TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA ĐIỆN – ĐIỆN TỬ

Châu Minh Lộc

THIẾT KẾ CÁNH TAY ROBOT

ĐỒ ÁN CHUYÊN NGÀNH

KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA ĐIỆN – ĐIỆN TỬ

SINH VIÊN
CHÂU MINH LỘC-42000407

THIẾT KẾ CÁNH TAY ROBOT

ĐỒ ÁN CHUYÊN NGÀNH
KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA

Người hướng dẫn
ThS. Thiều Quang Trí

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023

LỜI CẢM ƠN

Em xin chân thành cảm ơn thầy Thiều Quang Trí đã hỗ trợ em trong quá trình thực hiện đồ án chuyên ngành. Em cảm ơn sự chỉ dẫn tận tình của thầy, thầy luôn sẵn sàng trả lời những thắc mắc của em cũng như hướng dẫn em thực hiện đồ án chuyên ngành một cách chi tiết và phù hợp với yêu cầu môn học do nhà trường đề ra. Tuy thầy có nhiều việc bận nhưng vẫn thu xếp thời gian để hỗ trợ cho tụi em trong quá trình thực hiện đồ án lần này. Em mong sau này còn vẫn có thể nhận được sự hướng dẫn của thầy trong đồ án tốt nghiệp.

TP. Hồ Chí Minh, ngày 04 tháng 12 năm 2023

Tác giả Châu Minh Lộc

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH
TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khoa học của ThS Thiều Quang Trí. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong Đồ án hệ thống nhúng còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung Đồ án chuyên ngành của mình. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

TP. Hồ Chí Minh, ngày 04 tháng 12 năm 2023

Tác giả

(ký tên và ghi rõ họ tên)

TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA ĐIỆN –ĐIỆN TỬ

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

PHIẾU THEO DÕI TIẾN ĐỘ THỰC HIỆN ĐỒ ÁN CHUYÊN NGÀNH

Họ tên sinh viên: Châu Minh Lộc MSSV: 42000407
Ngành học: Kỹ thuật điều khiển và tự động hóa
Địa chỉ liên lạc: 2056/1/6A Huỳnh Tấn Phát, TP.HCM Đơn vị: ĐH Tôn Đức Thắng
Họ tên GVHD: Thiều Quang Trí
Tên đề tài: Thiết kế cánh tay robot

Tuần/Ngày	**Khối lượng**		**GVHD ký**
	Đã thực hiện	**Tiếp tục thực hiện**	
Tuần 1 11-17/09/2023	- Thống nhất đề tài - Thiết kế lịch trình trong 10 tuần - Tìm hiểu đề tài - Viết chương I của bài báo cáo	- Tìm hiểu cấu trúc robot 3 bậc - Lắp ráp và gán tọa độ cho tay robot trong phần mềm SolidWorks - Tạo Simulink cho cánh tay robot trong phần mềm Matlab	
Tuần 2 18-24/09/2023	- Tìm hiểu cấu trúc robot 3 bậc - Lắp ráp và gán tọa độ cho tay robot trong phần mềm SolidWorks - Tạo Simulink cho cánh tay robot	- Thiết kế giao diện người dùng GUIDE trong Matlab - Tìm bảng DH cho cánh tay robot - Giải bài toán động học thuận cho cánh tay robot	

	trong phần mềm Matlab		
Tuần 3 25/09/2023- 01/10/2023	- Thiết kế giao diện người dùng GUIDE trong Matlab - Tìm bảng DH cho cánh tay robot - Giải bài toán động học thuận cho cánh tay robot	- Giải bài toán động học nghịch cho cánh tay robot - Thi công mô hình robot	
Tuần 4 02-08/10/2023	- Giải bài toán động học nghịch cho cánh tay robot - Thi công mô hình robot	- Tìm hiểu cấu trúc, nguyên lý hoạt động, sơ đồ chân của các linh kiện - Thiết kế mạch nguyên lý - Thi công, lắp đặt phần cứng - Viết chương II của bài báo cáo - Báo cáo 50% đồ án chuyên ngành	
Tuần 5 09-15/10/2023	- Tìm hiểu cấu trúc, nguyên lý hoạt động, sơ đồ chân của các linh kiện	- Lưu đồ giải thuật - Viết chương trình điều khiển cho vi điều khiển Arduino Uno	

	- Thiết kế mạch nguyên lý - Thi công, lắp đặt phần cứng - Viết chương II của bài báo cáo - Báo cáo 50% đồ án chuyên ngành	- Viết chương III của bài báo cáo	
Kiểm tra giữa kỳ	Đánh giá khối lượng hoàn thành……..% Đồ án chuyên ngành		
Tuần 6 30/10/2023- 05/11/2023	- Lưu đồ giải thuật - Viết chương trình điều khiển cho vi điều khiển Arduino Uno - Viết chương III của bài báo cáo	- Kết nối giữa phần mềm Matlab trên máy tính với vi điều khiển Arduino Uno thông qua giao tiếp Serial - Viết chương IV của bài báo cáo	
Tuần 7 06-12/11/2023	- Kết nối giữa phần mềm Matlab trên máy tính với vi điều khiển Arduino Uno thông qua giao tiếp Serial - Viết chương IV của bài báo cáo	- Kết hợp cánh tay robot với xử lý ảnh Viết chương V của bài báo cáo	

Tuần 8 13-19/11/2023	- Kết hợp cánh tay robot với xử lý ảnh - Viết chương V của bài báo cáo	- Kết hợp cánh tay robot với xử lý ảnh Hoàn thiện báo cáo	
Tuần 9 20-26/11/2023	- Kết hợp cánh tay robot với xử lý ảnh - Hoàn thiện báo cáo	- Trình bày poster Hoàn thiện đề tài	
Tuần 10 27/11/2023- 03/12/2023	- Trình bày poster - Hoàn thiện đề tài		
Nộp Đồ án chuyên ngành	Đã hoàn thành……..% Đồ án chuyên ngành		

THIẾT KẾ CÁNH TAY ROBOT

TÓM TẮT

Bài báo cáo về đề tài thiết kế cánh tay robot này bao gồm năm chương, tư liệu tham khảo và mục lục. Bên trong báo cáo cũng bao gồm các hình ảnh minh họa các liên kiện đã sử dụng, hình ảnh từ các phần mềm hỗ trợ đồ án và các bảng biểu về thông số kỹ thuật cũng như sơ đồ chân của các linh kiện sử dụng. Chương đầu tiên của báo cáo là giới thiệu về hệ thống bãi giữ xe tự động, mục đích, đối tượng, phạm vi nghiên cứu và dự kiến kết quả. Tiếp theo ở chương hai là về thiết kế và thi công gồm sơ đồ khối phần cứng và các linh kiện sử dụng, thông số kỹ thuật của các linh kiện và thiết kế mạch nguyên lý cũng như là thi công mô hình. Chương ba là giải thuật và điều khiển bao gồm mô tả hoạt động của hệ thống và giải thích lưu đồ giải thuật. Kế đến là chương bốn báo cáo về việc thực nghiệm gồm trình tự thao tác và kết quả thu được. Cuối cùng ở chương 5 là nêu ra các ưu nhược điểm và hướng phát triển trong tương lai của đề tài về đề tài bãi giữ xe tự động.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC CÁC CHỮ VIẾT TẮT

DH	Denavit – Hartenberg

CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1 Giới thiệu đề tài

Với sự phát triển của khoa học kỹ thuật thì vai trò và ứng dụng của cánh tay robot trong môi trường công nghiệp ngày càng quan trọng. Cùng với sự phát triển nhanh chóng của công nghệ đã sản xuất ra nhiều cánh tay robot linh hoạt và mạnh mẽ với chức năng thực hiện các nhiệm vụ khác nhau trong quá trình sản xuất và đời sống của con người.

Cánh tay robot đại diện cho sự tiến bộ đột phá trong quá trình tự động hóa trong sản xuất. Các cánh tay robot hiện nay được trang bị với nhiều bậc tự do tạo nên sự linh hoạt trong chuyển động, các cánh tay robot không chỉ thay thế lao động thủ công trong các tác vụ lặp đi lặp lại mà còn tăng độ chính xác cũng như năng suất. Các cánh tay robot được lắp đặt trong các dây chuyền sản xuất như SCARA, Delta, … mang lại sự linh hoạt và hiệu suất cho quy trình sản xuất.

Về ứng dụng của các cánh tay robot trong sản suất công nghiệp rất đa dạng từ lắp ráp các chi tiết cho ô tô, hàn các chi tiết thành phần đến kiểm tra chất lượng đầu cuối của sản phẩm. Ngoài ra trong quá trình sản xuất điện tử, cánh tay robot còn có khả năng trong công việc thao tác với những linh kiện điện tử có kích thước siêu nhỏ, lắp ráp các bo mạch điện tử và kiểm tra chất lượng bo mạch điện tử theo yêu cầu kỹ thuật cần thiết. Thêm vào đó, các cánh tay robot còn được tích hợp cảm biến và trí tuệ nhân tạo nhằm tạo ra cánh tay robot thông minh. Cảm biến giúp các cánh tay robot nhận biết môi trường xung quanh cũng như tương tác với các đối tượng khác nhau và công nghệ trí tuệ nhân tạo thì cho phép các cánh tay robot tự học, tối ưu hóa hiệu suất trong quá trình theo thời gian.

1.2 Mục đích nghiên cứu

Đề tài của em bao gồm mô hình cách tay robot, vi điều khiển, động cơ bước, các driver điều khiển động cơ bước, các công tắc hành trình và bộ nguồn 24V-10A. Cánh tay robot sẽ được điều khiển bởi vi điều khiển và được kết nối mới máy tính để giao tiếp nhận và truyền dữ liệu giữa tay robot và máy tính. Camera sẽ chụp và xử lý ảnh báo về tọa độ tâm của vật cho máy tính, máy tính sẽ tính toán bài toán động học nghịch tìm ra các góc xoay của các khớp thông qua phần mềm Matlab và sẽ gửi dữ liệu các góc xoay lên cổng Serial, vi điều khiển sẽ nhận dữ liệu trên cổng serial và xuất tín hiệu điều khiển các driver, các driver điều khiển các động cơ bước để điều khiển các góc xoay của cánh tay robot. Giới hạn của các góc xoay của cánh tay robot sẽ được giới hạn bởi các công tắc hành trình. Sự chuyển động của cánh tay robot sẽ được mô phỏng lại thông qua công cụ Simulink trên phần mềm Matlab. Mọi sự điều khiển sẽ được thực hiện thông qua bảng thiết kế giao diện người dùng GUIDE của Matlab do người dùng tạo ra.

