DANH MỤC BẢNG BIỂU

DANH MỤC HÌNH ẢNH

DANH MỤC TỪ VIẾT TẮT

3D	:	3 chiều (3 direction)
ROS	:	Hệ điều hành robot (Robot Operating System)
TCP	:	Giao thức điều khiển truyền dẫn (Transmission Control Protocol)
IP	:	Giao thức Internet (Internet Protocol)
D-H	:	Denavit – Hartenberg

MỤC LỤC

PHẦN MỞ ĐẦU

i. Đặt vấn đề

Sản xuất thông minh là một nền tảng cơ bản và quan trọng của nền công nghiệp hiện đại và kinh tế toàn cầu. Một trong những thách thức chính của sản xuất thông minh là kết nối không gian vật lý và không gian số. Sự phát triển nhanh chóng của mô phỏng, thu thập dữ liệu, truyền thông dữ liệu và các công nghệ tiên tiến khác đã tạo ra sự tương tác lớn hơn giữa không gian vật lý và không gian số. Digital twin đang ngày càng được sử dụng rộng rãi trong nhiều lĩnh vực và ngành công nghiệp khác nhau, và nhận được sự quan tâm từ cả giới học thuật và ngành công nghiệp.

Hiện nay, nhiều doanh nghiệp tại Việt Nam như ASIM, Ahamove, OneFin, Ngân hàng VIB, SDC, Crystal Bay, Joolux, DFAR, Time Universal, Ai20X và HoiAn đã triển khai và áp dụng thành công bản sao số trong quy trình quản lý và sản xuất. Các công ty lớn ở nước ngoài như General Electric, IBM, PTC, Microsoft Corporation, Siemens AG, ANSYS, SAP, Oracle, Robert Bosch, SWIM.AI và ZenithCities cũng đang áp dụng hiệu quả các công nghệ bản sao số.

Tuy nhiên, tại Việt Nam, các nghiên cứu về phát triển hệ thống robot thông minh vẫn đang ở giai đoạn phát triển ban đầu và tập trung vào các phần nhỏ trong cấu trúc của bản sao số. Một cấu trúc tổng thể cho bản sao số chưa được nghiên cứu và phát triển toàn diện. Do đó, việc nghiên cứu và phát triển nền tảng (framework) cho bản sao số là cần thiết.

ii. Nghiên cứu trong và ngoài nước

- Ngoài nước

Digital twin hay bản sao kỹ thuật số là một đại diện kỹ thuật số của một đối tượng hoặc hệ thống vật lý. Về bản chất, digital twin là một chương trình máy tính lấy dữ liệu trong thế giới thực về một đối tượng hoặc hệ thống vật lý làm đầu vào và tạo ra các dự đoán hoặc mô phỏng đầu ra về cách đối tượng hoặc hệ thống vật lý đó sẽ bị ảnh hưởng bởi những đầu vào đó.

Cơ sở lý thuyết của bản sao số đến từ các ngành khoa học khác nhau như khoa học thông tin, kỹ thuật sản xuất, khoa học dữ liệu và khoa học máy tính. Các lý thuyết phù hợp nhất được chia làm bốn phần [1]: (1) mô hình digital twin, mô phỏng, xác minh, …; (2) Hợp nhất dữ liệu; (3) Tương tác và cộng tác; và (4) Dịch vụ.

Mô hình hóa và mô phỏng digital twin là cơ sở của việc phát triển các digital twin trong thực tế. Nhiều nhà nghiên cứu đã đề xuất các kiến trúc mô hình khác nhau. Schroeder và cộng sự [2] đã đề xuất một kiến trúc mô hình digital bao gồm năm lớp (tức là lớp thiết bị, lớp giao diện người dùng, lớp dịch vụ web, lớp truy vấn và lớp kho dữ liệu) để quản lý dữ liệu digital twin. Họ cũng phát triển hệ thống thực tế tăng cường để hiển thị thông tin thời gian thực. Họ cũng đề xuất trao đổi dữ liệu giữa các hệ thống không đồng nhất thông qua AutomationML. Phương pháp này gồm 3 giai đoạn: (1) Tạo mô hình; (2) Xác nhận mô hình; Và (3) phát triển hệ thống thông tin. Yun và cộng sự [3] cũng đã đề xuất một mô hình hóa kiến trúc cho các nền tảng digital twin lớn bao gồm một khuôn khổ hợp tác phân tán và một cơ chế truyền thông. Moreno và cộng sự [4] đã sử dụng một máy đục lỗ để giới thiệu quy trình từng bước để xây dựng mô hình digital twin. Quy trình gồm 5 bước: (1) Xây dựng mô hình 3D; (2) Trích xuất hành vi; (3) Mô hình hóa sự tương tác giữa máy đục lỗ và các phần tử chuyển động; (4) Mô hình hoạt động; Và (5) Mô phỏng. X. Wang và cộng sự [5] đã sử dụng các nền tảng mã nguồn mở ROS-Gazebo để mô phỏng robot trực tuyến. Garg và cộng sự [6] sử dụng thực tế ảo giúp tăng tính trải nghiệm và tương tác khi mô phỏng và tương tác với robot từ xa. Ngoài ra mô hình digital twin cần được đánh giá đúng mức để đảm bảo độ chính xác của nó trong việc phản ánh thực tế ảo và thực. Do đó, Smarslok và cộng sự [7] đã đề xuất một khuôn khổ để lượng hóa lỗi và đánh giá độ tin cậy, bao gồm một bộ thước đo để đo độ trung thực của các mô hình bản sao số.

Hợp nhất dữ liệu là một công nghệ quan trọng vì digital twin phải xử lý một khối lượng dữ liệu rất lớn từ nhiều nguồn khác nhau như thiết bị, môi trường vật lý, dữ liệu trong quá khứ, …. Ricks và cộng sự [8] đã đề xuất một kỹ thuật giảm thứ tự cho digital twin được áp dụng trong phương pháp tổng quát hóa độ trung thực cao của tế bào để nâng cao hiệu quả xử lý dữ liệu. Cai và cộng sự [9] đã phát triển một phương pháp tích hợp dữ liệu cảm biến và dữ liệu sản xuất làm cơ sở xây dựng digital twin của máy phay

đứng, nơi dữ liệu cảm biến được sử dụng để theo dõi các hoạt động gia công và dự đoán độ nhám bề mặt.

Dröder và cộng sự [10] đã ứng dụng thuật machine leaning để hiệu chuẩn human-robot. Bielefeldt và cộng sự [11] đã đề xuất một phương pháp đánh giá không phá hủy (NDE) để phát hiện các vết nứt do mỏi. Một nghiên cứu điển hình về cánh máy bay chỉ ra rằng phương pháp này có thể giảm số lượng tính toán một cách hiệu quả. Seshadri và Krishnamurthy [12] đã sử dụng các phản ứng sóng có hướng dẫn để đưa ra các dự đoán trong thời gian thực. Họ tích hợp dữ liệu cảm biến, dữ liệu đầu vào và dữ liệu ảo để mô tả một đối tượng vật lý và chẩn đoán kích thước thiệt hại, vị trí và các thông tin hư hỏng khác.

Pairet et al. [13] giới thiệu công nghệ rô-bốt ngoài khơi sử dụng bộ mô phỏng trung tâm ORCA để đào tạo và thử nghiệm các giải pháp hợp tác giữa người và máy thống nhất ba loại hệ thống sản xuất trên nền tảng kỹ thuật số đôi. Voinov và cộng sự [14] đã nghiên cứu một phương pháp cung cấp khả năng quản lý đáng tin cậy của các hệ thống IoT phức tạp. Martinez et al. [15] đã nghiên cứu tác động của bộ đôi kỹ thuật số đối với đổi mới mô hình kinh doanh dịch vụ y tế bằng cách hiểu đầy đủ cách thiết lập, triển khai và sử dụng sức khỏe số và hiểu tác động của sức khỏe số trong kinh doanh dịch vụ doanh nghiệp. Biesinger và cộng sự [16] đã giới thiệu bộ đôi kỹ thuật số của hệ thống sản xuất body-in-white để đạt được sự tích hợp nhanh chóng của những chiếc xe hơi mới.

- Trong nước

Ở Việt Nam, việc nghiên cứu liên quan đến bản sao số cũng đã ngày càng thu hút được nhiều quan tâm của cộng đồng khoa học cũng như doanh nghiệp, đặc biệt là các doanh nghiệp công nghệ. Công ty Robot3T [17], một công ty khởi nghiệp đi đầu trong lĩnh vực robot tại Việt Nam đã phát triển được các sản phẩm robot hợp tác (Colab robot), ứng dụng cả trong công nghiệp và dịch vụ. Và gần đây, việc nghiên cứu phát triển hệ thống bản sao số cho robot cũng đang được công ty này quan tâm và nghiên cứu phát triển. Cho đến nay việc nghiên cứu phát triển các hệ thống robot thông minh đã và đang được nhiều nhóm nghiên cứu đẩy mạnh. Tuy nhiên các nghiên cứu này hiện nay chỉ tập

trung phát triển một số phần trong cấu trúc của một hệ thống bản sao số, như chỉ tập trung nghiên cứu phát triển các mô hình điều khiển thông minh [18-20] hoặc tập trung ứng dụng trí tuệ nhân tạo vào nhận dạng, xử lý ảnh [21-23] hoặc mô phỏng [24,25] hoặc phát triển các nền tảng IOT với công nghệ Edge Computing hoặc Cloud Computing [26-28]. Một bản sao số với cấu trúc tổng thể hiện nay vẫn chưa được nghiên cứu phát triển toàn diện tại Việt Nam.

iii. Mục tiêu đề tài

Nghiên cứu phát triển mô hình bản sao số cho robot. Trong đó tập trung:

- Xây dựng một cấu trúc bản sao số cho robot
- Phát triển mô hình số robot bao gồm mô hình hình học 3D và mô hình điều khiển.
- Tích hợp mô hình số Robot vào môi trường ảo, đồng thời kết nối vào nền tảng mã nguồn mở ROS.
- Kết nối, truyền thông dữ liệu 2 chiều online theo thời gian thực giữa hệ thống robot thực và robot ảo, trong đó tập trung dữ liệu về góc biến khớp, thông số vị trí của điểm cuối tay máy (end-effector) dựa trên nền tảng ROS thông qua bộ giao thức Socket TCP.

iv. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu:

Mô hình bản sao số của robot

- Phạm vi nghiên cứu:

Đề tài tập trung nghiên cứu kết nối, thu thập dữ liệu các biến khớp và truyền dữ liệu 2 chiều theo thời gian thực giữa mô hình số và mô hình vật lý.

v. Phương pháp nghiên cứu

- Nghiên cứu thực nghiệm (Experimental research)

Thiết lập các thí nghiệm và thực hiện kết nối và tương tác với robot thực để thu thập dữ liệu. Bằng cách tiến hành các thí nghiệm và kiểm soát các yếu tố để kiểm tra, đánh

giá và thu thập dữ liệu về hoạt động của robot. Các dữ liệu này sẽ phục vụ cho việc phân tích ở các bước tiếp theo.

- Nghiên cứu trường hợp (Case study research)

Chia nghiên cứu thành hai trường hợp: một trường hợp khi robot hoạt động bình thường, và một trường hợp khi robot gặp lỗi. Bên cạnh đó, xem xét một trường hợp khác khi robot hoạt động theo một quỹ đạo nhất định. Việc nghiên cứu các trường hợp này sẽ giúp bạn hiểu rõ hơn về hoạt động và hiệu suất của robot trong các tình huống khác nhau và làm dữ liệu cho việc tối ưu trong những bước phát triển sau.

- Nghiên cứu phân tích (Analytical research)

Tập trung vào việc phân tích và đánh giá các dữ liệu thu thập được từ hoạt động của robot. Sử dụng các phương pháp thống kê và phân tích để so sánh và tìm ra độ trễ, sai số giữa mô hình thực và mô hình ảo trong quá trình hoạt động của robot. Bằng cách phân tích dữ liệu, những nhận xét và kết luận về hiệu suất và độ chính xác của robot được đưa ra.

vi. Mô tả hệ thống

Hệ thống gồm 3 thành phần chính: (1) Phần ảo; (2) Phần thực; và (3) Phần xử lý trung tâm. Phần ảo là một phần mềm hỗ có khả năng tưởng tác, điều khiển và mô phỏng. Phần thực là đối tượng vật lý, trong đề tài là robot 2 bậc tự do, có khả năng truyền và nhận dữ liệu cảm biến và tín hiệu điều khiển. Phần xử lý trung tâm dùng để xử lý, kết nối giữa phần thực và phần ảo và lưu trữ dữ liệu. Việc sử bộ xử lý trung tâm giúp tăng khả năng mở rộng và ứng dụng của hệ thống. Việc kết nối và truyền thống dữ liệu 2 chiều online theo thời gian thực giữa bản sao số và bộ xử lý trung tâm thông qua socket TCP. Và việc kết nối và truyền thông dữ liệu giữa bộ điều khiển trên arduino mega 2560 và bộ xử lý trung tâm thông qua giao thức UART. Hoạt động của hệ thống như sau:

- Khi hệ thống đã khởi động thì bản sao số và robot thực sẽ kết nối với bộ xử lý trung tâm. Nếu kết nối thất bại thì hệ thống sẽ thông báo lỗi và chờ kết nối lại. Khi đã kết nối thành công hệ thống sẽ thiết lập các kết nối và dữ liệu bắt đầu được truyền giữa phần bản sao số và robot thực thông qua bộ xử lý trung tâm.

Đồng thời dữ liệu sẽ được xử lý và lưu trữ tại bộ nhớ cục bộ và sao lưu định kỳ lên đám mây.

- Khi người dùng điều khiển robot từ bản sao số, tín hiệu sẽ được gửi đến bộ điều khiển trung tâm. Ở đây tín hiệu sẽ được xử lý thành giá trị xung để truyền đến bộ điều khiển của robot thực. Bộ điều khiển sẽ tạo xung điều khiển để điều khiển mạch động lực từ đó điều khiển động cơ. Các dữ liệu về encoder được thu thập và truyền về bộ điều khiển để hình thành bộ điều khiển vòng kín. Các giá trị này sẽ được bộ điều khiển gửi đến bộ xử lý trung tâm. Tại đây dữ liệu cảm biến được xử lý và tính toán sau đó được truyền đến bản sao số. Khi nhận được dữ liệu, bộ điều khiển của bản sao số sẽ điều khiển mô hình đúng với các thông số nhận được. Ngoài ra các thông số cũng được thể hiện trên màn hình để người dùng theo dõi và giám sát.

vii. Nội dung nghiên cứu

Trong nghiên cứu này, một kiến trúc bản sao số được đề xuất để xây dựng và phát triển mô hình bản sao số cho robot.

Từ kiến trúc hệ thống, nghiên cứu được chia thành những nhiệm vụ chính như sau: Thiết kế và chế tạo robot arm; Xây dựng bản sao số cho mô hình robot; Kết nối và truyền thông dữ liệu; Đánh giá và kết luận.

PHẦN NỘI DUNG

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1.1. Bản sao số

Digital twin hay bản sao số là một đại diện kỹ thuật số của một đối tượng hoặc hệ thống vật lý. Về bản chất, digital twin là một chương trình máy tính lấy dữ liệu trong thế giới thực về một đối tượng hoặc hệ thống vật lý làm đầu vào và tạo ra các dự đoán hoặc mô phỏng đầu ra về cách đối tượng hoặc hệ thống vật lý đó sẽ bị ảnh hưởng bởi những đầu vào. Cơ sở lý thuyết của digital twin đến từ các ngành khoa học khác nhau như khoa học thông tin, kỹ thuật sản xuất, khoa học dữ liệu và khoa học máy tính. Các lý thuyết phù hợp nhất được chia làm 4 phần [1]: (1) mô hình digital twin, mô phỏng, xác minh, …; (2) Hợp nhất dữ liệu; (3) Tương tác và cộng tác; và (4) Dịch vụ.

1.2. Tổng quan robotics

- Khái niệm robotics

Định nghĩa theo tiêu chuẩn AFNOR (Pháp): Robot công nghiệp là một cơ cấu chuyển động tự động có thể lập trình, lặp lại các chương trình, tổng hợp các chương trình đặt ra trên các trục toạ độ; có khả năng định vị, định hướng, di chuyển các đối tượng vật chất: chi tiết, dao cụ, gá lắp . . . theo những hành trình thay đổi đã chương trình hoá nhằm thực hiện các nhiệm vụ công nghệ khác nhau.

Định nghĩa theo RIA (Robot institute of America): Robot là một tay máy vạn năng có thể lặp lại các chương trình được thiết kế để di chuyển vật liệu, chi tiết, dụng cụ hoặc các thiết bị chuyên dùng thông qua các chương trình chuyển động có thể thay đổi để hoàn thành các nhiệm vụ khác nhau.

Định nghĩa theo GOCT 25686-85 (Nga): Robot công nghiệp là một máy tự động, được đặt cố định hoặc di động được, liên kết giữa một tay máy và một hệ thống điều khiển theo chương trình, có thể lập trình lại để hoàn thành các chức năng vận động và điều khiển trong quá trình sản xuất.

Có thể nói Robot công nghiệp là một máy tự động linh hoạt thay thế từng phần hoặc toàn bộ các hoạt động cơ bắp và hoạt động trí tuệ của con người trong nhiều khả năng thích nghi khác nhau.

