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:

NHẬN DIỆN CỬ CHỈ TAY ĐỂ ĐIỀU KHIỂN ROBOT ABB CRB15000

GV hướng dẫn: **TS. GIÁP QUANG HUY**

KS hướng dẫn: **KS. LÊ PHƯỚC SINH**

Sinh viên thực hiện:

CHÂU PHƯỚC NHẬT	**105200307**	**20TDH1**
NGUYỄN TRUNG HIẾU	**105200331**	**20TDH2**
TRẦN VĂN SANG	**105200342**	**20TDH2**

Đà Nẵng, 06/2025

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 CAPSTONE

TT	Họ tên sinh viên	Số thẻ SV	Lớp	Ngành
1	Châu Phước Nhật	105200307	20TDH1	Kỹ thuật điều khiển và tự động hóa
2	Nguyễn Trung Hiếu	105200331	20TDH2	Kỹ thuật điều khiển và tự động hóa
3	Trần Văn Sang	105200342	20TDH2	Kỹ thuật điều khiển và tự động hóa

1. Tên đề tài đồ án:

   Nhận diện cử chỉ tay để điều khiển robot ABB CRB15000

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	Châu Phước Nhật	Lên ý tưởng, chọn thiết bị cho việc làm mạch điện điều khiển.
2	Nguyễn Trung Hiếu	Thiết kế sơ đồ mạch điện, vẽ sơ đồ nguyên lý, làm mạch PCB.
3	Trần Văn Sang	Thi công, kiểm nghiệm. Hoàn thiện, chỉnh sửa báo cáo kết quả.

b. Phần riêng:

TT	Họ tên sinh viên	Nội dung
1	Châu Phước Nhật	Nghiên cứu, lập trình giao tiếp giữa Visual Studio với robot studio sử dụng nền tảng robot web servie. Thiết kế chức năng nhận diện cử chỉ tay sử dụng ngôn ngữ python. Thiết kế giao diện điều khiển sử dụng Visual Studio
2	Nguyễn Trung Hiếu	Nghiên cứu, tính toán toạ độ các vị trí cánh tay cẳng tay. Lập trình gửi nhận dữ liệu giữa thiết bị phần cứng và máy tính bằng ngôn ngữ python.
3	Trần Văn Sang	Nghiên cứu, xây dựng các lệnh điều khiển robot trên Visual Studio như khởi tạo dữ liệu kết nối với robot, thiết lập tốc độ di chuyển, bắt đầu và dừng quá trình hoạt động, giải phóng tài nguyên.

5. Người hướng dẫn:

- TS. Giáp Quang Huy
- KS. Lê Phước Sinh

6. Ngày giao nhiệm vụ đồ án: 14/3/2025

7. Ngày hoàn thành đồ án:

Đà Nẵng, ngày tháng năm 2023

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

TS. Giáp Quang Huy **TS. Giáp Quang Huy**

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: Châu Phước Nhật Số thẻ SV: 105200307
Nguyễn Trung Hiếu Số thẻ SV: 105200331
Trần Văn Sang Số thẻ SV: 105200342

Tên đề tài ĐATN: **Nhận diện cử chỉ tay để điều khiển robot ABB CRB15000**

Họ tên người HD1: TS. Giáp Quang Huy

Đơn vị: Khoa Điện

Họ tên người HD2: KS. Lê Phước Sinh

Đơn vị: Công ty TNHH Tự Động Hóa Nhật Tri

Tuần	**Ngày**	**Khối lượng**		**GVHD ký tên**
		Đã thực hiện (%)	**Tiếp tục thực hiện (%)**	
1	17/3 - 23/3	Nhận đề tài	Nghiên cứu lý thuyết liên quan đến đề tài	
2	24/3 - 30/3	Nghiên cứu lập trình Visual Studio Nghiên cứu và lập trình điều khiển robot trên phần mềm Robot Studio bằng ngôn ngữ Rapid	Tiếp tục thực hiện dự án. Làm thuyết minh chương 1 và 2.	
3	31/3 – 6/4	Hoàn thành thuyết minh chương 1 và 2. Thiết kế mạch điện và làm mạch cảm biến.	Nghiên cứu lập trình thiết kế giao diện. Nghiên cứu lập trình các cảm biến.	

4	6/4 – 7/4	Duyệt lần 1: Đánh giá khối lượng hoàn thành 25 %: Được tiếp tục làm ĐATN□Không tiếp tục thực hiện ĐATN □		
5	7/4 – 13/4	Nghiên cứu và lập trình giao tiếp EGM giữa 2 phần mềm Visual Studio Code và RobotStudio2025. Nghiên cứu và lập trình giao tiếp giữa box điều khiển và máy tính.	Lập trình mô phỏng điều khiển cho Robot ABB. Tính toán toạ độ cho robot dựa trên toạ độ các khớp tay.	
6	14/4 – 20/4	Nghiên cứu và lập trình giao tiếp RWS giữa 2 phần mềm Visual Studio Code và RobotStudio2025.	Nghiên cứu lập trình thiết kế giao diện. Nghiên cứu phát triển các tính năng mới	
7	21/4 – 22/4	Duyệt lần 2: Đánh giá khối lượng hoàn thành 50 % : Được tiếp tục làm ĐATN□Không tiếp tục thực hiện ĐATN □		
8	22/4 – 27/4	Nghiên cứu và lập trình thiết kế giao diện với PyQt5	Nghiên cứu lập trình thiết kế giao diện. Nghiên cứu phát triển các tính năng mới	
9	21/4 – 4/5	Nghiên cứu, xây dựng tính năng nhận diện cử chỉ người dùng bằng xử lý ảnh.	Nghiên cứu lập trình thiết kế giao diện. Nghiên cứu phát triển các tính năng mới Làm thuyết minh chương 3 và 4.	
10	5/5 – 11/5	Kết nối giao diện điều khiển với robot thông qua nền tảng RWS	Kiểm tra, tối ưu các thuật toán trên giao diện điều khiển và trên phần mềm điều khiển robot.	

11	12/5 – 18/5	Khắc phục các lỗi kết nối với Robot.	Kiểm tra, tối ưu các thuật toán trên giao diện điều khiển và trên phần mềm điều khiển robot. Làm thuyết minh chương 5 và 6	
12	19/5 – 20/5	Duyệt lần 3: Đánh giá khối lượng hoàn thành 90 %: Được tiếp tục làm ĐATN□ Không tiếp tục thực hiện ĐATN □		
13	21/5 – 31/5	Tối ưu hóa code trên Visual Studio và Robot Studio.	Quay video demo. Kiểm tra tính chính xác của quá trình mapping toạ độ.	
14	1/6 – 14/6	Kiểm tra, chạy thử Robot ABB và khắc phục các lỗi nếu có. Hoàn thành quay video demo.	Chỉnh sửa bản thuyết minh	
15	15/6 – 20/6	Hoàn thành thuyết minh đồ án tốt nghiệp		

TÓM TẮT

Tên đề tài: **Nhận diện cử chỉ tay để điều khiển robot ABB CRB150000.**

Sinh viên thực hiện:	Châu Phước Nhật	105200307	20TDH1
	Nguyễn Trung Hiếu	105200331	20TDH2
	Trần Văn Sang	105200342	20TDH2

Nội dung tóm tắt:

Dự án "Nhận diện cử chỉ tay để điều khiển robot ABB CRB15000" phát triển một hệ thống điều khiển robot tiên tiến bằng cách kết hợp cảm biến MPU và xử lý ảnh để nhận diện cử chỉ tay người dùng. Hệ thống này cho phép điều khiển robot ABB CRB15000 một cách chính xác và nhanh chóng thông qua giao thức EGM, đồng thời tối ưu hóa trải nghiệm người dùng với bộ cử chỉ đơn giản, dễ thực hiện và một giao diện đồ họa giám sát thân thiện. Đề tài không chỉ hướng đến mở rộng ứng dụng robot trong công nghiệp mà còn mang giá trị học thuật quan trọng giúp sinh viên nắm vững kiến thức về robot, lập trình điều khiển (bao gồm vi điều khiển, ứng dụng và robot công nghiệp), xử lý tín hiệu, xử lý ảnh, đồng thời rèn luyện kỹ năng làm việc nhóm và giải quyết vấn đề thực tế.

1. **Lý do chọn đề tài.**

Đề tài này kết hợp giữa công nghệ robot và khả năng xử lý ảnh, tạo ra một sự giao thoa độc đáo giữa hai lĩnh vực này. Điều này mang lại cơ hội khám phá và khai thác tiềm năng sáng tạo của cả hai lĩnh vực, đồng thời đề tài này cũng giúp chúng em rèn luyện kỹ năng làm việc nhóm, vì nó yêu cầu sự phối hợp giữa các thành viên để phát triển cả hai phần mềm và đảm bảo tính khả thi của đồ án.

2. **Các vấn đề nghiên cứu.** Nghiên cứu, lập trình cho Rob ot ABB CRB 15000 - Thiết kế điều khiển, giám sát cho Robot ABB - Xây dựng giao diện xử lý ảnh bằng phần mềm Visual Studio Code - Thiết kế hệ thống phần cứng cho việc ước tính toạ độ - Nghiên cứu, xây dựng giao diện điều khiển bằng phần mềm Visual Studio Code - Nghiên cứu cách trao đổi dữ liệu giữa Python (Chạy trên PC) và Robot ABB
3. **Phương pháp nghiên cứu.**

Nghiên cứu tài liệu để thu thập thông tin, phân tích và đánh giá về hệ thống robot, từ đó tiến hành xây dựng các giải pháp để thực hiện và giải quyết các vấn đề cụ thể mà dự án đặt ra. Xây dựng mô hình robot và tiến hành chạy thử trên thiết bị thực tế. Bằng cách kết hợp các phương pháp nghiên cứu tài liệu, phân tích hệ thống, xây dựng giải pháp và thực hiện thử nghiệm, chúng em hy vọng sẽ đạt kết quả tốt nhất trong việc ứng dụng robot một cách thông minh hơn.

LỜI NÓI ĐẦU VÀ CẢM ƠN

Để có thể thực hiện và hoàn thành đề tài này, nhóm xin gửi lời chân thành cảm ơn các thầy cô trong Khoa Điện đã tạo những điều kiện tốt nhất cho chúng em hoàn thành đề tài. Những kiến thức bổ ích mà các Thầy cô dạy, nó được áp dụng vào đề tài đồ án tốt nghiệp rất nhiều, từ những kiến thức nhỏ nhặt cho tới những bài học lớn. Một lần nữa, nhóm xin được gửi lời cảm ơn đến tất cả thầy cô, nếu không có thầy cô thì chắc giờ này nhóm sẽ khó có thể hoàn thành đề tài này.

Ngoài sự cố gắng của bản thân, nhóm em không thể nào không nhắc đến công lao đã vạch ra hướng đi cho đề tài và hướng dẫn từng yêu cầu của đề tài mà thầy TS. Giáp Quang Huy đã truyền đạt cho nhóm em những kiến thức hết sức bổ ích và những ứng dụng thực tế. Thầy TS. Giáp Quang Huy ân cần chỉ bảo tận tình. Giải thích rõ ràng những chỗ mà nhóm em chưa hiểu.

Tiếp theo nhóm cũng xin cảm ơn tới các Anh kĩ thuật của công ty Tự Động Hóa Nhật Tri đã tạo điều kiện giúp đỡ, từ những tài liệu liên quan tới đề tài cho tới những kinh nghiệm sống thực tế. Nhờ đó mà nhóm mới có thể phát triển được.

Mặc dù nhóm em đã cố gắng hoàn thành tốt đề tài này một cách hoàn chỉnh nhất, nhưng cũng không thể tránh những sai sót nhất định trong công tác nghiên cứu, tiếp cận thực tế, cũng như những hạn chế về kiến thức lẫn thời gian thực hiện. Rất mong nhận được sự góp ý của quý thầy cô để đề tài này được hoàn chỉnh.

Xin chân thành cảm ơn!

LỜI CAM ĐOAN LIÊM CHÍNH HỌC THUẬT

Em xin cam đoan đề tài: "**NHẬN DIỆN CỬ CHỈ TAY ĐỂ ĐIỀU KHIỂN ROBOT ABB CRB15000**" là sự nghiên cứu của nhóm chúng em dưới sự hướng dẫn của TS. Giáp Quang Huy (Khoa Điện – Trường Đại học Bách Khoa – Đại học Đà Nẵng) và KS Lê Phước Sinh (Công ty TNHH Tự Động Hóa Nhật Tri). Ngoài ra có sự giúp đỡ của bạn bè và không có bất cứ sự sao chép của người khác.

Đề tài, nội dung báo cáo là sản phẩm mà nhóm em đã nghiên cứu trong quá trình học tập tại trường.

Phần sử dụng tài liệu tham khảo đã được nêu rõ trong mục "Tài liệu tham khảo". Các số liệu và kết quả trong báo cáo này là hoàn toàn trung thực. Chúng em xin chịu hoàn toàn trách nhiệm và kỷ luật của bộ môn và nhà trường nếu như có vấn đề xảy ra.

Sinh viên thực hiện

(Kí và ghi rõ họ tên)

MỤC LỤC

DANH SÁCH CÁC BẢNG, HÌNH VẼ

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

CHỮ VIẾT TẮT:

EGM: Externally Guided Motion

RWS: Robot Web Services

HTTP: HyperText Transfer Protocol

API: Application Programming Interface

URL: Uniform Resource Locator

REST: Representational State Transfer

MỞ ĐẦU

1. Lý do chọn đề tài

Trong bối cảnh công nghiệp hiện đại và xu hướng phát triển các hệ thống tự động hóa thông minh, công nghệ điều khiển bằng cử chỉ đang dần trở thành một trong những hướng tiếp cận tiên tiến và đầy tiềm năng. Việc điều khiển robot bằng cử chỉ tay không chỉ mở ra khả năng tương tác tự nhiên, trực quan giữa con người và máy móc mà còn mang lại tính linh hoạt và tiện lợi cao trong các ứng dụng thực tiễn như sản xuất, chăm sóc sức khỏe và hỗ trợ người khuyết tật.

Đề tài "Nhận diện cử chỉ tay để điều khiển robot ABB CRB15000" được lựa chọn nhằm tiếp cận và ứng dụng các công nghệ cảm biến IMU vào một hệ thống điều khiển thực tế. Đây là một đề tài liên ngành, kết hợp giữa kỹ thuật điện tử, cơ điện tử, và lập trình điều khiển robot, từ đó giúp các thành viên trong nhóm nâng cao kỹ năng chuyên môn như xử lý tín hiệu, lập trình nhúng, truyền thông không dây và lập trình robot công nghiệp. Đồng thời, đề tài cũng tạo cơ hội để rèn luyện kỹ năng làm việc nhóm, quản lý dự án, và giải quyết vấn đề trong quá trình thiết kế và triển khai hệ thống thực tế.

2. Mục tiêu

Phát triển hệ thống nhận diện cử chỉ tay dựa trên sự kết hợp giữa cảm biến MPU và xử lý ảnh.

Điều khiển Robot ABB thông qua giao tiếp EGM để robot phản hồi chính xác, nhanh với cử chỉ.

Tối ưu trải nghiệm người dùng bằng việc thiết kế bộ cử chỉ đơn giản, dễ thực hiện và ghi nhớ, phù hợp với đối tượng người dùng đa dạng.

Xây dựng giao diện đồ họa đơn giản để cài đặt, giám sát trạng thái hệ thống.

Mở rộng khả năng ứng dụng robot vào các hệ thống công nghiệp.

3. Ý nghĩa

Đồ án giúp sinh viên nắm vững kiến thức về cấu trúc và nguyên lý hoạt động của robot cũng như ra cơ hội cho sinh viên học hỏi và rèn kỹ năng lập trình robot, bao gồm việc tạo và điều khiển các chuyển động, tương tác I/O và giao tiếp với các phần mềm điều khiển khác nhau. Việc thực hiện dự án sẽ cung cấp cho sinh viên kinh nghiệm và kiến thức quý báu về lập trình robot thực tế. Bên cạnh đó sinh viên còn học hỏi được những kiến thức về xử lý ảnh. Ngoài ra, đồ án này còn mang lại ý nghĩa quan trọng trong việc biết cách lập trình vi điều khiển, lập trình ứng dụng.

4. Bố cục của đề tài

Bố cục của đề tài ngoài phần lời mở đầu và kết luận chung, nội dung của đề tài được chia thành 5 chương như sau:

Mở đầu:

Chương 1: Cơ sở lý thuyết và tổng quan đề tài

Chương 2: Thiết kế phần cứng, ước lượng định hướng tay người và truyền dữ liệu thời gian thực

Chương 3: Thiết kế và triển khai hệ thống phần mềm giao tiếp điều khiển robot ABB sử dụng RWS.

Chương 4: Thử nghiệm và đánh giá

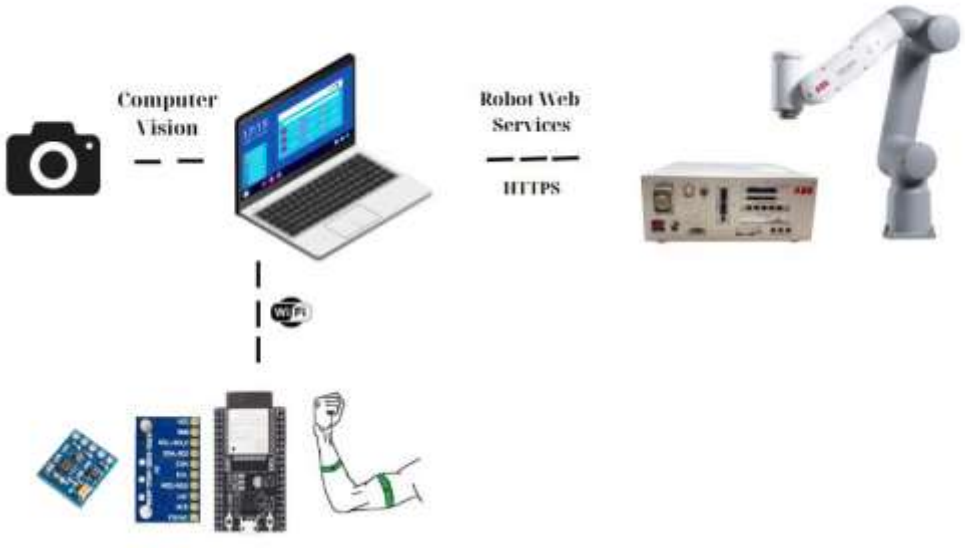
Chương 5: Kết luận và hướng phát triển

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT VÀ TỔNG QUAN ĐỀ TÀI

1.1. Tổng quan đề tài

Ngày nay với phát triển mạnh mẽ của công nghệ việc ứng dụng các phương thức điều khiển tiên tiến đang ngày càng được quan tâm và nghiên cứu sâu rộng. Khả năng điều khiển robot thông qua cử chỉ tay mang lại sự tương tác mới mẻ, trực quan, linh hoạt và thân thiện hơn giữa con người và máy móc. Đề tài được triển khai nhằm nghiên cứu và hiện thực hóa giải pháp điều khiển robot sử dụng cảm biến IMU. Đây là một hướng tiếp cận liên ngành, góp phần phát triển các kỹ năng như xử lý tín hiệu, lập trình và điều khiển robot công nghiệp, đồng thời nâng cao khả năng làm việc nhóm và giải quyết bài toán thực tế trong kỹ thuật điều khiển hiện đại.

Hệ thống được thiết kế với kiến trúc tổng thể bao gồm nhiều thành phần phần cứng và phần mềm, phối hợp nhịp nhàng nhằm thực hiện chức năng nhận diện cử chỉ tay, truyền lệnh điều khiển và giám sát robot ABB CRB15000. Sơ đồ tổng quan và vai trò của từng thành phần được trình bày trong Hình 1.1 và Bảng 1.1 dưới đây.



Hình 1.1. Tổng quan hệ thống

Bảng 1.1. Vai trò các thành phần trong hệ thống

Thành phần	Vai trò chính
Camera	Thu nhận ảnh đầu vào
OpenCv	Tiền xử lý ảnh
MediaPipe	Nhận diện và trích xuất cử chỉ tay
Thuật toán điều khiển	Phân tích cử chỉ và nhận lệnh cho robot
PyQt5	Giao diện người dùng, hiển thị video và điều khiển
RWS	Giao tiếp điều khiển và giám sát giữa PC và Robot
EGM	Giao tiếp thời gian thực, truyền nhận dữ liệu vị trí và điều khiển chuyển động giữa PC và robot
Bộ điều khiển Robot	Nhận lệnh và điều phối hoạt động robot
Robot ABB CRB15000	Thực thi hành động vật lí
ESP32	Thu thập và truyền dữ liệu cảm biến đến PC
MPU9250 và QMC5883L	Cung cấp dữ liệu gia tốc, con quay hồi chuyển và từ kế

Hệ thống điều khiển robot sử dụng cảm biến hai cảm biến MPU9250 và QMC5883L để đo đạc và tính toán toạ độ từ cánh tay người dùng, sau đó truyền dữ liệu đến vi điều khiển ESP32. ESP32 xử lý sơ bộ và gửi tín hiệu qua WiFi đến máy tính. Máy tính có nhiệm vụ tiếp nhận dữ liệu, phân tích và gửi toạ độ vừa tính được cho robot để CPU robot có thể đưa cánh tay đến vị trí xác định. Bên cạnh đó việc xử lý các hiệu lệnh tay thông qua xử lý ảnh cũng được máy tính tiếp nhận và gửi các lệnh này được gửi tới bộ điều khiển của robot ABB thông qua giao thức HTTPS sử dụng Robot Web Service (RWS) để điều khiển robot. Cuối cùng, cánh tay robot ABB thực hiện các chuyển động tương ứng với cử động người dùng yêu cầu. Hệ thống này cho phép điều khiển robot từ xa bằng tín hiệu sinh học, ứng dụng trong các lĩnh vực y tế, sản xuất như phục hồi chức năng hay điều khiển không tiếp xúc.

Mục tiêu

Phát triển hệ thống nhận diện cử chỉ tay thông qua phương pháp: Đồng bộ chuyển động giữa tay người và cánh tay robot bằng cách sử dụng hai cảm biến

MPU9250 và QMC5883L để theo dõi và tính toán vị trí cổ tay người dùng trong không gian ba chiều. Ứng dụng xử lý ảnh vào việc điều khiển robot thông qua các cử chỉ tay.

Thiết kế hệ thống giao tiếp không dây giữa vi điều khiển ESP32 và máy chủ điều khiển robot thông qua kết nối Wifi, đảm bảo độ ổn định và tốc độ truyền dữ liệu.

Tích hợp và triển khai điều khiển robot ABB thông qua giao thức Externally Guided Motion (EGM), đảm bảo robot có thể phản hồi chính xác và nhanh chóng với từng cử chỉ được nhận diện.

Đánh giá hiệu suất hệ thống thông qua việc mô phỏng và thử nghiệm thực tế, từ đó điều chỉnh thuật toán để tối ưu độ chính xác và độ trễ trong điều khiển.

Tạo nền tảng ứng dụng thực tiễn trong các lĩnh vực như tự động hóa sản xuất, y tế phục hồi chức năng, và điều khiển không chạm.

1.2. Giới thiệu về thiết bị sử dụng

1.2.1. Giới thiệu về robot ABB CRB15000

1.2.1.1. Tổng quan Robot CRB15000 (GoFa)

Robot CRB15000 là một robot công nghiệp nhỏ gọn và linh hoạt, được sản xuất bởi hãng ABB. Với khối lượng chỉ khoảng 54kg, độ chính xác cao và tốc độ nhanh, CRB15000 có thể được sử dụng trong nhiều ứng dụng khác nhau, bao gồm hàn, cắt, đóng gói, định vị và lắp ráp. Robot GoFa CRB15000 là một sự kết hợp hàng loạt các tính năng cho phép con người có thể vận hành nó một cách an toàn, trực tiếp mà không cần quá nhiều nhân công hay một hệ thống hàng rào che chắn phức tạp. GoFa có thể liên tục chia sẻ không gian làm việc với mọi người, mang lại cho nó sự linh hoạt và hiệu quả tối đa. Robot và con người hợp tác trong các nhiệm vụ giống nhau mà không ảnh hưởng đến năng suất nhưng vẫn đảm bảo độ an toàn. Đặc điểm này rất quan trọng trong các ứng dụng cần độ an toàn cao như tích hợp với máy laser hoặc một dây chuyền làm việc cần sự can thiệp của con người.

Hình 1.2. Robot GoFa CRB 15000

Thông số kỹ thuật:

Bảng 1.2. Thống số kỹ thuật Robot GoFa CRB 15000

Số trục	6
Bảo vệ an toàn	IP54
Lắp đặt	Có thể lắp đặt bất cứ vị trí nào, bao gồm: trên bàn làm việc, tường và trần nhà
Bộ điều khiển	OmniCore C30
Điện năng tiêu thụ	24A/220V
Tín hiệu kết nối	4 tín hiệu (IO, Fieldbus hoặc Ethernet)
Chất lượng lắp ráp	Tiêu chuẩn ISO 9409-1-50
Chức năng an toàn	SafeMove Collaborative Chứng nhận an toàn Loại 3, PL d
Tầm làm việc (mm)	950
Khối lượng có thể tải (kg)	5

1.2.1.2. Giới thiệu bộ điều khiển OmniCore C30

Hình 1.3. Bộ điều khiển Ominicore C30

OmniCore C30 là bộ điều khiển chính của robot GoFa CRB15000. Bộ điều khiển này có thiết kế nhỏ gọn nhưng mạnh mẽ, tích hợp khả năng điều khiển chính xác và linh hoạt, đáp ứng yêu cầu của các hệ thống tự động hóa hiện đại. C30 được sử dụng rộng rãi trong ngành Thực phẩm & Đồ uống, Điện tử và các ngành công nghiệp khác, hỗ trợ robot cộng tác, Delta, SCARA và các robot khớp nối lên đến IRB 1600.

