

ĐẠI HỌC QUỐC GIA VIỆT NAM
TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG



BÙI MINH ĐỨC

ĐỒ ÁN TỐT NGHIỆP
GIẢI PHÁP ĐỊNH VỊ TRONG NHÀ CHO ROBOT DI ĐỘNG
MECANUM SỬ DỤNG THỊ GIÁC MÁY TÍNH

(INDOOR LOCALIZATION FOR MECANUM-WHEELED
MOBILE ROBOT: A VISION-BASED LOCALIZATION
FRAMEWORK)

CỦ NHÂN NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

TP. HỒ CHÍ MINH, 2025

ĐẠI HỌC QUỐC GIA VIỆT NAM
TRƯỜNG ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỀU KHIỂN TỰ ĐỘNG

BÙI MINH ĐỨC - 2110130

ĐỒ ÁN TỐT NGHIỆP
GIẢI PHÁP ĐỊNH VỊ TRONG NHÀ CHO ROBOT DI ĐỘNG
MECANUM SỬ DỤNG THỊ GIÁC MÁY TÍNH
(INDOOR LOCALIZATION FOR MECANUM-WHEELED
MOBILE ROBOT: A VISION-BASED LOCALIZATION
FRAMEWORK)

CỦ NHÂN NGÀNH KỸ THUẬT ĐIỀU KHIỂN & TỰ ĐỘNG HÓA

GIẢNG VIÊN HƯỚNG DẪN
TS. NGUYỄN TRỌNG TÀI

TP. HỒ CHÍ MINH, 2025

**KHÓA LUẬN TỐT NGHIỆP ĐƯỢC HOÀN THÀNH TẠI
TRƯỜNG ĐẠI HỌC BÁCH KHOA -ĐHQG -HCM**

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

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

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

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

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

1.
2.
3.
4.
5.

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

CHỦ TỊCH HỘI ĐỒNG

CHỦ NHIỆM BỘ MÔN.....

TP. HCM, ngày....tháng....năm.....
NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP
CỦA CÁN BỘ HƯỚNG DẪN

Tên Đồ án:

**GIẢI PHÁP ĐỊNH VỊ TRONG NHÀ CHO ROBOT DI ĐỘNG
MECANUM SỬ DỤNG THỊ GIÁC MÁY TÍNH**
**(INDOOR LOCALIZATION FOR MECANUM-WHEELED MOBILE
ROBOT: A VISION-BASED LOCALIZATION FRAMEWORK)**

Nhóm sinh viên thực hiện:

Bùi Minh Đức

2110130

Cán bộ hướng dẫn:

TS. Nguyễn Trọng Tài

Đánh giá Đồ án:

1. Về cuốn báo cáo:

Số trang _____

Số chương _____

Số bảng số liệu _____

Số hình vẽ _____

Số tài liệu tham khảo _____

Sản phẩm _____

Một số nhận xét về hình thức cuốn báo cáo:

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

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

2. Về nội dung Đồ án:

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

.....

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

3. Về tính ứng dụng:

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

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

4. Về thái độ làm việc của sinh viên:

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

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

Đánh giá chung: Đồ án đạt/không đạt yêu cầu của một Đồ án tốt nghiệp cử nhân, xếp loại Giỏi/Khá/Trung bình

Điểm của từng sinh viên:

Bùi Minh Đức:/10

Cán bộ hướng dẫn
(Ký và ghi rõ họ tên)

TP. HCM, ngày....tháng....năm.....
NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP
CỦA CÁN BỘ PHẢN BIỆN

Tên Đồ án:

**GIẢI PHÁP ĐỊNH VỊ TRONG NHÀ CHO ROBOT DI ĐỘNG
MECANUM SỬ DỤNG THỊ GIÁC MÁY TÍNH**
**(INDOOR LOCALIZATION FOR MECANUM-WHEELED MOBILE
ROBOT: A VISION-BASED LOCALIZATION FRAMEWORK)**

Nhóm sinh viên thực hiện:

Bùi Minh Đức

2110130

Cán bộ phản biện:

Đánh giá Đồ án:

1. Về cuốn báo cáo:

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

Một số nhận xét về hình thức cuốn báo cáo:

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

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

2. Về nội dung Đồ án:

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

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

3. Về tính ứng dụng:

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

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

4. Về thái độ làm việc của sinh viên:

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

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

Đánh giá chung: Đồ án đạt/không đạt yêu cầu của một Đồ án tốt nghiệp cử nhân, xếp loại Giỏi/Khá/Trung bình

Điểm của từng sinh viên:

Bùi Minh Đức:...../10

Người nhận xét
(Ký và ghi rõ họ tên)

TP. HCM, ngày....tháng....năm.....

ĐỀ CƯƠNG CHI TIẾT

TÊN ĐỀ ÁN: GIẢI PHÁP ĐỊNH VỊ TRONG NHÀ CHO ROBOT DI ĐỘNG MECANUM SỬ DỤNG THỊ GIÁC MÁY TÍNH

Cán bộ hướng dẫn: TS. Nguyễn Trọng Tài

Thời gian thực hiện: Từ ngày 01/05/2025 đến ngày 30/08/2025

Sinh viên thực hiện: Bùi Minh Đức

Nội dung đề tài: (Mô tả chi tiết mục tiêu, phạm vi, đối tượng, phương pháp thực hiện, kết quả mong đợi của đề tài)

Mục tiêu:

Tóm tắt các mô hình toán học của robot di động bốn bánh Mecanum, bao gồm động học và động lực học.

Đề xuất bộ điều khiển Back-stepping kết hợp điều khiển trượt để bám vị trí và tốc độ của robot.

Xây dựng hệ thống định vị sử dụng camera gắn trên robot để nhận diện và ước tính vị trí cũng như hướng của các ArUco Marker được bố trí trên bản đồ, từ đó xác định chính xác vị trí và hướng của robot trong hệ tọa độ thế giới thực.

Đề xuất phương pháp tích hợp dữ liệu từ IMU, encoder và thông tin vị trí, góc hướng thu được từ ArUco Marker, kết hợp xử lý bằng bộ ước lượng Extended Kalman Filter hoặc các biến thể nâng cao như Adaptive Extended Kalman Filter và Extended Kalman Filter chi bình phương, nhằm ước tính chính xác và ổn định hơn góc hướng cũng như trạng thái chuyển động của robot.

Xây dựng mô hình thực tế của robot để đánh giá hiệu quả của phương pháp điều khiển.

Phân tích kết quả mô phỏng và thực nghiệm.

Phạm vi của đề tài:

Đồ án Tốt nghiệp tập trung vào việc định vị và điều hướng Robot Mecanum. Trọng tâm của đề tài là xây dựng và tích hợp hệ thống định vị dựa trên camera với Aruco Marker, kết hợp với thông tin từ encoder và IMU, sau đó sử dụng bộ ước lượng EKF và AEKF để ước lượng chính xác vị trí và góc hướng của robot. Việc định vị chính xác là nền tảng quan trọng để nâng cao hiệu quả của các bộ điều khiển trong bài toán bám quỹ đạo. Song song với đó, đề tài triển khai và so sánh hiệu quả của hai phương pháp điều khiển phi tuyến Backstepping và điều khiển trượt (SMC) dựa trên mô hình toán học của robot.

Đối tượng:

Đối tượng của Đồ án Tốt nghiệp là một Robot Mecanum 4 bánh, được truyền động độc lập bởi bốn động cơ DC. Nhờ cấu trúc bánh Mecanum, robot có khả năng di chuyển đa hướng (omnidirectional) như tiến, lùi, đi ngang, quay tại chỗ và kết hợp nhiều dạng chuyển động. Robot được trang bị hệ thống định vị dựa trên camera gắn phía trước robot để phát hiện các Aruco Marker trên bản đồ, đồng thời sử dụng bộ ước lượng ước lượng trạng thái EKF để kết hợp thông tin từ camera, encoder và IMU, cho phép ước lượng chính xác vị trí và góc hướng của robot trong không gian.

Phương pháp thực hiện:

Thiết lập mô hình toán học của Robot Mecanum, bao gồm động học và động lực học, làm cơ sở lý thuyết cho việc thiết kế bộ điều khiển.

Thiết kế và mô phỏng bộ điều khiển Backstepping kết hợp điều khiển trượt, đánh giá hiệu quả trong bài toán bám quỹ đạo.

Triển khai bộ điều khiển trên Robot Mecanum thực tế, áp dụng cho hệ thống điều khiển động lực học.

Xây dựng hệ thống camera thu nhận thông tin từ Aruco Marker và kết hợp với dữ liệu cảm biến thông qua EKF để định vị chính xác.

Tiến hành thực nghiệm, thu thập dữ liệu và phân tích để đánh giá hiệu suất điều khiển, so sánh giữa mô phỏng và thực tế.

Kết quả mong muốn:

Một hệ thống định vị robot mecanum hoàn chỉnh với sai số vị trí và hướng thấp và khả năng bám quỹ đạo mong muốn.

Kế hoạch thực hiện: (Mô tả kế hoạch làm việc và phân công công việc cho từng sinh viên tham gia)

Bùi Minh Đức

Giai đoạn 1: Nghiên cứu động học và động lực học của robot Mecanum. Nghiên cứu thuật toán điều khiển Back-stepping cho robot Mecanum.

Giai đoạn 2: Mô phỏng thuật toán điều khiển trong môi trường MATLAB/Simulink. Nghiên cứu thuật toán định vị nhận diện AruCo marker.

Giai đoạn 3: Nghiên cứu về ROS để xây dựng cấu trúc hệ thống. Nghiên cứu bộ ước lượng trạng thái EKF (Extended Kalman Filter) và các biến thể của EKF.

Giai đoạn 4: Xây dựng phần cứng để triển khai bộ điều khiển động lực học cho mô hình robot thực tế. Nghiên cứu vi điều khiển STM32, thiết kế mạch kết nối STM32F411 và driver motor

Giai đoạn 5: Tiến hành thực nghiệm để phân tích và đánh giá kết quả.

Giai đoạn 6: Xây dựng giao diện điều khiển để làm giao diện quan sát và điều khiển robot.

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

DANH SÁCH HỘI ĐỒNG BẢO VỆ ĐỒ ÁN

Hội đồng chấm Đồ án tốt nghiệp, thành lập theo Quyết định số ngày.....
của Hiệu trưởng Trường Đại học Bách Khoa TP. HCM.

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

LỜI CẢM ƠN

Đầu tiên, em xin chân thành cảm ơn các thầy cô giảng viên tại Trường Đại học Bách Khoa Thành phố Hồ Chí Minh, đặc biệt là những thầy, cô thuộc Khoa Điện - Điện tử, vì sự hướng dẫn tận tình, lời khuyên rất có giá trị và sự kiên nhẫn hỗ trợ của các thầy, cô trong suốt thời gian học tại trường. Những kiến thức và kinh nghiệm của quý thầy cô là hành trang quý báu cho chúng em trên con đường học tập, nghiên cứu và phát triển sau này.

Tiếp theo, em xin chân thành gửi lời cảm ơn đến thầy Nguyễn Trọng Tài đã tận tình hướng dẫn, giúp đỡ về các kiến thức và ý tưởng trong suốt quá trình thực hiện đề tài. Thầy đã tạo điều kiện và môi trường thuận lợi nhất giúp em hoàn thành những mục tiêu mà đề tài đề ra.

Ngoài ra, em xin gửi lời tri ân sâu sắc nhất đến gia đình, vì đã là nguồn động viên to lớn về mặt tinh thần và trụ cột vững chắc, giúp em vững tâm trên con đường thực hiện đề tài. Cảm ơn các bạn, anh chị đã nhiệt tình chia sẻ cả về vật chất lẫn tinh thần trong những lúc khó khăn, nhờ đó em có thể hoàn thiện và thực hiện đề tài này một cách toàn diện và chu đáo nhất.

Em xin chân thành cảm ơn!

TP. HCM, ngày 10 tháng 9 năm 2025

Sinh viên

Bùi Minh Đức

MỤC LỤC

TÓM TẮT ĐỀ TÀI	1
CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN	2
1.1. Lý do chọn đề tài	2
1.2. Tính cấp thiết của việc sử dụng bánh xe Mecanum	3
1.3. So sánh bánh xe Mecanum và bánh xe thông thường	4
1.4. Ứng dụng thực tiễn	5
1.5. Mục tiêu của Đề án tốt nghiệp	7
1.6. Cấu trúc tổng quan của hệ thống	8
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	11
2.1. Mô hình động học và động lực học hệ robot di động	11
2.1.1. Mô hình toán học robot Mecanum	11
2.1.2. Mô hình động học	11
2.1.3. Mô hình động lực học	13
2.2 Điều khiển backstepping	14
2.2.1. Phương trình động học và động lực học cho bộ điều khiển	15
2.2.2. Thiết kế bộ điều khiển backstepping	15
2.3. Điều khiển trượt	16
2.3.1. Mô hình động lực học bánh xe	17
2.3.2. Sai số và mặt trượt	17
2.3.3. Luật điều khiển Sliding Mode	17
2.4. AruCo Marker	18
2.4.1. Hệ thống định vị bằng Camera	18
2.4.2. Tổng quan về phương pháp Aruko marker	18
2.4.3. Camera calibration	19
2.4.4. Đo lường vị trí bằng AruCo Marker	22
2.5. Sử dụng các bộ ước lượng Kalman	25
2.5.1. EKF	25
2.5.2. AEKF	28
2.5.3. EKF kết hợp với kiểm định chi bình phương	28
2.6. Mô hình hóa và ước lượng trạng thái từ dữ liệu cảm biến	29
2.6.1. Tính toán định hướng từ IMU	29
2.6.2. Tính toán vận tốc và vị trí từ encoder	29
CHƯƠNG 3: MÔ PHỎNG MATLAB	31
3.1. Tổng quan mô hình và thông số mô phỏng	31
3.2 Kết quả mô phỏng	31

CHƯƠNG 4: THI CÔNG PHẦN CỨNG	35
4.1. Sơ đồ kết nối	35
4.1.1. Sơ đồ công suất	35
2.5.2. Sơ đồ kết nối các thành phần	36
4.2. Thiết kế mạch cho phần cứng	37
4.3. Chi tiết phần cứng của hệ thống	38
4.3.1. Máy tính công nghiệp (IPC)	38
4.3.2. Động cơ encoder	39
4.3.3. Vi điều khiển (MCU)	40
4.3.4. Cảm biến định hướng (IMU)	40
4.3.5 Camera phát hiện ArUco	41
4.3.6. 9 Channel Servo Shield Driver Board for Arduino ATmega 2560	42
CHƯƠNG 5: THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ	43
5.1. Các bước thực nghiệm	43
5.2. Thiết kế giao diện người dùng	46
5.3. Kết quả thực nghiệm	47
5.3.1 Kết quả đáp ứng vận tốc của bộ điều khiển trượt	48
5.3.2. Quỹ đạo đường thẳng theo trục Y	49
5.3.3. Quỹ đạo đường thẳng theo trục X	51
5.3.4. Quỹ đạo hình vuông	54
5.3.5. Quỹ đạo hình tròn	61
5.3.6. Phương pháp đánh giá sai số	65
CHƯƠNG 6: KẾT LUẬN	67
6.1. Kết quả đạt được	67
6.2. Những hạn chế	67
6.3. Hướng phát triển	68
TÀI LIỆU THAM KHẢO	69

DANH MỤC HÌNH ẢNH

Hình 1.1: Mecanum robot.....	2
Hình 1.2: Bánh xe Mecanum.....	3
Hình 1.3: Các hướng di chuyển của xe mecanum	3
Hình 1.4: Các robot Kiva của Amazon đang vận chuyển các kệ hàng.....	6
Hình 1.5: Ứng dụng trong công nghiệp của mecanum robot	7
Hình 1.6: Tổng quan sơ đồ tín hiệu của hệ thống.....	9
Hình 1.7: Sơ đồ tín hiệu của hệ thống của hệ thống khi thực hiện thực tế.....	9
Hình 1.8: Cấu trúc phần cứng của mô hình thực tế	10
Hình 2.1: Mô hình động học của robot mecanum	13
Hình 2.2: Mô hình động lực học robot mecanum.....	13
Hình 2.3: AruCo Marker kích thước 4x4.....	18
Hình 2.4: Camera Calibration Flowchart	20
Hình 2.5: Camera Calibration Flowchart	21
Hình 2.6: Hình ảnh calib checker board	22
Hình 2.7: Kết quả Camera calibration	22
Hình 2.8: Mối quan hệ giữa tọa độ tâm ArUco và tọa độ máy ảnh	23
Hình 2.9: Sơ đồ khối 2 phương pháp AEKF và EKF kết hợp kiểm định χ^2	27
Bảng 3.1: Thông số mô phỏng.....	31
Hình 3.2: Kết quả mô phỏng bám quỹ đạo hình vuông.....	32
Hình 3.3: Sai số vị trí và góc mô phỏng bám quỹ đạo hình vuông	32
Hình 3.4: Kết quả mô phỏng bám quỹ đạo hình tròn	32
Hình 3.5: Sai số vị trí và góc mô phỏng bám quỹ đạo hình tròn	32
Hình 3.6: Kết quả mô phỏng bám quỹ đạo hình elipse	33
Hình 3.7: Sai số vị trí và góc mô phỏng bám quỹ đạo hình elipse	33
Hình 3.8: Kết quả mô phỏng bám quỹ đạo hình số 8	33
Hình 3.9: Sai số vị trí và góc mô phỏng bám quỹ đạo hình số 8.....	34
Hình 4.1: Sơ đồ khối công suất	35
Hình 4.2: Sơ đồ kết nối các thành phần.....	36
Hình 4.3: Thiết kế sơ đồ mạch cho phần cứng	37
Hình 4.4: Mạch thực tế sau khi hàn	37
Hình 4.5: Mạch khi được sử dụng	38
Hình 4.6: Máy tính nhúng công nghiệp	38
Hình 4.7: Động cơ DC giảm tốc CHR-GM37-520 (with Encoder)	39
Hình 4.8: Vi điều khiển STM32F411VG Discovery	40
Hình 4.8: Cảm biến gia tốc BNO055	41

Hình 4.9: Webcam HD 1080	41
Hình 4.10: 4Motor&9Servo Driver board	42
Hình 5.1: Hệ điều hành Ubuntu.....	43
Hình 5.2: Phần mềm STM32CubeIDE.....	43
Hình 5.3: Phần mềm Anydesk	44
Hình 5.4: Mô hình robot kết hợp với định vị bằng ArUco marker trong hệ tọa độ toàn cục (global frame)	44
Hình 5.5: Setup cho hình tròn góc thay đổi.....	45
Hình 5.6: Setup cho hình vuông góc không thay đổi	46
Hình 5.7: Giao diện thiết kế bằng Qt5	47
Hình 5.8: Mô hình robot thực nghiệm.....	48
Hình 5.9: Đáp ứng vận tốc của bánh xe trước khi áp dụng bộ điều khiển trượt	48
Hình 5.10: Sai số vận tốc của bánh xe trước khi áp dụng bộ điều khiển trượt.....	48
Hình 5.11: Đáp ứng vận tốc của bánh xe sau khi áp dụng bộ điều khiển trượt.....	49
Hình 5.12: Sai số vận tốc của bánh xe sau khi áp dụng bộ điều khiển trượt.....	49
Hình 5.13: Quỹ đạo robot bám quỹ đạo đường thẳng	50
Hình 5.14: Đáp ứng robot quỹ đạo đường thẳng theo trục X.....	50
Hình 5.15: Đáp ứng robot quỹ đạo đường thẳng theo trục Y	51
Hình 5.16: Đáp ứng robot quỹ đạo đường thẳng theo góc yaw.....	51
Hình 5.17: Quỹ đạo robot bám quỹ đạo đường ngang	52
Hình 5.18: Đáp ứng robot quỹ đạo đường ngang theo trục X	52
Hình 5.19: Đáp ứng robot quỹ đạo đường ngang theo trục Y	53
Hình 5.20: Đáp ứng robot quỹ đạo đường ngang theo góc yaw	53
Hình 5.21: Quỹ đạo robot bám quỹ đạo hình vuông với EKF	55
Hình 5.22: Đáp ứng robot quỹ đạo hình vuông theo trục X với EKF	55
Hình 5.23: Đáp ứng robot quỹ đạo hình vuông theo trục Y với EKF	56
Hình 5.24: Đáp ứng robot quỹ đạo hình vuông theo góc yaw với EKF	56
Hình 5.25: Quỹ đạo robot bám quỹ đạo hình vuông với AEKF.....	57
Hình 5.26: Đáp ứng robot quỹ đạo hình vuông theo trục X với AEKF	57
Hình 5.27: Đáp ứng robot quỹ đạo hình vuông theo trục Y với AEKF	58
Hình 5.28: Đáp ứng robot quỹ đạo hình vuông theo góc yaw với AEKF	58
Hình 5.29: Quỹ đạo robot bám quỹ đạo hình vuông với EKF chi bình phương	59
Hình 5.30: Đáp ứng robot quỹ đạo hình vuông theo trục X với EKF chi bình phương	59
Hình 5.31: Đáp ứng robot quỹ đạo hình vuông theo trục Y với EKF chi bình phương	60
Hình 5.32: Đáp ứng robot quỹ đạo hình vuông theo góc yaw với EKF chi bình phương	60
Hình 5.33: Quỹ đạo robot bám quỹ đạo hình tròn với EKF.....	61

Hình 5.34: Đáp ứng robot quỹ đạo tròn theo trục x, y, yaw với EKF	62
Hình 5.35: Quỹ đạo robot bám quỹ đạo hình tròn với AEKF	62
Hình 5.36: Đáp ứng robot quỹ đạo tròn theo trục x, y, yaw với AEKF.....	63
Hình 5.37: Quỹ đạo robot bám quỹ đạo hình tròn với EKF chỉ bình phương.....	63
Hình 5.38: Đáp ứng robot quỹ đạo tròn theo trục x, y, yaw với EKF chỉ bình	64
Hình 5.39: Giá trị χ^2 và χ^2 threshold	65

DANH MỤC BẢNG

Bảng 1.1: So sánh bánh xe Mecanum và bánh xe thông thường.....	4
Bảng 1.2: Bảng tổng quan phần cứng của hệ thống	10
Bảng 2.1. Bảng thông số kỹ thuật.....	11
Bảng 3.1: Thông số mô phỏng	31
Bảng 3.2 Hiệu suất bám quỹ đạo của bộ điều khiển khi mô phỏng	34
Bảng 4.1: Thông số kỹ thuật máy tính nhúng.....	39
Bảng 4.2: Thông số kỹ thuật động cơ CHR-GM37-520	39
Bảng 4.3: Thông số kỹ thuật vi điều khiển STM32F411	40
Bảng 4.4: Thông số kỹ thuật cảm biến gia tốc BNO055	41
Bảng 4.5: Thông số kỹ thuật webcam	42
Bảng 4.6: Thông số kỹ thuật 4Motor&9Servo Driver board.....	42
Bảng 5.1: Kết quả đáp ứng vận tốc bánh xe	49
Bảng 5.2: Kết quả đáp ứng theo đường thẳng	51
Bảng 5.3: Kết quả đáp ứng theo đường ngang	53
Bảng 5.4: Thông số lựa chọn cho các bộ ước lượng	54
Bảng 5.5: Kết quả sai số của quỹ đạo hình vuông cho 3 bộ ước lượng	66
Bảng 5.6: Kết quả sai số của quỹ đạo hình tròn cho 3 bộ ước lượng	66

DANH MỤC VIẾT TẮT

Từ khóa	Tên Tiếng Anh
FMWMR	Four Mecanum-Wheeled Mobile Robot
IMU	Inertial Measurement Unit
GUI	Graphical User Interface
EKF	Extended Kalman Filter
AEKF	Adaptive Extended Kalman Filter
UART	Universal Asynchronous Receiver / Transmitter
SMC	Sliding mode control
TCP/IP	Transmission Control Protocol/Internet Protocol
DC	Direct Current
ID	Identification

TÓM TẮT ĐỀ TÀI

Trong những năm gần đây, nhu cầu ứng dụng robot di động trong các lĩnh vực dịch vụ, vận chuyển, kiểm kho hay hướng dẫn trong các tòa nhà, khách sạn, bệnh viện ngày càng tăng cao. Một trong những thách thức lớn nhất của robot di động hoạt động trong không gian trong nhà chính là khả năng định vị chính xác, do hệ thống định vị vệ tinh toàn cầu (GPS) không thể hoạt động trong môi trường khép kín. Việc đảm bảo robot có thể xác định chính xác vị trí và hướng di chuyển của mình là yếu tố then chốt để thực hiện các tác vụ tự động một cách an toàn và hiệu quả.

Trong luận văn này, một giải pháp định vị chính xác cho robot di động bốn bánh sử dụng cơ cấu bánh Mecanum được đề xuất và triển khai. Giải pháp được xây dựng trên cơ sở tích hợp đa cảm biến, bao gồm cảm biến quán tính (IMU), encoder và camera (nhận diện marker ArUco). Để kết hợp và xử lý thông tin từ các cảm biến, hệ thống sử dụng các bộ ước lượng như Kalman mở rộng (EKF), Kalman mở rộng thích nghi (AEKF) và EKF kết hợp kiểm định thống kê χ^2 -testing. Cách tiếp cận này cho phép hạn chế và hiệu chỉnh các sai số tích lũy riêng lẻ của từng loại cảm biến, từ đó cải thiện độ tin cậy và độ chính xác trong việc xác định vị trí của robot.

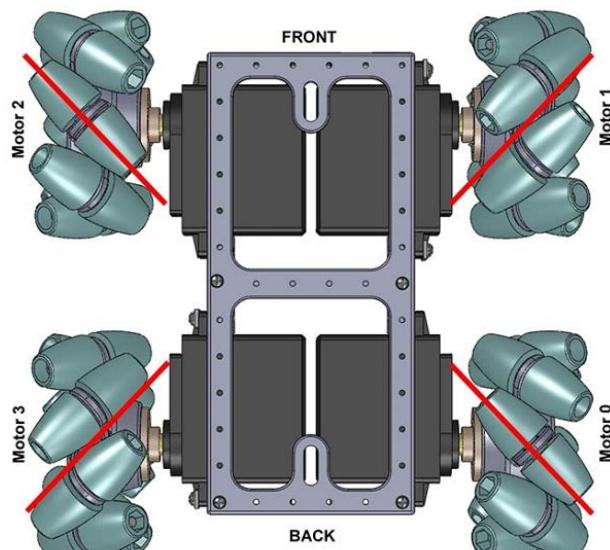
Bên cạnh việc giải quyết bài toán định vị, luận văn cũng xây dựng và áp dụng bộ điều khiển Backstepping kết hợp bộ điều khiển trượt nhằm điều khiển chuyển động của robot. Phương pháp này giúp robot bám theo quỹ đạo đặt trước một cách ổn định và chính xác, đồng thời đảm bảo tính linh hoạt trong các tình huống vận hành thực tế.

Kết quả nghiên cứu cho thấy, giải pháp định vị và điều khiển được đề xuất không chỉ đáp ứng tốt yêu cầu hoạt động của robot trong môi trường trong nhà mà còn có tiềm năng ứng dụng làm nền tảng phát triển các hệ thống xe tự hành hoặc robot dịch vụ trong nhiều lĩnh vực khác nhau. Đặc biệt, việc khắc phục hạn chế về định vị trong không gian khép kín mang lại ý nghĩa thực tiễn quan trọng, góp phần thúc đẩy việc ứng dụng robot di động vào đời sống và sản xuất.

CHƯƠNG 1: GIỚI THIỆU TỔNG QUAN

1.1. Lý do chọn đề tài

Robot Mecanum là một loại robot di động có cấu trúc đặc biệt, cho phép di chuyển linh hoạt theo mọi hướng mà không cần thay đổi tư thế quay của thân robot. Nhờ đặc tính ưu việt này, robot Mecanum ngày càng trở thành một đối tượng nghiên cứu quan trọng trong lĩnh vực robot di động. Với khả năng cơ động cao, loại robot này đã và đang được ứng dụng rộng rãi trong nhiều lĩnh vực như công nghiệp tự động hóa, dịch vụ, y tế và giáo dục. Bên cạnh đó, robot Mecanum cũng thường được lựa chọn làm nền tảng để nghiên cứu các thuật toán điều khiển, thử nghiệm những phương pháp định vị và tích hợp các công nghệ mới trong lĩnh vực robot học.

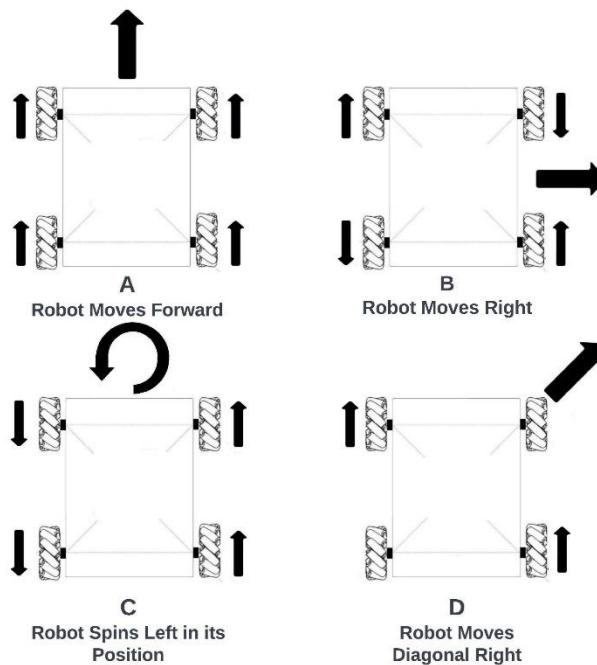


Hình 1.1: Mecanum robot

Bánh xe Mecanum được thiết kế đặc biệt với nhiều con lăn nhỏ bố trí xung quanh và đặt nghiêng một góc 45° so với trục quay chính của bánh. Dựa vào hướng sắp xếp các con lăn, bánh xe Mecanum có thể được phân thành hai loại: bánh trái và bánh phải. Trong quá trình hoạt động, các bánh xe này quay với tốc độ và hướng khác nhau, từ đó tạo ra lực tổng hợp theo phuơng mong muôn, giúp robot có khả năng di chuyển theo mọi hướng trên mặt phẳng. Nhờ đặc tính này, robot sử dụng bánh xe Mecanum có thể dễ dàng tránh chướng ngại vật, thực hiện các thao tác phức tạp như quay tại chỗ hoặc di chuyển theo quỹ đạo đặc biệt (ví dụ: đường zigzag), đáp ứng tốt các yêu cầu linh hoạt trong điều khiển và vận hành.