Robot công nghiệp có khả năng chương trình hoá linh hoạt trên nhiều trục chuyển động, biểu thị cho số bậc tự do của chúng. Robot công nghiệp được trang bị những bàn tay máy hoặc các cơ cấu chấp hành, giải quyết những nhiệm vụ xác định trong các quá trình công nghệ : hoặc trực tiếp tham gia thực hiện các nguyên công (sơn, hàn, phun phủ, rót kim loại vào khuôn đúc, lắp ráp máy . . .) hoặc phục vụ các quá trình công nghệ (tháo lắp chi tiết gia công, dao cụ, đồ gá . . .) với những thao tác cầm nắm, vận chuyển và trao đổi các đối tượng với các trạm công nghệ, trong một hệ thống máy tự động linh hoạt, được gọi là "Hệ thống tự động linh hoạt robot hoá" cho phép thích ứng nhanh và thao tác đơn giản khi nhiệm vụ sản xuất thay đổi.

- Bậc tự do

Bậc tự do là số khả năng chuyển động của một cơ cấu (chuyển động quay hoặc tịnh tiến). Để dịch chuyển được một vật thể trong không gian, cơ cấu chấp hành của robot phải đạt được một số bậc tự do. Nói chung cơ hệ của robot là một cơ cấu hở, do đó bậc tự do của nó có thể tính theo công thức:

$$w = 6n - \sum_{i=1}^{5} ip_i \quad (1.1)$$

Ở đây: n – Số khâu động

p_i – Số khớp loại i (i = 1, 2, …, 5: Số bậc tự do bị hạn chế).

Đối với các cơ cấu có các khâu được nối với nhau bằng khớp quay hoặc tịnh tiến (khớp động loại 5) thì số bậc tự do bằng với số khâu động... Đối với cơ cấu hở, số bậc tự do bằng tổng số bậc tự do của các khớp động.

Để định vị và định hướng khâu chấp hành cuối một cách tuỳ ý trong không gian 3 chiều robot cần có 6 bậc tự do, trong đó 3 bậc tự do để định vị và 3 bậc tự do để định hướng. Một số công việc đơn giản nâng hạ, sắp xếp... có thể yêu cầu số bậc tự do ít hơn. Các robot hàn, sơn... thường yêu cầu 6 bậc tự do. Trong một số trường hợp cần sự khéo

léo, linh hoạt hoặc khi cần phải tối ưu hoá quỹ đạo... người ta dùng robot với số bậc tự do lớn hơn 6.

- Hệ tọa độ

Mỗi robot thường bao gồm nhiều khâu (links) liên kết với nhau qua các khớp (joints), tạo thành một xích động học xuất phát từ một khâu cơ bản (base) đứng yên. Hệ toạ độ gắn với khâu cơ bản gọi là hệ toạ độ cơ bản (hay hệ toạ độ chuẩn). Các hệ toạ độ trung gian khác gắn với các khâu động gọi là hệ toạ độ suy rộng. Trong từng thời điểm hoạt động, các toạ độ suy rộng xác định cấu hình của robot bằng các chuyển dịch dài hoặc các chuyển dịch góc của các khớp tịnh tiến hoặc khớp quay. Các toạ độ suy rộng còn được gọi là biến khớp.

Các hệ toạ độ gắn trên các khâu của robot phải tuân theo qui tắc bàn tay phải: Dùng tay phải, nắm hai ngón tay út và áp út vào lòng bàn tay, xoè 3 ngón : cái, trỏ và giữa theo 3 phương vuông góc nhau, nếu chọn ngón cái là phương và chiều của trục z, thì ngón trỏ chỉ phương, chiều của trục x và ngón giữa sẽ biểu thị phương, chiều của trục.

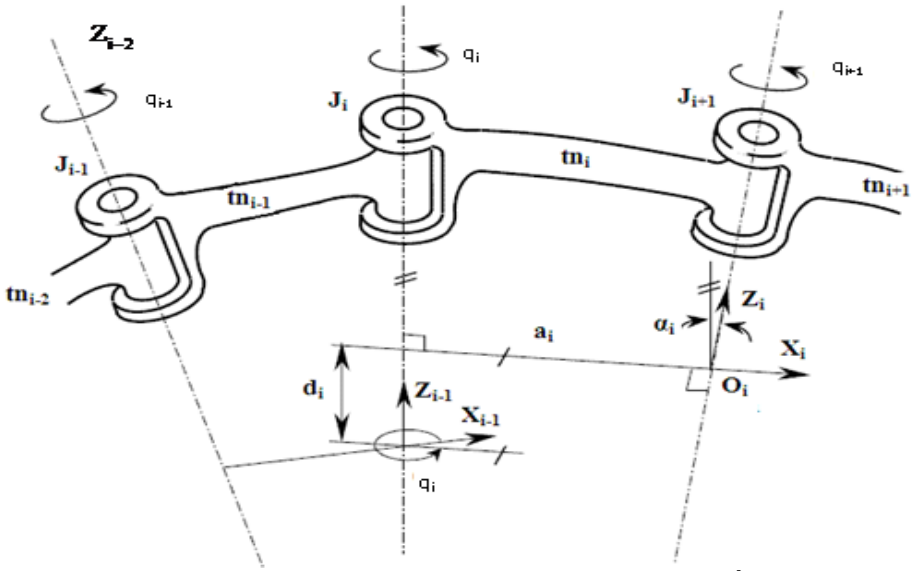
Trong robot ta thường dùng chữ O và chỉ số n để chỉ hệ toạ độ gắn trên khâu thứ n. Như vậy hệ toạ độ cơ bản (Hệ toạ độ gắn với khâu cố định) sẽ được ký hiệu là O0; hệ toạ độ gắn trên các khâu trung gian tương ứng sẽ là O1, O2,..., On-1, Hệ toạ độ gắn trên khâu chấp hành cuối ký hiệu là On.

- Động học thuận robot

Động học robot là bài toán nghiên cứu chuyển động của robot mà không quan tâm đến tính chất về lực tương tác cũng như về khối lượng mà chỉ xét đến cấu trúc hình học của robot. Mục đích của bài toán động học thuận là xác định vị trí của khâu tác động cuối của Robot khi biết các biến khớp của Robot. Các bước thực hiện bài toán động học thuận cho tay máy:

- Bước 1: Xác định số khớp và số thanh nối

- Bước 2: Gắn lên các thanh nối từ 0 đến n các hệ trục tọa độ. Ví dụ: Thanh nối i (i = 0 ÷ n) gắn hệ trục O_i, X_i, Y_i, Z_i



Hình 1.1. Xây dựng hệ tọa độ thanh nối

q_i: là góc quay của thanh nối thứ i

d_i: là độ lệch khâu,

a_i: là độ dài đường vuông góc chung giữaZ_{i-1}vàZ_i,

α_i: là góc vặn của thanh nối.

Cách xác định trụcZ_i: là trục mà xung quanh nó khớp thứ i+1 quay hoặc dọc theo nó khớp (i = 1÷ n-1) tịnh tiến.

Z_0: trục mà xung quanh nó khớp 1quay,

O_0: tâm hệ trục tọa độ quy chiếu, chọn một điểm cố định trên đế Robot,

Z_1: trục mà xung quanh nó khớp 2 quay hoặc khớp tịnh tiến,

Z_{n-1}: trục mà xung quanh nó khớp n quay,

Z_n: trùng phương với Z_{n-1}

Cách xác định trục X_i: Trục X thường được đặt dọc theo pháp tuyến chung và hướng từ khớp i đến i+1. Trong trường hợp các trục khớp cắt nhau thì trục X chọn theo tích vectơ Z_{i-1} x Z_i. Cách xác định trục Y_i: Xác định theo quy tắc bàn tay phải.

- Bước 3: Xác định các biến khớp

  Khớp quay tương ứng với biến khớp quay q, khớp tịnh tiến tương ứng với biến khớp tịnh tiến d.

- Bước 4: Xác định quan hệ giữa hai khung tọa độ i và i-1

  Hệ trục tọa độ i và hệ trục tọa độ i-1 giữa hai khâu nối tiếp nhau có quan hệ với nhau bằng phép biến đổi đồng nhất, theo trình tự sau:

  Quay xung quanh trục Z_{i-1} một góc θ_i sao cho trục X_{i-1} trùng với phương của trục X_i.

  Tịnh tiến dọc theo trục Z_{i-1} một đoạn d_i để gốc khung tọa độ mới trùng chân pháp tuyến chung trục i-1 và i, ($X_{i-1} \equiv X_i$).

  Tịnh tiến dọc theo trục X_{i-1} một đoạn a_i, ($O_{i-1} \equiv O_i$).

  Quay xung quanh trục X_{i-1} một góc α_i sao cho trục Z_{i-1} trùng với trục Z_i.

  Các phép biến đổi trên được thực hiện so với khung tọa độ hiện tại, do đó phép biến đổi tổng hợp được xác định như sau :

$$A_i = \mathrm{Rot}_z(q)\mathrm{Trans}_z(d)\mathrm{Trans}_x(a)\mathrm{Rot}_x(\alpha)$$

$$^{i-1}A_i = Rot_z(\theta)Trans_z(d)Trans_x(a)Rot_x(\alpha)$$

$$= \begin{bmatrix} c\theta_i & -s\,\theta_i & 0 & 0 \\ s\,\theta_i & c\,\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\,\alpha_i & -s\alpha_i & 0 \\ 0 & s\alpha_i & c\,\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a\cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.2)$$

  Denavit J.& Hartenberg R.S đã gọi biến đổi đồng nhất là ma trận A, mô tả bởi phép quay và phép tịnh tiến tương đối giữa hệ tọa độ của hai khâu liên tiếp. Denavit J.& Hartenberg R.S đề xuất dùng ma trận thuần nhất 4x4 để mô tả quan hệ giữa 2 khâu liên tiếp trong cơ cấu không gian. Pieper D.L đã dùng ma trận

thuần nhất 4x4 trong nghiên cứu Robot. Trong bài toán động học Robot định luật Denavit Hartenber đóng vai trò rất quan trọng. Định luật này cho chúng ta cách xác định các hệ trục toạ độ đặt lên các khớp và từ đó xác định được vị trí của các thanh nối và khâu tác động cuối trong các hệ trục toạ độ.

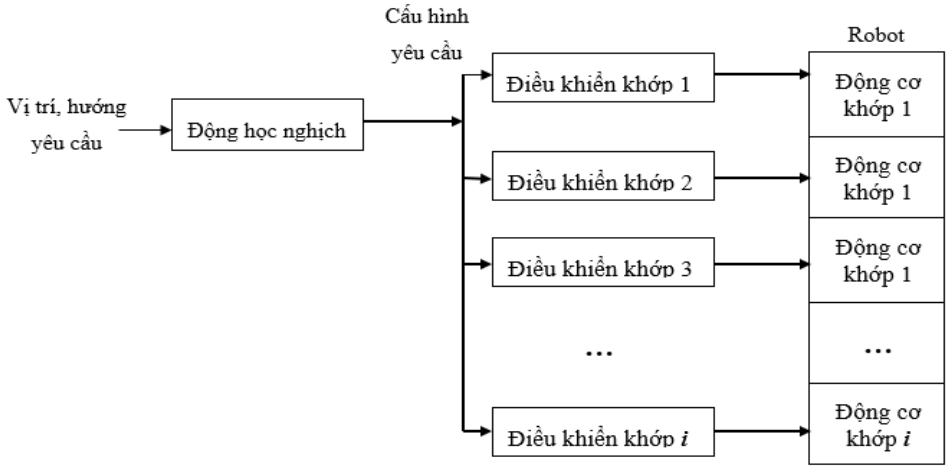
- Bước 5: Xác định phương trình động học thuận cho tay máy

Ma trận đồng nhất mô tả hướng và vị trí của Robot trong hệ tọa độ {O}

$$T_n = T_n^0 = A_1^0.A_2^1.A_3^2 \dots .A_n^{n-1} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.3)$$

- Động học nghịch robot

Bởi vì tính chất quan trọng trong việc điều khiển robot nên bài toán động học nghịch được nghiên cứu từ những năm 1960. Sự phức tạp của bài toán phụ thuộc vào cấu trúc của robot. Ở đây số bậc của robot đóng vai trò quan trọng trong bài toán này. Sự phức tạp của bài toán động học nghịch tăng khi số lượng bậc tự do tăng. Sắp xếp các trục khớp cũng ảnh hưởng khá lớn đến độ phức tạp của bài toán động học nghịch. Một số robot có cấu trúc cho các phương trình của bài toán động học thuận đơn giản có thể giải bài toán động học nghịch bằng phương pháp đại số. Pieper vào năm 1968 đã chỉ ra rằng bất cứ robot sáu bậc tự do nào có ba trục liên tiếp song song hoặc vuông góc nhau thì có thể giảm thành đa thức bậc 4. Thì bài toán robot có thể đạt được với dạng vòng kín.



Hình 1.2. Mô hình bài toán động học nghịch

Để điều khiển robot di chuyển theo các vị trí mong muốn của tay máy robot trong không gian, cần xác định các giá trị biến khớp tương ứng với vị trí và hướng của tay robot mong muốn. Đây là nội dung của bài toán động học ngược. Bài toán động học ngược thường khó giải và không có lời giải tổng quát cho mọi robot. Ta có thể khái quát động học ngược như sau:

$$A_1A_2A_3 \dots A_i = P \quad (1.4)$$

Với $\boldsymbol{A_i = A_i(q_i)}$ là hàm của biến $\boldsymbol{q_i}$ là ma trận chuyển từ hệ trục i-1 đến i và P là ma trận vị trí và hướng của cơ cấu chấp hành cuối. Chúng ta có phương trình mô tả tương đương vị trí và hướng như sau:

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{n} & \boldsymbol{o} & \boldsymbol{a} & \boldsymbol{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.5)$$

Quá trình giải bài toán động học nghịch:

Khi quân tích động học của robot, bước đầu tiên luôn phải giải bài toán động học thuận liên hệ với vị trí và hướng của cơ cấu chấp hành cuối như là giá trị đã biết với các biến khớp. Các biến khớp với các khớp quay $q_i = \theta_i$ và khớp tịnh tiến $q_i = d_i$. Với robot n khớp. phương trình động học thuận là:

$$P = f(q_1, q_2, q_3, \dots, q_n) \quad (1.6)$$

Các phương trình của bài toán động học thuận sau khi tính toán sẽ được xem xét cẩn thận để xác định xem có biến nào có thể tính toán từ các phương trình này không. Tuy nhiên, các phương trình động học là phi tuyến vì vậy không phải dễ dàng giải. Thường sẽ sử dụng một vài phương trình để giải bằng phương pháp đại số. Mục đích là đưa ra các phương trình để giải các biến khớp một cách dễ nhất. Phương pháp đại số để giải bài toán ngược như sau:

Giả sử ta muốn đưa tay máy đến một vị trí gọi là $P(p_x, p_y, p_z)$.Ta có ma trận biểu diễn hướng và vị trí của tay máy tại vị trí đó như sau:

$$T_n = A_1^0.A_2^1.A_3^2 \dots .A_n^{n-1} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.7)$$

Trong đó:

$\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$ là ma trận biểu diễn hướng của tay máy.

$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$ là ma trận biểu diễn vị trí của tay máy.

Nhân cả hai vế cho ma trận nghịch đảo A_1^{-1} ta được:

$$A_1^{-1}.T_n = A_2^1.A_3^2 \dots .A_n^{n-1} \quad (1.8)$$

$$A_1^{-1}.\begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = A_2^1.A_3^2 \dots .A_n^{n-1} \quad (1.9)$$

Cứ như vậy lần lượt ta nhân các ma trận nghịch đảo của từng khớp để tìm được hết tất cả các biến khớp θ_i. Nhiều phương trình của bài toán động học có dạng:

$$a.s_i + b.c_i = d \quad (1.10)$$

Với a, b, d là giá trị đã biết hoặc hằng số và s_i, c_i là giá trị sin, cos của biến khớp θ_i. Ngoài ra, chúng ta thường chuển qua đơn biến như sau:

$$u = \tan\frac{\theta_i}{2} \quad (1.11)$$

$$sin\theta_i = \frac{1-u^2}{1+u^2} \quad (1.12)$$

$$cos\theta_i = \frac{2u}{1+u^2} \quad (1.13)$$

Thay phương trình (2.5-2.8), chúng ta có:

$$a(1-u^2) + 2bu = d(1+u^2) \quad (1.14)$$

Sắp xếp`theo bậc của biến u:

$$(a+c)u^2 + 2bu + (d-a) = 0 \quad (1.15)$$

Giải phương trình bậc 2 chúng ta có:

$$u = \frac{b \pm \sqrt{b^2 + a^2 - d^2}}{a+c} \quad (1.16)$$

Khi $a^2 + b^2 \geq d^2$ thì phương trình (10) có 2 lời giải như sau:

$$\theta_1 = atan2\left(d; \pm\sqrt{a^2 + b^2 - d^2}\right) - atan2(b; a) \quad (1.17)$$

Các trường hợp chúng ta xác định theo định lý Pythagore:

$$S_i^2 + C_i^2 = 1 \ (1.18)$$

Chúng ta có thể giải theo lượng giác. Với a=0 thì:

$$c_i = \frac{d}{b} \rightarrow s_i = \pm\sqrt{(1 - c_i^2)} \ (1.19)$$

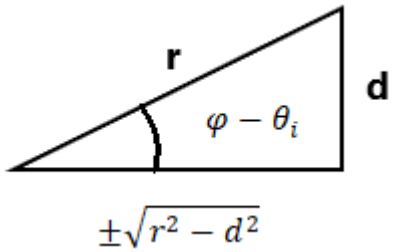
Tương tự ta có:

$$s_i = \frac{d}{b} \rightarrow c_i = \pm\sqrt{(1 - s_i^2)} \quad (1.21)$$

$$\rightarrow \theta_i = atan2(c_i; s_i) = atan2\left[\pm\sqrt{\left(1 - \frac{d^2}{b^2}\right)}, \frac{d}{b}\right] \ (1.22)$$

Đối với dạng $p_y \cos(\theta_i) - p_x \sin(\theta_i) = d$, ta thay $p_x = r.\cos(\varphi)$, $p_y = r.\sin(\varphi)$ vào phương trình (21) (trong đó : $\varphi = atan2(p_y, p_x)$ và $r^2 = {p_x}^2 + {p_y}^2$).

