ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN

ĐỒ ÁN TỐT NGHIỆP
CAPSTONE PROJECT

NGÀNH: KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA

ĐỀ TÀI:

NGHIÊN CỨU ĐIỀU KHIỂN CÁNH TAY ROBOT TỰ ĐỘNG RÓT BIA SỬ DỤNG AI

Người hướng dẫn: **TS. TRẦN THỊ MINH DUNG**
Sinh viên thực hiện:
1. PHAN PHƯỚC BẮC – MSSV: 105200322 – LỚP: 20TDH2
2. VÕ BÁ PHÚC – MSSV: 105200339 – LỚP: 20TDH2

Đà Nẵng, 6/2025

TÓM TẮT

Tên đề tài: Nghiên cứu điều khiển cánh tay robot tự động rót bia sử dụng AI

Sinh viên thực hiện: Võ Bá Phúc Số thẻ SV:105200339

Phan Phước Bắc Số thẻ SV: 105200322

Lớp: 20TDH2

Trong đề tài này, hệ thống cánh tay robot tự động rót bia được nghiên cứu và phát triển nhằm phục vụ các dây chuyền tự động hóa trong lĩnh vực dịch vụ. Hệ thống ứng dụng các thuật toán trí tuệ nhân tạo (AI) kết hợp với xử lý ảnh nhằm nhận diện chính xác vị trí ly bia và xác định lượng bia đã rót.

Hình ảnh khu vực làm việc được thu nhận thông qua hệ thống camera, sau đó mô hình AI thực hiện nhận diện miệng ly và thân ly trong ảnh. Thông qua các thuật toán xử lý ảnh và phân tích màu sắc trong không gian HSV, hệ thống xác định chính xác vị trí ly và tính toán lượng bia hiện có trong ly.

Dữ liệu xử lý được truyền về bộ điều khiển trung tâm, thực hiện tính toán động học và điều khiển các khớp của cánh tay robot di chuyển chính xác đến vị trí rót bia. Trong quá trình vận hành, hệ thống liên tục giám sát mực bia và điều chỉnh tốc độ rót phù hợp nhằm đảm bảo chất lượng sản phẩm và tránh hiện tượng tràn bia.

Việc kết hợp giữa trí tuệ nhân tạo, thị giác máy tính và công nghệ điều khiển chuyển động chính xác giúp hệ thống vận hành ổn định, đạt độ chính xác cao và đáp ứng tốt yêu cầu tự động hóa trong thực tế.

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

TT	Họ tên sinh viên	Số thẻ SV	Lớp	Ngành
1	Võ Bá Phúc	105200339	20TDH2	Kỹ thuật Điều khiển và Tự động hóa
2	Phan Phước Bắc	105200322	20TDH2	Kỹ thuật Điều khiển và Tự động hóa

1. Tên đề tài đồ án:

Nghiên cứu điều khiển cánh tay robot tự động rót bia sử dụng AI

2. *Đề tài thuộc diện:* ☐*Có ký kết thỏa thuận sở hữu trí tuệ đối với kết quả thực hiện*

3. Các số liệu và dữ liệu ban đầu:

……..……..……...

...……

…..……..……..……

4. Nội dung các phần thuyết minh và tính toán:

a. Phần chung:

TT	Họ tên sinh viên	Nội dung
1	Phan Phước Bắc Võ Bá Phúc	- Tổng quan về đề tài - Nghiên cứu đưa ra giải pháp - Thi công mô hình - Lập trình điều khiển Robot

b. Phần riêng:

TT	Họ tên sinh viên	Nội dung
1	Phan Phước Bắc	- Mô hình hóa cánh tay Robot - Thiết kế bộ điều khiển, xây dựng mô hình mô phỏng MATLAB, SolidWorks - Thiết kế cánh tay Robot - Viết báo cáo chương 3, 4
2	Võ Bá Phúc	- Thiết kế sơ đồ công nghệ - Tính chọn thiết bị - Huấn luyện mô hình AI nhận diện ly, lập trình đọc và chuyển đổi tọa độ ly, xử lý ảnh đọc % bia trong ly - Viết báo cáo chương 1, 2, 5

5. Các bản vẽ, đồ thị (ghi rõ các loại và kích thước bản vẽ):

a. Phần chung:

TT	Họ tên sinh viên	Nội dung
1	Phan Phước Bắc Võ Bá Phúc	

b. Phần riêng:

TT	Họ tên sinh viên	Nội dung
1	Phan Phước Bắc	Bản vẽ chi tiết các khâu của cánh tay Robot
2	Võ Bá Phúc	Sơ đồ công nghệ

6. Họ tên ngườihướng dẫn:	*Phần/ Nội dung:*
TS. Trần Thị Minh Dung	

7. Ngày giao nhiệm vụ đồ án: 6/3/2024

8. Ngày hoàn thành đồ án: 26/6/2024

Đà Nẵng, ngày tháng 6 năm 2025

Trưởng Bộ môn Tự động hóa **Người hướng dẫn**

TS. Giáp Quang Huy TS. Trần Thị Minh Dung

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

PHIẾU KIỂM SOÁT TIẾN ĐỘ LÀM ĐỒ ÁN TỐT NGHIỆP

(Phiếu dành cho người hướng dẫn/sinh viên)

Họ tên sinh viên: Phan Phước Bắc Số thẻ SV:105200322

Võ Bá Phúc Số thẻ SV : 105200339

Tên đề tài ĐATN:

Nghiên cứu điều khiển cánh tay robot tự động rót bia sử dụng AI

Họ tên người HD: TS. Trần Thị Minh Dung Đơn vị: Trường Đại học Bách Khoa - ĐHĐN

Tuần	Ngày	Khối lượng		GVHD ký tên
		đã thực hiện (%)	tiếp tục thực hiện (%)	
1	3/3/2025	Nhận đề tài Tìm hiểu sơ lược về hệ thống	Xác định giải pháp và hướng tiếp cận	
2	10/3/2025	Tìm hiểu tổng quan về robot Các yêu cầu cho điều khiển robot	Đọc, tham khảo tài liệu trên internet	
3	17/3/2025	Ứng dụng của robot Đưa ra đề tài	Viết báo cáo chương 1	
4	24/3/2025	Duyệt lần 1: Đánh giá khối lượng hoàn thành _____ % : Được tiếp tục làm ĐATN ϒ Không tiếp tục thực hiện ĐATN ϒ		
5	3/4/2025	Hoàn thành báo cáo chương 1	Liệt kê những khó khăn trong đề tài	
6	10/4/2025	Đưa ra những giải pháp để giải quyết khó khăn trong đồ án	Tìm hiểu sơ đồ công nghệ, quy trình công nghệ	
7	17/4/2025	Thiết kế sơ đồ công nghệ, sơ đồ kết nối	Tính chọn thiết bị Viết báo cáo chương 2	
8	24/4/2025	Duyệt lần 2: Đánh giá khối lượng hoàn thành _____ % : Được tiếp tục làm ĐATN ϒ Không tiếp tục thực hiện ĐATN ϒ		
9	3/5/2025	Hoàn thành báo cáo chương 2 Tính toán động học thuận	Nghiên cứu động học nghịch Nghiên cứu động lực học	
10	10/5/2025	Tính toán động học nghịch Tính toán động lực học	Thiết kế bộ điều khiển Nghiên cứu mô hình AI	
11	17/5/2025	Thiết kế mô hình AI Triển khai hệ thống AI	Viết báo cáo chương 3	
12	24/5/2025	Duyệt lần 3: Đánh giá khối lượng hoàn thành _____ % : Được tiếp tục làm ĐATN ϒ Không tiếp tục thực hiện ĐATN ϒ		

13	1/6/2025	Hoàn thành báo cáo chương 3 Tiến hành mô phỏng trên Matlab	Tiến hành mô phỏng trên simcape	
14	8/6/2025	Đưa ra kết quả và phân tích so sánh	Tiến hành thi công mô hình	
15	15/6/2025	So sánh với kết quả thực nghiệm	Hoàn thiện nội dung báo cáo	

LỜI NÓI ĐẦU VÀ CẢM ƠN

Trong những năm gần đây, cùng với sự phát triển mạnh mẽ của công nghệ trí tuệ nhân tạo (AI) và tự động hóa, các hệ thống robot ngày càng được ứng dụng rộng rãi trong đời sống và sản xuất. Xuất phát từ thực tiễn đó, đề tài "Nghiên cứu điều khiển cánh tay robot tự động rót bia sử dụng AI" được thực hiện nhằm ứng dụng các thành tựu công nghệ hiện đại, góp phần nâng cao hiệu quả và mức độ tự động hóa trong lĩnh vực phục vụ.

Trong quá trình thực hiện, nhóm đã nghiên cứu và vận dụng các kỹ thuật xử lý ảnh, nhận dạng đối tượng bằng mô hình học sâu, kết hợp với hệ thống điều khiển chuyển động chính xác nhằm xây dựng nên một hệ thống cánh tay robot có khả năng nhận diện vị trí ly, giám sát mực bia và điều khiển thao tác rót bia tự động một cách ổn định và hiệu quả.

Để hoàn thành đề tài này, nhóm xin bày tỏ lòng biết ơn sâu sắc đến quý thầy cô trong Khoa Điện, đặc biệt là Tiến sĩ Trần Thị Minh Dung, người đã tận tình hướng dẫn, hỗ trợ và tạo điều kiện thuận lợi cho nhóm trong suốt quá trình nghiên cứu và thực hiện. Nhóm cũng xin gửi lời cảm ơn chân thành đến gia đình và bạn bè đã luôn đồng hành, động viên trong suốt thời gian học tập và nghiên cứu.

Mặc dù đã nỗ lực hoàn thành, song do giới hạn về thời gian và kinh nghiệm, đề tài chắc chắn vẫn còn những thiếu sót. Nhóm rất mong nhận được những ý kiến đóng góp quý báu từ quý thầy cô và các bạn để tiếp tục hoàn thiện hơn trong tương lai.

Nhóm chúng em xin chân thành cảm ơn!

LỜI CAM ĐOAN LIÊM CHÍNH HỌC THUẬT

Chúng em xin cam đoan đồ án tốt nghiệp với đề tài *"Nghiên cứu điều khiển cánh tay robot tự động rót bia sử dụng AI"* là kết quả của quá trình học tập, nghiên cứu và làm việc nghiêm túc của chính nhóm chúng em. Trong quá trình thực hiện, nhóm đã tuân thủ đầy đủ các quy định về liêm chính học thuật, không sao chép nội dung từ các công trình nghiên cứu khác mà không trích dẫn hoặc được sự cho phép.

Mọi số liệu, hình ảnh, kết quả nghiên cứu, tính toán trong đồ án đều được thu thập, xử lý và trình bày trung thực, đúng với thực tế nghiên cứu. Nếu có bất kỳ vi phạm nào liên quan đến vấn đề liêm chính học thuật, nhóm chúng em hoàn toàn chịu trách nhiệm trước Hội đồng và các quy định của nhà trường.

Sinh viên thực hiện

MỤC LỤC

DANH SÁCH CÁC BẢNG

DANH SÁCH HÌNH ẢNH

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

KÝ HIỆU:

Ký hiệu	Giải thích
c	cos
s	sin
R_n^0	Ma trận xoay
P_n^0	Ma trận vị trí
T_n^0	Ma trận tổng
L_i	Chiều dài khớp thứ i
θ_i	Góc quay của khớp thứ i
p_x, p_y, p_z	Tọa độ cánh tay Robot
${}^i\omega_i$	Vận tốc góc
${}^iv_i$	Vận tốc dài
${}^{C_i}I_i$	Momen
m	Khối lượng
g	Gia tốc trọng trường
K	Động năng
U	Thế năng
u	Tín hiệu điều khiển
e	Sai số

CHỮ VIẾT TẮT:

Chữ viết tắt	Giải nghĩa
AI	Artificial Intelligence (Trí tuệ nhân tạo)
EMG	Electromyography (Điện cơ đồ)
IoT	Internet of Things (Mạng lưới vạn vật kết nối Internet)
IFR	International Federation of Robotics (Liên đoàn Robot Quốc tế)
F&B	Food and Beverage (Thực phẩm và Đồ uống)

DIY	Do It Yourself (Tự làm, tự thực hiện)
YOLO	You Only Look Once (Thuật toán phát hiện đối tượng thời gian thực)
UART	Universal Asynchronous Receiver Transmitter (Bộ thu phát không đồng bộ)
TensorRT	Tensor Runtime (Thư viện tăng tốc AI của NVIDIA)
ONNX	Open Neural Network Exchange (Định dạng trao đổi mô hình mạng nơ-ron mở)
PWM	Pulse Width Modulation (Điều chế độ rộng xung)
DH	Denavit-Hartenberg (Phương pháp tham số hóa động học robot)
PID	Proportional-Integral-Derivative (Điều khiển tỉ lệ – tích phân – vi phân)
HSV	Hue-Saturation-Value (Không gian màu sắc – độ bão hòa – giá trị sáng)
PUL	Pulse (Xung điều khiển step motor)
DIR	Direction (Hướng quay motor)
CNN	Convolutional Neural Network (Mạng nơ-ron tích chập)
SSD	Single Shot MultiBox Detector (Bộ phát hiện vật thể một lần duy nhất)
NMS	Non-Maximum Suppression (Thuật toán triệt tiêu cực đại)
HTĐ	Hệ Tọa Độ
COCO	Common Objects in Context (Tập dữ liệu đối tượng trong bối cảnh)
IoU	Intersection over Union (Đo độ chồng lấp của các bounding box)
mAP	mean Average Precision (Trung bình độ chính xác trung bình)
DOF	Degree of Freedom (Bậc tự do)
PD	Proportional-Derivative (Điều khiển tỉ lệ – vi phân)

MỞ ĐẦU

Trong những năm gần đây, cùng với sự phát triển mạnh mẽ của khoa học kỹ thuật, việc ứng dụng robot công nghiệp vào sản xuất và dịch vụ ngày càng phổ biến. Đặc biệt, các hệ thống robot kết hợp xử lý hình ảnh và điều khiển chính xác đã mang lại nhiều giải pháp hiệu quả trong tự động hóa các quy trình phức tạp.

Mục đích thực hiện đề tài

Nghiên cứu, thiết kế và xây dựng hệ thống điều khiển cánh tay robot thực hiện thao tác rót bia tự động dựa trên nhận diện hình ảnh, góp phần nâng cao khả năng ứng dụng thực tế của robot trong lĩnh vực phục vụ và sản xuất.

Mục tiêu đề tài:

+ Thiết kế, lắp ráp hệ thống robot.
+ Xây dựng hệ thống nhận diện vị trí ly và lượng bia trong ly bằng xử lý ảnh và AI.
+ Xây dựng thuật toán điều khiển robot rót bia chính xác, đảm bảo an toàn và ổn định.

Phạm vi và đối tượng nghiên cứu:

+ Đối tượng nghiên cứu: hệ thống robot 4 bậc tự do kết hợp với camera và xử lý hình ảnh.
+ Phạm vi: nhận diện ly, xác định vị trí ly trong vùng làm việc và điều khiển robot rót bia.

Phương pháp nghiên cứu:

+ Nghiên cứu lý thuyết về robot công nghiệp, xử lý ảnh và thị giác máy tính.
+ Ứng dụng OpenCV và AI (YOLOv8) trong xử lý ảnh.
+ Mô phỏng và điều khiển thực tế hệ thống robot.
+ Đánh giá qua các thí nghiệm thực tế.

Cấu trúc đồ án tốt nghiệp:

Ngoài phần mở đầu, kết luận và tài liệu tham khảo, đồ án gồm:

+ Chương 1: Tổng quan đề tài
+ Chương 2: Nghiên cứu và thiết kế cánh tay robot rót bia
+ Chương 3: Cơ sở lý thuyết tính toán robot rót bia
+ Chương 4: Mô phỏng hệ thống robot rót bia
+ Chương 5: Thiết kế và thi công mô hình

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

1.1. Tổng quan về cánh tay robot

1.1.1. Giới thiệu

Cánh tay robot là một trong những cấu trúc phổ biến và quan trọng nhất trong lĩnh vực robot học và tự động hóa. Thiết bị này mô phỏng chuyển động và chức năng của cánh tay người, bao gồm các khớp nối và bộ điều khiển, nhằm thực hiện các tác vụ cơ học chính xác, linh hoạt và có thể lặp lại. Trong bối cảnh cuộc cách mạng công nghiệp 4.0 và hướng tới 5.0, cánh tay robot đóng vai trò then chốt trong việc tối ưu hóa sản xuất, hỗ trợ y tế, phục vụ con người và nhiều lĩnh vực khác.

Cánh tay robot được ứng dụng rộng rãi trong nhiều lĩnh vực

- ✓ Trong công nghiệp: robot thực hiện các công việc lặp đi lặp lại, nguy hiểm hoặc yêu cầu độ chính xác cao như hàn, lắp ráp, đóng gói, kiểm tra sản phẩm.
- ✓ Trong y tế: robot phẫu thuật (như da Vinci), hỗ trợ phục hồi chức năng, hỗ trợ chăm sóc người bệnh.
- ✓ Trong dịch vụ: robot pha chế, phục vụ tại nhà hàng, khách sạn, chăm sóc người cao tuổi.
- ✓ Trong nghiên cứu và giáo dục: mô phỏng, thí nghiệm, đào tạo kỹ sư và nhà nghiên cứu.

Mỗi lĩnh vực sử dụng yêu cầu các đặc điểm kỹ thuật khác nhau: độ chính xác, tốc độ, lực tác động, khả năng phản hồi, và mức độ an toàn khi làm việc với con người.

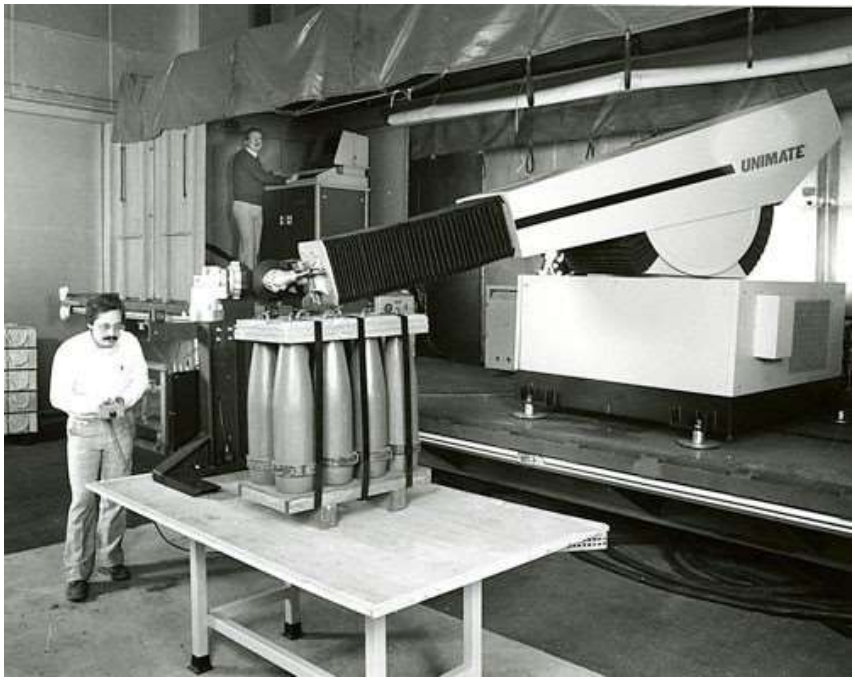
Hình 1.1: Ứng dụng của cánh tay Robot trong công nghiệp

1.1.2 Lịch sử phát triển

a) Giai đoạn đầu phát triển (1950–1970)

Cánh tay robot hiện đại bắt đầu hình thành vào giữa thế kỷ 20. Năm 1954, kỹ sư người Mỹ George Devol đã phát minh ra một thiết bị điều khiển từ xa có thể tái lập chuyển động, được gọi là Unimate – robot công nghiệp đầu tiên trên thế giới.

Unimate được đưa vào sử dụng tại nhà máy sản xuất ô tô General Motors vào năm 1961, thực hiện công việc hàn trên dây chuyền sản xuất. Đây là bước ngoặt lịch sử đánh dấu sự ra đời của robot công nghiệp hiện đại.



Hình 1.2: Robot Unimate, robot công nghiệp đầu tiên trong nhà xưởng

b) Giai đoạn phát triển mạnh (1970–1990)

Trong giai đoạn này, các hãng công nghệ lớn trên thế giới như FANUC (Nhật Bản), KUKA (Đức), ABB (Thụy Sĩ) bắt đầu phát triển hàng loạt cánh tay robot phục vụ cho các nhà máy. Sự ra đời của vi điều khiển và lập trình số giúp robot trở nên thông minh và chính xác hơn.

Cánh tay robot được ứng dụng để thực hiện các thao tác như hàn, lắp ráp, phun sơn, đóng gói… góp phần làm thay đổi toàn diện phương thức sản xuất truyền thống.

Hình 1.3: Robot KUKA trong ngành công nghiệp oto

c) Giai đoạn hiện đại (1990–nay)

Từ những năm 1990 đến nay, công nghệ robot phát triển vượt bậc. Các cánh tay robot ngày nay không chỉ nhanh và chính xác mà còn được tích hợp AI, thị giác máy, cảm biến lực, thậm chí có khả năng học tập và tự điều chỉnh hành vi.

Ngoài robot công nghiệp, một thế hệ mới gọi là robot cộng tác (collaborative robots – cobots) được phát triển. Những robot này có thể làm việc cùng con người mà không cần hàng rào bảo vệ, giúp tăng tính linh hoạt trong sản xuất.

Hình 1.4: Một Robot tích hợp thị giác máy tính

1.2. Điều khiển phối hợp

Trong các hệ thống tự động hiện đại, đặc biệt là robot và các hệ thống đa tác tử (multi-agent systems), khái niệm điều khiển phối hợp ngày càng trở nên quan trọng. Điều khiển phối hợp cho phép nhiều đối tượng cùng thực hiện một nhiệm vụ một cách đồng bộ, tối ưu hóa hiệu suất, nâng cao độ chính xác và tính linh hoạt của hệ thống.

Điều khiển phối hợp (Coordinated Control) là một phương pháp điều khiển trong đó nhiều đối tượng (robot, thiết bị chấp hành, hay các hệ thống con) tương tác với nhau nhằm thực hiện một mục tiêu chung.

Điều khiển phối hợp yêu cầu

– Trao đổi thông tin giữa các đối tượng.
– Thiết kế luật điều khiển phù hợp để đạt mục tiêu chung.
– Cân bằng giữa tính độc lập và tính đồng bộ của các phần tử.

1.2.2. Các phương pháp điều khiển phối hợp hiện nay

a) Điều khiển phân tán (Distributed Control)

Mỗi tác nhân có bộ điều khiển riêng, sử dụng giao thức đồng thuận (consensus protocols) để trao đổi thông tin với các tác nhân lân cận, hướng tới mục tiêu chung. Các thuật toán như giao thức đồng thuận trung bình (average consensus) hoặc tối đa/hối thiểu (max/min consensus) thường được áp dụng.

- **Ứng dụng**

– Dây chuyền sản xuất công nghiệp, ví dụ: cánh tay robot lắp ráp ô tô.
– Hệ thống robot y tế trong phẫu thuật chính xác.

Hình 1.5: Cánh tay robot lắp ráp ô tô

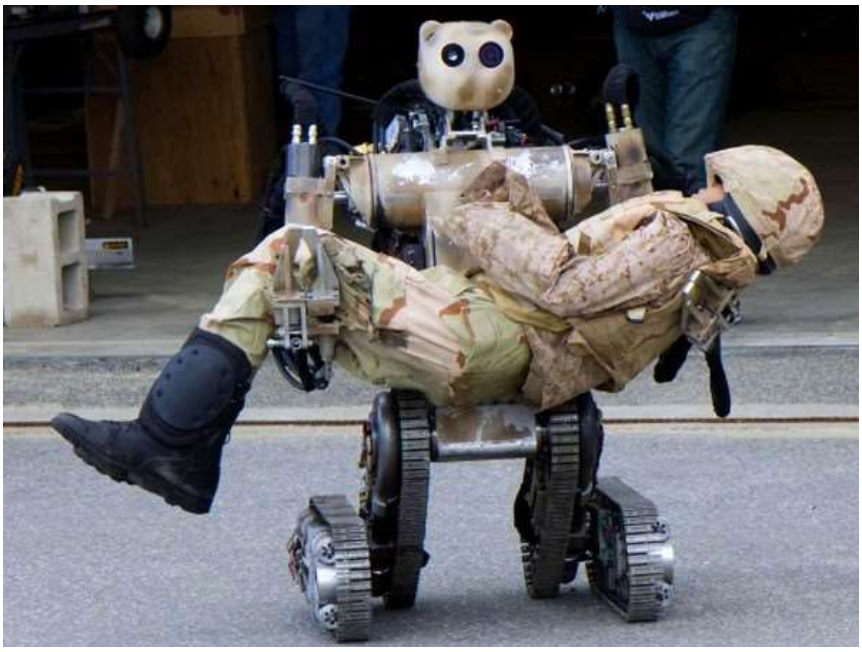
b) Điều khiển phi tập trung (Decentralized Control)

Trong phương pháp này, mỗi robot có bộ điều khiển riêng, tự xử lý thông tin cục bộ và phối hợp với các robot khác thông qua giao tiếp. Các robot ra quyết định dựa trên thông tin từ cảm biến của chính mình và dữ liệu nhận được từ các đơn vị khác.

- **Ứng dụng**

– Đàn robot (swarm robotics) trong nhiệm vụ tìm kiếm và cứu hộ.

– Hệ thống robot di động tự hành trong kho bãi hoặc logistics.



Hình 1.6: Robot cứu hộ

c) Điều khiển phân cấp (Hierarchical Control)

Phương pháp này kết hợp ưu điểm của điều khiển tập trung và phân tán, chia hệ thống thành các cấp bậc. Cấp cao chịu trách nhiệm lập kế hoạch chiến lược và ra quyết định tổng thể, trong khi cấp thấp thực hiện các nhiệm vụ cụ thể dựa trên lệnh từ cấp cao.

- **Ứng dụng**

– Nhà máy thông minh (smart factory) với nhiều robot phối hợp.

– Hệ thống giao thông thông minh với các phương tiện tự hành.

d) Điều khiển dựa trên hành vi (Behavior-based Control)

Phương pháp này dựa trên việc thiết kế các hành vi đơn giản (như tránh chướng ngại vật, di chuyển đến mục tiêu, hoặc hợp tác với robot khác) và kết hợp chúng để đạt được mục tiêu chung. Mỗi robot hoạt động dựa trên tập hợp các hành vi được lập trình sẵn.

- **Ứng dụng**

– Robot tìm kiếm và cứu hộ trong môi trường nguy hiểm.

– Đàn robot tự tổ chức trong nông nghiệp hoặc giám sát môi trường.

e) Điều khiển dựa trên học máy và trí tuệ nhân tạo

Sử dụng các thuật toán học máy, chẳng hạn như học sâu hoặc học tăng cường để robot tự học cách phối hợp dựa trên dữ liệu hoặc kinh nghiệm thực tế. Các robot có thể thích nghi với môi trường và tối ưu hóa hành vi theo thời gian.

- **Ứng dụng**

– Robot tự hành trong giao thông đô thị.

– Hệ thống robot hợp tác thông minh trong sản xuất hoặc khám phá không gian.

1.2.3. Các yêu cầu kỹ thuật cho điều khiển phối hợp

- **Độ chính xác**

+ Robot phải thực hiện nhiệm vụ với sai số tối thiểu (ví dụ: sai số vị trí dưới 1 mm trong sản xuất công nghiệp).

+ Đảm bảo các robot phối hợp đúng với mục tiêu chung, đặc biệt trong các ứng dụng yêu cầu độ chính xác cao như phẫu thuật hoặc lắp ráp.

- **Tốc độ**

+ Thời gian phản hồi nhanh để đảm bảo phối hợp thời gian thực.

+ Tốc độ xử lý và truyền dữ liệu giữa các robot hoặc từ bộ điều khiển trung tâm phải đủ nhanh để đáp ứng yêu cầu nhiệm vụ.

- **Độ ổn định**

+ Hệ thống phải duy trì hoạt động ổn định, không bị gián đoạn hoặc lỗi khi có nhiễu, thay đổi môi trường hoặc mất kết nối tạm thời.

+ Đảm bảo tính liên tục trong phối hợp, tránh xung đột giữa các robot.

- **Khả năng mở rộng**

+ Hệ thống phải hỗ trợ tăng số lượng robot mà không làm giảm hiệu suất.

+ Phù hợp với các ứng dụng từ vài robot đến hàng trăm robot (như đàn robot).

- **Giao tiếp**

+ Băng thông giao tiếp cao, độ trễ.

+ Giao thức giao tiếp đáng tin cậy để chia sẻ dữ liệu như vị trí, trạng thái.

- **Độ linh hoạt**

+ Khả năng thích nghi với các môi trường động hoặc không xác định (ví dụ: robot cứu hộ trong môi trường thay đổi).

+ Hỗ trợ tái cấu hình nhiệm vụ mà không cần lập trình lại toàn bộ hệ thống.

- **Tiết kiệm năng lượng**

+ Tối ưu hóa tiêu thụ năng lượng trong tính toán, giao tiếp và chuyển động để kéo dài thời gian hoạt động.

+ Đặc biệt quan trọng với robot di động hoặc đàn robot.

- **An toàn và bảo mật**

+ Đảm bảo robot phối hợp không gây nguy hiểm cho con người hoặc môi trường.

+ Bảo vệ hệ thống khỏi các cuộc tấn công mạng, đặc biệt trong các hệ thống sử dụng AI hoặc giao tiếp không dây.

- **Khả năng đồng bộ**

+ Các robot phải hoạt động đồng bộ về thời gian và không gian (ví dụ: sai số đồng bộ dưới 1 ms trong sản xuất).

+ Đảm bảo các hành động phối hợp (như di chuyển hoặc thao tác) diễn ra đúng trình tự.

1.3. Ứng dụng và đặc thù kỹ thuật theo từng lĩnh vực

Cánh tay robot là một trong những cấu trúc phổ biến nhất trong lĩnh vực robot học, với khả năng mô phỏng chuyển động của cánh tay con người để thực hiện các nhiệm vụ tự động hóa. Trong nhiều thập kỷ qua, cánh tay robot đã được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau, mỗi lĩnh vực lại đặt ra những yêu cầu kỹ thuật riêng biệt nhằm đáp ứng đặc thù về môi trường hoạt động, mức độ chính xác, tính linh hoạt và độ an toàn. Trong phần này, ta sẽ phân tích bốn nhóm lĩnh vực ứng dụng chính: công nghiệp, y tế, dịch vụ và nghiên cứu.

1.3.1. Ứng dụng trong công nghiệp

Trong ngành công nghiệp, đặc biệt là sản xuất và lắp ráp, cánh tay robot được ứng dụng phổ biến trong các công đoạn như: gắp – đặt (pick and place), lắp ráp linh kiện, hàn tự động, phun sơn, kiểm tra sản phẩm, đóng gói và vận chuyển hàng hóa. Những hệ thống robot công nghiệp này thường làm việc theo chu kỳ liên tục, yêu cầu tốc độ và độ chính xác cao để đảm bảo chất lượng và năng suất.

- **Các yêu cầu kỹ thuật đặc trưng:**

+ Độ chính xác cao: Trong các dây chuyền lắp ráp vi mô như linh kiện điện tử, sai số cho phép chỉ từ 0.01 – 0.05 mm.

+ Tốc độ và tính lặp lại: Robot phải thực hiện các tác vụ lặp lại với thời gian ngắn và độ ổn định cao.

Hình 1.7: Cánh tay robot bóc xếp hàng hóa

1.3.2. Ứng dụng trong y tế

Trong lĩnh vực y tế, cánh tay robot đóng vai trò hỗ trợ hoặc thay thế bác sĩ trong các công việc yêu cầu độ chính xác và ổn định cao như phẫu thuật, chẩn đoán hình ảnh, phục hồi chức năng, phân phát thuốc hoặc hỗ trợ người khuyết tật. Một ví dụ nổi bật là hệ thống robot phẫu thuật Da Vinci, cho phép bác sĩ điều khiển từ xa các cánh tay robot để thực hiện thao tác mổ chính xác tới từng milimet.

- **Yêu cầu kỹ thuật đặc trưng:**

+ Độ chính xác và độ ổn định cực cao: Phẫu thuật yêu cầu độ sai lệch tối đa chỉ từ 0.1 mm, thậm chí nhỏ hơn.

+ Điều khiển lực tinh vi: Các động tác cần mượt mà, chính xác và có phản hồi lực để không gây tổn thương mô mềm.

Hình 1.8: Robot phẫu thuật Da Vinci

1.3.3. Ứng dụng trong dịch vụ

Robot dịch vụ đang ngày càng phát triển, đặc biệt trong các lĩnh vực như nhà hàng, khách sạn, chăm sóc người già, hỗ trợ gia đình, hoặc các quầy tiếp tân tự động. Các robot này thường có thiết kế thân thiện, dễ tương tác và có khả năng thực hiện các nhiệm vụ đơn giản như mang đồ, pha chế nước, nói chuyện, hoặc hỗ trợ người dùng thực hiện các thao tác cơ bản.

- **Yêu cầu kỹ thuật đặc trưng**

+ Giao tiếp với con người: Robot cần nhận diện giọng nói, gương mặt, cử chỉ, hoặc điều hướng trong không gian có người di chuyển.

+ An toàn và mềm dẻo: Do hoạt động gần con người, robot cần hạn chế lực va chạm và sử dụng vật liệu mềm, nhẹ.

Hình 1.9: Robot pha chế

1.3.4. Ứng dụng trong nghiên cứu và đào tạo

Robot dạng cánh tay được sử dụng phổ biến trong các phòng thí nghiệm, viện nghiên cứu và cơ sở đào tạo để nghiên cứu các thuật toán điều khiển, trí tuệ nhân tạo, mô phỏng chuyển động con người, hoặc tích hợp các công nghệ cảm biến mới. Nhiều trường đại học sử dụng robot này để giảng dạy môn học như điều khiển robot, xử lý ảnh, lập trình hệ thống nhúng, ROS...

- **Yêu cầu kỹ thuật đặc trưng:**

+ Tính linh hoạt và tùy biến: Cần hỗ trợ nhiều cấu hình khác nhau, dễ thay đổi mô hình cơ khí, phần mềm.

+ Tích hợp dễ dàng: Hỗ trợ giao tiếp với phần mềm mô phỏng như MATLAB, ROS, hoặc các ngôn ngữ điều khiển phổ biến (Python, C++).

1.4. Tình hình trong nước và ngoài nước

- **Tình hình quốc tế**

Ở Hoa Kỳ

+ Barrett Technology: Công ty này nổi tiếng với cánh tay robot WAM (Whole Arm Manipulation) 7 bậc tự do, được thiết kế để tương tác trực tiếp với con người và được ứng dụng trong các lĩnh vực như phẫu thuật và phục hồi chức năng.

Ở Nhật Bản

+ Kawasaki Robotics: Là một trong những công ty tiên phong trong lĩnh vực robot công nghiệp, Kawasaki đã phát triển robot công nghiệp đầu tiên của Nhật Bản vào năm 1969, mở đường cho sự phát triển mạnh mẽ của ngành công nghiệp robot tại quốc gia này.

+ Cyberdyne: Công ty này nổi tiếng với sản phẩm HAL (Hybrid Assistive Limb), một bộ khung xương trợ lực sử dụng trong y tế để hỗ trợ người khuyết tật vận động.

Ở Đức

+ KUKA: Là một trong những nhà sản xuất robot công nghiệp hàng đầu thế giới, KUKA cung cấp các giải pháp tự động hóa toàn diện cho nhiều ngành công nghiệp, bao gồm ô tô, điện tử và logistics.

+ Franka Emika: Công ty này phát triển robot cộng tác Franka Research 3, nổi bật với khả năng cảm nhận lực chính xác và dễ dàng tích hợp vào các ứng dụng nghiên cứu và giáo dục.

- **Tình hình trong nước**

+ Viện Hàn lâm Khoa học và Công nghệ Việt Nam (VAST): Đã phát triển cánh tay robot 6 bậc tự do SM6, ứng dụng trong dây chuyền sản xuất phân bón vi sinh.

+ Trường Đại học Công nghiệp Hà Nội: Nghiên cứu thiết kế cánh tay robot điều khiển bằng tín hiệu điện cơ (EMG) để hỗ trợ người khuyết tật, mở ra hướng đi mới trong ứng dụng robot y tế tại Việt Nam.

+ Công ty TNHH Robot Việt Nam (VNRobotics): Phát triển bộ kit robot giáo dục, được Sở Khoa học và Công nghệ TP.HCM chấp thuận, phục vụ cho việc học tập và nghiên cứu của học sinh, sinh viên.

Bảng 1.1: So sánh công nghệ trong và ngoài nước

Tiêu chí	Quốc tế	Việt Nam
Mức độ phát triển	Công nghệ tiên tiến, tích hợp AI, cảm biến lực, và khả năng học máy.	Đang trong giai đoạn nghiên cứu và phát triển, tập trung vào ứng dụng cụ thể.
Ứng dụng	Đa dạng: công nghiệp, y tế, dịch vụ, nghiên cứu.	Chủ yếu trong giáo dục, nghiên cứu và một số ứng dụng công nghiệp.
Khả năng sản xuất	Sản xuất hàng loạt với tiêu chuẩn quốc tế.	Sản xuất quy mô nhỏ, chủ yếu phục vụ nghiên cứu và đào tạo.
Chi phí	Cao, phù hợp với các doanh nghiệp lớn.	Thấp hơn, phù hợp với điều kiện kinh tế trong nước.
Đội ngũ nhân lực	Chuyên gia giàu kinh nghiệm, đội ngũ R&D mạnh.	Đang phát triển, cần đào tạo thêm về chuyên môn và kỹ năng.

1.5. Lựa chọn và xác định đề tài

Nhằm đáp ứng nhu cầu tự động hóa trong phục vụ đồ uống, nhóm tiến hành nghiên cứu hệ thống cánh tay robot có khả năng rót bia tự động bằng cách ứng dụng công nghệ xử lý ảnh và trí tuệ nhân tạo (AI).

Đề tài được lựa chọn:

"Nghiên cứu điều khiển cánh tay robot tự động rót bia sử dụng AI"

- **Cơ sở lựa chọn đề tài**

Trong xu thế phát triển của công nghệ robot dịch vụ, việc tự động hóa các tác vụ lặp đi lặp lại như rót bia, phục vụ đồ uống đang ngày càng được quan tâm tại các nhà hàng, quán bar và không gian dịch vụ thông minh. Các bài toán này không chỉ đòi hỏi robot thực hiện thao tác chính xác, mà còn cần khả năng nhận thức được môi trường – như vị trí ly và lượng bia đã có trong ly.

Từ thực tế đó, đề tài hướng đến việc xây dựng một hệ thống sử dụng cánh tay robot 4 bậc tự do, kết hợp với thị giác máy để nhận diện và xác định vị trí ly, đồng thời có thể điều chỉnh lượng bia rót dựa trên việc phân tích hình ảnh thực tế của ly.

- **Khoảng trống nghiên cứu**

+ Phần lớn các đề tài hiện nay dừng lại ở việc robot thực hiện thao tác rót theo thời gian cố định, chưa tích hợp khả năng phân tích mức nước trong ly để điều chỉnh lượng rót.

+ Trong nước, số lượng nghiên cứu tích hợp xử lý ảnh – điều khiển robot – giám sát phản hồi trong một hệ thống đồng bộ còn hạn chế, đặc biệt với cấu hình robot bậc tự do thấp hơn.

- **Tính khả thi**

+ Về phần cứng: Cánh tay robot 4 bậc tự do có thể được chế tạo từ vật liệu nhẹ, sử dụng servo kết hợp STM32 và Raspberry Pi để điều khiển – chi phí thấp và dễ chế tạo.

+ Về xử lý ảnh: Dùng OpenCV và mô hình YOLOv8 để nhận diện ly và xác định vị trí miệng ly để robot rót chính xác vào vị trí.

+ Về đo lượng bia: Sử dụng xử lý ảnh (tương phản giữa phần có bia và phần rỗng) và mô hình AI đơn giản để xác định phần trăm mức đầy của ly.