Hình 1.2: Bánh xe Mecanum



Hình 1.3: Các hướng di chuyển của xe mecanum

1.2. Tính cấp thiết của việc sử dụng bánh xe Mecanum

Bánh xe Mecanum mang lại cho robot khả năng di chuyển theo bất kỳ hướng nào mà không cần thay đổi hướng của thân xe. Tính năng này đặc biệt hữu ích trong các ứng dụng thực tế như robot giao hàng tự động, phương tiện vận chuyển trong nhà máy, lĩnh vực xây dựng, sản xuất cũng như y tế. Trong những môi trường làm việc có không gian hạn chế, bánh xe Mecanum cho phép robot dễ dàng vượt qua rào cản về di chuyển, đồng thời thực hiện các thao tác phức tạp một cách linh hoạt và hiệu quả.

1.3. So sánh bánh xe Mecanum và bánh xe thông thường

Bảng 1.1: So sánh bánh xe Mecanum và bánh xe thông thường

Tiêu chí	Bánh xe Mecanum	Bánh xe thông thường
Cấu tạo	Gồm nhiều con lăn đặt nghiêng 45° quanh vành, chế tạo phức tạp và yêu cầu chính xác.	Đơn giản, chỉ gồm vành và lốp, dễ chế tạo và lắp đặt.
Khả năng di chuyển	Di chuyển đa hướng: tiến, lùi, ngang, chéo, quay tại chỗ mà không cần đổi hướng thân robot.	Chỉ di chuyển tiến, lùi và rẽ nhở quay thân xe.
Ứng dụng	Robot dịch vụ, robot công nghiệp trong không gian hẹp, robot nghiên cứu điều khiển và định vị.	Xe đẩy, robot đơn giản, các hệ thống cần tải trọng lớn nhưng ít yêu cầu linh hoạt.
Độ phức tạp	Cao, cần thuật toán điều khiển và đồng bộ chính xác giữa các bánh.	Thấp, dễ điều khiển và bảo trì.
Khả năng tải trọng	Thấp hơn bánh thường do các con lăn hạn chế diện tích tiếp xúc và độ bền.	Cao, thích hợp cho vận chuyển nặng.
Giá cả	Thường có giá đắt hơn do cấu trúc phức tạp	Thường có giá thành thấp hơn

Có thể nhận thấy rằng công nghệ bánh xe Mecanum mang lại nhiều ưu điểm vượt trội và đã được ứng dụng rộng rãi trong các hệ thống robot di động tiên tiến cũng như các giải pháp định vị tự động. Nhờ khả năng di chuyển đa hướng, loại bánh xe này cho phép robot vận hành linh hoạt, trơn tru và hiệu quả trên bề mặt phẳng.

Tuy nhiên, công nghệ bánh xe Mecanum hiện chưa được triển khai phổ biến trong lĩnh vực ô tô thương mại. Nguyên nhân chủ yếu xuất phát từ những hạn chế về tính ứng dụng thực tế, sự khác biệt về nhu cầu thị trường cũng như các yếu tố liên quan đến sự tiện nghi và an toàn của người sử dụng. Người tiêu dùng thường ưu tiên các tiêu chí như tiết kiệm nhiên liệu, độ an toàn và sự thoải mái trong vận hành. Trong khi đó, công nghệ bánh xe Mecanum chưa mang lại lợi ích rõ ràng ở những khía cạnh này, thậm chí có thể làm giảm độ êm ái và ổn định khi di chuyển.

Bên cạnh đó, không phải mọi điều kiện đường sá đều thích hợp cho việc sử dụng bánh xe Mecanum. Trong các tình huống đặc thù như phanh khẩn cấp hoặc đổi hướng đột ngột, bánh xe truyền thống thường cho độ ổn định cao hơn, hạn chế hiện tượng trượt ngang. Ngược lại, do đặc điểm cấu tạo, bánh xe Mecanum có thể làm giảm độ bám đường trong các tình huống này, dẫn đến nguy cơ mất ổn định và gia tăng khả năng xảy ra tai nạn. Đây chính là một trong những rào cản quan trọng khiến công nghệ bánh xe Mecanum chưa thể áp dụng rộng rãi trong các phương tiện vận tải thương mại.

1.4. Ứng dụng thực tiễn

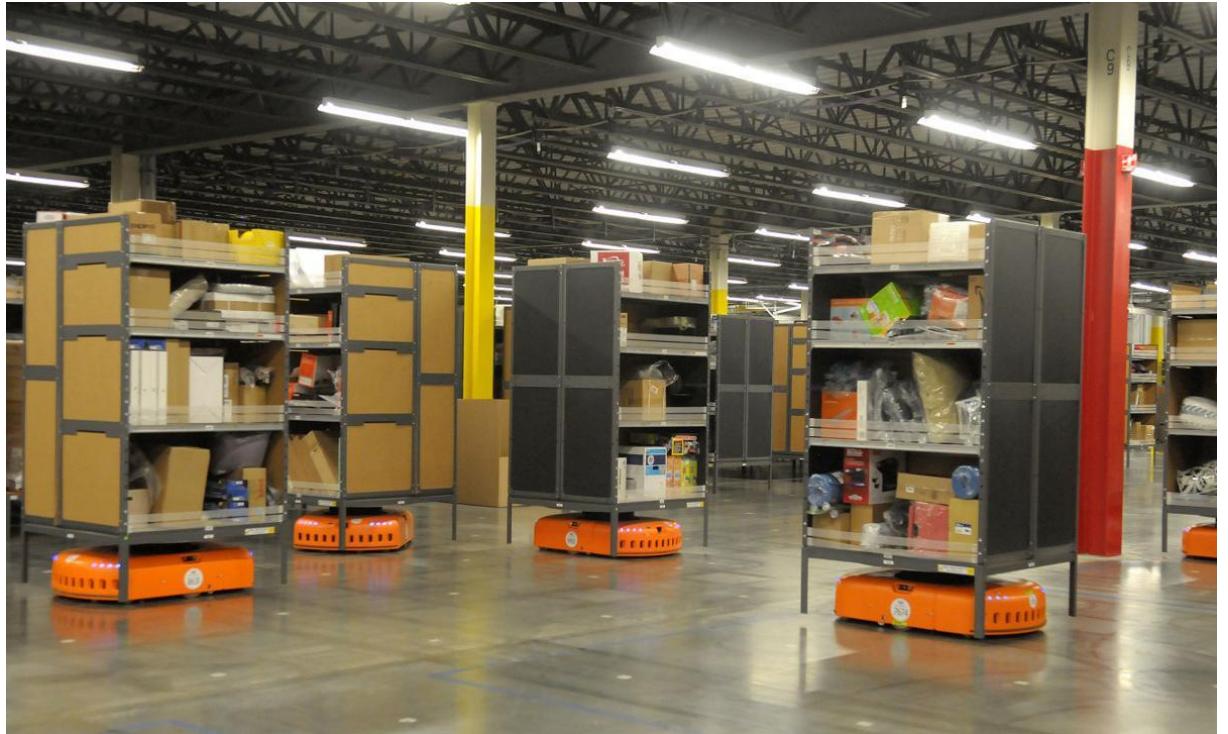
Robot Mecanum hiện được ứng dụng rộng rãi trong các nghiên cứu liên quan đến robot di động và robot tự hành. Các hướng nghiên cứu tập trung chủ yếu vào việc thiết kế, phát triển và đánh giá hiệu quả ứng dụng của robot Mecanum trong nhiều lĩnh vực khác nhau, chẳng hạn như điều khiển tự động, kiểm tra và giám sát trong nhà máy, hay các hệ thống điều khiển dựa trên tầm nhìn máy tính.

Bên cạnh đó, còn có nhiều công trình nghiên cứu chuyên sâu về các khía cạnh khác của robot Mecanum, bao gồm điều khiển chuyển động, định vị, cũng như các thuật toán điều hướng trong môi trường phức tạp. Những nghiên cứu này đã đóng góp quan trọng vào việc hoàn thiện công nghệ robot Mecanum, từ đó mở rộng phạm vi ứng dụng trong thực tiễn.

Dưới đây là một số ứng dụng tiêu biểu của robot Mecanum trong đời sống và sản xuất:

Trong lĩnh vực công nghiệp, bánh xe Mecanum có thể được ứng dụng trên các phương tiện nâng hạ, chẳng hạn như xe nâng hàng công nghiệp. Nhờ đặc tính di chuyển đa hướng, loại bánh xe này mang lại hiệu quả vượt trội trong các tình huống yêu cầu khả năng cơ động cao trong không gian hạn chế, chẳng hạn khi vận chuyển hàng hóa

dài qua các lối đi hẹp hoặc cửa có kích thước tiêu chuẩn. Bên cạnh đó, khả năng di chuyển linh hoạt theo mọi hướng giúp phương tiện trang bị bánh Mecanum trở thành lựa chọn tối ưu để làm việc trong những môi trường chật hẹp, nơi việc quay đầu gần như không thể thực hiện và đòi hỏi sự điều khiển chính xác.



Hình 1.4: Các robot Kiva của Amazon đang vận chuyển các kệ hàng

Mỗi robot Kiva có khối lượng khoảng 145 kg, được thiết kế để di chuyển linh hoạt dưới sàn nhà kho và đảm nhận nhiệm vụ vận chuyển các kệ hàng nặng tới 340 kg với chiều cao vượt quá một người trưởng thành. Quá trình vận hành của hệ thống diễn ra một cách tuần tự, khoa học và chính xác, tạo ra một khung cảnh nhộn nhịp giống như trong các nhà máy hiện đại thường thấy trong phim khoa học viễn tưởng. Vai trò của con người trong quy trình chủ yếu chỉ dừng lại ở việc sắp xếp hàng hóa lên kệ, còn toàn bộ công việc vận chuyển và sắp xếp được robot đảm nhiệm.

Theo báo cáo của Amazon, hiện nay công ty đã triển khai hơn 15.000 robot Kiva tại các trung tâm phân phối và nhà kho trên khắp nước Mỹ. Trung bình, mỗi robot Kiva có thể vận chuyển khoảng 700.000 sản phẩm mỗi ngày, và trong giai đoạn cao điểm con số này có thể tăng lên đến 1,5 triệu sản phẩm. Việc ứng dụng robot trong kho hàng mang lại nhiều lợi ích đáng kể, bao gồm tiết kiệm diện tích do không cần duy trì lối đi cho con người, từ đó gia tăng khả năng lưu trữ kệ hàng. Đồng thời, hệ thống robot giúp giảm

thiểu thời gian chờ, nâng cao hiệu quả luân chuyển hàng hóa và đảm bảo tính liên tục trong hoạt động logistics.



Hình 1.5: Ứng dụng trong công nghiệp của mecanum robot

Trong lĩnh vực y tế, robot di động sử dụng bánh Mecanum (FMWMR) có thể được ứng dụng làm xe lăn điện, mang lại sự chủ động trong di chuyển cho người cao tuổi. Điều này giúp họ tham gia các hoạt động tự chăm sóc, sinh hoạt giải trí, giao tiếp xã hội và nâng cao giá trị bản thân. Nhờ đó, chất lượng cuộc sống của những người có khả năng đi lại hạn chế, thiếu sức bền hoặc không thể tự đẩy xe lăn được cải thiện rõ rệt. Ngoài ra, FMWMR còn hỗ trợ hiệu quả cho những bệnh nhân có khuyết tật thể chất hoặc tinh thần nặng.

Trong lĩnh vực giáo dục, FMWMR được sử dụng như một nền tảng di động linh hoạt, phục vụ nghiên cứu và thử nghiệm các thuật toán điều hướng robot trong môi trường trong nhà.

1.5. Mục tiêu của Đồ án tốt nghiệp

- Tóm tắt các mô hình toán học của robot di động bốn bánh Mecanum, bao gồm động học và động lực học.
- Đề xuất bộ điều khiển Back-stepping kết hợp điều khiển trượt để bám vị trí và tốc độ của robot.

- Xây dựng hệ thống định vị sử dụng camera gắn trên robot để nhận diện và ước tính vị trí cũng như hướng của các ArUco Marker được bố trí trên bản đồ, từ đó xác định chính xác vị trí và hướng của FMWMR trong hệ tọa độ thế giới thực.
- Đề xuất phương pháp tích hợp dữ liệu từ IMU, encoder và thông tin vị trí, góc hướng thu được từ ArUco Marker, kết hợp xử lý bằng bộ ước lượng EKF hoặc các biến thể nâng cao như AEKF và EKF chi bình phương, nhằm ước tính chính xác và ổn định hơn góc hướng cũng như trạng thái chuyển động của FMWMR.
- Xây dựng mô hình thực tế của FMWMR để đánh giá hiệu quả của phương pháp điều khiển.
- Tối ưu hóa tốc độ xử lý, giảm thiểu sai số bám vị trí và hướng.
- Phân tích kết quả mô phỏng và thực nghiệm

1.6. Cấu trúc tổng quan của hệ thống

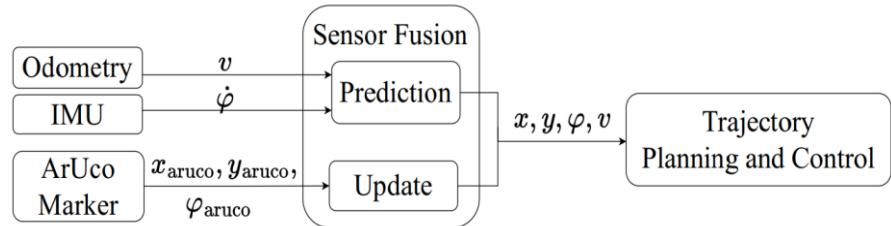
Hệ thống định vị trong nhà được triển khai trên một nền tảng robot di động bánh Mecanum cho phép di chuyển linh hoạt trong không gian hẹp và đa hướng. Robot được trang bị một máy tính công nghiệp (IPC) tích hợp bên trong để đảm nhiệm việc xử lý tập trung toàn bộ các thuật toán điều khiển, định vị và lập kế hoạch chuyển động. Ngoài ra, robot còn sử dụng một camera RGB độ phân giải 1080p để phát hiện các marker ArUco phân bố trong môi trường làm việc, giúp cung cấp thông tin vị trí và hướng nhìn tương đối so với các điểm mốc đã biết. Để hỗ trợ việc xác định hướng quay của robot, cảm biến quán tính BNO055 được tích hợp để đo tốc độ quay quanh trục yaw, cung cấp dữ liệu định hướng trong thời gian thực.

Hệ thống truyền động của robot bao gồm bốn động cơ CHR-GM37-520, mỗi động cơ được gắn encoder để đo vận tốc bánh xe và cung cấp phản hồi vị trí chính xác. Các động cơ này được điều khiển thông qua một bo mạch Arduino ATmega 2560, chịu trách nhiệm thu thập dữ liệu encoder và gửi đến IPC để xử lý cao cấp hơn.

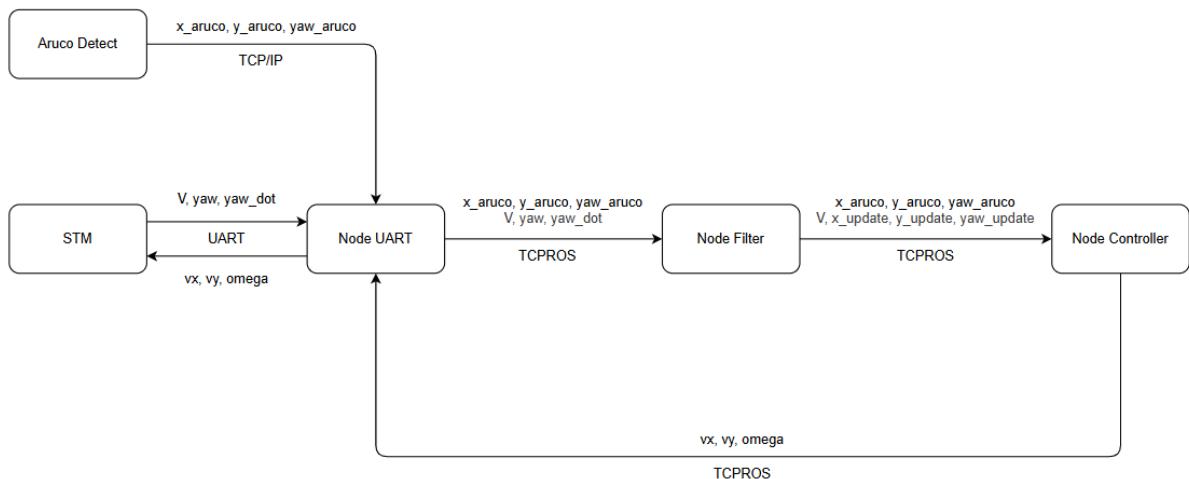
Trong hệ thống này, quá trình định vị dựa vào ba nguồn cảm biến chính:

- Vận tốc bánh xe thu được từ encoder của các động cơ.
- Tốc độ quay quanh trục yaw đo bởi cảm biến IMU BNO055.
- Thông tin vị trí và hướng nhìn thu được từ camera thông qua việc phát hiện các marker ArUco.

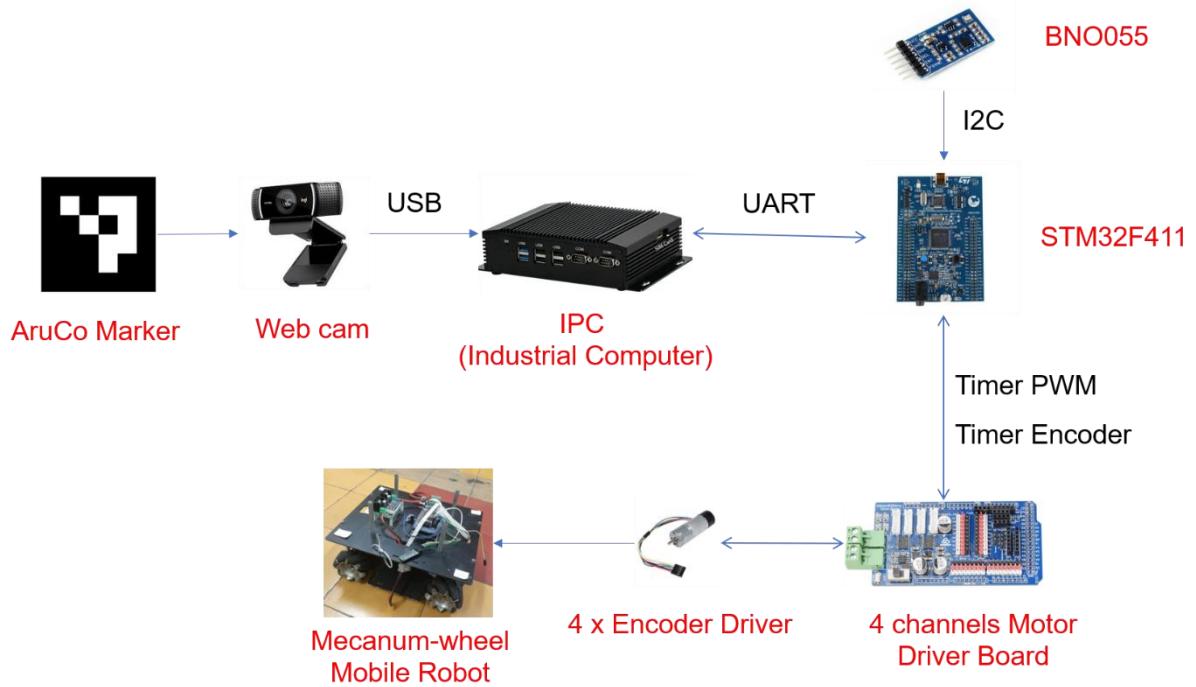
Cụ thể, các giá trị vận tốc bánh xe và tốc độ yaw được sử dụng trong bước dự đoán (prediction step) của mô-đun tích hợp cảm biến (sensor fusion), giúp dự đoán trạng thái chuyển động hiện tại của robot. Sau đó, bước cập nhật (update step) sẽ sử dụng các phép đo vị trí và góc phương vị từ kết quả nhận diện marker để hiệu chỉnh và làm chính xác hơn ước lượng trạng thái. Trạng thái hợp nhất cuối cùng, bao gồm vị trí (x, y), góc phương vị (θ), và vận tốc tuyến tính (v), được truyền đến bộ lập kế hoạch quỹ đạo để xác định đường đi tối ưu và gửi lệnh đến hệ thống điều khiển động cơ, đảm bảo robot di chuyển chính xác đến các vị trí mục tiêu trong không gian trong nhà.



Hình 1.6: Tổng quan sơ đồ tín hiệu của hệ thống



Hình 1.7: Sơ đồ tín hiệu của hệ thống của hệ thống khi thực hiện thực tế



Hình 1.8: Cấu trúc phần cứng của mô hình thực tế

Bảng 1.2: Bảng tổng quan phần cứng của hệ thống

Thành phần		Tên thiết bị
IPC	CPU	Intel® Core i7-10510U CPU
	GPU	Intel® UHD Graphics
	RAM	16GB DDR4-2400
Speed Motor		CHR-GM37-520 (with Encoder)
MCU		STM32F411
IMU		BNO055
Camera		Webcam HD 1080p

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Mô hình động học và động lực học hệ robot di động

2.1.1. Mô hình toán học robot Mecanum

Trong phạm vi của đồ án tốt nghiệp này, đối tượng nghiên cứu là một cấu hình robot bốn bánh Mecanum, trong đó các bánh xe được bố trí đối xứng qua tâm của khung robot. Mô hình động học của hệ thống được minh họa trong Hình 2.1, trong khi mô hình động lực học được thể hiện ở Hình 2.2. Các thông số kỹ thuật đặc trưng của robot được tổng hợp và trình bày trong Bảng 2.1.

Bảng 2.1. Bảng thông số kỹ thuật

Ký hiệu	Ý nghĩa	Độ lớn	Đơn vị
x	Tọa độ toàn cục theo phương x		m
y	Tọa độ toàn cục theo phương y		m
θ	Góc lệch của robot so với tọa độ gốc		rad
$\omega_i (i = 1, \dots, 4)$	Tốc độ góc bánh xe thứ i		rad/s
r	Bán kính bánh xe	0.035	m
m	Tổng khối lượng của robot	6	Kg
L1	Khoảng cách từ trục Byb đến bánh xe	0.11	m
L2	Khoảng cách từ trục Bxb đến bánh xe	0.1	m
g	Gia tốc trọng trường	9.8	m/s ²
I	Mô men quán tính của robot	0.22	Kg.m ²
I _b	Mô men quán tính của bánh xe	0.02328	Kg.m ²

2.1.2. Mô hình động học

Phân tích động học của robot di động bốn bánh Mecanum đóng vai trò quan trọng trong việc mô tả chuyển động của khung xe cũng như xây dựng và thiết kế các thuật toán điều khiển phù hợp. Các phương trình động học thuận và động học nghịch của hệ thống được trình bày lần lượt trong (2.1) và (2.2).

Cụ thể, phương trình (2.1) biểu diễn ma trận động học thuận của robot di động bốn bánh Mecanum (Four Mecanum Wheeled Mobile Robot – FMWMR). Trong phương trình này, vận tốc tịnh tiến và vận tốc quay của robot được xác định từ tốc độ

góc của bốn bánh xe cùng với bán kính bánh. Đồng thời, vận tốc góc của robot còn phụ thuộc vào kích thước hình học của robot, bao gồm chiều dài cơ sở và chiều rộng cơ sở.

Ma trận động học thuận của vận tốc robot trên trực tiếp bộ với vận tốc từng bánh xe:

$$\begin{bmatrix} u \\ v \\ \omega \end{bmatrix} = \frac{r}{4} \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1 & -\frac{1}{L_1+L_2} & \frac{1}{L_1+L_2} & \frac{1}{L_1+L_2} \\ \frac{1}{L_1+L_2} & -\frac{1}{L_1+L_2} & -\frac{1}{L_1+L_2} & \frac{1}{L_1+L_2} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} \quad (2.1)$$

Ma trận động học nghịch của vận tốc robot trên trực tiếp bộ với vận tốc từng bánh xe:

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} -1 & 1 & L_1 + L_2 \\ 1 & 1 & -(L_1 + L_2) \\ -1 & 1 & -(L_1 + L_2) \\ 1 & 1 & L_1 + L_2 \end{bmatrix} \begin{bmatrix} u \\ v \\ \omega \end{bmatrix} \quad (2.2)$$

Xét mô hình robot di động gắn với các hệ trục tọa độ toàn cục x_i/y_i và hệ tọa độ cục bộ robot x_b/y_b như Hình 2.1.

Các thông số của hệ gồm:

- x/y là khoảng dịch chuyển theo trục x_i/y_i trong hệ tọa độ x_i/y_i ;
- θ là góc quay của thân robot trong hệ trục tọa độ toàn cục x_i/y_i ;
- u/v là vận tốc dịch chuyển theo trục x_b/y_b trong hệ tọa độ x_b/y_b ;
- ω là vận tốc góc của robot trong hệ tọa độ x_b/y_b .

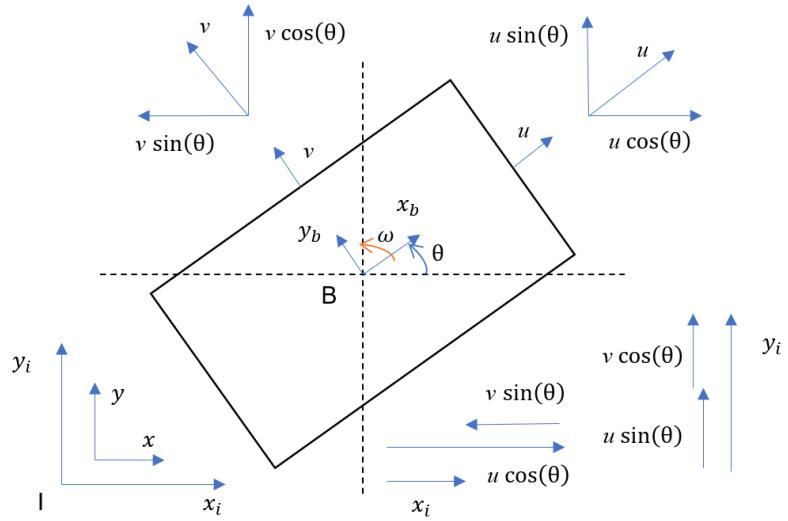
Các phương trình động học của hệ robot di động thể hiện mối liên hệ của vận tốc và góc quay trong hệ trục tọa độ cục bộ robot (B) và hệ trục tọa độ toàn cục (I):

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u \\ v \\ \omega \end{bmatrix}$$

$$\dot{\eta} = J(\theta) \cdot \zeta$$

Trong đó:

$$\eta = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}; \zeta = \begin{bmatrix} u \\ v \\ \omega \end{bmatrix}; J(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



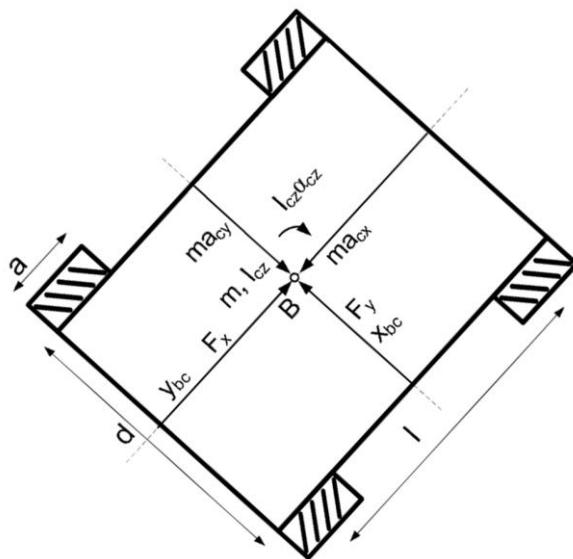
Hình 2.1: Mô hình động học của robot mecanum

2.1.3. Mô hình động lực học

Xét mô hình mô hình động lực học robot di động gắn với các hệ trục tọa độ như hình 2.2. Đặt B là tâm hình học của rô bốt.

Các thông số của hệ gồm:

- m là khối lượng của robot;
- l và d là chiều dài và chiều rộng robot;
- I_{cz} là mô men quán tính robot;
- ω và α_{cz} là vận tốc và gia tốc góc robot;
- a_{cx}/a_{cy} là gia tốc theo trục x/y của robot;
- a là đường kính của bánh xe.



Hình 2.2: Mô hình động lực học robot mecanum

Để xây dựng mô hình động lực học cho robot di động, ta sử dụng phương pháp Lagrange.

$$\frac{d}{dt} \left(\frac{\partial L}{\partial q_i} \right) - \frac{\partial L}{\partial q_i} = F_i$$

Trong đó:

- L là hàm Lagrange, được tính từ hiệu của động năng (KE) và thế năng (PE);
- q_i là các biến trạng thái của robot;
- Động năng KE của robot được tính gồm cả phần tịnh tiến và quay;
- Thế năng PE của robot được giả định là bằng 0 do chuyển động trên mặt phẳng.

$$KE = 0.5m[\dot{x}_c^2 + \dot{y}_c^2] + 0.5I_\alpha\omega^2; PE = 0$$

Sử dụng quan hệ giữa vận tốc trong hệ tọa độ robot và hệ tọa độ toàn cục:

$$\dot{x}_c = u - y_{bc}\omega; \dot{y}_c = v + x_{bc}\omega$$

Thay các biểu thức vận tốc vào công thức động năng:

$$L = KE - PE = 0.5m(u^2 - 2u\omega y_{bc} + v^2 + 2v\omega x_{bc} + \omega^2[x_{bc}^2 + y_{bc}^2]) + 0.5I_\alpha\omega^2$$

Sử dụng công thức Lagrange với các biến u, v, ω ta tính được các thành phần của véc-tơ lực hay mô-men tác động:

$$F_x = \frac{d}{dt} \frac{\partial L}{\partial u} = m(\dot{u} - v\omega - x_{bc}\omega^2 - y_{bc}\dot{\omega})$$

$$F_y = \frac{d}{dt} \frac{\partial L}{\partial v} = m(\dot{v} + u\omega - y_{bc}\omega^2 + x_{bc}\dot{\omega})$$

$$M_z = \frac{d}{dt} \frac{\partial L}{\partial \omega} = I_\alpha\dot{\omega} + m(x_{bc}\dot{v} - y_{bc}\dot{u}) + m\dot{\omega}(x_{bc}^2 + y_{bc}^2)$$

Phương trình động lực học tổng quát của robot sẽ là:

$$D\ddot{\zeta} + n(\zeta) = \tau$$

Trong đó:

$$D = \begin{bmatrix} m & 0 & -my_{bc} \\ 0 & m & mx_{bc} \\ -my_{bc} & mx_{bc} & I_\alpha + m(x_{bc}^2 + y_{bc}^2) \end{bmatrix};$$

$$n(\zeta) = \begin{bmatrix} -m(v + x_{bc}\omega) \\ m(u - y_{bc}\omega) \\ m\omega(x_{bc}u + y_{bc}v) \end{bmatrix}; \tau = \begin{bmatrix} F_x \\ F_y \\ M_z \end{bmatrix} [4]$$

2.2 Điều khiển backstepping

Bộ điều khiển Backstepping là một phương pháp điều khiển phi tuyến hiện đại, thường được áp dụng cho các hệ thống có tính phi tuyến và chịu ảnh hưởng của nhiều

yếu tố động lực học phức tạp. Ưu điểm nổi bật của phương pháp này là xây dựng bộ điều khiển dựa trên tiếp cận Lyapunov, giúp đảm bảo tính ổn định toàn cục của hệ thống. Nguyên lý cơ bản của Backstepping là thiết kế điều khiển theo từng bước (step-by-step), trong đó mỗi bước được xây dựng thông qua việc xác định các biến ảo (virtual control) để ổn định dần các trạng thái trung gian trước khi đạt đến trạng thái mong muốn cuối cùng.