Phương trình sẽ có dạng: $\sin(\varphi - \theta_i) = d/r$. Ta dựa vào góc lượng giác và định lý Pythagore để xác định được biến θ_i:



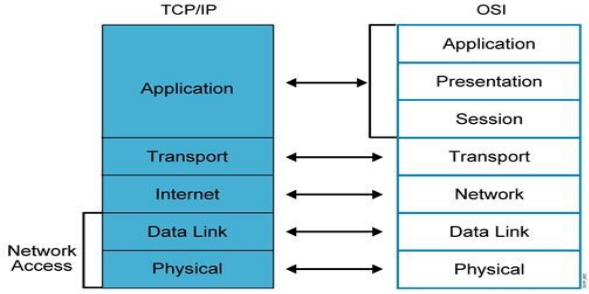
Hình 1.3. Góc lượng giác tính toán θ_i

1.3. Giao thức truyền thông

- Mô hình TCP/IP

Mô hình TCP/IP tiêu chuẩn bao gồm 4 tầng được chồng lên nhau, bắt đầu từ tầng thấp nhất là:

- Tầng 1: Tầng vật lý (Physical): Còn được gọi là tầng liên kết dữ liệu là tầng thấp nhất trong mô hình TCP/IP. Tầng này chịu trách nhiệm truyền dữ liệu giữa hai thiết bị trong cùng một mạng. Tại đây, các gói dữ liệu được đóng vào khung (gọi là Frame) và được định tuyến đi đến đích đã được chỉ định ban đầu.
- Tầng 2: Tầng mạng (Network): Tầng Internet đảm nhận việc truyền tải dữ liệu một cách hợp lý. Các giao thức của tầng này bao gồm IP (Internet Protocol), ICMP (Internet Control Message Protocol), IGMP (Internet Group Message Protocol).
- Tầng 3: Tầng giao vận (Transport): Tầng dữ liệu hoạt động thông qua hai giao thức chính là TCP (Transmission Control Protocol) và UDP (User Datagram Protocol). TCP sẽ đảm bảo chất lượng truyền gửi gói tin, tuy nhiên lại mất thời gian khá lâu để thực hiện các thủ tục kiểm soát dữ liệu. Ngược lại, UDP lại cho tốc độ truyền tải nhanh nhưng lại không đảm bảo được chất lượng dữ liệu. Ở tầng này, TCP và UDP sẽ hỗ trợ nhau phân luồng dữ liệu.
- Tầng 4: Tầng ứng dụng (Application): Tầng Application hay còn gọi là tầng ứng dụng. Tầng ứng dụng đảm nhận vai trò giao tiếp dữ liệu giữa 2 máy khác nhau thông qua các dịch vụ mạng khác nhau (duyệt web, chay hay các giao thức trao đổi dữ liệu SMTP, SSH, FTP…). Dữ liệu khi đến được tầng 4 sẽ được định dạng để kết nối theo kiểu Byte nối Byte. Các thông tin định tuyến tại đây sẽ giúp xác định đường đi đúng của một gói tin.



Hình 1.4. Cấu trúc TCP/IP và OSI

- TCP/IP
  - Giao thức truyền thông TCP/IP (Transmission Control Protocol/Internet Protocol) là bộ giao thức tiêu chuẩn được sử dụng để liên lạc và truyền thông giữa các thiết bị trong mạng Internet. Nó cung cấp một cơ sở lý thuyết và quy tắc để phân đoạn, gói tin hóa, gửi và nhận dữ liệu qua mạng. Dưới đây là một số khái niệm cơ bản và cơ sở lý thuyết của giao thức truyền thông TCP/IP.
- Giao thức IP (Internet Protocol)
  - IP là một giao thức lớp mạng trong mô hình TCP/IP.
  - Nhiệm vụ chính của giao thức IP là định vị và định tuyến gói tin qua mạng dựa trên địa chỉ IP của các thiết bị.
  - Giao thức IP đảm bảo việc gửi và nhận các gói tin qua mạng Internet.
  - Giao thức TCP (Transmission Control Protocol):
  - TCP là một giao thức lớp giao vận trong mô hình TCP/IP.
  - Giao thức TCP tạo và quản lý các kết nối tin cậy (reliable) giữa các thiết bị trong mạng.
  - Nó đảm bảo việc chia nhỏ và lắp ghép các đoạn (segments) dữ liệu, xác nhận nhận dạng gói tin, điều khiển luồng dữ liệu và khắc phục các lỗi truyền thông.
- Giao thức UDP (User Datagram Protocol)
  - UDP là một giao thức lớp giao vận trong mô hình TCP/IP.
  - UDP cung cấp một dịch vụ không tin cậy (unreliable) cho việc truyền thông qua mạng.
  - Không giống như TCP, giao thức UDP không đảm bảo tính toàn vẹn dữ liệu, xác nhận gói tin và điều khiển luồng. Nó giúp truyền dữ liệu nhanh chóng và hiệu quả, nhưng không đảm bảo dữ liệu sẽ được gửi và nhận đúng trình tự.
- Địa chỉ IP

- Mỗi thiết bị trong mạng Internet được gán một địa chỉ IP duy nhất để xác định và định vị nó trong mạng.
- Địa chỉ IP gồm hai phần: địa chỉ mạng (network address) và địa chỉ máy (host address).
- Địa chỉ IP có thể là IPv4 (32-bit) hoặc IPv6 (128-bit) dựa trên phiên bản giao thức IP được sử dụng.
- Giao thức HTTP (Hypertext Transfer Protocol):
- HTTP là một giao thức ứng dụng trong mô hình TCP/IP.
- Giao thức HTTP được sử dụng để truyền tải và truy cập các trang web và dữ liệu trên Internet.
- Nó hoạt động trên cơ sở gửi yêu cầu từ khách hàng (client) và nhận và trả lời yêu cầu từ máy chủ (server).
- Các khái niệm trên cung cấp một cái nhìn tổng quan về cơ sở lý thuyết của giao thức truyền thông TCP/IP. Giao thức này là cơ sở cho việc truyền thông và giao tiếp trong mạng Internet, và nó đóng vai trò quan trọng trong việc truyền tải dữ liệu và xác định địa chỉ của các thiết bị trong mạng.

- UART
  - UART (Universal Asynchronous Receiver/Transmitter) là một giao thức truyền thông được sử dụng để giao tiếp giữa các thiết bị trong hệ thống điện tử. Nó là một giao thức đồng bộ hướng dòng (serial) có mục đích chính là truyền dữ liệu qua một đường truyền duy nhất.
  - Giao thức UART sử dụng hai tín hiệu chính là tín hiệu TX (Transmit) để truyền dữ liệu và tín hiệu RX (Receive) để nhận dữ liệu. Điều này cho phép hai thiết bị kết nối trực tiếp với nhau để truyền thông tin qua một đường dây đơn.
  - UART hoạt động ở chế độ không đồng bộ, nghĩa là không có tín hiệu đồng hồ (clock) chung giữa các thiết bị. Thay vào đó, nó sử dụng một chuỗi bit khởi đầu (start bit) để đồng bộ hóa dữ liệu và một hoặc nhiều bit dừng (stop bit) để phân

biệt các khung dữ liệu. Dữ liệu được truyền trong các khung dữ liệu có độ dài cố định, thường là 8 bit, và được gửi theo thứ tự từ bit thấp đến bit cao (LSB-first).

- Giao thức UART hỗ trợ nhiều tốc độ truyền khác nhau, như 9600 bps (bit per second), 115200 bps, 1 Mbps, v.v. Tốc độ truyền thông thường được đặt trước khi gửi dữ liệu và cần phải khớp với cấu hình tốc độ của cả hai thiết bị để đảm bảo truyền thông chính xác.
- UART thường được sử dụng trong nhiều ứng dụng như truyền thông giữa vi xử lý và các thiết bị ngoại vi như cảm biến, mạch điều khiển, module truyền thông không dây và nhiều thiết bị điện tử khác.

1.4. Cảm biến và thuật toán điều khiển

- Encoder

Encoder hay còn gọi là bộ mã hóa, là một bộ cảm biến chuyển động cơ học tạo ra tín hiệu kỹ thuật số đáp ứng với chuyển động. Là một thiết bị cơ điện có khả năng làm biến đổi chuyển động thành tín hiệu số hoặc xung. Có hai loại bộ mã hóa: tuyến tính và quay. Encoder tuyến tính đáp ứng chuyển động dọc theo một đường dẫn, còn Encoder quay thì đáp ứng với chuyển động quay.

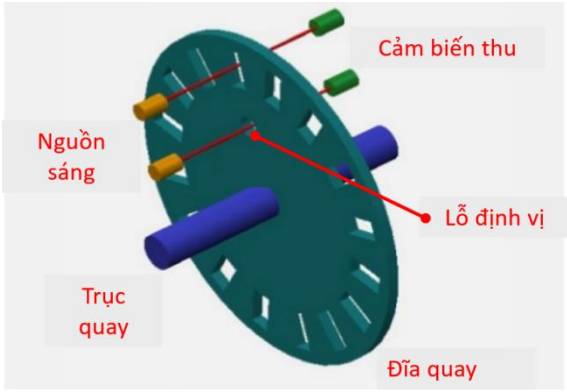
Encoder mục đích dùng để quản lý vị trí góc của một đĩa quay, có thể là đĩa quay của bánh xe, trục động cơ hay bất kỳ thiết bị nào cần xác định vị trí góc. Một bộ mã hóa thường được phân loại theo các phương tiện đầu ra của nó, gồm 2 loại chính: Encoder tuyệt đối và Encoder tương đối.

Encoder tuyệt đối (absolute encoder) sử dụng đĩa theo mã nhị phân hoặc mã Gray. Có kết cấu gồm: bộ phát ánh sáng (LED), đĩa mã hóa (có chứa dải băng mang tín hiệu), một bộ thu ánh sáng nhạy với ánh sáng phát ra. Đĩa mã hóa ở Encoder được chế tạo từ vật liệu trong suốt, người ta đã chia mặt đĩa thành các góc đều nhau cùng các đường tròn đồng tâm. Ưu điểm: giữ được giá trị tuyệt đối khi Encoder mất nguồn. Nhược điểm: giá thành cao vì chế tạo phức tạp, đọc tín hiệu khó.

Encoder tương đối (incremental encoder): phát ra tín hiệu tăng dần hoặc theo chu kỳ. Đĩa mã hóa bao gồm một dải băng tạo xung, thường được chia thành nhiều lỗ bằng nhau

và được cách đều nhau. Chất liệu có thể là trong suốt để giúp ánh sáng chiếu qua. Là Encoder chỉ có 1,2 hoặc tối đa 3 vòng lỗ, và thường có thêm một lỗ định vị. Ưu điểm: giá thành rẻ, chế tạo đơn giản, xử lý tín hiệu trả về dễ dàng. Nhược điểm: dễ bị sai lệch về xung khi trả về. Sẽ tích lũy sai số khi hoạt động lâu dài.

Nguyên lý hoạt động: Khi Encoder chuyển động bộ chuyển đổi sẽ xử lý các chuyển động và chuyển thành các tín hiệu điện. Các tín hiệu này sẽ được truyền đến các thiết bị điều khiển PLC và được xử lý để biểu thị các giá trị cần đo bằng chương trình riêng biệt. Đối với các tín hiệu có ánh sáng chiếu qua hay không có ánh sáng chiếu qua. Người ta vẫn có thể ghi nhận được đèn Led có chiếu qua lỗ này hay không. Hơn thế nữa, số xung đếm được và tăng lên được tính bằng số lần mà ánh sáng bị cắt. Ví dụ: trên đĩa có 1 lỗ duy nhất, khi mỗi lần con mắt thu nhận được 1 tín hiệu đèn Led thì có nghĩa là đĩa đã quay được 1 vòng.



Hình 1.5. Cấu tạo của đĩa quay trong encoder

Bởi vậy, đây chính là nguyên lý hoạt động của Encoder cơ bản, còn đối với nhiều chủng loại Encoder khác thì khi đĩa quay có nhiều lỗ hơn khi đó tín hiệu thu nhận sẽ khác hơn.

Bộ mã hóa Encoder trở thành một nguồn quan trọng cho nhiều ứng dụng. Dù là liên quan đến tốc độ, hướng hay khoảng cách, khả năng của Encoder cho phép người sử dụng thông tin này để kiểm soát chính xác. Encoder có thể ứng dụng biểu thị tốc độ, đo lường, đếm số lượng hay trong các lĩnh vực và ngành nghề như cơ khí, ô tô, ….

- PID

Bộ điều khiển PID (Proportional-Integral-Derivative) là một phương pháp điều khiển phản hồi phổ biến trong hệ thống điều khiển tự động. Nó kết hợp ba thành phần chính:

tỷ lệ (Proportional), tích phân (Integral) và vi sai (Derivative), để điều chỉnh đầu ra điều khiển dựa trên sự chênh lệch giữa giá trị đặt và giá trị đo của hệ thống. Dưới đây là mô tả cơ sở lý thuyết của mỗi thành phần trong bộ điều khiển PID:

- Tỷ lệ (Proportional):

  Thành phần tỷ lệ phản hồi trực tiếp vào sự chênh lệch (error) giữa giá trị đặt (setpoint) và giá trị đo hiện tại của hệ thống.

  Tỷ lệ tỷ lệ thuận với error và tạo ra tín hiệu điều khiển tỷ lệ với độ lớn của error.

  Điều chỉnh tham số tỷ lệ (Kp) sẽ ảnh hưởng trực tiếp đến tốc độ đáp ứng và độ ổn định của hệ thống. Tham số Kp quá lớn có thể gây dao động và không ổn định, trong khi Kp quá nhỏ có thể làm cho hệ thống chậm và không đạt được giá trị đặt mong muốn.

- Tích phân (Integral):

  Thành phần tích phân tính toán và tích lũy các giá trị error theo thời gian.

  Tích phân giúp loại bỏ sự chênh lệch trung bình (error trung bình) và đưa hệ thống về một trạng thái cân bằng.

  Điều chỉnh tham số tích phân (Ki) ảnh hưởng đến khả năng đạt được độ chính xác ổn định và khả năng phục hồi của hệ thống. Tham số Ki quá lớn có thể gây dao động và không ổn định, trong khi Ki quá nhỏ có thể làm cho hệ thống chậm và không đạt được giá trị đặt mong muốn.

- Vi sai (Derivative):

  Thành phần vi sai ước lượng tỷ lệ thay đổi của error theo thời gian.

  Vi sai giúp dự đoán xu hướng thay đổi của hệ thống và giảm tác động của thay đổi nhanh chóng trên đầu ra điều khiển.

  Điều chỉnh tham số vi sai (Kd) ảnh hưởng đến khả năng kháng nhiễu và tốc độ đáp ứng của hệ thống. Tham số Kd quá lớn có thể làm hệ thống nhạy cảm với nhiễu và dao động, trong khi Kd quá nhỏ có thể không đủ để ổn định và đáp ứng tốt.

Kết hợp ba thành phần trên, bộ điều khiển PID tính toán đầu ra điều khiển bằng cách tổng hợp các thành phần tỷ lệ, tích phân và vi sai như sau:

$$u(v) = Kp * e + Ki * \int e(t) + Kd * e(t)\frac{d}{dt} \quad (1.23)$$

Trong công thức trên, Kp, Ki và Kd là các tham số điều chỉnh của bộ điều khiển PID. Các tham số này được tinh chỉnh để đạt được hiệu suất tốt và ổn định của hệ thống điều khiển.

Qua việc điều chỉnh các tham số Kp, Ki và Kd, bộ điều khiển PID có thể điều khiển hệ thống để đạt được đáp ứng chính xác, ổn định và nhanh chóng đối với các biến đổi và nhiễu trong hệ thống điều khiển tự động.

- Thuật toán Ziegler-Nichols

Thuật toán Ziegler-Nichols là một phương pháp phổ biến được sử dụng để tinh chỉnh các thông số của bộ điều khiển PID (Proportional-Integral-Derivative) trong hệ thống điều khiển. Phương pháp này được đề xuất bởi John G. Ziegler và Nathaniel B. Nichols vào năm 1942. Thuật toán Ziegler-Nichols dựa trên phản hồi hệ thống và sử dụng phản ứng của hệ thống để xác định các thông số PID tối ưu. Thuật toán Ziegler-Nichols được mô tả dưới đây:

- Chế độ tinh chỉnh:

  Bước đầu tiên trong thuật toán Ziegler-Nichols là đặt bộ điều khiển vào chế độ tinh chỉnh (autotuning) bằng cách vô hiệu hóa phần tử tích phân (Integral) và phần tử vi sai (Derivative) của PID, chỉ để lại phần tử tỷ lệ (Proportional).

  Trong chế độ này, bộ điều khiển sẽ chỉ sử dụng thông số Kp (hệ số tỷ lệ) và điều chỉnh giá trị của Kp để tạo ra tín hiệu điều khiển phù hợp cho hệ thống.