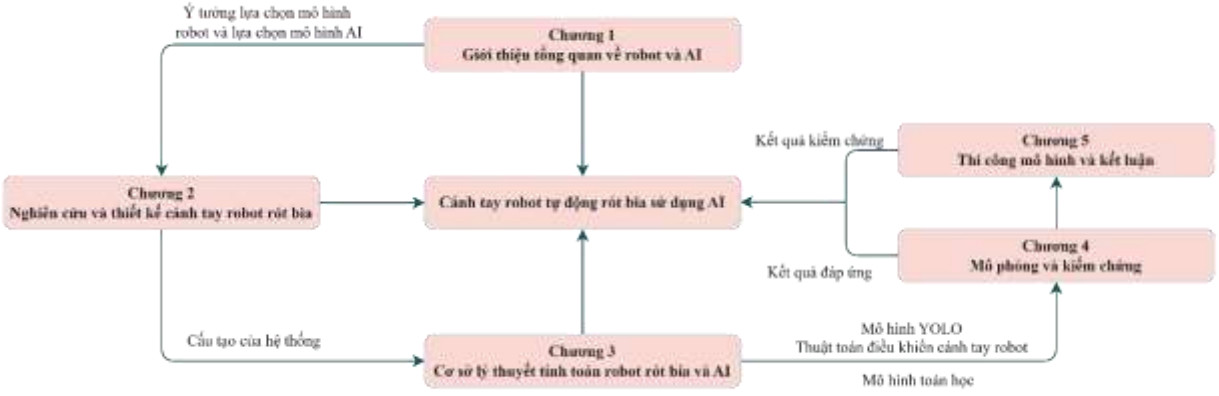
- **Giá trị thực tiễn**

+ Ứng dụng thực tế: Có thể ứng dụng trong không gian quán bar, nhà hàng tự phục vụ, tiết kiệm nhân lực, tăng trải nghiệm khách hàng.

+ Tính giáo dục và học thuật: Là một bài toán tổng hợp các kỹ năng về cơ điện tử, tự động hóa, thị giác máy và điều khiển robot – phù hợp với sinh viên kỹ thuật.

+ Tính mở rộng: Có thể nâng cấp cho các ứng dụng khác như rót nước, pha chế tự động, phục vụ trong lĩnh vực dịch vụ thông minh.

Sơ đồ cấu trúc của ĐATN:



Hình 1.10: Lưu đồ kết nối công việc

CHƯƠNG 2: NGHIÊN CỨU VÀ THIẾT KẾ CÁNH TAY ROBOT RÓT BIA

2.1. Bối cảnh

2.1.1. Xu hướng công nghệ trong thời đại Công nghiệp 4.0

Trong kỷ nguyên Công nghiệp 4.0, các hệ thống sản xuất và dịch vụ đang dần chuyển đổi theo hướng tự động hóa thông minh – nơi robot không chỉ thực hiện hành động theo chương trình có sẵn mà còn có khả năng cảm nhận, phân tích và tự đưa ra quyết định dựa trên dữ liệu đầu vào từ môi trường. Ba xu hướng công nghệ then chốt đang thúc đẩy sự phát triển của robot, đặc biệt là cánh tay robot, bao gồm:

- Trí tuệ nhân tạo (AI): Cho phép robot học cách nhận biết vật thể, phân tích hình ảnh, và tự ra quyết định.
- Internet of Things (IoT): Kết nối robot với các thiết bị khác như camera, cảm biến hoặc hệ thống giám sát từ xa, tạo nên môi trường điều khiển linh hoạt, có thể mở rộng.
- Tự động hóa tích hợp (Cyber-Physical Systems): Tăng cường khả năng điều phối giữa phần cứng (cánh tay robot) và phần mềm (xử lý ảnh, điều khiển AI), góp phần tạo ra hệ thống sản xuất dịch vụ thông minh.

2.1.2. Đánh giá nhu cầu thị trường

❖ **Trên thế giới**

Theo báo cáo từ nhiều tổ chức nghiên cứu công nghệ như IFR (International Federation of Robotics) và MarketsandMarkets, thị trường robot dịch vụ và robot cộng tác (cobot) đang tăng trưởng mạnh mẽ:

- Các quốc gia như Nhật Bản, Đức, Hàn Quốc, Mỹ đã đưa robot vào các mô hình dịch vụ tự động, từ nhà hàng sushi cho đến hệ thống pha cà phê tự phục vụ bằng robot.
- Robot tích hợp thị giác và AI có mức tăng trưởng kép hàng năm (CAGR) ấn tượng >20%.

❖ **Ở Việt Nam**

- Trong nước, nhu cầu về robot giá rẻ, linh hoạt, dễ tích hợp trong môi trường nhỏ (nhà hàng, quán bar, nhà máy vừa và nhỏ) đang ngày càng tăng.
- Các lĩnh vực như F&B (thực phẩm – đồ uống) đang tìm kiếm giải pháp tối ưu hóa vận hành, nhất là sau đại dịch COVID-19.
- Tuy nhiên, hiện tại phần lớn hệ thống robot ở Việt Nam còn đơn giản, chưa tích hợp AI hay thị giác máy một cách thực sự hiệu quả – tạo ra khoảng trống lớn cho nghiên cứu và phát triển.

→ Trong bối cảnh chuyển đổi số toàn cầu và nhu cầu thị trường thực tiễn, việc phát triển một hệ thống cánh tay robot 4 bậc tự do tích hợp xử lý ảnh và AI để rót bia tự động không chỉ là một bài toán kỹ thuật thú vị mà còn có giá trị ứng dụng cao, có thể nhân rộng cho nhiều mô hình dịch vụ thông minh tại Việt Nam và quốc tế.

2.2. Đặt vấn đề

Mặc dù cánh tay robot đã được phát triển và ứng dụng từ nhiều thập kỷ qua, tuy nhiên trong các ứng dụng dịch vụ đời sống như rót bia, pha chế, phục vụ khách hàng,... nhiều mô hình robot hiện tại vẫn chưa thể đáp ứng đầy đủ các yêu cầu về độ chính xác, tốc độ và chi phí. Một số hạn chế chính có thể liệt kê như sau:

2.2.1. Các hạn chế phổ biến

- **Chi phí đầu tư cao**

  Các robot thương mại đến từ Nhật Bản, Mỹ, Đức tuy có độ chính xác và độ tin cậy cao nhưng thường có giá từ vài chục triệu đến hàng trăm triệu đồng.

- **Độ chính xác chưa đáp ứng yêu cầu rót chất lỏng**

  Việc không tích hợp cảm biến lực hoặc thị giác chính xác làm giảm khả năng điều chỉnh vị trí theo thời gian thực.

- **Tốc độ xử lý chậm, độ trễ cao**

  Thời gian từ khi nhận diện hình ảnh → xử lý → phản hồi → điều khiển có thể dẫn đến độ trễ 1–2 giây, gây sai lệch nếu ly bị di chuyển hoặc rung nhẹ.

- **Khả năng thích ứng kém trong môi trường thay đổi**

  Robot không tích hợp AI hoặc học máy sẽ không thể tự điều chỉnh theo các thay đổi như: góc camera thay đổi, ánh sáng yếu,...

2.3. Thách thức

Trong quá trình triển khai hệ thống cánh tay robot 4 bậc tự do ứng dụng thị giác máy và trí tuệ nhân tạo (AI) để thực hiện nhiệm vụ rót bia chính xác vào ly, nhóm nghiên cứu đã xác định nhiều khó khăn lớn liên quan đến cả mặt kỹ thuật, chi phí triển khai, và tích hợp công nghệ phức tạp. Những thử thách này không chỉ ảnh hưởng đến tiến độ và chất lượng đề tài, mà còn là các rào cản lớn nếu muốn mở rộng ứng dụng vào thực tế sản xuất và dịch vụ.

2.3.1. Khó khăn kỹ thuật

a) Độ trễ hệ thống và phản hồi thời gian thực

Một hệ thống robot tự động rót bia phải hoạt động gần như theo thời gian thực. Quá trình bao gồm nhiều bước liên tục như: camera ghi hình → mô hình AI xác định vị trí ly → tính toán tọa độ thực → truyền lệnh đến robot → robot thực hiện thao tác. Mỗi bước đều có độ trễ riêng:

- Độ trễ xử lý ảnh: Nếu sử dụng mô hình như YOLOv8, việc xử lý ảnh đầu vào có thể mất từ 100 đến 500ms tùy thuộc cấu hình phần cứng.
- Độ trễ truyền thông: Giao tiếp giữa các thiết bị như camera – máy tính – vi điều khiển cũng tạo ra độ trễ, đặc biệt nếu dùng giao tiếp nối tiếp tốc độ thấp (UART).

Tổng hợp các yếu tố trên có thể dẫn đến độ trễ tổng lên đến 1–2 giây, gây sai lệch nghiêm trọng nếu ly bị dịch chuyển hoặc robot rót sai vị trí.

b) Sai số từ cảm biến và truyền động

- Sai số định vị từ camera: Các camera thông thường không được hiệu chuẩn chính xác sẽ có lỗi do méo hình, ánh sáng yếu, hoặc ảnh bị nhiễu. Điều này khiến việc tính toán tọa độ ly trong không gian thực không chính xác.
- Độ sai lệch từ động cơ: Động cơ thường gặp sai số ±1–3 độ góc, tương đương 5–10mm ở đầu cánh tay – gây ảnh hưởng nghiêm trọng đến vị trí miệng chai khi rót.

c) Độ phức tạp trong thuật toán điều khiển robot

Việc điều khiển một robot 4 bậc tự do tuy đơn giản hơn robot công nghiệp 6 bậc, nhưng vẫn yêu cầu xử lý:

- Động học nghịch để tính toán góc cần quay từ tọa độ thực của ly.
- Quy hoạch chuyển động để di chuyển mượt mà, không gây đổ bia, lệch hướng.
- Đồng bộ hóa các trục để đưa miệng chai đến đúng vị trí miệng ly, với độ nghiêng vừa đủ để rót đúng lượng bia.

2.3.2. Thách thức về chi phí triển khai

a) Chi phí phần cứng và vật liệu

Để đảm bảo hệ thống hoạt động ổn định và chính xác, một số thành phần bắt buộc phải đầu tư đúng mức:

- Camera có độ phân giải đủ cao và ổn định khung hình, đặc biệt là trong điều kiện ánh sáng yếu hoặc môi trường ngoài trời.
- Động cơ servo chất lượng tốt để giảm sai số định vị.
- Khung cơ khí nhẹ, chắc chắn, thường làm từ nhôm định hình hoặc nhựa in 3D chất lượng cao, có thể đội chi phí lên nhiều lần so với in nhựa giá rẻ.

b) Chi phí nhân lực và đào tạo

- Hệ thống cần người có khả năng lập trình Python (xử lý ảnh), C/C++ (vi điều khiển), kiến thức về AI (huấn luyện mô hình YOLO) và cơ khí động học (robot).

c) Chi phí bảo trì và thay thế

- Các thiết bị như servo, khung in 3D dễ hư hỏng khi hoạt động liên tục.
- Việc bảo trì, thay thế phụ tùng nếu không được thiết kế modul hóa sẽ phức tạp và tốn công.

2.3.3. Khó khăn trong tích hợp AI và phần mềm điều khiển

a) Tối ưu mô hình AI để chạy thực tế

+ Mặc dù mô hình YOLOv8 có thể chạy trên CPU hoặc các thiết bị như Jetson Nano, Raspberry Pi 4, nhưng vẫn cần tối ưu hóa tốc độ bằng cách giảm kích thước mô hình hoặc dùng định dạng TensorRT, ONNX.

+ Việc triển khai mô hình AI cần hiểu biết về môi trường ảo, thư viện OpenCV, PyTorch, hoặc Ultralytics.

b) Kết nối đa nền tảng

+ Camera thường giao tiếp qua USB nhưng vi điều khiển điều khiển robot lại qua UART, PWM → cần phần mềm trung gian xử lý việc đọc ảnh, suy luận AI, tính toán góc và gửi lệnh điều khiển.

c) Đồng bộ hóa thời gian thực

+ Mỗi bộ phận (camera, AI, vi điều khiển) có thời gian xử lý riêng, việc đồng bộ dữ liệu giữa các thành phần này là một thử thách lớn.

+ Một lỗi nhỏ về thời gian có thể khiến robot rót sai hoàn toàn – ví dụ AI phát hiện ly ở vị trí cũ nhưng ly đã bị người dùng dịch chuyển trong thực tế.

2.4. Giải pháp

Trong bối cảnh xu hướng tự động hóa hiện nay, giải pháp đề xuất nhằm hiện thực hóa một hệ thống cánh tay robot 4 bậc tự do thông minh, có khả năng tự động nhận diện ly, xác định vị trí và lượng bia trong ly, từ đó điều khiển động cơ để rót lượng bia chính xác cần thiết. Hệ thống kết hợp đồng thời ba mảng kỹ thuật chính: điều khiển động học chính xác, thị giác máy ứng dụng AI, và thiết kế phần cứng tối ưu cho thao tác thực tế.

2.4.1. Giải pháp điều khiển

a) Xây dựng mô hình động học bằng phương pháp Denavit-Hartenberg (DH)

✓ Để điều khiển chính xác vị trí đầu robot, nhóm đề tài sử dụng mô hình động học thuận/nghịch dựa trên phương pháp DH.

✓ Thông qua mô hình này, tọa độ của miệng ly được chuyển đổi thành chuỗi góc quay của từng khớp trên robot.

✓ Việc xây dựng bảng DH giúp đơn giản hóa việc tính toán ma trận biến đổi giữa các khớp, từ đó giảm tải cho vi điều khiển khi tính toán thời gian thực.

b) Điều khiển động cơ bằng động cơ servo AC công nghiệp

✓ Hệ thống sử dụng 4 động cơ servo AC, mỗi động cơ điều khiển một bậc tự do (DOF) của cánh tay robot.

✓ Các động cơ servo được điều khiển thông qua bộ điều khiển servo công nghiệp, đảm bảo khả năng điều khiển chính xác vị trí, tốc độ và mô-men xoắn theo yêu cầu.

✓ Bộ điều khiển trung tâm tính toán các tham số động học và phát tín hiệu điều khiển thông qua giao thức xung (Pulse/Direction) đến bộ driver servo.

✓ Hệ thống động cơ servo AC công nghiệp cho phép cánh tay robot vận hành với độ chính xác cao, chuyển động mượt mà, ổn định, chịu được tải trọng lớn và thích hợp cho các ứng dụng tự động hóa trong công nghiệp.

✓ Giải pháp sử dụng servo AC giúp nâng cao hiệu suất vận hành, độ tin cậy và tuổi thọ cho toàn bộ hệ thống robot.

c) Ổn định chuyển động bằng thuật toán PID

✓ Đối với những chuyển động yêu cầu độ chính xác cao và mượt, đặc biệt là khi robot dừng đúng vị trí để rót bia, thuật toán PID được sử dụng để tinh chỉnh độ lệch giữa góc đặt và góc thực tế.

✓ Thuật toán PID giúp đảm bảo robot không rung lắc khi dừng, và thao tác rót bia diễn ra trơn tru, không bị giật.

2.4.2. Thị giác máy và AI

a) Nhận diện miệng ly bằng YOLOv8

- Mô hình YOLOv8 được huấn luyện với ảnh thực tế để nhận diện miệng ly – đối tượng hình tròn với đường kính xác định.
- Sau khi xác định bounding box quanh miệng ly, hệ thống tính toán tọa độ tâm và quy đổi từ tọa độ ảnh sang hệ trục robot để điều khiển đầu rót di chuyển chính xác.

b) Xác định lượng bia bằng xử lý hình ảnh trong bounding box thân ly

- Sau khi robot định vị được vị trí ly, hệ thống tiếp tục nhận diện thân ly và trích xuất ảnh nằm trong bounding box này.
- Các bước xử lý ảnh như sau:

+ Chuyển ảnh sang không gian màu HSV để dễ phân biệt màu sắc.

+ Sử dụng hàm cv2.inRange() để lọc vùng màu vàng – đại diện cho bia.

+ Các màu khác được gán giá trị 0 (màu đen) nhằm loại bỏ nhiễu.

+ Đếm số lượng pixel màu vàng, so sánh với chiều cao của ly để ước lượng phần trăm bia còn lại.

- Từ kết quả này, hệ thống tính toán lượng bia cần rót thêm cho đủ mức mong muốn.

2.4.3. Giải pháp cơ khí và vật liệu

✓ Toàn bộ cánh tay robot được chế tạo từ vật liệu sắt, nhằm:

+ Tăng độ cứng vững khi nâng chai bia – vật nặng và tròn.

+ Chịu tải tốt, hạn chế biến dạng trong quá trình di chuyển, rót bia.

+ Tăng tuổi thọ sử dụng trong môi trường có độ ẩm cao như quán bar, nhà hàng.

- ✓ Bộ gắp cũng được thiết kế chắc chắn để giữ chặt chai bia, tránh trượt khi nghiêng để rót.

2.4.4. Tích hợp phần cứng và phần mềm

a) Phần cứng

- **Hệ thống thu nhận hình ảnh:** Sử dụng camera kết nối qua cổng USB, lắp đặt phía trước vùng làm việc của robot để thu thập hình ảnh phục vụ cho quá trình nhận diện và xử lý.
- **Bộ xử lý hình ảnh:** Raspberry Pi 4 đảm nhiệm chức năng xử lý ảnh, vận hành các thuật toán nhận dạng đối tượng (YOLOv8) và phân tích màu sắc để xác định vị trí ly và lượng bia cần rót.
- **Bộ vi điều khiển điều khiển chuyển động:** STM32F103C8T6 thực hiện tính toán động học và điều khiển chuyển động các khớp robot dựa trên dữ liệu nhận được từ bộ xử lý hình ảnh, tạo tín hiệu điều khiển truyền đến các bộ driver servo.
- **Hệ thống truyền động:**

  + Trang bị 4 động cơ servo AC Delta ECMA-C20604SS, mỗi động cơ điều khiển một bậc tự do (DOF) trên cánh tay robot.

  + Công suất mỗi động cơ: 400W; điện áp cấp: 220V AC; dòng điện vận hành định mức: 2.6A; dòng điện cực đại: 7.8A.
- **Bộ điều khiển động cơ (servo driver):**

  + Sử dụng bộ điều khiển servo Delta ASD-A2-0421-L chuyên dụng, công suất 400W, điện áp cấp 220V AC (1 pha).

  + Hỗ trợ nhiều chế độ điều khiển vị trí, tốc độ, mô-men xoắn, đồng thời có khả năng giao tiếp truyền thông công nghiệp như EtherCAT, CANopen, bảo đảm khả năng mở rộng và tích hợp cao.
- **Hệ thống cấp nguồn:** Trang bị bộ nguồn ổn áp 5V cấp nguồn cho vi điều khiển, camera và các thiết bị ngoại vi khác trong hệ thống.

b) Phần mềm

- Python + OpenCV cho xử lý ảnh đầu vào.
- YOLOv8 (Ultralytics) thực hiện nhận diện đối tượng trong thời gian thực.
- Thuật toán xử lý màu HSV để xác định lượng bia.
- Giao tiếp giữa Raspberry Pi và vi điều khiển STM32F103C8T6 qua UART (Serial) để truyền lệnh góc quay từ mô hình DH.

→ Tổng kết giải pháp

Giải pháp đề tài không chỉ dừng lại ở việc điều khiển robot cơ bản, mà còn kết hợp AI, thị giác máy và điều khiển chính xác, giúp robot có thể:

- Tự phát hiện ly và ước lượng lượng bia.
- Tính toán góc di chuyển chính xác.
- Rót bia với lượng vừa đủ, hạn chế tràn hoặc rót thừa.

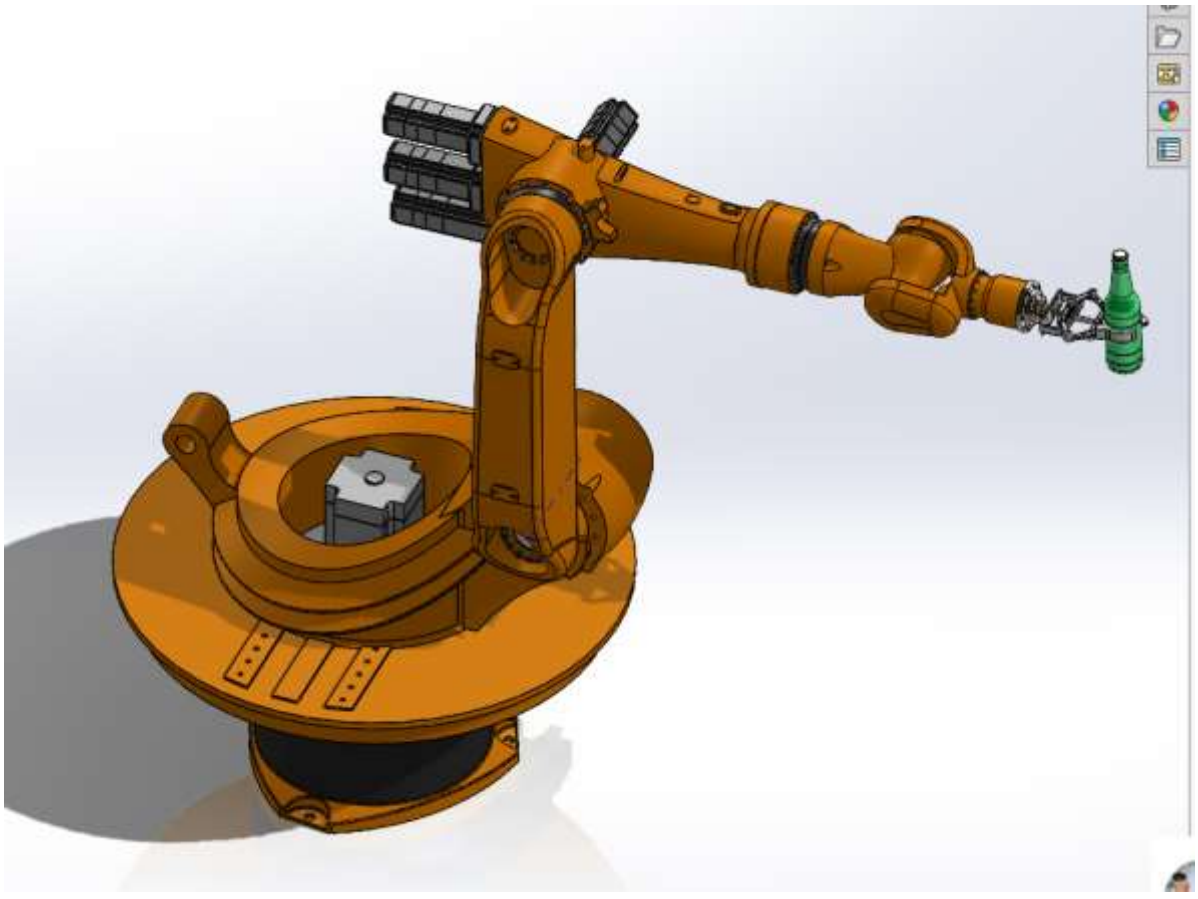
Hệ thống mang tính thực tiễn cao, có thể áp dụng cho các mô hình nhà hàng thông minh, bar tự động, hoặc dây chuyền sản xuất đóng chai, đồng thời dễ mở rộng trong tương lai.

2.5. Thiết kế

2.5.1. Sơ đồ công nghệ

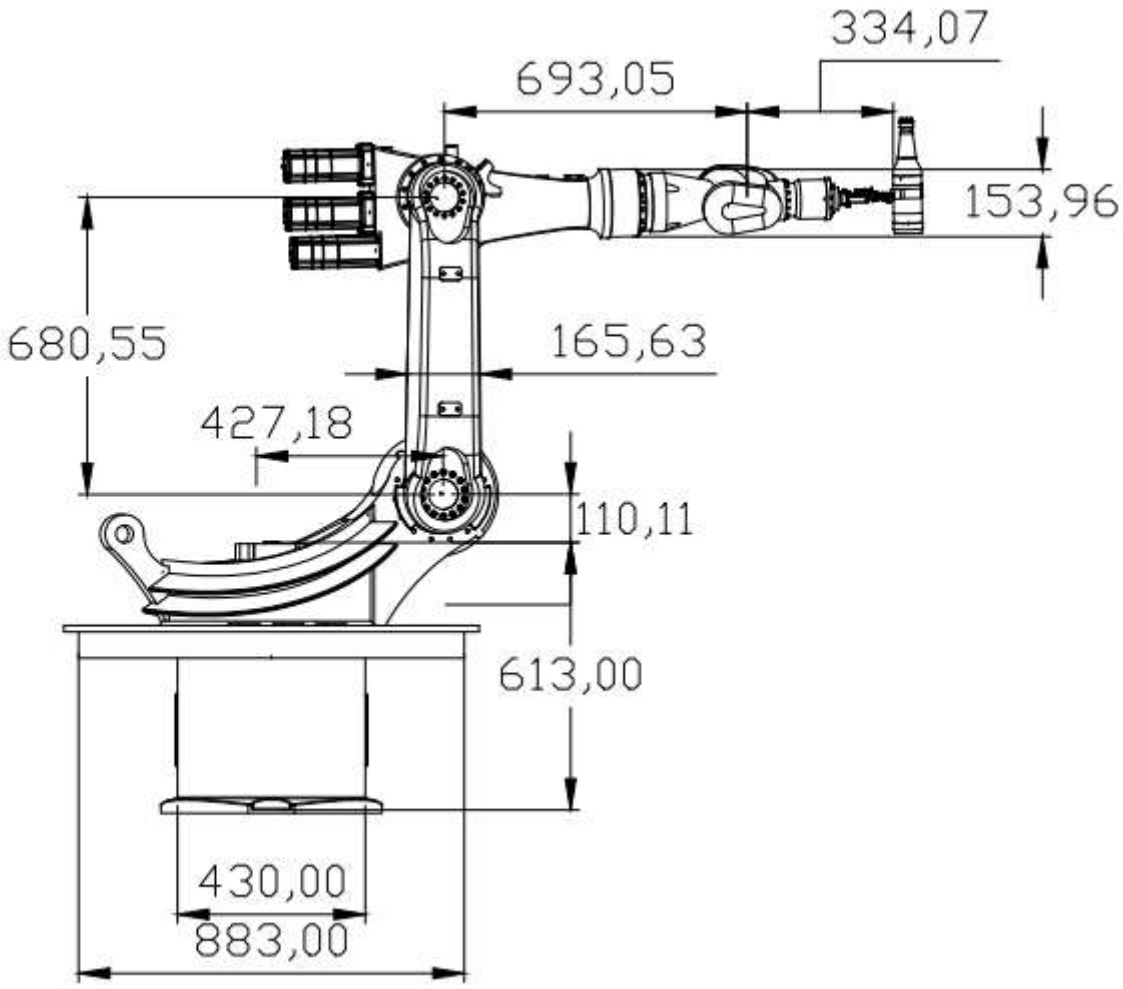
Quy trình hoạt động của hệ thống được mô tả thông qua sơ đồ công nghệ dưới đây.

Để thể hiện rõ cấu trúc cơ khí và hình dạng tổng thể của cánh tay robot, nhóm vẽ sơ đồ công nghệ nhằm thể hiện vị trí các khớp, chiều cao từng đoạn tay và các thành phần cơ khí quan trọng, phục vụ cho quá trình thiết kế, chế tạo và lắp ráp.



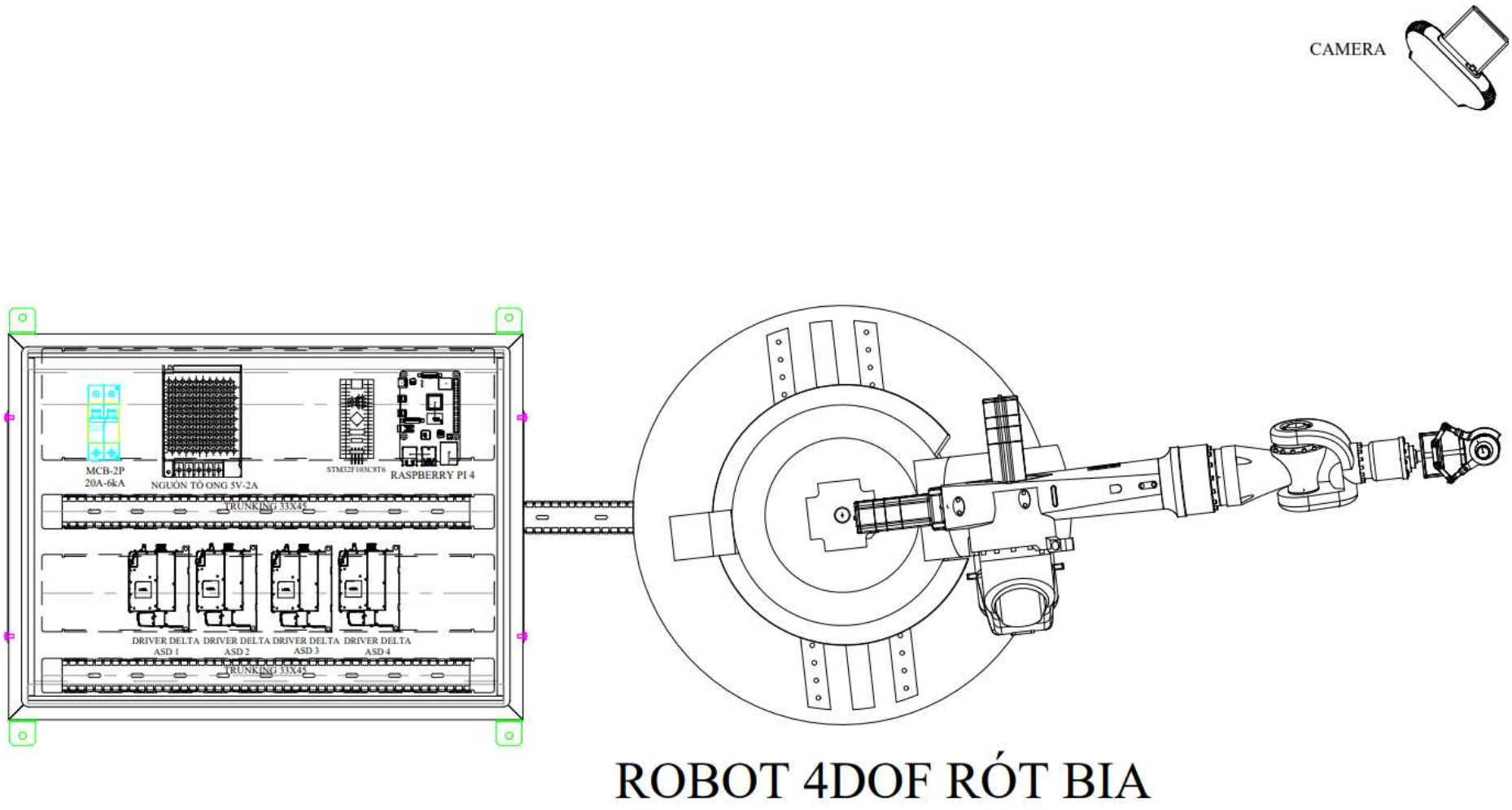
Hình 2.1: Sơ đồ công nghệ cánh tay robot tự động rót bia

Để phục vụ cho quá trình thiết kế, chế tạo và lắp ráp, nhóm đã thực hiện bản vẽ thể hiện kích thước chi tiết của cánh tay robot



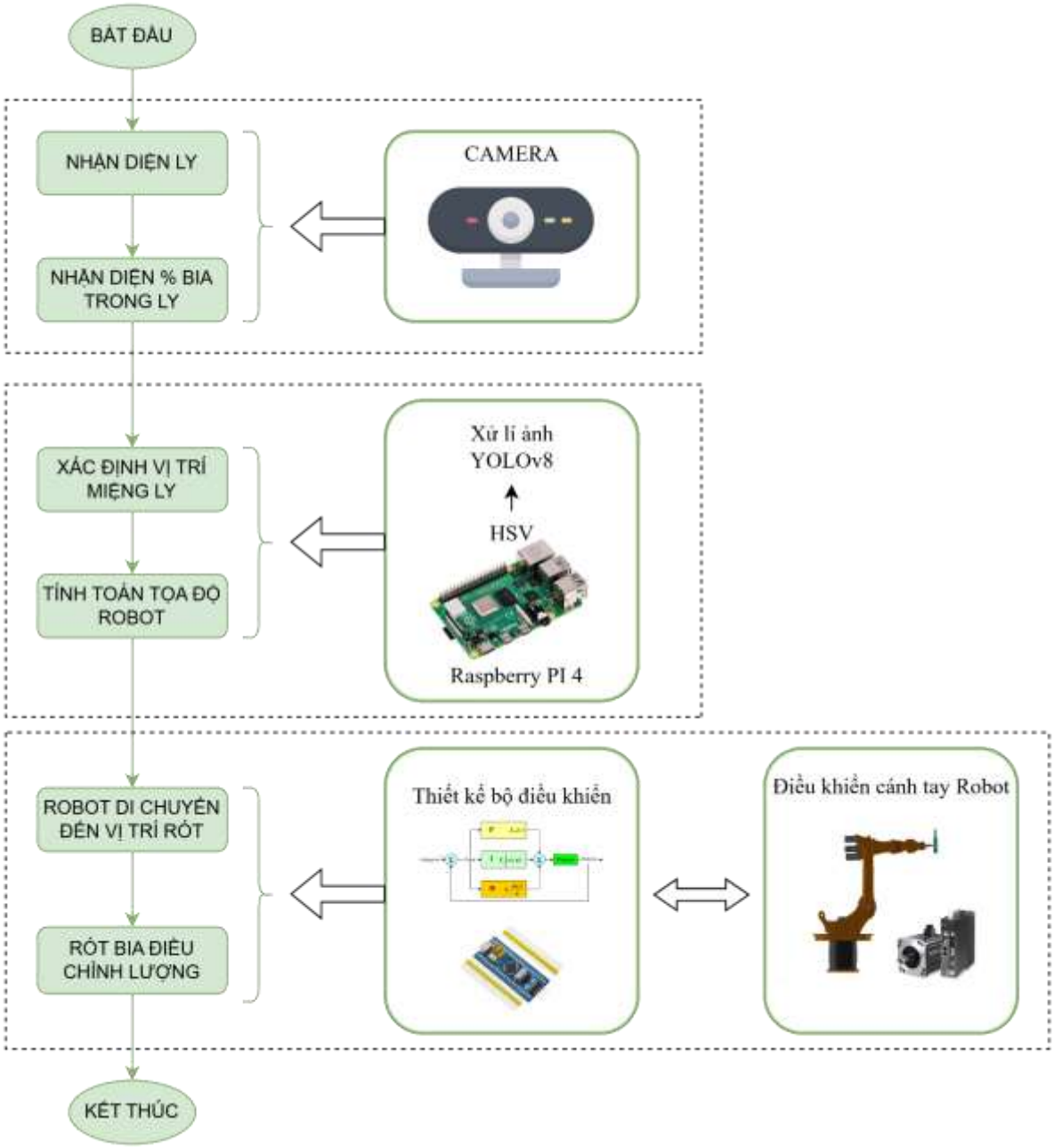
Hình 2.2: Kích thước của cánh tay Robot

Do sơ đồ công nghệ không thể hiện đầy đủ vị trí của tủ điều khiển cũng như cách bố trí các thiết bị trên mặt phẳng làm việc, nhóm đã xây dựng thêm hình chiếu bằng để thể hiện rõ hơn mối liên hệ và vị trí tương đối giữa cánh tay robot, tủ điều khiển và các thiết bị ngoại vi khác.



Hình 2.3: Sơ đồ công nghệ tủ điều khiển

Để hệ thống hoạt động một cách trơn tru và chính xác, các bước xử lý cần được thực hiện theo một trình tự nhất định. Phần dưới đây trình bày quy trình công nghệ của hệ thống cánh tay robot rót bia.



Hình 2.4: Quy trình công nghệ

Sơ đồ mô tả toàn bộ quy trình hoạt động của hệ thống rót bia tự động, sử dụng sự kết hợp giữa camera AI, xử lý ảnh bằng trí tuệ nhân tạo, tính toán điều khiển và cánh tay robot thực thi. Đây là một hệ thống tích hợp nhiều lĩnh vực công nghệ như thị giác máy tính (Computer Vision), điều khiển tự động, robot học và trí tuệ nhân tạo (AI).

a) Camera AI – Khối thu nhận dữ liệu hình ảnh

Đây là thiết bị đầu vào của toàn bộ hệ thống, sử dụng camera (thường là webcam độ phân giải cao) để thu nhận hình ảnh thực tế tại khu vực làm việc – nơi đặt ly bia. Camera hoạt động liên tục và ghi lại hình ảnh thời gian thực. Dữ liệu hình ảnh này sẽ là đầu vào để xử lý nhận dạng đối tượng và phân tích mức chất lỏng. Vị trí lắp đặt camera được tính toán sao cho góc nhìn bao quát được toàn bộ khu vực ly bia, đồng thời tránh nhiễu từ ánh sáng môi trường.

b) Xử lý ảnh – Nhận diện đối tượng và phân tích mức bia

Khối xử lý ảnh được thực hiện trên máy tính hoặc vi điều khiển có hỗ trợ AI (Raspberry Pi 4). Tại đây, hai kỹ thuật xử lý ảnh chính được sử dụng:

✓ **Nhận diện đối tượng bằng YOLOv8**

+ YOLOv8n là mô hình AI hiện đại, phiên bản nhẹ (nano), tối ưu cho thiết bị có tài nguyên hạn chế.

+ Mô hình được huấn luyện để nhận diện miệng ly và thân ly trong ảnh.

+ Sau khi nhận diện, hệ thống trả về tọa độ bounding box (tọa độ pixel) của các đối tượng, cụ thể là:

– Tọa độ chính xác của miệng ly – để định vị điểm cần rót.

– Tọa độ thân ly – để phân tích mức bia trong ly.

✓ **Phân tích mức bia bằng xử lý màu HSV**

+ Sau khi biết vùng chứa thân ly, hệ thống chuyển ảnh sang không gian màu HSV để lọc màu vàng đặc trưng của bia.

+ Kỹ thuật phân ngưỡng HSV giúp loại bỏ các vùng không phải là bia, giữ lại vùng chứa chất lỏng.

+ Bằng cách đo chiều cao vùng màu vàng so với chiều cao toàn bộ thân ly, hệ thống ước lượng phần trăm lượng bia đang có trong ly.

c) Tính toán điều khiển – Chuyển đổi tọa độ và điều khiển động cơ

Sau khi xử lý ảnh xong, hệ thống sẽ thực hiện các tác vụ điều khiển, bao gồm:

• **Chuyển đổi tọa độ ảnh sang tọa độ robot**

+ Do camera và robot nằm ở hai hệ quy chiếu khác nhau, nên cần chuyển đổi từ tọa độ pixel sang tọa độ thực tế.

+ Quá trình này sử dụng ma trận biến đổi tọa độ.

• **Tính toán vị trí và góc nghiêng phù hợp**

+ Dựa trên vị trí miệng ly và mức bia hiện tại, hệ thống tính toán:

– Vị trí cần di chuyển tới.

– Góc nghiêng để rót đúng lượng bia còn thiếu.

• **Điều khiển phần cứng**

+ Các tín hiệu điều khiển được gửi đến driver điều khiển động cơ Driver Delta ASD điều khiển các trục X, Y, Z.

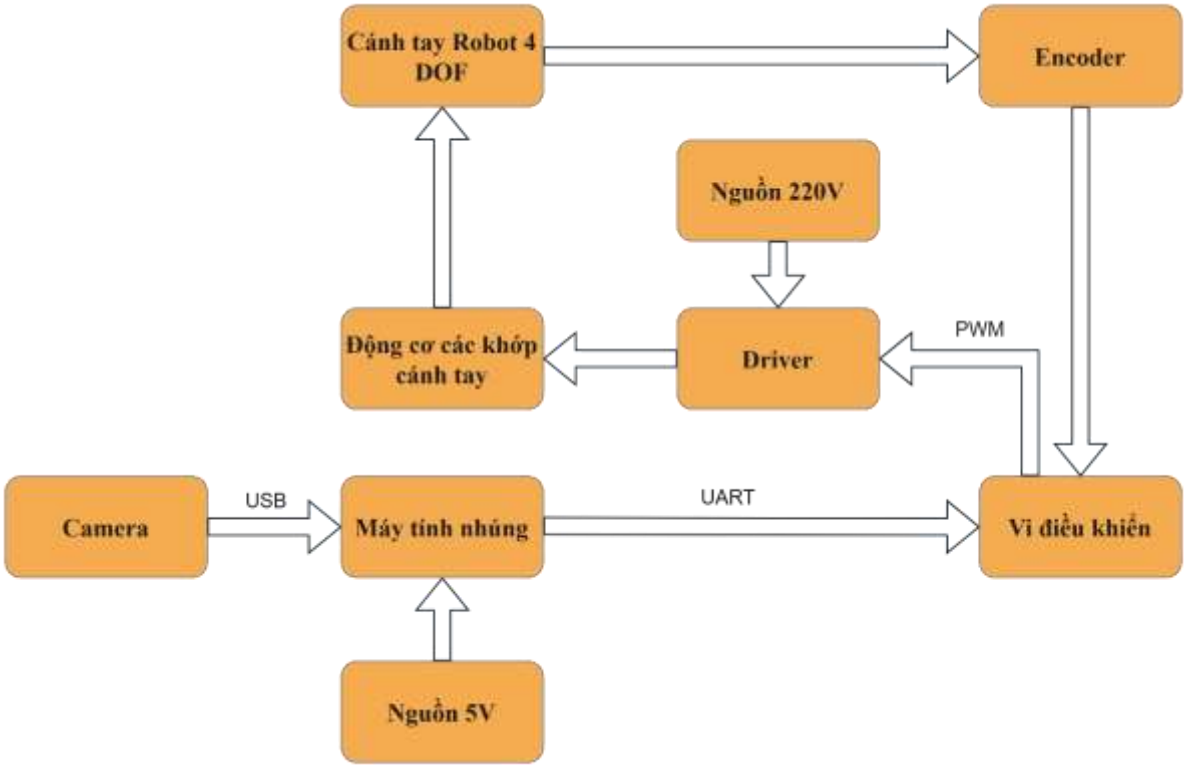
+ Hệ thống tích hợp thuật toán PID hoặc điều khiển theo quỹ đạo để đảm bảo cánh tay di chuyển mượt mà, chính xác.

d) Cánh tay robot – Thực thi hành động rót bia

Đây là phần tử chấp hành của toàn hệ thống:

- Cánh tay nhận lệnh từ bộ điều khiển và di chuyển chai đến đúng vị trí miệng ly.
- Sau khi đến đúng vị trí, robot nghiêng chai bia cố định để rót bia vào ly.
- Hệ thống giám sát hình ảnh để cập nhật mức bia trong ly theo thời gian thực.
- Khi hệ thống xác định đã rót đủ, robot sẽ dừng rót và quay về vị trí ban đầu.

