc.

- Bắt đầu với $K_I = K_D = 0$.
- Tăng K_P đến khi hệ thống giao động nhẹ.
- Thêm K_I để loại bỏ sai số duy trì.
- Cuối cùng tăng K_D để giảm giao động.

5.3. Giải thuật truyền nhận

Giải thuật này được xây dựng dựa trên 3 phần chính:

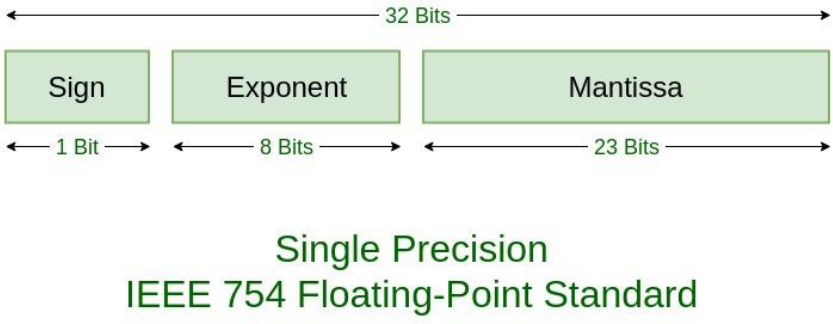
- Mã hóa thông tin.
- Frame truyền nhận dữ liệu.
- Bộ đệm.

5.3.1. Mã hóa thông tin

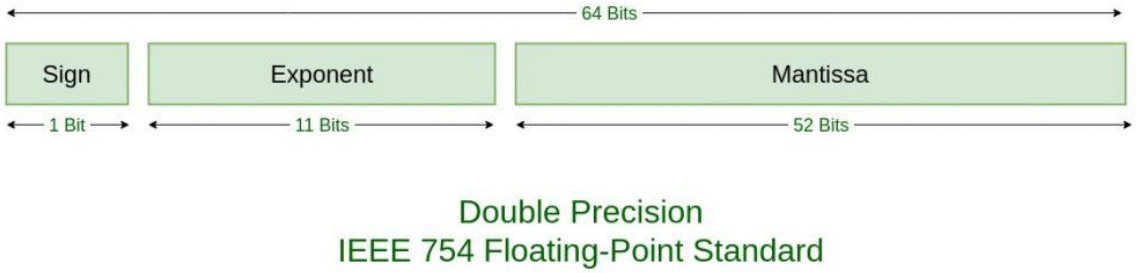
Thông tin cần xử lý khi truyền nhận là tọa độ của vật, bao gồm x và y. Đây là hai biến có kết quả là số thực, vì thế để có thể truyền nhận thông tin giao tiếp giữa Raspberry Pi và STM32, ta cần mã hóa đoạn thông tin này. Để mã hóa số thực ta cần

chọn một phương pháp hoặc một chuẩn, để từ quy tắc đó ta có thể đọc hoặc hiểu được thông tin sau mã hóa.

Chuẩn được sử dụng cho đồ án này là chuẩn IEEE 754. IEEE 754 là chuẩn quốc tế dùng để biểu diễn số thực trong máy tính và các hệ thống nhúng. Chuẩn này định nghĩa các cách biểu diễn số dấu phẩy động với độ chính xác đơn (single-precision) và độ chính xác kép (double-precision).



Hình 5.2. Số IEEE 754 với độ chính xác đơn.



Hình 5.3. Số IEEE 754 với độ chính xác kép.

Đối với đồ án này, dữ liệu cần đọc không quá phức tạp và không yêu cầu về chính xác tuyệt đối nên sẽ sử dụng số IEEE 754 với độ chính xác đơn. Số IEEE 754 với độ chính xác đơn được biểu diễn như sau:

Số 32-bit: 1-bit dấu, 8-bit mũ, 23-bit trị số.

$$(-1)^S.(1.M).2^{E-127}.$$

Trong đó:

- **S (Sign):** Bit dấu (0 cho số dương, 1 cho số âm).
- **E (Exponent):** Mũ, được mã hóa dạng dịch bias (bias = 127).
- **M (Mantissa):** Trị số, biểu diễn phần thập phân.