- Phương pháp Relay:

  Trong chế độ tinh chỉnh, bộ điều khiển sẽ đặt tín hiệu điều khiển đầu ra thành một tín hiệu Relay với hai mức cao (high) và thấp (low) cố định.

Ban đầu, tín hiệu Relay sẽ được đặt ở mức thấp. Khi giá trị đầu ra của hệ thống vượt qua một ngưỡng (relay high), tín hiệu Relay sẽ chuyển sang mức cao. Ngược lại, khi giá trị đầu ra giảm xuống dưới một ngưỡng (relay low), tín hiệu Relay sẽ chuyển về mức thấp.

Quá trình chuyển mạch của tín hiệu Relay sẽ tạo ra một đáp ứng dao động của hệ thống.

- Xác định các thông số PID tối ưu:

Dựa trên đáp ứng dao động của hệ thống, thuật toán Ziegler-Nichols sử dụng ba phương pháp khác nhau để xác định các thông số PID tối ưu: phương pháp Relay dạng vạch (Ultimate Gain Method), phương pháp Relay dạng vòng (Ultimate Period Method) và phương pháp Relay dạng bán-vạch (Ultimate Semi-Period Method).

Trong phạm vi câu trả lời này, chúng ta sẽ tập trung vào phương pháp Relay dạng vạch (Ultimate Gain Method), phương pháp thông dụng nhất.

- Phương pháp Relay dạng vạch:

Trong phương pháp Relay dạng vạch, ta đo hai giá trị sau quá trình chuyển mạch Relay:

Amplitude: Độ lớn của đáp ứng dao động, đo từ giá trị đặt tới đỉnh hoặc từ giá trị đặt tới đáy.

Period: Chu kỳ hoặc thời gian một chu kỳ đáp ứng dao động.

Dựa trên hai giá trị này, ta có thể tính toán các thông số PID tối ưu như sau:

- Kp (Proportional Gain):

$$Kp = 0.6 * Kc \quad (1.24)$$

- Ki (Integral Gain):

$$Ki = 1.2 * Kc * Tu \quad (1.25)$$

- Kd (Derivative Gain):

$$Kd = 0.075 * Kc * Tu \quad (1.26)$$

Trong đó, Kc là hệ số tỷ lệ tương ứng với đáp ứng dao động (Amplitude), và Tu là chu kỳ hoặc thời gian một chu kỳ đáp ứng dao động (Period).

- Tinh chỉnh và kiểm tra:

Sau khi xác định được các thông số PID tối ưu, bạn có thể đặt lại bộ điều khiển với các thông số này.

Tiếp theo, bạn nên kiểm tra và điều chỉnh lại các thông số nếu cần thiết để đảm bảo rằng hệ thống PID hoạt động tốt trên nhiều giá trị error khác nhau.

Phương pháp Ziegler-Nichols Relay là một phương pháp đơn giản và hiệu quả để tinh chỉnh PID, nhưng cần lưu ý rằng nó có thể tạo ra đáp ứng dao động trong quá trình tinh chỉnh. Do đó, việc kiểm tra và điều chỉnh thêm có thể được thực hiện để đảm bảo rằng hệ thống PID hoạt động ổn định và đáp ứng mong muốn trên nhiều giá trị error khác nhau.

1.5. Phần mềm và nền tảng hỗ trợ

- Solidworks

SolidWorks là một phần mềm thiết kế và mô phỏng 3D được sử dụng rộng rãi trong lĩnh vực kỹ thuật và công nghiệp. Nó được phát triển bởi Dassault Systèmes và ra mắt lần đầu vào năm 1995. SolidWorks cung cấp một giao diện người dùng dễ sử dụng và nhiều công cụ mạnh mẽ để thiết kế và mô phỏng các sản phẩm 3D.

Với SolidWorks, người dùng có thể tạo ra các bản vẽ kỹ thuật, mô hình 3D, và bộ mô phỏng động để kiểm tra tính năng và hiệu suất của sản phẩm trước khi đưa vào sản xuất. Phần mềm này hỗ trợ nhiều kỹ thuật thiết kế, bao gồm thiết kế cơ khí, điện tử, điện, và ngành công nghiệp chế tạo. Các tính năng chính của SolidWorks bao gồm:

- Thiết kế 3D: SolidWorks cho phép người dùng tạo ra các mô hình 3D chi tiết và tổ hợp sản phẩm.
- Tạo bản vẽ kỹ thuật: Người dùng có thể tạo bản vẽ kỹ thuật 2D từ các mô hình 3D và trình bày thông tin chi tiết về sản phẩm.

- Mô phỏng và kiểm tra tính toàn vẹn: SolidWorks cung cấp các công cụ mô phỏng để kiểm tra tính toàn vẹn cơ học, độ bền, và độ tin cậy của sản phẩm.
- Hợp tác và quản lý dự án: Phần mềm cho phép nhiều người dùng làm việc cùng nhau trên cùng một dự án và quản lý phiên bản sản phẩm.
- Tích hợp sản xuất: SolidWorks hỗ trợ kết nối với các công cụ gia công và máy móc để chuyển đổi mô hình 3D thành các chương trình gia công và máy gia công.

SolidWorks đã trở thành một trong những phần mềm thiết kế và mô phỏng 3D phổ biến nhất trên thế giới và được sử dụng trong nhiều lĩnh vực, bao gồm ô tô, hàng không vũ trụ, công nghệ y tế, điện tử, và chế tạo.

- ROS

ROS (Robot Operating System) là một hệ thống phần mềm mã nguồn mở được phát triển để hỗ trợ xây dựng và điều khiển các robot. Dưới đây là cơ sở lý thuyết cơ bản về ROS:

- Kiến trúc ROS:

  ROS được thiết kế dựa trên kiến trúc phân tán (distributed architecture). Nó cho phép các phần mềm và module riêng lẻ giao tiếp với nhau thông qua giao thức truyền thông ROS.

  Kiến trúc phân tán của ROS cho phép phát triển, triển khai và quản lý các thành phần riêng lẻ của hệ thống robot một cách linh hoạt và dễ dàng.

- Node và Topic:

  Trong ROS, các chương trình chạy độc lập được gọi là nodes. Mỗi node thực hiện một công việc cụ thể như điều khiển động cơ, xử lý dữ liệu cảm biến, hoặc hiển thị giao diện người dùng.

  Các nodes giao tiếp với nhau bằng cách gửi và nhận các message thông qua các topic. Một topic là một kênh truyền thông giữa các nodes để truyền tải thông tin như dữ liệu cảm biến, tín hiệu điều khiển, hoặc trạng thái.

- Service:

  Ngoài giao tiếp thông qua topic, ROS cung cấp cơ chế gọi dịch vụ (service) giữa các nodes. Một service cho phép một node yêu cầu một dịch vụ từ một node khác và nhận được kết quả trả về.

  Cơ chế dịch vụ giúp các nodes tương tác trực tiếp với nhau và chia sẻ các chức năng và tài nguyên.

- Package và Catkin:

  ROS sử dụng khái niệm package để tổ chức và quản lý mã nguồn, các tài nguyên và các thành phần của một dự án.

  Công cụ build system của ROS được gọi là Catkin. Nó cho phép xây dựng và quản lý các package và phụ thuộc giữa chúng.

- Launch file:

  Launch file là một tập tin cấu hình XML được sử dụng để khởi chạy và cấu hình một hoặc nhiều nodes cùng lúc. Nó giúp đơn giản hóa việc khởi động và cấu hình các thành phần của hệ thống ROS.

- Công cụ và thư viện hỗ trợ:

  ROS cung cấp nhiều công cụ và thư viện hỗ trợ phát triển và kiểm thử robot, bao gồm các công cụ quản lý gói, công cụ mô phỏng, công cụ kiểm tra và gỡ lỗi.

- Unity

Unity là một công cụ phát triển trò chơi và ứng dụng tương tác 3D/2D đa nền tảng. Dưới đây là cơ sở lý thuyết cơ bản về Unity:

- Game Object:

  Trong Unity, mọi đối tượng trong một trò chơi hay ứng dụng được gọi là Game Object.

  Mỗi Game Object có các thuộc tính và thành phần riêng, và có thể chứa các Game Object con.

- Component:

  Component là các thành phần cơ bản của một Game Object trong Unity.

  Mỗi Component đại diện cho một chức năng hoặc tính năng cụ thể như hiển thị đồ họa, xử lý vật lý, xử lý logic, hoặc điều khiển âm thanh.

- Scene:

  Scene là một màn chơi hoặc môi trường 3D/2D trong Unity.

  Một Scene có thể chứa nhiều Game Object và được sử dụng để tổ chức và quản lý các thành phần và tài nguyên trong trò chơi hay ứng dụng.

- Script:

  Script là một tập tin mã nguồn viết bằng ngôn ngữ lập trình như C# hoặc JavaScript.

  Script được sử dụng để xử lý logic và tương tác với các thành phần của Game Object trong Unity.

- Physics:

  Unity hỗ trợ hệ thống vật lý tích hợp sẵn.

  Hệ thống vật lý Unity cho phép xử lý va chạm, tương tác vật lý, và mô phỏng các hiệu ứng vật lý trong trò chơi hay ứng dụng.

- Animation:

  Unity cho phép tạo và quản lý các hoạt cảnh (animation) cho các Game Object.

  Hoạt cảnh Unity có thể được sử dụng để tạo ra chuyển động, biểu cảm và hiệu ứng đặc biệt cho các đối tượng trong trò chơi hay ứng dụng.

- Asset:

  Asset là các tài nguyên như hình ảnh, âm thanh, mô hình 3D, hoặc các tài nguyên khác được sử dụng trong Unity.

Các Asset có thể được nhập, quản lý và sử dụng để xây dựng và thiết kế các trò chơi hay ứng dụng.

Các khái niệm trên cung cấp cơ sở lý thuyết cơ bản về Unity. Hiểu về cơ sở lý thuyết này giúp người dùng sử dụng và tận dụng tối đa các tính năng và công cụ của Unity để phát triển trò chơi và ứng dụng đa nền tảng.

- MongoDB

MongoDB là một hệ thống quản lý cơ sở dữ liệu (DBMS) phổ biến và mã nguồn mở, thuộc hệ thống không quan hệ (NoSQL), cung cấp hiệu suất cao, khả năng mở rộng và linh hoạt trong việc xử lý lượng lớn dữ liệu phi cấu trúc. MongoDB thuộc loại cơ sở dữ liệu hướng tài liệu, nghĩa là nó lưu trữ dữ liệu dưới dạng tài liệu linh hoạt, bán cấu trúc được gọi là BSON (Binary JSON). MongoDB được phát triển bởi MongoDB Inc. và được ra mắt vào năm 2009.

Dưới đây là một số tính năng và khái niệm chính liên quan đến MongoDB:

- Hướng tài liệu: MongoDB lưu trữ dữ liệu dưới dạng các tài liệu JSON gọi là tài liệu BSON. BSON là một phiên bản nhị phân của tài liệu JSON hỗ trợ nhiều loại dữ liệu và cấu trúc khác nhau.
- Khả năng mở rộng: MongoDB cung cấp khả năng mở rộng theo chiều ngang thông qua sharding. Sharding cho phép phân phối dữ liệu trên nhiều máy chủ hoặc cụm, cho phép xử lý lượng dữ liệu lớn và tương thích với các tập dữ liệu khổng lồ.
- Linh hoạt: MongoDB có cấu trúc schema linh hoạt, cho phép thay đổi cấu trúc của các tài liệu trong một bộ sưu tập một cách linh hoạt. Mỗi tài liệu trong một bộ sưu tập có thể có cấu trúc khác nhau, phù hợp với các mô hình dữ liệu phát triển.
- Truy vấn: MongoDB hỗ trợ ngôn ngữ truy vấn mạnh mẽ với nhiều tính năng phong phú để truy xuất và thao tác dữ liệu. Nó cung cấp nhiều toán tử truy vấn, bao gồm lọc, sắp xếp, tổng hợp và tìm kiếm văn bản.

- Chỉ mục: MongoDB cho phép tạo chỉ mục trên các trường để cải thiện hiệu suất truy vấn. Nó hỗ trợ nhiều loại chỉ mục, bao gồm chỉ mục trên một trường, chỉ mục ghép, chỉ mục đa khóa, chỉ mục địa lý và chỉ mục văn bản.
- Sao chép: MongoDB cung cấp tính năng sao chép tích hợp để đảm bảo sẵn sàng cao và khả năng chịu lỗi. Nó hỗ trợ replica sets, là một nhóm các phiên bản MongoDB duy trì cùng một tập dữ liệu, cho phép chuyển giao tự động trong trường hợp máy chủ chính gặp sự cố.
- Giao dịch ACID: Bắt đầu từ phiên bản 4.0, MongoDB giới thiệu giao dịch ACID đa tài liệu, cho phép thực hiện nhiều hoạt động theo cách giao dịch, đảm bảo tính nhất quán dữ liệu.
- Khung thống kê: Aggregation Framework của MongoDB cung cấp một tập hợp các toán tử và giai đoạn mạnh mẽ để tổng hợp dữ liệu, bao gồm nhóm, lọc, chuyển đổi và kết hợp dữ liệu từ nhiều bộ sưu tập.
- MongoDB Atlas: MongoDB Atlas là một dịch vụ cơ sở dữ liệu quản lý đám mây được cung cấp hoàn toàn bởi MongoDB Inc. Nó cho phép triển khai, mở rộng và quản lý các cụm MongoDB trên đám mây, giúp giảm bớt công việc quản lý hạ tầng.

Cộng đồng và hệ sinh thái: MongoDB có một cộng đồng sôi nổi và tích cực, cung cấp tài liệu, hướng dẫn và nguồn tài nguyên phong phú. Nó cũng tích hợp với nhiều ngôn ngữ lập trình và khung phát triển thông qua các trình điều khiển chính thức và được cộng đồng hỗ trợ.

MongoDB được sử dụng rộng rãi trong các ứng dụng khác nhau, bao gồm phát triển web, hệ thống quản lý nội dung, phân tích thời gian thực, IoT (Internet of Things) và ứng dụng di động. Mô hình dữ liệu linh hoạt và khả năng mở rộng của nó làm cho nó phù hợp với việc xử lý nhiều loại dữ liệu và dự án quy mô lớn.

CHƯƠNG 2. TỔNG QUAN HỆ THỐNG

Hệ thống được chia làm 3 phần chính: Bản sao số, bộ xử lý trung tâm và mô hình robot thực. Việc xử lý tính toán được xử lý tại bộ xử lý trung tâm giúp bản sao số hoạt động tốt hơn và khả năng mở rộng dễ dàng hơn. Ngoài ra điều đó còn giúp việc sử dụng nhiều bản sao số cùng một lúc trên nhiều thiết bị khác nhau. Tổng quan hệ thống được mô tả tại hình 2.1 và lưu đồ giải thuật tại hình 2.2 và 2.3.

- Bản sao số

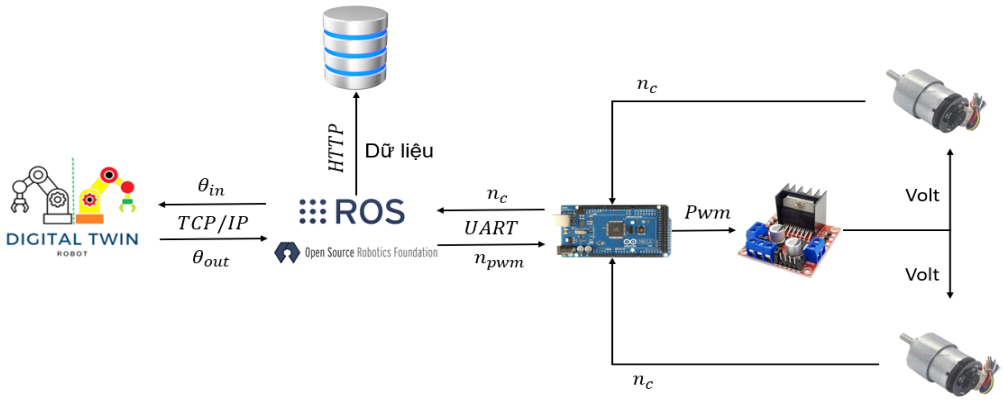
Bản sao số là một phần mềm mô phỏng trạng thái hình học của robot theo dữ liệu thời gian thực. Ngoài ra còn cung cấp cho người dùng các tính năng điều khiển và mô phỏng trước quá trình di chuyển của robot. Người dùng cũng có thể xử dụng chế độ 360 để dùng cho các thiết bị thực tế ảo.

- Bộ xử lý trung tâm

Đây là phần giúp chuẩn hóa dữ liệu, xử lý và tính toán. Ngoài ra còn hỗ trợ lưu trữ và sao lưu dữ liệu lên nền tảng cloud và server. Việc xử lý trên bộ xử lý trung tâm sẽ giúp mở rộng với nhiều loại robot một cách dễ dàng và tăng tốc độ xử lý của hệ thống.

