

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CƠ KHÍ – BỘ MÔN CƠ ĐIỆN TỬ

...•••



LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

ĐIỀU KHIỂN HỒI TIẾP ẢNH CHO ROBOT GẮP VẬT

SỬ DỤNG IMAGE MOMENT

(Image moment based Visual Servoing for robot grasping object)

GVHD: TS. Lê Đức Hạnh

SVTH: Trần Tiến Trung

MSSV: 1713722

TP.HCM, 2021

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CƠ KHÍ – BỘ MÔN CƠ ĐIỆN TỬ

••••••



LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

ĐIỀU KHIỂN HỒI TIẾP ẢNH CHO ROBOT GẮP VẬT
SỬ DỤNG IMAGE MOMENTS

(Image moments based Visual Servoing for robot graping object)

GVHD: TS. Lê Đức Hạnh

SVTH: Trần Tiên Trung

MSSV: 1713722

TP.HCM, 2021

Số: ____ /ĐHBK-CK

Bộ môn: Cơ Điện Tử**NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP**

(Chú ý: sinh viên phải dán tờ này vào trang thứ nhất của bản thuyết minh)

HỌ VÀ TÊN: Trần Tiến Trung**MSSV:** 1713722**NGÀNH:** Cơ Điện Tử**LÓP:** CK17CD1

1. Đầu đề luận văn: Điều khiển hồi tiếp ảnh cho robot gấp vật sử dụng image moment
(image moment based Visual servoing for robot grasping object)

Số liệu:

- Vật gấp đứng yên
- Thời gian hội tụ nhanh <5s

2. Nhiệm vụ:

- Tìm hiểu tổng quan và nguyên lý hoạt động của tay máy
- Thiết kế đồ gá đầu hút.
- Thiết kế mạch điều khiển và chương trình điều khiển.
- Thực nghiệm kiểm nghiệm (nếu có điều kiện)
- Số bản vẽ dự kiến: 4 , gồm:
 - + 1 bản vẽ A0, về: Phân tích, lựa chọn phương án thiết kế.
 - + 1 bản vẽ A0, về: Kết cấu cơ khí.
 - + 1 bản vẽ A0, về: Bản vẽ mạch điện.
 - + 1 bản vẽ A0, về Lưu đồ và giải thuật điều khiển.

3. Ngày giao nhiệm vụ luận văn: 01/03/2021

4. Ngày hoàn thành nhiệm vụ: 11/06/2021

5. Họ và tên người hướng dẫn:

- Lê Đức Hạnh , BM Cơ điện tử

Phần hướng dẫn:

100%

Nội dung và yêu cầu LVTN đã được thông qua Bộ môn.

Ngày 01 tháng 03 năm 2021

CHỦ NHIỆM BỘ MÔN
(Ký và ghi rõ họ tên)

Nguyễn Quốc Chí

NGƯỜI HƯỚNG DẪN CHÍNH
(Ký và ghi rõ họ tên)

Lê Đức Hạnh

PHẦN DÀNH CHO KHOA, BỘ MÔN

Người duyệt (chấm sơ bộ):

Đơn vị:

Ngày bảo vệ:

Điểm tổng kết:

Nơi lưu trữ luận án:

LỜI CẢM ƠN

Trước khi trình bày nội dung Luận Văn Tốt Nghiệp của bản thân, tôi xin bày tỏ sự biết ơn sâu sắc đối với Thầy TS. Lê Đức Hạnh. Cảm ơn Thầy đã tận tình chỉ bảo, hướng dẫn, giúp đỡ tôi trong quá trình thực hiện Luận Văn. Chính những kiến thức, kinh nghiệm và sự nhiệt tình của thầy đã giúp ích cho tôi rất nhiều trong việc tháo gỡ những khúc mắc mà nhiều lúc tưởng chừng khó long giải quyết.

Tôi xin cảm ơn trung tâm Đào tạo Khu Công Nghệ Cao – SHTP Training Center đã cho tôi một môi trường học tập, làm việc năng động cùng với những máy móc thiết bị tiên tiến mà trước nay tôi không có cơ hội được sử dụng. Tôi xin chân thành cảm ơn anh Đỗ Tân Khoa nói riêng và phòng VJTC thuộc trung tâm đào tạo nói chung, những người đã giúp đỡ tôi rất nhiều ngay từ ngày đầu thực hiện Luận Văn Tốt Nghiệp.

Hơn nữa, là một sinh viên Bách Khoa, chuyên ngành Cơ Điện Tử, tôi xin gửi lời cảm ơn chân thành đến tất cả quý thầy cô giảng dạy tại bộ môn Cơ Điện Tử nói riêng và trường Đại học Bách Khoa nói chung vì những kiến thức mà thầy cô đã truyền đạt cho tôi.

Lời cuối cùng, con xin được cảm ơn gia đình đã là chỗ dựa vững chắc, là sự động viên to lớn về tinh thần để cùng con vượt qua những khó khăn thử thách trong thời gian học tập tại trường Đại học Bách Khoa. Con xin cảm ơn rất nhiều.

Tp HCM, ngày 20 tháng 07 năm 2021

Trần Tiến Trung

TÓM TẮT LUẬN VĂN

Đề tài Luận Văn Tốt Nghiệp “Điều khiển hồi tiếp ảnh cho robot gấp vật sử dụng image moment” trình bày về phương pháp điều khiển hồi tiếp ảnh sử dụng bộ đặc trưng thị giác là image moments trên cánh tay robot 6 bậc tự do Nachi MZ07 có cấu hình camera eye – in – hand, Vật thể được sử dụng để hồi tiếp là vật thể phẳng 2D được làm từ mica và có hình dạng phức tạp. Mô phỏng của đề tài được thực hiện bằng phần mềm Matlab, sử dụng hai bộ công cụ là Robotics Toolbox và Machine Vision Toolbox của tác giả P. Corke. Nội dung luận văn bao gồm 11 chương, cụ thể như sau:

Chương 1 Tổng quan

Chương 2 Lựa chọn phương án

Chương 3 Phân tích động học tay máy

Chương 4 Giải thuật Image moments based Visual Servoing

Chương 5 Xử lý ảnh

Chương 6 Camera calibration

Chương 7 Hệ thống điện và cơ cấu chấp hành

Chương 8 Giao tiếp và điều khiển robot

Chương 9 Mô phỏng

Chương 10 Thực nghiệm và kết quả

Chương 11 Kết luận

MỤC LỤC

LỜI CẢM ƠN	i
TÓM TẮT LUẬN VĂN	ii
DANH MỤC HÌNH ẢNH	vi
DANH MỤC BẢNG BIỂU	ix
CHƯƠNG 1. TỔNG QUAN	1
1.1. Giới thiệu chung	1
1.2. Tình hình nghiên cứu trong nước	4
1.3. Tình hình nghiên cứu ở nước ngoài	4
1.4. Sơ lược về robot công nghiệp	6
1.4.1. Định nghĩa robot công nghiệp	6
1.4.2. Phân loại robot công nghiệp	6
1.5. Mục tiêu, nhiệm vụ và phạm vi đề tài	11
1.5.1. Đầu đề luận văn	11
1.5.2. Mục tiêu luận văn	11
1.5.3. Nhiệm vụ luận văn	11
1.5.4. Phạm vi đề tài	12
1.6. Tổ chức luận văn.....	12
CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN	13
2.1. Lựa chọn loại robot.....	13
2.1.1. Phương án 1: Robot Delta	13
2.1.2. Phương án 2: SCARA robot	14
2.1.3. Phương án 3: Robot khớp bản lề	15
2.2. Lựa chọn cơ cấu gấp vật	16
2.2.1. Phương án 1: Tay gấp	16

2.2.2. Phương án 2: Giác hút	17
2.2.3. Phương án 3: Nam châm điện.....	17
2.3. Lựa chọn cấu hình camera.....	18
2.3.1. Phương án 1: Camera được gắn ở vị trí cố định	18
2.3.2. Camera được gắn trực tiếp trên tay máy	19
CHƯƠNG 3. ĐỘNG HỌC TAY MÁY.....	21
3.1. Thông số kỹ thuật robot Nachi MZ07	21
3.2. Phân tích động học thuận tay máy	24
3.3. Ma trận Jacobian.....	27
3.4. Góc roll, pitch, yaw	29
CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING	31
4.1. Lý thuyết chung về Visual Servoing	31
4.2. Visual servoing sử dụng moments ảnh.....	33
4.2.1. Giới thiệu về moments ảnh.....	33
4.2.2. Moments khối tâm và các bất biến	34
4.2.3. Thuật toán Visual Servoing sử dụng moments ảnh	36
4.2.4. Cải tiến giải thuật.....	43
CHƯƠNG 5. XỬ LÝ ẢNH	45
CHƯƠNG 6. CAMERA CALIBRATION	48
6.1. Lý thuyết.....	48
6.2. Tiến hành	50
CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH	54
7.1. Hệ thống điện	54
7.2. Tính toán giác hút	57
7.3. Tính toán đồ gá	60

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT	61
8.1. Giao tiếp với robot Nachi MZ07	61
8.2. Phương thức giao tiếp socket trên robot Nachi MZ07.	61
8.3. Điều khiển robot Nachi MZ07.....	63
CHƯƠNG 9. MÔ PHỎNG	70
9.1. Các bước mô phỏng	70
9.2 Kết quả mô phỏng.....	73
CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ.....	76
CHƯƠNG 11. KẾT LUẬN	82
11.1 Kết quả đạt được	82
11.2 Hạn chế của luận văn	82
11.3 Hướng phát triển đề tài	82
TÀI LIỆU THAM KHẢO.....	84

DANH MỤC HÌNH ẢNH

Hình 1.1 Các lĩnh vực liên quan đến Visual Servoing	2
Hình 1.2 Visual Servoing Platform	5
Hình 1.3 Cartesian/Gantry Robots	7
Hình 1.4 Spherical/Polar Robots	7
Hình 1.5 Cylindrical Robots.....	8
Hình 1.6 SCARA Robots	8
Hình 1.7 Articulated Robots (Robot khớp bản lề)	9
Hình 1.8 Parallel Robots (Robot song song)	9
Hình 2.1 Delta robot	13
Hình 2.2 Mitsubishi SCARA Robot.....	14
Hình 2.3 Robot khớp bản lề Nachi MZ07	15
Hình 2.4 Tay gấp thủy lực SMC MHZ2.....	16
Hình 2.5 Giác hút chân không FESTO.....	17
Hình 2.6 SMC Magnetic gripper	17
Hình 2.7 Eye to hand	18
Hình 2.8 Eye in hand	19
Hình 3.1 Vùng làm việc của robot Nachi MZ07	23
Hình 3.2 Cấu hình gốc tọa độ tay máy	24
Hình 3.3 Định nghĩa tham số bảng DH	25
Hình 4.1 Ví dụ về Image Moments	33
Hình 5.1 Vật thể sử dụng trong thực nghiệm	45
Hình 5.2 Không gian màu HSV	46
Hình 5.3 Kết quả sau khi lọc màu HSV	46
Hình 5.4 Hình ảnh thu được trước và sau khi lọc nhiễu.....	47

Hình 6.1 Pattern sử dụng để calib camera	50
Hình 6.2 Nhận diện ma trận điểm tròn của 1 trong số các hình mẫu thu được	52
Hình 7.1 Sơ đồ kết nối robot và CFD Controller	54
Hình 7.2 Cổng kết nối robot với các thiết bị ngoại vi	54
Hình 7.3 Sơ đồ kết nối Teach pendant với CFD Controller.....	55
Hình 7.4 Hai loại Teach pendant	55
Hình 7.5 Cổng nguồn của CFD Controller.....	56
Hình 7.6 Hệ thống khí nén của robot Nachi MZ07	57
Hình 7.7 Sơ đồ hoạt động của giác hút.....	57
Hình 7.8 Giác hút ESG của hãng FESTO	59
Hình 7.9 Biểu đồ tính toán đồ gá đầu tool.....	60
Hình 8.1 Mô hình giao tiếp điều khiển robot	65
Hình 8.2 Chương trình con khởi tạo socket và nhận vị trí của robot	66
Hình 8.3 Chương trình con ra lệnh di chuyển robot và điều khiển giác hút	67
Hình 8.4 Lưu đồ giải thuật của chương trình USERTASK	68
Hình 8.5 Lưu đồ giải thuật của chương trình robot.....	69
Hình 9.1 Hình ảnh dung để tạo dữ liệu vật thể.....	70
Hình 9.2 Hình thu được sau khi tìm đường bao của vật thể	71
Hình 9.3 Vị trí ban đầu của mô phỏng	73
Hình 9.4 Kết quả của mô phỏng	73
Hình 9.5 Giá trị của các đặc trưng s trong quá trình mô phỏng	74
Hình 9.6 Sai số của các đặc trưng trong quá trình mô phỏng.....	74
Hình 9.7 Góc của các khớp robot trong quá trình mô phỏng	75
Hình 9.8 Kết quả mô phỏng của một số mẫu vật thể khác	75
Hình 10.1 Vị trí ban đầu của robot và vật thể	76

Hình 10.2 Tiến hành lấy mẫu vật thể ở vị trí mong muốn	77
Hình 10.3 Đặt vật thể nghiêng một góc nhỏ so với mặt bàn	78
Hình 10.4 Vị trí của robot và vật thể trước khi tiến hành hồi tiếp	78
Hình 10.5 Vị trí của robot và vật thể sau khi tiến hành hồi tiếp.....	79
Hình 10.6 Tiến hành gấp vật thể	79
Hình 10.7 Tiến hành thả vật kết thúc thực nghiệm	80
Hình 10.8 Vị trí vật thể sau khi thả	80
Hình 10.9 Giá trị sai số của các đặc trưng trong quá trình thực nghiệm.....	81

DANH MỤC BẢNG BIỂU

Bảng 2.1 Lựa chọn phương án đề tài.....	20
Bảng 3.1 Thông số kỹ thuật của robot Nachi MZ07	21
Bảng 3.2 Thông số Denavit – Hartenberg của robot Nachi MZ07	24
Bảng 6.1 Tọa độ của P1, P2, P3	50
Bảng 6.2 Các vị trí lấy ảnh	51
Bảng 6.3 Vector xoay và vector tịnh tiến của từng ảnh mẫu	53
Bảng 9.1 Giá trị các đặc trưng mẫu s^*	74
Bảng 10.1 Giá trị các đặc trưng mẫu \tilde{s}	77

CHƯƠNG 1. TỔNG QUAN

CHƯƠNG 1. TỔNG QUAN

1.1. Giới thiệu chung

Ngày nay, với sự phát triển không ngừng của khoa học và công nghệ, robot ngày càng đóng vai trò quan trọng trong mọi lĩnh vực của đời sống, xã hội, thay thế dần sức lao động của con người tạo ra các sản phẩm chất lượng và có độ chính xác cao. Do đó, các yêu cầu về tính linh hoạt, độ chính xác và sự mạnh mẽ của robot ngày càng cao và có thể hoạt động trong nhiều môi trường khác nhau. Để đáp ứng các yêu cầu như thế, các robot thường được tích hợp các cảm biến để đo đạc và cảm nhận môi trường xung quanh (đặc biệt khi robot hoạt động trong môi trường không cấu trúc). Các loại cảm biến thường được sử dụng là cảm biến lực, cảm biến siêu âm, laser, cảm biến thị giác... Trong đây, cảm biến thị giác (ví dụ như camera) là phổ biến nhất, bởi vì cách cảm biến thị giác cảm nhận môi trường gần giống với con người và cho phép sự đo đạc không cần đến tiếp xúc trực tiếp.

Sử dụng điều khiển dựa trên thị giác giúp robot có thể cảm nhận và phản ứng với môi trường phức tạp, không biết trước. Camera bắt đầu được sử dụng trong điều khiển robot từ những năm sáu mươi của thế kỷ trước. Thông tin thị giác được sử dụng trong bộ điều khiển vòng hở, robot “nhìn” và sau đấy “di chuyển”, do đó độ chính xác phụ thuộc trực tiếp vào độ chính xác của cảm biến hình ảnh và của robot. Điều khiển hồi tiếp sử dụng thông tin thị giác (điều khiển vòng kín) chỉ bắt đầu phát triển vào những năm 90 giúp tăng độ chính xác của hệ thống. Hệ thống điều khiển vòng kín này được gọi là Visual Servoing.

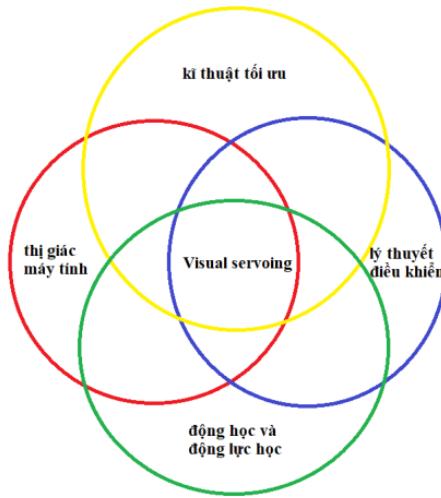
Visual servoing là một kỹ thuật để điều khiển robot, trong đây thông tin hồi tiếp thị giác được sử dụng trong vòng điều khiển kín giúp cải thiện độ chính xác. Đây là một kỹ thuật với sự kết hợp của nhiều lĩnh vực khác nhau như thị giác máy tính, động học và động lực học, lý thuyết điều khiển và các kỹ thuật tối ưu hóa (Hình 1.1). Sự thiếu chính xác của cơ cấu cơ khí có thể được bù đắp khi sử dụng kỹ thuật visual servoing. Dữ liệu thị giác được hồi tiếp trong sơ đồ điều khiển giúp hệ thống đạt được độ chính xác yêu cầu cho dù hệ thống cơ khí chưa thật sự hoàn chỉnh.

Một tác vụ robot trong visual servoing sử dụng các đặc trưng hình ảnh trích xuất từ đối tượng để điều khiển chuyển động của robot/camera. Có nhiều nghiên cứu khác

CHƯƠNG 1. TỔNG QUAN

nhau tập trung vào việc lựa chọn các đặc trưng hình ảnh, cấu trúc của sơ đồ điều khiển, khảo sát sự mạnh mẽ đối với các sai số hay sự không chắc chắn khi hiệu chỉnh robot và camera. Một hệ thống visual servoing có thể được thiết kế sử dụng các đặc trưng 2D như tọa độ các điểm trong hình ảnh, các đặc trưng 3D như tọa độ Cartesian hoặc kết hợp cả hai. Dựa trên sự lựa chọn các đặc trưng và sơ đồ điều khiển, hệ thống robot sẽ có các hành vi khác nhau. Do đó sẽ gây ra các khó khăn để có thể thiết kế và lựa chọn được các thông tin thị giác và sơ đồ điều khiển cho ra đáp ứng phù hợp.

Sử dụng thị giác trong điều khiển robot giúp giải quyết được nhiều vấn đề khác nhau nhờ vào thông tin thị giác được thu thập mà không có sự tiếp xúc với môi trường. Tuy nhiên, một số vấn đề cần được xem xét khi sử dụng cảm biến thị giác trong điều khiển robot. Những vấn đề này bao gồm sự ổn định cục bộ hoặc toàn cầu, quỹ đạo của robot và của các tính năng hình ảnh phải phù hợp, tách rời tối đa giữa thông tin thị giác và bậc tự do được điều khiển, tránh sự kỳ dị và cực tiểu cục bộ của ma trận điều khiển (ma trận tương tác), đảm bảo các tính năng nằm trong trường nhìn của camera, tránh va chạm và tránh giới hạn các góc khớp của robot.



Hình 1.1 Các lĩnh vực liên quan đến Visual Servoing

Visual servoing (VS) là một kỹ thuật điều khiển hồi tiếp sử dụng thông tin thị giác giúp tăng cường độ chính xác và tính linh động của hệ thống robot. Các tác vụ điều khiển trong visual servoing được định nghĩa để điều khiển vị trí của robot trong khung tọa độ toàn cục hoặc trong khung tọa độ của đối tượng mục tiêu, sử dụng các đặc trưng thị giác được trích xuất từ hình ảnh theo thời gian thực. Hàm sai số trong visual servoing

CHƯƠNG 1. TỔNG QUAN

được định nghĩa là sai số giữa các đặc trưng thị giác hiện tại và đặc trưng thị giác tại vị trí mong muốn. Mục tiêu của sơ đồ điều khiển dựa trên thị giác là điều chỉnh sai số và triệt tiêu nó (sai số bằng không). Một hay nhiều camera có thể được sử dụng để thu thập thông tin thị giác từ đối tượng để điều khiển robot.

Dựa trên cách thông tin thị giác được sử dụng, Visual Servoing được phân thành hai sơ đồ điều khiển chính là position based visual servoing (PBVS) và image based visual servoing (IBVS)[11, 12, 13, 22]. Sự kết hợp của hai sơ đồ điều khiển tạo thành một sơ đồ lai gọi là 2-1/2D visual servoing[19, 26]. Trong một hệ thống PBVS, đầu vào được tính toán trong không gian 3D (vì lý do này, phương pháp này có thể được gọi là 3D visual servoing). Vị trí của đối tượng so với camera được ước lượng từ các đặc trưng hình ảnh của đối tượng. Có nhiều phương pháp khác nhau để xác định vị trí của đối tượng. Tất cả các phương pháp đều yêu cầu biết mô hình hình học hoàn chỉnh của đối tượng và các thông số calib của camera. Với hệ thống điều khiển IBVS, đầu vào được tính toán trong không gian ảnh 2D (vì lý do này, phương pháp này còn gọi là 2D visual servoing). Phương pháp này được biết đến rất mạnh mẽ đối với sai số calib camera và robot. Tuy nhiên sự hội tụ của nó về mặt lý thuyết chỉ được đảm bảo trong một vùng (hoàn toàn không thể phân tích để xác định) xung quanh vị trí mong muốn. Ngoại trừ các trường hợp đơn giản, việc phân tích độ ổn định liên quan đến lỗi hiệu chỉnh là một điều bất khả thi, vì hệ thống được ghép nối và phi tuyến.

Hai khía cạnh chính ảnh hưởng lớn đến hành vi của hệ thống visual servoing là: việc lựa chọn các đặc trưng thị giác được sử dụng làm đầu vào của bộ điều khiển và sơ đồ điều khiển được thiết kế. Cùng một bộ các đặc trưng, hệ thống sẽ có các hành vi khác nhau khi được sử dụng trong các sơ đồ điều khiển khác nhau và cùng một luật điều khiển sẽ cho các hành vi khác nhau khi xem xét các đặc trưng khác nhau. Hành vi thu được với sự kết hợp của các lựa chọn này thường không như mong muốn: chọn một bộ đặc trưng cụ thể hoặc sơ đồ điều khiển cụ thể có thể dẫn đến một số vấn đề về tính ổn định và hội tụ.

Trong phạm vi luận văn này, sẽ ứng dụng IBVS với bộ đặc trưng thị giác được chọn là Image moments hay còn được gọi là Image moments based visual servoing để điều khiển robot nhận diện và gấp vật 2D có hình dạng phức tạp.

CHƯƠNG 1. TỔNG QUAN

1.2. Tình hình nghiên cứu trong nước

Hệ robot-camera gắn trên bệ di động cũng như visual servoing đã được một số viện nghiên cứu, một số trường đại học và đặc biệt là các trường, viện nghiên cứu trong lĩnh vực kĩ thuật quân sự thuộc bộ quốc phòng quan tâm, nghiên cứu:

Các viện nghiên cứu, chế tạo thuộc bộ quốc phòng đã nghiên cứu, chế tạo bệ robot-camera để triển khai ứng dụng và triển khai trên thực địa trong lĩnh vực quân sự. Robot-camera cũng được quan tâm, nghiên cứu trong các trường đại học như: đại học Bách Khoa Hà Nội, đại học Công Nghệ - đại học Quốc Gia Hà Nội, đại học Bách Khoa – Đại học quốc gia Thành Phố Hồ Chí Minh, đại học Bách Khoa Đà Nẵng. Các trường đại học trên có một số luận văn thạc sĩ và luận án tiến sĩ nghiên cứu về lĩnh vực này.

Trong các viện nghiên cứu thì lĩnh vực này cũng có một số bài báo, báo cáo khoa học tại các hội nghị trong nước và quốc tế, đặc biệt là các kết quả nghiên cứu khoa học tại Viện Công Nghệ Thông Tin – Viện Hàn Lâm Khoa Học Công Nghệ Việt Nam và đại học Bách Khoa Hà nội được công bố. Trong luận văn thạc sĩ của tác giả Võ Duy Công [24] thuộc Đại học Bách Khoa – đại học quốc gia Thành phố Hồ Chí Minh tác giả đã đi sâu vào nghiên cứu, xem xét các kỹ thuật phổ biến trong Visual Servoing và thực hiện các đánh giá dựa trên hiệu quả và khả năng giải quyết các vấn đề gặp phải trong visual servoing của các kỹ thuật.

1.3. Tình hình nghiên cứu ở nước ngoài

Ở nước ngoài kỹ thuật điều khiển Visual Servoing đã được đưa vào nghiên cứu rất sớm và ứng dụng trong điều khiển robot. Cụ thể những nhà khoa học nổi tiếng trong lĩnh vực như François Chaumette, Seth Hutchinson, Peter I. Corke,... Đã công bố rất nhiều bài báo hay các báo cáo nghiên cứu trong lĩnh vực này.

Các bài báo như: “A tutorial on visual servo control” [20], “Visual servo control, part I: Basic approaches” [11], “Visual servo control, part II: Advanced approaches” [12], đã nghiên cứu và đưa đến cho người đọc những kiến thức cơ bản về visual servoing cụ thể là 2 kỹ thuật chính của visual servoing là PBVS và IBVS. Cho đến những bài báo về các kỹ thuật kết hợp cả IBVS lẫn PBVS 2-1/2 D visual servoing [19]. Hay những bài báo về các kỹ thuật khác của Visual servoing như Homography-based Visual

CHƯƠNG 1. TỔNG QUAN

servoing “Homography-based 2D Visual Servoing” [21], Partitioned Visual Servoing(PVS) “A new partitioned approach to image-based visual servo control” [22], Shorted Path Visual servoing “New shortest-path approaches to visual servoing” [23]. Rất nhiều những bài báo nghiên cứu được công bố với nội dung là các đề suất về các kỹ thuật mới để cải thiện nhưng hạn chế của các kỹ thuật ra đời trước của Visual servoing.

Trong rất nhiều những bài báo về Visual Servoing có những bài báo liên quan đến đề tài luận văn này như trong “Image moments: a general and useful set of features for visual servoing” [14] của tác giả François Chaumette đã thảo luận về việc sử dụng image moments để thành lập biểu thức của ma trận visual Jacobian. Biểu thức này cho phép tách rời các bậc tự do dựa trên loại moment được chọn. Hay trong bài báo “Point-based and region-based image moments for visual servoing of planar objects” [17] của tác giả O. Tahri và François Chaumette đã trình bày việc sử dụng moment invariants để thiết kế một sơ đồ điều khiển 2D visual servoing tách rời.

Bên cạnh những bài báo những nghiên cứu về Visual servoing một số toolbox dành cho matlab phục vụ cho việc nghiên cứu và học tập về Visual servoing cũng được ra đời cụ thể là Robotics toolbox [9] và Machine vision toolbox [10] của Peter I. Corke.

Một thư viện có tên là ViSP(Visual Servoing Platform) là một thư viện đa nền tảng cho phép tạo mẫu và phát triển các ứng dụng của Visual servoing được nghiên cứu và thực hiện bởi IRISA – nhóm Inria Rainbow được ra mắt.



Hình 1.2 Visual Servoing Platform

CHƯƠNG 1. TỔNG QUAN

1.4. Sơ lược về robot công nghiệp

1.4.1. Định nghĩa robot công nghiệp

Theo viện Robotics của Mỹ RIA định nghĩa: Robot công nghiệp là một thiết bị đa chức năng, có khả năng tái lập trình, được thiết kế để di chuyển vật liệu, tay gấp hoặc các thiết bị chuyên biệt khác thông qua các chuyển động đã được lập trình khác nhau cho từng tác vụ đa dạng khác nhau.

Nhìn từ quan điểm của người kỹ sư: Robots là các thiết bị phức tạp, đa năng. Nó bao gồm kết cấu cơ khí, hệ thống cảm biến và hệ thống điều khiển tự động. Các nguyên lý cơ bản của robotics dựa trên kết quả nghiên cứu của các báo cáo trong lĩnh vực cơ khí, điện, tự động hóa, toán và khoa học máy tính. Để được xem là một robot, một thiết bị phải thỏa mãn các tiêu chí sau:

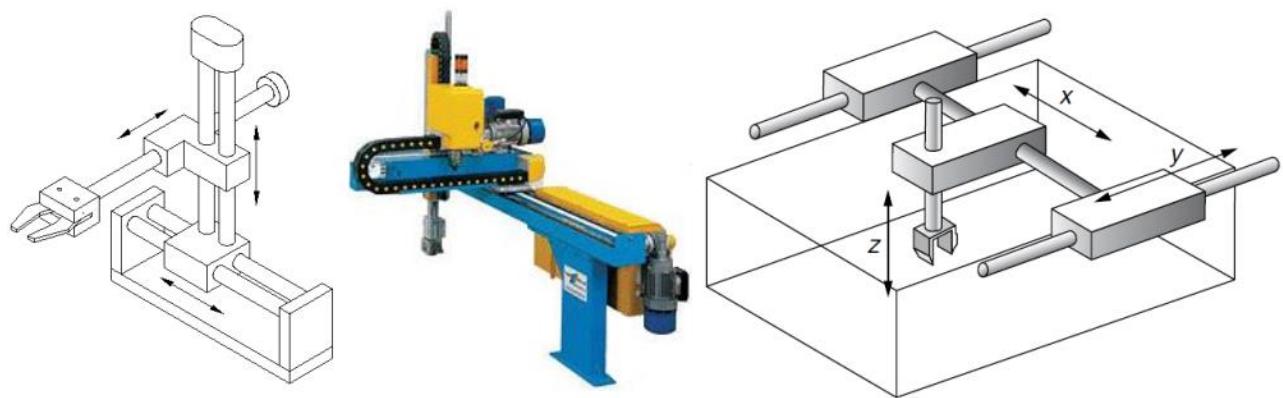
- Tái lập trình được: được thiết kế sao cho các chuyển động được lập trình hoặc các chức năng phụ có thể thay đổi, mà không cần phải thay thế phần cứng.
- Tính đa dụng: khả năng làm việc, thích nghi với nhiều ứng dụng khác nhau khi thay thế một vài thiết bị phần cứng ví dụ như tool.
- Cảm biến: có khả năng lấy thông tin từ môi trường xung quanh.
- Tương tác được với con người.

1.4.2. Phân loại robot công nghiệp

1.4.2.1. Phân loại theo cấu trúc cơ khí:

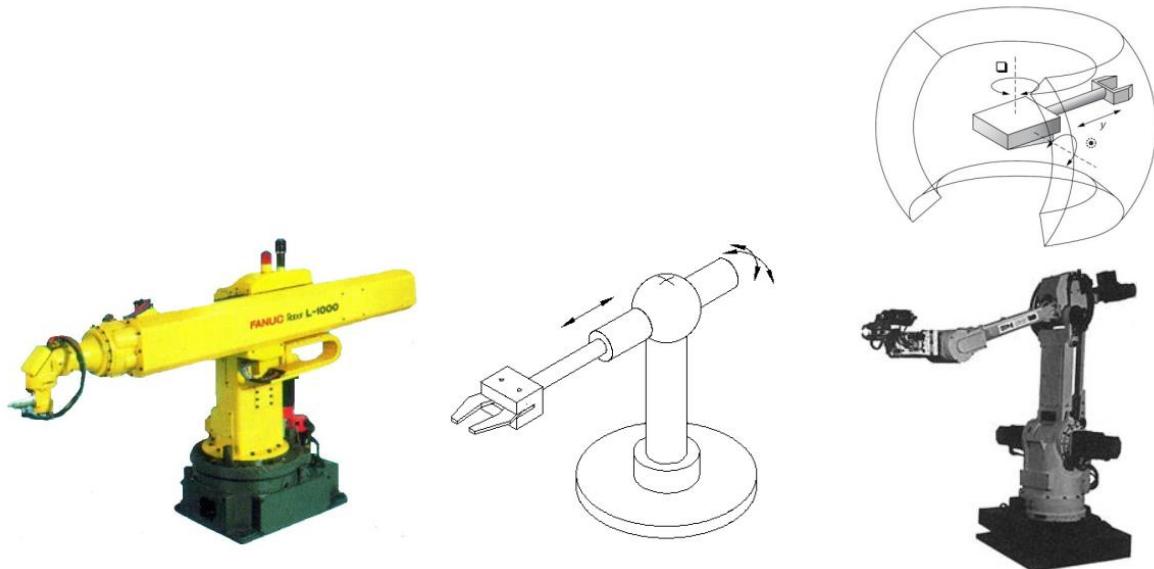
- Cartesian/Gantry Robots: Dạng robot có cấu trúc 3 khớp tịnh tiến theo phương của các trục tọa độ Descartes. Không gian làm việc có dạng hình hộp chữ nhật. Cartesian/Gantry Robot có kết cấu đơn giản, độ cứng vững cao, độ tuyến tính trong điều khiển cao, thường được dùng trong các ứng dụng gấp và đặt vật, lắp ráp, hàn hồ quang,...

CHƯƠNG 1. TỔNG QUAN



Hình 1.3 *Cartesian/Gantry Robots*

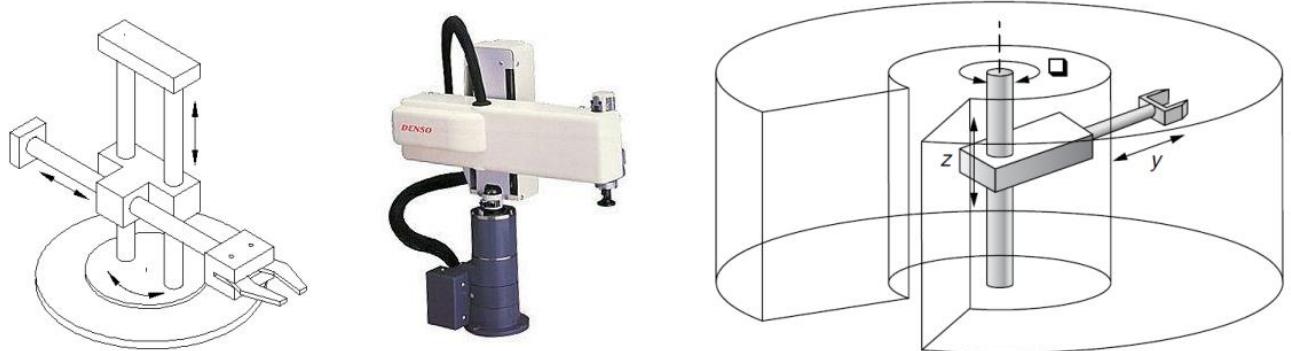
- Spherical/Polar Robots: Dạng robot có không gian làm việc hình cầu có 2 khớp quay và 1 khớp tịnh tiến.



Hình 1.4 *Spherical/Polar Robots*

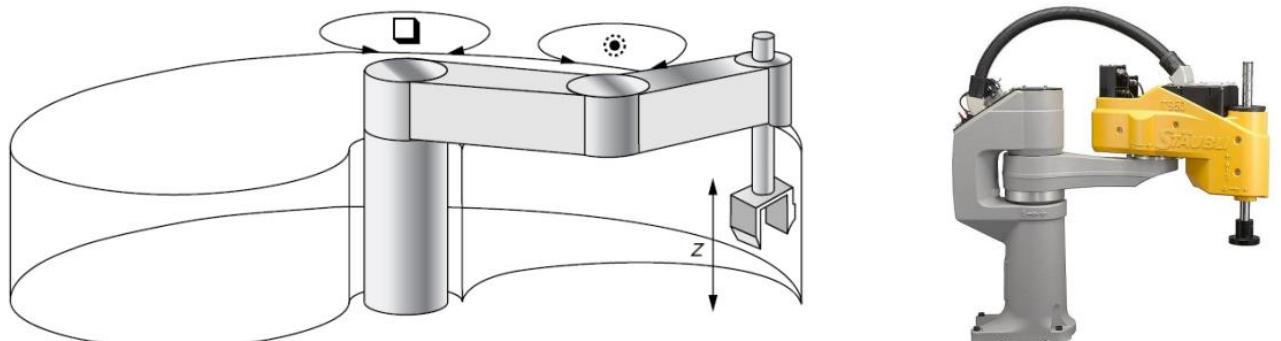
CHƯƠNG 1. TỔNG QUAN

- Cylindrical Robots: Dạng robot có không gian làm việc dạng hình trụ rỗng, có 1 khớp quay và 2 khớp tịnh tiến, được ứng dụng chủ yếu trong nâng hạ và sắp xếp sản phẩm.