Để người đọc dễ hình dung cách hệ thống hoạt động, nhóm đã xây dựng sơ đồ kết nối tổng quan. Sơ đồ này giúp thấy rõ các thiết bị sử dụng trong hệ thống và cách chúng được liên kết với nhau.



Hình 2.5: Sơ đồ kết nối tổng quan của hệ thống

2.5.2. Sơ đồ kết nối

Hệ thống bao gồm các thiết bị chính: USB Camera, Raspberry PI 4, STM32F103C8T6, Driver Delta ASD-A2-0421-L, động cơ servo AC Delta ECMA-C20604SS cánh tay robot và nguồn tổ ong 5V-2A. Các thiết bị được kết nối như sau:

Camera nối với Raspberry PI 4

+ Camera được kết nối trực tiếp vào cổng USB của Raspberry Pi 4.

+ Dữ liệu hình ảnh thu nhận từ camera được truyền vào Raspberry Pi 4 để thực hiện quá trình xử lý ảnh và nhận diện đối tượng.

Raspberry PI 4 nối với Vi điều khiển

+ Sử dụng chuẩn giao tiếp nối tiếp UART (Serial).
+ Raspberry Pi 4 sau khi xử lý ảnh và xác định được các tham số vị trí, sẽ truyền dữ liệu tọa độ và lệnh điều khiển sang STM32 thông qua cổng UART để điều khiển chuyển động robot.

Nguồn tổ ong 24V – 5A đấu với điện lưới 220V

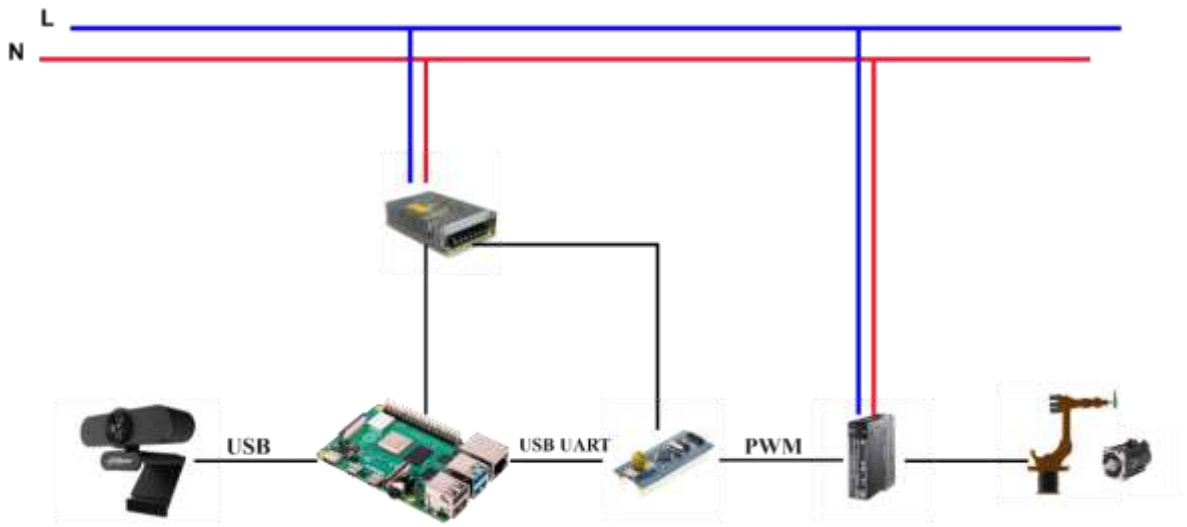
+ Sử dụng nguồn tổ ong 5V-2A để cấp nguồn ổn định cho Raspberry Pi 4, STM32F103C8T6 và các mạch điều khiển phụ trợ.
+ Bộ nguồn được đấu nối vào điện lưới 220V AC với các chân kết nối: L (dây pha), N (dây trung tính) và GND (tiếp đất).
+ Đảm bảo các yêu cầu an toàn điện trong quá trình đấu nối với dây dẫn 2 lõi bọc cách điện.

Kết nối Delta ASD-A2-0421-L với động cơ servo AC Delta ECMA-C20604SS

+ **Kết nối nguồn động lực:** Đầu ra U, V, W của driver servo được nối tương ứng với các đầu U, V, W của động cơ servo.
+ **Kết nối tín hiệu phản hồi encoder:** Cáp encoder từ động cơ được nối về cổng CN2 của driver servo, bao gồm các tín hiệu A+, A-, B+, B-, Z+, Z-, cung cấp thông tin phản hồi vị trí chính xác về bộ điều khiển.

Số lượng: Lặp lại cho 4 cặp ASDA-A2 và động cơ (mỗi khớp một cặp).

Từ nguyên lý kết nối giữa các thiết bị, sơ đồ đấu dây dưới đây trình bày chi tiết cách các phần tử trong hệ thống được cấp nguồn và truyền tín hiệu điều khiển.



Hình 2.6: Sơ đồ nối dây

CHƯƠNG 3: CƠ SỞ LÝ THUYẾT TÍNH TOÁN ROBOT RÓT BIA

Trong chương 3 nhóm sẽ làm rõ những vấn đề sau

Hình 3.1: Cấu trúc chương 3

3.1. Mô hình hóa cánh tay robot

Mục đích: Xác định mối quan hệ giữa các biến khớp (góc quay hoặc khoảng cách trượt) và vị trí/hướng của điểm đầu cuối trong không gian.

3.1.1 Lý thuyết động học thuận

a) Lý thuyết

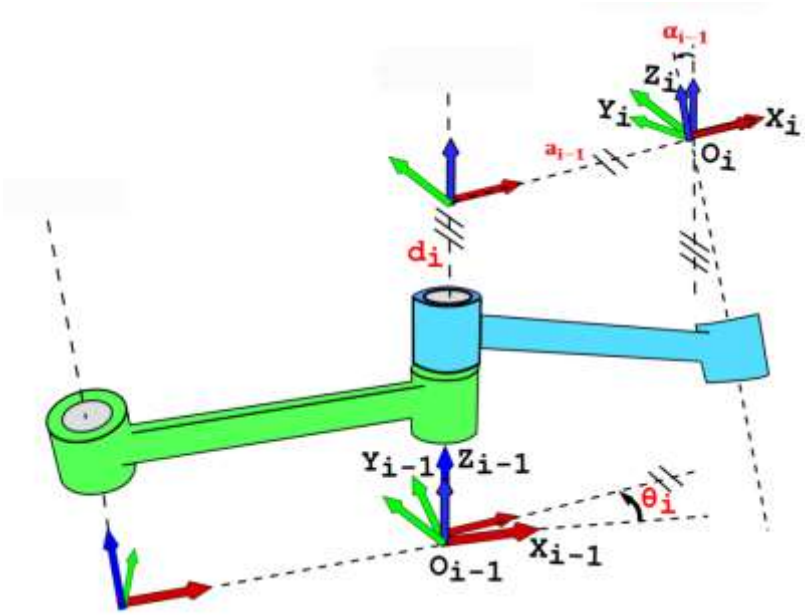
Bài toán động học thuận nhằm xác định vị trí và hướng của điểm đầu cuối trong hệ tọa độ Descartes so với hệ tọa độ gốc, dựa trên các tham số khớp và khâu đã cho. Các bước giải bằng phương pháp Denavit-Hartenberg (DH) được tóm tắt như sau

Hình 3.2: Các bước giải bằng phương pháp DH

Bước 1: Xác định số bậc tự do của robot và tại mỗi khớp là chuyển động gì, thông thường có hai chuyển động là quay và tịnh tiến.

Bước 2: Gắn các hệ trục tại mỗi khâu hoặc mỗi khớp cho robot.

Bước 3: Xác định các thông số của Denavit-Hartenberg.



Hình 3.3: Mô hình thiết lập hệ trục Denavit-Hartenberg (DH) cho robot

Tính toán động học thuận bằng phương pháp Denavit-Hartenberg là phương pháp đặt các trục tọa độ tại các khớp của robot. Ta có ma trận chuyển đổi tổng quát giữa hệ trục thứ {i-1} và hệ trục thứ {i}

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.1) $$

Bảng 3.1: Chú thích

θ_i	Góc quay quanh trục Z_{i-1} để đưa hệ trục về vị trí mong muốn.
α_{i-1}	Góc xoay quanh trục x_i từ trục z_{i-1} đến trục z_i
d_i	Khoảng cách dọc theo trục Z_{i-1} từ gốc O_{i-1} đến giao điểm với trục x_i
a_{i-1}	Khoảng cách dọc theo trục x_i từ gốc O_i đến giao điểm với trục z_i

Bảng 3.2: Bộ tham số Denavit-Hartenberg

i	a_{i-1}	a_{i-1}	d_i	θ_i
1	a_0	a_0	d_1	θ_1
2	a_1	a_1	d_2	θ_2
…	…	…	…	…
n	a_{n-1}	a_{n-1}	d_n	θ_n

Bước 4: Tính toán các ma trận chuyển đổi của từng khâu.

Bước 5: Tìm ra mối liên hệ giữa hệ trục tọa độ gốc và hệ trục tọa độ cuối cùng.

Tìm ma trận tổng T_n^0

+ Nhân các ma trận chuyển đổi để tìm mối quan hệ giữa hệ tọa độ góc và hệ tọa độ cuối

$$ T_n^0 = T_1^0 \times T_2^1 \times ... \times T_n^{n-1} \quad (3.2) $$

+ Ma trận T_n^0 gồm

Ma trận xoay R_n^0 : Hướng đầu cuối

Ma trận vị trí P_n^0 : Vị trí đầu cuối

Xác định mỗi khớp là chuyển động quay (rotational) hay tịnh tiến (translational).

+ Lập bảng tham số DH

- **Tính ma trận chuyển đổi đồng nhất** T_i^{i-1}

+ Sử dụng công thức ma trận chuyển đổi DH

$$ T_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_{i-1} & \sin\theta_i\sin\alpha_{i-1} & \alpha_{i-1} \\ \sin\theta_i & \cos\theta_i\cos\alpha_{i-1} & -\cos\theta_i\sin\alpha_{i-1} & 0 \\ 0 & \sin\alpha_{i-1} & \cos\alpha_{i-1} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.3) $$

+ Tính T_i^{i-1} cho từng khâu từ i = 1 đến i = n

- **Tính ma trận tổng** T_n^0

+ Nhân các ma trận chuyển đổi để tìm mối quan hệ giữa hệ tọa độ gốc và hệ tọa độ cuối

$$T_n^0 = T_1^0 \times T_2^1 \times ... \times T_n^{n-1} \tag{3.4}$$

Tính toán động học thuận cho cánh tay máy

+ Ma trận xoay:

$$ {}^A_B R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.5}$$

${}^A_B R$ là ma trận xoay 3x3 thể hiện phép xoay từ hệ trục tọa độ B sang hệ tọa độ A

+ Ma trận vị trí:

$$ {}^A P_{BORG} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{3.6}$$

${}^A P_{BORG}$ là ma trận 3x1 biểu thị điểm P của hệ tọa độ B trong hệ tọa độ A.

+ Ma trận chuyển đổi:

$$ {}^A_B T = \begin{bmatrix} & {}^A_B R & & {}^A P_{BORG} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

${}^A_B T$ là ma trận chuyển đổi đồng nhất biến đổi từ hệ tọa độ B sang hệ tọa độ A,

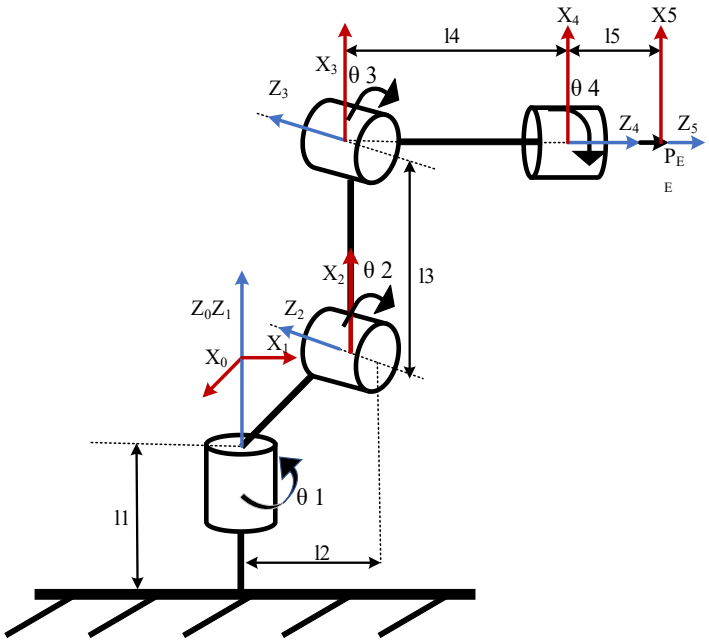
- Để biến đổi giữa hệ trục (n-1) sang hệ trục n, ta có:

$$ {}^{n-1}_n T = R_{X_{i-1}}(\alpha_{i-1}).T_{X_{i-1}}(a_{i-1}).R_{Z_i}(\theta_i).T_{Z_i}(d_i) \tag{3.8}$$

b) Tính toán động học thuận cho cánh tay robot rót bia

Bước 1: Robot có tổng cộng 4 bậc tự do, tại mỗi khớp đều là chuyển động xoay.

Bước 2: Đặt các hệ trục cho Robot.



Hình 3.4: Đặt hệ trục tọa độ cho robot

Bảng 3.3: Chú thích các khoảng cách và hướng trong hệ tọa độ robot

d_1	khoảng cách giữa hai trục $\hat{Z}_1$ đến $\hat{Z}_2$ tính dọc theo trục $\hat{X}_1$
d_2	khoảng cách giữa hai mặt phẳng chứa $\hat{X}_0$ đến $\hat{X}_1$ cùng vuông góc với trục $\hat{Z}_0$
d_3	khoảng cách giữa hai trục $\hat{Z}_3$ đến $\hat{Z}_4$ tính dọc theo trục $\hat{X}_3$
L_2	khoảng cách giữa hai trục

Bước 3: Xác định các thông số của Denavit-Hartenberg.

Bảng 3.4: Thông số bảng DH

i	a_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	L_2	−90	0	θ_2
3	L_3	0	0	θ_3
4	0	−90	L_4	θ_4
5	0	0	L_5	0

Bước 4: Tính toán các ma trận chuyển đổi của từng khâu.

Ta có ma trận chuyển đổi tổng quát từ hệ $i-1$ sang hệ thứ Z_i

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & \alpha_{i-1} \\ s\theta_i c\alpha_{i-1} & s\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.9) $$

Ma trận chuyển đổi đồng nhất từ hệ 1 sang hệ 0

$$ {}^{0}_{1}T = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.10) $$

Ma trận chuyển đổi đồng nhất từ hệ 2 sang hệ 1:

$$ {}^{1}_{2}T = \begin{bmatrix} c_2 & -s_2 & 0 & L_2 \\ 0 & 0 & 1 & 0 \\ -s_2 & -c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.11) $$

Ma trận chuyển đổi đồng nhất từ hệ 3 sang hệ 2:

$$ {}^{2}_{3}T = \begin{bmatrix} c_3 & -s_3 & 0 & L_3 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.12) $$

Ma trận chuyển đổi đồng nhất từ hệ 4 sang hệ 3:

$$ {}^{3}_{4}T = \begin{bmatrix} c_4 & -s_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_4 & -c_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.13) $$

Ma trận chuyển đổi đồng nhất từ hệ 5 sang hệ 4:

$$ {}^{4}_{5}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.14) $$

Bước 5: Tìm ra mối liên hệ giữa hệ trục tọa độ gốc và hệ trục tọa độ cuối cùng của Robot.

Ta có ma trận chuyển đổi đồng nhất từ hệ 5 sang hệ 0 là

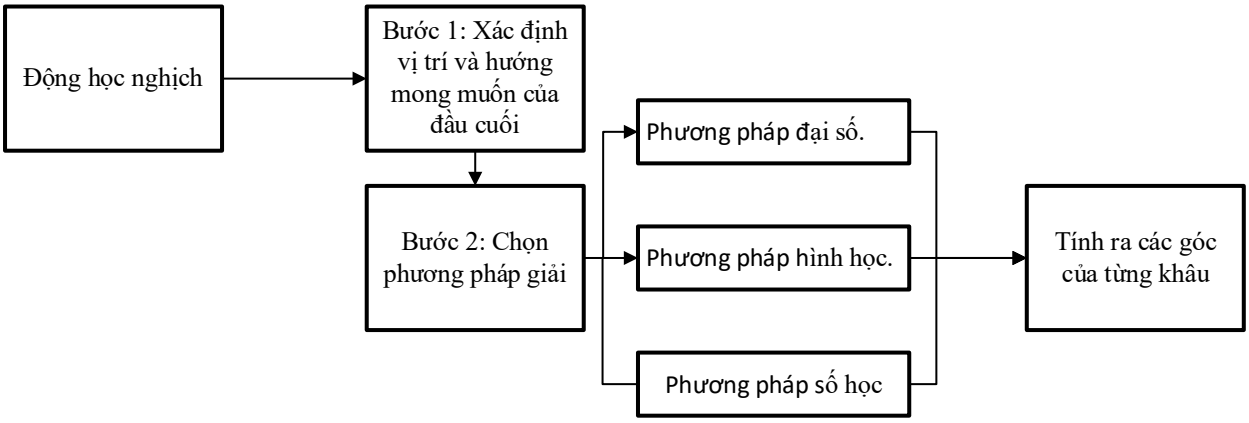
$$ {}^{0}_{5}T = {}^{0}_{1}T^{1}_{2}T^{2}_{3}T^{3}_{4}T^{4}_{5}T \quad (3.15) $$

$$= \begin{bmatrix} s_1s_4 - c_4(c_1s_2s_3 - c_1c_2c_3) & c_4s_1 + s_4(c_1s_2s_3 - c_1c_2c_3) & -s_{23}c_1 & c_1(L_2 - L_4s_{23} - L_5s_{23} + L_2c_3) \\ -c_1s_4 - c_4(s_1s_2s_3 - s_1c_2c_3) & s_4(s_1s_2s_3 - s_1c_2c_3) - c_1c_4 & -s_{23}s_1 & s_1(L_2 - L_4s_{23} - L_5s_{23} + L_2c_3) \\ -s_{23}c_4 & s_{23}s_4 & -c_{23} & -L_4c_{23} - L_5c_{23} - L_3s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Vị trí điểm đầu cuối so với hệ 0:

$$^0P_5 = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} c_1(L_2 - L_4s_{23} - L_5s_{23} + L_3c_2) \\ s_1(L_2 - L_4s_{23} - L_5s_{23} + L_2c_3) \\ -L_4c_{23} - L_5c_{23} - L_3s_2 \end{bmatrix} \quad (3.16)$$

3.1.2 Động học nghịch



Hình 3.5: Cấu trúc mục 3.1.2

Bài toán động học nghịch là tìm tất cả tập hợp vị trí góc quay tại các khớp, khi cho trước vị trí và hướng của điểm cuối robot để tạo ra chuyển động cho cơ cấu chấp hành. Tuy nhiên đối với bài toán động học nghịch sẽ khó giải quyết hơn vì tại mỗi điểm đầu cuối của robot sẽ có thể cho ra nhiều tập hợp các góc quay khác nhau tại mỗi khớp.

Các phương pháp giải bài toán động học nghịch: Ngày nay có rất nhiều phương pháp để giải bài toán về động học nghịch tuy nhiên vẫn dựa trên 3 phương pháp chính sau:

- ✓ **Phương pháp đại số.**
- ✓ Phương pháp hình học.
- ✓ Phương pháp số học.

- Tính toán động học nghịch

Tính động học nghịch (Inverse Kinematics) bằng phương pháp đại số

- Tính θ_1
- Tính θ_3

$\rightarrow$ Tính θ_2

$\rightarrow$ Dựa vào góc rót bia tìm được θ_4

Đặt

$$\begin{aligned} n_x &= p_y c_1 + p_x s_1 \\ n_y &= p_y c_1 - p_x s_1 \\ n_z &= p_z \\ m_x &= L_2 - L_4 s_{23} - L_5 s_{23} + L_3 c_2 \\ m_y &= 0 \\ m_z &= -L_4 c_{23} - L_5 c_{23} - L_3 s_2 \end{aligned} \tag{3.17}$$

- ✓ **Tính** θ_1

Từ $n_y = m_y$ ta có

$$p_y c_1 - p_x s_1 = 0 \tag{3.18}$$

Chuyển vế ta được

$$\frac{p_y}{p_x} = \frac{s_1}{c_1} \tag{3.19}$$

Góc θ_1 tìm được là

$$\theta_1 = arctg2(p_y, p_x) \tag{3.20}$$

- ✓ **Tính** θ_3

Ta có $n_x = m_x$ và $n_z = m_z$ hay

$$\begin{cases} p_x c_1 + p_y s_1 = L_2 - L_4 s_{23} - L_5 s_{23} + L_3 c_2 \\ p_z = -L_4 c_{23} - L_5 c_{23} - L_3 s_2 \end{cases} => \begin{cases} p_x c_1 + p_y s_1 - L_2 = L_3 c_2 - (L_5 + L_4) s_{23} \\ p_z = -L_3 s_2 - (L_5 + L_4) c_{23} \end{cases} (*)$$

Bình phương 2 vế hệ phương trình ta được:

$$\begin{cases} (p_x c_1 + p_y s_1 - L_2)^2 = (L_3 c_2 - (L_5 + L_4) s_{23})^2 \\ {p_z}^2 = (-L_3 s_2 - (L_5 + L_4) c_{23})^2 \end{cases}$$

Từ (*) cộng vế theo vế ta được:

$$(p_x c_1 + p_y s_1 - L_2)^2 + {p_z}^2 = (L_3 c_2 - (L_5 + L_4) s_{23})^2 + (-L_3 s_2 - (L_5 + L_4) c_{23})^2$$

$$(p_x c_1 + p_y s_1 - L_2)^2 + {p_z}^2 = (L_3 c_2)^2 - 2L_3 c_2((L_5 + L_4) s_{23}) + ((L_5 + L_4) s_{23})^2 + (L_3 s_2)^2 + 2((L_5 + L_4) c_{23}) L_3 s_2 + ((L_5 + L_4) c_{23})^2$$

$$(p_x c_1 + p_y s_1 - L_2)^2 + {p_z}^2 = {L_3}^2({c_2}^2 + {s_2}^2) + (L_5 + L_4)^2({c_{23}}^2 + {s_{23}}^2) - 2L_3 c_2 (L_5 + L_4) s_{23} + 2(L_5 + L_4) c_{23} L_3 s_2$$

$$(p_x c_1 + p_y s_1 - L_2)^2 + {p_z}^2 - {L_3}^2 - (L_5 + L_4)^2 = -2L_3 c_2 (L_5 + L_4) s_{23} + 2(L_5 + L_4) c_{23} L_3 s_2$$

Đặt

$$A = (p_x c_1 + p_y s_1 - L_2)^2 + {p_z}^2 - {L_3}^2 - (L_5 + L_4)^2$$

$$A = 2L_3 s_2 (L_5 + L_4) c_{23} - 2(L_5 + L_4) s_{23} L_3 c_2 = 2L_3 (L_5 + L_4)(s_2 c_{23} - c_2 s_{23}) = -2L_3 (L_5 + L_4) s_3$$

$$=> s_3 = -\frac{A}{2L_3(L_5+L_4)} = \frac{(p_x c_1 + p_y s_1 - L_2)^2 + {p_z}^2 - {L_3}^2 - (L_5+L_4)^2}{2L_3(L_5+L_4)}$$

Ta có

$${s_3}^2 + {c_3}^2 = 1 => c_3 = \pm\sqrt{1-{s_3}^2}$$

$$\theta_3 = \left[arctg2(s_3, c_3); arctg(s_3, -c_3)\right]$$

Chọn nghiệm:

$$\theta_3 = arctg2(s_3, c_3) \tag{3.21}$$

$\rightarrow$ Tính θ_2

Từ góc θ_3 vừa tìm được ta tính được góc θ_2 dựa vào phương trình (*)

$$\begin{cases} p_x c_1 + p_y s_1 - L_2 = L_3 c_2 - (L_5+L_4)s_{23} \\ p_z = -L_3 s_2 - (L_5+L_4)c_{23} \end{cases} => \begin{cases} B = L_3 c_2 - (L_5+L_4)s_{23} \\ C = -L_3 s_2 - (L_5+L_4)c_{23} \end{cases} => \begin{cases} B = L_3 c_2 - (L_5+L_4)(c_2 s_3 + s_2 c_3) \\ C = -L_3 s_2 - (L_5+L_4)(c_2 c_3 + s_2 s_3) \end{cases}$$

$$=> \begin{cases} B = L_3 c_2 - (L_5+L_4)c_2 s_3 - (L_5+L_4)s_2 c_3 \\ C = -L_3 s_2 - (L_5+L_4)c_2 c_3 + (L_5+L_4)s_2 s_3 \end{cases} => \begin{cases} B = (L_3 - (L_5+L_4)s_3)c_2 - (L_5+L_4)s_2 c_3 \\ C = (-L_3 + (L_5+L_4)s_3)s_2 - (L_5+L_4)c_2 c_3 \end{cases}$$

$$=> \begin{cases} c_2 = \dfrac{B + (L_5+L_4)s_2 c_3}{L_3 - (L_5+L_4)s_3} \\ s_2 = -\dfrac{C + (L_5+L_4)c_2 c_3}{L_3 - (L_5+L_4)s_3} \end{cases} \tag{3.22}$$

Thay s_2 vào c_2 ta được:

$$c_2 = \frac{B + (L_5+L_4)s_2 c_3}{L_3 - (L_5+L_4)s_3} = \frac{B + (L_5+L_4)\left(-\dfrac{C + (L_5+L_4)c_2 c_3}{L_3 - (L_5+L_4)s_3}\right)c_3}{L_3 - (L_5+L_4)s_3}$$

$$= \frac{B(L_3 - (L_5+L_4)s_3) + (L_5+L_4)(-C - (L_5+L_4)c_2 c_3)c_3}{(L_3 - (L_5+L_4)s_3)^2}$$

$$= \frac{B(L_3 - (L_5+L_4)s_3) - C(L_5+L_4)c_3 - (L_5+L_4)^2 {c_3}^2 c_2}{(L_3 - (L_5+L_4)s_3)^2}$$

$$= \frac{B(L_3 - (L_5+L_4)s_3) - C(L_5+L_4)c_3}{(L_3 - (L_5+L_4)s_3)^2} - \frac{(L_5+L_4)^2 {c_3}^2}{(L_3 - (L_5+L_4)s_3)^2} c_2$$

$$=> c_2\left(1 + \frac{(L_5+L_4)^2 {c_3}^2}{(L_3 - (L_5+L_4)s_3)^2}\right) = \frac{B(L_3 - (L_5+L_4)s_3) - C(L_5+L_4)c_3}{(L_3 - (L_5+L_4)s_3)^2}$$

$$=> c_2\left((L_3 - (L_5+L_4)s_3)^2 + (L_5+L_4)^2 {c_3}^2\right) = B(L_3 - (L_5+L_4)s_3) - C(L_5+L_4)c_3$$

$$=> c_2 = \frac{(p_x c_1 + p_y s_1 - L_2)(L_3 - (L_5+L_4)s_3) - p_z(L_5+L_4)c_3}{(L_3 - (L_5+L_4)s_3)^2 + (L_5+L_4)^2 {c_3}^2}$$

Thay c_2 vào s_2 ta được:

$$s_2 = \frac{-C-(L_5+L_4)\left(\frac{B+(L_5+L_4)s_2c_3}{L_3-(L_5+L_4)s_3}\right)c_3}{L_3-(L_5+L_4)s_3}$$

$$= \frac{-C(L_3-(L_5+L_4)s_3)-(L_5+L_4)(B+(L_5+L_4)s_2c_3)c_3}{(L_3-(L_5+L_4)s_3)^2}$$

$$= \frac{-C(L_3-(L_5+L_4)s_3)-B(L_5+L_4)c_3-(L_5+L_4)^2 s_2 c_3^{\,2}}{(L_3-(L_5+L_4)s_3)^2}$$

$$= \frac{-C(L_3-(L_5+L_4)s_3)-B(L_5+L_4)c_3}{(L_3-(L_5+L_4)s_3)^2} - \frac{(L_5+L_4)^2 c_3^{\,2}}{(L_3-(L_5+L_4)s_3)^2}$$

$$\Rightarrow s_2\left(1+\frac{(L_5+L_4)^2 c_3^{\,2}}{(L_3-(L_5+L_4)s_3)^2}\right) = \frac{-C(L_3-(L_5+L_4)s_3)-B(L_5+L_4)c_3}{(L_3-(L_5+L_4)s_3)^2}$$

$$\Rightarrow s_2\left((L_3-(L_5+L_4)s_3)^2+(L_5+L_4)^2 c_3^{\,2}\right) = -C(L_3-(L_5+L_4)s_3)-B(L_5+L_4)c_3$$

$$\Rightarrow s_2 = \frac{-p_z(L_3-(L_5+L_4)s_3)-(p_x c_1+p_y s_1-L_2)-B(L_5+L_4)c_3}{L_3-(L_5+L_4)s_3)^2+(L_5+L_4)^2 c_3^{\,2}}$$

Ta tính được góc θ_2

$$\theta_2 = arctg2(s_2, c_2) \tag{3.23}$$

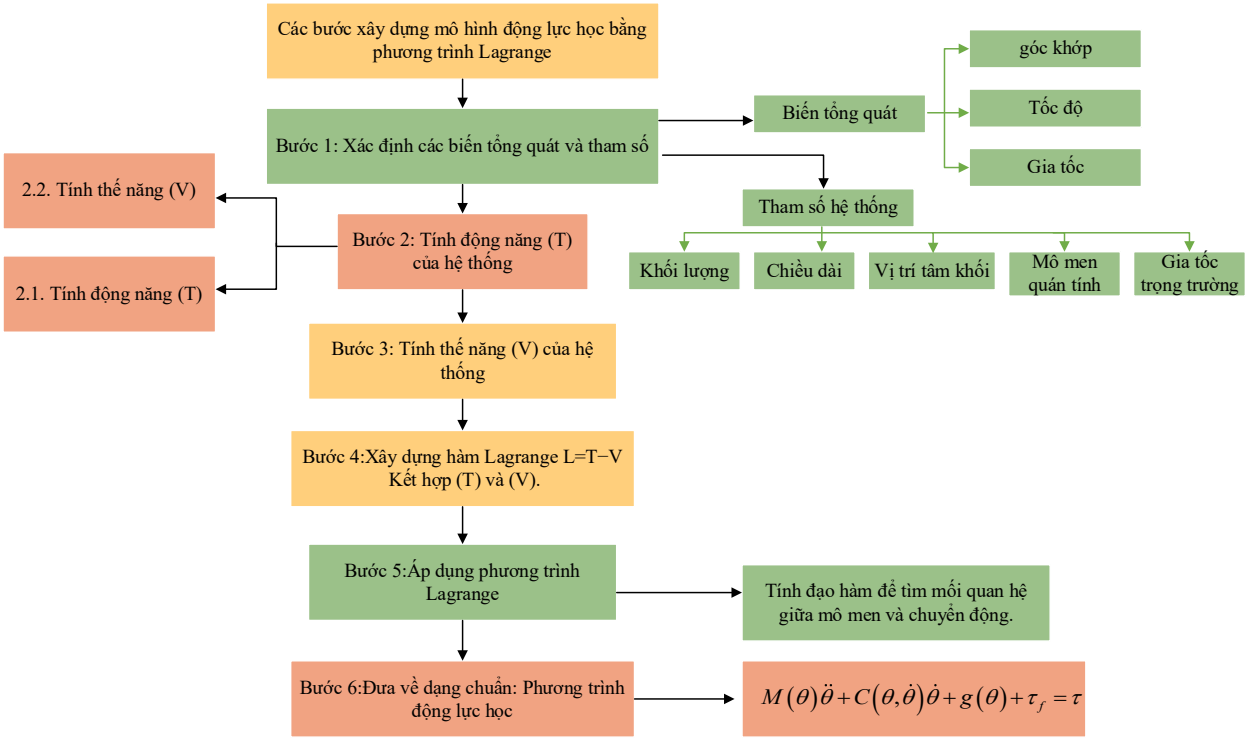
Tìm góc θ_4

Dựa vào góc rót bia tìm được θ_4

Kết luận

$$\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_4\end{bmatrix} = \begin{bmatrix}arctg2(p_y, p_x)\\ arctg2(s_3, c_3)\\ arctg2(s_2, c_2)\\ \theta_4\end{bmatrix} \tag{3.24}$$

3.1.3 Mô hình động lực học



Hình 3.6: Cấu trúc mục 3.1.3

a) Lý thuyết

Động lực học trong cánh tay máy là nghiên cứu về các lực liên kết với cánh tay máy. Để thực hiện một nhiệm vụ cụ thể, cánh tay máy được tăng tốc từ trạng thái nghỉ để bắt đầu chuyển động, sau đó khâu cuối của cánh tay máy có thể được yêu cầu di chuyển với vận tốc không đổi và sau đó được giảm tốc để đưa nói về trạng thái xác lập tại điểm mong muốn. Chuyển động như vậy đòi hỏi sự biến đổi của mô-men xoắn tại các khớp bằng cơ cầu truyền động theo quỹ đạo mong muốn.

Phương trình động lực học nghịch (Inverse Dynamics) là phương trình với đầu vào là tín hiệu đặt bao gồm vị trí, vận tốc, gia tốc của từng góc khớp; ngõ ra là tín hiệu mô-men điều khiển. Quỹ đạo với sự thay đổi của vị trí, vận tốc và gia tốc được đưa ra và mô-men xoắn cần thiết tại các khớp thao tác để di chuyển dọc theo quỹ đạo mong muốn.

Phương trình động lực học thuận (Forward Dynamics) là phương trình với đầu vào là tín hiệu mô-men điều khiển; ngõ ra bao gồm vị trí, vận tốc, gia tốc của từng góc khớp. Các tín hiệu điều khiển mô-men xoắn được đưa ra và chuyển động của cánh tay máy phải được tìm thấy, nó có thể liên quan đến việc tìm kiếm vị trí, vận tốc và gia tốc.

Có hai phương pháp để tìm phương trình động lực học của hệ cánh tay máy đó là phương trình Lagrangian dựa theo năng lượng động năng và thế năng và phương trình Newton-Euler dựa theo lực và mô-men tác động lên từng khâu của cánh tay máy. Ở đây

nhóm tiến hành phân tích động lực học của hệ theo phương pháp sử dụng phương trình Lagrangian.

Vận tốc góc và vận tốc dài của cánh tay máy.

Vận tốc góc của khâu $i+1$ so với hệ trục tọa độ $\{i+1\}$

$$^{i+1}\omega_{i+1} = {}^{i+1}_{i}R\,{}^{i}\omega_i + \dot{\theta}_{i+1}\,{}^{i+1}\hat{z}_{i+1} \tag{3.25}$$

Vận tốc góc của khâu $i+1$ so với hệ trục toa độ góc

$$^{i+1}\omega_{i+1} = {}^{i}_{i+1}R\,{}^{i+1}\omega_{i+1} \tag{3.26}$$

Vận tốc dài của khâu $i+1$ so với hệ trục tọa độ $\{i+1\}$

$$^{i+1}\upsilon_{i+1} = {}^{i+1}_{i}R\left({}^{i}\upsilon_i + {}^{i}\omega_i \times {}^{i}p_{i+1}\right) \tag{3.27}$$

Vận tốc dài của khâu $i+1$ so với hệ trục tọa độ gốc

$$^{i+1}\upsilon_{i+1} = {}^{i}_{i+1}R\,{}^{i+1}\upsilon_{i+1} \tag{3.28}$$

Vị trí trọng tâm tại khâu i+1 so với hệ trục tọa độ gốc

$$^{i}p_{C_{i+1}} = {}^{i}_{i+1}T\,{}^{i+1}p_{C_{i+1}} \tag{3.29}$$

Vận tốc dài tại trọng tâm khâu i+1 so với hệ trục tọa độ gốc

$$^{i+1}\upsilon_{C_{i+1}} = {}^{i}_{i+1}R\left({}^{i+1}\upsilon_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{i+1}p_{C_{i+1}}\right) \tag{3.30}$$

Bảng 3.5: Các đại lượng động lực học trong thuật toán Newton-Euler

$^{\mathbf{i+1}}_{\mathbf{i}}R$	ma trận xoay của hệ trục thứ {i} lên hệ trục thứ {i+1}
$^{i+1}\omega_{i+1}$	vận tốc góc của khâu thứ {i+1}
$^{i}\omega_i$	vận tốc góc của khâu thứ {i}
$\dot{\theta}_{i+1}$	vận tốc góc tại khớp thứ {i+1}
$^{i+1}\hat{z}_{i+1}$	vecto của trục Z tại hệ trục thứ {i+1}
$^{i+1}v_{i+1}$	vận tốc tuyến tính ở điểm gốc của hệ trục thứ {i+1}
$^{i}v_i$	vận tốc tuyến tính ở điểm gốc của hệ trục thứ {i}
$^{\mathbf{i}}P_{\mathbf{i+1}}$	vị trí điểm gốc của hệ trục thứ {i+1} chiếu lên hệ trục thứ {i}
$^{\mathbf{i+1}}P_{\mathbf{c_{i+1}}}$	vị trí trọng tâm của khớp {i+1} chiều lên tọa độ {i}.

Công thức tính toán động lực học cho cánh tay máy.