Trong trường hợp robot di động bốn bánh Mecanum, việc áp dụng bộ điều khiển Backstepping cho phép xử lý đồng thời các quan hệ động học và động lực học phức tạp của hệ thống, từ đó nâng cao khả năng bám quỹ đạo và giảm thiểu sai số trong quá trình điều khiển. Cách tiếp cận này đặc biệt phù hợp trong bối cảnh robot hoạt động trong môi trường thực tế có nhiều nhiễu loạn và bất định.

2.2.1. Phương trình động học và động lực học cho bộ điều khiển

Hệ phương trình của robot được mô tả như sau:

$$\begin{cases} \dot{\eta} = J(\eta) \cdot \zeta \\ \dot{\zeta} = D^{-1}[\tau - n(\zeta)] \end{cases}$$

Xét một trường hợp cụ thể: Hệ có điểm cân bằng $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

$$\begin{cases} \dot{x} = f_1(x_1) + g_1(x_1)x_2 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)u \end{cases}$$

Trong đó:

- $x_1 = \eta, x_2 = \zeta, u = \tau$
- $f_1(x_1) = 0, g_1(x_1) = J(\eta)$
- $f_2(x_1, x_2) = D^{-1}[-n(\zeta)], g_2(x_1, x_2) = D^{-1}$
- $J(\eta)$: là ma trận Jacobian

$$J(\eta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Giá trị tham chiếu $x_{1d} = \eta_d$, đạo hàm bậc 1 và 2 tương ứng là $\dot{x}_{1d} = \dot{\eta}_d, \ddot{x}_{1d} = \ddot{\eta}_d$. Sai số điều khiển được định nghĩa $e_1 = x_{1d} - x_1$. Mục tiêu điều khiển là $e_1 \rightarrow 0$ và $\dot{e}_1 \rightarrow 0$.

2.2.2. Thiết kế bộ điều khiển backstepping

Tốc độ thay đổi của sai số giữa tín hiệu mong muốn và tín hiệu thực tế là:

$$\dot{e}_1 = \dot{x}_{1d} - \dot{x}_1 = \dot{x}_{1d} - g_1(x_1)x_2.$$

Nhận thấy $e_1 \rightarrow 0$ chọn hàm Lyapunov có dạng:

$$V_1 = 0.5e_1^2.$$

Đạo hàm V_1 ta được: $\dot{V}_1 = e_1 \dot{e}_1 = e_1(\dot{x}_{1d} - g_1(x_1)x_2)$.

Ta nhận thấy $\dot{V}_1 = -k_1 e_1^2 < 0$, ($k_1 > 0$) nếu ta chọn:

$$\dot{x}_{1d} - g_1(x_1)x_2 = -k_1 e_1.$$

Do đã chọn $\dot{x}_{1d} - g_1(x_1)x_2 = -k_1 e_1$, ta chọn tín hiệu vào bộ điều khiển:

$$x_{2d} = g_1^{-1}(x_1)(\dot{x}_{1d} + k_1 e_1).$$

Khi $x_2 \rightarrow x_{2d}$ ta được sai lệch giữa tín hiệu đặt và tín hiệu phản hồi $e_2 = x_{2d} - x_2$, sau đó ta lấy đạo hàm tín hiệu này ta được:

$$\dot{e}_2 = \dot{x}_{2d} - \dot{x}_2 = \dot{x}_{2d} - f_2(x, t) - g_2(x, t)u.$$

Chọn hàm Lyapunov có dạng:

$$V_2 = V_1 + 0.5e_2^2 = 0.5(e_1^2 + e_2^2).$$

Đạo hàm V_2 ta được:

$$\dot{V}_2 = -k_1 e_1^2 + e_2 \cdot (g_1^T(x_1)e_1 + \dot{x}_{2d} - f_2(x, t) - g_2(x, t)u).$$

Nhận thấy $\dot{V}_2 = -k_1 e_1^2 - k_2 e_2^2 < 0$, ($k_2 > 0$) nếu ta chọn:

$$g_1^T(x_1)e_1 + \dot{x}_{2d} - f_2(x, t) - g_2(x, t)u = -k_2 e_2$$

Từ đó ta chọn bộ điều khiển có dạng:

$$u = g_2^{-1}(x, t)(k_2 e_2 + \dot{x}_{2d} + g_1^T(x_1)e_1 - f_2(x, t))$$

Từ phương trình trên kết hợp với $\dot{x}_{2d} = g_1^{-1}(x_1)(\ddot{x}_{1d} + k_1 \dot{e}_1)$, ta rút ra luật điều khiển có dạng sau:

$$u = g_2^{-1}(x, t)(k_2 e_2 + g_1^{-1}(x_1)(\ddot{x}_{1d} + k_1 \dot{e}_1) + g_1^T(x_1)e_1 - f_2(x, t)) [4]$$

Phương trình của luật điều khiển thỏa mãn điều kiện ổn định Lyapunov như sau:

$$u = D[k_2 e_2 + J^{-1}(\eta)(\dot{\eta}_d + k_1 \dot{e}_1) + J^T(\eta) \cdot e_1] + n(\zeta)$$

2.3. Điều khiển trượt

Bên cạnh phương pháp Backstepping cho vòng điều khiển ngoài, nhằm tạo ra tín hiệu vận tốc bánh xe tham chiếu $\omega_{i,d}$, một phương pháp điều khiển phi tuyến phổ biến khác được áp dụng cho vòng điều khiển trong là điều khiển trượt (Sliding Mode Control – SMC). Phương pháp này có ưu điểm nổi bật ở khả năng đảm bảo tính bền vững của hệ thống ngay cả trong điều kiện có nhiễu và bất định tham số. Ý tưởng chính của SMC là thiết kế một mặt trượt (sliding surface) trong không gian trạng thái, sao cho khi trạng

thái hệ thống tiến tới và duy trì trên mặt trượt này, hệ luôn ổn định và đáp ứng mong muốn.

2.3.1. Mô hình động lực học bánh xe

Đối với mỗi bánh xe Mecanum, mô hình động lực học có thể được biểu diễn dưới dạng:

$$J_\omega \dot{\omega}_i + D_\omega \omega_i = \tau_i$$

Trong đó:

- ω_i là vận tốc góc của bánh xe thứ i
- J_ω là mô men quán tính của bánh xe
- D_ω là hệ số ma sát nhót
- τ_i là mô men điều khiển từ động cơ

Nhiệm vụ của bộ điều khiển vòng trong là đảm bảo vận tốc thực tế của bánh xe ω_i bám theo vận tốc tham chiếu $\omega_{i,d}$ được sinh ra từ vòng Backstepping.

2.3.2. Sai số và mặt trượt

Sai số vận tốc được định nghĩa:

$$e_i = \omega_{i,d} - \omega_i$$

Mặt trượt được thiết kế như sau:

$$s_i(t) = \dot{e}_i + \lambda e_i$$

Trong đó $\lambda > 0$ là hằng số điều chỉnh tốc độ hội tụ. Khi $s_i(t) \rightarrow 0$, cả sai số và đạo hàm của sai số tiến về 0, tức là bánh xe bám theo vận tốc mong muốn.

2.3.3. Luật điều khiển Sliding Mode

Để đảm bảo điều kiện trượt, luật điều khiển được thiết kế sao cho thỏa mãn bất đẳng thức:

$$s_i \dot{s}_i \leq -\eta |s_i|, \eta > 0$$

Theo đó, tín hiệu điều khiển mô men động cơ có dạng:

$$\tau_i = J_\omega (\dot{\omega}_{i,d} - \lambda e_i) + D_\omega \omega_i + k sgn(s_i)$$

Trong đó:

- $J_\omega (\dot{\omega}_{i,d} - \lambda e_i)$ đảm bảo bám theo quỹ đạo tham chiếu.
- $D_\omega \omega_i$ bù trừ lực cản.
- $k sgn(s_i)$ đảm bảo bền vững và khả năng chống nhiễu.

2.4. AruCo Marker

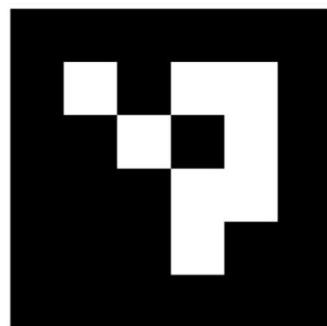
2.4.1. Hệ thống định vị bằng Camera

Hệ thống định vị bằng camera sử dụng một camera gắn trên xe nhận diện các ArUco marker trên bản đồ. Bằng cách phân tích vị trí của marker, vị trí và hướng của robot trong hệ tọa độ toàn cầu có thể được xác định. Việc ước lượng góc bước dựa trên camera này chính xác và không bị trôi lệch, vì nó cung cấp một tham chiếu tuyệt đối đến khung toàn cầu. Tuy nhiên, nó dễ bị che khuất, có góc nhìn hạn chế và có thể gây ra độ trễ trong xử lý hình ảnh.

Đầu tiên, để việc đọc ảnh và tìm ra pose của robot chính xác, chúng ta cần calibration camera, từ việc calibration camera, ta sẽ có được hai thông số quan trọng là: Intrinsic matrix of camera và Distortion coefficients.

2.4.2. Tổng quan về phương pháp Aruco marker

Một ArUco Marker là một điểm đánh dấu hình vuông tổng hợp bao gồm một đường viền đen rộng và một ma trận nhị phân bên trong nền trắng để xác định số nhận dạng (ID) của nó. Đường viền đen giúp cho việc phát hiện hình ảnh nhanh chóng, và việc mã hóa nhị phân cho phép xác định vị trí và áp dụng các kỹ thuật phát hiện và hiệu chỉnh.



Hình 2.3: Aruco Marker kích thước 4x4

Kích thước của marker quyết định kích thước ma trận bên trong. Ví dụ, một marker 4x4 được xây dựng từ một ma trận 16-bit. Mã này có thể xoay theo nhiều hướng khác nhau trong môi trường. Để giải mã dữ liệu được truyền một cách chính xác, cần xác định góc xoay ban đầu, khiến mỗi góc rõ ràng được xác định rõ ràng. Mã nhị phân giải quyết vấn đề này.

Aruco Marker là một công nghệ phổ biến trong thị giác máy tính và Thực tế Tăng cường (AR). Được phát triển bởi thư viện OpenCV, Aruco Marker là một loại mã

vạch có thể nhận dạng được được sử dụng để theo dõi và xác định vị trí trong các ứng dụng AR và trí tuệ nhân tạo.

Mỗi ArUco Marker là một hình vuông đen trắng được chia thành một lưới các ô nhỏ, với mỗi ô có hai trạng thái: đen hoặc trắng. Qua cấu trúc đặc biệt này, mỗi marker mã hóa một giá trị nhận dạng duy nhất. Khi một máy ảnh chụp một hình ảnh chứa một ArUco Marker, thuật toán nhận dạng có thể xác định vị trí và góc quay của marker trong không gian 3D.

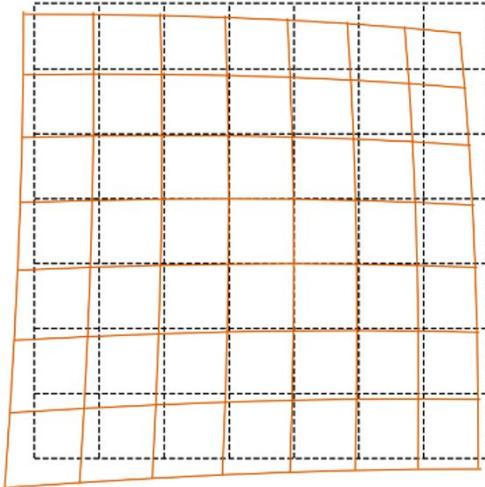
Các ứng dụng của ArUco Marker rất đa dạng. Trong thực tế tăng cường, nó có thể được sử dụng để tích hợp các đối tượng 3D vào các hình ảnh thực tế, tạo ra các trải nghiệm ảo tương tác trên các thiết bị di động và các nền tảng AR khác. Ngoài ra, ArUco Marker được sử dụng trong các hệ thống robot để theo dõi vị trí và hướng của robot trong một môi trường 3D, hỗ trợ trong điều hướng và định vị.

Với khả năng nhận dạng chính xác và hiệu suất tính toán nhanh, ArUco Marker đã trở thành một công cụ mạnh mẽ trong lĩnh vực thị giác máy tính và AR. Việc sử dụng ArUco Marker mở ra nhiều cơ hội sáng tạo cho việc phát triển các ứng dụng thực tế tăng cường, định vị robot và các lĩnh vực khác sử dụng công nghệ thị giác máy tính.

2.4.3. Camera calibration

Một số loại ống kính máy ảnh có thể gây ra các dạng biến dạng đáng kể trong ảnh chụp, trong đó phổ biến nhất là biến dạng xuyên tâm (radial distortion) và biến dạng tiếp tuyến (tangential distortion).

Biến dạng xuyên tâm thường thể hiện dưới dạng cong vênh của các đường thẳng, và mức độ cong này tăng dần khi điểm ảnh nằm xa khỏi tâm ảnh. Ví dụ, ở Hình 2.4 minh họa biến của một bàn cờ, mặc dù các cạnh được đánh dấu bằng các đường thẳng màu đỏ, song ta có thể quan sát thấy biến của bàn cờ không trùng khớp với các đường này mà bị cong lồi ra, thể hiện rõ sự méo dạng xuyên tâm.



Hình 2.4: Camera Calibration Flowchart

Hiện tượng này có thể được mô tả toán học như sau:

$$x_{distorted} = x(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

$$y_{distorted} = y(1 + k_1r^2 + k_2r^4 + k_3r^6)$$

Trong khi đó, biến dạng tiếp tuyến xuất hiện khi ống kính không được lắp đặt hoặc căn chỉnh hoàn toàn song song với mặt phẳng ảnh. Điều này dẫn đến việc một số vùng trong ảnh xuất hiện lệch vị trí, khiến chúng có cảm giác gần hơn hoặc xa hơn so với thực tế. Biến dạng tiếp tuyến được mô tả bởi:

$$x_{distorted} = x + (2p_1xy + p_2(r^2 + 2x^2))$$

$$y_{distorted} = y + (p_1(r^2 + 2y^2) + 2p_2xy)$$

Tóm lại, quá trình hiệu chỉnh biến dạng đòi hỏi xác định năm hệ số biến dạng bao gồm:

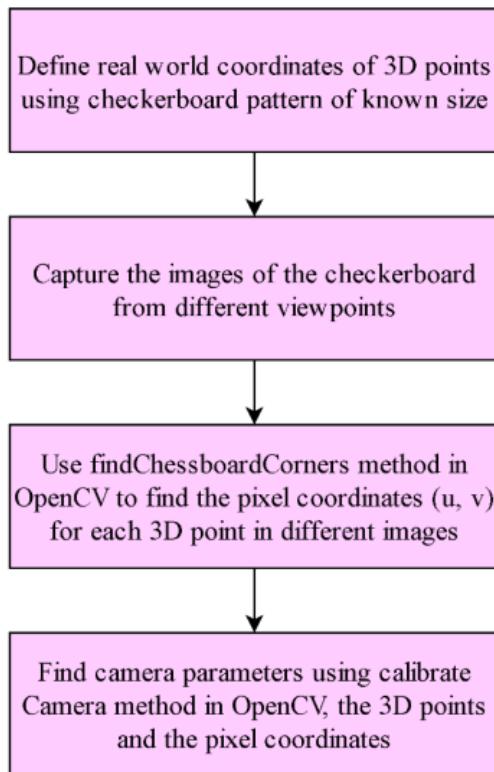
$$distortion coefficients = [k_1, k_2, p_1, p_2, k_3]$$

Việc hiệu chuẩn camera là một bước quan trọng trong thị giác máy tính và xử lý ảnh, đòi hỏi xác định các tham số nội tại và ngoại tại của một máy ảnh. Các tham số nội tại bao gồm độ dài tiêu cự, điểm chính, và các hệ số biến dạng, trong khi các tham số ngoại tại bao gồm vị trí và hướng của máy ảnh trong không gian ba chiều.

Hiệu chuẩn camera là cần thiết để hiệu chỉnh các biến dạng và không chính xác mà có thể xuất hiện trong bất kỳ hệ thống camera nào. Những biến dạng này có thể gây ra sai số trong các đo lường và việc phát hiện đối tượng, dẫn đến kết quả không chính xác. Hiệu chuẩn giúp đảm bảo rằng các đo lường của camera là chính xác và đáng tin cậy.

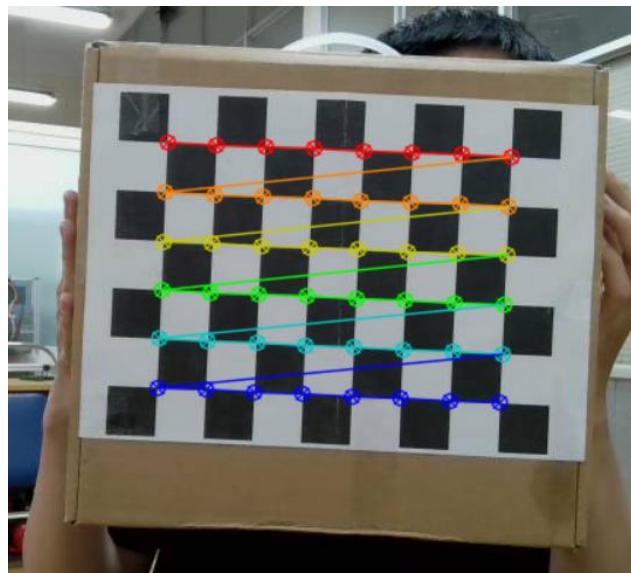
Trong OpenCV, việc hiệu chuẩn camera thường được thực hiện bằng một kỹ thuật gọi là hiệu chuẩn bàn cờ cờ vua. Điều này liên quan đến việc chụp nhiều hình ảnh của một mẫu bàn cờ cờ vua từ các góc độ và hướng khác nhau và sử dụng chúng để ước lượng các thông số nội tại và ngoại tại của camera.

Camera Calibration Flowchart



Hình 2.5: Camera Calibration Flowchart

Kết quả của việc hiệu chuẩn là ta thu được ma trận camera: một ma trận 3×3 với các thành phần là khoảng cách tiêu điểm và tọa độ trung tâm của camera (các tham số nội tại), và các hệ số méo (distortion coefficients): một vector gồm 5 hoặc nhiều hơn các thành phần mô hình hóa sự méo của camera của bạn. Điều này là một bước quan trọng để thực hiện ước lượng tư thế của camera, như ta sẽ đề cập phía dưới, ta sẽ cần biết các tham số hiệu chuẩn của camera. Đó chính là ma trận camera và các hệ số méo.



Hình 2.6: Hình ảnh calib checker board

```
K: !!opencv-matrix
rows: 3
cols: 3
dt: d
data: [ 591.55070674671776, 0., 325.3678451553609, 0.,
        592.88278307839482, 257.09178056444006, 0., 0., 1. ]
D: !!opencv-matrix
rows: 1
cols: 5
dt: d
data: [ 0.24926886187398647, -1.5914981566495681,
        -0.001376346729257804, 0.0042357208958515302, 2.6839020936373701 ]
```

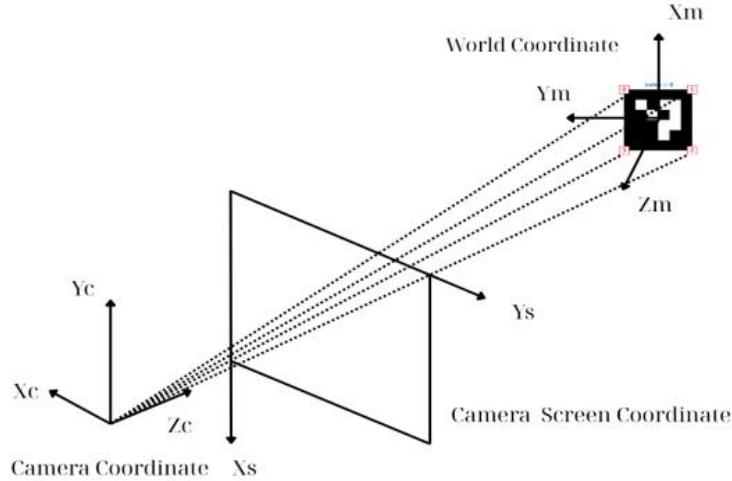
Hình 2.7: Kết quả Camera calibration

Kết quả hiệu chuẩn trên tạm chấp nhận được khi camera hoạt động ở chế độ video tiêu chuẩn 640x480. Trong cấu hình này, các tọa độ giao điểm của trực quang học của camera trên mặt phẳng sàn C (cx, cy) trùng với trung tâm của mặt phẳng hình ảnh. Tuy nhiên, các hệ số méo của hình ảnh không hoàn hảo, và có một hệ số méo có độ lớn lớn hơn 1, chẳng hạn như 2.683902, có thể dẫn đến sự không chính xác trong việc xác định vị trí và hướng của camera. Mặc dù vậy, đây là các kết quả hiệu chuẩn camera tốt nhất đạt được trong dự án này.

2.4.4. Đo lường vị trí bằng Aruco Marker

Mỗi quan hệ hình học giữa một điểm ba chiều (3D) trong không gian và điểm ảnh hai chiều (2D) tương ứng trên mặt phẳng ảnh của camera được mô tả thông qua mô hình camera lỗ kim (pinhole camera model). Đây là một mô hình toán học được đơn giản hóa nhưng mang tính tổng quát, cho phép biểu diễn quá trình chiếu từ không gian

3D xuống mặt phẳng 2D. Nhờ đặc tính này, mô hình camera lỗ kim được ứng dụng rộng rãi trong lĩnh vực thị giác máy tính, đặc biệt trong các bài toán hiệu chỉnh nội tại và ngoại tại của camera, tái tạo hình học 3D, cũng như định vị và dẫn đường dựa trên thị giác.



Hình 2.8: Mối quan hệ giữa tọa độ tâm ArUco và tọa độ máy ảnh

Chúng ta có $\{X_c, Y_c, Z_c\}$ lần lượt là giá trị các góc marker corners sau khi được nhận diện trong hệ trục tọa độ của camera. $\{X_s, Y_s\}$ là tọa độ các corners trong hệ trục tọa độ pixel của mặt phẳng ảnh. Dưới đây là công thức biến đổi từ hệ tọa độ 4 marker corners (x_{ci}, y_{ci}, z_{ci}) trong camera frame, và tọa độ điểm pixel 2D (x_{si}, y_{si}) trong camera-screen frame.

$$\begin{bmatrix} x_{si} \\ y_{si} \\ 1 \end{bmatrix} = \begin{pmatrix} f_x & 0 & x_{s0} \\ 0 & f_y & y_{s0} \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x_{ci} \\ y_{ci} \\ z_{ci} \\ 1 \end{bmatrix}$$

Trong đó, $i = \{1,2,3,4\}$ lần lượt là 4 của marker corners points, (x_{s0}, y_{s0}) là tọa độ tâm ảnh trong hệ tọa độ ảnh và f_x, f_y tiêu cự của trục X_s và Y_s . 4 tham số này là 4 tham số nằm trong intrinsics matrix của camera. Ngoài ra, r_{ij} và t_i lần lượt là các tham số của 2 ma trận là: rotation matrix và translation matrix. Việc xác định ma trận ngoại tại R_c và T_c về cơ bản là một bài toán ước lượng tư thế camera từ các điểm tương ứng (Perspective-n-Point - PnP). Bài toán này thường được giải bằng hàm solvePnP của thư viện OpenCV, hoặc là bằng nhiều phương pháp giải khác nhau. Sau khi tính được ma trận R_c và T_c , góc tương đối φ_1 và khoảng cách d từ AruCo so với robot. Dựa vào công thức sau:

$$\varphi_2 = \arctan\left(\frac{-r_{31}}{\sqrt{r_{32}^2 + r_{33}^2}}\right)$$

$$d = \sqrt{(t_1 + l_{lat})^2 + (t_3 + l_{lon})^2}$$

Khi đó, l_{lat} và l_{lon} biểu thị khoảng cách theo chiều ngang và chiều dọc (đơn vị mét) từ vị trí camera đến tâm trực thực của xe. Mỗi Aruco sẽ được pre-define trước một giá trị trong world frame bao gồm (x_0, y_0, φ_0) . Vị trí của robot trong global frame map được tính bằng khoảng cách d , góc tương đối φ_1 và vị trí tuyệt đối của mỗi marker Aruco. Tiếp theo, ta sẽ tính góc của đầu xe so trong hệ tọa độ global frame là φ_{robot} .

$$\varphi_{robot} = \varphi_1 + \varphi_0$$

Trong lúc setup, vị trí của camera là cố định và thẳng hàng so với tâm của robot, do đó loát sẽ bằng 0. Góc φ_2 và φ_3 sẽ được tính như sau:

$$\varphi_2 = \arctan\left(\frac{t_1}{t_3 + l_{lon}}\right)$$

$$\varphi_3 = \varphi_{robot} - \varphi_2$$

Cuối cùng, ta sẽ tính được vị trí của xe trong hệ trục tọa độ global frame, được đo bằng mét.

$$x_{robot} = x_0 - d \cos(\varphi_3)$$

$$y_{robot} = y_0 - d \sin(\varphi_3)$$

Vì các frame khi được đọc về sẽ bị nhiễu bởi ánh sáng, màu sắc và các yếu tố ngoại cảnh, do đó để giá trị pose được tính toán một cách ổn định và chính xác, chúng tôi đề xuất sử dụng thêm các bộ ước lượng như Low Pass Filter với hệ số alpha là 0.5 cho cả vị trí x , y và góc φ

Trong các tình huống mà robot không thể phát hiện được các marker Aruco, hệ thống sẽ chuyển sang sử dụng phương pháp khác để ước lượng tư thế (pose) hiện tại của robot. Phương pháp này dựa trên dữ liệu cảm biến từ IMU (bộ đo quán tính) và các encoder gắn ở bánh xe để tính toán vị trí và hướng di chuyển của robot. Cụ thể, vận tốc tuyến tính (linear velocity) của robot được suy ra từ các encoder, vốn ghi lại số vòng quay của bánh xe, qua đó cho biết quãng đường mà robot đã di chuyển trong một khoảng thời gian ngắn. Trong khi đó, vận tốc góc (angular velocity) – tức là tốc độ quay của

robot quanh trục thẳng đứng – được đo trực tiếp từ cảm biến IMU. Dựa trên hai thông tin này, hệ thống sẽ cập nhật tư thế của robot tại thời điểm $t + \Delta t$ bằng cách tính toán sự thay đổi vị trí và góc quay so với thời điểm trước đó.

$$x_{t+\Delta t} = x_t + (v_x \cdot \cos(\varphi t) - v_y \cdot \sin(\varphi t)) \cdot \Delta t$$

$$y_{t+\Delta t} = y_t + (v_x \cdot \sin(\varphi t) + v_y \cdot \cos(\varphi t)) \cdot \Delta t$$

$$\varphi_{t+\Delta t} = \varphi_t + \omega_t \cdot \Delta t$$

Phương pháp này tuy không phụ thuộc vào tín hiệu từ camera hoặc các nguồn dữ liệu bên ngoài, nhưng lại có nhược điểm là sai số tích lũy sẽ tăng dần theo thời gian nếu không được hiệu chỉnh. Vì vậy, phương pháp này thường chỉ được sử dụng như một giải pháp tạm thời hoặc bổ trợ, trong trường hợp hệ thống nhận diện ArUco marker bị gián đoạn.

2.5. Sử dụng các bộ ước lượng Kalman

2.5.1. EKF

Bộ ước lượng Kalman mở rộng (EKF) là một thuật toán ước lượng trạng thái được sử dụng rộng rãi trong các hệ thống phi tuyến, đặc biệt trong các bài toán định vị robot, theo dõi đối tượng, và dẫn đường. Mục tiêu của EKF là ước lượng trạng thái hệ thống từ các đầu vào điều khiển và đo lường bị nhiễu, sao cho sai số là nhỏ nhất.

2.5.1.1. Bước dự đoán

Đây là bước đầu tiên trong chu trình EKF tại mỗi thời điểm t . EKF sử dụng mô hình động học để dự đoán trạng thái mới của hệ thống và dự đoán ma trận hiệp phương sai tương ứng.

$$x_{t|t-1} = f(x_{t-1}, u_t)$$

$$P_{t|t-1} = F_t P_{t-1} F_t^T + Q$$

Trong đó:

- $x_{t|t-1}$ là trạng thái ước lượng sơ bộ tại thời điểm t , chưa được cập nhật bởi cảm biến;
- $f(x_{t-1}, u_t)$ là mô hình phi tuyến mô tả hệ thống, mô hình được sử dụng là a kinematic single-track model [1];

- Q_t là ma trận nhiễu quá trình;
- F_t là biến đổi Jacobian $f(x_{t-1}, u_t)$, tuyênh hóa thông qua sự mở rộng Taylor bậc nhất xung quanh trạng thái x_{t-1} .

