

ĐẠI HỌC QUỐC GIA TP. HCM  
TRƯỜNG ĐẠI HỌC BÁCH KHOA

-----  
NGÔ ĐĂNG HIỀN

**CẢI THIỆN ĐỘ CHÍNH XÁC HỆ THỐNG ĐỊNH VỊ CHO ROBOT DI  
ĐỘNG TỰ ĐỘNG DỰA TRÊN PHƯƠNG PHÁP HỢP NHẤT  
NHIỀU CẢM BIẾN**

**IMPROVING THE ACCURACY OF THE AUTONOMOUS MOBILE  
ROBOT LOCALIZATION SYSTEMS BASED ON THE MULTIPLE  
SENSOR FUSION METHODS**

Chuyên ngành: Kỹ Thuật Điều Khiển Và Tự Động Hóa

Mã số: 60520216

LUẬN VĂN THẠC SĨ

TP. HỒ CHÍ MINH, tháng 09 năm 2020

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

Cán bộ hướng dẫn khoa học: **TS. Nguyễn Trọng Tài**

Cán bộ chấm nhận xét 1: **PGS.TS Huỳnh Thái Hoàng**

Cán bộ chấm nhận xét 2: **PGS.TS Nguyễn Tấn Lũy**

Luận văn thạc sĩ được bảo vệ tại Trường Đại học Bách Khoa, ĐHQG  
Tp. Hồ Chí Minh, ngày 04 tháng 09 năm 2020.

Thành phần Hội đồng đánh giá luận văn thạc sĩ gồm:

1. Chủ tịch: PGS.TS Nguyễn Thanh Phương (ĐH KTCN TpHCM)
2. Thư ký: TS. Trần Ngọc Huy
3. Phản biện 1: PGS.TS Huỳnh Thái Hoàng
4. Phản biện 2: PGS.TS Nguyễn Tấn Lũy (ĐH Công nghiệp TpHCM)
5. Ủy viên: TS. Nguyễn Hoàng Giáp

Xác nhận của Chủ tịch Hội đồng đánh giá luận văn và Trưởng Khoa  
quản lý chuyên ngành sau khi luận văn đã được sửa chữa (nếu có):

**CHỦ TỊCH HỘI ĐỒNG**

**TRƯỞNG KHOA**

## NHIỆM VỤ LUẬN VĂN THẠC SĨ

Họ tên học viên: Ngô Đăng Hiền MSHV: 1670782  
Ngày, tháng, năm sinh: 20/08/1988 Nơi sinh: Đăk Lăk  
Chuyên ngành: Kỹ Thuật Điều Khiển Và Tự Động Hóa Mã số: 60520216

### I. TÊN ĐỀ TÀI:

CẢI THIỆN ĐỘ CHÍNH XÁC HỆ THỐNG ĐỊNH VỊ CHO ROBOT DI ĐỘNG TỰ  
ĐỘNG DỰA TRÊN PHƯƠNG PHÁP HỢP NHẤT NHIỀU CẢM BIẾN

IMPROVING THE ACCURACY OF THE AUTONOMOUS MOBILE ROBOT  
LOCALIZATION SYSTEMS BASED ON THE MULTIPLE SENSOR FUSION  
METHODS

### II. NHIỆM VỤ VÀ NỘI DUNG

- Nghiên cứu, xây dựng, triển khai phương pháp tổng hợp đa cảm biến dựa trên bộ lọc Kalman mở rộng (EKF), thực nghiệm trên mô hình robot di động tự hành với các loại cảm biến như: Encoder, IMU, UWB, Lidar.

III. NGÀY GIAO NHIỆM VỤ: 10/02/2020

IV. NGÀY HOÀN THÀNH NHIỆM VỤ: 07/08/2020

V. CÁN BỘ HƯỚNG DẪN: TS. Nguyễn Trọng Tài

CÁN BỘ HƯỚNG DẪN  
(Họ tên và chữ ký)

Tp. HCM, ngày tháng 09 năm 2020  
CHỦ NHIỆM BỘ MÔN ĐÀO TẠO  
(Họ tên và chữ ký)

TRƯỜNG KHOA  
(Họ tên và chữ ký)

## LỜI CẢM ƠN

Để hoàn thành luận văn này, tôi xin gửi lời cảm ơn tới:

Quý thầy, cô trong Bộ môn Điều Khiển và Tự Động Hóa, trường Đại học Bách Khoa Tp. Hồ Chí Minh, những người đã hết lòng truyền đạt những kiến thức quý báu cho tôi trong suốt thời gian học tập tại trường. Đặc biệt, tôi xin gửi lời cảm ơn chân thành và trân trọng nhất đến **TS. Nguyễn Trọng Tài**, đã tận tình hướng dẫn, giúp đỡ tôi trong suốt quá trình nghiên cứu và thực hiện đề tài. Ngoài ra, tôi xin được gửi lời cảm ơn chân thành nhất đến thầy **PGS.TS Huỳnh Thái Hoàng**, thầy **TS. Nguyễn Vĩnh Hảo**, thầy **TS. Trương Đình Châu**, thầy **TS. Ngô Đình Trí**, đã giúp đỡ, định hướng lại cho tôi toàn bộ về kiến thức nền, khả năng nghiên cứu khoa học, khả năng khai thác những thiết bị công nghiệp hiện đại, những kiến thức vô cùng quý giá mà thời đại học tôi chưa từng được trang bị, nó sẽ là hành trang cho tôi trên con đường học tập, công tác và nghiên cứu sau này.

Tôi xin gửi lời cảm ơn chân thành đến tập thể khoa Vũ Khí Dưới Nước, Học Viện Hải Quân, là đơn vị nơi tôi công tác, đã luôn tạo điều kiện, giúp đỡ hết sức trong suốt quá trình tôi công tác tại đơn vị, cũng như học tập tại trường Đại học Bách Khoa.

Tôi xin gửi lời cảm ơn chân thành và sâu sắc nhất đến Vợ và con gái nhỏ, đã luôn là nguồn động viên to lớn nhất. Đồng thời, con cũng xin được gửi lời cảm ơn chân thành đến Bố Mẹ đã luôn đồng hành, động viên con trong suốt quá trình con học tập, công tác và phát triển.

Cuối cùng, tôi xin gửi lời cảm ơn đến tất cả các anh chị em lớp cao học Tự động hóa, khóa 2017, 2018 đã luôn giúp đỡ, hỗ trợ tôi trong suốt quá trình học tập lý thuyết tại trường. Đặc biệt gửi lời cảm ơn đến một người bạn, cũng là đồng nghiệp, bạn Hồ Sỹ Thông - 1670848, Khóa 2016, Lớp cao học Kỹ thuật điện tử, đã luôn giúp đỡ và đồng hành cùng nhau trong suốt thời gian cả hai học tập tại trường.

Xin chân thành cảm ơn !

Tp. HCM, ngày 20 tháng 08 năm 2020

**Ngô Đăng Hiền**

## TÓM TẮT LUẬN VĂN

Luận văn tập trung vào nghiên cứu và triển khai một giải pháp để nâng cao độ chính xác cho hệ thống định vị robot di động, bằng cách thức hợp nhất nhiều cảm biến với nhau thông qua bộ lọc Kalman mở rộng (EKF).

Hệ thống được mô hình hóa, đánh giá thử nghiệm trên mô phỏng simulink của matlab; python và được triển khai trên hệ thống máy tính nhúng Linux ở tầng điều khiển High Level và MCU STM32F405 ở tầng Low level.

Mô hình được thực nghiệm là Robot di động kiểu vi sai, được điều khiển bằng bộ điều khiển PID, sử dụng các cảm biến như Encoder, MPU9250, UWB, Lidar cho bài toán EKF Fusion. Ngoài ra cảm biến Lidar còn sử dụng cho điều hướng di chuyển, tránh vật cản.

## ABSTRACT

The project focuses on researching and implementing solutions to improve the accuracy of the positioning robot system, by pairing multiple sensors together through the extended Kalman filter (EKF).

The system is modeled, evaluated on the matlab's simulation of the link; python and it are implemented on a Linux embedded computer system on the controller upper layer and the STM32F405 MCU on the Low layer.

The model done is robot dynamic differential, controlled by PID control, using sensors such as Encoder, MPU9250, UWB, Lidar for the EKF Fusion problem. In addition, Lidar Sensor is also used to guide the way, avoid obstacles.

## **LỜI CAM ĐOAN**

Tôi xin cam đoan rằng luận văn: "**CẢI THIỆN ĐỘ CHÍNH XÁC HỆ THỐNG ĐỊNH VỊ CHO ROBOT DI ĐỘNG TỰ ĐỘNG DỰA TRÊN PHƯƠNG PHÁP HỢP NHẤT NHIỀU CẢM BIẾN**" là kết quả nghiên cứu do tôi thực hiện với sự hướng dẫn của thầy TS. Nguyễn Trọng Tài.

Những tài liệu tham khảo, nội dung trích dẫn được tôi trình bày chi tiết, cụ thể. Còn lại những nội dung khác chưa từng được công bố hoặc sử dụng để nhận bằng cấp ở những nơi khác.

Nếu phát hiện có bất kỳ sự gian lận nào, tôi xin hoàn toàn chịu trách nhiệm về nội dung luận văn của mình. Trường đại học Bách khoa Tp. Hồ Chí Minh không liên quan đến những vi phạm (nếu có) về tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện.

Tp. Hồ Chí Minh, ngày 20 tháng 08 năm 2020

HỌC VIÊN

**Ngô Đăng Hiền**

## MỤC LỤC

Chương 1 .....	1
TỔNG QUAN VỀ ĐỀ TÀI.....	1
1.1. Đặt vấn đề .....	1
1.2. Tính cấp thiết .....	2
1.3. Những nghiên cứu liên quan .....	3
1.4. Mục tiêu nghiên cứu.....	7
1.5. Đối tượng và phạm vi nghiên cứu.....	7
1.6. Cấu trúc của Luận văn .....	7
Chương 2 .....	9
CƠ SỞ LÝ THUYẾT.....	9
2.1. Mô hình toán của hệ thống Robot.....	9
2.1.1. Mô hình động học của Robot.....	9
2.1.2. Mô hình động học của động cơ điện một chiều (DC Motor) .....	11
2.2. Tổng hợp cảm biến bằng bộ lọc Kalman mở rộng EKF .....	13
2.2.1. Lý thuyết bộ lọc Kalman mở rộng (EKF).....	13
2.2.2. Ứng dụng bộ lọc EKF vào mô hình Robot.....	15
Chương 3 .....	20
XÂY DỰNG MÔ HÌNH VÀ BỘ ĐIỀU KHIỂN CHO ROBOT .....	20
3.1. Tổng quan về mô hình và thiết bị sử dụng trong Robot .....	20
3.2. Xây dựng bộ điều khiển cho robot.....	22
3.2.1. Nhận dạng hàm truyền động cơ.....	22
3.2.2. Lựa chọn tham số PID cho động cơ .....	22
3.2.3. Đánh giá đáp ứng của động cơ ở các điều kiện khác nhau .....	25
3.3. Tính toán góc yaw cho Robot từ IMU MPU9250 .....	27
3.4. Nhận giá trị vị trí XY cho Robot từ UWB DWM1001C .....	29
3.5. Tính toán vị trí XY và hướng Yaw cho Robot từ biến Lidar.....	30
CHƯƠNG 4 .....	32
MÔ PHỎNG HỆ THỐNG .....	32
4.1. Mô phỏng mô hình Robot .....	32
4.1.1. Sơ đồ Simulink của Robot.....	32

4.1.2. Đánh giá mô hình mô phỏng và mô hình thực tế của robot .....	33
4.2. Mô phỏng quá trình hợp nhất dữ liệu với bộ lọc EKF.....	37
4.2.1. Xây dựng chương trình mô phỏng .....	37
4.2.2. EKF với phép đo XY từ UWB Tag .....	38
4.2.3. EKF với phép đo góc yaw từ IMU .....	39
4.2.4. EKF với phép đo XY-Theta.....	42
4.3. Kết luận phần mô phỏng hệ thống.....	43
CHƯƠNG 5 .....	44
THỰC NGHIỆM TỔNG HỢP CẢM BIẾN CHO ĐỊNH VỊ ROBOT DI ĐỘNG .....	44
5.1. Kiểm tra nhiễu UWB và độ trôi góc yaw từ MPU9250 .....	44
5.1.1. Kiểm tra độ chính xác của UWB .....	44
5.1.2. Kiểm tra sự chính xác và độ trôi góc yaw của IMU .....	46
5.2. Triển khai bộ lọc EKF cho những quỹ đạo di chuyển định sẵn .....	48
5.2.1. Phương pháp thực hiện .....	48
5.2.2. Quỹ đạo di chuyển theo vòng tròn .....	48
5.2.3. Quỹ đạo di chuyển theo các góc vuông .....	54
5.2.4. Quỹ đạo di chuyển liên tục nhiều vòng .....	55
5.3. Xác định vị trí Robot với cảm biến Lidar .....	57
5.3.1. Vẽ bản đồ môi trường.....	57
5.4.2. Định vị AMCL cho Robot.....	60
5.4. Hợp nhất các cảm biến cho quá trình định vị Robot .....	61
5.4.1. Phương pháp thực hiện .....	61
5.4.2. Đánh giá kết quả định vị .....	63
5.5. Đánh giá kết quả định vị với bài toán điều hướng Robot.....	64
5.6. Nhận xét và kết luận.....	67
Chương 6 .....	69
KẾT LUẬN VÀ PHƯƠNG HƯỚNG PHÁT TRIỂN .....	69
6.1. Đánh giá kết quả thực hiện .....	69
6.2. Những mặt hạn chế .....	69
6.3. Phương hướng phát triển.....	70
TÀI LIỆU THAM KHẢO.....	71

PHỤ LỤC 1 .....	75
1.1. Danh sách file data dùng nhận dạng mô hình động cơ (ident) .....	75
1.2. Danh sách m-file và data thực hiện quá trình mô phỏng, đánh giá .....	75
1.3. Danh sách m-file và data thực hiện đánh giá odometry .....	75
1.4. Danh sách m-file và data cho đánh giá bộ lọc Madgwick .....	75
1.5. Dánh sách m-file và data cho đánh giá phương sai nhiễu UWB .....	76
1.6. Danh sách bagfile cho các quỹ đạo di chuyển khác nhau .....	76
1.7. Danh sách bagfile cho bộ lọc EKF ở các quỹ đạo di chuyển khác nhau ....	76
1.8. Danh sách bagfile cho EKF Fusion tổng hợp có sử dụng Lidar .....	77
1.9. Dánh sách bagfile quá trình điều hướng đến các điểm ABCD .....	77
PHỤ LỤC 2 .....	78
Code Matlab Function trên Matlab cho ModelRobot.slx .....	78

## DANH MỤC HÌNH VẼ

Hình 1. 1 EKF giữa odometry , IMU và UWB [3] .....	4
Hình 1. 2 Ba phương pháp tiếp cận khác nhau của EKF [8, p.4, fig.2] .....	5
Hình 1. 3 Kết quả đánh giá EKF với Encoder,GPS và Compass [1, table.2] .....	6
Hình 2. 1 Hệ quy chiếu toàn cục $O_GX_GY_G$ và hệ quy chiếu robot $O_RX_RY_R$ .....	9
Hình 2. 2 Sơ đồ điện của động cơ điện 1 chiều .....	11
Hình 2. 3 Sơ đồ thuật toán bộ lọc EKF .....	14
Hình 2. 4 Sơ đồ quá trình EKF Fusion cho Robot .....	19
Hình 3. 1 Hình ảnh thực tế phần bệ đỡ robot và mạch điều khiển .....	21
Hình 3. 2 Hình ảnh thực tế tổng quát của Robot.....	21
Hình 3. 3 Mô hình điều khiển với bộ điều khiển PIDz .....	23
Hình 3. 4 Đáp ứng của hệ thống khi thêm bộ điều khiển PID .....	24
Hình 3. 5 So sánh bộ PID rời rạc được triển khai dưới MCU với Simulink .....	24
Hình 3. 6 Đáp ứng tốc độ của các động cơ khi ở PWM = 150 .....	25
Hình 3. 7 Đáp ứng tốc độ của các động cơ khi ở PWM =200 .....	26
Hình 3. 8 Đáp ứng tốc độ của các động cơ khi PWM = 250.....	26
Hình 3. 9 Đáp ứng tốc độ của các động cơ khi ở PWM =500 .....	27
Hình 3. 10 Kết quả đánh giá 03 bộ lọc cho IMU [26,p.4, table 1].....	28
Hình 3. 11 Đánh giá RMSE cho 03 bộ lọc cho IMU [28, p.6, table 2] .....	28
Hình 3. 12 Module IMU MPU9250 .....	28
Hình 3. 13 Module UWB DWM1001C của Deacaware .....	29
Hình 3. 14 Cảm biến RPLidar A1M8.....	30
Hình 3. 15 Sơ đồ gói AMCL Localization của ROS .....	31
Hình 4. 1 Sơ đồ simulink mô hình hóa hệ thống.....	32
Hình 4. 2 Sơ đồ mô phỏng bên trong khối DC Motor System.....	32
Hình 4. 3 Quỹ đạo mô phỏng của robot với toolbox.....	33
Hình 4. 4 Vận tốc góc của 02 bánh xe khi quỹ đạo là hình vuông .....	33
Hình 4. 5 Đáp ứng của robot khi chạy theo một vòng tròn .....	34
Hình 4. 6 Đáp ứng của robot khi chạy theo một hình vuông.....	35

Hình 4. 7 Đáp ứng của robot khi chạy theo một hình số tám.....	35
Hình 4. 8 Biểu đồ phân phối nhiễu của $\nu$ [m/s] và $\omega$ [rad/s] .....	36
Hình 4. 9 Các phương thức kết hợp EKF cho mô phỏng thử nghiệm .....	38
Hình 4. 10 Kết quả hợp nhất Odometry và UWB.....	39
Hình 4. 11 EKF kết nối với góc yaw (có trôi) từ IMU.....	40
Hình 4. 12 EKF kết hợp với góc yaw (không bị trôi) từ IMU.....	41
Hình 4. 13 Kết quả EKF hợp nhất với phép đo đầy đủ $[x; y; \theta]$ .....	43
Hình 5. 1 Sơ đồ bố trí các thiết bị UWB làm Anchor .....	44
Hình 5. 2 UWB Tag Robot trong không gian UWB Anchor.....	44
Hình 5. 3 Giá trị đo được và biểu đồ phân phối nhiễu của UWB .....	45
Hình 5. 4 Tọa độ của Robot và phân phối đo lường của UWB .....	45
Hình 5. 5 Đánh giá sự trôi góc Yaw từ IMU .....	47
Hình 5. 6 Giá trị đo lường góc yaw khi quay trở ở các góc khác nhau .....	47
Hình 5. 7 So sánh quỹ đạo di chuyển khi sử dụng EKF với UWB .....	49
Hình 5. 8 So sánh quỹ đạo di chuyển khi sử dụng EKF với IMU .....	50
Hình 5. 9 Quỹ đạo di chuyển khi EKF với UWB và EKF với IMU.....	51
Hình 5. 10 Quỹ đạo di chuyển khi EKF với UWB+IMU.....	52
Hình 5. 11 Quỹ đạo di chuyển khi EKF với IMU và EKF với UWB+IMU .....	53
Hình 5. 12 EKF với phép đo UWB kết hợp IMU quỹ đạo hình chữ nhật.....	54
Hình 5. 13 EKF với phép đo UWB kết hợp IMU quỹ đạo hình chữ P .....	55
Hình 5. 14 EKF với quỹ đạo di chuyển theo hình tròn 03 lần .....	56
Hình 5. 15 EKF với quỹ đạo di chuyển theo hình chữ nhật 03 lần .....	57
Hình 5. 16 Hai phương pháp thử nghiệm lấy bản đồ.....	58
Hình 5. 17 Khu vực thử nghiệm Robot (sảnh Học viện quân y, Q.10).....	59
Hình 5. 18 Bản đồ môi trường đã được xây dựng.....	59
Hình 5. 19 Phương pháp cải tiến cho gói AMCL.....	60
Hình 5. 20 Quá trình ước lượng vị trí của AMCL .....	60
Hình 5. 21 Phương pháp tổng hợp EKF Fusion.....	61
Hình 5. 22 Sơ đồ bố trí 06 UWB Anchor trên bản đồ.....	62

Hình 5. 23 EKF với UWB+IMU khi quỹ đạo hình vuông .....	63
Hình 5. 24 EKF với UWB+IMU khi quỹ đạo hình chữ nhật.....	64
Hình 5. 25 Điều hướng Robot từ A đến C .....	65
Hình 5. 26 Điều hướng các đỉnh hình vuông ABCD .....	66
Hình 5. 27 Điều hướng các đỉnh hình chữ nhật ABCD .....	67

## DANH MỤC BẢNG

Bảng 1. 1 Một số bài báo liên quan đến Fusion sensor với Kalman .....	3
Bảng 2. 1 Ý nghĩa các tham số của robot .....	10
Bảng 2. 2 Ý nghĩa các tham số của động cơ DC .....	11
Bảng 3. 1 Thông số các thành phần sử dụng trong Robot.....	20
Bảng 3. 2 Hàm truyền đã nhận dạng của động cơ Trái và Phải.....	22
Bảng 3. 3. Hệ số PID tính toán và đáp ứng của động cơ.....	23
Bảng 4. 1 Phương sai nhiễu đo lường $v$ và $\omega$ của robot.....	36
Bảng 4. 2 Thông tin thử nghiệm mô phỏng EKF với UWB .....	38
Bảng 4. 3 So sánh RMSE giữa Dead Reckoning và EKF với UWB .....	39
Bảng 4. 4 Thông tin mô phỏng EKF với IMU.....	40
Bảng 4. 5 So sánh RMSE giữa Dead Reckoning và EKF với IMU .....	40
Bảng 4. 6 So sánh RMSE giữa Dead Reckoning và EKF với IMU (không trôi) ....	41
Bảng 4. 7 Thông tin mô phỏng EKF với phép đo XY-Yaw .....	42
Bảng 4. 8 Kết quả đánh giá RMSE khi sử dụng EKF với phép đo $[x; y; \theta]$ .....	43
Bảng 5. 1 Thông số R, Q thử nghiệm cho EKF với UWB .....	48
Bảng 5. 2 Thông số R, Q thử nghiệm cho EKF với IMU .....	49
Bảng 5. 3 Thông số R và Q khi EKF với UWB hợp nhất IMU .....	51
Bảng 5. 4 Ma trận <b>R</b> , <b>Q</b> cho thử nghiệm tổng hợp .....	61
Bảng 5. 5 Vị trí các điểm ABCD theo $[x; y; \theta]$ .....	64
Bảng 5. 6 Kết quả điều hướng Lần 1 từ A đến C.....	65
Bảng 5. 7 Sai số của EKF tại các đỉnh hình vuông ABCD.....	66
Bảng 5. 8 Sai số của EKF tại các đỉnh hình chữ nhật ABCD .....	67

## DANH MỤC CHỮ VIẾT TẮT

AGV	Autonomous Guided Vehicles (Xe vận tải tự động)
AHRS	Attitude and Heading Reference System
ASYNC	Asynchronous (không đồng bộ)
IMU	Inertial Measurement Unit
LIDAR	Light Detection And Ranging
MCU	Micro Controller Unit
PID	Proportional Integral Derivative Controller
UART	Universal Asynchronous Receiver-Transmitter
EKF	Extended Kalman Filter (Bộ lọc Kalman mở rộng)
GPS	Global Positioning System (Hệ định vị toàn cầu)
Gyro	Gyroscope (Con quay hồi chuyển)
Accel	Accelerometer (Cảm biến gia tốc)
KF	Kalman Filter (Bộ lọc Kalman)
PWM	Pulse Width Modulation (Điều chế độ rộng xung)
UWB	Ultra-Wideband (băng tần siêu rộng)
LRF	Laser Range Finder (cảm biến đo xa laser)
SLAM	Simultaneous Localization And Mapping
RMSE	Root Mean Square Error
RRMSE	Relative Root Mean Square Error
ROS	Robot Operating System (Hệ điều hành cho Robot)
PCB	Printed Circuit Board (Bản mạch điện tử)

## Chương 1

# TỔNG QUAN VỀ ĐỀ TÀI

### 1.1. Đặt vấn đề

Một thách thức quan trọng đối với tất cả các robot di động là khả năng trả lời câu hỏi: Tôi đang ở đâu? Hệ thống định vị hay bản địa hóa (Localization) đóng một vai trò quan trọng trong khung điều hướng của robot di động tự trị. Bởi vì, nó cung cấp thông tin quan trọng cho các hệ thống còn lại của khung điều hướng.

Bản địa hóa (Localization) là vấn đề ước tính vị trí và hướng của robot so với môi trường hoạt động của nó từ các quan sát cảm biến. Do đó, để bản địa hóa thành công, robot phải xác định chính xác cả mô hình chuyển động và mô hình đo lường. Tuy nhiên, có một số lỗi không xác định vẫn còn, dẫn đến sự không chắc chắn trong ước tính tư thế robot theo thời gian. Mỗi lỗi dựa trên quy tắc phân phối, là Gauss hoặc không phải Gauss. Do đó, để cải thiện độ chính xác của hệ thống định vị trong môi trường động, cần xác định nguyên nhân gây nhiễu, loại phân phối nhiễu và cách loại bỏ nhiễu đó.

Thông thường để cải thiện độ chính xác của hệ thống định vị tư thế robot trong môi trường động, robot di động được trang bị nhiều loại cảm biến, như encoder, GPS, IMU, Compass, các loại cảm biến không dây như WiFi, BLE, RF, LRF, Ultrasonic, UWB, sử dụng thị giác máy tính với vision camera... Mỗi loại cảm biến chỉ đo một hoặc hai tham số của môi trường với độ chính xác khác nhau, bằng cách kết hợp dữ liệu từ nhiều nguồn cảm biến được trang bị trên robot, có thể ước tính tư thế tổng thể của Robot với sai số nhỏ nhất có thể. Phương pháp hợp nhất nhiều nguồn dữ liệu từ cảm biến chủ yếu sử dụng các bộ lọc như Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), phương pháp định vị giữa phép đo với các điểm mốc (landmark) chủ yếu sử dụng Particle Filter (PF).

### 1.2. Tính cấp thiết

Bài toán dẫn đường, điều hướng cho robot từ một điểm xuất phát tới đích một cách an toàn được coi là bài toán chính trong nghiên cứu robot hiện nay. Đặc biệt là trong các ứng dụng robot AGV, robot phục vụ, tiếp tân ..., robot cần phải di chuyển một cách chính xác để thực hiện nhiệm vụ và vẫn đảm bảo sự an toàn cho chính robot, vật dụng và con người xung quanh.

Bài toán chung của dẫn đường robot di động là việc trả lời 3 câu hỏi: “Where am I? (robot đang ở đâu)”, “Where am I going? (robot sẽ đi tới đâu)”, và “How should I get there? (robot sẽ đi tới đó như thế nào)”. Muốn giải quyết được bài toán này thì robot phải tự xác định được vị trí của mình trong môi trường hoạt động (quá trình localization), có được bản đồ môi trường hoạt động (quá trình mapping), vạch ra được quỹ đạo di chuyển tới đích (quá trình path planning), xuất tín hiệu điều khiển theo quỹ đạo đã có (quá trình path controlling) và phải tránh được vật cản trên quỹ đạo di chuyển (quá trình obstacle avoidance).

Việc nghiên cứu sử dụng các cảm biến và hệ thống phần cứng hiện đại cũng như phát triển các giải thuật phần mềm nhằm tăng độ tin cậy khi giải quyết câu hỏi thứ nhất: *Robot đang ở đâu?* Tư thế của robot (vị trí và hướng) được tính toán từ các cảm biến đặt trên robot. Mỗi cảm biến sẽ có những ưu nhược điểm với các cảm biến còn lại và luôn tồn tại nhiều đo lường ảnh hưởng đến độ chính xác của phép đo. Kết quả định vị nhận được của từng cảm biến riêng lẻ thường bị giới hạn về độ chính xác và tin cậy. Ví dụ, *cảm biến siêu âm* là một thiết bị có giá thành thấp và có lợi thế khi cho được kết quả đo nhanh hơn so với các thiết bị khác, tuy nhiên độ chính xác thấp. Camera (*stereo camera*) có thể xác định khoảng cách gián tiếp trên cơ sở tính toán các tọa độ điểm ảnh, nhưng lại đòi hỏi một quá trình tính toán lớn với độ chính xác của kết quả không cao và cũng lệ thuộc nhiều về điều kiện ánh sáng. Cảm biến đo xa laser LRF (*laser range finder*) có khả năng thu thập đo đặc khoảng cách với tốc độ và độ chính xác cao, kết quả không phụ thuộc quá nhiều vào điều kiện

môi trường, tuy nhiên cảm biến này không phát hiện được với các vật liệu trong suốt, không phát hiện được những vật có kết cấu không giống nhau theo chiều dọc (như bàn, các khung dầm ngang...). Cảm biến hướng từ (la bàn) bị bóp méo gần đường tải điện hoặc kết cấu thép làm ảnh hưởng trực tiếp đến phép đo góc khi sử dụng ở môi trường trong nhà, hoặc nơi có nhiều vật thể kim loại.

Những công trình nghiên cứu gần đây, phương pháp *tổng hợp cảm biến* (sensor fusion) đã được áp dụng nhằm nâng cao độ chính xác và tin cậy của các ước lượng trạng thái robot. Mục đích của tổng hợp cảm biến là thực hiện một kiến trúc cảm nhận mới trên cơ sở tổng hợp đa thông tin từ cảm biến cho việc nhận dạng môi trường. Hiện nay, hầu hết các giải thuật cho quá trình hợp nhất cảm biến được phát triển dựa trên suy luận xác suất như EKF, UKF, PF... Trong đó áp dụng bộ lọc Kalman mở rộng EKF [1-14] là giải pháp thông dụng nhất để hợp nhất các nguồn dữ liệu từ cảm biến và đưa ra tư thế của robot trong môi trường hoạt động.

Đề tài sẽ đi sâu vào nghiên cứu vấn đề ứng dụng bộ lọc EKF cho quá trình hợp nhất (fusion) dữ liệu từ các nguồn cảm biến khác nhau để nâng cao độ chính xác cho quá trình định vị tư thế robot (Localization).

### 1.3. Những nghiên cứu liên quan

Bảng 1. 1 Một số bài báo liên quan đến Fusion sensor với Kalman

Hợp nhất cảm biến (Sensor Fusion)	Tài liệu
Odometry + IMU (gyro)	[2]
Odometry + IMU + Lidar	[3]
Odometry + IMU + UWB (landmark)	[4]
Odometry + IMU + Beacon (landmark)	[5]
Odometry + UWB (landmark)	[6]
Odometry + Ultrasonic sensor	[7]
Odometry + IMU + GPS	[8]
Odometry + IMU (AHRS)+ GPS	[9]
Odometry+Compass+GPS	[10]

Odometry + 02 IMU + 02 GPS	[11]
Odometry + IMU+Compass+GPS	[12]
Vision + IMU +UWB (landmark)	[13]
Vision + Odometry+IMU+Compass	[14]
Vision + Odometry + compass + Lidar	[1]

Trong [3], Tác giả trình bày một phương pháp cho vấn đề nội địa hóa robot di động tự động (AGV). Bằng cách sử dụng EKF để hợp nhất dữ liệu từ các điểm mốc – landmark (thu được thông qua cảm biến Lidar SICK) kết hợp với thông tin từ odometry (encoder) và IMU (đọc góc yaw). Trong bài này, tác giả đã sử dụng rất nhiều cột mốc đặt theo quy luật, để Lidar có thể nhận được nhiều thông tin nhất về landmark. Việc này sẽ tạo ra thuận lợi cho quá trình Localization và SLAM. Tương tự [3] [5] các tác giả đều kết hợp các nguồn thông tin từ Odometry + IMU + Landmark, chỉ khác nhau về công nghệ sóng không dây (UWB và MarvelMind Ultrasonic Beacon). Trong [3] Khi sử dụng thêm sóng băng thông rộng UWB cho sai số đến 15cm. Tương tự, trong [6] tác giả sử dụng 06 UWB làm 06 điểm mốc (landmark) kết hợp với thông tin Odometry (không có IMU) hợp nhất dữ liệu với EKF cho sai số 17.75cm.