Tính toán tổng năng lượng động năng của hệ.

$$K(\theta,\dot{\theta}) = \sum_{i=1}^{n}\left(\frac{1}{2}m_i(\upsilon_{c_i})^T\upsilon_{c_i} + \frac{1}{2}({}^{i}\omega_i)^T\,{}^{c_i}I_i\,{}^{i}\omega_i\right) \tag{3.31}$$

Tính toán tổng năng lượng thế năng của hệ.

$$U(\theta) = \sum_{i=1}^{n}\left(-m_i\,{}^{0}g^T\,{}^{0}p_{c_i} + U_{ref_i}\right) \tag{3.32}$$

Từ kết quả tính toán động năng và thế năng của hệ thay vào phương trình chuyển động Larange của hệ.

$$L\left(\theta,\dot{\theta}\right)=K\left(\theta,\dot{\theta}\right)-U\left(\theta\right) \tag{3.33}$$

Bảng 3.6: Chú thích các đại lượng động lực học của robot

m_i	là khối lượng của khâu thứ {i}
υ_{C_i}	vận tốc dài tại trọng tâm khâu thứ {i}
${}^{i}\omega_i$	vận tốc góc tại khâu thứ{i}
${}^{C_i}I_i$	mo-men quán tính tại trọng tâm khâu thứ {i}
g	vec-tơ gia tốc trọng trường
${}^{0}p_{C_i}$	vị trí trọng tâm khâu thứ {i} so với hệ tọa độ gốc {0}
U_{ref}	gốc thế năng

Tính phương trình động lực học của hệ

$$\frac{d}{dt}\left(\frac{\partial L\left(\theta,\dot{\theta}\right)}{\partial\dot{\theta}}\right)-\left(\frac{\partial L\left(\theta,\dot{\theta}\right)}{\partial\theta}\right)=\tau \tag{3.34}$$

Phương trình vi phân của cánh tay máy đưa về dạng tổng quát

$$M\left(\theta\right)\ddot{\theta}+C\left(\theta,\dot{\theta}\right)\dot{\theta}+g\left(\theta\right)+\tau_f+J^TF_e=\tau \tag{3.35}$$

Bảng 3.7: Chú thích các đại lượng trong phương trình động lực học robot

$\tau\in\Re^{n\times1}$	vec-tơ vận tốc tại mỗi khớp quay
$\theta\in\Re^{n\times1}$	vec-tơ vị trí của mỗi khớp quay
$\dot{\theta}\in\Re^{n\times1}$	véc-tơ vận tốc của mỗi khớp quay
$M\left(\theta\right)\in\Re^{n\times n}$	ma trận quán tính có tính chất đối xứng và xác định dương
$C(\theta,\dot{\theta})\in\Re^{n\times n}$	ma trận Coriolis/centrifugal
$g\left(\theta\right)\in\Re^{n\times1}$	vec-tơ trọng trường
$\tau_{\mathbf{f}}=F_v\dot{\theta}+F_csign\left(\dot{\theta}\right)$	với $F_v=diag([b_1,b_2,...b_n])$ là hệ số ma sát nhớt
$F_{\mathbf{c}}=diag\left(\left[c_1,c_2,...,c_n\right]\right)\in\Re^{n\times n}$	là hệ số ma sát tĩnh
J	ma trận Jacobia
F_e	ngoại lực tác động lên robot tại khâu cuối

Phương trình động lực học thuận của hệ như sau:

$$\ddot{\theta}=M\left(\theta\right)^{-1}\left(\tau-C\left(\theta,\dot{\theta}\right)\dot{\theta}-g\left(\theta\right)-\tau_f-J^TF_e\right) \tag{3.36}$$

Phương trình động lực học nghịch của hệ như sau:

$$\tau = M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) + \tau_f + J^T F_e \tag{3.37}$$

b) Tính toán động lực học cánh tay máy

❖ **Điều kiện ban đầu**

Vận tốc góc của hệ tọa độ gốc như sau:

$$^0\omega_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{3.38}$$

Vận tốc dài tham chiếu cho hệ tọa độ gốc:

$$^0\upsilon_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{3.39}$$

Gia tốc trọng trường của hệ tọa độ gốc theo phương Z và hướng xuống:

$$g = [\,0 \quad 0 \quad g\,] \tag{3.40}$$

Tọa độ trọng tâm tại mỗi khâu:

$$\begin{aligned} ^1p_{c_1} &= \begin{bmatrix} x_1 & y_1 & z_1 \end{bmatrix}^T \\ ^2p_{c_2} &= \begin{bmatrix} x_2 & y_2 & z_2 \end{bmatrix}^T \\ ^3p_{c_3} &= \begin{bmatrix} x_3 & y_3 & z_3 \end{bmatrix}^T \\ ^4p_{c_4} &= \begin{bmatrix} x_4 & y_4 & z_4 \end{bmatrix}^T \end{aligned} \tag{3.41}$$

Moment quán tính tại mỗi khâu:

$$^{c_1}I_1 = \begin{bmatrix} I_{xx_1} & -I_{xy_1} & -I_{xz_1} \\ -I_{xy_1} & I_{yy_1} & -I_{yz_1} \\ -I_{xz_1} & -I_{yz_1} & I_{zz_1} \end{bmatrix} \tag{3.42}$$

$$^{c_2}I_2 = \begin{bmatrix} I_{xx_2} & -I_{xy_2} & -I_{xz_2} \\ -I_{xy_2} & I_{yy_2} & -I_{yz_2} \\ -I_{xz_2} & -I_{yz_2} & I_{zz_2} \end{bmatrix} \tag{3.43}$$

$$^{c_3}I_3 = \begin{bmatrix} I_{xx_3} & -I_{xy_3} & -I_{xz_3} \\ -I_{xy_3} & I_{yy_3} & -I_{yz_3} \\ -I_{xz_3} & -I_{yz_3} & I_{zz_3} \end{bmatrix} \tag{3.44}$$

$$^{c_3}I_3 = \begin{bmatrix} I_{xx_3} & -I_{xy_3} & -I_{xz_3} \\ -I_{xy_3} & I_{yy_3} & -I_{yz_3} \\ -I_{xz_3} & -I_{yz_3} & I_{zz_3} \end{bmatrix} \tag{3.45}$$

Bảng 3.8: Chú thích

I_{xx_i}	moment quán tính tác động lên trục X của hệ trục tọa độ thứ {i}
I_{yy_i}	moment quán tính tác động lên trục Y của hệ trục tọa độ thứ {i}
I_{zz_i}	moment quán tính tác động lên trục Z của hệ trục tọa độ thứ {i}
I_{xy_i}	moment quán tính tác động lên trục XY của hệ trục tọa độ thứ {i}
I_{xz_i}	moment quán tính tác động lên trục XZ của hệ trục tọa độ thứ {i}
I_{yz_i}	moment quán tính tác động lên trục YZ của hệ trục tọa độ thứ {i}

✓ **Tại khâu 1**

Vận tốc góc của khâu 1 so với hệ trục tọa độ {1}

$$^1\omega_1 = {}^1_0R\,{}^0\omega_0 + \dot{\theta}_1\,{}^1\hat{z}_1 \tag{3.46}$$

Vận tốc góc của khâu 1 so với hệ trục toa độ góc

$$^1\omega_1 = {}^0_1R\,{}^1\omega_1 \tag{3.47}$$

Vận tốc dài của khâu 1 so với hệ trục tọa độ {1}

$$^1\upsilon_1 = {}^1_0R\left({}^0\upsilon_0 + {}^0\,\omega_0 \times {}^0p_1\right) \tag{3.48}$$

Vận tốc dài của khâu 1 so với hệ trục tọa độ gốc

$$^1\upsilon_1 = {}^0_1R\,{}^1\upsilon_1 \tag{3.49}$$

Vị trí trọng tâm tại khâu 1 so với hệ trục tọa độ gốc

$$^0p_{C_1} = {}^0_1T\,{}^1p_{C_1} \tag{3.50}$$

Vận tốc dài tại trọng tâm khâu 1 so với hệ trục tọa độ gốc

$$^1\upsilon_{C_1} = {}^0_1R\left({}^1\upsilon_1 + {}^1\,\omega_1 \times {}^1p_{C_1}\right) \tag{3.51}$$

✓ **Tại khâu 2:**

Vận tốc góc của khâu 2 so với hệ trục tọa độ {2}

$$^2\omega_2 = {}^2_1R\,{}^1\omega_1 + \dot{\theta}_2\,{}^2\hat{z}_2 \tag{3.52}$$

Vận tốc góc của khâu 2 so với hệ trục toa độ gốc

$$^2\omega_2 = {}^1_2R\,{}^2\omega_2 \tag{3.53}$$

Vận tốc dài của khâu 2 so với hệ trục tọa độ {2}

$$^2\upsilon_2 = {}^2_1R\left({}^1\upsilon_1 + {}^1\,\omega_1 \times {}^1p_2\right) \tag{3.54}$$

Vận tốc dài của khâu 2 so với hệ trục tọa độ gốc

$$^2\upsilon_2 = {}^1_2R\,{}^2\upsilon_2 \tag{3.55}$$

Vị trí trọng tâm tại khâu 2 so với hệ trục tọa độ gốc

$$^{0}p_{C_2} = {}^{0}_{2}T\,{}^{2}p_{C_2} \tag{3.56}$$

Vận tốc dài tại trọng tâm khâu 2 so với hệ trục tọa độ gốc

$$^{2}\upsilon_{C_2} = {}^{0}_{2}R\left({}^{2}\upsilon_2 + {}^{2}\omega_2 \times {}^{2}p_{C_2}\right) \tag{3.57}$$

✓ **Tại khâu 3:**

Vận tốc góc của khâu 3 so với hệ trục tọa độ {3}

$$^{3}\omega_3 = {}^{3}_{2}R\,{}^{2}\omega_2 + \dot{\theta}_3\,{}^{3}\hat{z}_3 \tag{3.58}$$

Vận tốc góc của khâu 3 so với hệ trục toa độ góc

$$^{3}\omega_3 = {}^{2}_{3}R\,{}^{3}\omega_3 \tag{3.59}$$

Vận tốc dài của khâu 3 so với hệ trục tọa độ {3}

$$^{3}\upsilon_3 = {}^{3}_{2}R\left({}^{2}\upsilon_2 + {}^{2}\omega_2 \times {}^{2}p_3\right) \tag{3.60}$$

Vận tốc dài của khâu 3 so với hệ trục tọa độ gốc

$$^{3}\upsilon_3 = {}^{2}_{3}R\,{}^{3}\upsilon_3 \tag{3.61}$$

Vị trí trọng tâm tại khâu 3 so với hệ trục tọa độ gốc

$$^{0}p_{C_3} = {}^{0}_{3}T\,{}^{3}p_{C_3} \tag{3.62}$$

Vận tốc dài tại trọng tâm khâu 3 so với hệ trục tọa độ gốc

$$^{3}\upsilon_{C_3} = {}^{0}_{3}R\left({}^{3}\upsilon_3 + {}^{3}\omega_3 \times {}^{3}p_{C_3}\right) \tag{3.63}$$

✓ **Tại khâu 4:**

Vận tốc góc của khâu 4 so với hệ trục tọa độ {4}

$$^{4}\omega_4 = {}^{4}_{3}R\,{}^{3}\omega_3 + \dot{\theta}_4\,{}^{4}\hat{z}_4 \tag{3.64}$$

Vận tốc góc của khâu 4 so với hệ trục toa độ góc

$$^{4}\omega_4 = {}^{3}_{4}R\,{}^{4}\omega_4 \tag{3.65}$$

Vận tốc dài của khâu 4 so với hệ trục tọa độ {4}

$$^{4}\upsilon_4 = {}^{4}_{3}R\left({}^{3}\upsilon_3 + {}^{3}\omega_3 \times {}^{3}p_4\right) \tag{3.66}$$

Vận tốc dài của khâu 4 so với hệ trục tọa độ gốc

$$^{4}\upsilon_4 = {}^{3}_{4}R\,{}^{4}\upsilon_4 \tag{3.67}$$

Vị trí trọng tâm tại khâu 4 so với hệ trục tọa độ gốc

$$^{0}p_{C_4} = {}^{0}_{4}T\,{}^{4}p_{C_4} \tag{3.68}$$

Vận tốc dài tại trọng tâm khâu 4 so với hệ trục tọa độ gốc

$$^{4}\upsilon_{C_4} = {}^{0}_{4}R\left({}^{4}\upsilon_4 + {}^{4}\omega_4 \times {}^{4}p_{C_4}\right) \tag{3.69}$$

❖ **Tính toán động lực học robot.**

Tính toán tổng năng lượng động năng dựa vào công thức với n = 4.

$$K(\theta,\dot{\theta}) = \sum_{i=1}^{n}\left(\frac{1}{2}m_i {\upsilon_{c_i}}^T \upsilon_{c_i} + \frac{1}{2}{}^i\omega_i^T {}^{c_i}I_i^i\omega_i\right) \quad (3.70)$$

Tính toán tổng năng lượng thế năng dựa vào công thức n=4.

$$U(\theta) = \sum_{i=1}^{n}\left(-m_i({}^0g)^T {}^0p_{c_i} + U_{ref_i}\right) \quad (3.71)$$

Từ kết quả tính toán động năng và thế năng của hệ thay vào phương trình chuyển động Larange.

$$L(\theta,\dot{\theta}) = K(\theta,\dot{\theta}) - U(\theta) \quad (3.72)$$

Tính phương trình động lực học của hệ.

$$\frac{d}{dt}\left(\frac{\partial L(\theta,\dot{\theta})}{\partial\dot{\theta}}\right) - \left(\frac{\partial L(\theta,\dot{\theta})}{\partial\theta}\right) = \tau \quad (3.73)$$

Phương trình vi phân của cánh tay máy đưa về dạng tổng quát

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) + \tau_f = \tau \quad (3.74)$$

Phương trình động lực học thuận của hệ như sau:

$$\ddot{\theta} = M(\theta)^{-1}(\tau - C(\theta,\dot{\theta})\dot{\theta} - g(\theta) - F_v\dot{\theta} - F_c sign(\dot{\theta})) \quad (3.75)$$

Phương trình động lực học nghịch của hệ như sau:

$$\tau = M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) + F_v\dot{\theta} + F_c sign(\dot{\theta}) \quad (3.76)$$

3.2. Thiết kế bộ điều khiển PD

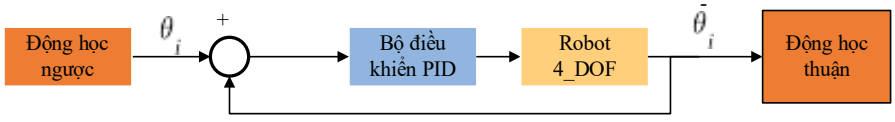
3.2.1 Bộ điều khiển PID

Bộ điều khiển PID là một phương pháp điều khiển phổ biến trong công nghiệp và robot học, được sử dụng để điều chỉnh đầu ra của hệ thống (góc khớp của robot 4-DOF rót bia) nhằm đạt giá trị mong muốn với sai số nhỏ nhất, được trình bày công thức

$$u_{PID} = K_p e(t) + K_D\frac{de(t)}{dt} + K_I\int e(t)dt \quad (3.77)$$

Trong đó, u là tín hiệu điều khiển và e là sai lệch điều khiển. Tín hiệu điều khiển là tổng của 3 thành phần: tỉ lệ, vi phân và tích phân.

Nhiệm vụ của người thiết kế bộ điều khiển PID được xác định bởi công thức trên, là chọn lựa bộ ba giá trị $\{K_P, K_I, K_D\}$ thỏa mãn các yêu cầu về chất lượng điều khiển được trình bày trong sơ đồ điều khiển sau



Hình 3.7: Sơ đồ điều khiển

Trong sơ đồ điều khiển có khối động học ngược để chuyển đổi từ vị trí ban đầu sang các góc tham chiếu ngõ ra của robot 4dof có khối động học thuận để chuyển đổi các góc thực sang vị trí thực tại tâm chuyển động

Thiết kế bộ điều khiển vi phân tỉ lệ cho hệ thống

Đặt $x_1 = \theta \in \Re^{4\times1}$; $x_2 = \dot{\theta} \in \Re^{4\times1}$ và x_2 = động lực học cánh tay máy được thể hiện theo phương trình không gian trạng thái như sau.

$$\begin{cases} x_1 = x_2 \\ \dot{x}_2 = M^{-1}(x_1)(u - C(x_1, x_2)x_2 - g(x_1) - F_v x_2 - F_c sign(x_2)) \end{cases} \quad (3.78)$$

Sai số vị trí:

$$e = x_{1d} - x_1 \quad (3.79)$$

3.2.2 Bộ điều khiển PD

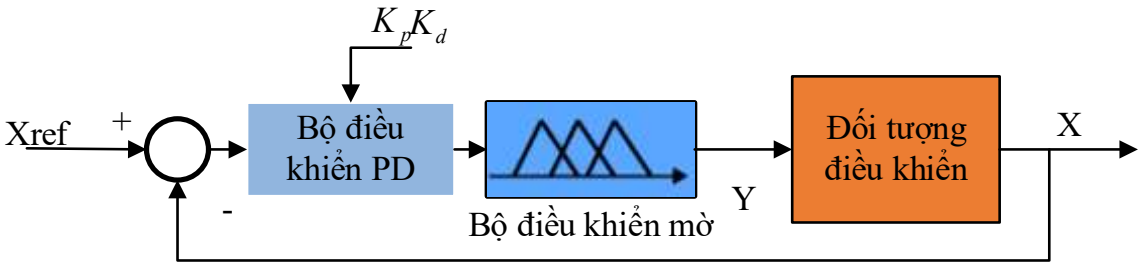
Trong công nghiệp, hiện tượng bão hòa tích phân trong các bộ điều khiển PID làm cho hệ kín có độ quá điều chỉnh lớn, thời gian điều chỉnh tăng lên và có thể gây ra mất ổn định để tránh việc sai sót trong quá trình điều khiển bỏ qua khâu tích phân, xét bộ điều khiển vi phân tỉ lệ được đưa ra như sau.

$$\tau = K_p e + K_d \dot{e} \quad (3.80)$$

Bảng 3.9: Chú thích

$K_p = diag\left(\left[K_{p1}, K_{p2}, K_{p3}, K_{p4}\right]\right) \in \Re^{4\times4}$	ma trận đường chéo xác định dương là hệ số khâu tỉ lệ
$K_d = diag\left(\left[K_{d1}, K_{d2}, K_{d3}, K_{d4}\right]\right) \in \Re^{4\times4}$	là ma trận đường chéo xác định dương là hệ số khâu vi phân.

3.2.3:Bộ điều khiển tự điều chỉnh FUZZY – PD cho mô hình cánh tay robot rot bia



Hình 3.8: Bộ điều khiển tự điều chỉnh FUZZY – PID

Điều khiển PD tự điều chỉnh mờ dựa trên điều khiển PID kinh điển và sử dụng các quy tắc suy luận mờ để làm cho các tham số PID tự chỉnh định dựa trên sai lệch $e(t)$ và đạo hàm $de(t)$.

- Bộ điều khiển Fuzzy lấy sai số góc và đạo hàm sai số góc để điều chỉnh hệ số góc ra. Từ đó, các hệ số góc ra thay đổi theo thời gian được đưa vào khối khuếch đại tin hiệu để đưa ra giá trị điều khiển.

Việc xây dựng các hàm liên thuộc, các khoảng giá trị của biến vật lý và biến ngôn ngữ dựa trên kinh nghiệm chỉnh định như sau:

Bảng 3.10: Quy ước luật điều khiển

Âm nhiều	Âm vừa	Âm ít	Zero	Dương ít	Dương vừa	Dương nhiều
NB	NM	NS	ZO	PS	PM	PB

Luật điều khiển Mờ PD (Fuzzy - PD): Mục tiêu làm hệ phản hồi nhanh hơn khi sai số lớn và đáp ứng trong thời gian sớ m nhất.

+ E là sai số.
+ dE là đạo hàm sai số

Giá trị: Đầu ra u là giá trị điều khiển góc.

Bảng 3.11: Luật điều khiển

E \ dE	**NB**	**NM**	**NS**	**Z**	**PS**	**PM**	**PB**
NB	PB	PB	PM	PM	PS	Z	Z
NM	PB	PM	PM	PS	Z	Z	NS
NS	PM	PM	PS	Z	NS	NM	NM
Z	PM	PS	Z	Z	Z	NS	NM
PS	PS	Z	NS	Z	NS	NM	NM
PM	Z	NS	NM	NM	NM	NM	NB
PB	Z	NM	NM	NM	NB	NB	NB

Giải thích luật:

Mục tiêu:

+ Khi sai số lớn (NB hoặc PB), hệ cần phản hồi nhanh, nên giá trị đầu ra u được tăng mạnh (ví dụ: PB khi E = NB và dE = NB).
+ Khi sai số gần 0 (ZO), giá trị góc được giữ nguyên (ZO) để tránh giao động.
+ Khi sai số lớn, giá trị góc ra tăng để đáp ứng nhanh hơn, nhưng cần mượt mà để giảm dao động.

Phân tích một số quy tắc:

+ E = NB, dE = NB → u = PB: Khi sai số âm lớn và tăng nhanh (đạo hàm âm lớn), hệ cần điều chỉnh mạnh về dương để bù đắp nhanh.
+ E = ZO, dE = ZO → u = ZO: Khi sai số và tốc độ thay đổi gần 0, giữ nguyên góc để hệ ổn định.

+ E = PS, dE = NS → u = NS: Khi sai số dương nhỏ và giảm (đạo hàm âm), điều chỉnh giảm nhẹ để tránh vượt quá.
+ E = PB, dE = PB → u = NB: Khi sai số dương lớn và tăng nhanh, giảm mạnh để kiềm chế.

Kết quả:

Phản hồi nhanh khi sai số lớn: Các quy tắc ở góc NB và PB (ví dụ: PB, PM) đảm bảo điều chỉnh mạnh, phù hợp với mục tiêu.

Giữ nguyên góc khi sai số gần 0: Quy tắc E = ZO, dE = ZO → u = ZO giúp ổn định hệ.

3.3. Ứng dụng AI trong nhận diện ly và ước lượng lượng bia

3.3.1. Mô hình YOLO (YOU ONLY LOOK ONCE)

Mô hình YOLO, do Joseph Redmon phát triển lần đầu tiên vào năm 2016, là một kiến trúc mạng nơ-ron tích chập (CNN) mang tính đột phá trong lĩnh vực phát hiện và phân loại đối tượng. Điểm đặc biệt của YOLO nằm ở khả năng xử lý toàn bộ hình ảnh chỉ trong một lần truyền qua mạng (one-stage detector), cho phép mô hình vừa xác định nhãn đối tượng vừa dự đoán chính xác vị trí thông qua các khung giới hạn (bounding box). Mặc dù so với một số mô hình hai giai đoạn khác, YOLO đôi khi có thể thua kém một chút về độ chính xác tuyệt đối, nhưng bù lại nó vượt trội về tốc độ xử lý và khả năng nhận diện theo thời gian thực.

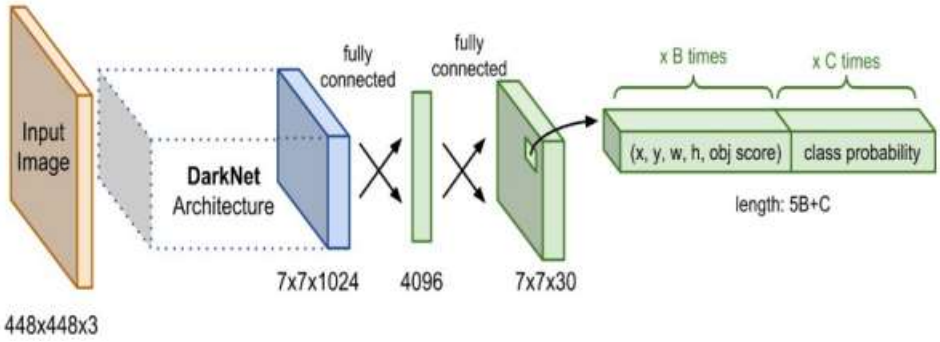
Trải qua nhiều thế hệ cải tiến, YOLO ngày càng được hoàn thiện cả về kiến trúc lẫn hiệu năng. Trong khuôn khổ đề tài, nhóm lựa chọn sử dụng phiên bản YOLOv8 — phiên bản mới nhất với nhiều cải tiến đáng kể về độ chính xác, khả năng tối ưu tham số cũng như tốc độ huấn luyện và nhận diện. Việc áp dụng YOLOv8 nhằm tận dụng những ưu thế của mô hình trong việc nhận dạng và xác định chính xác vị trí của các đối tượng, qua đó nâng cao hiệu quả toàn diện cho hệ thống xử lý hình ảnh và điều khiển robot.

3.3.2. Kiến trúc mô hình YOLO

YOLO (viết tắt của You Only Look Once) là một trong những mô hình mạng nơ-ron sâu nổi bật trong lĩnh vực phát hiện đối tượng (object detection). Khác với các phương pháp hai giai đoạn như R-CNN hay Faster R-CNN, YOLO xử lý bài toán phát hiện đối tượng chỉ trong một lần truyền mạng duy nhất, giúp đạt được tốc độ xử lý rất cao, phù hợp cho ứng dụng thời gian thực.

- Base Network : Là chuỗi các lớp convolutional (với batch normalization và activation Leaky ReLU) kèm theo một số fully-connected layer, chịu trách nhiệm trích xuất đặc trưng từ ảnh đầu vào.

- Extra Layers (Detection Head) : Là các lớp bổ trợ phía sau mạng cơ sở, đảm nhiệm vai trò dự đoán vị trí, kích thước và lớp đối tượng trên các đặc trưng đã được trích xuất.



Hình 3.9: Sơ đồ cấu trúc mạng YOLO

YOLOv3 sử dụng Darknet-53 làm mạng nền với 53 lớp tích chập liên tiếp, mỗi lớp kèm Batch Normalization và hàm kích hoạt Leaky ReLU, đồng thời áp dụng các lớp tích chập stride = 2 để giảm kích thước feature map mà vẫn bảo toàn thông tin đặc trưng. Đầu vào ảnh phổ biến trong YOLOv3 là 416x416 hoặc 608x608 . Trong quá trình xử lý, kích thước ảnh bị giảm dần theo cấp số nhân, dẫn đến việc thu được các feature map ở nhiều mức độ khác nhau:

- Với đầu vào 416x416 : Feature map lần lượt là 13x13 , 26x26 , 52x52 .
- Với đầu vào 608x608 : Feature map lần lượt là 19x19 , 38x38 , 76x76 .

Các feature map này giúp mô hình phát hiện đối tượng ở nhiều tỉ lệ khác nhau, cải thiện đáng kể khả năng nhận diện vật thể nhỏ và ở xa.

3.3.3. OUTPUT của YOLO

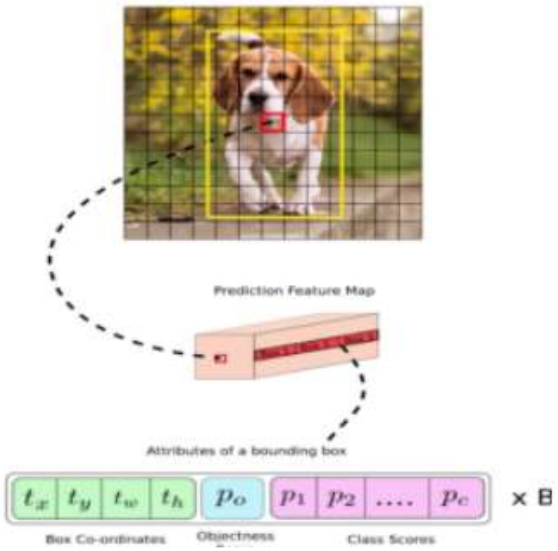
Output của mô hình YOLO là một véc tơ sẽ bao gồm các thành phần:

$$y^T = \left[p_{0,} \left\langle t_x, t_y, t_w, t_h \right\rangle, \left\langle p_1, p_2, ..., p_c \right\rangle \right] (3.81)$$

Trong đó:

- $\theta_1 = a\tan 2\left(\frac{z_{J_1}}{y_{F_1} - y_{J_1}} \right)$ (3.82) là xác suất dự báo vật thể xuất hiện trong bounding box.
- $\left\langle t_x, t_y, t_w, t_h \right\rangle$ giúp xác định bounding box. Trong đó t_x, t_y là tọa độ tâm và t_w, t_h là kích thước rộng, dài của bounding box.
- $\left\langle p_1, p_2, ..., p_c \right\rangle$ là véc tơ phân phối xác suất dự báo của các classes.

Việc hiểu output khá là quan trọng để chúng ta cấu hình tham số chuẩn xác khi huấn luyện model qua các open source như darknet. Như vậy output sẽ được xác định theo số lượng classes theo công thức (n_class+5). Nếu huấn luyện 80 classes thì bạn sẽ có output là 85. Trường hợp bạn áp dụng 3 anchors/cell thì số lượng tham số output sẽ là:



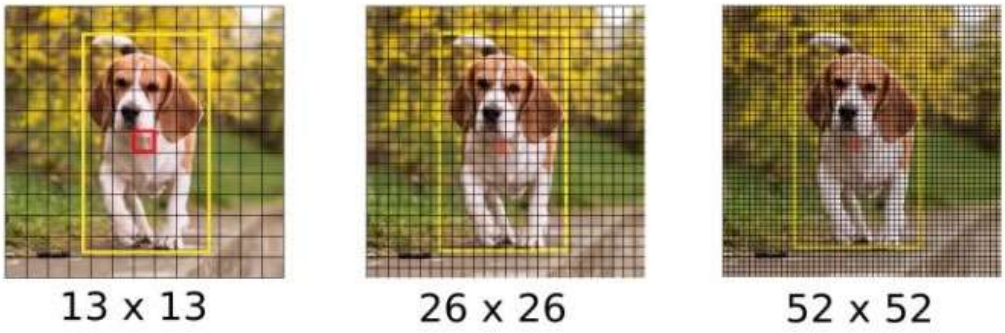
Hình 3.10: Kiến trúc một output của model

3.3.4 Feature map

Tương tự như SSD, YOLO cũng dự đoán vật thể trên nhiều feature map. Trên mỗi cell của feature map sẽ có 3 anchor box để dự đoán vật thể, như vậy ta tính được số lượng các anchor box trong mô hình YOLO là (3 feature map x 3 anchor box) = 9. Từ đó ta có thể tính được số lượng anchor box trên một bức ảnh là:

(13x13+26x26+52x52) x 3=10647 (anchor box)

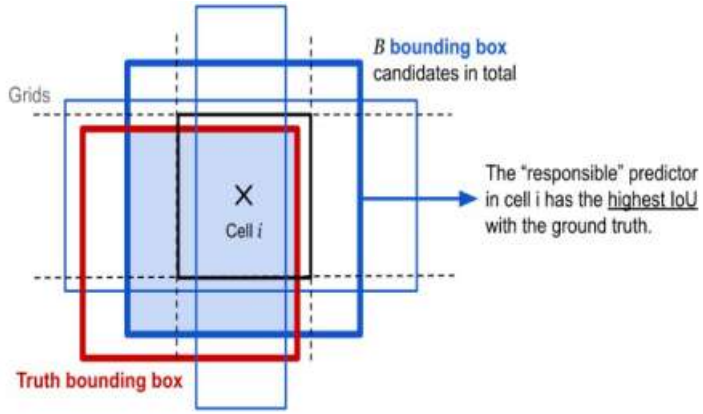
Với số lượng lớn thế này thì quá trình huấn luyện sẽ rất chậm bởi ta phải dự báo nhãn và bounding box trên từng anchor box.



Hình 3.11: Các feature maps của mạng YOLOv3 với input shape là 416x416, output là 3 feature maps

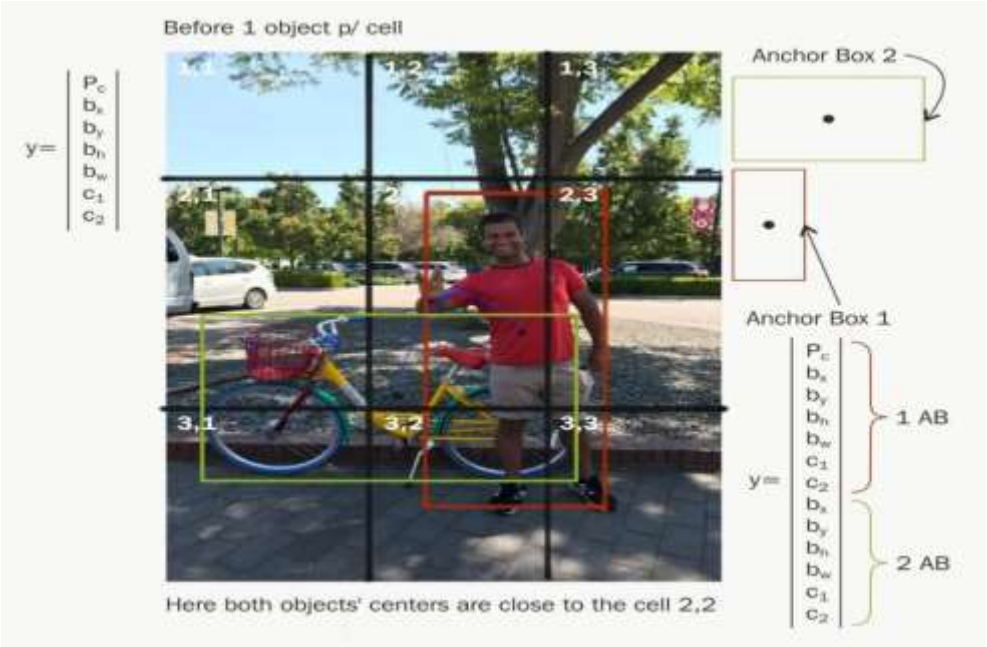
3.3.5. Anchor Box

YOLO sẽ cần các anchor box để ước lượng và tìm bounding box cho vật thể. Ban đầu các anchor box này sẽ xác định và bao quanh vật thể, sau đó thuật toán regression bounding box sẽ tìm ra bounding box dự đoán cho vật thể. Mỗi vật thể sẽ được phân bổ về một anchor box. Trong trường hợp có nhiều anchor box cùng bao quanh vật thể thì thuật toán sẽ chọn ra anchor box có IOU với ground truth cao nhất.



Hình 3.12: Cách xác định anchor box cho vật thể

Mô tả: Từ cell i ta xác định được 3 anchor box viền xanh. Tuy nhiên chỉ anchor box viền xanh đậm là được chọn vì có IOU với ground truth bounding box là cao nhất. Bên cạnh đó, mỗi vật thể còn được phân bố về một cell trên feature map chứa điểm mid point của vật thể. Từ cell đó ta có thể xác định các anchor box của đối tượng. Tóm lại, để xác định vật thể ta cần quan tâm đến anchor box và cell.



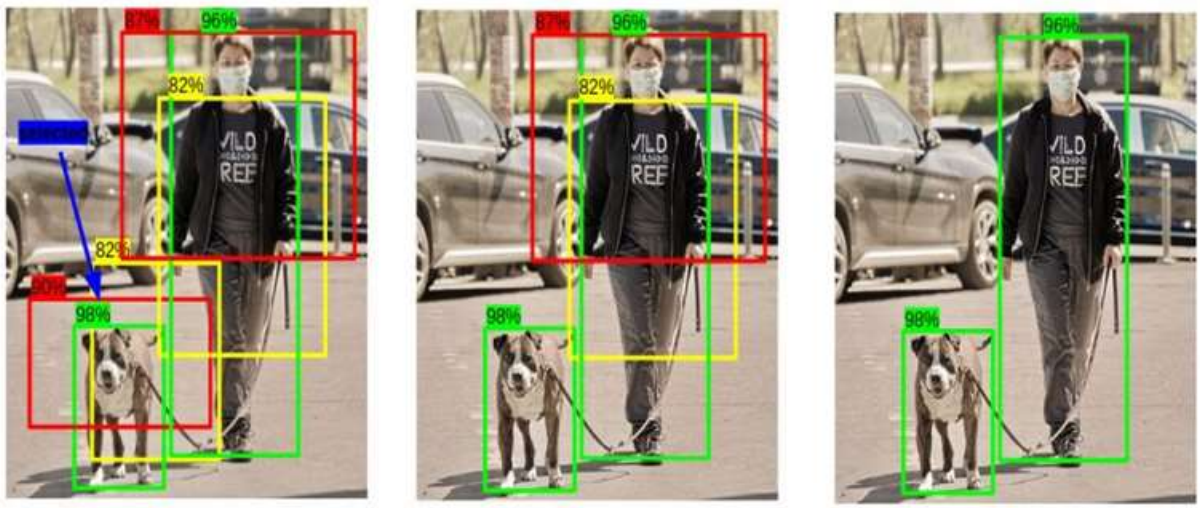
Hình 3.13: Cách xác định vật thể của YOLO

3.3.6. Non-max suppression

Non Maximum Suppression (NMS) là một bước hậu xử lý mà hầu hết các thuật toán Object Detection sau này đều sử dụng. Mục tiêu của NMS là lựa chọn một bounding box thích hợp nhất cho đối tượng.

NMS tính theo 2 tiêu chí: Objectiveness score được trả về bởi model -Overlap hoặc IOU giữa các bounding box.

Do thuật toán YOLO dự báo ra rất nhiều bounding box trên một bức ảnh nên đối với những cell có vị trí gần nhau, khả năng các khung hình bị overlap là rất cao. Trong trường hợp đó YOLO sẽ cần đến non-max suppression để giảm bớt số lượng các khung hình được sinh ra

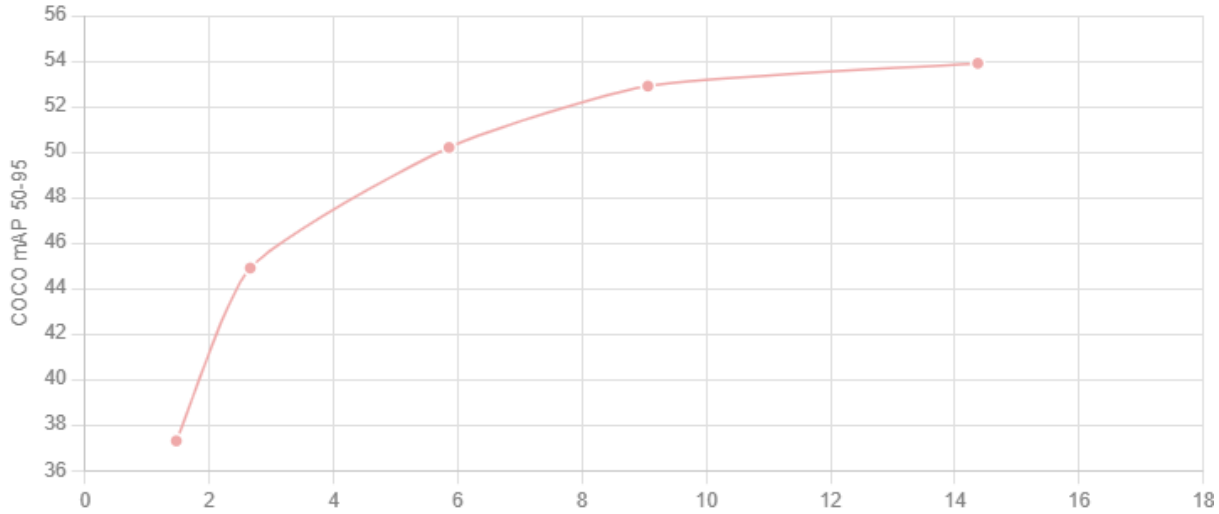


Hình 3.14: Non-max suppression từ 3 bounding box xuống còn 1 bounding box

Nếu chỉ số này lớn hơn ngưỡng threshold thì điều đó chứng tỏ 2 bounding boxes đang overlap nhau rất cao. Ta sẽ xóa các bouding có xác suất thấp hơn và giữ lại bounding có xác suất cao nhất. Cuối cùng, ta thu được một bounding box duy nhất cho một vật thể.

3.3.7. Phân loại YOLOv8

Có 5 mô hình khác nhau: YOLO8n, YOLO8s, YOLO8m, YOLO8l, YOLO8x. Đầu tiên là nhỏ nhất và kém chính xác nhất, cuối cùng là lớn nhất với độ chính xác lớn nhất. Tất cả các mô hình đều chạy trên PyTorch.



Hình 3.15: So sánh hiệu suất 5 mô hình YOLOv8

Phiên bản YOLOv8 đánh dấu những bước phát triển vượt bậc về cả tốc độ suy luận lẫn độ chính xác khi so sánh với các thế hệ trước. Trong các bài kiểm tra benchmark,

YOLOv8 được đối chiếu với các phiên bản từ YOLOv5 đến YOLOv7, cho thấy sự cải thiện rõ rệt. Kết quả minh họa trên Hình 2.20 cho thấy YOLOv8 liên tục vượt lên dẫn đầu, đạt độ chính xác mAP trên tập COCO cao hơn, đồng thời vẫn giữ được tốc độ xử lý nhanh hơn so với các phiên bản tiền nhiệm.

Cụ thể, các phiên bản khác nhau của YOLOv8 (gồm yolov8n, yolov8s, yolov8m và yolov8x) tạo thành một dải hiệu năng rõ ràng, trong đó mỗi biến thể đạt mAP50–95 cao hơn ở mức độ trễ tương ứng. Đáng chú ý, phiên bản yolov8x đạt được khoảng 53-54% mAP50–95 với độ trễ rất thấp, vượt trội hoàn toàn so với các thế hệ trước.

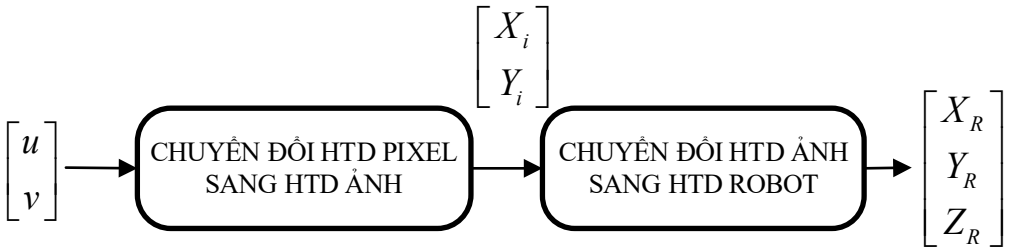
Đối với các mô hình tầm trung như yolov8m, hiệu năng cũng được cải thiện đáng kể khi cho độ chính xác tương đương với những mô hình kích thước lớn hơn của các phiên bản cũ, nhưng với thời gian xử lý ngắn hơn nhiều. Đặc biệt, ở dải độ trễ rất thấp (từ 2–6ms), yolov8s vẫn duy trì được độ chính xác khoảng 46-47% mAP50–95, mức mà trước đây chỉ có thể đạt được bởi những mô hình kém chính xác hơn. Đây là yếu tố quan trọng với những ứng dụng thời gian thực đòi hỏi sự cân bằng giữa tốc độ và độ chính xác.

Bên cạnh đó, đường cong hiệu năng của YOLOv8 cũng cho thấy khả năng mở rộng linh hoạt giữa các biến thể, giúp người dùng có thể dễ dàng lựa chọn cấu hình mô hình phù hợp với từng yêu cầu cụ thể về hiệu suất và thời gian xử lý.

3.3.8. Phương pháp chuyển đổi hệ toạ độ xác định vị trí đối tượng

Mục đích của việc chuyển đổi hệ tọa độ là lấy được vị trí của vật thể từ tọa độ camera đem chuyển sang tọa độ Robot để điều khiển Robot di chuyển đến tiếp cận đối tượng và xử lý.

Có rất nhiều phương pháp chuyển đổi hệ tọa độ camera sang hệ tọa độ Robot. Nhóm đã chọn **phương pháp chuyển đổi dựa theo mô hình thực tế**, đây là phương pháp chuyển đổi dựa trên vị trí các thiết bị trong thực tế chứ không dựa vào các thông số có sẵn của camera.