1.3 Đối tượng nghiên cứu

- Vi điều khiển Arduino Uno
- Shield mở rộng CNC V3 cho Arduino Uno
- Module driver A4988
- Động cơ bước NEMA 17 size 42 1.8°
- Động cơ bước 28byj-48 12V
- Công tắc hành trình V-151-1C25
- Nguồn tổ ong 12V-10A

1.4 Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài cánh tay robot giới hạn cho các cánh tay cỡ nhỏ, các động cơ bước hoạt động với tốc độ chậm, tốc độ truyền nhận dữ liệu giữa vi điều khiển và phần mềm Matlab thông qua cổng Serial với độ trễ là từ 0.01s tới 0.02s. Việc giới hạn các góc xoay của cánh tay robot sẽ bằng các công tắc hành trình. Cơ

cấu chấp hành của cánh tay robot là tay kẹp và được điều khiển bởi động cơ bước thông qua truyền động bánh răng, chuyển từ chuyển động xoay tròn thành chuyển động trượt. Mọi sự điều khiển sẽ được thực hiện thông qua bảng thiết kế giao diện người dùng GUIDE của Matlab do người dùng tạo ra.

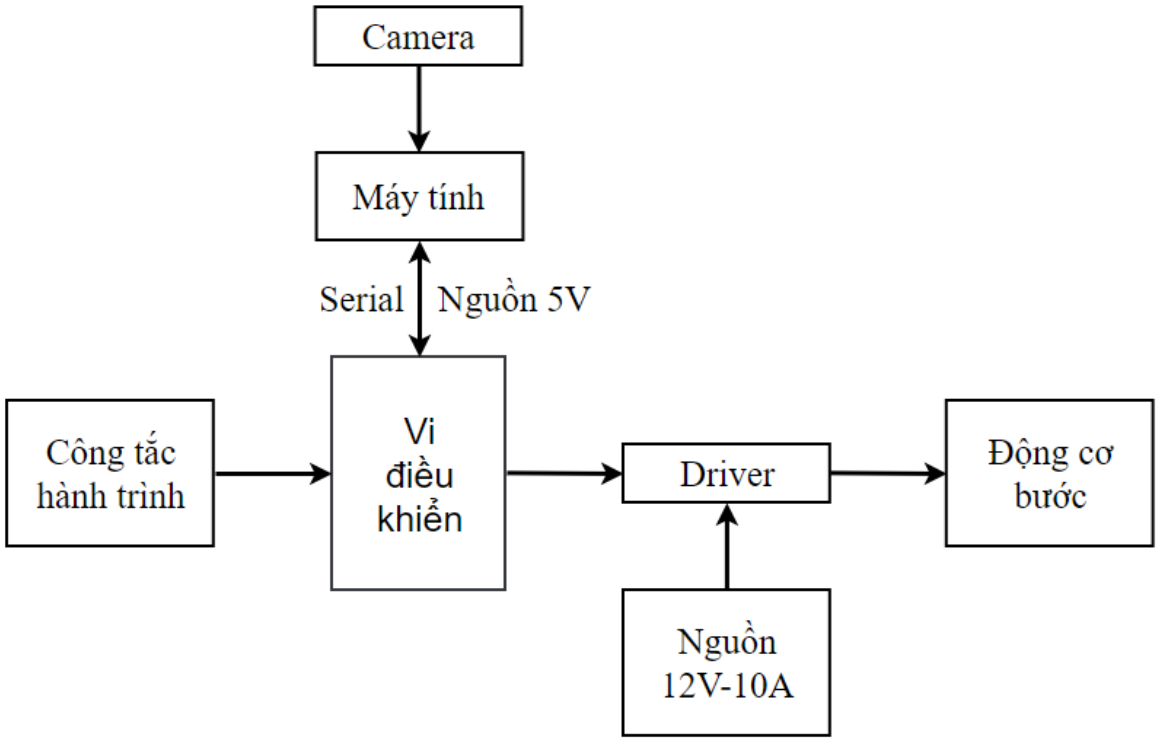
1.5 Dự kiến kết quả

Mô hình sẽ bao gồm mô hình cách tay robot, vi điều khiển, động cơ bước, các driver điều khiển động cơ bước, các công tắc hành trình và bộ nguồn 24V-10A. Mô hình cánh tay robot sẽ có bốn công tắc hành trình cho bốn động cơ bước. Hệ thống sẽ nhận tọa độ tâm của vật và điều khiển cánh tay robot đến vị trí vật để gắp vật và để vào các hộp tương đương với các màu sắc khác nhau. Cuối cùng mọi thông tin về vị trí, tọa độ sẽ được điều khiển, hiển thị và mô phỏng trên phần mềm Matlab.

CHƯƠNG 2. THIẾT KẾ VÀ THI CÔNG

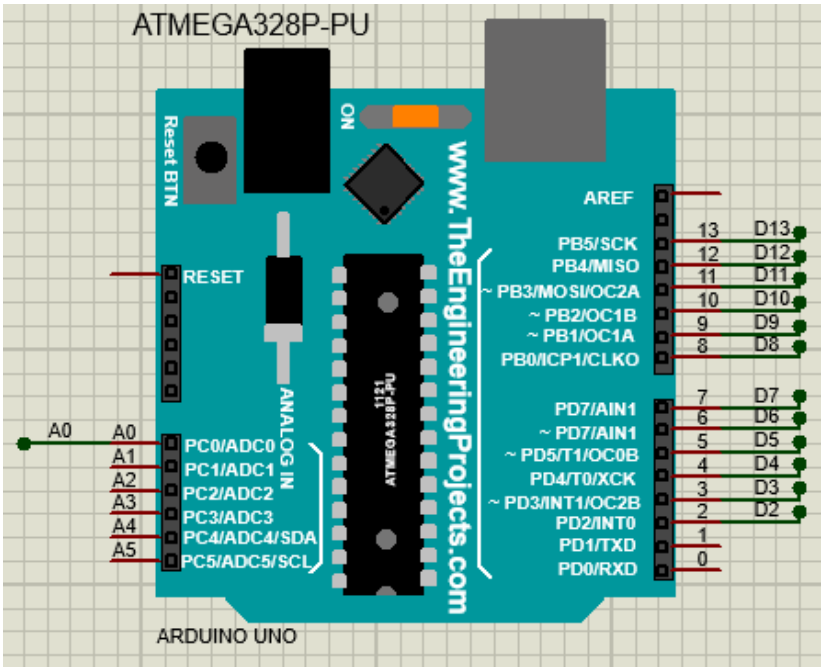
2.1 Sơ đồ khối của hệ thống

Dưới đây là sơ đồ khối của hệ thống với trung tâm là khối vi điều khiển, các khối đầu vào gồm khối nút nhấn, khối cảm biến hồng ngoại, khối đồng hồ thời gian thực, khối nguồn DC, và các khối ngõ ra bao gồm khối động cơ servo, khối màn hình LCD, khối còi báo cháy.



Hình 2-1. Sơ đồ khối của hệ thống

2.1.1 Khối vi điều khiển



Hình 2-2. Khối vi điều khiển

2.1.1.1 Chức năng khối vi điều khiển

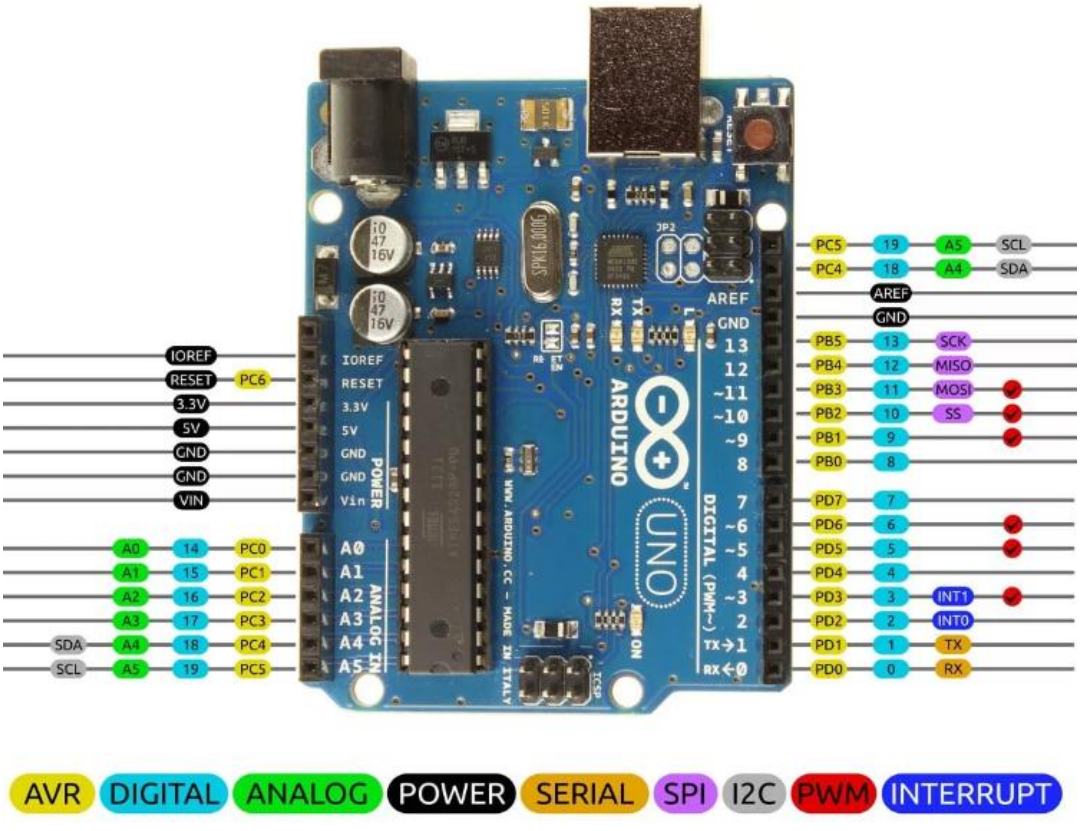
Arduino Uno tích hợp sẵn thạch anh với tần số là 16MHz nhằm làm nguồn tạo dao động cho chip ATMEGA328P. Khối vi điều khiển sẽ nhận dữ liệu từ máy tính trên cổng Serial baudrate 9600 để điều khiển các động cơ bước thông qua các driver A4988, đồng thời sẽ xuất dữ liệu lên cổng Serial baudrate 9600 để phần mềm Matlab trong máy tính nhận dữ liệu đó rồi mô phỏng lại vị trí, chuyển động của cánh tay robot.