Dưới đây là mô hình a kinematic single-track model:

$$f(x_{t-1}, u_t) = \begin{bmatrix} x_{t-1} + \Delta t(v_{t-1}\cos(\varphi_{t-1})) \\ y_{t-1} + \Delta t(v_{t-1}\sin(\varphi_{t-1})) \\ \varphi_{t-1} + \Delta t\omega_t \\ v_{t-1} + \Delta v \end{bmatrix}$$

Trong đó, Δt là bước thời gian, tốc độ yaw ω_t [deg/sec] được đo lường bởi IMU, được lấy từ đầu vào điều khiển của hệ thống $u_t = \omega_t$. Tốc độ $v_{t-1} + \Delta v$ [m/sec] thu được trực tiếp từ wheel encoders.

2.5.1.2. Bước cập nhật

Tại thời điểm t , hệ thống nhận được dữ liệu cảm biến (camera) $z_t = [x_{aruco}, y_{aruco}, \varphi_{aruco}]^T$, ta sử dụng dữ liệu này trong bước cập nhật trạng thái ước lượng x_t và ma trận hiệp phương sai P_t . Trước tiên ta tính sai số đo lường $d_t = z_t - H_t x_{t|t-1}$, trong đó $H_t = [I_{3x3} \ O_{3x1}]$ nhằm lấy 3 thông số $[x, y, \varphi]^T$ của $x_{t|t-1}$. Innovation chính là độ lệch giữa đo lường thực tế và giá trị dự đoán của đo lường. nếu giá trị d_t nhỏ, thì đo lường gần khớp với dự đoán, mô hình và cảm biến đều hoạt động ổn. Nếu d_t lớn, có thể bị nhiễu, outlier hoặc mô hình chưa chính xác.

Ta không thể đánh giá innovation lớn/nhỏ có tốt hay không nếu không biết mức "bình thường" là bao nhiêu. Do đó chúng ta cần tính thêm ma trận hiệp phương sai của sai số đo (Measurement innovation uncertainty),

$$S_t = H_t P_{t|t-1} H_t^T + R_t$$

Ma trận S_t mô tả độ bất định (uncertainty) của sai số đo lường d_t . Nó bao gồm Sự không chắc chắn đến từ chính trạng thái dự đoán P và sự không chắc chắn đến từ cảm biến R. Hay nói cách khác Innovation d_t là một giá trị đo sai số và S_t là độ tin cậy của sai số đó.

Kalman Gain $K_{t-1} = P_{t-1}H_t^TS_t^{-1}$ là trọng số quyết định mức độ tin tưởng vào đo lường mới so với dự đoán từ mô hình. Nếu cảm biến đáng tin, K_t lớn, cập nhật mạnh và ngược lại. Kalman Gain chính là chìa khóa của hiệu suất bộ ước lượng Kalman.

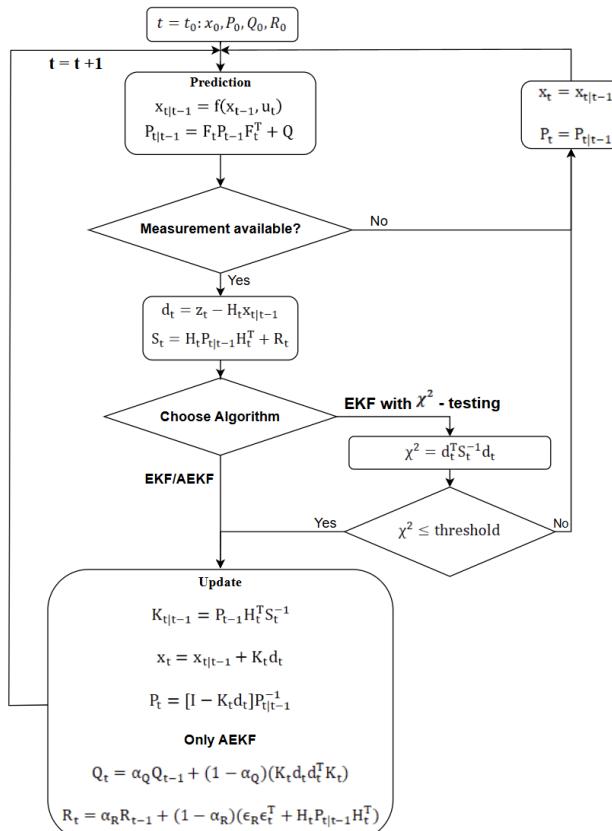
Cuối cùng, ta cập nhật trạng thái ước lượng và ma trận hiệp phương sai:

$$x_t = x_{t|t-1} + K_t d_t$$

$$P_t = [I - K_t d_t] P_{t|t-1}^{-1}$$

Trạng thái sau cập nhật luôn nằm giữa dự đoán và đo lường, theo tỷ lệ niềm tin vào mỗi bên. Sau khi cập nhật ma trận hiệp phương sai, độ bất định của trạng thái giảm xuống. Đây là lý do Kalman Filter hội tụ dần: mỗi lần cập nhật, nó giảm độ không chắc chắn.

Trong luận văn này, 2 phương pháp ước lượng cải tiến từ phương pháp EKF được đề xuất để tăng khả năng định vị và bám quỹ đạo cho hệ thống. Phương pháp AEKF (Adaptive extended Kalman Filter) và Extended Kalman Filter kết hợp kiểm định χ^2 . Sơ đồ khái 2 phương pháp như hình 2.9:



Hình 2.9: Sơ đồ khái 2 phương pháp AEKF và EKF kết hợp kiểm định χ^2

2.5.2. AEKF

Bên cạnh EKF, chúng tôi còn thử nghiệm bộ ước lượng AEKF [2] với các ma trận nhiễu quá trình và ma trận nhiễu phép đo được cập nhật một cách thích ứng thay vì được thiết lập cố định như trong EKF. Theo phương pháp trong [3], chúng tôi tính sai số đo lường giữa \mathbf{z}_t và trạng thái đã cập nhật tại thời điểm t gọi là residual. Đây là sai số giữa đo lường thực tế và giá trị đo được suy ra từ trạng thái đã cập nhật.

$$\varepsilon_t = \mathbf{z}_t - H_t \mathbf{x}_t$$

Do đó ma trận nhiễu quá trình và phép đo được tính toán thích ứng trong phần *cập nhật* như:

$$Q_t = a_Q Q_{t-1} + (1 - a_Q)(K_t d_t d_t^T K_t) \text{ (process noise matrices)}$$

$$R_t = a_R R_{t-1} + (1 - a_R)(\varepsilon_t \varepsilon_t^T + H_t P_{t|t-1} H_t^T) \text{ (measurement noise matrices)}$$

Trong đó, K_t là Kalman Gain, $P_{t|t-1}$ là ma trận hiệp phương sai tương ứng với $\mathbf{x}_{t|t-1}$, và $0 < \alpha_{Q,R} \leq 1$ là hệ số quên (forgetting factors). Nếu $\alpha_{Q,R} = 1$ thì hệ thống không thích ứng (non-adaptive), hoạt động giống như EKF.

2.5.3. EKF kết hợp với kiểm định chi bình phương

Trong Extended Kalman Filter with χ^2 -testing [4], quá trình kiểm tra Chi bình phương (χ^2) được sử dụng để phát hiện lỗi trong các đo lường cảm biến. Bộ ước lượng tính giá trị χ^2 từ sai số innovation giữa đo lường thực tế và giá trị dự đoán, đồng thời sử dụng ma trận hiệp phương sai của sai số để chuẩn hóa giá trị này. Việc tính toán theo công thức:

$$\chi^2 = d_t^T S_t^{-1} d_t$$

$$\chi^2 \leq threshold$$

Nếu giá trị χ^2 lớn hơn một ngưỡng $threshold$ định sẵn (xác định bằng thực nghiệm), đo lường đó được coi là outlier và bị loại bỏ khỏi quá trình correction của EKF. Khi đó, trạng thái hệ thống không bị cập nhật theo thông tin sai lệch. Ngược lại, nếu giá trị χ^2 nhỏ hơn hoặc bằng $threshold$, EKF sẽ thực hiện correction bình thường. Phương pháp này giúp bảo vệ hệ thống định vị khỏi các sai số lớn đột ngột, duy trì độ ổn định và độ chính xác cao trong các điều kiện hoạt động thực tế đầy nhiễu.

2.6. Mô hình hóa và ước lượng trạng thái từ dữ liệu cảm biến

2.6.1. Tính toán định hướng từ IMU

Cảm biến quán tính (Inertial Measurement Unit – IMU) cung cấp thông tin về gia tốc tuyến tính và tốc độ quay góc, từ đó có thể tính toán góc phương vị (yaw) và định hướng của robot. Trong hệ thống này, cảm biến BNO055 được sử dụng với khả năng xuất trực tiếp giá trị Euler angles hoặc quaternion, giúp đơn giản hóa việc xử lý.

Khi sử dụng tốc độ góc quanh trục z (yaw rate) thu được từ con quay hồi chuyển (gyroscope), góc phương vị $\theta(t)$ tại thời điểm t có thể được tính xấp xỉ bằng phương pháp tích phân số:

$$\theta(t) = \theta(t_0) + \int_{t_0}^t \omega_z(\tau) d\tau \approx \theta(t_0) + \sum_{k=1}^n \omega_z(k) \cdot \Delta t$$

Trong đó:

- $\omega_z(k)$: tốc độ quay quanh trục z tại bước thời gian thứ k
- Δt : chu kỳ lấy mẫu (sampling period)

Tuy nhiên, vì phép tích phân này dễ bị lệch do nhiễu và trôi (drift), hệ thống sử dụng kết hợp thông tin từ marker (ArUco) để hiệu chỉnh lại góc phương vị trong bước update của bộ ước lượng hợp nhất.

2.6.2. Tính toán vận tốc và vị trí từ encoder

Encoder gắn vào mỗi động cơ giúp đo được số xung phát ra theo mỗi vòng quay, từ đó suy ra được vận tốc góc và vận tốc tuyến tính của từng bánh xe. Với encoder có độ phân giải N xung/vòng, số xung đọc được trong khoảng thời gian Δt là ΔC , ta có:

Góc quay bánh xe:

$$\Delta\theta = \frac{2\pi \cdot \Delta C}{N}$$

Vận tốc góc bánh xe:

$$\omega = \frac{\Delta\theta}{\Delta t} = \frac{2\pi \cdot \Delta C}{N \cdot \Delta t}$$

Vận tốc tuyến tính của bánh xe (với r là bán kính bánh xe):

$$v = r \cdot \omega = \frac{2\pi r \cdot \Delta C}{N \cdot \Delta t}$$

Khi đã biết vận tốc từng bánh, ta áp dụng mô hình động học của robot Mecanum để tính toán vận tốc toàn cục (v_x, v_y, ω) của thân robot:

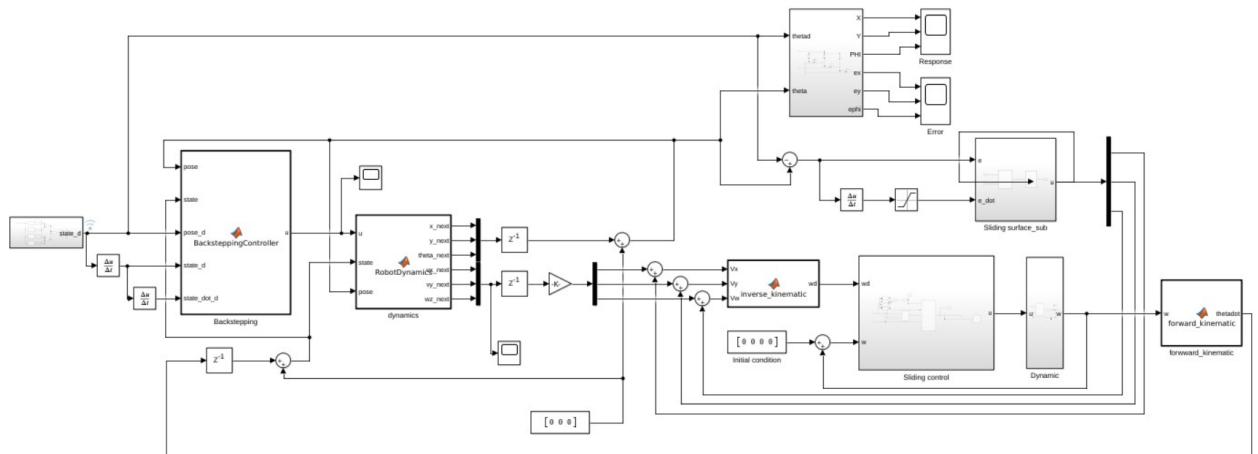
$$\begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} = \frac{r}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -\frac{1}{l+w} & \frac{1}{l+w} & -\frac{1}{l+w} & \frac{1}{l+w} \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}$$

Trong đó:

- r : bán kính bánh xe
- l, w : khoảng cách từ tâm robot đến bánh theo trực dọc và ngang
- $\omega_1 \rightarrow \omega_4$: vận tốc góc của 4 bánh (theo thứ tự trái-trước, phải-trước, phải-sau, trái-sau)

CHƯƠNG 3: MÔ PHỎNG MATLAB

3.1. Tổng quan mô hình và thông số mô phỏng



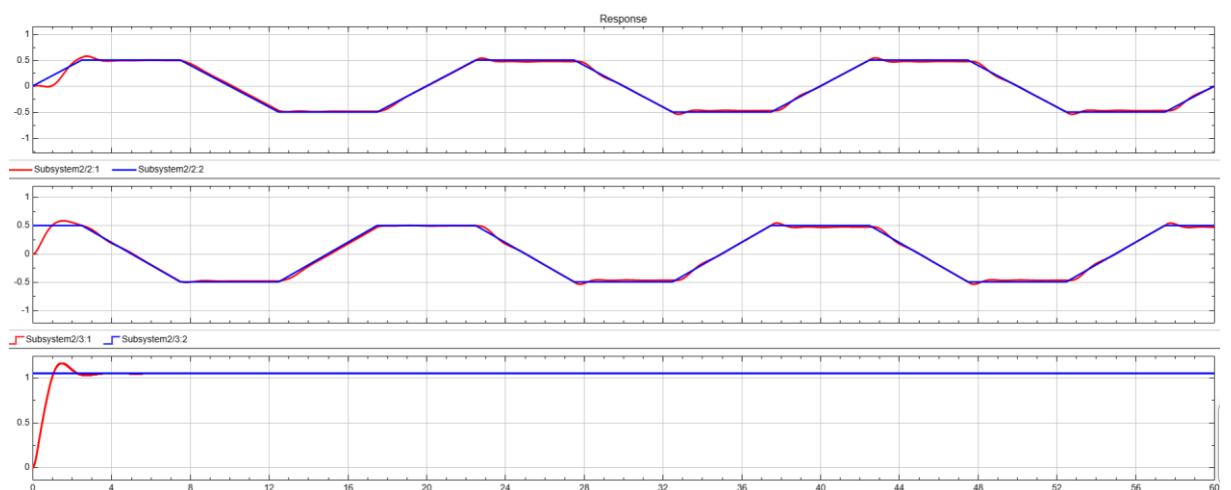
Hình 3.1: Sơ đồ khái quát tổng quan

Bảng 3.1: Thông số mô phỏng

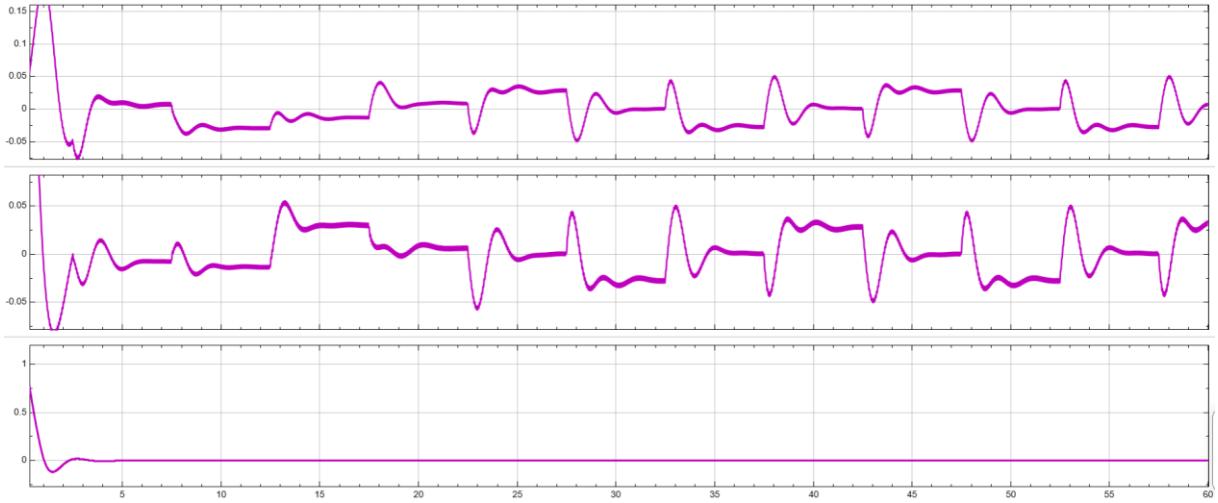
Ký hiệu	Ý nghĩa	Độ lớn	Đơn vị
r	Bán kính bánh xe	0.035	m
m	Tổng khối lượng của robot	6	Kg
L1	Khoảng cách từ trục Byb đến bánh xe	0.11	m
L2	Khoảng cách từ trục Bxb đến bánh xe	0.1	m
I	Mô men quán tính của robot	0.22	Kg.m ²
I _b	Mô men quán tính của bánh xe	0.02328	Kg.m ²

3.2 Kết quả mô phỏng

- Hình vuông

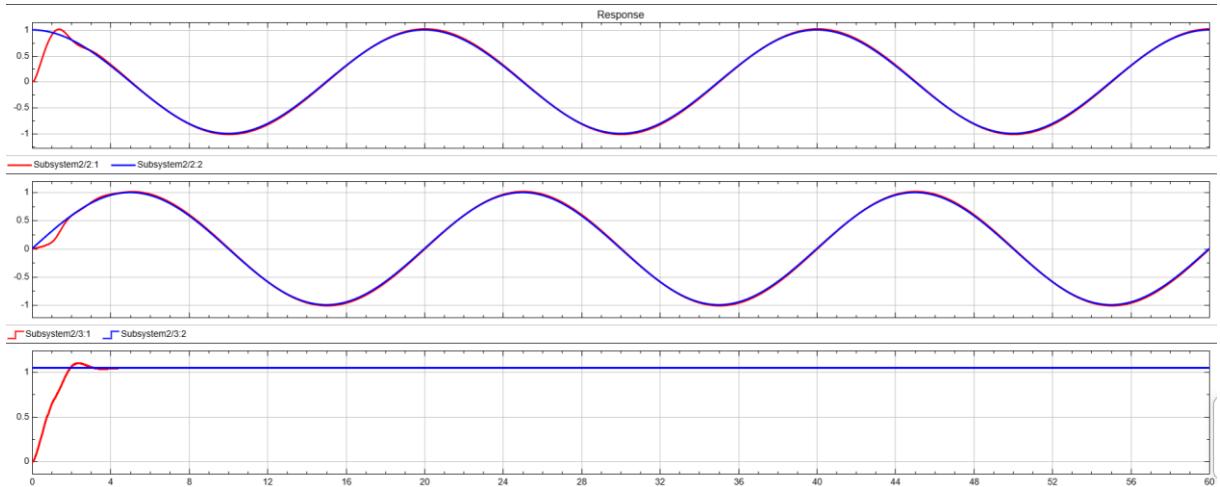


Hình 3.2: Kết quả mô phỏng bám quỹ đạo hình vuông

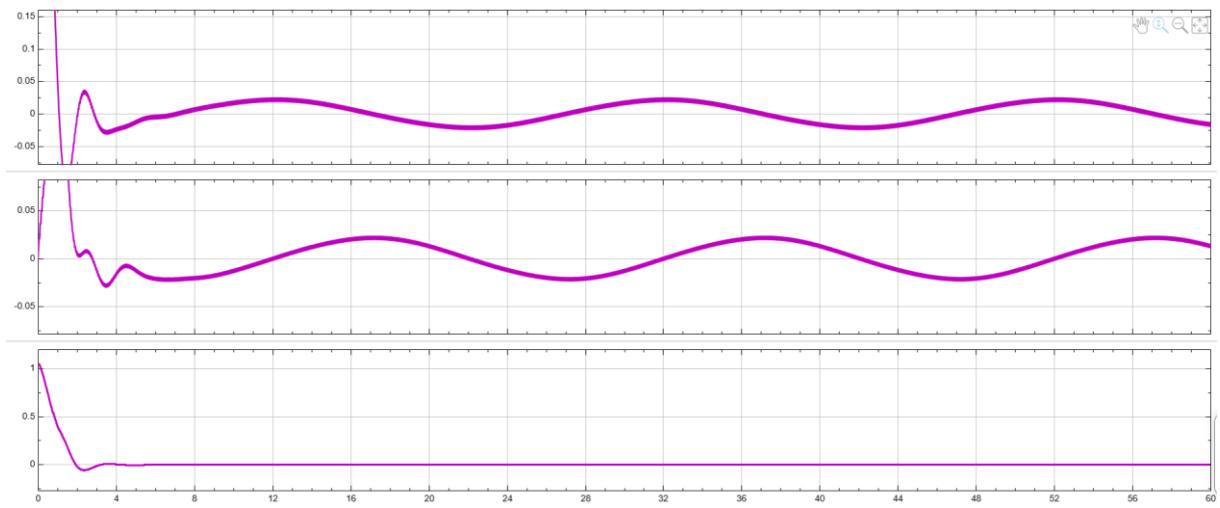


Hình 3.3: Sai số vị trí và góc mô phỏng bám quỹ đạo hình vuông

- Hình tròn

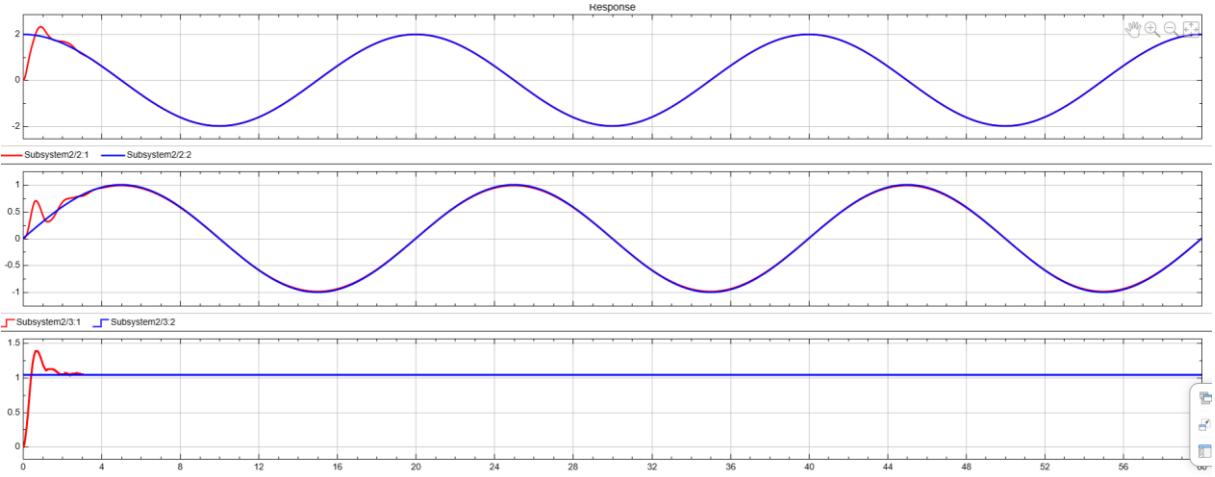


Hình 3.4: Kết quả mô phỏng bám quỹ đạo hình tròn

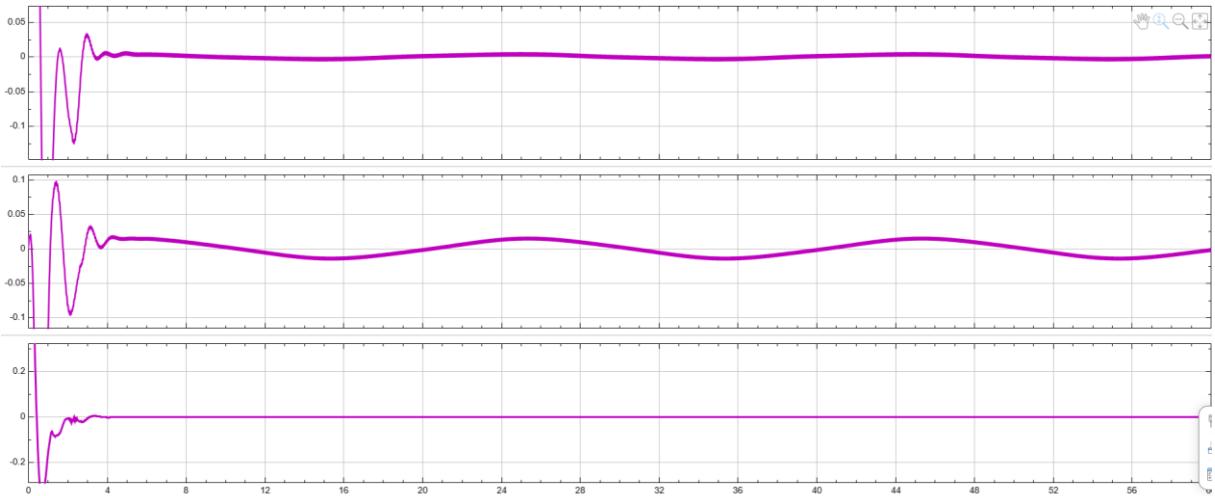


Hình 3.5: Sai số vị trí và góc mô phỏng bám quỹ đạo hình tròn

- Hình ellipse

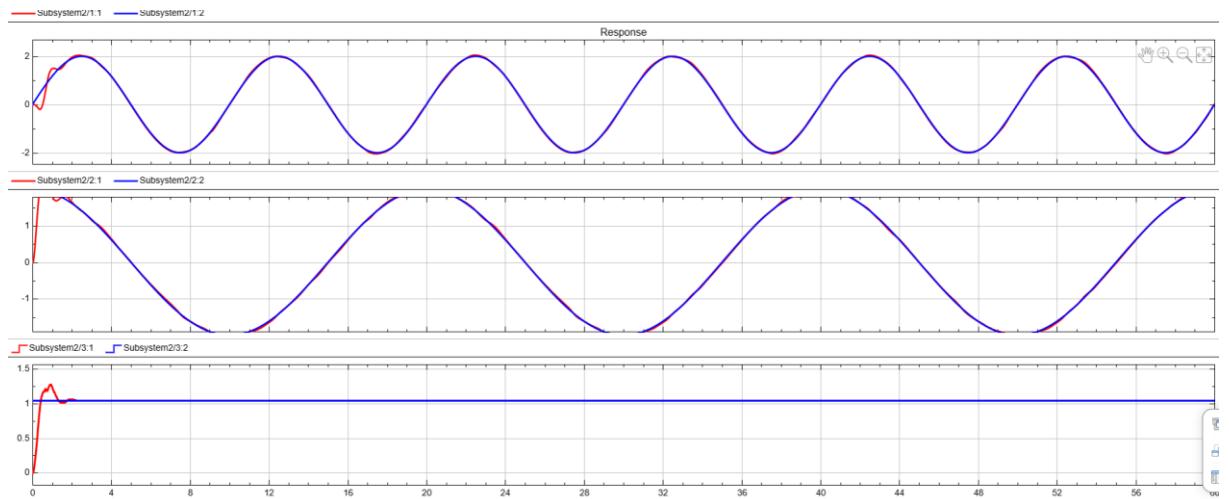


Hình 3.6: Kết quả mô phỏng bám quỹ đạo hình ellipse

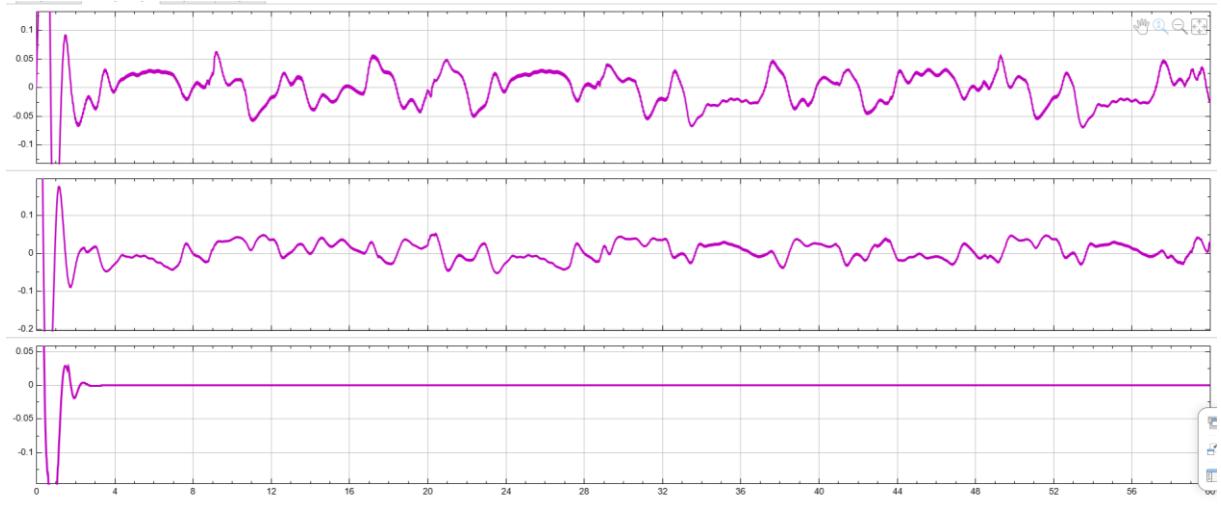


Hình 3.7: Sai số vị trí và góc mô phỏng bám quỹ đạo hình ellipse

- Hình số 8



Hình 3.8: Kết quả mô phỏng bám quỹ đạo hình số 8



Hình 3.9: Sai số vị trí và góc mô phỏng bám quỹ đạo hình số 8

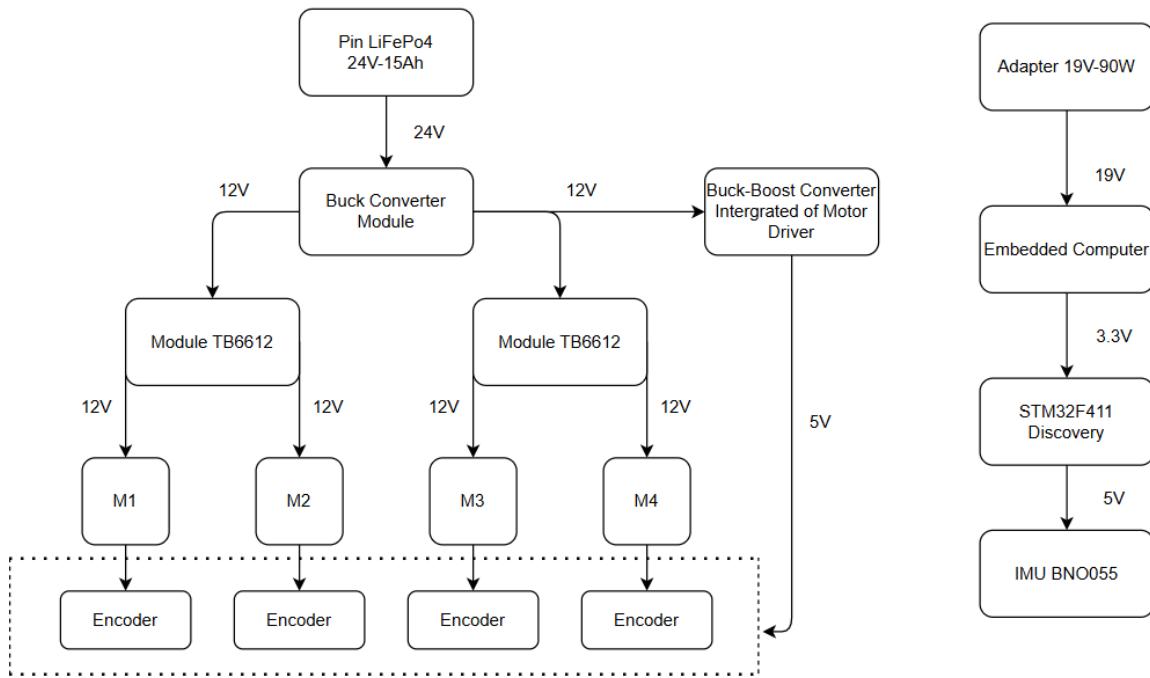
Bảng 3.1 Hiệu suất bám quỹ đạo của bộ điều khiển khi mô phỏng