Thông số kỹ thuật:

Bảng 1.3. Thông số kỹ thuật bộ điều khiển Omnicore C30

Hạng mục	Thông số
Phần cứng	Hệ thống đa xử lý PCI Bus Bộ nhớ Flash dung lượng lớn
Phần mềm điều khiển	RobotWare OS thời gian thực Ngôn ngữ lập trình RAPID
Điện áp cung cấp	220V/300VAC, 50-60Hz
Trọng lượng	28.5kg
Nhiệt độ môi trường	0°C đến + 45°C
Độ ẩm tương đối	Max 95%
Tiểu chuẩn bảo vệ	IP20
Cổng I/O	16 DI/16 DO

1.2.1.3. FlexPendant – Thiết bị điều khiển cầm tay

Hình 1.4. Tay cầm điều khiển Robot Gofa CRB 15000

FlexPendant là thiết bị điều khiển cầm tay được thiết kế nhỏ gọn, trực quan, giúp người vận hành dễ dàng giao tiếp và điều khiển robot GoFa CRB15000. Thiết bị này hỗ trợ màn hình cảm ứng đa điểm, cho phép truy cập nhanh các chức năng chính, thiết kế giao diện điều hành cá nhân hóa, tạo các giải pháp tùy chỉnh và biểu tượng riêng. FlexPendant cũng hỗ trợ hoán đổi nóng giữa nhiều robot, tăng tính linh hoạt và thuận tiện khi sử dụng trong các dây chuyền sản xuất.

1.2.2. Giới *thiệu* cảm biến MPU9050.

MPU9250 là cảm biến IMU 9 trục do InvenSense (nay thuộc TDK) sản xuất, được sử dụng phổ biến trong các hệ thống điều khiển chuyển động. Cảm biến này tích hợp ba thành phần trong một chip: gia tốc kế 3 trục đo gia tốc tuyến tính, con quay hồi chuyển 3 trục đo tốc độ quay góc, và từ kế 3 trục (dùng chip AK8963) đo từ trường Trái Đất để xác định phương hướng. Với khả năng cung cấp 9 Degrees of Freedom (9DOF), MPU9250 cho phép thu thập dữ liệu chính xác về vị trí và chuyển động trong không gian 3 chiều.

Hình 1.5. Cảm biến MPU9250

Thông số kỹ thuật:

Bảng 1.4. Bảng thông số kỹ thuật cảm biến MPU9250

Điện áp sử dụng	Giao tiếp	Cảm biến gia tốc	Cảm biến con quay	Cảm biến từ kế
3-5V	I2C	±2g, ±4g, ±8g, ±16g	±250, ±500, ±1000, ±2000°/s	±4800μT

Ứng dụng trong đồ án:

Sử dụng cảm biến MPU9250 để thu thập dữ liệu gia tốc và gyro, từ đó tính toán và ước lượng góc nghiêng Roll và Pitch của cánh tay người đeo.

1.2.3. Cảm biến la bàn số QMC-5883L

QMC5883L là cảm biến từ kế 3 trục với bộ chuyển đổi ADC 16-bit, tích hợp công nghệ AMR từ Honeywell. Chip này có kích thước rất nhỏ (3×3×0.9 mm) và phù hợp cho các ứng dụng điều hướng như drone, robot, thiết bị cầm tay. Điện áp cấp hoạt động khoảng 2.16–3.6 V. Xử lý điện áp I2C và thực hiện các hiệu chuẩn đúng cách.

Vì cảm biến MPU9250 gặp vấn đề khi đo giá trị Yaw nên nhóm chúng đã sử dụng thêm cảm biến QMC5883L để đo và tính giá trị Yaw.

Hình 1.6. Cảm biến la bàn QMC5883L

Thông số kỹ thuật:

Bảng 1.5. Thông số cảm biến QMC5883L

Điện áp sử dụng	Giao tiếp	Cảm biến từ kế
3-5V	I2C	±4800μT

Ứng dụng trong đồ án:

Sử dụng cảm biến QMC5883L để thu thập dữ liệu từ trường, hỗ trợ tính toán và ước lượng góc Yaw của cánh tay người đeo.

1.2.4. Vi điều khiển ESP 32

ESP32 là một series các vi điều khiển trên một vi mạch giá rẻ, năng lượng thấp có tích hợp WiFi và dual-mode Bluetooth (tạm dịch: Bluetooth chế độ kép). Dòng ESP32 sử dụng bộ vi xử lý Tensilica Xtensa LX6 có hai biến thể lõi kép và lõi đơn, và bao gồm các công tắc antenna tích hợp, RF balun, bộ khuếch đại công suất, bộ khuếch đại thu nhiễu thấp, bộ lọc và module quản lý năng lượng.

Hình 1.7. Chip xử lí vi điều khiển ESP32

Thông số kỹ thuật:

Bảng 1.6. Thông số cở bản của chip ESP32

Điện áp hoạt động	Chip	Tần số	Số chân GPIO	Giao tiếp
5V DC	Tensilica Xtensa LX6	80 - 240 MHz	34	UART, I2C, SPI, PWM…

Ứng dụng trong đồ án:

ESP32 được sử dụng để thu thập dữ liệu từ cảm biến và truyền dữ liệu đến hệ thống điều khiển thông qua kết nôi không dây.

1.3. Giới thiệu về tính năng Externally Guided Motion (EGM)

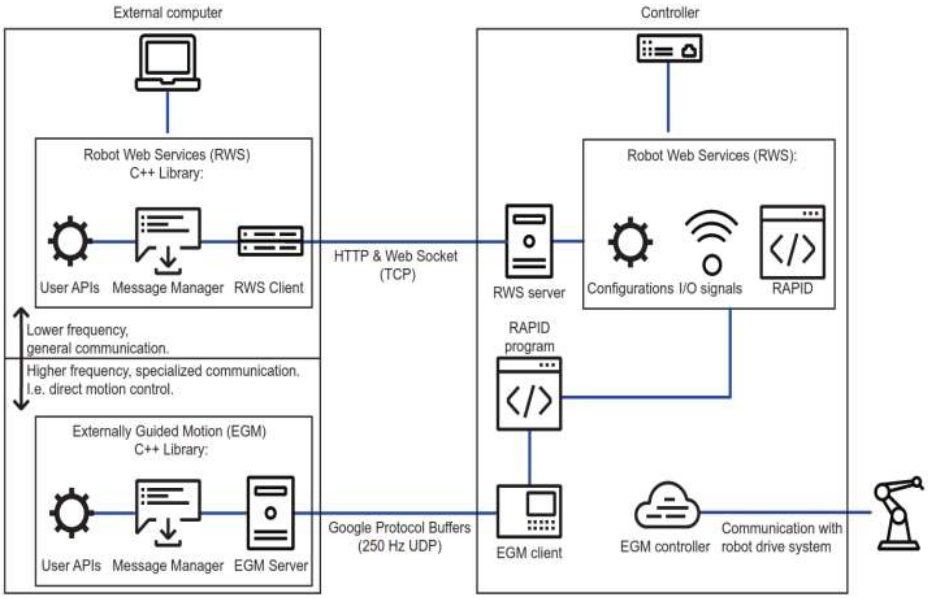
1.3.1. Tổng quan về tính năng EGM Guided Motion

EGM, viết tắt của Externally Guided Motion, là một tính năng của robot ABB cho phép các thiết bị bên ngoài tương tác và điều khiển robot trong thời gian thực thông

qua mạng. Nó sử dụng giao thức UDP và Google Protocol Buffers để truyền dữ liệu giữa robot và máy tính bên ngoài.

Externally Guided Motion (EGM) bao gồm 3 tính năng khác nhau:

- EGM Position Stream: Vị trí hiện tại và kế hoạch của các đối tượng cơ khí trong RAPID Task được gửi đến một thiết bị bên ngoài.
- EGM Position Guidance: Robot không thực hiện theo đường đi được lập trình trong RAPID mà thực hiện theo đường đi do một thiết bị bên ngoài tạo ra
- EGM Path Correction: Đường đi của robot được lập trình được sửa đổi/ hiệu chỉnh bởi dữ liệu của các thiết bị bên ngoài.



Hình 1.8. Sơ đồ tổng quan của EGM

Ưu điểm của EGM:

- Tốc độ cao, độ trễ thấp: EGM Position Guidance cho phép đọc/ghi vị trí mỗi 4 ms, với độ trễ điều khiển khoảng 10-20 ms và chỉ mất khoảng 20 ms để robot phản hồi vị trí mới.
- Bỏ qua lập kế hoạch đường đi: Giao diện cấp thấp này không cần kế hoạch đường đi sẵn, giúp điều khiển robot tức thì phù hợp cho hàn laser, theo dõi đường may, hay bắt bám vật thể di chuyển.

- Đa chế độ điều khiển: Hỗ trợ Joint mode (góc trục) và Pose mode (tư thế). Trong Pose mode, có thể thiết lập khung tham chiếu (Tool, Work Object, Correction, Sensor) để điều khiển chính xác.
- Tích hợp điều khiển khác: Có thể kết hợp cùng các logic hay chế độ như Force Control thông qua tham số mở rộng (ví dụ \NoWaitCond).
- Giao tiếp linh hoạt: Dùng giao thức UdpUc để truyền nhận dữ liệu vị trí với thiết bị ngoài một cách nhanh chóng.

Nhược điểm:

- Không thể sử dụng tính năng EGM Position Guidance để điều khiển một đơn vị cơ khí trong một đối tượng làm việc đang di chuyển.
- Nếu robot tiến gần đến một điểm kỳ dị, tức là khi hai trục của robot gần như song song với nhau, chuyển động của robot sẽ bị dừng lại và một thông báo lỗi sẽ xuất hiện. Trong tình huống đó, cách duy nhất là điều khiển robot ra khỏi điểm kỳ dị bằng tay (jogging).

1.3.2. Tổng quan về tính năng EGM Position Guidance

1.3.2.1. Giới thiệu về tính năng EGM Position Guidance

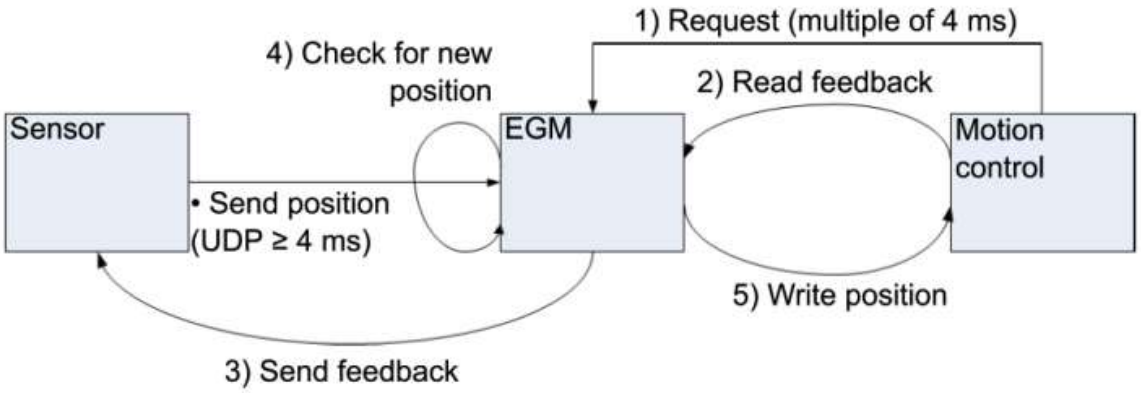
EGM Position Guidance được thiết kế cho người dùng nâng cao và cung cấp giao diện điều khiển cấp thấp cho bộ điều khiển robot ABB. Tính năng này bỏ qua bộ lập kế hoạch đường đi bên trong và cho phép thiết bị bên ngoài trực tiếp hướng dẫn chuyển động của robot theo thời gian thực, đảm bảo độ phản hồi nhanh và chuyển động mượt mà. EGM Position Guidance cho phép gửi và nhận dữ liệu với chu kỳ lên đến 4 ms, với độ trễ điều khiển điển hình chỉ khoảng 10 đến 20 ms tùy loại robot. Khoảng thời gian từ khi ghi một vị trí mới đến khi vị trí đó bắt đầu ảnh hưởng đến chuyển động thực tế của robot thường chỉ khoảng 20 ms. Ngoài ra, tính năng này cung cấp độ chính xác tuyệt đối, giúp robot đạt đúng vị trí mục tiêu như được chỉ định từ thiết bị điều khiển bên ngoài.

EGM Position Guidance cũng cho phép đọc vị trí từ hệ thống chuyển động (motion system) và ghi vị trí đến hệ thống chuyển động với tốc độ cao, đáp ứng các yêu cầu điều khiển chuyển động chính xác trong thời gian thực. Các tham chiếu điều khiển có thể được xác định bằng giá trị các trục hoặc tư thế, trong đó tư thế có thể được định nghĩa trong bất kỳ work object nào không di chuyển trong quá trình thực hiện EGM Position Guidance. Đây là giải pháp lý tưởng cho các ứng dụng cần điều khiển

chuyển động robot với tốc độ cao và độ trễ thấp so với các phương pháp điều khiển ngoài khác.

1.3.2.2. Quy trình trao đổi dữ liệu vị trí qua UdpUc

Luồng trao đổi dữ liệu vị trí qua kết nối UdpUc được minh họa trong sơ đồ bên dưới. Khi sử dụng giao thức UdpUc, dữ liệu được truyền qua mạng dưới dạng các gói UDP, với chu kỳ truyền tối thiểu là 4 ms, đảm bảo khả năng phản hồi nhanh và phù hợp cho các ứng dụng điều khiển thời gian thực.



Hình 1.9. Luồng dữ liệu UdpUc interface

Bước 1: Motion control gọi EGM

Bước 2: EGM đọc dữ liệu phản hồi từ motion control

Bước 3: EGM gửi dữ liệu phản hồi đến cảm biến

Bước 4: EGM kiểm tra UDP từng đợt để tìm tin nhắn từ cảm biến

Bước 5: Nếu có tin nhắn, EGM đọc tin nhắn tiếp theo và bước 5 ghi dữ liệu vị trí đến motion control. Nếu không có dữ liệu vị trí nào được gửi, motion control tiếp tục sử dụng dữ liệu vị trí gần nhất trước đó được ghi bởi EGM. Cảm biến gửi dữ liệu vị trí đến bộ điều khiển (EGM), khuyến nghị kết hợp bước này với bước 3. Như vậy cảm biến sẽ cùng pha với bộ điều khiển.

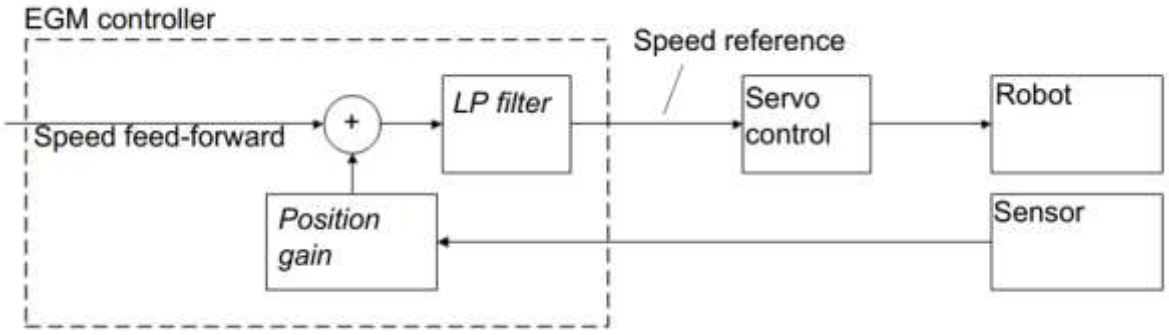
1.3.2.3. Thông số tác động đến vòng điều khiển vị trí EGM

Vòng điều khiển EGM (Enhanced Global Motion) trong robot ABB sử dụng các thông số điều chỉnh để đảm bảo độ chính xác và ổn định khi di chuyển theo yêu cầu từ cảm biến. Hai thông số quan trọng nhất ảnh hưởng đến vòng điều khiển này là:

Bảng 1.7. Bảng mô tả hai thông số ảnh hưởng đến vòng điều khiển EGM

Thông số	Mô tả
Default Proportional Position Gain	Thông số Position gain trong hình ảnh ảnh hưởng đến khả năng phản hồi khi di chuyển đến vị trí mục tiêu do cảm biến đưa ra liên quan đến vị trí hiện tại của robot. Giá trị càng cao thì phản hồi càng nhanh.
Default Low Pas Filter Bandwith Time	Thông số LP Filter trong hình là giá trị mặc định dung để lọc sự bù tốc độ từ EGM

Hai thông số này được minh họa rõ ràng trong hình dưới đây :



Hình 1.10. Sơ đồ mô tả thông số ảnh hưởng đến vòng điều khiển EGM

1.3.2.4. Hệ trục tọa độ trong EGM Position Guidance

EGM có thể hoạt động ở hai chế độ: joint mode và pose mode.

- Ở Joint mode, không cần sử dụng khung tham chiếu vì cả giá trị cảm biến và giá trị vị trí đều được biểu diễn dưới dạng góc từng trục (tính bằng độ), tương ứng với vị trí hiệu chuẩn (calibration position) của từng trục.
- Ở Pose mode, các khung tham chiếu là bắt buộc. Trong chế độ này, dữ liệu từ cảm biến và hướng thay đổi vị trí của robot luôn được xác định dựa trên một khung tham chiếu cụ thể.

Các hệ trục tọa độ này được định nghĩa thông qua lệnh EGMActPose trong RAPID với các thành phần chính như sau:

Bảng 1.8. Frame cho EGM Position Guidance

Frame	Mô tả
Tool	Dữ liệu tool sử dụng cho EGM được xác định bởi tham số \Tool.
Work object	Dữ liệu work object sử dụng cho EGM được xác định bởi tham số \Wobj
Correction	Frame sử dụng để xác định hướng di chuyển cuối cùng thì được xác định bởi tham số bắt buộc CorFrame
Sensor	Frame sử dụng để diễn tả dữ liệu cảm biến thì được xác định bởi tham số bắt buộc SensorFrame

1.4. Giới thiệu về Robot Web Services:

1.4.1. Tổng quan về Robot Web Services

Robot Web Services là một nền tảng cho phép các nhà phát triển tạo ra các ứng dụng tùy chỉnh của riêng họ để tương tác với bộ điều khiển robot. Robot Web Services cung cấp các API RESTful tận dụng giao thức HTTPS và các thông báo được tạo thành từ XHTML và JSON.

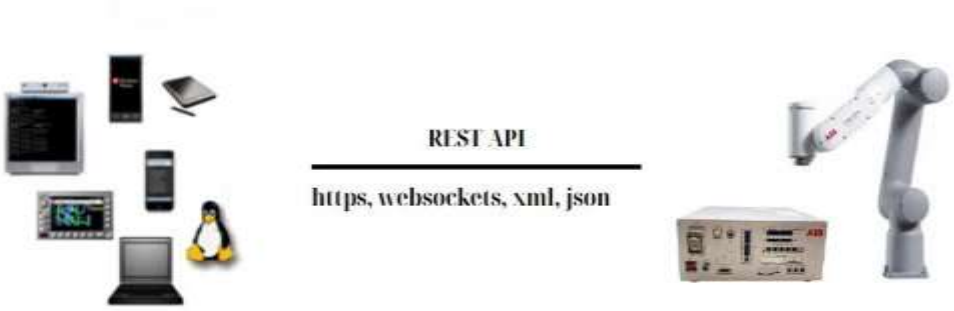
HTTP (HyperText Transfer Protocol) là một giao thức truyền thông hoạt động theo mô hình client–server, đóng vai trò trung gian giúp máy khách (client) giao tiếp với máy chủ (server) để gửi và nhận dữ liệu.

HTTPS (HTTP Secure) là phiên bản bảo mật của HTTP. HTTPS sử dụng giao thức TLS (hoặc SSL trước đây) để mã hóa dữ liệu truyền tải, giúp bảo vệ tính riêng tư và toàn vẹn dữ liệu khỏi bị nghe lén hoặc giả mạo.

API (Application Programming Interface) là tập hợp các quy tắc, giao diện cho phép các phần mềm hoặc hệ thống giao tiếp, trao đổi dữ liệu với nhau. API giúp các ứng dụng bên ngoài truy cập vào chức năng hoặc dữ liệu của một hệ thống mà không cần biết chi tiết bên trong.

REST API (Representational State Transfer API) là một loại API tuân theo các nguyên tắc thiết kế của REST, hoạt động trên nền giao thức HTTP/HTTPS. REST API:

- Sử dụng URL để đại diện cho các tài nguyên (resources).
- Sử dụng các phương thức HTTP (GET, POST, PUT, DELETE...) để thao tác với tài nguyên.
- Dữ liệu trả về thường ở định dạng JSON hoặc XML, dễ phân tích và xử lý.



Hình 1.11. Truyền thông robot với PC thông qua RWS

Robot Web Services tạo điều kiện cho việc giao tiếp độc lập với nền tảng và ngôn ngữ với bộ điều khiển robot. Trong REST, một URL xác định một tài nguyên. Biểu diễn dữ liệu ứng dụng được gửi từ bộ điều khiển robot có thể ở định dạng XHTML hoặc JSON. Robot Web Services không cung cấp thông tin định dạng về cách dữ liệu sẽ được hiển thị trong trình duyệt web.

Các API trong RWS tận dụng các phương thức HTTP chuẩn để thao tác với tài nguyên của robot:

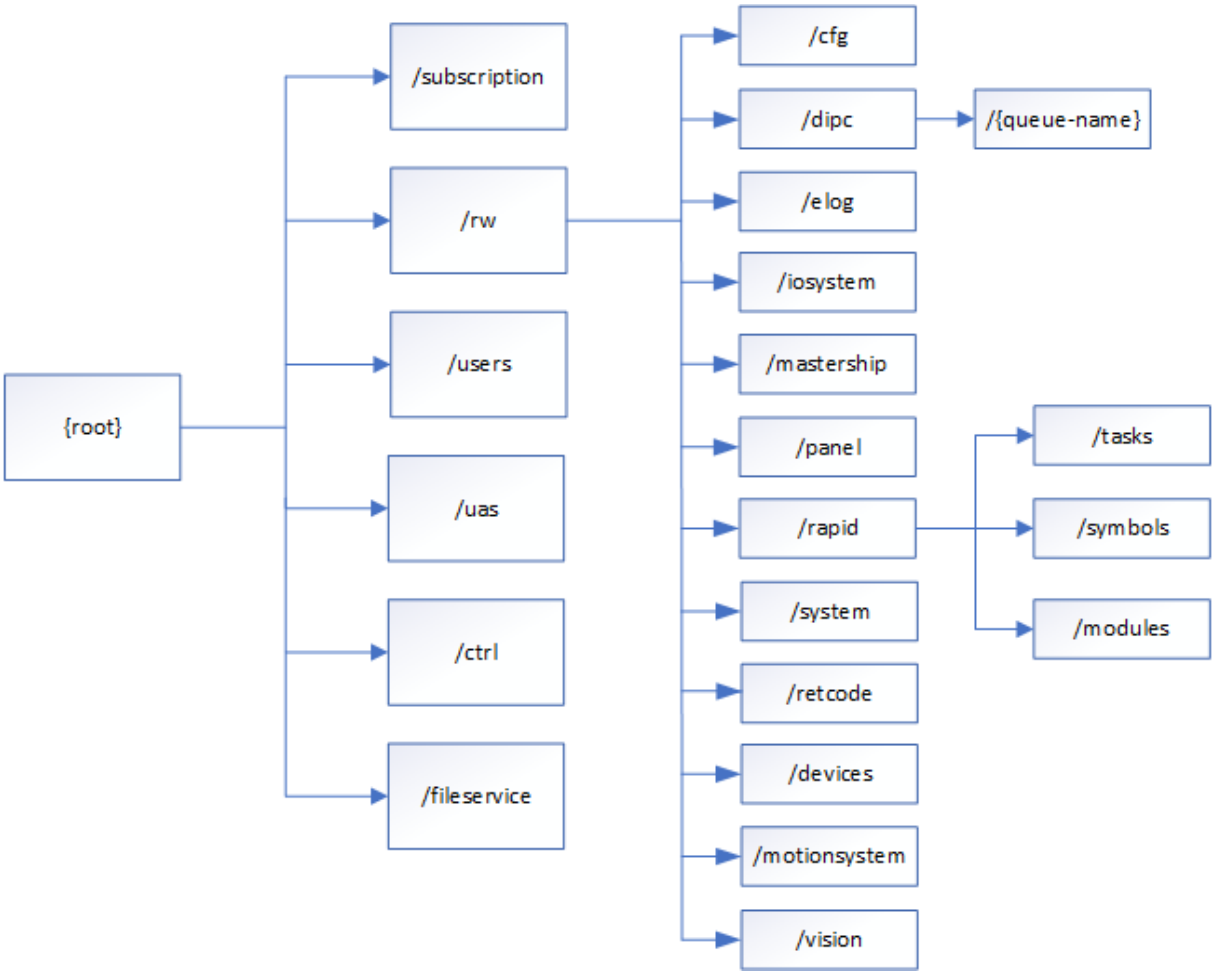
- GET: Lấy lại một tài nguyên
- PUT: Tạo hoặc cập nhật tài nguyên
- POST: Cập nhật tài nguyên
- DELETE: Xóa một tài nguyên
- OPTIONS: Phương pháp được hỗ trợ bởi một nguồn lực

Trong đó:

- GET và OPTIONS không làm thay đổi trạng thái tài nguyên.
- PUT, POST, DELETE sẽ làm thay đổi trạng thái tài nguyên.