Hình 1.5 Cylindrical Robots

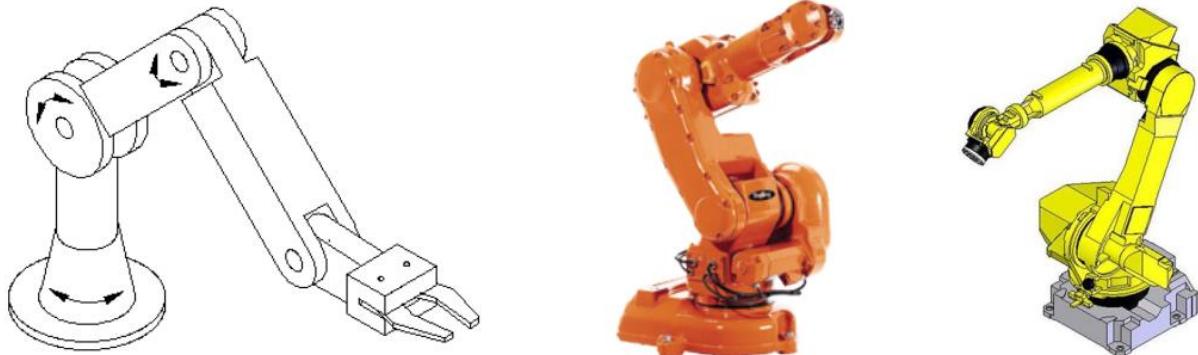
- SCARA Robots (Selective Compliance Assembly Robot Arm): Dạng robot thường được sử dụng trong các ứng dụng hạn chế về di chuyển thẳng đứng nhưng mức độ di chuyển ngang lớn. Thường được sử dụng cho các ứng dụng như gấp và đặt, lắp ráp,...



Hình 1.6 SCARA Robots

CHƯƠNG 1. TỔNG QUAN

- Articulated Robots (Robot khớp bản lề): Dạng robot có ít nhất 3 khớp quay. Là một dạng robot dựa trên cấu trúc nối tiếp. Độ cứng vững và linh hoạt cao, được sử dụng rộng rãi và ứng dụng nhiều.



Hình 1.7 Articulated Robots (Robot khớp bản lề)

- Parallel Robots (Robot song song): Là dạng robot bao gồm nhiều chuỗi động học kết nối bệ cố định với bộ phận đầu cuối. Robot cũng có thể được coi là một dạng khái quát không gian của liên kết bốn thanh.



Hình 1.8 Parallel Robots (Robot song song)

CHƯƠNG 1. TỔNG QUAN

1.4.2.2. Phân loại theo nguồn dẫn động

- Robot dùng nguồn cấp điện: Nguồn điện cấp cho robot là nguồn điện xoay chiều hoặc một chiều. Ưu điểm của robot loại này là thiết kế gọn, chạy êm, độ chính xác cao.

- Robot dùng nguồn khí nén: Là dạng robot được trang bị máy nén, bình chứa khí và động cơ kéo máy nén. Robot loại này thường được sử dụng trong các ứng dụng có tải trọng nhỏ. Có tay máy là các xy-lanh khí nén thực hiện chuyển động thẳng và chuyển động quay. Do khí nén là lưu chất nén được nên robot loại này thường sử dụng trong các thao tác gấp đặt không cần độ chính xác cao.

- Robot dùng nguồn thủy lực: Là dạng robot sử dụng nguồn thuỷ lực là lưu chất không nén được dầu ép. Hệ thống cần trang bị bơm để tạo áp lực dầu. Tay máy là các xy - lanh thuỷ lực chuyển động thẳng và quay động cơ dầu. robot loại này được sử dụng trong các ứng dụng có tải trọng lớn.

1.4.2.3. Phân loại theo cách điều khiển

- Điều khiển vòng hở: Truyền động bước được xác định từ đầu, quãng đường hoặc góc dịch chuyển tỷ lệ thuận với số xung điều khiển. Điều khiển vòng hở đơn giản nhưng độ chính xác thấp.

- Điều khiển vòng kín: Chuyển động của robot được xác định bằng cách sử dụng một vòng kín với tín hiệu được trả về nhờ cảm biến hoặc encoder. Tín hiệu hồi tiếp này giúp tăng độ chính xác.

1.4.2.4. Phân loại dựa trên loại ứng dụng

- Robot nâng.
- Robot hàn.
- Robot gấp đặt.
- Robot lắp ráp.
- Robot y tế, chăm sóc sức khỏe,...

CHƯƠNG 1. TỔNG QUAN

1.5. Mục tiêu, nhiệm vụ và phạm vi đề tài

1.5.1. Đầu đề luận văn

- Vật thể cần gấp làm bằng mica có hình dáng phức tạp
- Trọng lượng vật dưới 100g
- Vật thể được đặt trong vùng nhìn thấy của camera và vùng làm việc của robot 6 bậc tự do.
 - Vùng thả vật thể được đánh dấu bằng một đường bao có hình dạng tương tự vật thể.

1.5.2. Mục tiêu luận văn

- Áp dụng giải thuật Visual Servoing sử dụng image moment để điều khiển tay máy đến vị trí mong muốn với thời gian dưới 1 phút.
- Gấp và thả vật phẳng làm bằng mica có hình dáng phức tạp vào vị trí mong muốn.
- Sai số tối đa là 2mm.

1.5.3. Nhiệm vụ luận văn

- Tìm hiểu tổng quan và nguyên lý hoạt động của tay máy.
- Thiết kế đồ gá đầu hút.
- Thiết kế mạch điện và xây dựng chương trình điều khiển.
- Thực nghiệm kiểm nghiệm
- Số bản vẽ dự kiến :
 - 1 bản vẽ A0 : Phân tích, lựa chọn phương án thiết kế.
 - 1 bản vẽ A0 : Kết cấu cơ khí.
 - 1 bản vẽ A0 : Bản vẽ mạch điện.
 - 1 bản vẽ A0 : Lưu đồ giải thuật và điều khiển.

CHƯƠNG 1. TỔNG QUAN

1.5.4. Phạm vi đề tài

Đề tài ứng dụng các kiến thức về kỹ thuật robot, cơ điện và lập trình đã được dạy tại trường, đồng thời kết hợp với tham khảo và nghiên cứu các bài báo khoa học. Đề tài thiên về hướng kiểm tra tính đúng đắn và khả dụng của giải thuật đối với gấp thả vật đứng yên.

1.6. Tóm tắt các chương

Chương 1 Tổng quan

Chương 2 Lựa chọn phương án

Chương 3 Phân tích động học tay máy

Chương 4 Camera calibration

Chương 5 Xử lý ảnh

Chương 6 Giải thuật Image moment based Visual servoing

Chương 7 Hệ thống điện và cơ cấu chấp hành

Chương 8 Giao tiếp và điều khiển robot

Chương 9 Mô phỏng

Chương 10 Thực nghiệm và kết quả

Chương 11 Kết luận

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

2.1. Lựa chọn loại robot

Trong phần phân loại robot công nghiệp, chúng ta đã đề cập đến rất nhiều dạng robot đang được sử dụng trong công nghiệp hiện nay. Đối với ứng dụng gấp thả ta có các dạng robot Delta, SCARA, Robot khớp bản lề là được sử dụng nhiều nhất.

2.1.1. Phương án 1: Robot Delta

Robot Delta là một dạng robot song song, bao gồm ba cánh tay được nối với các khớp quay trên bệ cố định. Robot Delta thường được sử dụng cho các ứng dụng gấp thả ở tốc độ cao.



Hình 2.1 Delta robot

- **Ưu điểm:** Robot Delta là robot dạng song song nên có tốc độ làm việc rất cao. Các động cơ được gắn trên bệ làm cho các phần chuyển động của robot trở nên rất nhẹ.Thêm nữa việc tính toán động học ngược cho robot Delta khá đơn giản.

- **Nhược điểm:** Việc các bộ phận chuyển động của robot Delta nhẹ cũng làm cho độ cứng vững của robot thấp, ngoài ra vùng làm việc của robot Delta khá hạn chế. Robot Delta chỉ phù hợp cho các ứng dụng gấp thả ở tốc độ cao với tải siêu nhẹ.

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

2.1.2. Phương án 2: SCARA robot

SCARA robot là một dạng robot được biết đến phổ biến nhất với cấu hình bốn bậc tự do, bao gồm ba trục xoay và một trục tịnh tiến.



Hình 2.2 Mitsubishi SCARA Robot

- **Ưu điểm:** Robot SCARA chiếm diện tích làm việc nhỏ, có độ chính xác cao và tốc độ làm việc tốt. Phương trình động học của robot SCARA cũng đơn giản.

- **Nhược điểm:** Do không gian làm việc của robot SCARA là hình trụ nên nó không phù hợp với tất cả các ứng dụng gấp thả. Ngoài ra do đầu công tác vuông góc với mặt đất nên nó có hạn chế về tính linh hoạt. Do cấu trúc của nó nên nó thường chỉ nâng được tải trọng nhẹ.

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

2.1.3. Phương án 3: Robot khớp bắn lè

Robot khớp bắn lè là loại robot được sử dụng nhiều nhất trong công nghiệp hiện nay.



Hình 2.3 Robot khớp bắn lè Nachi MZ07

- **Ưu điểm:** Độ cứng vững cao, vùng làm việc rộng, số bậc tự do từ 4 đến 6 bậc nên độ linh hoạt cao, hay còn gọi là khả năng làm việc trong không gian linh hoạt.

- **Nhược điểm:** Hạn chế về tốc độ khi đặt cạnh các robot khác, động học tay máy phức tạp.

=> Kết luận: Mục tiêu của đề tài bao gồm ứng dụng gấp thả và kiểm nghiệm tính khả thi của giải thuật điều khiển hồi tiếp ảnh, nên sẽ lựa chọn dạng robot khớp bắn lè vì đây là dạng robot đáp ứng được độ linh hoạt và phạm vi hoạt động lớn phù hợp với các ứng dụng gấp thả.

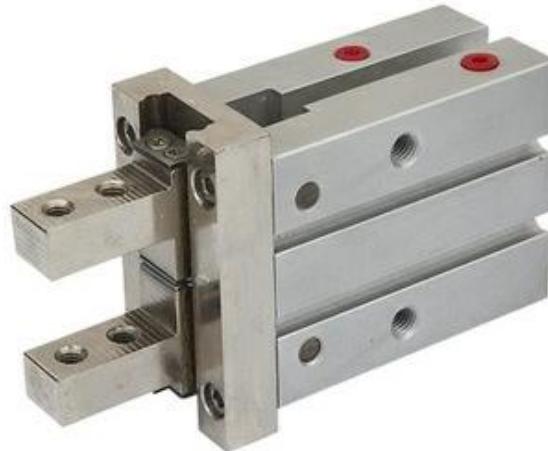
CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

2.2. Lựa chọn cơ cấu gấp vật

Trong điều khiển robot cơ cấu gấp vật thường được sử dụng nhất là tay gấp, giác hút và nam châm điện.

2.2.1. Phương án 1: Tay gấp

Cơ cấu tay gấp sử dụng các ngón tay để gấp vật, có thể có hai hoặc nhiều hơn các ngón tay để gấp vật phù hợp với yêu cầu làm việc.



Hình 2.4 Tay gấp thủy lực SMC MHZ2

- **Ưu điểm:** Linh hoạt, có thể tùy biến dễ dàng tùy theo mục đích sử dụng. Tùy ứng dụng mà có thể sử dụng tay gấp trợ lực điện hay trợ lực khí nén. Lực gấp chặt, có khả năng nâng được vật nặng.

- **Nhược điểm:** Cần xác định chính xác vị trí và hướng gấp giữa các ngón tay. Kích thước trống giữa các ngón tay gấp là giới hạn nên kích thước vật thể gấp được là giới hạn. Không kiểm soát được hành trình gấp, thường sẽ là tay gấp đi hết hành trình.

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

2.2.2. Phương án 2: Giác hút



Hình 2.5 Giác hút chân không FESTO

- **Ưu điểm:** Cấu tạo đơn giản hơn so với tay gấp, phù hợp với các vật thể có dạng mặt phẳng, giá thành rẻ. Không cần xác định hướng để có thể gấp vật, thay vào đó phải xác định trọng tâm của vật.

- **Nhược điểm:** Không hút được các vật có hình dạng phức tạp, thường chỉ dùng cho vật có hình dạng phẳng, không hút được các vật có khối lượng lớn.

2.2.3. Phương án 3: Nam châm điện



Hình 2.6 SMC Magnetic gripper

- **Ưu điểm:** Lực hút mạnh, dễ dàng lắp đặt và sử dụng, có thể hút được các vật có hình dạng phức tạp. Có thể tùy chỉnh lực hút.

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

- Nhược điểm: Chỉ hút được các vật có từ tính.

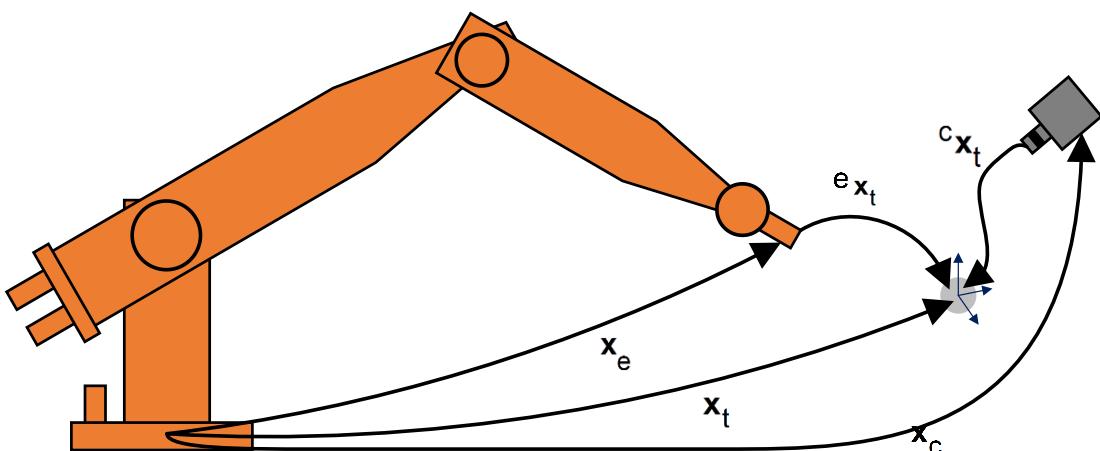
=> Kết luận: Do vật thể sử dụng trong khuôn khổ luận văn được làm bằng nhựa nên không thể sử dụng nam châm điện. Vật thể phẳng, có khối lượng nhỏ cùng hình dạng không phù hợp cho tay gấp, nên đề tài lựa chọn cơ cấu giác hút chân không trong quá trình thực hiện luận văn.

2.3. Lựa chọn cấu hình camera

Trong tác vụ điều khiển hồi tiếp ảnh có hai cấu hình gá đặt camera chủ yếu. Thứ nhất là camera được gắn cố định, thứ hai là camera được gắn trực tiếp trên cánh tay robot. Ứng với mỗi cấu hình gá đặt camera khác nhau sẽ có ứng dụng khác nhau và giải thuật điều khiển thích hợp.

2.3.1. Phương án 1: Camera được gắn ở vị trí cố định

Cấu hình gá đặt camera này thường được gọi với cái tên end-point open-loop. Đối với cấu hình camera được đặt ở vị trí cố định, vị trí và hướng của cát thể sẽ hoàn toàn do camera được gắn ở vị trí định sẵn ghi nhận rồi điều khiển robot đi đến vị trí mới bằng cách tính động học ngược như điều khiển thông thường.



Hình 2.7 Eye to hand

- Ưu điểm: Chỉ phải calib camera một lần, tốc độ làm việc cao, camera thường đặt ở vị trí có tầm nhìn rộng.

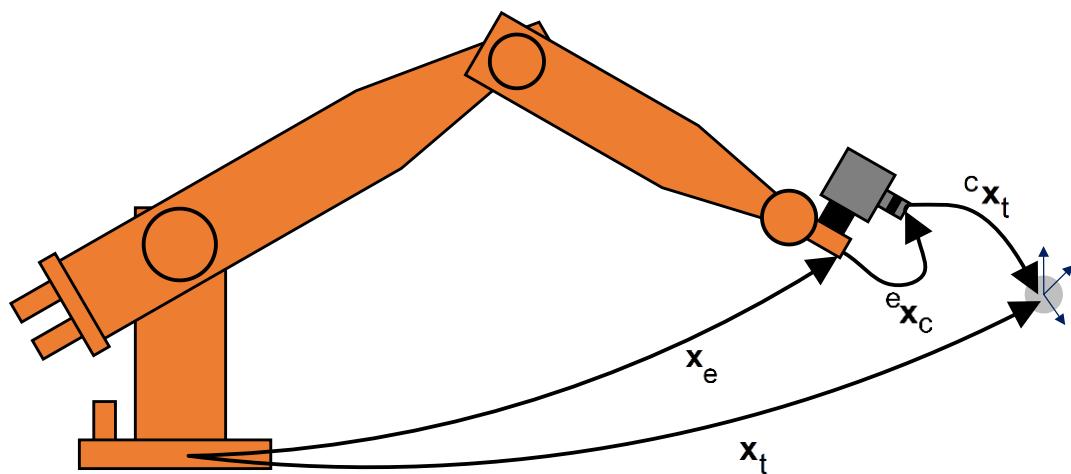
- Nhược điểm: Quá trình calib rất quan trọng để đảm bảo cho sự chính xác khi làm việc của hệ thống. Do giải thuật điều khiển là điều khiển vòng hở nên điều khiển tọa độ

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

đầu cuối là điều khiển mù, khó đảm bảo độ chính xác mong muốn. Đồng thời phải tìm được ma trận chuyển tọa độ từ camera về robot để chuyển tọa độ thu được bằng pixel trong camera thành tọa độ trong hệ tọa độ robot.

2.3.2. Camera được gắn trực tiếp trên tay máy

Cấu hình gá đặt camera này thường được gọi với cái tên end-point closed-loop hay eye-in-hand. End-point closed-loop nghĩa là việc điều khiển tọa độ đầu cuối camera sẽ có tín hiệu phản hồi, làm quá trình điều khiển trở thành một vòng kín. Eye-in-hand đơn giản là camera sẽ được gá trực tiếp trên tay robot.



Hình 2.8 Eye in hand

- **Ưu điểm:** Giải thuật điều khiển vòng kín cho phép điều khiển robot một cách chủ động. Không cần biết tư thế ban đầu của robot, vật thể.

- **Nhược điểm:** Tầm bao quát của cách gá đặt này là hạn chế so với cách gá camera cố định. Trường hợp vật bị trượt khỏi khung hình có thể diễn ra, tốc độ làm việc chậm hơn điều khiển vòng hở.

=> Kết luận: Để tài thực hiện hướng tới việc khắc phục các hạn chế về điều khiển theo phương pháp truyền thống như quá trình điều khiển là vòng hở, không thể điều khiển được nếu không biết được vị trí chính xác trong hệ tọa độ robot của vật thể, nên đề tài sẽ chọn cấu hình gá đặt camera gắn trực tiếp trên cánh tay robot.

CHƯƠNG 2. LỰA CHỌN PHƯƠNG ÁN

Bảng 2.1 Lựa chọn phương án để tài

Dạng robot sử dụng	Robot khớp bản lề
Cơ cấu gấp vật	Giác hút
Cấu hình camera	Eye-in-hand

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

3.1. Thông số kỹ thuật robot Nachi MZ07

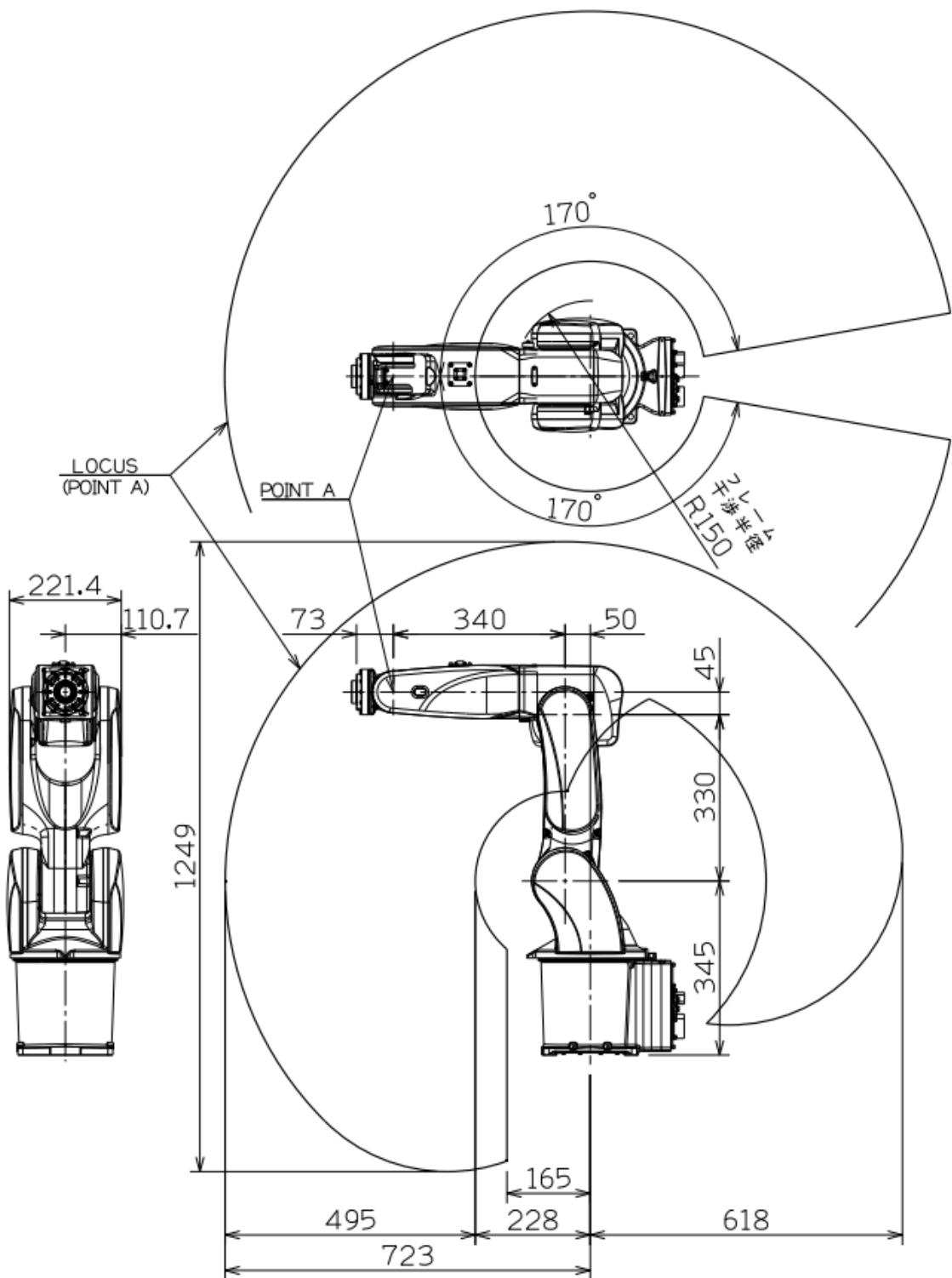
Bảng 3.1 Thông số kỹ thuật của robot Nachi MZ07

Tên	Thông số	
Dạng khớp	Xoay	
Số bậc tự do	6	
Động cơ	AC servo	
Vùng hoạt động tối đa	Axis 1	$\pm 170^\circ$
	Axis 2	$-135^\circ \sim 80^\circ$
	Axis 3	$-136^\circ \sim 270^\circ$
	Axis 4	$\pm 190^\circ$
	Axis 5	$\pm 120^\circ$
	Axis 6	$\pm 360^\circ$
Tốc độ tối đa	Axis 1	450 °/s
	Axis 2	380 °/s
	Axis 3	530 °/s
	Axis 4	550 °/s
	Axis 5	550 °/s
	Axis 6	1000 °/s
Tải tối đa	7 kg	

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

Moment tải tĩnh tối đa	Axis 4	16.6 N.m
	Axis 5	16.6 N.m
	Axis 6	9.4 N.m
Moment quán tính cho phép	Axis 4	0.47 kg.m ²
	Axis 5	0.47 kg.m ²
	Axis 6	0.15 kg.m ²
Độ hội tụ vị trí		±0.02 mm
Tầm với tối đa		723 mm
Óng khí nén		Φ6x2
Dây tín hiệu		10 wires
Dạng lắp đặt		Nèn/Tường/Nghiêng/Ngược
Điều kiện môi trường		Nhiệt độ: 0 – 45 °C Độ ẩm 20 tới 80% RH Đao động bề mặt làm việc: dưới 0.5G
Chỉ số IP		IP67
Độ ồn		70.2 dB
Khối lượng		30kg

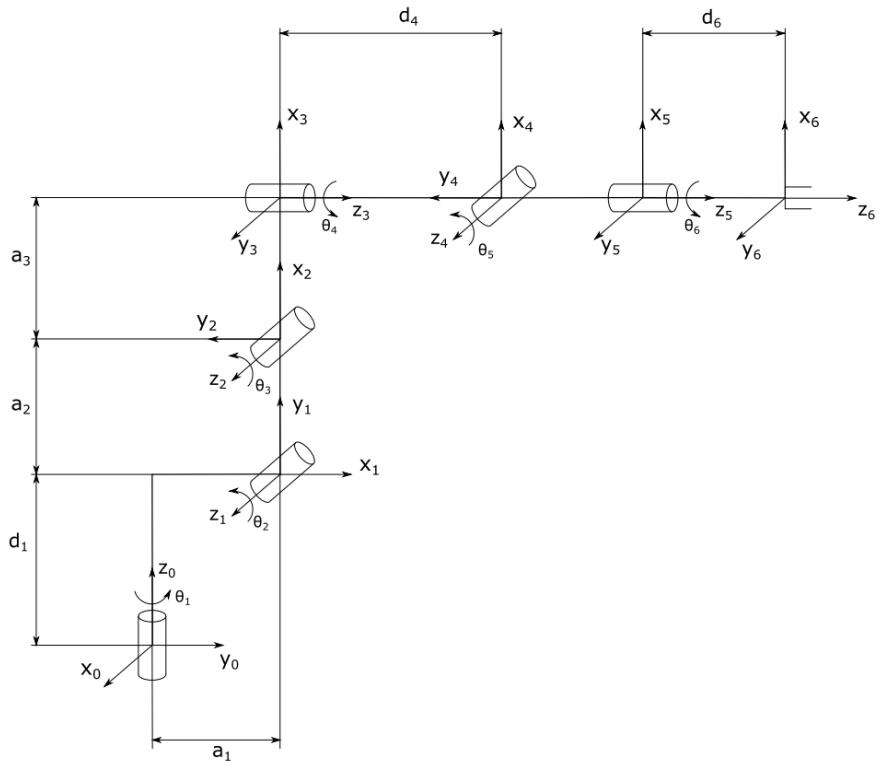
CHƯƠNG 3. ĐỘNG HỌC TAY MÁY



Hình 3.1 Vùng làm việc của robot Nachi MZ07

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

3.2. Phân tích động học thuận tay máy

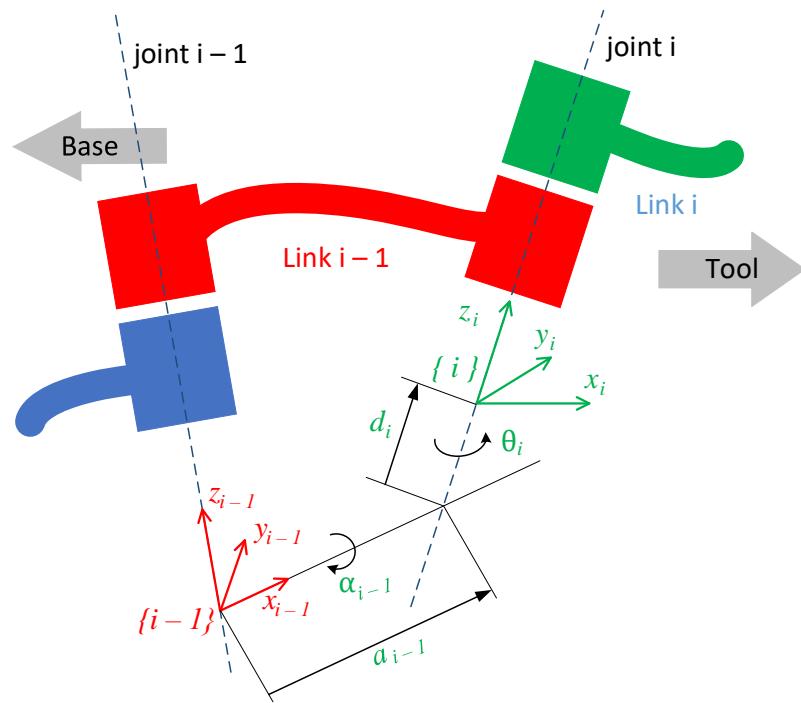


Hình 3.2 Cấu hình góc tọa độ tay máy

Bảng 3.2 Thông số Denavit – Hartenberg của robot Nachi MZ07

i	a_i (mm)	α_i (degree)	d_i	θ_i (degree)
1	50	90	345	θ_1
2	330	0	0	θ_2
3	45	90	0	θ_3
4	0	-90	340	θ_4
5	0	90	0	θ_5
6	0	0	73	θ_6

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY



Hình 3.3 Định nghĩa tham số bằng DH

- Độ dài khâu a_i (Link length): khoảng cách từ trục Z_{i-1} đến Z_i đo theo chiều trục X_i .
 - Góc xoắn α_i (Link twist): Góc giữa Z_{i-1} và Z_i đo theo Z_i .
 - Độ dời khâu d_i (Link offset): Khoảng cách từ trục X_{i-1} đến X_i đo theo chiều trục Z_i .
- 1.
- Góc quay khớp θ_i (Joint angle): Góc giữa X_{i-1} và X_i đo theo chiều trục Z_i .

Theo quy tắc D - H ma trận chuyển đổi từ tọa độ i về $i-1$:

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

Đặt $\sin \theta_i = s_i, \cos \theta_i = c_i, \sin(\theta_i + \theta_j) = s_{ij}, \cos(\theta_i + \theta_j) = c_{ij}$.

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

Từ đó ta có:

$$\begin{aligned} {}^0_1T &= \begin{bmatrix} c_1 & 0 & s_1 & a_1 c_1 \\ s_1 & 0 & -c_1 & a_1 s_1 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & {}^1_2T &= \begin{bmatrix} c_2 & -s_2 & 0 & a_2 c_2 \\ s_2 & c_2 & 0 & a_2 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^2_3T &= \begin{bmatrix} c_3 & 0 & s_3 & a_3 c_3 \\ s_3 & 0 & -c_3 & a_3 s_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & {}^3_4T &= \begin{bmatrix} c_4 & 0 & -s_4 & 0 \\ s_4 & 0 & c_4 & 0 \\ 0 & -1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^4_5T &= \begin{bmatrix} c_5 & 0 & s_5 & 0 \\ s_5 & 0 & -c_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, & {}^5_6T &= \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

Ngoài ra đề tài có sử dụng một cơ cấu giác hút có ma trận chuyển tọa độ như sau

$${}_{tool}^6T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_7 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.2)$$

d_7 là các thông số chuyển dịch của đầu công tác so với vị trí tâm của khớp thứ 6 của cánh tay robot. Các giá trị này có được sau quá trình calibration đầu công tác.

Ma trận chuyển đổi từ khâu 6 về gốc tọa độ của robot:

$${}_{tool}^0T = {}^0_1T \cdot {}^1_2T \cdot {}^2_3T \cdot {}^3_4T \cdot {}^4_5T \cdot {}^5_6T \cdot {}_{tool}^6T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.3)$$

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

Trong đó:

$$r_{11} = s_6(s_1c_4 - c_1s_4c_{23}) - c_6[c_1s_5s_{23} - c_5(s_1s_4 + c_1c_4c_{23})]$$

$$r_{12} = c_6(s_1c_4 - s_1s_4c_{23}) + s_6[c_1s_5s_{23} - c_5(s_1s_4 + c_1c_4c_{23})]$$

$$r_{13} = c_1c_5s_{23} + s_5(s_1s_4 + c_1c_4c_{23})$$

$$r_{21} = -c_6[s_1s_5s_{23} + c_5(c_1s_4 - s_1s_4c_{23})] - s_6(s_1s_4c_{23} + c_1c_4)$$

$$r_{22} = -c_6(c_1c_4 + s_1s_4c_{23}) + s_6[s_1s_5s_{23} + c_5(c_1s_4 - s_1s_4c_{23})]$$

$$r_{23} = s_1c_5s_{23} - s_5(c_1s_4 - s_1s_4c_{23})$$

$$r_{31} = c_6(c_{23}s_5 + s_{23}c_4c_5) - s_{23}s_4s_6$$

$$r_{32} = -s_6(c_{23}s_5 + s_{23}c_4c_5) - s_{23}s_4c_6$$

$$r_{33} = s_{23}c_4c_5 - c_{23}c_5$$

$$p_x = d_4c_1s_{23} + a_3c_1c_{23} + a_2c_1c_2 + a_1c_1 + (d_6 + d_7)[c_1c_5s_{23} + s_5(s_1s_4 + c_1c_4c_{23})]$$

$$p_y = d_4s_1s_{23} + a_3s_1c_{23} + a_2s_1c_2 + a_1s_1 + (d_6 + d_7)[s_1c_5s_{23} - s_5(c_1s_4 + s_1c_4c_{23})]$$

$$p_z = -d_4c_{23} + a_3s_{23} + a_2s_2 + d_1 - (d_6 + d_7)(c_5c_{23} - c_4s_5s_{23})]$$

3.3. Ma trận Jacobian

Đối với bài toán điều khiển hồi tiếp ảnh, chúng ta không biết được vị trí của vật thể trong hệ tọa độ cơ bản của robot, nên thông qua luật điều khiển, chúng ta khởi tạo 6 tham số về vận tốc tại đầu công tác, bao gồm 3 vận tốc tịnh tiến theo 3 trục và 3 vận tốc xoay. Vấn đề đặt ra là phải tìm được vận tốc của từng khớp sau khi có được vận tốc mong muốn ở tại đầu công tác, và ma trận Jacobian sẽ giúp chúng ta làm được điều đó.

Gọi vận tốc các khớp là \dot{q}

$$\dot{q} = [\dot{q}_1 \ \dot{q}_2 \ \dot{q}_3 \ \dot{q}_4 \ \dot{q}_5 \ \dot{q}_6]$$

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

Vận tốc của đầu công tác tổng hệ tọa độ đầu cuối là \dot{X}

$$\dot{X} = [\dot{X} \ \dot{Y} \ \dot{Z} \ \omega_X \ \omega_Y \ \omega_Z]$$

Động học vận tốc của robot được định nghĩa bởi mối quan hệ giữa vận tốc các khớp và vận tốc của đầu công tác theo công thức

$$\dot{X} = J\dot{q} \quad (3.4)$$

Đối với vận tốc góc

Sử dụng tham số bảng Denavit – Hartenberg của robot, vận tốc góc của khâu thứ i so với khâu i - 1 trong hệ tọa độ robot:

$${}^{i-1}\omega_i = {}^{i-1}z_{i-1} \dot{q} \quad (3.5)$$

Với ${}^{i-1}z_{i-1} [0 \ 0 \ 1]^T$ vì khớp của robot là khớp xoay.

Do vận tốc góc có tính chất cộng dồn, nên vận tốc góc của khớp thứ n trong hệ tọa độ robot được tính như sau

$${}^0\omega_n = \sum_{i=1}^n {}^0\omega_i = \sum_{i=1}^n {}^0R_{i-1}{}^{i-1}\omega_i = \sum_{i=1}^n {}^0R_{i-1}{}^{i-1}z_{i-1} \dot{q} = \sum_{i=1}^n {}^0z_{i-1} \dot{q}_i \quad (3.6)$$

So sánh với công thức biểu thị mối quan hệ giữa vận tốc các khớp và vận tốc của đầu công tác $\dot{X} = J\dot{q}$, ta có ma trận Jacobian trong hệ tọa độ robot có nửa dưới như sau, với $n = 6$ và ${}^0J_{\omega 0} = {}^0R_{i-1}{}^{i-1}z_{i-1}$

$${}^0J_{\omega} = \begin{bmatrix} {}^0J_{\omega 0} & {}^0J_{\omega 1} & {}^0J_{\omega 2} & {}^0J_{\omega 3} & {}^0J_{\omega 4} & {}^0J_{\omega 5} \end{bmatrix} \quad (3.7)$$

Đối với vận tốc dài:

Vận tốc của khâu thứ i là vận tốc của gốc tọa độ B_i gắn với khâu i. Gọi ${}^{i-1}d_i$ là vị trí gốc tọa độ của khâu i so với khâu i-1

$${}^{i-1}\dot{X}_i = {}^0\omega_i {}^{i-1}d_i \quad (3.8)$$

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

Tương tự như vận tốc góc ta tìm được ma trận Jacobian đối với vận tốc dài có công thức như sau:

$${}^0J_v = \begin{bmatrix} {}^0J_{v0} & {}^0J_{v1} & {}^0J_{v2} & {}^0J_{v3} & {}^0J_{v4} & {}^0J_{v5} \end{bmatrix} \quad (3.9)$$

$$\text{Với } {}^0J_{vi} = {}^0R_{i-1}{}^{i-1}z_{i-1}{}^{i-1}d_i = {}^{i-1}z_{i-1}{}^{i-1}d_n$$

Sau khi có hai ma trận thành phần, ta có ma trận Jacobian của robot 6 bậc tự do trong hệ tọa độ của robot là

$${}^0J = \begin{bmatrix} {}^0J_v \\ {}^0J_\omega \end{bmatrix} \quad (3.10)$$

Trong đề tài, ta sẽ có vận tốc của đầu công tác trong hệ tọa độ của nó, nghĩa là trong hệ tọa độ của đầu công tác, đòi hỏi ta phải tìm ma trận Jacobian trong hệ tọa độ của đầu công tác, khi đó

$${}^0J = {}^0R_6 \times {}^nJ \Rightarrow {}^nJ = {}^0R_6^{-1} \times {}^0J \quad (3.11)$$

3.4. Góc roll, pitch, yaw

Robot Nachi MZ07 mô tả đầu công tác bằng 6 thông số X, Y, Z, roll, pitch, yaw. Trong đó các góc roll, pitch, yaw được quy ước như sau.