Hình 3.16: Phương pháp chuyển đổi hệ toạ độ xác định vị trí đối tượng

Ưu điểm của phương pháp:

- Phương trình chuyển đổi đơn giản, có thể dễ dàng chỉnh sửa khi muốn thay đổi vị trí camera và Robot.

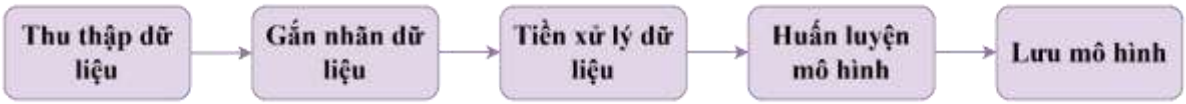
- Không cần người lập trình phải có kiến thức về xử lý các ma trận thông số của camera.
- Vì không phải dựa vào thông số của camera nên độ chính xác tương đối cao.

Nhược điểm của phương pháp:

- Vì là phương pháp chuyển đổi dựa theo mô hình thực tế nên không thể áp dụng phương pháp cho các mô hình khác. Đây là phương pháp riêng biệt.
- Khi muốn sử dụng phương pháp cho mô hình khác phải tiến hành lấy thông số và tính toán lại từ đầu.

3.3.9. Thiết kế và triển khai hệ thống

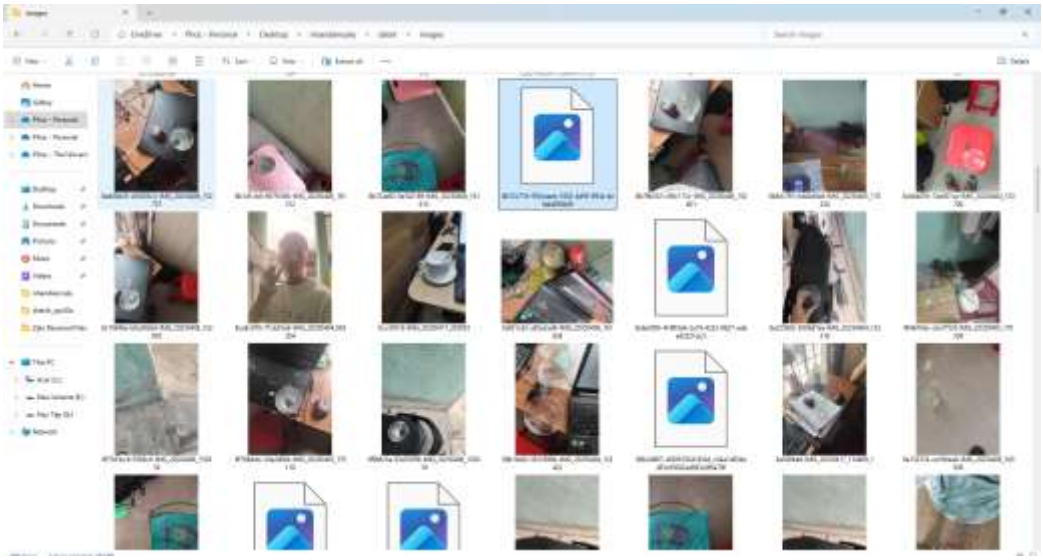
Mô hình 1: Nhận diện miệng ly bằng mô hình YOLOv8n



Hình 3.17: Quy trình huấn luyện AI

a) Thu thập dữ liệu

Việc thu thập ảnh chất lượng cao là bước quan trọng để xây dựng tập dữ liệu huấn luyện cho hệ thống nhận diện miệng ly bằng trí tuệ nhân tạo (AI). Chụp ảnh miệng ly từ nhiều góc độ, điều kiện ánh sáng khác nhau.



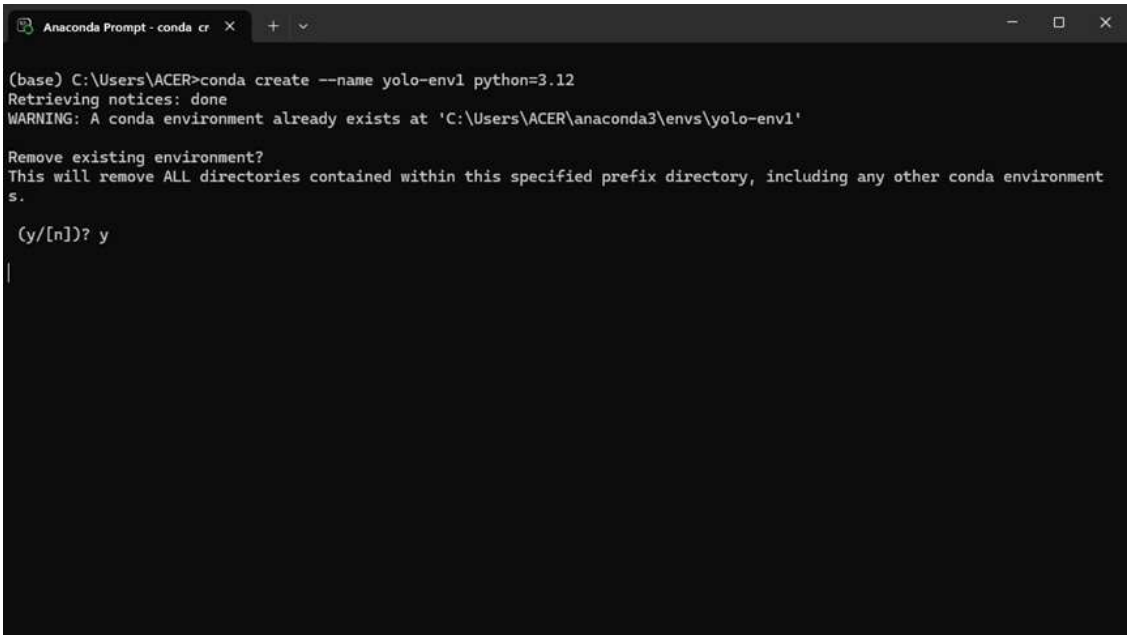
Hình 3.18: Dữ liệu hình ảnh được thu thập vào 1 file

Nhóm đã tiến hành thu thập tổng cộng 605 hình ảnh miệng ly, được chụp từ nhiều góc nhìn và trong các điều kiện ánh sáng đa dạng nhằm tăng cường tính tổng quát và khả năng thích ứng của mô hình.

b) Gắn nhãn dữ liệu

Việc gắn nhãn dữ liệu một cách chính xác và hiệu quả là bước nền tảng để xây dựng các hệ thống AI đáng tin cậy. Anaconda cung cấp môi trường tích hợp với các thư viện như pandas và Label Studio để tối ưu hóa quy trình gắn nhãn

Bước 1: Sau khi tải và cài đặt Anaconda trên máy tính, ta khởi động Anaconda prompt và gõ lệnh "conda create --name yolo-env1 python=3.12" để khởi tạo môi trường làm việc mới



Hình 3.19: Anaconda prompt

Sau đó dùng lệnh "conda activate yolo-env1" để sử dụng môi trường vừa khởi tạo.

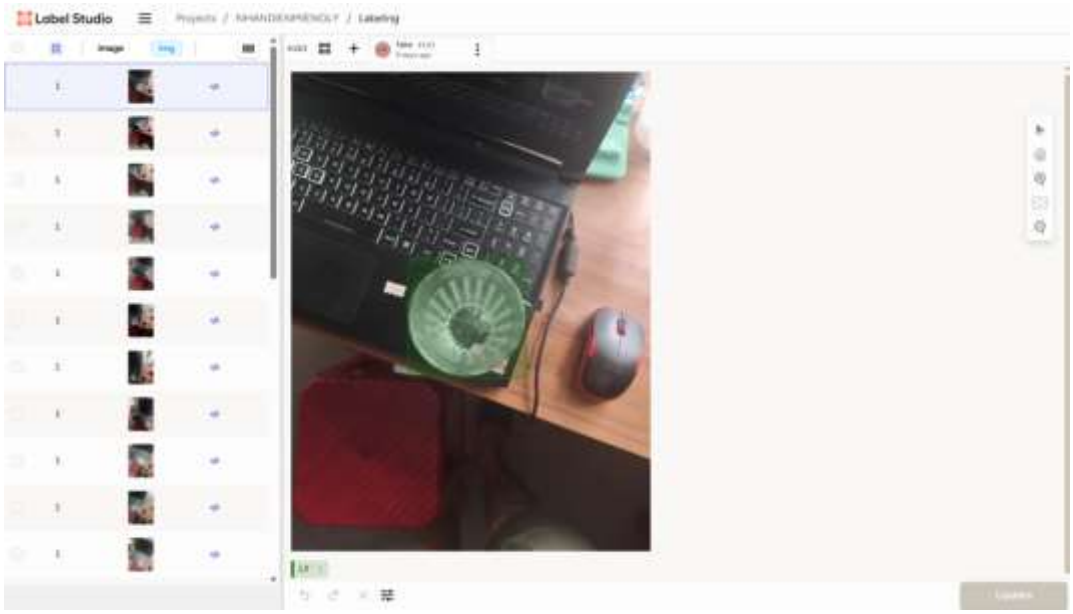
Bước 2: Tải và cài đặt Label Studio vào môi trường vừa khởi tạo bằng dòng lệnh "pip install label-studio"

Việc cài đặt này có thể mất vài phút để thực hiện.

Bước 3: Sử dụng lệnh "label-studio start" để khởi tạo và chạy label studio và chương trình sẽ chạy 1 trang web có đường dẫn như trong ảnh

Hình 3.20: Đường dẫn đến trang web

Đăng nhập và sử dụng label studio để gắn nhãn



Hình 3.21: Tiến hành gắn nhãn

Sau khi hoàn tất quá trình gán nhãn cho tập dữ liệu, nhóm đã tiến hành xuất các tệp nhãn về máy cục bộ để chuẩn bị cho bước huấn luyện mô hình.

c) Tiền xử lý dữ liệu

- Giải nén dữ liệu

Bộ dữ liệu đã được nén trong tệp data.zip và được giải nén trực tiếp vào thư mục /content/custom_data trên môi trường Google Colab.

Thao tác này giúp tổ chức dữ liệu huấn luyện một cách gọn gàng và thuận tiện cho các bước tiền xử lý và huấn luyện mô hình sau đó.

- Phân chia dữ liệu

Để chia bộ dữ liệu ban đầu thành tập huấn luyện và tập kiểm tra, một script có sẵn từ GitHub là train_val_split.py đã được sử dụng. Script này giúp tự động phân chia dữ liệu với tỉ lệ 90% dành cho huấn luyện và 10% cho kiểm tra

Kết quả: Dữ liệu được tổ chức lại thành các thư mục chuẩn cho YOLO, bao gồm:

+ train/images, train/labels – chứa ảnh và nhãn dùng để huấn luyện
+ validation/images, validation/labels – chứa ảnh và nhãn dùng để kiểm tra mô hình

- Tạo tệp cấu hình YAML cho dữ liệu

Để mô hình YOLO có thể hiểu cấu trúc của dữ liệu huấn luyện, cần tạo file cấu hình data.yaml. File này chứa các thông tin quan trọng như đường dẫn dữ liệu, số lượng lớp và tên các lớp.

Một hàm Python đã được xây dựng nhằm tự động tạo file data.yaml dựa trên nội dung của file classes.txt. File classes.txt bao gồm danh sách tên các lớp đối tượng cần nhận diện (trong trường hợp này là: mouth_cup).

d) Huấn luyện mô hình

- Cài đặt thư viện cần thiết

Trước khi bắt đầu huấn luyện, thư viện ultralytics – cung cấp các tiện ích để sử dụng mô hình YOLOv8

- Cấu hình huấn luyện

Mô hình: YOLOv8n (yolov8n.pt) – là phiên bản nhẹ (nano) của YOLOv8, thích hợp cho các hệ thống yêu cầu tốc độ cao và tài nguyên thấp.

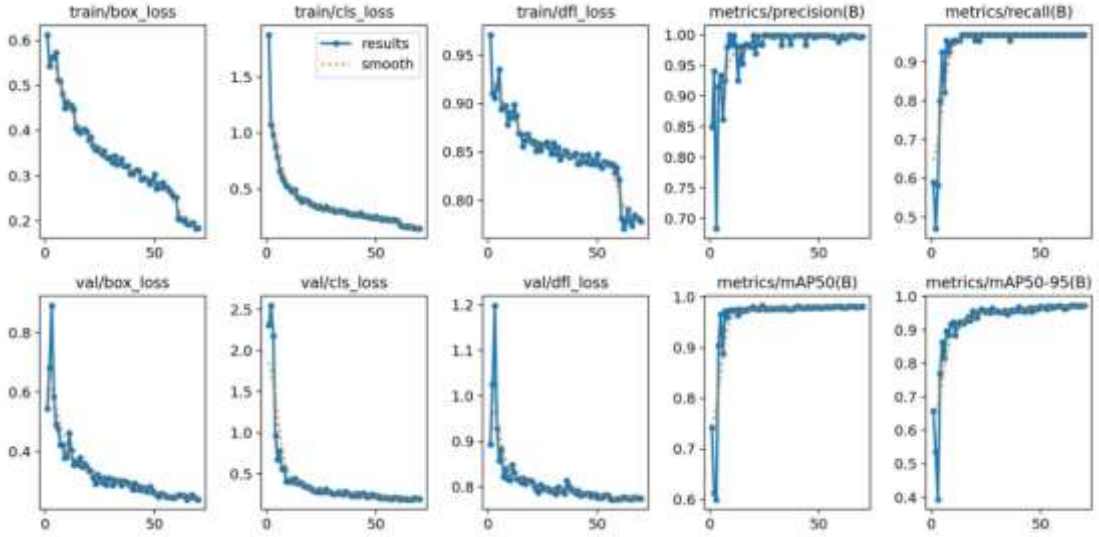
Số vòng lặp (epochs): 70

Kích thước ảnh: 320×320 pixel

File cấu hình dữ liệu: /content/data.yaml, bao gồm thông tin đường dẫn dữ liệu, số lượng và tên các lớp.

- Thực thi huấn luyện

Kết quả của quá trình huấn luyện



Hình 3.22: Kết quả của quá trình huấn luyện nhận diện miệng ly

Nhận xét

Trong quá trình huấn luyện, mô hình có sự hội tụ ổn định và đạt kết quả tốt. Các chỉ số loss trên tập huấn luyện (train) và tập kiểm tra (validation) đều giảm dần theo số vòng lặp (epoch) và ổn định về cuối. Cụ thể:

+ Hàm mất mát box_loss giảm từ 0.6 xuống còn khoảng 0.2.
+ Hàm mất mát cls_loss giảm mạnh từ khoảng 1.5 (train) và 2.5 (val) về dưới 0.3.
+ Hàm mất mát dfl_loss giảm ổn định từ khoảng 1.0 về gần 0.8.

Các chỉ số đánh giá độ chính xác đều đạt giá trị cao:

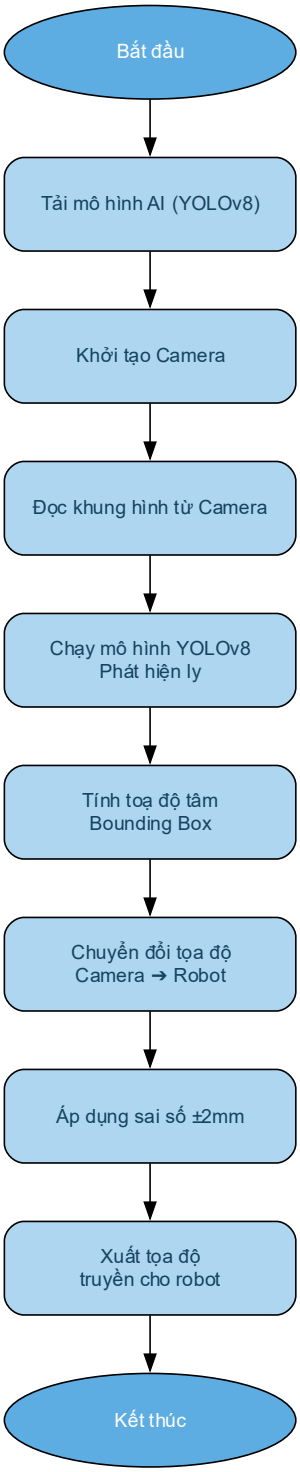
+ Precision (độ chính xác) đạt trên 95%.
+ Recall (độ bao phủ) đạt trên 90%.

+ mAP50 (mean Average Precision với IoU = 50%) đạt gần 100%.
+ mAP50-95 đạt trên 90%, phản ánh khả năng tổng quát hóa tốt của mô hình.

Kết quả trên cho thấy mô hình đã học tốt, không có hiện tượng quá khớp (overfitting), và có thể sẵn sàng áp dụng vào bài toán nhận diện miệng ly trong thực tế.

Hiển thị và đọc tọa độ ly

a) Lưu đồ thuật toán



Hình 3.23: Lưu đồ thuật toán nhận diện miệng ly

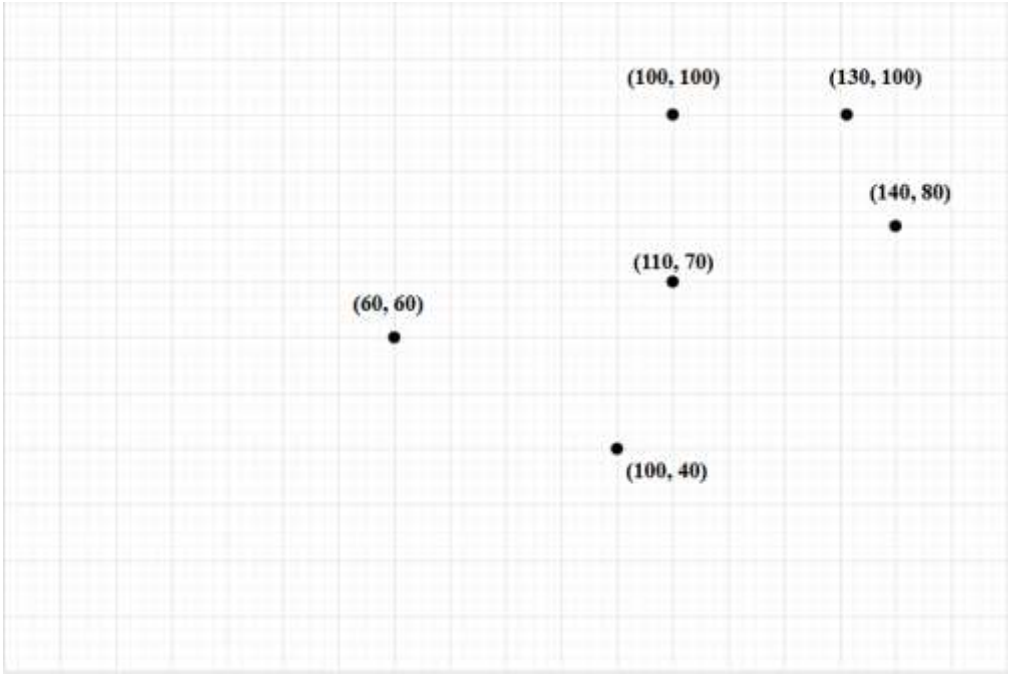
b) Chuyển đổi tọa độ từ ảnh (pixel) sang hệ toạ độ thực (mm)

Bài toán đặt ra là làm sao để xử lý được tọa độ của tâm miệng ly để chuyển từ tọa độ pixel sang tọa độ số để truyền qua cho robot.

Để chuyển đổi chính xác tọa độ tâm miệng ly từ ảnh về hệ tọa độ thực của robot, nhóm sử dụng phép biến đổi đồng nhất hình học (Homography). Phương pháp này cho phép ánh xạ các điểm trên mặt phẳng ảnh sang mặt phẳng làm việc thực tế, kể cả khi camera đặt nghiêng hoặc có biến dạng phối cảnh.

Quy trình thực hiện

- Chọn trước 6 điểm chuẩn trên mặt phẳng làm việc và đo chính xác tọa độ thực của chúng.



Hình 3.24: Lấy tọa độ 6 điểm trên thực tế

- Lấy tọa độ pixel tương ứng của 6 điểm này trên ảnh.
- Sử dụng hàm cv2.findHomography() trong OpenCV để tính toán ma trận Homography.
- Dùng ma trận này để chuyển đổi các tọa độ pixel thu được từ mô hình AI sang tọa độ thực của robot.

Việc sử dụng Homography giúp hệ thống hoạt động chính xác hơn và giảm sai số khi camera không đặt vuông góc hoàn toàn với mặt phẳng làm việc.

c) Kết quả

Đây là kết quả thử lại với tọa độ (60,60), cho thấy sai số trong phép đo là 2mm (phù hợp trong bài toán nhận diện miệng ly)

Hình 3.25: Chạy thử mô hình nhận diện miệng ly

- Nhận xét

+ Hiệu suất giảm trong điều kiện ánh sáng cực đoan
+ Chưa nhận diện tốt với các loại ly có thiết kế đặc biệt
+ Cần cải thiện độ chính xác với góc nhìn nghiêng

- Hướng phát triển trong tương lai

+ Tích hợp nhận diện 3D để xác định cả độ sâu và hướng của miệng ly
+ Phát triển mô hình siêu nhẹ (<1MB) cho thiết bị IoT
+ Mở rộng mô hình để nhận diện các thuộc tính khác của ly như vết nứt, biến dạng

Mô hình 2: ĐO LƯỜNG % BIA CÓ TRONG LY

Quy trình thực hiện

- Thu thập và chuẩn bị dữ liệu

+ Số lượng ảnh: 1000 ảnh
+ Nguồn dữ liệu: Chụp trực tiếp, dữ liệu mở, và tổng hợp
+ Điều kiện ánh sáng đa dạng (trong nhà, ngoài trời,...)
+ Các góc độ chụp khác nhau
+ Nền đa dạng (bàn, quầy bar, tay cầm,...)

→ Sử dụng công cụ LabelImg để gán nhãn hộp giới hạn cho ly

→ Phân chia dữ liệu: 90% huấn luyện, 10% kiểm tra

- Huấn luyện mô hình

+ Mô hình: Sử dụng YOLOv8n (phiên bản nhẹ) từ Ultralytics.

+ Môi trường: Google Colab với GPU (NVIDIA RTX 3060 hoặc tương đương) để huấn luyện.

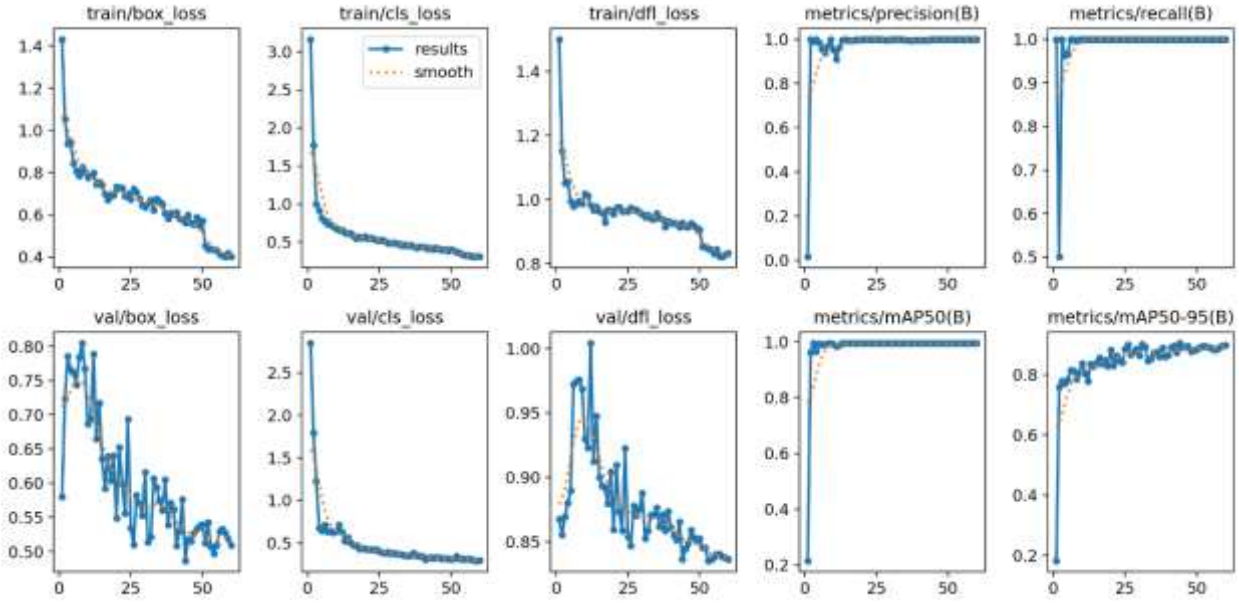
- Tham số huấn luyện:

+ Epochs: 60

+ Kích thước ảnh: 320x320.

+ Batch size: Mặc định của YOLOv8n

Kết quả của quá trình huấn luyện



Hình 3.26: Kết quả mô hình nhận diện thân ly

- **Nhận xét**

Loss function (train/val)

+ Các hàm mất mát (box_loss, cls_loss, dfl_loss) ở cả tập huấn luyện (train) và kiểm tra (val) đều giảm dần ổn định theo số epoch.

+ Trong giai đoạn đầu (0–20 epoch), các giá trị loss giảm nhanh, sau đó tiếp tục giảm dần và hội tụ ổn định sau khoảng 50 epoch.

+ Val/box_loss dao động nhẹ ở các epoch sau, thể hiện mô hình vẫn còn điều chỉnh nhẹ nhưng không có dấu hiệu overfitting.

Precision và Recall:

+ Precision và Recall đều tăng nhanh ngay từ đầu và đạt giá trị gần 1 sau khoảng 20 epoch, chứng tỏ mô hình nhận diện tốt, chính xác cao và độ bao phủ tốt.

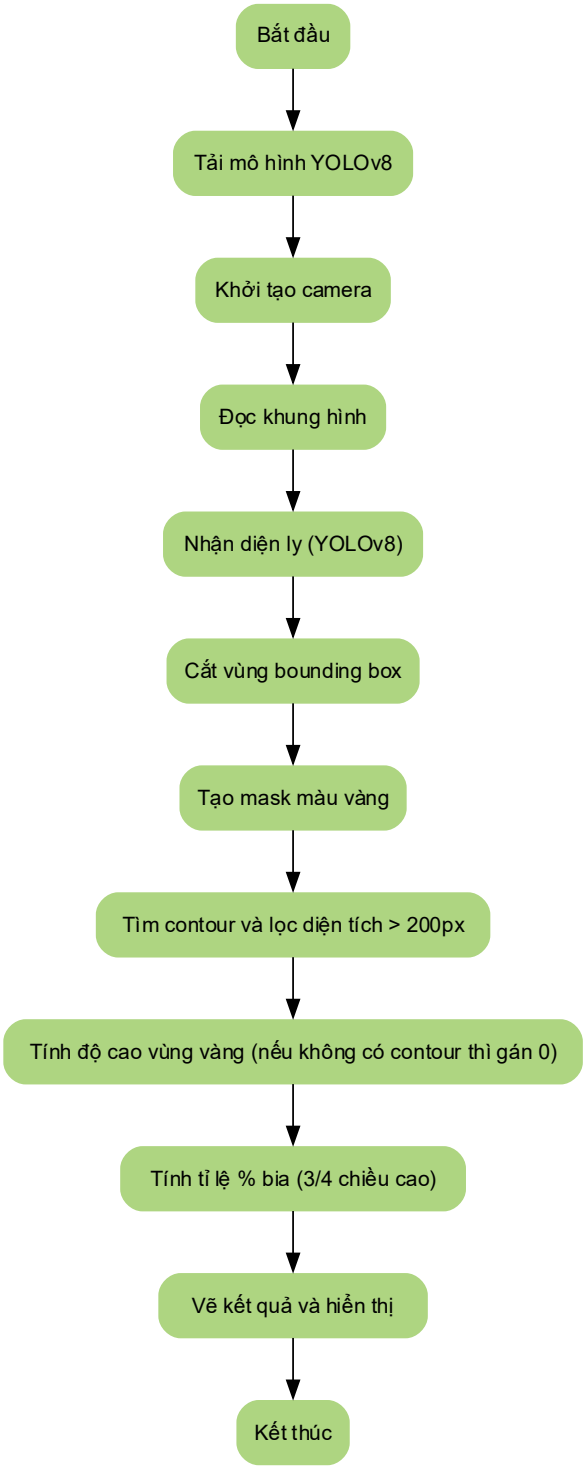
mAP (Mean Average Precision):

+ mAP50 nhanh chóng đạt gần 1, thể hiện khả năng nhận diện rất chính xác với ngưỡng IoU 50%.

+ mAP50-95 tăng dần đều và đạt mức trên 0.8 vào cuối quá trình huấn luyện, phản ánh mô hình giữ được độ chính xác cao trên nhiều mức IoU khác nhau.

Xác định phần trăm bia có trong ly

a) Lưu đồ thuật toán



Hình 3.27: Lưu đồ thuật toán đọc % bia

b) Xử lý ngưỡng để đọc phần trăm bia trong ly

- Phương pháp xử lý ngưỡng

Chuyển đổi và phân đoạn màu vàng: Vùng ảnh từ bounding box được chuyển đổi sang không gian màu HSV để tách biệt vùng bia. Chuyển đổi sang HSV để phân đoạn màu vàng dựa trên Hue (15-35), Saturation (50-255), và Value (80-255). Tạo mask nhị phân, gán pixel trong phạm vi HSV là 255 (vùng bia) và ngoài phạm vi là 0.

Phân tích vùng bia: Vùng màu vàng được phân tích để xác định chiều cao và loại bỏ nhiễu. Tìm contour trong mask nhị phân và lọc các contour có diện tích lớn hơn 200 pixel để loại bỏ nhiễu như bọt bia. Đo chiều cao vùng bia bằng cách xác định điểm cao nhất (tọa độ y lớn nhất) và thấp nhất (tọa độ y nhỏ nhất).

Tính tỉ lệ và hiển thị kết quả: Tỷ lệ bia được tính toán theo công thức và hiển thị trực quan trên khung hình video.

Tính tỷ lệ phần trăm theo công thức

$$\text{Ti le} = \left(\frac{\text{Chieu cao vung bia}}{\frac{3}{4}\text{Chieu cao bounding box}} \right) \times 100 \qquad (3.83)$$

Công thức sử dụng 3/4 chiều cao bounding box làm mức đầy tham chiếu để ước lượng tỷ lệ bia.

Hiển thị tỷ lệ trên video, dùng màu xanh nếu tỷ lệ ≥ 100% và đỏ nếu < 100%, kèm vùng bia được làm nổi bật.

c) Kết quả

Hình 3.28: Kết quả đọc % bia trong ly

- Ưu điểm

+ Phương pháp cung cấp kết quả nhanh và đáng tin cậy trong điều kiện ánh sáng ổn định.

+ Hiệu quả, tốc độ cao, phù hợp với xử lý thời gian thực.

+ Chính xác khi ánh sáng ổn định nhờ phân đoạn HSV và lọc contour.

- Hạn chế

+ Phương pháp gặp thách thức trong môi trường ánh sáng phức tạp và khi có bọt bia.

+ Phụ thuộc ánh sáng, phạm vi HSV cố định kém linh hoạt khi ánh sáng thay đổi.

+ Nhiễu từ bọt bia do màu sắc tương tự vùng bia, gây sai lệch phân đoạn.

Bảng 3.12: Góc nghiêng với từng % bia

% bia có trong ly	Góc θ_4
0-30%	35
30-50%	45
50-80%	60
80-90%	75
90%	50
100%	0

Góc nghiêng θ_4 của cánh tay robot được xác định thông qua quá trình đo lường thực nghiệm thay vì tính toán lý thuyết. Nhóm tiến hành thử rót bia vào ly ở các mức đầy khác nhau, quan sát dòng chảy và điều chỉnh góc nghiêng thủ công đến khi đạt hiệu quả tốt nhất (bia chảy đều, không trào, không sủi bọt). Các góc này được ghi nhận và lập thành bảng tra (Bảng 3.12), ứng với từng mức phần trăm bia trong ly.

Phương pháp đo thực tế này giúp phản ánh chính xác điều kiện môi trường và đặc tính vật lý như độ sánh, tốc độ chảy, hình dạng ly, cho kết quả ổn định hơn so với mô phỏng. Hệ thống sau khi áp dụng bảng góc nghiêng đã thực hiện quá trình rót bia chính xác và mượt mà, đáp ứng tốt yêu cầu thực tế.

CHƯƠNG 4: MÔ PHỎNG

4.1. Mô phỏng kiểm chứng dựa trên MATLAB Simscape Multibody

4.1.1. Mô phỏng kiểm chứng tính toán động học thuận.

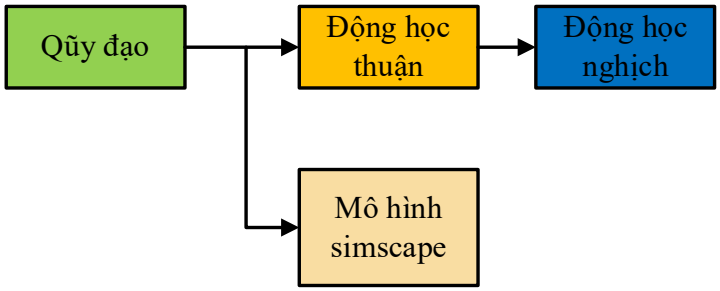
Để kiểm chứng tính đúng đắn của việc tính toán động học thuận chúng ta đi so sánh vị trí điểm đầu cuối từ phương trình động học thuận đã tính toán với vị trí điểm đầu cuối trên mô hình mô phỏng Simscape.

Mô phỏng kiểm chứng tính toán động học thuận với 2 trường hợp góc đặt:

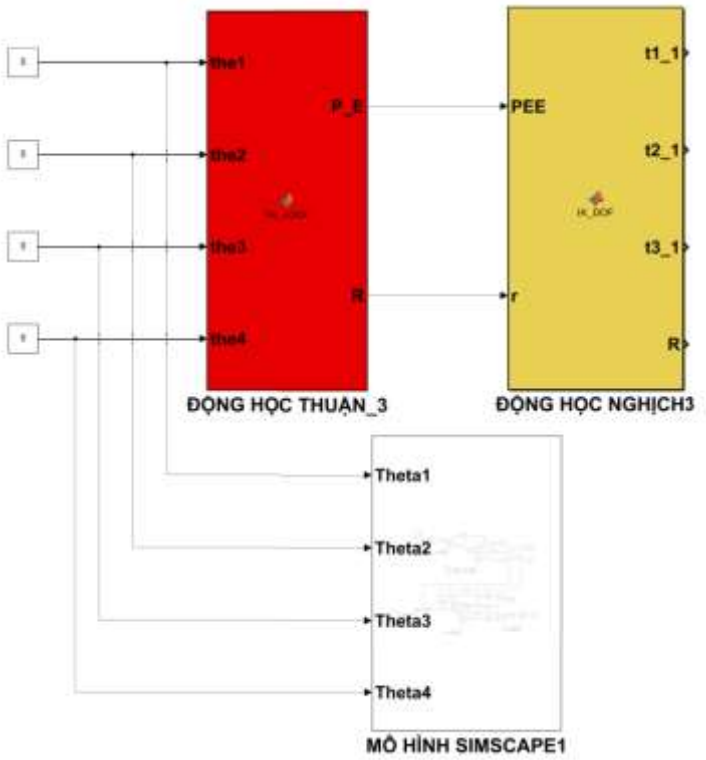
Trường hợp 1: Giá trị đặt 1 $\theta_1 = 0, \theta_2 = 0, \theta_3 = 0, \theta_4 = 90$

Trường hợp 2: Giá trị đặt 2 $\theta_1 = 0, \theta_2 = 0, \theta_3 = -90, \theta_4 = 90$

Từ 2 kết quả kiểm nghiệm trên để kết luận tính chính xác của mô hình toán động học thuận.



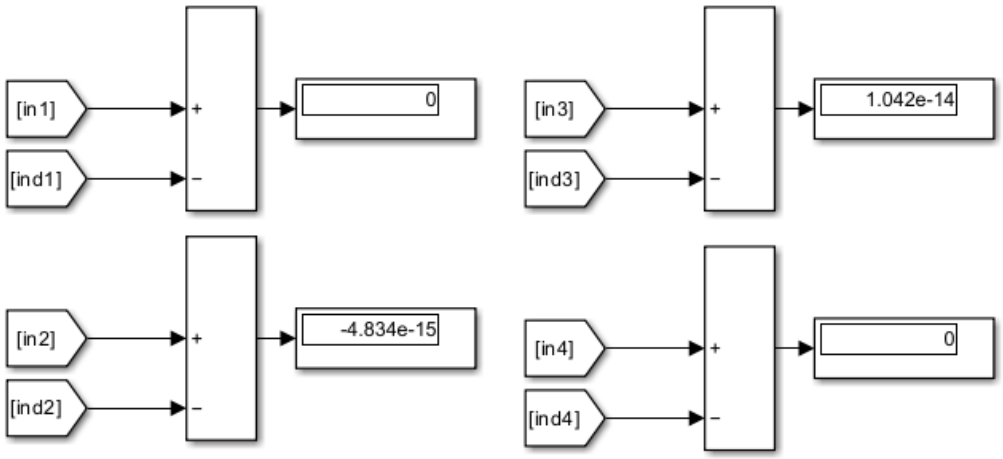
Hình 4.1: Sơ đồ khối kiểm chứng tính toán động học thuận



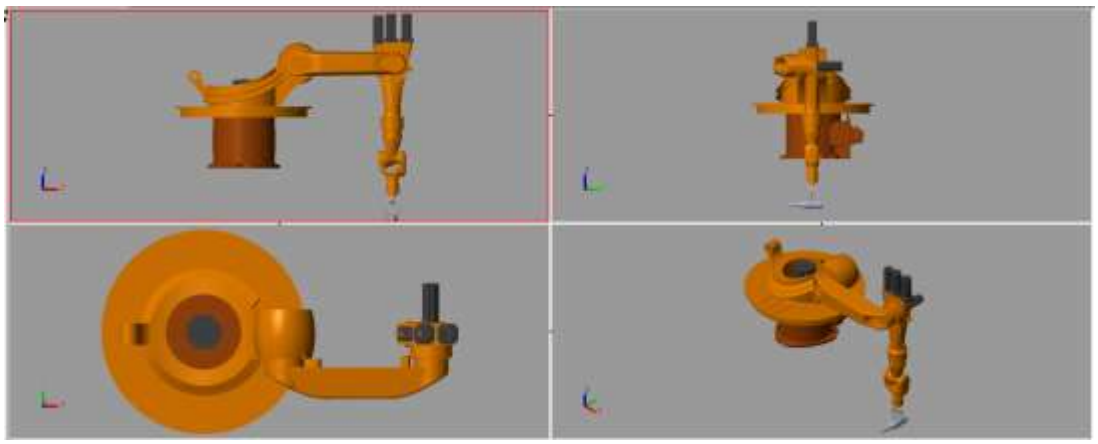
Hình 4.2: Mô hình kiểm chứng tính toán động học thuận trên matlab simulink

Ta cho các ngõ vào của khối "Động học thuận" là vị trí góc quay tại mỗi khớp ,ngõ ra là vị trí của điểm đầu cuối của robot. Đồng thời ngõ ra của khối "Động học thuận" cũng là ngõ vào của khối "Động học nghịch" và ngõ ra của khối này là vị trí góc quay tại các khớp. Để biết tính đúng sai của phương trình động học mà ta đã tính toán,ta lấy ngõ ra của khối "Động học thuận" là vị trí của điểm đầu cuối so sánh với vị trí ta tính toán được trên mô hình thực. Kiểm chứng việc tính toán phương trình động học thuận

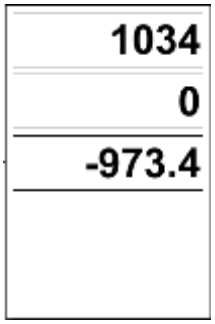
Giá trị đặt 1 $\theta_1 = 0, \theta_2 = 0, \theta_3 = 0, \theta_4 = 90$



Hình 4.3: Kết quả mô phỏng trên Matlab/Simulink



Hình 4.4: Hình dạng robot với các góc quay trên



Hình 4.5: Vị trí điểm cuối của robot

Tại các khớp của robot tất cả vị trí góc quay bằng 0 nên robot có hình dạng như ban đầu. Ta có thể tính được vị trí điểm P so với trục X_0 của hệ tọa độ gốc:

$P_x = L_2 + L_3 = 376.11 + 657.5 = 1033.61$

Vị trí điểm P so với trục Y_0:

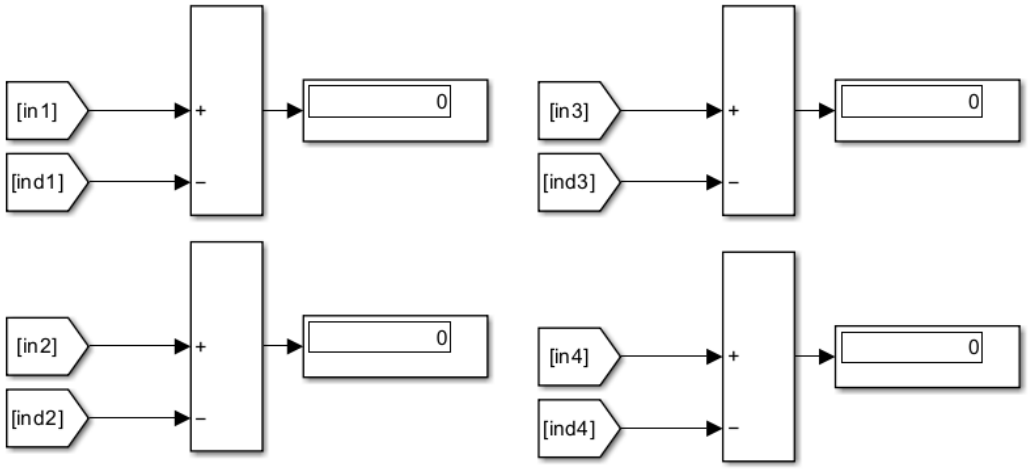
$P_y = 1033.61$

Vị trí điểm P so với trục Z_0

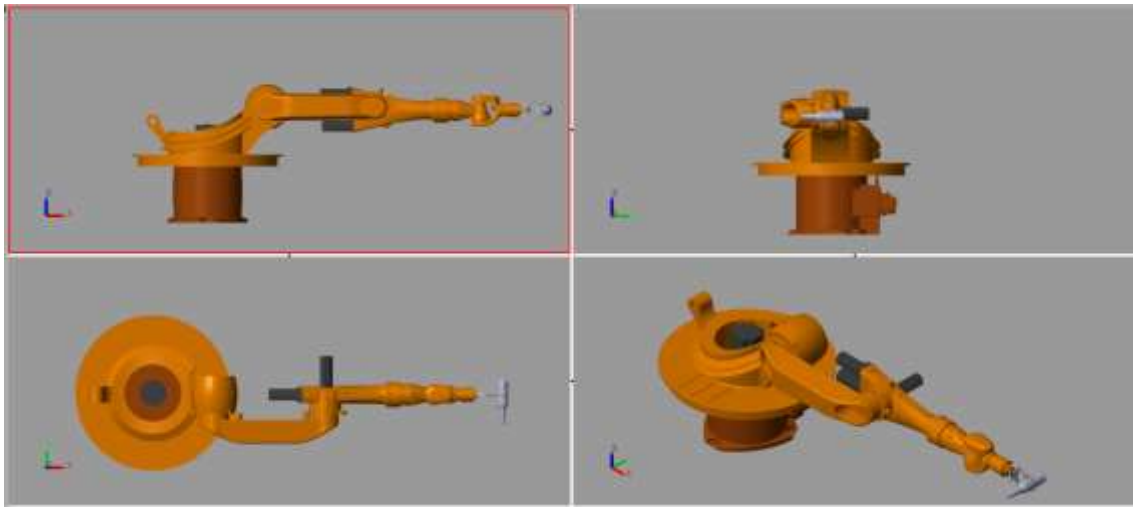
$P_z = -(L_4 + L_5) = -(430.43 + 543.01) = -973.44$

Từ đó ta tính được vị trí điểm đầu cuối dựa trên thông số mô hình thực có tọa độ $P(P_x, P_y, P_z)$ hay $P(1034, -973.4)$ sau đó ta so sánh vị trí điểm P tính toán với kết quả mô phỏng được từ phương trình động học thuận có thể thấy hai kết hoàn toàn gần giống nhau nên phương trình động học thuận ta đã tính toán là hoàn toàn đúng.