Robot Web Services hỗ trợ giao thức Websockets được bảo mật để gửi các thay đổi trạng thái cho người dùng

1.4.2. Các dịch vụ chính của Robot Web Services (RWS)



Hình 1.12. Các dịch vụ chính của Robot Web Service

User Service: Quản lý xác thực người dùng, phiên đăng nhập, đăng xuất.

User Authentication Service (UAS): Quản lý người dùng, xác thực, phân quyền.

Robotware Service: Truy xuất, giám sát trạng thái robot; quản lý chương trình RAPID; điều khiển chuyển động; quản lý I/O; truy xuất User Panel; lấy thông tin phiên bản.

File Service: Quản lý file trên controller (upload, download, xóa).

Subscription Service: Đăng ký nhận thông báo thay đổi trạng thái (Websocket, long-polling).

Controller Service: Thực hiện các lệnh điều khiển cấp cao cho robot như điều khiển chuyển động (motion), tín hiệu I/O, và các thao tác hệ thống khác.

1.4.3. Đặc điểm giao tiếp

Robot Web Services (RWS) hỗ trợ giao tiếp qua cả HTTP (port 80) và HTTPS (port 443), trong đó HTTPS được khuyến nghị sử dụng nhằm đảm bảo tính bảo mật của quá trình trao đổi dữ liệu. Khi sử dụng HTTPS, bộ điều khiển robot cần được cấu hình với chứng chỉ SSL hợp lệ để thiết lập kết nối an toàn giữa máy khách và máy chủ.

RWS áp dụng cơ chế xác thực Basic Authentication thông qua phần tiêu đề (header) của gói tin HTTP/HTTPS. Thông tin xác thực bao gồm tên đăng nhập (username) và mật khẩu (password) của tài khoản đã được cấu hình trên bộ điều khiển robot, đảm bảo rằng chỉ những người dùng có quyền mới có thể truy cập và thao tác trên robot.

Cấu trúc cơ bản của một lệnh RWS:

- URL**:** https://{ip}:{port}/ kèm endpoint cụ thể của API.
- Phương thức HTTPS: Có thể dùng GET để đọc giá trị từ robot, POST để ghi giá trị đến robot.
- Headers: Xác định cách mã hóa dữ liệu gửi và nhận, cùng thông tin xác thực. Bao gồm Content-Type, Accept và Authorization.
- Data: Được sử dụng trong các phương thức như POST hoặc PUT, dữ liệu thường được định dạng dưới dạng XML hoặc form urlencoded tùy theo yêu cầu của API

Khi nhận phản hồi từ bộ điều khiển robot, dữ liệu trả về có thể ở định dạng XML hoặc JSON, tùy thuộc vào giá trị được khai báo trong trường Accept của header. Đối với các lệnh yêu cầu gửi dữ liệu trong phần thân (body), định dạng XML thường được yêu cầu nhằm đảm bảo tính nhất quán và khả năng phân tích cú pháp trên hệ thống robot.

1.5. Giới thiệu về các thư viện sử dụng

Trong hệ thống này, chúng em sử dụng ngôn ngữ lập trình Python cùng với một số thư viện hỗ trợ để giao tiếp và điều khiển robot, xử lý dữ liệu và xây dựng giao diện.

1.5.1. PyQt5 – Bộ công cụ giao diện Python dựa trên Qt5

PyQt5 là thư viện Python do Riverbank Computing phát triển, đóng vai trò là wrapper cho Qt5. Nó cung cấp hơn 620 lớp, hỗ trợ đầy đủ tính năng từ Qt Designer — từ các thành phần giao diện cơ bản (nút, menu, hộp thoại) đến các hệ thống phức tạp như xử lý sự kiện, đồ họa 2D/3D, hiển thị ảnh/video, đa luồng và mạng. Với PyQt5,

lập trình viên có thể xây dựng ứng dụng GUI chuyên nghiệp bằng Python, đồng thời tận dụng sức mạnh của Qt — lý tưởng cho các lĩnh vực như AI, xử lý ảnh và điều khiển robot.

Tính năng chính:

- Thiết kế giao diện với nhiều widget, hỗ trợ Qt Designer kéo-thả.
- Signal-slot rõ ràng, dễ tách biệt giao diện và logic.
- Đa luồng, giao diện mượt khi xử lý song song.
- Dễ tích hợp OpenCV, requests, websocket…
- Chạy trên nhiều hệ điều hành (Windows, Linux, macOS).

Ứng dụng PyQt5 trong đồ án:

- Xây dựng giao diện chính của hệ thống để giám sát và điều khiển robot.
- Hiển thị hình ảnh từ camera và kết quả nhận diện từ MediaPipe.
- Gửi và nhận dữ liệu điều khiển thông qua Robot Web Services (RWS) sử dụng giao thức HTTP/HTTPS và Externally Guided Motion (EGM) sử dụng giao thức UDP.
- Thiết kế giao diện bằng Qt Designer giúp rút ngắn thời gian phát triển, dễ bảo trì, dễ mở rộng khi cần bổ sung chức năng mới.

1.5.2. OpenCV – Thư viện xử lý ảnh và thị giác máy tính

OpenCV là thư viện mã nguồn mở do Intel phát triển từ năm 2000, chuyên xử lý ảnh, video và thị giác máy. Hỗ trợ đa nền tảng (Windows, Linux, macOS, Android, iOS) và đa ngôn ngữ (C++, Python, Java...), OpenCV cung cấp hàng nghìn hàm xử lý như chuyển đổi màu, làm mịn, phát hiện cạnh, theo dõi chuyển động và giao tiếp camera real-time.

Tính năng chính:

- Xử lý ảnh: làm mờ, lọc, phát hiện cạnh (Canny), biến đổi hình học...
- Xử lý video: đọc/ghi video, trích xuất khung hình...
- Phát hiện vật thể: face detection, object tracking...
- Kết hợp với Deep Learning: DNN module hỗ trợ chạy các model như YOLO, SSD...

Ứng dụng trong đồ án:

- Tiền xử lý ảnh từ webcam: chuyển đổi màu, làm mịn, cắt vùng quan tâm trước khi đưa vào MediaPipe Hands.

1.5.3. MediaPipe (Google):

MediaPipe là framework mã nguồn mở của Google (2019), tối ưu CPU/GPU, hỗ trợ nhiều nền tảng (C++, Python, Android, iOS, Web). MediaPipe cung cấp nhiều mô-đun có model huấn luyện sẵn, hoạt động real-time, chính xác cao.

Một trong những mô-đun mạnh nhất của MediaPipe là MediaPipe Hands, có khả năng phát hiện vị trí lòng bàn tay bằng mô hình CNN. Dự đoán tọa độ 3D của 21 điểm xương trên bàn tay (gọi là landmarks). Theo dõi liên tục khung hình để giảm độ trễ.

Hình 1.13. Cấu trúc Pipeline Mediapipe Hands

Ứng dụng trong đồ án:

- Phát hiện và theo dõi bàn tay qua webcam, dựa vào 21 điểm landmark để nhận diện cử chỉ.
- Cử chỉ mở tay: gửi lệnh mở gripper; cử chỉ nắm tay: gửi lệnh đóng gripper và nhiều cử chỉ khác
- Đảm bảo điều khiển real-time, đáp ứng nhanh, chính xác.

1.6. Giới thiệu về phần mềm sử dụng

1.6.1. Robot Studio

Phần mềm Robot Studio là một công cụ phát triển và mô phỏng robot do ABB phát triển. Nó được thiết kế đặc biệt để hỗ trợ việc lập trình, mô phỏng và điều khiển các robot của ABB. Robot Studio cung cấp một môi trường tích hợp cho các nhà phát triển robot để tạo, mô phỏng và tối ưu hóa các ứng dụng robot. Nó cho phép người dùng thiết kế các chương trình điều khiển robot bằng cách sử dụng ngôn ngữ lập trình Rapid, mô phỏng các hành động của robot trong môi trường 3D, và kiểm tra các chương trình trước khi triển khai thực tế trên robot thực.

1.6.2. Visual Studio Code

Visual Studio Code (VS Code) là một môi trường phát triển mã nguồn mở, nhẹ và mạnh mẽ, được phát triển bởi Microsoft. Trong hệ thống này, VS Code được sử dụng để lập trình và phát triển các ứng dụng điều khiển robot, bao gồm các đoạn mã Python giao tiếp với Robot Web Services (RWS) và Externally Guided Motion (EGM). Visual Studio Code hỗ trợ nhiều tính năng như gợi ý mã, kiểm tra cú pháp, quản lý dự án và tích hợp các công cụ quản lý phiên bản (như Git), giúp tăng hiệu quả lập trình và dễ dàng bảo trì mã nguồn.

1.6.3. Arduino IDE

Arduino IDE là một môi trường phát triển tích hợp đơn giản, được sử dụng để viết, biên dịch và nạp chương trình cho các vi điều khiển như ESP32. Trong hệ thống này, Arduino IDE hỗ trợ lập trình vi điều khiển ESP32 để thu thập dữ liệu từ các cảm biến (MPU9050, QMC-5883L) và truyền dữ liệu về hệ thống điều khiển robot. Arduino IDE có giao diện trực quan, dễ sử dụng, phù hợp cho cả người mới bắt đầu và các kỹ sư phát triển hệ thống nhúng.

Kết luận chương 1

Chương đã trình bày các cơ sở lý thuyết và tổng quan công nghệ làm nền tảng cho việc xây dựng hệ thống điều khiển và giám sát robot ABB CRB15000. Bên cạnh đó, giao thức EGM (Externally Guided Motion) đã được phân tích như một giải pháp điều khiển chuyển động ngoài, có độ phản hồi nhanh và độ trễ thấp, phù hợp với các ứng dụng điều khiển theo thời gian thực. Các công cụ và nền tảng phần mềm như Python, Visual Studio Code, và đặc biệt là RobotStudio – môi trường lập trình, mô phỏng và tối ưu hóa robot ABB cũng đã được đề cập. Sự kết hợp giữa Robot Web Services và lập trình python trên Visual Studio Code giúp xây dựng giao diện tương tác mạnh mẽ giữa người dùng và robot. Qua chương này, chúng ta đã tiến hành phân tích tổng quan về mô hình hệ thống, xác định rõ yêu cầu thiết kế và nguyên lý hoạt động của các thành phần trong mô hình hệ thống điều khiển giám sát và thu thập dữ liệu. Đồng thời, việc lựa chọn phần cứng phù hợp cũng đã được cân nhắc kỹ lưỡng để đáp ứng yêu cầu thực tế, đảm bảo tính ổn định, hiệu quả và khả năng mở rộng cho hệ thống. Kiến trúc tổng thể của hệ thống điều khiển robot bằng EGM đã được mô tả chi tiết. Những cơ sở lý thuyết và thực tiễn được trình bày trong chương này sẽ là nền tảng quan trọng để tiến hành các bước tiếp theo trong quá trình thiết kế, xây dựng và triển khai hệ thống ở các chương sau.

CHƯƠNG 2: THIẾT KẾ PHẦN CỨNG, ƯỚC LƯỢNG ĐỊNH HƯỚNG TAY NGƯỜI VÀ TRUYỀN DỮ LIỆU THỜI GIAN THỰC

2.1. Thiết kế box đeo tay định hướng

Để đảm bảo việc thu thập dữ liệu chuyển động tay người được chính xác và ổn định, một bộ (box) đeo tay chuyên dụng đã được nhóm thiết kế nhằm cố định các cảm biến lên tay người dùng trong quá trình vận hành. Thiết kế này không chỉ giúp cảm biến giữ nguyên vị trí tương đối theo thời gian, mà còn hạn chế tối đa rung động ngoài ý muốn, từ đó cải thiện chất lượng tín hiệu đầu vào cho hệ thống xử lý.

Box đeo tay có kích thước 36 x 61 x 25 mm, nhỏ gọn và phù hợp để đeo ở cổ tay hoặc mu bàn tay mà không gây vướng víu hay khó chịu cho người sử dụng trong quá trình làm việc. Kích thước này đủ để tích hợp toàn bộ các thành phần như mạch chứa vi điều khiển ESP32, cảm biến MPU9250, QMC5883L, pin, mạch nguồn và mạch nạp, đồng thời đảm bảo tính thẩm mỹ, thuận tiện khi đeo lâu dài và không ảnh hưởng đến thao tác tự nhiên của người vận hành.

Hình 2.1. Hình dạng Box đeo tay

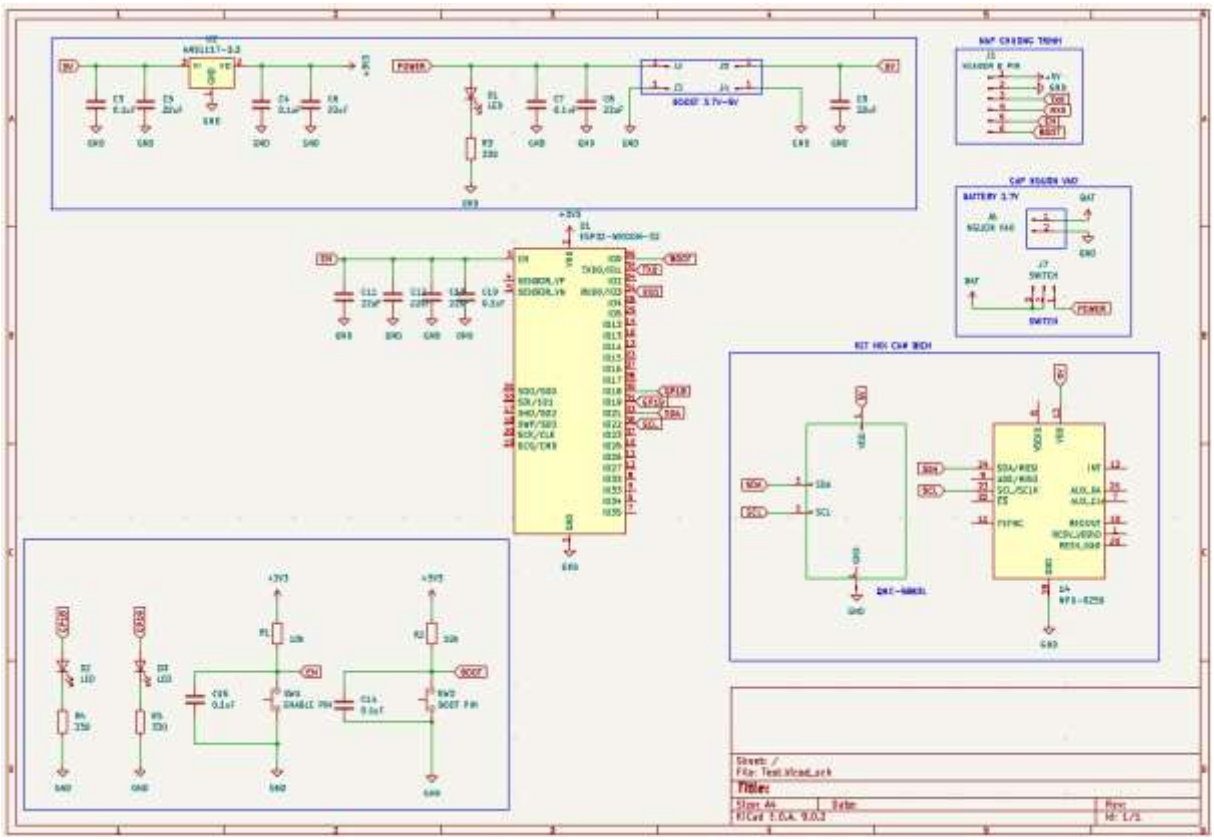
2.1.1. Mạch nguyên lí lấy tín hiệu từ cảm biến

Mạch nguyên lý được thiết kế nhằm thu thập tín hiệu từ các cảm biến và cung cấp nguồn phù hợp cho các module trong hệ thống. Bảng 2.1 liệt kê các linh kiện chính được sử dụng trong mạch, và sơ đồ nguyên lý được trình bày bên dưới để minh họa mối liên kết giữa các linh kiện này.

Bảng 2.1. Danh sách các linh kiện

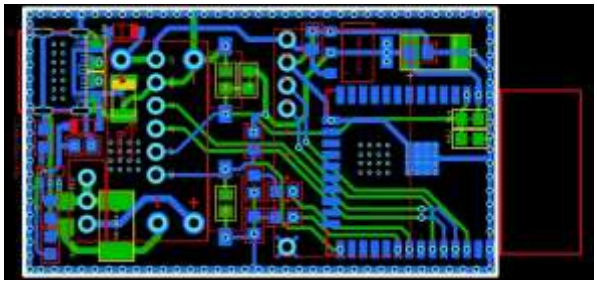
STT	Tên sản phẩm
1	Module ESP32-WROOM 32D
2	Module gia tốc góc MPU-9250
3	Module la bàn QMC-5883L
4	Mạch tăng áp 3.7V lên 5V/8V/9V/12V 0.3A
5	Ic ổn áp AMS1117-3V3 - SMD
6	Nút nhấn 2 chân SMD 3x6x2.5
7	Công tắc gạt 3 chân mini-2.54mm

- Sơ đồ nguyên lí:



Hình 2.2. Sơ đồ mạch nguyên lí

2.1.2. Mạch PCB hoàn chỉnh:



Hình 2.3. Sơ đồ mạch PCB

- Box đeo tay và mạch điều khiển sau khi hoàn thành:

Hình 2.4. Box đeo tay và mạch hoàn chỉnh

2.2. Bộ lọc nhiễu trong quá trình tính toán ước lượng

Bộ lọc Kalman là một công cụ toán học được phát triển bởi Rudolph E. Kalman vào năm 1960 để thực hiện phép ước tính của một biến quan sát bằng cách sử dụng một nhóm các phương trình dự báo – cập nhật (prediction – update). Nếu nhiễu có đặc trưng là nhiễu Gaussian và vấn đề là bộ lọc chỉ có thể mô tả theo dạng tuyến tính, thì bộ lọc ước lượng Kalman sẽ được coi là giải pháp tối ưu trong việc tính toán truy hồi kết quả phép ước đoán trạng thái của một quá trình sao cho trung bình phương sai của độ lệch (giữa giá trị thực và giá trị ước đoán) là nhỏ nhất. Trong trường hợp hệ phi tuyến, bộ lọc Kalman phải được tuyến tính hóa và giải pháp này được xem là bộ ước lượng trạng thái đặc biệt.

Kalman Filter (KF) là một thuật toán ước lượng trạng thái của một hệ thống động trong môi trường có nhiễu. KF gốc chỉ áp dụng được cho các hệ tuyến tính. Extended Kalman Filter (EKF) mở rộng Kalman Filter để áp dụng cho hệ phi tuyến, bằng cách xấp xỉ tuyến tính các hàm phi tuyến thông qua khai triển Taylor cấp 1 (Jacobian).

EKF gồm 2 bước chính:

- Dự đoán (Prediction): Dự đoán trạng thái tiếp theo từ trạng thái hiện tại và mô hình hệ thống.
- Cập nhật (Update): Sử dụng đo lường thực tế từ cảm biến để hiệu chỉnh lại dự đoán.

Trong EKF, vì hệ thống là phi tuyến nên các phương trình trạng thái và đo lường không còn tuyến tính mà thường được mô tả bằng các hàm phi tuyến. Do đó, tại mỗi bước thời gian, bộ lọc EKF sẽ tuyến tính hóa các phương trình này bằng cách sử dụng khai triển Taylor cấp một xung quanh trạng thái dự đoán hiện tại. Cụ thể, EKF tính các ma trận Jacobian tương ứng với các hàm phi tuyến để thay thế cho các ma trận cố định như trong bộ lọc Kalman tuyến tính.

Mô hình toán học phi tuyến trong EKF gồm:

Phương trình trạng thái:

$$x^k = f(x_{k-1}, u_k) + w_k \tag{2.1}$$

Trong đó:

x^k là trạng thái thời điểm k

u_{k-1} là đầu vào điều khiển

w_k và v_k là nhiễu hệ thống và nhiễu đo lường, thường giả định có phân phối Gaussian với hiệp phương sai lần lượt là Q và R

Phương trình đo lường:

$$z_k = h(x_k) + v_k \tag{2.2}$$

Trong đó:

z_k: Vector đo lường từ cảm biến tại thời điểm kk.

$h(x_k)$: Hàm phi tuyến (hoặc tuyến tính) ánh xạ trạng thái x_k sang không gian đo lường.

v_k: Nhiễu đo lường, phân phối Gaussian với hiệp phương sai R

$v_k \sim N(o, R)$

Dự đoán (Prediction):

Dự đoán trạng thái tiếp theo:

$$x_{k|k-1} = f(x_{k-1,u_k}) \tag{2.3}$$

Ma trận Jacobian của hàm trạng thái:

$$F_k = \frac{\partial f}{\partial x}\bigg|_{x_{k-1,u_k}} \tag{2.4}$$

Cập nhật ma trận hiệp phương sai:

$$P_{k|k-1} = F_k . P_{k-1} . F_k^T + Q_k \tag{2.5}$$

Cập nhật (Correction):

Ma trận Jacobian của hàm đo lường:

$$H_k = \frac{\partial h}{\partial x}\bigg|_{x_{k|k-1}} \tag{2.6}$$

Tính trọng số Kalman:

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \tag{2.7}$$

Cập nhật trạng thái ước lượng:

$$x_k = x_{k|k-1} + K_k (z_k - h(x_{k|k-1})) \tag{2.8}$$

Cập nhật ma trận hiệp phương sai:

$$P_k = (1 - K_k H_k) P_{k|k-1} \tag{2.9}$$

Nhờ cơ chế tuyến tính hóa này, bộ lọc Kalman mở rộng có thể áp dụng được trong các hệ thống mà các mối quan hệ giữa trạng thái và đo lường không tuyến tính. Tuy nhiên, do chỉ sử dụng khai triển bậc nhất nên sai số có thể tích lũy nếu hệ phi tuyến mạnh hoặc ước lượng ban đầu không chính xác. Dù vậy, EKF vẫn là công cụ phổ biến trong các ứng dụng như dẫn đường quán tính, định vị GPS, robot tự hành và các hệ thống điều khiển nhúng có yêu cầu cao về độ chính xác.

Ứng dụng:

- Xác định tư thế (pose estimation) cho robot di động
- Theo dõi quỹ đạo tên lửa, UAV
- Ước lượng trạng thái hệ thống cảm biến quán tính (IMU)
- Hệ thống định vị (GPS + IMU fusion)

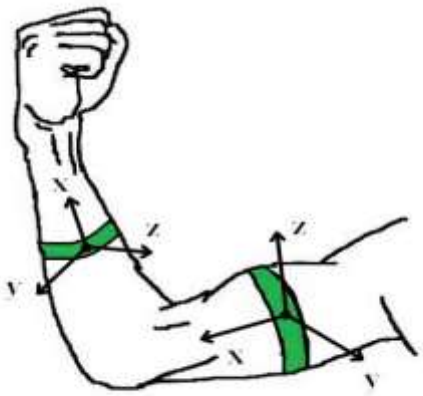
2.3. Tính toán và ước lượng định hướng tay người bằng IMU

IMU tích hợp gồm gia tốc kế (accelerometer), con quay hồi chuyển (gyroscope) và thường có thêm từ kế (magnetometer) để đo các đại lượng chuyển động. Cảm biến gia tốc đo gia tốc tuyến tính theo ba trục, cho biết độ nghiêng tĩnh so với trọng trường,

con quay hồi chuyển đo tốc độ góc quay của thiết bị, từ kế đo cường độ trường từ Trái Đất để xác định hướng. Thông tin từ các cảm biến này cho phép xác định tư thế (roll/pitch/yaw) của thiết bị trong không gian 3 chiều.

Ở hệ thống này, các góc chênh lệch của từng cánh tay người so với vị trí chuẩn và chiều dài cánh tay được giả định là như nhau tại các vị trí đã được xác định (gán box). Giả định này giúp đơn giản hóa quá trình tính toán động học và xác định quỹ đạo chuyển động của tay người, đồng thời đảm bảo tính nhất quán khi so sánh và ánh xạ chuyển động từ người sang robot. Việc gán box tại các vị trí cố định còn hỗ trợ việc thiết lập hệ tọa độ tham chiếu, qua đó nâng cao độ chính xác trong quá trình điều khiển.

Dùng cảm biến đầu tiên (gắn ở bắp tay) làm mốc gốc tọa độ. Cảm biến thứ hai (gắn ở cánh tay) sẽ cung cấp dữ liệu vị trí tương đối trong không gian.



Hình 2.5. Vị trí đeo box trên tay

Gyroscope cung cấp vận tốc góc $(\omega_x, \omega_y, \omega_z)$ cho bước dự đoán của bộ lọc EKF. Kết quả quaternion dự đoán bước đầu được chuẩn hóa. Accelerometer cung cấp gia tốc (a_x, a_y, a_z) cho bước hiệu chỉnh, nơi quaternion dự đoán đã chuẩn hóa trước đó được hiệu chỉnh dựa trên sai số giữa gia tốc đo được và vector trọng lực dự đoán. Quaternion sau hiệu chỉnh được chuẩn hóa và từ đó các góc roll và pitch được tính toán bằng công thức chuyển đổi Euler.