Ví dụ minh họa:

85.125

85 = 1010101

0,125 = 001

85,125 = 1010101,001

= 1,010101001 x 2^6 (tức dịch trái 6-bit)

Bit dấu (S) = 0

Số mũ lệch (E) 127 + 6 = 133

133 = 10000101

Mantisa chuẩn hóa (M) = 010101001

Chúng ta sẽ thêm 0 để hoàn thành 23 bit

Độ chính xác đơn IEEE 754 là: 0 10000101 01010100100000000000000

Điều này có thể được viết ở dạng thập lục phân **42AA4000**

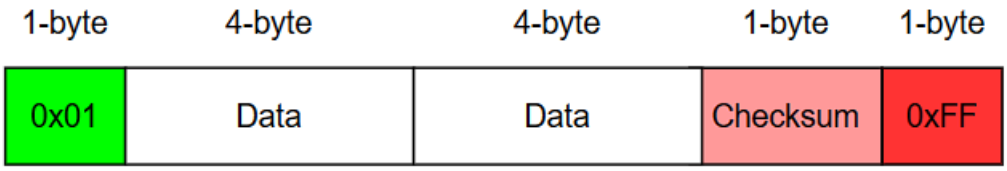
Khi truyền dữ liệu qua UART, cách sắp xếp byte (endianness) đóng vai trò quan trọng trong việc đảm bảo dữ liệu được giải mã chính xác ở đầu nhận. Hai chuẩn phổ biến nhất là Big-Endian và Little-Endian. Trong đồ án này, chuỗi truyền được sắp xếp theo Big-Endian. Trong ví dụ trên, khi dữ liệu được truyền theo Big-Endian (MSB trước) sẽ là:

Truyền byte theo thứ tự: 0x42, 0xAA, 0x40, 0x00.

5.3.2. Frame truyền dữ liệu

Mỗi khung (frame) trong giao thức UART chứa các thành phần quan trọng giúp dữ liệu được truyền chính xác, bao gồm bit dữ liệu, bit bắt đầu, bit dừng, và đôi khi là bit kiểm tra lỗi. Mục này sẽ trình bày chi tiết về cấu trúc của một khung dữ liệu trong giao thức UART và vai trò của từng thành phần trong quá trình truyền nhận dữ liệu trong đồ án.

Đối với trường hợp truyền đi từ Raspberry Pi sang STM32, ta có khung dữ liệu như sau:



Hình 5.4. Frame truyền dữ liệu.

Trong đó:

- Hai biến Data sẽ lần lượt là x và y sau khi được mã hóa theo chuẩn IEEE 754 32-bit
- Biến Checksum được thêm sau khi frame truyền đi đã điền xong biến y. Checksum sẽ tính tổng byte gồm byte bắt đầu và 8-byte dữ liệu.

Đối với STM32, sau khi nhận được frame thực hiện đọc byte bắt đầu và byte kết thúc trước, nếu bằng đúng như frame đã quy định trước đó, thì tiến hành đọc byte Checksum để so sánh với tổng số byte quy định nhận được.

5.3.3. Bộ đệm

Bộ đệm (buffer) là một phần không thể thiếu trong các hệ thống xử lý dữ liệu, giúp tạm thời lưu trữ dữ liệu để đồng bộ hóa tốc độ giữa các thành phần hoặc xử lý dữ liệu không đồng bộ. Tùy thuộc vào cách triển khai, bộ đệm có thể được thực hiện dưới dạng phần mềm hoặc phần cứng, với mỗi loại có những ưu điểm và nhược điểm riêng.