2.1.1.2 Sơ đồ chân của Arduino Uno

Arduino Uno là board vi điều khiển phổ biến sử dụng chip xử lý ATMEGA328P với 32KB bộ nhớ Flash, 1KB EEPROM và 2KB SRAM. Arduino Uno sử dụng cổng USB type B, cổng Serial với baudrate là 9600, I2C, SPI nên Arduino Uno có thể kết nối với nhiều linh kiện khác nhau. Điện áp hoạt động của Arduino Uno là 5VDC. Phần mềm dùng để lập trình cho Arduino Uno là Arduino IDE dễ dàng sử dụng cho người mới bắt đầu với thao viết và tác nạp chương trình nhanh chóng. Ngoài ra cộng

đồng hỗ trợ rộng rãi với nhiều thư viện hỗ trợ và mã nguồn mở, tạo sự linh hoạt và dễ dàng trong quá trình làm việc với board vi điều khiển Arduino Uno.

Hình 2-3. Hình ảnh thực tế của vi điều khiển Arduino Uno



Hình 2-4. Sơ đồ chân của vi điều khiển Arduino Uno

Bảng 2-1: Sơ đồ chân của vi điều khiển Arduino Uno

Tên chân	Mô tả
RX <- 0	Chân Digital nhưng Arduino Uno chủ yếu sử dụng chần này cùng chân TX -> 1 nhằm truyền dữ liệu trong UART TTL truyền thông nối tiếp.
TX -> 1	Chân Digital nhưng Arduino Uno chủ yếu sử dụng chần này cùng chân RX <- 0 nhằm truyền dữ liệu trong UART TTL truyền thông nối tiếp.
2	Chân Digital và hỗ trợ chức năng ngắt ngoài INT0.
~3	Chân Digital và hỗ trợ chức năng ngắt ngoài INT1, điều khiển độ rộng xung PWM.
4	Chân Digital.
~5	Chân Digital và hỗ trợ chức năng điều khiển độ rộng xung PWM.
~6	Chân Digital và hỗ trợ chức năng điều khiển độ rộng xung PWM.
7	Chân Digital.
8	Chân Digital.
~9	Chân Digital và hỗ trợ chức năng điều khiển độ rộng xung PWM.
~10	Chân Digital và hỗ trợ chức năng điều khiển độ rộng xung PWM. Chân này còn dùng để giao tiếp SPI (giao diện ngoại vi nối tiếp) bằng thư viện SPI.
~11	Chân Digital và hỗ trợ chức năng điều khiển độ rộng xung PWM. Chân này còn dùng để giao tiếp SPI (giao diện ngoại vi nối tiếp) bằng thư viện SPI.
12	Chân Digital và dùng để giao tiếp SPI (giao diện ngoại vi nối tiếp) bằng thư viện SPI.

13	Chân Digital và dùng để giao tiếp SPI (giao diện ngoại vi nối tiếp) bằng thư viện SPI.
GND	Chân GND (nối đất).
AREF	Chân tham chiếu tương tự trên các chân A0-A5.
IOREF	Chân tham chiếu điện áp đầu vào của các chân A0-A5.
RESET	Chân reset vi điều khiển.
3.3V	Chân cung cấp điện áp 3.3VDC.
5V	Chân cung cấp điện áp 5VDC.
GND	Chân GND (nối đất).
GND	Chân GND (nối đất).
VIN	Chân cấp điện áp vào vi điều khiển.
A0	Chân Analog
A1	Chân Analog
A2	Chân Analog
A3	Chân Analog
A4	Chân Analog và hỗ trợ chức năng giao tiếp I2C cùng với chân A5.
A5	Chân Analog và hỗ trợ chức năng giao tiếp I2C cùng với chân A4.

2.1.1.3 Thông số kỹ thuật Arduino Uno

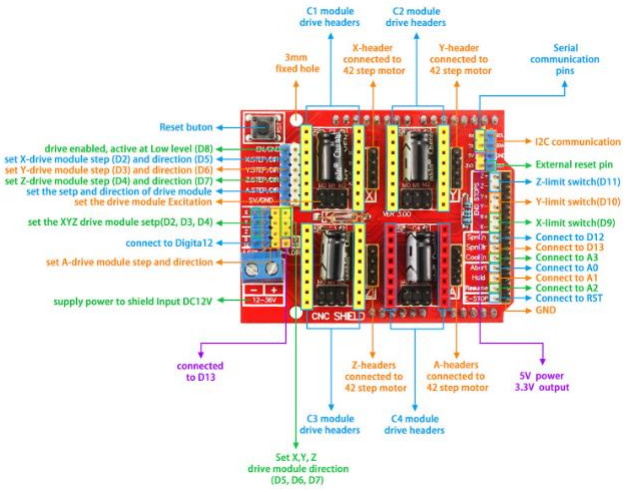
Bảng 2-2: Thông số kỹ thuật của vi điều khiển Arduino Uno

Loại CPU	ATMEGA328P
Bộ nhớ Flash	32 KB
SRAM	2 KB
EEPROM	1 KB
Số lượng chân vi điều khiển	28 chân
Thạch anh tích hợp	16 MHz
Đầu vào/ra kỹ thuật số	14 chân

Số chân hỗ trợ PWM	6 chân
Đầu vào/ra tương tự	6 chân
Hỗ trợ giao tiếp UART	1 module
Hỗ trợ giao tiếp SPI	1 module
Hỗ trợ giao tiếp I2C	1 module
Đèn LED tích hợp	Có
Dòng điện cho mỗi chân I/O	20mA
Điện áp hoạt động	5V

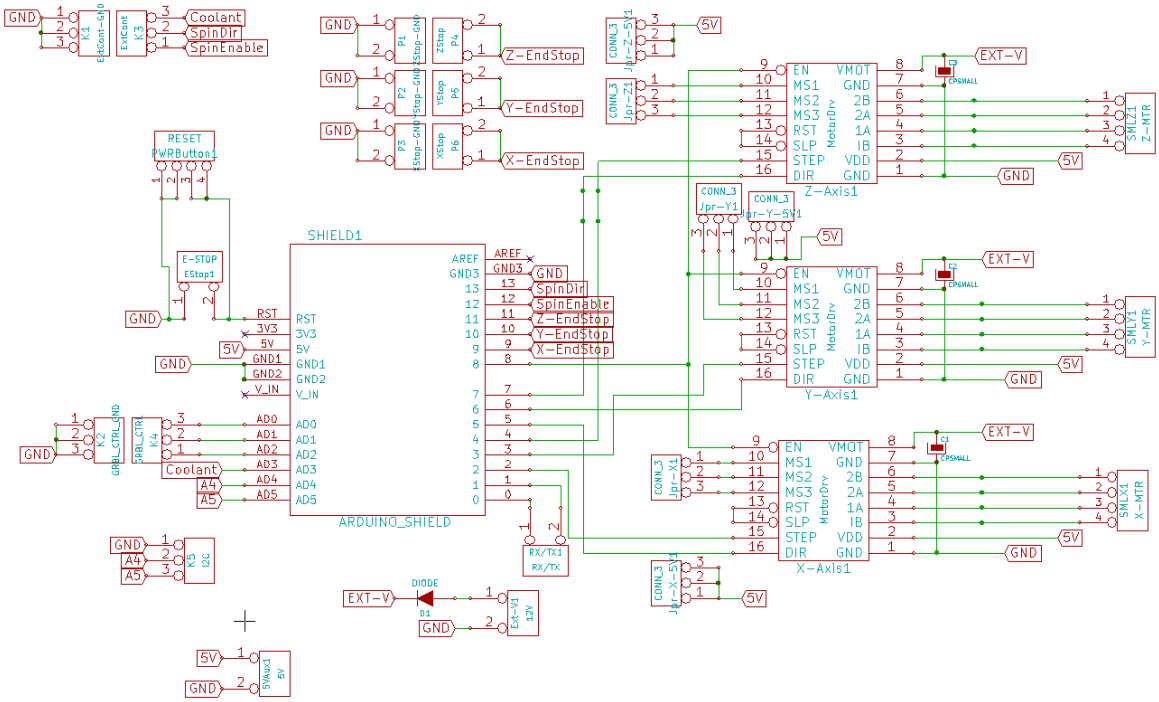
2.1.1.4 Shield CNC V3

Để giảm bớt dây dẫn thì em sử dụng bo mở rộng điều khiển các module driver dành cho Arduino Uno. Bo Shield CNC V3 sẽ tích hợp thêm các tụ lọc cho các module driver của động cơ bước nhằm giảm bớt nhiễu trong quá trình truyền xung điều khiển các động cơ bước. Ngoài ra bo còn có các hàng rào ra chân loại Male giúp kết nối giữa các chân trong hệ thống dễ dàng và không cần nối dây dài dẫn đến đỡ rối cho mô hình. Cuối cùng bo mở rộng này còn có cổng cấp điện áp đầu vào để điều khiển các module driver từ 12VDC tới 36VDC và có cầu chì bảo vệ.



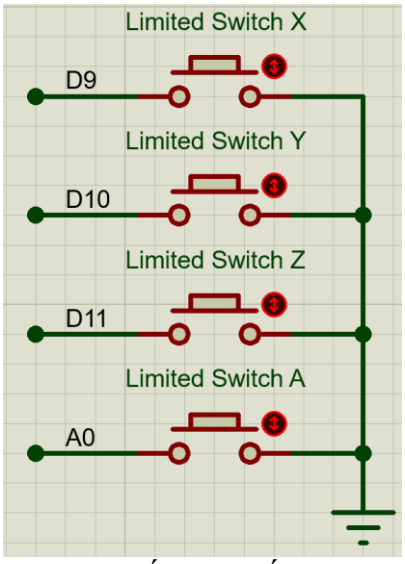
Hình 2-5. Hình ảnh thực tế của Shield CNC V3

Sơ đồ nguyên lý của bo mở rộng Shield CNC V3 từ nhà sản xuất.



Hình 2-6. Sơ đồ nguyên lý của Shield CNC V3

2.1.2 Khối công tắc hành trình



Hình 2-7. Khối công tắc hành trình

2.1.2.1 Chức năng khối công tắc hành trình

Ở khối công tắc hành trình công tắc được nối với điện trở kéo lên tích hợp sẵn trong Arduino Uno có giá trị từ 20 kΩ tới 50kΩ rồi nối với nguồn 5V và chân còn lại

thì nối với GND. Các công tắc này có chức năng giới hạn góc xoay tối đa và tối thiểu của cánh tay robot. Khi nhấn nút thì tính hiệu trả về các chân I/O của vi điều khiển sẽ chuyển từ 1 sang 0 tương ứng với chuyển từ mức cao xuống mức thấp.