Quỹ đạo	Sai số x (m)	Sai số y (m)	Sai số góc yaw (rad)
Vuông	0.03 – 0.05	0.03 – 0.05	0.05
Tròn	0.03 – 0.04	0.03 – 0.04	0.03
Elipse	0.02 – 0.03	0.02 – 0.03	0.02
Số 8	0.05 – 0.06	0.05 – 0.06	0.04

CHƯƠNG 4: THI CÔNG PHẦN CỨNG

4.1. Sơ đồ kết nối

4.1.1. Sơ đồ công suất

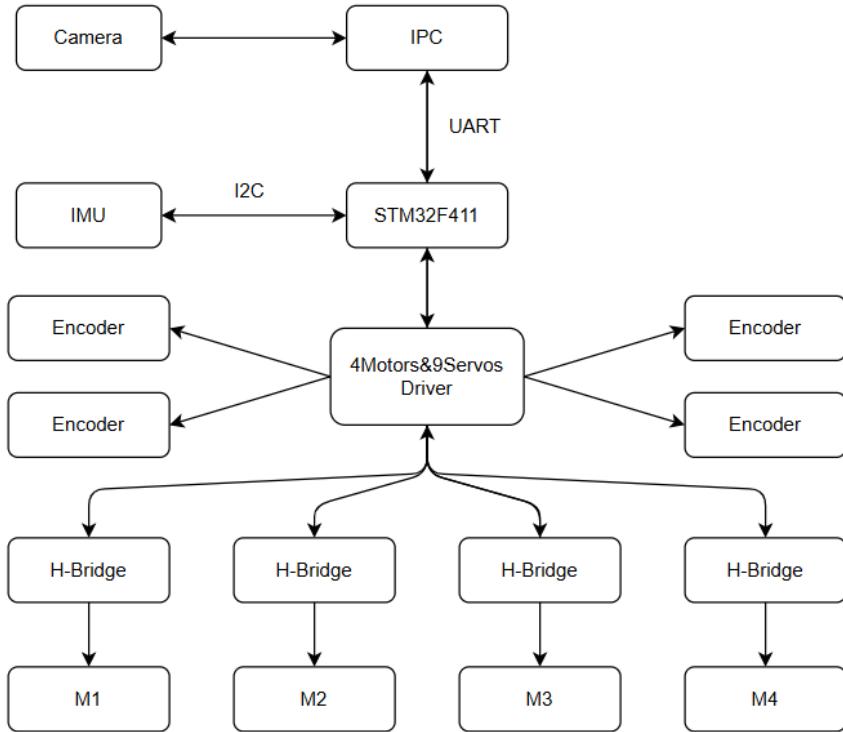


Hình 4.1: Sơ đồ khái công suất

Hệ thống cung cấp nguồn điện cho xe FMWMR

- Toàn bộ nguồn năng lượng của xe FMWMR được cung cấp từ pin LiFePo4 24V – 15Ah và Adapter 19V-90W. Pin LiFePo4 có ưu điểm dung lượng lớn và khả năng cung cấp dòng cao, đáp ứng tốt yêu cầu khi bốn động cơ hoạt động đồng thời ở chế độ mô-men xoắn cực đại và tốc độ cao.
 - Nguồn điện 24V từ pin trước tiên được đưa qua mạch hạ áp Buck Converter, tạo ra mức điện áp ổn định 12V để cung cấp cho các mô-đun điều khiển động cơ TB6612. Các mô-đun này sử dụng điện áp 12V làm nguồn cấp cho động cơ (M1, M2, M3, M4), đảm bảo công suất hoạt động ổn định và hiệu quả.
 - Song song đó, nguồn 12V cũng được đưa qua mạch Buck-Boost Converter tích hợp trên board điều khiển động cơ. Mạch này có nhiệm vụ chuyển đổi từ 12V xuống 5V ổn định, dùng để cấp cho các encoder gắn trên động cơ, sử dụng nguồn 5V để hoạt động và xuất tín hiệu xung về STM32F411.
 - Adapter 19V-90W đóng vai trò cung cấp nguồn điện cho máy tính nhúng phục vụ mạch điều khiển STM32F411. Máy tính nhúng cấp nguồn 3V3 ổn định cho STM32F411 và từ đó vi xử lí cấp nguồn 5V cho module IMU BNO055.

2.5.2. Sơ đồ kết nối các thành phần



Hình 4.2: Sơ đồ kết nối các thành phần

Trong hệ thống FMWMR, máy tính nhúng được xem là trung tâm xử lý và điều phối, đóng vai trò đại diện cho hệ thống truyền thông của toàn bộ cấu trúc phần cứng. Các kết nối chính trong hệ thống được mô tả như sau:

- Kết nối giữa camera và máy tính nhúng:

Camera USB được kết nối trực tiếp với máy tính nhúng để ghi lại các khung hình trong phạm vi hoạt động của robot. Dữ liệu hình ảnh được xử lý để trích xuất thông tin từ ArUco Marker, phục vụ cho việc ước lượng vị trí và góc hướng của robot.

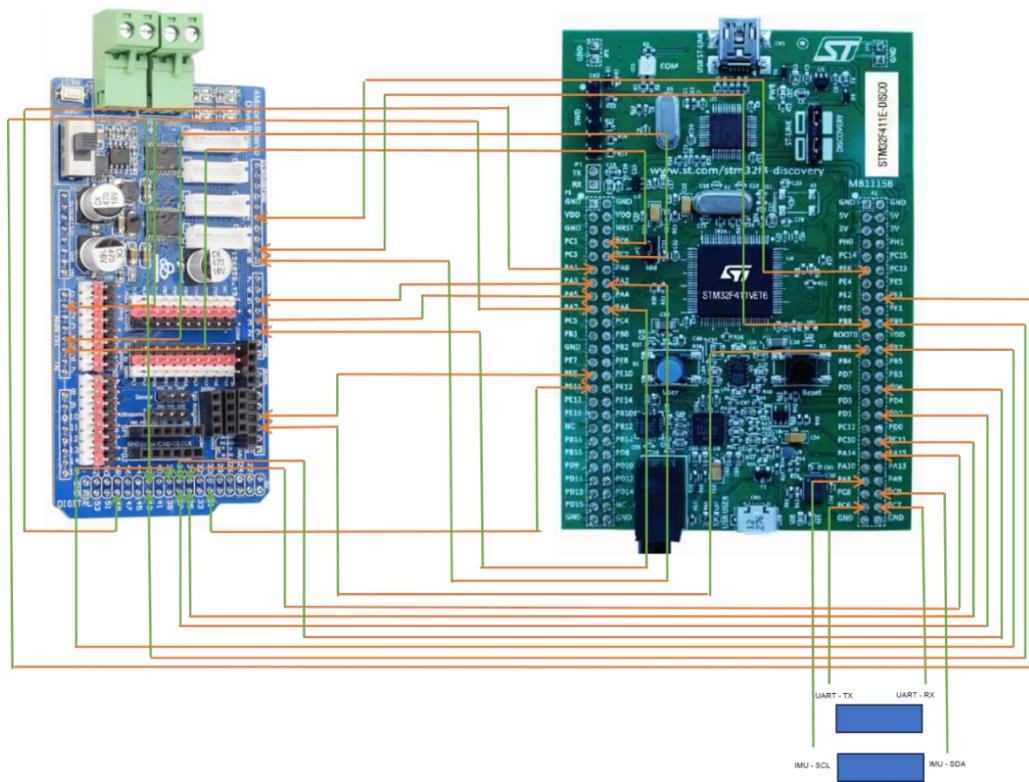
- Kết nối giữa máy tính nhúng và STM32F411:

Máy tính nhúng và vi điều khiển STM32F411 Discovery giao tiếp thông qua UART. Trong đó, máy tính nhúng gửi lệnh điều khiển (vận tốc tham chiếu) đến STM32F411, đồng thời nhận lại dữ liệu phản hồi về trạng thái vận hành của robot.

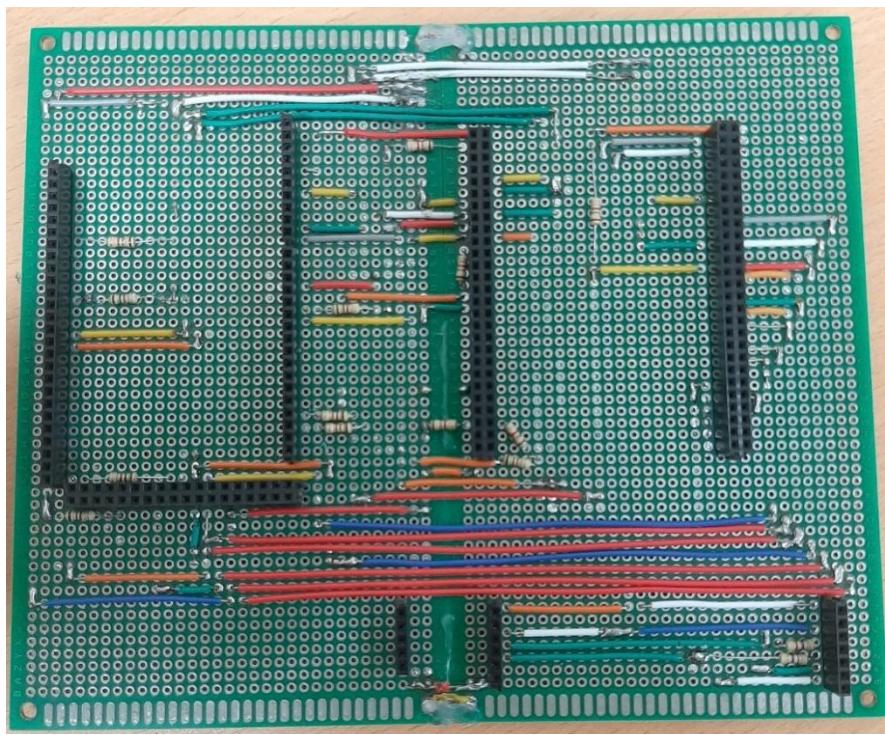
- Kết nối của STM32F411 với phần cứng ngoại vi:

STM32F411 xuất tín hiệu PWM và điều khiển hướng tới board driver để vận hành độc lập từng bánh xe, đồng thời thu nhận tín hiệu từ bốn encoder để đo vận tốc thực tế và phục vụ phản hồi điều khiển. Bên cạnh đó, STM32F411 giao tiếp với cảm biến IMU BNO055 qua I2C để thu thập dữ liệu quán tính, hỗ trợ cho quá trình ước lượng trạng thái chuyển động của robot.

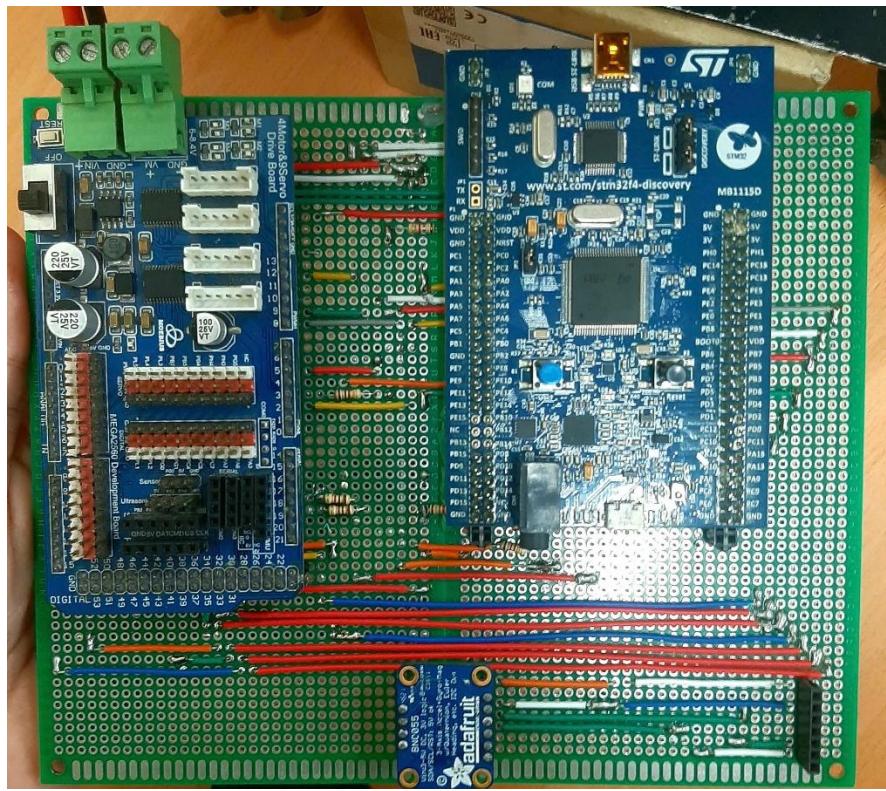
4.2. Thiết kế mạch cho phần cứng



Hình 4.3: Thiết kế sơ đồ mạch cho phần cứng



Hình 4.4: Mạch thực tế sau khi hàn



Hình 4.5: Mạch khi được sử dụng

4.3. Chi tiết phần cứng của hệ thống

Hệ thống robot Mecanum được xây dựng từ nhiều thành phần phần cứng được lựa chọn cẩn thận nhằm đảm bảo hiệu suất xử lý, độ tin cậy và khả năng mở rộng của toàn bộ hệ thống định vị và điều khiển. Các thành phần chính bao gồm:

4.3.1. Máy tính công nghiệp (IPC)

Bộ xử lý trung tâm của hệ thống là một máy tính công nghiệp (IPC) có hiệu suất cao, đảm nhiệm việc xử lý các thuật toán định vị, dẫn đường và điều khiển robot theo thời gian thực. Cấu hình chi tiết của IPC như sau:



Hình 4.6: Máy tính nhúng công nghiệp

Bảng 4.1: Thông số kỹ thuật máy tính nhúng

Thành phần	Thông số kỹ thuật
CPU	Intel® Core i7-10510U, 64-bit, 4 nhân 8 luồng
GPU tích hợp	Intel® UHD Graphics
RAM	16GB DDR4, bus 2400 MHz
Vai trò	Xử lý thuật toán định vị, lập kế hoạch và điều khiển

4.3.2. Động cơ encoder

Hệ thống truyền động sử dụng các động cơ DC loại CHR-GM37-520, tích hợp sẵn encoder nhằm cung cấp phản hồi vận tốc chính xác phục vụ cho việc điều khiển.



Hình 4.7: Động cơ DC giảm tốc CHR-GM37-520 (with Encoder)

Bảng 4.2: Thông số kỹ thuật động cơ CHR-GM37-520

Thông số	Giá trị
Nguồn cung cấp	12V DC
Tốc độ không tải	200 vòng/phút (RPM)
Loại encoder	Encoder quang học 2 kênh
Độ phân giải encoder	240 xung/vòng
Vai trò	Tạo mô men, đọc tốc độ

4.3.3. Vị điều khiển (MCU)

Vị điều khiển chính được sử dụng trong hệ thống là STM32F411, đóng vai trò trung gian giữa các cảm biến, động cơ và IPC:



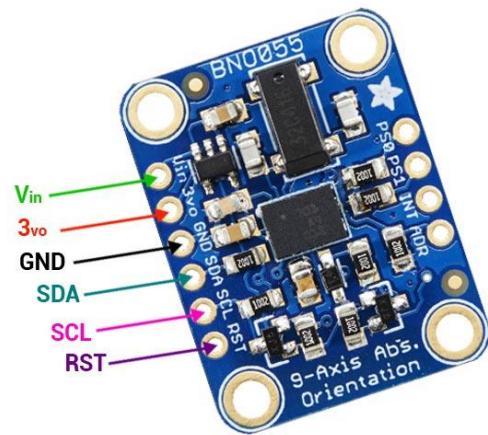
Hình 4.8: Vị điều khiển STM32F411VG Discovery

Bảng 4.3: Thông số kỹ thuật vị điều khiển STM32F411

Thông số	Giá trị
Vị xử lý chính	ARM Cortex-M4 32-bit, STM32F411
Tốc độ xung nhịp	Lên đến 100 MHz
Số chân Digital I/O	100
Số chân Analog Input	3 ADC 12-bit, tối đa 16 kênh chuyển đổi
Timer	11 Timer
Giao tiếp UART	3 USART/UART
Giao tiếp I2C	3 I2C
Vai trò	Điều khiển động cơ, đọc encoder, giao tiếp IPC

4.3.4. Cảm biến định hướng (IMU)

Để thu thập thông tin về góc quay của robot (yaw), hệ thống sử dụng cảm biến quán tính BNO055.



Hình 4.8: Cảm biến gia tốc BNO055

Bảng 4.4: Thông số kỹ thuật cảm biến gia tốc BNO055

Thông số	Giá trị
Loại cảm biến	IMU 9 trục (Accelerometer, Gyroscope, Magnetometer)
Dải đo góc quay (yaw)	$\pm 250^\circ/\text{s}$
Độ phân giải gyroscope	16 bits
Giao tiếp	I2C (100–400 kHz) / UART (3.0 Mbps)
Đầu ra dữ liệu	Quaternion, Euler angles, Linear Acceleration
Cấp nguồn	3.3V hoặc 5V
Tốc độ cập nhật	Tối đa 100 Hz (orientation)
Vai trò	Cung cấp tốc độ góc yaw cho định vị và điều hướng

4.3.5 Camera phát hiện ArUco

Hệ thống sử dụng một webcam HD để phát hiện các marker ArUco bố trí trong môi trường nhằm định vị chính xác vị trí và hướng của robot:



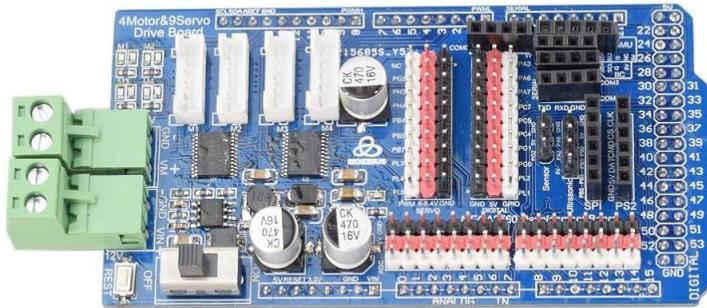
Hình 4.9: Webcam HD 1080

Bảng 4.5: Thông số kỹ thuật webcam

Thông số	Giá trị
Model	Webcam HD 1080p
Độ phân giải tối đa	1920 × 1080 pixels @ 30 fps
Ống kính	Tiêu cự cố định, FOV ~70°
Giao tiếp	USB 2.0/3.0
Tốc độ khung hình	30 fps (Full HD), hỗ trợ giảm độ trễ thực thi
Vai trò	Nhận diện ArUco marker để định vị tuyệt đối

4.3.6. 9 Channel Servo Shield Driver Board for Arduino ATmega 2560

Driver Board có chứa 2 module TB6612 dùng để điều khiển 4 Motor đồng thời hỗ trợ đọc giá trị encoder thông qua 4 port M1, M2, M3, M4.



Hình 4.10: 4Motor&9Servo Driver board

Bảng 4.6: Thông số kỹ thuật 4Motor&9Servo Driver board

Thông số	Giá trị
Nguồn cung cấp	12V DC
Hỗ trợ điều khiển motor	4 động cơ motor DC và 4 encoders
Giao tiếp UART	2 USART/UART
Vai trò	Điều khiển 4 động cơ và đọc encoder

CHƯƠNG 5: THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

5.1. Các bước thực nghiệm

Hệ thống xe tự hành sử dụng bánh Mecanum được thiết lập và triển khai trong môi trường Ubuntu – một hệ điều hành mã nguồn mở, miễn phí, dựa trên nền tảng Linux. Ubuntu nổi bật bởi giao diện thân thiện, tính bảo mật cao và khả năng tương thích tốt với nhiều ứng dụng lập trình, trí tuệ nhân tạo và robot. Nhờ đó, Ubuntu thường được lựa chọn để phát triển các hệ thống nhúng và điều khiển robot.



Hình 5.1: Hệ điều hành Ubuntu

Sau khi khởi động Ubuntu, thiết bị USB cần được gắn (attach) từ môi trường Windows sang Ubuntu nhằm phục vụ quá trình truyền và nhận dữ liệu qua giao tiếp UART.

Khi kết nối UART được thiết lập thành công, luận văn sử dụng phần mềm STM32CubeIDE để lập trình, gỡ lỗi và nạp mã điều khiển xuống bo mạch điều khiển chính của robot.



Hình 5.2: Phần mềm STM32CubeIDE

Trong quá trình thực nghiệm, ba cửa sổ terminal được khởi tạo độc lập để thực hiện:

- Giao tiếp UART thu nhận dữ liệu từ bo mạch.

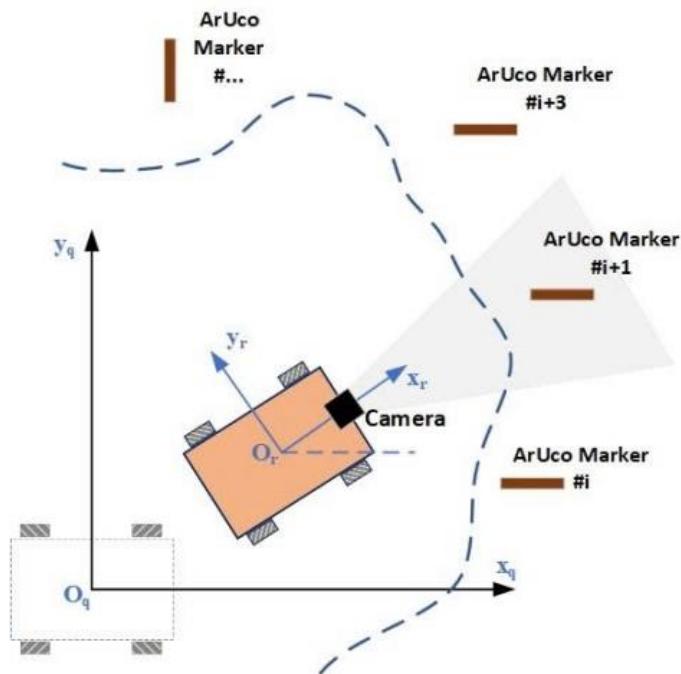
- Chạy bộ ước lượng xử lý dữ liệu (như bộ ước lượng Kalman).
- Thực thi bộ điều khiển điều hướng robot.

Để thuận tiện cho việc vận hành robot mà không cần sử dụng trực tiếp bàn phím, chuột hoặc màn hình tại máy tính nhúng, phần mềm AnyDesk được sử dụng nhằm điều khiển từ xa hệ thống. AnyDesk là một phần mềm truy cập máy tính từ xa thông qua internet, cho phép người dùng điều khiển máy tính mục tiêu một cách nhanh chóng và an toàn. Việc sử dụng AnyDesk giúp nâng cao tính linh hoạt trong việc theo dõi và điều chỉnh hệ thống từ xa.



Hình 5.3: Phần mềm Anydesk

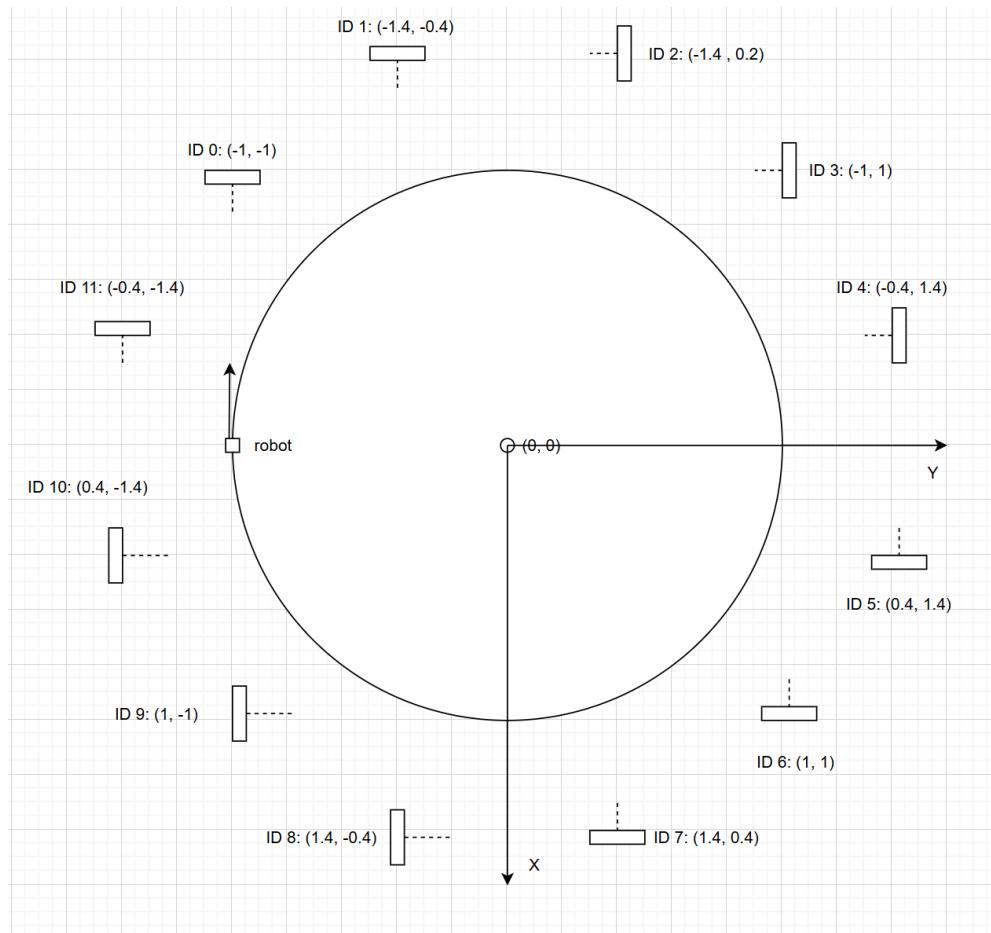
Hệ thống thị giác sử dụng camera được vận hành trên môi trường Windows. Để kết nối và trao đổi dữ liệu với các node ROS trong Ubuntu, chúng tôi sử dụng giao thức Socket TCP/IP. Qua đó, dữ liệu vị trí được trích xuất từ ảnh ArUco sẽ được gửi từ máy Windows đến máy Ubuntu để các node ROS xử lý tiếp.



Hình 5.4: Mô hình robot kết hợp với định vị bằng ArUco marker trong hệ tọa độ toàn cục (global frame)

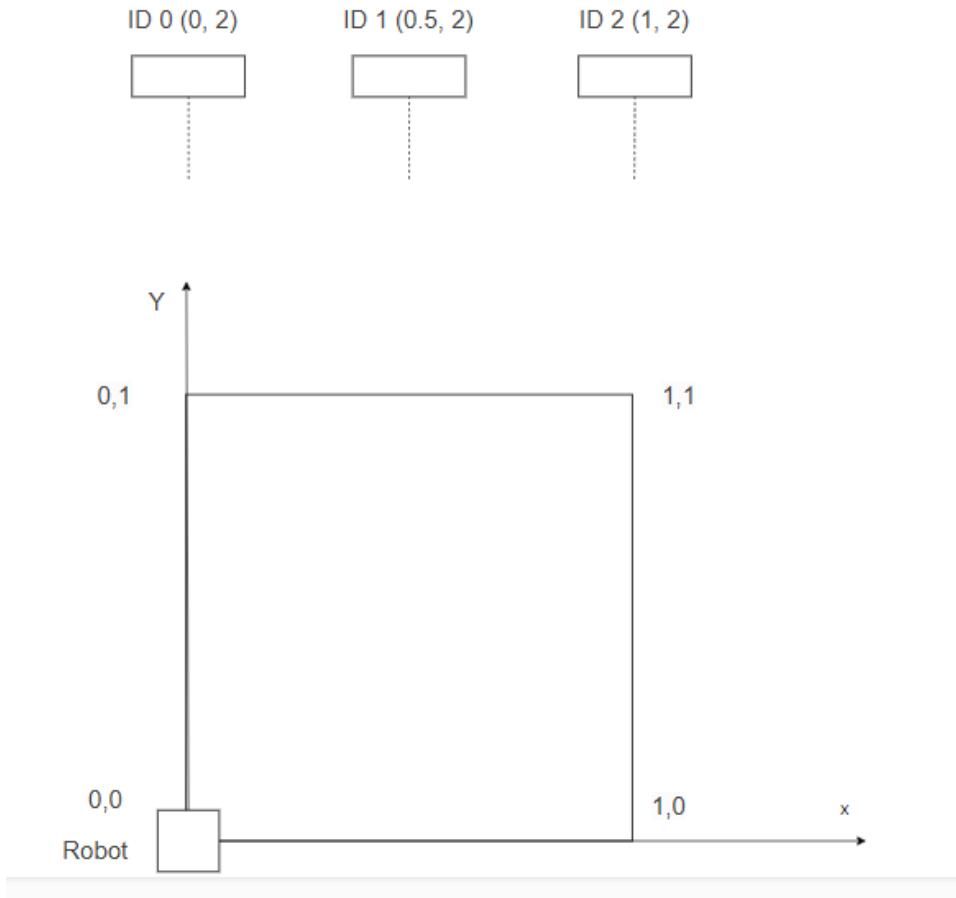
Để phục vụ mục tiêu định vị và dẫn đường, chúng tôi thiết lập bản đồ bằng 2 cách bố trí marker ArUco.