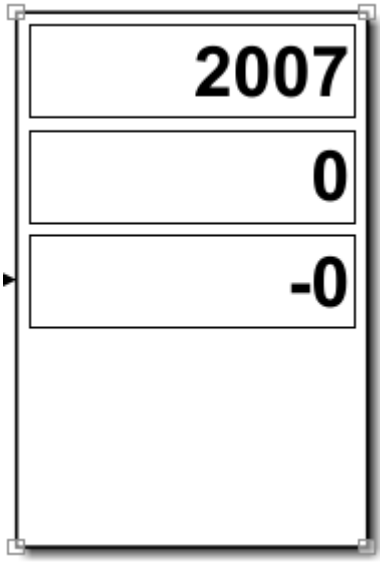
Giá trị đặt 2: $\theta_1 = 0, \theta_2 = 0, \theta_3 = -90, \theta_4 = 90$



Hình 4.6: Kết quả mô phỏng trên Matlab/Simulink



Hình 4.7: Hình dạng robot với các góc quay trên



Hình 4.8: Vị trí điểm cuối cuổi robot

Với giá trị đặt 2, tại khớp 3 quay góc -90, nên robot có hình dạng như hình 4.7. Với hình dáng trên robot có điểm đầu cuối xa gốc tọa độ nhất.ta có thể tính được vị trí điểm P so với trục X_0 của hệ tọa độ gốc:

$$P_x = L_2 + L_3 + L_4 + L_5 = 376.11 + 657.5 + 430.43 + 543.01 = 2007.5$$

Vị trí điểm P so với trục Y_0: $P_y = 0$

vị trí điểm P so với trục Z_0: $P_z = 0$

Từ đó ta tính được vị trí điểm đầu cuối dựa trên thông số mô hình thực có tọa độ $P(P_x, P_y, P_z)$ hay $P(2007.5, 0)$ sau đó ta so sánh vị trí điểm P tính toán với kết quả mô phỏng được từ phương trình động học thuận có thể thấy hai kết quả hoàn toàn giống nhau nên phương trình động học thuận ta đã tính toán là hoàn toàn đúng.

Kết luận: Từ 2 kết quả kiểm nghiệm trên ta chứng minh được giá trị tính toán mô hình động học thuận hoàn toàn chính xác.

4.1.2. Kiểm chứng động học nghịch

Kiểm chứng mô hình động học nghịch với 2 trường hợp:

Bước 1: Xác định vị trí mong muốn

Trường hợp 1: $P_{EE}\begin{bmatrix}906.5 & 329.9 & -1392\end{bmatrix}^T$

Trường hợp 2: $P_{EE}\begin{bmatrix}2007 & 0 & 0\end{bmatrix}^T$

Bước 2: Giải phương trình động học nghịch

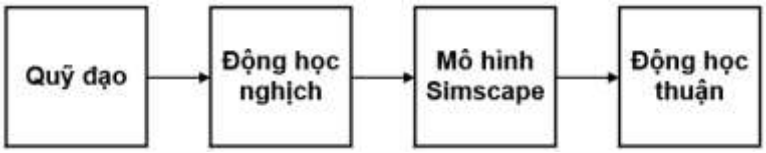
Bước 3:Đưa giá trị góc tính toán từ khối động học nghịch vào mô hình

Simscape để xuất ra hình ảnh mô phỏng vị trí điểm cuối.

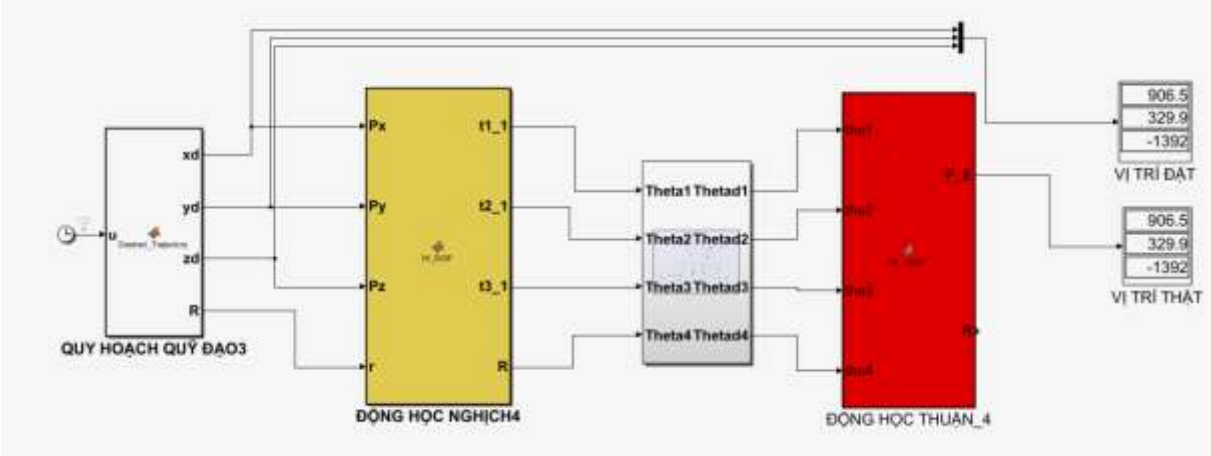
Bước 4:Dùng mô hình động học thuận để tính ra giá trị điểm cuối.

Bước 5: So sánh giá trị vị trí đặt và vị trí tính được từ khối động học thuận thu được.

⇨ Kết luận tính chính xác của mô hình toán động học nghịch.



Hình 4.9: Sơ đồ khối kiểm chứng tính toán động học nghịch



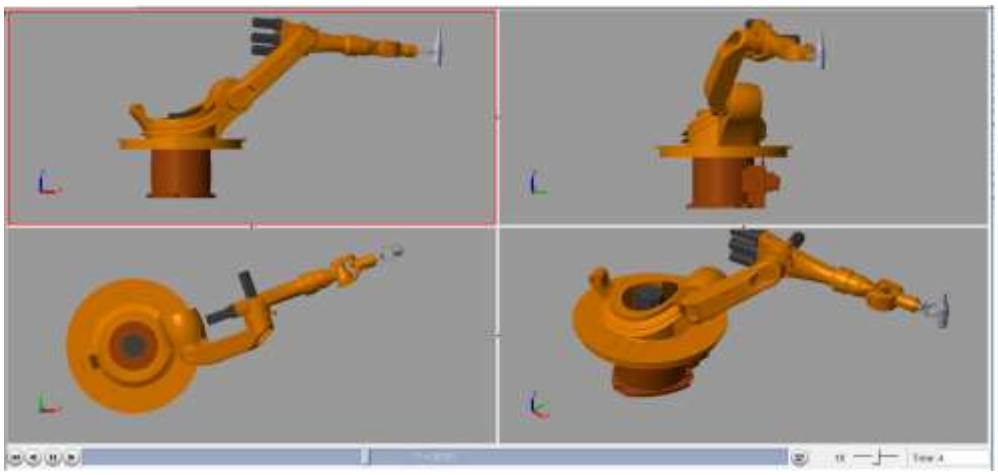
Hình 4.10: Mô hình kiểm chứng động học nghịch trên Matlab/Simulink.

Khối quy hoạch quỹ đạo: đưa vào giá trị tọa độ mong muốn. Khối động học nghịch nhận giá trị từ khối quy hoạch quỹ đạo tọa độ mong muốn, để từ đó tính được các góc $\theta_1, \theta_2, \theta_3, \theta_4$,phù hợp với từng bộ nghiệm. Khối động học thuận là khối nhận giá trị góc khớp được tính từ khối mô phỏng chuyển của robot và feedback về giá trị góc khớp.

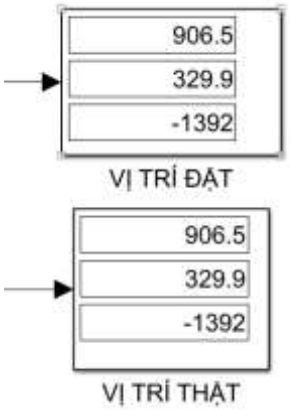
Kiểm chứng mô hình với 2 trường hợp sau:

Trường hợp 1: đưa giá trị đầu vào khối quy hoạch quỹ đạo là: $P_{EE}\,[906.5 \quad 329.9 \quad -1392]^T$

Kết quả:

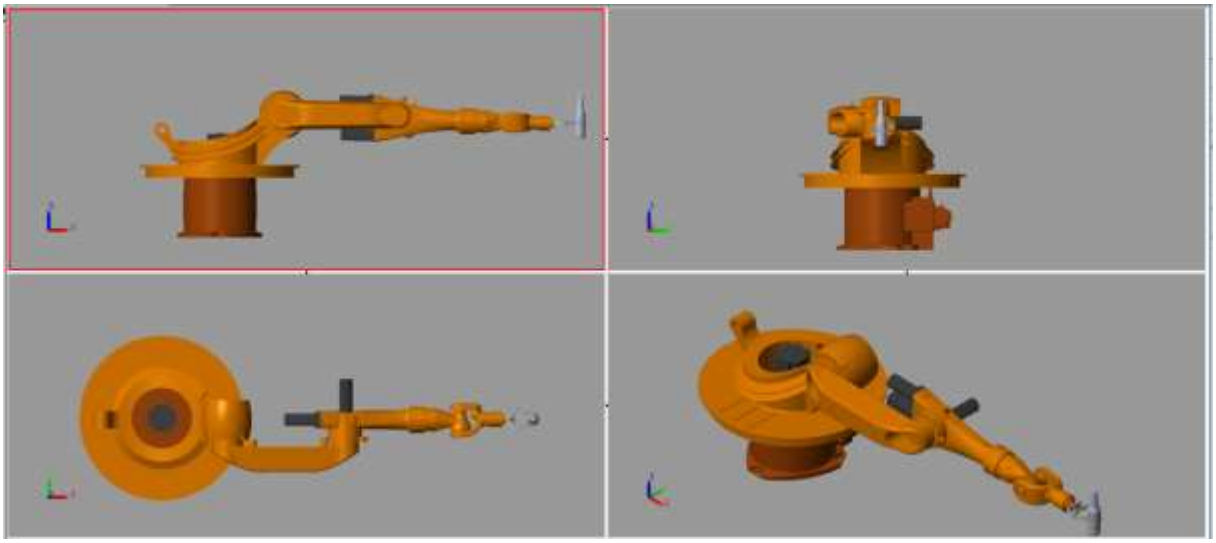


Hình 4.11: Kết quả kiểm chứng Simscape

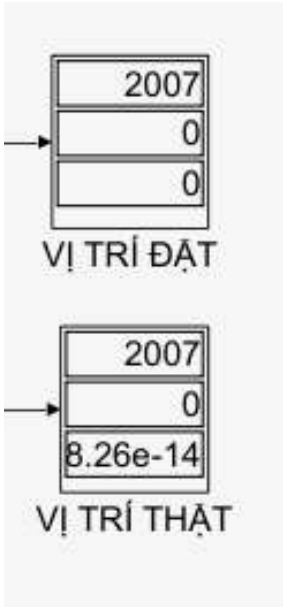


Hình 4.12: Kết quả kiểm chứng Simulink

Trường hợp 2: đưa giá trị đầu vào khối quy hoạch quỹ đạo là: $P_{EE}[2007 \quad 0 \quad 0]^T$



Hình 4.13: Kết quả kiểm chứng Simscape



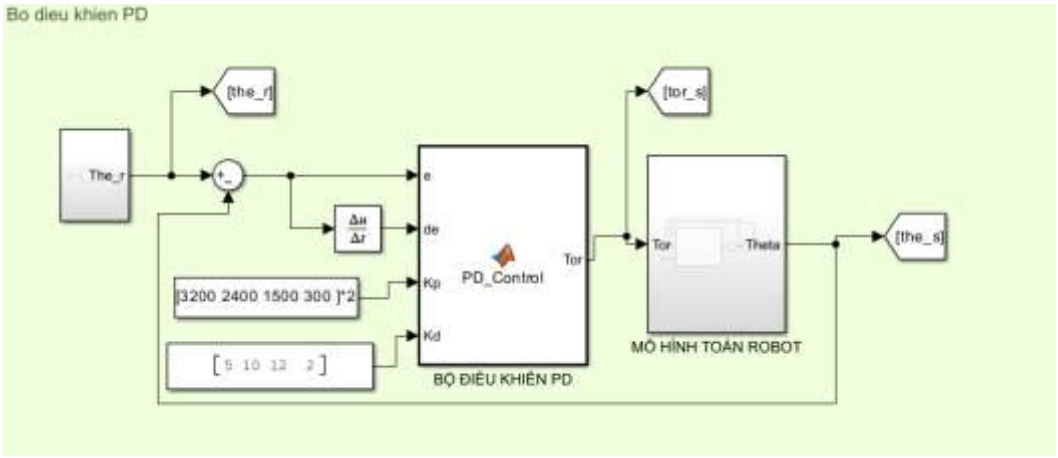
Hình 4.14: Kết quả kiểm chứng Simulink

Khi so sánh kết quả của vị trí điểm đầu cuối giữa giá trị đặt đầu vào với giá trị đầu ra khi đi qua các khối là hai kết quả giống nhau và kết quả giá trị các vị trí góc quay θ mô phỏng được khi so sánh với góc quay tại các khớp trên mô hình simscape thì giống. Từ những phân tích trên, ta có thể kết luận rằng phương trình động học đã tính toán là đúng.

4.1.3. Mô phỏng kiểm chứng thiết kế bộ điều khiển PD cho hệ thống

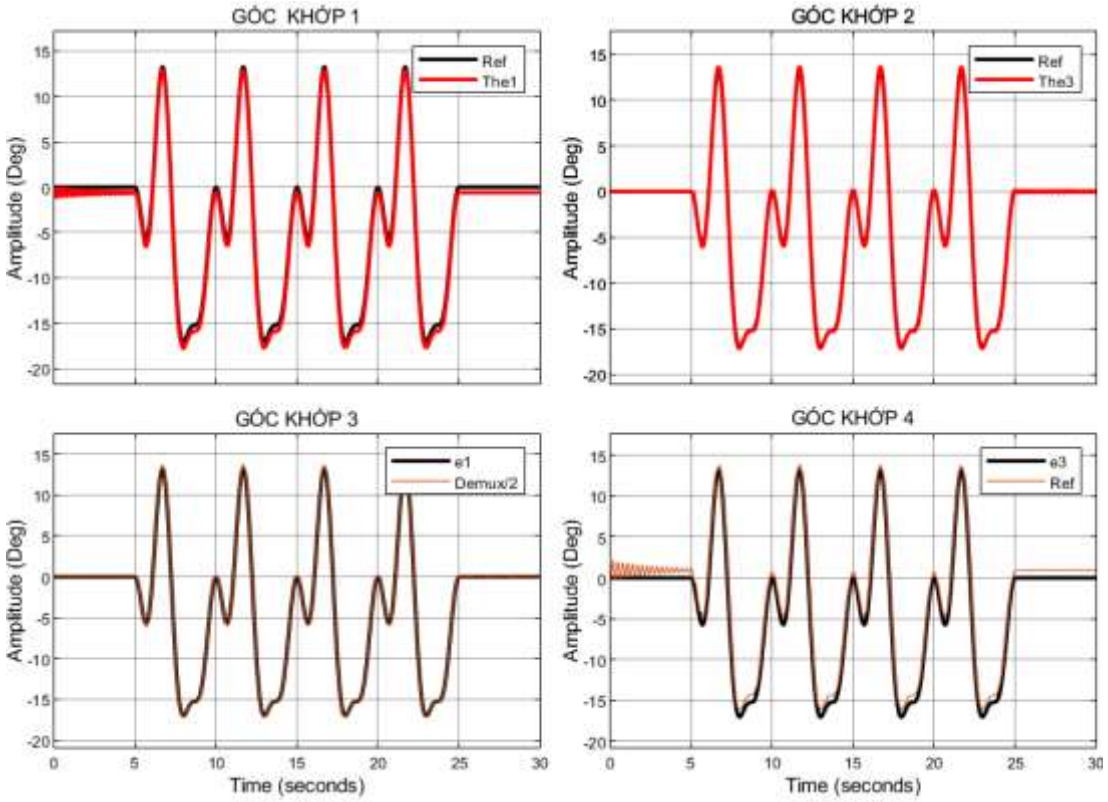
Mô phỏng kiểm chứng thiết kế bộ điều khiển PD dựa vào sơ đồ khối như hình dưới đây.

Hình 4.15: Sơ đồ khối bộ điều khiển PD



Hình 4.16: Sơ đồ mô phỏng kiểm chứng bộ điều khiển PD.

Đáp ứng ngõ ra của từng góc tại mỗi khớp được thể hiện như sau,với đường nét liền màu đen là tín hiệu đặt cho từng góc quay thông qua quỹ đạo được quy hoạch sẵn và đường màu đỏ nét đứt là đáp ứng ngõ ra của mô hình.

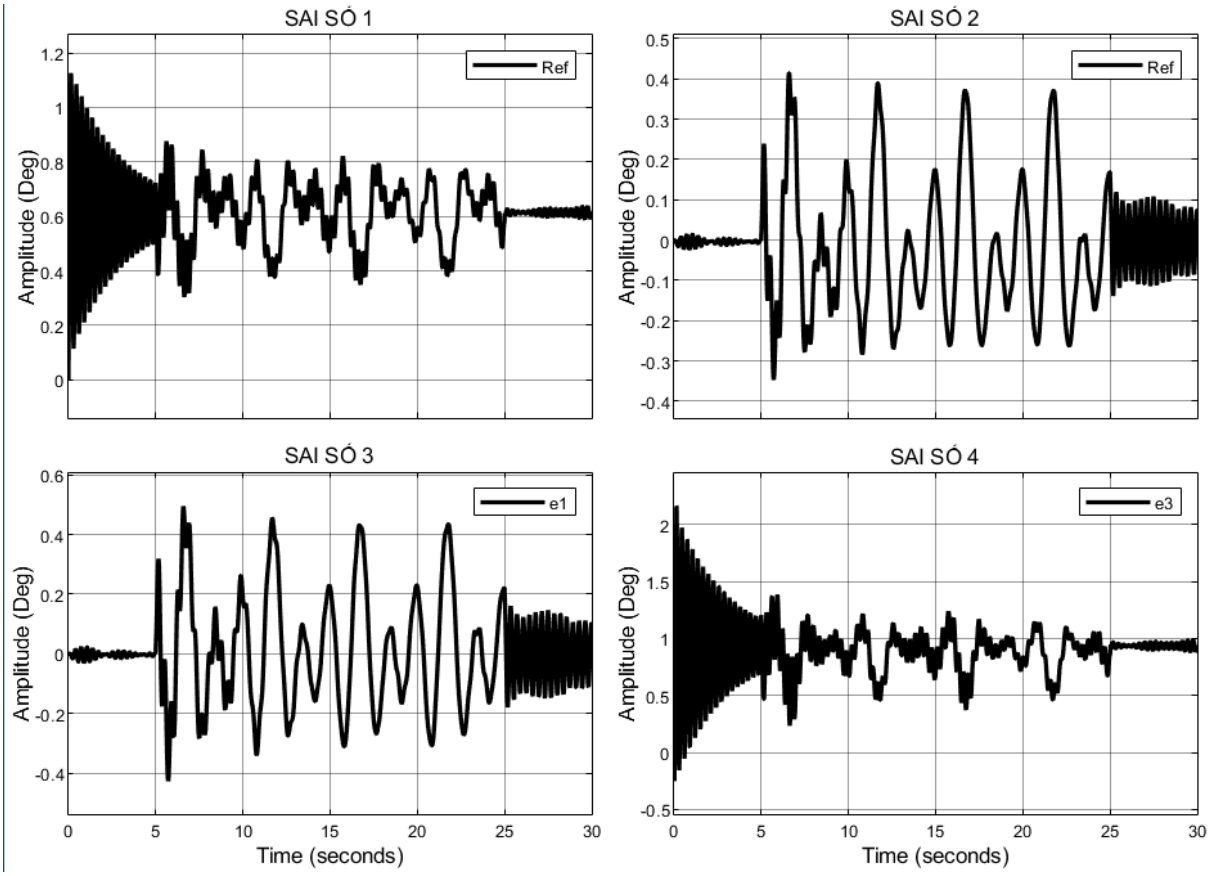


Hình 4.17: Tín hiệu điều khiển và đáp ứng Khớp 1; Khớp 2; Khớp 3; Khớp 4

Từ kết quả mô phỏng trên, có thể rút ra một số nhận xét:

- **Đối với khớp 1 (Hình a):** Tín hiệu điều khiển và đáp ứng thực tế (θ_1) bám khá sát nhau trong suốt quá trình hoạt động. Sai số nhỏ và nhanh chóng được triệt tiêu sau mỗi lần thay đổi tín hiệu điều khiển. Điều này cho thấy bộ điều khiển thực hiện tốt việc theo dõi quỹ đạo mong muốn.
- **Đối với khớp 2 (Hình b):** Đáp ứng (θ_3) cũng bám sát tín hiệu tham chiếu Ref. Tuy có một vài dao động nhẹ ở các điểm chuyển tiếp nhưng nhìn chung hệ thống vẫn đảm bảo độ chính xác điều khiển tốt.
- **Đối với khớp 3 (Hình c):** Đường cong đáp ứng Demux/2 gần như trùng khít với e1, thể hiện sự chính xác rất cao trong điều khiển khớp 3. Sai số không đáng kể.
- **Đối với khớp 4 (Hình d):** Đáp ứng e3 và tín hiệu Ref gần như trùng nhau. Hệ thống điều khiển có độ ổn định cao, đáp ứng nhanh và bám sát tín hiệu tham chiếu trong toàn bộ quá trình.

Tổng kết: Hệ thống điều khiển đã đảm bảo được khả năng điều khiển chính xác vị trí các khớp. Sai số bám quỹ đạo nhỏ, dao động sau quá độ nhanh chóng bị dập tắt. Điều này chứng tỏ rằng bộ điều khiển được thiết kế có độ chính xác cao và phù hợp để ứng dụng điều khiển chuyển động cho cánh tay robot 4 bậc tự do.

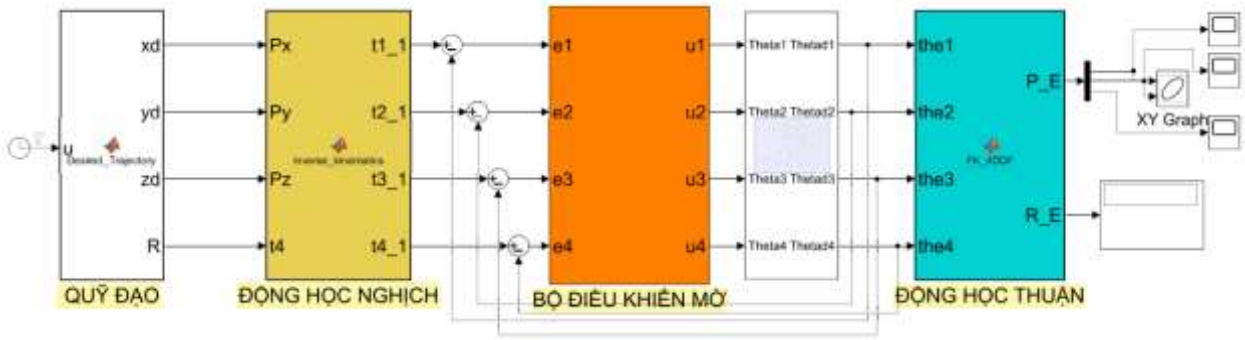


Hình 4.18: Giá trị sai số góc quay Khớp 1; Khớp 2; Khớp 3; Khớp 4

Từ kết quả mô phỏng giá trị sai số trên, có thể rút ra các nhận xét sau:

- **Sai số khớp 1 (Hình a):**
  Sai số ban đầu dao động khá lớn do tác động quá độ ban đầu, sau đó biên độ dao động giảm dần và hội tụ về mức ổn định xấp xỉ khoảng 0.6 độ. Độ hội tụ nhanh, sai số cuối nhỏ chứng tỏ bộ điều khiển hoạt động hiệu quả.
- **Sai số khớp 2 (Hình b):**
  Biên độ dao động ban đầu khoảng ±0.3 độ, sau đó cũng giảm dần và hội tụ về mức nhỏ hơn. Quá trình điều chỉnh dao động nhanh chóng được dập tắt và đạt sai số ổn định nhỏ.
- **Sai số khớp 3 (Hình c):**
  Biên độ dao động nhỏ so với các khớp trước, dao động ban đầu khoảng ±0.2 độ, sau đó hội tụ nhanh về 0. Độ chính xác của điều khiển khớp 3 là khá cao.
- **Sai số khớp 4 (Hình d):**
  Biên độ dao động ban đầu lớn nhất trong các khớp (~1.5 - 2 độ), nhưng quá trình hội tụ diễn ra ổn định và giảm dần về mức sai số nhỏ hơn 0.5 độ. Bộ điều khiển vẫn đảm bảo khả năng triệt tiêu sai số tốt.

Tổng kết: Mặc dù ban đầu các khớp đều xuất hiện dao động quá độ, nhưng sau đó hệ thống đã nhanh chóng ổn định, sai số góc quay được triệt tiêu đáng kể. Điều này chứng minh rằng bộ điều khiển được thiết kế có khả năng thích ứng tốt với hệ thống robot nhiều bậc tự do, đồng thời đảm bảo độ chính xác điều khiển trong quá trình làm việc.



Hình 4.19: Sơ đồ mô phỏng matlab

Phân tích sơ đồ

a) Quy Đạo (Quỹ đạo mong muốn)

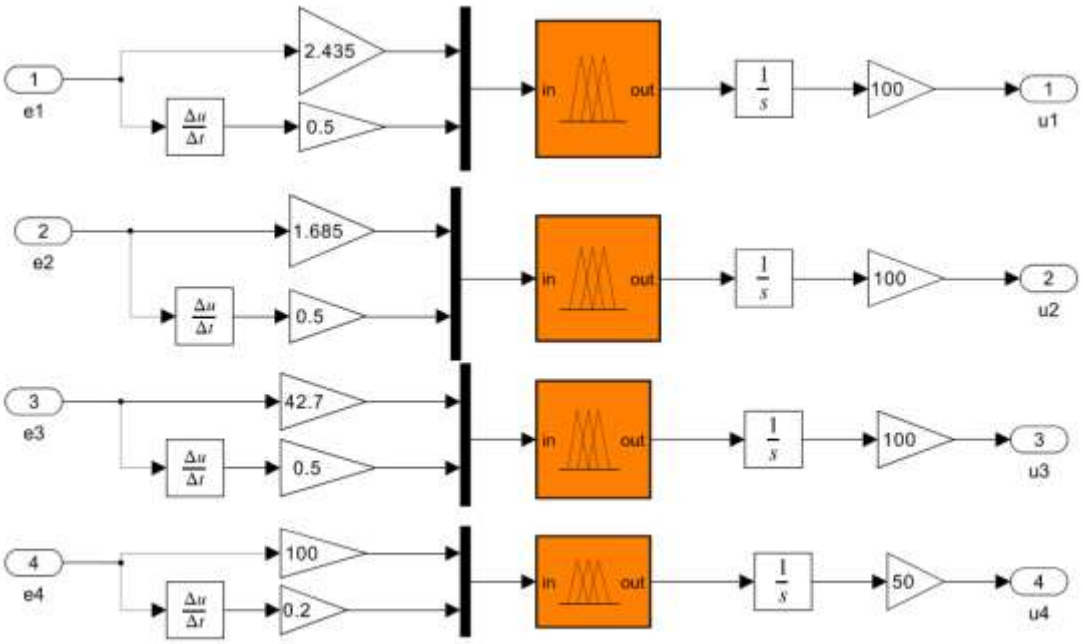
+ Tọa độ ly bia nhận từ bộ xử lí AI ->x,y,z
+ Mức bia đọc được,qua bộ điều khiển AI-> góc quay t4(góc rót bia)

b) Động Học Nghịch (Động học nghịch đảo)

+ Các hàm tính toán sẽ tính ra các giá trị góc quay của từng biến khớp của robot 4dof rót bia từ các giá trị tọa độ x,y,z được gửi về từ xử lí AI.

c) Bộ Điều Khiển (Mô Hình Bộ Điều Khiển)

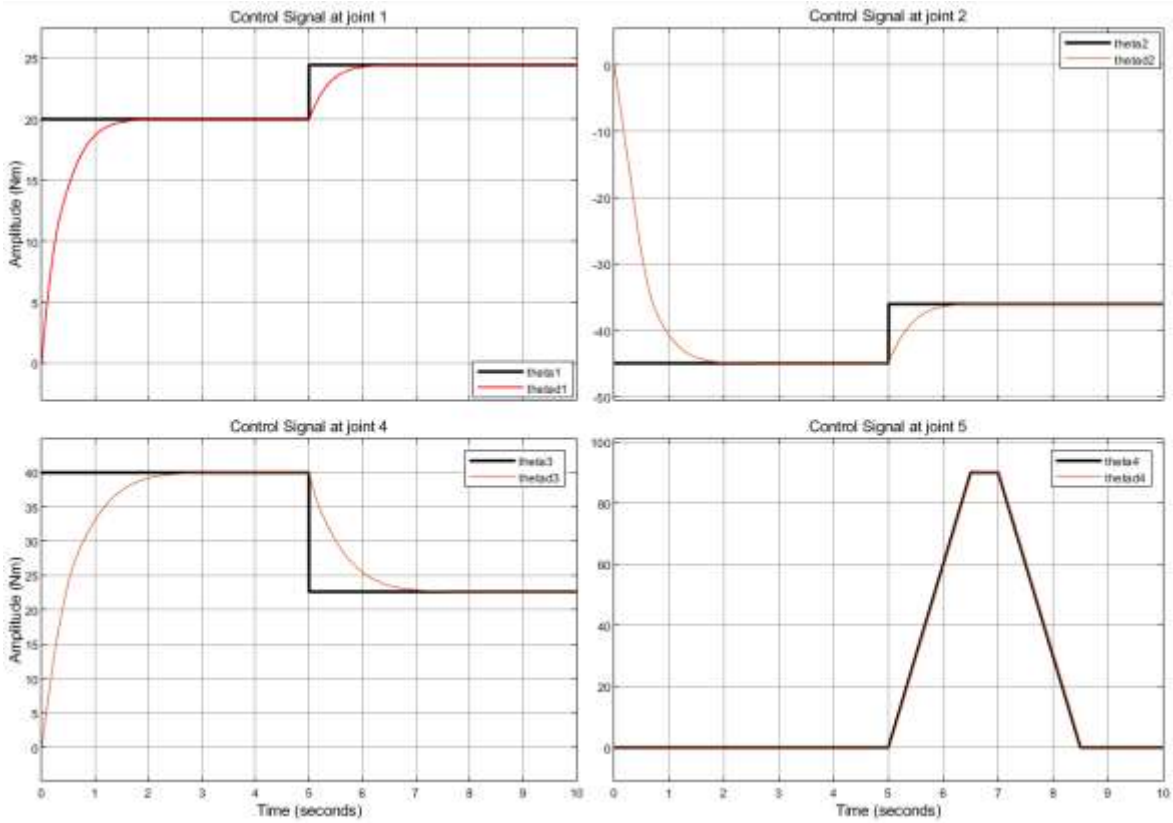
+ Bộ điều khiển:hệ số Kp,Kd được xác định bằng phương pháp thử sai.Qua nhiều lần mô phỏng thử nghiệm,ta chọn được bộ thông số Kp,Kd tối ưu nhất:



Hình 4.20: Bộ thông số Kp, Kd

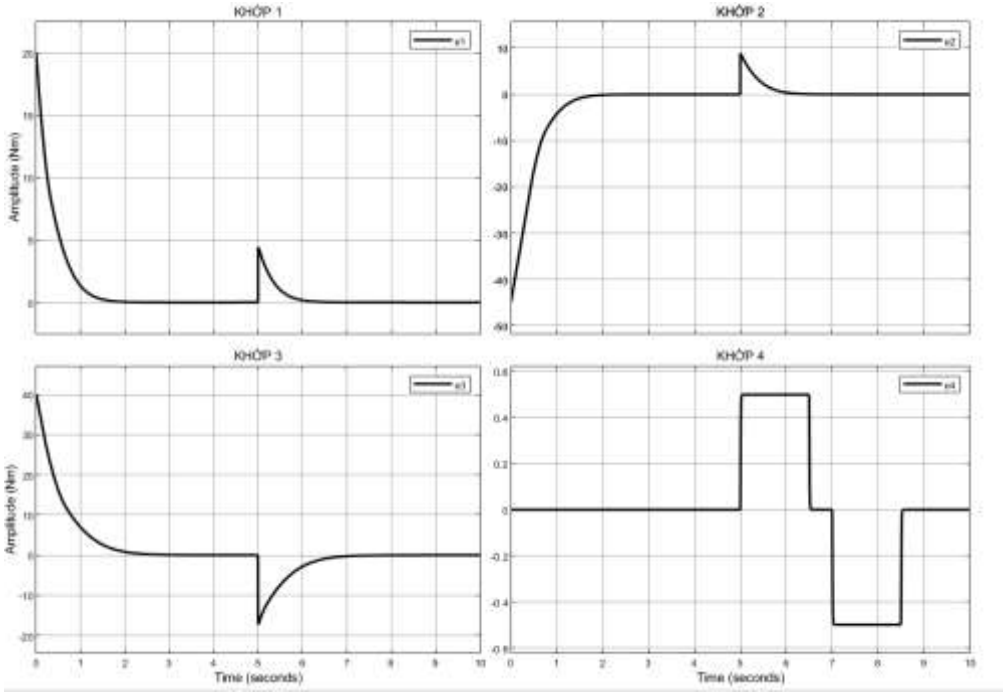
KẾT QUẢ MÔ PHỎNG:

Hình 4.21: Hình ảnh robot robia tới vị trí đặt



Hình 4.22: Hình Giá trị đặt và đáp ứng ngõ ra của hệ thống

Khi vị trí đặt thay đổi thì tín hiệu điều khiển thay đổi nhanh chóng bám theo giá trị đặt.



Hình 4.23: Sai số giữa giá trị đặt và đáp ứng ngõ ra của hệ thống

Bộ điều khiển có thể hiện khả năng bám theo tốt khi có thay đổi, với sai số ban đầu giảm nhanh và tiến về 0

Kết luận: Hệ thống phản hồi nhanh khi sai số lớn, phù hợp với mục tiêu thiết kế bộ điều khiển PD mờ. Sau khi đạt giá trị mong muốn, các khớp (1, 2, 3, 4) ổn định mà không xuất hiện dao động rõ rệt, cho thấy hiệu quả điều khiển. Ngoài ra, hệ thống còn đảm bảo độ chính xác vị trí trong quá trình di chuyển, giảm thiểu thời gian quá độ và sai số trạng thái dừng. Khả năng thích ứng tốt với các nhiễu loạn nhỏ và đặc tính phi tuyến của cơ cấu robot giúp nâng cao độ tin cậy khi vận hành trong môi trường thực tế. Kết quả mô phỏng và thực nghiệm đều xác nhận tính ổn định, độ chính xác và hiệu quả của hệ thống điều khiển đề xuất.

CHƯƠNG 5: THI CÔNG MÔ HÌNH

Ở chương 2 của đề tài,nhóm đã tính chọn thiết bị và linh kiện dựa trên các yêu cầu thực tế đối với một cách tay robot rót bia ứng dụng trong công nghiệp.Các yêu cầu này bao gồm khả năng mang tải,hoạt động ổn định trong môi trường công nghiệp,kích thước phù hợp với không gian làm việc trong nhà hang,quán ăn….,hoạt động liên tục trong thời gian dài,đảm bảo độ tin cậy.Do đó các thiết bị được lựa chọn có công suất phù hợp với thực tế của đề tài,độ bền cơ khí,công suất cao,đáp ứng được các điều kiện trong môi trường công nghiệp.

Tuy nhiên, để phù hợp với phạm vi và điều kiện thực hiện,trong bối cảnh giới hạn về tài nguyên,chi phí và thời gian triển khai,hệ thống được nhóm thực hiện thi công ở mô hình thu nhỏ.Trong mô hình thu nhỏ này vẫn đảm bảo nguyên lý hoạt động và cấu trúc điều khiển cơ bản nhưng có một số hạn chế về khả năng linh hoạt,thời gian đáp ứng,độ trễ và độ bền vật liệu.Nhưng nhìn chung mô hình vẫn đánh giá khách quan về mô hình đề tài.