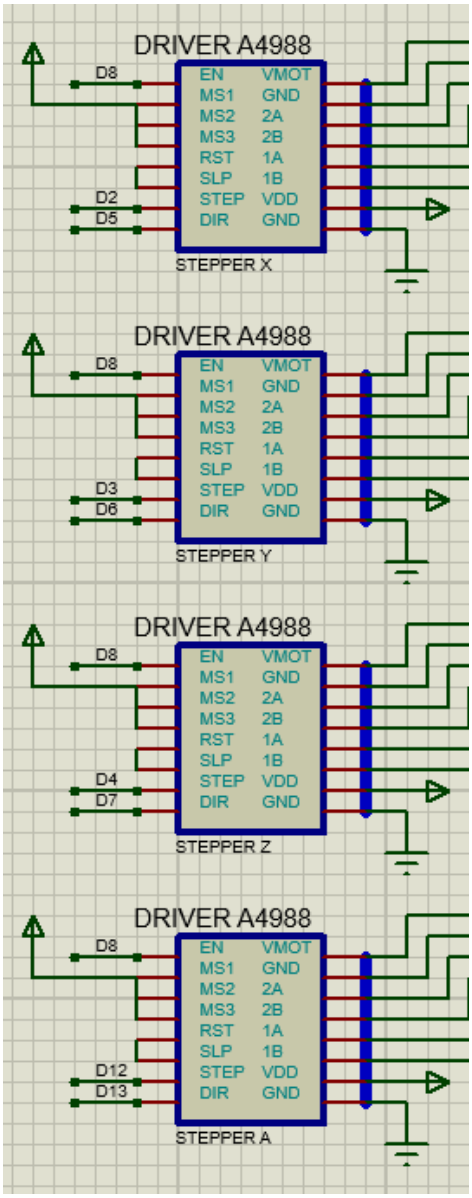
Hình 2-8. Hình ảnh thực tế của công tắc hành trình

Các công tắc hành trình được nối với vi điều khiển Arduino Uno như sau:

Bảng 2-3: Sơ đồ đấu nối của công tắc hành trình

Tên công tắc hành trình	Chân của Arduino Uno
Limited Switch X	~9
Limited Switch Y	~10
Limited Switch Z	~11
Limited Switch A	A0

2.1.3 Khối driver



Hình 2-9. Khối driver

2.1.3.1 Chức năng khối driver

Ở khối driver gồm module driver A4988 để điều khiển bốn động cơ bước tương ứng với các khớp chuyển động và cơ cấu chấp hành tay kẹp của cánh tay robot.

Vi điều khiển Arduino Uno điều khiển các động cơ bước thông qua các module driver A4988 bằng các chân Enable, Step và Dir của module. Các chân của các module được nối với vi điều khiển qua các chân như sau:

Bảng 2-4: Sơ đồ đấu nối của module driver A4988

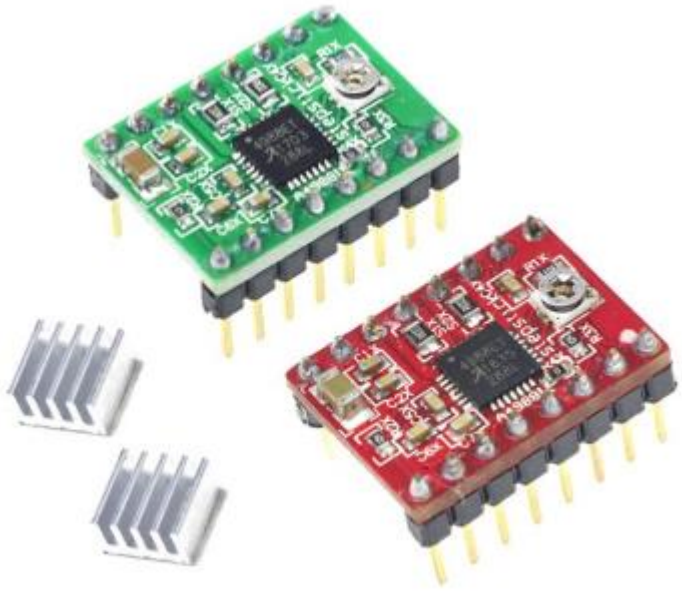
Tên chân	Chân của Arduino Uno
EN X	8
STEP X	2
DIR X	~5
EN Y	8
STEP Y	~3
DIR Y	~6
EN Z	8
STEP Z	4
DIR Z	7
EN A	8
STEP A	12
DIR A	13

2.1.3.2 Module driver A4988

Module driver A4988 là module điều khiển động cơ bước thường được sử dụng trong các ứng dụng như máy in 3D, máy CNC,... Nó có thể hoạt động ở mức điện áp rộng từ 8V đến 35V với khả năng điều khiển động cơ bước unipolar hay bipolar.

Module này còn hỗ trợ điều khiển động cơ bước ở chế độ vi bước (MicroStep) nhằm điều khiển chính xác hơn và giảm độ rung của động cơ. Ngoài ra còn cung cấp chức năng bảo vệ quá nhiệt giữ cho module an toàn trong khi làm việc ở nhiệt độ cao.

Module driver A4988 có tác dụng dễ thấy nhất là giảm số chân cần dùng để điều khiển cho một động cơ bước từ sáu chân xuống còn hai chân là Step và Dir của module driver A4988. Module này được nối vào bo mở rộng Shield CNC V3 để đơn giản việc lắp đặt và hạn chế số lượng dây nối cần dùng.



Hình 2-10. Hình ảnh thực tế của module driver A4988

Hình 2-11. Sơ đồ chân của module driver A4988

Bảng 2-5: Sơ đồ chân của module driver A4988

Tên chân	Mô tả
ENABLE	Chân khởi động cho module.
MS1	Chân chọn chế độ điều khiển vi bước.
MS2	Chân chọn chế độ điều khiển vi bước.
MS3	Chân chọn chế độ điều khiển vi bước.
RESET	Chân khởi động lại module.

SLEEP	Chân cho động cơ vào chế độ tiết kiệm năng lương (tiêu thụ điện năng ít nhất).
STEP	Chân nhận bước từ vi điều khiển.
DIR	Chân quy định hướng quay của động cơ bước.
VMOT	Chân cấp điện cho động cơ bước.
GND	Nối GND (nối đất).
2B	Nối vào dây B của cuộn dây thứ 2 của động cơ bước.
2A	Nối vào dây A của cuộn dây thứ 2 của động cơ bước.
1A	Nối vào dây A của cuộn dây thứ 1 của động cơ bước.
1B	Nối vào dây B của cuộn dây thứ 1 của động cơ bước.
VDD	Chân cấp điện hoạt động cho module với điện áp 5VDC.
GND	Nối GND (nối đất).
VREF	Điều chỉnh điện áp trên biến trở để điều chỉnh dòng qua động cơ bước.

2.1.3.3 Thông số kỹ thuật module driver A4988

Bảng 2-6: Thông số kỹ thuật của module driver A4988

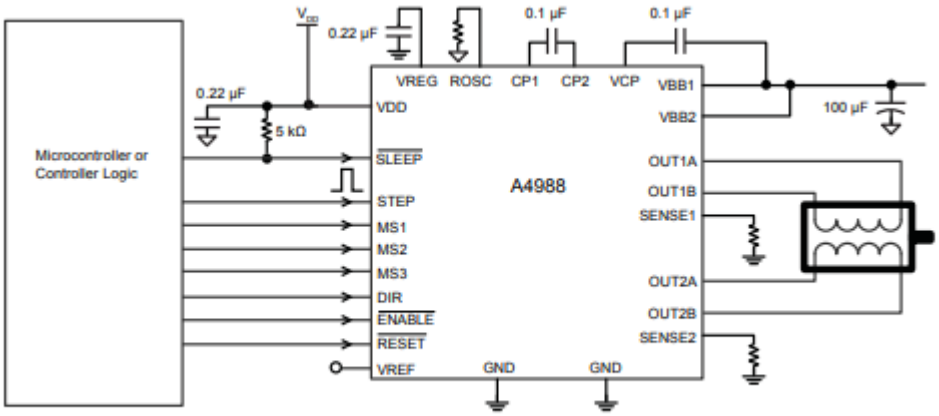
Điện áp cung cấp	8 tới 35VDC
Dòng ra tối đa	±2A
Điện áp tham chiếu	5.5VDC
Nhiệt độ hoạt động	-20 tới 85°C

Bảng 2-7: Bảng lựa chọn chế độ vi bước của module driver A4988

MS1	MS2	MS3	Microstep Resolution
L	L	L	Full Step
H	L	L	1/2 Step
L	H	L	1/4 Step
H	H	L	1/8 Step
H	H	H	1/16 Step

2.1.3.4 Sơ đồ nguyên lý module driver A4988

Dưới đây là sơ đồ nguyên lý của module driver A4988 từ nhà sản xuất.



Hình 2-12. Sơ đồ nguyên lý của module driver A4988

2.1.3.5 Tính toán dòng qua động cơ bước

Động cơ bước NEMA 17 size 42 1.8°thông thường sẽ có dòng cực đại là 800mA tương ứng là 0.8A nhưng ta thường chỉ sử dụng 80% dòng cực dại để đảm bảo an toàn cho động cơ bước nên dòng điện qua động cơ bước sẽ là:

$$I_{\text{Động cơ bước}} = 80\% \times I_{Max} = 80\% \times 0.8 = 0.64\ (A)$$

Điện trở trên module driver A4988 có giá trị là RS = 100mΩ tương ứng 0.1Ω. Theo công thức được cung cấp trong datasheet của module driver A4988 từ nhà sản xuất thì ta có công thức điều chỉnh điện áp trên biến trở VREF tương ứng với dòng điện mong muốn qua động cơ bước như sau:

$$I_{\text{Động cơ bước}} = \frac{V_{REF}}{8 \times R_S}$$

$$\Longrightarrow V_{REF} = 8 \times R_S \times I_{\text{Động cơ bước}} = 8 \times 0.1 \times 0.64 = 0.512\ (V)$$

Vậy ta vặn biến trở trên module cho đến khi đạt được giá trị điện áp rơi trên biến trở là 0.512V.

Động cơ bước 28BYJ-48 12V thông thường sẽ có dòng cực đại là 150mA tương ứng là 0.15A nhưng ta thường chỉ sử dụng 80% dòng cực dại để đảm bảo an toàn cho động cơ bước nên dòng điện qua động cơ bước sẽ là:

$$I_{Động\ cơ\ bước} = 80\% \times I_{Max} = 80\% \times 0.15 = 0.12\ (A)$$

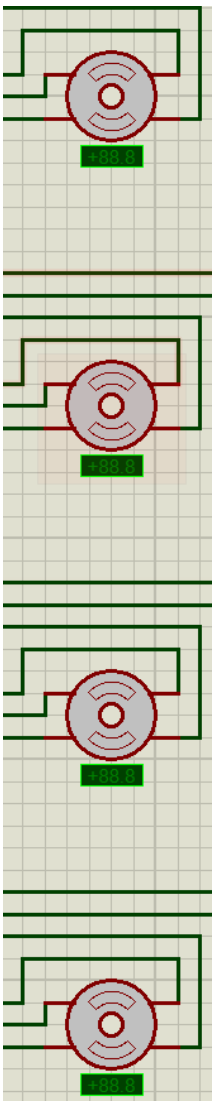
Điện trở trên module driver A4988 có giá trị là RS = 100mΩ tương ứng 0.1Ω. Theo công thức được cung cấp trong datasheet của module driver A4988 từ nhà sản xuất thì ta có công thức điều chỉnh điện áp trên biến trở VREF tương ứng với dòng điện mong muốn qua động cơ bước như sau:

$$I_{Động\ cơ\ bước} = \frac{V_{REF}}{8 \times R_S}$$

$$\Longrightarrow V_{REF} = 8 \times R_S \times I_{Động\ cơ\ bước} = 8 \times 0.1 \times 0.12 = 0.096\ (V)$$

Vậy ta vặn biến trở trên module cho đến khi đạt được giá trị điện áp rơi trên biến trở là 0.096V.