- Góc roll là góc quay quanh trục z.
- Góc pitch là góc quay quanh trục y.
- Góc yaw là góc quay quanh trục x.

Thứ tự quay là x => y => z.

Đặt $\cos(\text{roll}) = c_r$, $\cos(\text{pitch}) = c_p$, $\cos(\text{yaw}) = c_y$.

$\sin(\text{roll}) = s_r$, $\sin(\text{pitch}) = s_p$, $\sin(\text{yaw}) = s_y$.

CHƯƠNG 3. ĐỘNG HỌC TAY MÁY

Ma trận xoay R của đầu công tác so với hệ tọa độ cơ sở có dạng:

$${}^0_6R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} c_p c_y & s_r s_p c_y - c_p s_y & c_r s_p c_y + s_r s_y \\ c_p s_y & s_r s_p c_y + c_p c_y & c_r s_p c_y - s_r c_y \\ -s_p & s_r c_p & c_r c_p \end{bmatrix} \quad (3.12)$$

Ngược lại nếu có ma trận xoay ${}^0_6R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$, ta có thể tìm được các góc roll, pitch, yaw theo công thức:

$$\begin{cases} roll = \arctan\left(\frac{r_{21}}{r_{11}}\right) \\ pitch = -\arcsin(r_{31}) \\ yaw = \arctan\left(\frac{r_{32}}{r_{31}}\right) \end{cases} \quad (3.13)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

4.1. Lý thuyết chung về Visual Servoing

Để thực hiện một nhiệm vụ trong visual servoing, một tập hợp các đặc trưng thị giác được lựa chọn từ hình ảnh cho phép điều khiển các bậc tự do mong muốn. Một luật điều khiển cũng được thiết kế để các đặc trưng thị giác s bằng với các đặc trưng mong muốn s^* . Do đó, tất cả các tác vụ visual servoing đều nhằm mục tiêu triệt tiêu sai số $e(t)$ được định nghĩa [11]:

$$e(t) = s(m(t), a) - s^* \quad (4.1)$$

Trong đó vector $m(t)$ là một tập hợp các giá trị trong hình ảnh (ví dụ như tọa độ điểm ảnh, hoặc diện tích, trọng tâm và các đặc trưng hình học khác của đối tượng). Các giá trị hình ảnh này phụ thuộc vào vị trí giữa camera và môi trường. Chúng được sử dụng để tính toán vector $s(m(t), a)$ của các đặc trưng thị giác, trong đây a là tập các thông số biểu diễn cho sự hiểu biết thêm về hệ thống (ví dụ như ma trận thông số nội của camera hoặc mô hình ba chiều của đối tượng). Vector s^* chứa các giá trị mong muốn của các đặc trưng mà có thể có giá trị không đổi nếu mục tiêu cố định hoặc thay đổi nếu đối tượng di chuyển theo một quỹ đạo nhất định.

Khi các biến điều khiển được định nghĩa trong hệ tọa độ của camera, để tối thiểu sai số $e(t)$, cách tiếp cận đơn giản nhất là điều khiển vận tốc của robot (cũng là vận tốc camera). Quan hệ giữa đạo hàm của vector s và vận tốc camera $v_c = (v_c, \omega_c)$ được cho bởi [11]:

$$\dot{s} = L_s v_c \quad (4.2)$$

Trong đó $L_s \in R^{k \times n}$ là ma trận tương tác (interaction matrix) hay ma trận Jacobian phụ thuộc vào các đặc trưng s .

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Từ công thức (4.1) và (4.2), ta có mối quan hệ giữa vận tốc camera và sai số e được cho bởi công thức:

$$\dot{e} = L_s v_c \quad (4.3)$$

Để đảm bảo tiệt tiêu được một cách tốt nhất sai số của vector đặc trưng, sử dụng luật điều khiển theo phương trình bậc nhất sau [11]:

$$\dot{e} = -\lambda e \quad (4.4)$$

Từ công thức (4.2) và (4.4), ta có được mối quan hệ giữa vận tốc camera và sai số e được cho bởi công thức:

$$v_c = -\lambda L_s^+ e = -\lambda L_s^+ (s - s^*) \quad (4.5)$$

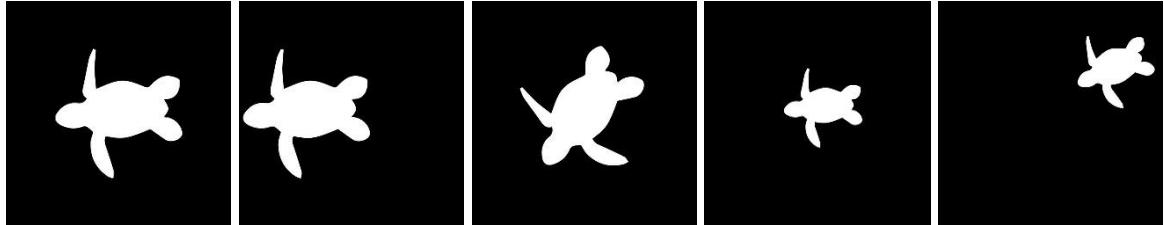
Với $L_s^+ = (L_s^T L_s)^{-1} L_s^T$ là ma trận giả nghịch đảo của L_s . Trong thực tế ta không thể biết chính xác được các ma trận L_s , L_s^+ . Vì thế các ma trận này sẽ được ước lượng hay xấp xỉ trong thực tế. Ta kí hiệu \hat{L}_s^+ là ma trận xấp xỉ của L_s^+ trong thực tế. Vì thế nên luật điều khiển trong thực tế được cho bởi công thức:

$$v_c = -\lambda \hat{L}_s^+ e = -\lambda \hat{L}_s^+ (s - s^*) \quad (4.6)$$

4.2. Visual servoing sử dụng moments ảnh

4.2.1. Giới thiệu về moments ảnh

Xét một ví dụ đơn giản với cá bức ảnh đen trắng dưới đây:



Hình 4.1 Ví dụ về Image Moments

Giác quan con người có thể dễ dàng nhận ra đây là các bức ảnh của cùng một vật thể được dịch chuyển, xoay hay thu nhỏ nhưng để thực hiện được điều này trên máy tính thì không hề đơn giản. Các thuật toán xử lý ảnh thông thường làm việc trên giá trị các pixel và do đó gặp khó khăn ngay cả với việc phân biệt hai bức ảnh có mô tả cùng một vật thể hay không vì chỉ cần một dịch chuyển nhỏ của vật thể trong ảnh sẽ làm cho ma trận pixel khác nhau hoàn toàn.

Để giải quyết vấn đề này, người ta nghĩ đến việc tính toán các thông số đặc trưng của một bức ảnh và làm việc trên các thông số đó. Một trong các công cụ thường dùng nhất là tính moments ảnh.

Về mặt toán học, moments trong không gian hai chiều được định nghĩa như sau:

$$m_{pq} = \iint_S x^p y^q f(x, y) dx dy \quad (4.7)$$

Trong đó S là diện tích toàn bộ vật thể, $f(x, y)$ là một hàm biểu thị cường độ của một đại lượng nào đó tại điểm (x, y) . Đối với một vật thể thì $f(x, y)$ có thể là mật độ khối lượng. Còn đối với một bức ảnh, $f(x, y)$ có thể là độ sáng của pixel. Về lý thuyết, công thức trên có thể áp dụng được cả với ảnh xám nhưng trên thực tế người ta chỉ tính moments của ảnh đen trắng nên dưới đây ta chỉ xét đen trắng, do đó có thể coi $f(x, y) = 1$. Hơn nữa, một bức ảnh số sẽ không liên tục theo không gian, nên trên thực tế, người ta dùng công thức dưới đây để tính moments ảnh:

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

$$m_{pq} = \sum_x \sum_y x^p y^q \quad (4.8)$$

Moment m_{pq} có bậc là $(p + q)$. Moment ảnh đơn giản nhất là moment bậc 0, $m_{00} = \sum_x \sum_y 1$, có thể hiểu giá trị của m_{00} theo hai cách:

- Số pixel trắng có trong hình.
- Diện tích của vật thể trong hình.

Có thể thấy rằng cho dù vật thể trong ảnh có bị dịch chuyển hay xoay thì giá trị của m_{00} vẫn không đổi, m_{00} được gọi là bất biến với phép tịnh tiến và phép xoay. Do đó trong số các bức ảnh ở trên thì ba ảnh đầu cho cùng một giá trị m_{00} . Nhưng khi vật thể bị phóng to hay thu nhỏ thì giá trị của m_{00} cũng thay đổi theo, hơn nữa tồn tại rất nhiều trường hợp hai vật thể có cùng diện tích, hay nói cách khác cùng giá trị m_{00} nhưng chúng hoàn toàn khác nhau. Như vậy, một mình m_{00} không thể hoàn toàn đại diện cho một bức ảnh mà chỉ thể hiện một đặc trưng, một “góc nhìn” vật thể.

Ngoài m_{00} , ta còn có thể tính được các đặc trưng vật lý khác của ảnh từ moments, chẳng hạn:

$$\begin{cases} x_g = \frac{m_{10}}{m_{00}} \\ y_g = \frac{m_{01}}{m_{00}} \end{cases} \quad (4.9)$$

là tọa độ khói tâm của vật thể trong ảnh.

Không những thế, từ moments ảnh ta còn có thể tính được các đặc trưng mà tuy không có ý nghĩa vật lý cụ thể nhưng lại đóng vai trò quan trọng trong tính toán.

4.2.2. Moments khói tâm và các bất biến

Công thức moments được đề cập ở trên là moments thuần túy. Còn một loại moments nữa cũng được dùng rất nhiều là moments khói tâm có công thức như sau:

$$\mu_{pq} = \iint_S (x - x_g)^p (y - y_g)^q f(x, y) dx dy \quad (4.10)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Hay đối với ảnh đen trắng:

$$\mu_{pq} = \sum_x \sum_y (x - x_g)^p (y - y_g)^q \quad (4.11)$$

Moments khói tâm là các bất biến đối với phép tịnh tiến. Moments khói tâm có thể dùng để suy ra được các bất biến đối với cả phép tịnh tiến và vị tự theo công thức sau:

$$\eta_{pq} = \frac{\mu_{pq}}{m_{00}^{(p+q)/2+1}} \quad (4.12)$$

Các η_{pq} được gọi là moments khói tâm chuẩn hóa.

Ngoài ra moments khói tâm còn có thể dùng để tính toán các bất biến đối với cả tịnh tiến và xoay, dưới đây là một vài trong số đó:

$$i_1 = \mu_{20}\mu_{02} - \mu_{11}^2 \quad (4.13)$$

$$i_2 = -\mu_{30}\mu_{12} + \mu_{21}^2 - \mu_{03}\mu_{21} + \mu_{12}^2 \quad (4.14)$$

$$i_3 = 3\mu_{30}\mu_{12} + \mu_{30}^2 + 3\mu_{03}\mu_{21} + \mu_{03}^2 \quad (4.15)$$

Từ 3 bất biến trên còn có thể tính được các bất biến đối với cả ba phép biến đổi tịnh tiến, xoay và vị tự:

$$i_{n1} = \frac{i_2}{i_1^{8/10}}, i_{n2} = \frac{i_3}{i_1^{8/10}}, i_{n3} = \frac{i_3}{i_2}, i_{n4} = \frac{i_3}{m_{00}^5}, i_{n5} = \frac{i_2}{m_{00}^5}, i_{n6} = \frac{i_1}{m_{00}^4} \quad (4.16)$$

M=Ngoài 3 biến đổi cơ bản là tịnh tiến, xoay và vị tự còn có các biến đổi khác như biến đổi đối xứng, phép biến hình affine,... và moments ảnh cũng có thể dùng để tính được các bất biến đối với các loại biến đổi này. Trong thư viện OpenCV có hỗ trợ các đặc trưng ảnh đặc biệt được gọi là các Hu moments, bao gồm 7 moment.

$$\phi_1 = \eta_{20} + \eta_{02} \quad (4.17)$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \quad (4.18)$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \quad (4.19)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \quad (4.20)$$

$$\begin{aligned} \phi_5 = & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) \left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2 \right] \\ & + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 \right] \end{aligned} \quad (4.21)$$

$$\begin{aligned} \phi_6 = & (\eta_{20} - \eta_{02}) \left[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 \right] \\ & + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \end{aligned} \quad (4.22)$$

$$\begin{aligned} \phi_7 = & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12}) \left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2 \right] \\ & - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03}) \left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 \right] \end{aligned} \quad (4.23)$$

Chúng là các bất biến đối với phép tịnh tiến, xoay, vị tự và 6 moment đầu bất biến với cả phép đối xứng, moment thứ 7 sẽ đổi dấu khi gấp phép đối xứng.

Các bất biến nói trên có ý nghĩa to lớn trong lĩnh vực xử lý ảnh. Lấy ví dụ hai bức ảnh có giá trị Hu moments giống nhau nhưng một đặc trưng nào đó không bất biến với phép xoay có giá trị khác nhau thì có thể kết luận hai bức ảnh mô tả cùng một vật thể và bức ảnh thứ hai bị xoay một góc so với bức ảnh thứ nhất.

Qua ví dụ trên có thể thấy được tầm quan trọng của moments ảnh, chúng thường được sử dụng trong các thuật toán nhận dạng và template matching. Ngoài ra chúng còn có thể được dùng trong các thuật toán Visual Servoing.

4.2.3. Thuật toán Visual Servoing sử dụng moments ảnh

Như đã nói ở trên, mỗi đặc trưng không thể đại diện hoàn toàn cho vật thể nên ta phải sử dụng nhiều đặc trưng và điều khiển vị trí camera sao cho tất cả các đặc trưng ấy đều đạt được giá trị mong muốn, khi đó camera cũng sẽ nằm ở vị trí mong muốn. Đó là ý tưởng cơ bản của việc ứng dụng moments ảnh vào Visual Servoing. Phương pháp này theo lý thuyết có thể hoạt động với vật thể có hình dạng phức tạp, đồng thời không đòi hỏi phải đánh dấu trước.

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Như đã trình bày ở phần 6.1, ta có vector các đặc trưng là $s = [s_1, s_2, \dots, s_n]^T$, ma trận tương tác L_s .

$$\dot{e} = L_s v_c \quad (4.24)$$

Mục tiêu điều khiển là để cho các đặc trưng s tiến gần đến các giá trị mong muốn s^* . Đặt $e = s - s^*$, chọn hàm Lyapunov xác định dương [14]:

$$V = \frac{1}{2} e^T e = \frac{1}{2} \left[(s_1 - s_1^*)^2 + (s_2 - s_2^*)^2 + \dots + (s_n - s_n^*)^2 \right] \quad (4.25)$$

Suy ra:

$$\dot{V} = e^T \dot{e} = e^T \dot{s} = e^T L_s v_c \quad (4.26)$$

Đặt luật điều khiển như sau [14]:

$$v_c = -\lambda L_s^{-1} e \quad (4.27)$$

Thực tế các thông số cần điều khiển là vận tốc góc các khớp của robot, có thể tính được từ nghịch đảo ma trận Jacobian:

$$\dot{\theta} = J^{-1} v_c = -\lambda J^{-1} L_s^{-1} e \quad (4.28)$$

Với λ là một số nguyên dương. Khi đó:

$$\dot{V} = -\lambda e^T e \leq 0 \quad (4.29)$$

Vậy hệ ổn định.

Do camera trong không gian ba chiều có 6 bậc tự do nên cần 6 thông số đặc trưng của ảnh $s = [s_1, s_2, s_3, s_4, s_5, s_6]^T$ để ma trận tương tác L_s khả nghịch. Đồng thời cần phải tính ma trận $L_s = [L_{s_1}, L_{s_2}, L_{s_3}, L_{s_4}, L_{s_5}, L_{s_6}]^T$ của 6 thông số này. Với L_{s_i} là vector liên hệ giữa độ biến thiên của đặc trưng s_i với dịch chuyển của camera:

$$\dot{s}_i = L_{s_i} v_c \quad (4.30)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Do đặc trưng ảnh là một hàm của các moments ảnh $s_i = F_i(m_{pq}, \mu_{pq})$ nên luôn có thể biểu diễn $\dot{s}_i = G_i(m_{pq}, \mu_{pq})\dot{m}_{pq} + H_i(m_{pq}, \mu_{pq})\dot{\mu}_{pq}$.

Từ đó, có thể quy về bài toán tìm vector liên hệ $L_{m_{pq}}$ và $L_{\mu_{pq}}$ giữa độ biến thiên của moments ảnh với dịch chuyển của camera.

$$\dot{m}_{pq} = L_{m_{pq}} v_c \quad (4.31)$$

$$\dot{\mu}_{pq} = L_{\mu_{pq}} v_c \quad (4.32)$$

Tác giả F. Chaumette đã dựa vào định lý Green để xác định được dạng giải tích của các vector liên hệ này đối với các vật thể phẳng [14]:

$$L_{m_{pq}} = [m_{vx}, m_{vy}, m_{vz}, m_{\omega x}, m_{\omega y}, m_{\omega z}] \quad (4.33)$$

Với:

$$\left\{ \begin{array}{l} m_{vx} = -p(Am_{pq} + Bm_{p-1,q+1} + Cm_{p-1,q}) - Am_{pq} \\ m_{vy} = -q(Am_{p+1,q-1} + Bm_{pq} + Cm_{p,q-1}) - Bm_{pq} \\ m_{vz} = (p+q+3)(Am_{p+1,q} + Bm_{p,q+1} + Cm_{pq}) - Cm_{pq} \\ m_{\omega x} = (p+q+3)m_{p,q+1} + qm_{p,q-1} \\ m_{\omega y} = (p+q+3)m_{p+1,q} - pm_{p-1,q} \\ m_{\omega z} = pm_{p-1,q+1} - qm_{p+1,q-1} \end{array} \right. \quad (4.34)$$

Đối với moments khói tâm [11]:

$$L_{\mu_{pq}} = [\mu_{vx}, \mu_{vy}, \mu_{vz}, \mu_{\omega x}, \mu_{\omega y}, \mu_{\omega z}] \quad (4.35)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Với:

$$\left\{ \begin{array}{l} \mu_{vx} = -(p+1)A\mu_{pq} - pB\mu_{p-1,q+1} \\ \mu_{vy} = -qA\mu_{p+1,q-1} - (p+1)B\mu_{pq} \\ \mu_{vz} = -A\mu_{\omega y} + B\mu_{\omega x} + (p+q+2)C\mu_{pq} \\ \mu_{\omega x} = (p+q+3)\mu_{p,q+1} + px_g\mu_{p-1,q+1} + (p+2q+3)y_g\mu_{pq} - 4pn_{11}\mu_{p-1,q} - 4qn_{02}\mu_{p,q-1} \\ \mu_{\omega y} = -(p+q+3)\mu_{p+1,q} - (2p+q+3)x_g\mu_{pq} - py_g\mu_{p+1,q-1} + 4pn_{20}\mu_{p-1,q} + 4qn_{11}\mu_{p,q-1} \\ \mu_{\omega z} = p\mu_{p-1,q+1} - q\mu_{p+1,q-1} \end{array} \right. \quad (4.36)$$

Trong đó:

- $n_{pq} = \mu_{pq} / m_{00}$
- A, B, C là ba hệ số của phương trình mặt phẳng $\frac{1}{Z} = Ax + By + C$. Trường hợp mặt phẳng vật thể song song với mặt phẳng ảnh của camera thì $A = B = 0$.
- (x_g, y_g) là tọa độ khói tâm của vật thể.

Đến đây ta đã biết được phương pháp tìm ma trận tương tác L_s của các đặc trưng ảnh. Vấn đề còn lại là lựa chọn các đặc trưng ảnh phù hợp. Việc lựa chọn các đặc trưng ảnh phải tuân theo các tiêu chí sau:

- Các đặc trưng phải được lựa chọn sao cho ma trận L_s có cấu trúc tách rời, tức là các hang và cột của ma trận phải khác biệt rõ ràng để tránh nguy cơ L_s bị suy biến.
- Condition number của L_s phải nhỏ để một thay đổi nhỏ ở vé phải của phương trình (3) sẽ dẫn đến 1 thay đổi nhỏ ở vé trái. Như vậy sẽ đảm bảo độ mượt và độ ổn định trong hoạt động của cánh tay robot.

Các tác giả trước đó đã nghiên cứu về đề tài này đã đề xuất cách chọn như sau [15]:

$$s = [x_n, y_n, a_n, s_4, s_5, \theta] \quad (4.37)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Trong đó [15]:

- $a_n = Z^* \sqrt{\frac{m_{00}^*}{m_{00}}}$, với m_{00}^* là diện tích mong muốn của vật thể trong ảnh và Z^* là khoảng cách mong muốn từ camera đến vật thể, a_n là một bất biến đối với tịnh tiến và xoay, chủ yếu được dùng để điều khiển vận tốc v_z của camera.
- $x_n = x_g a_n$, $y_n = y_g a_n$ với (x_g, y_g) là moment khối tâm của vật thể, x_n và y_n chủ yếu được dùng để điều khiển vận tốc v_x và v_y của camera.
- s_4, s_5 là hai bất biến với cả ba phép biến đổi tịnh tiến, xoay và vị tự. Được dùng để điều khiển tốc độ xoay ω_x và ω_y của camera.
- θ là góc quay của vật thể, dùng để điều khiển tốc độ xoay ω_z của camera và được tính theo công thức:

$$\theta = \frac{1}{2} \arctan \left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \right) \quad (4.38)$$

Có rất nhiều cách chọn s_4 và s_5 do tồn tại nhiều bất biến đối với các phép biến đổi tịnh tiến, xoay và vị tự. Ví dụ chúng có thể được chọn từ tập hợp các bất biến đã đề cập ở mục (4.2.2). Luận văn này lựa chọn hai Hu moments đầu tiên trong số 7 Hu moments làm s_4 và s_5 do Hu moments được thư viện OpenCV hỗ trợ sẵn và hai Hu moments đầu có công thức đơn giản nhất, thuận tiện cho việc tính toán ma trận tương tác.

Với cách chọn đặc trưng như trên thì ma trận tương tác khi mặt phẳng vật thể song song với mặt phẳng ảnh được cho bởi [15], [17]:

$$L_s = \begin{bmatrix} L_{x_n} \\ L_{y_n} \\ L_{a_n} \\ L_{s_4} \\ L_{s_5} \\ L_\theta \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & a_n \varepsilon_{11} & -a_n (1 + \varepsilon_{12}) & y_n \\ 0 & -1 & 0 & a_n (1 + \varepsilon_{21}) & -a_n \varepsilon_{22} & -x_n \\ 0 & 0 & -1 & -a_n \varepsilon_{31} & a_n \varepsilon_{32} & 0 \\ 0 & 0 & 0 & s_{4\omega x} & s_{4\omega y} & 0 \\ 0 & 0 & 0 & s_{5\omega x} & s_{5\omega y} & 0 \\ 0 & 0 & 0 & \theta_{\omega x} & \theta_{\omega y} & -1 \end{bmatrix} \quad (4.39)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Với:

- $\varepsilon_{11} = \varepsilon_{22} = 4n_{11} - \frac{x_g y_g}{2}$
- $\varepsilon_{12} = 4n_{20} - \frac{x_g^2}{2}$
- $\varepsilon_{21} = 4n_{02} - \frac{y_g^2}{2}$
- $\varepsilon_{31} = \frac{3y_g}{2}$
- $\varepsilon_{32} = \frac{3x_g}{2}$
- $n_{pq} = \frac{\mu_{pq}}{m_{00}}$

θ_{ω_x} và θ_{ω_y} được cho bởi công thức [17]:

$$\left\{ \begin{array}{l} \theta_{\omega_x} = \frac{\beta(\mu_{12}(\mu_{20} - \mu_{02}) + \mu_{11}(\mu_{03} - \mu_{21})) + \gamma x_g (\mu_{02}(\mu_{20} - \mu_{02}) - 2\mu_{11}^2) + \gamma y_g \mu_{11}(\mu_{20} + \mu_{02})}{d} \\ \theta_{\omega_y} = \frac{\beta(\mu_{21}(\mu_{02} - \mu_{20}) + \mu_{11}(\mu_{30} - \mu_{12})) + \gamma x_g \mu_{11}(\mu_{20} + \mu_{02}) + \gamma(\mu_{20}(\mu_{02} - \mu_{20}) - 2\mu_{11}^2)}{d} \\ d = (\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2 \end{array} \right. \quad (4.40)$$

Trong đó $\beta = 5, \gamma = 1$ đối với vật thể đặc và $\beta = 4, \gamma = 2$ đối với vật thể rời rạc (vật thể dooming tập hợp các điểm rời rạc) [17].

Các thành phần $s_{4\omega_x}, s_{4\omega_y}, s_{5\omega_x}, s_{5\omega_y}$ được tính như sau:

Công thức s_4 và s_5 được cho bởi:

$$s_4 = \eta_{20} + \eta_{02}$$

$$s_5 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

Đạo hàm 2 vé:

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

$$\dot{s}_4 = \dot{\eta}_{20} + \dot{\eta}_{02}$$

$$\Rightarrow L_{s_4} v_c = L_{\eta_{20}} v_c + L_{\eta_{02}} v_c$$

$$\Rightarrow L_{s_4} = L_{\eta_{20}} + L_{\eta_{02}} \quad (4.41)$$

$$\dot{s}_5 = 2(\eta_{20} - \eta_{02})(\dot{\eta}_{20} + \dot{\eta}_{02}) + 4\eta_{11}\dot{\eta}_{11}$$

$$\Rightarrow L_{s_5} v_c = 2(\eta_{20} - \eta_{02})(L_{\eta_{20}} v_c - L_{\eta_{02}} v_c) + 4\eta_{11}L_{\eta_{11}} v_c$$

$$\Rightarrow L_{s_5} = 2(\eta_{20} - \eta_{02})(L_{\eta_{20}} - L_{\eta_{02}}) + 4\eta_{11}L_{\eta_{11}} \quad (4.42)$$

Trong đó:

$$\eta_{pq} = \frac{\mu_{pq}}{m_{00}^{(p+q)/2+1}}$$

$$\Rightarrow \eta_{pq} = \frac{\dot{\mu}_{pq}}{m_{00}^{(p+q)/2+1}} - \left(\frac{p+q}{2} + 1 \right) \frac{\mu_{pq}}{m_{00}^{(p+q)/2+1}} m_{00}^{\bullet}$$

$$\Rightarrow L_{n_{pq}} = \frac{L_{\mu_{pq}}}{m_{00}^{(p+q)/2+1}} - \left(\frac{p+q}{2} + 1 \right) \frac{\mu_{pq}}{m_{00}^{(p+q)/2+1}} L_{m_{00}} \quad (4.43)$$

Với $L_{m_{00}}$ và $L_{\mu_{pq}}$ có thể tính được bằng công thức (4.33) và (4.35). Cuối cùng ta có
được:

$$s_{4\omega x} = \frac{2x_g\mu_{11} + y_g\mu_{02} - y_g\mu_{20} + 5\mu_{03} + 5\mu_{21}}{m_{00}^2} \quad (4.44)$$

$$s_{4\omega y} = \frac{x_g\mu_{02} - x_g\mu_{20} - 2y_g\mu_{11} - 5\mu_{12} - 5\mu_{30}}{m_{00}^2} \quad (4.45)$$

$$s_{5\omega x} = \frac{8\eta_{11}(x_g\mu_{02} + 5\mu_{12}) - 2(\eta_{02} - \eta_{20})(2x_g\mu_{11} - y_g\mu_{02} - y_g\mu_{20} - 5\mu_{03} + 5\mu_{21})}{m_{00}^2} \quad (4.46)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

$$s_{5\omega y} = \frac{-8\eta_{11}(y_g\mu_{20} + 5\mu_{21}) + 2(\eta_{02} - \eta_{20})(x_g\mu_{02} + x_g\mu_{20} - 2y_g\mu_{11} - 5\mu_{12} + 5\mu_{30})}{m_{00}^2} \quad (4.47)$$

Lý do người ta chỉ quan tâm đến ma trận tương tác khi mặt phẳng vật thể song song với mặt phẳng ảnh là bởi trong thực tế thường hướng đến việc điều khiển cánh tay robot đến vị trí sao cho đầu công tác vuông góc với mặt phẳng vật thể. Hơn nữa, trong quá trình điều khiển, khó có cách nào biết được các hệ số A, B trong phương trình mặt phẳng của vật thể để tính ra ma trận L_s tại từng thời điểm nên có thể coi L_s tại A = B = 0 như một xấp xỉ của L_s thực, và thường được ký hiệu là L_s^\parallel . Thuật toán điều khiển vẫn có khả năng hoạt động hiệu quả nhờ đặc tính triệt tiêu sai số mô hình của các hệ điều khiển hồi tiếp vòng kín.

4.2.4. Cải tiến giải thuật

Tuy mô phỏng cho thấy giải thuật đề cập ở trên hoạt động rất tốt, nhưng thực nghiệm lại chỉ ra điều ngược lại. Lý do là bởi giải thuật trên có tính chất khuếch đại nhiễu. Vì condition number của L_s thường rất lớn, cho nên chỉ cần sai lệch nhỏ ở một thành phần trong e , sẽ dẫn đến sai lệch lớn ở cả 6 thành phần của v_c .

Do đó, thay vì sử dụng như giải thuật đề cập trên, luận văn này đề xuất sử dụng một giải thuật Visual Servoing sử dụng moments ảnh mới, lấy ý tưởng từ giải thuật 2D $\frac{1}{2}$ của tác giả Malis [19]. Thay vì gộp chung các đặc trưng lại và tính theo dạng ma trận, ta sẽ tách các đặc trưng ra và điều khiển các vận tốc tương ứng, cụ thể:

- Dùng đặc trưng x_n để điều khiển thành phần vận tốc v_x của camera:

$$v_x = \lambda_{x_n} (x_n - x_n^*) \quad (4.48)$$

- Dùng đặc trưng y_n để điều khiển thành phần vận tốc v_y của camera:

$$v_y = \lambda_{y_n} (y_n - y_n^*) \quad (4.49)$$

- Dùng đặc trưng a_n để điều khiển thành phần vận tốc v_z của camera:

$$v_z = \lambda_{a_n} (a_n - a_n^*) \quad (4.50)$$

CHƯƠNG 4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

- Dùng đặc trưng θ để điều khiển thành phần vận tốc ω_z của camera:

$$\omega_z = \lambda_\theta (\theta - \theta^*) \quad (4.51)$$

- Dùng đặc trưng s_4 và s_5 để điều khiển thành phần vận tốc ω_x và ω_y của camera:

$$\begin{bmatrix} \omega_x \\ \omega_y \end{bmatrix} = -\lambda_{45} \begin{bmatrix} s_{4\omega x} & s_{4\omega y} \\ s_{5\omega x} & s_{5\omega y} \end{bmatrix}^{-1} \begin{bmatrix} s_4 - s_4^* \\ s_5 - s_5^* \end{bmatrix} \quad (4.52)$$

Bằng cách này, nhiều ở thành phần nào sẽ chỉ ảnh hưởng ở thành phần đó và không làm sai lệch các thành phần khác.

CHƯƠNG 5. XỬ LÝ ẢNH

CHƯƠNG 5. XỬ LÝ ẢNH

Vật thể sử dụng trong thực nghiệm được làm từ mica màu xanh nước biển và có hình dạng con mèo, vị trí thả vật được đánh dấu bằng hình dạng tương tự nhưng có màu xanh lá cây. Vị trí thả được vẽ sẵn vùng sai số có thể chấp nhận được như trong hình 5.1.

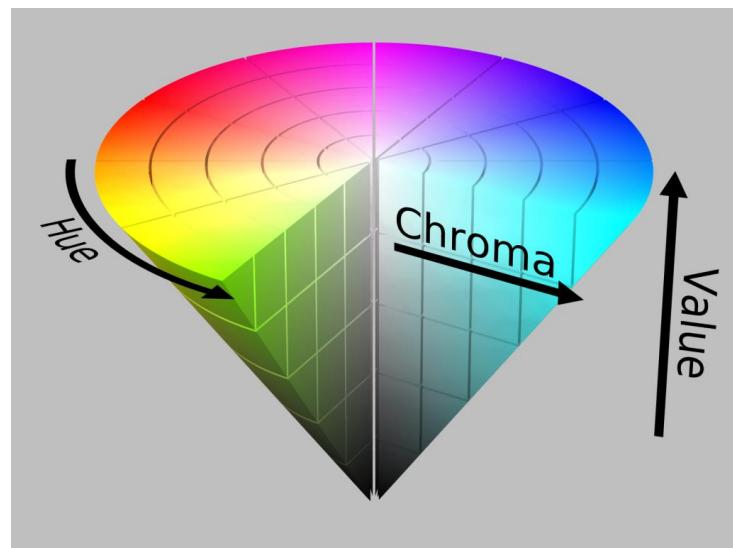


Hình 5.1 Vật thể sử dụng trong thực nghiệm

Để có thể áp dụng giải thuật Visual Servoing sử dụng image moment. Hình ảnh sau khi lấy về từ camera cần phải được xử lý để nhận biết được vị trí của vật thể. Cụ thể hình ảnh thu được từ camera sau khi được xử lý sẽ cho ra một hình dạng binary với vật thể được chuyển thành màu trắng và tất cả các pixel còn lại có màu đen. Để nhận diện vật thể và xử lý để đưa ra hình ảnh đầu vào cho giải thuật, ta sử dụng phương pháp lọc màu HSV.

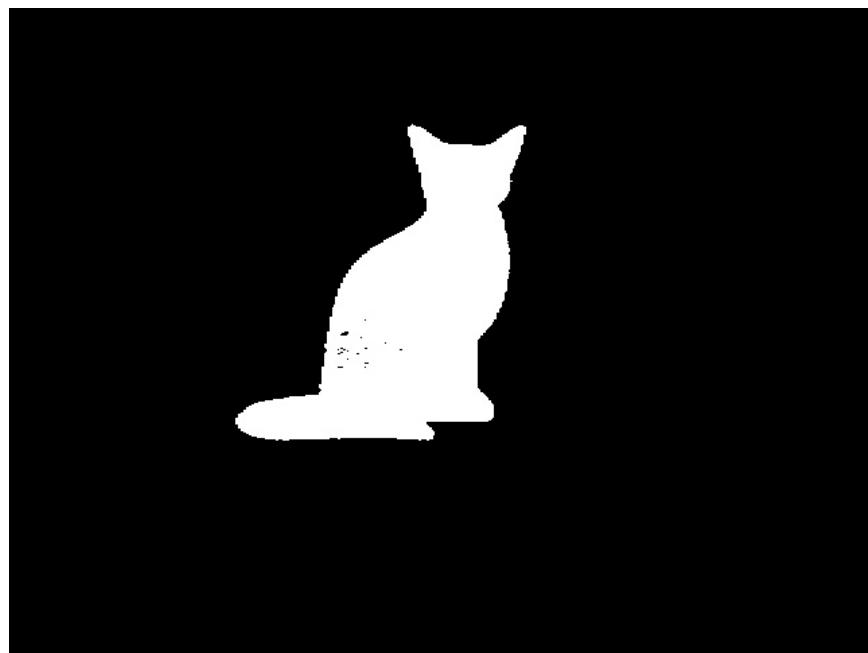
HSV là không gian màu dựa trên 3 thông số chính là vùng màu (Hue), độ bão hòa (Saturation) và giá trị hay độ sáng màu (Value). Trong không gian này, mỗi điểm tương đương với mỗi màu sẽ có giá trị khác nhau. Từ đó, ta có thể dễ dàng tách các màu riêng biệt với nhau.