Quaternion biểu diễn hướng của thiết bị được ký hiệu là:

$$q = [q_0, q_1, q_2, q_3]^T \tag{2.10}$$

Để giảm nhiễu và sai số trôi dạt, các tín hiệu từ đa cảm biến được kết hợp bằng bộ lọc Kalman mở rộng (EKF), giúp tối ưu cho hệ thống phi tuyến bằng cách tuyến tính hóa xung quanh nghiệm dự đoán. Trong EKF, trạng thái bao gồm các đại lượng tư thế (thường là quaternion hoặc ba góc Euler) và có thể bổ sung thêm sai số lệch (bias) của con quay hồi chuyển. Mô hình tiến triển trạng thái dựa trên tích phân tín hiệu con quay, trong khi mô hình đo lường sử dụng tín hiệu từ gia tốc kế và từ kế. Phương trình động học liên tục:

$$\dot{q} = \frac{1}{2}\Omega q \ \ [12] \tag{2.11}$$

Trong đó Ω là ma trận vận tốc góc:

$$\Omega = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix} \tag{2.12}$$

Ta có khai triển Taylor bậc 1 với chu kỳ lấy mẫu T_s như sau:

$$A = e^{\Omega T_s} \cong I + A.T_s + \frac{1}{2!}.(A.T_s)^2 + \frac{1}{3!}.(A.T_s)^3 + ... \tag{2.13}$$

Sử dụng khai triển trên ta được:

$$q_k = (I + \frac{\Omega_k T_s}{2})q_{k-1} + W_k \tag{2.14}$$

Với $W_k \sim N(0,Q)$ là nhiễu quá trình Gaussian, có nguồn gốc từ sai số mô hình động học và nhiễu cảm biến con quay. Công thức ma trận hiệp phương sai Q :

$$Q = \begin{bmatrix} \sigma_{\omega_x}^2 & 0 & 0 & 0 \\ 0 & \sigma_{\omega_y}^2 & 0 & 0 \\ 0 & 0 & \sigma_{\omega_z}^2 & 0 \\ 0 & 0 & 0 & \sigma_{\omega_z}^2 \end{bmatrix} \tag{2.15}$$

Trong đó :

- $\sigma_{\omega_x}^2, \sigma_{\omega_y}^2, \sigma_{\omega_z}^2$: Độ lệch chuẩn của nhiễu gyroscope trên các trục (rad/s)
- T_s : chu kỳ lấy mẫu

Mô hình đo lường cho vector đo lường y_k gồm gia tốc hiệu chỉnh a_g và góc lệch ψ:

$$\begin{bmatrix} a_x \\ a_y \\ a_z \\ \psi \end{bmatrix} = \begin{bmatrix} 2(q_1q_3 - q_0q_2) \\ 2(q_2q_3 + q_0q_1) \\ q_0^2 - q_1^2 - q_2^2 + q_3^2 \\ a\tan 2(\dfrac{2(q_1q_2 + q_3q_0)}{q_0^2 + q_1^2 - q_2^2 - q_3^2} \end{bmatrix} + \vec{v} \tag{2.16}$$

Công thức này có thể được viết lại ngắn gọn như sau :

$$y_k = g(q_k) + v_k = \begin{bmatrix} 2(q_1q_3 - q_0q_2) \\ 2(q_2q_3 + q_0q_1) \\ q_0^2 - q_1^2 - q_2^2 + q_3^2 \\ a\tan 2(\dfrac{2(q_1q_2 + q_3q_0)}{q_0^2 + q_1^2 - q_2^2 - q_3^2} \end{bmatrix} + v_k \tag{2.17}$$

Với $v_k \sim N(0, R)$ là nhiễu đo lường là nhiễu đo lường Gaussian, xuất phát từ sai số cảm biến gia tốc và từ kế. Công thức ma trận hiệp phương sai R :

$$R = \begin{bmatrix} \sigma_{a_x}^2 & 0 & 0 & 0 \\ 0 & \sigma_{a_y}^2 & 0 & 0 \\ 0 & 0 & \sigma_{a_z}^2 & 0 \\ 0 & 0 & 0 & \sigma_{\psi}^2 \end{bmatrix} \tag{2.18}$$

Trong đó :

- $\sigma_{a_x}^2, \sigma_{a_y}^2, \sigma_{a_z}^2$: Độ lệch chuẩn nhiễu gia tốc(m/s^2)
- σ_{ψ}^2 : Độ lệch chuẩn nhiễu góc lệch (rad)

Từ các công thức trên, chúng ta có được quy trình hiệu chỉnh bởi EKF như sau:

Bảng 2.2. Quy trình hiệu chỉnh bởi EKF

Bước	Công thức
Dự đoán trạng thái	$\hat{q}_{\bar{k}} = (I + \dfrac{\Omega_k T_s}{2})\hat{q}_{k-1}$
Dự đoán hiệp phương sai	$P_{\bar{k}} = A_k P_{k-1} A_k^T + Q$
Tính hệ số Kalman	$K_k = P_{\bar{k}} G_k^T (G_k P_{\bar{k}} G_k^T + R)^{-1}$
Cập nhật trạng thái	$\hat{q}_k = \hat{q}_{\bar{k}} + K_k(y_k - g(\hat{q}_{\bar{k}}))$
Cập nhật hiệp phương sai	$P_k = (I - K_k G_k) P_{\bar{k}}$

Trong đó: P_k: Ma trận hiệp phương sai

Q: Ma trận hiệp phương sai nhiễu quá trình

A_k : Jacobian của mô hình trạng thái

G_k : Jacobian của $\{g(\hat{q}_{\bar{k}})\}$ đạo hàm riêng theo $\hat{q}$

R : Ma trận hiệp phương sai nhiễu đo lường

Sau khi ước lượng quaternion, có thể chuyển đổi sang góc Euler bằng công thức:

$$\begin{aligned}\phi &= a\tan 2\left(\frac{2(q_0q_1+q_3q_2)}{q_0^2-q_1^2-q_2^2+q_3^2}\right)\\ \theta &= \arcsin(2(q_0q_2-q_1q_3))\\ \psi &= a\tan 2\left(\frac{2(q_1q_2+q_3q_0)}{q_0^2+q_1^2-q_2^2-q_3^2}\right)\end{aligned} \tag{2.19}$$

Góc Euler thu được có thể chuyển đổi thành ma trận quay theo quy ước ZYX như sau:

$$R = R_z(\psi)R_y(\phi)R_x(\theta) \tag{2.20}$$

Các thành phần của ma trận quay được khai triển cụ thể như sau :

$$\begin{bmatrix}\cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi-\sin\psi\cos\phi & \cos\psi\sin\theta\cos\phi+\sin\psi\sin\phi\\ \sin\psi\cos\phi & \sin\psi\sin\theta\sin\phi+\cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi-\cos\psi\sin\phi\\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi\end{bmatrix} \tag{2.21}$$

Dựa trên ma trận quay thu được, vị trí cổ tay trong không gian được tính toán theo mô hình động học:

$$\overrightarrow{P_{wrist}} = R_{shoulder}.\overrightarrow{L_1} + R_{forearm}.\overrightarrow{L_2} \tag{2.22}$$

Các thành phần của vị trí cổ tay lần lượt được triển khai:

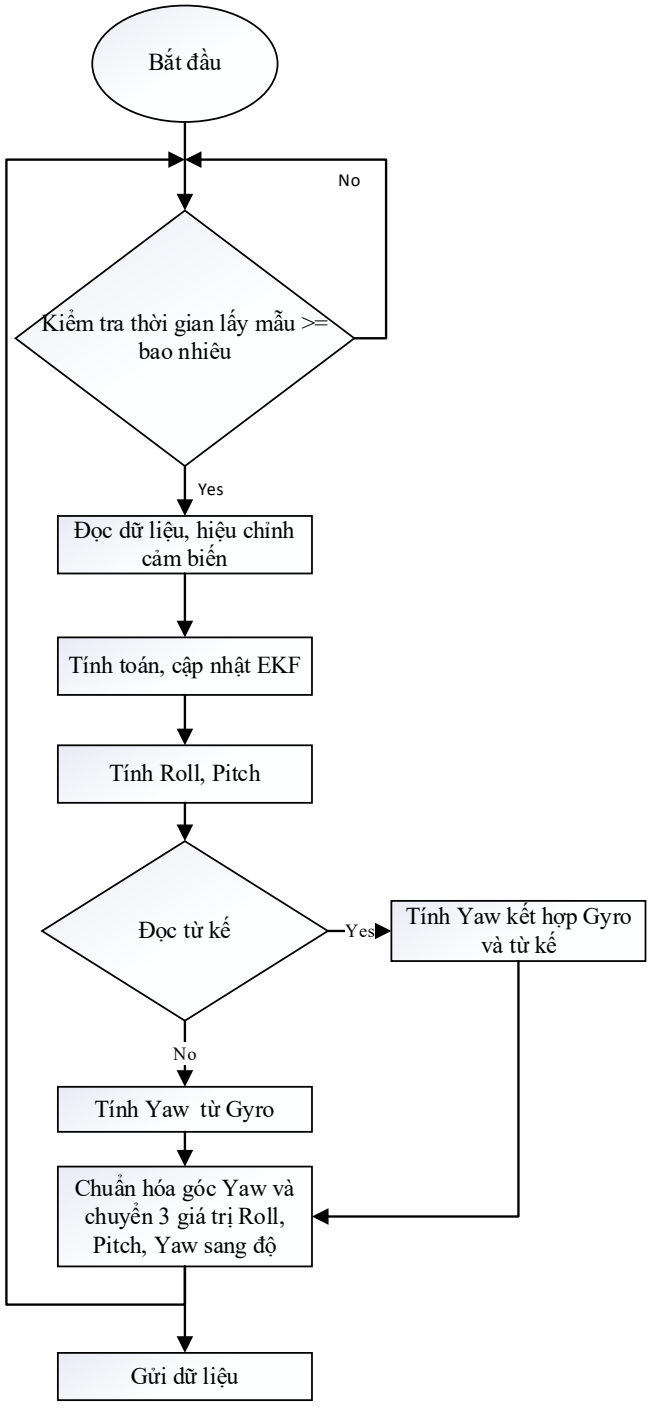
$$\begin{bmatrix}P_x\\P_y\\P_z\end{bmatrix} = \begin{bmatrix}R_{11}^s & R_{12}^s & R_{13}^s\\ R_{21}^s & R_{22}^s & R_{23}^s\\ R_{31}^s & R_{32}^s & R_{33}^s\end{bmatrix}\begin{bmatrix}L_1\\0\\0\end{bmatrix} + \begin{bmatrix}R_{11}^f & R_{12}^f & R_{13}^f\\ R_{21}^f & R_{22}^f & R_{23}^f\\ R_{31}^f & R_{32}^f & R_{33}^f\end{bmatrix}\begin{bmatrix}L_2\\0\\0\end{bmatrix} \tag{2.23}$$

Tính toán thành phần tọa độ :

$$\begin{aligned}&\text{Tọa độ X: } P_x = (R_{12}^s.L_1) + (R_{12}^f.L_2)\\ &\text{Tọa độ Y: } P_y = (R_{22}^s.L_1) + (R_{22}^f.L_2)\\ &\text{Tọa độ Z: } P_z = (R_{32}^s.L_1) + (R_{32}^f.L_2)\end{aligned} \tag{2.24}$$

2.4. Lưu đồ thuật toán

2.4.1. Lưu đồ thu thập dữ liệu từ cảm biến MPU 9250

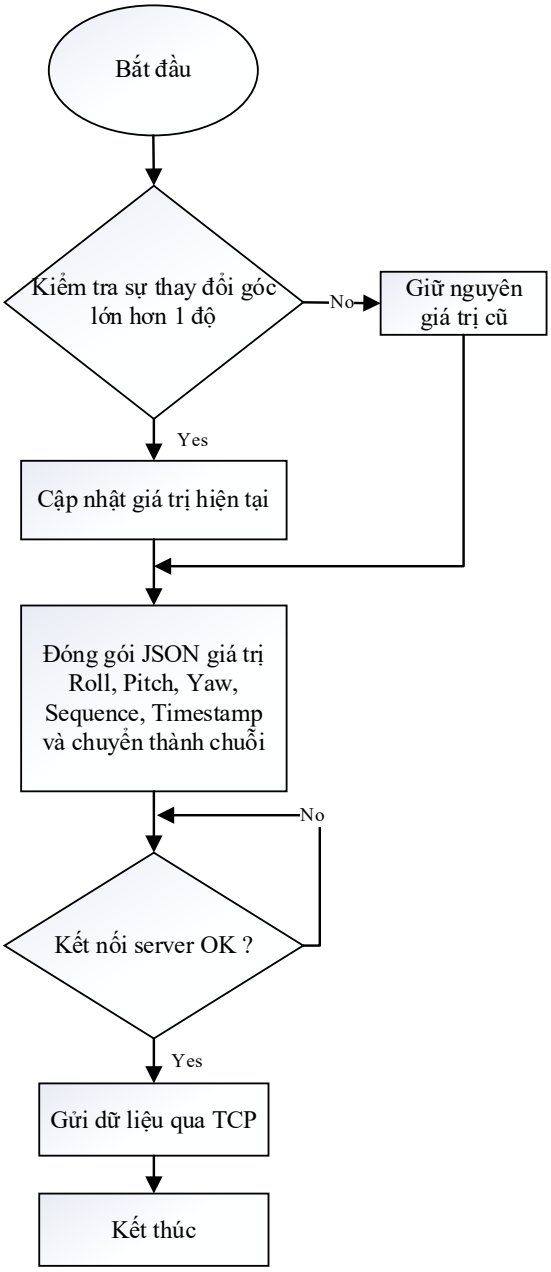


Hình 2.6. Lưu đồ thu thập dữ liệu từ MPU9250 và QMC5883L

Sơ đồ thuật toán trên mô tả quá trình xử lý dữ liệu từ cảm biến IMU bằng cách sử dụng bộ lọc Kalman mở rộng (EKF). Dữ liệu thu được từ cảm biến sẽ được hiệu

chỉnh và cập nhật vào bộ lọc EKF để ước lượng chính xác trạng thái quay của hệ thống. Sau khi chuyển đổi quaternions sang Euler, các góc Roll và Pitch được trích xuất từ EKF, trong khi góc Yaw được tính dựa trên Gyroscope kết hợp từ kế (nếu có). Hệ thống cũng thực hiện chuẩn hóa góc Yaw và gửi dữ liệu đầu ra phục vụ điều khiển robot. Thuật toán đảm bảo tính chính xác, ổn định và phản hồi thời gian thực trong quá trình điều khiển.

2.4.2. Lưu đồ đóng gói dữ liệu và gửi sang PC của ESP32



Hình 2.7. Lưu đồ đóng gói dữ liệu gửi sang PC

Hệ thống triển khai cơ chế truyền dữ liệu tối ưu cho thông tin định hướng IMU sau xử lý. Bắt đầu bằng việc kiểm tra sự thay đổi góc Euler, dữ liệu chỉ được xử lý tiếp nếu ít nhất một góc thay đổi lớn hơn 2 độ so với giá trị trước đó nhằm loại bỏ bớt các gói tin dư thừa khi vật thể ổn định. Nếu thay đổi không đạt ngưỡng, hệ thống duy trì giá trị 0 và bỏ qua truyền dữ liệu. Ngược lại, giá trị mới được cập nhật và đóng gói thành cấu trúc JSON chuẩn hóa bao gồm: Góc định hướng (Pitch, Roll, Yaw), số thứ tự gói (Sequence) phát hiện mất mát dữ liệu và Timestamp đồng bộ thời gian.

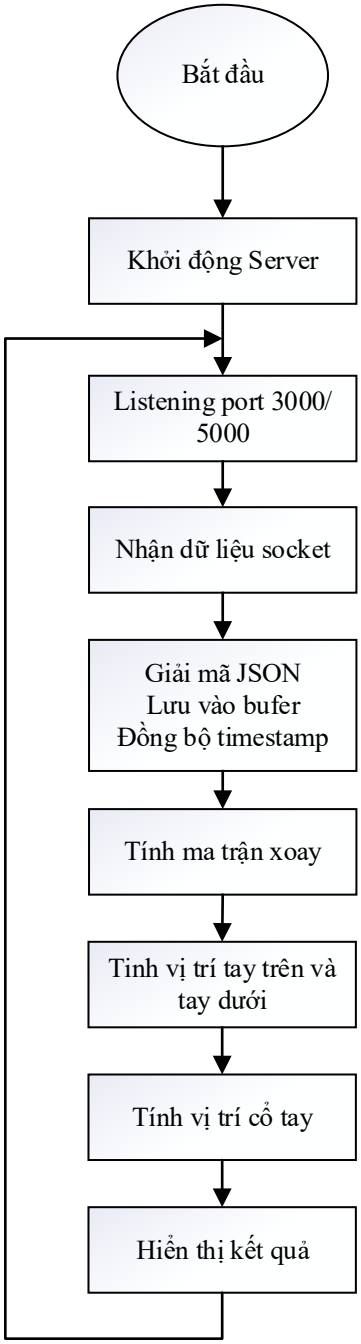
Gói JSON được chuyển đổi thành chuỗi byte trước khi hệ thống kiểm tra trạng thái kết nối TCP đến server. Chỉ khi kết nối hoạt động, dữ liệu mới được truyền đi qua giao thức hướng kết nối (TCP) đảm bảo tính toàn vẹn thông tin.

2.4.3. Lưu đồ nhận dữ liệu và xử lý toạ độ trên PC

Sau khi khởi động máy chủ, hệ thống lắng nghe kết nối từ IMU qua socket, nhận dữ liệu ở dạng JSON và lưu vào bộ đệm. Tiếp theo, hệ thống đồng bộ mốc thời gian nhằm đảm bảo tính nhất quán dữ liệu giữa các cảm biến.

Giải mã JSON và dữ liệ Roll, Pitch, Yaw thành ma trận xoay.

Hệ thống tiến hành xác định ma trận tương đối giữa các khung tham chiếu, từ đó ước lượng vị trí khớp khuỷu và cổ tay, đồng thời áp dụng các giới hạn vận động sinh học để đảm bảo chuyển động nằm trong phạm vi cho phép.



Hình 2.8. Lưu đồ nhận dữ liệu và xử lý tọa độ trên PC

Kết luận chương 2

Trong chương này, đã trình bày chi tiết nguyên lý hoạt động và ứng dụng của bộ lọc Kalman mở rộng (EKF) trong việc ước lượng định hướng của cảm biến MPU9250 và QMC5883L. Các bước xây dựng mô hình toán học, tuyến tính hóa phương trình phi tuyến thông qua khai triển Taylor, và quá trình dự đoán – cập nhật đã được làm rõ. Bên cạnh đó, chương cũng mô tả việc triển khai thuật toán EKF trên nền tảng ESP32 để xử lý dữ liệu cảm biến theo thời gian thực, bao gồm việc chuyển đổi quaternion sang góc Euler và truyền dữ liệu qua Wi-Fi theo định dạng JSON.

Phần cứng của hệ thống (ESP32 kết hợp MPU9250 và QMC5883L) đã được thiết kế và tích hợp hoàn chỉnh. Đặc biệt, một box đeo tay chuyên dụng với kích thước 36 x 61 x 25 mm đã được thiết kế nhằm cố định các thành phần phần cứng trên tay người dùng, đảm bảo cảm biến giữ nguyên vị trí tương đối, giảm thiểu rung động ngoài ý muốn, đồng thời tạo sự thoải mái, gọn nhẹ và thuận tiện khi đeo trong thời gian dài.

Kết quả thu được cho thấy khả năng áp dụng EKF hiệu quả trong việc theo dõi tư thế cánh tay người, tạo nền tảng cho các ứng dụng điều khiển robot hoặc thiết bị tương tác thông minh.

CHƯƠNG 3: THIẾT KẾ VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM GIAO TIẾP ĐIỀU KHIỂN ROBOT ABB SỬ DỤNG RWS

3.1. Lập trình giao tiếp PC và Robot thông qua RWS

3.1.1. Kiến trúc giao tiếp phần mềm với Robot ABB qua RWS

Trong hệ thống phần mềm điều khiển robot ABB CRB15000, giao tiếp giữa máy tính (PC) và robot được thực hiện thông qua Robot Web Services (RWS). Kiến trúc giao tiếp được thiết kế theo nguyên tắc mở, sử dụng giao thức web tiêu chuẩn nhằm đảm bảo khả năng mở rộng, tương thích đa nền tảng và dễ dàng tích hợp với các hệ thống phần mềm khác. Hệ thống gồm các thành phần kiến trúc.

Client (PC): Phần mềm điều khiển được phát triển bằng Python, sử dụng:

- Thư viện requests để tạo các HTTP request.
- Thư viện PyQt5 để xây dựng giao diện đồ họa giám sát và điều khiển robot.

Network Layer: Giao tiếp được thực hiện qua giao thức HTTPS (port 443), sử dụng Basic Authentication để xác thực và SSL/TLS để đảm bảo an toàn dữ liệu

Server (Robot Controller): REST API endpoints đây là các URL được client gọi để truy xuất trạng thái, điều khiển I/O, giám sát RAPID…

Quá trình giao tiếp giữa phần mềm PC và robot được thực hiện theo các bước:

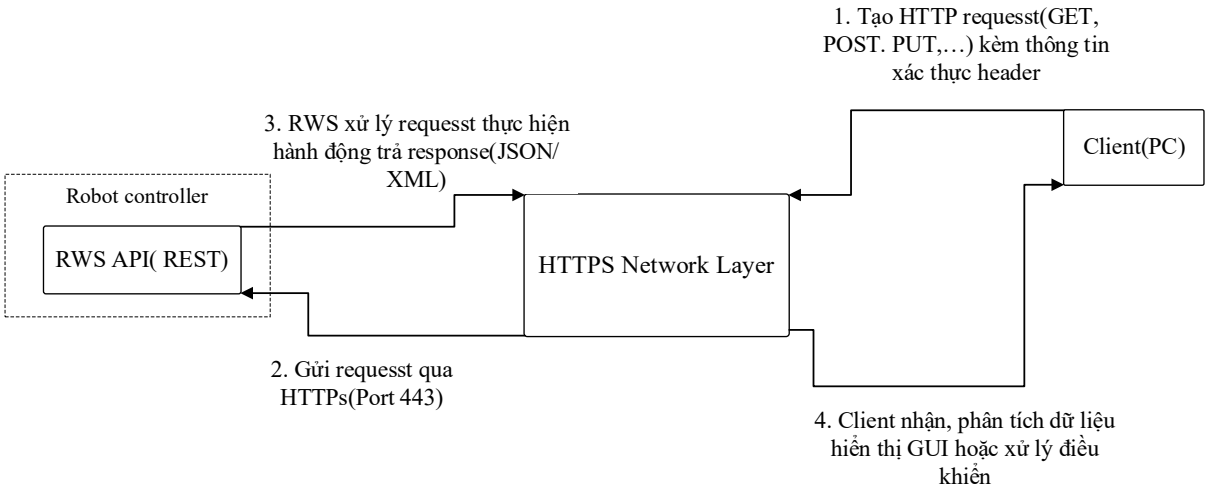
Bước 1: Phần mềm PC tạo request với phương thức HTTP phù hợp (GET, POST, PUT, DELETE) và thông tin xác thực trong header.

Bước 2: Request được gửi qua mạng dưới dạng HTTPS đến bộ điều khiển robot.

Robot controller thông qua RWS xử lý request và trả về kết quả dưới dạng JSON hoặc XML.

Bước 3: Phần mềm PC phân tích dữ liệu phản hồi và hiển thị kết quả trên giao diện người dùng hoặc thực hiện các hành động điều khiển tiếp theo.

Quy trình giao tiếp giữa chương trình Python trên PC và bộ điều khiển robot qua RWS API được thể hiện trực quan bằng lưu đồ thuật toán dưới đây :



Hình 3.1. Lưu đồ thuật toán giao tiếp Python với RWS API qua HTTPS

3.1.2. Các API chính được sử dụng trong hệ thống

3.1.2.1. Quản lý trạng thái của Panel Service

Bảng 3.1. Các hàm quản lý trạng thái bộ điều khiển

Chức năng	Phương thức HTTP	API Endpoint
Thiết lập và cập nhập trạng thái của bộ điều khiển.	POST, GET	/rw/panel/ctrl-state
Thiết lập và cập nhập chế độ vận hành (Auto/Man).	POST, GET	/rw/panel/opmode
Xác nhận chuyển chế độ (Bắt buộc khi sang chế độ auto)	POST	/rw/panel/opmode/acknowledge
Đặt tỷ lệ tốc độ robot (0-100%).	POST, GET	/rw/panel/speedratio
Thiết lập giải phóng quyền điều khiển (Mastership release)	POST	/rw/mastership/release
Thiết lập quyền điều khiển	POST	/rw/mastership/request

3.1.2.2. Thực thi chương trình RAPID

Bảng 3.2. Các hàm thực thi chương RAPID

Chức năng	Phương thức HTTP	API Endpoint
Bắt đầu thực thi chương trình RAPID.	POST	/rw/rapid/execution/start
Dừng thực thi chương trình RAPID.	POST	/rw/rapid/execution/stop
Reset con trỏ của chương trình Rapid.	POST	/rw/rapid/execution/resetpp
Cập nhập trạng thái thực thi chương trình RAPID (Start/Stop).	GET	/rw/rapid/execution

3.1.2.3. Quản lý I/O

Bảng 3.3. Các hàm quản lý I/O

Chức năng	Phương thức HTTP	API Endpoint
Thiết lập và cập nhật trạng thái I/O	POST, GET	/rw/iosystem/signals/{signal-path}

3.1.2.4. Đăng ký quyền người dùng

Bảng 3.4. API đăng ký quyền người dùng

Chức năng	Phương thức HTTP	API Endpoint
Đăng ký người dùng cục bộ, dùng để điều khiển chế độ vận hành.	POST	/users/register/local

3.1.2.5. Đăng ký tài nguyên

Bảng 3.5. Hàm chức năng đăng ký các thay đổi tài nguyên

Chức năng	Phương thức HTTP	API Endpoint
Đăng ký các thay đổi trên các tài nguyên khác nhau. Cập nhập thay đổi liên tục thông qua Websocket.	POST	/Subscription

Để đăng ký quyền theo dõi thay đổi tài nguyên, client gửi POST /subscription kèm thông tin tài nguyên cần giám sát. Server phản hồi với danh sách sự kiện ban đầu và Location header, client mở WebSocket theo URI trong Location và duy trì kết nối để nhận thông báo sự kiện liên tục.