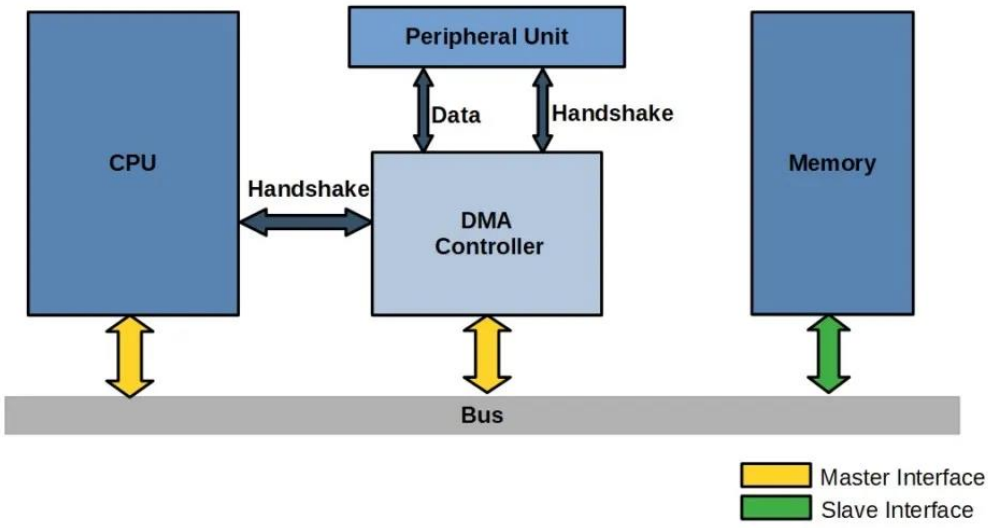
Bộ đệm phần cứng được sử dụng trong trường hợp này là:

UART FIFO (First In, First Out):

- Dữ liệu được nhận từ UART ban đầu sẽ được lưu trữ trong FIFO, một loại bộ đệm phần cứng.
- FIFO hoạt động độc lập và giúp giảm thiểu tình trạng mất dữ liệu khi tốc độ truyền nhận khác nhau.

DMA Controller:

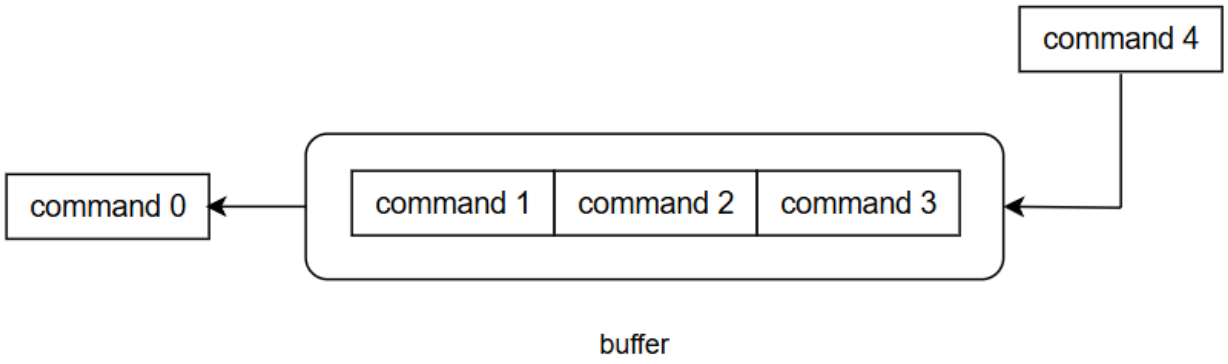
- DMA sử dụng bộ đệm phần cứng của nó để chuyển dữ liệu từ UART FIFO vào RAM (bộ đệm phần mềm) mà không cần CPU can thiệp.
- DMA đảm bảo rằng dữ liệu được truyền nhanh chóng và hiệu quả.



Hình 5.5. Hình thức hoạt động của DMA.

Bộ đệm phần mềm nhằm tạo không gian lưu trữ lệnh gửi xuống từ máy tính, xử

lý các lệnh được tuần tự và tránh mất mát lệnh.



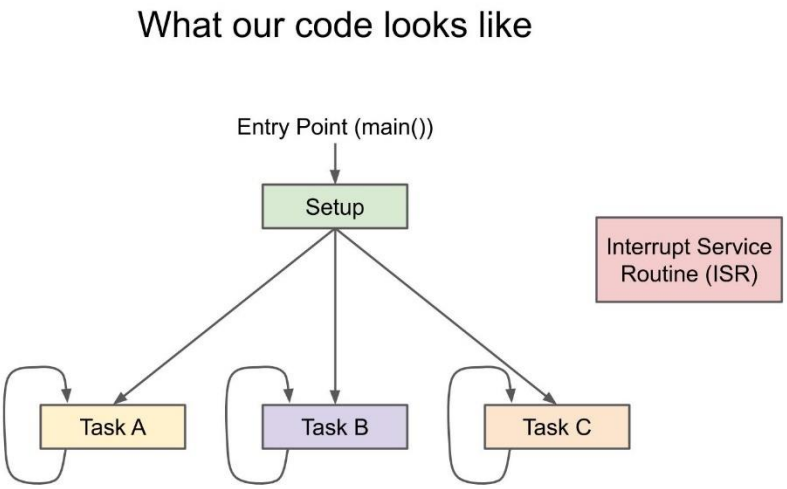
Hình 5.6. Bộ đệm phần mềm.