**Table 1.** The maximum error statistical results of the three positioning methods.

Positioning Methods Experimental Cases		IMU	UWB	EKF
Situation I	1(a)	63.1 cm	59.6 cm	16.2 cm
	2(b)	79.9 cm	58.4 cm	14.2 cm
	3(c)	63.3 cm	62.6 cm	14.7 cm
Situation II	4(d)	59.4 cm	103.5 cm	15.5 cm
	5(e)	61.3 cm	82.3 cm	15.0 cm

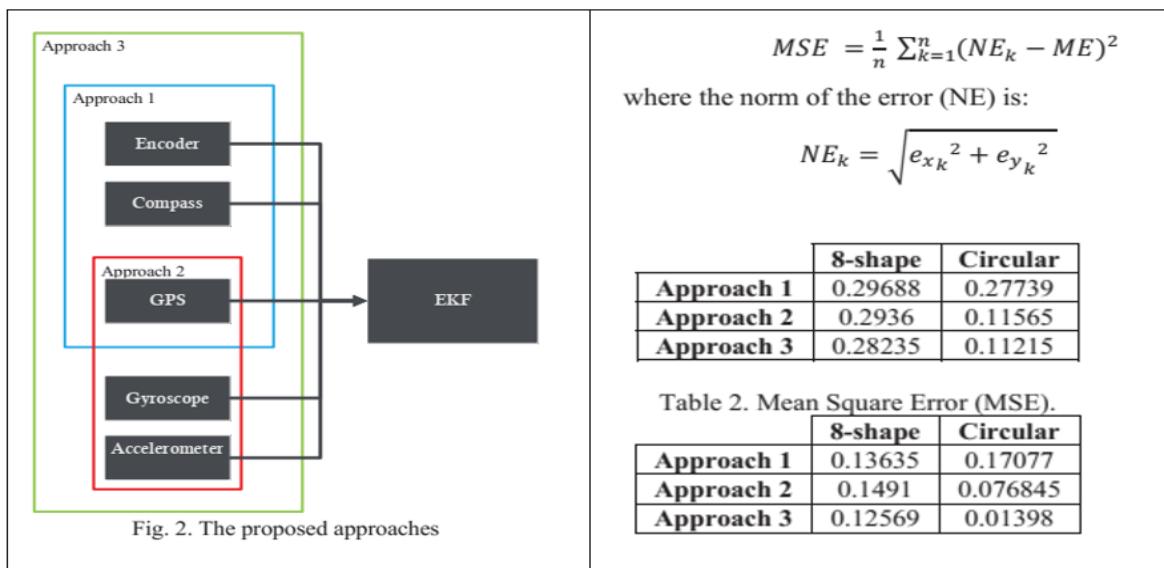
Hình 1. 1 EKF giữa odometry , IMU và UWB [3]

Trong [8] tác giả trình bày thuật toán bản địa hóa robot, sử dụng Bộ lọc Kalman mở rộng (EKF) để hợp nhất dữ liệu từ bộ mã hóa bánh xe quang, con

quay hồi chuyển, gia tốc kế và GPS. Hiệu suất của thuật toán được đánh giá bằng thực nghiệm bằng cách sử dụng robot di động SeekurJr<sup>1</sup>.

Trong [5] nhóm tác giả đến từ ĐHQG Hà nội, sử dụng EKF kết hợp tín hiệu IPS ultrasonic beacons với bộ mã hóa và cảm biến IMU-MPU6050 để xác định vị trí và robot định hướng chạy trên hệ điều hành Robot ROS sử dụng Robot UET-FuSo. Kết quả thu được cho thấy hiệu quả của phương pháp được đề xuất khi nội địa hóa robot ổn định và chính xác hơn so với chỉ sử dụng tín hiệu IPS. Bản địa hóa này sẽ được áp dụng để lập bản đồ và điều hướng robot di động trong robot hướng dẫn triển lãm.

Trong [12], tác giả đưa ra 03 cách tiếp cận khác nhau như hình 1.2. Kết quả cho thấy phương pháp 3 vượt trội so với hai phương pháp khác. Kết luận trong bài báo này là nhiều phép đo hơn sẽ cho độ chính xác ước tính tốt hơn. Ngoài ra, nếu một cảm biến thất bại, thuật toán vẫn cho kết quả chính xác khi sử dụng phương pháp tiếp cận 3.



Hình 1.2 Ba phương pháp tiếp cận khác nhau của EKF [8, p.4, fig.2]

<sup>1</sup> <https://www.generationrobots.com/media/SeekurJr-Datasheet-RevB.pdf>

Trong [10] nhóm tác giả triển khai EKF với Odometry, Compass, GPS, đánh giá bằng MSE, kết quả như sau:

MEAN SQUARE ERROR FOR THE THREE APPROACHES

Sensors	Sinusoidal trajectory	Circular trajectory
Encoder	6.1162	5.9312
Encoder + GPS	0.0289	0.0217
Encoder + Compass	0.0189	0.0243
Encoder + GPS + Compass	0.0081	0.0115

Hình 1. 3 Kết quả đánh giá EKF với Encoder, GPS và Compass [1, table.2]

Trong [14], tác giả sử dụng vector trạng thái  $\mathbf{x}$  và vector điều khiển  $\mathbf{u}$  là:  $\mathbf{x} = [x, y, z, \varphi, \theta, \psi]^T$      $\mathbf{u} = [\Delta s_L, \Delta s_R, \omega_x, \omega_y, \omega_z]^T$ , sử dụng EKF để hợp nhất nhiều dữ liệu qua các trường hợp, đánh giá bằng RRMSE, việc tính toán các landmark sử dụng kỹ thuật thị giác máy tính. Kết quả thu được cho sai số rất nhỏ trong trường hợp đầy đủ nhất là 2.3 mm.

Qua phân tích một số bài báo trên, các bài báo đều rất mới chủ yếu là xuất bản năm 2018-2020, rõ ràng vấn đề Localization đang được rất quan tâm và có nhiều giải pháp được đưa ra để nâng cao độ chính xác, từ sử dụng nhiều loại cảm biến chính xác khác nhau, đến xử lý ảnh, các thuật toán hợp nhất như EKF, UKF, PF, Học máy, Deep-Learning ...

### 1.4. Mục tiêu nghiên cứu

Từ các xuất phát điểm nêu trên, đề tài tập trung vào mục đích sau:

*Nghiên cứu thực nghiệm, cải thiện độ chính xác hệ thống định vị cho robot di động tự hành hai bánh, dựa trên phương pháp hợp nhất nhiều cảm biến sử dụng bộ lọc Kalman mở rộng (EKF).*

Đề tài sẽ tiến hành các nội dung như:

- Xây dựng một mô hình robot di động vi sai bao gồm 02 bánh xe chủ động và 01 bánh xe tự do (caster) chứa nhiều loại cảm biến như: Encoder, IMU MPU-9250, UWB DWM1001C, RPLidar A1M8-R5.
- Triển khai phương pháp tổng hợp đa cảm biến dựa trên bộ lọc Kalman mở rộng (EKF) cho mô hình Robot đã xây dựng.
- Cho Robot di chuyển theo quỹ đạo định trước hoặc quỹ đạo điều hướng, đánh giá độ chính xác của phương pháp đã triển khai bằng đo đạc trực tiếp.

### 1.5. Đối tượng và phạm vi nghiên cứu

- Bộ lọc EKF và bài toán kết hợp nhiều dữ liệu từ các cảm biến.
- Mô hình robot di động có kiểu vi sai (02 bánh chủ động và 01 bánh caster)
- Embedded MCU, Linux, ROS (Robot Operating System)
- Phương pháp xác định vị trí, hướng với các cảm biến như: Encoder, IMU, UWB, Lidar

### 1.6. Cấu trúc của Luận văn

#### Chương 1: Giới thiệu tổng quan về đề tài

Giới thiệu sơ lược về đề tài, các nghiên cứu liên quan trong và ngoài nước. Trình bày mục tiêu, đối tượng và phạm vi nghiên cứu.

#### Chương 2: Cơ sở lý thuyết

Trình bày mô hình toán mô tả của hệ thống, bao gồm mô hình toán của động cơ DC và mô hình toán tổng thể của Robot.

Lý thuyết về bộ lọc EKF và ứng dụng vào mô hình Robot.

### **Chương 3: Xây dựng mô hình và bộ điều khiển cho Robot**

Giới thiệu về mô hình và các thành phần Robot đã xây dựng.

Nhận dạng hàm truyền cho từng động cơ, xây dựng bộ điều khiển PID và đánh giá đáp ứng của hệ thống.

Phương pháp sử dụng các cảm biến trên Robot như MPU9250, UWB và Lidar.

### **Chương 4: Mô phỏng hệ thống**

Sử dụng Matlab Simulink để mô phỏng quá trình hoạt động của Robot

Sử dụng Python để mô phỏng quá trình hợp nhất các cảm biến với EKF

### **Chương 5: Thực nghiệm tổng hợp cảm biến cho mô hình đã xây dựng**

Triển khai thiết bị UWB trên môi trường robot hoạt động; kiểm tra các điều kiện về cảm biến.

Cho Robot di chuyển theo các quỹ đạo khác nhau; đánh giá kết quả định vị bằng RMSE và đo lường trực tiếp bằng tay.

### **Chương 6: Kết luận và hướng phát triển**

Đánh giá về ưu nhược điểm đề tài đã thực hiện

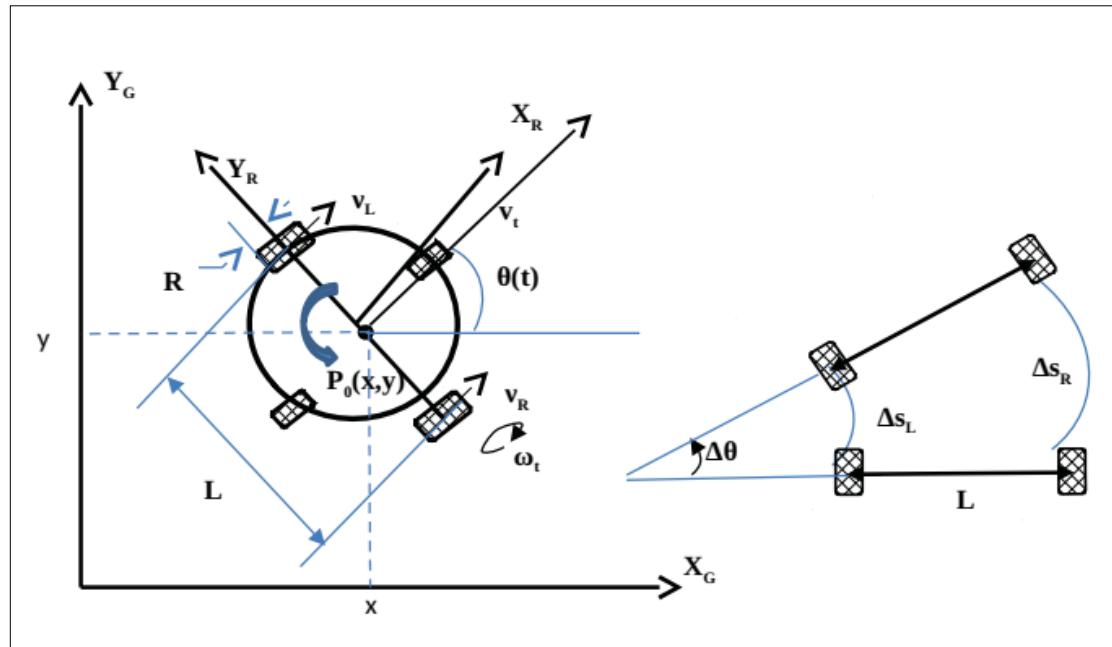
Hướng hoàn thiện và phát triển đề tài

## Chương 2 CƠ SỞ LÝ THUYẾT

### 2.1. Mô hình toán của hệ thống Robot

#### 2.1.1. Mô hình động học của Robot

##### 2.1.1.1. Phương trình trạng thái rời rạc của Robot



Hình 2. 1 Hệ quy chiếu toàn cục  $O_GX_GY_G$  và hệ quy chiếu robot  $O_RX_RY_R$ <sup>2</sup>

Robot được thiết kế là dạng robot di động có 02 bánh xe chủ động kiểu vi sai (differential drive wheeled robot) cùng bán kính R và cách nhau một khoảng L, tâm của robot là trung điểm của đoạn thẳng nối từ tâm trực giữa 02 bánh xe, các bánh xe còn lại (một đến hai) được thả tự do. Từ [1] [10] [15], phương trình trạng thái của robot tại thời điểm  $k$  trong hệ tọa độ toàn cục được cập nhập từ thời điểm  $k-1$  như sau:

<sup>2</sup> Tham khảo hình tại [10, p. 2] fig. 1

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} \Delta S_k \cos(\theta_{k-1} + \Delta\theta_k / 2) \\ \Delta S_k \sin(\theta_{k-1} + \Delta\theta_k / 2) \\ \Delta\theta_k \end{bmatrix} \quad (2.1)$$

Trong đó tư thế của robot (tọa độ và hướng) trong hệ tọa độ toàn cục được biểu diễn bởi vector trạng thái  $\mathbf{x} = [x, y, \theta]^T$  với  $(x, y)$  là tọa độ tâm robot và  $\theta$  là góc hướng của robot (đơn vị rad) chiều dương là chiều ngược kim đồng hồ, có giá trị từ  $(-\pi \div \pi)$ . Các số gia dịch chuyển  $(\Delta S_k, \Delta\theta_k)$  chính là quãng đường và góc dịch chuyển được sau một thời gian lấy mẫu  $T_s$  (giây),  $\Delta S_k = v_{k-1} T_s$  và  $\Delta\theta_k = \omega_{k-1} T_s$ , trong đó gọi  $\mathbf{u}_k = [v_k, \omega_k]^T$  là tín hiệu điều khiển vận tốc dài (linear velocity [m/s]) và vận tốc góc (angular velocity [rad/s]) của robot,  $(v_k, \omega_k)$  được tính toán dựa vào tốc độ của 02 bánh xe trái và phải (dựa vào thông tin phản hồi từ encoder trên mỗi trục quay của bánh xe).

Từ các lập luận trên, phương trình trạng thái của robot (system model) ở thời điểm  $k$  như sau:

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} v_{k-1} T_s \cos(\theta_{k-1} + 0.5 T_s \omega_{k-1}) \\ v_{k-1} T_s \sin(\theta_{k-1} + 0.5 T_s \omega_{k-1}) \\ T_s \omega_{k-1} \end{bmatrix} \quad (2.2)$$

### 2.1.1.2. Động học thuận và nghịch của Robot (Robot Kinematics)

➤ Động học thuận (Forward Kinematics)

$$v = \frac{R}{2}(\omega_R + \omega_L); \omega = \frac{R}{L}(\omega_R - \omega_L) \quad (2.3)$$

➤ Động học nghịch (Inverse Kinematics)

$$\omega_L = \frac{1}{R}(v - \frac{\omega L}{2}); \omega_R = \frac{1}{R}(v + \frac{\omega L}{2}) \quad (2.4)$$

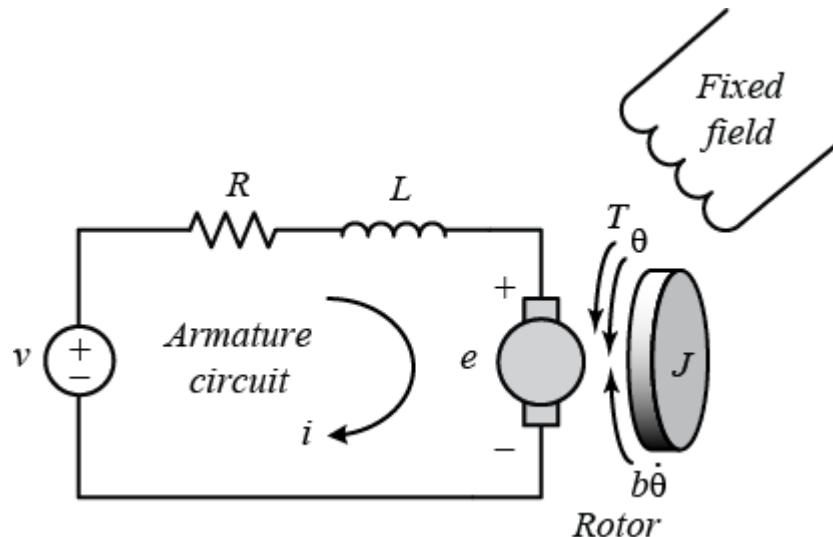
Trong đó:

Bảng 2. 1 Ý nghĩa các tham số của robot

Ký hiệu	Ý nghĩa	Đơn vị
$\omega_L$	Vận tốc góc của bánh xe bên trái	(rad/s)

$\omega_R$	Vận tốc góc của bánh xe bên phải	(rad/s)
$v$	Vận tốc tuyến tính (dài) của robot	(m/s)
$\omega$	Vận tốc góc của robot	(rad/s)
L	Khoản cách giữa 02 bánh xe	(m)

### 2.1.2. Mô hình động học của động cơ điện một chiều (DC Motor)



Hình 2. 2 Sơ đồ điện của động cơ điện 1 chiều <sup>3</sup>

Từ [16-19], Hàm truyền động cơ DC với ngõ vào điện áp U [Volt] và ngõ ra tốc độ  $\Omega$  [rad/s] như sau:

$$G(s) = \frac{\Omega(s)}{U(s)} = \frac{K_T}{JR_a s^2 + (bL_a + JR_a)s + (bR_a + K_T K_b)} \quad (2.5)$$

Trong đó:

Bảng 2. 2 Ý nghĩa các tham số của động cơ DC

Ký hiệu	Ý nghĩa	Đơn vị
$R_a$	Điện trở phần ứng	Ohm
$L_a$	Điện cảm phần ứng	H
$J$	Mô men quán tính của trục động cơ	Kg.m <sup>2</sup>
$b$	Hệ số ma sát	N.m.s

<sup>3</sup> <http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling>

$K_T$	Hằng số mô men xoắn	N.m/A
$K_b$	Hằng số sức phản điện (EMF)	V/rad/s
$U$	Điện áp ngõ vào	V
$\Omega$	Tốc độ quay của trục động cơ	rad/s

Thông thường, điện cảm của động cơ tương đối nhỏ so với quán tính của động cơ nên có thể bỏ qua ở tần số thấp. Do đó, hàm truyền tốc độ động DC có thể xấp xỉ như hệ thống bậc 1:

$$G(s) = \frac{\Omega(s)}{U(s)} = \frac{K_T}{JR_a s + (bR_a + K_T K_b)} = \frac{K}{\tau s + 1} \quad (2.6)$$

$$\text{Trong đó: } K = \frac{K_T}{(bR_a + K_T K_b)}; \tau = \frac{JR_a}{(bR_a + K_T K_b)} \quad (2.7)$$

Trong đề tài này, sử dụng mạch điều khiển cầu H (H- Bridge Controller) để cấp điện áp cho động cơ DC, nên  $U = \frac{PWM}{1000} * VoltageSupply$  [V], với PWM là giá trị băm từ MCU (giá trị từ -1000 đến +1000); VoltageSupply là nguồn cung cấp cho động cơ, nên phương trình (2.6) có thể được viết lại:

$$\Omega(s) = \frac{K}{\tau s + 1} U(s) = \frac{K * VoltageSupply * PWM}{1000 * (\tau s + 1)} = \frac{PWM}{H(\tau s + 1)} \quad (2.8)$$

Với  $H = \frac{1000}{K * VoltageSupply}$ , có thể xem đây là một hằng số, lúc này có thể tính toán lại hàm với tín hiệu đầu vào là PWM [-1000 ÷ 1000] và ngõ ra là  $\Omega$  [rad/s] như sau:  $G(s) = \frac{\Omega(s)}{PWM(s)} = \frac{H}{\tau s + 1}$ ; với H là một hằng số độ lợi mới. Như vậy, việc tính toán ước lượng hàm truyền cho động cơ dựa vào tín hiệu điều khiển PWM sẽ tiện lợi hơn.

## 2.2. Tổng hợp cảm biến bằng bộ lọc Kalman mở rộng EKF

### 2.2.1. Lý thuyết bộ lọc Kalman mở rộng (EKF)

Tham khảo [10] [15] [20-25], phương trình trạng thái của hệ thống với vector trạng thái  $\mathbf{x}_k \in \Re^n$  là phương trình sai phân ngẫu nhiên phi tuyến, có dạng:  $\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1})$  (2.9)

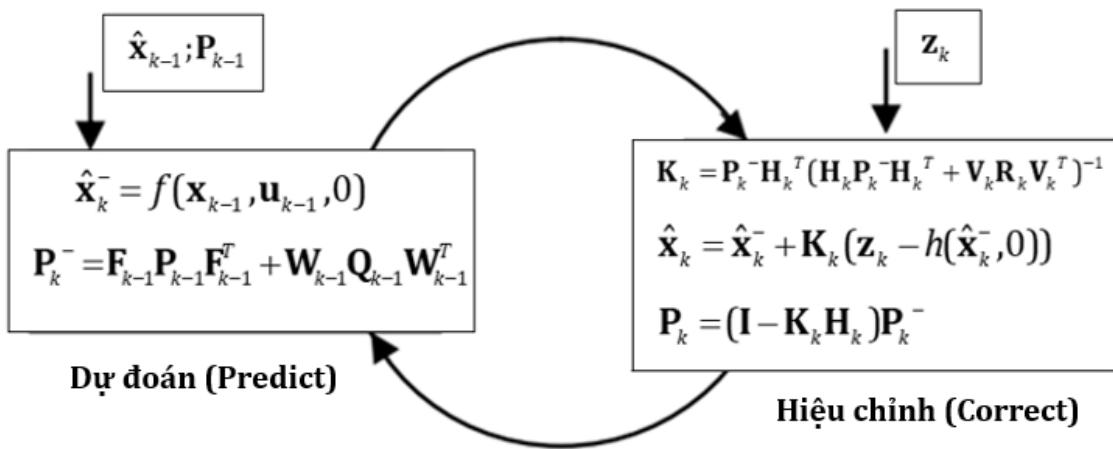
$$\text{Và phép đo } \mathbf{z}_k \in \Re^m \text{ là: } \mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k) \quad (2.10)$$

Trong đó  $f$  là hàm hệ thống với  $\mathbf{x}_k$  là vector trạng thái,  $\mathbf{u}_k$  là vector điều khiển, các biến ngẫu nhiên  $\mathbf{w}_k$  và  $\mathbf{v}_k$  cũng được giả định là nhiễu quá trình và nhiễu đo độc lập với nhau, là nhiễu trắng có phân phối xác suất chuẩn (Gauss):

$$\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k); \quad \mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k); \quad E(\mathbf{w}_k, \mathbf{v}_k) = 0; \quad (2.11)$$

Hàm phi tuyến  $f$  liên quan đến trạng thái ở bước thời gian hiện tại  $k$  đến trạng thái ở bước thời gian trước đó  $k-1$ . Nó còn bao gồm các tham số đầu vào  $\mathbf{u}_k$  và nhiễu quá trình  $\mathbf{w}_k$ . Hàm phi tuyến  $h$  của phép đo trong phương trình (2.10) biểu diễn mối liên quan giữa trạng thái  $\mathbf{x}_k$  với phép đo  $\mathbf{z}_k$ .

Quá trình thực hiện của bộ lọc Kalman mở rộng là một vòng lặp đệ quy với 02 giai đoạn là pha dự đoán (với các phương trình cập nhật thời gian) và pha hiệu chỉnh (với các phương trình cập nhật số liệu) như trên hình 2.3.



Hình 2.3 Sơ đồ thuật toán bộ lọc EKF

❖ **Pha dự đoán (prediction):** ở mỗi bước thời gian  $k$ , bộ lọc truyền trạng thái  $x$  và hiệp phương sai  $P$  của hệ thống ở bước trước tới bước hiện tại khi sử dụng các phương trình cập nhật thời gian:

$$\hat{\mathbf{x}}_k^- = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0) \quad (2.12)$$

$$\mathbf{P}_k^- = \mathbf{F}_{k-1} \mathbf{P}_{k-1} \mathbf{F}_{k-1}^T + \mathbf{W}_{k-1} \mathbf{Q}_{k-1} \mathbf{W}_{k-1}^T \quad (2.13)$$

❖ **Pha hiệu chỉnh (Correction):** Bộ lọc sửa ước lượng trạng thái tiên nghiệm với các phép đo  $\mathbf{z}_k$  bởi các phương trình cập nhật số đo sau:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{V}_k \mathbf{R}_k \mathbf{V}_k^T)^{-1} \quad (2.14)$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k (\mathbf{z}_k - h(\hat{\mathbf{x}}_k^-, 0)) \quad (2.15)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^- \quad (2.16)$$

Trong đó:

$\hat{\mathbf{x}}_k^- \in \mathbb{R}^n$  là ước lượng trạng thái tiên nghiệm ở bước  $k$  nhận được từ tiên nghiệm quá trình ở bước  $k$ .

$\hat{\mathbf{x}}_k \in \mathbb{R}^n$  là ước lượng trạng thái hậu nghiệm ở bước  $k$  nhận được sau phép đo  $\mathbf{z}_k$

$\mathbf{P}_k^-$  là ma trận hiệp biến của sai số ước lượng trạng thái tiên nghiệm.

$\mathbf{P}_k$  là ma trận hiệp biến của sai số ước lượng trạng thái hậu nghiệm.

$\mathbf{Q}_{k-1}$  là ma trận hiệp phương sai nhiễu đầu vào.

**R**<sub>k</sub> là ma trận hiệp phương sai của nhiễu đo.

**K**<sub>k</sub> là hệ số lọc Kalman.

**I** là ma trận đơn vị.

**F**<sub>k</sub> là ma trận Jacobian của các đạo hàm riêng của hàm *f* đối với **x**

**H**<sub>k</sub> là ma trận Jacobian của các đạo hàm riêng của hàm *h* đối với **x**

**W**<sub>k</sub> là ma trận Jacobian của các đạo hàm riêng của *f* theo **w**

**V**<sub>k</sub> là ma trận Jacobian của các đạo hàm riêng của *h* theo **v**

Các ma trận được tính như sau:

$$\mathbf{F}_k = \frac{\partial f}{\partial \mathbf{x}_{k-1}}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0); \quad \mathbf{W}_k = \frac{\partial f}{\partial \mathbf{w}}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0) \quad (2.17)$$

$$\mathbf{H}_k = \frac{\partial h}{\partial \mathbf{x}_k}(\mathbf{x}_k, 0); \quad \mathbf{V}_k = \frac{\partial h}{\partial \mathbf{v}}(\mathbf{x}_k, 0) \quad (2.18)$$

### 2.2.2. Ứng dụng bộ lọc EKF vào mô hình Robot

Như trình bày ở phần 2.1.1, mô hình hệ thống của Robot trong trường

hợp này như sau:  $\mathbf{x}_k = [x_k \quad y_k \quad \theta_k]^T$  là vector trạng thái của hệ thống.

$\mathbf{u}_k = [v_k \quad \omega_k]^T$  là vector điều khiển;  $\mathbf{w}_k = [\varepsilon_v \quad \varepsilon_\omega]^T$  là vector nhiễu quá trình của hệ thống, được cho là nhiễu Gauss, có phân phối chuẩn  $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)$ ; Nhiều này chủ yếu đến từ việc đo lường không chính xác các tham số như encoder của 02 động cơ trái phải, bán kính bánh xe, khoảng cách tiếp xúc của 02 bánh xe và các tham số mặt sàn, ma sát..., **Q**<sub>k</sub> (ma trận hiệp phương sai

nhiễu đầu vào)<sup>4</sup> có kích thước [2x2],  $\mathbf{Q}_k = \text{cov}(v_k, \omega_k) = \begin{bmatrix} \sigma_v & 0 \\ 0 & \sigma_\omega \end{bmatrix}$ ,

$\sigma_v$  và  $\sigma_\omega$  sẽ được tìm bằng thực nghiệm.

---

<sup>4</sup> Covariance = cov(X,Y) = Ma trận hiệp phương sai

$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1})$  là phương trình trạng thái của hệ thống.

Kết hợp với công thức có được từ (2.2), phương trình trạng thái đầy đủ của hệ thống:

$$\mathbf{x}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} (v_{k-1} + \varepsilon_v)T_s \cos(\theta_{k-1} + 0.5T_s(\omega_{k-1} + \varepsilon_\omega)) \\ (v_{k-1} + \varepsilon_v)T_s \sin(\theta_{k-1} + 0.5T_s(\omega_{k-1} + \varepsilon_\omega)) \\ T_s(\omega_{k-1} + \varepsilon_\omega) \end{bmatrix} \quad (2.19)$$

❖ **Khởi tạo ban đầu:**

Robot được khởi tạo từ vị trí ban đầu tại bước  $k = 0$ , với hiệu phương sai  $\mathbf{P}_0$  và ước lượng trạng thái  $\hat{\mathbf{x}}_0$

❖ **Trong pha dự đoán (predict):**

Theo công thức (2.19):

$$\hat{\mathbf{x}}_k^- = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0) = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} v_{k-1}T_s \cos(\theta_{k-1} + 0.5T_s\omega_{k-1}) \\ v_{k-1}T_s \sin(\theta_{k-1} + 0.5T_s\omega_{k-1}) \\ T_s\omega_{k-1} \end{bmatrix} \quad (2.20)$$

Các ma trận khác được tính như sau:

$$\begin{aligned} \mathbf{F}_k &= \frac{\partial f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0)}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_x}{\partial x} & \frac{\partial f_x}{\partial y} & \frac{\partial f_x}{\partial \theta} \\ \frac{\partial f_y}{\partial x} & \frac{\partial f_y}{\partial y} & \frac{\partial f_y}{\partial \theta} \\ \frac{\partial f_\theta}{\partial x} & \frac{\partial f_\theta}{\partial y} & \frac{\partial f_\theta}{\partial \theta} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & -T_s v_{k-1} \sin(\theta_{k-1} + 0.5T_s\omega_{k-1}) \\ 0 & 1 & -T_s v_{k-1} \cos(\theta_{k-1} + 0.5T_s\omega_{k-1}) \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (2.21)$$

$$\mathbf{W}_k = \frac{\partial f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0)}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial f_x}{\partial \sigma_v} & \frac{\partial f_x}{\partial \sigma_\omega} \\ \frac{\partial f_y}{\partial \sigma_v} & \frac{\partial f_y}{\partial \sigma_\omega} \\ \frac{\partial f_\theta}{\partial \sigma_v} & \frac{\partial f_\theta}{\partial \sigma_\omega} \end{bmatrix}$$

$$= \begin{bmatrix} T_s \cos(\theta_{k-1} + 0.5T_s \omega_{k-1}) & -0.5v_{k-1} T_s^2 \sin(\theta_{k-1} + 0.5T_s \omega_{k-1}) \\ T_s \sin(\theta_{k-1} + 0.5T_s \omega_{k-1}) & 0.5v_{k-1} T_s^2 \cos(\theta_{k-1} + 0.5T_s \omega_{k-1}) \\ 0 & T_s \end{bmatrix} \quad (2.22)$$

❖ Trong pha hiệu chỉnh (Correct):

Trạng thái của hệ thống được quan sát bằng một số phép đo tạo nên vector  $\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k)$ , trong đó  $\mathbf{v}_k$  là nhiễu đo lường, được cho là độc với với  $\mathbf{w}_k$  và có phân phối Gauss,  $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$

➤ Trường hợp thông tin đo lường từ UWB:

Lúc này hàm  $\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k) = \begin{bmatrix} x_{UWB} \\ y_{UWB} \end{bmatrix} + \begin{bmatrix} \mathcal{E}_{uwb\_x} \\ \mathcal{E}_{uwb\_y} \end{bmatrix}$  (2.23)

$$\mathbf{H}_{UWB} = \frac{\partial h(\mathbf{x}_k, 0)}{\partial \mathbf{x}_k} = \begin{bmatrix} \frac{\partial h_x}{\partial x} & \frac{\partial h_x}{\partial y} & \frac{\partial h_x}{\partial \theta} \\ \frac{\partial h_y}{\partial x} & \frac{\partial h_y}{\partial y} & \frac{\partial h_y}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (2.24)$$

$$\mathbf{V}_{UWB} = \frac{\partial h(\mathbf{x}_k, 0)}{\partial \mathbf{v}_k} = \begin{bmatrix} \frac{\partial h_x}{\partial \mathcal{E}_{uwb\_x}} & \frac{\partial h_x}{\partial \mathcal{E}_{uwb\_y}} \\ \frac{\partial h_y}{\partial \mathcal{E}_{uwb\_x}} & \frac{\partial h_y}{\partial \mathcal{E}_{uwb\_y}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.25)$$

Chọn  $\mathbf{R}_{UWB}$  [2x2] chính bằng phương sai<sup>5</sup> nhiễu đo của UWB .

$$\mathbf{R}_{UWB} = \begin{bmatrix} \text{var}(uwb\_x) & 0 \\ 0 & \text{var}(uwb\_y) \end{bmatrix} \quad (2.26)$$

$$\Rightarrow \mathbf{V}_{UWB} * \mathbf{R}_{UWB} * \mathbf{V}_{UWB}^T = \mathbf{R}_{UWB} = \begin{bmatrix} \text{var}(uwb\_x) & 0 \\ 0 & \text{var}(uwb\_y) \end{bmatrix}$$

- Trường hợp thông tin đo lường từ IMU (góc yaw)

$$\text{Lúc này hàm } \mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k) = [\theta_k] + [\varepsilon_{IMU}] \quad (2.27)$$

với  $\theta_k$  là góc hướng (yaw) của Robot,  $\varepsilon_{IMU}$  là nhiễu đo lường.