- Cách thứ nhất bố trí các marker ID từ 0 đến 11 trên mặt phẳng làm việc. Các marker được đặt sao cho đảm bảo robot có thể quan sát được khi di chuyển theo quỹ đạo hình tròn với bán kính 1 mét. Sơ đồ bố trí marker được minh họa trong Hình 5.2.



Hình 5.5: Setup cho hình tròn góc thay đổi

- Cách thứ hai bố trí các marker ID từ 0 đến 2 trên mặt phẳng làm việc. Các marker được đặt sao cho đảm bảo robot có thể quan sát được khi di chuyển theo quỹ đạo hình vuông với cạnh 1 mét. Sơ đồ bố trí marker được minh họa trong Hình 5.3.



Hình 5.6: Setup cho hình vuông góc không thay đổi

Sau khi hoàn tất thiết lập môi trường phần mềm và bố trí không gian thực nghiệm, chúng tôi tiến hành chạy đồng thời các chương trình trên ba terminal, khởi chạy camera trên Windows và nạp chương trình điều khiển xuống bo mạch qua Arduino IDE. Khi hệ thống robot hoàn tất một chu kỳ di chuyển, toàn bộ dữ liệu được ghi nhận và xử lý để hiển thị kết quả.

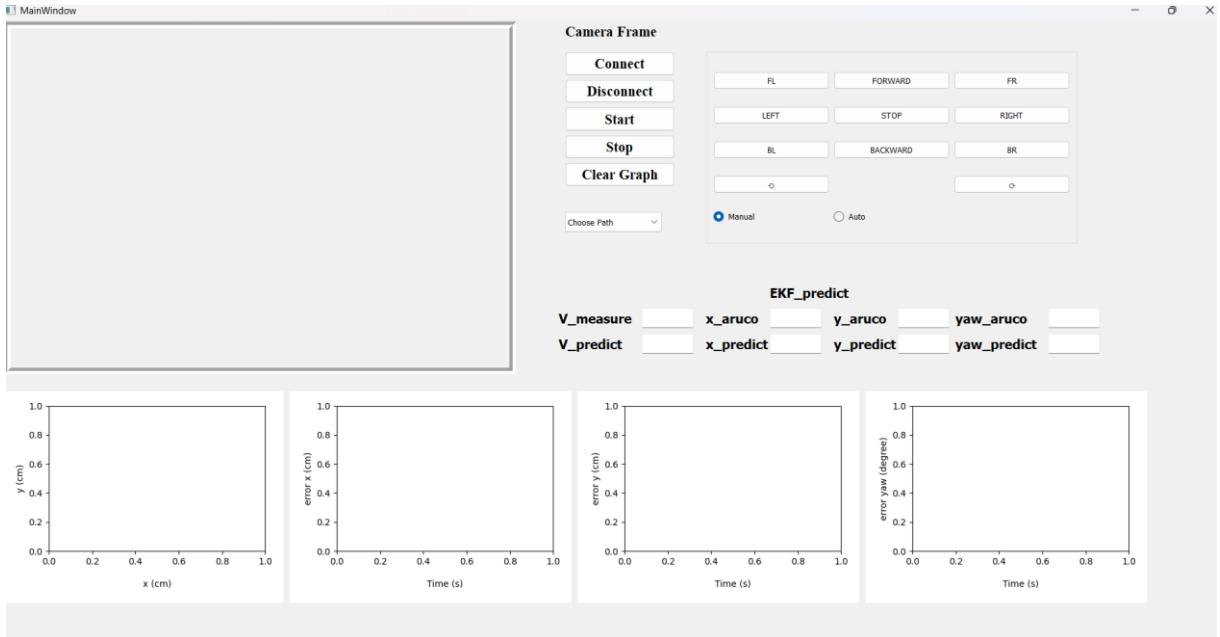
Các kết quả thu được bao gồm:

- Đồ thị thể hiện vị trí thực tế (theo ArUco), vị trí sau khi lọc Kalman, và vị trí mong muốn.
- Đồ thị sai số vị trí và sai số góc theo thời gian.

5.2. Thiết kế giao diện người dùng

Nhằm dễ dàng hơn cho việc điều khiển và quan sát, một giao diện người dùng được thiết kế để hỗ trợ người dùng. Trong hệ thống, giao diện người dùng được phát triển trực tiếp bằng PyQt5 trên Python, thay vì sử dụng công cụ thiết kế Qt Designer. Các thành phần giao diện như nút chức năng, khung hiển thị hình ảnh và các ô hiển thị trạng thái được xây dựng thông qua mã lập trình, giúp linh hoạt hơn trong việc tùy chỉnh

cũng như tích hợp với các thuật toán xử lý và điều khiển robot. Nhờ vậy, giao diện vừa đảm bảo tính trực quan, vừa dễ dàng mở rộng khi bổ sung chức năng mới.



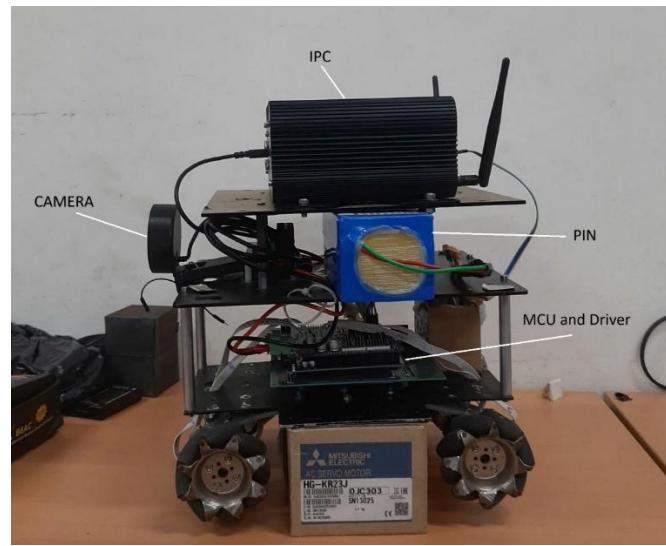
Hình 5.7: Giao diện thiết kế bằng Qt5

Như được minh họa trong Hình 5.4 ở trên, giao diện người dùng bao gồm một số thành phần sau:

- Camera frame: Hiển thị các khung video từ camera để người dùng quan sát chuyển động của robot.
- Graphs: Bốn đồ thị riêng biệt hiển thị quỹ đạo, sai số theo trục x, y và sai số góc quay.
- Control buttons: Có một số nút được đặt tên là “Connect”, “Disconnect”, “Clear graph”, “Start”, “Stop” và các nút điều khiển bằng tay.
- EKF Parameter: Phần này dùng để hiển thị các giá trị hiện tại như giá trị vị trí, góc từ aruco, giá trị vị trí và góc được ước tính thông qua bộ ước lượng.

5.3. Kết quả thực nghiệm

Hiệu suất của hệ thống điều khiển được đánh giá trong bốn tình huống khác nhau, bao gồm quỹ đạo đường thẳng theo trục X, quỹ đạo đường thẳng theo trục Y, quỹ đạo hình tròn và quỹ đạo hình vuông, được kiểm tra trong phần kết quả của các thí nghiệm. Đồng thời kiểm tra tính chính xác khi sử dụng lần lượt 3 bộ ước lượng EKF, AEKF và EKF chi bình phương cho các quỹ đạo hình tròn và hình vuông.



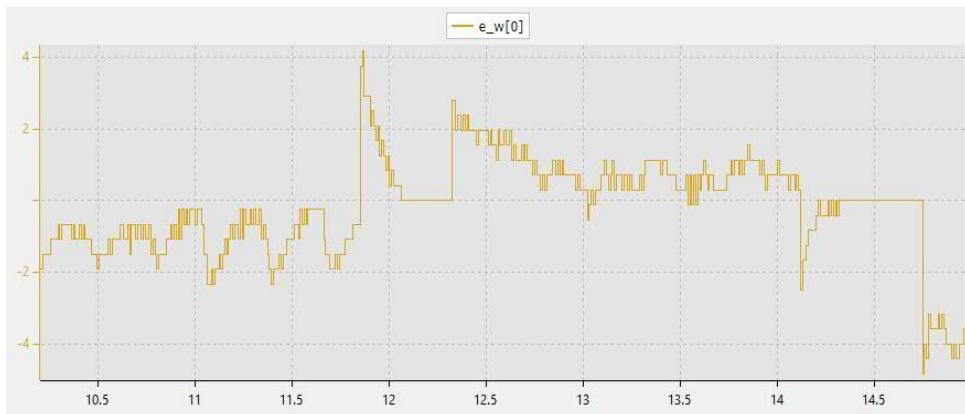
Hình 5.8: Mô hình robot thực nghiệm

5.3.1 Kết quả đáp ứng vận tốc của bộ điều khiển trượt

Đầu tiên ta sẽ kiểm tra đáp ứng của bộ điều khiển trượt trong việc điều khiển vận tốc bánh xe đến vận tốc mong muốn.



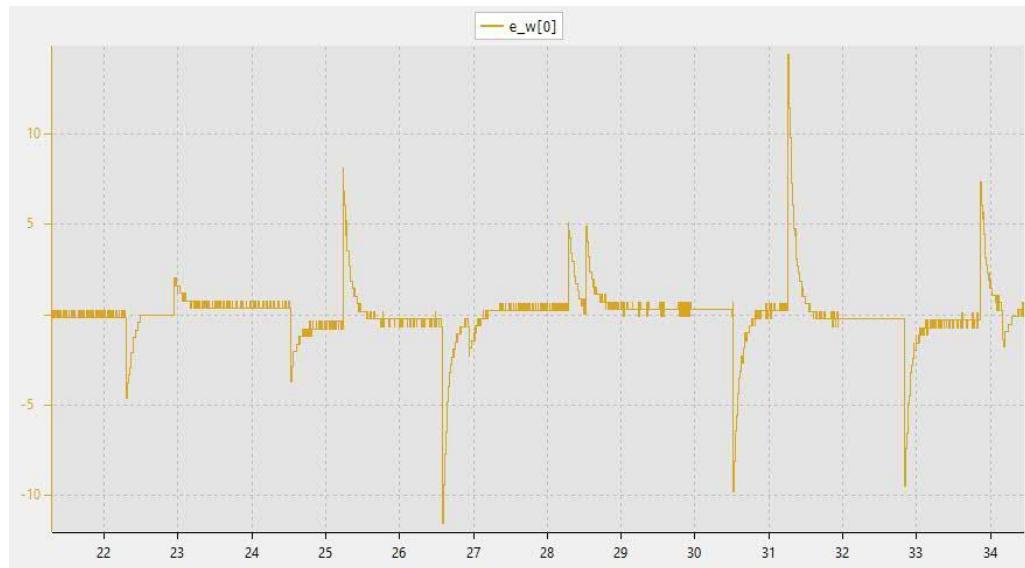
Hình 5.9: Đáp ứng vận tốc của bánh xe trước khi áp dụng bộ điều khiển trượt



Hình 5.10: Sai số vận tốc của bánh xe trước khi áp dụng bộ điều khiển trượt



Hình 5.11: Đáp ứng vận tốc của bánh xe sau khi áp dụng bộ điều khiển trượt



Hình 5.12: Sai số vận tốc của bánh xe sau khi áp dụng bộ điều khiển trượt

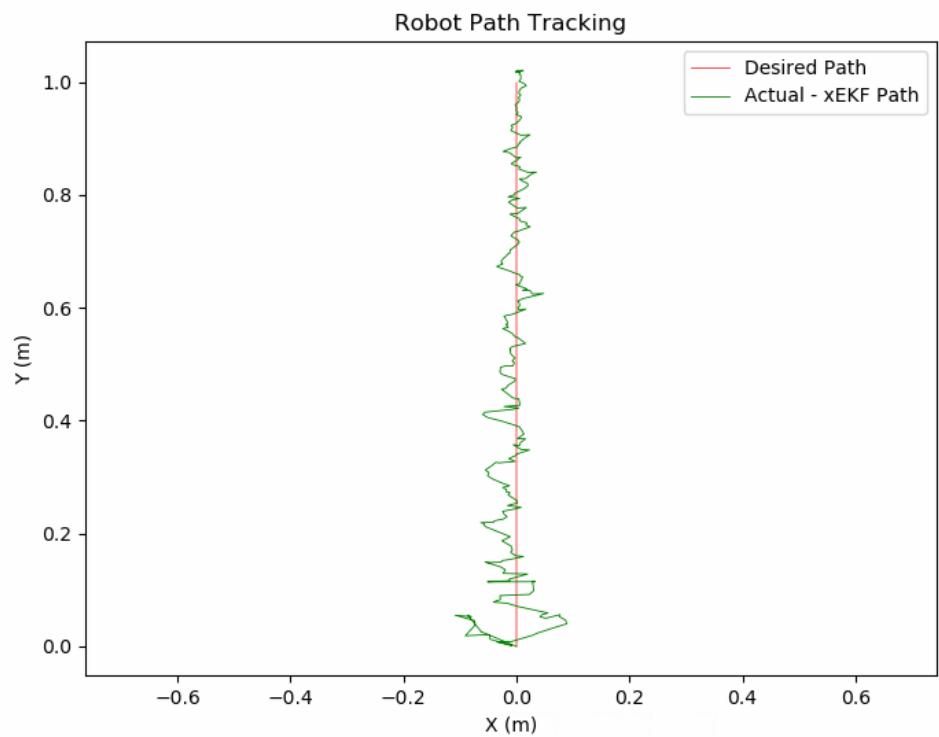
Như quan sát từ hình 5.9 và 5.11, vận tốc bánh xe đạt được tốc độ ổn định hơn và bám sát với vận tốc mong muốn. Sai số vận tốc cũng được cải thiện sau khi áp dụng bộ điều khiển trượt.

Bảng 5.1: Kết quả đáp ứng vận tốc bánh xe

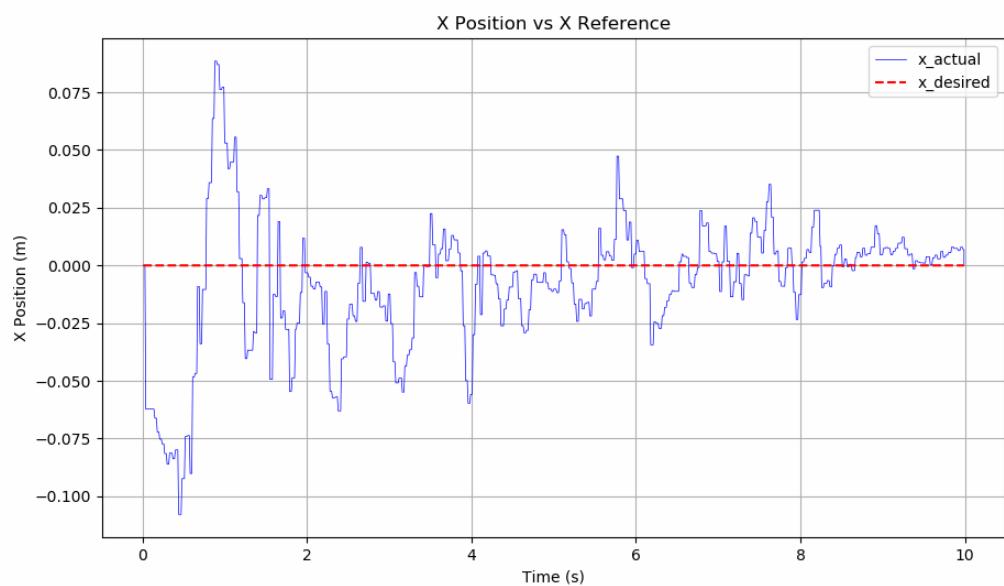
	$\bar{t}_r(s)$	$\bar{t}_s(s)$	Độ vọt lồ (%)	Sai số xác lập(rad/s)
Without SMC	0.45	0.5	5	0.8
With SMC	0.3	0.35	No	0.07

5.3.2. Quỹ đạo đường thẳng theo trục Y

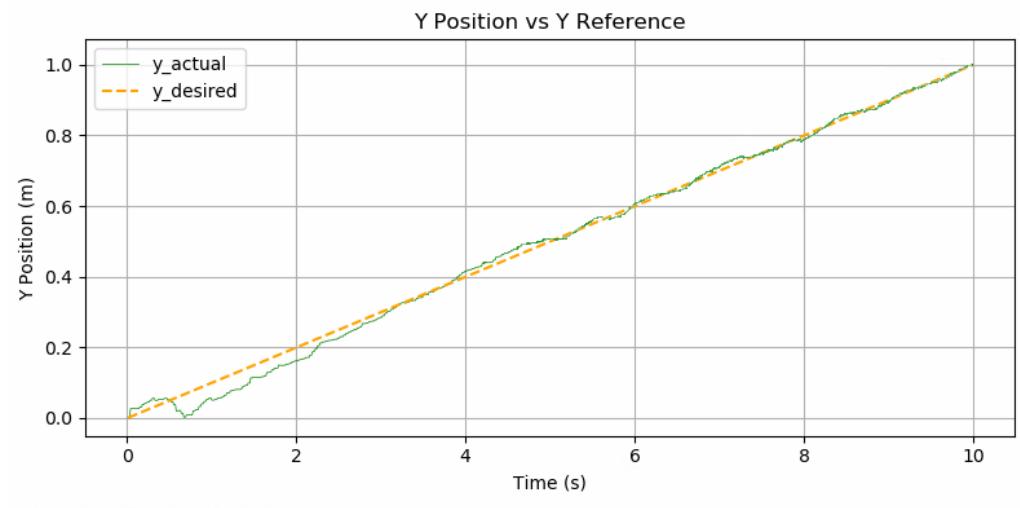
Trong thí nghiệm đầu tiên, FMWMR được giao nhiệm vụ điều hướng đến một vị trí mục tiêu cụ thể (0m; 1m) và giữ nguyên góc quay là 90 độ



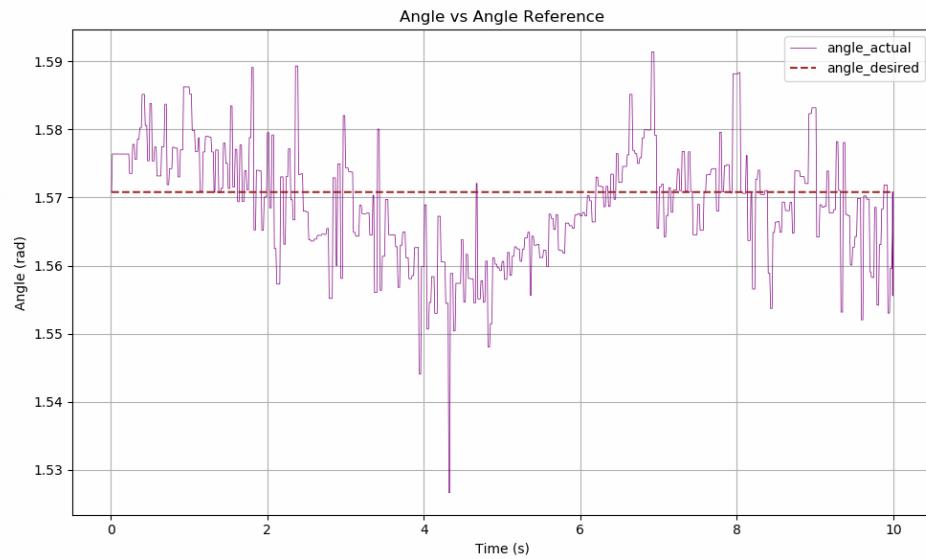
Hình 5.13: Quỹ đạo robot bám quỹ đạo đường thẳng



Hình 5.14: Đáp ứng robot quỹ đạo đường thẳng theo trục X



Hình 5.15: Đáp ứng robot quỹ đạo đường thẳng theo trục Y



Hình 5.16: Đáp ứng robot quỹ đạo đường thẳng theo góc yaw

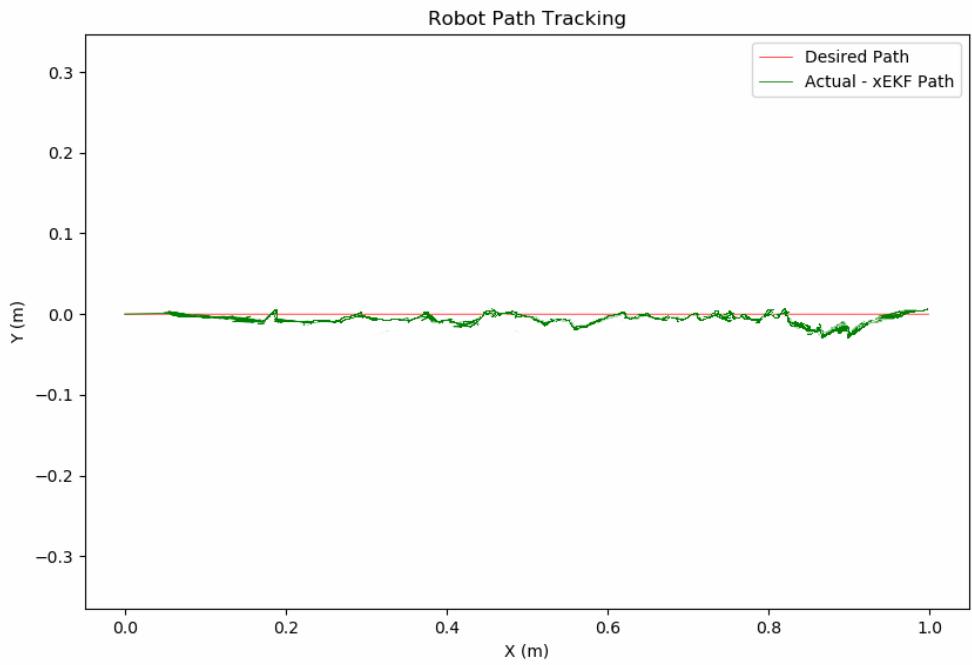
Nhận xét: Robot đã bám quỹ đạo theo đường thẳng với sai số theo trục Y thấp, tuy nhiên còn dao động nhỏ với trục X và góc yaw

Bảng 5.2: Kết quả đáp ứng theo đường thẳng

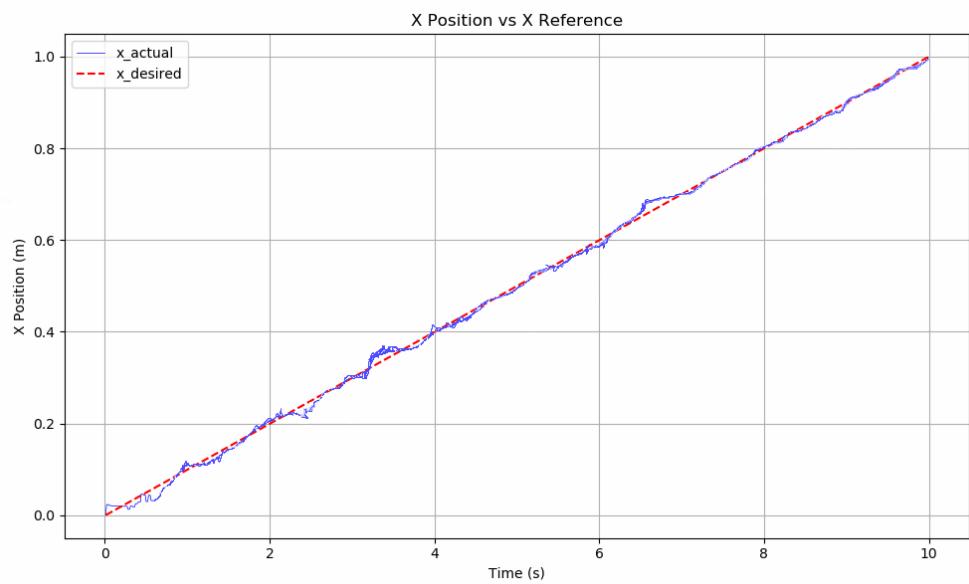
Sai số theo trục X (m)	Sai số theo trục Y (m)	Sai số góc yaw (rad)
0.02 – 0.03	0.025 -0.05	0.01 - 0.02

5.3.3. Quỹ đạo đường thẳng theo trục X

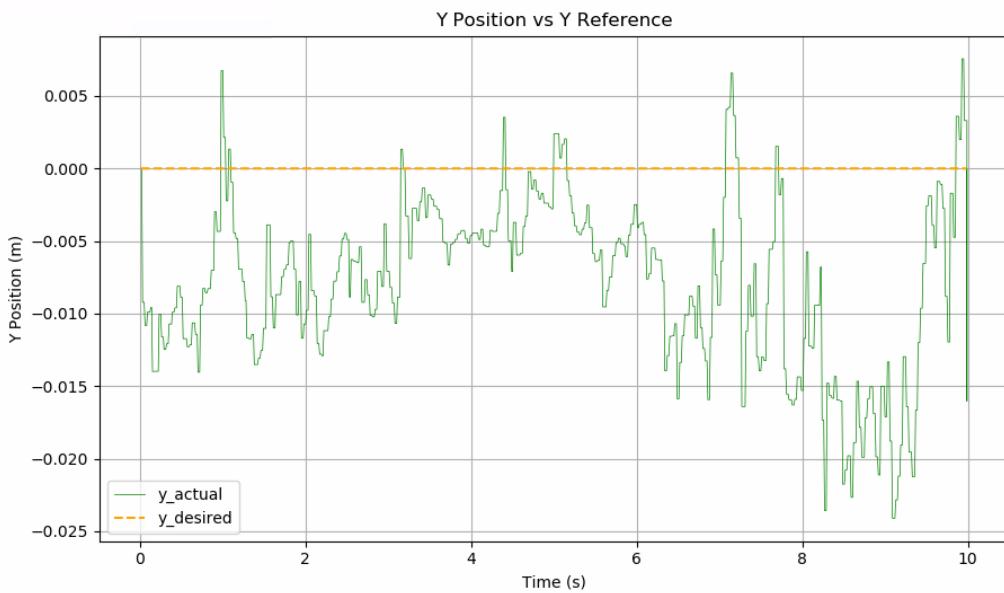
Trong thí nghiệm đầu tiên, FMWMR được giao nhiệm vụ điều hướng đến một vị trí mục tiêu cụ thể (1m; 0m) và giữ nguyên góc quay là 90 độ



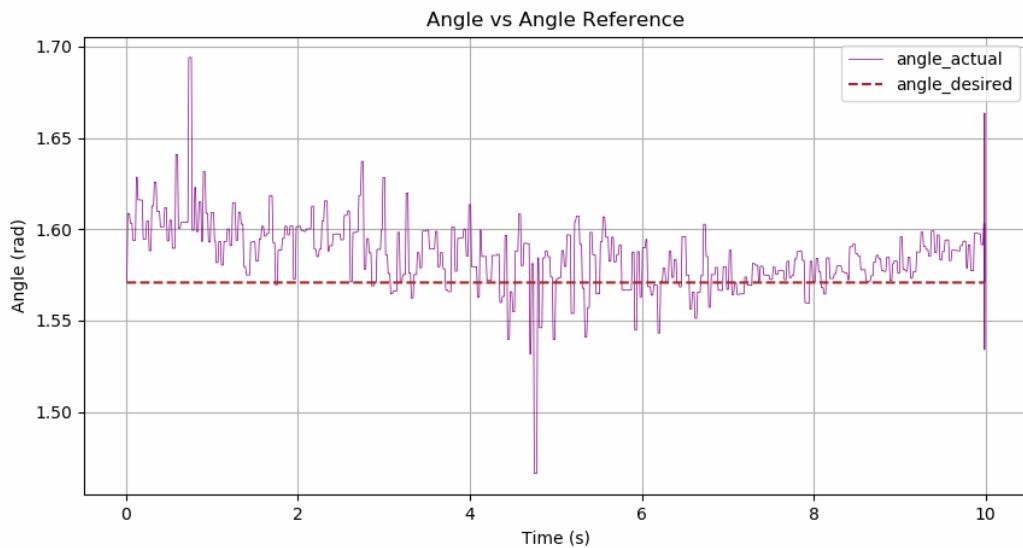
Hình 5.17: Quỹ đạo robot bám quỹ đạo đường ngang



Hình 5.18: Dáp ứng robot quỹ đạo đường ngang theo trục X



Hình 5.19: Đáp ứng robot quỹ đạo đường ngang theo trục Y



Hình 5.20: Đáp ứng robot quỹ đạo đường ngang theo góc yaw

Nhận xét: Robot đã bám quỹ đạo theo đường ngang với sai số theo trục Y thấp, tuy nhiên còn dao động nhỏ với trục X và góc yaw. Dao động của góc yaw cao hơn đáng kể do góc nhìn của camera tới aruco thay đổi qua quá trình di chuyển ngang.

Bảng 5.3: Kết quả đáp ứng theo đường ngang

Sai số theo trục X (m)	Sai số theo trục Y (m)	Sai số góc yaw (rad)
0.02 – 0.03	0.01 -0.02	0.03 - 0.05

5.3.4. Quỹ đạo hình vuông

Sau khi thực hiện với quỹ đạo đường thẳng và đường nằm ngang, ta tiếp tục thực nghiệm với quỹ đạo phức tạp hơn là đường vuông. Với quỹ đạo này ta sẽ sử dụng setup 3 aruco tham chiếu như đã trình bày ở mục 5.1 setup 2.