CHƯƠNG 5. XỬ LÝ ẢNH



Hình 5.2 Không gian màu HSV

Sau khi nhận ảnh màu từ camera, sử dụng các hàm được thư viện OpenCV hỗ trợ để chuyển hình ảnh sang không gian màu HSV (Hue – Saturation – Value). Sau đó, thiết lập một khoảng cho trước để tách các pixel có giá trị nằm trong khoảng. Vì vật thể được sử dụng có màu xanh nước biển nên ta thiết lập khoảng giá trị HSV để tách các pixel màu xanh nước biển.



Hình 5.3 Kết quả sau khi lọc màu HSV

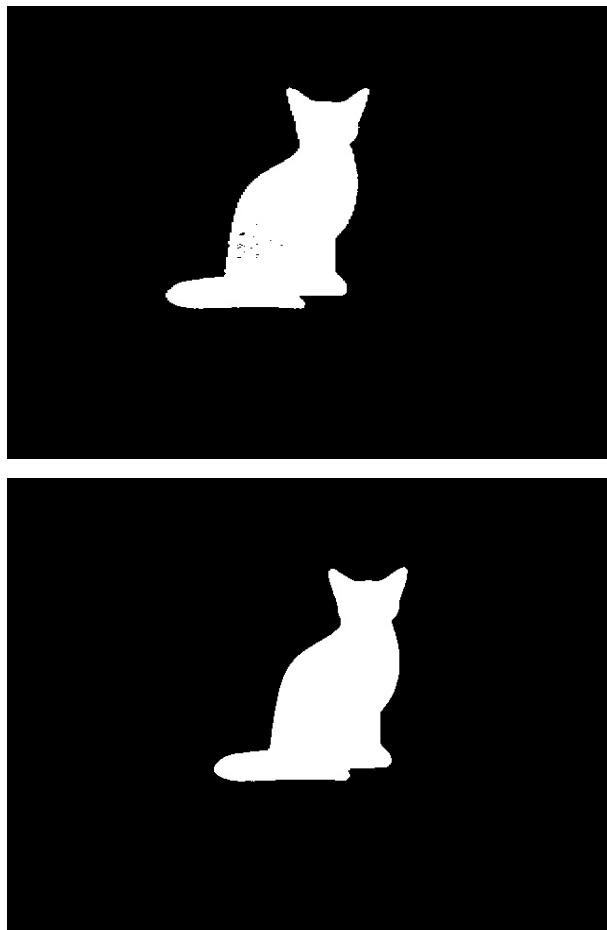
Sau khi lọc ảnh HSV, có thể thấy kết quả cho ra vẫn còn nhiều. Có thể chia nhiều thành 2 loại: các pixel trắng nằm ngoài vật thể và các pixel đen nằm trong vật thể. Để

CHƯƠNG 5. XỬ LÝ ẢNH

loại bỏ các nhiễu dạng này, ta sử dụng thuật toán *opening* và *closing*. *Opening* và *closing* là 2 thuật toán sử dụng kết hợp các thuật toán *erode* và *dilate*. Trong đó:

- *Erode* là thuật toán mở rộng vùng tối (vùng đen) của ảnh.
- *Dilate* là thuật toán mở rộng vùng sáng (vùng trắng) của ảnh.
- *Opening* là thuật toán sử dụng lần lượt *erode* rồi đến *dilate*.
- *Closing* là thuật toán sử dụng lần lượt *dilate* rồi đến *erode*.

Như vậy, nếu sử *opening* sẽ khử được các nhiễu trắng nhỏ nằm ngoài vật thể và sử dụng *closing* sẽ khử được các vùng đen nằm trong vật thể. Sử dụng lần lượt *opening* đến *closing* ta thu được kết quả như hình bên dưới



Hình 5.4 Hình ảnh thu được trước và sau khi lọc nhiễu

Có thể thấy sau khi khử nhiễu thì các pixel trắng và đen riêng lẻ đã biến mất, kết quả thu được là hình ảnh chỉ có duy nhất vật thể.

CHƯƠNG 6. CAMERA CALIBRATION

CHƯƠNG 6. CAMERA CALIBRATION

6.1. Lý thuyết

Trong giải thuật Visual Servoing, hình ảnh được ghi lại bằng camera và xử lý để tìm ra các thông số điều khiển, lúc này các thông số này đang ở trong hệ tọa độ camera, để điều khiển được robot ta cần phải tìm được vị trí của camera trong hệ tọa độ cơ sở của robot tức là cần tìm ${}^0T_{cam}$. Mà:

$${}^0T_{cam} = {}^0T_{tool} \cdot {}^{tool}T_{cam} \quad (6.1)$$

${}^0T_{tool}$ có thể tính được bằng cách giải bài toán động học thuận robot (3.3) đã được trình bày ở chương 3. Do đó mục đích cuối cùng của camera calibration là tìm ma trận chuyển từ hệ tọa độ của camera về đầu công tác ${}^{tool}T_{cam}$. Phương pháp đo thông thường sẽ cho kết quả không chính xác do không biết được vị trí chính xác của thấu kính camera. Thay vào đó, phương pháp tìm thông qua một hệ tọa độ trung gian, thường là của pattern ma trận điểm tròn hoặc mẫu checkerboard cho ra kết quả rất chính xác.

Các bước thực hiện camera calibration như sau:

- Bước 1: Đặt cố định mẫu pattern trong cùng hoạt động của robot.
- Bước 2: Di chuyển đầu công tác của robot lần lượt đến gốc tọa độ P_1 của pattern, một điểm P_2 nằm trên trục x, một điểm P_3 nằm trên trục y nếu ghi nhận tọa độ của các điểm đó trong hệ tọa độ robot. Từ tọa độ của 3 điểm trên có thể tính được các vector đơn vị ${}^0x_{pattern}$, ${}^0y_{pattern}$, ${}^0z_{pattern}$ của hệ tọa độ gắn với mẫu pattern:

$${}^0x_{pattern} = \frac{P_2 - P_1}{|P_2 - P_1|} \quad (6.2)$$

$${}^0y_{pattern} = \frac{P_3 - P_1}{|P_3 - P_1|} \quad (6.3)$$

$${}^0z_{pattern} = {}^0x_{pattern} \times {}^0y_{pattern} \quad (6.4)$$

Từ 3 điểm trên ta suy ra được ma trận chuyển hệ tọa độ từ hệ tọa độ gắn với pattern về hệ tọa độ gắn với robot:

CHƯƠNG 6. CAMERA CALIBRATION

$${}_{pattern}^0T = \begin{bmatrix} {}^0x_{pattern} & {}^0y_{pattern} & {}^0z_{pattern} & P_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.5)$$

- Bước 3: Di chuyển robot đến các vị trí lấy mẫu (khoảng 15 vị trí) sao cho pattern nằm trong tầm nhìn của camera, đọc ảnh camera thu được cùng với tư thế của robot tại từng vị trí. Từ các thông số vị trí của robot (X, Y, Z, roll, pitch, yaw), tính được ma trận chuyển tọa độ từ đầu công tác về tọa độ robot ${}_{tool}^0T_i$ tại từng vị trí.
- Bước 4: Từ các hình ảnh thu được, nhận diện tọa độ pixel của các điểm mẫu trên pattern bằng cách sử dụng hàm *findChessboardCorners* hoặc *findCircleGrid* (tùy thuộc vào mẫu pattern) được hỗ trợ bởi thư viện OpenCV, sau đó áp dụng thuật toán calibrate camera để map tọa độ pixel và tọa độ thực bằng cách sử dụng hàm *CalibrateCamera* của thư viện OpenCV. Sau khi áp dụng kết quả trả về ta sẽ thu được ma trận thông số nội của camera, ma trận xoay R_c , ma trận tịnh tiến t_c của hình mẫu. Từ ma trận xoay và ma trận tịnh tiến ta có thể tìm được ma trận ${}_{pattern}^{cam}T$.

$${}_{pattern}^{cam}T = \begin{bmatrix} R_c & t_c \\ 0 & 1 \end{bmatrix} \quad (6.6)$$

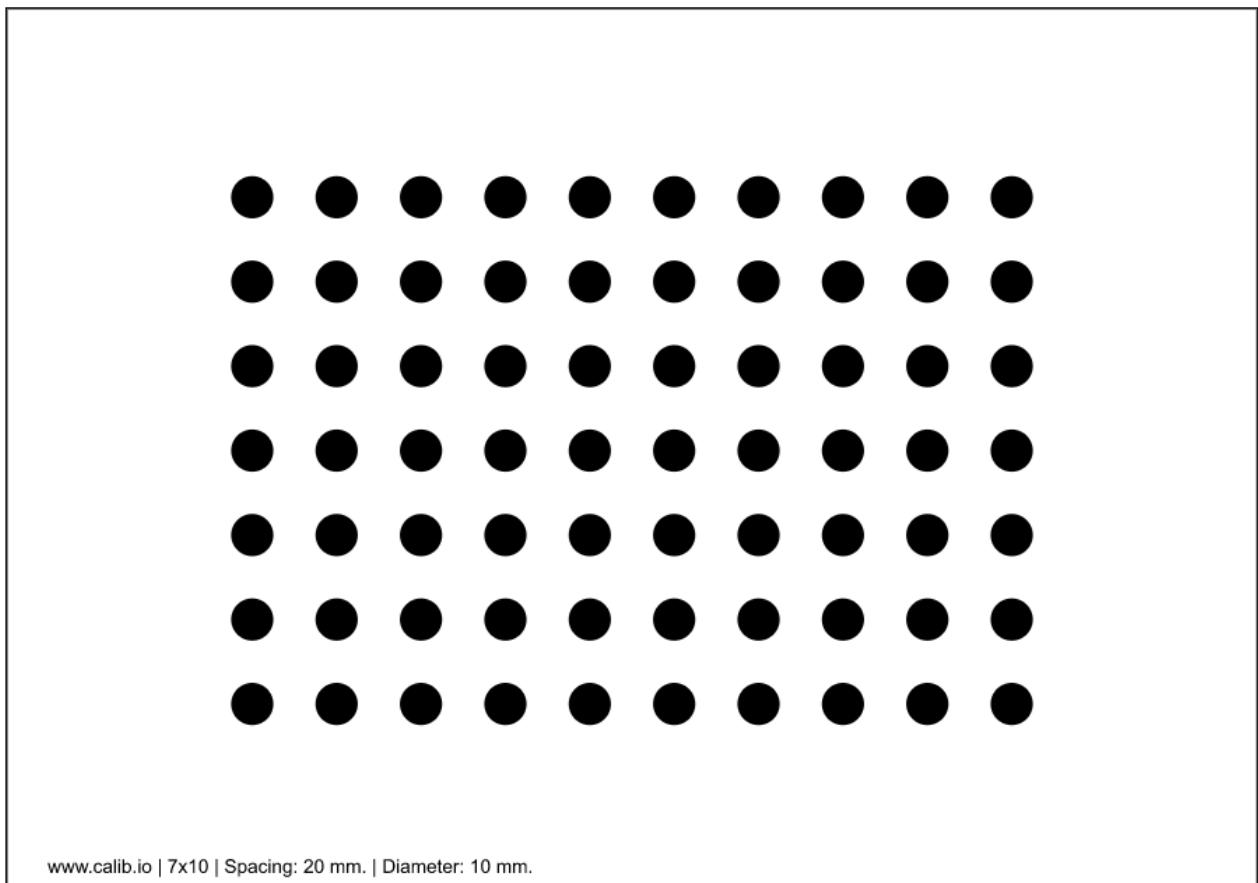
- Bước 5: Tính ma trận chuyển tọa độ camera về đầu công tác:

$$\begin{aligned} {}_{tool}^0T \cdot {}_{cam}^{tool}T \cdot {}_{pattern}^{cam}T &= {}_{pattern}^0T \\ \Rightarrow {}_{cam}^{tool}T &= pinv({}_{tool}^0T) \cdot {}_{pattern}^0T \cdot {}_{pattern}^{cam}T^{-1} \end{aligned} \quad (6.7)$$

CHƯƠNG 6. CAMERA CALIBRATION

6.2. Tiền hành

Đề tài sử dụng pattern là ma trận điểm tròn 10×7 , khoảng cách giữa các điểm tròn là 20mm như hình ...



Hình 6.1 Pattern sử dụng để calib camera

Bảng 6.1 Tọa độ của P1, P2, P3

Điểm	X (mm)	Y (mm)	Z (mm)
P1	522.12	99.52	-2.74
P2	521.54	-80.34	-2.53
P3	400.97	98.44	-2.67

CHƯƠNG 6. CAMERA CALIBRATION

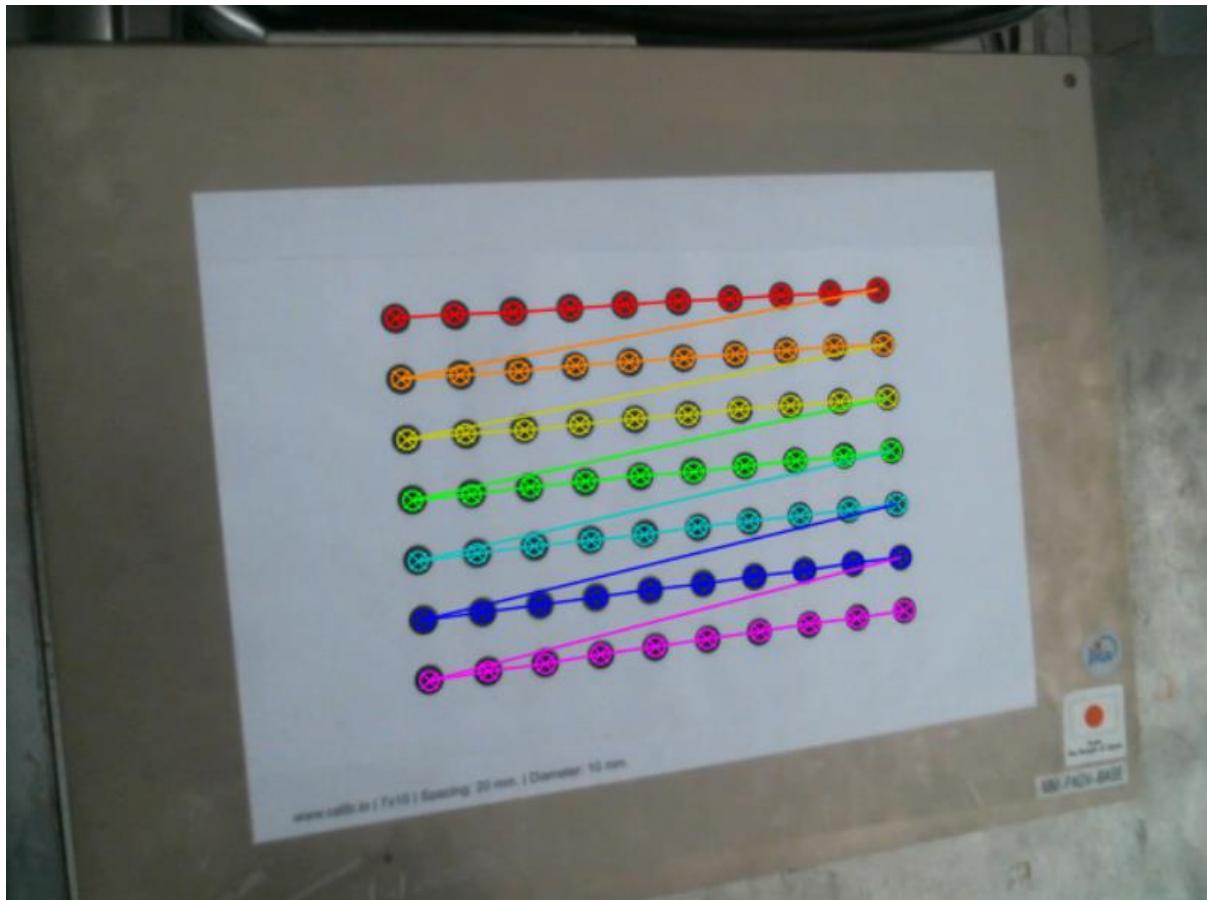
Áp dụng công thức (6.2), (6.3), (6.4), (6.5) ta tính được:

$${}_{pattern}^0T = \begin{bmatrix} 0 & -1 & 0 & 522.12 \\ -1 & -0.01 & 0 & 99.53 \\ 0 & 0 & -1 & -2.74 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Bảng 6.2 Các vị trí lấy ảnh

STT	X (mm)	Y (mm)	Z (mm)	roll (độ)	pitch (độ)	yaw (độ)
1	390.177	0.239412	463.562	0.336493	0.00695086	-180.0
2	369.891	9.0331	418.64	7.52302	1.38578	-176.854
3	396.354	42.5722	389.197	-25.3156	1.18006	179.32
4	386.272	13.0966	332.86	12.23	0.0427438	-179.996
5	393.232	25.5277	333.075	-21.4781	0.0191225	-180.0
6	326.504	35.4743	332.777	-0.650745	-15.2712	178.583
7	371.028	-106.66	283.403	-1.01765	-1.15968	-159.514
8	389.486	90.4222	284.936	-0.211237	-1.4598	166.376
9	356.956	136.965	300.868	-19.5115	-11.0537	166.305
10	445.855	23.5461	354.536	24.8615	9.00519	178.551
11	404.223	19.4721	266.994	-0.351574	1.15994	178.901
12	408.297	37.1654	353.401	23.1424	3.95279	172.335
13	447.62	25.7723	354.695	-22.4068	7.7836	-178.76
14	475.853	-55.6014	321.777	-0.987266	13.1731	-167.082
15	343.012	37.2823	323.496	20.9334	-5.22624	171.785

CHƯƠNG 6. CAMERA CALIBRATION



Hình 6.2 Nhận diện ma trận điểm tròn của 1 trong số các hình mẫu thu được

Tiến hành tính toán tọa độ pixel của ma trận điểm tròn bằng hàm *findCirclesGrid*. Sau đó sử dụng hàm *CalibrateCamera* của thư viện OpenCV để tính thông số nội và thông số ngoại của camera. Thông số ngoại được trả về dưới dạng các vector xoay và vector tịnh tiến được liệt kê dưới **bảng 6.3**. Trong đó vector xoay là 1 cách biểu diễn rất gọn của ma trận xoay, để chuyển từ vector xoay thành ma trận xoay sử dụng hàm *Rodrigues* của thư viện OpenCV.

CHƯƠNG 6. CAMERA CALIBRATION

Bảng 6.3 Vector xoay và vector tịnh tiến của từng ảnh mẫu

STT	Vector xoay	Vector tịnh tiến
1	[0.03, -0.01, -0.02]	[-108.90, -63.95, 488.97]
2	[0.06, 0.04, 0.11]	[-55.97, -105.72, 440.66]
3	[0.06, -0.01, -0.47]	[-116.90, -27.08, 411.33]
4	[0.04, -0.01, 0.19]	[-62.67, -82.86, 355.90]
5	[0.04, -0.01, -0.40]	[-122.18, -21.49, 358.62]
6	[-0.23, -0.03, -0.04]	[-80.88, -32.42, 396.70]
7	[0.03, 0.35, -0.04]	[-104.86, -71.95, 363.53]
8	[0.01, -0.24, -0.03]	[-82.66, -52.32, -37.69]
9	[-0.12, -0.26, -0.39]	[-102.50, -37.69, 340.03]
10	[0.19, -0.07, 0.40]	[-53.67, -87.38, 356.76]
11	[0.06, -0.02, -0.03]	[-91.78, -50.53, 286.66]
12	[0.07, -0.16, 0.38]	[-66.10, -84.57, 363.84]
13	[0.17, 0.05, -0.42]	[-94.93, -19.71, 373.48]
14	[0.27, 0.21, -0.06]	[-91.70, -47.29, 353.90]
15	[-0.07, -0.14, 0.32]	[-49.82, -90.69, 360.18]

Sau khi tìm được các ma trận xoay và vector tịnh tiến dễ dàng xây dựng được các ma trận chuyển tọa độ ${}_{pattern}^{cam}T$. Sau đó áp dụng công thức (4.7) ta tính được ma trận chuyển hệ tọa độ từ camera về đầu công tác:

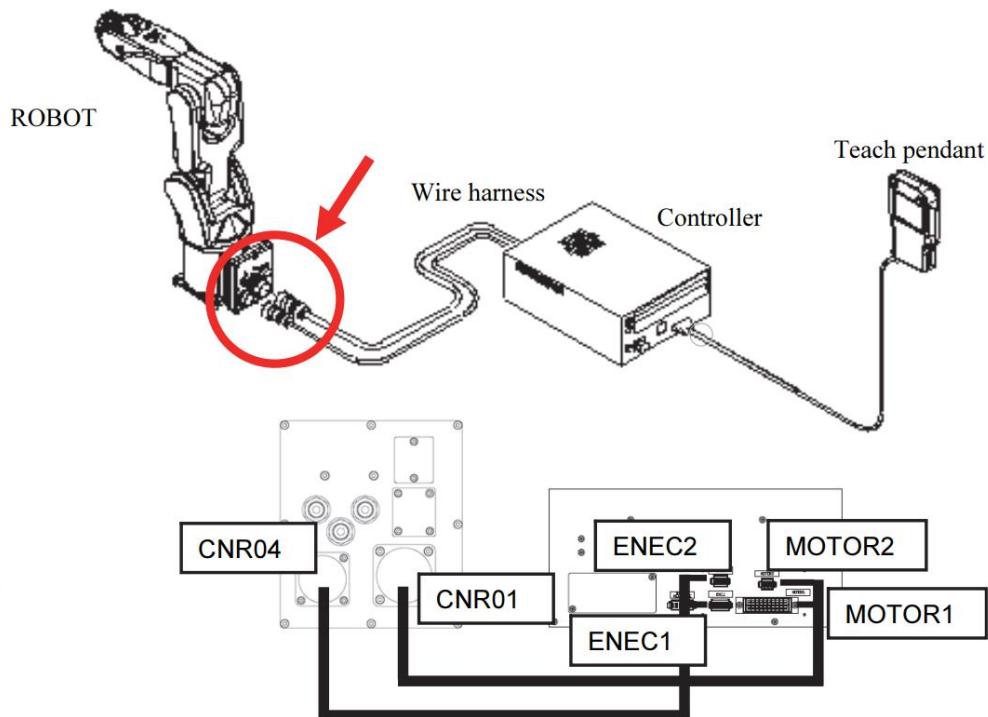
$${}_{cam}^{tool}T = \begin{bmatrix} -0.02 & -1.01 & -0.04 & 82.48 \\ 1. & -0.02 & 0.01 & 2.84 \\ -0.02 & -0.03 & 1. & -25.39 \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH

CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH

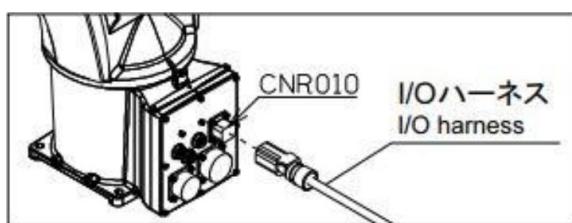
7.1. Hệ thống điện

Đối với dòng robot Nachi MZ07, robot và bộ điều khiển là CFD Controller được kết nối như **hình 7.1** [1]:



Hình 7.1 Sơ đồ kết nối giữa robot và CFD Controller

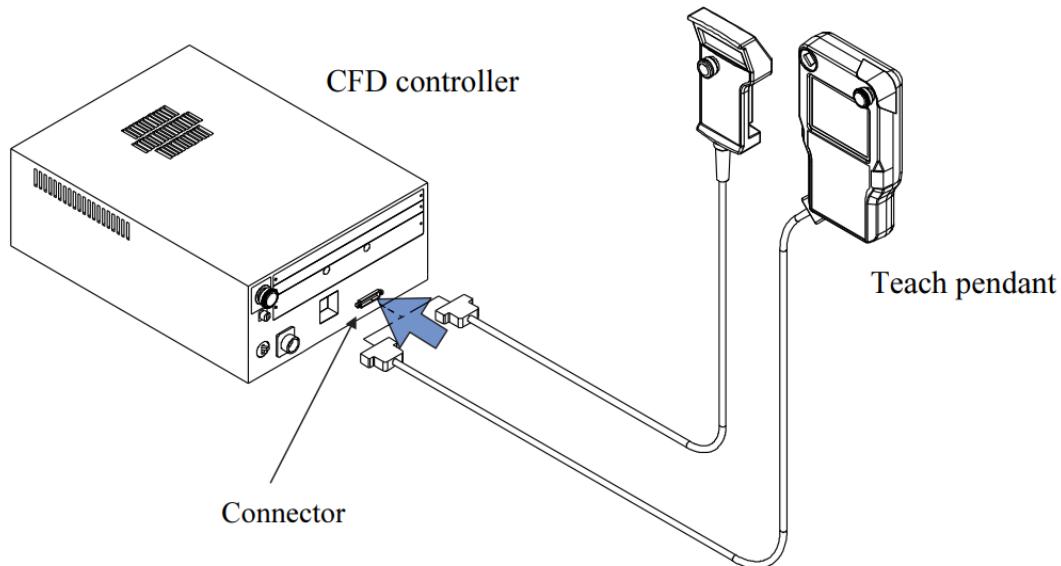
Trong đó, cổng ENC1 và ENC2 trên CFD Controller nối với cổng CNR04 trên robot, cổng MOTOR1 và MOTOR2 trên CFD Controller nối với cổng CNR01 trên robot. Nếu cấu hình robot sử dụng các thiết bị ngoại vi như PLC,... thì các thiết bị đó sẽ kết nối vào cổng CNR010 trên robot.



Hình 7.2 Cổng kết nối robot với các thiết bị ngoại vi

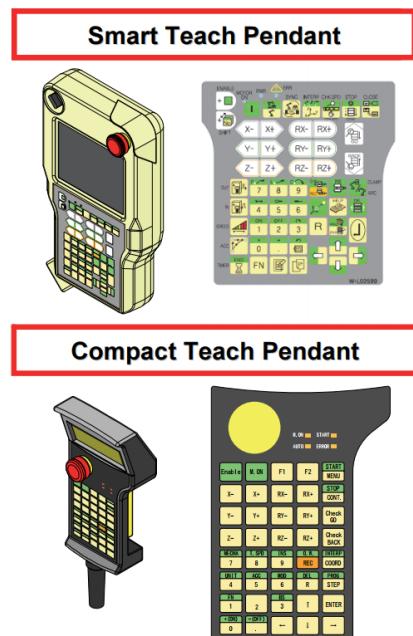
Robot Nachi có hai dòng tay cầm điều khiển là Compact TP và Smart TP. Để điều khiển được robot, kết nối chúng vào cổng TEACH PENDANT trên CFD Controller.

CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH



Hình 7.3 Sơ đồ kết nối Teach pendant với CFD Controller

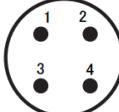
Để tài sử dụng Smart TP với nhiều ưu điểm vượt trội như màn hình màu, kích thước lớn, độ phân giải cao, chức năng chia nhỏ màn hình,...



Hình 7.4 Hai loại Teach pendant

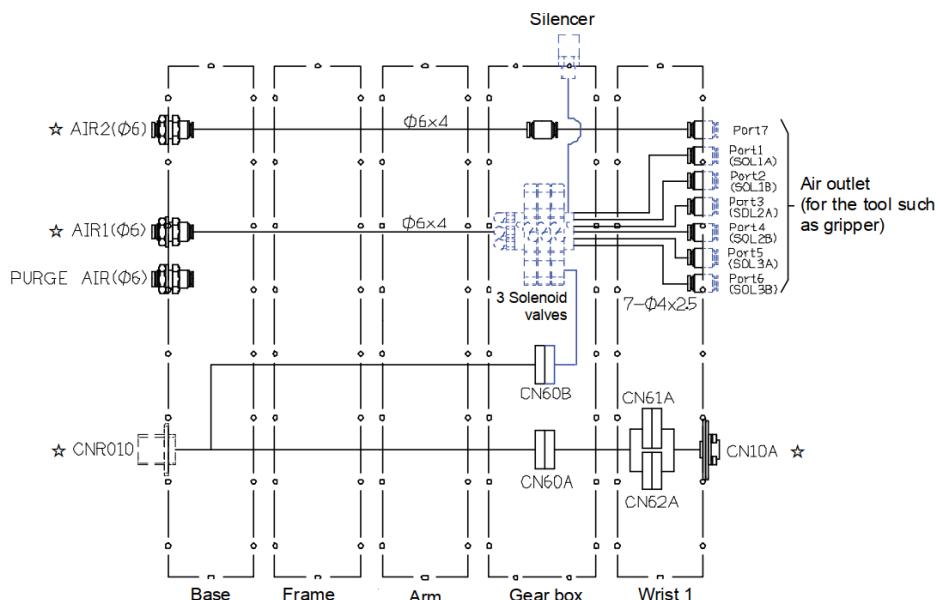
Cổng nguồn của CFD Controller có 4 chân. Đường kính dây cáp có thể sử dụng là từ 10 – 12,5mm, loại kết nối được thiết kế bởi hãng Nanaboshi Electric với số hiệu NJC–204–PF. Để tài sử dụng điện một pha nên nối dây vào chân 1 và 3 của cổng nguồn, chân 4 được nối đất.

CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH

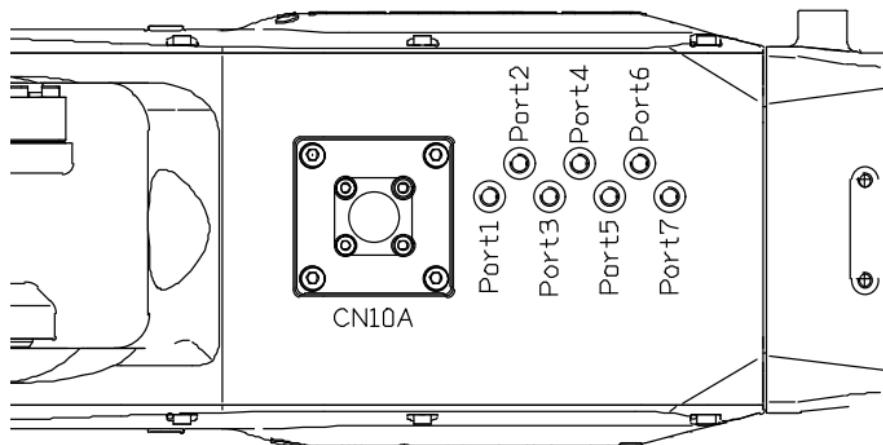
Connector pin layout Seeing from the soldering side	PIN No.	Connection	
		3-phase AC200V	Single-phase AC200V
	1	AC200V R-Phase (RED)	AC200V R-phase (RED)
	2	AC200V S-phase (WHITE)	—
Connector type Nanaboshi Electric Mfg.Co.Ltd NJC-204-PF	3	AC200V T-phase (BLACK)	AC200V T-phase (BLACK)
	4	Ground (Y/G)	Ground (G/Y)

Hình 7.5 Cổng nguồn của CFD Controller

Robot Nachi MZ07 có hỗ trợ hệ thống khí nén tích hợp như biểu đồ hình bên dưới. Trong đó, van solenoid được sử dụng là loại SYJ3220-5GR-M3 của hãng SMC, có thể điều khiển từ cổng CNR010 ở bệ robot. Van solenoid này có thể được kích thông qua User Outputs từ Smart TP thông qua cổng CNR010. Cụ thể, 09 → SOL1A tương ứng cổng 1, 010 → SOL1B tương ứng cổng 2.



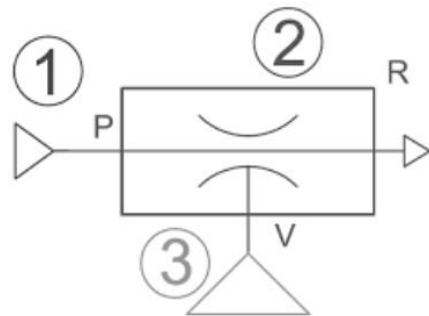
CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH



Hình 7.6 Hệ thống khí nén của robot Nachi MZ07

7.2. Tính toán giác hút

Sơ đồ hoạt động của giác hút:



Hình 7.7 Sơ đồ hoạt động của giác hút

Trong đó, van hút chân không có nguyên lý hoạt động giống như ống Venturi. Cổng P (1) được cung cấp dòng khí nén có áp suất cao, dòng khí này khi di chuyển ngang qua cổng V (3) bị sụt áp, làm cho áp suất tại cổng V thấp hơn áp suất khí quyển, tạo lực hút vật lên. Cổng R(2) có tác dụng giảm tiếng ồn trong quá trình làm việc.

Trước hết, ta cần tính lực hút cần thiết để có thể hút được vật theo công thức sau:

$$F_{TH} = m \left(g + \frac{a}{\mu} \right) s \quad (7.1)$$

Trong đó:

CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH

- m : Khối lượng vật cần gấp, theo đề bài ta lấy bằng 0.1 kg.
- g : gia tốc trọng trường, có giá trị 9.81 kg.m/s^2 .
- a : gia tốc di chuyển, ước tính cao nhất là 5 m/s^2 .
- μ : hệ số ma sát, với bề mặt phẳng không ướt, lấy bằng 0.5.
- s : hệ số an toàn, cân nhắc đến việc gấp vật hơi nghiêng so với mặt đất, chọn giá trị bằng 3.

Như vậy:

$$F_{TH} = 0.1 \times \left(9.81 + \frac{5}{0.5} \right) \times 3 = 5.943 \text{ (N)}$$

Để đảm bảo rằng vật không bị rớt trong quá trình gấp, lực hút của giác hút phải lớn hơn hoặc bằng lực giữ lý thuyết. Công thức tính lực hút như sau:

$$F = \Delta P \times A \quad (7.2)$$

Trong đó:

- ΔP : Độ lớn của áp suất chân không.
- A : Diện tích bề mặt hút của giác hút.

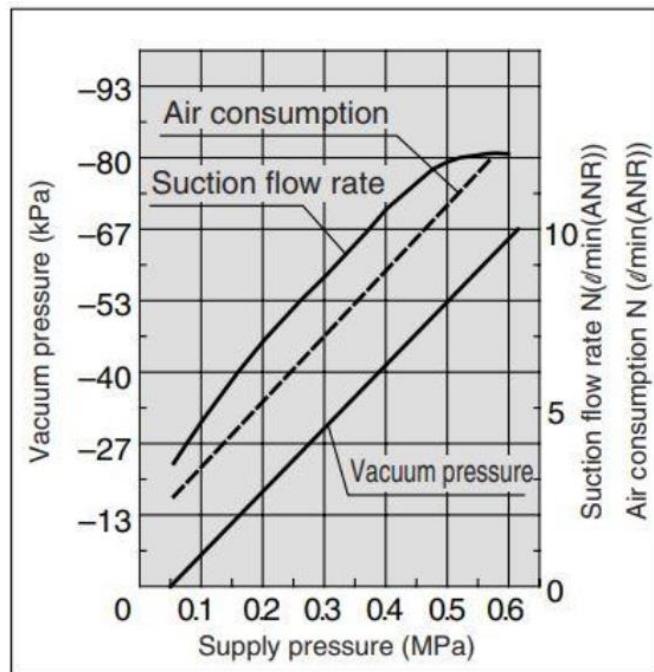
Đề tài sử dụng máy nén khí có sẵn là dòng Hitachi Bebicon, khí được đưa qua bộ lọc khí, có áp suất đầu ra từ 4 đến 6 bar. Dò trên đồ thị, với áp suất ở cổng P (1) từ 4 đến 6 bar, đề tài sử dụng van hút chân không SMC ZU05L. Áp suất chân không tạo ra khi sử dụng van SMC ZU05L là từ -40 đến -67 kPa. Chọn ngưỡng dưới dưới để tính toán, ta có:

$$40 \times 10^3 \times \frac{\pi}{4} \times d^2 \geq 5.943$$

$$\Leftrightarrow \frac{\pi d^2}{4} \geq 1.486 \times 10^{-4}$$

$$\Leftrightarrow d \geq 14$$

CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH



Như vậy, giác hút sử dụng trong đè tài là loại ESG tiêu chuẩn của hãng FESTO có đường kính $d = 20$ mm đủ để giữ vật ổn định.