$$\mathbf{H}_{Yaw} = \begin{bmatrix} \frac{\partial h_x}{\partial x} & \frac{\partial h_x}{\partial y} & \frac{\partial h_x}{\partial \theta} \end{bmatrix} = [0 \ 0 \ 1] \quad (2.28)$$

$$\mathbf{V}_{YAW} = \frac{\partial h(\mathbf{x}_k, 0)}{\partial \mathbf{v}_k} = \begin{bmatrix} \frac{\partial h_x}{\partial \varepsilon_{IMU}} \end{bmatrix} = [1] \quad (2.29)$$

Chọn  $\mathbf{R}_{YAW}$  [1x1] chính bằng phương sai của của nhiễu đo lường

$$\text{góc yaw, } \mathbf{R}_{YAW} = [\text{var}(yaw\_imu)] \quad (2.30)$$

$$\Rightarrow \mathbf{V}_{YAW} * \mathbf{R}_{YAW} * \mathbf{V}_{YAW}^T = \mathbf{R}_{YAW} = [\text{var}(yaw\_imu)]$$

- Trường hợp thông tin đo lường bao gồm cả xy và  $\theta$

$$\text{Lúc này hàm } \mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k) = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_\theta \end{bmatrix} \quad (2.31)$$

với  $\varepsilon_x, \varepsilon_y, \varepsilon_\theta$  là nhiễu đo lường.

$$\mathbf{H}_{xy\theta} = \frac{\partial h(\mathbf{x}_k, 0)}{\partial \mathbf{x}_k} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.32)$$

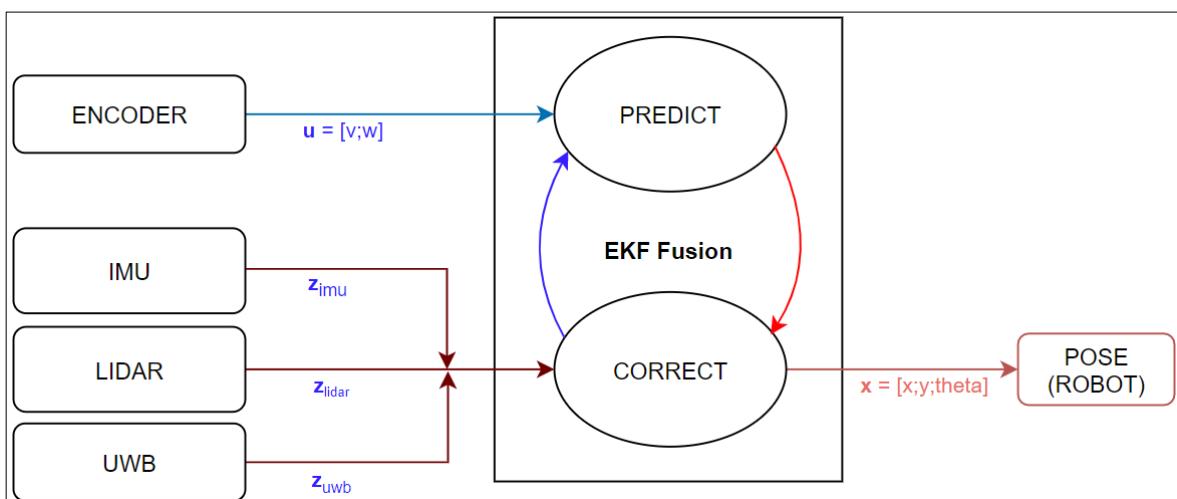
---

<sup>5</sup> Variance = var =  $\sigma^2$ ;  $\mathcal{N}(\mu, \sigma^2)$

$$\mathbf{V}_{XY\theta} = \frac{\partial h(\mathbf{x}_k, 0)}{\partial \mathbf{v}_k} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.33)$$

$$\mathbf{R}_{XY\theta} = \begin{bmatrix} \text{var}(sensor\_x) & 0 & 0 \\ 0 & \text{var}(sensor\_y) & 0 \\ 0 & 0 & \text{var}(sensor\_\theta) \end{bmatrix}$$

$$\Rightarrow \mathbf{V}_{XY\theta} * \mathbf{R}_{XY\theta} * \mathbf{V}_{XY\theta}^T = \mathbf{R}_{XY\theta}$$



Hình 2.4 Sơ đồ quá trình EKF Fusion cho Robot

**Nhận xét 1:** Trong cả 3 trường hợp biểu thức  $\mathbf{V} * \mathbf{R} * \mathbf{V}^T = \mathbf{R}$ , vì vậy trong công thức (2.14) chỉ cần sử dụng mỗi  $\mathbf{R}$  cho đơn giản tính toán. Tùy thuộc theo cách thức cập nhập (correct) mà sử dụng ma trận R cho phù hợp (kích thước phù hợp). Đối với  $\mathbf{W} * \mathbf{Q} * \mathbf{W}^T$  ở công thức (2.13) sử dụng trong quá trình predict, thay vì tính biểu thức ma trận trên, để cho đơn giản sử dụng ma trận Q kích thước 3x3 với giá trị cố định được khởi tạo ngay từ đầu.

**Nhận xét 2:** Cảm biến encoder (đọc tốc độ quay của 02 bánh xe) cung cấp tín hiệu điều khiển  $\mathbf{u} = [v; \omega]$  chỉ sử dụng trong quá trình predict, 03 phép đo còn lại từ IMU, UWB, Lidar sử dụng để cập nhập (update/correct) lại vị trí thật của Robot.

### Chương 3

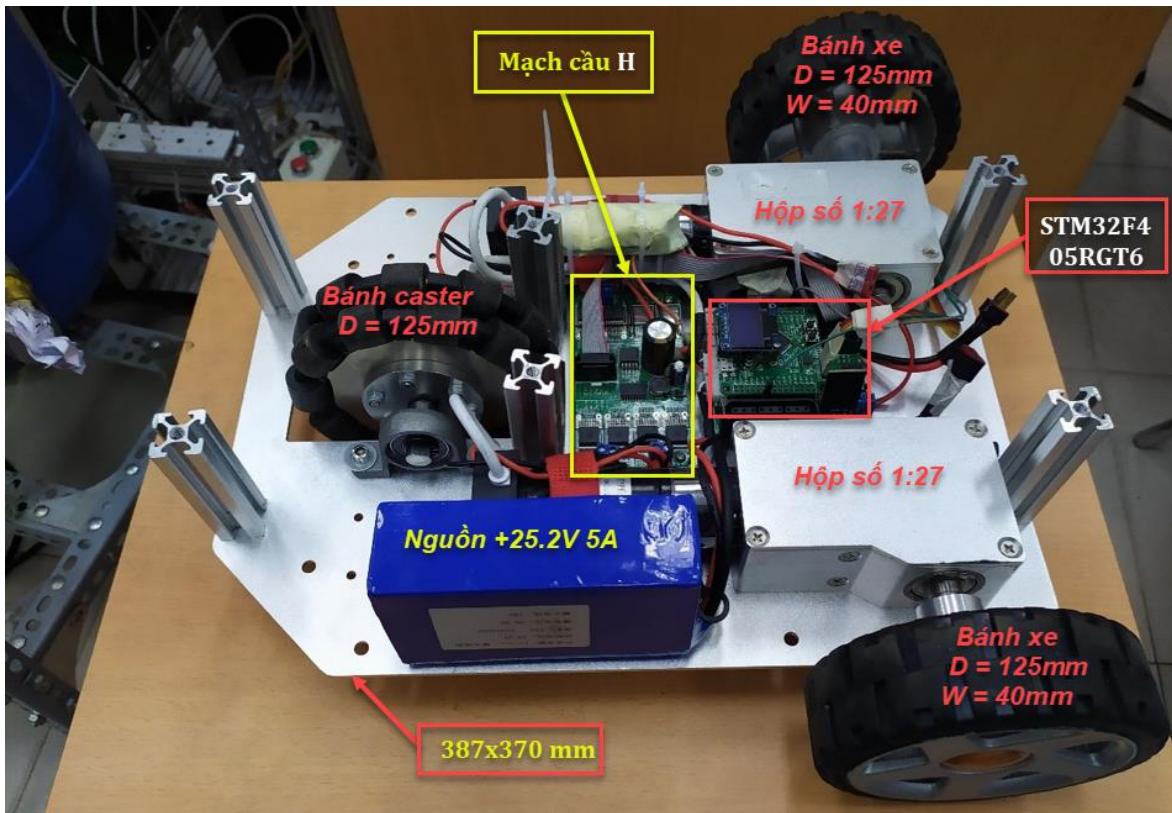
## XÂY DỰNG MÔ HÌNH VÀ BỘ ĐIỀU KHIỂN CHO ROBOT

### 3.1. Tổng quan về mô hình và thiết bị sử dụng trong Robot

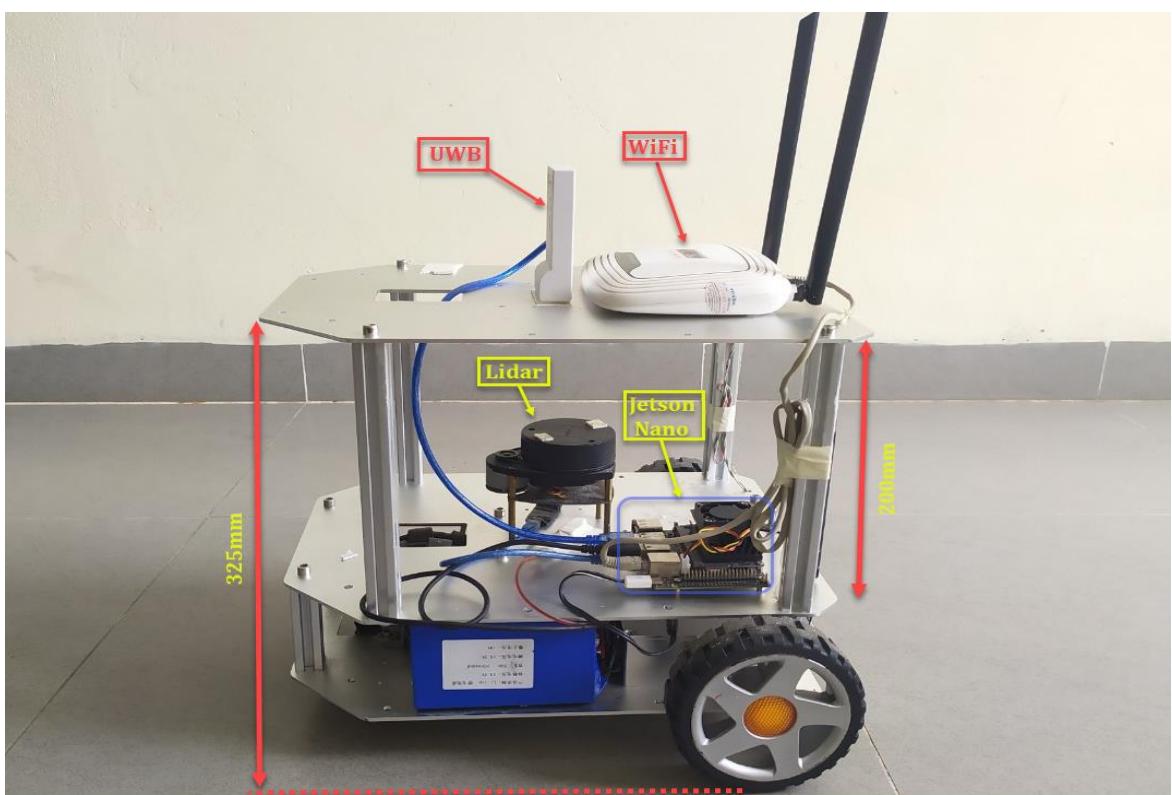
Bảng 3. 1 Thông số các thành phần sử dụng trong Robot

Items	Specification <sup>6</sup>	
Khung sườn robot và động cơ	Kích thước dài x rộng x cao(mm)	387 x 370 x 325
	Khối lượng (kg)	5.62 Kg
	Tải trọng (kg)	25 Kg
	Tốc độ dài tối đa của robot (v)	2.4 m/s
	Tốc độ quay tối đa của robot (w)	13.6 rad/s
	Actuator: sử dụng 02 DC Motor MD36NP27P; vận tốc góc tối đa (qua đổi tốc): 40 rad/s	
	Độ phân giải encoder	500 xung /vòng
	Kích thước bánh xe (wheel)	Diam = 125 (mm) Width = 40 (mm)
	Kích thước bánh xe caster	125 x 40 (mm)
JETSON (ROS)	<b>JETSON NANO</b> , CPU Quad-core ARM A57 @ 1.43 GHz GPU 128-core Maxwell; Memory 4 GB 64-bit LPDDR4	
MCU (STM32F4)	<b>STM32F405RGT6</b> , Core: ARM®32-bit Cortex®-M4 CPU with FPU, frequency up to 168 MHz, 210 DMIPS/1.25 DMIPS/MHz and DSP instructions	
IMU	<b>MPU9250 (09DOF)</b> , Gyroscope 3Axis, Accelerometer 3Axis, Magnetometer 3Axis	
UWB	<b>DWM1001C</b> , Decawave	
Lidar	Laser Distance Sensor, <b>RPLIDAR A1M8</b> của Salmtec.com	
H-Bridge	Điện áp 8 - 32V, Công suất 200W, Dòng điện liên tục 8A.	
Battery	<b>Pin Lion 25.2V - 5A</b> dùng cấp nguồn cho toàn bộ hệ thống và cấp nguồn +9V cho WiFi-Router.	
WiFi	Router <b>WiFi của Tp-link</b> , nguồn phụ 9V, sử dụng khi cần thiết muốn điều khiển từ xa với độ ổn định tốt nhất.	

<sup>6</sup> Các thông tin về tốc độ tối đa được đo trực tiếp thông qua dụng cụ khi Robot sử dụng nguồn +25.2V



Hình 3. 1 Hình ảnh thực tế phần bệ đỡ robot và mạch điều khiển



Hình 3. 2 Hình ảnh thực tế tổng quát của Robot

### 3.2. Xây dựng bộ điều khiển cho robot

#### 3.2.1. Nhận dạng hàm truyền động cơ

Như đã trình bày ở phần (2.1.1) và (2.1.2), toàn bộ quá trình điều hướng robot chính là quá trình điều khiển tốc độ quay của 02 bánh xe trái và phải ( $\omega_L, \omega_R$ ). Để có thể nhận dạng hàm truyền theo công thức (2.8) để tài sử dụng công cụ nhận dạng **System Identification**<sup>7</sup> và kết hợp với simulink để lấy dữ liệu từ 02 bánh xe.

Tốc độ tối đa của các bánh xe có thể đặt được 40 rad/s, ở tốc độ này Robot có thể chạy đến 2.4 m/s. Đề tài sẽ chọn nhận dạng ở một tốc độ chậm hơn, phù hợp với quá trình di chuyển, điều hướng cho Robot. Lựa chọn tốc độ Robot 0.5 m/s tương ứng với  $\omega_L, \omega_R \approx 9 \text{ rad/s}$  tại PWM = 250, thời gian lấy mẫu  $T_s = 0.005\text{s}$  (5ms). Các Files và data cần thiết cho quá trình nhận dạng này ở Phụ lục 1, tiểu mục 1.

Bảng 3. 2 Hàm truyền đã nhận dạng của động cơ Trái và Phải

Hàm truyền động cơ bên Trái	Hàm truyền động cơ bên Phải
$0.005571 (+/- 7.995e-05) z^{-1}$ <hr/> $1 - 0.8037 (+/- 0.002868) z^{-1}$ Sample time: 0.005 seconds Fit to estimation data: 96.44% FPE: 0.01465, MSE: 0.01459	$0.007311 (+/- 0.00013) z^{-1}$ <hr/> $1 - 0.7937 (+/- 0.003738) z^{-1}$ Sample time: 0.005 seconds Fit to estimation data: 95.63% FPE: 0.03452, MSE: 0.03438

#### 3.2.2. Lựa chọn tham số PID cho động cơ

Để đơn giản nhưng vẫn đảm bảo được tính chất điều khiển ổn định, đề tài sử dụng bộ PID để điều tốc cho 02 bánh xe. Việc tìm Gain PID sử dụng PID Tuner của Matlab. Chọn các thông số tính toán cho PID<sup>8</sup> như sau:

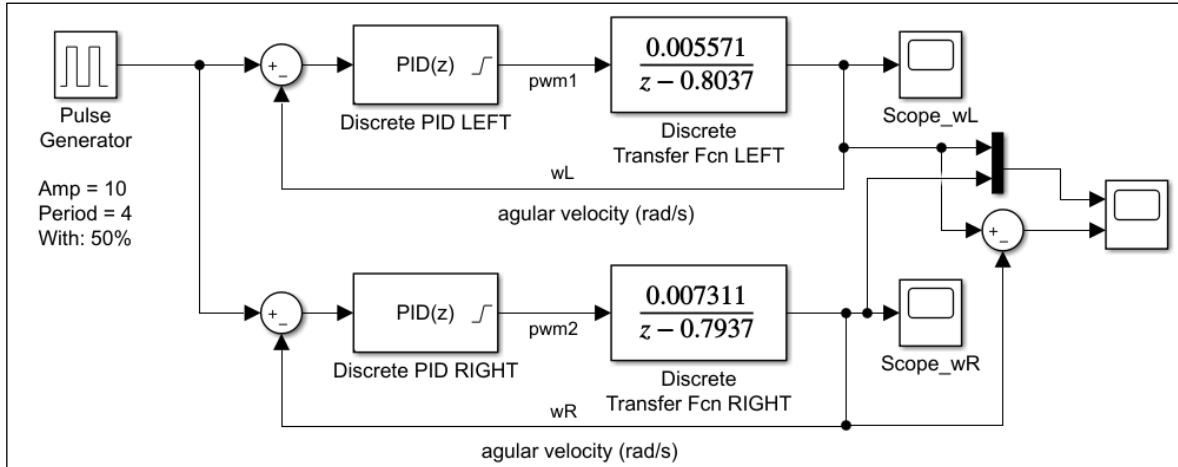
---

<sup>7</sup> Ref: <https://www.mathworks.com/help/ident/gs/about-system-identification.html>

<sup>8</sup> Ref: <https://www.mathworks.com/help/control/ug/discrete-time-proportional-integral-derivative-pid-controller.html>

- Thời gian lấy mẫu (Sample time):  $T_s = 0.005$  (giây)
- Phương pháp tính tích phân (Integrator methods) : Trapezoidal
- Phương pháp tính đạo hàm (derivative ): BackwardEuler

$$G_{PID}(z) = P + I \frac{T_s}{2} \frac{z+1}{z-1} + D \cdot \frac{1}{T_s} \frac{z-1}{z} \quad (3.1)$$

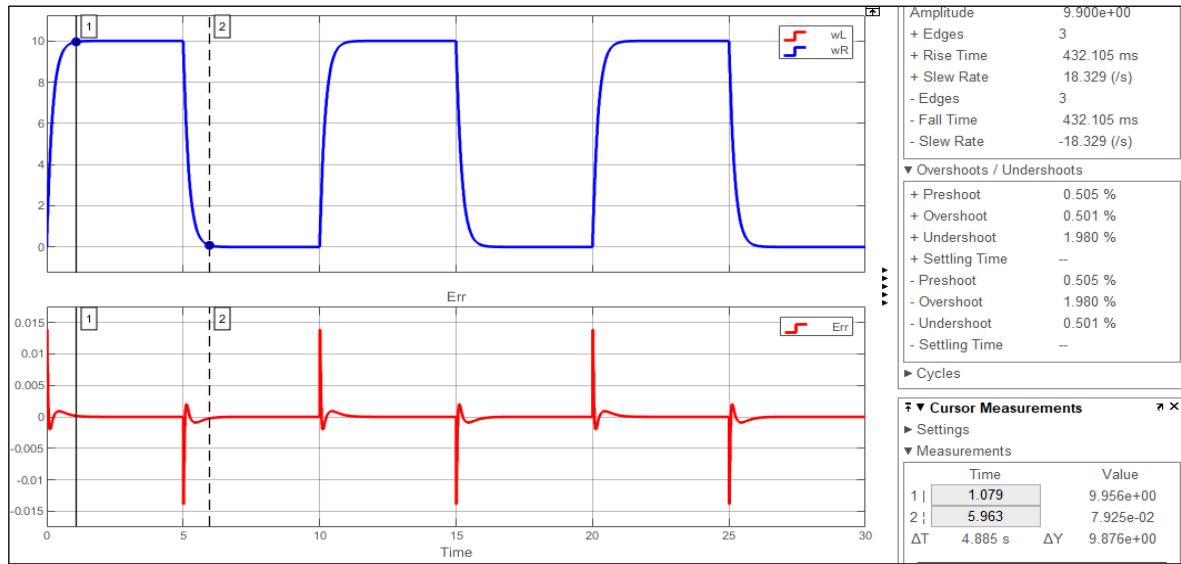


Hình 3. 3 Mô hình điều khiển với bộ điều khiển PIDz

Với yêu cầu đặt ra cho bộ điều khiển như: thời gian đáp ứng khoản 1s, vọt lố < 0.5%, bộ gain cho PID có được kết quả như bên dưới bảng 3.3.

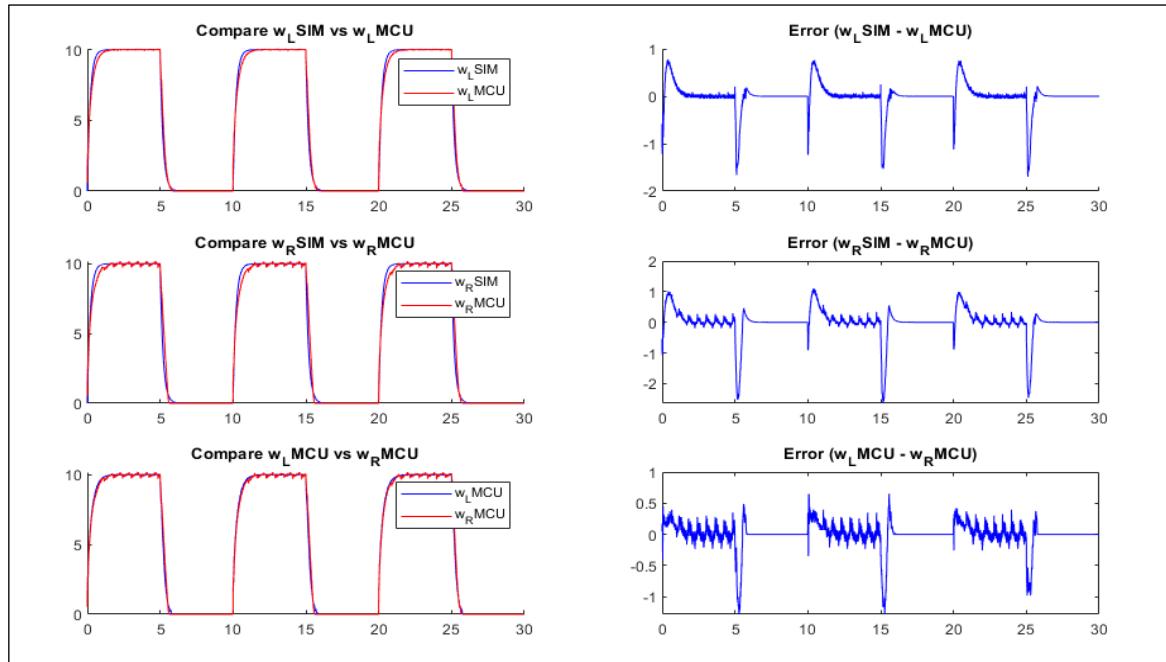
Bảng 3. 3. Hệ số PID tính toán và đáp ứng của động cơ

Động cơ	Hàm truyền	Gain PID			Đáp ứng (Response)
		K	I	D	
Trái	$Gz = \frac{0.005571}{z - 0.8037}$	4.4892	176.7686	0.0344	Rise time: 0.44s Setting time: 0.785s Peak: 1 Gain margin: Inf dB Phase margin: 90 deg Closed-loop: Stable
Phải	$Gz = \frac{0.007311}{z - 0.7937}$	3.4207	141.5116	0.0251	



Hình 3.4 Đáp ứng của hệ thống khi thêm bộ điều khiển PID

Để đảm bảo chắc chắn hệ thống làm việc chính xác với mô hình trên, cần nạp các bộ Gain PID xuống dưới MCU và so sánh đánh giá với kết quả mô phỏng trên simulink (Files và data thuộc Phụ lục 1, tiểu mục 2).



Hình 3.5 So sánh bộ PID rời rạc được triển khai dưới MCU với Simulink

**Nhận xét:** Cơ bản bộ PID sử dụng dưới MCU với Gain ( $K_p$ ,  $K_i$ ,  $K_d$ ) được tính toán trên mô hình nhận dạng với đáp ứng trên mô phỏng là tương đồng.

Đáp ứng của động cơ bên trái và bên phải gần như đè lên nhau. Tuy nhiên phần động cơ bên phải (màu đỏ) có răng cưa nhiều khi hệ thống đã vào trạng thái xác lập. Việc này có thể lý giải: do bản thân trực của bánh xe bên phải đang bị cong vênh, nên động cơ chạy rất lắc (đảo), gây ra gai răng cưa (tốc độ bánh xe không đều). Mặc dù vậy, nhưng bộ điều khiển PID vẫn hoạt động tốt, vẫn đưa được 02 bánh xe về trạng thái đáp ứng gần như nhau.

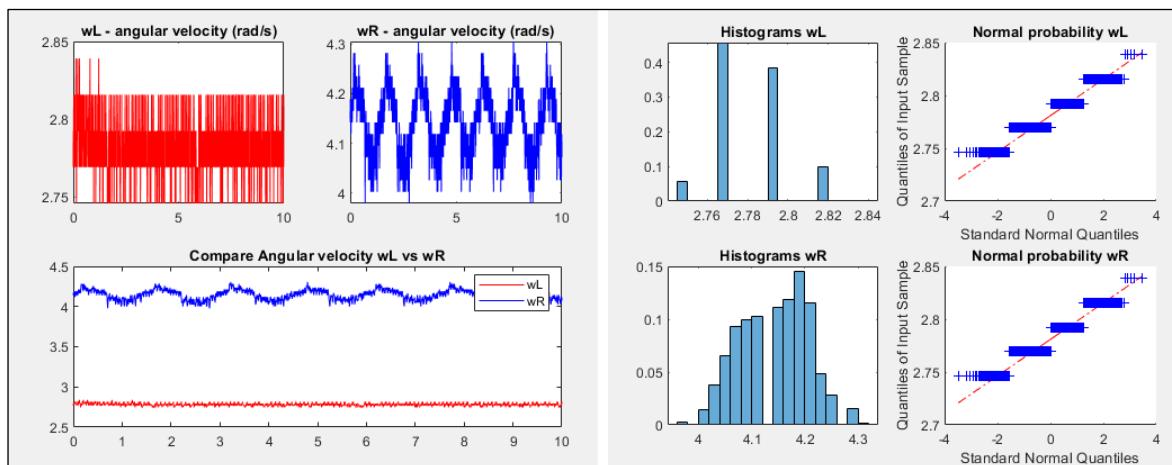
### 3.2.3. Đánh giá đáp ứng của động cơ ở các điều kiện khác nhau

Đây là phần đánh giá độc lập với bộ PID phía trên, mục đích kiểm tra sự đáp ứng của động cơ về kết cấu cơ khí, mạch điều khiển cầu H, nguồn điện cung cấp. Từ đó đánh giá sự phân phối nhiễu trên mỗi phép đo thông qua encoder. M-file và tập dữ liệu cho phần này ở Phụ lục 1, tiểu mục 2.

- ❖ Thủ 02 động cơ ở PWM cố định = 150

Các giá trị đo  $\omega_L$  và  $\omega_R$  dao động trên nền nhiễu theo phân phối chuẩn:

$$\mathbf{w}_L \sim \mathcal{N}(2.78164, 0.017724^2); \quad \mathbf{w}_R \sim \mathcal{N}(4.14362, 0.065157^2);$$

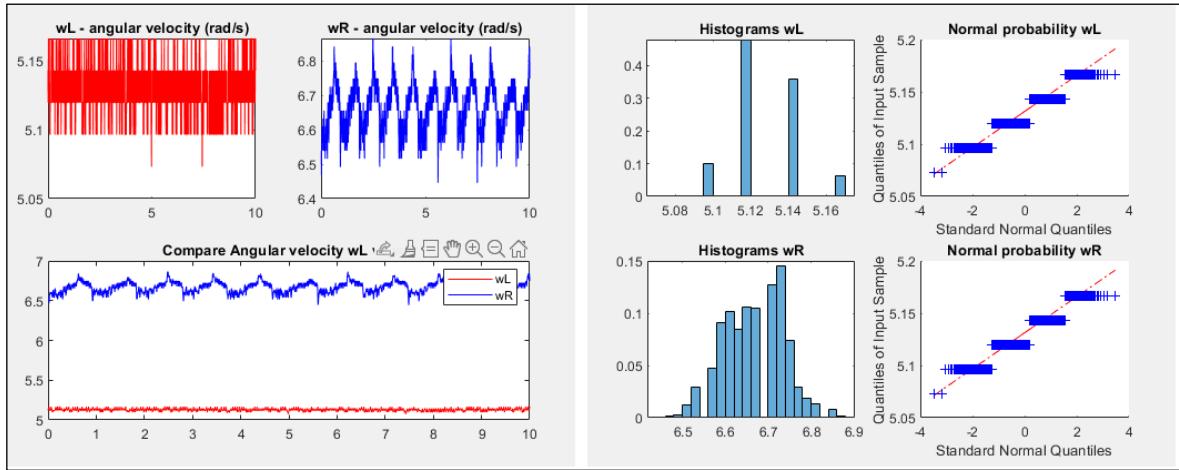


Hình 3.6 Đáp ứng tốc độ của các động cơ khi ở PWM = 150

- ❖ Thủ 02 động cơ ở PWM cố định = 200

Các giá trị đo  $\omega_L$  và  $\omega_R$  dao động trên nền nhiễu theo phân phối chuẩn:

$$\mathbf{w}_L \sim \mathcal{N}(5.12846, 0.0175005^2); \quad \mathbf{w}_R \sim \mathcal{N}(6.66757, 0.0702647^2);$$

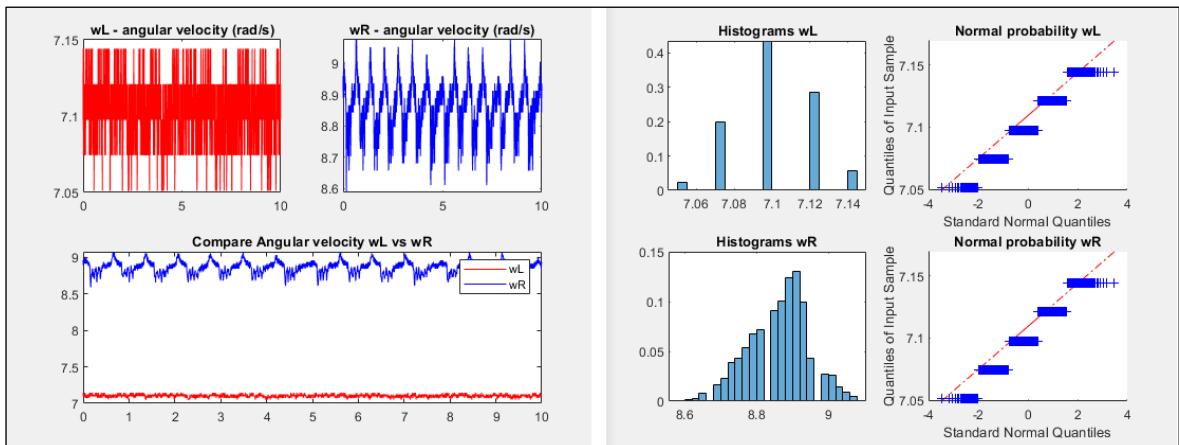


Hình 3. 7 Đáp ứng tốc độ của các động cơ khi ở PWM =200

❖ Thủ 02 động cơ ở PWM cố định = 250

Các giá trị đo  $\omega_L$  và  $\omega_R$  dao động trên nền nhiễu theo phân phối chuẩn:

$$\mathbf{w}_L \sim \mathcal{N}(7.10129, 0.0205832^2); \quad \mathbf{w}_R \sim \mathcal{N}(8.86238, 0.0847669^2);$$

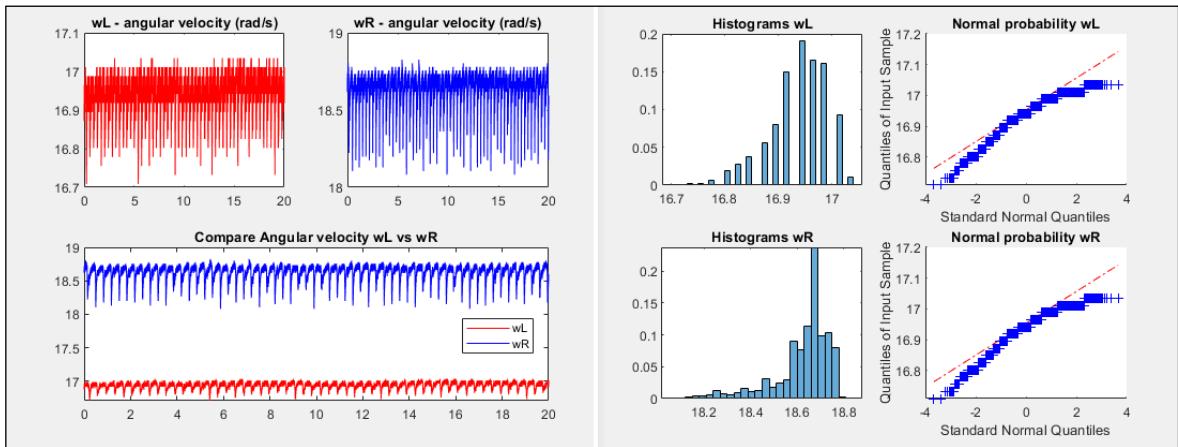


Hình 3. 8 Đáp ứng tốc độ của các động cơ khi PWM = 250

❖ Thủ 02 động cơ ở PWM cố định = 500

Các giá trị đo  $\omega_L$  và  $\omega_R$  dao động trên nền nhiễu theo phân phối chuẩn:

$$\mathbf{w}_L \sim \mathcal{N}(16.9378, 0.0537814^2); \quad \mathbf{w}_R \sim \mathcal{N}(18.6253, 0.123034^2);$$



Hình 3.9 Đáp ứng tốc độ của các động cơ khi ở PWM =500

**Nhận xét:** Dữ liệu đo lường cơ bản tuân theo phân phối chuẩn Gauss, điều này chứng tỏ mô hình đo lường đang được lập trình dưới MCU thực hiện khá chính xác. Độ lệch chuẩn của các tín hiệu  $\omega_L$  và  $\omega_R$  sẽ thay đổi khi động cơ chạy ở những tốc độ khác nhau. Và độ lệch chuẩn không giống nhau giữa các động cơ, động cơ bên phải (màu xanh) dao động khá mạnh (do kết cấu cơ khí).