2.1.4 Khối động cơ bước



Hình 2-13. Khối động cơ bước

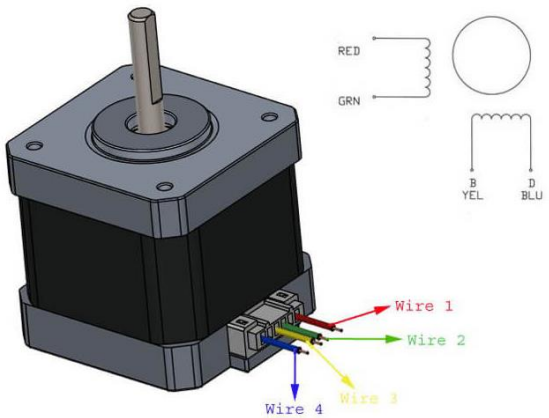
2.1.4.1 Chức năng khối động cơ bước

Ở khối động cơ bước sẽ bao gồm bốn động cơ bước đóng vai trò là các khớp chuyển động và cơ cấu chấp hành tay kẹp của cánh tay robot. Mỗi động cơ bước gồm bốn dây tức tương ứng với hai cặp dây. Bốn dây này sẽ được nối với các chân 1A, 1B, 2A và 2B của module driver A4988.

2.1.4.2 Động cơ bước NEMA 17 size 42 1.8°

Động cơ bước NEMA 17 size 42 1.8° là loại động cơ có kích thước tiêu chuẩn là 42mm x 42mm với độ phân giải là 1.8 độ/bước ở chế độ FullStep. Động cơ được thiết kế cho ứng dụng máy in 3D, máy CNC hoạt động hiệu quả hơn với các module driver điều khiển như DRV8825, A4988, … ở chế độ vi bước nhằm tăng độ chính xác cho hệ thống.

Hình 2-14. Hình ảnh thực tế của động cơ bước NEMA 17 size 42 1.8°



Hình 2-15. Sơ đồ dây của động cơ bước NEMA 17 size 42 1.8°

Bảng 2-8: Sơ đồ dây động cơ bước NEMA 17 size 42 1.8°

Màu dây	Mô tả
Đỏ	Dây A của cuộn dây thứ 1 (kí hiệu 1A)
Xanh lá	Dây B của cuộn dây thứ 1 (kí hiệu 1B)
Vàng	Dây A của cuộn dây thứ 2 (kí hiệu 2A)
Xanh dương	Dây B của cuộn dây thứ 1 (kí hiệu 2B)

2.1.4.3 Thông số kỹ thuật động cơ bước NEMA 17 size 42 1.8°

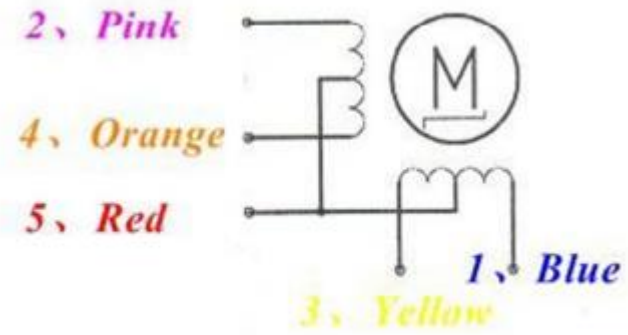
Bảng 2-9: Thông số kỹ thuật động cơ bước NEMA 17 size 42 1.8°

Điện áp cung cấp	6-14VDC (±10%)
Công suất tối đa	1W
Giới hạn tải	0.28kg
Giới hạn tải từ đầu trục	20N
Quán tính của rotor	57g/cm^3
Momen giữ tại dòng điện liên tục	0.35Nm
Momen giữ tại dòng điện đỉnh	0.5Nm
Dòng điện đầu ra liên tục	1.8A
Dòng điện đầu ra đỉnh (tùy ứng dụng)	3.5A
Chất liệu bánh răng	Kim loại
Góc bước	1.8°
Khối lượng	0.37kg

2.1.4.4 Động cơ bước 28byj-48 12V

Động cơ bước 28byj-48 12V là loại động cơ có kích thước tiêu chuẩn là 28mm x 19mm với độ phân giải là 5.625 độ/bước ở chế độ FullStep. Động cơ được thiết kế cho ứng dụng máy in 3D, máy CNC hoạt động hiệu quả hơn với các module driver điều khiển như ULN2003, DRV8825, A4988, … ở chế độ vi bước nhằm tăng độ chính xác cho hệ thống.

Hình 2-16. Hình ảnh thực tế của động cơ bước 28byj-48 12V



Hình 2-17. Sơ đồ dây động cơ bước 28byj-48 12V

Bảng 2-10: Sơ đồ dây động cơ bước 28byj-48 12V

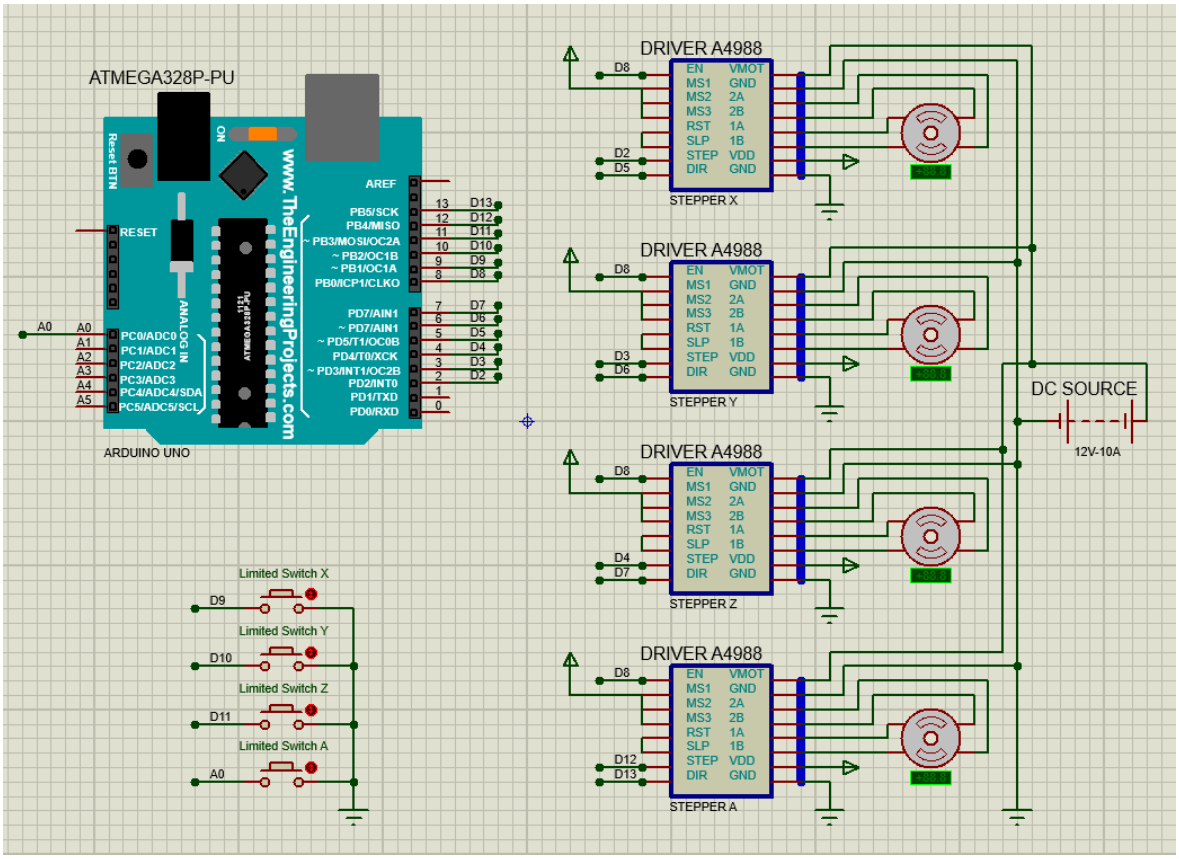
Màu dây	Mô tả
Xanh dương	Dây A của cuộn dây thứ 1 (kí hiệu 1A)
Vàng	Dây B của cuộn dây thứ 1 (kí hiệu 1B)
Hồng	Dây A của cuộn dây thứ 2 (kí hiệu 2A)
Cam	Dây B của cuộn dây thứ 1 (kí hiệu 2B)
Đỏ	Dây chung cho cả hai cuộn dây

2.1.4.5 Thông số kỹ thuật động cơ bước 28byj-48 12V

Bảng 2-11: Thông số kỹ thuật động cơ bước 28byj-48 12V

Điện áp cung cấp	12VDC
Momen xoắn	450gf.cm
Tần số hoạt động	100Hz
Số pha	4 pha
Chất liệu bánh răng	Kim loại
Góc bước	5.625°