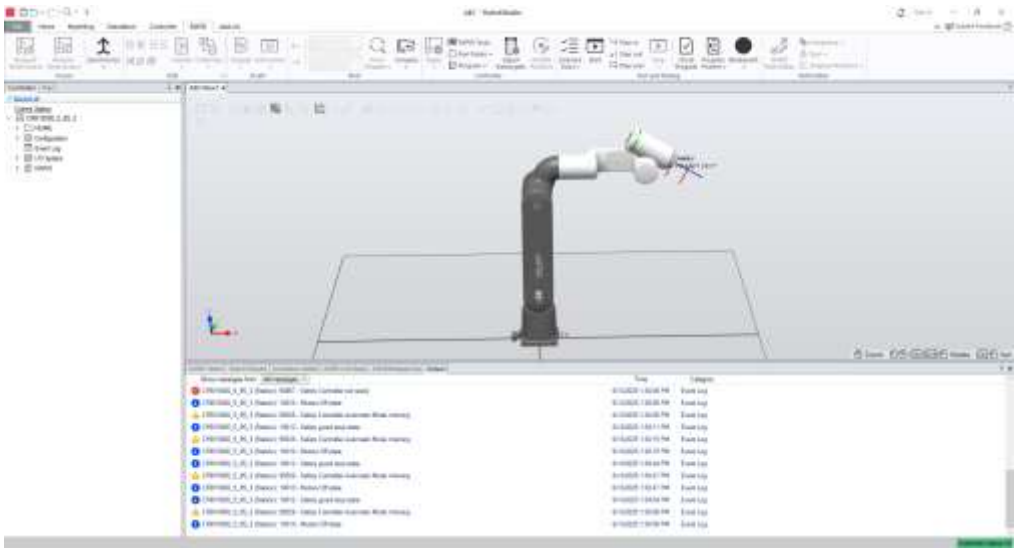
Mỗi kết nối WebSocket tương ứng với một nhóm subscription, có thể theo dõi tối đa 1000 tài nguyên với mức ưu tiên thấp (p=0) hoặc trung bình (p=1).

3.2. Thiết lập cấu hình và lập trình trên Robotstudio

3.2.1. Cấu hình EGM trên RobotStudio

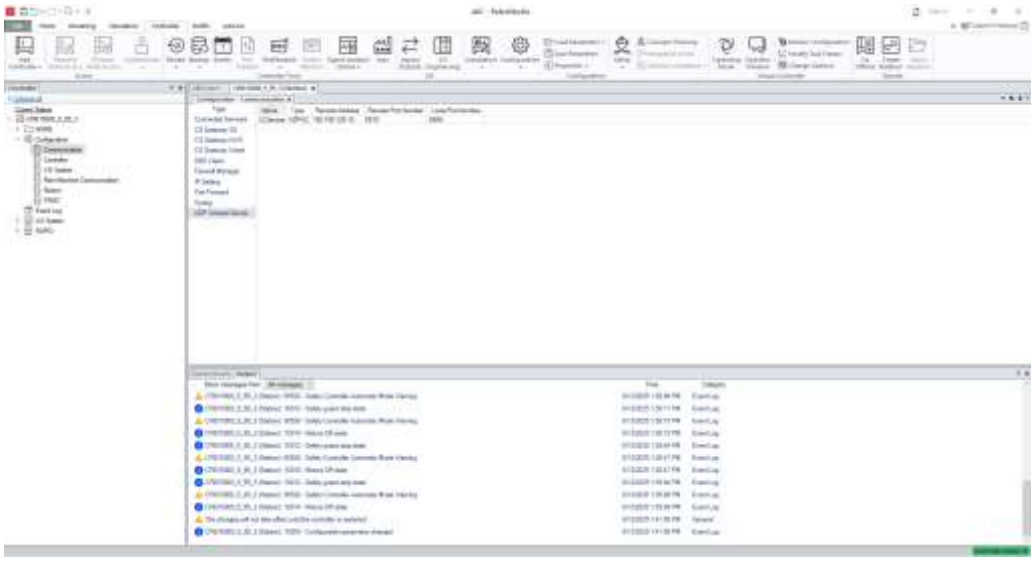
Các cấu hình cần thiết cho EGM :

Bước 1: Với bộ điều khiển thật tiến hành thêm bộ điều khiển bằng cách chọn vào tab Controller > Add Controller > One Click Connect.



Hình 3.2. Kết nối bộ điều khiển Robot

Bước 2: Thiết lập cổng và địa chỉ cho Thiết bị UDP. Ở Tab Controller, chọn Configuration > Communication > UDP Unicast Device để thiết lập địa chỉ IP và Port (địa chỉ IP này là địa chỉ của PC điều khiển để giao tiếp với Controller của Robot). Sau khi thiết lập hoàn tất, tiến hành Reset bộ điều khiển.



Hình 3.3. Thiết lập cấu hình cho thiết bị UDP

3.2.2. Các câu lệnh Rapid ở RobotStudio:

Các câu lệnh sử dụng trong chương trình Rapid:

- **EGMActPose:**

EGMActPose được sử dụng để khởi động phiên giao tiếp EGM và xác định các thông tin cố định, giúp robot thực hiện các chuyển động theo giá trị đặt từ máy tính với độ trễ thấp và phản hồi nhanh. Các thông tin này thường chỉ cần thiết lập một lần và có thể dùng cho nhiều chu kỳ điều khiển.

Cú pháp cơ bản**:**

EGMActPose EGMid [\Tool] [\WObj] [\TLoad] [\DOtoSensor] [\DataToSensor] [\DIfromSensor] [\DataFromSensor] CorrFrame CorrFrType SensorFrame SensorFrType [\x] [\y] [\z] [\rx] [\ry] [\rz] [\LpFilter] [\SampleRate] [\MaxPosDeviation] [\MaxSpeedDeviation]

Các tham số chính:

Bảng 3.6. Các tham số chính của EGMActPose

EGMid	Biến định danh cho kết nối EGM. Kiểu dữ liệu egmident
Tool	Công cụ được sử dụng cho các chuyển động được thực hiện với hướng dẫn EGMRunPose. Mặc định là tool0
WObj	WorkObject - hệ tọa độ làm việc. Nếu không khai báo, sử dụng hệ tọa độ toàn cục (world).
TLoad	Tải được sử dụng cho các chuyển động được thực hiện với lệnh EGMRunPose. Nếu không khai báo, lấy từ tooldata
CorrFrame	Khung hiệu chỉnh.
CorrFrType	Loại khung của khung hiệu chỉnh
SensorFrame	Khung cảm biến
SensorFrType	Loại khung của khung cảm biến
\x, \y, \z	Sai số vị trí cho phép (mm, mặc định ±1.0 mm)
\rx, \ry, \rz	Sai số góc cho phép (°, mặc định ±0.5°)
LpFilter	Bộ lọc thông thấp (Hz) giúp giảm nhiễu từ cảm biến. Mặc định lấy từ EGMSetup
SampleRate	Tần suất lấy mẫu dữ liệu từ cảm biến. Đơn vị là bội số của 4ms
MaxPosDeviation	Giới hạn lệch vị trí tối đa (°).
MaxSpeedDeviation	Giới hạn thay đổi vận tốc tối đa (°/s) trên các khớp.

- **EGMRunPose:**

EGMRunPose thực hiện chuyển động được dẫn hướng bằng dữ liệu vị trí từ PC, đưa robot từ một điểm xác định đến vị trí đích trong một tiến trình EGM cụ thể. Lệnh này cho phép xác định trước các hướng và tư thế có thể được điều chỉnh trong quá trình di chuyển.

Cú pháp cơ bản**:**

EGMRunPose EGMid, Mode [\NoWaitCond] [\x] [\y] [\z] [\rx] [\ry] [\rz] [\CondTime] [\RampInTime] [\RampOutTime] [\Offset] [\PosCorrGain]

Các tham số chính:

Bảng 3.7. Các tham số chính của EGMRunPose

EGMid	Biến định danh cho kết nối EGM. Kiểu dữ liệu egmident
Mode	Kiểu dữ liệu egmstopmode. Xác định cách kết thúc chuyển động
NoWaitCond	Cho phép RAPID tiếp tục mà không chờ đến khi chuyển động hoàn tất. Phải gọi EGMWaitCond sau đó để đồng bộ
\x, \y, \z	Chuyển động theo hướng x, y và z.
\rx, \ry, \rz	Định hướng lại xung quanh các trục x, y và z.
CondTime	Cho phép RAPID tiếp tục mà không chờ đến khi chuyển động hoàn tất. Phải gọi EGMWaitCond sau đó để đồng bộ. Giá trị mặc định là 1 giây.
RampInTime	Xác định tốc độ bắt đầu của chuyển động trong vài giây
RampOutTime	Xác định trong vài giây tốc độ giảm tốc của EGM sẽ được thực hiện
Offset	Một pose tĩnh để cộng thêm vào dữ liệu cảm biến
PosCorrGain	Hệ số hiệu chỉnh vị trí. Giá trị từ 0 đến 1. Mặc định là 1

- **EGMGetId**

EGMGetId được sử dụng để đặt trước một định danh EGM (EGMid). Định danh này sau đó sẽ được sử dụng trong tất cả các lệnh và hàm RAPID EGM khác để xác định một tiến trình EGM cụ thể đang được kết nối với tác vụ chuyển động RAPID mà từ đó nó được sử dụng. Một egmident được nhận diện bằng tên của nó, nghĩa là khi gọi EGMGetId lần thứ hai hoặc thứ ba với cùng một egmident, sẽ không tạo ra một tiến trình EGM mới cũng như không thay đổi nội dung của nó.

Để giải phóng một egmident nhằm cho phép các tiến trình EGM khác sử dụng, cần sử dụng lệnh RAPID EGMReset.

Cú pháp cơ bản: EGMGetId EGMid

Tham số chính:

EGMid: Biến định danh cho kết nối EGM. Kiểu dữ liệu egmident.

- **EGMSetupUC**

EGMSetupUC được sử dụng để thiết lập giao thức UdpUc cho một tiến trình EGM cụ thể (EGMid) làm nguồn cung cấp giá trị vị trí đích, mà robot và tối đa 6 trục phụ sẽ được dẫn hướng đến.

Cú pháp cơ bản:

EGMSetupUC MecUnit, EGMid, ExtConfigName, UCDevice [\Joint] | [\Pose] | [\PathCorr] [\APTR | \LATR] [\CommTimeout>]

Các tham số chính:

Bảng 3.8. Các tham số chính của EGMSetupUC

MecUnit	Tên đơn vị cơ khí
EGMid	Biến định danh cho kết nối EGM. Kiểu dữ liệu egmident
ExtConfigName	Tên cấu hình giao tiếp định nghĩa trong system parameters
UCDevice	Tên thiết bị UdpUc.
Joint	Chọn chuyển động khớp để hướng dẫn vị trí
Pose	Chọn chuyển động tư thế để hướng dẫn vị trí
PathCorr	Chọn hiệu chỉnh đường dẫn
APTR	Thiết lập loại cảm biến At-point tracker để hiệu chỉnh đường dẫn
LATR	Thiết lập loại cảm biến Look-ahead tracker để hiệu chỉnh đường dẫn
CommTimeout	Thời gian timeout (giây) nếu không nhận được dữ liệu UDP

- **EGMGetState**

EGMGetState trong RAPID cho phép bạn kiểm tra trạng thái hiện tại của một quá trình EGM (được xác định bằng egmident). Thuộc kiểu dữ liệu egmstate.

Cú pháp cơ bản: EGMGetState (EGMid)

Các tham số chính:

EGMid: Biến định danh cho kết nối EGM. Kiểu dữ liệu egmident.

Sự chuyển đổi giữa các trạng thái:

3.2.3. Các kiểu dữ liệu EGM đặc trưng

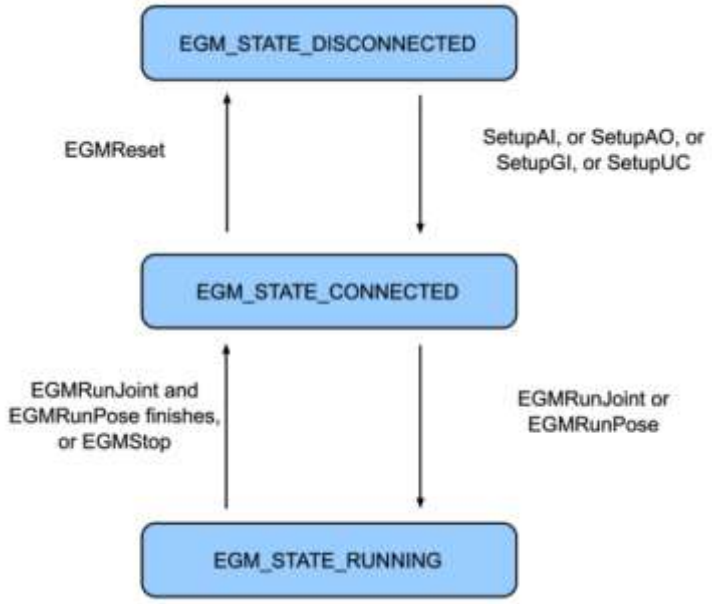
- **Egmstate**

Egmstate được sử dụng để xác định trạng thái của các hiệu chỉnh và đo đạc cảm biến trong EGM. Egmstate là giá trị trả về của hàm EGMGetState.

Bảng 3.9. Các giá trị trạng thái của một chu trình EGM

Giá trị	Mô tả
EGM_STATE_DISCONNECTED	Trạng thái EGM của tiến trình chưa được xác định. Chưa có thiết lập nào.
EGM_STATE_CONNECTED	Tiến trình EGM đã được thiết lập nhưng chưa hoạt động.
EGM_STATE_RUNNING	Tiến trình EGM đang hoạt động. Robot đang được điều khiển.

Sự chuyển dổi giữa các trạng thái



Hình 3.4. Sự chuyển đổi giữa các trạng thái của EGM

- **Egmident**

Egmident dùng để xác định một tiến trình EGM cụ thể.

Một egmident được đặt trước bằng lệnh EGMGetId. Sau đó, nó được dùng để liên kết và xác định chuỗi các lệnh như: EGMSetupXX, EGMActX, EGMRunX, và EGMReset cùng thuộc về một tiến trình EGM duy nhất.

Một egmident được xác định bằng tên biến của nó. Việc gọi EGMGetId nhiều lần với cùng một tên egmident sẽ không tạo tiến trình mới và không thay đổi nội dung của tiến trình hiện tại.

Chỉ có lệnh EGMReset mới có thể giải phóng một egmident để tiến trình khác có thể sử dụng lại.

- **Egmmin_max**

Egm_minmax được dùng để xác định tiêu chí hội tụ (convergence criteria) cho việc dừng quá trình EGM. Nó giúp robot biết khi nào sai số vị trí nằm trong giới hạn cho phép, và có thể coi là đã đạt đến mục tiêu.

Egm_minmax thường được sử dụng trong các lệnh như EGMActJoint và EGMActPose.

- **Egmstopmode**

Egmstopmode được dùng để xác định chế độ dừng cho các lệnh điều khiển hiệu chỉnh và dữ liệu đo từ cảm biến trong EGM.

Egmstopmode thường được sử dụng trong các lệnh như EGMRunJoint, EGMRunPose và EGMStop.

Bảng 3.10. Các giá trị được xác định của kiểu dữ liệu EGMstopmode

Giá trị	Mô tả
EGM_STOP_HOLD	Giữ nguyên vị trí cuối cùng khi dừng EGM. Robot không quay về vị trí ban đầu.
EGM_STOP_RAMP_DOWN	Trở về vị trí bắt đầu của EGM một cách từ từ.

- **Egmframetype**

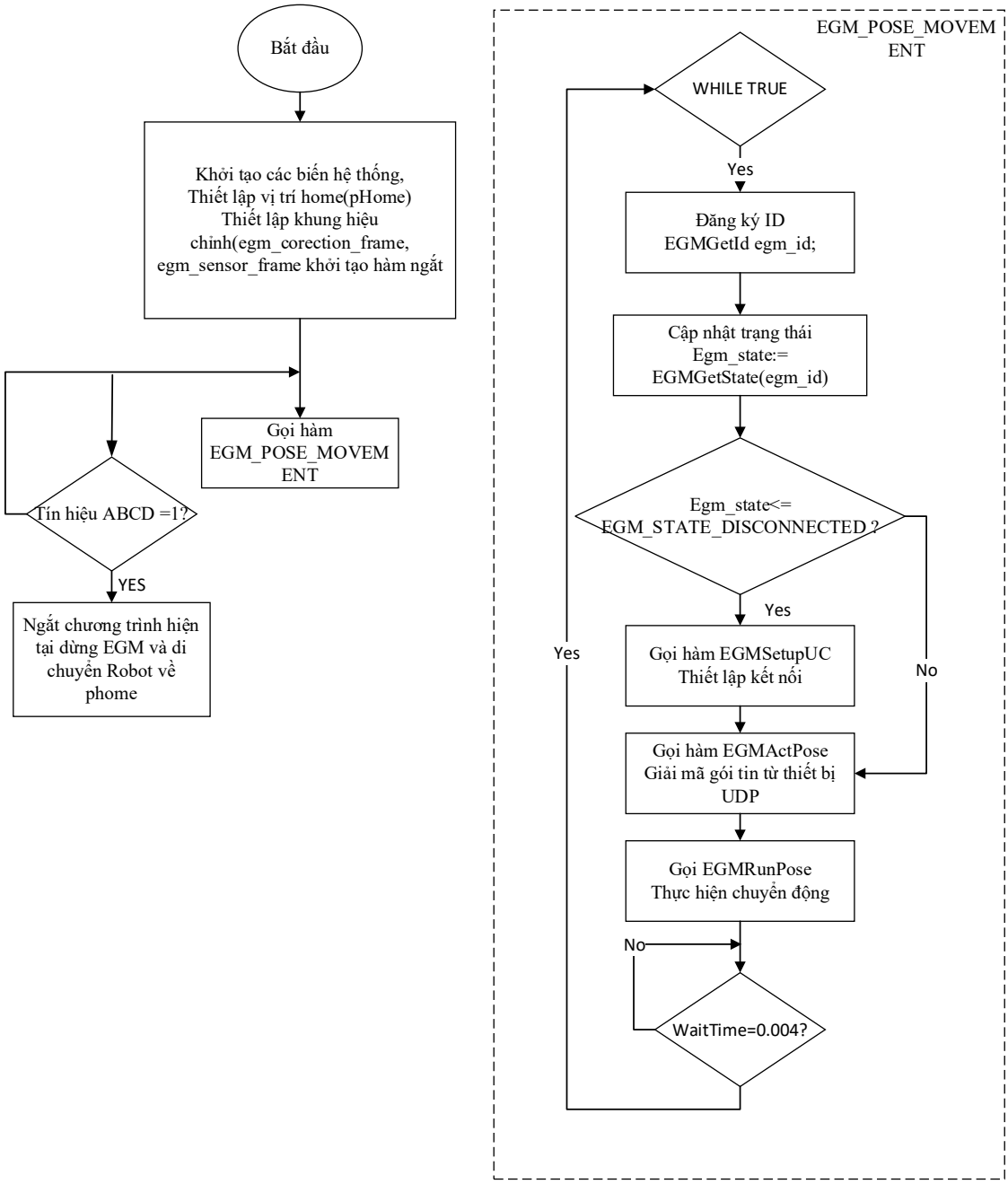
Đối với các hệ quy chiếu bắt buộc như CorrFrame và SensorFrame, cần phải xác định rõ xem chúng được định nghĩa tương ứng hệ quy chiếu nào. Thông tin này được

xác định bằng cách sử dụng các hệ quy chiếu có sẵn, được định nghĩa trước trong kiểu dữ liệu egmframetype.

Bảng 3.11. Các hệ quy chiếu được định nghĩa trong kiểu dữ liệu egmframetype

Giá trị	Mô tả
EGM_FRAME_BASE	Hệ quy chiếu được định nghĩa so với base frame (pose mode)
EGM_FRAME_TOOL	Hệ quy chiếu được định nghĩa so với tool0 (pose mode)
EGM_FRAME_WOBJ	Hệ quy chiếu được định nghĩa so với work object đang hoạt động (pose mode)
EGM_FRAME_WORLD	Hệ quy chiếu được định nghĩa so với world frame (pose mode)
EGM_FRAME_JOINT	Các giá trị là giá trị khớp (joint mode)

Lưu đồ thuật toán



Hình 3.5. Lưu đồ thuật toán chương trình RAPID

3.3. Giao tiếp giữa PC và Robot qua Externally Guided Motion (EGM)

3.3.1. Giới thiệu về giao thức EGM Sensor:

Giao thức EGM Sensor (Externally Guided Motion Sensor) được thiết kế nhằm phục vụ giao tiếp tốc độ cao giữa robot controller và các thiết bị ngoại vi (như sensor

hoặc máy tính), với mục tiêu truyền dữ liệu điều khiển theo thời gian thực và giảm thiểu tối đa độ trễ. Điểm đặc trưng nổi bật của giao thức này là không tuân theo cơ chế request/response, vì vậy ngay sau khi sensor nhận được gói tin đầu tiên từ robot, nó có thể chủ động gửi dữ liệu liên tục mà không cần chờ phản hồi.

Dữ liệu trong giao thức được mã hóa bằng Google Protocol Buffers (Protobuf) một giải pháp mã hóa nhanh, gọn nhẹ, hỗ trợ đa ngôn ngữ và có cấu trúc linh hoạt nhờ các trường tùy chọn (optional field), giúp đóng gói dữ liệu hiệu quả hơn nhiều so với XML. Trong quá trình giao tiếp, hai cấu trúc dữ liệu chính được sử dụng là EgmRobot (robot gửi) và EgmSensor (sensor gửi).

Về cơ chế hoạt động, sau khi nhận gói tin đầu tiên từ robot (là một data message), sensor có thể gửi dữ liệu tại bất kỳ thời điểm nào và với tần số bất kỳ, hoàn toàn không cần đồng bộ theo yêu cầu từ robot. Giao tiếp không yêu cầu cơ chế kết nối hay ngắt kết nối riêng biệt mà chỉ đơn giản là luồng dữ liệu hai chiều hoạt động độc lập giữa robot và sensor. Mặc định, robot gửi và đọc dữ liệu từ sensor theo chu kỳ 4 ms (tương đương tần số 250 Hz), và chu kỳ này có thể điều chỉnh thông qua tham số SampleRate trong các lệnh RAPID như EGMStreamStart, EGMActJoint hoặc EGMActPose. Mỗi motion task sử dụng một kênh UDP riêng để đảm bảo dữ liệu được truyền tải ổn định và chính xác.

Một số loại gói tin được sử dụng trong giao thức:

Bảng 3.12. Các gói tin EGM chính được sử dụng

Tên gói tin	Chức năng
EgmSensor	Dữ liệu gửi từ máy tính đến robot.
EgmRobot	Dữ liệu gửi từ robot đến máy tính
EgmHeader	Thông tin tiêu đề gói tin (số thứ tự, loại message, thời gian) được sử dụng chung ở EgmSensor và EgmRobot
Cartesian	Cấu trúc lưu vị trí và góc quay (x, y, z và rx, ry, rz).
EgmPlanned	Chứa thông tin về chuyển động đã được lên kế hoạch.
EgmFeedBack	Phản hồi thực tế từ robot (bao gồm vị trí và trạng thái).

3.3.2. Xây dựng chương trình giao tiếp EGM Sensor bằng Python

Chức năng hệ thống: Hệ thống sử dụng ngôn ngữ Python, thông qua giao thức EGM (Externally Guided Motion) để thực hiện hai chức năng chính trong điều khiển robot :

- Thứ nhất, hệ thống nhận phản hồi từ robot dưới dạng các gói tin EgmRobot gửi qua giao thức UDP. Mỗi gói tin này có thể chứa thông tin vị trí thực tế (x, y, z), góc quay thực tế (rx, ry, rz) của TCP (tool center point) và trạng thái robot. Dữ liệu nhận được được phân tích bằng protobuf, sau đó hệ thống cập nhật giá trị vị trí, góc quay và kiểm tra trạng thái của robot để phản ứng phù hợp.
- Thứ hai, hệ thống tạo và gửi các gói tin EgmSensor đến robot, trong đó chứa thông tin vị trí mục tiêu, góc quay mục tiêu hoặc cấu hình khớp, cùng với sequence number (seqno) để đồng bộ. Gói tin được đóng gói bằng protobuf rồi gửi qua socket UDP đến robot với tần số điều khiển 250 Hz (mỗi 4 ms một gói tin).