### 3.3. Tính toán góc yaw cho Robot từ IMU MPU9250

Robot được thiết kế để hoạt động với môi trường indoor, xưởng, nhà máy, những nơi sẽ không có tín hiệu GPS, và có nhiều sắt (Fe) từ lớn. Vì vậy chỉ cần sử dụng IMU 6DOF (03 của Accel và 03 của Gyro) là đủ. MPU6050 là một lựa chọn phù hợp, tuy nhiên MPU9250 9DOF ra đời sau có độ chính xác tốt hơn. Vì vậy, đề tài lựa chọn MPU9250<sup>9</sup> của TDK InvenSense.

Trong [26-28] các tác giả đã sử dụng các bộ lọc khác nhau để tính toán AHRS dựa trên các giá trị cảm biến từ IMU. Đáng chú ý nhất là 03 phương pháp phổ biến nhất đó là: Lọc Kalman, Madgwick [29] và Mahony [30]. Theo như bảng 1 [26, p.4, Table 1] thì Madgwick cho đáp ứng lọc là tốt nhất.

---

<sup>9</sup> <https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-9250/>

AHRS	roll (deg)	pitch (deg)	yaw (deg)
Basic	4.3135 / 0.0353	2.0937 / 1.9871	4.4740 / 0.0000
Madgwick	0.5896 / 0.0226	0.3123 / 0.3525	0.5787 / 0.0230
Mahony	0.3813 / 0.1233	0.6464 / 0.6550	0.4522 / 0.1238

Hình 3. 10 Kết quả đánh giá 03 bộ lọc cho IMU [26,p.4, table 1]

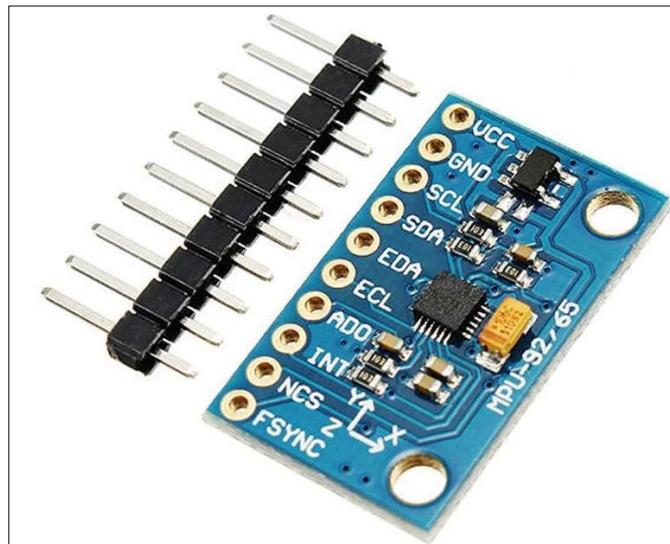
Kết quả tại [28, p.6, Table 2] thì bộ lọc EKF xử lý tốt nhất rồi đến Madgwick.

Euler angles [°]	EKF	Madgwick	Mahony
Roll	5.05	5.54	5.87
Pitch	3.24	3.93	4.53
Yaw	5.93	6.27	6.66

Table 2. Dynamic RMSE with noisy measurements.

Hình 3. 11 Đánh giá RMSE cho 03 bộ lọc cho IMU [28, p.6, table 2]

Madgwick và Mahony là 02 bộ lọc có thời gian xử lý tốt nhất, thích hợp cho nền tảng MCU (với tần số hoạt động thấp). Bên cạnh đó thư viện<sup>10</sup> cho 02 bộ lọc này được public rất nhiều. Vì vậy, để tài sử dụng bộ lọc Madgwick cho việc tính toán góc yaw từ IMU.



Hình 3. 12 Module IMU MPU9250

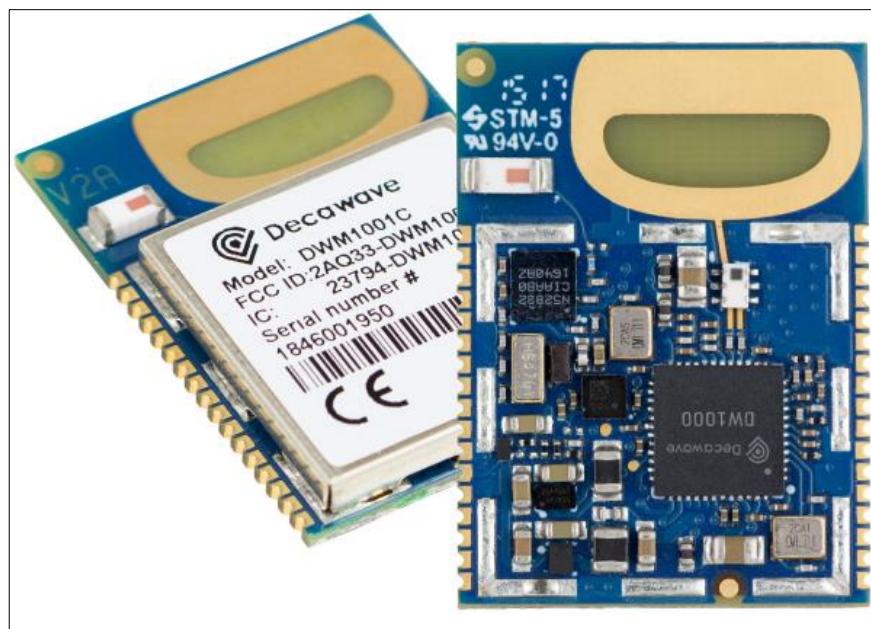
---

<sup>10</sup> Ref lib madgwick: <https://x-io.co.uk/open-source-imu-and-ahrs-algorithms/>

Trên robot được trang bị 01 Module IMU MPU9250, được kết nối với board nhúng Linux Jestson Nano qua chuẩn giao tiếp I2C ở tốc độ 400kHz. Sử dụng bộ lọc Madgwick với thời gian lấy mẫu 200Hz, lập trình bằng ngôn ngữ C++ và python. Kết quả đánh giá được giới thiệu ở chương 5.

#### 3.4. Nhận giá trị vị trí XY cho Robot từ UWB DWM1001C

Để tài sử dụng Module UWB DWM1001C<sup>11</sup> của hãng Decawave, sử dụng 06 Anchor để thiết lập không gian tọa độ và 01 Tag được gắn phía trên Robot để xác định tọa độ của Robot trong khu vực tọa độ của UWB. Với thời gian đáp ứng khoảng 100 ms (10Hz), sai số dao động trong khoảng 10cm<sup>12</sup>, tại một thời điểm có thể bắt đến 04 Anchor để tính toán ra vị trí. UWB rất thích hợp với những bài toán định vị local, indoor, ở những nơi mà sóng GPS không thể tới.



Hình 3. 13 Module UWB DWM1001C của Decawave

---

<sup>11</sup> <https://www.decawave.com/product/dwm1001-module/>

<sup>12</sup> Ref: <https://www.decawave.com/dwm1001/systemoverview/>, page 10, table 1.

### 3.5. Tính toán vị trí XY và hướng Yaw cho Robot từ biến Lidar

Robot sử dụng cảm biến RPLidar A1M8-R5<sup>13</sup> của hãng Salmtec với khoảng cách quét đến 12m, Tần số quét tối đa 10Hz với 8000 mẫu dữ liệu, độ phân giải góc đến 1 độ và độ chính xác về khoảng cách là 0.2 cm.



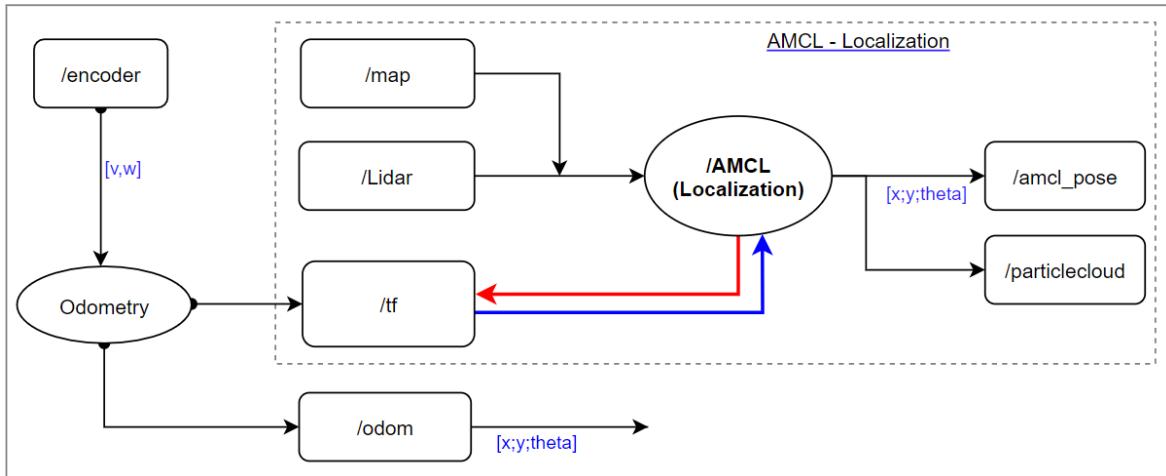
Hình 3. 14 Cảm biến RPLidar A1M8

Để có được vị trí và hướng của Robot trong môi trường hoạt động từ cảm biến Lidar, cần phải có bản đồ (/map) trước. Quá trình định vị này, đề tài sử dụng gói AMCL Localization<sup>14</sup> của ROS để thực hiện.

---

<sup>13</sup> Thông tin kỹ thuật về Lidar tại <https://www.slamtec.com/en/Lidar/A1>

<sup>14</sup> <http://wiki.ros.org/amcl>



Hình 3. 15 Sơ đồ gói AMCL Localization của ROS

Nội dung bản tin `/amcl_pose` bao gồm tọa độ  $[x; y; \theta]$ , kết quả này được dùng hợp nhất (fusion) trong thuật toán EKF. Tuy nhiên, dữ liệu từ `/amcl_pose` rất dễ bị sai khi thông tin từ odometry không chính xác như: bánh xe bị trượt, tốc độ di chuyển đọc về không đúng ..., Bên cạnh đó dữ liệu này đáp ứng rất chậm (<1Hz), gửi về không thường xuyên và dừng gửi hẳn khi Robot không di chuyển; Khi quá trình localization sai về vị trí và hướng, dữ liệu vẫn được gửi về. Mặc dù dễ bị sai trong quá trình localization, nhưng nếu quá trình ước lượng (estimate) này chính xác thì dữ liệu có được từ AMCL lại chính xác hơn UWB về tọa độ XY và chính xác hơn IMU về hướng (góc yaw của IMU bị trôi). Vì vậy, dữ liệu có được từ AMCL cần được xử lý trước khi đưa vào EKF để fusion và tận dụng góc hướng này (khi quá trình estimation chính xác) để bù trôi góc yaw của IMU. Nội dung này sẽ được trình bày tại chương 5.

## CHƯƠNG 4

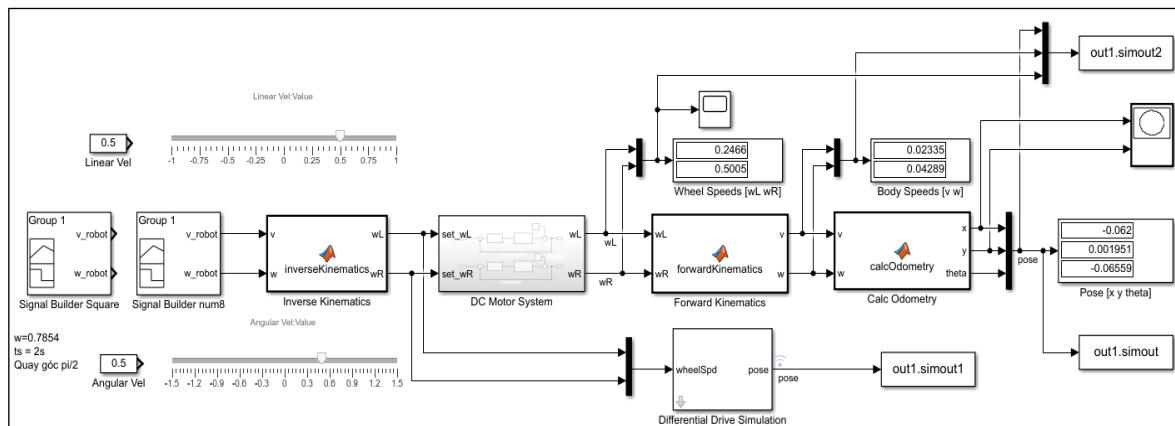
### MÔ PHỎNG HỆ THỐNG

#### 4.1. Mô phỏng mô hình Robot

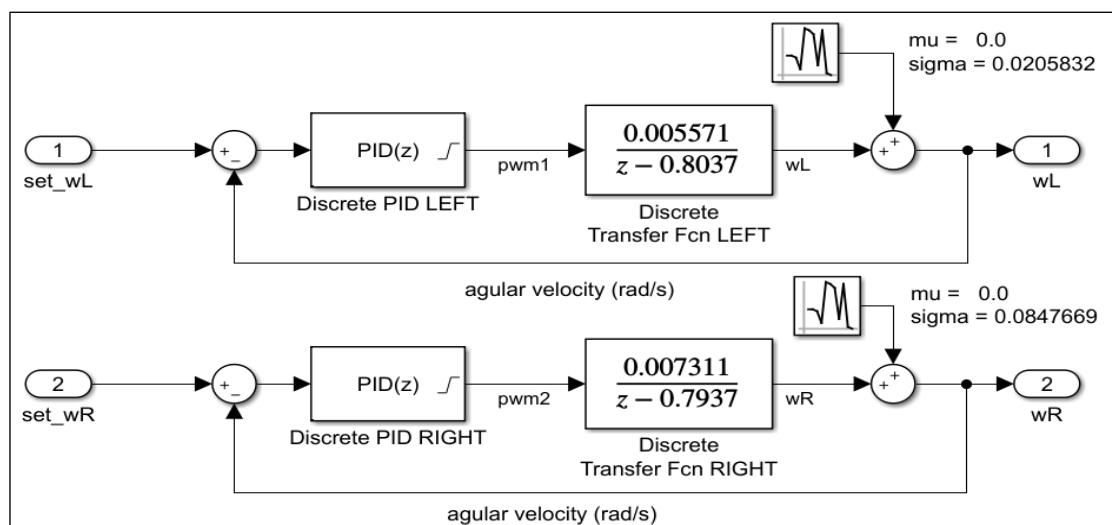
##### 4.1.1. Sơ đồ Simulink của Robot

Mô hình động học của Robot đã được trình bày ở phần (2.1), sử dụng các công thức (2.3) và (2.4) để tính động học thuận và động học ngược của robot. Đối với công thức (2.2) sử dụng để tính toán Odometry. Sơ đồ thí nghiệm simulink như hình 4.1 bên dưới. Thêm tín hiệu nhiễu hệ thống với giá trị có được khi thử PWM = 250 ở thử nghiệm (3.2.3) là:

$$\varepsilon_{w_L} \sim \mathcal{N}(0.0, 0.0205832^2); \quad \varepsilon_{w_R} \sim \mathcal{N}(0.0, 0.0847669^2);$$



Hình 4. 1 Sơ đồ simulink mô hình hóa hệ thống



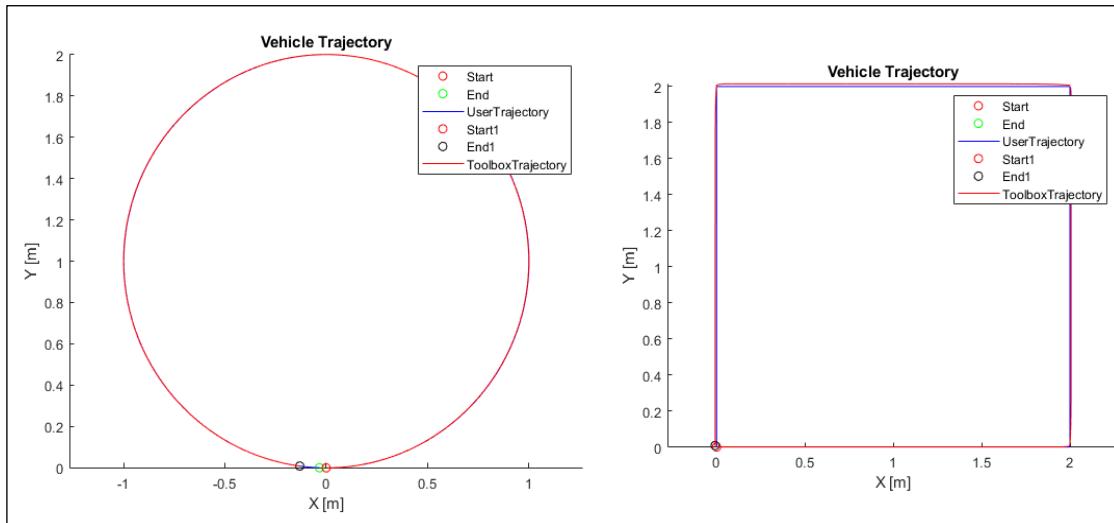
Hình 4. 2 Sơ đồ mô phỏng bên trong khối DC Motor System

Toàn bộ phần code của Matlab Function được giới thiệu ở Phụ lục 2, tiểu mục 1; riêng khối *Differential Drive Simulation*<sup>15</sup> sử dụng toolbox của Matlab, dùng để kiểm tra chéo với các khối đã được lập trình.

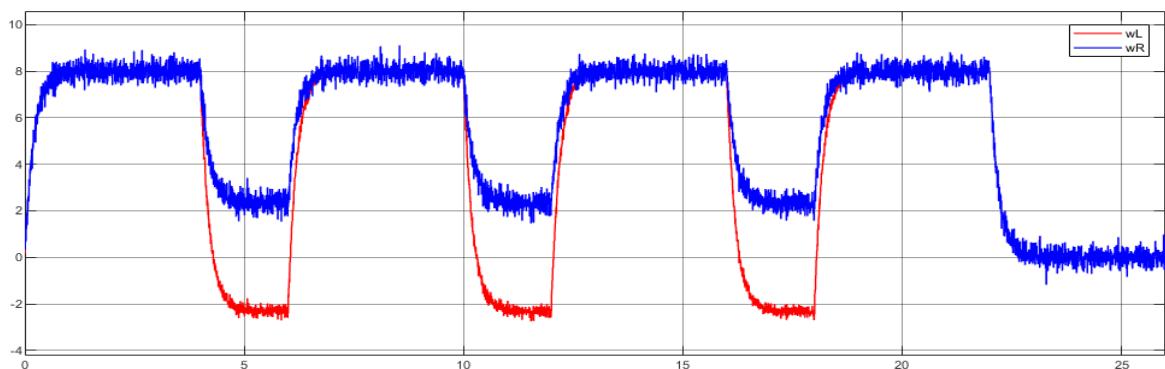
#### 4.1.2. Đánh giá mô hình mô phỏng và mô hình thực tế của robot

##### 4.1.2.1. Kiểm tra sự đúng đắn của mô hình simulink đã xây dựng

Cho mô hình chạy với 02 quỹ đạo khác nhau là vòng tròn và hình vuông.



Hình 4.3 Quỹ đạo mô phỏng của robot với toolbox



Hình 4.4 Vận tốc góc của 02 bánh xe khi quỹ đạo là hình vuông

**Nhận xét 1:** Mô hình xây dựng trên simulink đã chính xác. Đối với quỹ đạo là đường tròn thì mô hình robot sẽ đáp ứng chậm hơn một chút (điểm kết

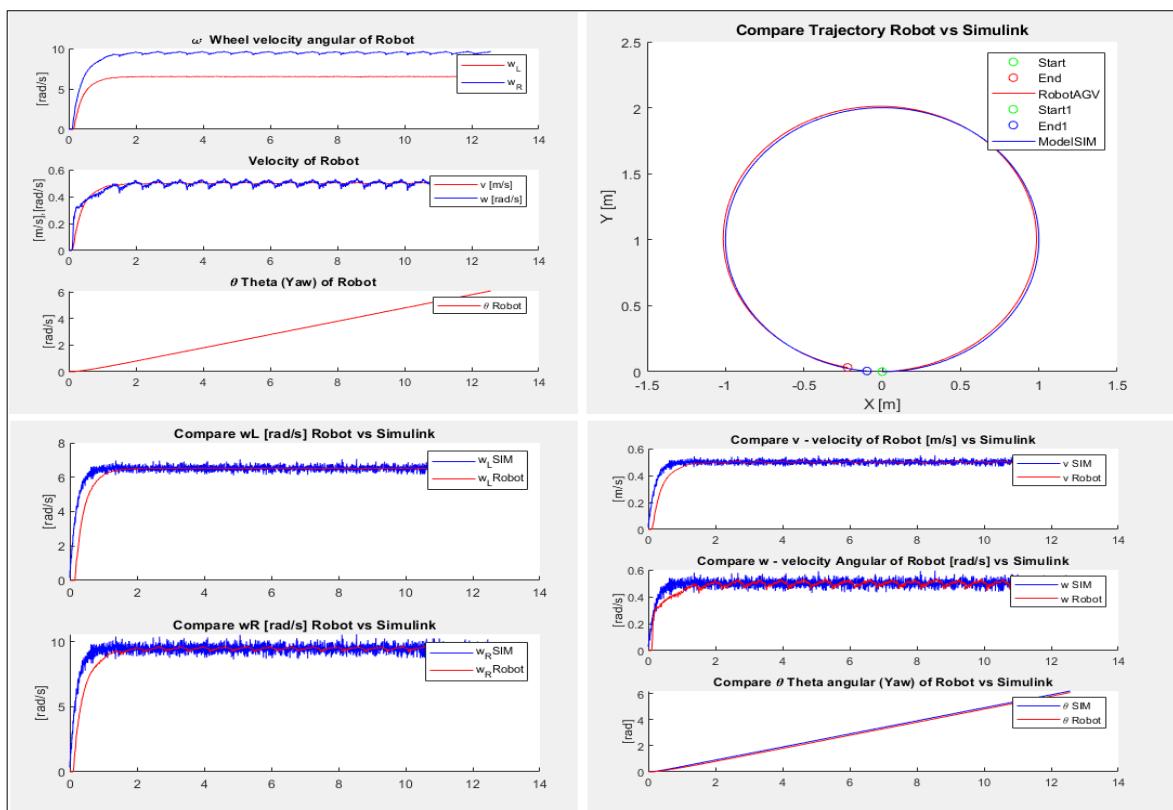
---

<sup>15</sup> Ref: <https://www.mathworks.com/matlabcentral/fileexchange/66586-mobile-robotics-simulation-toolbox>

thúc vòng tròn màu đen, không trùng với màu xanh). Còn đối với quỹ đạo hình vuông quỹ đạo robot bị bo tròn ở các góc, và quỹ đạo không hoàn toàn trùng khớp với mô phỏng sử dụng toolbox. Việc này hoàn toàn chính xác, toolbox coi đối tượng như 1 điểm, còn mô hình robot có thời gian đáp ứng mỗi khi đặt tốc độ mới. (phụ thuộc vào bộ điều khiển và kết cấu vật lý).

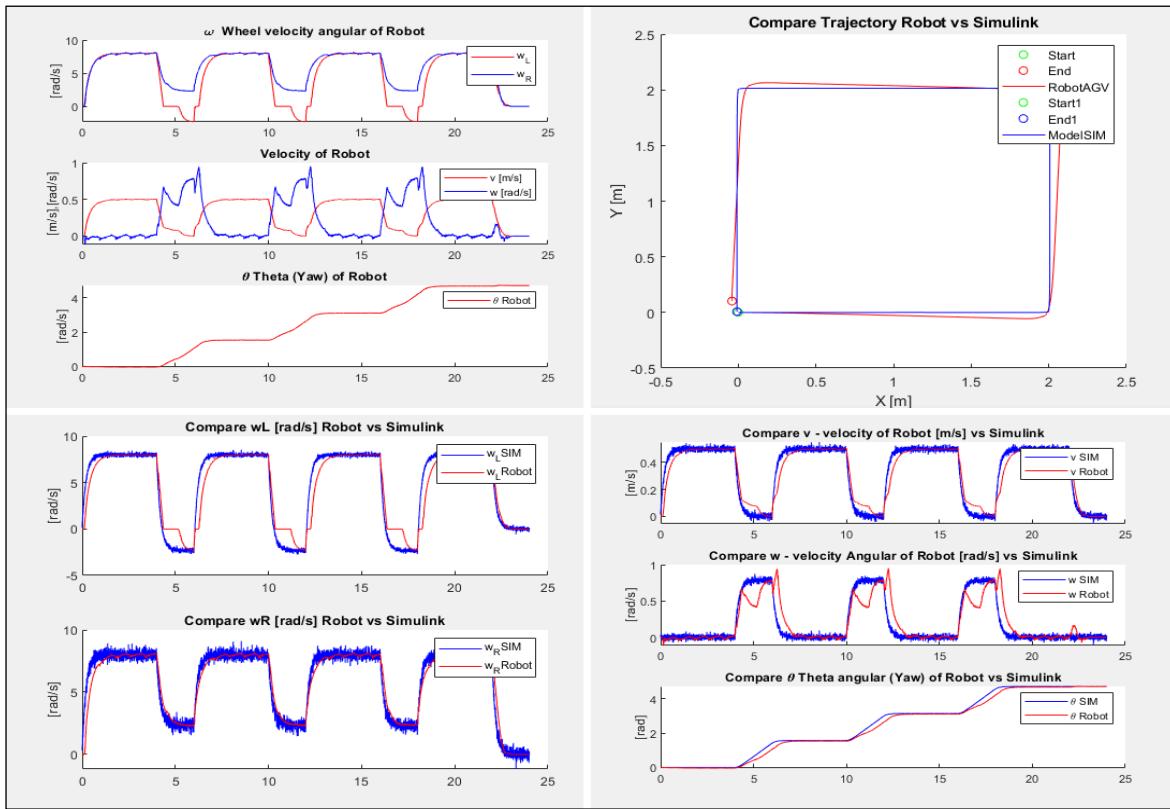
**Nhận xét 2:** Khi thêm nhiều vào cả 02 điểm đo tốc độ bánh xe, quỹ đạo vẫn được tính toán chính xác. Nhiều đo theo phân phối chuẩn và khá nhỏ trong khi thời gian xử lý của hệ thống khá nhanh (5ms) nên không bị ảnh hưởng đến việc tính toán quỹ đạo. Toàn bộ tập dữ liệu đánh giá được lưu lại trong quá trình kiểm tra, và được liệt kê ở phụ lục 1, tiểu mục 3 (các file data).

#### 4.1.2.2. Đánh giá đáp ứng của robot theo các quỹ đạo khác nhau

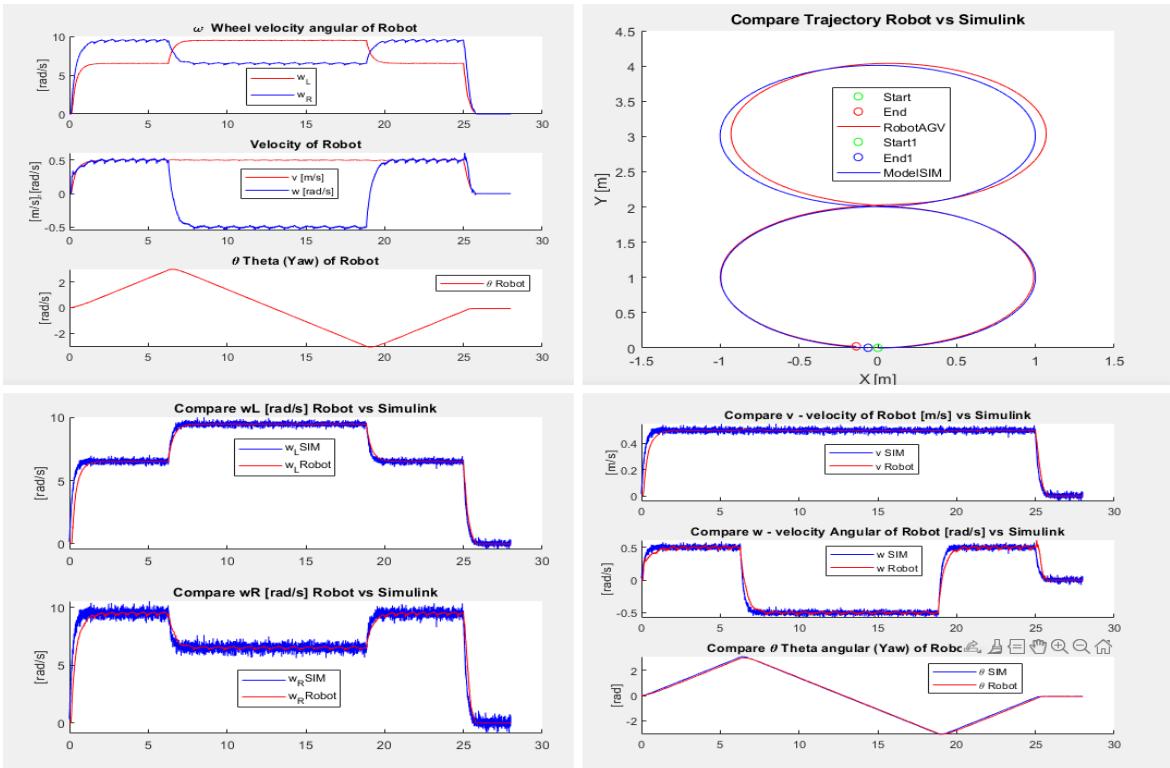


Hình 4.5 Đáp ứng của robot khi chạy theo một vòng tròn

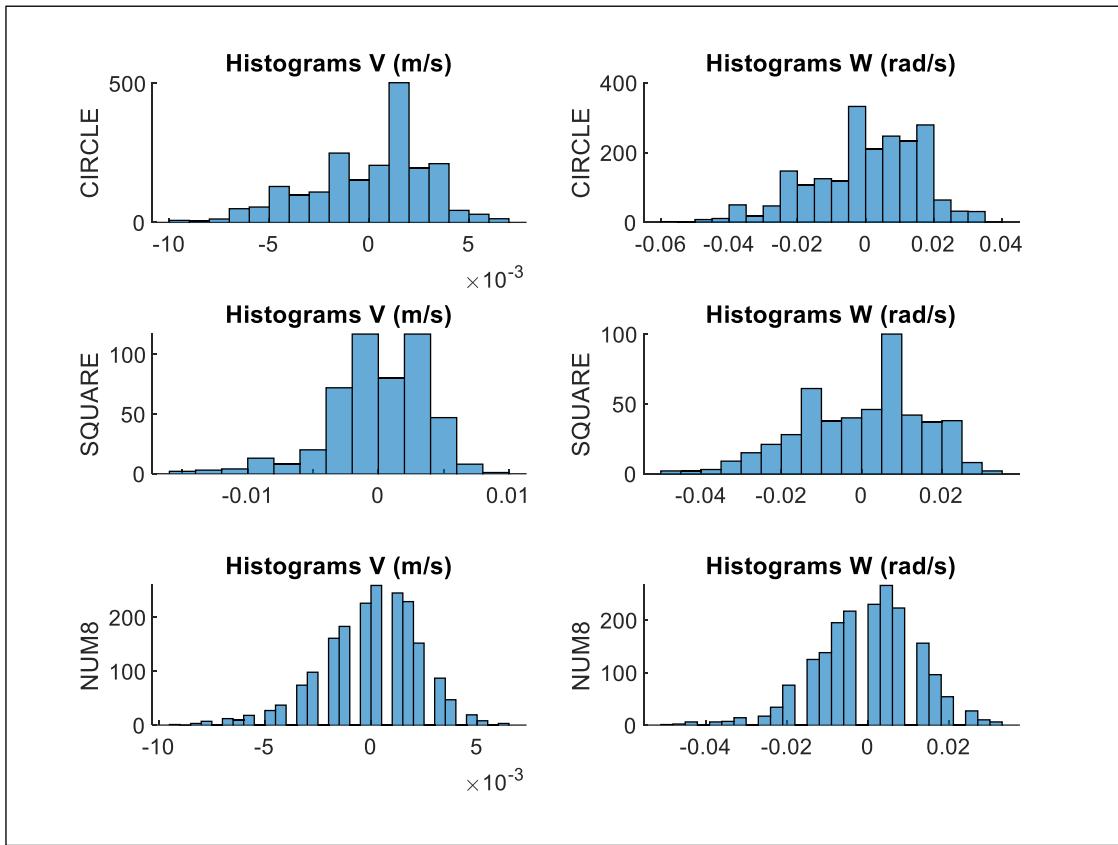
## Chương 4: MÔ PHỎNG HỆ THỐNG



Hình 4. 6 Đáp ứng của robot khi chạy theo một hình vuông



Hình 4. 7 Đáp ứng của robot khi chạy theo một hình số tám



Hình 4.8 Biểu đồ phân phối nhiễu của  $v$  [m/s] và  $\omega$  [rad/s]

Bảng 4.1 Phương sai nhiễu đo lường  $v$  và  $\omega$  của robot

Quỹ đạo	Phương sai $v$ [m/s]	Phương sai $\omega$ [rad/s]
Hình tròn	$0.00292152^2$	$0.0158411^2$
Hình vuông	$0.00369928^2$	$0.0153147^2$
Hình số 8	$0.00230419^2$	$0.0122931^2$

**Nhận xét :** Hệ thống mô phỏng và đáp ứng thật bên dưới Robot gần như giống nhau về các hành vi điều khiển, cũng như quỹ đạo di chuyển. Điều này chứng tỏ hệ thống được mô hình hóa chính xác. Tuy nhiên hệ thống này đang được thử ở điều kiện lý tưởng (không tải, 2 bánh xe động cơ được quay tự do không có ma sát với mặt sàn). Khi robot hoạt động trong một không gian xác định sẽ tồn tại những yếu tố bất định, ảnh hưởng trực tiếp đến các phép

đo làm cho phần tính toán vị trí, hướng của robot bị cộng dồn sai số, dẫn đến sai cả về vị trí và phương hướng.