Mỗi lệnh hợp lệ (được kiểm tra ký tự bắt đầu và kết thúc) sẽ được lưu trong bộ đệm. Lệnh được lưu trữ theo quy tắc của hàng đợi – queue. Mỗi lệnh ở phía trước hàng đợi được xử lý sẽ giải phóng bộ nhớ và toàn bộ dữ liệu được dịch về trước. Dữ liệu chỉ được di chuyển theo một chiều từ sau bộ đệm ra trước bộ đệm như *Hình 5.6*.

5.4. Giao tiếp giữa hai luồng

Phần cứng được chia ra làm hai luồng xử lý chạy dưới FreeRTOS, là phiên bản hệ điều hành RTOS dành cho các vi điều khiển như STM32. Việc chia luồng giúp chia chương trình thành các nhiệm vụ độc lập. Mỗi nhiệm vụ xử lý một chức năng cụ thể, giúp mã nguồn dễ viết, kiểm tra và bảo trì. Ngoài ra việc sử dụng chia luồng và xử lý trên RTOS giúp tránh xung đột tài nguyên, hệ điều hành sử dụng các cơ chế như mutex, semaphore hoặc queue để quản lý truy cập đồng thời đến tài nguyên.

Khi viết 1 chương trình sử dụng chia luồng chạy trên RTOS, sơ đồ giải thuật của chương trình chúng ta thường trông như thế này:

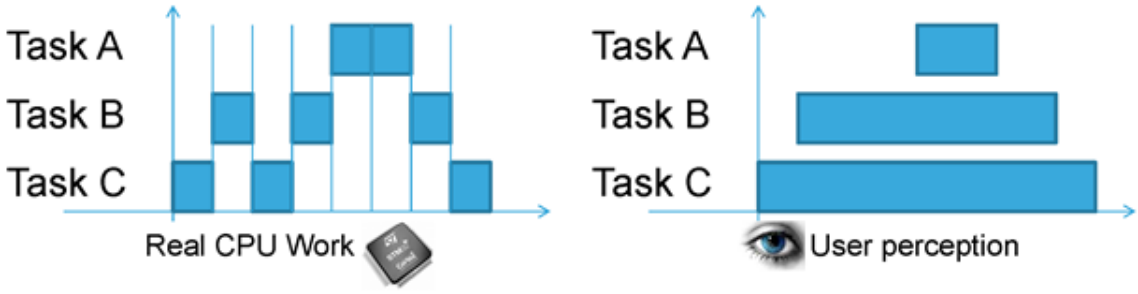


Hình 5.7. Sơ đồ khi sử dụng chia luồng trong RTOS.

Nhưng thực tế mỗi task sẽ được lên lịch để có thể chạy ở những thời gian khác nhau. Kernel sẽ có nhiệm vụ quản lý nhiều task cùng chạy 1 lúc, mỗi task thường chạy mất vài ms. Tại lúc kết thúc task thường:

- Lưu trạng thái task.
- Thanh ghi CPU sẽ load trạng thái của task tiếp theo.
- Task tiếp theo cần khoảng vài ms để thực hiện.