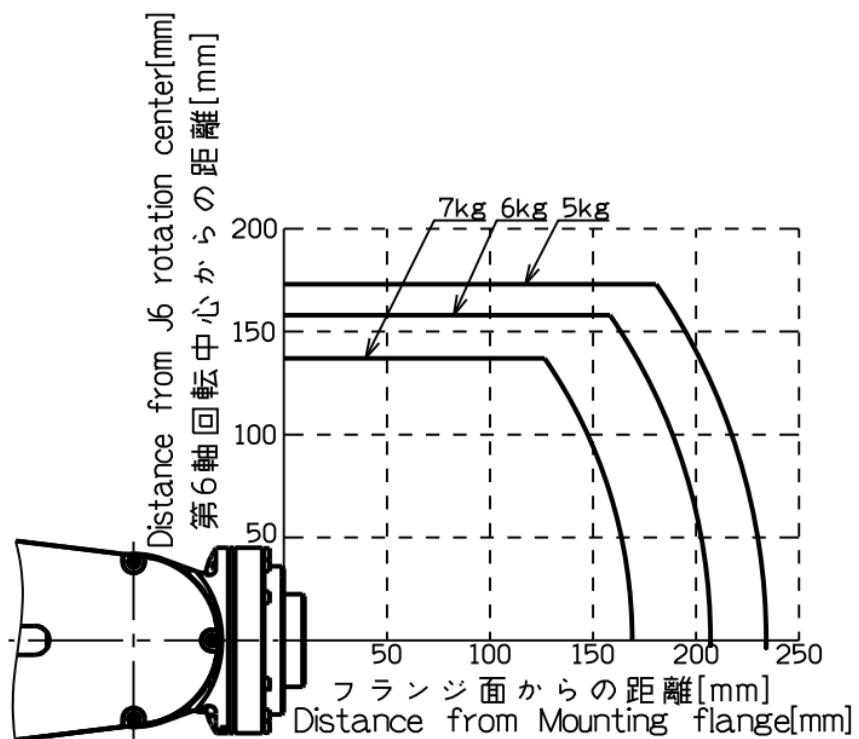


Hình 7.8 Giác hút ESG của hãng FESTO

CHƯƠNG 7. HỆ THỐNG ĐIỆN VÀ CƠ CẤU CHẤP HÀNH

7.3. Tính toán đồ gá

Khâu 6 của tay máy Nachi MZ07 được thiết kế với mặt bích có 4 bu long M5 và 2 lỗ định vị. Đồ gá trung gian và đồ gá được thiết kế có tổng chiều dài là 144 mm và khối lượng rơi vào khoảng 2kg. Camera có khối lượng 138g được gắn vào đồ gá được in 3D. Trọng tâm camera nằm cách trục Z của khâu 6 ít hơn 100 mm. Tính thêm cả giác hút và vật gấp thì tổng chiều dài từ mặt bích trở ra là nhỏ hơn 200 mm, khối lượng tổng cộng chưa tới 3kg.



Hình 7.9 Biểu đồ tính toán đồ gá đầu tool

Dựa vào biểu đồ có thể thấy trọng tâm của phần tử mặt bích trở ra nằm hoàn toàn trong miền 6kg. Do đó có thể kết luận tải trọng này là an toàn khi vận hành robot.

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT

8.1. Giao tiếp với robot Nachi MZ07

Đối với robot Nachi MZ07, nhà sản xuất hỗ trợ hai phương thức giao tiếp, một là thư viện OpenNR – IF, hai là phương thức giao tiếp socket.

OpenNR – IF là một thư viện được hãng Nachi thiết kế dành riêng cho việc điều khiển robot bằng máy tính, với mục đích dễ dàng tạo ra các ứng dụng để kết nối với CFD Controller và các thiết bị ngoại vi khác. Thư viện có khả năng thiết kế các hoạt động cho cánh tay robot, bao gồm ở cả chế độ Playback, giám sát hoạt động, trạng thái làm việc cũng như bảng tín hiệu I/O của robot.

Tuy nhiên để sử dụng ứng dụng, cần phải có được bản quyền sử dụng của thư viện và chỉ có một máy tính có địa chỉ MAC được cấp phép mới có quyền sử dụng toàn bộ chức năng của thư viện.

Do giới hạn về vấn đề bản quyền, nên đề tài chọn sử dụng giao tiếp với robot thông qua phương thức giao tiếp socket.

8.2. Phương thức giao tiếp socket trên robot Nachi MZ07.

Bộ điều khiển của robot là CFD Controller có hỗ trợ cổng giao tiếp ethernet cho phép giao tiếp Socket bằng các lệnh ứng dụng khác nhau từ chương trình bên ngoài để trao đổi nhiều loại dữ liệu như tọa độ vị trí, hướng,...

Socket là một giao diện lập trình ứng dụng (API – Application Programming Interface). Nó được giới thiệu lần đầu tiên trong phiên bản UNIX – BSD 4.2 dưới dạng các hàm hệ thống cú pháp ngôn ngữ C(socket(), bind(), connect(), send(), receive(), read(), write(), close(),...). Ngày nay, Socket được hỗ trợ trong hầu hết các hệ điều hành như MS Windows, Linux,... và được sử dụng trong nhiều ngôn ngữ lập trình khác nhau như: C, C++, Java, Visual Basic, Visual C++,...

Socket cho phép thiết lập các kênh giao tiếp mà hai đầu kênh được kết nối bởi 2 cổng (port). Thông qua các cổng này một quá trình có thể nhận và gửi dữ liệu với các quá trình khác.

Có 2 giao thức thường sử dụng nhất trong socket là TCP và UDP.

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT

- TCP (Transmission Control Protocol): là một bộ các công được thiết bị sử dụng để giao tiếp qua internet và hầu hết các mạng cục bộ. TCP cung cấp cho các ứng dụng cách để truyền (và nhận) một luồng gói thông tin đã được đặt hàng và kiểm tra lỗi qua mạng. Giao thức TCP có độ tin cậy cao, các gói tin được gửi bawngf TCP sẽ được theo dõi do vậy dữ liệu sẽ không bị mất hoặc hỏng trong quá trình truyền nhận. Đó là lí do tại sao tệp tin tải xuống không bị hỏng ngay cả khi mạng có vấn đề. Tất nhiên, nếu bên nhận hoàn toàn ngoại tuyến, máy tính sẽ từ bỏ một thông báo lỗi ghi không thể giao tiếp với máy chủ lưu trữ từ xa.

- UDP (User Datagram Protocol): được sử dụng để truyền nhận một luồng dữ liệu nhanh hơn bằng cách bỏ qua kiểm tra lỗi. UDP được sử dụng khi yêu cầu cần tốc độ truyền nhận nhanh và không cần thiết sửa lỗi. Ví dụ, UDP thường được sử dụng cho các chương trình phát sóng trực tiếp và game online.

Vì để dữ liệu được truyền chính xác, an toàn và không cần tốc độ cao trong mỗi lần chuyển nên chọn giao thức socket TCP.

Có 4 giai đoạn giao tiếp TCP:

- Giai đoạn 1: Server tạo socket, gắn số hiệu cổng và lắng nghe yêu cầu kết nối. Server sẵn sàng phục vụ client. Giai đoạn gồm các hàm khởi tạo sau:

- + socket(): Server yêu cầu tạo một socket để có thể sử dụng dịch vụ của tầng dịch chuyển.
- + bind(): Server yêu cầu gán số hiệu cổng cho socket.
- + listen(): Server lắng nghe các yêu cầu kết nối từ các client trên cổng đã được gán.

- Giai đoạn 2: Client tạo socket, yêu cầu thiết lập một kết nối với server. Giai đoạn gồm các hàm:

- + connect(): Client gửi yêu cầu kết nối đến server có địa chỉ IP và Port xác định.
- + accept(): Server chấp nhận kết nối của client, khi đó một kênh giao tiếp ảo được hình thành, Client và server có thể trao đổi thông tin với nhau thông qua kênh ảo.

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT

- + socket(): Client yêu cầu tạo một socket để có thể sử dụng các dịch vụ của tầng dịch chuyển, thông thường hệ thống tự động gán một số hiệu rảnh cho socket của client.
- Giai đoạn 3: Trao đổi thông tin giữa server và client. Giai đoạn gồm các hàm:
 - + read(): Sau khi chấp nhận yêu cầu kết nối, server thực hiện lệnh và nghẽn cho đến khi có thông điệp yêu cầu (Request Message) từ client gửi đến.
 - + write(): Dùng để gửi dữ liệu xuống client. Client chờ nhận thông điệp kết quả từ server thông qua lệnh read().
- Giai đoạn 4: Kết thúc. Cổng kết nối sẽ bị xoá khi dùng hàm close().

8.3. Điều khiển robot Nachi MZ07

Trong CFD Controller, tồn tại hai loại chương trình, một là chương trình điều khiển hoạt động của robot gọi là Robot Language Program (RLP), hai là chương trình giám sát và truyền nhận dữ liệu gọi là USERTASK [3].

Robot Language Program (RLP): Chương trình/hàm người dùng định nghĩa cho CFD controller biên dịch và chạy dựa trên các câu lệnh cung cấp sẵn như di chuyển robot, kích I/O, kích valve khí nén có sẵn trong robot, chạy internal PLC,... Do yêu cầu về giám sát và an toàn, RLP chỉ cho phép chạy tuần tự và chỉ chạy được duy nhất một RLP tại một thời điểm (việc gọi một RLP từ một RLP khác vẫn được xem như là việc chạy tuần tự một chương trình vì luồng điều khiển chỉ theo một hướng) cho đến khi RLP đó kết thúc mới cho phép chạy RLP khác.

USERTASK: Chương trình cho CFD controller biên dịch và chạy dựa trên các câu lệnh cung cấp sẵn (trừ lệnh di chuyển robot, I/O, internal PLC). USERTASK được viết bằng robot language sau đó biên dịch sang dạng thực thi. Chương trình chạy độc lập hoặc song song với RLP, có thể chạy tối đa 4 User task cùng một thời điểm, nếu có xung đột sẽ ưu tiên User task có Priority cao hơn. Chức năng chính của các USERTASK dùng để tạo giao diện giám sát và xử lý lỗi trên teach pendant. Các câu lệnh khởi tạo Socket Communication và truyền nhận dữ liệu chỉ có thể được gọi từ các USERTASK.

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT

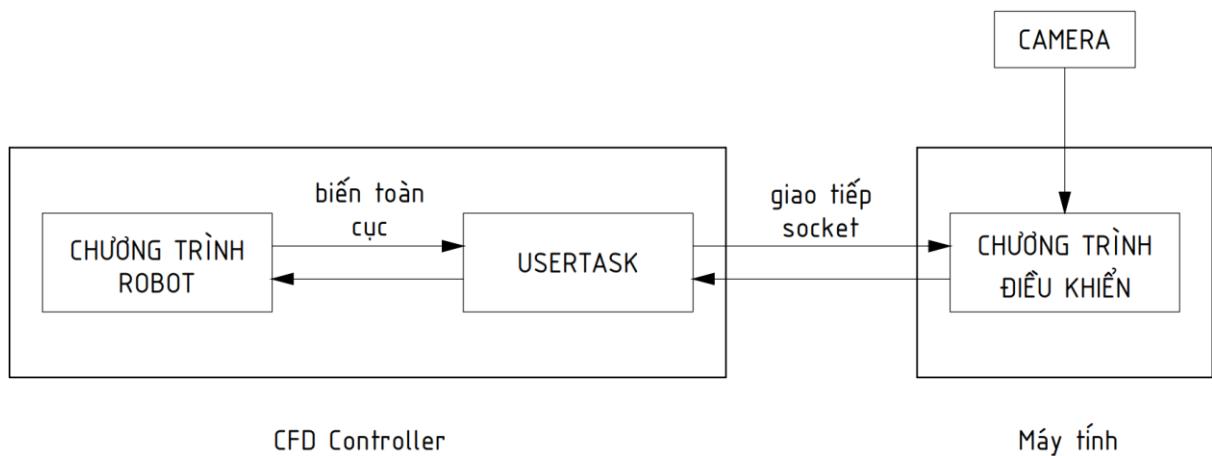
Một số câu lệnh liên quan đến phương thức giao tiếp socket được sử dụng. Cấu trúc các lệnh này được viết trong robot language [3].

Tên hàm	Chức năng	Thông số
SOCKCREATE P1,P2	Tạo socket	P1: Socket number (1 đến 16) P2: Giao thức (0: TCP, 1: UDP)
SOCKCLOSE P1	Đóng socket	P1: Socket number
SOCKBIND P1,P2	Tạo cổng chờ kết nối	P1: Socket number P2: Port number (1 đến 65535)
SOCKWAIT P1,P2,P3	Chờ kết nối	P1: Socket number P2: Socket wait number (1 đến 16) P3: Time period (0 – 20 giây)
SOCKCONNECT P1,P2,P3,P4	Kết nối đến server socket	P1:Socket number P2: IP address (1 – 254) P3: Port number (1 – 65535) P4: Timer period
SOCKSENDSTR P1,P2,P3,P4,P5,P6	Truyền data dạng chuỗi	P1: Socket number P2: String data P3: Data length P4: Timeout period P5: Save variables P6: String terminator
SOCKRECV P1,P2,P3,P4,P5	Chờ nhận data	P1: Socket number

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT

		P2: Buffer number
		P3: Data length
		P4: Timeout period
		P5: Data size

Chương trình Robot và USERTASK sử dụng chung các biến toàn cục, nêu các biến này có thể dung để giao tiếp giữa 2 loại chương trình trên.



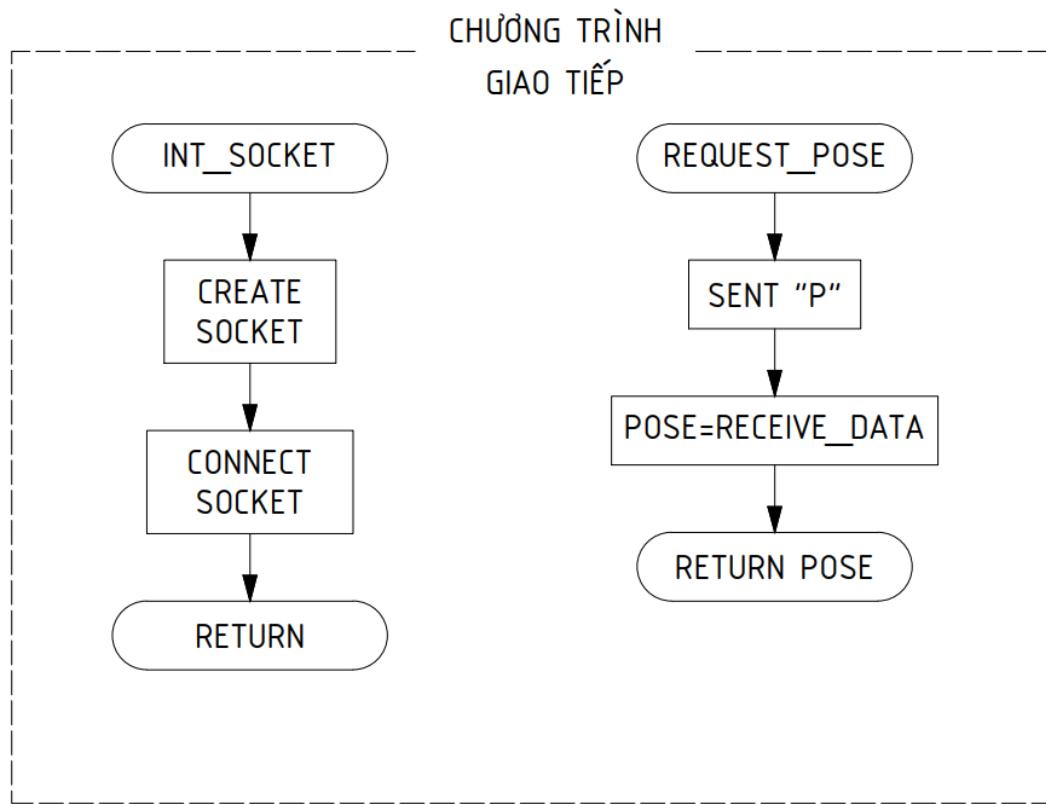
Hình 8.1 Mô hình giao tiếp điều khiển robot

Hình 8.1 thể hiện sơ đồ giao tiếp và điều khiển robot, trong đó cần tổng cộng 3 chương trình, 1 trên máy tính và 2 trên CFD Controller.

Luận văn sử dụng Robot mà cụ thể là CFD Controller dưới vai trò server, khởi tạo socket và chờ kết nối từ client là máy tính.

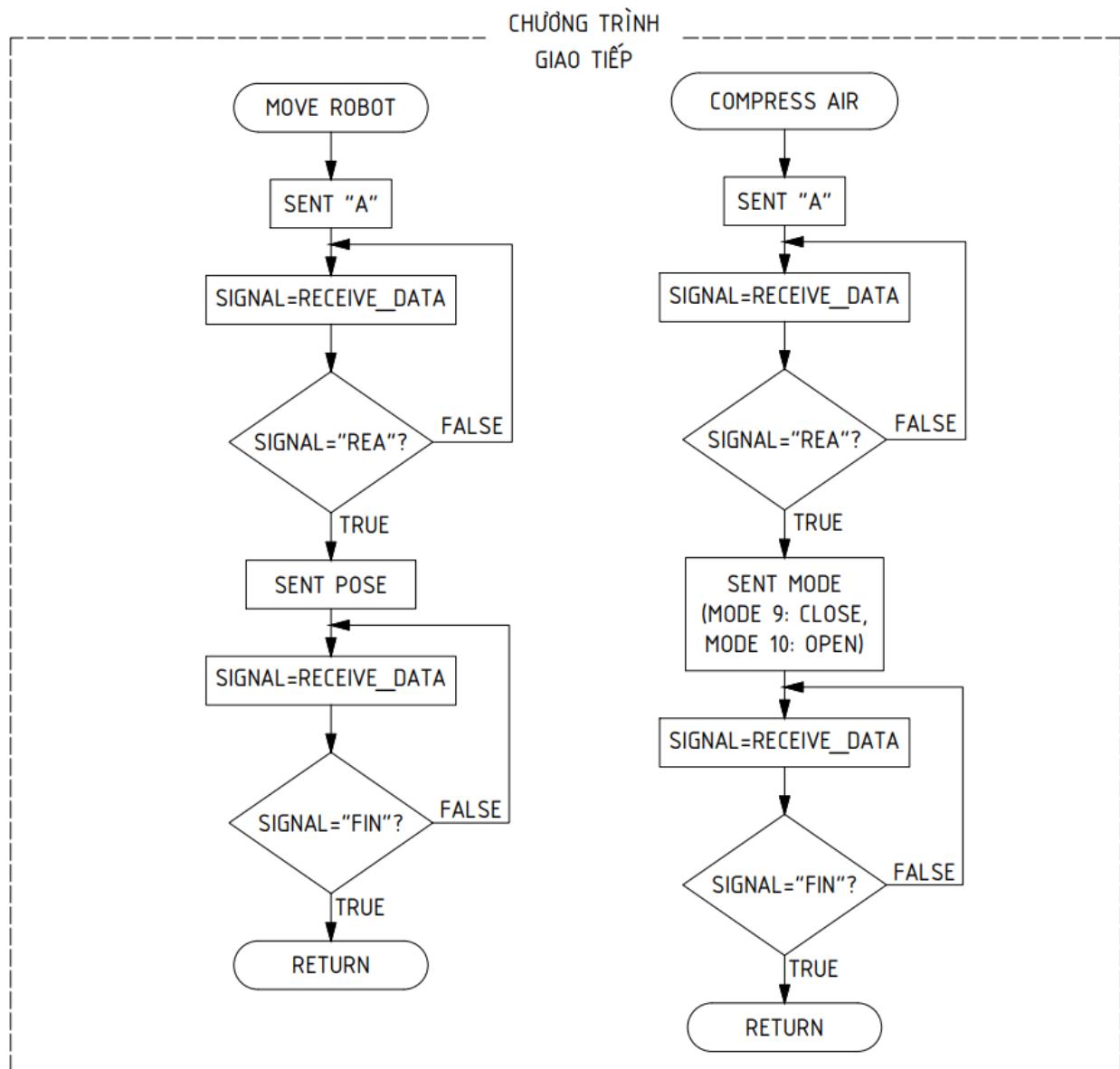
Chương trình trên máy tính có nhiệm vụ xử lý hình ảnh thu được từ camera, khởi tạo socket với vai trò là client và kết nối với server là CFD Controller, gửi dữ liệu đến cho CFD Controller hoặc nhận dữ liệu từ CFD Controller để xử lý. USERTASK dùng để khởi tạo socket trên CFD Controller với vai trò là server và chờ kết nối từ client là máy tính, nhận và gửi dữ liệu về máy tính. Chương trình robot chịu trách nhiệm di chuyển robot và điều khiển giác hút.

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT



Hình 8.2 Chương trình con khởi tạo socket và nhận vị trí của robot

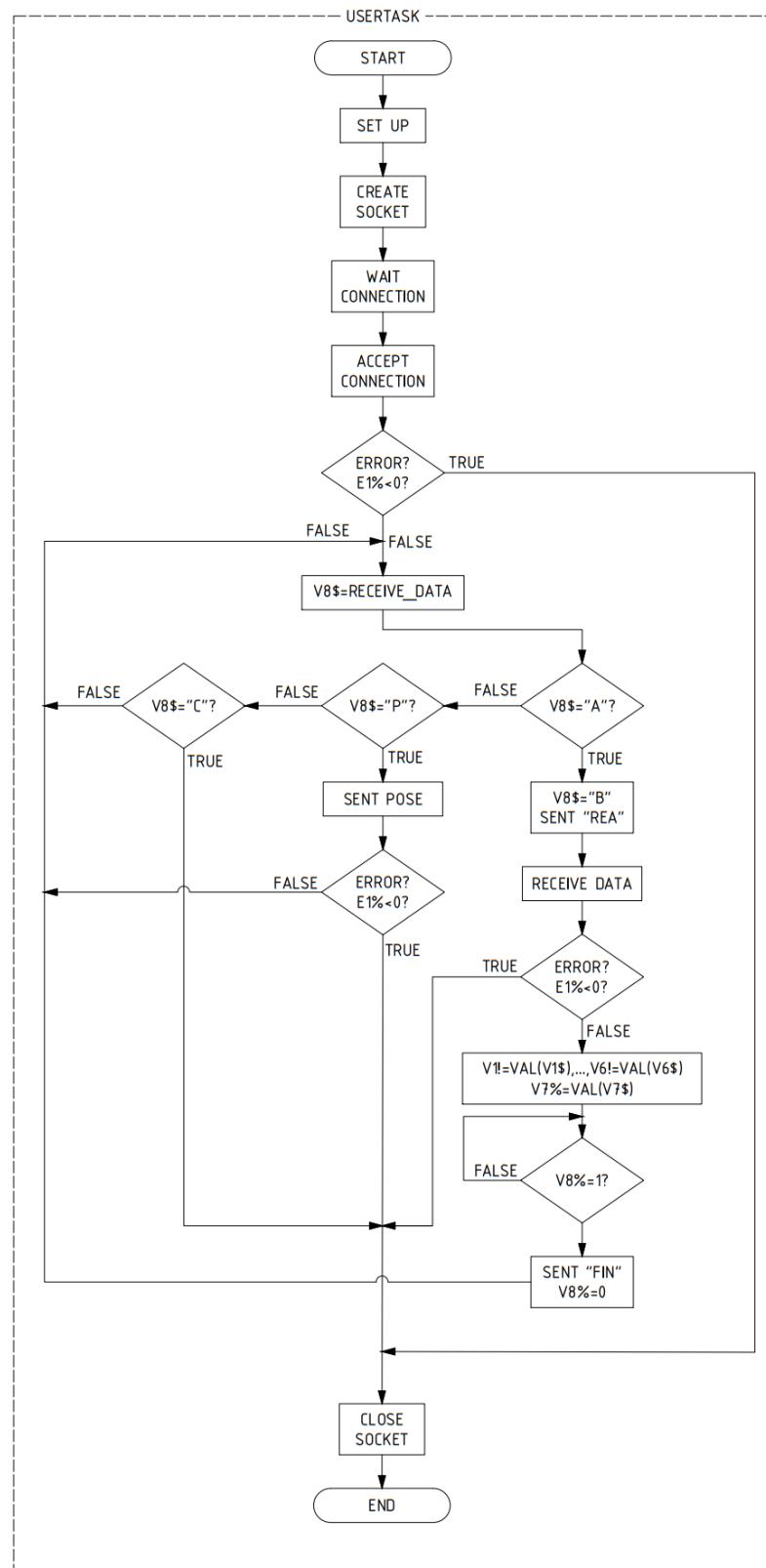
CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT



Hình 8.3 Chương trình con ra lệnh di chuyển robot và điều khiển giác hút

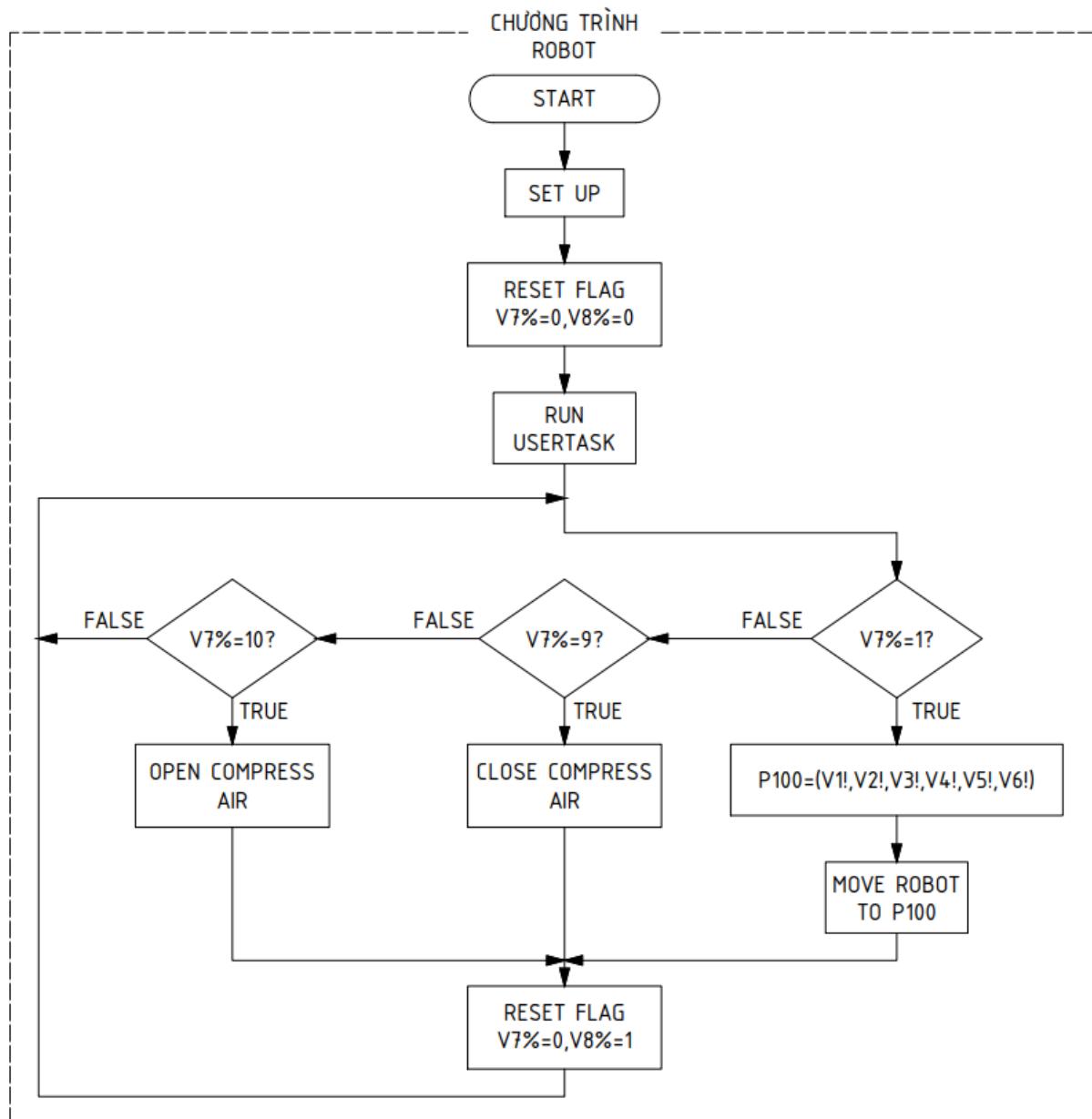
Các chương trình con trên máy tính để khởi tạo socket, gửi và nhận vị trí robot, ra lệnh di chuyển robot, điều khiển van khí của giác hút được thể hiện ở **hình 8.2** và **hình 8.3**. Lưu đồ giải thuật cụ thể của USERTASK và chương trình robot được biểu diễn ở **hình 8.4** và **hình 8.5**.

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT



Hình 8.4 Lưu đồ giải thuật của chương trình USERTASK

CHƯƠNG 8. GIAO TIẾP VÀ ĐIỀU KHIỂN ROBOT



Hình 8.5 Lưu đồ giải thuật của chương trình robot

CHƯƠNG 9. MÔ PHỎNG

CHƯƠNG 9. MÔ PHỎNG

Để thực hiện mô phỏng giải thuật để tài sử dụng phần mềm MATLAB, đồng thời sử dụng hai toolbox là Machine Vision Toolbox và Robotics Toolbox của tác giả Peter Corke. Bên cạnh đó sử dụng phần mềm đồ họa Adobe Illustrator và ngôn ngữ lập trình Python để tạo ra dữ liệu về vật thể. Mục tiêu của của mô phỏng là dựa vào hình ảnh thu được từ camera để điều khiển cánh tay robot sao cho vị trí của camera tiến đến vị trí mong muốn định trước.

9.1. Các bước mô phỏng

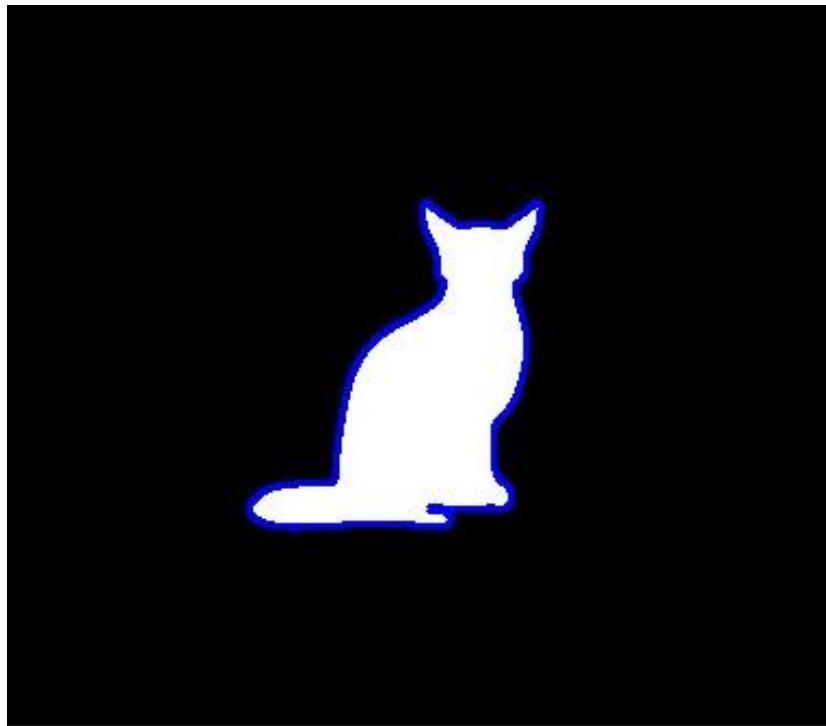
Để tạo dữ liệu về vật thể ta sử dụng phần mềm Adobe Illustrator để ghép định dạng vector của vật thể mà ta muốn vào một khung nền để tạo ra một hình với nền màu đen và vật thể có màu trắng như **hình 9.1** bên dưới.



Hình 9.1 Hình ảnh dùng để tạo dữ liệu vật thể

CHƯƠNG 9. MÔ PHỎNG

Sau khi đã có được hình như **hình 9.1**, sử dụng ngôn ngữ lập trình Python và hàm *findContours* được hỗ trợ bởi thư viện thư viện OpenCV để tìm đường bao (contour) của vật thể như **hình 9.2**.



Hình 9.2 Hình thu được sau khi tìm đường bao của vật thể

Sau khi có được dữ liệu về đường bao của vật thể lưu dữ liệu đường bao đó thành một chuỗi các điểm hai chiều và lưu vào file có định dạng “.xlsx”. Sau đó dùng MATLAB để chuyển file đó thành file có định dạng “.mat” của MATLAB.

Vị trí mong muốn định trước là camera cách mặt phẳng vật thể 0.5m và trục z của camera vuông góc với mặt phẳng vật thể.

Quá trình mô phỏng được thực hiện theo các bước sau:

- Bước 1: Load dữ liệu vật thể

- Bước 2: Sử dụng toolbox Robotics Toolbox để xây dựng đối tượng robot có thông số DH lấy từ bảng 3.2. Đồng thời camera được xem như đầu công tác để thuận tiện cho việc tính Jacobian nên ma trận chuyển vị của đầu công tác của đối tượng robot được gán bằng $T_{tool} = {}^6T_{cam}^{tool}$. Lưu ý rằng ma trận chuyển vị phải chuyển sang đơn vị mét.

Robot để ở tư thế ban đầu $\theta = \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{\pi}{2} & 0 \end{bmatrix}$.

CHƯƠNG 9. MÔ PHỎNG

- Bước 3: Sử dụng toolbox Machine Vision Toolbox để tạo một đối tượng camera với thông số mặc định.

- Bước 4: Cho camera nhìn vật thể ở vị trí mong muốn, dung phương pháp *project* của đối tượng camera để tính hình ảnh của vật thể qua camera. Từ hình ảnh thu được, tính đặc trưng mong muốn s^* và các thông số $s_{4\omega_x}, s_{4\omega_y}, s_{5\omega_x}, s_{5\omega_y}$ như đề cập ở chương 6.

- Bước 5: Đặt vật thể ở vị trí bắt đầu mô phỏng, vị trí này có ma trận chuyển vị so với camera là:

$$T_{init} = t_x(0.1) \cdot t_y(-0.2) \cdot t_z(0.5) \cdot R_y\left(\frac{\pi}{10}\right) \cdot R_x\left(\frac{-\pi}{12}\right) \cdot R_z\left(\frac{-\pi}{10}\right)$$

- Bước 6: Tính đặc trưng s .

- Bước 7: Tính sai số $e = s - s^*$. Kiểm tra điều kiện $|e| \leq e_{min}$, nếu thỏa thì dừng mô phỏng.

- Bước 8: Tính vận tốc camera theo (4.48), (4.49), (4.50), (4.51), (4.52) và giới hạn trên dưới cho các giá trị vận tốc.

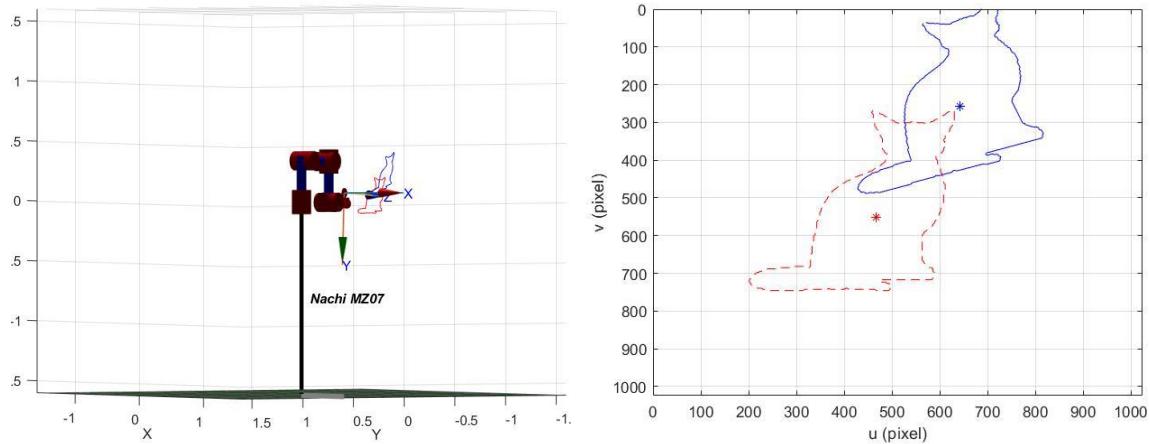
- Bước 9: Tính vận tốc các khớp theo công thức (4.28)

- Bước 10: Di chuyển robot đến vị trí mới: $\theta = \theta + d\theta$. Quay lại bước 6.

CHƯƠNG 9. MÔ PHỎNG

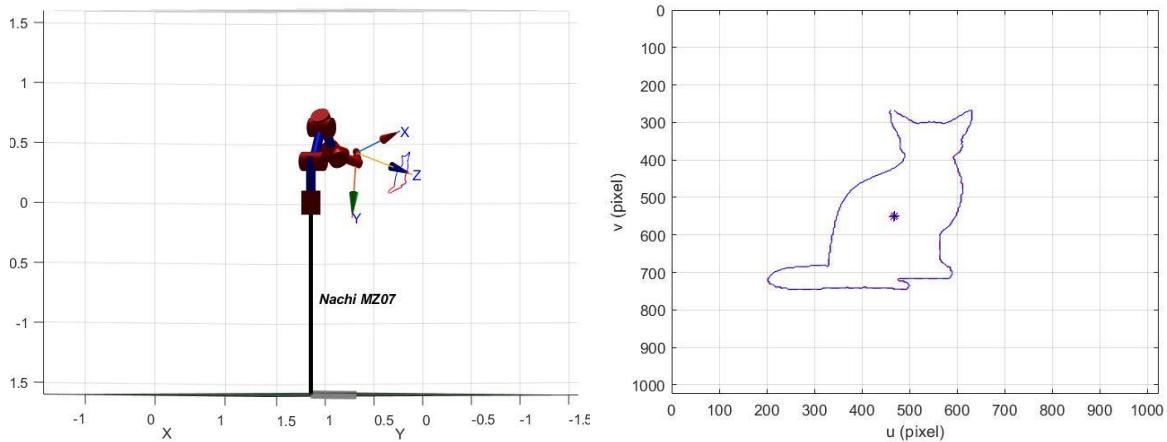
9.2 Kết quả mô phỏng

Quá trình mô phỏng chỉ mất 54 bước để hội tụ.