3.3.2.1. Thiết lập ban đầu cho chương trình giao tiếp

Để thực hiện giao tiếp bằng giao thức EGM, ABB cung cấp một tệp có tên "egm.proto", đây là tệp định nghĩa giao thức dùng để mô tả các message truyền qua UDP giữa robot và PC. Để sử dụng, cần biên dịch tệp này bằng trình biên dịch Protocol Buffers để tạo ra mã nguồn Python tương ứng.

Cách cài đặt: cần cài thư viện protobuf bằng lệnh pip install protobuf và tải công cụ protoc từ trang chính thức của Google.

Cách sử dụng: chạy lệnh protoc_python_out = egm.proto (để biên dịch tệp proto thành mã Python.). Từ đó tạo ra file egm_pb2.py để import và sử dụng trong chương trình.

3.3.2.2. Cấu trúc lớp chính trong chương trình Python giao tiếp EGM

Ở đây nhóm sử dụng chương trình lấy từ thư viện mã nguồn mở abb_egm_pyclient(do Anton Tetov Johansson phát triển) để làm thư viện giao tiếp với robot ABB thông qua giao thức EGM (Externally Guided Motion).

Trong thư viện, EGMClient là lớp trung tâm được thiết kế nhằm thiết lập, gửi và nhận các gói tin điều khiển/giám sát giữa PC và robot ABB thông qua giao thức EGM (Externally Guided Motion). Nó giúp xây dựng các lệnh điều khiển chuyển động và thu thập phản hồi trạng thái robot theo thời gian thực. Lớp EGMClient gồm các nhóm chức năng chính:

- **Nhóm chức năng Khởi tạo giao tiếp (InitializeClient):**

Chức năng: Chuẩn bị socket, bind cổng, và thiết lập các biến nội bộ của EGMClient.

Thành phần chính:

Bảng 3.13. Nhóm chức năng Khởi tạo giao tiếp (InitializeClient)

udp_socket	Socket UDP để gửi/nhận dữ liệu
udp_port	Cổng giao tiếp (mặc định 6510)
robot_controller_address	Địa chỉ IP của robot controller (xác định khi nhận gói tin đầu tiên)
send_counter	Số thứ tự gói tin gửi (seqno)
last_state	Lưu gói phản hồi mới nhất từ robot
is_stopped	Trạng thái robot (True nếu MCI_STOPPED)

- **Nhóm chức năng Tạo gói tin (CreateSensorMessage):**

Chức năng: Tạo gói tin dạng EgmSensor để gửi đến robot.

Thành phần chính:

Bảng 3.14. Nhóm chức năng Tạo gói tin (CreateSensorMessage)

seq_num	Số thứ tự của gói tin (auto tăng)
header.tm_type	Loại gói tin (PLANNED_POSITION)
planned.cartesian	Vị trí mục tiêu (x, y, z)
planned.euler	Góc quay mục tiêu (rx, ry, rz)
planned.joints	Góc các khớp mục tiêu (nếu điều khiển dạng joint)
planned.externalJoints	Góc các external joint (nếu có)

- **Nhóm chức năng Gửi gói tin (SendMessage):**

  Chức năng: Đóng gói, serialize và gửi dữ liệu điều khiển tới robot qua UDP.

  Thành phần chính:

Bảng 3.15. Nhóm chức năng Gửi gói tin (SendMessage)

send_msg(message)	Gửi một message protobuf tới robot (địa chỉ đã biết)
send_planned_configuration(joint_list)	Gửi góc khớp mục tiêu
send_planned_frame(x, y, z, rx, ry, rz)	Gửi vị trí + góc mục tiêu Descartes
send_cartesian_path(path, time_between_points)	Gửi dãy điểm để robot chạy theo quỹ đạo
reset_sequence_counter()	Reset seq_num về 0.

- **Nhóm chức năng Nhận gói tin (ReceiveMessage):**

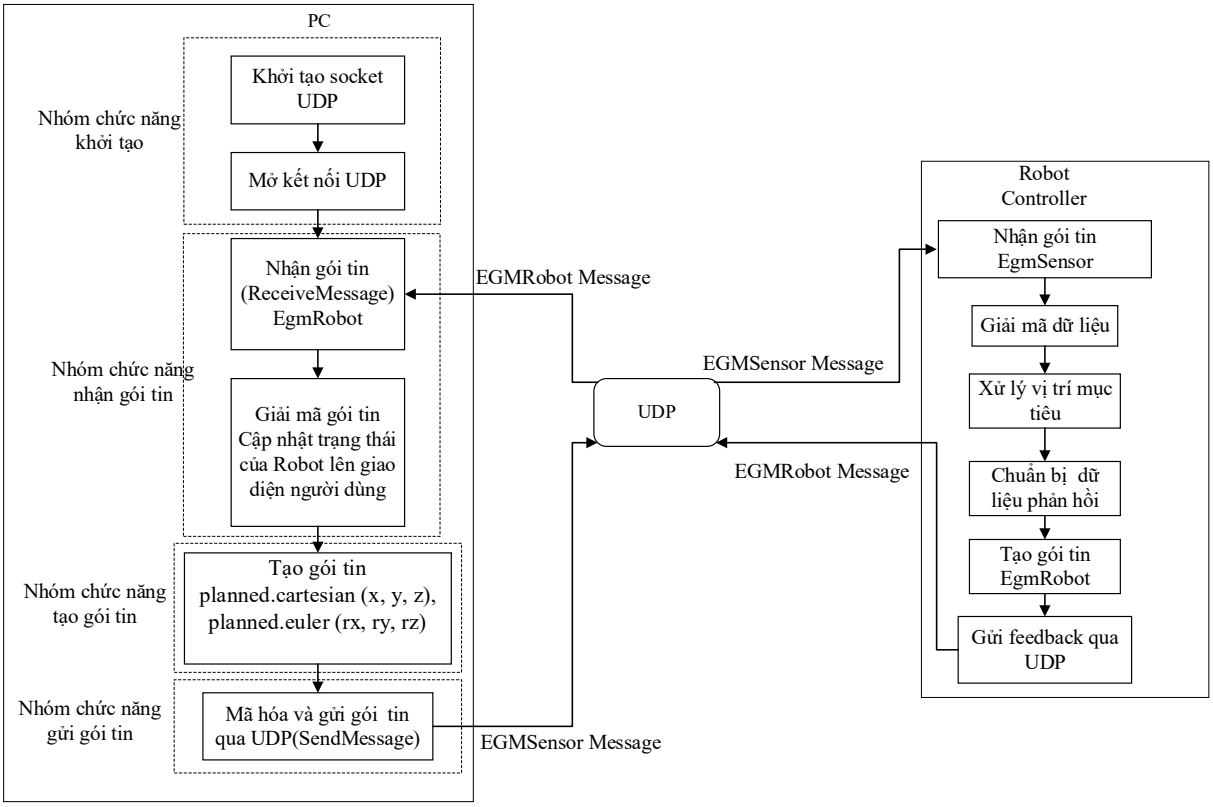
  Chức năng: Nhận và xử lý gói tin phản hồi từ robot.

  Thành phần chính:

Bảng 3.16. Nhóm chức năng Nhận gói tin (ReceiveMessage)

get_last_packet()	Lấy gói tin mới nhất từ buffer UDP (bỏ qua gói cũ)
receive_msg()	Xử lý gói EgmRobot từ dữ liệu nhận được

Để trực quan hóa hoạt động của chương trình giao tiếp Python với EGM, nhóm đã xây dựng lưu đồ thể hiện luồng xử lý chính trong lớp EGMClient. Lưu đồ mô tả quá trình từ khởi tạo, tạo gói tin điều khiển, gửi gói tin đến nhận và xử lý phản hồi từ robot, qua đó giúp hình dung rõ cấu trúc chương trình và mối liên hệ giữa các nhóm chức năng chính của EGMClient.



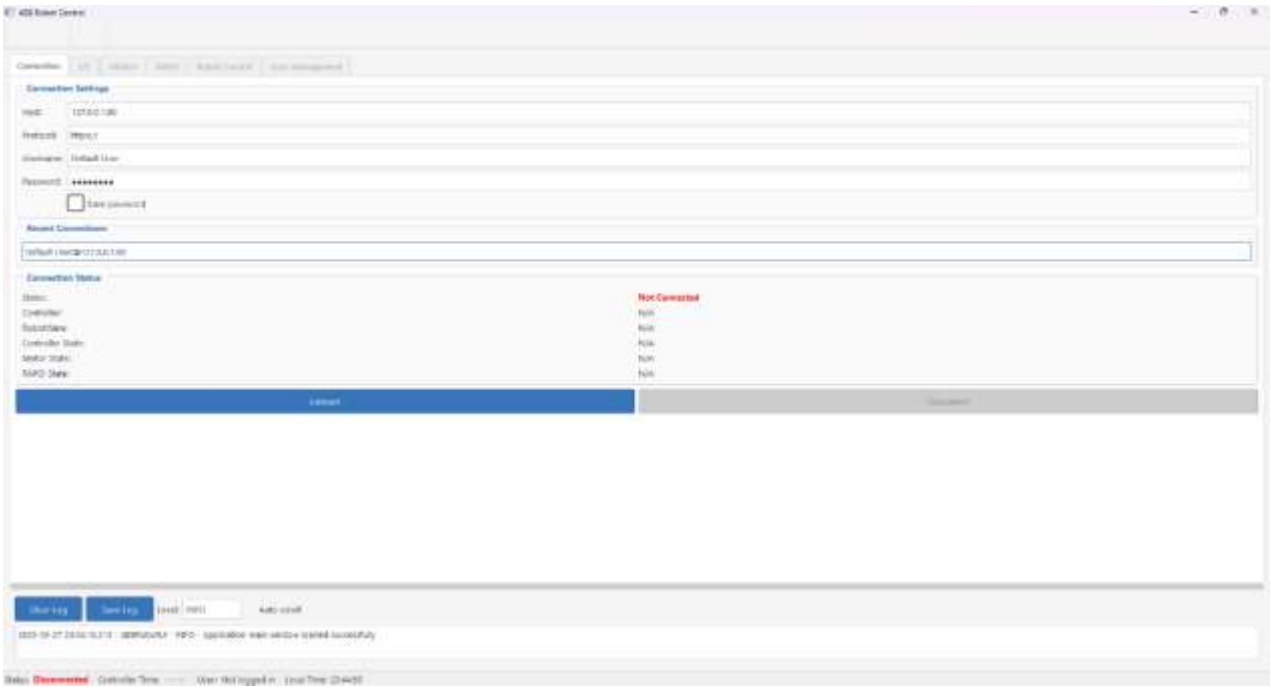
Hình 3.6: Lưu đồ giao tiếp giữa Python và Robot qua EGM

3.4. Thiết kế giao diện phần mềm giao tiếp trên PC

Giao diện phần mềm trên PC được thiết kế theo cấu trúc đơn giản, nhằm đảm bảo tính trực quan, dễ sử dụng và đáp ứng linh hoạt nhu cầu của người dùng từ cơ bản đến nâng cao. Mỗi tab đại diện cho một nhóm chức năng cụ thể, các tab chính trong giao diện bao gồm:

3.4.1. Tab Connection

Đây là nơi người dùng thực hiện việc kết nối và ngắt kết nối với robot công nghiệp ABB. Đây là bước đầu tiên cần thiết để có thể điều khiển hoặc giám sát trạng thái của robot thông qua phần mềm.



Hình 3.7. Giao diện đăng nhập

Giao diện đăng nhập gồm các chức năng chính:

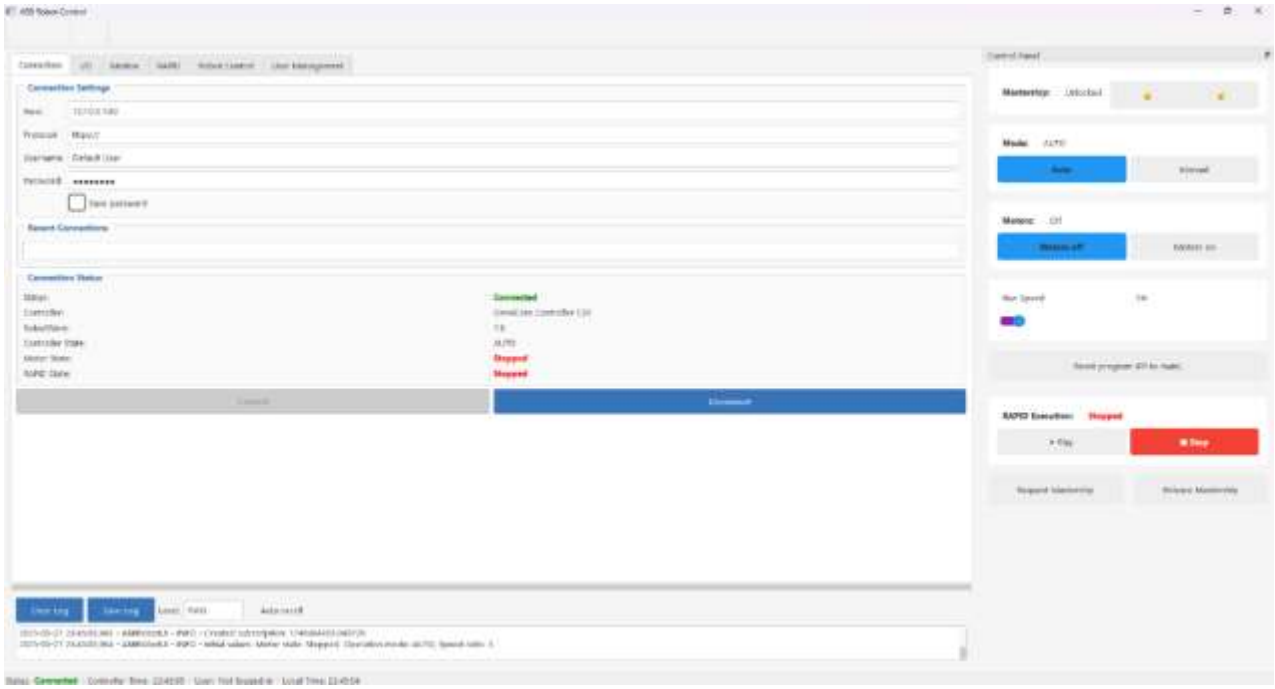
- Cấu hình kết nối: Cho phép người dùng nhập các thông số cần thiết để thiết lập kết nối với robot, bao gồm địa chỉ IP, giao thức (HTTP/HTTPS), tên đăng nhập, mật khẩu và tùy chọn lưu mật khẩu. Ngoài ra, giao diện hiển thị các cấu hình đã sử dụng trước đó để thuận tiện cho việc kết nối lại. Người dùng có thể dễ dàng thực hiện kết nối (Connect) hoặc ngắt kết nối (Disconnect) với robot.

- Trạng thái: Hiển thị tình trạng kết nối hiện tại giữa PC và robot (đã kết nối/chưa kết nối), đồng thời cung cấp các thông tin trạng thái chính của robot như chế độ hoạt động, lỗi hiện thời hoặc tín hiệu khẩn cấp (nếu có).
- Nhật ký: Ghi nhận và hiển thị toàn bộ quá trình giao tiếp giữa PC và robot, bao gồm các lệnh đã gửi, phản hồi nhận được và thông tin cảnh báo/lỗi phát sinh.

3.4.2. Tab Control Panel

Sau khi đăng nhập thành công, giao diện tab Control Panel sẽ hiển thị, cho phép người dùng theo dõi và điều khiển các chế độ vận hành của robot. Tab này gồm các chức năng chính:

- Giám sát chế độ hoạt động: Tab panel cung cấp thông tin về chế độ Auto/Manual, trạng thái động cơ (Motors On/Off), tốc độ chạy (Run Speed) và tình trạng thực thi chương trình RAPID.
- Điều khiển trực tiếp: Người dùng có thể trực tiếp chuyển đổi chế độ, bật/tắt động cơ, điều chỉnh tốc độ và thực thi/dừng chương trình RAPID ngay trên panel, hỗ trợ vận hành nhanh chóng và thuận tiện.

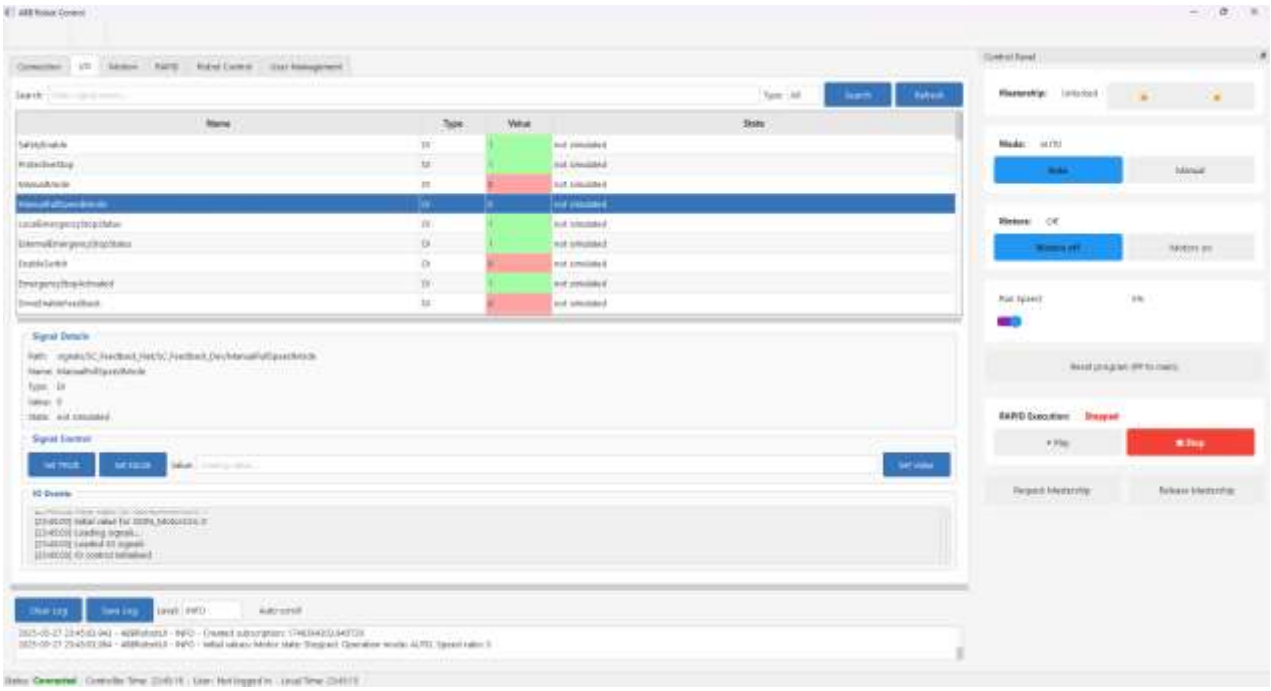


Hình 3.8. Giao diện tab Control Panel sau khi log in vào

Sau khi xác thực quyền truy cập, người dùng có thể truy cập vào các tab với các chức năng quan trọng.

3.4.3. I/O Control

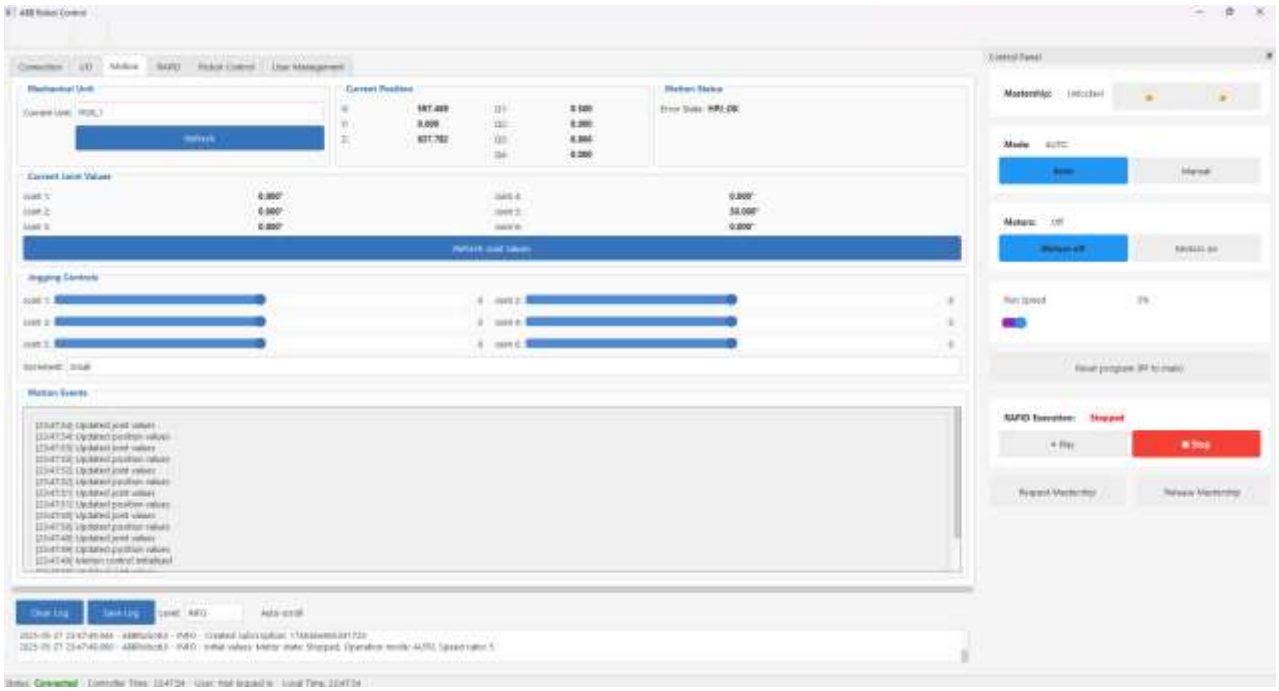
Cho phép người dùng giám sát và điều khiển các tín hiệu I/O (Input/Output) của robot. Người dùng có thể theo dõi trạng thái tín hiệu vào (input), điều khiển tín hiệu ra (output) và thực hiện các thao tác I/O trực tiếp để phục vụ quá trình vận hành hoặc kiểm tra hệ thống.



Hình 3.9. Tab điều khiển I/O

3.4.4. Motion Tab

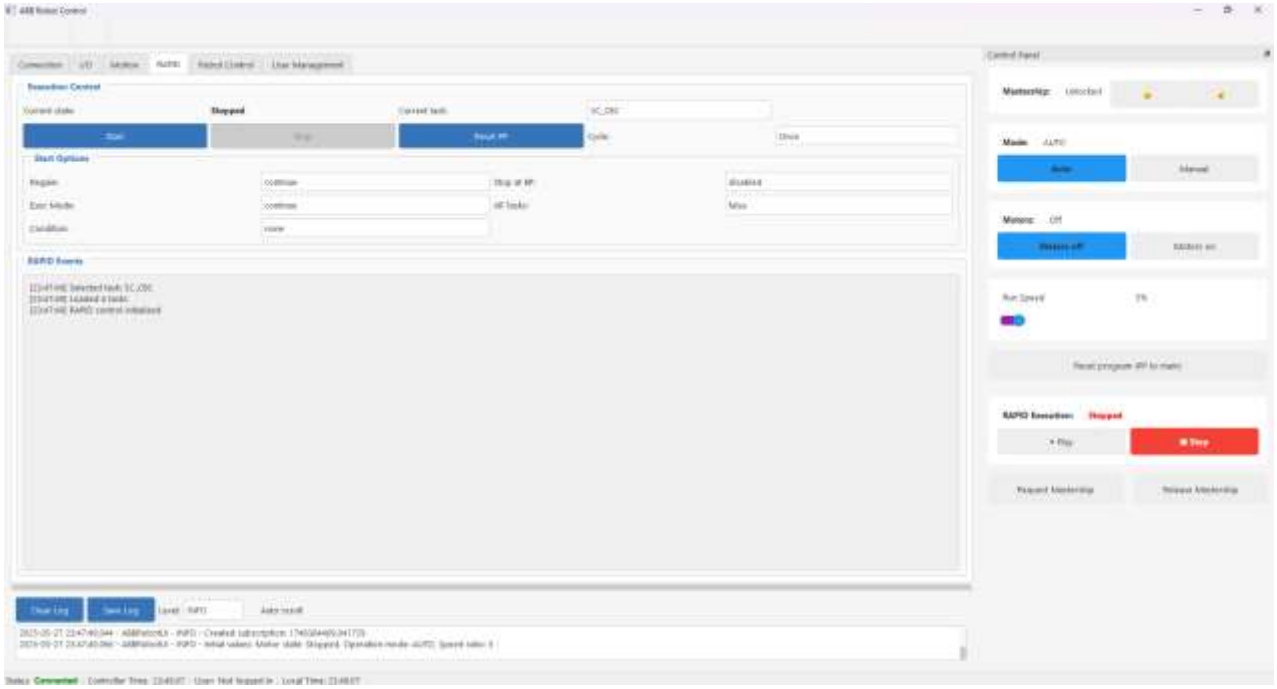
Cho phép người dùng giám sát và điều khiển toàn diện các thông số liên quan đến chuyển động của robot. Người dùng có thể theo dõi vị trí hiện tại của robot trong không gian 3D (tọa độ X, Y, Z), góc các khớp (Joint 1–6), vận tốc di chuyển và tình trạng chuyển động thực tế. Tab này cũng hỗ trợ điều khiển bằng tay (jogging) với thanh trượt cho từng khớp, cho phép thực hiện các điều chỉnh chính xác trong quá trình setup hoặc bảo trì. Bên cạnh đó, hệ thống ghi lại các sự kiện chuyển động và cảnh báo, hỗ trợ người dùng giám sát, phân tích và xử lý sự cố kịp thời, đảm bảo quá trình vận hành an toàn và hiệu quả.