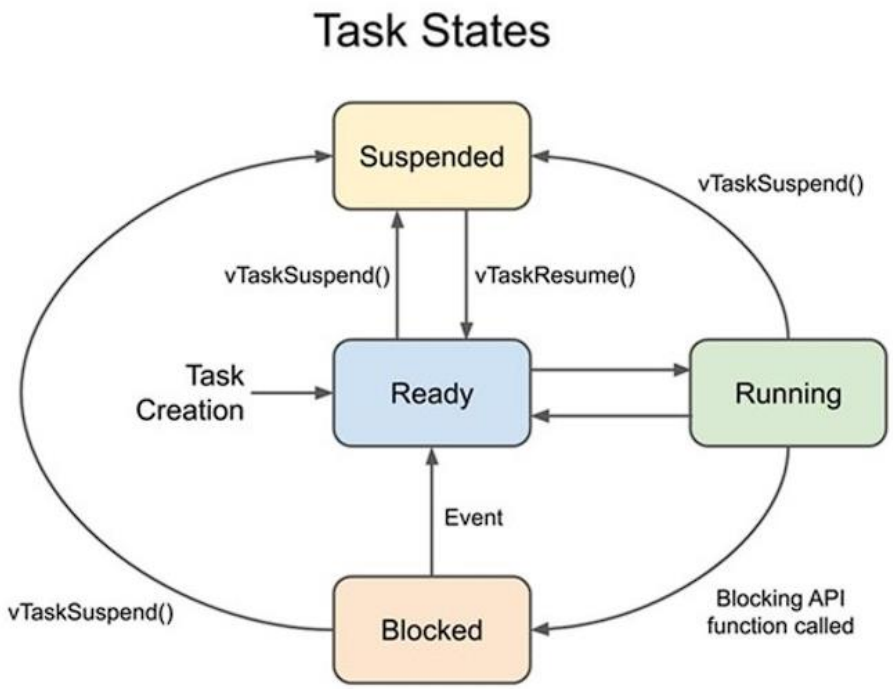
Vì CPU thực hiện tác vụ rất nhanh nên dưới góc nhìn người dùng thì hầu như các task là được thực hiện 1 cách liên tục.



Hình 5.8. Lên lịch chạy mỗi task.

Một task trong RTOS thường có những trạng thái như sau:

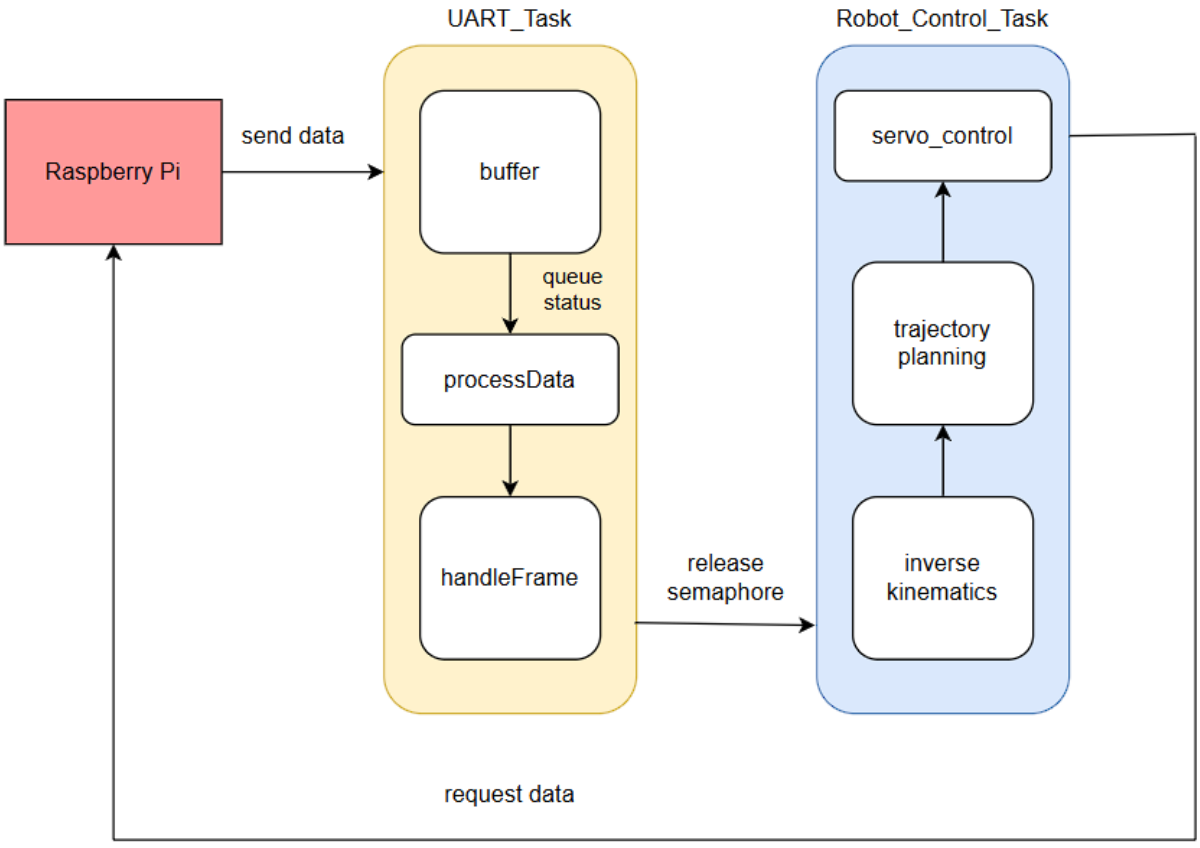
- Running: Đang thực thi.
- Ready: Sẵn sàng thực thi.
- Waiting: Chờ sự kiện.
- Inactive: Không được kích hoạt.