2.2 Sơ đồ nguyên lý tổng quát



Hình 2-18. Sơ đồ nguyên lý tổng quát

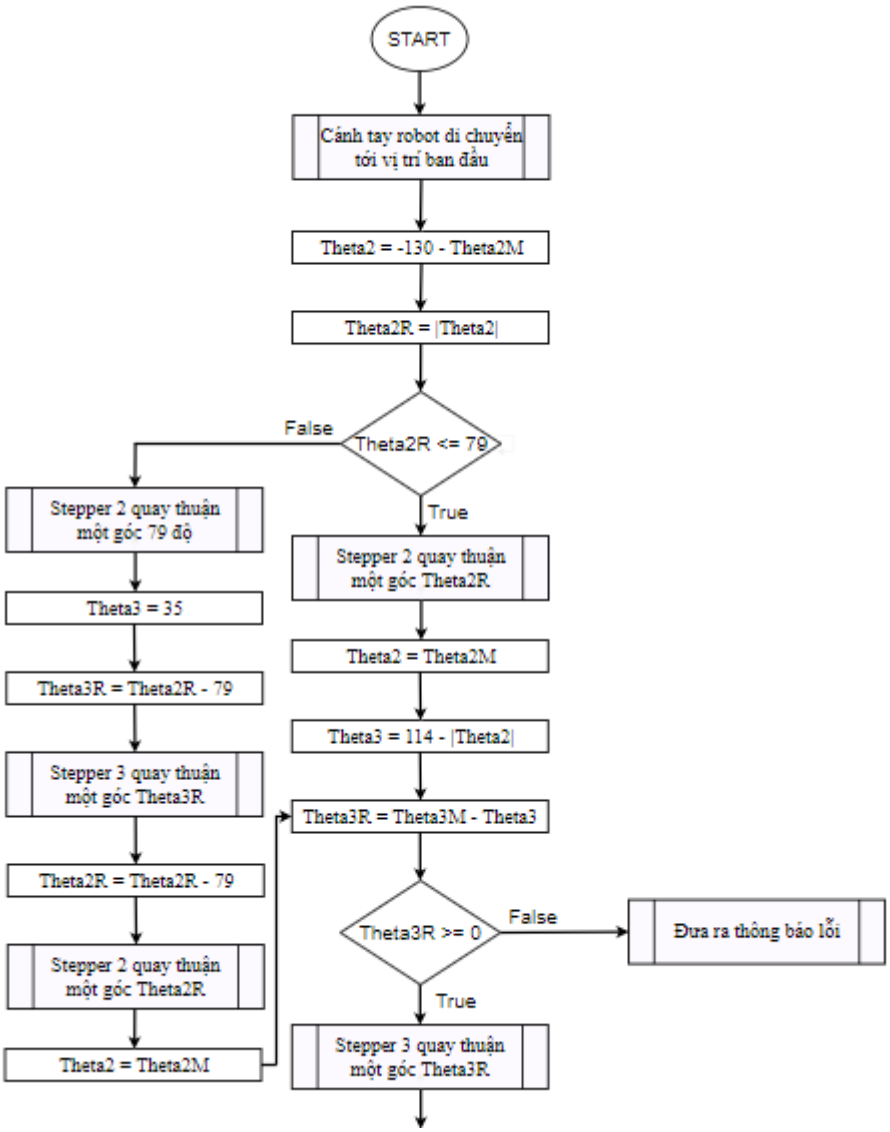
CHƯƠNG 3. GIẢI THUẬT VÀ ĐIỀU KHIỂN

3.1 Hoạt động của hệ thống

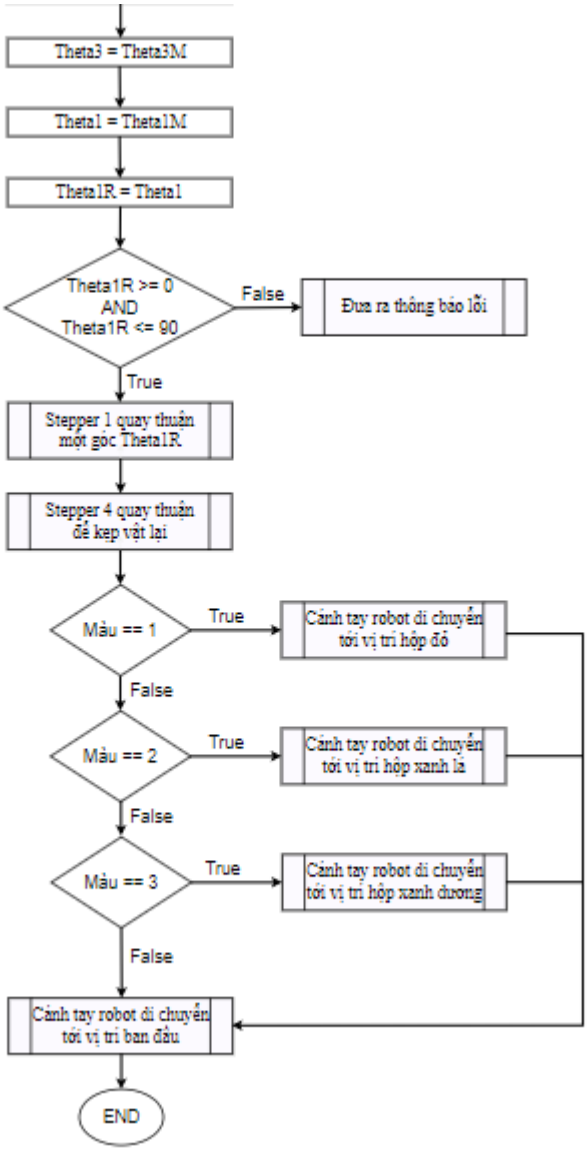
Hệ thống bắt đầu hoạt động nhấn các nút nhấn trên giao diện người dùng GUIDE (RUN, HOME, FK và IK). Khi nhấn nút nhấn RUN thì máy tính sẽ nhận tọa độ tâm của vật thể (xử lý bởi camera) về phần mềm Matlab trong máy tính. Matlab sẽ tính toán động học nghịch để chuyển tọa độ nhận được từ ảnh và chuyển thành các góc xoay của từng khớp của cánh tay robot. Máy tính sẽ gửi dữ liệu các góc này tới công cụ mô phỏng Simulink trong Matlab và cổng Serial. Vi điều khiển Arduino Uno sẽ nhận dữ liệu các góc từ cổng Serial và chạy chương trình điều khiển trong vi điều khiển Arduino Uno để điều khiển cánh tay robot tới vị trí tâm của vật. Mỗi khi vi điều khiển truyền một xung để điều khiển một động cơ bước thì vi điều khiển sẽ gửi dữ liệu góc của các động cơ bước (tức góc xoay của các khớp cánh tay robot) lên cổng Serial, phần mềm Matlab trong máy tính sẽ nhận dữ liệu trên Serial và truyền vào công cụ mô phỏng Simulink trong Matlab để mô phỏng cánh tay robot trên máy tính chạy theo giống với cánh tay thực tế. Khi cơ cấu chấp hành của cánh tay robot đến đúng vị trí mong muốn, tay kẹp sẽ đóng lại kẹp vào vật, sau đó cánh tay gắp vật đến các hộp khác nhau tương ứng với màu sắc của vật thể. Sau đó cánh tay quay lại vị trí ban đầu (vị trí HOME), nếu có thêm vật thể được đưa vào vùng xử lý ảnh thì cánh tay tiếp tục thực hiện quá trình trên hoặc cánh tay sẽ dừng hoạt động khi nhấn nút CLOSE trên giao diện người dùng GUIDE của Matlab.

3.2 Lưu đồ giải thuật trên vi điều khiển

Dưới đây là lưu đồ giải thuật của đề tài thiết kế cánh tay robot. Khi nhấn nút nhấn RUN trên giao diện người dùng GUIDE thì hệ thống sẽ bắt đầu hoạt động, sau đó Matlab gửi dữ liệu các góc xoay lên cổng Serial. Vi điều khiển nhận dữ liệu trên cổng Serial và điều khiển cánh tay robot.



Hình 3-1. Lưu đồ giải thuật trên vi điều khiển 1



Hình 3-2. Lưu đồ giải thuật trên vi điều khiển 2

3.3 Thiết kế mô hình

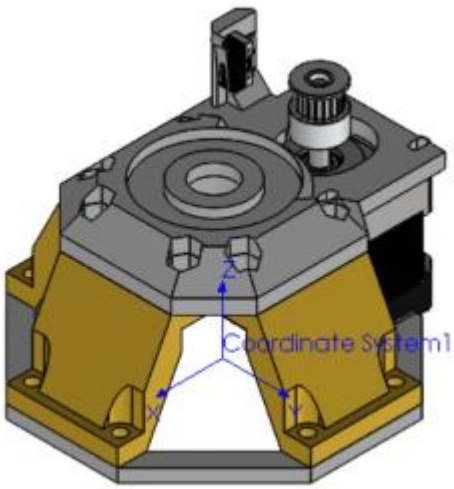
3.3.1 Mô hình cánh tay robot trên SolidWorks

Dưới đây là hình ảnh cánh tay robot được vẽ và lắp ráp trong môi trường SolidWorks với góc nhìn là isometic có trục Z hướng lên trên.

Hình 3-3. Mô hình cánh tay robot trong SolidWorks

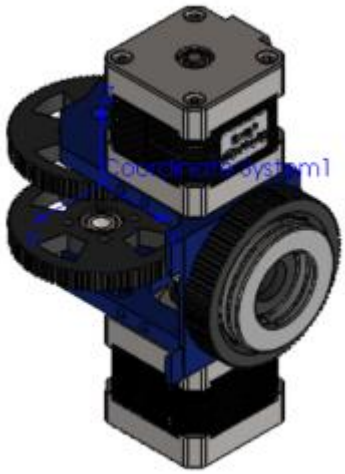
3.3.2 Gán hệ tọa độ cho các khớp của cánh tay robot

Dưới đây là hình ảnh gán hệ tạo độ cho chân đế của cánh tay robot trong môi trường SolidWorks với góc nhìn là isometic có trục Z hướng lên trên.



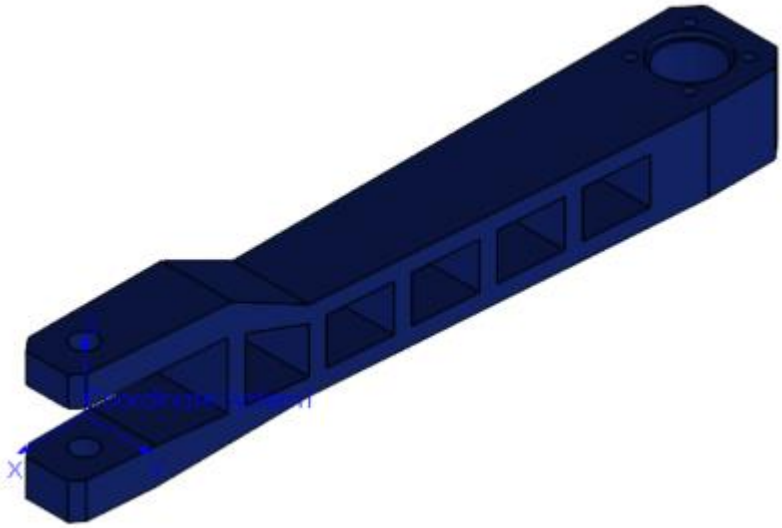
Hình 3-4. Hệ tọa độ chân đế của cánh tay robot

Dưới đây là hình ảnh gán hệ tạo độ cho phần thân của cánh tay robot trong môi trường SolidWorks với góc nhìn là isometic có trục Z hướng lên trên.