Hình 3.10. Tab điều khiển thủ công

3.4.5. RAPID Tab

Giúp người dùng quản lý và theo dõi quá trình thực thi chương trình RAPID trên robot một cách trực quan và hiệu quả. Tại đây, người dùng có thể dễ dàng khởi động, dừng, đặt lại biến toàn cục hoặc chạy chương trình ở chế độ vòng lặp. Tab này cũng cho phép thiết lập các tùy chọn khi bắt đầu thực thi, như chế độ chạy, điều kiện khởi động hoặc cách xử lý khi giành lại quyền kiểm soát. Ngoài ra, toàn bộ sự kiện liên quan đến quá trình thực thi, cảnh báo hoặc lỗi — đều được ghi lại để hỗ trợ giám sát, phân tích và xử lý sự cố kịp thời, đảm bảo robot vận hành an toàn và chính xác.



Hình 3.11. Giao diện minh họa chức năng mở rộng

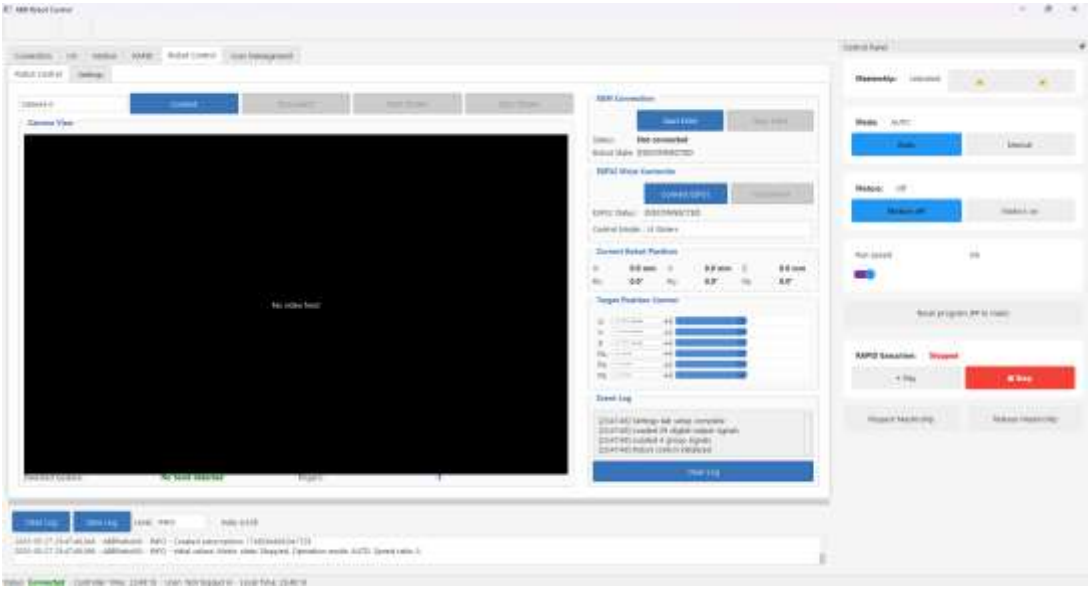
3.4.6. Robot Control Tab

3.4.6.1. Robot Control Main Tab

Cung cấp giao diện điều khiển trực quan, cho phép người dùng:

- Camera View: Hiển thị hình ảnh trực tiếp từ camera gắn trên máy tính, đồng thời tích hợp xử lý ảnh bằng thư viện OpenCV và MediaPipe để nhận diện cử chỉ tay và đối tượng trong không gian, từ đó tạo lệnh điều khiển robot theo thời gian thực. Việc kết nối/ngắt kết nối camera được thực hiện nhanh chóng, hỗ trợ quan sát chính xác trong quá trình vận hành.
- EGM Connection**:** Quản lý giao tiếp giữa robot và máy tính, thiết lập kết nối EGM để robot nhận lệnh điều khiển từ thiết bị ngoài. Người dùng có thể dễ dàng khởi động hoặc dừng chế độ EGM thông qua giao diện, đảm bảo truyền lệnh an toàn và đồng bộ giữa robot và hệ thống điều khiển.
- Robot Position Control: Phần này hiển thị và quản lý vị trí hiện tại và mục tiêu của robot dưới dạng tọa độ (X, Y, Z) và góc quay (Rx, Ry, Rz), với giá trị cập nhật liên tục khi robot di chuyển. Người dùng có thể linh hoạt chuyển đổi chế độ điều khiển (Control Mode) giữa:
    - Manual (Slider): Điều chỉnh vị trí mục tiêu thủ công qua thanh trượt cho từng trục và góc.

- Auto (ESP32): Vị trí mục tiêu được tính toán tự động từ dữ liệu ESP32 gửi đến. Người dùng có thể kết nối hoặc ngắt kết nối ESP32 để bật/tắt chế độ nhận dữ liệu điều khiển tự động.



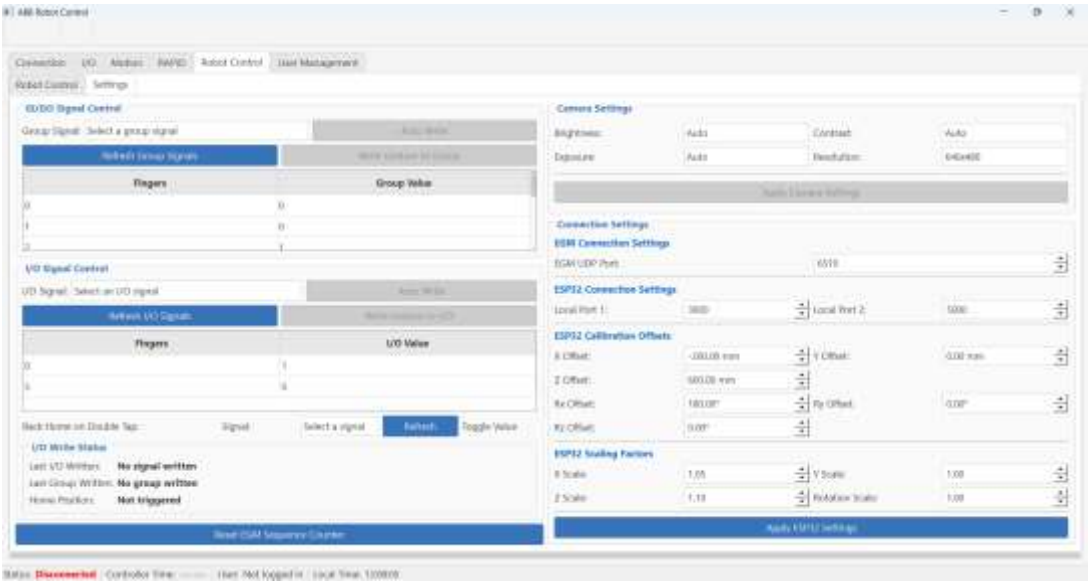
Hình 3.12. Giao diện kết nối và hiển thị camera

3.4.6.2. Robot Control Settings Tab

Tab này cho phép người dùng giám sát, điều chỉnh tín hiệu I/O và cấu hình các tham số kết nối, hiệu chỉnh để đảm bảo robot vận hành chính xác theo yêu cầu.

- Signal Control**:** Hiển thị danh sách nhóm tín hiệu (GO) và tín hiệu I/O, cho phép người dùng lựa chọn và điều khiển tín hiệu phù hợp để thực hiện các tác vụ đã thiết kế sẵn. Hệ thống tích hợp MediaPipe để nhận diện cử chỉ bàn tay người vận hành (như nắm tay để đóng gripper, mở tay để mở gripper, double tap hai ngón để kích hoạt hoặc hủy kích hoạt khóa an toàn). Giá trị của các cử chỉ này được ánh xạ trực tiếp sang tín hiệu đầu ra điều khiển robot (I/O, GO), từ đó tự động xử lý và gửi setpoint điều khiển, giúp robot thực thi lệnh chính xác, nhanh chóng và đảm bảo an toàn trong quá trình vận hành.
- Camera Settings: Cho phép điều chỉnh các thông số camera như độ sáng, phơi sáng, độ tương phản, độ phân giải, giúp tối ưu chất lượng hình ảnh phục vụ giám sát và xử lý hình ảnh.

- EGM Connection Settings: Cấu hình chính xác các thông số kết nối với ESP32 như cổng UDP, IP cục bộ, đảm bảo robot giao tiếp ổn định với thiết bị điều khiển bên ngoài.
- ESP32 Calibration & Scaling**:** Người dùng có thể nhập các giá trị bù chỉnh (offset) và tỷ số phóng đại (scaling) cho các trục X, Y, Z và góc quay Rx, Ry, Rz nhằm hiệu chỉnh sai lệch và điều chỉnh phạm vi chuyển động của robot khi nhận lệnh từ ESP32. Sau khi cấu hình các thông số này, người dùng nhấn Apply ESP32 Settings để lưu và áp dụng ngay, đảm bảo robot vận hành chính xác và linh hoạt theo thiết lập mới.



Hình 3.13. Giao diện cấu hình tín hiệu I/O

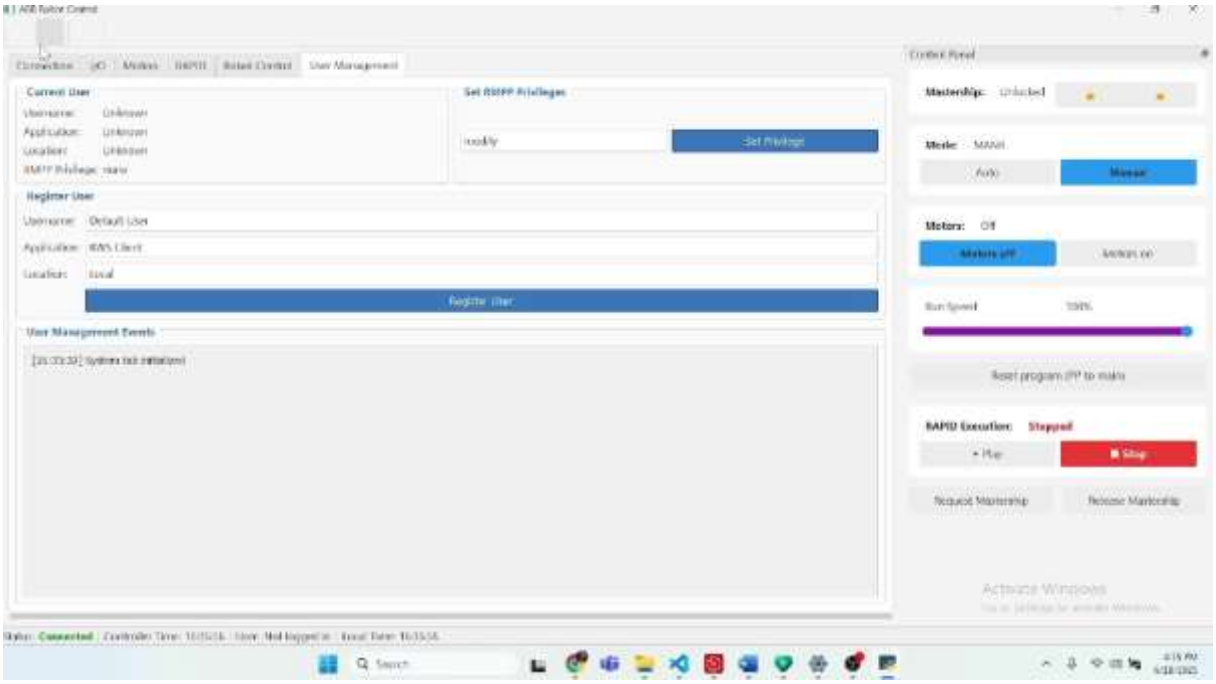
3.4.7. User Management Tab

Tab này cho phép người dùng quản lý tài khoản, phân quyền truy cập và giám sát hoạt động người dùng nhằm đảm bảo an toàn, bảo mật và vận hành hệ thống robot hiệu quả.

- Current User Information**:** Hiển thị rõ ràng thông tin về người dùng hiện tại như username, ứng dụng đang sử dụng, vị trí truy cập và quyền hạn RMPP. Điều này hỗ trợ người vận hành hoặc quản trị viên giám sát và xác định nhanh quyền của người đang đăng nhập.
- Register User: Hỗ trợ người quản trị dễ dàng tạo thêm tài khoản bằng cách nhập tên người dùng, chọn ứng dụng liên quan và xác định vị trí truy cập

(Local hoặc Remote). Đây là bước cần thiết để mở rộng quyền truy cập cho nhân sự hoặc thiết bị mới trong hệ thống.

- Set RMPP Privileges: Cho phép cài đặt mức độ truy cập cụ thể cho từng tài khoản. Người quản trị có thể nhập tên người dùng và chọn quyền RMPP tương ứng, giúp kiểm soát chính xác những gì người dùng đó có thể thực hiện trên robot, từ theo dõi đến điều khiển trực tiếp.



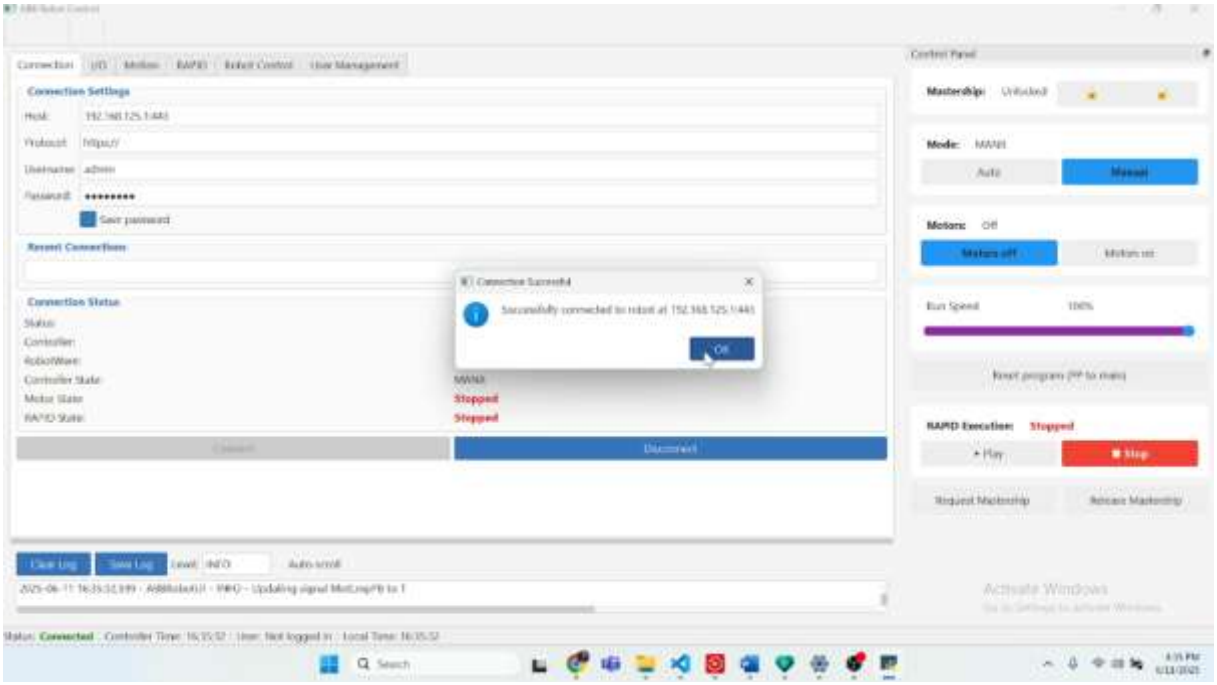
Hình 3. 14. Giao diện quản lý người dùng

Kết luận chương 3

Chương đã hoàn thiện hệ thống phần mềm điều khiển robot ABB CRB15000 thông qua Robot Web Services (RWS), triển khai thành công giao tiếp RESTful API (HTTP/HTTPS) để quản lý trạng thái robot, điều khiển chuyển động và truy xuất dữ liệu RAPID/I/O với độ trễ dưới 50 ms. Giao diện điều khiển tích hợp (GUI) trên nền tảng Python/PyQt5 cung cấp các module chức năng: kết nối robot, điều khiển tổng quát, giám sát chuyển động và xử lý camera/I/O, cho phép tương tác trực quan thông qua nhận diện cử chỉ tay sử dụng thư viện MediaPipe và OpenCV ánh xạ cử chỉ "mở/nắm tay" thành tín hiệu điều khiển gripper. Hệ thống được kiểm thử ổn định trên Robot Studio, đạt sai số vị trí <12 mm và độ trễ toàn phần 300–400 ms, tạo nền tảng ứng dụng cho các giải pháp công nghiệp 4.0 kết hợp tương tác tự nhiên và giám sát đa nhiệm, với hướng phát triển tương lai tập trung giảm độ trễ xử lý ảnh và mở rộng nhận diện đa cử chỉ. Hệ thống đã cung cấp giải pháp điều khiển robot trực quan, linh hoạt thông qua giao diện phần mềm và tương tác cử chỉ tự nhiên.

CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM VÀ ĐÁNH GIÁ

4.1. Kiểm tra chức năng giao diện hệ thống



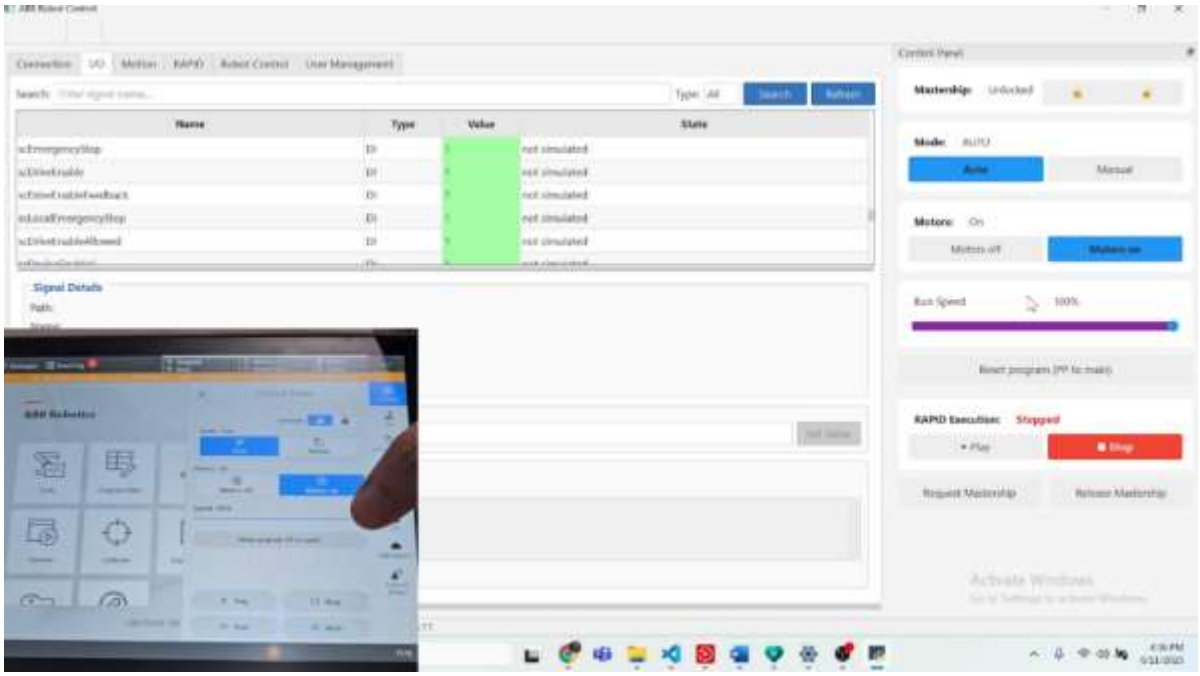
Hình 4.1. Giao diện khi đăng nhập thành công

Trên giao diện đã cho thấy quá trình kết nối thành công giữa máy tính và bộ điều khiển robot ABB thông qua phần mềm ABB Robot Control. Người dùng đã thực hiện đăng nhập bằng tài khoản quản trị viên (admin), xác thực thông tin qua giao thức bảo mật HTTPS cho phép thực hiện các thao tác lập trình và điều khiển robot.

Sau khi kết nối thành công, người dùng có thể:

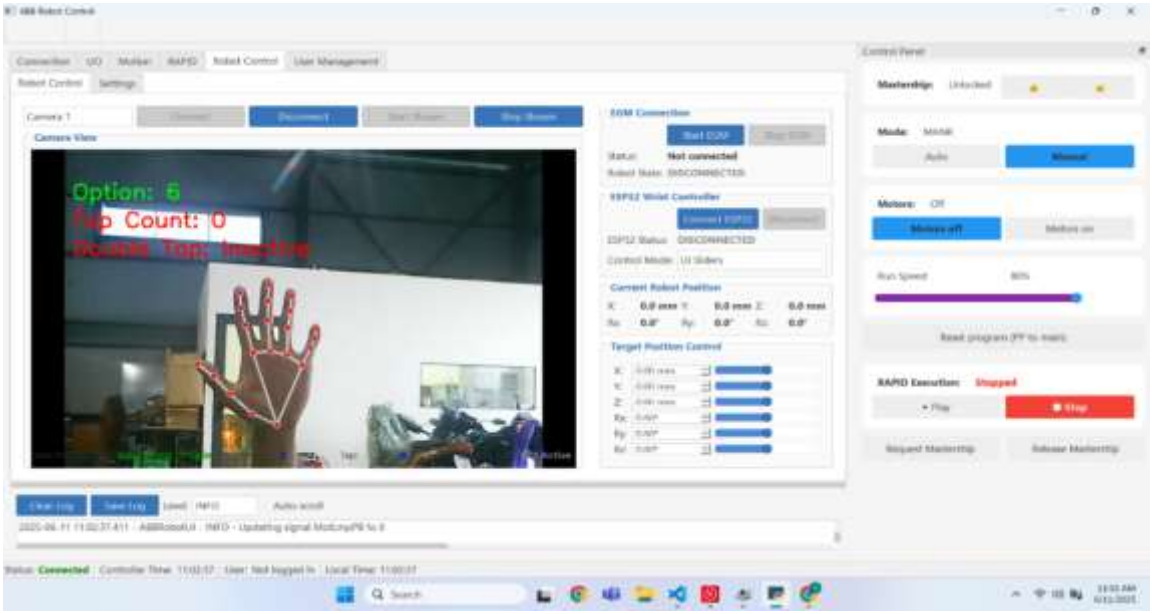
- Chọn chế độ vận hành: Chuyển đổi giữa chế độ Auto hoặc Manual.
- Bật/tắt động cơ robot: Điều khiển trạng thái mô-tơ (Motors on/off).
- Chạy hoặc dừng chương trình RAPID: Cho phép điều khiển và theo dõi tiến trình thực thi mã lệnh RAPID.
- Điều chỉnh tốc độ thực thi chương trình: Với thanh trượt Run Speed (0–100%).

Giao diện này là bước khởi đầu quan trọng để người vận hành có thể giám sát, lập trình, và điều khiển robot trong môi trường công nghiệp một cách an toàn và hiệu quả.



Hình 4.2. Đồng bộ điều khiển giữa Flexpendant và giao diện điều khiển

Hình ảnh minh họa thể hiện sự đồng bộ hóa trạng thái điều khiển giữa giao diện phần mềm ABB Robot Control trên máy tính và tay cầm FlexPendant điều khiển robot ABB. Cả hai giao diện đều hiển thị trạng thái giống nhau tại cùng thời điểm. Việc đồng bộ này giúp đảm bảo mọi thao tác điều khiển và giám sát robot có thể thực hiện được đồng thời từ giao diện máy tính lẫn thiết bị cầm tay, từ đó tăng độ linh hoạt và độ tin cậy trong quá trình vận hành và giám sát hệ thống robot.



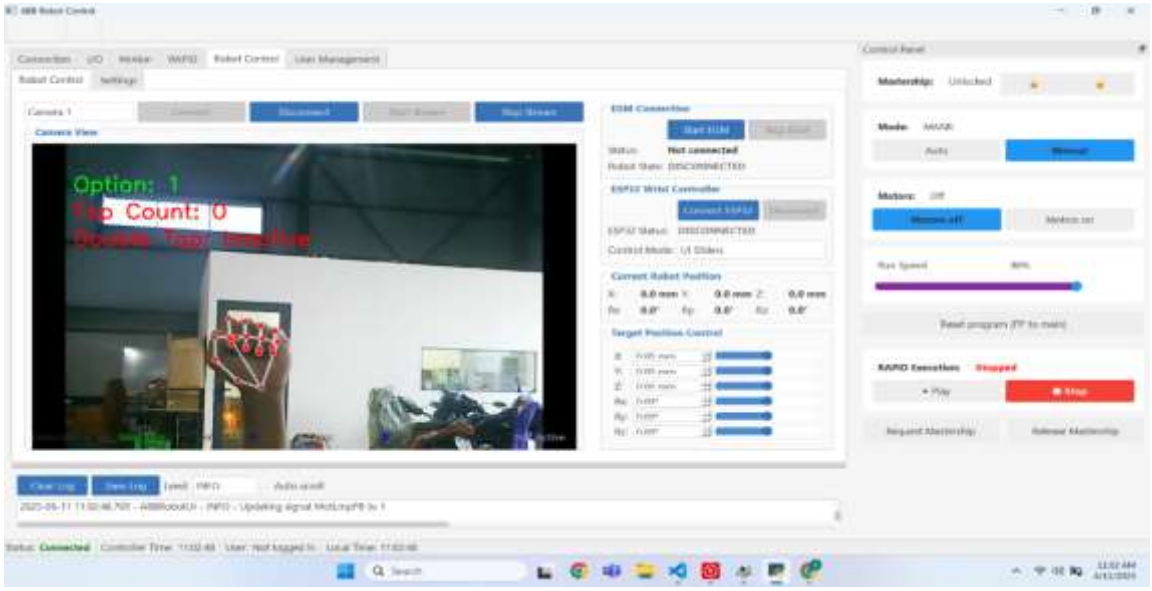
Hình 4.3. Nhận diện cử chỉ tay thông qua xử lí ảnh

Camera thu hình tay người điều khiển và xử lý trực tiếp thông qua thư viện MediaPipe để xác định 21 điểm đặc trưng (landmarks) trên bàn tay. Tốc độ xử lý đạt 25–30 FPS, đảm bảo độ trễ thấp trong quá trình phản hồi. Hệ thống dễ dàng phân biệt các cử chỉ đơn giản như "mở bàn tay", "chạm", "nhấn đúp", đồng thời hiển thị trực quan số lần thực hiện và trạng thái hiện hành.