### 4.2. Mô phỏng quá trình hợp nhất dữ liệu với bộ lọc EKF

#### 4.2.1. Xây dựng chương trình mô phỏng

Thuật toán EKF cho việc hợp nhất các dữ liệu đo được như thông tin đo được từ robot thông qua encoder để tính toán quỹ đạo dịch chuyển, quá trình này gọi là Dead Reckoning<sup>16</sup>, và các thông tin khác như UWB (tọa độ x,y), IMU (góc  $\theta$ ) được xây dựng trên ngôn ngữ thông dịch python<sup>17</sup> chạy trên máy tính nhúng Linux (Jetson Nano). Vì vậy, phần mô phỏng này sẽ được xây dựng dựng trên chính ngôn ngữ python để tiện cho quá trình đánh giá thử nghiệm. Riêng đối với cảm biến Lidar dữ liệu mô phỏng (tạo giả) được sử dụng bằng chính dữ liệu UWB và IMU hợp lại. Sử dụng RMSE<sup>18</sup> làm phương pháp đánh giá sai số.

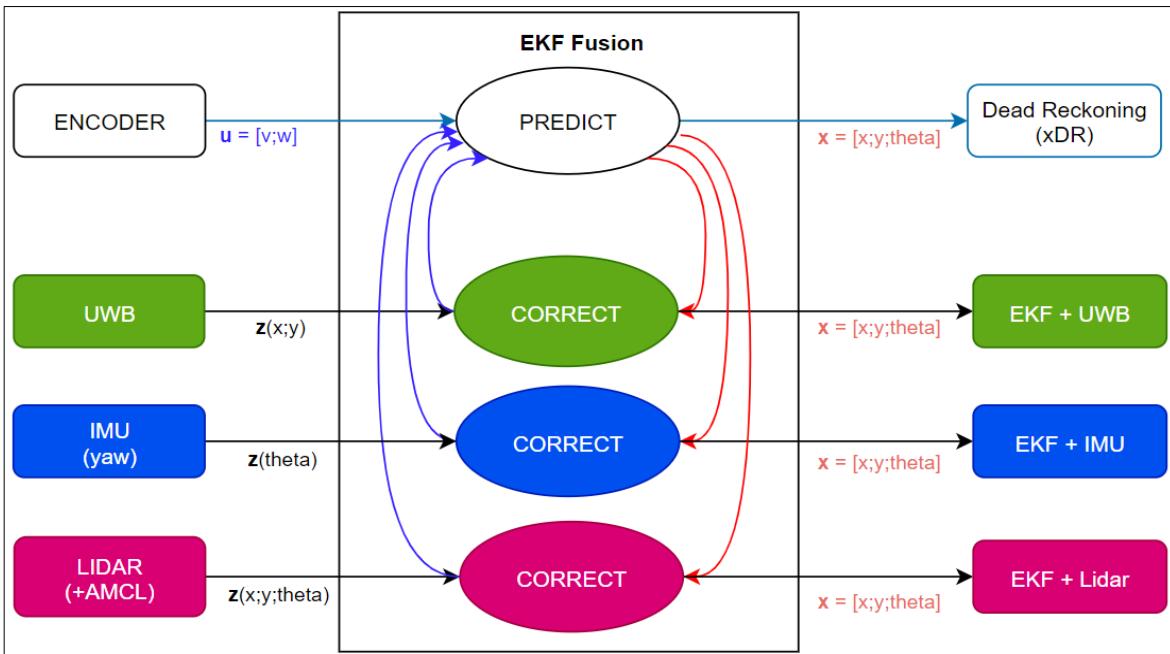
Quá trình mô phỏng chia thành 03 trường hợp độc lập như hình 4.9, bao gồm quá trình EKF fusion với UWB (phép đo  $[x; y]$ ); EKF fusion với góc yaw từ IMU (phép đo  $[\theta]$ ) và cuối cùng là EKF fusion với phép đo đầy đủ  $[x; y; \theta]$  từ UWB kết hợp với IMU hoặc quá trình AMCL với Lidar.

---

<sup>16</sup> [https://en.wikipedia.org/wiki/Dead\\_reckoning](https://en.wikipedia.org/wiki/Dead_reckoning)

<sup>17</sup> <https://www.python.org/>

<sup>18</sup> [https://en.wikipedia.org/wiki/Root-mean-square\\_deviation](https://en.wikipedia.org/wiki/Root-mean-square_deviation)



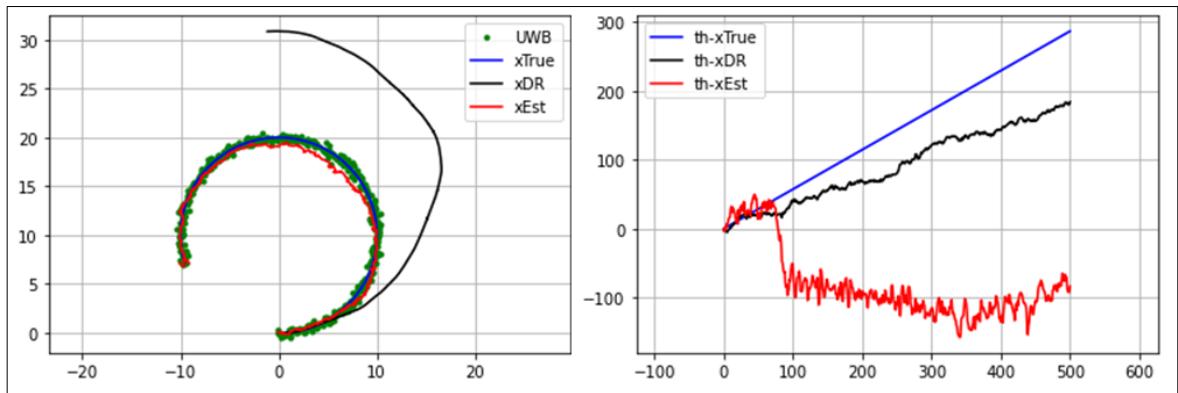
Hình 4. 9 Các phương thức kết hợp EKF cho mô phỏng thử nghiệm

#### 4.2.2. EKF với phép đo XY từ UWB Tag

Giả sử thông tin thử nghiệm cho mô phỏng như bảng 4.2, thuật toán EKF sẽ hợp nhất dữ liệu Encoder (trong quá trình predict) với thông tin đo lường từ UWB. Kết quả được so sánh với quá trình Dead Reckoning (kí hiệu xDR).

Bảng 4. 2 Thông tin thử nghiệm mô phỏng EKF với UWB

STT	Thông tin cảm biến và nhiễu	Ghi chú
1	Tín hiệu điều khiển, cố định với $v = 1.0$ m/s; $\omega = 0.1$ rad/s, với nhiễu $\sim \mathcal{N} \begin{bmatrix} 1.0^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	
2	UWB, với nhiễu $\sim \mathcal{N} \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	Sử dụng hàm <i>np.random.randn</i>
3	Ma trận $\mathbf{Q} = \begin{bmatrix} 0.1^2 & 0 & 0 \\ 0 & 0.1^2 & 0 \\ 0 & 0 & 0.52359^2 \end{bmatrix}$	$30^\circ = 0.52359$ rad
4	Ma trận $\mathbf{R}_{UWB} = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	



Hình 4.10 Kết quả hợp nhất Odometry và UWB

Bảng 4.3 So sánh RMSE giữa Dead Reckoning và EKF với UWB

Dead Reckoning (xDR)	EKF Fusion + UWB
7.954419586017109	2.666140930543576

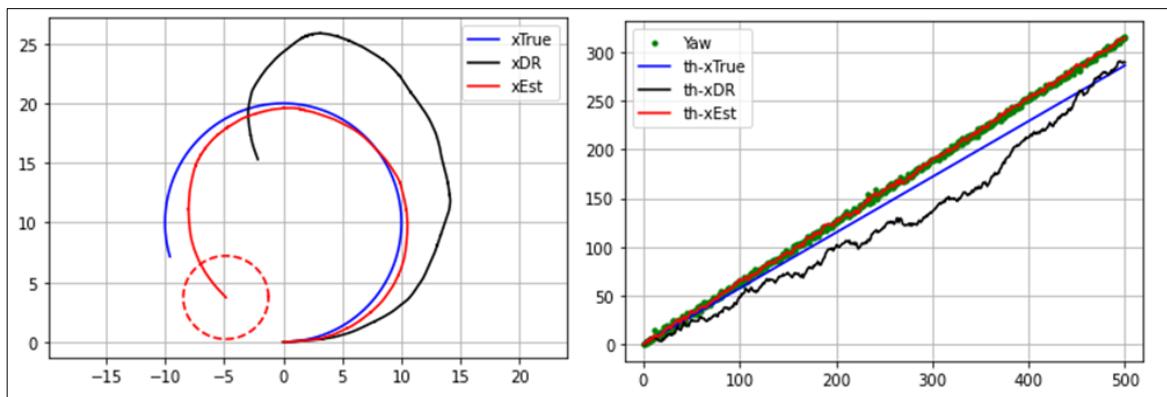
**Nhận xét:** Giá trị ước lượng XY có được từ EKF (xEst - màu đỏ) cơ bản bám theo tín hiệu đo lường UWB (chấm tròn nhỏ - màu xanh green) và gần với quỹ đạo di chuyển thật (xTrue – màu xanh blue) trong khi giá trị ước tính từ quá trình Dead Reckoning (xDR) đã đi ra xa với sai số lớn. Tuy nhiên, quá trình EKF này chỉ giúp cho giá trị ước lượng XY (xEst) về gần với giá trị thật (xTrue), còn giá trị về góc hướng (th-xEst, góc hướng robot - màu đỏ) không bám theo góc hướng thật (th-xTrue) và thay đổi không có quy luật. Mặc dù vậy quá trình EKF khi có phép đo từ UWB mang đến độ chính xác tốt hơn nhiều với xDR, khi mà tổng sai số cả quỹ đạo (RMSE) của EKF nhỏ hơn quá trình xDR.

#### 4.2.3. EKF với phép đo góc yaw từ IMU

Trong phần này chỉ sử dụng EKF với phép đo lường từ IMU (cụ thể là góc hướng  $\theta$ ), với các thông tin mô phỏng thử nghiệm như sau:

Bảng 4. 4 Thông tin mô phỏng EKF với IMU

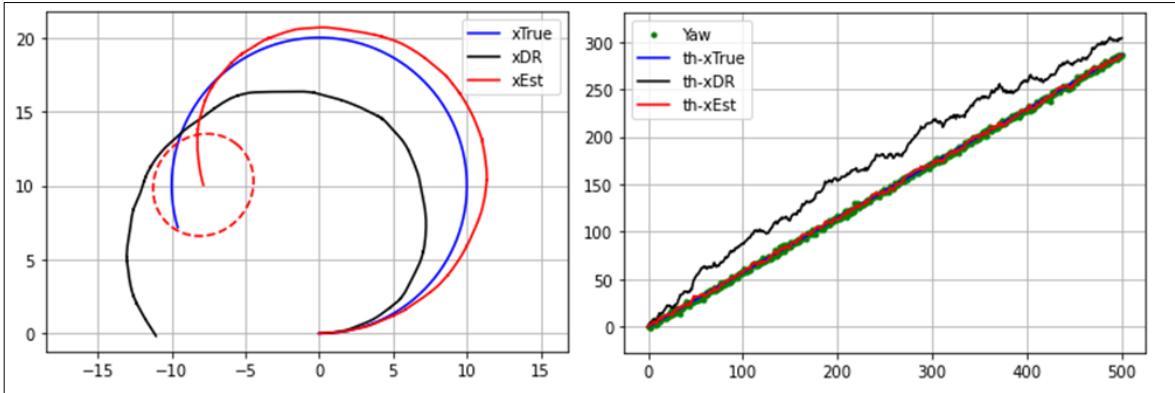
STT	Thông tin cảm biến và nhiễu	Ghi chú
1	Tín hiệu điều khiển, cố định với $v = 1.0 \text{ m/s}$ ; $\omega = 0.1 \text{ rad/s}$ , với nhiễu $\sim \mathcal{N} \begin{bmatrix} 1.0^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	
2	IMU, với nhiễu $\mathcal{N} [0.1745^2]$ Độ trôi góc yaw: $0.6^0/\text{giây}$	$10^0 = 0.1745 \text{ rad}$ Sử dụng hàm <code>np.random.randn</code>
3	Ma trận $\mathbf{Q} = \begin{bmatrix} 0.1^2 & 0 & 0 \\ 0 & 0.1^2 & 0 \\ 0 & 0 & 0.52359^2 \end{bmatrix}$	$30^0 = 0.52359 \text{ rad}$
4	Ma trận $\mathbf{R}_{IMU} = [2^2]$	



Hình 4. 11 EKF kết nối với góc yaw (có trôi) từ IMU

Bảng 4. 5 So sánh RMSE giữa Dead Reckoning và EKF với IMU

Dead Reckoning (xDR)	EKF Fusion + IMU
4.482774656682976	1.4009653650456957



Hình 4.12 EKF kết hợp với góc yaw (không bị trôi) từ IMU

Bảng 4.6 So sánh RMSE giữa Dead Reckoning và EKF với IMU (không trôi)

Dead Reckoning (xDR)	EKF Fusion
4.394423913922045	0.9941149335497097

**Nhận xét 1:** Trong cả 02 trường hợp khi góc yaw có trôi hoặc không trôi thì góc hướng ước tính của robot có được từ EKF (th-xEst, đường màu đỏ) vẫn bám theo giá trị góc yaw từ IMU (yaw – chấm tròn nhỏ, màu xanh green). Mặc dù không đặt niềm tin nhiều vào IMU, bằng việc cho  $\mathbf{R}_{IMU} = [4]$  nhưng quá trình ước lượng vẫn bám theo IMU. Bởi vì giá trị mô phỏng  $\mathbf{Q}_\theta = [0.52359^2]$  (nhiều đến  $30^0$ ) cũng không đặt niềm tin nhiều về hướng của quá trình predict hệ thống. Điều này chứng tỏ giá trị  $\mathbf{R}$  và  $\mathbf{Q}$  quyết định rất nhiều về kết quả của quá trình ước lượng. Bằng cách thay đổi giá trị  $\mathbf{R}_{IMU}$  lên rất lớn và  $\mathbf{Q}_\theta$  xuống rất nhỏ, có thể tính được góc trôi yaw (từ IMU) khi Robot đứng yên.

**Nhận xét 2:** Mặc dù góc hướng của Robot đã được bám chính xác theo góc yaw của IMU, tuy nhiên ước lượng về vị trí XY (xEst – màu đỏ) của EKF cũng không được cải thiện là bao nhiêu. Vị trí quỹ đạo di chuyển không bám theo quỹ đạo thật (xTrue – màu xanh) nhưng vẫn tốt hơn nhiều so với quỹ

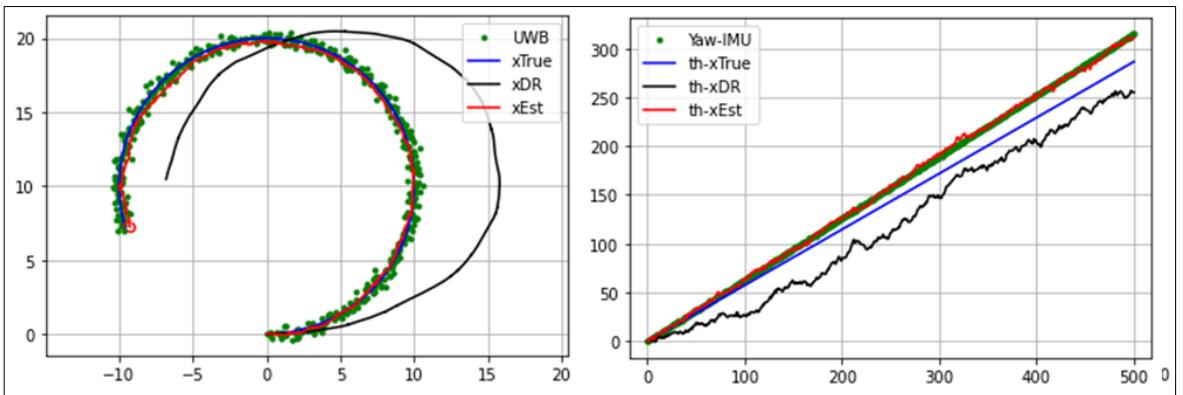
đạo được tính từ Dead Reckoning (xDR – màu đen), với tổng sai số quỹ đạo RMSE tốt hơn quá trình xDR.

#### 4.2.4. EKF với phép đo XY-Theta

Phép đo đầy đủ  $[x; y; \theta]$  có thể đến từ việc hợp nhất dữ liệu của UWB và IMU hoặc kết quả của quá trình AMCL Localization với Lidar. UWB có tần số cập nhật 10Hz (100ms) và IMU có tần số cập nhật 200Hz (50ms), nhanh hơn 20 lần so với UWB. Vì vậy, trong pha hiệu chỉnh của EKF (correct/update) với UWB sẽ thêm phép đo  $[\theta]$  đến từ IMU, để quá trình ước lượng được chính xác nhất.

Bảng 4. 7 Thông tin mô phỏng EKF với phép đo XY-Yaw

STT	Thông tin cảm biến và nhiễu	Ghi chú
1	Tín hiệu điều khiển, cố định với $v = 1.0 \text{ m/s}$ ; $\omega = 0.1 \text{ rad/s}$ , với nhiễu $\sim \mathcal{N} \begin{bmatrix} 1.0^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	
2	Phân phối nhiễu đo $\sim \mathcal{N} \begin{bmatrix} 0.5^2 & 0 & 0 \\ 0 & 0.5^2 & 0 \\ 0 & 0 & 0.5236^2 \end{bmatrix}$	
3	Ma trận $\mathbf{Q}_{xy\theta} = \begin{bmatrix} 0.1^2 & 0 & 0 \\ 0 & 0.1^2 & 0 \\ 0 & 0 & 0.52359^2 \end{bmatrix}$	$30^\circ = 0.52359 \text{ rad}$
4	Ma trận $\mathbf{R}_{xy\theta} = \begin{bmatrix} 1^2 & 0 & 0 \\ 0 & 1^2 & 0 \\ 0 & 0 & 1^2 \end{bmatrix}$	



Hình 4.13 Kết quả EKF hợp nhất với phép đo đầy đủ  $[x; y; \theta]$

Bảng 4.8 Kết quả đánh giá RMSE khi sử dụng EKF với phép đo  $[x; y; \theta]$

Dead Reckoning (xDR)	EKF Fusion
3.8924378650458094	0.2623247685302017

**Nhận xét:** Tư thế robot ước tính của quá trình EKF Fusion (đường xEst và đường th-xEst, màu đỏ) đã bám gần đúng nhất với quỹ đạo di chuyển thật (xTrue và th-xTrue, màu xanh blue). Quá trình EKF (xEst) với ước lượng về tư thế chính xác hơn nhiều, với tổng sai số RMSE của cả quỹ đạo nhỏ hơn nhiều so với quá trình Dead Reckoning (xDR, màu đen). Tuy nhiên, trong thực tế cần phải thử nghiệm và hiệu chỉnh 02 ma trận **R** và **Q** để đạt được kết quả mong muốn.

#### 4.3. Kết luận phần mô phỏng hệ thống

Quá trình mô phỏng chỉ mang tính chất kiểm tra độ chính xác của thuật toán đã xây dựng. Thông tin giả sử cho quá trình mô phỏng EKF không sát với thực tế của mô hình, các tham số được lựa chọn một cách có ý đồ để thể hiện tốt nhất quá trình hợp nhất (fusion) cho từng cảm biến đơn lẻ. Vì vậy, những bộ tham số này không được sử dụng lại trong quá trình thực nghiệm.

Việc hiệu chỉnh các ma trận **R** và **Q** sẽ được tiến hành bằng thực nghiệm thông qua quá trình chạy thử nghiệm. Tùy những trường hợp mà giá trị của **R** và **Q** sẽ khác nhau. Ở chương 5 sẽ trình bày về quá trình này.

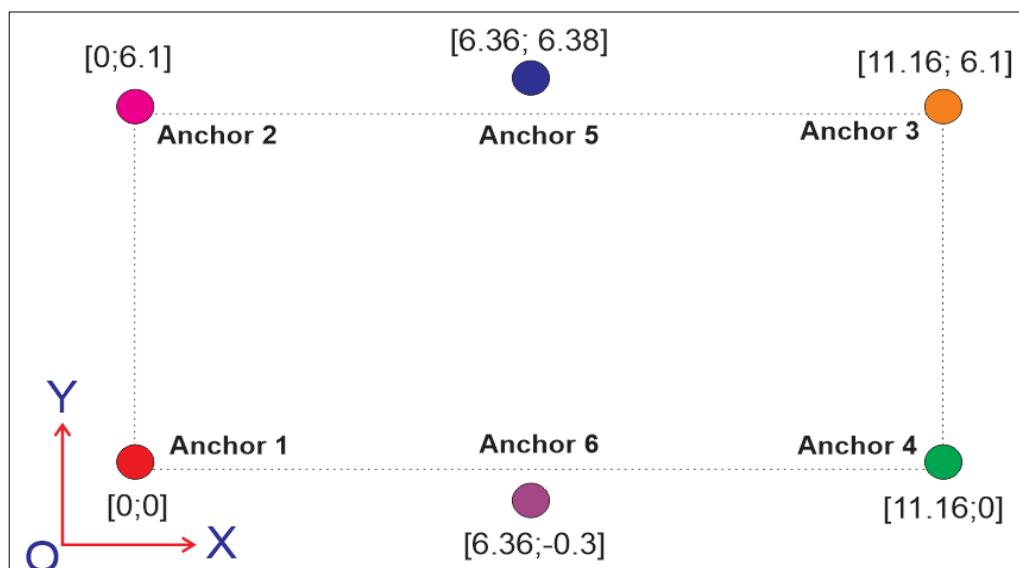
## CHƯƠNG 5

### THỰC NGHIỆM TỔNG HỢP CẢM BIẾN CHO ĐỊNH VỊ ROBOT DI ĐỘNG

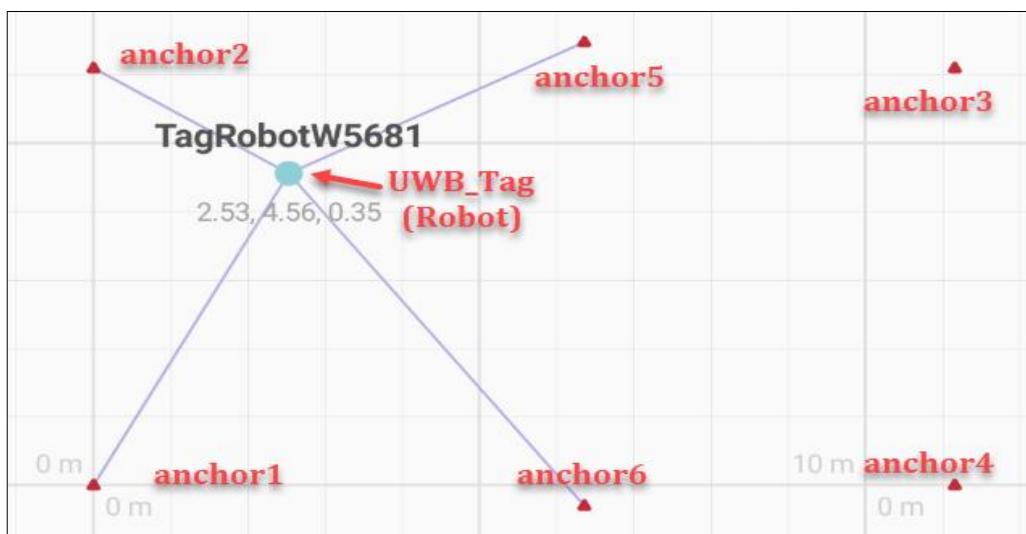
#### 5.1. Kiểm tra nhiễu UWB và độ trôi góc yaw từ MPU9250

##### 5.1.1. Kiểm tra độ chính xác của UWB

Môi trường thử nghiệm sử dụng 07 thiết bị UWB DWM1001C, trong đó 06 thiết bị làm Anchor treo trên tường ở độ cao 180cm và 01 thiết bị làm Tag được gắn trên Robot. Phạm vi thử nghiệm 10x12m được bố trí như sau:

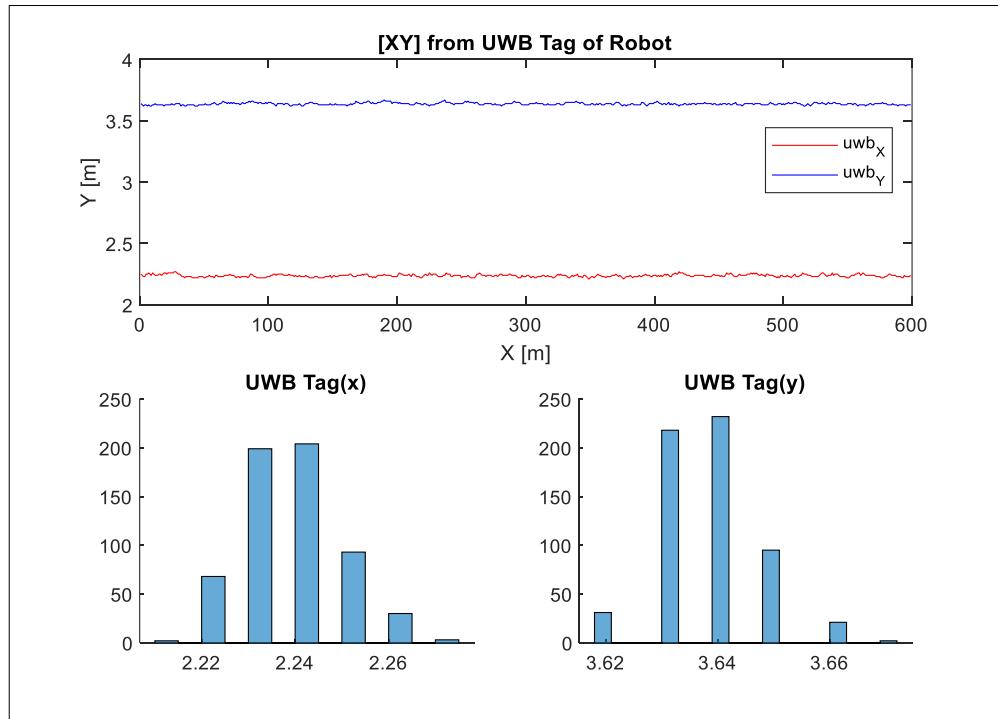


Hình 5. 1 Sơ đồ bố trí các thiết bị UWB làm Anchor

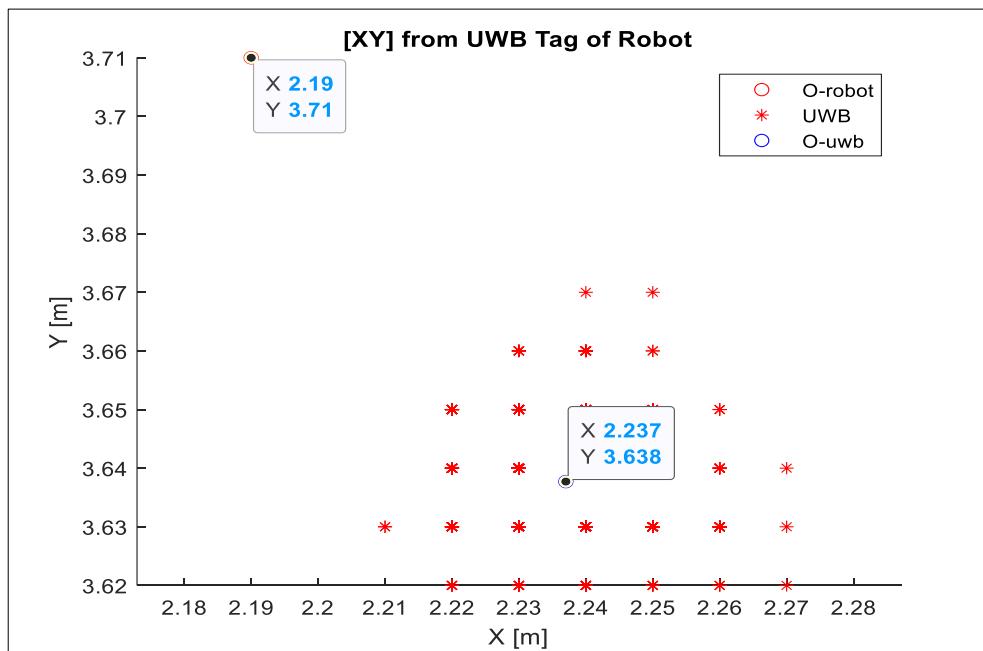


Hình 5. 2 UWB Tag Robot trong không gian UWB Anchor

Phương pháp kiểm tra như sau: đặt Robot chính xác ở vị trí đã biết trước, tại vị trí  $XY = [2.19; 3.71]$  (m). Tiến hành lấy mẫu liên tục trong thời gian 298 giây (với 599 mẫu dữ liệu) với tần số update của UWB là 2Hz.