- Mô hình robot vật lý

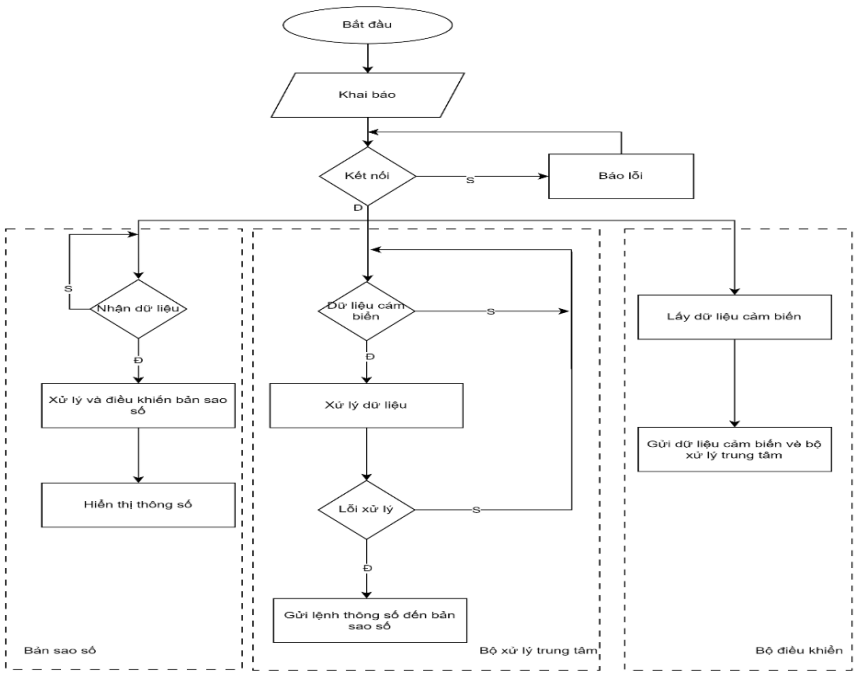
Mô hình robot vật lý bao gồm bộ điều khiển, cảm biến và các cơ cấu chấp hành. Đây là nơi thực hiện các yêu cầu từ người dùng.



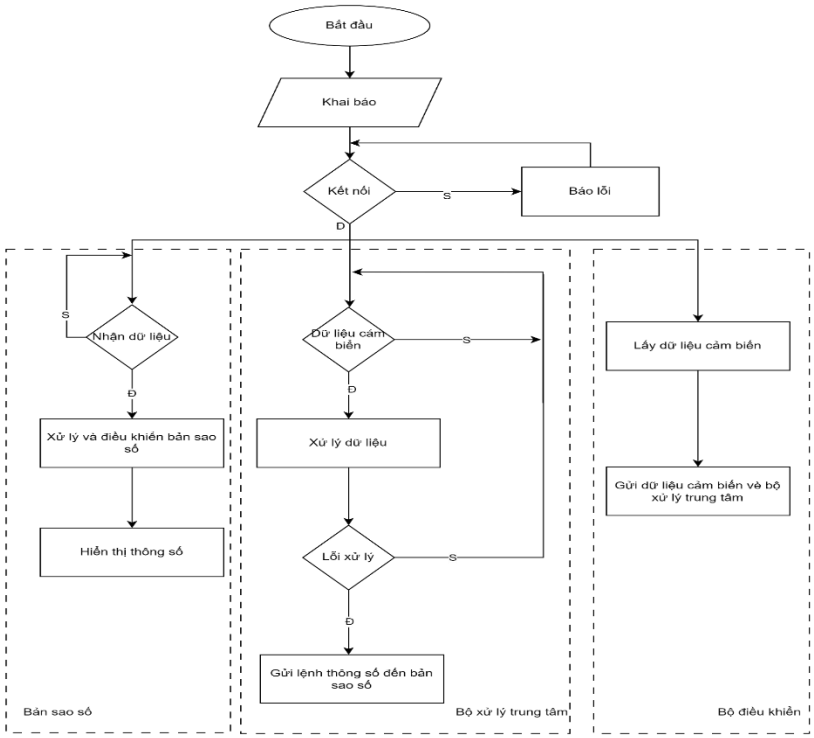
Hình 2.1. Tổng quan hệ thống

Trong đó, θ_{in} là giá trị góc quay robot ảo cần thực hiện; θ_{out} là giá trị góc quay robot thực ảo cần thực hiện; ROS là bộ xử lý trung tâm có cài đặt Ros; n_{pwm} là số xung

bộ điều khiển cần xuất; n_c là số xung encoder đếm được; Pwm là xung cấp cho mạch động lực. Volt (V) là điện áp cấp cho động cơ.



Hình 2.2. Lưu đồ điều khiển từ bản sao số sang robot vật lý



Hình 2.3. Lưu đồ giải thuật điều khiển từ robot vật lý sang bản sao số

CHƯƠNG 3. XÂY DỰNG ĐỐI TƯỢNG VẬT LÝ

3.1. Yêu cầu thiết kế

Mục tiêu đề tài tập trung vào nghiên cứu phát triển bản sao số cho robot trong đó tập trung vào truyền thông dữ liệu 2 chiều theo thời gian thực. Vì vậy mô hình được xây dựng để chứng minh và tăng tính thuyết phục của đề tài. Do đó thiết kế chỉ yêu cầu một robot arm từ 2 tự do trở lên, có khả năng thu thập dữ liệu vị trí góc khớp, kết nối và truyền thông dữ liệu 2 chiều theo thời gian thực.

3.2. Lựa chọn động cơ và mạch động lực

Mục tiêu của nghiên cứu là phát triển bản sao số cho robot arm. Vì vậy để tiết kiệm thời gian và chi phí nên robot chỉ yêu cầu ít nhất 2 bậc tự do với khả năng phản hồi vị trí quay của các khớp. Động cơ DC Servo JGB37-520 12VDC với hộp giảm tốc 168:1 được sử dụng với giá thành thấp và có thể dễ dàng mua được mà vẫn đáp ứng đủ yêu cầu cần thiết. Thông số về động cơ và encoder được trình bày tại bảng 3.1 và bảng 3.2.

Bảng 3.1. Thông số kỹ thuật động cơ

STT	Thông số	Giá trị
1	Điện áp định mức	12 VDC
2	Dòng tối đa	1.2 A
3	Tỷ số truyền	168:1
4	Tốc độ sau khi giảm tốc	35 rpm
5	Tốc độ tối đa khi có tải	28 rpm
6	Moment định mức	2.5 Kg.cm
7	Moment tối đa	10 Kg.cm
8	Đường kính trục	6 mm

Bảng 3.2. Thông số Encoder

STT	Thông số	Giá trị
1	Điện áp định mức	5 VDC
2	Số xung	11
3	Số kênh	2 kênh AB

3.3. Tính toán thiết kế cơ khí

- Thiết kế robot arm

Phần mềm solidword được sử dụng để vẽ bản thiết kế. Vật liệu được sử dụng là mica trong với độ dày 5mm. Bản vẽ gồm 7 chi tiết được gia công bởi máy cắt laser (xem thêm tại phụ lục). Đế được sử dụng là hộp điện 20x20 cm để chứa bộ điều khiển và mạch động lực. Ngoài ra còn được đục lỗ để cấp nguồn và cấp tín hiệu điều khiển từ ngoài. Thông số mô hình robot arm được trình bày tại bảng 3.3.

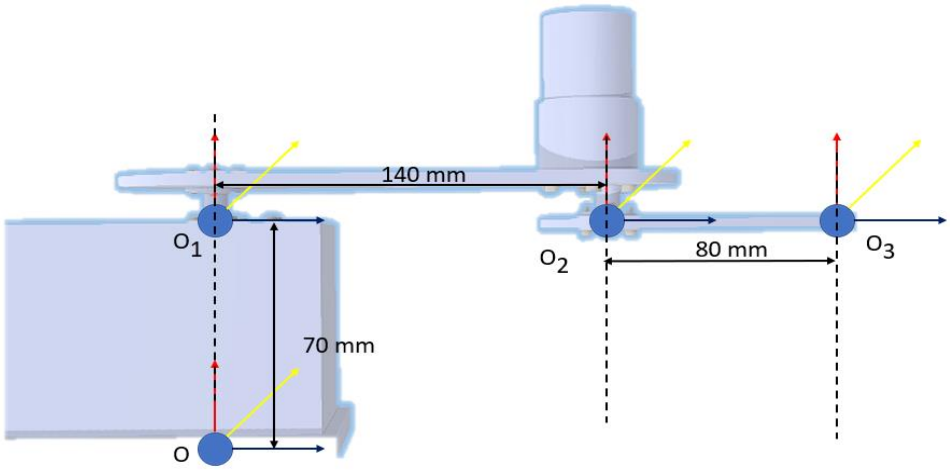
Bảng 3.3. Bảng thông số robot arm

STT	Thông số	Giá trị
1	Link 1	140 mm
2	Link 2	80 mm
3	Giới hạn khớp 1	[-90,90]
4	Giới hạn khớp 2	[-180,180]

- Tính toán động học thuận

Để tính toán động học thuận (forward kinematics) cho một robot arm 2 bậc tự do, cần biết thông tin về chiều dài các khớp cơ và góc quay của chúng. Điều này cho phép

chúng ta tính toán vị trí và hướng của tay robot dựa trên các góc quay của các khớp cơ. Tọa độ và khoảng cách được trình bày tại hình 3.1.



Hình 3.1. Hệ trục tọa độ và khích thước

Bảng 3.4. Bảng D-H của robot

Khớp	θ_i(deg)	d_i(mm)	a_i(mm)	α_i(deg)
1	θ_1	140	70	0
2	θ_2	80	0	0

Giả sử chúng ta có hai khớp cơ trong robot arm, và chúng ta gọi góc quay của khớp thứ nhất là θ1 và góc quay của khớp thứ hai là θ2. Chúng ta cũng cần biết chiều dài các đoạn cơ của robot arm, gọi là l1 và l2 (Thông số l1 và l2 được lấy từ bản thiết kế). Để tính toán động học thuận, chúng ta sử dụng các công thức sau:

$$x = l1 * cos(\theta 1) + l2 * cos(\theta 1 + \theta 2) = 160 * cos(\theta 1) + 80 * cos(\theta 1 + \theta 2) \quad (3.1)$$

$$y = l1 * sin(\theta 1) + l2 * sin(\theta 1 + \theta 2) = 160 * sin(\theta 1) + 80 * sin(\theta 1 + \theta 2) \quad (3.2)$$

Do robot có kết cấu 2 bậc tự do như hình 3.1 nên tọa độ z = 70 mm.

Trong đó, x và y là tọa độ của đầu tay robot. Với các giá trị của θ1 và θ2, chúng ta có thể tính được vị trí của đầu tay robot trong không gian.Ngoài ra, chúng ta cũng có thể tính toán hướng của tay robot bằng cách sử dụng công thức sau:

$$\theta = \theta 1 + \theta 2 \quad (3.3)$$

Trong đó, θ là góc quay của tay robot so với trục thẳng đứng. Điều này cho phép chúng ta xác định vị trí và hướng của tay robot dựa trên các góc quay của các khớp cơ và các thông số về chiều dài.

- Tính toán động học nghịch

Để tính toán động học nghịch cho robot arm 2 bậc tự do bằng phương pháp Newton-Raphson, cần sử dụng các bước sau:

Xác định giá trị ban đầu cho góc quay θ1 và θ2. Đây có thể là giá trị ban đầu gần với vị trí mục tiêu hoặc giá trị gần với giải pháp hiện tại.

Tính toán giá trị f1 và f2 dựa trên các phương trình sau:

$$f1(\theta1, \theta2) = l1 * cos(\theta1) + l2 * cos(\theta1 + \theta2) - x \quad (3.4)$$

$$f2(\theta1, \theta2) = l1 * sin(\theta1) + l2 * sin(\theta1 + \theta2) - y \quad (3.5)$$

Tính toán ma trận Jacobian J của hệ phương trình theo công thức sau:

$$J = \left[\left[\frac{\partial f1}{\partial \theta1}, \frac{\partial f1}{\partial \theta2}\right], \left[\frac{\partial f2}{\partial \theta1}, \frac{\partial f2}{\partial \theta2}\right]\right] \quad (3.6)$$

Trong đó, ∂f1/∂θ1, ∂f1/∂θ2, ∂f2/∂θ1, và ∂f2/∂θ2 là các đạo hàm riêng của f1 và f2 theo θ1 và θ2.

Tính toán ma trận delta θ (đại lượng thay đổi góc quay) bằng cách giải hệ phương trình tuyến tính J * delta θ = - [f1, f2].

Cập nhật giá trị góc quay θ1 và θ2 bằng cách thực hiện phép cộng:

$$\theta1 = \theta1 + delta\,\theta1, \theta2 = \theta2 + delta\,\theta2 \quad (3.7)$$

Lặp lại các bước từ 2 đến 5 cho đến khi giá trị f1 và f2 đạt đến mức chấp nhận được (gần với 0) hoặc đạt đến số lần lặp tối đa được định trước.

Kiểm tra xem giá trị góc quay θ1 và θ2 có nằm trong giới hạn cho phép không. Nếu không, áp dụng ràng buộc để đảm bảo nằm trong giới hạn cho phép.

Quá trình này được lặp lại để tìm giá trị góc quay tốt nhất để đạt được vị trí và hướng mong muốn của tay robot.

3.4. Xây dựng bộ điều khiển

- Mạch điều khiển và mạch động lực

Mạch điều khiển gồm có một vi điều khiển và một mạch động lực để điều khiển động. Vi điều khiển được sử dụng là vi điều khiển arduino mega (bảng 3.5) và mạch động lực là mạch L298N (bảng 3.6). Sơ đồ chân và quản mạch điều khiển được trình bày tại hình bảng 3.7 và hình 3.2.

Bảng 3.5. Thông số vi điều khiển ardino mega

STT	Thông số	Giá trị
1	Vi điều khiển	Atmega2560
2	Điện áp hoạt động	5 VDC
3	Nguồn vào (Khuyên dùng)	7-12 VDC
4	Nguồn vào (Giới hạn)	6-20 VDC
5	Số chân Digital I/O	54 (15 chân pwm)
6	Số chân vào Analog	16
7	Dòng DC mỗi chân I/O	20 mA
8	Dòng DC cho chân 3.3 V	50 mA
9	Bộ nhớ Flash	256 KB
10	SRAM	8 KB
11	EEPROM	4 KB
12	Clock Speed	16 MHz

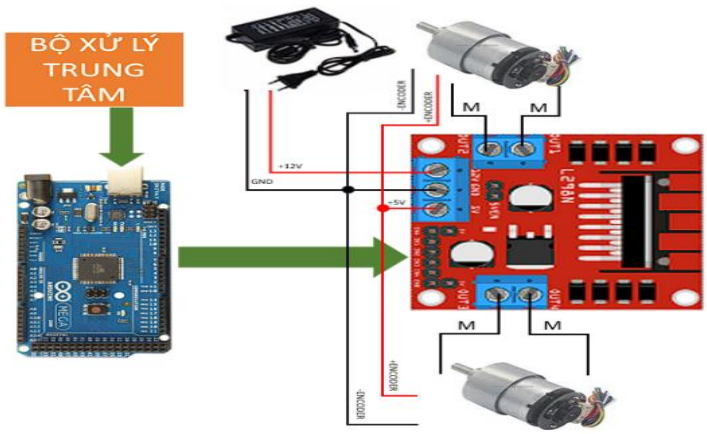
Bảng 3.6. Thông số mạch động lực L298N

STT	Thông số	Giá trị
1	Driver Chip	Double H-bridge L298N
2	Công suất tối đa	25 W

3	Nguồn động cơ tối đa	46 VDC
4	Nguồn dòng động cơ tối đa	2 A
5	Nguồn cấp	5-35 VDC
6	Dòng	2 A

Bảng 3.7. Bảng kết nối các thiết bị

STT	Arduino Mega	Mạch động lực	Động cơ & Encoder	Nguồn 12 VDC
1	Pin 2	-	Chân A Encoder 1	-
2	Pin 3	-	Chân A Encoder 2	-
3	Pin 4	-	Chân B Encoder 1	-
4	Pin 5	-	Chân B Encoder 2	-
5	Pin 6	IN1	-	-
6	Pin 7	IN2	-	-
7	Pin 8	IN3	-	-
8	Pin 9	IN4	-	-
9	Pin 10	PWM1	-	-
10	Pin 11	PWM2	-	-
11	GND	GND	GND Encoder 1 & 2	GND
12	-	12 VDC	-	VCC



13	-	5 VDC	VCC Encoder 1 & 2	-
14	-	M1	Chân + động cơ 1	-
15	-	M2	Chân - động cơ 1	-
16	-	M3	Chân + động cơ 2	-
17	-	M4	Chân - động cơ 2	-

Hình 3.2. Sơ đồ kết nối phần cứng mạch điều khiển

- Bộ điều khiển PID

Thuật toán PID được sử dụng để phát triển bộ điều khiển động cơ. Tuy nhiên, bộ điều khiển PID cần có các thông số PID để tối ưu và giúp hệ thống hoạt động ổn định. Để tìm thông số PID, một hàm tìm thông số PID tự động đã được phát triển và tích hợp vào hệ thống. Hàm này được phát triển dựa trên thuật toán Ziegler-Nichols. Việc tích hợp trực tiếp vào hệ thống giúp cho việc tối ưu và hiệu chỉnh thông số PID một cách nhanh chóng, linh hoạt, dễ dàng và tối ưu.

Do đông cơ có encoder là 11 xung và tỷ số truyền là 168:1 nên 1 vòng của khớp quay là 11*168 = 1848 xung. Vậy 1 độ tương ứng với 1848/360 ≈ 5 xung. Do giá trị xung từ 0-255 và điện áp đầu vào của động cơ là 12V nên:

$$pwm\ (xung\ điều\ khiển) = \frac{u}{12} * 255 \quad (3.8)$$

Thông tin về điều khiển dựa trên u(V) được trình bày tại bảng 3.8.