Trước hết ta sẽ chọn thông số cho các bộ ước lượng:

Bảng 5.4: Thông số lựa chọn cho các bộ ước lượng

Thông số	Giá trị
χ^2 threshold	0.03
Q_{t0}	[0.2, 0.2, 0.1, 0.02]
R_{t0}	[0.005, 0.005, 0.001]
P_{t0}	[0.1, 0.1, 0.1, 0.1]
α_Q	0.93
α_R	1

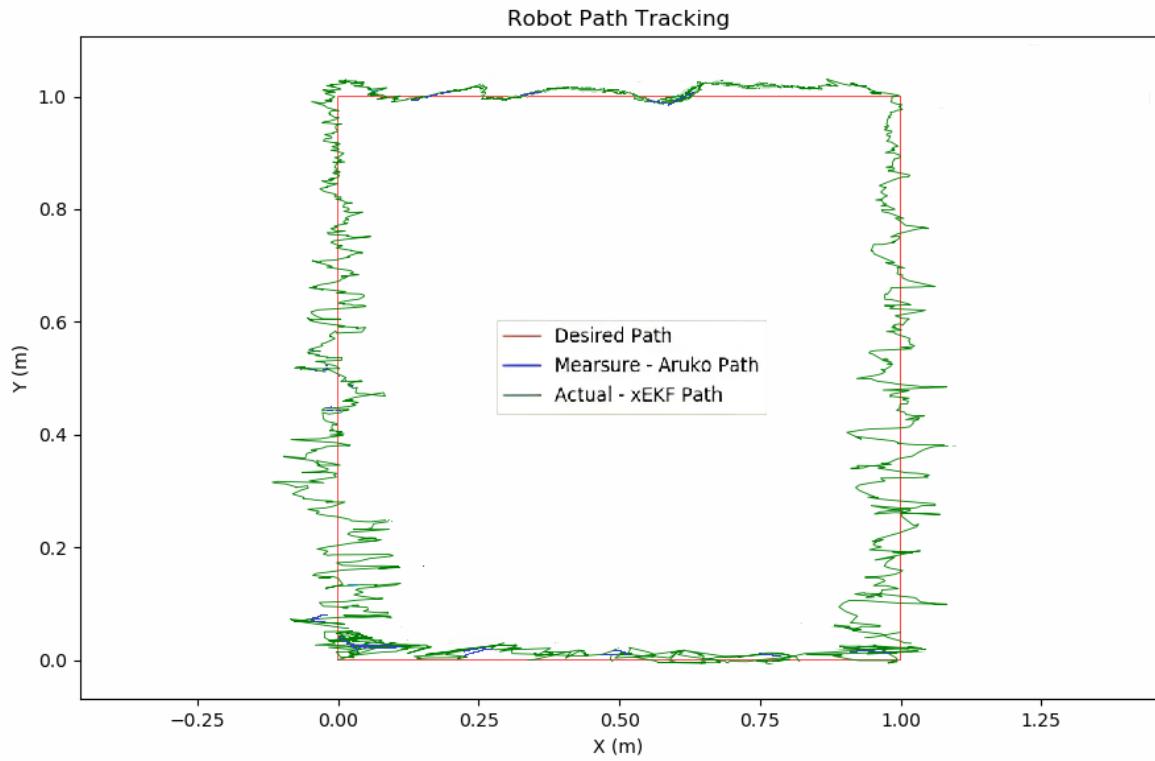
Sau khi lựa chọn được ma trận hiệp phương sai nhiều quá trình Q_t và nhiều do lường R_t phù hợp (như trong Bảng 1), các ma trận này được áp dụng đồng đều cho cả ba phương pháp.

Đối với EKF và EKF với ngưỡng χ^2 , các ma trận Q_t và R_t được giữ không đổi trong suốt thời gian thực nghiệm, tức là $Q_t = Q_{t0}$ và $R_t = R_{t0}$. Ngược lại, AEKF liên tục điều chỉnh các ma trận này bằng cách sử dụng các hệ số tỷ lệ α_Q và α_R . Để tìm được giá trị phù hợp, α_Q và α_R được lựa chọn trong khoảng [0, 1], với giá trị lớn hơn 1 thì sẽ gây ra tính mất ổn định. Do đó sau khi xác định được khoảng giá trị sẽ nằm trong khoảng [0:1], sử dụng phương pháp Grid Search. Grid Search là phương pháp thử nghiệm toàn bộ các kết hợp có thể của các tham số trong một lưới xác định trước sau đó chọn ra tổ hợp có hiệu suất tốt nhất.

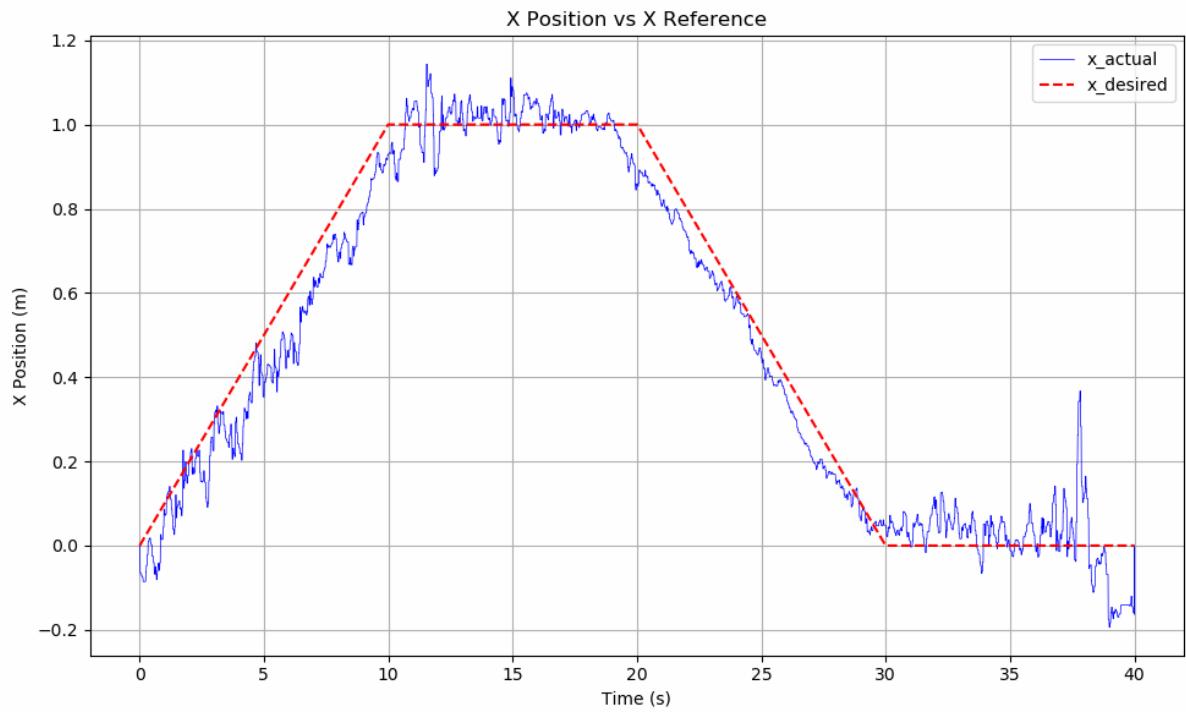
Cấu hình tối ưu để bám theo quỹ đạo tròn được xác định là $\{\alpha_Q, \alpha_R\} = \{0.93, 1\}$. Đối với phương pháp EKF với ngưỡng χ^2 , giá trị ngưỡng được đặt là 0.03.

Dưới đây là kết quả thực nghiệm với quỹ đạo hình vuông sử dụng 3 bộ ước lượng EKF, AEKF, EKF chi bình phương:

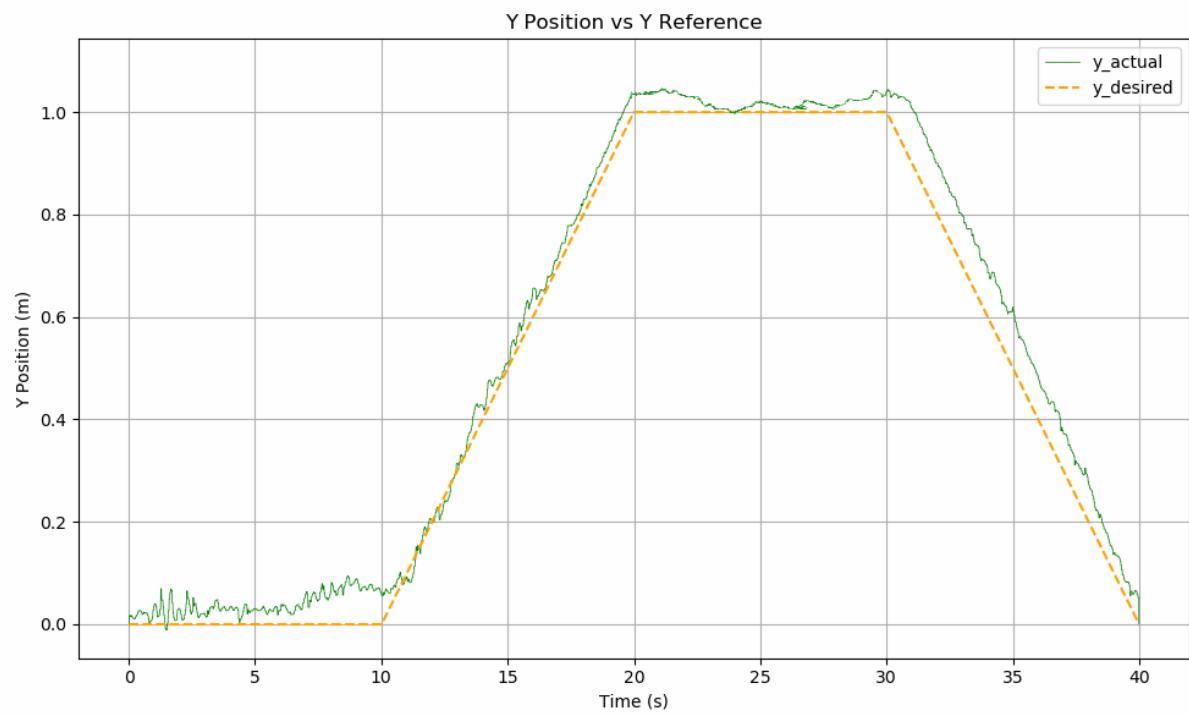
- Bộ uớc lượng EKF



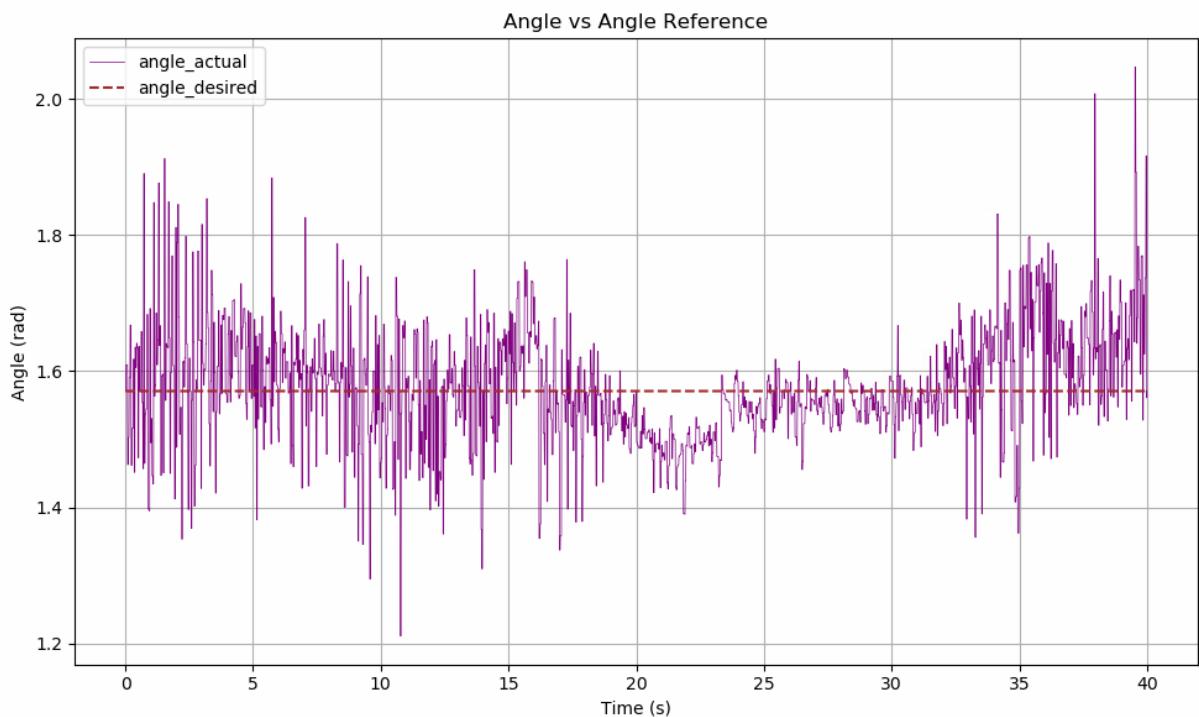
Hình 5.21: Quỹ đạo robot bám quỹ đạo hình vuông với EKF



Hình 5.22: Đáp ứng robot quỹ đạo hình vuông theo trục X với EKF

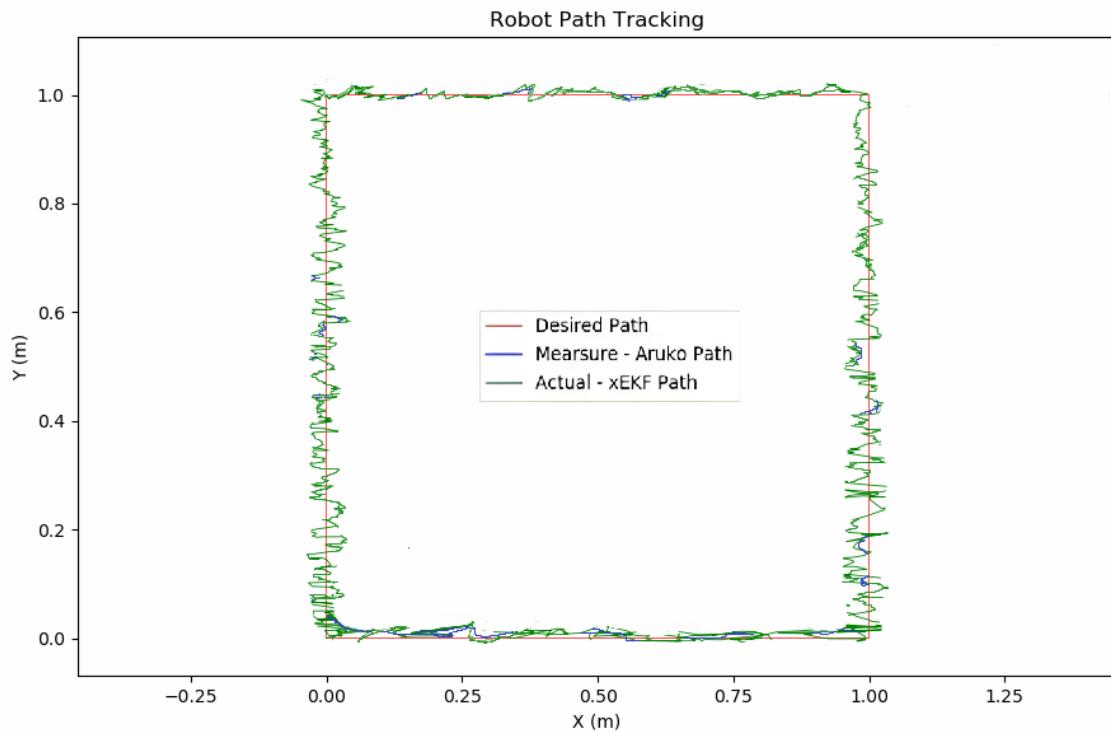


Hình 5.23: Đáp ứng robot quỹ đạo hình vuông theo trục Y với EKF

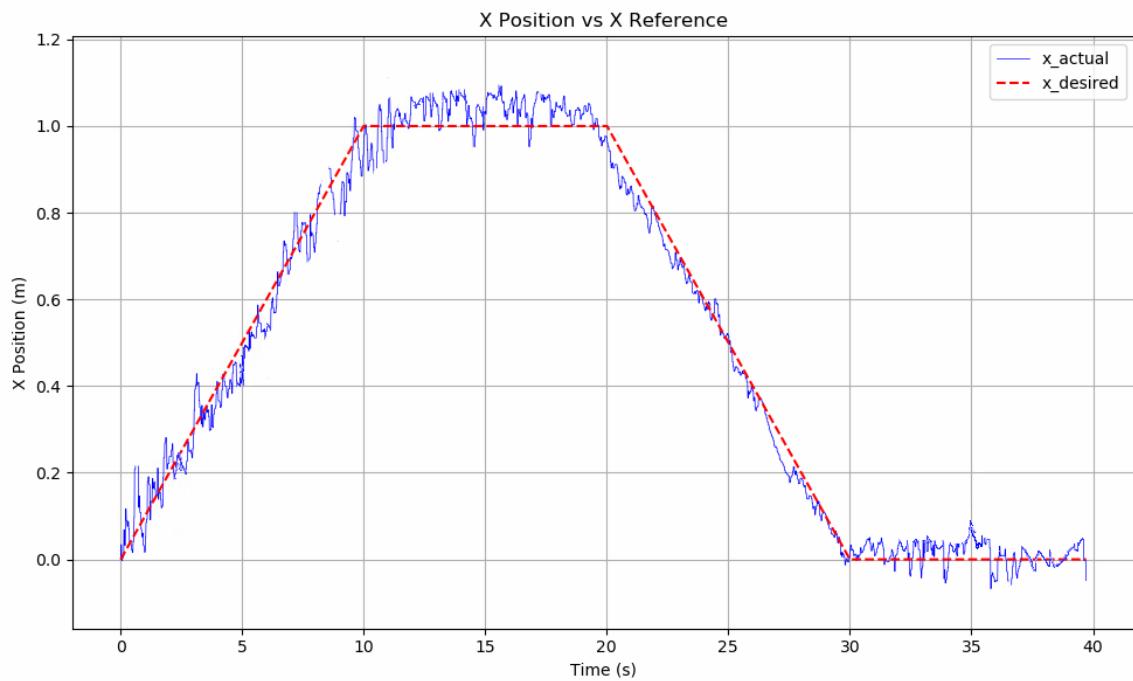


Hình 5.24: Đáp ứng robot quỹ đạo hình vuông theo góc yaw với EKF

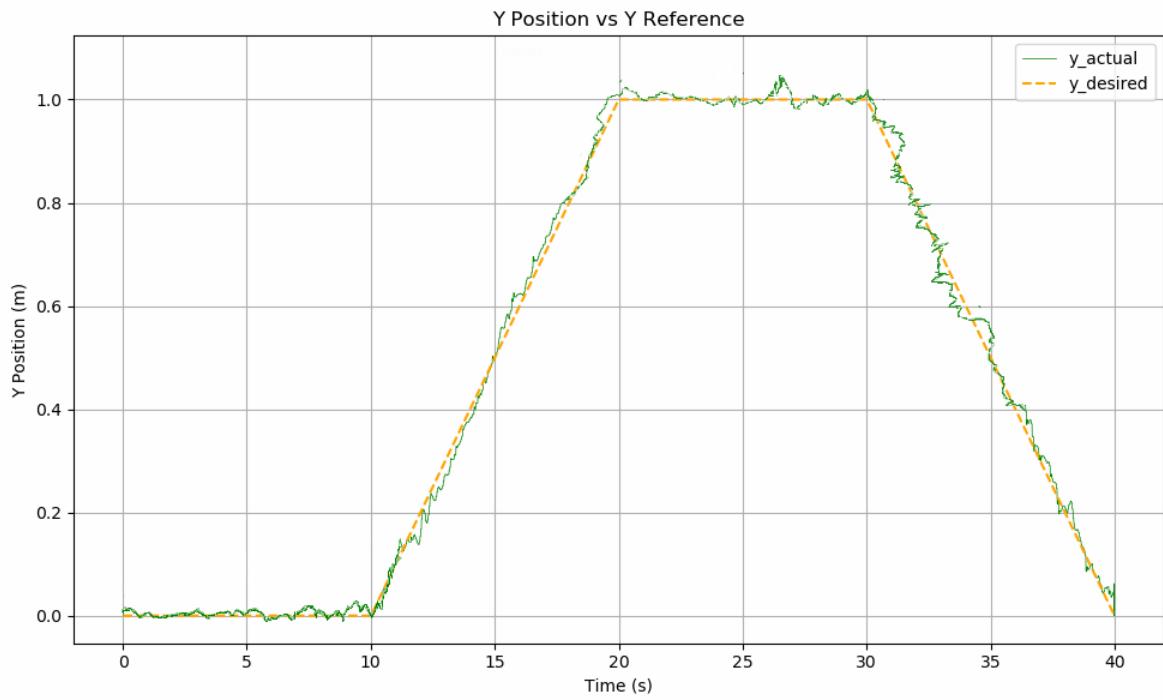
- Bộ uớc lượng AEKF



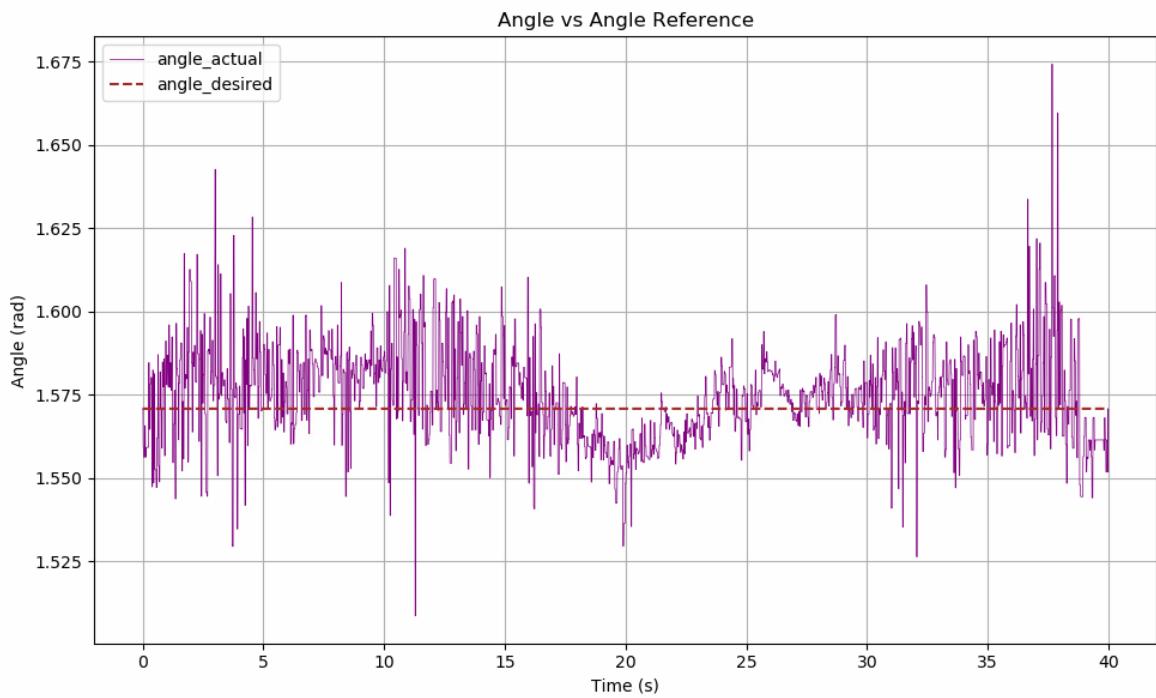
Hình 5.25: Quỹ đạo robot bám quỹ đạo hình vuông với AEKF



Hình 5.26: Dáp ứng robot quỹ đạo hình vuông theo trục X với AEKF

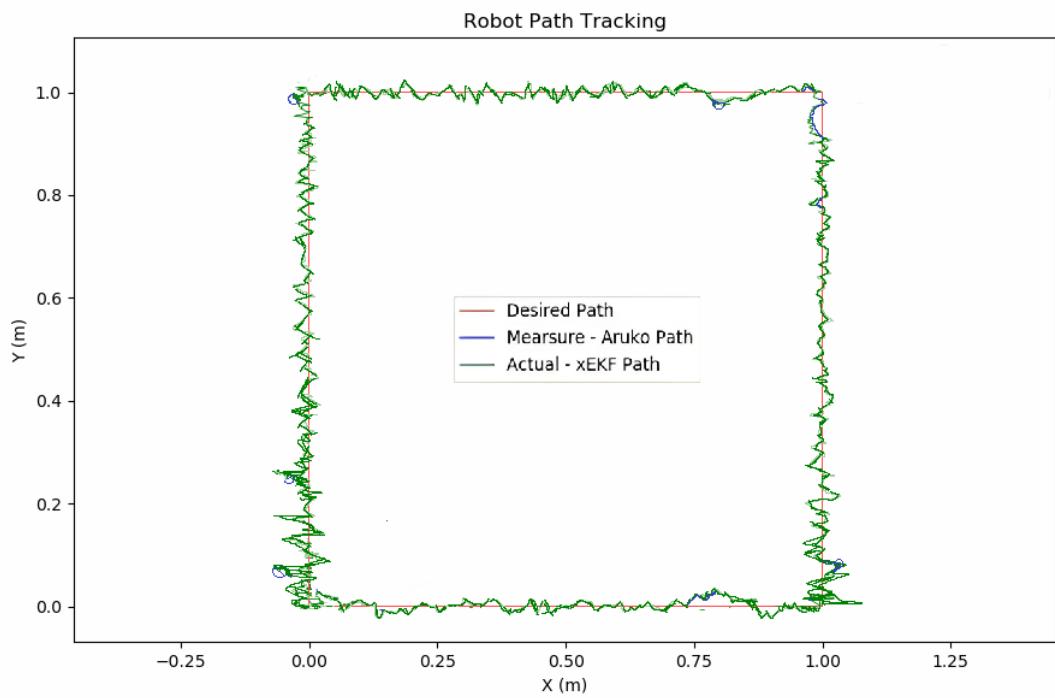


Hình 5.27: Dáp ứng robot quỹ đạo hình vuông theo trục Y với AEKF

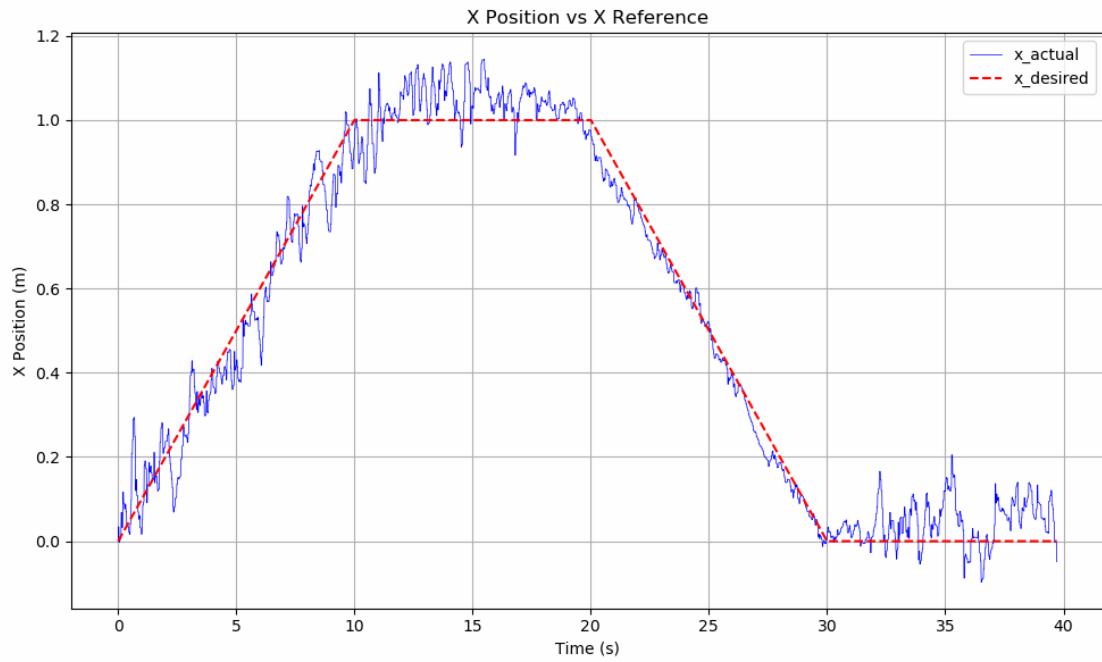


Hình 5.28: Dáp ứng robot quỹ đạo hình vuông theo góc yaw với AEKF

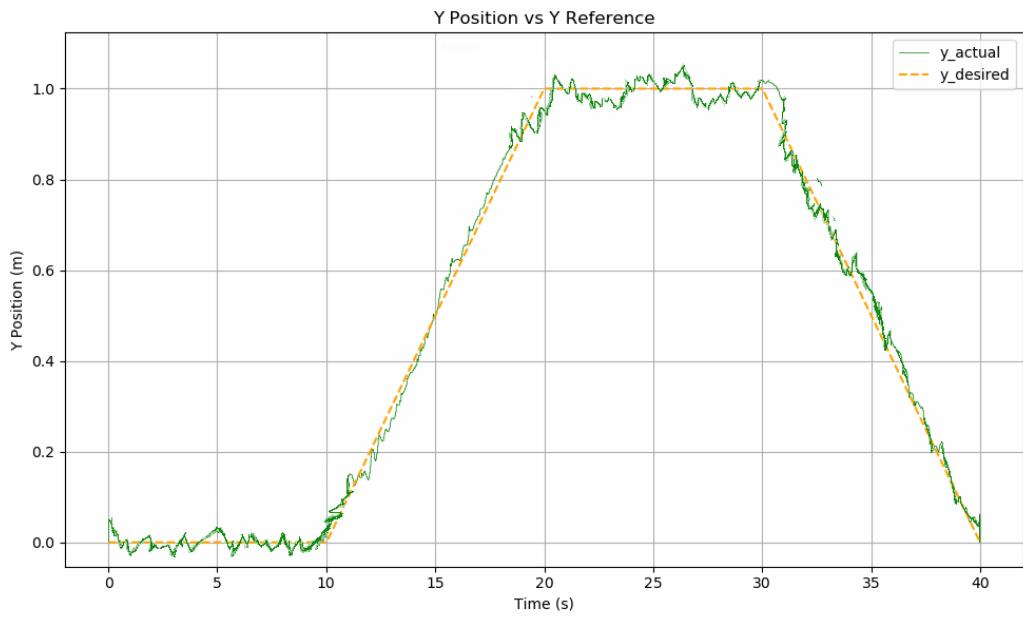
- Bộ uớc lượng EKF chi bình phương



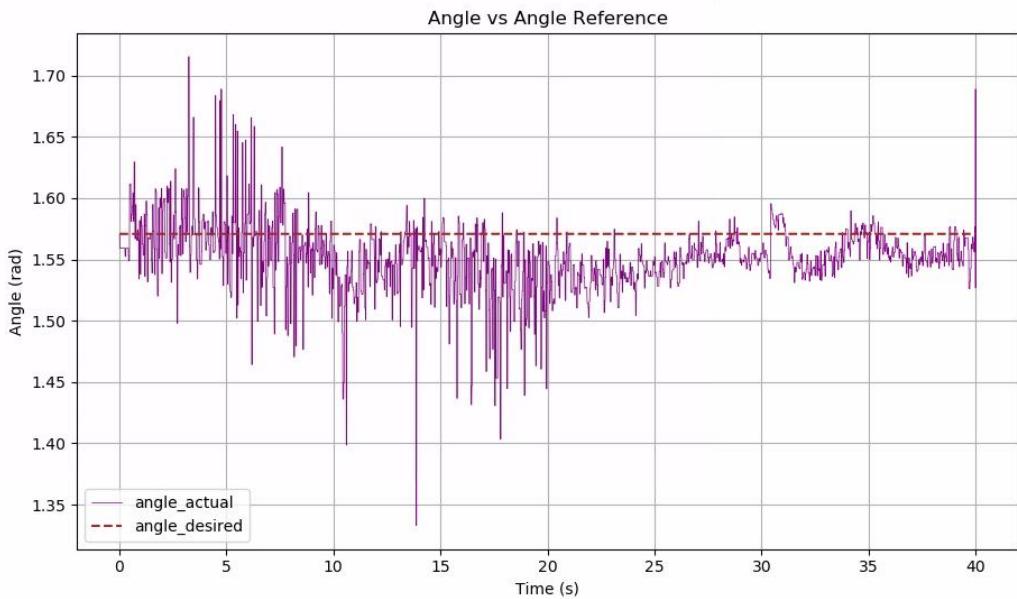
Hình 5.29: Quỹ đạo robot bám quỹ đạo hình vuông với EKF chi bình phương