Hình 4.4. Điều khiển robot thông qua cử chỉ tay

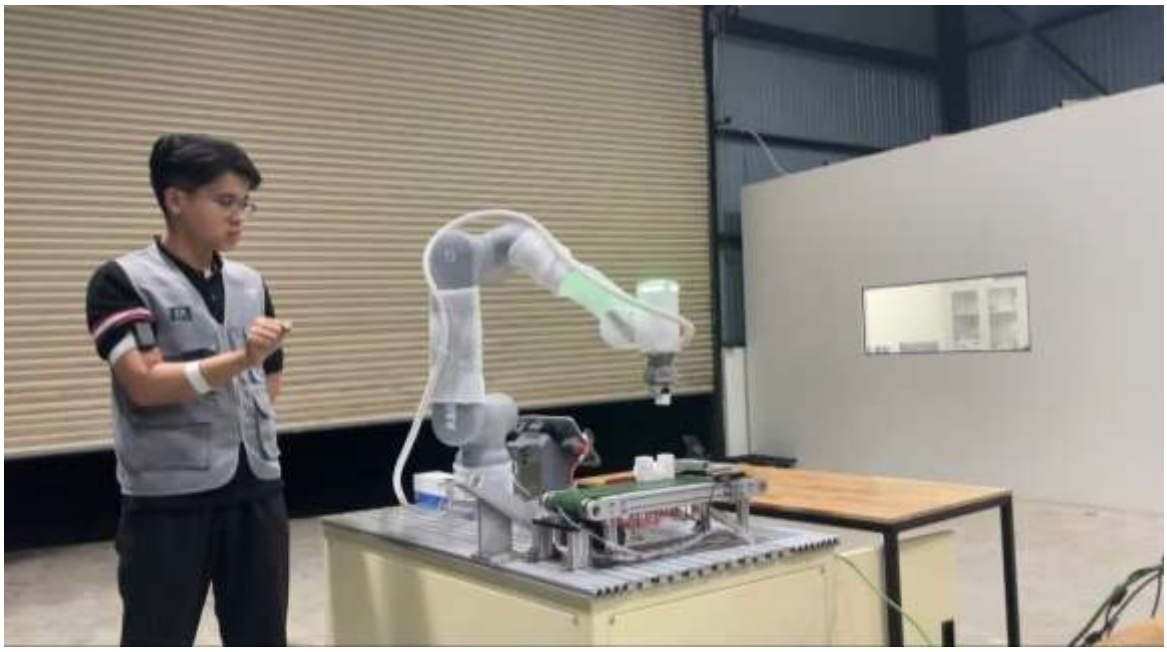
Thực tế khi người dùng tương tác với cánh tay robot ABB CRB 15000 trong môi trường xưởng sản xuất. Tư thế giơ tay cùng với thiết bị IMU gắn trên tay cho thấy khả

năng linh hoạt trong quá trình hoạt động. Khoảng cách an toàn giữa người và robot được duy trì tốt, và robot đã ở chế độ sẵn sàng làm việc. Điều này phản ánh sự phối hợp nhịp nhàng giữa phần cứng (robot, camera, IMU, ESP32) và phần mềm điều khiển (RAPID, giao diện người dùng UI, truyền dữ liệu Wi-Fi), tạo nên một hệ thống điều khiển cử chỉ không tiếp xúc hiệu quả, an toàn và có tính ứng dụng cao trong các môi trường sản xuất thông minh.



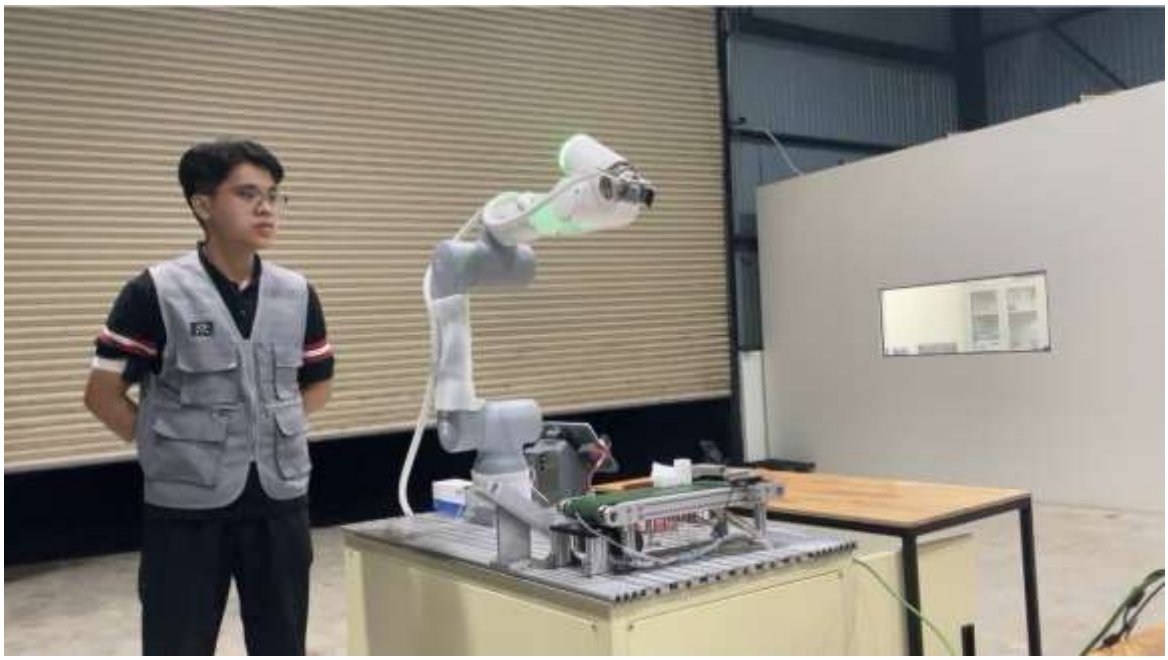
Hình 4.5. Giao diện nhận diện cử chỉ tay

Giao diện nhận diện cử chỉ "nắm tay" được hệ thống phát hiện chính xác với option 1. Thư viện MediaPipe tiếp tục xác định các điểm landmark trên bàn tay với độ chính xác cao và cập nhật trạng thái nhận diện theo thời gian thực. Cử chỉ "nắm tay" được quy ước là tín hiệu để thực hiện hành động “gắp vật” trong chuỗi thao tác điều khiển robot. Việc chuyển đổi từ trạng thái "bàn tay mở" sang "nắm tay" cho phép robot phản hồi hành động một cách linh hoạt, tạo ra trải nghiệm điều khiển tự nhiên và trực quan cho người vận hành.



Hình 4.6. Điều khiển robot gắp vật bằng cử chỉ tay

Cho thấy quá trình robot thực hiện hành động gắp vật dựa trên tín hiệu cử chỉ đầu vào. Người điều khiển đã thực hiện cử chỉ "nắm tay" và cánh tay robot ABB đã phản hồi bằng cách di chuyển xuống vị trí gắp và tiến hành hành động gắp một đối tượng trên băng chuyền. Hệ thống đèn trạng thái màu xanh lá vẫn duy trì, cho thấy robot vẫn đang trong trạng thái sẵn sàng và hoạt động ổn định.



Hình 4.7. Cánh tay robot về vị trí home sau khi nhận lệnh theo option

Tổng thể hệ thống cho thấy khả năng phản hồi chính xác và ổn định trước các tín hiệu điều khiển cử chỉ, giảm thiểu độ trễ giữa thao tác người dùng và hành động robot. Điều này chứng minh hiệu quả của mô hình điều khiển dựa trên thị giác máy và tiềm năng ứng dụng trong các hệ thống sản xuất thông minh không tiếp xúc.

4.2. Đánh giá độ trễ và độ chính xác của hệ thống

4.2.1. Đánh giá độ trễ

Hệ thống được thiết kế với sự kết hợp giữa hai cảm biến IMU và một khối xử lý thị giác máy tính. Cảm biến IMU có nhiệm vụ thu nhận chuyển động của cánh tay người điều khiển để tính toán tọa độ đầu vào cho robot, trong khi đó hệ thống thị giác máy tính nhận dạng vật thể và điều khiển các tín hiệu I/O để hỗ trợ quá trình thao tác.

Trong quá trình thử nghiệm thực tế, độ trễ của hệ thống được đo đạc bằng cách xác định thời gian từ lúc người điều khiển bắt đầu chuyển động đến khi robot hoàn tất hành vi tương ứng. Kết quả cho thấy các thành phần đóng góp chính vào độ trễ bao gồm:

- Xử lý dữ liệu IMU: Sử dụng thuật toán Madgwick hoặc EKF để tính toán góc và tư thế tay, với thời gian xử lý trung bình từ 30 đến 50 ms.
- Truyền dữ liệu và điều khiển robot: Quá trình truyền dữ liệu (qua UART/WiFi) và xử lý tọa độ để điều khiển robot diễn ra trong khoảng 50 đến 70 ms.
- Xử lý thị giác máy tính: Sử dụng camera để phát hiện vật thể và xử lý ảnh trong thời gian trung bình từ 150 đến 250 ms, tùy theo điều kiện ánh sáng và độ phức tạp cảnh vật.

Tổng hợp lại, độ trễ trung bình toàn hệ thống nằm trong khoảng 300 đến 400 ms. Đây là một mức trễ tương đối thấp, đáp ứng tốt cho các ứng dụng điều khiển robot cộng tác, bán thời gian thực như gắp thả vật thể hoặc điều khiển theo cử chỉ.

Nhận xét: Độ trễ của hệ thống hiện tại là chấp nhận được cho các bài toán tương tác người – máy thông thường. Tuy nhiên, đối với các ứng dụng yêu cầu thời gian phản hồi tức thời (dưới 100 ms), cần tối ưu thêm các khâu xử lý ảnh hoặc sử dụng phần cứng mạnh hơn (ví dụ: camera công nghiệp, vi xử lý AI chuyên dụng).

4.2.2. Đánh giá độ chính xác

Độ chính xác của hệ thống được đánh giá bằng cách so sánh tọa độ thực tế mà robot thực hiện (vị trí gắp vật) với tọa độ mục tiêu được tính toán từ cảm biến IMU và

hệ thống thị giác. Việc đo đạc được thực hiện bằng cách đặt các vật thể tại các vị trí đã biết trước, sau đó tiến hành đo sai số giữa điểm gắp thực tế và điểm lý tưởng.

Kết quả thực nghiệm cho thấy:

- Sai số trung bình do IMU gây ra: Khoảng 3-5 mm, với độ ổn định cao khi người điều khiển di chuyển tay với tốc độ vừa phải.
- Sai số trung bình từ khối thị giác máy tính: Phụ thuộc nhiều vào điều kiện ánh sáng, độ phân giải camera và thuật toán để phát hiện vật thể.

Nhận xét: Sai số tổng hợp nhỏ cho thấy hệ thống hoàn toàn phù hợp để ứng dụng trong các nhiệm vụ điều khiển robot cộng tác yêu cầu chính xác vừa phải như phân loại sản phẩm, gắp linh kiện, hoặc thao tác lắp ráp đơn giản. Trong đó, thành phần thị giác máy tính đóng vai trò quyết định đến độ chính xác tổng thể và cần được hiệu chỉnh kỹ lưỡng hơn thông qua:

- Cải thiện ánh sáng môi trường.
- Nâng cấp độ phân giải camera.
- Tối ưu thuật toán xử lý ảnh (ví dụ: sử dụng mô hình nhẹ hơn hoặc thêm bước lọc nhiễu).

4.2.3. Các lỗi phát sinh và hướng khắc phục.

Trong quá trình triển khai và vận hành hệ thống điều khiển robot sử dụng kết hợp cảm biến IMU và thị giác máy tính, một số lỗi đã được ghi nhận và phân tích như sau:

Bảng 4.1. Các lỗi phát sinh và biện pháp khắc phục

Lỗi phát sinh	Nguyên nhân	Hướng khắc phục
Nhiễu tín hiệu từ cảm biến IMU	Cảm biến chưa được cố định chắc chắn Tốc độ lấy mẫu thấp Thiếu lọc nhiễu	Gắn chắc IMU bằng đai hoặc băng dán Tăng tần số lấy mẫu lên Sử dụng bộ lọc EKF, lọc Butterworth
Sai số từ hệ thống thị giác máy tính	Ánh sáng môi trường không ổn định Độ phân giải camera thấp	Bổ sung đèn chiếu sáng định hướng

	Thuật toán phát hiện đơn giản	Nâng cấp camera
Hiệu chuẩn không chính xác	Không có bước hiệu chuẩn khi khởi động Góc đặt camera không đúng chuẩn	Thiết lập hiệu chuẩn tự động ban đầu Cố định camera và robot theo vị trí chuẩn hóa

Kết luận chương 4

Hệ thống điều khiển robot kết hợp IMU và thị giác máy tính đáp ứng yêu cầu cơ bản cho ứng dụng công nghiệp nhẹ. Độ trễ toàn hệ thống phù hợp với tác vụ bán thời gian thực như gắp thả vật thể, trong đó khâu xử lý thị giác là một bước phát triển mới. Độ chính xác ở mức chấp nhận được, nhưng phụ thuộc lớn vào điều kiện ánh sáng và chất lượng camera. Các lỗi chủ yếu liên quan đến nhiễu IMU, sai số thị giác và truyền dữ liệu, có thể khắc phục qua nâng cấp phần cứng (camera, đèn chiếu), tối ưu thuật toán, và hiệu chuẩn hệ thống. Hướng phát triển tương lai cần tập trung giảm độ trễ xử lý ảnh để mở rộng sang ứng dụng đòi hỏi thời gian thực.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Tổng kết kết quả đạt được

Đồ án "Nhận diện cử chỉ tay để điều khiển robot ABB CRB15000" đã đạt được các kết quả quan trọng sau:

5.1.1. Thiết kế hệ thống phần cứng hoàn chỉnh

Chế tạo thành công thiết bị đeo tay tích hợp các cảm biến MPU9250 và QMC5883L nhằm thu thập dữ liệu chuyển động của tay người dùng theo thời gian thực. Tín hiệu từ cảm biến được xử lý và truyền tới vi điều khiển ESP32, đóng vai trò trung tâm thu thập, xử lý sơ bộ và truyền dữ liệu về máy tính điều khiển thông qua WiFi.

Hệ thống đã xây dựng mạch PCB tối ưu với thiết kế bố trí linh kiện hợp lý giúp giảm thiểu giao thoa tín hiệu giữa các đường mạch, đảm bảo độ ổn định và chính xác cao khi thu thập dữ liệu chuyển động.

Thiết bị đeo tay được thiết kế với kiểu dáng gọn nhẹ, dễ dàng đeo và tháo lắp, không gây cản trở cử động tay. Kích thước và trọng lượng được tối ưu để tạo cảm giác thoải mái khi sử dụng lâu dài đảm bảo an toàn và độ bền trong điều kiện làm việc.

5.1.2. Phát triển phần mềm điều khiển:

Xây dựng giao diện điều khiển trên Visual Studio sử dụng ngôn ngữ Python, kết hợp với các thư viện mạnh như OpenCV và MediaPipe để thực hiện việc xử lý ảnh và nhận diện cử chỉ tay theo thời gian thực.

Giao diện được thiết kế trực quan, dễ sử dụng, cho phép người vận hành dễ dàng theo dõi tín hiệu từ cảm biến, trạng thái của robot, cũng như thực hiện các thao tác điều khiển chỉ với các cử chỉ đơn giản.

Hệ thống kết hợp giữa việc xử lý hình ảnh và điều khiển I/O robot, giúp các lệnh điều khiển được thực thi linh hoạt và chính xác. Việc tích hợp thuật toán xử lý dữ liệu từ cảm biến cùng với kết quả nhận diện từ camera đã giúp hệ thống đồng bộ đa nguồn dữ liệu một cách hiệu quả, đảm bảo độ tin cậy cao trong các tình huống vận hành thực tế.

Triển khai thành công giao tiếp giữa PC và robot ABB thông qua Robot Web Services (RWS) và Externally Guided Motion (EGM).

Việc lựa chọn kết nối không dây giúp tăng tính linh hoạt cho hệ thống, dễ dàng triển khai trong các môi trường sản xuất mà không cần bổ sung dây dẫn tín hiệu phức tạp.

5.1.3. Kết quả thử nghiệm

Hệ thống đã nhận diện chính xác các cử chỉ tay cơ bản, bao gồm: mở gripper, đóng gripper, dừng chạy EGM, và di chuyển theo các trục XYZ. Các cử chỉ được nhận dạng với độ chính xác cao trong điều kiện ánh sáng tiêu chuẩn, đồng thời đảm bảo khả năng phân biệt rõ ràng giữa các thao tác, hạn chế nhầm lẫn trong quá trình điều khiển.

Hệ thống còn tồn tại độ trễ nhưng vẫn đáp ứng được những yêu cầu của các tác vụ điều khiển trong công nghiệp đơn giản. Bên cạnh đó, sai số vị trí trung bình thấp (dưới 5 mm) trong môi trường ánh sáng ổn định cho thấy hệ thống hoàn toàn có khả năng ứng dụng trong các bài toán yêu cầu độ chính xác vừa phải.

Đồ án đã chứng minh tính khả thi của phương pháp điều khiển robot không chạm trong công nghiệp, một hướng đi hiện đại trong xu thế phát triển của các hệ thống sản xuất thông minh. Giải pháp này mở ra nhiều tiềm năng ứng dụng trong lĩnh vực tương tác người – máy (HMI), đặc biệt trong các môi trường làm việc đặc thù như: nhà máy sản xuất trong môi trường nguy hiểm, khu vực hạn chế tiếp xúc, hoặc các dây chuyền cần tối ưu hóa thao tác điều khiển mà không phụ thuộc vào các thiết bị ngoại vi truyền thống.

Đề tài vừa góp phần mang lại tính thực tiễn mà còn tạo nền tảng vững chắc cho việc phát triển các hệ thống điều khiển không chạm, góp phần nâng cao an toàn lao động, hiệu quả sản xuất, và hỗ trợ triển khai các mô hình sản xuất theo định hướng công nghiệp hiện đại.

5.2. Hạn chế của đề tài

Mặc dù hệ thống nhận diện cử chỉ tay để điều khiển robot ABB CRB15000 đã được hoàn thiện và có thể vận hành trên thực tế, tuy nhiên trong quá trình triển khai và thử nghiệm, nhóm đã gặp phải một số hạn chế nhất định như sau:

Độ trễ trong nhận diện và điều khiển: Dữ liệu thu thập từ các cảm biến chuyển động (MPU9250, QMC5883L) kết hợp với dữ liệu xử lý ảnh từ MediaPipe khi truyền qua mạng đến máy tính chủ vẫn tồn tại độ trễ gây ảnh hưởng tới tính thời gian thực của hệ thống.

Độ chính xác của nhận diện cử chỉ ảnh: Hệ thống sử dụng MediaPipe để nhận diện cử chỉ dựa trên hình ảnh từ camera, tuy nhiên độ chính xác vẫn bị ảnh hưởng trong các điều kiện ánh sáng không ổn định (ví dụ như ánh sáng ngược, ánh sáng yếu, hoặc chói sáng). Ngoài ra, một số cử chỉ phức tạp hoặc cử chỉ có sự che khuất một phần bàn tay dễ gặp nhầm lẫn cho việc nhận dạng dẫn đến có thể khiến robot nhận lệnh sai, ảnh hưởng đến tính an toàn và hiệu quả của hệ thống.

Giới hạn về phần cứng: Hệ thống chỉ sử dụng hai cảm biến chính gắn trên tay, do đó chưa thể bao quát toàn bộ chuyển động của cánh tay hoặc các khớp. Bên cạnh đó, ESP32 với năng lực xử lý vừa phải chỉ đủ đảm đương vai trò thu thập và truyền dữ liệu cơ bản, khi hệ thống yêu cầu xử lý dữ liệu lớn hơn hoặc thực hiện các thuật toán nhận diện tại chỗ, ESP32 không đáp ứng kịp về tốc độ xử lý và bộ nhớ.

Tính tương thích: Hệ thống có thể hoạt động ổn định trong môi trường mô phỏng và thử nghiệm, nhưng khi triển khai thực tế ở môi trường công nghiệp thì yêu cầu độ tin cậy của kết nối và sự an toàn cần được tăng lên. Những yếu tố như nhiễu điện từ, nhiễu mạng không dây, hoặc các yếu tố vật lý (va đập, rung động) có thể ảnh hưởng đến khả năng duy trì kết nối và độ chính xác của hệ thống.

5.3. Đề xuất hướng phát triển trong tương lai

Nâng cấp khả năng xử lý ảnh: Hệ thống có thể tích hợp các mô hình nhận diện hiện đại hơn như YOLOv8 để tăng độ chính xác trong việc nhận diện vật thể trong những môi trường có điều kiện ánh sáng phức tạp hoặc nhiều yếu tố gây nhiễu. Đồng thời, việc áp dụng các mô hình học như CNN (Convolutional Neural Network) vào nhận diện cử chỉ tay sẽ giúp hệ thống cải thiện độ chính xác hơn, giảm thiểu sai lệch và tăng tính thích ứng với nhiều điều kiện môi trường khác nhau.

Cải tiến phần cứng: Có thể thay thế cảm biến đang sử dụng bằng các phiên bản IMU hiện đại hơn nâng cao độ chính xác và ổn định cao hơn. Việc sử dụng các vi điều khiển có năng lực tính toán mạnh hơn như STM32H7, Raspberry Pi 4 hoặc các module tích hợp AI sẽ cho phép thực hiện xử lý dữ liệu một cách nhanh chóng và tăng tốc độ phản hồi.

Phát triển ứng dụng: Phần mềm điều khiển hiện tại có thể được phát triển thành Web app góp phần giúp người dùng dễ tiếp cận hệ thống, cho phép điều khiển robot từ nhiều loại thiết bị (PC, laptop, máy tính bảng, điện thoại), đồng thời mở rộng tính ứng dụng trong các môi trường sản xuất linh hoạt.

Ứng dụng vào thực tiễn: Tiếp tục triển khai hệ thống vào các ứng dụng cụ thể như: hỗ trợ phục hồi chức năng cho người khuyết tật, dây chuyền sản xuất tự động có tương tác người – máy, hoặc các khâu vận hành không chạm.

PHỤ LỤC

Phụ lục 1: Tài liệu tham khảo

[1] M. A. Sobh, Introduction to Robotics, Middle Tennessee State University Pressbooks. [Online].

Available: https://mtsu.pressbooks.pub/robotics/chapter/chapter-3/

[2] L. Sciavicco and B. Siciliano, Robotics: Modelling, Planning and Control (Extract). [Online].

Available: https://people.disim.univaq.it/~costanzo.manes/EDU_stuff/Robotics_Modelling,%20Planning%20and%20Control_Sciavicco_extract.pdf

[3] D. Wang, Forward Kinematics. Clemson University Open Textbooks. [Online].

Available: https://opentextbooks.clemson.edu/wangrobotics/chapter/forward-kinematics/

[4] ABB, RobotStudio API - Robot Web Services (RWS). [Online].

Available: https://developercenter.robotstudio.com/api/RWS

[5] Thrun S, Burgard W, Fox D. Probabilistic Robotics. Cambridge (MA): MIT Press; 2005.

Craig JJ. Introduction to Robotics: Mechanics and Control. 4th ed. Boston (MA): Pearson; 2017: https://docs.ufpr.br/~danielsantos/ProbabilisticRobotics.pdf

[6] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," University of North Carolina at Chapel Hill. [Online].

Available: https://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf

[7] C. Atkeson, "Robot Learning from Demonstration," YouTube, 2015. [Online].

Available: https://www.youtube.com/watch?v=d4EgbgTm0Bg

[8] K. Hashimoto, H. Osumi, K. Hashimoto, and T. Narikiyo, "Development of an omnidirectional mobile robot with step-climbing ability," ROBOMECH Journal, vol. 7, no. 1, pp. 1–14, 2020. [Online].

Available: https://robomechjournal.springeropen.com/articles/10.1186/s40648-020-00185y

[9] Wikipedia, "Quaternions and spatial rotation," Wikipedia, 2024. [Online]. Available: https://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation

[10] Wikipedia, "Conversion between quaternions and Euler angles," Wikipedia, 2024. [Online].

Available: https://en.wikipedia.org/wiki/Conversion_between_quaternions_and_Euler_angles

[11] https://gitlab.control.lth.se/robotlab/abb_egm_pyclient