Bảng 3.8. Chiều quay của động cơ và xung (Pwm) cấp cho mạch động lực theo các trường hợp của điện áp (u)

u (V)	Thông số	Giá trị
u > 0	Chiều	1
	Pwm	u / 12 * 255
u < 0	Chiều	-1
	Pwm	-u / 12 * 255
u = 0	Chiều	0
	Pwm	0

CHƯƠNG 4. XÂY DỰNG BẢN SAO SỐ

4.1. Xây dựng mô hình 3d robot và môi trường

Để tạo mô hình 3D cho bản sao số có nhiều giải pháp tuy nhiên có 2 phương pháp phổ biến hiện nay là quét đối tượng hoặc vẽ đối tượng (bảng 4.1). Tùy vào điều kiện hiện tại có thể lựa chọn phương pháp khác nhau. Trong nghiên cứu này sử dụng phương pháp vẽ đối tượng vì có bản thiết kế của robot.

Bảng 4.1. So sánh giữa phương pháp quét đối tượng và vẽ đối tượng

	Quét đối tượng	Vẽ đối tượng
Ưu điểm	Tiết kiệm thời gian và chi phí đối với hệ thống lớn và cấu tạo phức tạp Độ chính xác cao trong việc tái tạo hình dạng và chi tiết vật thể Chỉ cần 1 máy quét 3D để thu thập dữ liệu	Yêu cầu máy tính có cấu hình cao trở lên
Nhược điểm	Yêu cầu máy tính cấu hình mạnh để xử lý Thời gian xử lý và tạo mô hình 3D từ phần mềm tốn nhiều thời gian	Cần nhiều thiết bị đo đạc Thời gian và chi phí cao đối với hệ thống lớn Yêu cầu khả năng hội tụ trong việc tạo hình và kỹ năng thiết kế của người dùng để tái tạo vật thể với độ chính xác cao

Trong báo cáo này sử dụng mô hình robot được xây dựng từ các chi tiết của bản thiết kế. Do phần mềm unity và solidword dựa trên nền tảng đồ họa khác nhau, vì vậy cần đưa mô hình về chuẩn đồ họa của unity. Có nhiều giải pháp cho vấn đề này và ở đây, phần mềm blender được sử dụng làm giải pháp trung gian giúp đưa mô hình về chuẩn đồ họa của unity. Mô hình 3D được trình bày tại hình 6.3.

Lưu ý rằng có rất nhiều chuẩn định dạng được hỗ trợ trong nền tảng game như obj, fbx, usd, urdf, … Việc sử dụng từng loại định dạng sẽ có những ưu nhược điểm và phương pháp xử lý riêng. Trong nghiên cứu này chúng tôi sử dụng định dạng URDF. Bảng 4.2 so sánh một số định dạng trong vấn đề này.

Bảng 4.2. So sánh các loại định dạng cho đồ họa cho đồ họa hoạt hình

Loại định dạng	Ưu điểm	Nhược điểm
FBX (Filmbox)	Hỗ trợ nhiều tính năng như animation, vật liệu, độ phân giải cao, ánh sáng, …	Kích thước tệp có thể lớn đối với mô hình phức tạp
	Có khả năng tương thích với nhiều công cụ mô hình hóa 3D	
OBJ (Wavefront Object)	Hỗ trợ định dạng mesh cơ bản, vật liệu và texture	Không hỗ trợ animation và các tính năng cao cấp khác
glTF (GL Transmission Format)	Định dạng nguồn mở, hỗ trợ nhiều tính năng như geometry, vật liệu, animation, ánh sáng, …	Có thể gặp một số vấn đề về tương thích và hỗ trợ trong nền tảng game
	Hỗ trợ kích thước tệp nhỏ và tương thích với các công nghệ web khác nhau	
Collada (Digital Asset Exchange)	Định dạng đa năng, hỗ trợ nhiều tính năng như geometry, vật liệu, animation, cấu trúc phân cấp, …	Có thể gặp một số vấn đề về tương thích và hỗ trợ trong nền tảng game

	Có khả năng tương thích với nhiều công cụ mô hình hóa 3D	Kích thước tệp có thể lớn đối với mô hình phức tạp
DAE (COLLADA)	Định dạng nguồn mở, hỗ trợ nhiều tính năng như geometry, vật liệu, animation, cấu trúc phân cấp,…	Một số vấn đề có thể xảy ra trong quá trình đưa vào nền tảng game
	Có thể tương thích với nhiều công cụ mô hình hóa 3D	
3DS (3D Studio)	Có thể xuất bởi các phần mềm 3D khác nhau và hỗ trợ nhiều tính năng	Có thể mất một số thông tin và tính năng phức tạp khi đưa vào nền tảng game
BLEND (Blender)	cho phép lưu trữ tất cả các thông tin của mô hình như geometry, vật liệu, animation, ánh sáng, …	Không phải là định dạng chuẩn được hỗ trợ trực tiếp trong nền tảng game
STL (Stereolithography)	Hỗ trợ geometry ba chiều đơn giản và dễ sử dụng	Không hỗ trợ vật liệu, texture, animation hay tính năng phức tạp khác
UDS (Universal Scene Description)	cho phép lưu trữ mô hình 3D phức tạp với nhiều tính năng như animation, hiệu ứng, ánh sáng và vật liệu	sự hỗ trợ và tương thích có thể khác nhau giữa các phần mềm và nền tảng khác nhau
	UDS được tối ưu hóa để đạt hiệu suất tốt trong	Mô hình UDS có thể phức tạp và đòi hỏi hiểu

	việc xử lý và truyền tải cảnh 3D phức tạp	biết và kỹ năng để làm việc với nó
URDF (Unified Robot Description Format)	cho phép mô tả chi tiết về cấu trúc và các thành phần của robot	URDF không hỗ trợ mô tả hình dạng hình học phức tạp
	URDF cho phép mô hình hóa nhiều robot khác nhau trong cùng một tệp	Không cung cấp hỗ trợ đặc biệt cho mô tả các động cơ hoặc hình thức điều khiển chuyển động của robot

4.2. Xây dựng môi trường ảo

Để xây dựng một môi trường ảo cho bản sao số có thể sử dụng các mô hình 3D về animation có sẵn hoặc sử dụng phương pháp tương tự như xây dựng mô hình 3D ở trên. Tuy nhiên với các không gian tòa nhà hay phòng thí nghiệm có thể sử dụng phương pháp vẽ trên các nền tảng như Revit, Skepchup, … sau đó sử dụng các biện pháp trung gian để đưa vào môi trường ảo. Trong nghiêm cứu này sử dụng một bản thiết kế mẫu của Revit để làm không gian hoạt động cho đối tượng 3D.

Một giao diện người dùng cũng được xây dựng giúp người dùng giám sát, tương tác và điều khiển hệ thống với nhiều chức năng điều khiển khác nhau. Giao diện điều khiển gồm 2 chế độ xem chính:

- Chế độ giám sát và điều khiển

Gồm 2 phần là phần giao diện điều khiển và phần giám sát (hình 6.4). Các thông số về trạng thái robot được thể hiện lên màn hình. Ngoài ra trạng thái của mô hình 3D cũng thay đổi để người dùng dễ hình dung trạng thái của robot. Giao diện điều khiển giúp cho việc tương tác và điều khiển với robot từ xa thuận tiện và dễ dàng.

- Chế độ 360

Tương tác và điều khiển góc quay của camera (hình 6.5). Điều đó giúp tăng độ trải nghiệm và tính đắm chìm của hệ thống. Ngoài ra chế độ này áp dụng cho thực tế ảo.

Ngoài ra hệ thống còn thêm chế độ mô phỏng giúp người dùng xem trước được quá trình robot sẽ làm việc. Tuy nhiên do dữ liệu và tiết kiệm chi phí nên chức năng này so với thực tế còn cách xa nhau. Cần có thời gian để huấn luyện cho mô phỏng tiệm cận với thực tế.

4.3. Xây dựng bộ điều khiển

Bộ điều khiển phần ảo có chức năng điều khiển mô hình 3D của robot. Trong bộ điều khiển có 2 chế độ gồm: (1) Điều khiển mô hình 3D robot bằng việc sử dụng slider; (2) Điều khiển mô hình 3D robot sử dụng mã lệnh điều khiển với các thông số tùy chỉnh. Tuy nhiên các thông số bị giới hạn trong phạm vị hoạt động của robot. Nếu vượt quá giới hạn thì sẽ robot sẽ không hoạt động và gửi cảnh báo đến người dùng. Mã lệnh được trình bày tại bảng 5.1.

Ngoài ra việc xử lý va chạm cũng được xây dựng dựa trên lớp phủ vật lý giúp lấy thông tin khi có sự cố va chạm giữa các đối tượng trong môi trường ảo từ đó gửi thông báo đến bộ xử lý trung tâm để xử lý.

CHƯƠNG 5. KẾT NỐI VÀ XỬ LÝ DỮ LIỆU

5.1. Xây dựng bộ xử lý trung tâm

Bộ xử lý trung tâm được xây dựng và phát triển trên một máy tính, vi xử lý có cài đặt ROS. Phần mềm xử lý được đóng gói thành giúp cho người dùng dễ dàng sử dụng với các câu lệnh đơn giản. Bộ xử lý trung tâm xử lý những tác vụ sau:

- Kết nối và truyền thông dữ liệu 2 chiều theo thời gian thực giữa mô hình vật lý và mô hình số.
- Xử lý lệnh điều khiển từ người dung.
- Tính toán động học robot arm.
- Tính toán và gửi tín hiệu điều khiển bộ điều khiển.
- Xử lý lỗi phát sinh của hệ thống.
- Lưu trữ dữ liệu cục bộ và sao lưu dữ liệu trên cloud.

Bảng 5.1. Mã lệnh của robot

STT	Mã lệnh	Mô tả
1	1	Điều khiển dựa trên góc quay
2	2	Điều khiển dựa trên điểm cuối tay máy

5.2. Thiết lập kết nối

Hệ thống có 2 kết nối chính: 1. Kết nối giữa bộ xử lý trung tâm và bộ điều khiển; Và 2. Kết nối giữa bộ xử lý trung tâm và bản sao số.

- Kết nối giữa bộ xử lý trung tâm và bộ điều khiển

Để kết nối bộ điều khiển với bộ xử lý trung tâm, có thể sử dụng các phương pháp kết nối sau đây:

- Kết nối qua cổng Serial (UART): Bộ điều khiển được trang bị một cổng Serial (UART) để giao tiếp với các thiết bị khác. Có thể sử dụng một cáp USB-to-Serial (như USB-to-Serial TTL hoặc USB-to-Serial FTDI) để kết nối cổng USB của bộ điều khiển với cổng Serial trên bộ xử lý trung tâm. Sau đó, có thể sử dụng các

thư viện phần mềm như PySerial trong Python, SerialPort trong C# hoặc rosserial trong ros để giao tiếp với bộ điều khiển thông qua cổng Serial.

- Kết nối qua USB: Một số bộ điều khiển hỗ trợ kết nối trực tiếp qua cổng USB. Có thể sử dụng các thư viện phần mềm như PySerial, hoặc SerialPort để thiết lập kết nối và giao tiếp với Arduino qua cổng USB.
- Kết nối qua giao thức nối tiếp khác: Ngoài kết nối Serial, bộ điều khiển cũng có thể được kết nối với bộ xử lý trung tâm thông qua các giao thức nối tiếp khác như I2C, SPI hoặc Ethernet. Điều này yêu cầu bộ điều khiển và bộ xử lý trung tâm cùng hỗ trợ các giao thức này. Cần sử dụng các thư viện phần mềm và giao thức tương ứng để giao tiếp với bộ điều khiển qua giao thức nối tiếp được chọn.

Trong nghiên cứu này chúng tôi sử dụng thư viện rosserial trong ros để kết nối với bộ điều khiển thông qua công Serial (UART).

- Kết nối giữa bộ xử lý trung tâm và bản sao số

Để kết nối giữa bộ xử lý trung tâm và bản sao số, có thể sử dụng giao thức giao tiếp ROS để truyền và nhận dữ liệu giữa hai môi trường này. Dưới đây là một phương pháp phổ biến để thực hiện kết nối này:

- Cài đặt ROS: Đầu tiên, cần cài đặt và cấu hình ROS trên bộ xử lý trung tâm. ROS cung cấp các thư viện và công cụ để phát triển và chạy các ứng dụng robot. Có thể tải xuống phiên bản ROS phù hợp với hệ điều hành của bạn từ trang web chính thức của ROS.
- Xác định các gói ROS cần thiết: Cần xác định các gói ROS muốn sử dụng trong ứng dụng bản sao số. Các gói này cung cấp các thông điệp, dịch vụ và hành động cần thiết để truyền dữ liệu giữa bộ xử lý trung tâm và bản sao sô. Có một số gói ROS phổ biến cho việc kết nối như "rosbridge_suite", "rosbridge_library", "ros-sharp" và “ROS-TCP”.
- Xây dựng các chương trình bộ xử lý trung tâm và bản sao số tương thích: Trong ROS, cần viết các nút ROS để gửi và nhận thông tin giữa bộ xử lý trung tâm và bản sao số. Cần định nghĩa các thông điệp ROS, dịch vụ và hành động cần thiết

cho ứng dụng bản sao số. Trong bản sao số, cần xây dựng các chương trình để giao tiếp với bộ xử lý trung tâm qua giao thức TCP/IP.

Phương pháp này cho phép tích hợp giữa bộ xử lý trung tâm và bản sao số trong cùng một mạng cục bộ, giúp truyền dữ liệu và nhận dữ liệu giữa bộ xử lý trung tâm và bản sao số.

5.3. Ước tính vị trí góc khớp của robot

Việc ước tính vị trí động cơ giúp giảm sai số giữa mô hình thực và ảo ảnh hưởng bởi độ trễ của hệ thống. Dữ liệu đầu vào là vị trí góc tại thời điểm trước đó và vị trí góc tại thời điểm hiện tại, một điểm trong tương lai được tính toán. Thuật toán được trình bày như sau:

Đầu vào	:	P_{cur}, P_{pre}, v_{pre}, t_{cur}, t_{pre}, t_{fur}
Đầu ra	:	P_{fur}, P_{pre}, v_{pre}
Xử lý	:	$v = \frac{P_{cur} - P_{pre}}{t_{cur} - t_{pre}}$ $a = \frac{v_{cur} - v_{pre}}{t_{cur} - t_{pre}}$ $P_{fur} = P_{cur} + v * (t_{fur} - t_{cur}) + \frac{a*(t_{fur} - t_{cur})^2}{2}$ $P_{pre} = P_{cur}$ $v_{pre} = v$

Với: P_{cur} là vị trí góc quay hiện tại của khớp. P_{pre} là vị trí góc quay trước đó của khớp. v_{pre} là vận tốc tại thời điểm trước đó, t_{cur} là thời gian tại lúc lấy P_{cur}. t_{pre} là thời gian trước đó, t_{fur} là thời gian mà góc quay của khớp ảo được cập nhập. P_{fur} là giá trị góc quay tại thời điểm t_{fur}. v là giá trị vận tốc tức thời tại thời điểm t_{cur}. a là gia tốc tức thời tại thời điểm t_{cur}.

5.4. Lưu trữ dữ liệu

Để lưu trữ dữ liệu cục bộ có thể sử dụng các phương pháp như lưu trữ trong tệp văn bản, lưu trữ trong cơ sở dữ liệu, hoặc sử dụng các cấu trúc dữ liệu trong ngôn ngữ lập trình đang sử dụng.

- Lưu trữ trong tệp văn bản:

- Lưu trữ dữ liệu trong các tệp văn bản như CSV, JSON hoặc các định dạng tương tự. Bạn có thể ghi dữ liệu vào tệp và đọc dữ liệu từ tệp bằng cách sử dụng các hàm đọc/ghi tệp trong ngôn ngữ lập trình của bạn.

- Lưu trữ trong cơ sở dữ liệu:
  - Sử dụng các hệ quản trị cơ sở dữ liệu như MySQL, SQLite, PostgreSQL, hoặc MongoDB để lưu trữ và truy vấn dữ liệu cục bộ. Các hệ quản trị cơ sở dữ liệu này cung cấp các API và truy vấn để tương tác với dữ liệu.

- Sử dụng các cấu trúc dữ liệu trong ngôn ngữ lập trình:
  - Một số ngôn ngữ lập trình cung cấp các cấu trúc dữ liệu như danh sách (list), bộ (tuple), từ điển (dictionary) hoặc các cấu trúc dữ liệu tương tự. Bạn có thể lưu trữ dữ liệu trong các cấu trúc này và thao tác với chúng theo nhu cầu.

Trong nghiên cứu này sử dụng phương pháp kết hợp phương pháp 1 và 2 để lưu trữ và sao lưu dữ liệu lên cloud. Dữ liệu được lưu vào các tệp csv, sau một khoảng thời gian nhất định thì dữ liệu được sao lưu trên nền tảng cloud. Nền tảng cloud được xây dựng dựa trên MongoDB.

5.5. Kiểm tra và thông báo

Khi làm việc với kết nối giữa các hệ thống như kết nối giữa bộ xử lý trung tâm và bản sao số, có một số lỗi thường gặp có thể xảy ra, bao gồm lỗi kết nối, tràn dữ liệu, sai kiểu dữ liệu và không truyền hoặc nhận được dữ liệu. Dưới đây là một số cách để kiểm tra và thông báo về các lỗi này:

- Kiểm tra kết nối:
  - Kiểm tra xem cả hệ thống đang chạy và được kết nối đúng cách.
  - Sử dụng các công cụ như lệnh "rostopic list" trong ROS để kiểm tra xem các topic ROS có đúng và sẵn sàng cho việc truyền dữ liệu hay không.