Hình 5.30: Đáp ứng robot quỹ đạo hình vuông theo trục X với EKF chi bình phương



Hình 5.31: Đáp ứng robot quỹ đạo hình vuông theo trục Y với EKF chi bình phương



Hình 5.32: Đáp ứng robot quỹ đạo hình vuông theo góc yaw với EKF chi bình phương

Nhận xét:

- Robot đã thành công trong việc bám quỹ đạo hình vuông bán kính 1m với cả 3 bộ ước lượng EKF, AEKF, EKF chi bình phương.
- Trong đó có thể thấy AEKF và EKF chi bình phương đã cải thiện đáng kể khả năng bám quỹ đạo và định vị cho robot.
- AEKF cho kết quả sai số tốt nhất trong 3 bộ ước lượng chạy với quỹ đạo vuông

Bảng 5.5: Kết quả sai số của quỹ đạo hình vuông cho 3 bộ ước lượng

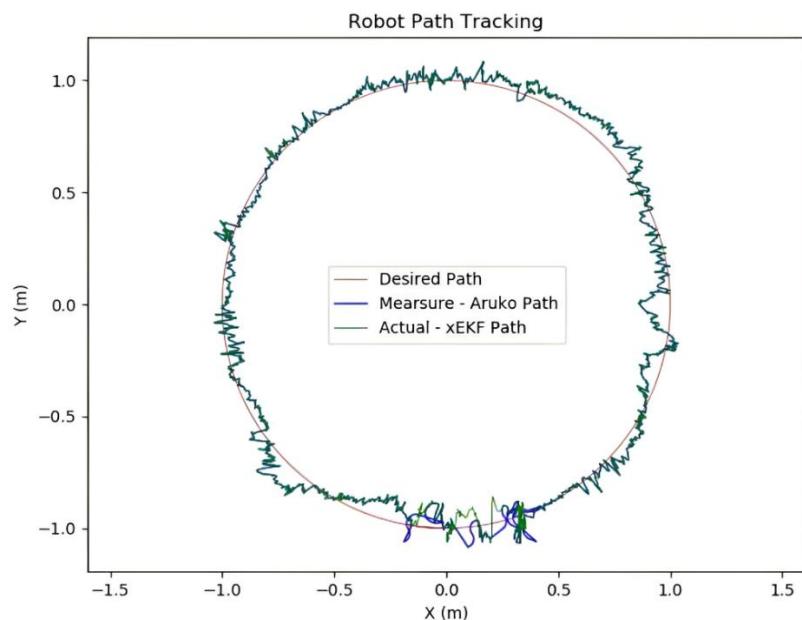
Bộ ước lượng thực hiện	Thời gian chạy (s)	Sai số x trung bình (m)	Sai số y trung bình (m)	Sai số yaw trung bình (rad)
EKF	40	0.0769	0.0908	0.25
AEKF	40	0.0343	0.0376	0.0657
EKF chi bình	40	0.0476	0.0457	0.0932

5.3.5. Quỹ đạo hình tròn

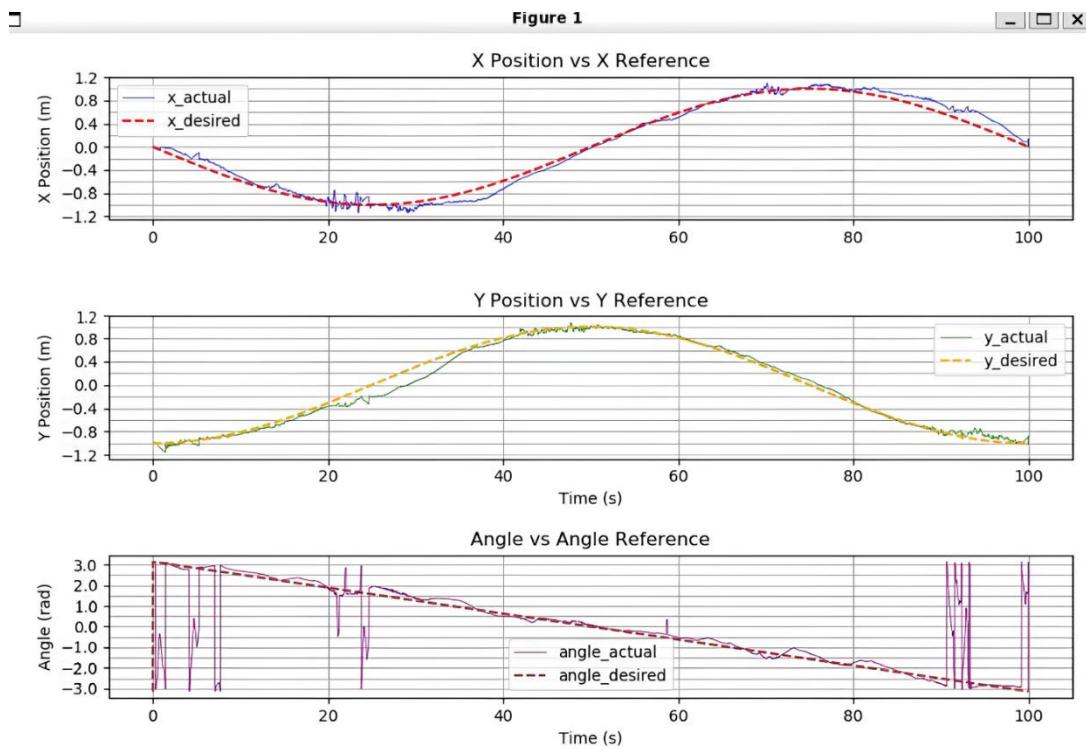
Cuối cùng ta sẽ xét khả năng định vị và bám quỹ đạo của hệ thống với quỹ đạo hình tròn. Với quỹ đạo này ta sẽ sử dụng setup 3 aruco tham chiếu như đã trình bày ở mục 5.1 setup 1.

Dưới đây là kết quả thực nghiệm với quỹ đạo hình vuông sử dụng 3 bộ ước lượng EKF, AEKF, EKF chi bình phương:

- Bộ ước lượng EKF



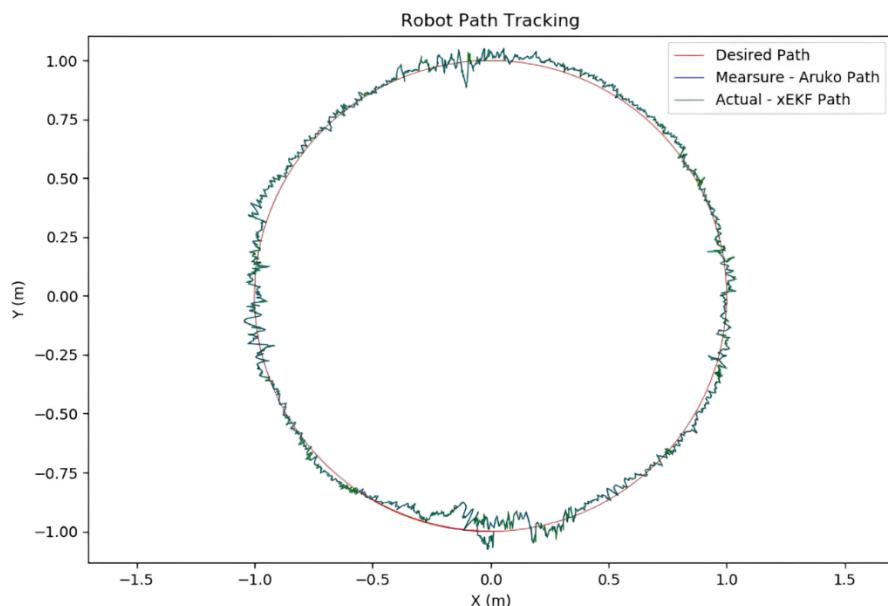
Hình 5.33: Quỹ đạo robot bám quỹ đạo hình tròn với EKF



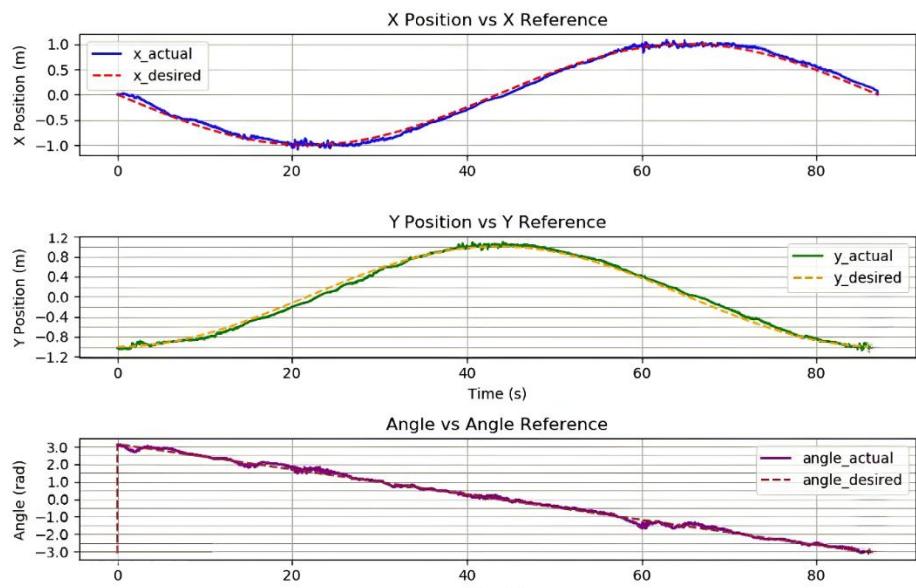
Hình 5.34: Đáp ứng robot quỹ đạo tròn theo trục x, y, yaw với EKF

Ở phương pháp EKF, robot về cơ bản vẫn bám theo quỹ đạo tham khảo, tuy nhiên sai số ước lượng vẫn tương đối đáng kể, đặc biệt ở góc hướng (yaw), do nhiều từ cảm biến chưa được khắc phục hoàn toàn.

- Bộ uớc lượng AEKF



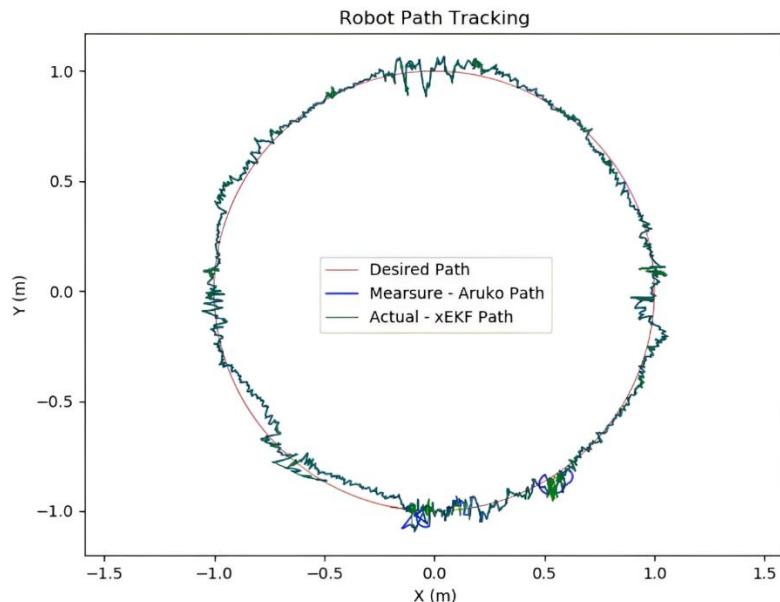
Hình 5.35: Quỹ đạo robot bám quỹ đạo hình tròn với AEKF



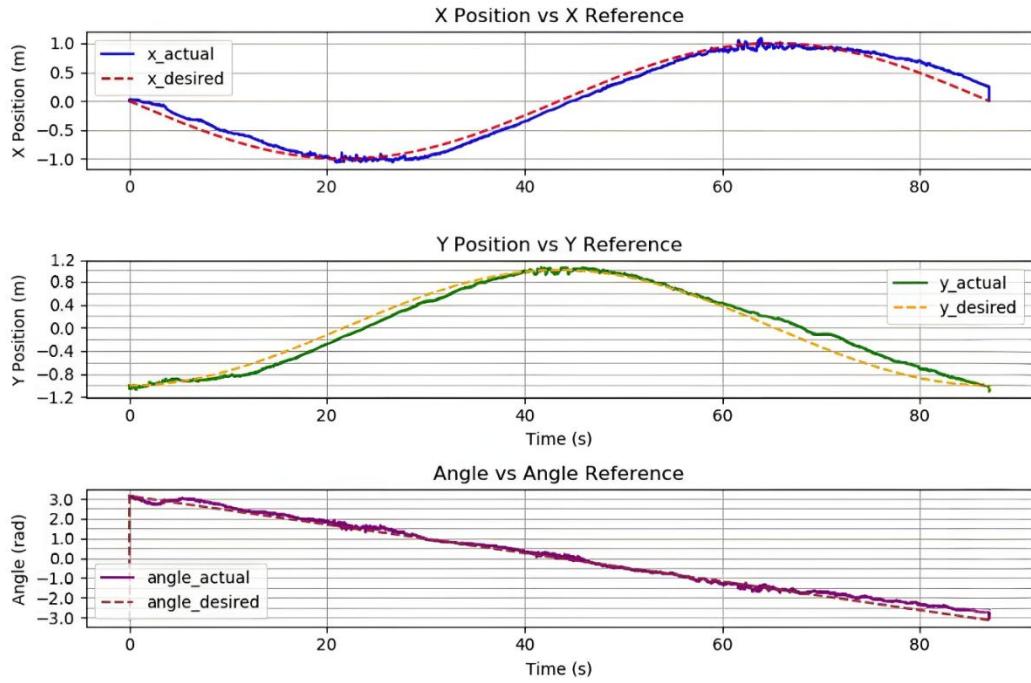
Hình 5.36: Đáp ứng robot quỹ đạo tròn theo trục x, y, yaw với AEKF

Nhận xét: Khả năng bám quỹ đạo được cải thiện đáng kể nhờ vào việc điều chỉnh động các ma trận Q_t và R_t , giúp bộ ước lượng thích ứng tốt hơn với nhiều từ môi trường. Quỹ đạo ước lượng bám theo quỹ đạo tham khảo một cách chính xác và mịn màng hơn so với EKF, qua đó phản ánh được tính ổn định và độ chính xác của hệ thống đã được cải thiện. Kết quả này minh chứng được hiệu quả của việc tinh chỉnh hiệp phương sai một cách động trong việc nâng cao chất lượng ước lượng

- Bộ ước lượng EKF chi bình phương



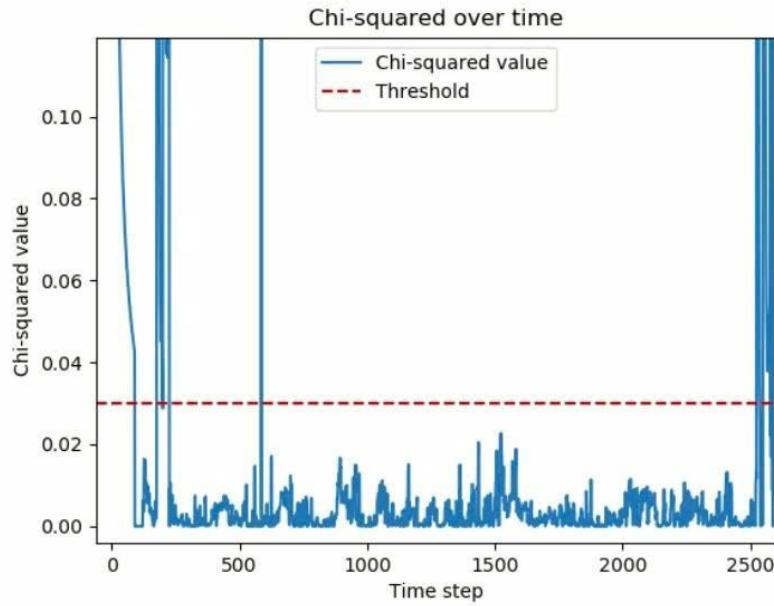
Hình 5.37: Quỹ đạo robot bám quỹ đạo hình tròn với EKF chi bình phương



Hình 5.38: Đáp ứng robot quỹ đạo tròn theo trục x, y, yaw với EKF chi bình

Nhận xét:

- Cả AEKF và EKF với ngưỡng χ^2 đều phù hợp để áp dụng cho mô hình robot mecanum, bởi chúng đem lại độ chính xác cao hơn và khả năng hoạt động robust hơn trước các nhiễu loạn từ môi trường.
- Cuối cùng, Hình 5.39 nhấn mạnh tầm quan trọng của việc lựa chọn một ngưỡng χ^2 phù hợp, bởi thông số này đóng vai trò then chốt trong việc loại bỏ các phép đo sai lệch vượt khỏi giới hạn chấp nhận được. Việc này trực tiếp giúp hệ thống điều khiển hoạt động mịn màng hơn.
 - Kết hợp với Hình 5.37, nhận thấy lúc ban đầu robot gặp một số nhiễu đo lường do mặt sàn không bằng phẳng, khiến giá trị χ^2 vượt quá ngưỡng. Tuy nhiên, sau khi được hiệu chỉnh, robot đã bám theo quỹ đạo một cách chính xác hơn.
 - Vì thế việc áp dụng ngưỡng χ^2 trong EKF đóng vai trò quan trọng trong việc kiểm soát ảnh hưởng của các phép đo sai, qua đó nâng cao độ chính xác của hệ thống định vị trong nhà.



Hình 5.39: Giá trị χ^2 và χ^2 threshold

5.3.6. Phương pháp đánh giá sai số

Để định lượng hiệu suất của từng phương pháp ước lượng, ta tiến hành tính toán sai số giữa giá trị ước lượng và giá trị thực của vị trí (x,y) và góc θ của robot tại mỗi thời điểm.

Giá trị thực (ground truth) được xác định dựa trên vị trí toàn cục đã biết của các marker ArUco được bố trí cố định trong môi trường. Ngược lại, giá trị ước lượng được thu nhận từ các bộ ước lượng tương ứng (ví dụ: bộ ước lượng Kalman, bộ ước lượng hạt, v.v.).

Tại mỗi thời điểm iii, ký hiệu (x_i, y_i, θ_i) là trạng thái thực của robot, trong khi $(\hat{x}_i, \hat{y}_i, \hat{\theta}_i)$ là trạng thái được ước lượng. Sai số tức thời được tính bằng trị tuyệt đối của hiệu giữa giá trị ước lượng và giá trị thực:

$$e_x(i) = |\hat{x}_i - x_i|, e_y(i) = |\hat{y}_i - y_i|, e_\theta(i) = |\hat{\theta}_i - \theta_i|$$

Từ các sai số tức thời này, chúng tôi tính toán hai đại lượng đặc trưng để đánh giá độ chính xác của hệ thống:

- Sai số trung bình (Mean Error): là trung bình cộng của các sai số tuyệt đối trong toàn bộ chuỗi thời gian, được xác định như sau:

- Sai số trung bình theo x: $x = \frac{1}{N} \sum_{i=1}^N e_x(i)$

- Sai số trung bình theo y : $y = \frac{1}{N} \sum_{i=1}^N e_y(i)$
- Sai số trung bình theo θ : $\theta = \frac{1}{N} \sum_{i=1}^N e_\theta(i)$

Bảng 5.6: Kết quả sai số của quỹ đạo hình tròn cho 3 bộ ước lượng

Bộ ước lượng thực hiện	Thời gian chạy (s)	Sai số x trung bình (m)	Sai số y trung bình (m)	Sai số theta trung bình (rad)
EKF	98	0.12	0.134	0.167
AEKF	96	0.047	0.043	0.053
EKF chi bình	92	0.056	0.061	0.062

CHƯƠNG 6: KẾT LUẬN

6.1. Kết quả đạt được

Trong phạm vi Đồ án Tốt nghiệp này, đồ án đã đạt được một số thành tựu đáng kể trong việc xây dựng một hệ thống định vị và điều khiển cho robot Mecanum.

- Đầu tiên, đồ án đã xây dựng thành công bộ điều khiển Backstepping kết hợp điều khiển trượt nhằm mục đích nâng cao độ chính xác và tính ổn định của việc điều khiển chuyển động của robot Mecanum.

- Đã mô phỏng robot Mecanum trong môi trường Matlab/Simulink để đánh giá hiệu suất bộ điều khiển. Một tập hợp các quỹ đạo tham chiếu đã được sử dụng để kiểm tra và đánh giá mức độ hiệu quả của bộ điều khiển trong việc bám theo quỹ đạo mong muốn cũng như duy trì tính ổn định động học của robot mecanum. Các kết quả mô phỏng thu được đóng vai trò như một nguồn thông tin quan trọng, phản ánh rõ ràng hiệu suất tương đối và đặc tính hoạt động bộ điều khiển được triển khai.

- Đã xây dựng được khung định vị bằng camera kết hợp dữ liệu từ cảm biến encoder và IMU thông qua các bộ ước lượng. Mang đến một giải pháp hiệu quả cho định vị robot trong nhà.

- Một mô hình robot mecanum thực nghiệm đã được xây dựng và các thí nghiệm đã được tiến hành với các bộ ước lượng khác nhau để xác nhận hiệu suất của hệ thống điều khiển và định vị. Kết quả của các thí nghiệm cho thấy sai số trung bình cho các quỹ đạo khá tốt cho thấy hệ thống đã hoạt động tương đối ổn định với các quỹ đạo khác nhau.

- Một giao diện người dùng trực quan (GUI) đã được phát triển bằng thư viện PyQt5 nhằm nâng cao khả năng tương tác giữa người dùng và hệ thống robot Mecanum. Giao diện này cho phép người vận hành quản lý robot cũng như theo dõi hiệu suất hoạt động của hệ thống trong thời gian thực thông qua một môi trường thân thiện và dễ sử dụng. Nhờ đó, tính khả dụng và mức độ tiện lợi của hệ thống điều khiển robot Mecanum đã được cải thiện đáng kể, góp phần hỗ trợ quá trình giám sát và vận hành hiệu quả hơn.

6.2. Những hạn chế

- Độ chính xác và độ tin cậy của hệ thống định vị phụ thuộc chặt chẽ vào hiệu năng hoạt động của camera và cảm biến quán tính (IMU). Tuy nhiên, cả hai loại cảm biến này đều tồn tại những hạn chế cố hữu, có thể ảnh hưởng đến chất lượng định vị tổng thể của

hệ thống FMWMR. Đối với camera, tín hiệu thu được rất dễ bị tác động bởi các yếu tố môi trường như sự thay đổi điều kiện chiếu sáng, hiện tượng che khuất hay nhiễu ảnh. Trong điều kiện ánh sáng yếu hoặc xảy ra hiện tượng lóa sáng, khả năng phát hiện và theo dõi đặc trưng hình ảnh của camera có thể bị suy giảm đáng kể, làm giảm độ chính xác trong quá trình định vị. Trong khi đó, cảm biến IMU thường gặp vấn đề về sai số tích lũy theo thời gian, đặc biệt là hiện tượng trôi và sai số bias. Các số đo từ gia tốc kế và con quay hồi chuyển có thể tích tụ sai lệch, dẫn đến hiện tượng trôi dần về vị trí và góc hướng, gây ảnh hưởng đến độ ổn định lâu dài của hệ thống định vị.

- Bộ điều khiển Backstepping kết hợp điều khiển trượt tuy ổn định nhưng sai số xác lập vẫn còn khá cao.
- Mô hình robot cải tiến với nhíp bánh xe giúp xe có thể đi trên những mặt đường gồ ghề, tuy nhiên điều này lại ảnh hưởng đến khả năng định vị của camera trên xe.
- Mạch thiết kế cho STM32F411 và Driver board còn nhiều do chưa xử lý tốt khi hàn và tính chất của board lõi.

6.3. Hướng phát triển

- Cải thiện bộ điều khiển để giảm sai số xác lập về gần bằng không để cải thiện kết quả robot khi thực nghiệm.
- Thiết kế mạch PCB cho mạch kết nối giữa STM32F411 và driver để giảm nhiễu cho hệ thống.
- Tích hợp thêm một số bộ ước lượng nâng cao để cải thiện khả năng định vị vị trí và góc cho phương pháp aruco marker.

TÀI LIỆU THAM KHẢO

- [1] X. Zhong, Y. Zhou, and H. Liu, “Design and recognition of artificial landmarks for reliable indoor self-localization of mobile robots,” *International Journal of Advanced Robotic Systems*, vol. 14, no. 1, p. 1729881417693489, 2017.
- [2] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM,” *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021
- [3] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, “A flexible and scalable SLAM system with full 3D motion estimation,” in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, 2011, pp. 155–160
- [4] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014
- [5] E. Olson, “AprilTag: A robust and flexible visual fiducial system,” in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 3400–3407.
- [6] B.-S. Cho, W.-s. Moon, W.-J. Seo, and K.-R. Baek, “A dead reckoning localization system for mobile robots using inertial sensors and wheel revolution encoding,” *Journal of Mechanical Science and Technology*, vol. 25, no. 11, pp. 2907–2917, Nov 2011.
- [7] P. Nazemzadeh, D. Fontanelli, D. Macii, and L. Palopoli, “Indoor localization of mobile robots through QR code detection and dead reckoning data fusion,” *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 6, pp. 2588–2599, 2017
- [8] N. Kayhani, A. Heins, W. Zhao, M. Nahangi, B. McCabe, and A. P. Schoellig, “Improved tag-based indoor localization of UAVs using extended Kalman filter,” in *Proceedings of the ISARC. International Symposium on Automation and Robotics in Construction, Banff, AB, Canada*, 2019, pp. 21–24

Proceedings of the ISARC. International Symposium on Automation and Robotics in Construction, Banff, AB, Canada, 2019, pp. 21–24.

- [9] A. Skobeleva, V. Ugrinovskii, and I. Petersen, “Extended Kalman filter for indoor and outdoor localization of a wheeled mobile robot,” in *2016 Australian Control Conference (AuCC)*, 2016, pp. 212–216
- [10] T. L. Song and J. L. Speyer, “The modified gain extended Kalman filter and parameter identification in linear systems,” *Automatica*, vol. 22, no. 1, pp. 59–75, 1986
- [11] S. Akhlaghi, N. Zhou, and Z. Huang, “Adaptive adjustment of noise covariance in Kalman filter for dynamic state estimation,” in *2017 IEEE Power and Energy Society General Meeting*, 2017.
- [12] B. Brumback and M. Srinath, “A Chi-square test for fault-detection in Kalman filters,” *IEEE Transactions on Automatic Control*, vol. 32, no. 6, pp. 552–554, 1987.
- [13] X. Li, S. Patel, D. Stronzek-Pfeifer and C. Büskens, "Indoor Localization for an Autonomous Model Car: A Marker-Based Multi-Sensor Fusion Framework," 2023 IEEE 19th International Conference on Automation Science and Engineering (CASE), Auckland, New Zealand, 2023, pp. 1-8, doi: 10.1109/CASE56687.2023.10260399.
- [14] Ren, J.; Wu, T.; Zhou, X.; Yang, C.; Sun, J.; Li, M.; Jiang, H.; Zhang, A. SLAM, “Path Planning Algorithm and Application Research of an Indoor Substation Wheeled Robot Navigation System”, *Electronics* 2022, 11, 1838.
<https://doi.org/10.3390/electronics11121838>.
- [15] Hà Huy Giáp, Nguyễn Quang Đại, “RESEARCH AND DESIGN APPLICATION BACKSTEPPING CONTROLLER FOR MOBILE ROBOT WITH FOUR MECANUM-WHEEL”, *Journal of SCIENCE & TECHNOLOGY*, Vol. 59 - No. 3, June 2023
- [16] Xingyang Lu, Xiangying Zhang, Guoliang Zhang, Songmin Jia, “Design of Adaptive Sliding Mode Controller for Four-Mecanum Wheel Mobile Robot”, 37th Chinese Control Conference