Tổng quan quy trình thi công mô hình:

5.1. Thiết bị sử dụng

Hình 5.1: Camera USB Dahua 2MP

Hình 5.2: Bộ xử lý AI

Hình 5.3: Arduino Mega 2560

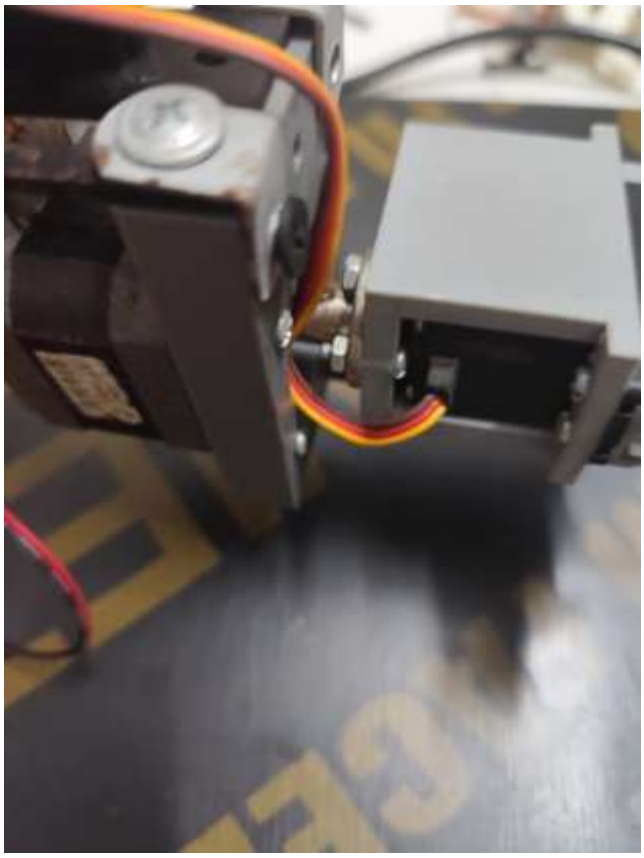
Hình 5.4: Driver điều khiển TB6600

Hình 5.5: Nguồn tổ ong 24V 5A

Hình 5.6: Step Motor

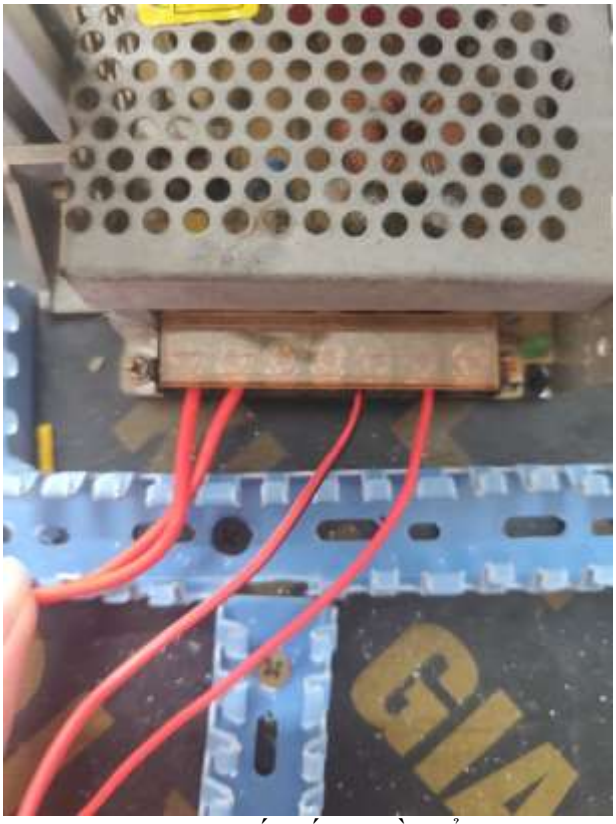
5.2. Chế tạo cánh tay robot 4DOF rót bia

Xây dựng cơ cấu 4 bậc tự do với khớp xoay tròn cuối để thực hiện nhiệm vụ rót bia.



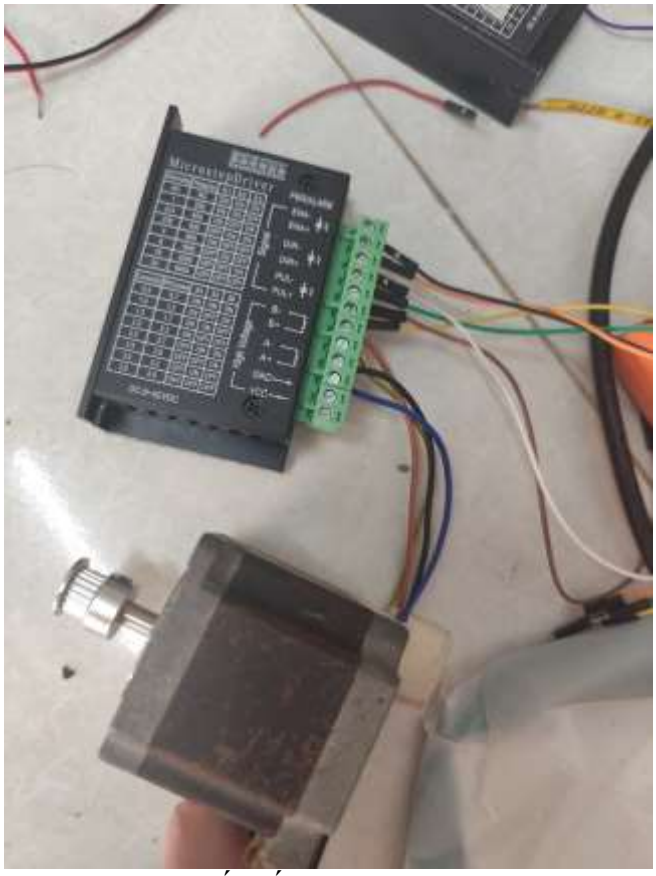
Hình 5.7: Khớp xoay tròn thực hiện nhiệm vụ rót bia

Hình 5.8: Kết nối camera với laptop

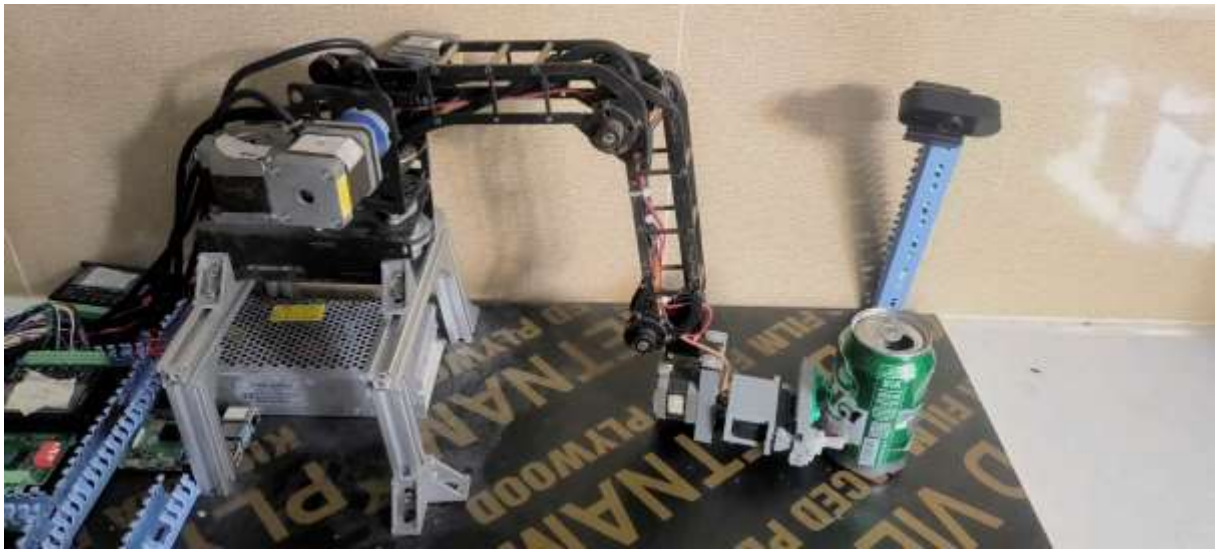


Hình 5.9: Kết nối nguồn tổ ong

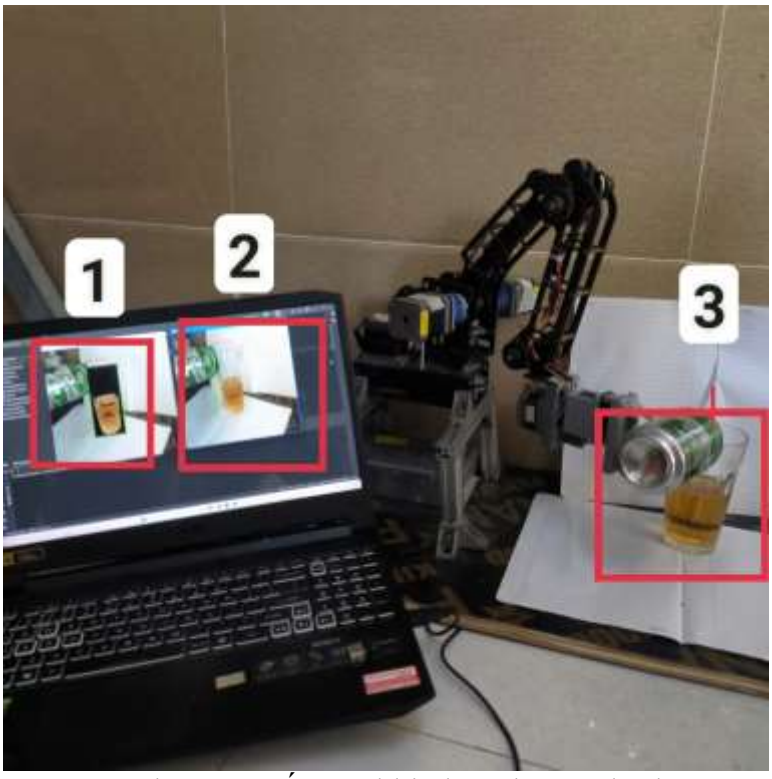
Hình 5.10: Kết laptop và Arduino Mega



Hình 5.11: Kết nối TB6600 và Step motor



Hình 5.12: Mô hình khi hoàn thiện



Hình 5.13: Kết quả khi chạy thử mô hình

Trong đó:

1 là kết quả của mô hình đọc % bia trong ly

2 là kết quả mô hình nhận diện miệng ly và chuyển đổi tọa độ

3 là hình ảnh thực tế của quá trình rót bia

Hình 5.14: Kết quả sẽ sai khi môi trường bị nhiễu

Kết luận:

Mô hình thực tế sau khi được chế tạo và lắp ráp đã cơ bản đáp ứng đúng yêu cầu thiết kế và mục tiêu đề ra. Các thành phần chính như khung cơ khí, động cơ, hệ thống điều khiển và camera nhận diện đều hoạt động ổn định. Cánh tay robot có thể thực hiện các thao tác cần thiết như di chuyển đến vị trí ly, điều chỉnh góc rót và thực hiện thao tác rót bia với độ chính xác tương đối cao. Việc sử dụng động cơ bước kết hợp với driver TB6600 cho phép điều khiển vị trí chính xác, giúp giảm thiểu rung lắc khi rót.

Hệ thống nhận diện hình ảnh sử dụng AI (YOLOv8) hoạt động hiệu quả trong việc phát hiện miệng ly và thân ly. Tọa độ thu được từ xử lý ảnh được chuyển đổi thành tọa độ thực trong không gian làm việc của robot, giúp cánh tay robot định vị chính xác. Ngoài ra, việc sử dụng xử lý ngưỡng màu để phát hiện lượng bia trong ly cũng mang lại kết quả khả quan, giúp hệ thống điều chỉnh lượng rót phù hợp với dung tích còn lại.

Tuy nhiên, trong quá trình vận hành thực tế, vẫn còn một số sai số nhất định do các yếu tố khách quan như: điều kiện ánh sáng môi trường thay đổi, nhiễu từ camera, độ trễ tín hiệu giữa các thiết bị, và giới hạn về cơ khí như độ cứng của khung hoặc sai số lắp ráp. Một số lần robot rót hơi lệch vị trí do sai lệch nhỏ trong chuyển đổi tọa độ hoặc khi ly đặt lệch so với vị trí tiêu chuẩn.

Nhìn chung, mô hình thực tế đã thể hiện được tính khả thi cao và thực hiện được các chức năng cơ bản đúng như mục tiêu đề tài. Kết quả này cho thấy hướng nghiên cứu và giải pháp kỹ thuật đã lựa chọn là hợp lý, có thể được cải tiến thêm để đạt độ chính xác và độ ổn định cao hơn trong các ứng dụng thực tế.

KẾT LUẬN

Sau quá trình nghiên cứu và thực hiện, đề tài "Thiết kế và điều khiển cánh tay robot 4 bậc tự do thực hiện nhiệm vụ nhận diện ly và rót bia tự động" đã hoàn thành các mục tiêu đặt ra ban đầu. Hệ thống đã được xây dựng và vận hành thử nghiệm thành công, đáp ứng các yêu cầu về độ chính xác, độ ổn định và khả năng hoạt động thực tế.

Trong quá trình thực hiện đề tài, nhóm nghiên cứu đã ứng dụng thành công mô hình trí tuệ nhân tạo YOLOv8 trong bài toán xử lý ảnh, giúp nhận diện chính xác vị trí miệng ly và tính toán được lượng bia còn lại trong ly với tốc độ xử lý nhanh, phù hợp cho các ứng dụng thời gian thực. Đồng thời, hệ thống cũng đã thiết kế và điều khiển thành công các động cơ bước thông qua driver TB6600, đảm bảo các khớp của cánh tay robot di chuyển ổn định, chính xác. Bộ điều khiển kết hợp giữa PID và Fuzzy Logic được xây dựng nhằm tối ưu hóa quá trình điều khiển rót bia, hạn chế dao động, nâng cao độ chính xác vị trí và ổn định hệ thống trong suốt quá trình vận hành.

Kết quả đạt được của đề tài cho thấy tiềm năng áp dụng của các giải pháp điều khiển thông minh kết hợp với trí tuệ nhân tạo vào thực tế sản xuất và tự động hóa công nghiệp. Mô hình xây dựng có khả năng mở rộng và phát triển thêm để ứng dụng vào nhiều bài toán tự động hóa khác nhau trong tương lai.

Mặc dù đã đạt được các mục tiêu đề ra, hệ thống vẫn còn tồn tại một số hạn chế nhất định, đặc biệt là sự phụ thuộc vào điều kiện ánh sáng môi trường trong quá trình nhận diện hình ảnh. Trong các nghiên cứu tiếp theo, cần tập trung cải thiện khả năng nhận diện trong điều kiện ánh sáng phức tạp, mở rộng hệ thống nhận diện với đa dạng loại ly, vị trí đặt ly cũng như các loại chất lỏng khác nhằm tăng tính linh hoạt, đa năng cho hệ thống.

Nhìn chung, đề tài đã hoàn thành tốt các nội dung nghiên cứu, thử nghiệm và kiểm chứng, đóng góp một giải pháp khả thi trong việc kết hợp giữa công nghệ AI và điều khiển robot vào các bài toán tự động hóa thực tế.

TÀI LIỆU THAM KHẢO

[1] N. H. Lộc, *Robot công nghiệp – Lập trình và điều khiển*, NXB Khoa học và Kỹ thuật, 2019.

[2] T. V. Hùng, *Cảm biến và xử lý tín hiệu trong hệ thống điều khiển tự động*, NXB Đại học Quốc gia TP. HCM, 2018.

[3] P. V. Hải và L. H. Phúc, *Lý thuyết điều khiển tự động*, NXB Bách khoa Hà Nội, 2017.

[4] N. M. Tân, *Xử lý ảnh số và ứng dụng*, NXB Giáo dục Việt Nam, 2021.

[5] OpenCV Documentation, "OpenCV: Open Source Computer Vision Library," [Online]. Available: https://docs.opencv.org.

[6] Ultralytics Documentation, "YOLOv8: Real-Time Object Detection," [Online]. Available: https://docs.ultralytics.com.

[7] Raspberry Pi Foundation, "Raspberry Pi Documentation," [Online]. Available: https://www.raspberrypi.com/documentation.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[9] A. Karpathy and L. Fei-Fei, "Deep Visual-Semantic Alignments for Generating Image Descriptions," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2015.

[10] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[11] F. Chollet, *Deep Learning with Python*, Manning Publications, 2018.

[12] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2010.

[13] Matplotlib, NumPy, OpenCV-Python Official Documentation. [Online]. Available: https://numpy.org, https://matplotlib.org, https://docs.opencv.org.

[14] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*, 2nd ed., Springer, 2017.

[15] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, Wiley, 2006.

[16] MathWorks, "Robotics System Toolbox Documentation," [Online]. Available: https://www.mathworks.com/help/robotics/.

[17] MathWorks, "Simulink Documentation," [Online]. Available: https://www.mathworks.com/help/simulink/

[18] Dassault Systèmes, “SolidWorks Documentation and Tutorials,” [Online]. Available: https://help.solidworks.com.
[19] J. J. Craig, *Introduction to Robotics: Mechanics and Control*, 4th ed., Pearson, 2017.

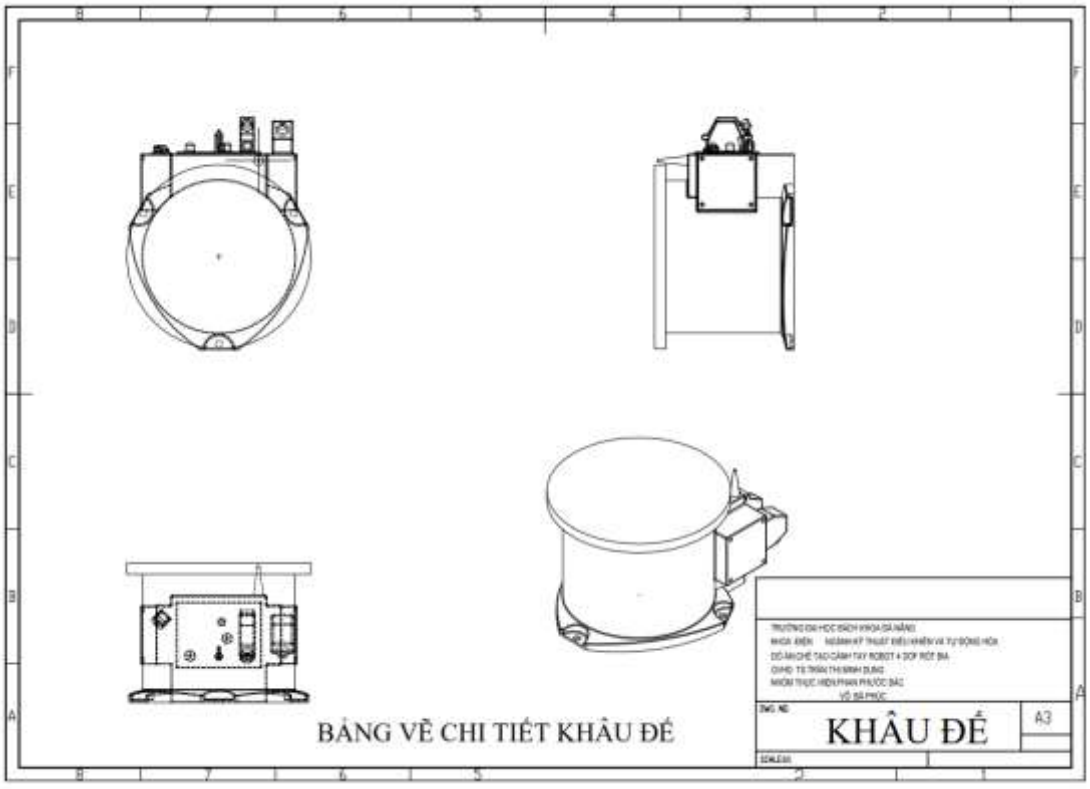
PHỤ LỤC 1

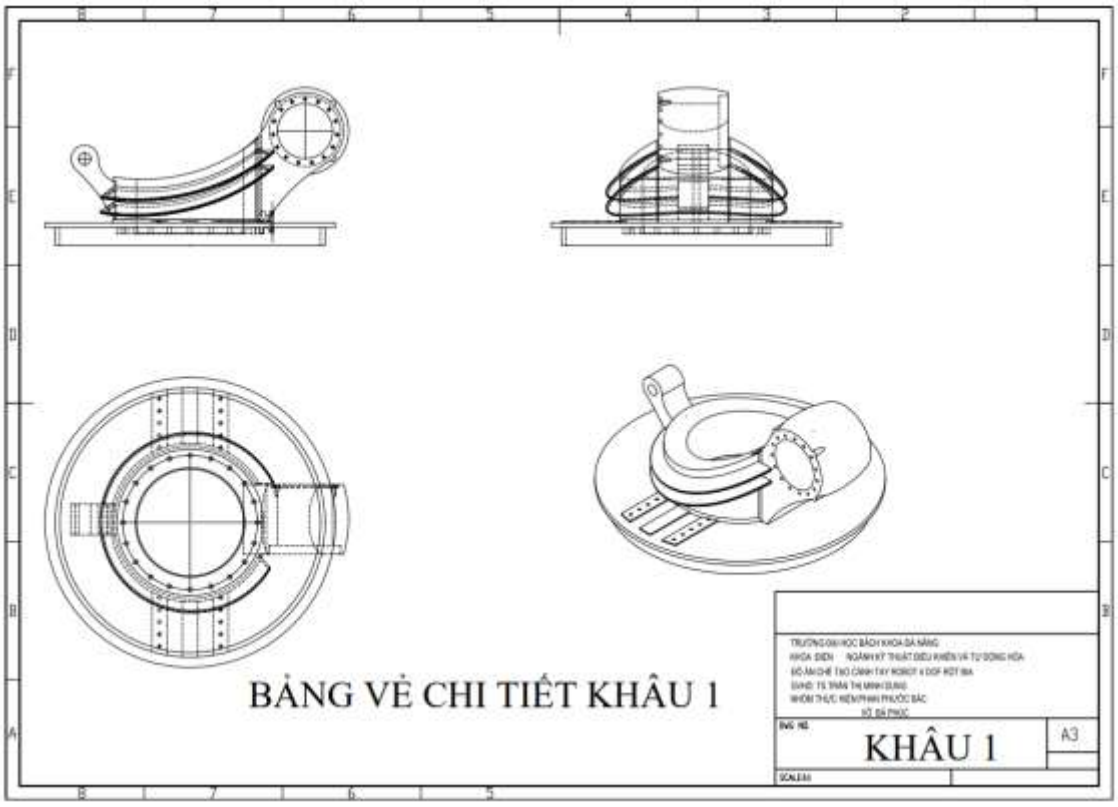
Tên thiết bị	Hình ảnh minh họa	Thông số chính	Giá thành
Nguồn tổ ong 5V 2A		Điện áp đầu ra: 5V DC Dòng điện tối đa: 2A Công suất tối đa: 10W Điện áp đầu vào: 100-240V AC ~ 50/60Hz Kiểu nguồn: Switching Power Supply (nguồn tổ ong) Hiệu suất: ~85% Bảo vệ: Chống quá dòng, quá áp, ngắn mạch	200.000 VNĐ
Driver Servo Delta ASD-A2-0421-L		Công suất: 400W Điện áp: 220V AC, 1 pha Dòng định mức: 2.6A Dòng cực đại: 7.8A Chế độ điều khiển: Vị trí, tốc độ, mô-men xoắn Giao tiếp: EtherCAT, CANopen Hỗ trợ kết nối encoder phản hồi	7.000.000 VNĐ (4)
Máy tính nhúng Raspberry Pi 4 Model B		CPU: Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit, 1.5GHz RAM: 4GB LPDDR4-3200 SDRAM Bộ nhớ lưu trữ: Thẻ nhớ microSD (tùy chọn) Cổng kết nối: 2x USB 3.0, 2x USB 2.0, Ethernet Gigabit, 2x micro-HDMI (4K) WiFi 802.11ac, Bluetooth 5.0 Điện áp cấp: 5V DC (USB-C), dòng tối đa 3A Kích thước: 85.6mm x 56.5mm	2.000.000 VNĐ

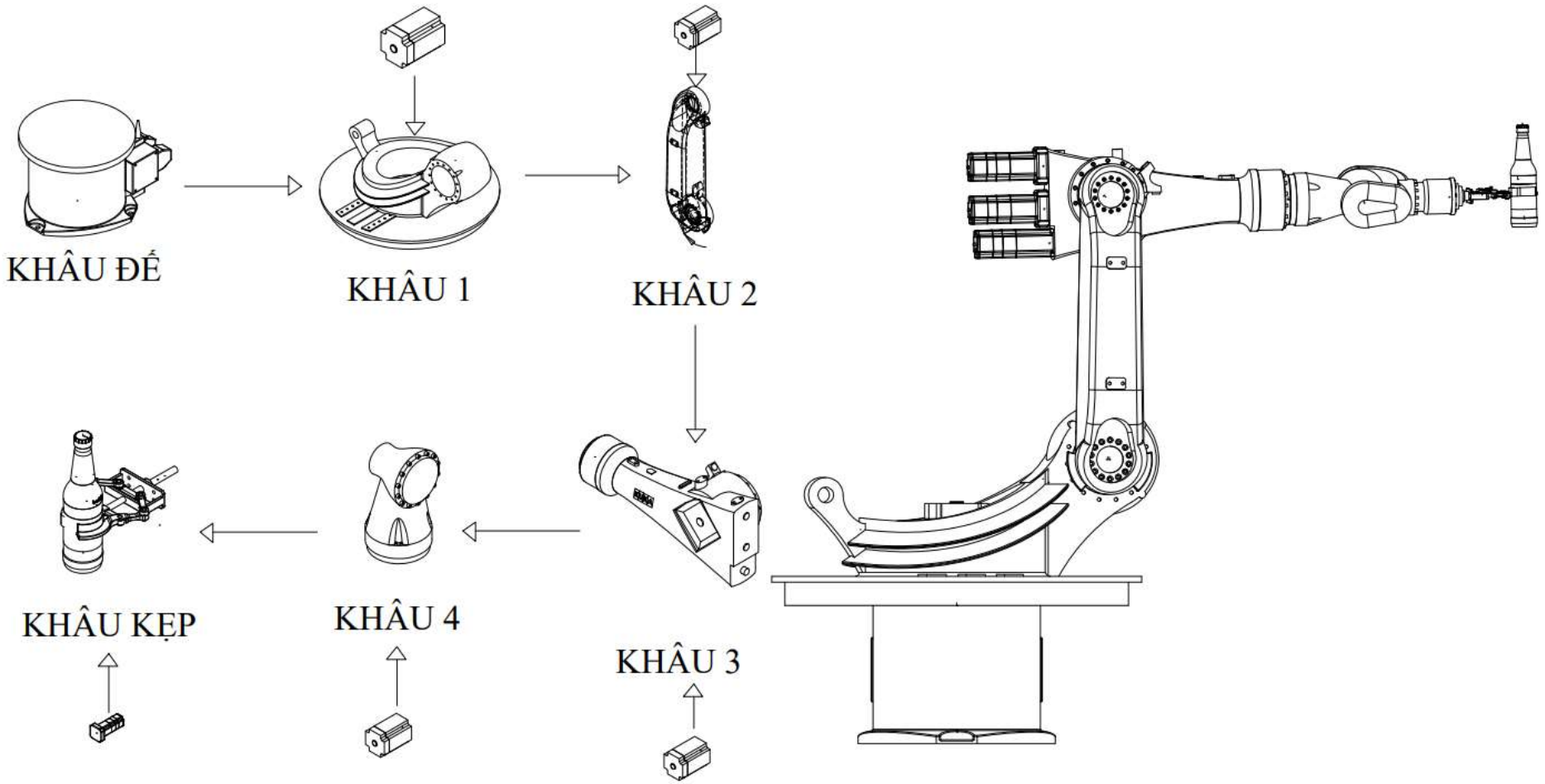
STM32F103C8T6 Blue Pill ARM Cortex-M3		Lõi xử lý: ARM Cortex-M3 Tốc độ xung nhịp: 72 MHz Bộ nhớ Flash: 64 KB RAM: 20 KB Điện áp hoạt động: 2.0V ~ 3.6V Giao tiếp: UART, SPI, I2C, PWM, ADC Số chân I/O: 37 chân I/O khả dụng Đóng gói: LQFP-48	200.000 VNĐ
Động cơ servo AC Delta ECMA-C20604SS		Công suất: 400W Điện áp định mức: 220V AC Dòng điện định mức: 2.6A Dòng điện cực đại: 7.8A Tốc độ định mức: 3000 vòng/phút (RPM) Tốc độ tối đa: 5000 RPM Độ phân giải encoder: 17-bit (131072 xung/vòng) Mô-men xoắn định mức: ~1.27 Nm Chuẩn bảo vệ: IP65 (chống bụi, nước nhẹ)	8.000.000 VNĐ (4)
Công tắc hành trình KW11-N		Điện áp định mức: 250V AC / 5A Điện áp sử dụng trong mạch điều khiển: 5V DC (thường dùng với Arduino) Loại tiếp điểm: 1 NO + 1 NC (thường mở + thường đóng) Độ bền cơ khí: ≥1,000,000 lần đóng/ngắt Kiểu tác động: cần gạt lò xo	10.000(4) VNĐ
Động Cơ RC Servo MG996R		Điện áp hoạt động: 4.8V ~ 7.2V DC Momen xoắn: + 9.4 kg.cm (tại 4.8V) + 11 kg.cm (tại 6V) Góc quay: ~0° đến 180° (tùy vào tín hiệu điều khiển) Tín hiệu điều khiển: PWM (Pulse Width Modulation) Tốc độ quay: ~0.17s/60° (tại 4.8V)	100.000 VNĐ

		Trọng lượng: ~55g Loại bánh răng: Bánh răng kim loại	
Webcam Logitech C922 FULLHD 1080P		Cảm biến: CMOS Full HD Độ phân giải: 1920x1080 (1080p @ 30fps), 720p @ 60fps Góc nhìn: 78° Lấy nét: Tự động Cân bằng sáng: Tự động điều chỉnh ánh sáng yếu Micro: Tích hợp 2 mic đa hướng Kết nối: USB 2.0 (tương thích Raspberry Pi) Nguồn cấp: 5V qua cổng USB	2.400.000 VNĐ
Dây cáp, jack nối			~1.000.000 VNĐ
Chi phí		**66.000.000 VNĐ**	

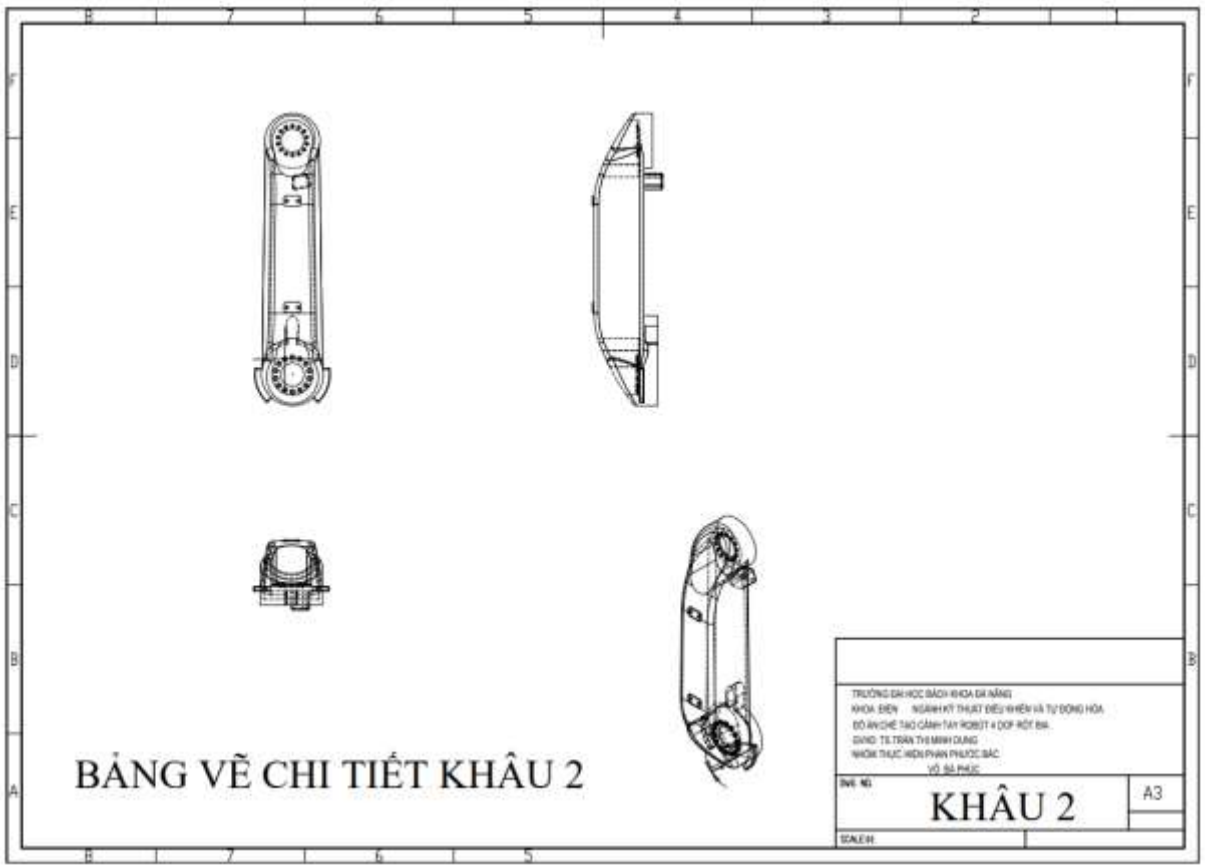
PHỤ LỤC 2

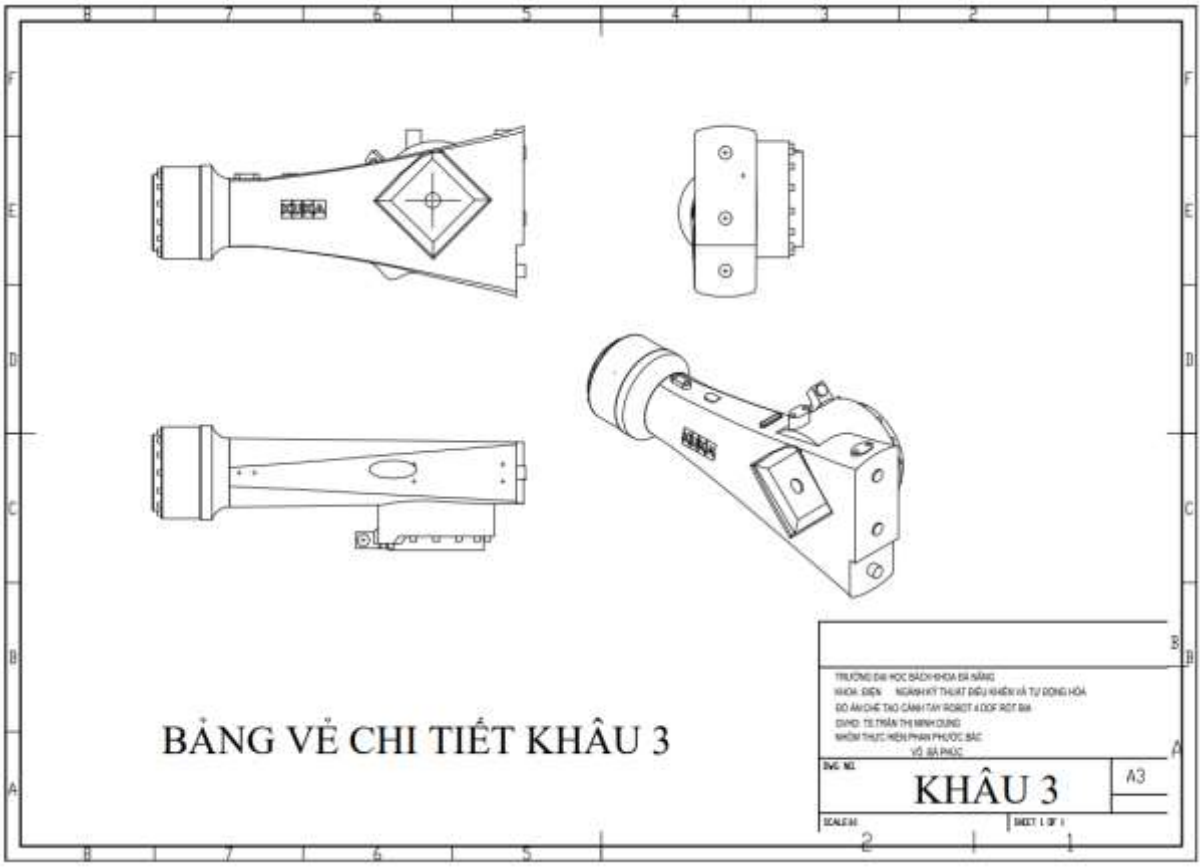

BẢNG VẼ CHI TIẾT KHÂU ĐẾ
KHÂU ĐẾ
A3

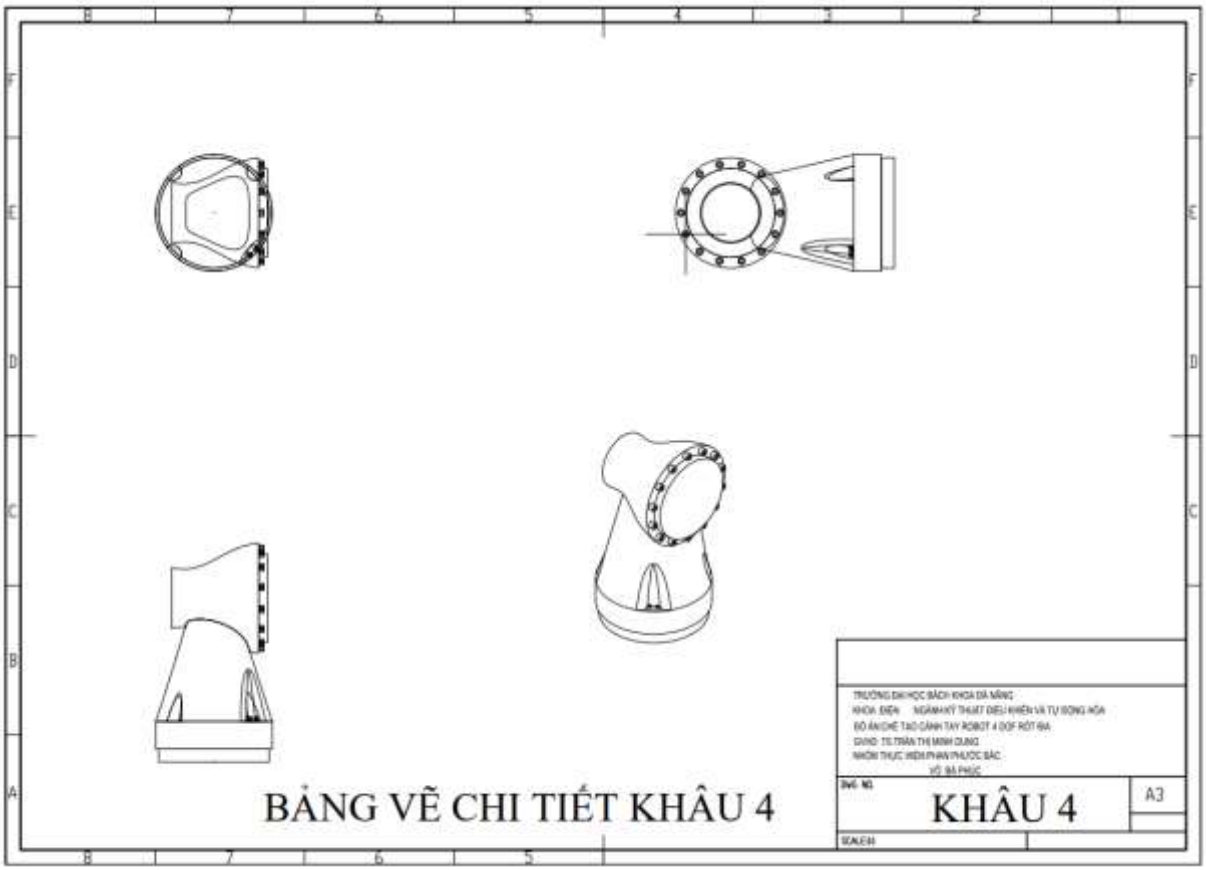

BẢNG VẼ CHI TIẾT KHÂU 1
KHÂU 1
A3

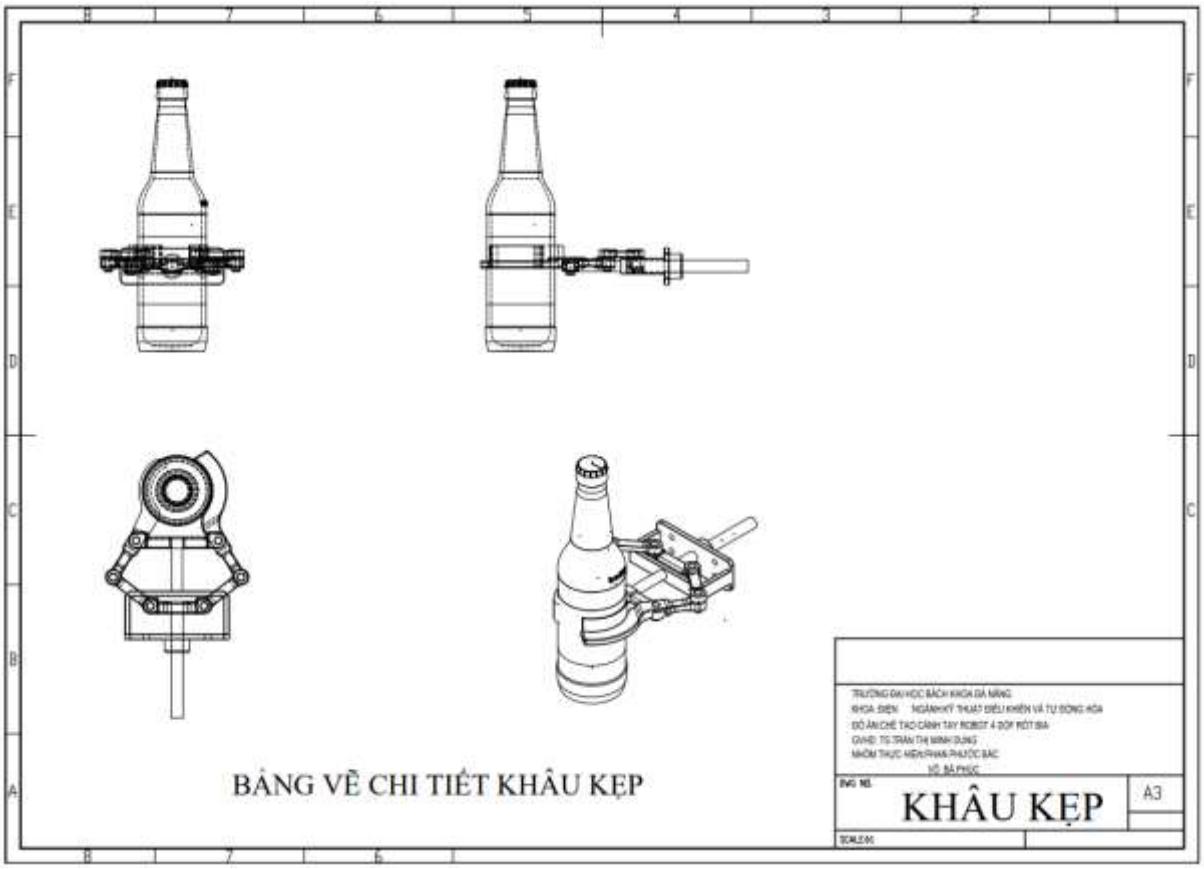


Bản vẽ chi tiết


BẢNG VẼ CHI TIẾT KHÂU 2
TRƯỜNG ĐẠI HỌC BÁCH KHOA ĐÀ NẴNG
KHOA: ĐIỆN NGÀNH KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA
ĐỒ ÁN CHẾ TẠO CÁNH TAY ROBOT 4 DOF RỐT BA
GVHD: TS.TRẦN THỊ MINH DUNG
NHÓM THỰC HIỆN:PHAN PHƯỚC BẮC
VÕ BÁ PHÚC
KHÂU 2
A3