Hình 9.3 Vị trí ban đầu của mô phỏng

Hình 9.3 thể hiện vị trí ban đầu của cánh tay robot, camera và vật thể. Trong đó, đường màu đỏ tượng trưng cho vị trí mong muốn của vật thể so với camera, đường màu xanh thể hiện vị trí thực của vật thể so với camera.



Hình 9.4 Kết quả của mô phỏng

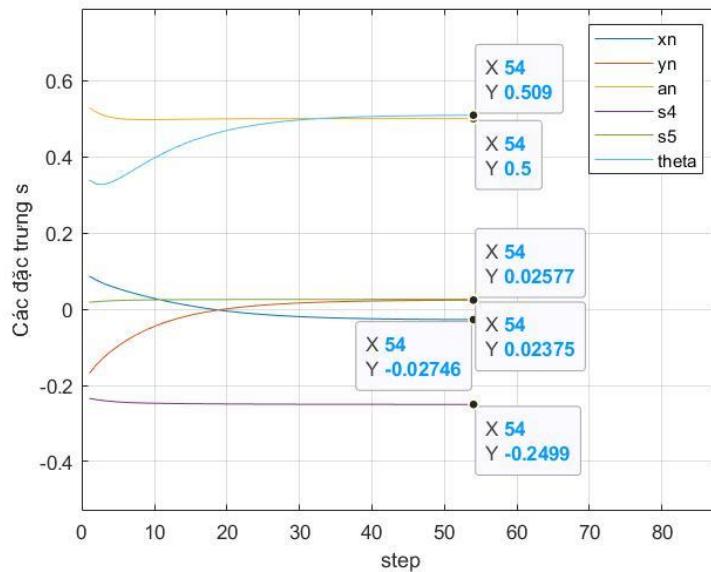
Quá trình mô phỏng mất 54 bước. **Hình 9.4** thể hiện vị trí của robot và vật thể khi kết thúc mô phỏng, có thể thấy camera đã tiến đến vị trí mong muốn, hình ảnh vật thể nhìn qua camera cũng khớp với hình ảnh mong muốn.

Giá trị của các đặc trưng mẫu mà ta mong muốn s^* được thể hiện ở **bảng 9.1**.

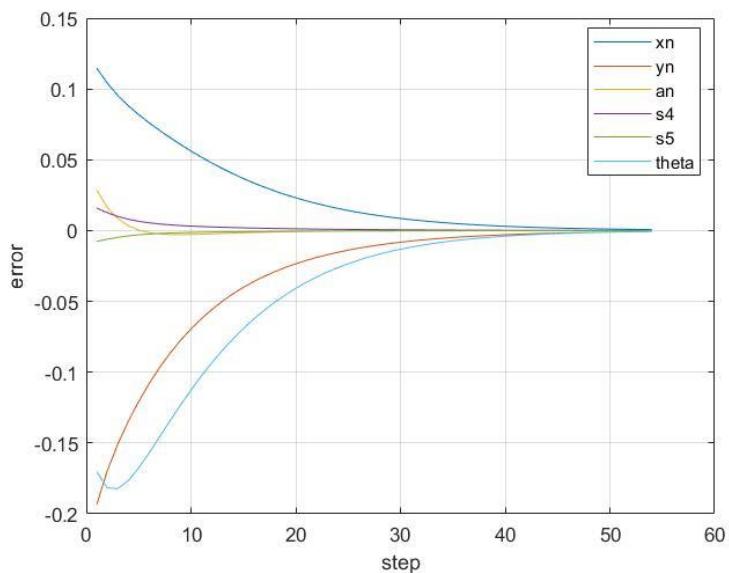
CHƯƠNG 9. MÔ PHỎNG

Bảng 9.1 Giá trị các đặc trưng mẫu s^*

x_n	y_n	a_n	s_4	s_5	θ
-0.0282	0.0244	0.5	-0.25	0.0258	0.5096

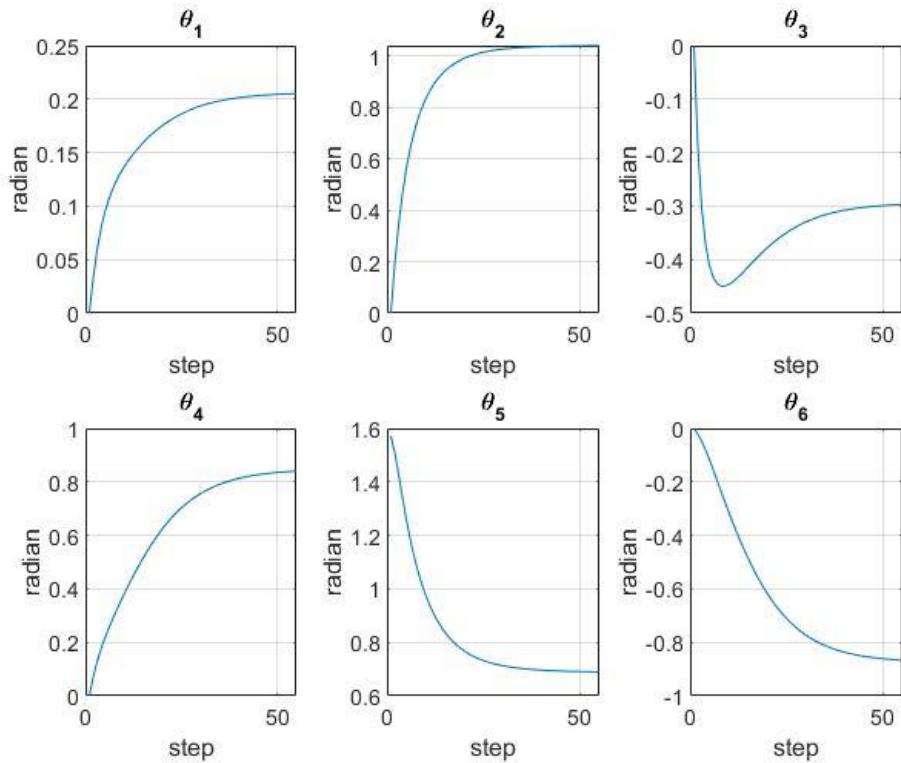


Hình 9.5 Giá trị của các đặc trưng s trong quá trình mô phỏng



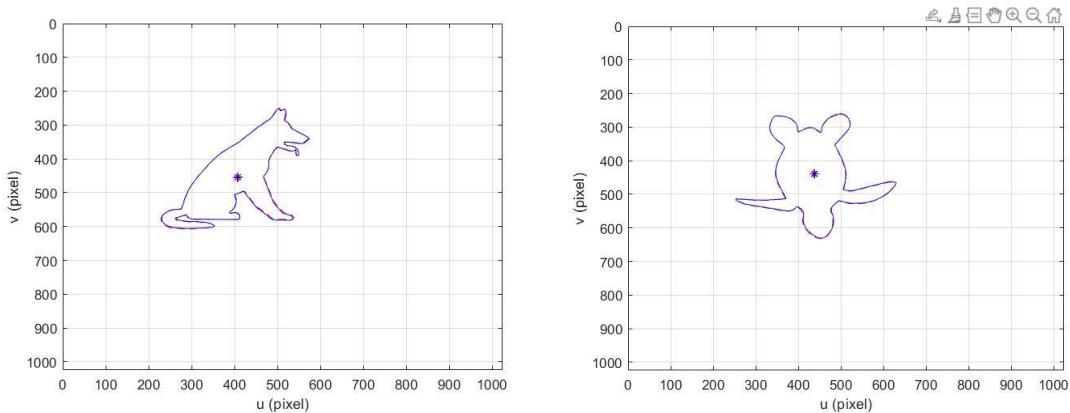
Hình 9.6 Sai số của các đặc trưng trong quá trình mô phỏng

CHƯƠNG 9. MÔ PHỎNG



Hình 9.7 Góc của các khớp robot trong quá trình mô phỏng

Hình 9.5 thể hiện giá trị của các đặc trưng s trong quá trình mô phỏng. Có thể thấy các đặc trưng đần dần hội tụ về giá trị của các đặc trưng mẫu s* được thể hiện ở **bảng 9.1**. **Hình 9.6** thể hiện sai số của các đặc trưng trong quá trình mô phỏng. Có thể thấy các sai số đều dần tiến về không và đạt giá trị sai số cho phép của mô phỏng ở bước 64.

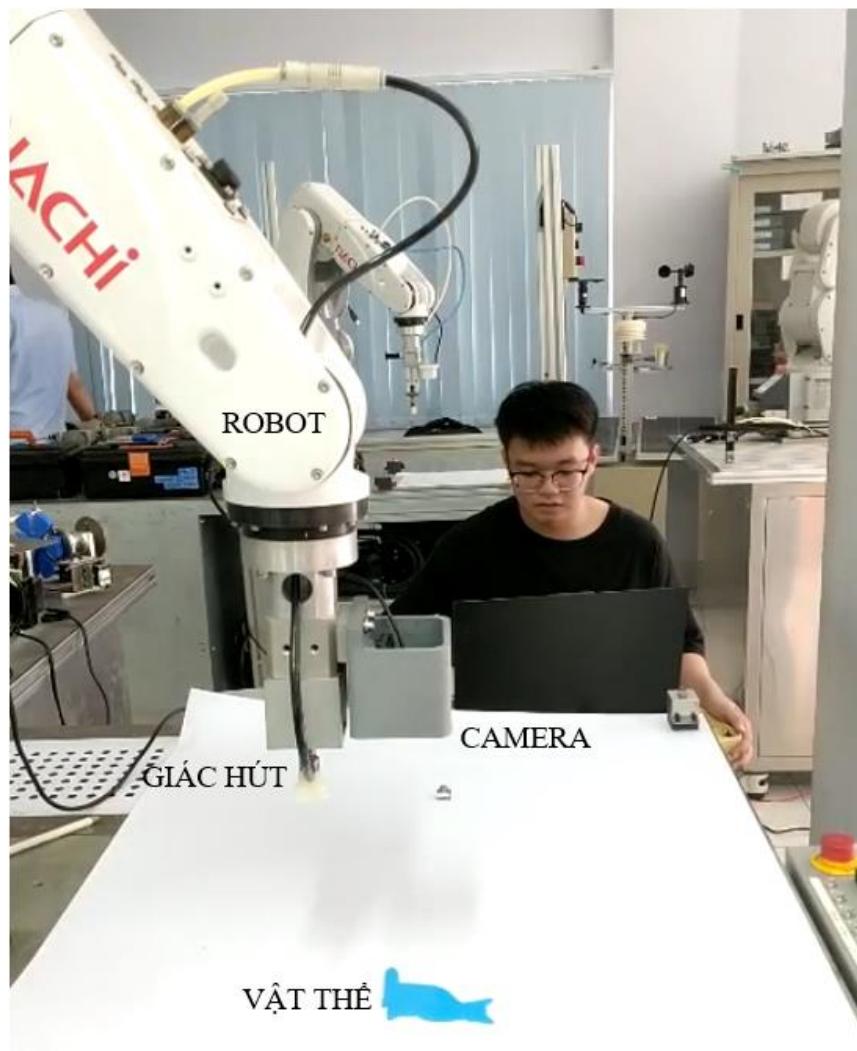


Hình 9.8 Kết quả mô phỏng của một số mẫu vật thể khác

CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ

CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ

Thực nghiệm được tiến hành với vật thể nằm trên mặt phẳng và vật thể có thể được đặt nằm nghiêng một góc nhỏ so với mặt phẳng nằm ngang. Vị trí ban đầu của robot như hình bên dưới.



Hình 10.1 Vị trí ban đầu của robot và vật thể

Các thành phần gồm: Robot nachi MZ07, camera, vật thể, giác hút.

Các bước tiến hành thực nghiệm:

- Bước 1: Kết nối máy tính và CFD Controller, chạy chương trình điều khiển và điều khiển robot về vị trí lấy mẫu được cài đặt từ trước.
- Bước 2: Lấy mẫu hình ảnh ở vị trí mong muốn và tính toán các đặc trưng mong muốn s*.

CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ

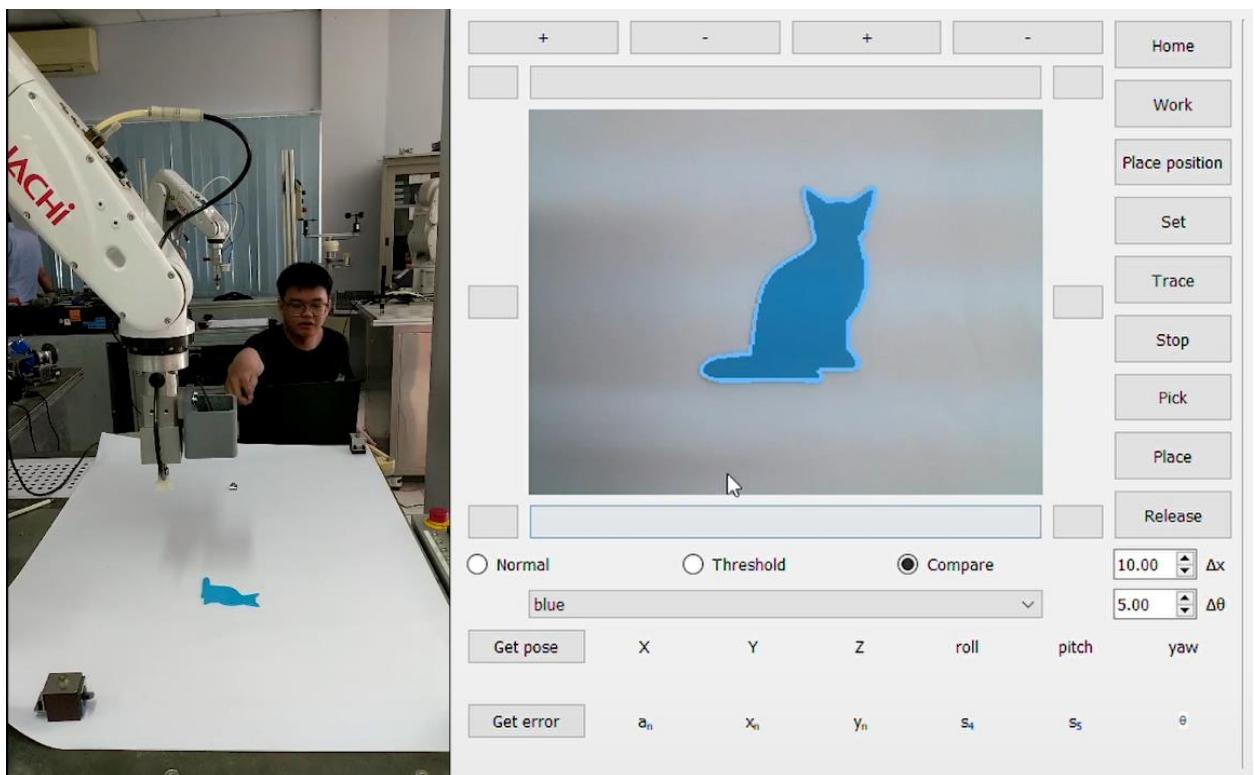
Bảng 10.1 Giá trị các đặc trưng mẫu s*

a _n	x _n	y _n	s ₄	s ₅	theta
6539475	347.32	238.6	0.00096	3.75.10 ⁻⁷	2.074

Chiều cao từ đầu công tác đến vật thể mong muốn $Z_d = 0.1856$ (m).

Ma trận tương tác hay ma trận Jacobian L có giá trị như sau:

$$L = \begin{bmatrix} -1 & 0 & 0 & -1.029 \times 10^7 & 1.389 \times 10^7 & 3.473 \times 10^2 \\ 0 & -1 & 0 & -5.784 \times 10^6 & 1.029 \times 10^7 & -6.539 \times 10^6 \\ 0 & 0 & -1 & -5.209 \times 10^2 & 9.809 \times 10^6 & 0 \\ 0 & 0 & 0 & -1.43 \times 10^{-1} & 2.646 \times 10^{-1} & 0 \\ 0 & 0 & 0 & -2.901 \times 10^{-4} & 4.827 \times 10^{-4} & 0 \\ 0 & 0 & 0 & -4.205 \times 10^2 & -1.859 \times 10^2 & -1 \end{bmatrix}$$



Hình 10.2 Tiến hành lấy mẫu vật thể ở vị trí mong muốn

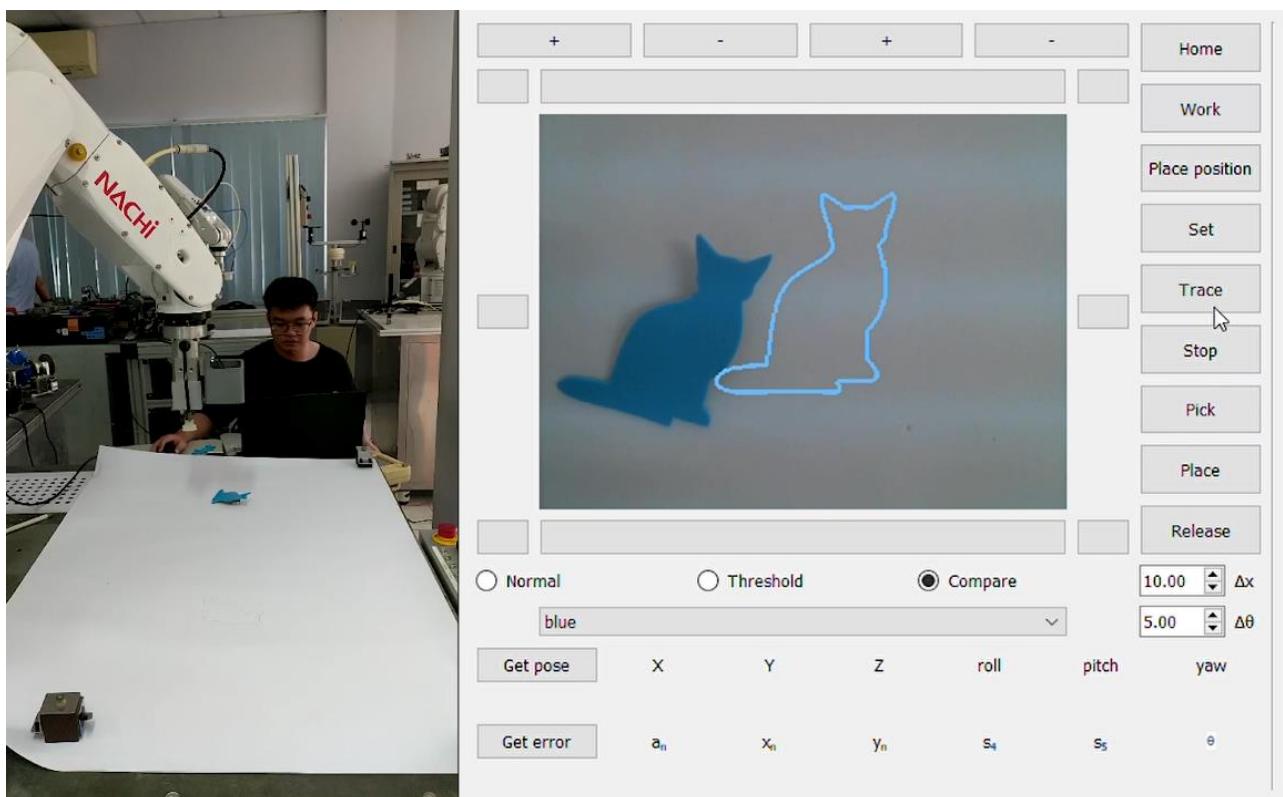
- Bước 3: Đưa vật thể đến vị trí làm việc được cài đặt trước và đặt vật thể nghiêng một góc nhỏ so với mặt phẳng bàn, xoay và dịch chuyển vật thể sao cho hình ảnh camera thu được khác với hình ảnh mẫu lấy ở bước 1.

CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ

- Bước 4: Điều khiển robot đến vị trí làm việc và tiến hành điều khiển hồi tiếp ảnh sao cho cánh tay robot về đúng vị trí mong muốn đã chọn ở bước 2.

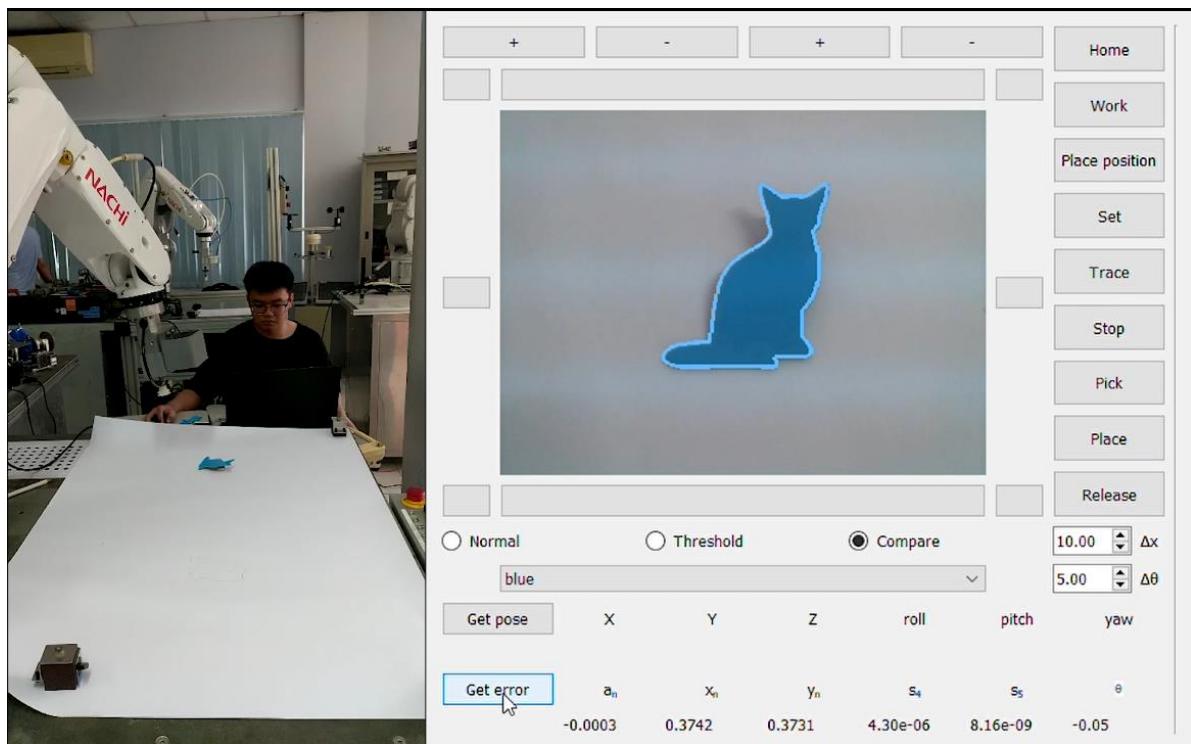


Hình 10.3 Đặt vật thể nghiêng một góc nhỏ so với mặt bàn



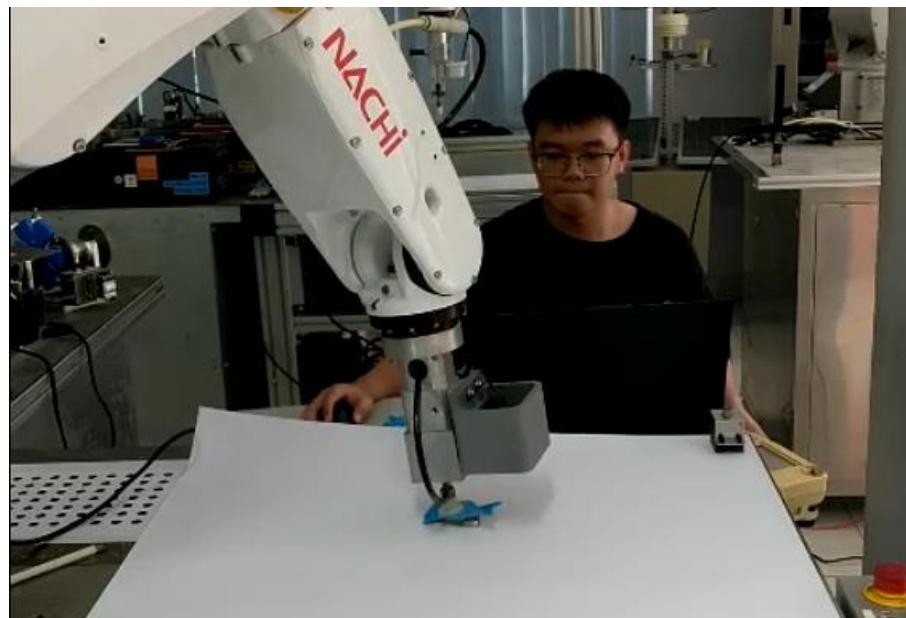
Hình 10.4 Vị trí của robot và vật thể trước khi tiến hành hồi tiếp

CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ



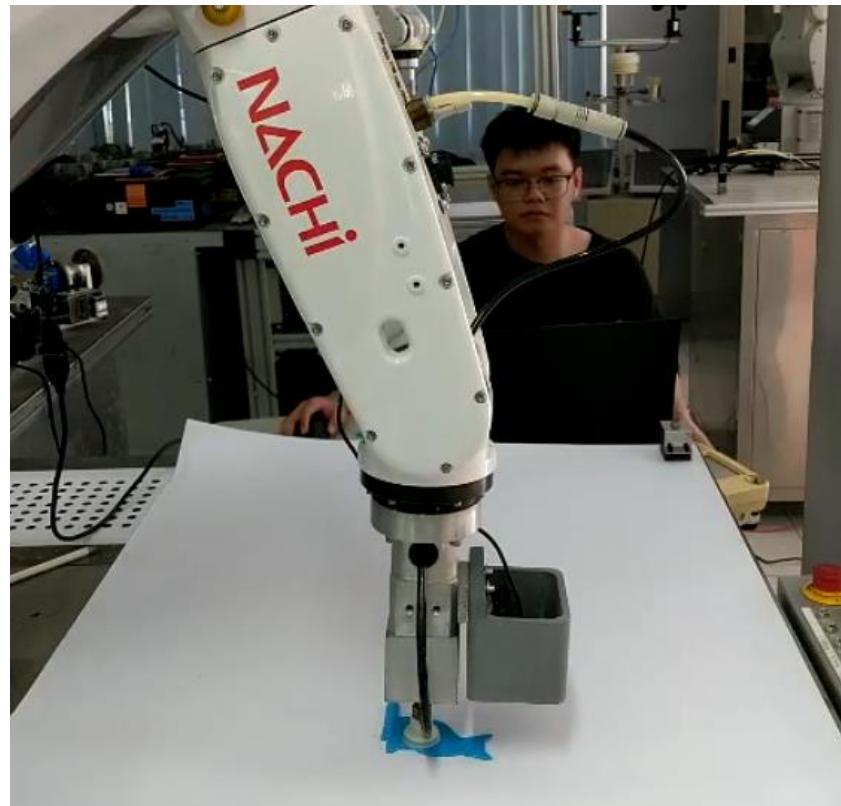
Hình 10.5 Vị trí của robot và vật thể sau khi tiến hành hồi tiếp

- Bước 5: Sau khi robot hồi tiếp về đúng vị trí mong muốn tiến hành gấp vật đưa đến vị trí thả, thả vật và kết thúc thực nghiệm.

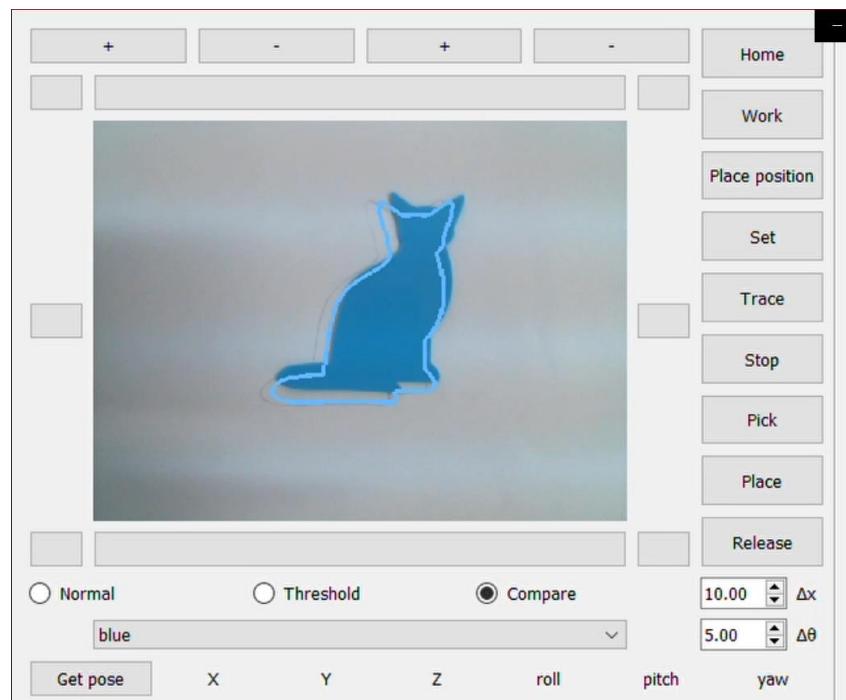


Hình 10.6 Tiến hành gấp vật thể

CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ

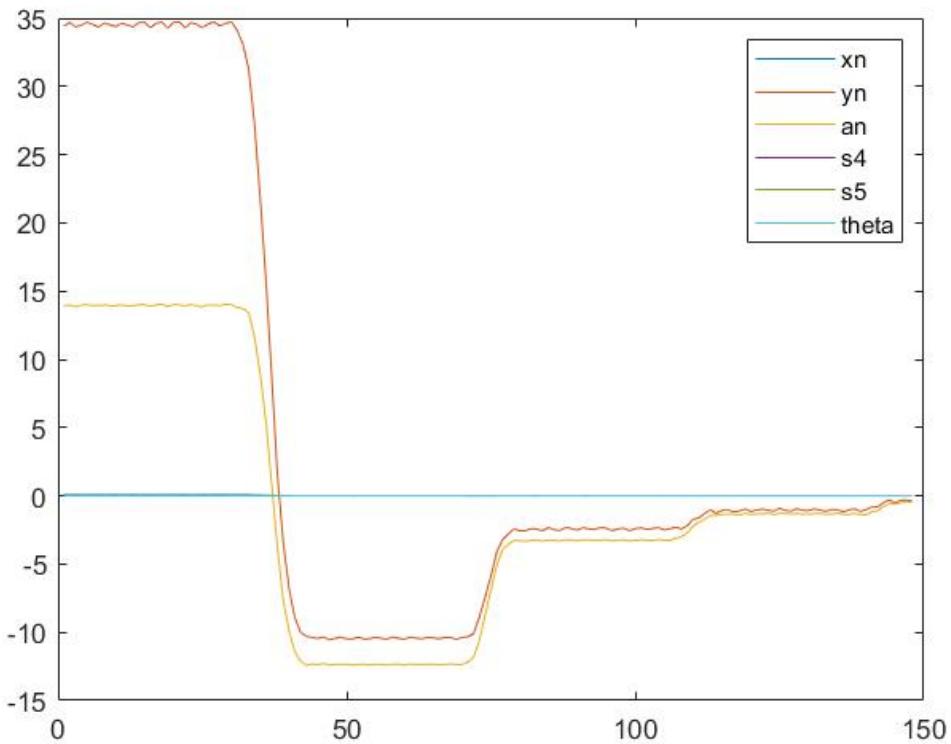


Hình 10.7 Tiến hành thả vật kết thúc thực nghiệm



Hình 10.8 Vị trí vật thể sau khi thả

CHƯƠNG 10. THỰC NGHIỆM VÀ KẾT QUẢ



Hình 10.9 Giá trị sai số của các đặc trưng trong quá trình thực nghiệm

Đánh giá kết quả thực nghiệm:

- Giải thuật chạy tốt, giá trị sai số của các đặc trưng thị giác trong quá trình thực nghiệm được thể hiện ở **hình 10.9** ta có thể thấy giá trị các sai số đều dần bị triệt tiêu và hội tụ về 0.
- Điều khiển robot hồi tiếp về đúng vị trí mong muốn trong thời gian nhiệm vụ luận văn đặt ra.
- Vị trí của vật thể sau khi thả bị lệch so với vị trí lấy mẫu ban đầu (như trên **hình 10.8**) do chưa có điều kiện để thực nghiệm hồi tiếp robot về vị trí thả vật và do khi lấy mẫu chưa căn chỉnh cho tâm của vật thể nằm đúng tâm khung hình của camera thu về dẫn đến khi gấp vật chưa thực sự đúng tâm.

CHƯƠNG 11. KẾT LUẬN

CHƯƠNG 11. KẾT LUẬN

11.1 Kết quả đạt được

- Xây dựng được hướng tiếp cận kỹ thuật điều khiển hồi tiếp Visual Servoing sử dụng bộ đặc trưng thị giác là image moments điều khiển robot hồi tiếp, nhận diện và gấp vật 2D có hình dạng phức tạp. Hoàn thành quá trình điều khiển hồi tiếp ảnh dưới thời gian 1 phút.

- Đề xuất được cách lọc nhiễu khi xử lý hình ảnh thu về từ camera để thu được hình ảnh đầu vào chất lượng cho giải thuật điều khiển.

- Thiết kế được đồ gá camera, đồ gá giác hút.

- Tìm hiểu, nghiên cứu về ngôn ngữ robot và giao tiếp được với robot bằng phương thức giao tiếp socket.

- Xây dựng được giao diện người dùng trực quan.

11.2 Hạn chế của luận văn

- Khuyết điểm của giải thuật điều khiển Visual Servoing là chưa tối ưu hóa đường đi của robot. Khi tiến hành hồi tiếp robot di chuyển còn bị ngắt quãng, giựt chưa được liên tục.

- Thời gian còn khá chậm so với chuẩn công nghiệp.

- Khi vật thể đặt quá lệch so với mẫu được lấy ban đầu quá trình hồi tiếp sẽ mất nhiều thời gian hoặc bị sai.

11.3 Hướng phát triển đề tài

- Thực nghiệm thêm hồi tiếp robot về vị trí thả vật để khi thả vật được chính xác hơn.

- Giảm thời gian hội tụ bằng cách tăng vận tốc robot và nội suy đường đi thay vì di chuyển point to point như hiện tại để robot di chuyển được liên tục mà không bị ngắt quãng.

CHƯƠNG 11. KẾT LUẬN

- Áp dụng thêm các giải thuật xử lý hình ảnh như *Watershed* để nhận diện được các vật thể nằm chồng lên nhau, tách riêng từng vật thể và điều khiển hồi tiếp đối với vật thể nằm trên cùng.