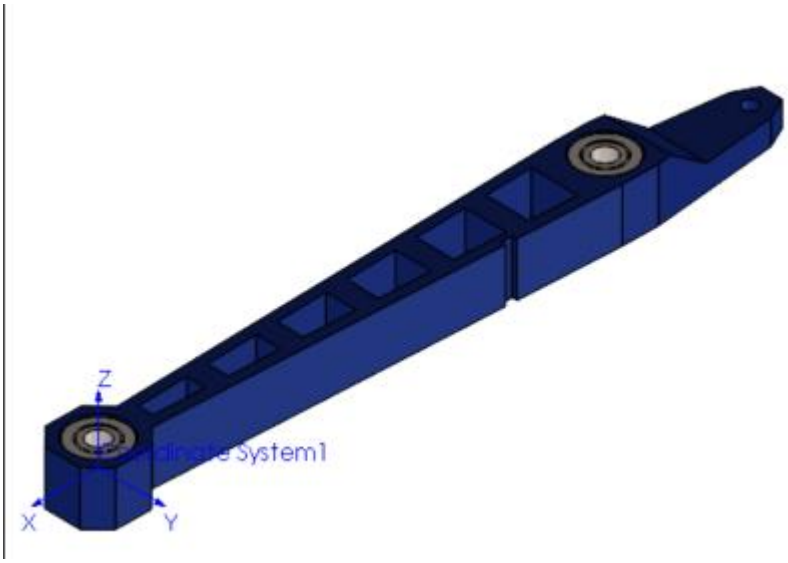
Hình 3-5. Hệ tọa độ phần thân đế của cánh tay robot

Dưới đây là hình ảnh gán hệ tạo độ cho trục nối thứ 1 (Link 1) của cánh tay robot trong môi trường SolidWorks với góc nhìn là isometic có trục Z hướng lên trên.



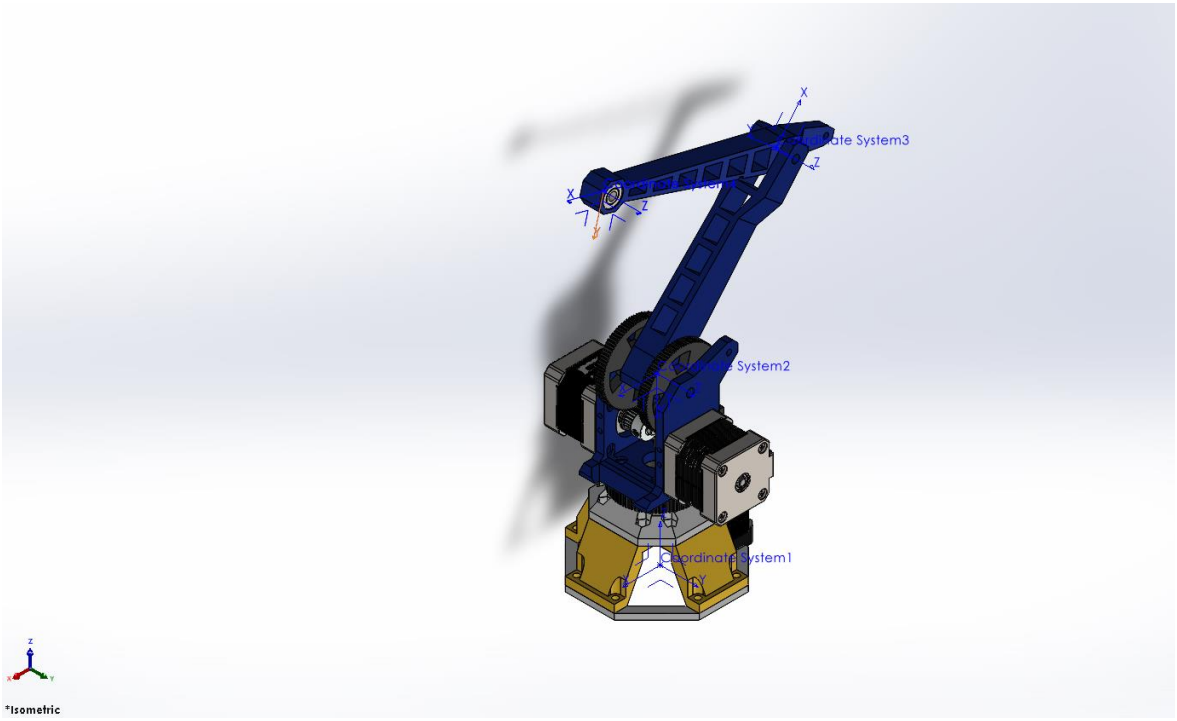
Hình 3-6. Hệ tọa độ trục nối thứ 1 của cánh tay robot

Dưới đây là hình ảnh gán hệ tạo độ cho trục nối thứ 2 (Link 2) của cánh tay robot trong môi trường SolidWorks với góc nhìn là isometic có trục Z hướng lên trên.



Hình 3-7. Hệ tọa độ trục nối thứ 2 của cánh tay robot

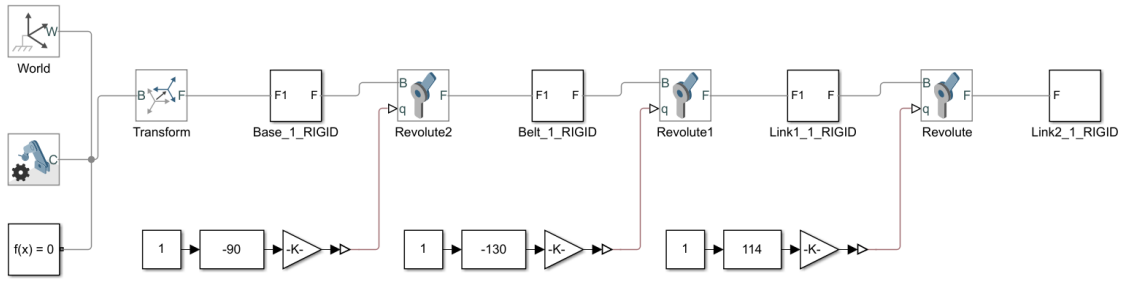
Dưới đây là hình ảnh gán hệ tạo độ hoàn chỉnh cho cánh tay robot trong môi trường SolidWorks với góc nhìn là isometic có trục Z hướng lên trên.



Hình 3-8. Hệ tọa độ hoàn chỉnh của cánh tay robot

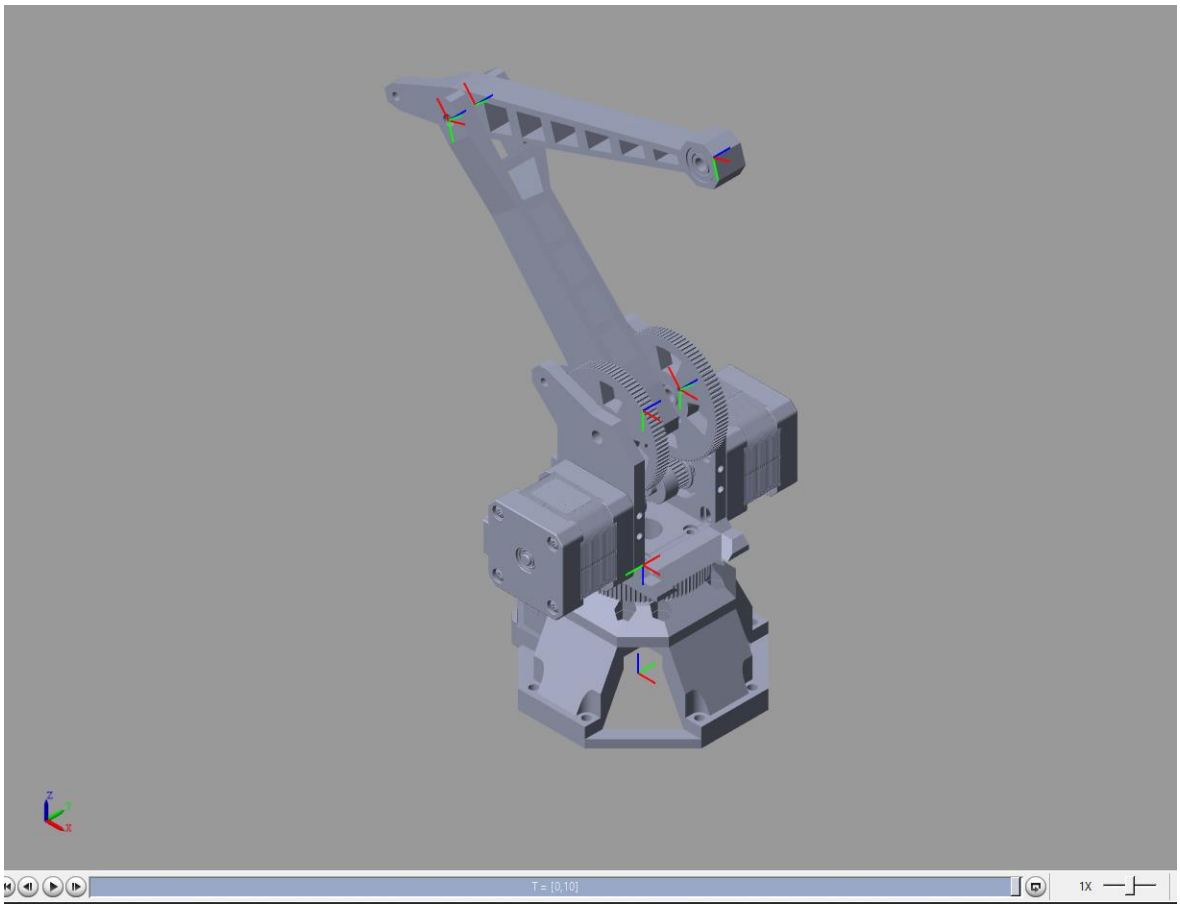
3.3.3 Tạo Simulink mô phỏng cho cánh tay robot

Dưới đây là hình ảnh mạch mô phỏng Simulink cho cánh tay robot trong phần mềm Matlab gồm các khối World, MechanismConfiguration, Solver Configuration, Transform, ReferenceFrame, Solid, Revolute, Constant, Slider Gain, Gain và Simulink-PS Converter.



Hình 3-9. Mạch mô phỏng Simulink của cánh tay robot

Dưới đây là hình ảnh đơn giản hóa của cánh tay robot trong môi trường Simulink của Matlab với góc nhìn là isometic có trục Z hướng lên trên.



Hình 3-10. Mô hình đơn giản hóa của cánh tay robot

3.3.4 Bảng DH (Denavit – Hartenberg của cánh tay robot

Dưới đây là bảng DH của cánh tay robot.