BẢNG VẼ CHI TIẾT KHÂU 3
TRƯỜNG ĐẠI HỌC BÁCH KHOA ĐÀ NẴNG
KHOA: ĐIỆN NGÀNH KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA
ĐỒ ÁN CHẾ TẠO CÁNH TAY ROBOT 4 DOF RỐT BA
GVHD: TS.TRẦN THỊ MINH DUNG
NHÓM THỰC HIỆN:PHAN PHƯỚC BẮC
VÕ BÁ PHÚC
KHÂU 3
A3


TRƯỜNG ĐẠI HỌC BÁCH KHOA ĐÀ NẴNG
KHOA ĐIỆN NGÀNH KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA
ĐỒ ÁN CHẾ TẠO CÁNH TAY ROBOT 4 ĐỘP RÓT BIA
GVHD: TS.TRẦN THỊ MINH DUNG
NHÓM THỰC HIỆN:PHAN PHƯỚC BẮC
VÕ BÁ PHÚC
DWG NO.
KHÂU 4
A3
SCALE:NI
BẢNG VẼ CHI TIẾT KHÂU 4


TRƯỜNG ĐẠI HỌC BÁCH KHOA ĐÀ NẴNG
KHOA ĐIỆN NGÀNH KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA
ĐỒ ÁN CHẾ TẠO CÁNH TAY ROBOT 4 ĐỘP RÓT BIA
GVHD: TS.TRẦN THỊ MINH DUNG
NHÓM THỰC HIỆN:PHAN PHƯỚC BẮC
VÕ BÁ PHÚC
DWG NO.
KHÂU KẸP
A3
SCALE:NI
BẢNG VẼ CHI TIẾT KHÂU KẸP

PHỤ LỤC 3

1.Code Raspberry PI 4
2.Code Arduino điều khiển cánh tay đến tọa độ đặt
3.Code matlab/Simulink

a) **Code khối động học thuận**
b) **Code khối động học nghịch**
c) **Code khối động lực học**

1.Code Raspberry PI 4 :

Code nhận diện ly và mức bia:

```
import cv2
import numpy as np
from ultralytics import YOLO
import os

# Đường dẫn đến mô hình YOLOv8 đã huấn luyện
model_path = r"C:\Users\ACER\OneDrive\Máy
tính\Phantrambia\my_model\train\weights\best.pt"

# Tải mô hình YOLOv8
model = YOLO(model_path)

# Khởi tạo USB camera
cap = cv2.VideoCapture(0)

if not cap.isOpened():
    print("Không thể mở USB camera!")
    exit()

# Hàm chuyển đổi khung màu và làm nổi bật màu vàng
def highlight_yellow(frame):
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
    lower_yellow = np.array([15, 50, 80])
    upper_yellow = np.array([35, 255, 255])
```

```
    mask = cv2.inRange(hsv, lower_yellow, upper_yellow)
    yellow_highlight = cv2.bitwise_and(frame, frame, mask=mask)
    return mask, yellow_highlight

while True:
    ret, frame = cap.read()
    if not ret:
        print("Không thể đọc khung hình từ camera!")
        break

    frame_with_highlight = frame.copy()
    results = model(frame)

    for result in results:
        boxes = result.boxes.xyxy.cpu().numpy()
        confidences = result.boxes.conf.cpu().numpy()
        class_ids = result.boxes.cls.cpu().numpy()

        for box, conf, cls_id in zip(boxes, confidences, class_ids):
            if conf > 0.8:
                x_min, y_min, x_max, y_max = map(int, box)
                bbox_region = frame[y_min:y_max, x_min:x_max]
                mask, yellow_highlight = highlight_yellow(bbox_region)

                contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
                valid_yellow_pixels_y = []
                for contour in contours:
                    area = cv2.contourArea(contour)
                    if area > 200:
                        y_coords = contour[:, :, 1].flatten()
                        valid_yellow_pixels_y.extend(y_coords)

                if len(valid_yellow_pixels_y) > 0:
                    yellow_top = np.min(valid_yellow_pixels_y)
                    yellow_bottom = np.max(valid_yellow_pixels_y)
```

```
            yellow_height = yellow_bottom - yellow_top
        else:
            yellow_height = 0

        bbox_height = y_max - y_min
        three_fourth_height = bbox_height * 0.75
        if three_fourth_height > 0:
            height_ratio = (yellow_height / three_fourth_height) * 100
        else:
            height_ratio = 0

        # Vẽ bounding box
        cv2.rectangle(frame, (x_min, y_min), (x_max, y_max), (0, 255, 0), 2)
        cv2.rectangle(frame_with_highlight, (x_min, y_min), (x_max, y_max), (0,
255, 0), 2)

        # Hiển thị tỷ lệ phần trăm với nhãn "% beer" và màu theo khoảng
        label = f"% beer: {height_ratio:.2f}%"
        if height_ratio <= 30:
            label_color = (0, 0, 255)  # Đỏ
        elif height_ratio <= 60:
            label_color = (0, 165, 255)  # Cam
        elif height_ratio <= 89:
            label_color = (0, 255, 255)  # Vàng
        else:
            label_color = (0, 255, 0)  # Xanh lá
        cv2.putText(frame, label, (x_min, y_min - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.9, label_color, 2)
        cv2.putText(frame_with_highlight, label, (x_min, y_min - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.9, label_color, 2)

        # Vẽ đường thẳng biểu thị chiều cao vùng màu vàng
        if yellow_height > 0:
            yellow_top_global = y_min + yellow_top
            yellow_bottom_global = y_min + yellow_bottom
```

```
                cv2.line(frame, (x_min + 5, yellow_top_global), (x_min + 5,
yellow_bottom_global), (0, 255, 255), 2)
                cv2.line(frame_with_highlight, (x_min + 5, yellow_top_global), (x_min +
5, yellow_bottom_global), (0, 255, 255), 2)

            frame_with_highlight[y_min:y_max, x_min:x_max] = yellow_highlight

    cv2.imshow("Beer Pour Detection", frame)
    cv2.imshow("Yellow Highlighted Region", frame_with_highlight)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

2.Code Arduino điều khiển cánh tay đến tọa độ đặt:

```
#include <AccelStepper.h>

#include <AccelStepper.h>

AccelStepper stepperX(1,2,5);
AccelStepper stepperY(1,3,6);
AccelStepper stepperZ(1,4,7);
class Robot_Control{
  private:
  const float pi = 3.14159;
  float Gear_ratio1 = 4.5;
  float Gear_ratio2 = 4.5;
  float Gear_ratio3 = 4.5;
  float ref_theta1 = -(97.5+1);
  float ref_theta2 = (137-5 );
  float ref_theta3 = -(114.2-5);

  float theta1 = 0, theta2 = 0, theta3 =0;
  int en = 8;
```

```
int dirX = 5;
int stepX = 2;
int endX = 9;
int dirY = 6;
int stepY = 3;
int endY = 10;
int dirZ = 7;
int stepZ = 4;
int endZ =11;
public:
   Robot_Control(int vx, int ax, int vy, int ay, int vz, int az)
{
    pinMode(en, OUTPUT);
    digitalWrite(en, LOW);
    pinMode(stepX, OUTPUT);
    pinMode(dirX, OUTPUT);
    pinMode(stepY, OUTPUT);
    pinMode(dirY, OUTPUT);
    pinMode(stepZ, OUTPUT);
    pinMode(dirZ, OUTPUT);
    stepperX.setMaxSpeed(vx);
    stepperX.setSpeed(vx);
    stepperX.setAcceleration(ax);

    stepperY.setMaxSpeed(vy);
    stepperY.setSpeed(vy);
    stepperY.setAcceleration(ay);

    stepperZ.setMaxSpeed(vz);
    stepperZ.setSpeed(vz);
    stepperZ.setAcceleration(az);

    pinMode(endX, INPUT_PULLUP);
    pinMode(endY, INPUT_PULLUP);
    pinMode(endZ, INPUT_PULLUP);
    stepperX.setCurrentPosition(0);
```

```
    stepperY.setCurrentPosition(0);
    stepperZ.setCurrentPosition(0);
}
void Config( int v,int a){
    stepperX.setMaxSpeed(v);
    stepperX.setSpeed(v);
    stepperX.setAcceleration(a);

    stepperY.setMaxSpeed(v);
    stepperY.setSpeed(v);
    stepperY.setAcceleration(a);

    stepperZ.setMaxSpeed(v);
    stepperZ.setSpeed(v);
    stepperZ.setAcceleration(a);
}
void CalibJ1()
{
  digitalWrite(dirX,HIGH);
  while(digitalRead(endX))
  {
   digitalWrite(stepX, HIGH);
   delayMicroseconds(1000);
   digitalWrite(stepX, LOW);
   delayMicroseconds(1000);
  }
  stepperX.setCurrentPosition(0);
}
void CalibJ2()
{
 digitalWrite(dirY, HIGH);
 while(digitalRead(endY))
 {
  digitalWrite(stepY, HIGH);
  delayMicroseconds(1000);
  digitalWrite(stepY, LOW);
```

```
    delayMicroseconds(1000);
   }
   stepperY.setCurrentPosition(0);
 }
 void CalibJ3()
 {
   digitalWrite(dirZ,LOW);
   while(digitalRead(endZ))
   {
     digitalWrite(stepZ, HIGH);
     delayMicroseconds(1000);
     digitalWrite(stepZ, LOW);
     delayMicroseconds(1000);
   }
   stepperZ.setCurrentPosition(0);
 }
 void Gripper_On(){
 }
 void Gripper_Off(){
 }

 void GetPosition(float theta1, float theta2, float theta3){
       int pulse_1 = Gear_ratio1*(theta1-ref_theta1)*200*16/360;
       int pulse_2 = Gear_ratio2*(theta2-ref_theta2)*200*16/360;
       int pulse_3 = Gear_ratio3*(theta3-ref_theta3)*200*16/360;
       stepperX.moveTo(-pulse_1);
       stepperY.moveTo(pulse_2);
       stepperZ.moveTo(-(pulse_3+pulse_2));

while((stepperX.distanceToGo()!=0)||(stepperY.distanceToGo()!=0)||(stepperZ.distanc
eToGo()!=0))
       {
         stepperX.run();
         stepperY.run();
         stepperZ.run();
        }
```

```
 }
 void Calib_home(){
  CalibJ1();
  CalibJ3();
  CalibJ2();
  GetPosition(0,125,-130);
 }
 void Inverse(float px, float py, float pz, float *theta1, float *theta2, float *theta3){
  float d1 = 9.6, a2 = 13.5, a3 = 14.7, a4 = 5.2, a5 = 6;
  *theta1 = atan2(py,px);
  px = px - a4*cos(*theta1);
  py = py - a4*sin(*theta1);
  pz = pz + a5;
  float r = sqrt(pow(px,2) + pow(py,2) + pow(pz-d1,2));
  *theta3 = -acos((pow(r,2) - pow(a2,2)- pow(a3,2))/(2*a2*a3));
  *theta2 = asin((pz-d1)/r) +  atan(a3*sin(abs(*theta3))/( a2 +
a3*cos(abs(*theta3))));
  *theta1 = round(*theta1*180*100/pi)/100;
  *theta2 = round(*theta2*180*100/pi)/100;
  *theta3 = round(*theta3*180*100/pi)/100;
 }
 void Path_Planning(float point_A[3], float point_B[3],float point_C[3], int type){
  if(type)
  {
   float s = sqrt(pow(point_B[0] - point_A[0],2)+pow(point_B[1] -
point_A[1],2)+pow(point_B[2] - point_A[2],2));
   float kx = (point_B[0]-point_A[0])/s;
   float ky = (point_B[1]-point_A[1])/s;
   float kz = (point_B[2]-point_A[2])/s;
   for (int i=0; i<=30; i++)
   {
    float x = point_A[0] + kx*(s/30)*i;
    float y = point_A[1] + ky*(s/30)*i;
    float z = point_A[2] + kz*(s/30)*i;
    Inverse(x, y, z, &theta1, &theta2, &theta3);
    Serial.println(theta1);
```

```
      Serial.println(theta2);
      Serial.println(theta3);
      GetPosition( theta1, theta2, theta3);
     }
    }
    else
    {
     float a1 = pow(point_B[0]-point_C[0],2) + pow(point_B[1]-point_C[1],2);
     float b1 = pow(point_A[0]-point_C[0],2) + pow(point_A[1]-point_C[1],2);
     float c1 = pow(point_A[0]-point_B[0],2) + pow(point_A[1]-point_B[1],2);
     float x_TS = a1*(b1+c1-a1)*point_A[0] + b1*(a1+c1-b1)*point_B[0] +
c1*(a1+b1-c1)*point_C[0];
     float y_TS = a1*(b1+c1-a1)*point_A[1] + b1*(a1+c1-b1)*point_B[1] +
c1*(a1+b1-c1)*point_C[1];
     float MS = a1*(b1+c1-a1) + b1*(a1+c1-b1) + c1*(a1+b1-c1);
     float x_O = x_TS/MS;
     float y_O = y_TS/MS;
     float R = sqrt(pow(x_O-point_A[0],2) + pow(y_O-point_A[1],2));
     float phi1 = atan2(point_A[1]-y_O, point_A[0]-x_O);
     float phi2 = atan2(point_B[1]-y_O, point_B[0]-x_O);
     float phi3 = atan2(point_C[1]-y_O, point_C[0]-x_O);
     float phi_max = max(phi1,phi2);
     phi_max = max(phi_max,phi3);
     float phi_min = min(phi1,phi2);
     phi_min = min(phi_min,phi3);
     for (int i=0;i<=50;i++)
     {
      float x = R*cos(phi_min+(phi_max - phi_min)*i/50) + x_O;
      float y = R*sin(phi_min+(phi_max - phi_min)*i/50) + y_O;
      float z = 5.2;
      Inverse(x, y, z, &theta1, &theta2, &theta3);
      Serial.println(theta1);
      Serial.println(theta2);
      Serial.println(theta3);
      Serial.println("/////");
      GetPosition(theta1, theta2, theta3);
```

```
    }

  }
 }
 void Classify_Object(int flag){
 switch (flag)
 {
  case 0:
   GetPosition(50,45,-100);
   break;
  case 1:
   GetPosition(65,45,-100);
   break;
  case 2:
   GetPosition(80,45,-100);
   break;
  default:
   break;
 }
 }
};
```

3.Code matlab/Simulink

a) Code khối động học thuận

```
function [P_E,R_E]=FK_4DOF(the1,the2,the3,the4)
%% Thông số của robot/mm
L1=393 ; L2=376.11; L3=657.5; L4=430.43; L5=543.01;

%% Lời giải động học thuận
P_E = [      cosd(the1)*(L2 - L4*sind(the2 + the3) - L5*sind(the2 + the3) +
L3*cosd(the2));
       sind(the1)*(L2 - L4*sind(the2 + the3) - L5*sind(the2 + the3) +
L3*cosd(the2));
       - L4*cosd(the2 + the3) - L5*cosd(the2 + the3) - L3*sind(the2)];

end
```

b) Code khối động học nghịch

```
function[t1_1,t2_1,t3_1,R]= IK_DOF(Px, Py, Pz,r)
%% Thông số của ROBOT/đơn vị mm
L1=393 ; L2=376.11; L3=657.5; L4=430.43; L5=543.01;
R = r
```

```
%% Bộ nghiệm số 1
%tim theta1 > 0
t1_1=atan2d(Py,Px);
%tim theta3 > 0
a=(Px*cosd(t1_1)+Py*sind(t1_1)-L2)^2+Pz^2-L3^2-(L5+L4)^2;
s3_1=-a/(2*L3*(L4+L5));
c3_1=sqrt(1-s3_1^2);
t3_1=atan2d(s3_1,c3_1);
%tim theta2 > 0
b=Px*cosd(t1_1)+Py*sind(t1_1)-L2;
c=Pz;
c2_1=(b*(L3-(L5+L4)*sind(t3_1))-c*(L4+L5)*cosd(t3_1))/((L3-
(L4+L5)*sind(t3_1))^2+(L4+L5)^2*cosd(t3_1)^2);
s2_1=(-c*(L3-(L4+L5)*sind(t3_1))-b*(L4+L5)*cosd(t3_1))/((L3-
(L4+L5)*sind(t3_1))^2+(L4+L5)^2*cosd(t3_1)^2);
t2_1=atan2d(s2_1,c2_1);
```

c) Code khối động lực học

```
function dde = Dynamics(e, de, Tor)

%%
e1=e(1); e2=e(2); e3=e(3); e4=e(4);
de1=de(1); de2=de(2); de3=de(3); de4=de(4);
%% thông số
m1=2;m2=2; m3=2;m4=2;%%kg
L1=1; L2=1;L3=1;L4=1;
g =-9.8;
b1=0.1; b2=0.1; b3=0.1; b4=0.1;
c1=1E-6; c2=1E-6; c3=1E-6; c4=1E-6;
b=[b1 0 0 0;
   0 b2 0 0;
   0 0 b3 0;
   0 0 0 b4];
c=[c1 0 0 0;
   0 c2 0 0;
   0 0 c3 0;
   0 0 0 c4];
M =[ (L2^2*m2)/8 + (L2^2*m3)/2 + (L2^2*m4)/2 + (L3^2*m3)/8 + (L3^2*m4)/2 +
(3*L4^2*m4)/16 - (L4^2*m4*cos(2*e2 + 2*e3 - 2*e4))/32 - (L4^2*m4*cos(2*e2 + 2*e3 +
2*e4))/32 + (L2^2*m2*cos(2*e2))/8 + (L2^2*m3*cos(2*e2))/2 + (L2^2*m4*cos(2*e2))/2
- (L4^2*m4*cos(2*e4))/16 - (L3^2*m3*cos(2*e2 + 2*e3))/8 + (L3^2*m4*cos(2*e2 +
2*e3))/2 - (L4^2*m4*cos(2*e2 + 2*e3))/16 - (L2*L4*m4*sin(e3 + e4))/4 -
(L2*L4*m4*sin(2*e2 + e3 - e4))/4 - (L3*L4*m4*sin(2*e2 + 2*e3 + e4))/4 +
L2*L3*m4*cos(e3) - (L2*L3*m3*sin(e3))/2 + L2*L3*m4*cos(2*e2 + e3) -
(L2*L3*m3*sin(2*e2 + e3))/2 - (L2*L4*m4*sin(e3 - e4))/4 - (L3*L4*m4*sin(2*e2 +
2*e3 - e4))/4 - (L2*L4*m4*sin(2*e2 + e3 + e4))/4, -(L4*m4*(4*L2*sin(e2)*sin(e4) +
4*L3*cos(e2)*sin(e3)*sin(e4) + 4*L3*cos(e3)*sin(e2)*sin(e4) +
L4*sin(2*e4)*cos(e2)*cos(e3) - L4*sin(2*e4)*sin(e2)*sin(e3)))/8, -
(L4*m4*(4*L3*cos(e2)*sin(e3)*sin(e4) + 4*L3*cos(e3)*sin(e2)*sin(e4) +
L4*sin(2*e4)*cos(e2)*cos(e3) - L4*sin(2*e4)*sin(e2)*sin(e3)))/8,
(L4*m4*(L4*cos(e2)*sin(e3) - 2*L2*cos(e2)*cos(e4) + L4*cos(e3)*sin(e2) -
2*L3*cos(e2)*cos(e3)*cos(e4) + 2*L3*cos(e4)*sin(e2)*sin(e3)))/4;

-(L4*m4*(4*L2*sin(e2)*sin(e4) + 4*L3*cos(e2)*sin(e3)*sin(e4) +
4*L3*cos(e3)*sin(e2)*sin(e4) + L4*sin(2*e4)*cos(e2)*cos(e3) -
L4*sin(2*e4)*sin(e2)*sin(e3)))/8,            (L2^2*m2)/4 + L2^2*m3 + L2^2*m4 +
(L3^2*m3)/4 + L3^2*m4 + (L4^2*m4*cos(e4)^2)/4 + 2*L2*L3*m4*cos(e3) -
L2*L3*m3*sin(e3) - L2*L4*m4*cos(e4)*sin(e3),                (L3^2*m3)/4 + L3^2*m4
+ (L4^2*m4*cos(e4)^2)/4 + L2*L3*m4*cos(e3) - (L2*L3*m3*sin(e3))/2 -
```

```
(L2*L4*m4*cos(e4)*sin(e3))/2,
-(L4*m4*sin(e4)*(L3 + L2*cos(e3)))/2;

-(L4*m4*(4*L3*cos(e2)*sin(e3)*sin(e4) + 4*L3*cos(e3)*sin(e2)*sin(e4) +
L4*sin(2*e4)*cos(e2)*cos(e3) - L4*sin(2*e4)*sin(e2)*sin(e3)))/8,
(L3^2*m3)/4 + L3^2*m4 + (L4^2*m4*cos(e4)^2)/4 + L2*L3*m4*cos(e3) -
(L2*L3*m3*sin(e3))/2 - (L2*L4*m4*cos(e4)*sin(e3))/2,
(L3^2*m3)/4 + L3^2*m4 + (L4^2*m4)/8 + (L4^2*m4*cos(2*e4))/8,
-(L3*L4*m4*sin(e4))/2;

(L4*m4*(L4*cos(e2)*sin(e3) - 2*L2*cos(e2)*cos(e4) + L4*cos(e3)*sin(e2) -
2*L3*cos(e2)*cos(e3)*cos(e4) + 2*L3*cos(e4)*sin(e2)*sin(e3)))/4,
-(L4*m4*sin(e4)*(L3 + L2*cos(e3)))/2,
-(L3*L4*m4*sin(e4))/2,
(L4^2*m4)/4];

G =[(g*m3*cos(e1)*(L3*sin(e2 + e3) - 2*L2*cos(e2)))/2 - g*m4*(cos(e1)*(L3*cos(e2 +
e3) + L2*cos(e2)) - (L4*(cos(e1)*cos(e2)*cos(e4)*sin(e3) - sin(e1)*sin(e4) +
cos(e1)*cos(e3)*cos(e4)*sin(e2)))/2) - (L2*g*m2*cos(e1)*cos(e2))/2;
                                                    (g*sin(e1)*(L3*m3*cos(e2 + e3) +
2*L3*m4*sin(e2 + e3) + L2*m2*sin(e2) + 2*L2*m3*sin(e2) + 2*L2*m4*sin(e2) +
L4*m4*cos(e2)*cos(e3)*cos(e4) - L4*m4*cos(e4)*sin(e2)*sin(e3)))/2;

g*m4*((L4*(cos(e2)*cos(e3)*cos(e4)*sin(e1) - cos(e4)*sin(e1)*sin(e2)*sin(e3)))/2 +
L3*sin(e2 + e3)*sin(e1)) + (L3*g*m3*cos(e2 + e3)*sin(e1))/2;

-(L4*g*m4*(cos(e2)*sin(e1)*sin(e3)*sin(e4) - cos(e1)*cos(e4) +
cos(e3)*sin(e1)*sin(e2)*sin(e4)))/2];

C =[ (L4^2*de4*m4*sin(2*e4))/8 - L2^2*de2*m3*sin(2*e2) - L2^2*de2*m4*sin(2*e2) -
(L2^2*de2*m2*sin(2*e2))/4 + (L3^2*de2*m3*sin(2*e2 + 2*e3))/4 -
L3^2*de2*m4*sin(2*e2 + 2*e3) + (L3^2*de3*m3*sin(2*e2 + 2*e3))/4 -
L3^2*de3*m4*sin(2*e2 + 2*e3) + (L4^2*de2*m4*sin(2*e2 + 2*e3))/8 +
(L4^2*de3*m4*sin(2*e2 + 2*e3))/8 + (L4^2*de2*m4*sin(2*e2 + 2*e3 - 2*e4))/16 +
(L4^2*de2*m4*sin(2*e2 + 2*e3 + 2*e4))/16 + (L4^2*de3*m4*sin(2*e2 + 2*e3 -
2*e4))/16 + (L4^2*de3*m4*sin(2*e2 + 2*e3 + 2*e4))/16 - (L4^2*de4*m4*sin(2*e2 +
2*e3 - 2*e4))/16 + (L4^2*de4*m4*sin(2*e2 + 2*e3 + 2*e4))/16 -
(L2*L4*de2*m4*cos(2*e2 + e3 + e4))/2 - (L2*L4*de3*m4*cos(2*e2 + e3 + e4))/4 -
(L2*L4*de4*m4*cos(2*e2 + e3 + e4))/4 - (L2*L4*de3*m4*cos(e3 + e4))/4 -
(L2*L4*de4*m4*cos(e3 + e4))/4 - (L2*L4*de2*m4*cos(2*e2 + e3 - e4))/2 -
(L2*L4*de3*m4*cos(2*e2 + e3 - e4))/4 + (L2*L4*de4*m4*cos(2*e2 + e3 - e4))/4 -
(L3*L4*de2*m4*cos(2*e2 + 2*e3 + e4))/2 - (L3*L4*de3*m4*cos(2*e2 + 2*e3 + e4))/2 -
(L3*L4*de4*m4*cos(2*e2 + 2*e3 + e4))/4 - (L2*L3*de3*m3*cos(e3))/2 -
L2*L3*de3*m4*sin(e3) - L2*L3*de2*m3*cos(2*e2 + e3) - (L2*L3*de3*m3*cos(2*e2 +
e3))/2 - (L2*L4*de3*m4*cos(e3 - e4))/4 + (L2*L4*de4*m4*cos(e3 - e4))/4 -
2*L2*L3*de2*m4*sin(2*e2 + e3) - L2*L3*de3*m4*sin(2*e2 + e3) -
(L3*L4*de2*m4*cos(2*e2 + 2*e3 - e4))/2 - (L3*L4*de3*m4*cos(2*e2 + 2*e3 - e4))/2 +
(L3*L4*de4*m4*cos(2*e2 + 2*e3 - e4))/4, (L4*de3*m4*(4*L3*sin(e2)*sin(e3)*sin(e4) -
4*L3*cos(e2)*cos(e3)*sin(e4) + L4*sin(2*e4)*cos(e2)*sin(e3) +
L4*sin(2*e4)*cos(e3)*sin(e2)))/8 + (L4*de2*m4*(4*L3*sin(e2)*sin(e3)*sin(e4) -
4*L3*cos(e2)*cos(e3)*sin(e4) - 4*L2*cos(e2)*sin(e4) + L4*sin(2*e4)*cos(e2)*sin(e3)
+ L4*sin(2*e4)*cos(e3)*sin(e2)))/8 - (L4*de4*m4*(4*L2*cos(e4)*sin(e2) +
4*L3*cos(e2)*cos(e4)*sin(e3) + 4*L3*cos(e3)*cos(e4)*sin(e2) +
2*L4*cos(e2)*cos(e3)*(2*cos(e4)^2 - 1) - 2*L4*sin(e2)*sin(e3)*(2*cos(e4)^2 -
1)))/8, (L4*de2*m4*(4*L3*sin(e2)*sin(e3)*sin(e4) - 4*L3*cos(e2)*cos(e3)*sin(e4) +
L4*sin(2*e4)*cos(e2)*sin(e3) + L4*sin(2*e4)*cos(e3)*sin(e2)))/8 +
(L4*de3*m4*(4*L3*sin(e2)*sin(e3)*sin(e4) - 4*L3*cos(e2)*cos(e3)*sin(e4) +
L4*sin(2*e4)*cos(e2)*sin(e3) + L4*sin(2*e4)*cos(e3)*sin(e2)))/8 -
(L4*de4*m4*(2*L3*sin(e2 + e3 - e4) + 2*L3*sin(e2 + e3 + e4) +
```

```
2*L4*cos(2*e4)*cos(e2 + e3)))/8, (L4*de3*m4*(L3*sin(e2 + e3 - e4) + L4*cos(e2 +
e3) + L3*sin(e2 + e3 + e4)))/4 + (L4*de2*m4*(L4*cos(e2)*cos(e3) +
2*L2*cos(e4)*sin(e2) - L4*sin(e2)*sin(e3) + 2*L3*cos(e2)*cos(e4)*sin(e3) +
2*L3*cos(e3)*cos(e4)*sin(e2)))/4 + (L4*de4*m4*sin(e4)*(L3*cos(e2 + e3) +
L2*cos(e2)))/2;

(L2^2*de1*m2*sin(2*e2))/8 + (L2^2*de1*m3*sin(2*e2))/2 + (L2^2*de1*m4*sin(2*e2))/2
+ (L4^2*de2*m4*cos(e2 + e3 - 2*e4))/32 - (L4^2*de2*m4*cos(e2 + e3 + 2*e4))/32 +
(L4^2*de3*m4*cos(e2 + e3 - 2*e4))/32 - (L4^2*de3*m4*cos(e2 + e3 + 2*e4))/32 -
(L4^2*de4*m4*cos(e2 + e3 - 2*e4))/8 - (L4^2*de4*m4*cos(e2 + e3 + 2*e4))/8 -
(L3^2*de1*m3*sin(2*e2 + 2*e3))/8 + (L3^2*de1*m4*sin(2*e2 + 2*e3))/2 -
(L4^2*de1*m4*sin(2*e2 + 2*e3))/16 - (L4^2*de4*m4*cos(e2 + e3))/8 -
(L4^2*de1*m4*sin(2*e2 + 2*e3 - 2*e4))/32 - (L4^2*de1*m4*sin(2*e2 + 2*e3 +
2*e4))/32 + (L2*L4*de1*m4*cos(2*e2 + e3 + e4))/4 + (L3*L4*de2*m4*sin(e2 + e3 -
e4))/8 + (L3*L4*de3*m4*sin(e2 + e3 - e4))/8 - (3*L3*L4*de4*m4*sin(e2 + e3 - e4))/8
- (L2*L4*de2*m4*sin(e2 + e4))/8 - (3*L2*L4*de4*m4*sin(e2 + e4))/8 +
(L2*L4*de1*m4*cos(2*e2 + e3 - e4))/4 + (L3*L4*de1*m4*cos(2*e2 + 2*e3 + e4))/4 +
(L2*L3*de1*m3*cos(2*e2 + e3))/2 + L2*L3*de1*m4*sin(2*e2 + e3) +
(L2*L4*de2*m4*sin(e2 - e4))/8 - (3*L2*L4*de4*m4*sin(e2 - e4))/8 -
(L3*L4*de2*m4*sin(e2 + e3 + e4))/8 - (L3*L4*de3*m4*sin(e2 + e3 + e4))/8 -
(3*L3*L4*de4*m4*sin(e2 + e3 + e4))/8 + (L3*L4*de1*m4*cos(2*e2 + 2*e3 - e4))/4,
- L2*de3*(L3*m3*cos(e3) + 2*L3*m4*sin(e3) + L4*m4*cos(e3)*cos(e4)) -
(L4*de1*m4*(4*L3*sin(e2)*sin(e3)*sin(e4) - 4*L3*cos(e2)*cos(e3)*sin(e4) -
4*L2*cos(e2)*sin(e4) + L4*sin(2*e4)*cos(e2)*sin(e3) +
L4*sin(2*e4)*cos(e3)*sin(e2)))/16 - (L4*de4*m4*sin(e4)*(L4*cos(e4) -
2*L2*sin(e3)))/2,
- (L2*de3*(L3*m3*cos(e3) + 2*L3*m4*sin(e3) + L4*m4*cos(e3)*cos(e4)))/2 -
(L4*de1*m4*(4*L3*sin(e2)*sin(e3)*sin(e4) - 4*L3*cos(e2)*cos(e3)*sin(e4) +
L4*sin(2*e4)*cos(e2)*sin(e3) + L4*sin(2*e4)*cos(e3)*sin(e2)))/16 -
(L4*de4*m4*sin(e4)*(L4*cos(e4) - L2*sin(e3)))/2,
(L2*L4*de3*m4*sin(e3)*sin(e4))/2 - (L4*de4*m4*cos(e4)*(L3 + L2*cos(e3)))/2 -
(L4*de1*m4*(L4*cos(e2)*cos(e3) + 2*L2*cos(e4)*sin(e2) - L4*sin(e2)*sin(e3) +
2*L3*cos(e2)*cos(e4)*sin(e3) + 2*L3*cos(e3)*cos(e4)*sin(e2)))/8;
(L4^2*de2*m4*cos(e2 + e3 - 2*e4))/32 - (L4^2*de2*m4*cos(e2 + e3 + 2*e4))/32 +
(L4^2*de3*m4*cos(e2 + e3 - 2*e4))/32 - (L4^2*de3*m4*cos(e2 + e3 + 2*e4))/32 -
(L4^2*de4*m4*cos(e2 + e3 - 2*e4))/8 - (L4^2*de4*m4*cos(e2 + e3 + 2*e4))/8 -
(L3^2*de1*m3*sin(2*e2 + 2*e3))/8 + (L3^2*de1*m4*sin(2*e2 + 2*e3))/2 -
(L4^2*de1*m4*sin(2*e2 + 2*e3))/16 - (L4^2*de4*m4*cos(e2 + e3))/8 -
(L4^2*de1*m4*sin(2*e2 + 2*e3 - 2*e4))/32 - (L4^2*de1*m4*sin(2*e2 + 2*e3 +
2*e4))/32 + (L2*L4*de1*m4*cos(2*e2 + e3 + e4))/8 + (L3*L4*de2*m4*sin(e2 + e3 -
e4))/8 + (L3*L4*de3*m4*sin(e2 + e3 - e4))/8 - (3*L3*L4*de4*m4*sin(e2 + e3 - e4))/8
+ (L2*L4*de1*m4*cos(e3 + e4))/8 + (L2*L4*de1*m4*cos(2*e2 + e3 - e4))/8 +
(L3*L4*de1*m4*cos(2*e2 + 2*e3 + e4))/4 + (L2*L3*de1*m3*cos(e3))/4 +
(L2*L3*de1*m4*sin(e3))/2 + (L2*L3*de1*m3*cos(2*e2 + e3))/4 + (L2*L4*de1*m4*cos(e3
- e4))/8 + (L2*L3*de1*m4*sin(2*e2 + e3))/2 - (L3*L4*de2*m4*sin(e2 + e3 + e4))/8 -
(L3*L4*de3*m4*sin(e2 + e3 + e4))/8 - (3*L3*L4*de4*m4*sin(e2 + e3 + e4))/8 +
(L3*L4*de1*m4*cos(2*e2 + 2*e3 - e4))/4,
(L2*L3*de2*m3*cos(e3))/2 - (L4^2*de4*m4*sin(2*e4))/4 - (L2*L3*de3*m3*cos(e3))/4 +
L2*L3*de2*m4*sin(e3) - (L2*L3*de3*m4*sin(e3))/2 + (L2*L4*de2*m4*cos(e3)*cos(e4))/2
- (L2*L4*de3*m4*cos(e3)*cos(e4))/4 + (L2*L4*de4*m4*sin(e3)*sin(e4))/4 -
(L4^2*de1*m4*cos(e2)*cos(e4)*sin(e3)*sin(e4))/8 -
(L4^2*de1*m4*cos(e3)*cos(e4)*sin(e2)*sin(e4))/8 +
(L3*L4*de1*m4*cos(e2)*cos(e3)*sin(e4))/4 -
(L3*L4*de1*m4*sin(e2)*sin(e3)*sin(e4))/4,
(L2*de2*(L3*m3*cos(e3) + 2*L3*m4*sin(e3) + L4*m4*cos(e3)*cos(e4)))/4 -
(L4^2*de4*m4*sin(2*e4))/4 - (L4*de1*m4*(4*L3*sin(e2)*sin(e3)*sin(e4) -
4*L3*cos(e2)*cos(e3)*sin(e4) + L4*sin(2*e4)*cos(e2)*sin(e3) +
L4*sin(2*e4)*cos(e3)*sin(e2)))/16,
- (L4*de1*m4*(L3*sin(e2 + e3 - e4) + L4*cos(e2 + e3) + L3*sin(e2 + e3 + e4)))/8 -
```

```
(L3*L4*de4*m4*cos(e4))/2 -(L2*L4*de2*m4*sin(e3)*sin(e4))/4;
-(L4*m4*(L4*de1*sin(2*e4) - 2*L4*de2*cos(e2)*cos(e3) - 4*L4*de3*cos(e2)*cos(e3) -
12*L2*de2*cos(e4)*sin(e2) - 4*L2*de4*cos(e2)*sin(e4) + 4*L2*de1*sin(e3)*sin(e4) +
2*L4*de2*sin(e2)*sin(e3) + 4*L4*de3*sin(e2)*sin(e3) +
4*L3*de1*cos(2*e2)*sin(2*e3)*sin(e4) + 4*L3*de1*cos(2*e3)*sin(2*e2)*sin(e4) -
12*L3*de2*cos(e2)*cos(e4)*sin(e3) - 12*L3*de2*cos(e3)*cos(e4)*sin(e2) -
12*L3*de3*cos(e2)*cos(e4)*sin(e3) - 12*L3*de3*cos(e3)*cos(e4)*sin(e2) -
4*L3*de4*cos(e2)*cos(e3)*sin(e4) + L4*de1*cos(2*e2)*cos(2*e3)*sin(2*e4) +
4*L3*de4*sin(e2)*sin(e3)*sin(e4) - L4*de1*sin(2*e2)*sin(2*e3)*sin(2*e4) -
2*L4*de3*cos(2*e4)*cos(e2)*cos(e3) - 4*L4*de2*cos(e2)*cos(e3)*cos(e4)^2 +
4*L2*de1*cos(2*e2)*sin(e3)*sin(e4) + 4*L2*de1*sin(2*e2)*cos(e3)*sin(e4) +
2*L4*de3*cos(2*e4)*sin(e2)*sin(e3) + 4*L4*de2*cos(e4)^2*sin(e2)*sin(e3)))/16,
(L4*de1*m4*(4*L2*cos(e4)*sin(e2) + 4*L3*cos(e2)*cos(e4)*sin(e3) +
4*L3*cos(e3)*cos(e4)*sin(e2) + 2*L4*cos(e2)*cos(e3)*(2*cos(e4)^2 - 1) -
2*L4*sin(e2)*sin(e3)*(2*cos(e4)^2 - 1)))/16 - (L4*de4*m4*cos(e4)*(L3 +
L2*cos(e3)))/4 + (L4*de2*m4*sin(e4)*(L4*cos(e4) - 2*L2*sin(e3)))/4 +
(L4*de3*m4*sin(e4)*(L4*cos(e4) - L2*sin(e3)))/4 +
(L2*L4*de3*m4*sin(e3)*sin(e4))/2,
(L4^2*de3*m4*sin(2*e4))/8 + (L4*de1*m4*(2*L3*sin(e2 + e3 - e4) + 2*L3*sin(e2 + e3
+ e4) + 2*L4*cos(2*e4)*cos(e2 + e3)))/16 - (L3*L4*de4*m4*cos(e4))/4 +
(L4*de2*m4*sin(e4)*(L4*cos(e4) - L2*sin(e3)))/4,
(L4*de2*m4*cos(e4)*(L3 + L2*cos(e3)))/4 - (L4*de1*m4*sin(e4)*(L3*cos(e2 + e3) +
L2*cos(e2)))/4 + (L3*L4*de3*m4*cos(e4))/4];
de = [de1,de2,de3,de4]';
dTor=b*de+c*sign(de);
%% phương trình động lực học
dde=inv(M)*(Tor - C*de - G - dTor);
```