TÀI LIỆU THAM KHẢO

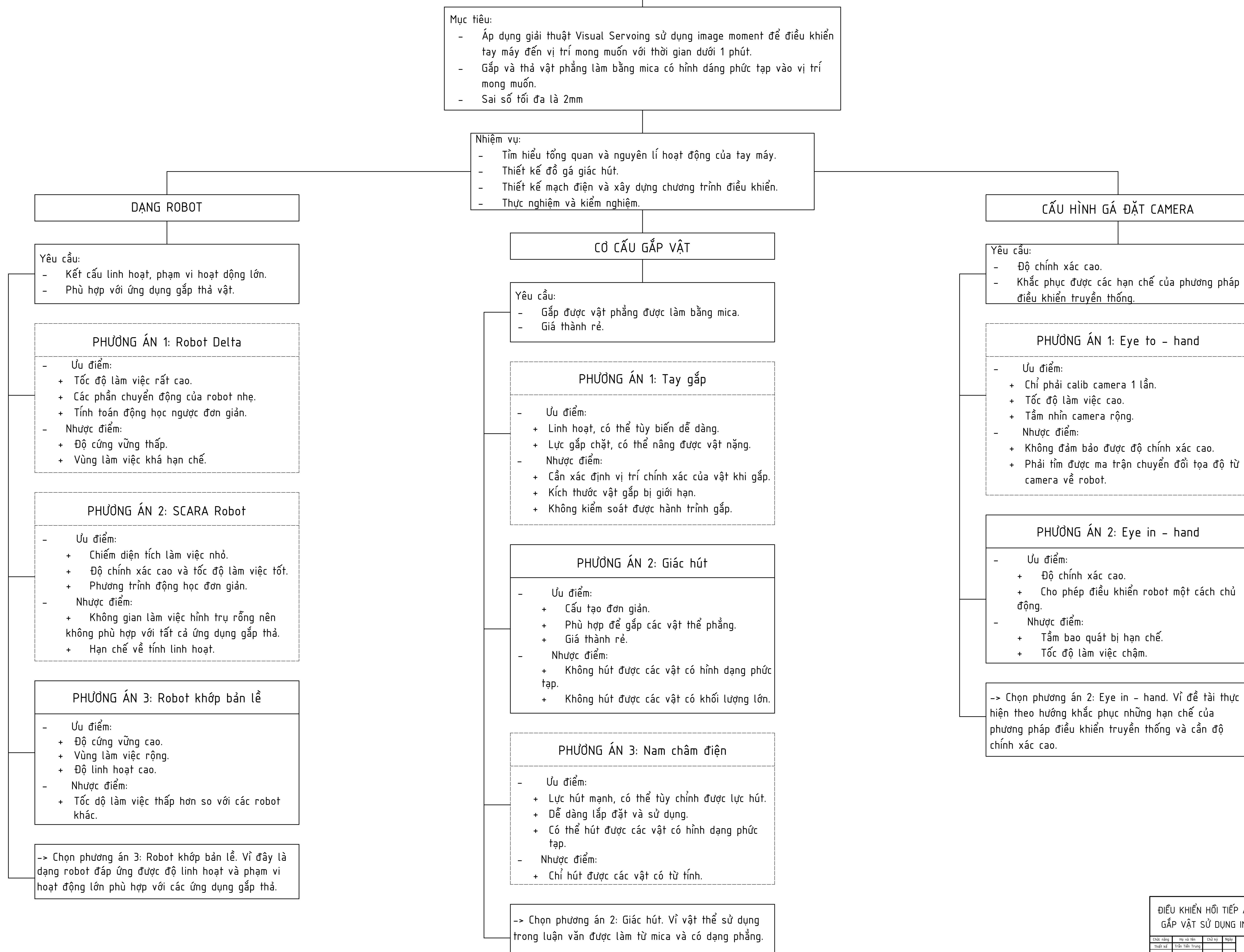
- [1] Nachi Fujikoshi, “Robot School CFD Standard Course”, Toyama: Author.
- [2] Nachi Fujikoshi, “CFD Controller Instruction Manual Basic Operations Manual”, Toyama: Author.
- [3] Nachi Fujikoshi, “FD Controller Instruction Manual Robot language”, Toyama: Author.
- [4] Nachi Fujikoshi, “Manipulator Instruction Manual”, Toyama: Author.
- [5] Nachi Fujikoshi, “FD Controller Instruction Manual Socket Communication”, Toyama: Author.
- [6] Nachi Fujikoshi, “FD Controller Instruction Manual User Task”, Toyama: Author.
- [7] Nachi Fujikoshi, “Manipulator Instruction Manual”, Toyama: Author.
- [8] P. Corke, “Robotic – Vision and Control: Fundamental Algorithms in Matlab”, Springer, 2011.
- [9] P. Corke, “Robotics Toolbox for MATLAB Release 10”, September 2020.
- [10] P. Corke, “Machine Vision Toolbox for MATLAB Release 3”, January 2015.
- [11] F. Chaumette and S. Hutchinson, “Visual servo control. I. Basic approaches” in IEEE Robotics & Automation Magazine, vol. 13, no. 4, pp. 82-90, December 2006.
- [12] F. Chaumette and S. Hutchinson, “Visual servo control. II. Advanced approaches [Tutorial]” in IEEE Robotics & Automation Magazine, vol. 14, no. 1, pp. 109-118, March 2007.
- [13] F. Chaumette, “A first step toward visual servoing using image moments. In IEEE/RSJ International Conference on Intelligent Robots and Systems”, 1, pages 378 - 383, 2002.
- [14] F. Chaumette, “Image moments: a general and useful set of features for visual servoing”, IEEE Transactions on Robotics, 20 (4): 713 - 723, Aug. 2004.

- [15] O. Tahri and F. Chaumette, "Image moments: generic descriptors for decoupled image-based visual servo", In IEEE International Conference on Robotics and Automation, 2, pages 1185 - 1190, April 2004.
- [16] O. Tahri, A. Y. Tamtsia, Y. Mezouar and C. Demonceaux, "Visual Servoing Based on Shifted Moments" in IEEE Transactions on Robotics, vol. 31, no. 3, pp. 798-804, June 2015.
- [17] O. Tahri and F. Chaumette, "Point-based and region-based image moments for visual servoing of planar objects", IEEE Transactions on Robotics, 21 (6): 1116 - 1127, Dec 2005.
- [18] Yimin Zhao, Wen-Fang Xie, and Sining Liu, "Image-based visual servoing using Improved image moments in 6-DOF robot systems", International Journal of Control, Automation and Systems, 11 (3): 586 - 596, 2013.
- [19] E. Malis, F. Chaumette and S. Boudet, "2 1/2 D visual servoing", IEEE Transactions on Robotics and Automation, vol. 15, no. 2, pp. 238-250, April 1999.
- [20] S. Hutchinson, G. D. Hager and P. I. Corke, "A tutorial on visual servo control," in IEEE Transactions on Robotics and Automation, vol. 12, no. 5, pp. 651-670, Oct. 1996.
- [21] S. Benhimane and E. Malis, "Homography-based 2D visual servoing," Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006., pp. 2397-2402, 2006.
- [22] P. I. Corke and S. A. Hutchinson, "A new partitioned approach to image-based visual servo control," in IEEE Transactions on Robotics and Automation, vol. 17, no. 4, pp. 507-515, Aug. 2001.
- [23] V. Kyrki, D. Kragic and H. I. Christensen, "New shortest-path approaches to visual servoing," 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), pp. 349-354 vol.1, 2004.
- [24] V. D. Công, "Nghiên cứu và đánh giá các kỹ thuật điều khiển Visual Servoing", Luận văn Thạc sĩ Kỹ thuật Cơ điện tử, Trường Đại học Bách Khoa (ĐHQG-HCM), TP.Hồ Chí Minh, 2020.

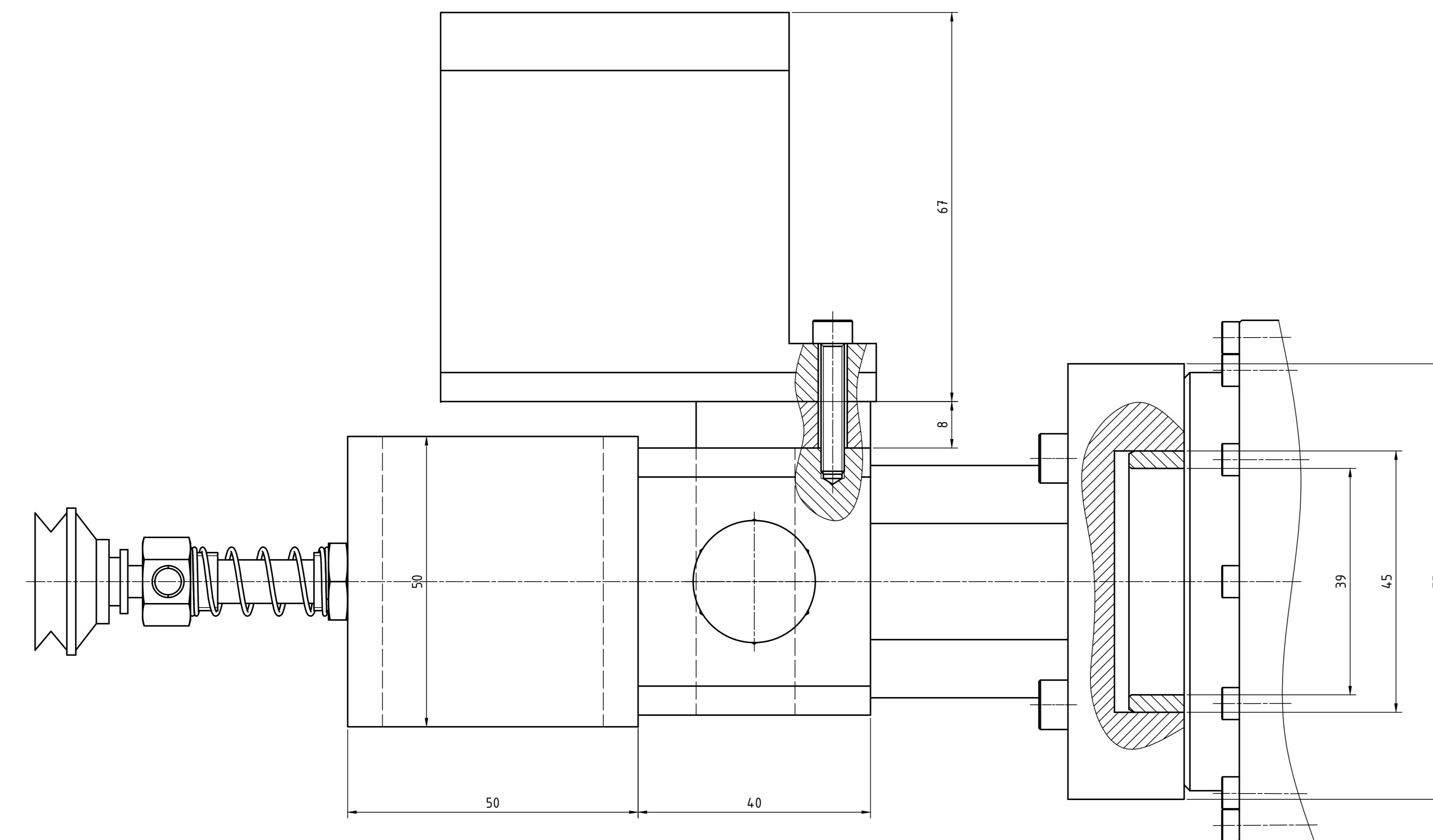
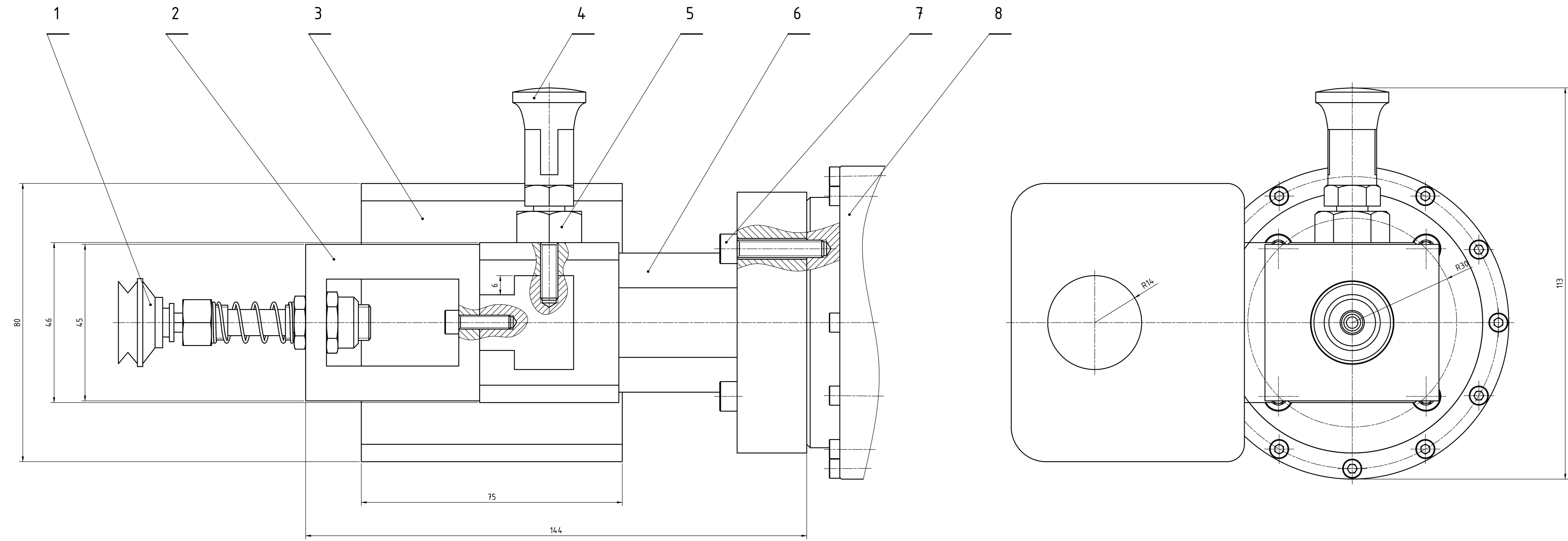
- [25] N. Đ. Toàn, “Điều khiển hồi tiếp ảnh cho robot công nghiệp sử dụng moment ảnh”, Luận văn tốt nghiệp Kỹ thuật Cơ điện tử, Trường Đại học Bách Khoa (ĐHQG - HCM), TP.Hồ Chí Minh, 2019.
- [26] Bruno Siciliano, Oussama Khatib (Eds.), “Springer Handbook of Robotics”, Springer – Verlag Berlin Heidelberg, 2016.

_____ END _____

IMAGE MOMENTS - BASED VISUAL SERVOING FOR ROBOT GRASPING OBJECT



ĐIỀU KHIỂN HỘI TIẾP ẢNH CHO ROBOT GẮP VẬT SỬ DỤNG IMAGE MOMENTS				LUẬN VĂN TỐT NGHIỆP	
Chức năng	Họ và tên	Chữ ký	Ngày	BẢN VẼ LƯU CHON PHƯƠNG ÁN	
Thiết kế	Trần Tiến Trung			Số lượng	Khối lượng
Hướng dẫn	Lê Đức Hạnh			1	11
Duyệt				SD tr	SD tr
				Toàn	
				SD tr	SD tr



8	Mặt bích robot	1	Thép	
7	Bulong lục giác chìm M5	4	Thép	
6	Intermediate mounting	1	Thép	
5	Đai ốc M10	1	Thép	
4	Indexing spring plunger	1	Nhựa	
3	Hộp chứa camera	1	Nhựa	
2	Gá giắc hút	1	Thép	
1	Cốc giắc hút	1	Cao su	
Số	Ký hiệu	Tên gọi	Số lượng	Vật liệu
				Ghi chú

ĐIỀU KHIỂN HỐI TIẾP ẢNH CHO ROBOT
GẮP VẬT SỬ DỤNG IMAGE MOMENTS

LUẬN VĂN TỐT
NGHIỆP

Chức năng	Họ và tên	Đã ký	Ngày	SD lượng	Khối lượng	Tỉ lệ
Thiết kế	Trần Tiến Trung					2:1
Hướng dẫn	Lê Đức Hạnh					1:1
Duyệt						