Hình 5.3 Giá trị đo được và biểu đồ phân phối nhiễu của UWB



Hình 5.4 Tọa độ của Robot và phân phối đo lường của UWB

Kết quả được như sau:

$$\mathbf{X}_{UWB} \approx \mathcal{N}[\mu = 2.23701; \sigma = 0.0106283]$$

$$\mathbf{Y}_{UWB} \approx \mathcal{N}[\mu = 3.63771; \sigma = 0.00921379]$$

 **Nhận xét:** Tọa độ thực của Robot không nằm trong vùng phân phối đo lường của UWB như hình 5.4; Sai lệch giữa vị trí của robot (0-robot) với trung bình của điểm đo (0-uwb) theo XY = [-0.0470; 0.0723] (m). Sai lệch trong lần thử nghiệm này < 10 cm. Phương sai nhiễu đo theo cả 2 chiều XY là khoảng 01cm. Như vậy, sai số đo lường từ UWB bao gồm cả sai lệch vị trí thực và nhiễu đo. Tuy nhiên kết quả trên là ở một phạm vi nhỏ với tần số update của UWB thấp (2Hz). Khi sử dụng trong một phạm vi lớn hơn với tần số cập nhập của UWB là 10Hz thì sai số này sẽ lớn hơn nhiều. Để đơn giản, chọn ma trận

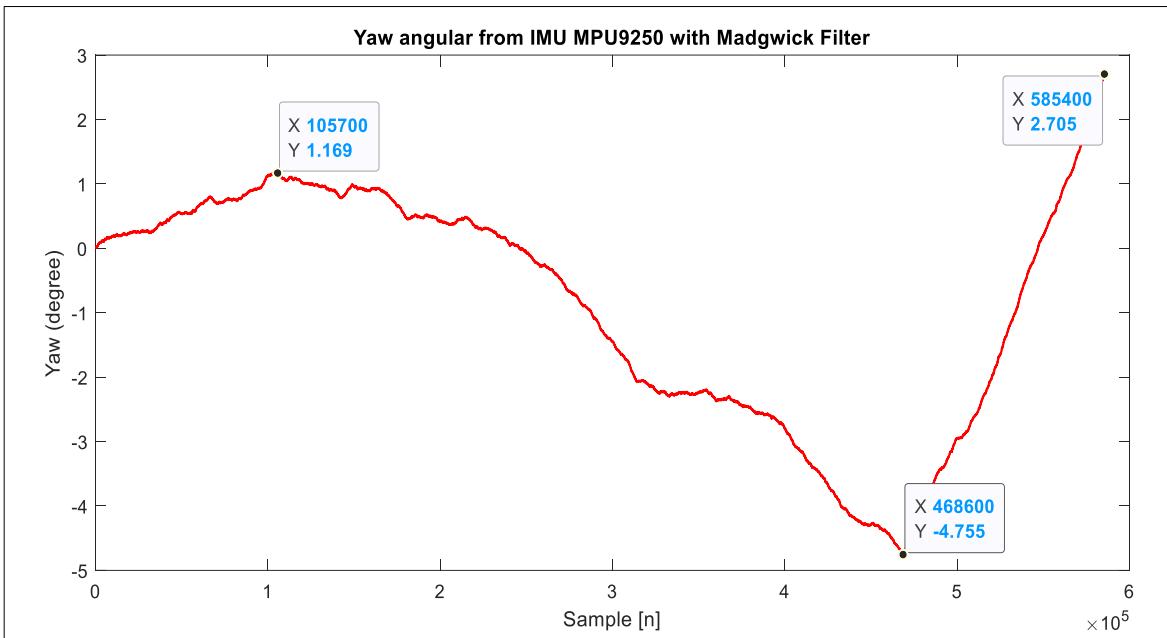
$$R_{UWB} = \begin{bmatrix} 0.316^2 & 0 \\ 0 & 0.316^2 \end{bmatrix}, \text{ tương ứng với nhiễu đo có phương sai bằng } 1\text{cm.}$$

### 5.1.2. Kiểm tra sự chính xác và độ trôi của góc yaw của IMU

Phương pháp kiểm tra như sau: đặt Robot đứng yên trên mặt sàn phẳng, cấp nguồn điện cho Robot khởi động và thực hiện xong quá trình calib bias cho IMU (thời gian calib khoảng 15 giây). Quá trình kiểm tra chia thành 02 giai đoạn là: kiểm tra độ trôi và độ chính xác về hướng của góc yaw.

#### 5.1.2.1. Kiểm tra độ trôi của góc yaw

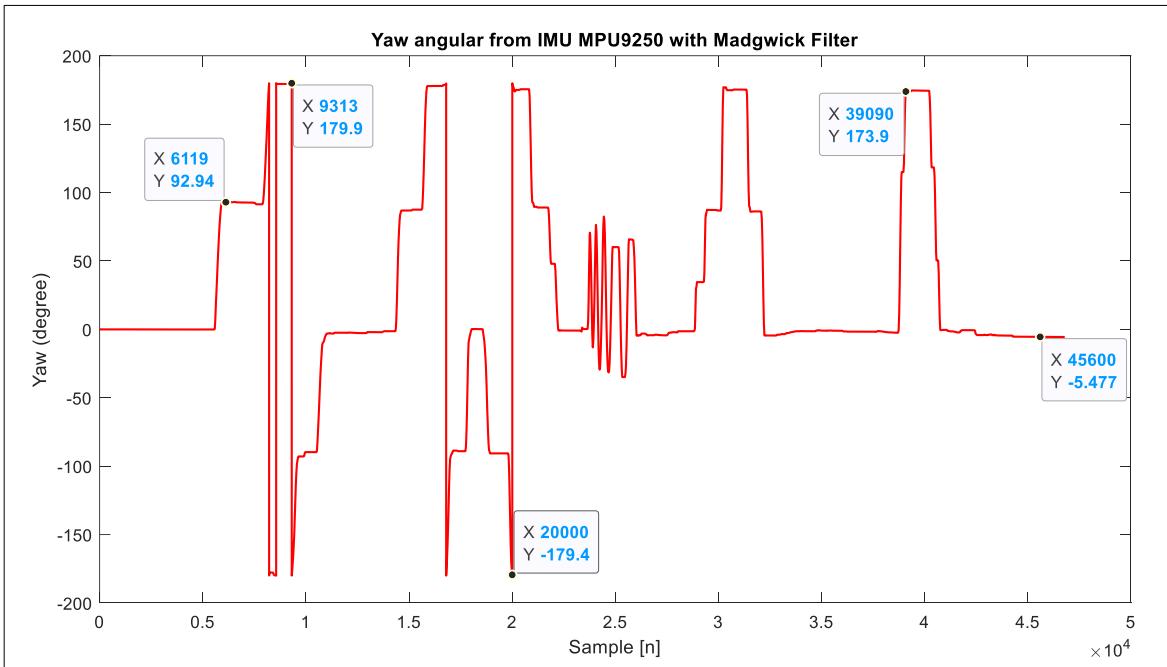
Tổng thời gian kiểm tra là: 01 giờ 08 phút 44 giây (4124 giây) với 585456 mẫu dữ liệu. Theo như hình 5.5, trong khoảng 12 phút đầu góc yaw trôi từ 0 lên  $1.2^\circ$  sau đó giảm dần đến  $-4.75^\circ$  tại phút thứ 58 và đảo hướng tăng dần đến  $2.7^\circ$  tại phút 68. Như vậy, sự trôi của góc yaw không theo một chiều và không có quy luật. Đánh giá chung độ trôi tĩnh của góc yaw khoảng  $5^\circ$  trong thời gian 1 giờ.



Hình 5. 5 Đánh giá sự trôi góoc Yaw từ IMU

### 5.1.2.2. Kiểm tra độ chính xác của góoc yaw từ IMU

Thử độ chính xác của góoc yaw IMU bằng cách quay trở robot liên tục ở các góoc  $0^\circ$ ;  $\pm 90^\circ$ ;  $\pm 180^\circ$  và xoay liên tục trong phạm vi nhỏ tạo nhiễu. Tổng thời gian kiểm tra là 5 phút 11 giây (311 giây) với 46792 mẫu dữ liệu.



Hình 5. 6 Giá trị đo lường góoc yaw khi quay trở ở các góoc khác nhau

 **Nhận xét:** ban đầu Robot ở hướng  $0^{\circ}$ , sau khi quay trở liên tục góc yaw của IMU cơ bản vẫn bám theo được góc hướng thật của robot. Góc hướng thực cuối cùng của Robot khoảng  $0^{\circ}$ , góc yaw đo được từ IMU là  $-5.4^{\circ}$ , với tốc độ xoay liên tục như vậy, sai lệch là  $-5.4^{\circ}$  có thể chấp nhận được.

## 5.2. Triển khai bộ lọc EKF cho những quỹ đạo di chuyển định sẵn

### 5.2.1. Phương pháp thực hiện

Robot được đặt trong khung tham chiếu của UWB, môi trường thử nghiệm như hình 5.1. Sử dụng các dụng cụ đo lường (thước kẹp, thước dây, bút lông) tiến hành đo đạc và vẽ các quỹ đạo định sẵn mà robot sẽ di chuyển.

Đưa Robot về chế độ không nhận lệnh điều khiển, mục đích thả tự do cho 02 bánh xe (vì không có bộ điều khiển tác động). Tiến hành di chuyển xe theo quỹ đạo đã định sẵn, với vận tốc đều khoảng 0.2 m/s.

Toàn bộ dữ liệu trong quá trình robot di chuyển được lưu trữ lại trong rosbag (một định dạng của ROS), sử dụng data này để thực nghiệm quá trình tối ưu cho 02 ma trận  $\mathbf{R}$  và  $\mathbf{Q}$ .

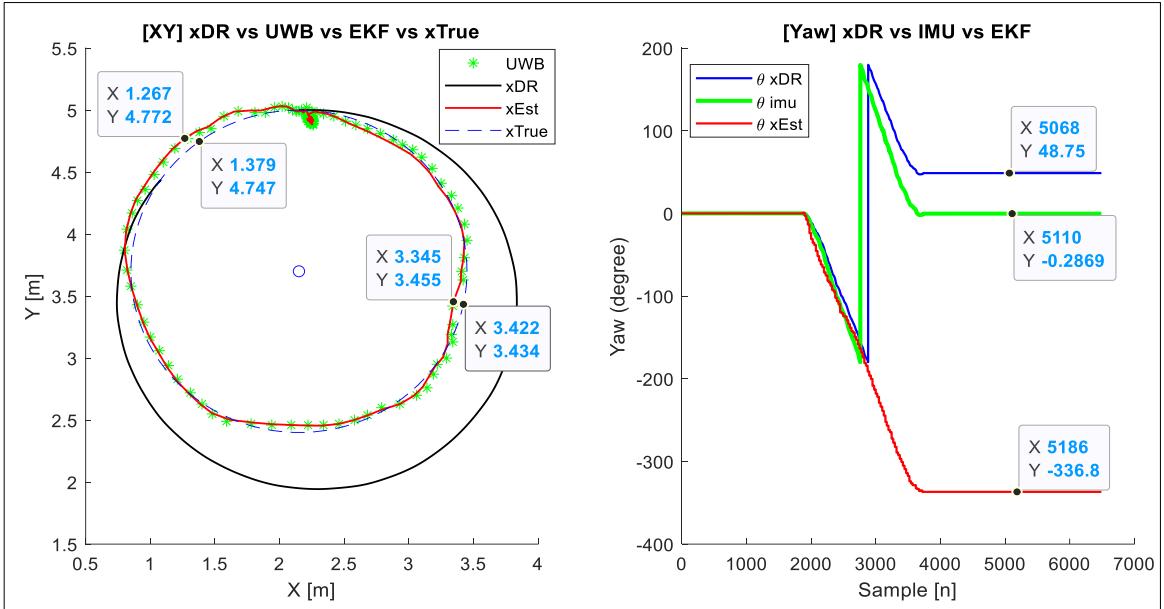
### 5.2.2. Quỹ đạo di chuyển theo vòng tròn

#### 5.2.2.1. Bộ lọc EKF với phép đo UWB

Robot được đặt chính xác tại vị trí  $XY\theta = [2.15; 5.0; 0.0]$ , và di chuyển theo quỹ đạo vòng tròn, có tâm  $\mathbf{O}_{XY} = [2.15; 3.70]$  (m) với bán kính  $R = 1.30$ m. Các ma trận  $\mathbf{R}$ ,  $\mathbf{Q}$  được lựa chọn sau khi thực nghiệm nhiều lần như sau:

Bảng 5. 1 Thông số R, Q thử nghiệm cho EKF với UWB

STT	Ma trận $\mathbf{R}, \mathbf{Q}$	Ghi chú
1	Ma trận $\mathbf{Q} = \begin{bmatrix} 0.01^2 & 0 & 0 \\ 0 & 0.01^2 & 0 \\ 0 & 0 & 0.02^2 \end{bmatrix}$	
2	Ma trận $\mathbf{R}_{UWB} = \begin{bmatrix} 0.31^2 & 0 \\ 0 & 0.31^2 \end{bmatrix}$	



Hình 5.7 So sánh quỹ đạo di chuyển khi sử dụng EKF với UWB

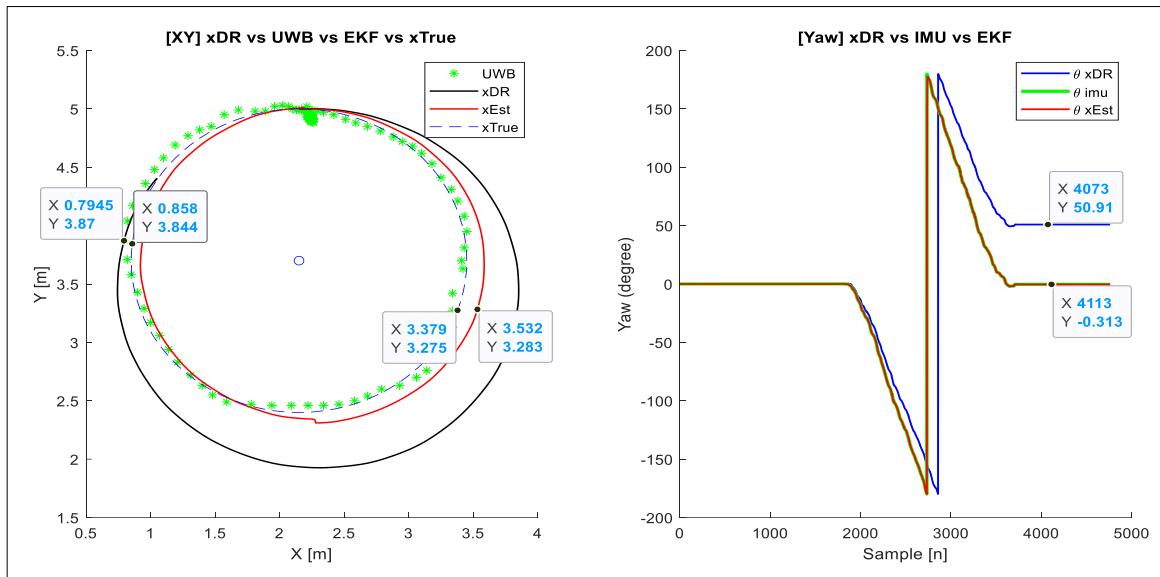
**Nhận xét:** Vị trí XY của Robot được ước tính qua bộ lọc EKF với phép đo từ UWB (xEst, màu đỏ), cơ bản đi theo các điểm đo từ UWB (UWB, kí hiệu \*, màu xanh green). Sai số lớn nhất của các điểm đo UWB so với quỹ đạo di chuyển thật (xEst, màu đỏ) là 11.2 (cm). Góc hướng ước tính của Robot không bám theo được góc hướng thật ( $\theta_{imu}$ , màu xanh green).

### 5.2.2.2. Bộ lọc EKF với phép đo IMU

Quỹ đạo di chuyển giống như trên, các ma trận R và Q được đặt lại như sau:

Bảng 5.2 Thông số R, Q thử nghiệm cho EKF với IMU

STT	Ma trận $R, Q$	Ghi chú
1	Ma trận $Q = \begin{bmatrix} 0.01^2 & 0 & 0 \\ 0 & 0.01^2 & 0 \\ 0 & 0 & 0.02^2 \end{bmatrix}$	Khi $v, \omega \neq 0.0$
2	Ma trận $Q = \begin{bmatrix} 0.001^2 & 0 & 0 \\ 0 & 0.001^2 & 0 \\ 0 & 0 & 0.001^2 \end{bmatrix}$	Khi $v = \omega = 0.0$
3	Ma trận $R_{IMU} = [10^8]$	Khi $v = \omega = 0.0$
4	Ma trận $R_{IMU} = [0.01]$	Khi $v, \omega \neq 0.0$

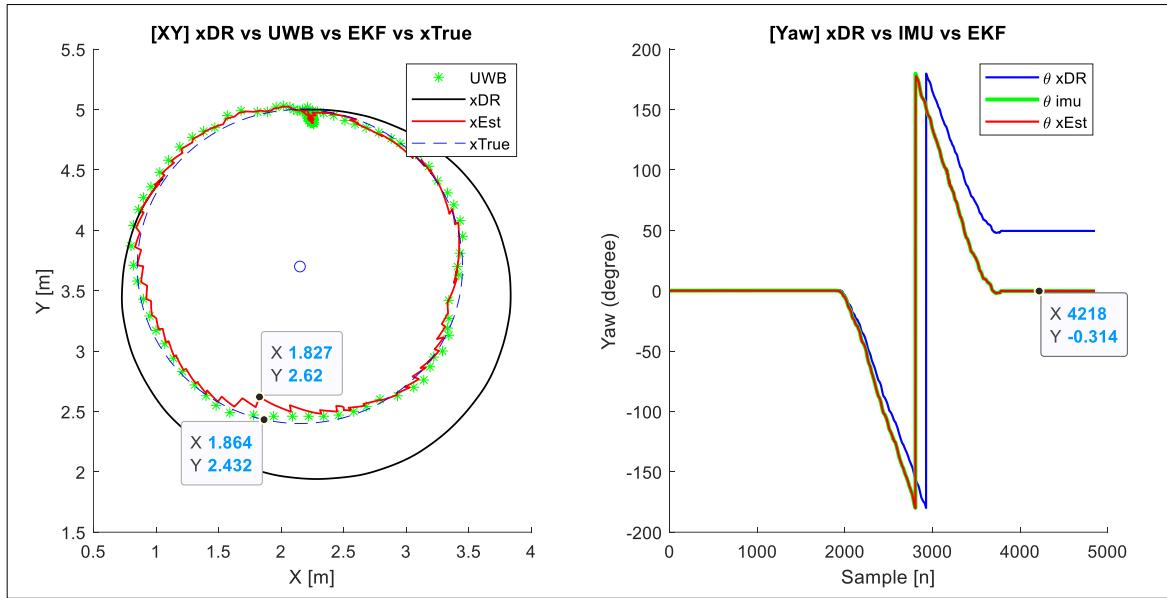


Hình 5.8 So sánh quỹ đạo di chuyển khi sử dụng EKF với IMU

**Nhận xét:** Góc hướng ước lượng ( $\theta_{xEst}$ , màu đỏ) bám theo đúng với góc hướng di chuyển thật ( $\theta_{imu}$ , màu xanh green). Tuy nhiên quỹ đạo di chuyển XY ước lượng (xEst) không đúng với quỹ đạo di chuyển thật (xTrue), sai số lớn nhất là 15,3 cm. Mặc dù vậy, quỹ đạo ước lượng này (xEst) vẫn tốt hơn rất nhiều so với quỹ đạo di chuyển khi tính bằng quá trình Dead Reckoning (xDR, màu đen).

### 5.2.2.3. Bộ lọc EKF với 02 phép đo độc lập từ UWB và IMU

Quỹ đạo di chuyển như trên, bộ tham số  $\mathbf{R}$  và  $\mathbf{Q}$  sử dụng theo bảng 5.2 khi tín hiệu đo lường đến từ UWB và sử dụng theo bảng 5.3 khi tín hiệu đo lường đến từ IMU. Hai quá trình EKF này được thực hiện độc lập và hiệu chỉnh (correct/update) chung một giá trị ước lượng về tư thế của Robot ( $XY\theta$ ).



Hình 5.9 Quỹ đạo di chuyển khi EKF với UWB và EKF với IMU

**Nhận xét:** Tư thế ước tính ( $x_{\text{Est}}$ ,  $\theta_{x_{\text{Est}}}$ ) đã đúng hơn về vị trí XY và chính xác về góc hướng (bám theo góc của  $\theta_{\text{imu}}$ ). Tuy nhiên quỹ đạo di chuyển XY còn bị răng cưa (không đúng với thực tế di chuyển của robot), sai số ước tính lớn nhất đến 18cm. Cần có 01 phương pháp để giá trị ước tính ( $x_{\text{Est}}$ ) được trơn mịn và chính xác hơn.

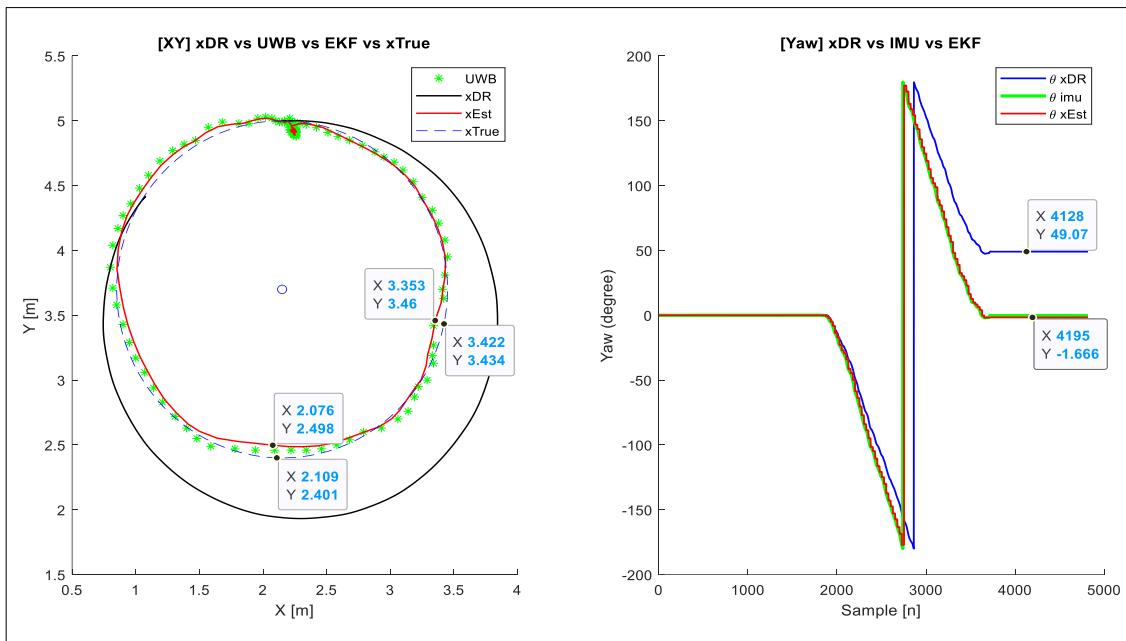
#### 5.2.2.4. Bộ lọc EKF với phép đo từ UWB kết hợp IMU

Phép đo lường được kết hợp giữa UWB (phép đo  $xy$ ) và IMU (phép đo  $\theta$ ) thành một phép đo duy nhất  $XY\theta$ . Quá trình correct/update chỉ thực hiện khi dữ liệu UWB được cập nhập mới. Thông số R và Q như sau:

Bảng 5.3 Thông số R và Q khi EKF với UWB hợp nhất IMU

STT	Ma trận $R, Q$	Ghi chú
1	Ma trận $Q = \begin{bmatrix} 0.01^2 & 0 & 0 \\ 0 & 0.01^2 & 0 \\ 0 & 0 & 0.02^2 \end{bmatrix}$	
2	Ma trận $R_{UWB\_IMU} = \begin{bmatrix} 0.31^2 & 0 & 0 \\ 0 & 0.31^2 & 0 \\ 0 & 0 & 10^4 \end{bmatrix}$	Khi $v = \omega = 0.0$

3	Ma trận $\mathbf{R}_{UWB\_IMU} = \begin{bmatrix} 0.31^2 & 0 & 0 \\ 0 & 0.31^2 & 0 \\ 0 & 0 & 0.001^2 \end{bmatrix}$	Khi $v, \omega \neq 0.0$
---	---	--------------------------

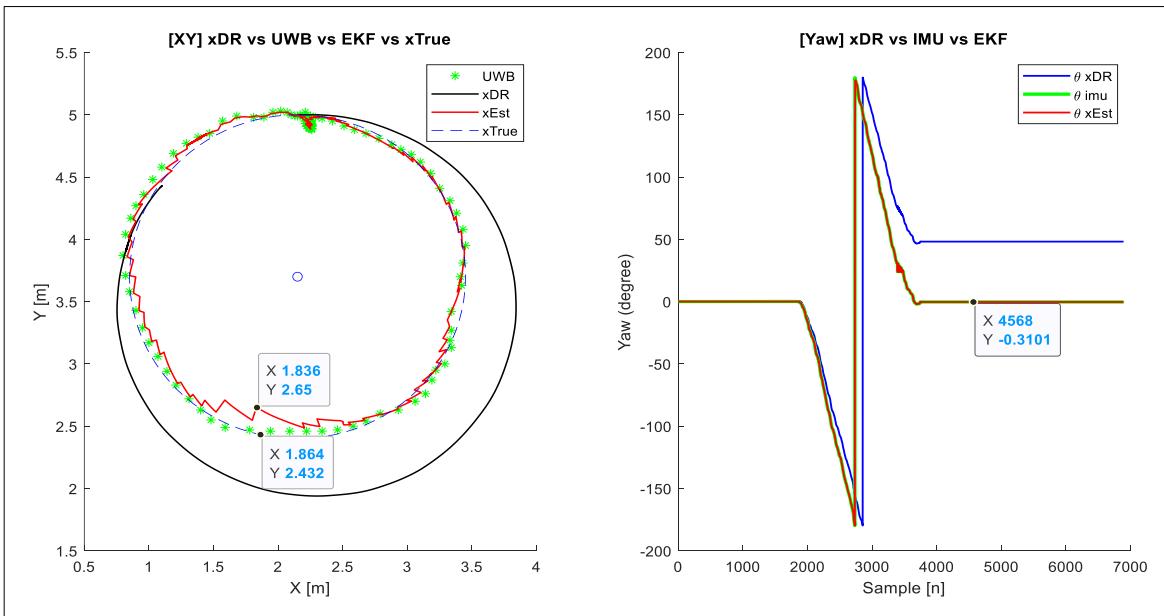


Hình 5. 10 Quỹ đạo di chuyển khi EKF với UWB+IMU

**Nhận xét:** Góc hướng ước tính ( $\theta_{xEst}$ ) chính xác với góc từ IMU ( $\theta_{imu}$ ) và quỹ đạo di chuyển XY ước tính (xEst) đã trơn, mịn và chính xác hơn nhiều. Sai số lớn nhất về vị trí ước tính so với quỹ đạo di chuyển thật (xTrue) là 9.7 cm.

#### 5.2.2.5. Bộ lọc EKF với phép đo từ UWB kết hợp IMU và IMU

Trong trường hợp này, bộ EKF sẽ thực hiện quá trình correct 02 lần độc lập cho một ước tính về robot (xEst) duy nhất. Lần thứ nhất correct với dữ liệu đo lường từ IMU, và lần thứ hai correct với dữ liệu đo lường từ UWB đã hợp nhất với góc yaw từ IMU (phép đo  $XY\theta$ ). Các tham số R và Q không thay đổi.



Hình 5.11 Quỹ đạo di chuyển khi EKF với IMU và EKF với UWB+IMU

**Nhận xét:** Trong thử nghiệm lần này, tư thế ước tính của robot (xEst) không tốt hơn so với 02 lần thử nghiệm trước. Góc hướng ước tính ( $\theta_{xEst}$ ) có một điểm lạ thường và sai số XY lớn nhất của xEst so với quỹ đạo di chuyển thật (xTrue, màu xanh blue) là 21.8 cm. Như vậy, phương pháp hợp nhất này không phù hợp để sử dụng.

#### 5.2.2.6. Kết luận thử nghiệm quỹ đạo vòng tròn

Thông qua quá trình thử nghiệm nhiều lần bằng cách thay đổi nội dung (giá trị) của các ma trận **R** và **Q** hệ thống cơ bản đã đáp ứng được yêu cầu đề ra khi chạy thử nghiệm với quỹ đạo tròn. Đây cũng chính là bộ tham số tối ưu nhất trong những lần thử nghiệm. Giá trị tham số này sẽ được sử dụng xuyên suốt trong các quỹ đạo khác.

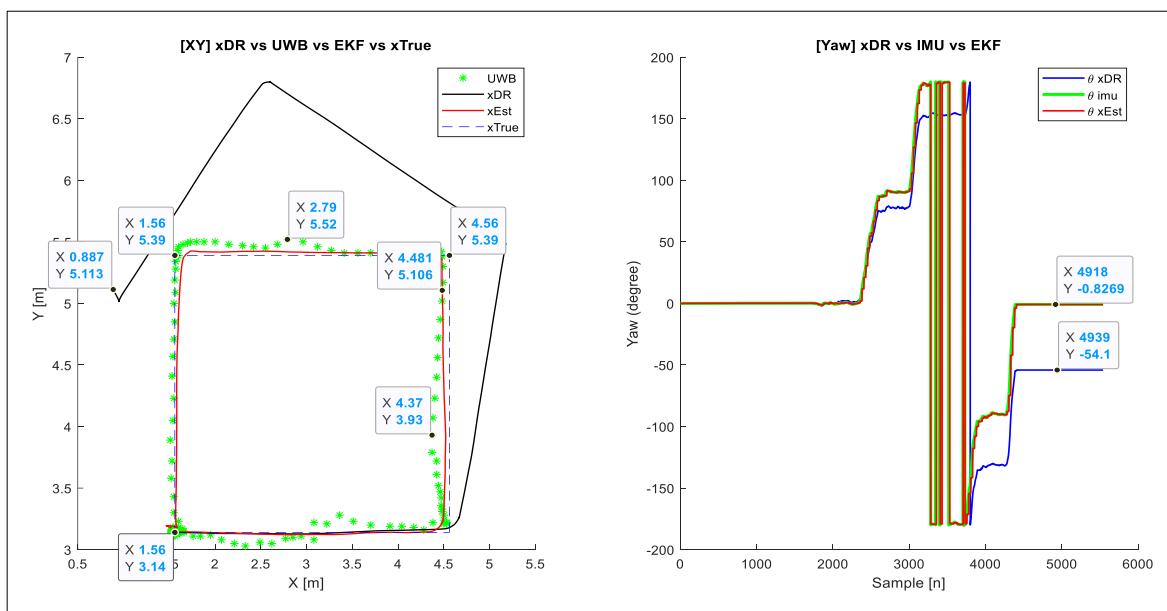
Kết quả hợp nhất dữ liệu trong trường hợp mục 5.2.2.4 (EKF với phép đo UWB hợp nhất IMU) cho kết quả ước lượng tốt nhất. Vì vậy, để tài sử dụng

phương pháp này cho quá trình correct tư thế robot khi có phép đo đến từ UWB (giá trị góc yaw của IMU sẽ được EKF fusion chung trong UWB).

### 5.2.3. Quỹ đạo di chuyển theo các góc vuông

#### 5.2.3.1. Quỹ đạo di chuyển theo hình chữ nhật

Cho robot di chuyển theo quỹ đạo hình chữ nhật có kích thước  $3.0 \times 2.25\text{m}$  tại vị trí cho trước  $XY\theta = [1.56; 3.14; 0.0]$ . Sử dụng bộ lọc EKF trong trường hợp hợp nhất dữ liệu với phép đo đầy đủ  $XY\theta$  của UWB và IMU. Thông số cho bộ lọc như bảng 5.4 mục 5.2.2.4

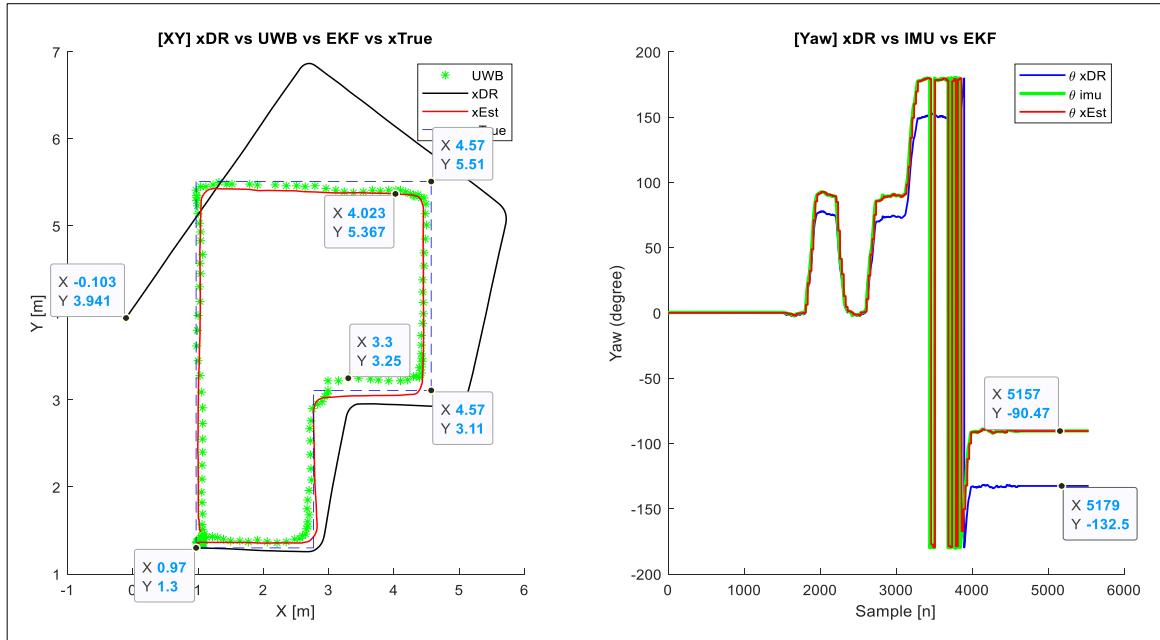


Hình 5. 12 EKF với phép đo UWB kết hợp IMU quỹ đạo hình chữ nhật

**Nhận xét:** Tư thế ước lượng của Robot cả về hướng ( $\theta_{xEst}$ ) và vị trí ( $xEst$ ) gần chính xác với quỹ đạo di chuyển thật ( $xTrue$ ). Sai số đo lường lớn nhất của UWB trong trường hợp này là 19cm. Sai số ước lượng qua EKF lớn nhất là 11cm. Như vậy quá trình EKF fusion đã khắc phục được những lỗi đo và đưa giá trị ước lượng hệ thống về gần nhất với giá trị thật.