Hình 5.9. Trạng thái Task.

Đồ án này được chia làm 2 tasks, gồm: UART_Task và Robot_Control_Task.

- UART_Task có nhiệm vụ giao tiếp với máy tính nhúng, kiểm soát quá trình truyền nhận và xử lý dữ liệu.
- Robot_Control_Task có nhiệm vụ xử lý các bài toán liên quan đến robot và kiểm soát việc xuất xung điều khiển động cơ theo bộ PID.



Hình 5.10. Các luồng chạy của chương trình.

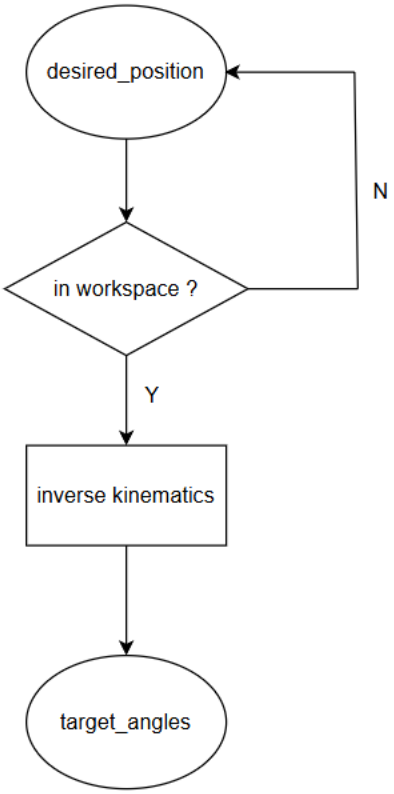
Semaphore là một công cụ được sử dụng trong khoa học máy tính để giúp quản lý cách các quy trình (hoặc chương trình) khác nhau chia sẻ tài nguyên, như bộ nhớ hoặc dữ liệu, mà không gây ra xung đột. Semaphore là một loại dữ liệu đồng bộ hóa đặc biệt chỉ có thể được sử dụng thông qua các nguyên hàm đồng bộ hóa cụ thể. Semaphore được sử dụng để triển khai các phần quan trọng, là các vùng mã phải được thực thi bởi chỉ một quy trình tại một thời điểm. Bằng cách sử dụng semaphore, các quy trình có thể phối hợp truy cập vào các tài nguyên được chia sẻ, như bộ nhớ được chia sẻ hoặc các thiết bị I/O.

Quá trình sử dụng Semaphores cung cấp hai hoạt động: wait (P) và signal (V). Hoạt động wait làm giảm giá trị của semaphore, và hoạt động signal làm tăng giá trị của semaphore. Khi giá trị của semaphore bằng không, bất kỳ tiến trình nào thực hiện hoạt động wait sẽ bị chặn cho đến khi một tiến trình khác thực hiện hoạt động signal.

Vì điều trên khi thực hiện xong UART_Task, cần giải phóng Semaphore để đến lượt Robot_Control_Task được thực hiện. Tuy nhiên đối với UART_Task, tín hiệu bắt đầu để thực hiện sẽ là kiểm tra hàng chờ Queue được xử lý từ buffer UART.