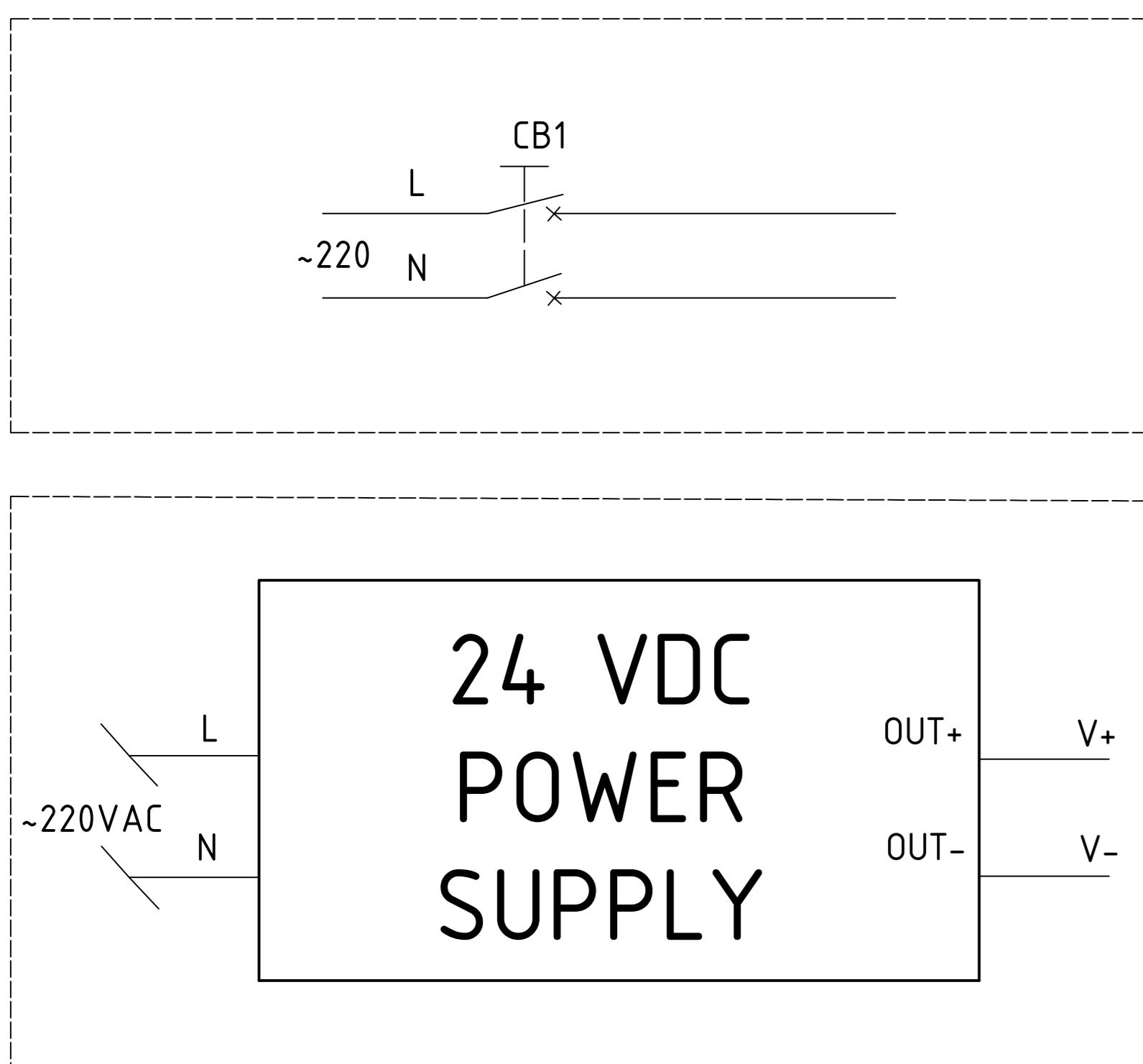
BẢN VẼ CƠ KHÍ

TRƯỜNG ĐẠI HỌC BÁCH KHOA
TP HỒ CHÍ MINH
Lớp CK17CD01

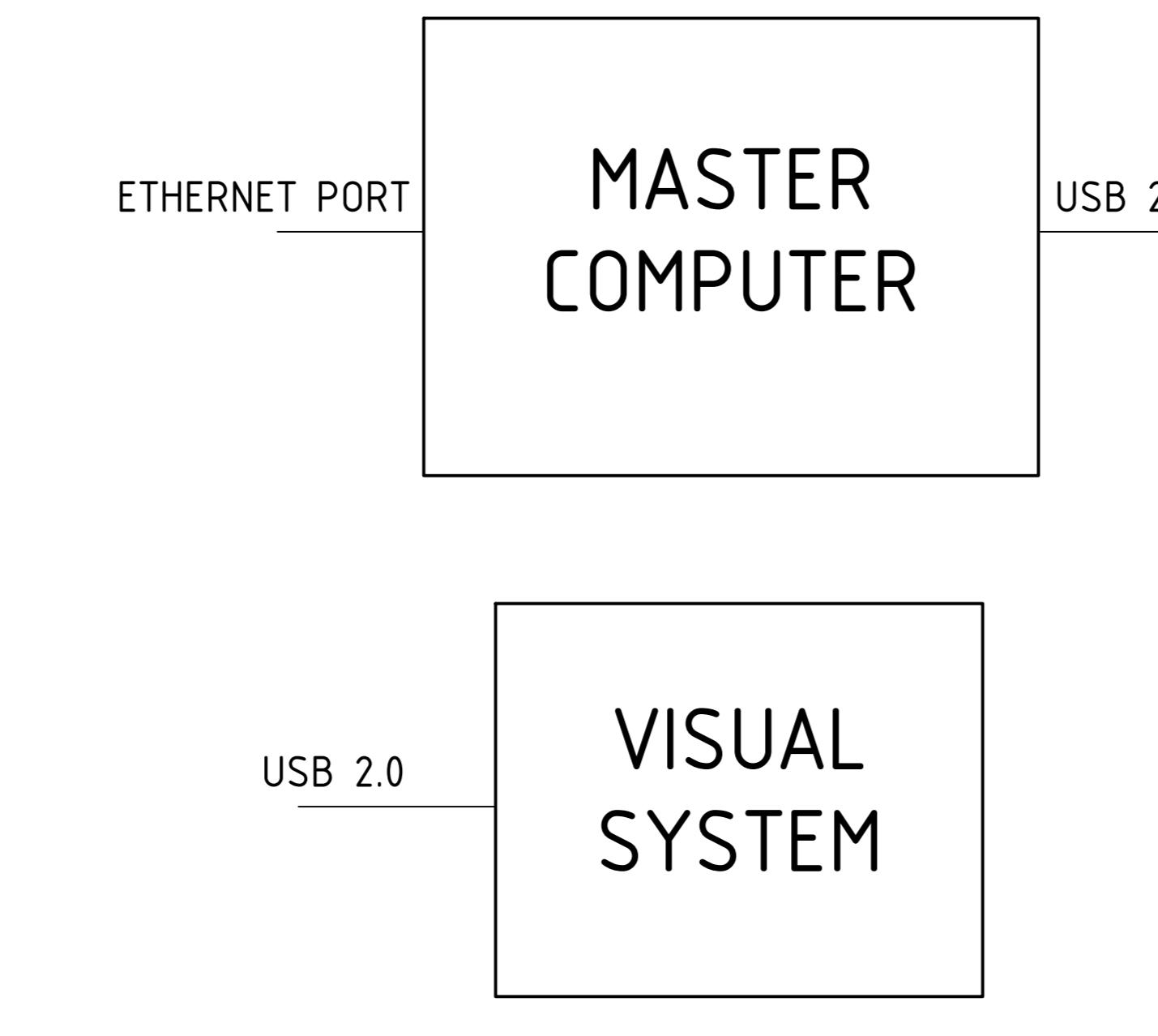
Khối kết nối robot



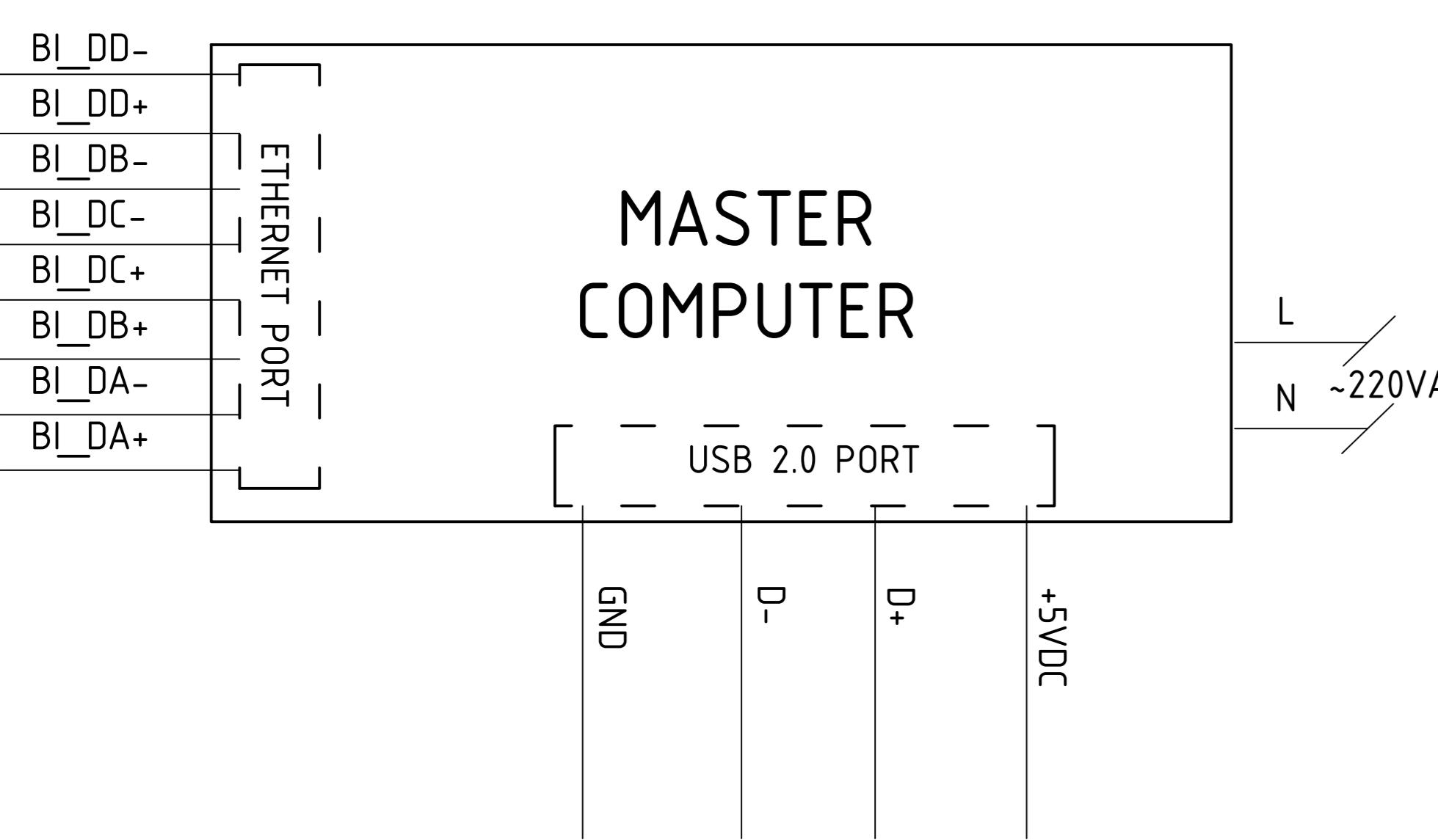
Khối nguồn



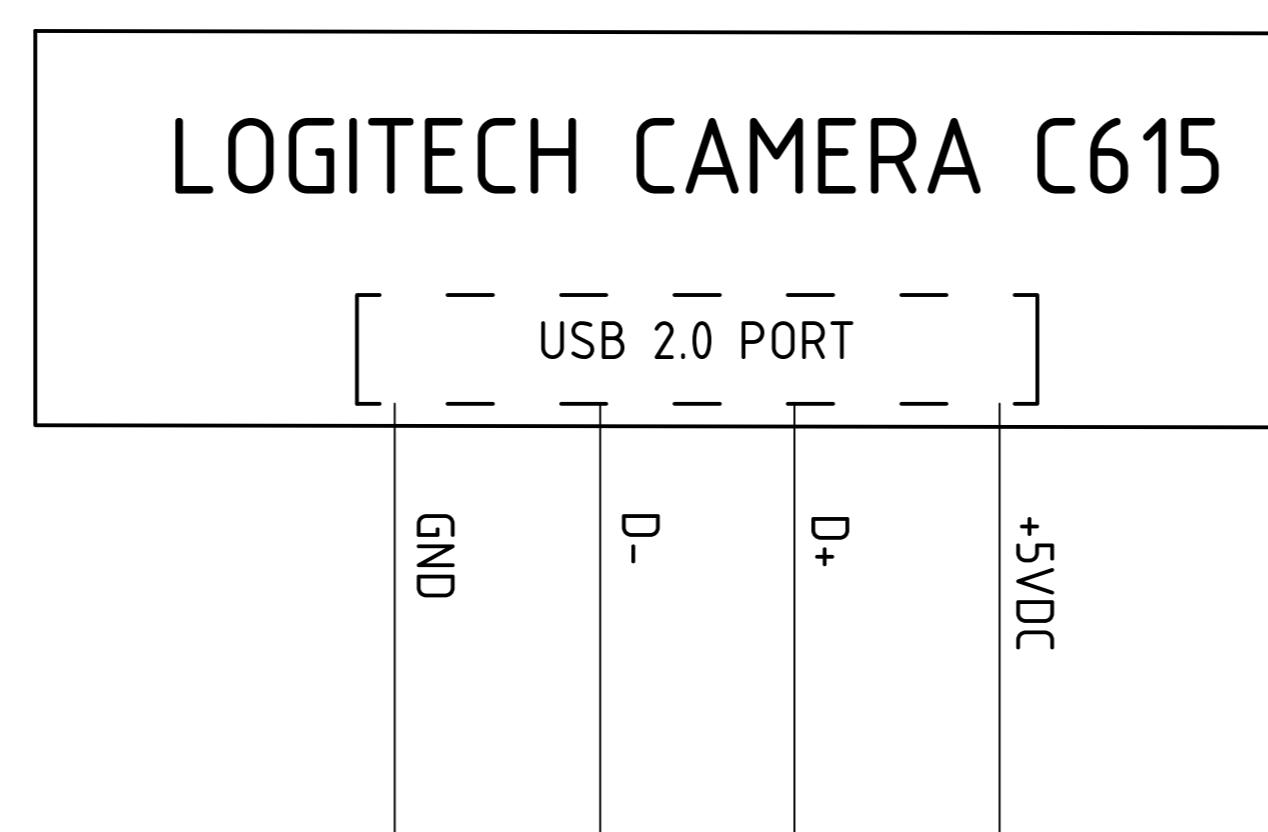
Khối Visual System



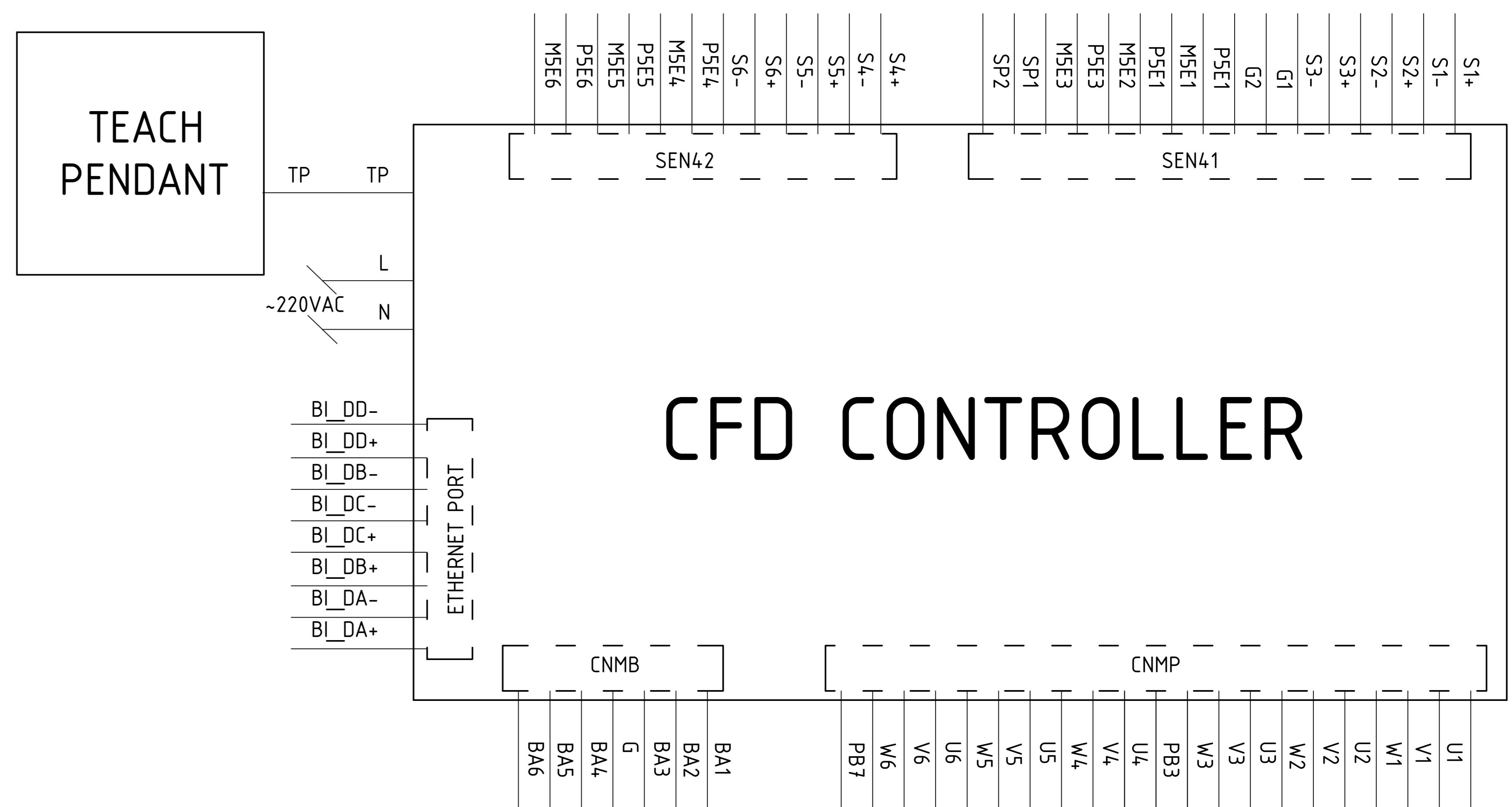
Khối master computer



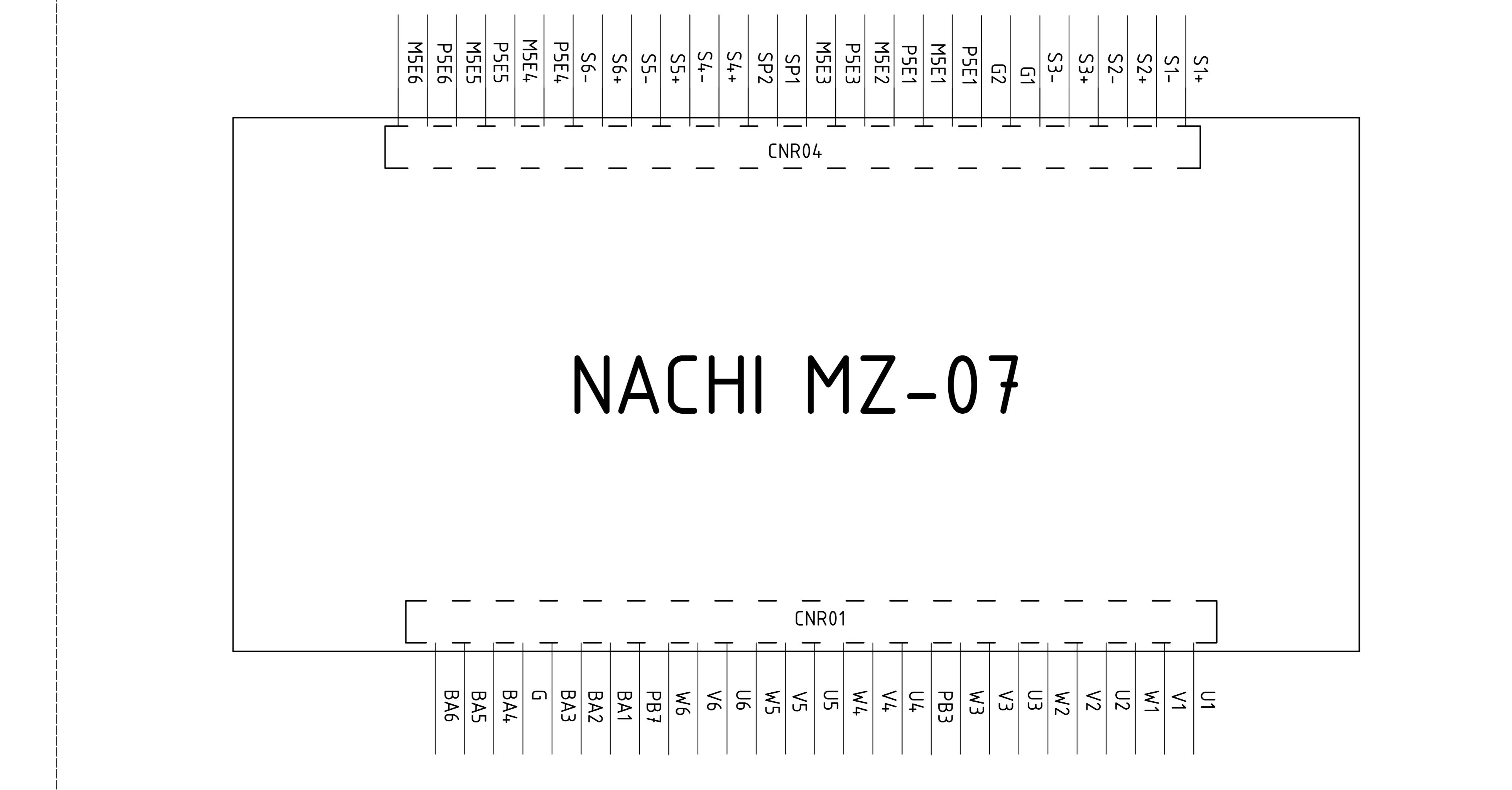
Khối camera



Khối CFD Controller



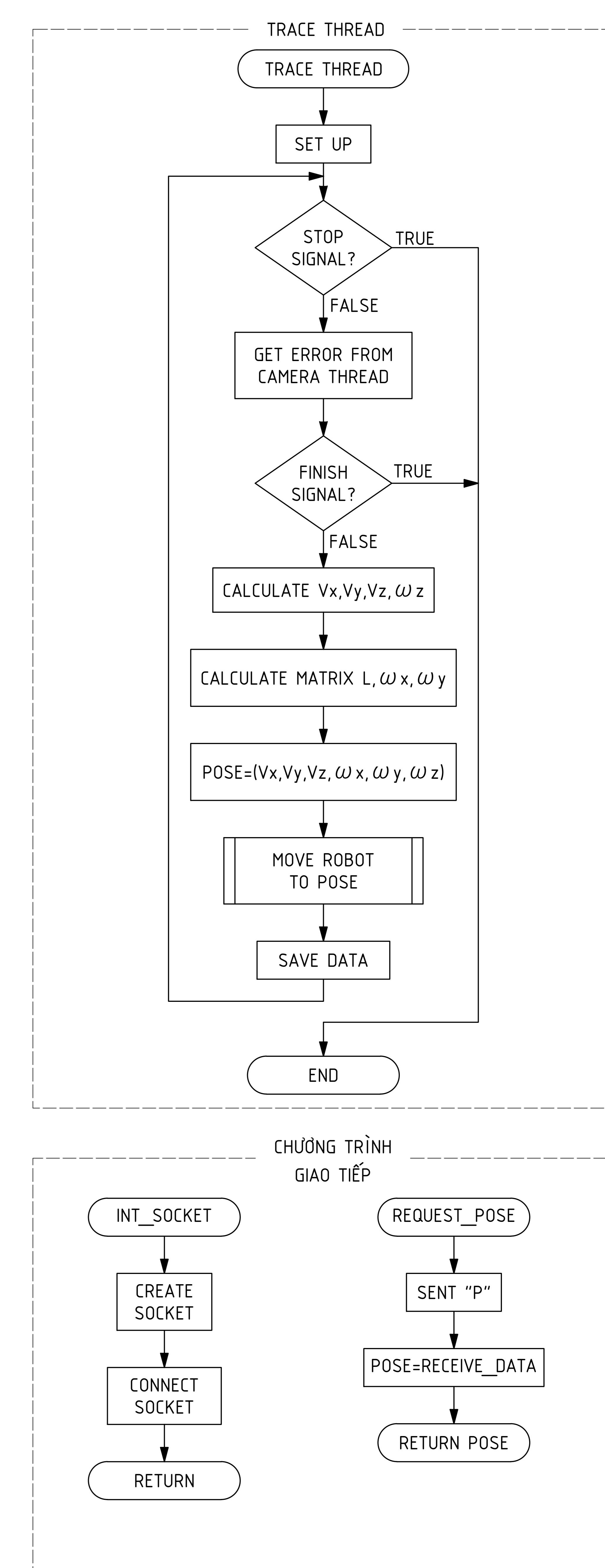
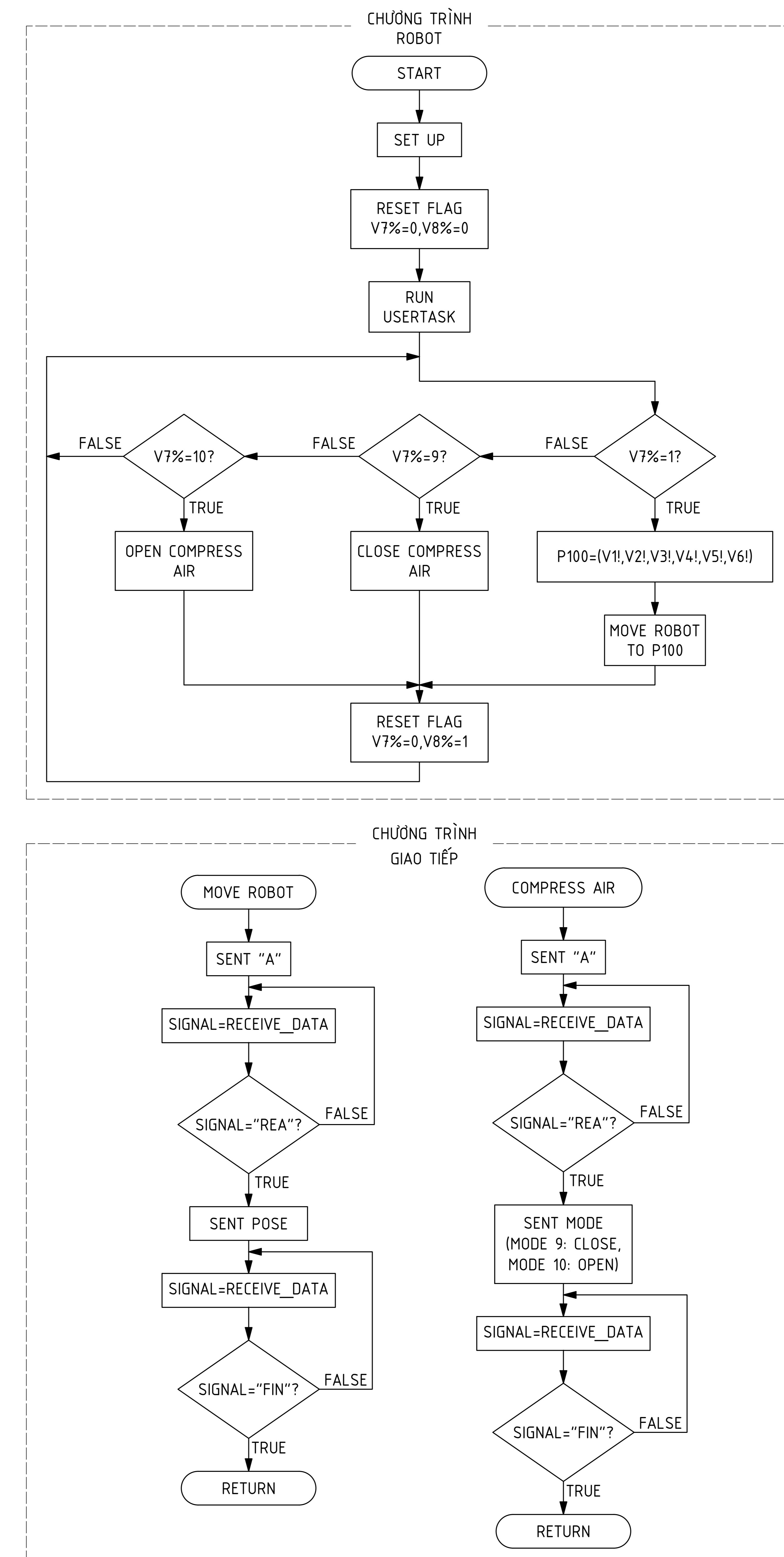
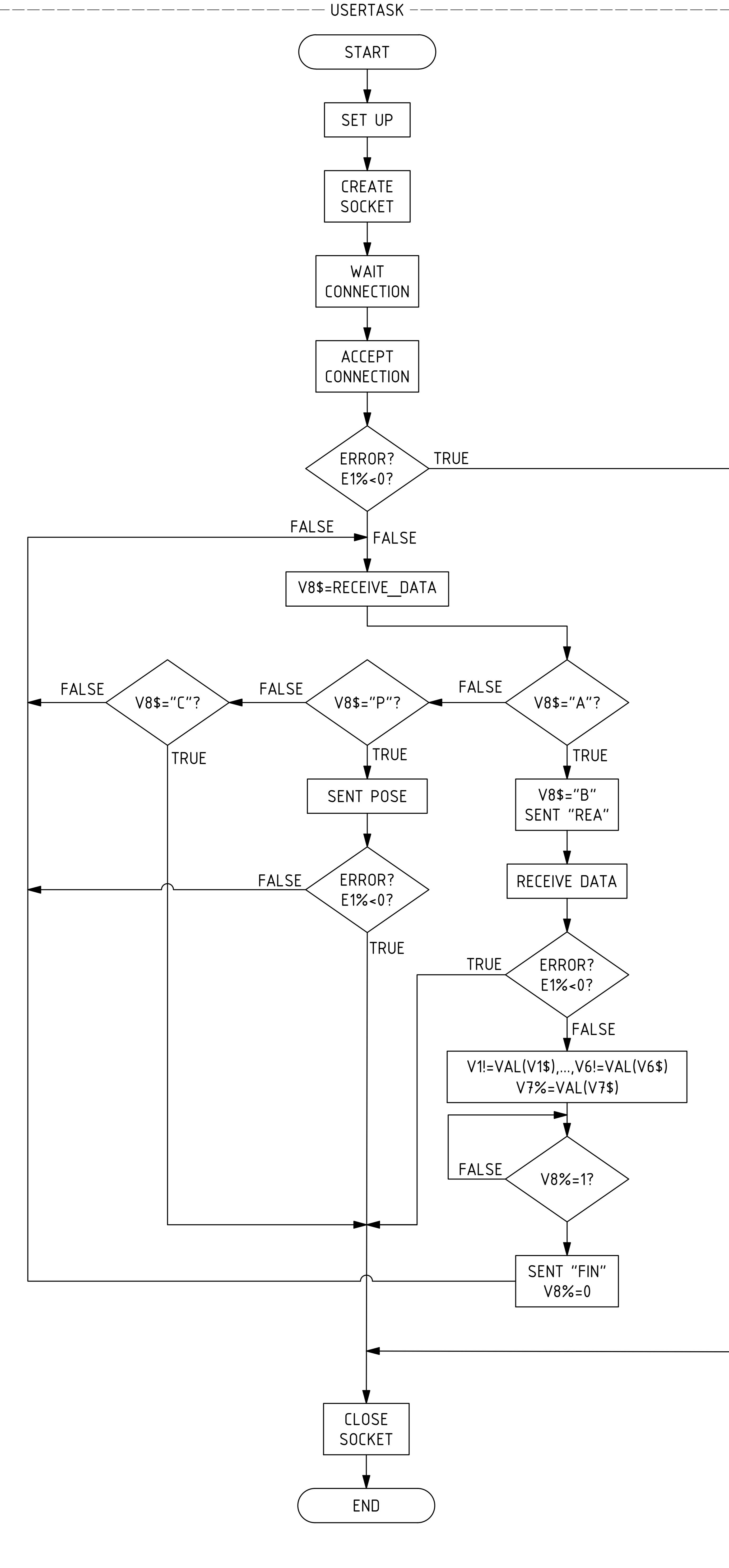
Khối cánh tay robot



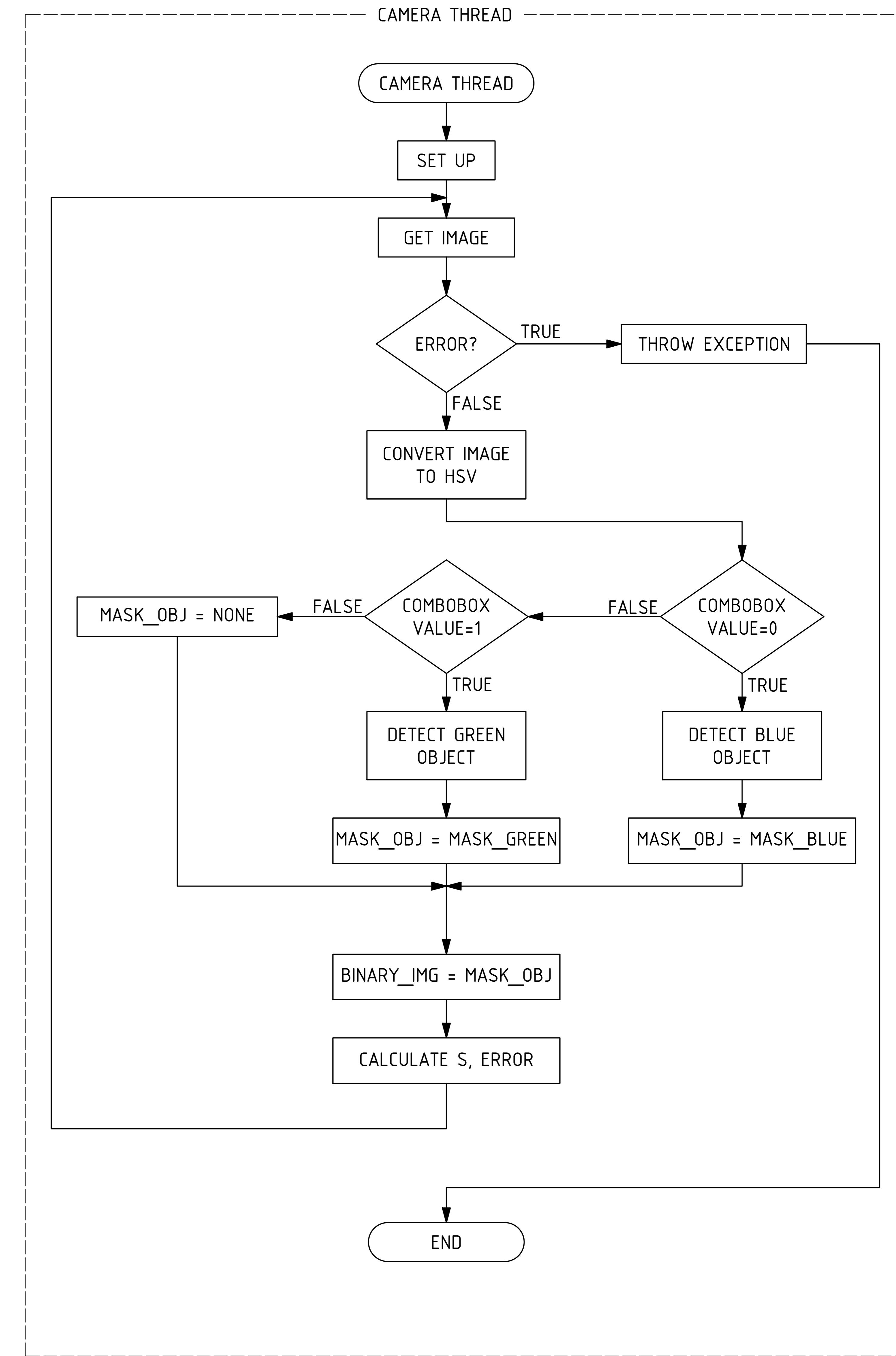
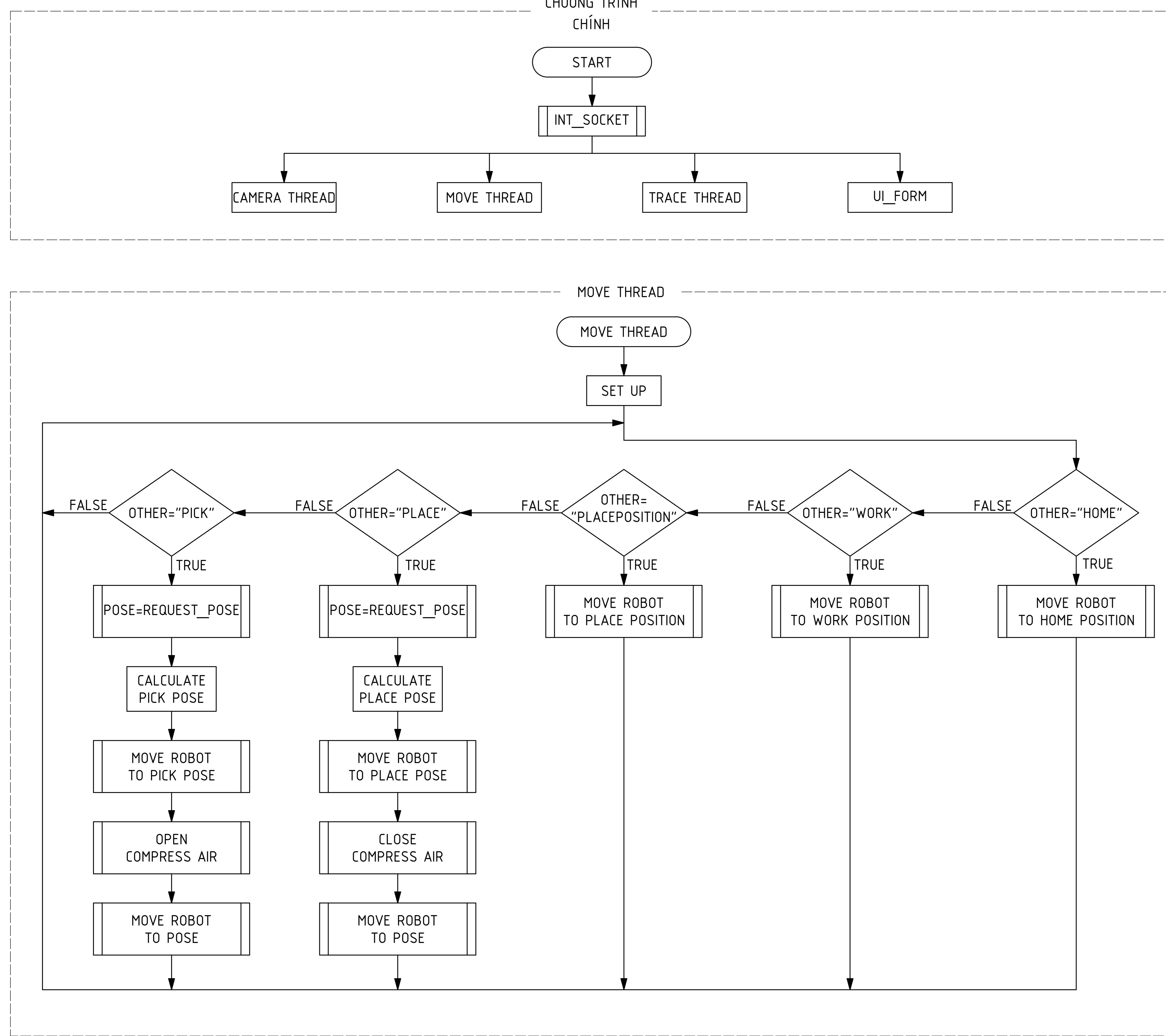
NACHI MZ-07

ĐIỀU KHIỂN HỘI TIẾP ẢNH CHO ROBOT GẮP VẬT SỬ DỤNG IMAGE MOMENTS					LUẬN VĂN TỐT NGHIỆP	
Chức năng	Họ và Họ	Chữ ký	Ngày		BẢN VẼ ĐIỆN	
Thiết kế	Trần Tiến Trung					Số lượng
Hàng sẵn	Lê Đức Hạnh					Khối lượng
Duyệt						Tổng
						11
						Số tờ 1

TRƯỜNG ĐẠI HỌC BÁCH KHOA
TP HỒ CHÍ MINH
Lớp CK17CD01



ĐIỀU KHIỂN HỒI TIẾP ẢNH CHO ROBOT GẮP VẬT SỬ DỤNG IMAGE MOMENTS				LUẬN VĂN TỐT NGHIỆP
Chức năng	Họ và tên	Chữ ký	Ngày	Số lượng
Thiết kế	Trần Tiến Trung			Khối lượng
				Tỉ lệ
Hướng dẫn	Lê Đức Hạnh			1:1
Duyệt				Tờ: 1
				Số tờ: 2
BẢN VẼ LUU ĐÔ VÀ GIẢI THUẬT				TRƯỜNG ĐẠI HỌC BÁCH KHOA TP HỒ CHÍ MINH Lớp CK17CD01



ĐIỀU KHIỂN HỐI TIẾP ẢNH CHO ROBOT GẮP VẬT SỬ DỤNG IMAGE MOMENTS				LUẬN VĂN TỐT NGHIỆP		
Chức năng	Họ và tên	Chữ ký	Ngày	BẢN VẼ LƯU ĐỒ VÀ GIẢI THUẬT	Số lượng	Khối lượng
Thiết kế	Trần Tiến Trung					Tỉ lệ
						1:1
Hướng dẫn	Lê Đức Hạnh				Tờ: 2	Số tờ: 2
Duyệt						



**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CƠ KHÍ – BỘ MÔN CƠ ĐIỆN TỬ**

LUẬN VĂN TỐT NGHIỆP

**ĐIỀU KHIỂN HỒI TIẾP ẢNH CHO ROBOT GẮP VẬT SỰ
DỤNG IMAGE MOMENT**

GVHD : TS. Lê Đức Hạnh
SVTH : Trần Tiến Trung
MSSV : 1713722
Email : trung.tran2899@hcmut.edu.vn
SĐT : 0358432114

TP. Hồ Chí Minh, ngày 27/07/2021

1



TÓM TẮT NỘI DUNG LUẬN VĂN

1. Tổng quan
2. Lựa chọn phương án
3. Đặc tả hệ thống
4. Giải thuật Image moment based Visual Servoing
5. Mô phỏng
6. Thực nghiệm và đánh giá kết quả
7. Kết luận và hướng phát triển



1. TỔNG QUAN

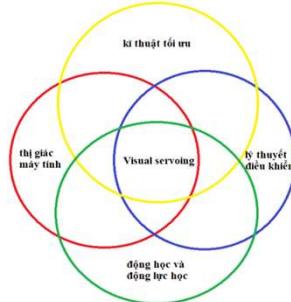
Giới thiệu Visual Servoing.

Visual servoing là một kỹ thuật để điều khiển robot, trong đó thông tin hồi tiếp thị giác được sử dụng trong vòng điều khiển kín giúp cải thiện độ chính xác. Đây là một kỹ thuật với sự kết hợp của nhiều lĩnh vực khác nhau như thị giác máy tính, động học và động lực học, lý thuyết điều khiển và các kỹ thuật tối ưu hóa.

Một tác vụ robot trong visual servoing sử dụng các đặc trưng hình ảnh trích xuất từ đối tượng để điều khiển chuyển động của robot/camera.

Hàm sai số trong visual servoing được định nghĩa là sai số giữa các đặc trưng thị giác hiện tại và đặc trưng thị giác tại vị trí mong muốn. Mục tiêu của sơ đồ điều khiển dựa trên thị giác là điều chỉnh sai số và triệt tiêu nó.

Dựa vào thông tin thị giác được lựa chọn, Visual Servoing được chia thành 2 sơ đồ chính là position based visual servoing và image based visual servoing. Trong đề tài này sẽ ứng dụng image based visual servoing với bộ đặc trưng thị giác là moment ảnh



1. TỔNG QUAN

Mục tiêu luận văn.

-Áp dụng giải thuật Visual Servoing sử dụng image moment để điều khiển tay máy đến vị trí mong muốn với thời gian dưới 1 phút.

-Gắp và thả vật phẳng làm bằng mica có hình dáng phức tạp vào vị trí mong muốn.

-Sai số tối đa là 2mm.

Nhiệm vụ luận văn.

-Tìm hiểu tổng quan và nguyên lý hoạt động của tay máy.

-Thiết kế đồ gá đầu hút.

-Thiết kế mạch điện và xây dựng chương trình điều khiển.

-Thực nghiệm kiểm nghiệm.



2. LỰA CHỌN PHƯƠNG ÁN

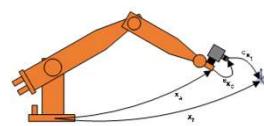
Lựa chọn loại robot.

Lựa chọn dạng robot khớp bắn lè vì đây là dạng robot đáp ứng được độ linh hoạt và phạm vi hoạt động lớn phù hợp với các ứng dụng gấp thả.



Lựa chọn cơ cấu gấp.

Vì vật thể được sử dụng trong đê tài được làm bằng mica phẳng nên ta chọn cơ cấu đê gấp vật là giác hút.



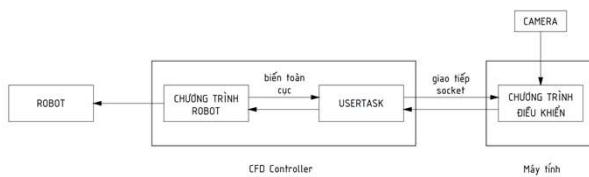
Lựa chọn cấu hình camera.

Đê tài thực hiện hướng tới việc khắc phục các hạn chế về điều khiển theo phương pháp truyền thống như quá trình điều khiển là vòng hở, không thể điều khiển được nếu không biết được vị trí chính xác trong hệ tọa độ robot của vật thể, nên đê tài sẽ chọn cấu hình gá đặt camera gắn trực tiếp trên cánh tay robot.

*Xem chi tiết trong bản vẽ lựa chọn phương án



3. ĐẶC TẨ HỆ THỐNG



Mô tả nguyên lý hoạt động của hệ thống.

Bước 1: Lấy mẫu vật thể và tính toán các đặc trưng thị giác mẫu s^* và ma trận tương tác L

Bước 2: Đưa vật thể và robot đến vùng làm việc và đặt vật thể lệch di so với mẫu đã lấy ở bước 1

Bước 2: Áp dụng giải thuật image moment based visual servoing để hồi tiếp vật.

Bước 3: Sau khi đã hồi tiếp được vật tiến hành điều khiển robot gấp thả vật



4. GIẢI THUẬT IMAGE MOMENT BASED VISUAL SERVOING

Lý thuyết chung của Visual Servoing.

$$\dot{s} = L_2 v_c, \quad \dot{e} = L_2 v_c, \quad \dot{\theta} = -\lambda e,$$

$$v_c = -\lambda L_2^+ e = -\lambda L_2^+ (s - s^*),$$

Giới thiệu về image moment.

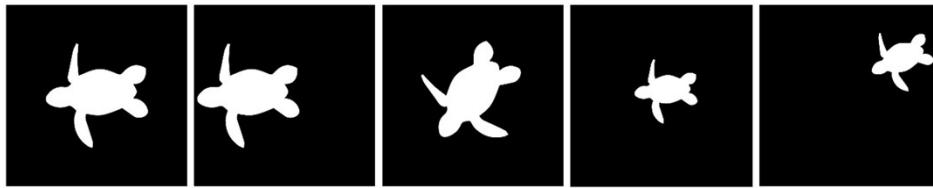
$$v_c = -\lambda \hat{L}_2^+ e = -\lambda \hat{L}_2^+ (s - s^*),$$

Giải thuật image moment based Visual Servoing

$$s = [x_n, y_n, a_n, s_4, s_5, \theta],$$

Cải tiến giải thuật

$$L_2 = \begin{bmatrix} L_{x_n} \\ L_{y_n} \\ L_{a_n} \\ L_{s_4} \\ L_{s_5} \\ L_\theta \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & a_n \epsilon_{11} & -a_n(1 + \epsilon_{12}) & y_n \\ 0 & -1 & 0 & a_n(1 + \epsilon_{21}) & -a_n \epsilon_{22} & -x_n \\ 0 & 0 & -1 & -a_n \epsilon_{31} & a_n \epsilon_{32} & 0 \\ 0 & 0 & 0 & s_{4\alpha x} & s_{4\alpha y} & 0 \\ 0 & 0 & 0 & s_{5\alpha x} & s_{5\alpha y} & 0 \\ 0 & 0 & 0 & \theta_{\alpha x} & \theta_{\alpha y} & -1 \end{bmatrix}$$



5. MÔ PHỎNG

Quá trình mô phỏng

Bước 1: Tạo dữ liệu vật thể và load vào Matlab

Bước 2: Sử dụng 2 toolbox là Robotics toolbox và Machine vision tool để tạo đối tượng robot và camera.

Bước 3: Cho camera nhìn vật thể và lấy mẫu, tính toán các đặc trưng thị giác mong muốn.

Bước 4: Đặt vật thể ở vị trí mô phỏng có ma trận chuyển vị so với camera được cho trước.

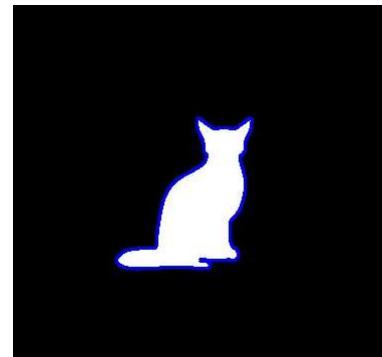
Bước 5: Tính toán các đặc trưng thị giác s.

Bước 6: Tính sai số

Bước 7: Tính vận tốc camera.

Bước 8: Tính góc xoay các khớp robot.

Bước 9: Di chuyển robot đến vị trí mới.



Giá trị các đặc trưng thị giác mong muốn

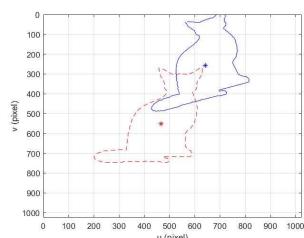
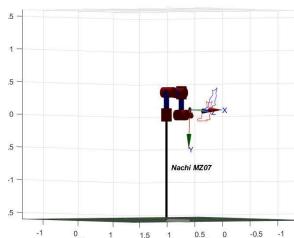
x_n	y_n	a_n	s_4	s_5	theta
-0.0282	0.0244	0.5	-0.25	0.0258	0.5096



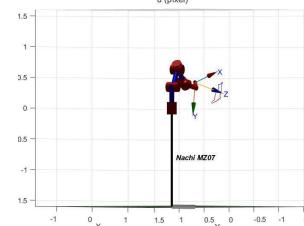
5. MÔ PHỎNG

Kết quả mô phỏng

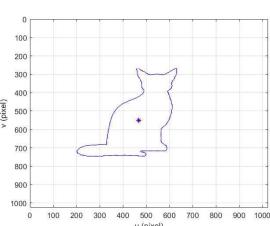
Quá trình mô phỏng mất 54 bước để hội tụ.



Sau khi tiến hành hồi tiếp



Trước khi tiến hành hồi tiếp



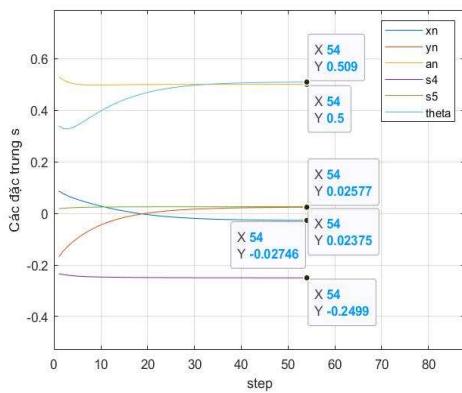
*Link video mô phỏng: <https://youtu.be/jJAJBk5i3O0>



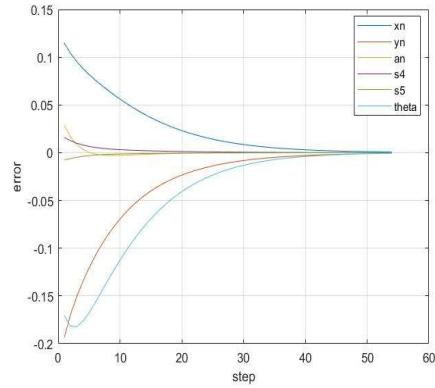
5. MÔ PHỎNG

Kết quả mô phỏng

Qua hai dữ liệu lưu về trong quá trình mô phỏng là giá trị các đặc trưng thị giác và giá trị của sai số ta thấy các đặc trưng thị giác đã dần hội tụ về giá trị mong muốn s^* lấy mẫu ban đầu. Giá trị của các sai số cũng dần bị tiêu tiêu và tiến về 0.



Giá trị của các đặc trưng thị giác s trong quá trình mô phỏng



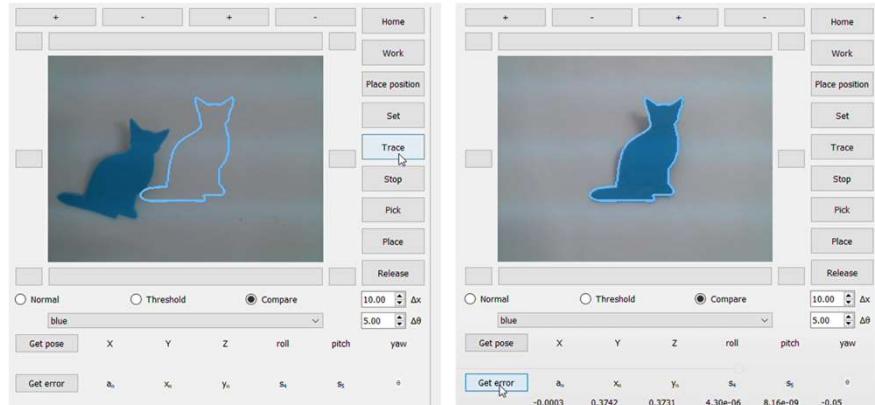
Giá trị của sai số e trong quá trình mô phỏng



6. THỰC NGHIỆM

Kết quả thực nghiệm

Hồi tiếp được robot về vị trí mong muốn, giải thuật chạy tốt.



Trước khi tiến hành hồi tiếp

Sau khi tiến hành hồi tiếp



6. THỰC NGHIỆM

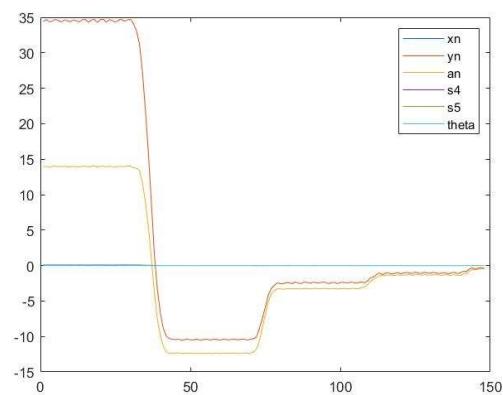
Kết quả thực nghiệm

Giá trị của các đặc trưng thị giác mong muốn s^* , chiều cao từ vật thể đến đầu công tác robot mong muốn, ma trận tương tác L của quá trình mô phỏng được thể hiện bên dưới. Sai số của quá trình thực nghiệm được thể hiện ở biểu đồ bên dưới. Từ biểu đồ ta có thể thấy sai số của các đặc trưng thị giác dần bị triệt tiêu và tiến về 0.

a_n	x_n	y_n	s_4	s_5	theta
6539475	347.32	238.6	0.00096	$3.75 \cdot 10^{-7}$	2.074

$$Z_d = 0.1856(m)$$

$$L = \begin{bmatrix} -1 & 0 & 0 & -1.029 \times 10^7 & 1.389 \times 10^7 & 3.473 \times 10^2 \\ 0 & -1 & 0 & -5.784 \times 10^6 & 1.029 \times 10^7 & -6.539 \times 10^6 \\ 0 & 0 & -1 & -5.209 \times 10^2 & 9.809 \times 10^6 & 0 \\ 0 & 0 & 0 & -1.43 \times 10^{-1} & 2.646 \times 10^{-1} & 0 \\ 0 & 0 & 0 & -2.901 \times 10^{-4} & 4.827 \times 10^{-4} & 0 \\ 0 & 0 & 0 & -4.205 \times 10^2 & -1.859 \times 10^2 & -1 \end{bmatrix}$$



*Link video thực nghiệm: <https://youtu.be/wBHwWFKJg6c>



7. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN ĐỀ TÀI

Kết quả đạt được.

- Xây dựng được hướng tiếp cận kỹ thuật điều khiển hồi tiếp Visual Servoing sử dụng bộ đặc trưng thị giác là image moments điều khiển robot hồi tiếp, nhận diện và gấp vật 2D có hình dạng phức tạp. Hoàn thành quá trình điều khiển hồi tiếp ảnh dưới thời gian 1 phút.
- Đề xuất được cách lọc nhiễu khi xử lý hình ảnh thu về từ camera để thu được hình ảnh đầu vào chất lượng cho giải thuật điều khiển.
- Thiết kế được đồ gá camera, đồ gá giác hút.
- Tìm hiểu, nghiên cứu về ngôn ngữ robot và giao tiếp được với robot bằng phương thức giao tiếp socket.
- Xây dựng được giao diện người dùng trực quan.

Hạn chế của luận văn.

- Khuyết điểm của giải thuật điều khiển Visual Servoing là chưa tối ưu hóa đường đi của robot. Khi tiến hành hồi tiếp robot di chuyển còn bị ngắt quãng, giựt chưa được liên tục.
- Thời gian còn khá chậm so với chuẩn công nghiệp.
- Khi vật thể đặt quá lệch so với mẫu được lấy ban đầu quá trình hồi tiếp sẽ mất nhiều thời gian hoặc bị sai.

Hướng phát triển đề tài.

- Thực nghiệm thêm hồi tiếp robot về vị trí thả vật để khi thả vật được chính xác hơn.
- Giảm thời gian hồi tự bằng cách tăng vận tốc robot và nội suy đường đi thay vì di chuyển point to point như hiện tại để robot di chuyển được liên tục mà không bị ngắt quãng.
- Áp dụng thêm các giải thuật xử lý hình ảnh như *Watershed* để nhận diện được các vật thể nằm chồng lên nhau, tách riêng từng vật thể và điều khiển hồi tiếp đối với vật thể nằm trên cùng.



THANK YOU