### 5.23.2. Quỹ đạo di chuyển theo hình chữ P

Cho robot di chuyển theo quỹ đạo hình chữ P định sẵn tại vị trí cho trước  $XY\theta = [0.97; 1.30; 0.0]$ . Các tham số của bộ lọc EKF tương tự như trường hợp quỹ đạo hình chữ nhật.



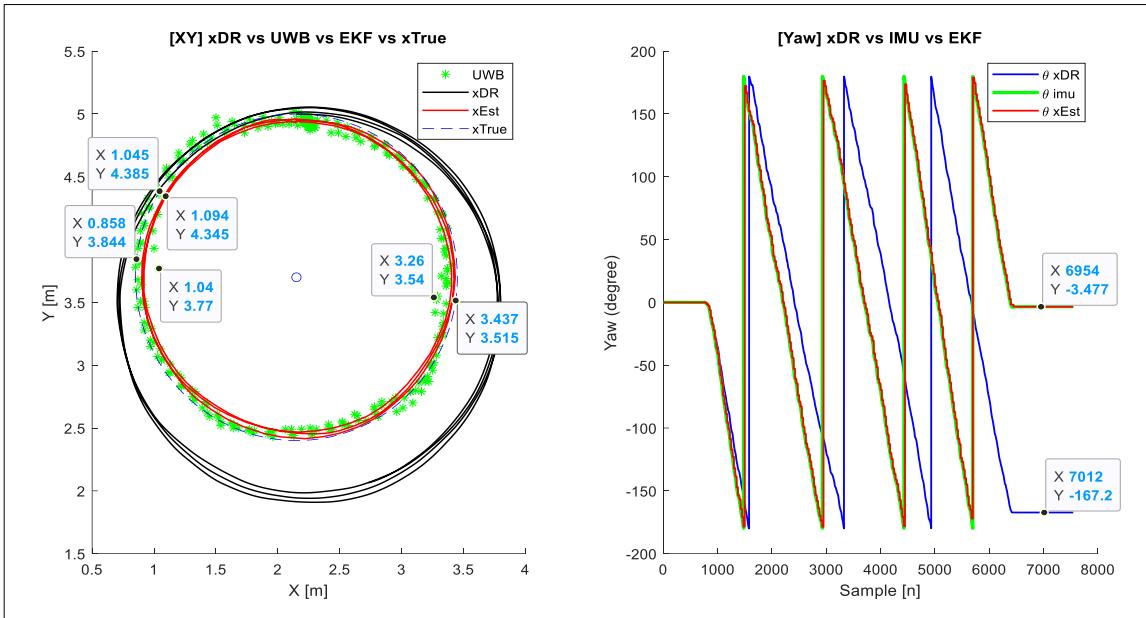
Hình 5.13 EKF với phép đo UWB kết hợp IMU quỹ đạo hình chữ P

**Nhận xét:** Tư thế ước lượng của Robot cả về hướng ( $\theta xEst$ ) và vị trí ( $xEst$ ) gần chính xác với quỹ đạo di chuyển thật ( $xTrue$ ). Sai số đo lường lớn nhất của UWB trong trường hợp này là 14cm. Sai số ước lượng qua EKF lớn nhất cũng là 14cm. Tuy sai số không được cải thiện nhưng quỹ đạo di chuyển của  $xEst$  hợp lý, đúng với thực tế di chuyển của Robot.

### 5.2.4. Quỹ đạo di chuyển liên tục nhiều vòng

#### 5.2.4.1. Di chuyển 03 vòng theo quỹ đạo tròn

Thực hiện quá trình thử nghiệm như mục 5.2.2.3, khi cho robot di chuyển theo hình tròn, chỉ khác quỹ đạo di chuyển được lặp lại 03 lần.

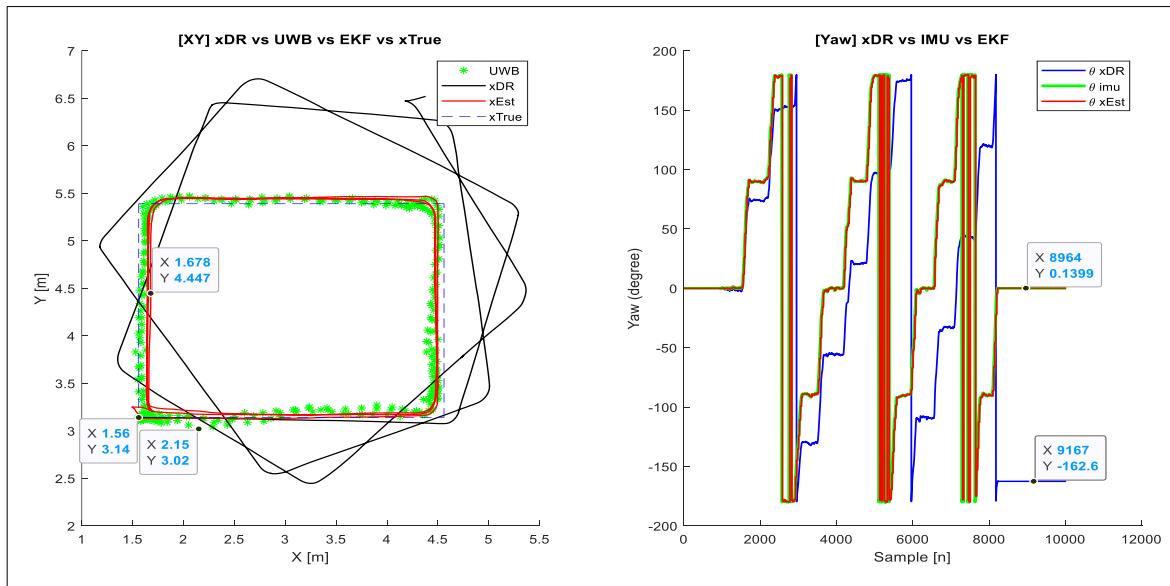


Hình 5. 14 EKF với quỹ đạo di chuyển theo hình tròn 03 lần

Nhận xét: Tư thế ước lượng của robot ( $x_{Est}$ ) đã khá chính xác về cả vị trí và góc hướng ( $\theta_{x_{Est}}$ ) so với quỹ đạo di chuyển thật ( $x_{True}$ ). Sai số đo lường lớn nhất của UWB là 12.2cm. Ước lượng tư thế Robot của EKF có sai số lớn nhất là 8.1cm.

#### 5.2.4.1. Di chuyển 03 vòng theo quỹ đạo hình chữ nhật

Thực hiện quá trình thử nghiệm như mục 5.2.3.1, khi cho robot di chuyển theo hình chữ nhật, chỉ khác quỹ đạo di chuyển được lặp lại 03 lần.



Hình 5. 15 EKF với quỹ đạo di chuyển theo hình chữ nhật 03 lần

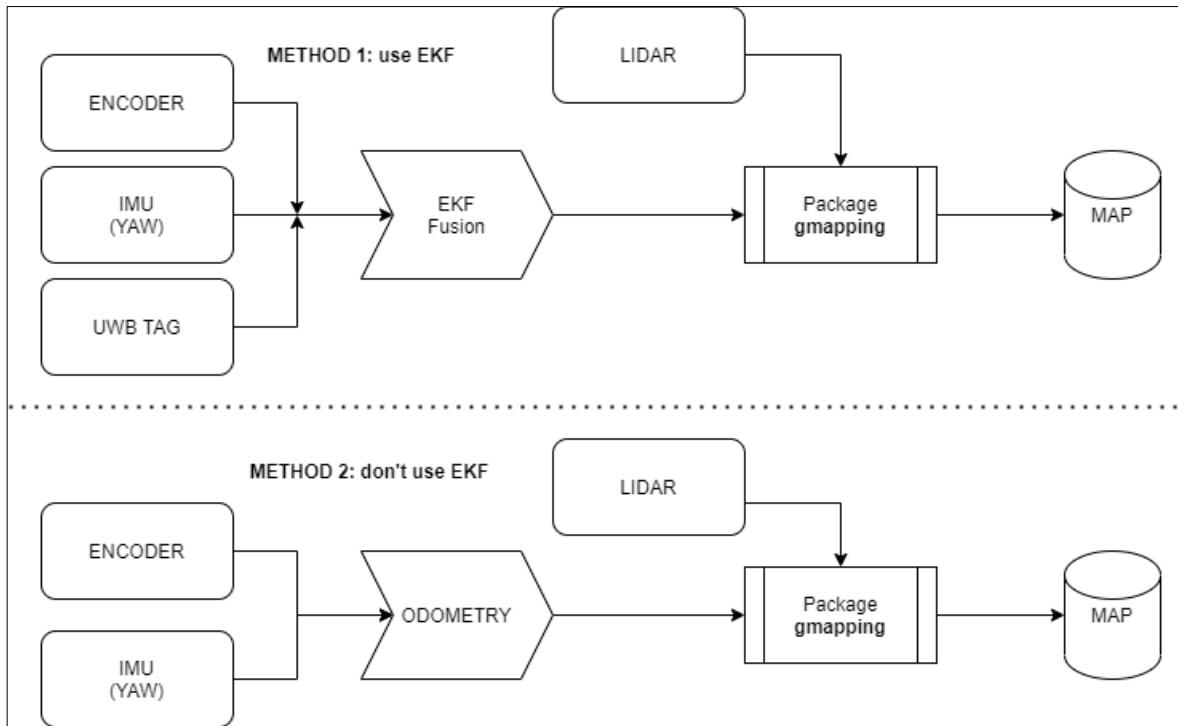
**Nhận xét:** Biên dạng của quỹ đạo di chuyển tương đối là trơn và mịn, góc hướng của robot ( $\theta_{xEst}$ ) được ước lượng chính xác theo góc yaw của imu ( $\theta_{imu}$ ). Tuy thế robot được ước lượng ( $x_{Ext}$ ,  $\theta_{xEst}$ ) không hoàn toàn lệ thuộc vào phép đo từ UWB. Sai số lớn nhất của UWB trong thử nghiệm này là 12cm. Sai số của quá trình ước lượng EKF là 11.8 cm. Quá trình Dead Reckoning thì đã quá sai khác về cả vị trí và hướng.

### 5.3. Xác định vị trí Robot với cảm biến Lidar

#### 5.3.1. Vẽ bản đồ môi trường

Khi sử dụng cảm biến Lidar để xác định vị trí của Robot, cần phải biết trước thông tin về môi trường mà robot sẽ hoạt động. Vì vậy, cần phải xây dựng bản đồ trước. Luận văn sử dụng gói gmapping<sup>19</sup> của ROS.

<sup>19</sup> [https://github.com/ros-perception/slam\\_gmapping.git](https://github.com/ros-perception/slam_gmapping.git)

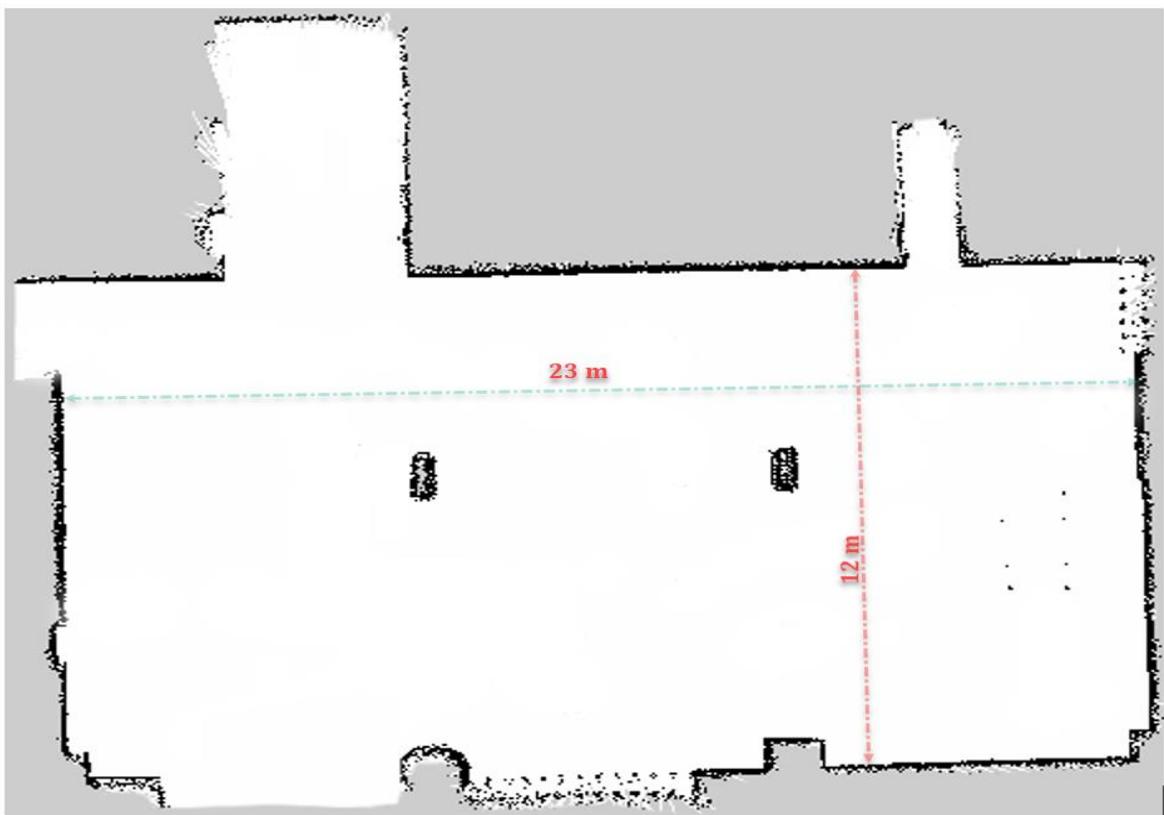


Hình 5. 16 Hai phương pháp thử nghiệm lấy bản đồ

Khu vực lấy bàn đồ tương đối rộng và trốn trãi, có những vùng đồng nhất với nhau. Nên quá trình lấy map để robot không bị mất phương hướng có 02 phương pháp đã được thử nghiệm, một là: lấy thông tin vị trí, hướng của robot từ bộ EKF cung cấp cho gói gmapping; Hai là: thay thế góc hướng robot bằng góc yaw của IMU và robot di chuyển với vận tốc thấp (0.2 m/s). Cả 2 phương pháp đều cho kết quả tốt như nhau, chỉ khác khi sử dụng thông tin lấy từ EKF thì robot có thể di chuyển với vận tốc cao hơn.



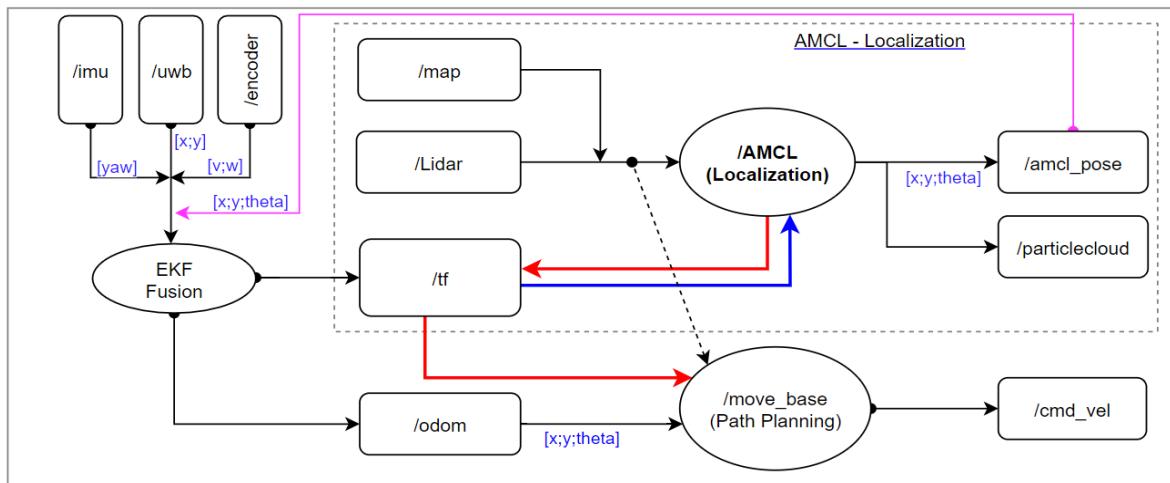
Hình 5. 17 Khu vực thử nghiệm Robot (sảnh Học viện quân y, Q.10)



Hình 5. 18 Bản đồ môi trường đã được xây dựng

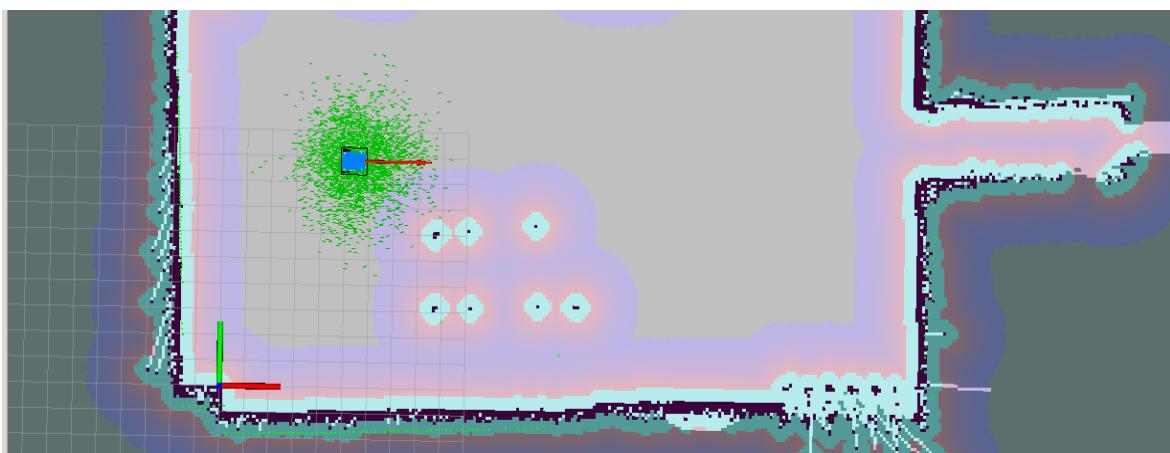
### 5.4.2. Định vị AMCL cho Robot

Quá trình AMCL Localization với cảm biến Lidar gặp nhũng bất lợi như mục 3.5 đã trình bày. Vì vậy, đề tài trình bày 01 phương pháp cải tiến như hình 5.25. Với phương pháp này, gói AMCL không cần phải khởi tạo vị trí ban đầu, bộ EKF Fusion sẽ cung cấp tọa độ và hướng gần chính xác nhất. Bên cạnh đó, nếu quá trình ước lượng của AMCL bị sai, hoặc robot bị dịch chuyển đi chỗ khác, hoặc bánh xe quay tự do (bị trượt, robot không di chuyển)... thì ngay sau đó bộ EKF Fusion sẽ cung cấp lại vị trí mới (vị trí chính xác) cho Robot.



Hình 5. 19 Phương pháp cải tiến cho gói AMCL

Dữ liệu sau khi AMCL Localization chính xác (/amcl\_pose) được sử dụng cập nhập trở lại bộ EKF Fusion (như một phép đo lường mới) cho chính gói AMCL và Move\_Base (gói điều hướng dùng dẫn đường cho Robot).



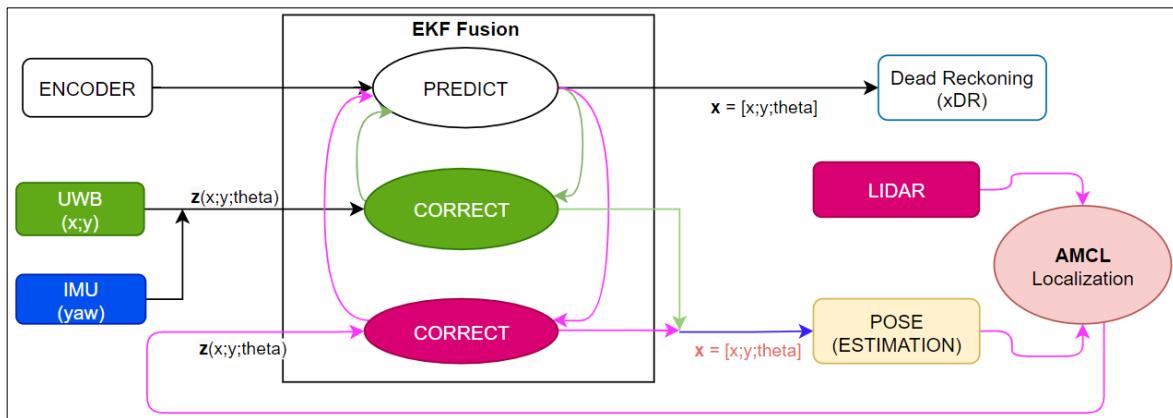
Hình 5. 20 Quá trình ước lượng vị trí của AMCL

## 5.4. Hợp nhất các cảm biến cho quá trình định vị Robot

### 5.4.1. Phương pháp thực hiện

Triển khai 06 UWB với không gian rộng hơn, bao phủ gần hết diện tích bản đồ đã thu thập (như hình bên dưới). Quỹ đạo di chuyển là hình vuông kích thước 7.2x7.2m và hình chữ nhật 7.2x13.2m, Robot sẽ di chuyển theo các quỹ đạo này đúng 01 vòng từ điểm bắt đầu có tọa độ  $[x; y; \theta] = [2.17; 3.70; 0.0]$  và kết thúc tại điểm đã xuất phát với góc hướng  $\theta = -90^\circ$ , phương pháp EKF fusion như hình 5.21 và tham số cho EKF như bảng 5.7

Kết quả dùng để đánh giá là EKF hợp nhất toàn bộ dữ liệu bao gồm UWB, IMU và kết quả trả về từ AMCL so với quỹ đạo di chuyển thực tế.

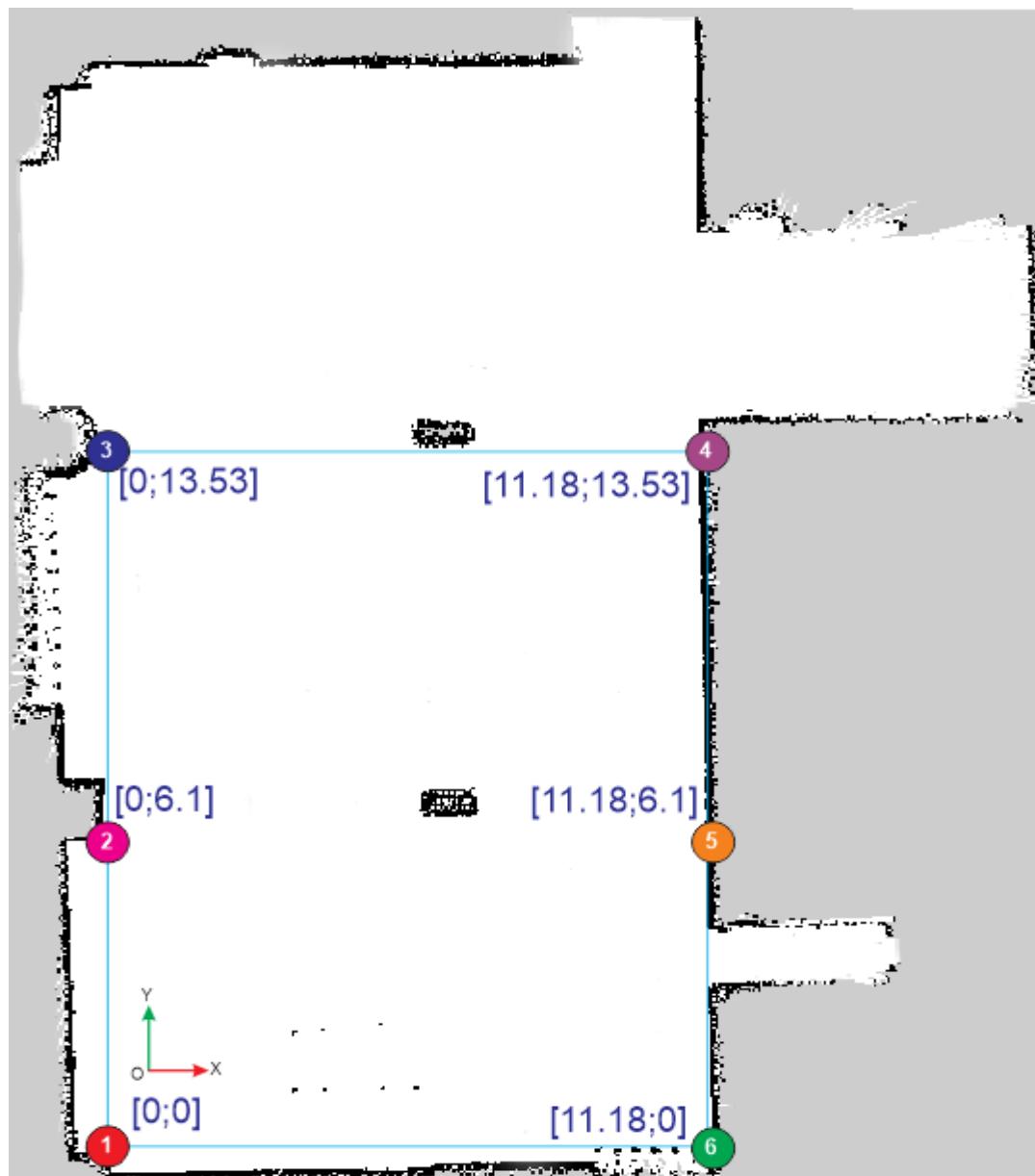


Hình 5. 21 Phương pháp tổng hợp EKF Fusion

Bảng 5. 4 Ma trận  $\mathbf{R}$ ,  $\mathbf{Q}$  cho thử nghiệm tổng hợp

STT	Ma trận $\mathbf{R}$ , $\mathbf{Q}$	Ghi chú
1	Ma trận $\mathbf{Q} = \begin{bmatrix} 0.01^2 & 0 & 0 \\ 0 & 0.01^2 & 0 \\ 0 & 0 & 0.02^2 \end{bmatrix}$	
2	Ma trận $\mathbf{R}_{UWB\_IMU} = \begin{bmatrix} 3.1^2 & 0 & 0 \\ 0 & 3.1^2 & 0 \\ 0 & 0 & 0.001 \end{bmatrix}$	$\mathbf{R}$ của UWB và IMU (khi $v = \omega = 0.0$ )
3	Ma trận $\mathbf{R}_{UWB\_IMU} = \begin{bmatrix} 3.1^2 & 0 & 0 \\ 0 & 3.1^2 & 0 \\ 0 & 0 & 10^4 \end{bmatrix}$	$\mathbf{R}$ của UWB và IMU (khi $v, \omega \neq 0.0$ )

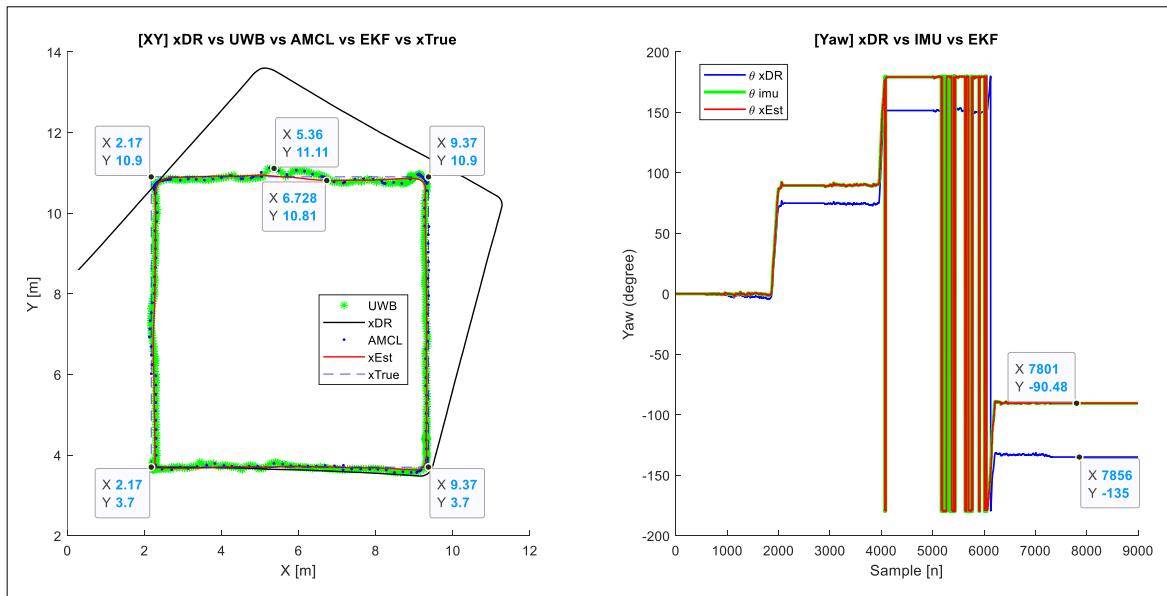
4	Ma trận $\mathbf{R}_{AMCL\_LIDAR} = \begin{bmatrix} 0.01^2 & 0 & 0 \\ 0 & 0.01^2 & 0 \\ 0 & 0 & 0.02^2 \end{bmatrix}$	$\mathbf{R}$ của Lidar
---	---	------------------------



Hình 5.22 Sơ đồ bố trí 06 UWB Anchor trên bản đồ

### 5.4.2. Đánh giá kết quả định vị

#### 5.4.2.1. EKF với quỹ đạo di chuyển hình vuông

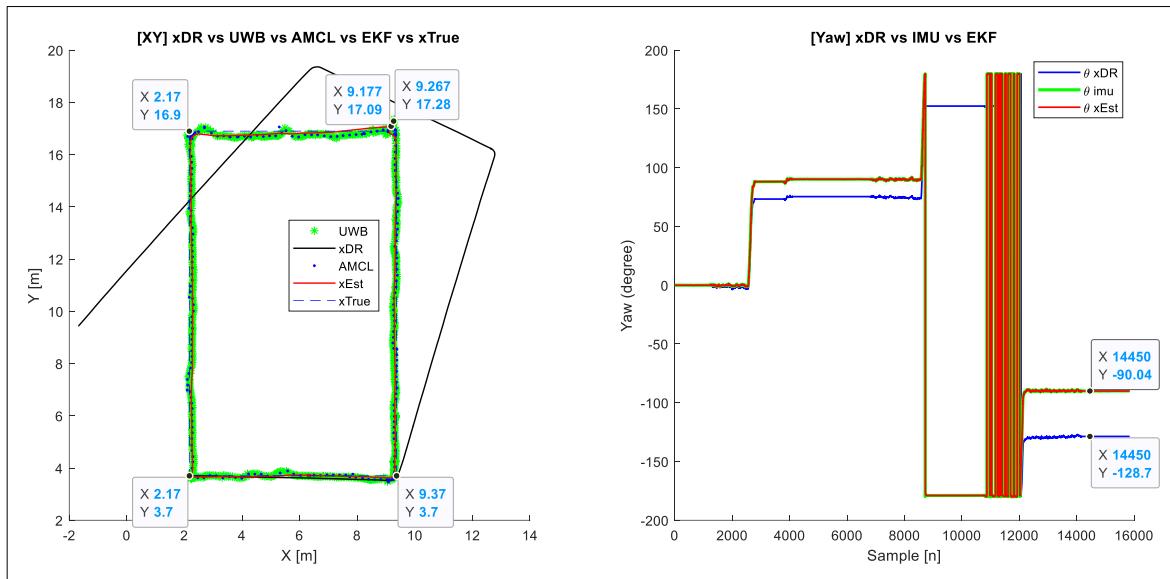


Hình 5. 23 EKF với UWB+IMU khi quỹ đạo hình vuông

**Nhận xét 1:** Tư thế về vị trí và hướng của AMCL (điểm chấm tròn, màu xanh blue) cơ bản bám theo quỹ đạo di chuyển thật. Việc này sẽ thuận lợi cho tầng điều hướng phía trên. Tuy nhiên dữ này đang chưa thực sự tốt. Vì trong quá trình gói AMCL hoạt động, đã được khởi tạo lại liên tục (vì quá trình ước lượng của AMCL thất bại) nên các điểm chấm của AMCL có xu hướng nghiêng về phía UWB.