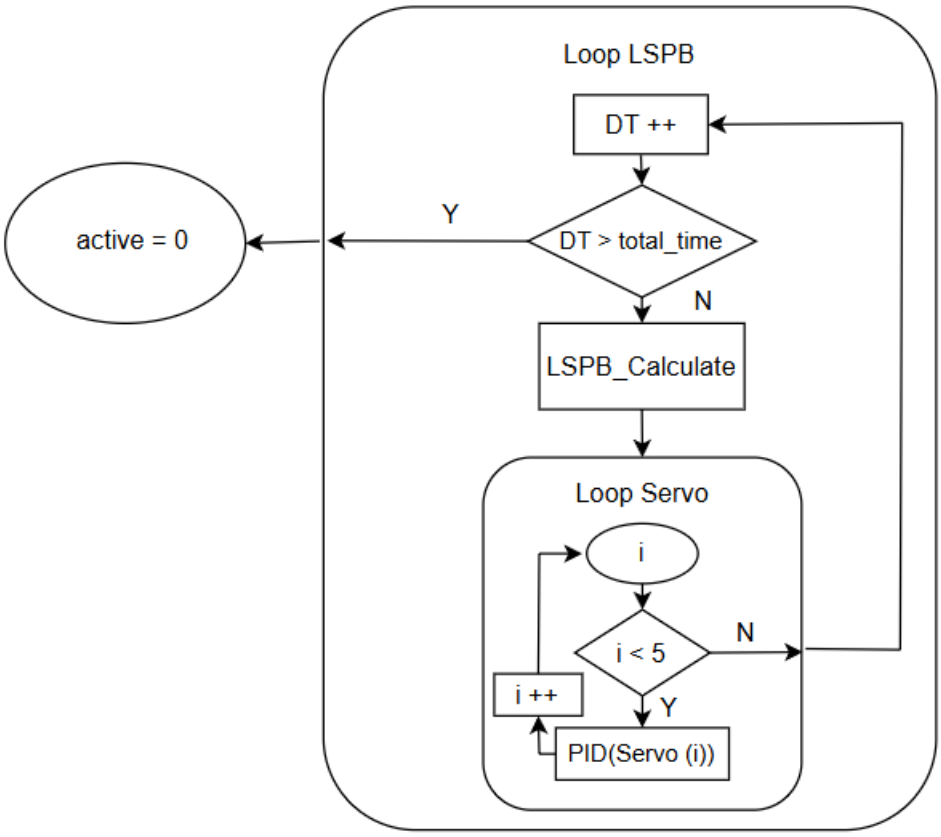
5.5. Thuật toán điều khiển

Các bài toán động học của cánh tay robot sẽ dựa trên tọa độ (x, y, z) để trả về giá trị góc θ của các góc khớp tương ứng. Tuy nhiên vì không phải giá trị (x, y, z) nào cũng có thể giải được ra các phương trình có nghiệm θ hoặc nếu có nghiệm thì các khớp sẽ di chuyển không đúng gây va chạm hư hỏng hệ thống. Vì vậy, trước khi giải các bài toán động học ta cần xác định các ngưỡng (x, y, z) để khoanh vùng trong thực tế để robot chạy đúng vùng hoạt động của nó.



Hình 5.11. Sơ đồ giải thuật tính toán góc quay mong muốn.

Sau khi có được góc mong muốn, hệ sẽ chuyển sang hoạch định quỹ đạo và điều khiển theo quỹ đạo đã chọn. Mỗi lần lặp của LSPB, hệ sẽ điều khiển cả 5 động cơ lần lượt theo vòng lặp nhỏ hơn. Mỗi động cơ sẽ được điều khiển bằng bộ PID đến góc tại mỗi thời điểm tương ứng.



Hình 5.12. Sơ đồ giải thuật hoạch định quỹ đạo.

Vòng lặp sẽ được thực hiện trong `total_time` đã được thiết lập trước đó, tại mỗi thời điểm `DT`, hàm `LSPB_Calculate` sẽ trả về một dãy các góc cần đạt tới. PID sẽ có nhiệm vụ điều khiển từng động cơ thông qua PCA 9685 đến được vị trí đó. Sau khi vòng lặp LSPB kết thúc, biến trạng thái đang hoạt động `active` sẽ được set về 0, báo hiệu sẵn sàng cho tọa độ tiếp theo.

CHƯƠNG 6. KẾT LUẬN