- Xử lý lỗi kết nối:
  - Sử dụng các hàm callback hoặc xử lý ngoại lệ để xử lý các lỗi kết nối.
  - Đảm bảo rằng các thiết bị mạng hoặc cổng kết nối được cấu hình đúng và hoạt động ổn định.

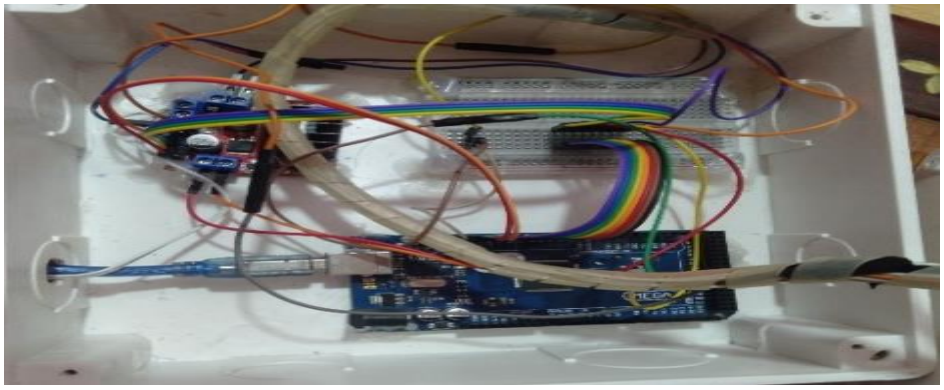
- Xử lý lỗi tràn dữ liệu:
  - Kiểm tra kích thước dữ liệu được truyền qua kết nối để đảm bảo rằng không có tràn dữ liệu xảy ra.
  - Sử dụng các công cụ như buffer hoặc cơ chế đồng bộ hóa để quản lý dữ liệu và tránh tràn dữ liệu.
- Xử lý lỗi chờ kết nối:
  - Sử dụng các hàm callback hoặc hệ thống đồng bộ hóa để kiểm tra và chờ đợi kết nối trước khi truyền dữ liệu.
  - Đặt thời gian chờ tối đa cho kết nối và xử lý các trường hợp lỗi khi không kết nối được trong khoảng thời gian đó.
- Thông báo lỗi:
  - Sử dụng hệ thống thông báo hoặc ghi log để thông báo và ghi lại các lỗi kết nối, tràn dữ liệu và chờ kết nối.
  - Hiển thị thông báo lỗi trực tiếp trên giao diện người dùng của bản sao số hoặc thông qua giao diện dòng lệnh của ROS.
  - Việc kiểm tra và thông báo về các lỗi này là quan trọng để quản lý và xử lý các vấn đề giao tiếp của hệ thống. Bằng cách sử dụng các cơ chế kiểm tra, xử lý lỗi và thông báo, có thể cải thiện tính ổn định và khả năng phản ứng của ứng dụng kết nối này. Hình 5.1 và 5.2 trình bày thông báo khi kết nối thành công và hiển thị thông báo lỗi khi không kết nối.
  - Các lỗi khi vận hành cũng được chuẩn hóa thành các Error Code (EC) tại bảng 5.2. Điều đó giúp việc xác định và giải quyết lỗi một cách nhanh chóng và dễ dàng.

Bảng 5.2. Mã lỗi của hệ thống

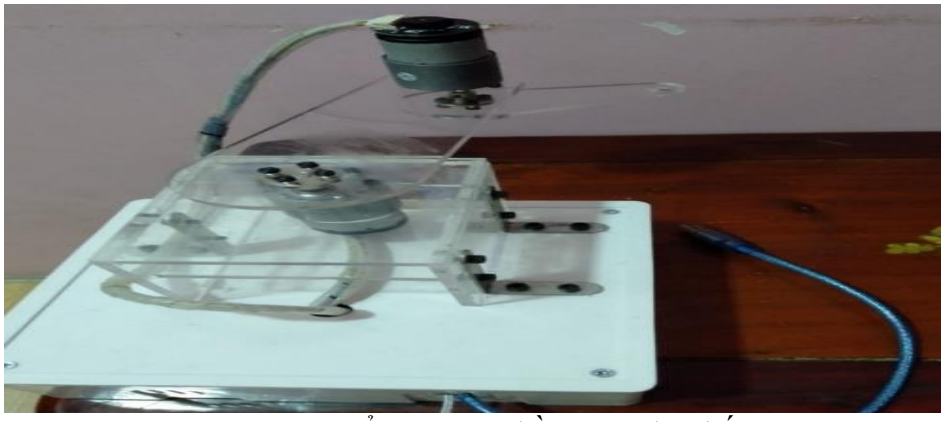
STT	EC	Loại lỗi
1	000	Không có kết nối với mạch điều khiển

2	001	Không có kết nối với bản sao số
3	002	Không tìm thấy giá trị quỹ đạo
4	003	Lỗi động cơ 1 không hoạt động
5	004	Lỗi động cơ 2 không hoạt động
6	005	Lỗi 2 động cơ không hoạt động
7	006	Lỗi truyền dữ liệu hệ thống

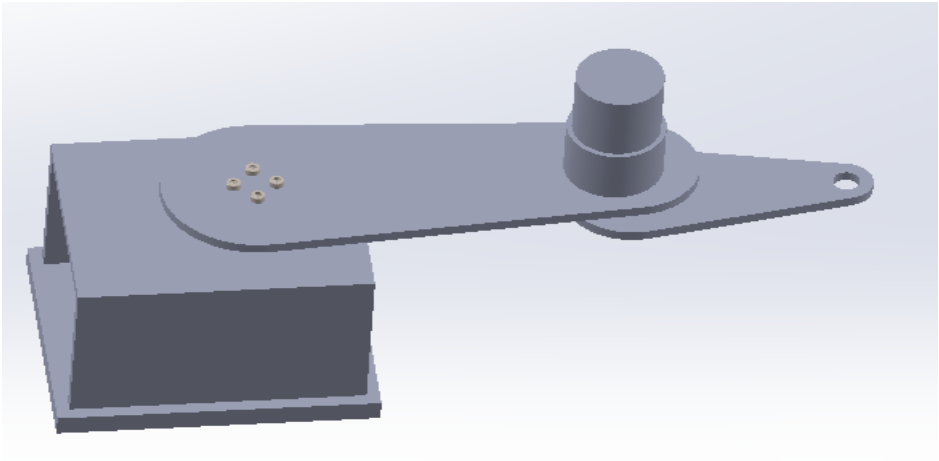
CHƯƠNG 6. KẾT QUẢ



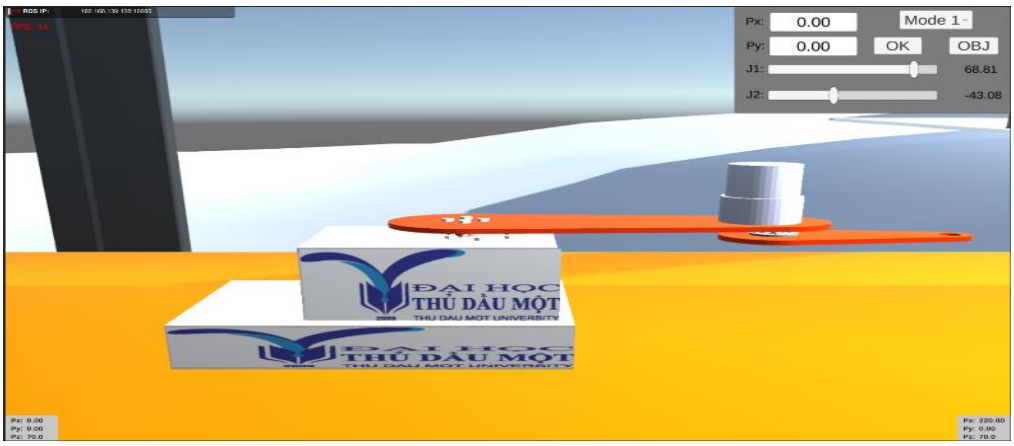
Hình 6.1. Kết nối mạch điều khiển và mạch động lực



Hình 6.2. Tổng quan phần cứng hệ thống

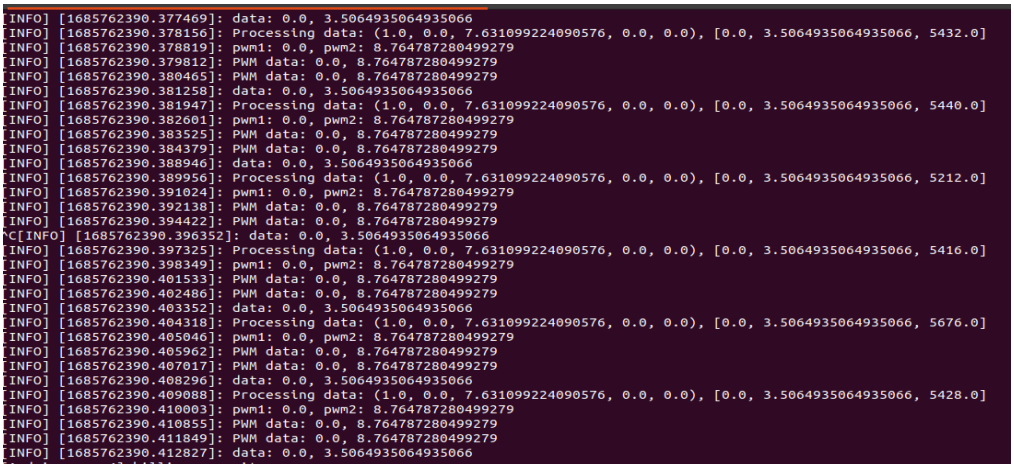


Hình 6.3. Mô hình 3D robot



Hình 6.4. Chế độ điều khiển và giám sát tại bảng điều khiển

Hình 6.5. Chế độ 360



Hình 6.6. Thông báo hệ thống trong quá trình hoạt động của bộ xử lý trung tâm

CHƯƠNG 7. ĐÁNH GIÁ VÀ THẢO LUẬN

7.1. Phương pháp đánh giá

- Bộ xử lý tính toán động học robot

Một công cụ giúp hỗ trợ kiểm tra và đánh giá việc tính toán động học robot được sử dụng giúp hỗ trợ kiểm tra và đánh giá việc tính toán của động học của hệ thống. Công cụ cho phép người dùng nhập đầu vào là vị trí góc của từng khớp và kết quả trả về là vị trí điểm cuối tay máy tương ứng sau đó vị trí điểm cuối này lại được đưa vào tính toán giúp kiểm tra xem góc tương ứng là bao nhiêu và so sánh với dữ liệu đầu phải xem có trùng khớp không.

- Độ trễ hệ thống

Dữ liệu được thu thập và phân tích về mặt thời gian. Dựa trên dữ liệu thời gian thu thập được sẽ vẽ thành các biểu đồ hỗ trợ việc phân tích. Thuật toán tính toán độ trễ hệ thống được trình bày như sau:

Đầu vào	:	$t_{pre},\ t_{cur}, t$=0
Đầu ra	:	$\Delta_{avd,}\Delta_t$
Vòng lặp	:	$\Delta_t = t_{cur} - t_{pre}$ $i = i + 1$ $t = t + \Delta_t$ $\Delta_{avd} = \frac{t}{i}$

Trong đó: t_{pre} là thời gian tại thời điểm trước đó; t_{cur} là thời gian tại thời điểm xét; Δ_t là độ trễ; Δ_{adv} là độ trễ trung bình; i là số lần dữ liệu được truyền và nhận; t là tổng độ trễ tương ứng với số lần i.

- Sai số hệ thống

Dữ liệu được thu thập và phân tích về mặt thời gian và giá trị tại thời điểm dữ liệu được nhận. Dựa trên dữ liệu đó sẽ được xử lý và chuẩn hóa sau đó vẽ thành biểu đồ để đánh giá sai số.

7.2. Đánh giá động học robot

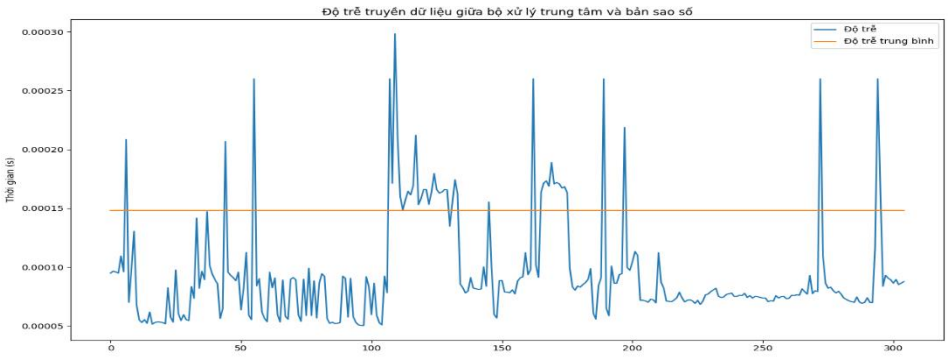
Bảng 7.1 cho thấy sự so sánh giữa việc tính toán giữa bộ xử lý trung tâm và công cụ hỗ trợ là gần giống nhau. Điều đó chứng tỏ việc tính toán động học robot của bộ xử lý trung tâm được chấp nhận.

Bảng 7.1. So sánh động học giữa bộ xử lý trung tâm và công cụ

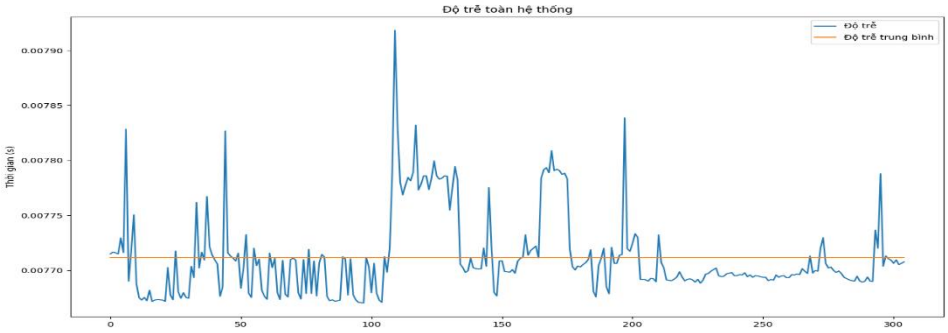
STT	Bộ xử lý trung tâm		Công cụ		Vị trí điểm cuối tay máy
	Bộ giá trị góc 1	Bộ giá trị góc 2	Bộ giá trị góc 1	Bộ giá trị góc 2	
1	[50, 50]	[85.5, -50]	[50, 50]	[85.5, -50]	[76.1, 186.03 ,70]
2	[49.5, 50]	[85, -50]	[49.49, 50]	[85.01, -50]	[77.73, 185.35, 70]
3	[9.99, 10]	[17.27, -10]	[10, 10]	[17.26, -9.99]	[213.05, 51.67, 70]
4	[-76, 89]	[-17, -89]	[-75.99, -89]	[-16.99, -89]	[111.82, -117.85, 70]
5	[-46.49, 88]	[12, -88]	[-47, 87.99]	[11.99, -88]	[156.29, -48.52, 70]
6	[58.28, 12]	[67, -12]	[58.28, 12.01]	[67, -12.01]	[100.59, 194.4, 70]

7.3. Đánh giá độ trễ hệ thống

Do độ trễ xử lý nên đã được điều chỉnh khoảng 50 Hz với độ trễ khoảng 20ms để tránh tràn dữ liệu. Độ trễ để xử lý khi nhận được tín hiệu là khoảng 5 ms. Độ trễ khi bộ xử lý gửi dữ liệu đến khi bản sao số nhận được dữ liệu là khoảng 0.15 ms. Thời gian bản sao số xử lý khi nhận được tín hiệu từ bộ xử lý trung tâm là 2.57 ms. Thời gian độ trễ để một dữ liệu từ thu thập đến lúc hoàn thành xử lý trên toàn bộ hệ thống khoảng 7.72 ms. Độ trễ hệ thống được thể hiện tại hình 7.1.



a)

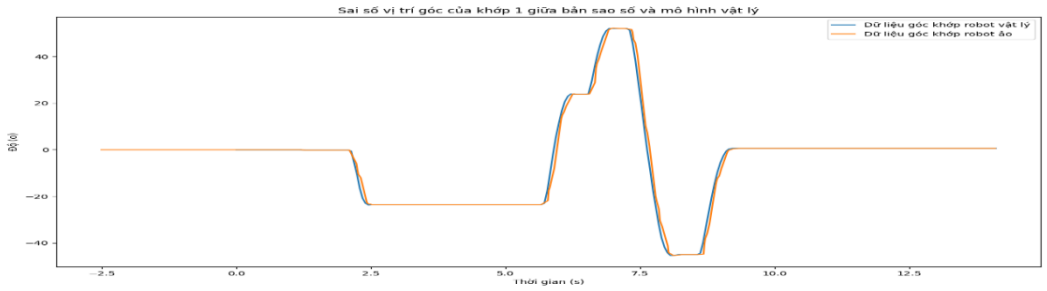


b)

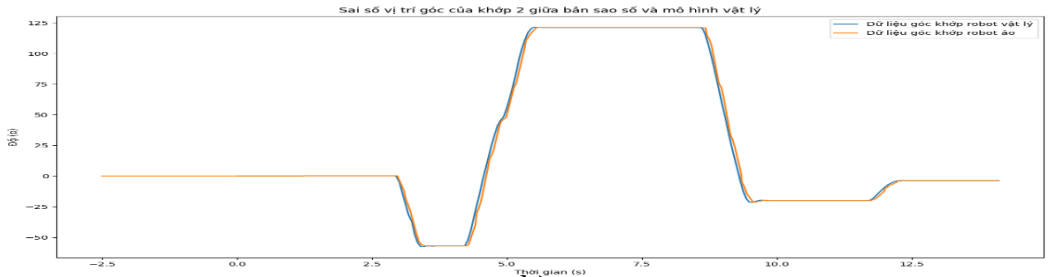
Hình 7.1. Độ trễ hệ thống. a) Độ trễ xử lý dữ liệu của bộ xử lý trung tâm trong khoảng 0.01-1.9 ms với độ trễ trung bình khoảng 0.15 ms; b) Đỗ trễ trên toàn hệ thống với độ trễ trung bình 7.72 ms.