**Nhận xét 2:** Tư thế ước lượng của Robot ( $x_{\text{Est}}$ ,  $\theta_{x_{\text{Est}}}$ ) với bộ lọc EKF cơ bản bám sát với quỹ đạo di chuyển thực tế ( $x_{\text{True}}$ ). Sai số đo lường lớn nhất của UWB là 21 cm. Sai số lớn nhất của  $x_{\text{Est}}$  ước lượng là 10cm.

### 5.4.2.2. EKF với quỹ đạo di chuyển hình chữ nhật



Hình 5. 24 EKF với UWB+IMU khi quỹ đạo hình chữ nhật

**Nhận xét:** Tương tự như nhận xét 1 ở quỹ đạo hình vuông đối với dữ liệu từ AMCL. Ngoài ra, tư thế ước lượng của Robot ( $x_{\text{Est}}$ ,  $\theta_{x_{\text{Est}}}$ ) với bộ lọc EKF cơ bản bám theo quỹ đạo di chuyển thực tế ( $x_{\text{True}}$ ). Sai số đo lường lớn nhất của UWB là 38cm và sai số lớn nhất của  $x_{\text{Est}}$  ước lượng là 19cm (khu vực góc đỉnh phía trên bên phải). Còn lại sai số duy trì trong khoảng nhỏ hơn 10cm.

### 5.5. Đánh giá kết quả định vị với bài toán điều hướng Robot

Quá trình thử nghiệm được chia thành 03 lần, cụ thể như sau:

Lần 1: Cho robot di chuyển từ điểm A đến điểm kết thúc là C;

Lần 2: Cho robot di chuyển đến các đỉnh hình chữ nhật ABCD, xuất phát từ A đến B-C-D và về lại A.

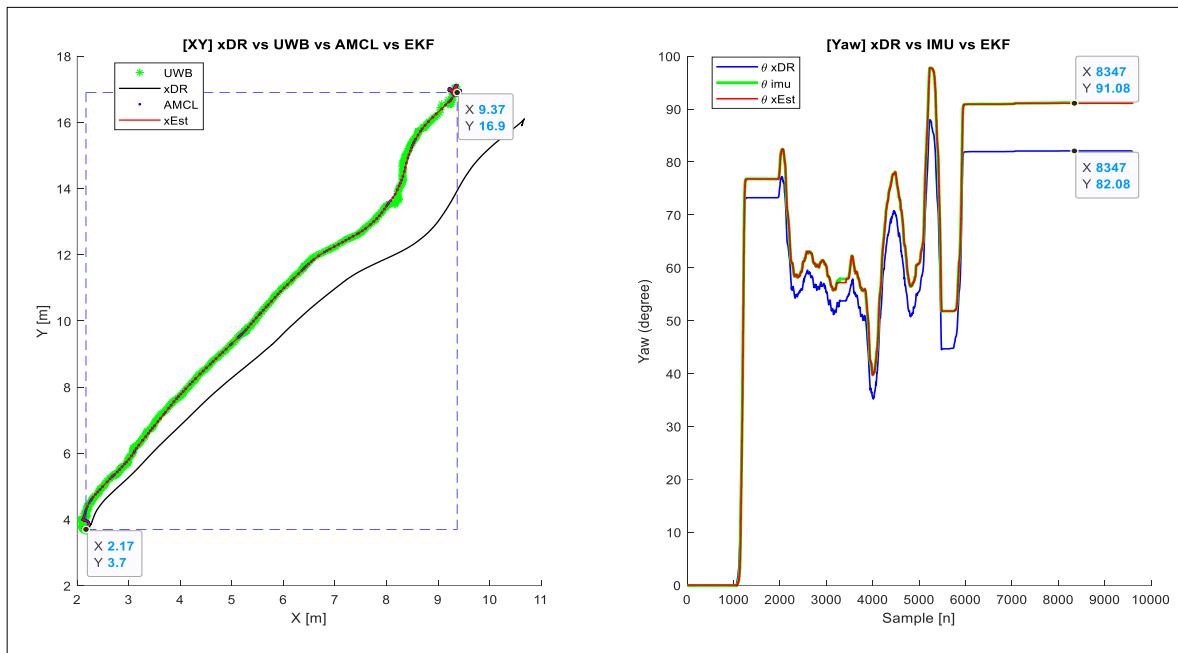
Lần 3: Tương tự như lần 2, nhưng với hình chữ nhật lớn hơn, các đỉnh C và D có sự thay đổi. Tọa độ các điểm như bảng 5.7. Quá trình điều hướng cho phép dung sai số là 7cm ( $xy\_goal\_tolerance: 0.07$ ).

Bảng 5. 5 Vị trí các điểm ABCD theo  $[x; y; \theta]$

STT	Điểm A	Điểm B	Điểm C	Điểm D
Lần 1	[2.17; 3.7; 0.0]		[9.37; 16.9; 1.57]	
Lần 2	[2.17; 3.7; 0.0]	[9.37; 3.70; 0.0]	[9.37; <b>10.9</b> ; 1.57]	[2.17; <b>10.9</b> ; 3.14]

## Chương 5: THỰC HIỆN TỔNG HỢP CẢM BIẾN CHO ĐỊNH VỊ ROBOT DI ĐỘNG

Lần 3	[2.17; 3.7; 0.0]	[9.37; 3.70; 0.0]	[9.37; <b>16.9</b> ; 1.57]	[2.17; <b>16.9</b> ; 3.14]
-------	------------------	-------------------	----------------------------	----------------------------

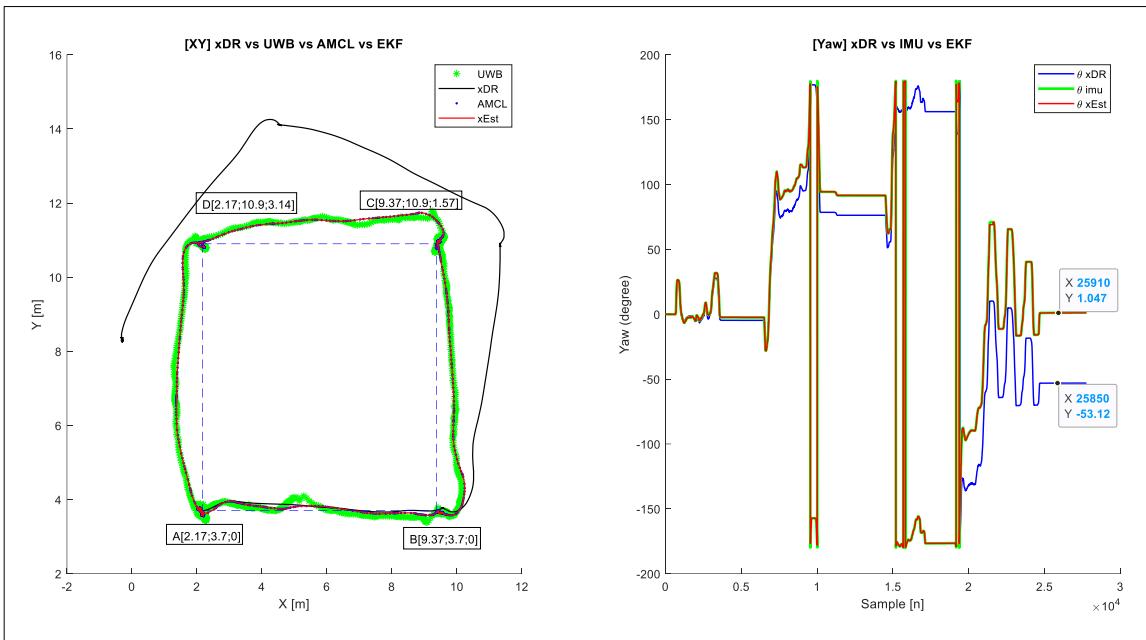


Hình 5. 25 Điều hướng Robot từ A đến C<sup>20</sup>

Bảng 5. 6 Kết quả điều hướng Lần 1 từ A đến C

Vị trí điểm cuối ước lượng (xEst) [ $x$ ; $y$ ; $\theta$ ] của Robot (xEst) tại C		
	Điểm C [9.37; 16.9; 1.57]	Sai số / đo tay
xEst	[9.3223; 17.0017; 1.5897]	[0.08; 0.0; 0.0]

<sup>20</sup> Rosbag: 2020-09-02-18-14-13.bag ; Video: VID\_20200902\_181422.mp4

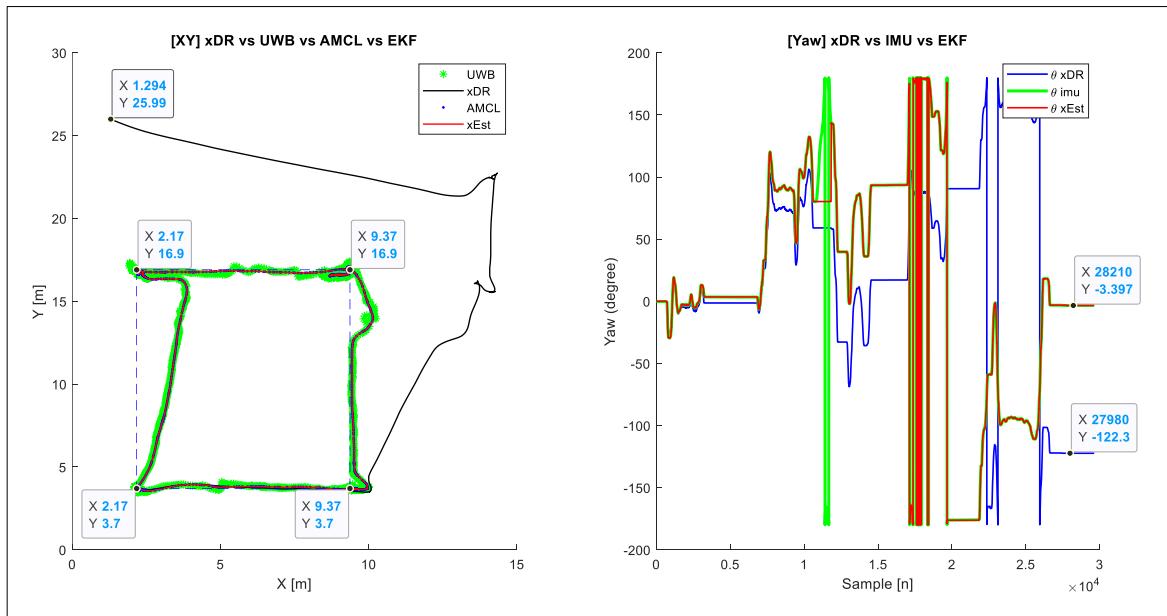


Hình 5. 26 Điều hướng các đỉnh hình vuông ABCD<sup>21</sup>

Bảng 5. 7 Sai số của EKF tại các đỉnh hình vuông ABCD

Vị trí điểm cuối ước lượng (xEst) [ $x; y; \theta$ ] của Robot		
Điểm	xEst [ $x; y; \theta$ ]	Sai số / đo tay
B [9.37; 3.70; 0.0]	[9.4875; 3.6543; -0.0425]	[0.055; 0.02; 0.0]
C [9.37; 10.9; 1.57]	[9.5943; 11.1801; 1.5467]	[0.11; 0.04; 1.57]
D [2.17; 10.9; 3.14]	[2.2208; 10.8098; -3.0827]	[0.07; 0.1; -3.0]
A [2.17; 3.7; 0.0]	[2.0775; 3.7658; 0.0207]	[0.04; 0.05; 0.0]

<sup>21</sup> Rosbag: 2020-09-03-00-48-34.bag; Video: VID\_20200903\_004836.mp4



Hình 5. 27 Điều hướng các đỉnh hình chữ nhật ABCD<sup>22</sup>

Bảng 5. 8 Sai số của EKF tại các đỉnh hình chữ nhật ABCD

Vị trí điểm cuối ước lượng (xEst) [ $x; y; \theta$ ] của Robot		
Điểm	xEst [ $x; y; \theta$ ]	Sai số / đo tay
B [9.37; 3.70; 0.0]	[9.3358; 3.7064; 0.2277]	[0.07;0.0;0.2]
C [9.37; 16.9; 1.57]	[9.3128; 16.8453; 1.6319]	[0.07;0.08;1.6]
D [2.17; 16.9; 3.14]	[2.2218 16.6910 -3.0754]	[0.10;0.0;-3.0]
A [2.17; 3.7; 0.0]	[2.2576; 3.5747; -0.0593]	[0.01;0.01;0.0]

## 5.6. Nhận xét và kết luận

Cơ bản bộ lọc EKF đã hợp nhất được các dữ liệu đưa đến từ các loại cảm biến khác nhau. Giá trị vị trí và hướng có được từ quá trình EKF (xEst) khá chính xác, sai số lớn nhất trong 03 lần thử nghiệm là 11cm (trường hợp hình vuông, tại đỉnh B), còn lại đa số đều nhỏ hơn 10cm. Thậm chí có những điểm rất chính xác như điểm A (1cm).

<sup>22</sup> Rosbag: 2020-09-03-00-20-00.bag ; Video: VID\_20200903\_002011.mp4

Quá trình điều hướng (sử dụng gói move\_base của ROS) hoạt động khá tốt trong không gian tương đối rộng và trống trải. Với không gian như vậy bộ AMCL thường sẽ không ước lượng chính xác vị trí Robot, dẫn đến quá trình navigation không thực hiện được. Tuy nhiên đề tài đã sử dụng kết quả của EKF (khi sử dụng Encoder, IMU, UWB) để khởi tạo lại gói AMCL mỗi khi nhận thấy thông tin về tư thế robot  $[x; y; \theta]$  của AMCL có sự sai khác lớn so với EKF.

## Chương 6

### KẾT LUẬN VÀ PHƯƠNG HƯỚNG PHÁT TRIỂN

#### 6.1. Đánh giá kết quả thực hiện

- Đã tìm hiểu, phân tích và xây dựng mô hình toán của của robot từ cơ sở mô hình toán của từng động cơ; đánh giá được kết quả xây dựng thông qua mô phỏng simulink và thực nghiệm trên robot
- Xác định được bộ tham số gain ( $K_p$ ,  $K_i$ ,  $K_d$ ) cho điều khiển PID dựa trên phương pháp nhận dạng hàm truyền cho từng đồng cơ kết hợp PID Tunner của Matlab. Đáp ứng của kết quả mô phỏng và thực nghiệm của bộ điều khiển cơ bản giống nhau.
- Triển khai được bộ lọc Madgwick cho IMU trên cả nền tảng MCU và Linux. Góc yaw thu được cơ bản nhỏ, đáp ứng được yêu cầu bài toán.
- Cơ bản hiểu lý thuyết và ứng dụng được bộ lọc EKF. Đã xây dựng được bài toán mô phỏng quá trình EKF cho từng cảm biến sát với mô hình của Robot.
- Bộ AMCL Localization của ROS được cải tiến cơ bản hoạt động tốt.
- Đã triển khai được bộ EKF trên mô hình thực tế cho các cảm biến như: Encoder, MPU9250, UWB DWM1001C, Lidar Samtec Rplidar A1M8.
- Quá trình định vị cho Robot cơ bản đã được cải thiện, đáp ứng được yêu cầu đề ra của Luận văn.
- Ngoài ra, đã triển khai được bộ điều hướng, tránh vật cản cho Robot trên cơ sở ứng dụng bộ định vị cho robot đã xây dựng.

#### 6.2. Những mặt hạn chế

- Robot cần biết góc hướng (góc yaw) ban đầu khi vừa mới khởi động (so với khung tham chiếu toàn cục) để quá trình AMCL Localization được chính xác. (Vị trí XY đã được bộ EKF cung cấp đến).
- Thông tin vị trí và hướng cung cấp từ bộ AMCL Localization đến gói EKF chưa thực sự ảnh hưởng nhiều đến hệ thống. Do thời gian cập nhật của gói rất chậm, và không gửi thường xuyên. Nên quá trình fusion của

EKF chủ yếu correct (update) lại giá trị XY của UWB (update đều đặn 2Hz hoặc 5Hz hoặc 10Hz), và update góc hướng từ IMU (200Hz)

- Giải pháp xử lý góc trôi góc yaw chưa triệt đệ, cần có giải pháp khác để cung cấp góc hướng tham chiếu cho robot để bù trôi góc yaw từ IMU khi robot hoạt động thời gian liên tục.

### 6.3. Phương hướng phát triển

- Bổ sung thêm cảm biến Camera 3D để vẽ bản đồ toàn diện hơn về môi trường, từ đó có thể xác định chính xác góc hướng của robot so với tham chiếu toàn cục.
- Tối ưu bộ AMCL cho đáp ứng tốt hơn, tận dụng được góc hướng từ bộ AMCL cho việc tính toán bias cho góc yaw từ IMU.
- Tối ưu tham số của ma trận  $R$ ,  $Q$  cho bộ lọc EKF.
- Ứng dụng kết quả nghiên cứu cho robot AGV hoạt động trong môi trường nhà máy.

## TÀI LIỆU THAM KHẢO

### Tiếng việt

- [1] T. T. Hoàng, “Nghiên cứu phương pháp tổng hợp cảm biến dùng cho kỹ thuật dẫn đường các robot di động,” LATS, Đại Học Công Nghệ, Đại Học Quốc Gia Hà Nội, Hà Nội, 2017.

### Tiếng anh

- [2] Hakyung Chung, L. Ojeda, and J. Borenstein, “Sensor fusion for mobile robot dead-reckoning with a precision-calibrated fiber optic gyroscope,” in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, Seoul, South Korea, 2001, vol. 4, pp. 3588–3593, doi: 10.1109/ROBOT.2001.933174.
- [3] H. Temeltas, “A REAL-TIME LOCALIZATION METHOD FOR AGVS IN SMART FACTORIES,” no. 2, p. 6, 2018.
- [4] Y. Sun, L. Guan, Z. Chang, C. Li, and Y. Gao, “Design of a Low-Cost Indoor Navigation System for Food Delivery Robot Based on Multi-Sensor Information Fusion,” *Sensors*, vol. 19, no. 22, p. 4980, Nov. 2019, doi: 10.3390/s19224980.
- [5] D. T. H. Quoc *et al.*, “Employing Extended Kalman Filter with Indoor Positioning System for Robot Localization Application,” p. 7.
- [6] H. Pointon, B. McLoughlin, C. Matthews, and F. Bezombes, “Towards a Model Based Sensor Measurement Variance Input for Extended Kalman Filter State Estimation,” *Drones*, vol. 3, no. 1, p. 19, Feb. 2019, doi: 10.3390/drones3010019.
- [7] K. Maatoug, M. Njah, and M. Jallouli, “Multisensor data fusion for electrical wheelchair localization using extended Kalman Filter,” in *2017 18<sup>th</sup> International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, Monastir, 2017, pp. 257–260, doi: 10.1109/STA.2017.8314970.

- [8] 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)*, Newcastle, Australia, 2016, pp. 212–216, doi: 10.1109/AUCC.2016.7868190.
- [9] M. M. I. Hasan, “Indoor and outdoor localization of a mobile robot fusing sensor data,” p. 47.
- [10] L. A. Nguyen, P. T. Dung, T. D. Ngo, and X. T. Truong, “Improving the accuracy of the autonomous mobile robot localization systems based on the multiple sensor fusion methods,” in *2019 3<sup>rd</sup> International Conference on Recent Advances in Signal Processing, Telecommunications & Computing (SigTelCom)*, Hanoi, Vietnam, 2019, pp. 33–37, doi: 10.1109/SIGTELCOM.2019.8696103.
- [11] T. Moore and D. Stouch, “A Generalized Extended Kalman Filter Implementation for the Robot Operating System,” in *Intelligent Autonomous Systems 13*, vol. 302, E. Menegatti, N. Michael, K. Berns, and H. Yamaguchi, Eds. Cham: Springer International Publishing, 2016, pp. 335–348.
- [12] E. I. Al Khatib, M. A. Jaradat, M. Abdel-Hafez, and M. Roigari, “Multiple sensor fusion for mobile robot localization and navigation using the Extended Kalman Filter,” in *2015 10th International Symposium on Mechatronics and its Applications (ISMA)*, Sharjah, United Arab Emirates, 2015, pp. 1–5, doi: 10.1109/ISMA.2015.7373480.
- [13] Y. Lee and D. Lim, “Vision/UWB/IMU sensor fusion based localization using an extended Kalman filter,” in *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*, Yunlin, Taiwan, 2019, pp. 401–403, doi: 10.1109/ECICE47484.2019.8942733.
- [14] D. Nemec, V. Šimák, A. Janota, M. Hruboš, and E. Bubeníková, “Precise localization of the mobile wheeled robot using sensor fusion of odometry, visual artificial landmarks and inertial sensors,” *Robotics and Autonomous Systems*, vol. 112, pp. 168–177, Feb. 2019, doi: 10.1016/j.robot.2018.11.019.

- [15] H. Temeltas, "A REAL-TIME LOCALIZATION METHOD FOR AGVS IN SMART FACTORIES," no. 2, p. 6, 2018.
- [116] A. Klee, "DEVELOPMENT OF A MOTOR SPEED CONTROL SYSTEM USING MATLAB AND SIMULINK, IMPLEMENTED WITH A DIGITAL SIGNAL PROCESSOR," p. 94.
- [17] T. Hellström, "Kinematics Equations for Differential Drive and Articulated Steering," p. 12.
- [18] M. I. Ribeiro and P. Lima, "KINEMATICS MODELS OF MOBILE ROBOTS," p. 16, 2002.
- [19] S. K. Malu and J. Majumdar, "Kinematics, Localization and Control of Differential Drive Mobile Robot," p. 9, 2014.
- [20] T. Moore and D. Stouch, "A Generalized Extended Kalman Filter Implementation for the Robot Operating System," in *Intelligent Autonomous Systems 13*, vol. 302, E. Menegatti, N. Michael, K. Berns, and H. Yamaguchi, Eds. Cham: Springer International Publishing, 2016, pp. 335–348.
- [21] M. I. Ribeiro, "Introduction to Kalman Filtering," p. 53.
- [22] M. R. Ananthasayanam, M. S. Mohan, N. Naik, and R. M. O. Gemson, "A heuristic reference recursive recipe for adaptively tuning the Kalman filter statistics part-1: formulation and simulation studies," *Sādhanā*, vol. 41, no. 12, pp. 1473–1490, Dec. 2016, doi: 10.1007/s12046-016-0562-z.
- [23] M. S. Mohan, N. Naik, R. M. O. Gemson, and M. R. Ananthasayanam, "A heuristic reference recursive recipe for adaptively tuning the Kalman filter statistics part-2: real data studies," *Sādhanā*, vol. 41, no. 12, pp. 1491–1507, Dec. 2016, doi: 10.1007/s12046-016-0563-y.
- [24] R. Masinjila, "Multirobot Localization Using Heuristically Tuned Extended Kalman Filter," p. 106.
- [25] Y. Laamari, K. Chafaa, and B. Athamena, "Particle swarm optimization of an extended Kalman filter for speed and rotor flux estimation of an induction motor drive," *Electr Eng*, vol. 97, no. 2, pp. 129–138, Jun. 2015, doi: 10.1007/s00202-014-0322-1.

- [26] T. Michel, H. Fourati, P. Geneves, and N. Layaida, “A comparative analysis of attitude estimation for pedestrian navigation with smartphones,” in *2015 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Banff, AB, Canada, Oct. 2015, pp. 1–10, doi: 10.1109/IPIN.2015.7346767.
- [27] S. Ludwig, K. Burnham, A. Jimenez, and P. Touma, “Comparison of attitude and heading reference systems using foot mounted MIMU sensor data: basic, Madgwick, and Mahony,” in *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2018*, Denver, United States, Mar. 2018, p. 96, doi: 10.1117/12.2296568.
- [28] “Cavallo et al. - 2014 - Experimental Comparison of Sensor Fusion Algorithm.pdf.” Accessed: Dec. 22, 2019. [Online]. Available: <http://folk.ntnu.no/skoge/prost/proceedings/ifac2014/media/files/1173.pdf>.
- [29] S. O. H. Madgwick, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays,” p. 32.
- [30] R. Mahony, T. Hamel, and J.-M. Pflimlin, “Nonlinear Complementary Filters on the Special Orthogonal Group,” *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, Jun. 2008, doi: 10.1109/TAC.2008.923738.

## PHỤ LỤC 1<sup>23</sup>

### **1.1. Danh sách file data dùng nhận dạng mô hình động cơ (ident)**

STT	Name: *.sid	Notes
1	getEstimation_Pwm_250.mat	Tệp dữ liệu chứa data lấy từ MCU
2	getEstimationModelMotorDC.slx	File dùng nhận dữ liệu qua UART
3	Ident_Motor_Pwm_250_LEFT.sid	File của indent cho động cơ Trái
4	Ident_Motor_Pwm_250_RIGHT.sid	File của indent cho động cơ Trái

### **1.2. Danh sách m-file và data thực hiện quá trình mô phỏng, đánh giá**

STT	Name: m-file	Name: *.mat
1	getEstimationModelMotor.m	getENCODER_PWM_150.mat
2	nt	getENCODER_PWM_250.mat
3	nt	getENCODER_PWM_500.mat
4	nt	getENCODER_PWM_900.mat

### **1.3. Danh sách m-file và data thực hiện đánh giá odometry**

STT	Name: m-file	Name: *.mat
1	get_Response_Robot_Odom_NEW.m	get_Odom_Run_Circle.mat
2	nt	get_Odom_Run_Square.mat
3	nt	get_Odom_Run_Num8.mat

### **1.4. Danh sách m-file và data cho đánh giá bộ lọc Madgwick**

STT	Name: m-file	Name: *.mat
1	Run_IMU_Madgwick_KF.m	IMU6050_NEW_RAW_RPY_NEW1.mat
2	getRPY_IMU6050_NEW2.slx	File lấy data từ MPU6050 qua UART

<sup>23</sup> Toàn bộ phần source code, database, báo cáo được lưu trữ tại:  
[https://github.com/hienclubvn/Thesis\\_Master\\_BKU\\_2020.git](https://github.com/hienclubvn/Thesis_Master_BKU_2020.git)

3	ExampleScriptHien_NEW.m	IMU6050_NEW_RAW_RPY.mat
---	-------------------------	-------------------------

Kiểm tra MPU9250 trong chương 5

STT	Name: m-file	Name: *.mat
1	test_imu_cx.m	2020-08-18-09-47-17.bag
2	test_imu_drift.m	2020-08-18-06-29-43.bag

### 1.5. Danh sách m-file và data cho đánh giá phương sai nhiễu UWB

STT	Name: m-file	Name: *.mat
1	run_uwb_new.m (chương 2)	uwb_test1.mat
2		uwb_test2.mat
3	test_uwb.m (chương 5)	read_uwb.bag

### 1.6. Danh sách bagfile cho các quỹ đạo di chuyển khác nhau

STT	Quỹ đạo/ m-file	Tên bagfile/m-file
1	Hình tròn 1 vòng	2020-08-19-04-28-55.bag
2	Hình tròn nhiều vòng	2020-08-19-04-41-31.bag
	m-file:	test_doluong.m
3	Hình vuông 1 vòng	2020-08-19-04-57-13.bag
4	Hình vuông nhiều vòng	2020-08-19-05-27-36.bag
5	Hình chữ P	2020-08-19-05-53-46.bag
	m-file:	test_doluong_hinhvuong.m

### 1.7. Danh sách bagfile cho bộ lọc EKF ở các quỹ đạo di chuyển khác nhau

STT	Quỹ đạo/ cách hợp nhất	Tên bagfile/m-file
1	Tròn 1 / EKF + UWB	2020-08-20-04-09-56.bag
2	Tròn 1 / EKF + UWB + IMU	2020-08-20-04-03-21.bag
3	Nhiều vòng / EKF + UWB	2020-08-20-04-13-41.bag
4	Nhiều vòng / EKF + UWB + IMU	2020-08-20-04-17-57.bag

	m-file:	test_ekf.m
5	Vuông 1 / EKF + UWB	2020-08-20-04-39-30.bag
6	Vuông 1 / EKF + UWB + IMU	2020-08-20-04-28-15.bag
7	Nhiều vuông / EKF + UWB	2020-08-20-04-48-05.bag
8	Nhiều vuông / EKF + UWB + IMU	2020-08-20-05-18-40.bag
	m-file:	test_ekf_square.m
9	Chữ P / EKF + UWB	2020-08-20-05-00-48.bag
10	Chữ P / EKF + UWB + IMU	2020-08-20-05-12-16.bag
	m-file:	test_ekf_P.m

### 1.8. Danh sách bagfile cho EKF Fusion tổng hợp có sử dụng Lidar

STT	Quỹ đạo/ m-file	Tên bagfile/m-file
1	Hình vuông lớn	2020-08-27-04-50-29.bag
2	Hình vuông nhỏ	2020-08-27-04-59-08.bag
	m-file:	test_ekf_full_sensor.m

### 1.9. Danh sách bagfile quá trình điều hướng đến các điểm ABCD

STT	Quỹ đạo/ m-file	Tên bagfile/m-file
1	A đến C	2020-09-02-18-14-13.bag
2	A đến BCDA (hình vuông)	2020-09-03-00-48-34.bag
	A đến BCDA (hình chữ nhật)	2020-09-03-00-20-00.bag
	m-file:	test_navigation.m

## PHỤ LỤC 2

### Code Matlab Function trên Matlab cho ModelRobot.slx

Tên function: MATLAB Function (Inverse Kinematics)

```
%hiennd, 04/08/2020
function [wL,wR] = inverseKinematics(v,w)
    wheelRadius = 0.0625;
    wheelBase = 0.37;
    % Calculates wheel speeds from linear and angular velocity
    wL = (v - w*wheelBase/2)/wheelRadius;
    wR = (v + w*wheelBase/2)/wheelRadius;
end
```

Tên function: MATLAB Function (Forward Kinematics)

```
%hiennd, 04/08/2020
function [v,w] = forwardKinematics(wL,wR)
    % Calculates linear and angular velocity from wheel speeds
    wheelRadius = 0.0625;
    wheelBase = 0.37;
    v = 0.5*wheelRadius*(wL+wR);
    w = (wR-wL)*wheelRadius/wheelBase;
end
```

Tên function: MATLAB Function (calcOdometry)

```
%hiennd, 04/08/2020
function [x,y,theta] = calcOdometry(v,w)
    persistent pose
    if isempty(pose)
        pose = zeros(3,1);      % Pose matrix
    end
    sampleTime = 0.005; %5ms
    velB = [v;0;w]; % Body velocities [vx;vy;w]
    % Convert from body to world
    %new code
    theta = pose(3) + 0.5*w*sampleTime; %,OK, too !!!
    vel = [cos(theta) -sin(theta) 0; sin(theta) cos(theta) 0;0 0
1]*velB;

    % Perform forward discrete integration step
    pose = pose + vel*sampleTime;
    if pose(3) >= 2*pi
        pose(3) = pose(3) - 2*pi;
    end
    if pose(3) <= -2*pi
        pose(3) = pose(3) + 2*pi;
    end
    %result
    x = pose(1);
    y = pose(2);
    theta = pose(3);
end
```

## **LÝ LỊCH TRÍCH NGANG**

**Họ và Tên:** Ngô Đăng Hiền

**Phái:** Nam

**Ngày sinh:** 20/08/1988

**Nơi sinh:** Đăk Lăk

**Địa chỉ liên lạc:** số nhà 813, Nhà công vụ Hải Quân, Vĩnh Nguyên, Nha Trang, Khánh Hòa.

**Email:** [danghien719@gmail.com](mailto:danghien719@gmail.com)

**Số điện thoại:** 0976.332.617

### **QUÁ TRÌNH ĐÀO TẠO**

Từ 2007-2012: Học đại học tại trường Học viện Hải Quân.  
Chuyên ngành: Vũ khí dưới nước

Từ 2017-2020: Học cao học tại trường Đại Học Bách Khoa-ĐHQG Tp.HCM.  
Chuyên ngành: Kỹ thuật Điều khiển và Tự động hóa.

### **QUÁ TRÌNH CÔNG TÁC**

Từ năm 2013 - nay: Trợ giảng tại trường Học Viện Hải Quân, Vĩnh Nguyên, Nha Trang, Khánh Hòa.