7.4. Đánh giá sai số giữa mô hình vật lý và mô hình ảo

Sai số giữa mô hình thực vào ảo được thể hiện tại hình 7.2 và 7.3 khi lấy robot thực làm chuẩn. Có thể thấy rằng sai số giữa mô hình thực và ảo là khá nhỏ và gần như là trùng nhau khi sử dụng robot thực điều khiển robot ảo. Ngược lại thì lấy robot ảo làm chuẩn thì sai số giữa mô hình ảo và mô hình thực tăng lên thể hiện tại hình 7.4. Vấn đề này là do việc tính toán và điều khiển con ảo khác so với chạy thực tế ở con thực. Điều này có thể khắc phục bằng việc huấn luyện robot ảo từ dữ liệu robot thực.

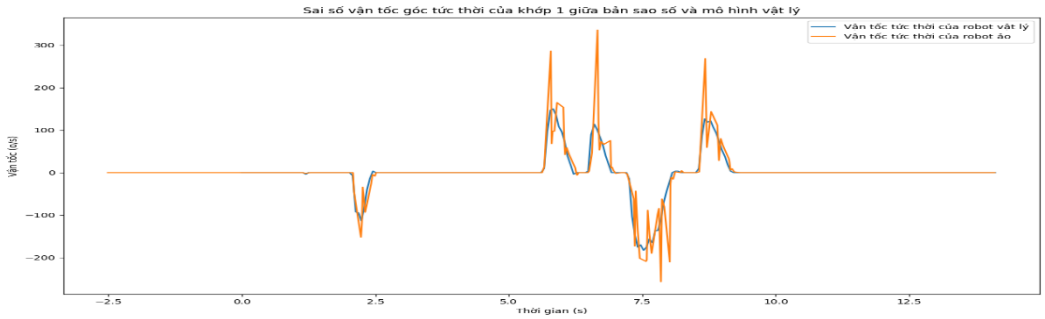


a)

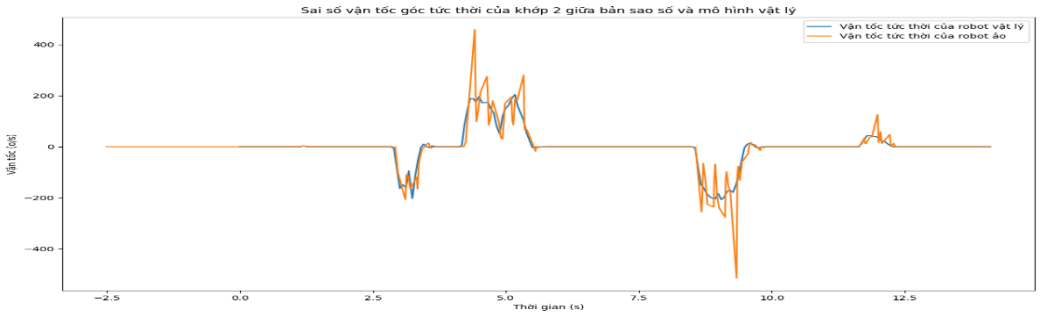


b)

Hình 7.2. Sai số vị trí góc của khớp giữa mô hình vật lý và mô hình bản sao số. a) Sai số vị trí góc tại khớp 1. b) Sai số vị trí góc tại khớp 2.

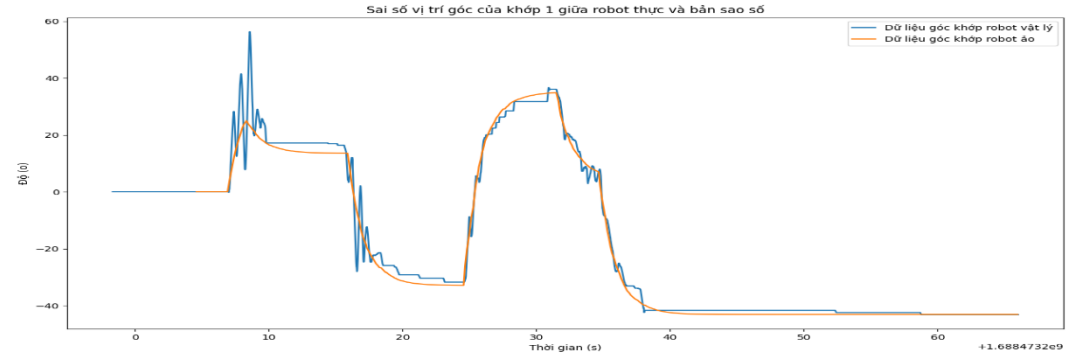


a)

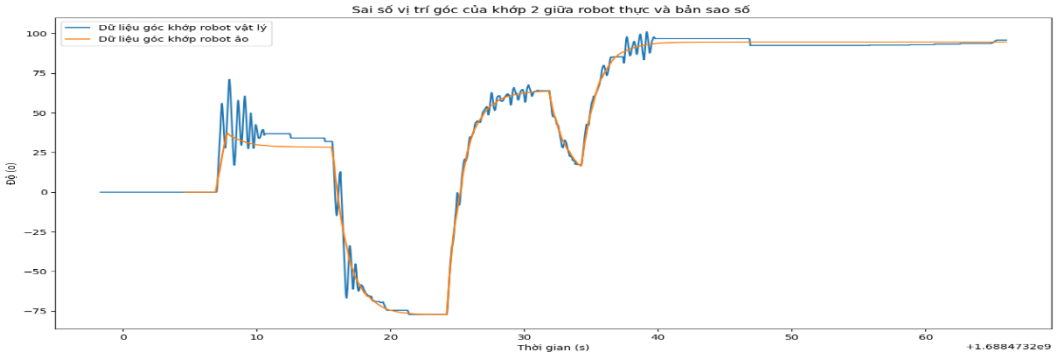


b)

Hình 7.3. Sai số vận tốc góc tức thời giữa tính toán mô phỏng và thực tế. a) Sai số vận góc tốc tức thời tại khớp 1. b) Sai số vận tốc góc tức thời tại khớp 2.



a)



b)

Hình 7.4. Sai số vị trí góc của khớp giữa mô hình vật lý và mô hình bản sao số. a) Sai số vị trí góc tại khớp 1. b) Sai số vị trí góc tại khớp 2.

CHƯƠNG 8. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

8.1. Kết luận

Nghiên cứu xây dựng được mô hình bản sao số cho robot với các mục tiêu đề ra với độ trễ khoảng 7.72 ms và sai số khá thấp. Tuy nhiên hệ thống chỉ mới thu thập dữ liệu từ một loại cảm biến, chưa đề cập đến các vấn đề dự báo, xử lý va chạm hay huấn luyện robot ảo. Ngoài ra việc sử dụng tính toán để tìm ví trí điểm cuối tay máy hay sự phụ thuộc vào encoder cũng là chưa đủ để đánh giá khách quan sai số giữa robot thực và ảo. Hệ thống cần được phát triển và cải tiến nhiều hơn nữa. Cần lưu ý khi chọn tần số lấy mẫu cảm biến cho bộ điều khiển trung tâm phải lớn hơn độ trễ của hệ thống để tránh tràn dữ liệu.

8.2. Hướng phát triển

Cải thiện các mô hình vật lý cũng như mô hình bản sao số để làm cho bản sao số giống mô hình thực hơn và cung cấp thông tin dự đoán bằng cách tích hợp các tính toán đa vật lý, phương pháp điều khiển hiện đại và áp dụng các kỹ thuật Máy học với nhiều dữ liệu thời gian thực hơn.

Ứng dụng VR/AR cho tương tác của hệ thống.

TÀI LIỆU THAM KHẢO

[1] Tao, Fei, et al. “Digital twin in industry: State-of-the-art.” *IEEE Transactions on Industrial Informatics*, 2018.

[2] G. Schroeder et al. “Visualising the digital twin using web services and augmented reality.” 1*4th International Conference on Industrial Informatics (INDIN)*, 2016.

[3] S. Yun, J. H. Park, et al. “Data-centric Middleware based Digital Twin Platform for Dependable Cyber-Physical Systems.” *Ninth International Conference on Ubiquitous and Future Networks*, 2017.

[4] Moreno, Aitor, et al. “Virtualisation process of a sheet metal punching machine within the Industry 4.0 vision.” *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 2017.

[5] Wang, Xi, et al. “Real-time process-level digital twin for collaborative human-robot construction work.” *Proceedings of the International Symposium on Automation and Robotics in Construction (IAARC)*, 2020.

[6] Garg, Gaurav, et al. “Digital twin for fanuc robots: Industrial robot programming and simulation using virtual reality.” *Sustainability*, 2021.

[7] B.P. Smarslok, et al. “Error Quantification and Confidence Assessment of Aerothermal Model Predictions for Hypersonic Aircraft.” *Structural Dynamics and Materials Conference*, 2013.

[8] T.M. Ricks, et al. “Computationally Efficient Solution of the High-Fidelity Generalized Method of Cells Micromechanics Relations.” *American Society of Composites-30th Technical Conference*, 2015.

[9] Y. Cai, B. Starly, et al. “Sensor Data and Information Fusion to Construct Digital-twins Virtual Machine Tools for Cyber physical Manufacturing.” *Procedia Manufacturing*, 2017.

[10] Dröder, Klaus, et al. “A machine learning-enhanced digital twin approach for human-robot-collaboration.” *Procedia Cirp*, 2018.

[11] B. Bielefeldt, et al. “Computationally Efficient Analysis of SMA Sensory Particles Embedded in Complex Aerostructures Using a Substructure Approach.” *ASME 2015 Conference on Smart Materials, Adaptive Structures and Intelligent Systems*, 2015.

[12] B.R. Seshadri, et al. “Structural Health Management of Damaged Aircraft Structures Using the Digital Twin Concept.” *AIAA/AHS Adaptive Structures Conference, Grapevine*, 2017.

[13] Pairet, Èric, et al. “A digital twin for human-robot interaction.” *14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2019.

[14] Voinov, Nikita, et al. “An approach to net-centric control automation of technological processes within industrial IoT systems.” *Advances in Manufacturing*, 2017.

[15] Martinez Hernandez, et al. “IService Business Model Innovation: The Digital Twin Technology.” *Apollo - University of Cambridge Repository*, 2019.

[16] Biesinger F, et al. “A case study for a digital twin of body-in-white production systems general concept for automated updating of planning projects in the digital factory.” *23rd international conference on emerging technologies and factory automation (ETFA)*, 2018.

[17] http://robot3t.com/en/

[18] Luu, Hoang, et al. “Optimization of Robotic Arm Trajectory Using Genetic Algorithm in Cosideration of Robot Controller.” *Measurement, Control, and Automation*, 2022.

[19] Ngon, Nguyen Chi, et al. “Điều khiển mờ thích nghi hệ cánh tay robot.” *TNU Journal of Science and Technology*, 2021.

[20] Thang, Nguyen Chien, et al. “Chỉnh định bộ điều khiển pid bằng hệ mờ áp dụng cho robot delta ba bậc tự do.” *TNU Journal of Science and Technology*, 2022.

[21] Dinh, Bao Minh, et al. “MAI_ARM: Robot tay máy thông minh sử dụng trí tuệ nhân tạo đa thể thức.” *Hội nghị Quốc gia lần thứ XXIV về Điện tử, Truyền thông và Công nghệ Thông tin*, 2021

[22] Loc, Pham Minh, et al. “Applying machine learning for hand posture recognition from a series of depth images.” *Science & Technology Development Journal-Economics-Law and Management*, 2022.

[23] Khoa, Phan Tran Dang, et al. “Robot nhổ cỏ tự động dựa trên phân tích ảnh sử dụng mô hình học sâu.” *Tạp chí Khoa học và Công nghệ-Đại học Đà Nẵng*, 2021.

[24] Hoa, Tran Dinh, et al. “Thiết kế, mô phỏng, chế tạo và điều khiển cánh tay robot 3 bậc tự do.” *Journal of Technical Education Science*, 2021.

[25] Quang, Nguyen Phung, et al. “Mô hình hóa và mô phỏng động lực học cho robot irb 120 dựa trên simscape multibody.” *TNU Journal of Science and Technology*, 2019.

[26] Ha, Quang Phuc, et al. “Lập trình điều khiển trên Arduino cho hệ vạn vật kết nối.” *Thanh Nien*, 2019.

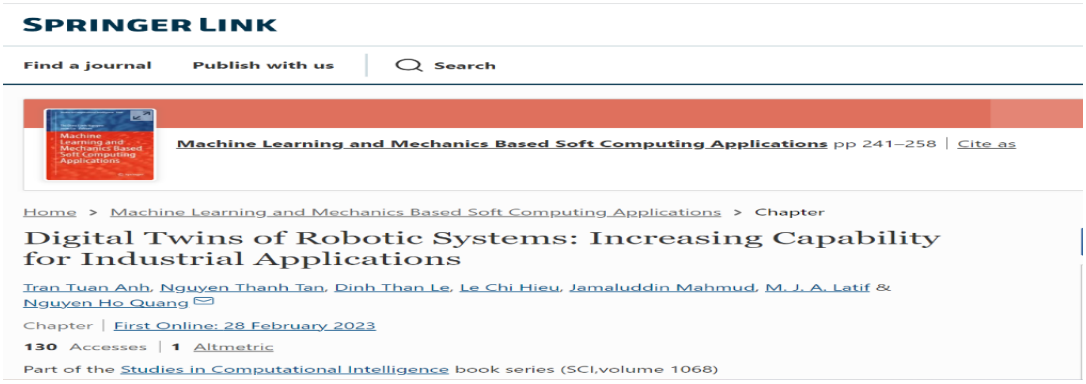
[27] Le, Đong Hung, et al. “Thiết kế và xây dựng xe tự hành thu thập nhiệt độ môi trường.” *Trường Cao đẳng Công nghệ thông tin hữu nghị Việt-Hàn*, 2019.

[28] Dinh, Bao Minh, et al. "Trợ lý ảo cho robot tiếp tân khách sạn dựa trên lý thuyết sRAM và nền tảng Dialogflow." *Hội nghị Quốc gia lần thứ XXIV về Điện tử, Truyền thông và Công nghệ Thông tin*, 2021.

PHỤ LỤC

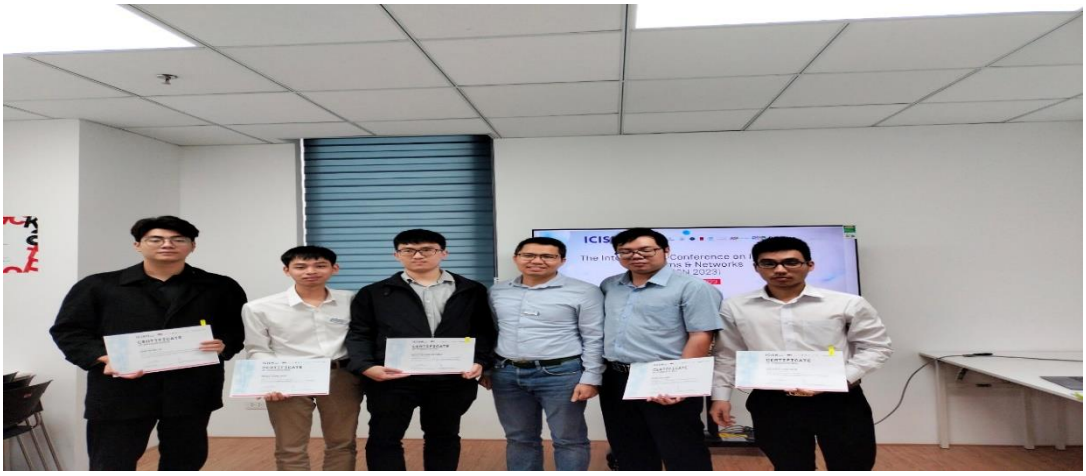
- **MINH CHỨNG BÀI BÁO:**

Anh, T.T. et al. (2023). Digital Twins of Robotic Systems: Increasing Capability for Industrial Applications. In: Nguyen, T.D.L., Lu, J. (eds) Machine Learning and Mechanics Based Soft Computing Applications. Studies in Computational Intelligence, vol 1068. Springer, Singapore. https://doi.org/10.1007/978-981-19-6450-3_23



- **TRÌNH BÀY HỘI THẢO KHOA HỌC QUỐC TẾ:**

Đã trình bày "A Conceptual Model Of Digital Twin For Potential Application In Healthcare:" tại hội thảo quốc tế ICISN 2022, dự kiến xuất bản vào tạp chí Intelligent Network System trong tháng 6.



- **LINK LƯU TRỮ DỮ ÁN:**

https://github.com/lamtacta2/NCKH_2021-2023.git

- **BẢN VẼ THIẾT KẾ CHI TIẾT ROBOT:**