Bảng 3-1: Bảng DH của cánh tay robot

Joint	ai	αi	di	Θi	WorkSpace
1	2.83mm	-90°	130.2mm	Θ1	0 → 90°
2	140.01mm	0°	-1.25mm	Θ2	-130 → 10°
3	140mm	0°	0.75mm	Θ3	35 → 140°

Gripper	-	-	-	-	0 → 47mm

3.3.5 Giới hạn tọa độ làm việc của cánh tay robot

Dưới đây là bảng giới hạn tọa độ làm việc của cánh tay robot.

Bảng 3-2: Bảng giới hạn tọa độ làm việc của cánh tay robot

Trục	Min	Max
X	0.5mm	269.8604mm
Y	-0.5mm	269.8604mm
Z	-34.1116mm	277.5974mm

3.3.6 Bài toán động học thuận của cánh tay robot

Ta có ma trận tính toán chuyển đổi của hai hệ tọa độ liền kề trong một cánh tay robot theo phương pháp cổ điển (Classic) như sau:

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos(\Theta i) & -\cos(\alpha i) \times \sin(\Theta i) & \sin(\alpha i) \times \sin(\Theta i) & ai \times \cos(\Theta i) \\ \sin(\Theta i) & \cos(\alpha i) \times \cos(\Theta i) & -\sin(\alpha i) \times \cos(\Theta i) & ai \times \sin(\Theta i) \\ 0 & \sin(\alpha i) & \cos(\alpha i) & di \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Ma trận chuyển đổi của hệ tọa độ thứ nhất (khớp xoay thứ nhất) so với hệ tọa độ góc (hệ tọa độ 0) như sau:

$$ {}^{0}_{1}T = \begin{bmatrix} \cos(\Theta 1) & -\cos(\alpha 1) \times \sin(\Theta 1) & \sin(\alpha 1) \times \sin(\Theta 1) & a1 \times \cos(\Theta 1) \\ \sin(\Theta 1) & \cos(\alpha 1) \times \cos(\Theta 1) & -\sin(\alpha 1) \times \cos(\Theta 1) & a1 \times \sin(\Theta 1) \\ 0 & \sin(\alpha 1) & \cos(\alpha 1) & d1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Ma trận chuyển đổi của hệ tọa độ thứ hai (khớp xoay thứ hai) so với hệ tọa độ thứ nhất (khớp xoay thứ nhất) như sau:

$$ {}^{1}_{2}T = \begin{bmatrix} \cos(\Theta 2) & -\cos(\alpha 2) \times \sin(\Theta 2) & \sin(\alpha 2) \times \sin(\Theta 2) & a2 \times \cos(\Theta 2) \\ \sin(\Theta 2) & \cos(\alpha 2) \times \cos(\Theta 2) & -\sin(\alpha 2) \times \cos(\Theta 2) & a2 \times \sin(\Theta 2) \\ 0 & \sin(\alpha 2) & \cos(\alpha 2) & d2 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Ma trận chuyển đổi của hệ tọa độ thứ ba (khớp xoay thứ ba) hay hệ tọa độ đầu cuối của cánh tay robot so với hệ tọa độ thứ hai (khớp xoay thứ hai) như sau:

$$ {}^{2}_{3}T = \begin{bmatrix} \cos(\Theta 3) & -\cos(\alpha 3) \times \sin(\Theta 3) & \sin(\alpha 3) \times \sin(\Theta 3) & a3 \times \cos(\Theta 3) \\ \sin(\Theta 3) & \cos(\alpha 3) \times \cos(\Theta 3) & -\sin(\alpha 3) \times \cos(\Theta 3) & a3 \times \sin(\Theta 3) \\ 0 & \sin(\alpha 3) & \cos(\alpha 3) & d3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Ma trận chuyển đổi của hệ tọa độ đầu cuối của cánh tay robot so với hệ tọa độ góc như sau:

$$ {}^{0}_{3}T = {}^{0}_{1}T \times {}^{1}_{2}T \times {}^{2}_{3}T = \begin{bmatrix} r11 & r12 & r13 & Px \\ r21 & r22 & r23 & Py \\ r31 & r32 & r33 & Pz \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Từ ma trận trên ta suy ra được tọa độ đầu cuối của cánh tay robot như sau:

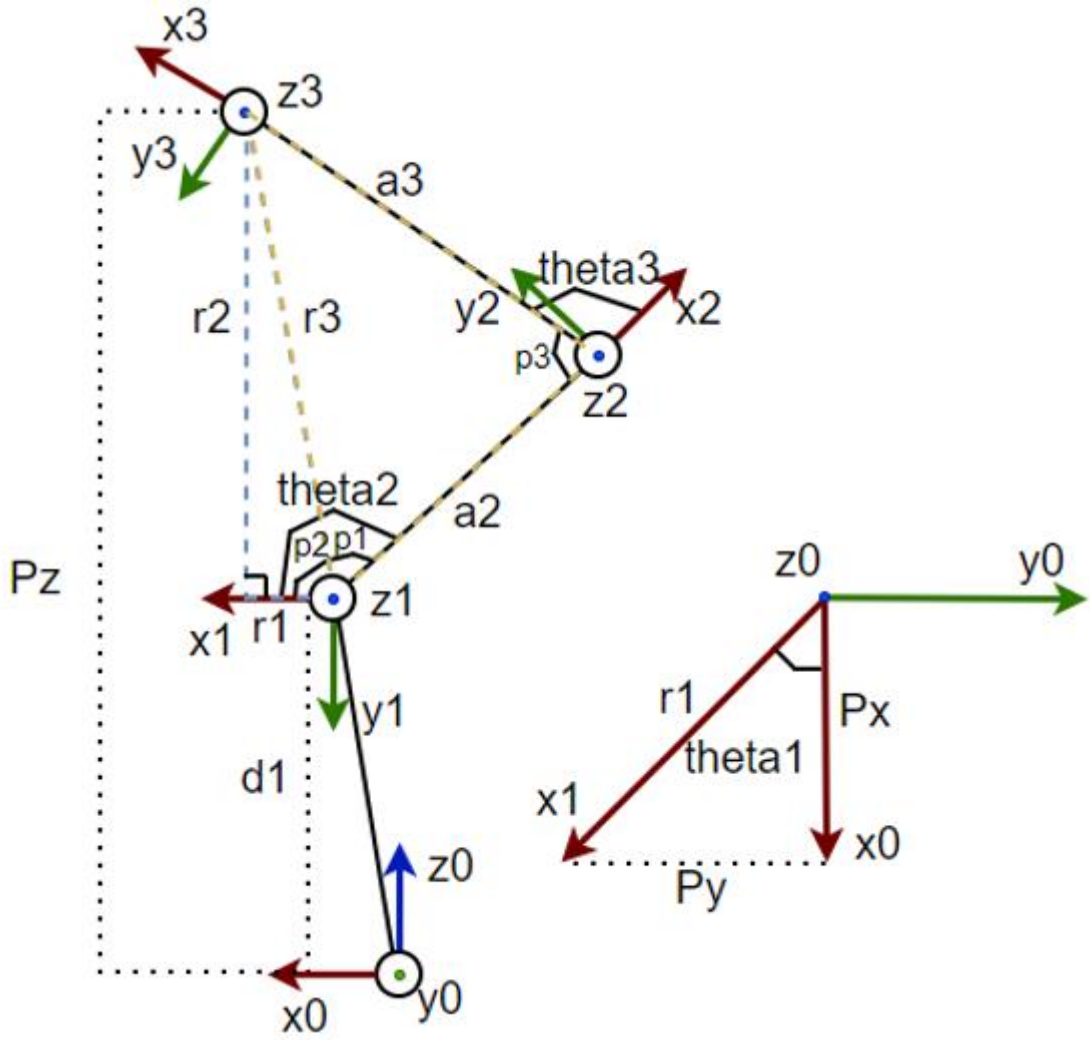
$$ Px = {}^{0}_{3}T(1,4) $$

$$ Py = {}^{0}_{3}T(2,4) $$

$$ Pz = {}^{0}_{3}T(3,4) $$

3.3.7 **Bài toán động học nghịch của cánh tay robot**

Dưới đây là hình ảnh hình chiếu cạnh (hình góc trái) và hình chiếu bằng (hình góc phải) của cánh tay robot nhằm phục vụ cho việc giải bài toán động học nghịch robot.



Hình 3-11. Hình chiếu cạnh và hình chiếu bằng của cánh tay robot

Tính toán góc theta 1 của cánh tay robot theo hình chiếu bằng như sau:

$$\theta 1 = \tan^{-1}\left(\frac{Py}{Px}\right)$$

Tính toán góc theta 2 của cánh tay robot theo hình chiếu cạnh như sau:

$$r1 = \sqrt{(Px)^2 + (Py)^2}$$

$$r2 = Pz - d1$$

$$\Phi 1 = cos^{-1}\left(\frac{(a3)^2 - (a2)^2 - (r3)^2}{-2 \times a2 \times r3}\right)$$

$$\Phi 2 = \tan^{-1}\left(\frac{r2}{r1}\right)$$

$$-\theta 2 = \Phi 1 + \Phi 2$$

$$\Longrightarrow \theta 2 = -(\Phi 1 + \Phi 2)$$

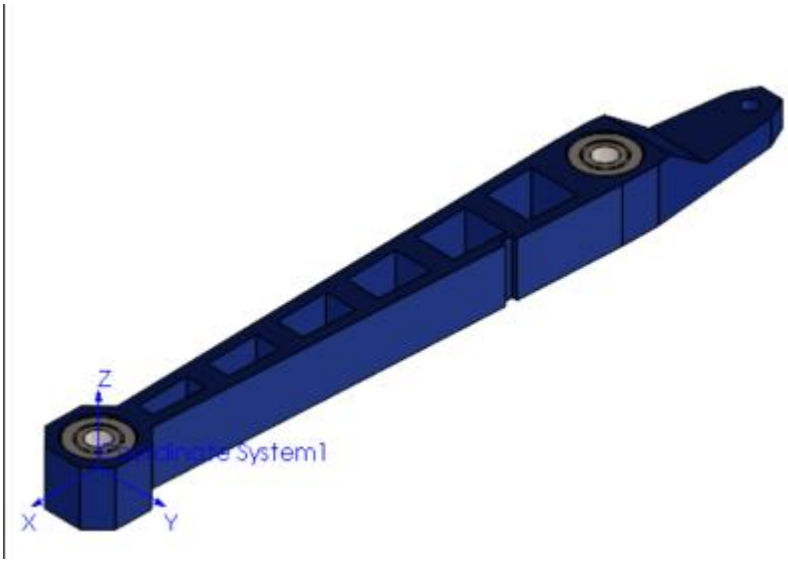
Tính toán góc theta 3 của cánh tay robot theo hình chiếu cạnh như sau:

$$\Phi 3 = cos^{-1}\left(\frac{(r3)^2 - (a2)^2 - (a3)^2}{-2 \times a2 \times a3}\right)$$

$$\theta 3 = 180 - \Phi 3$$

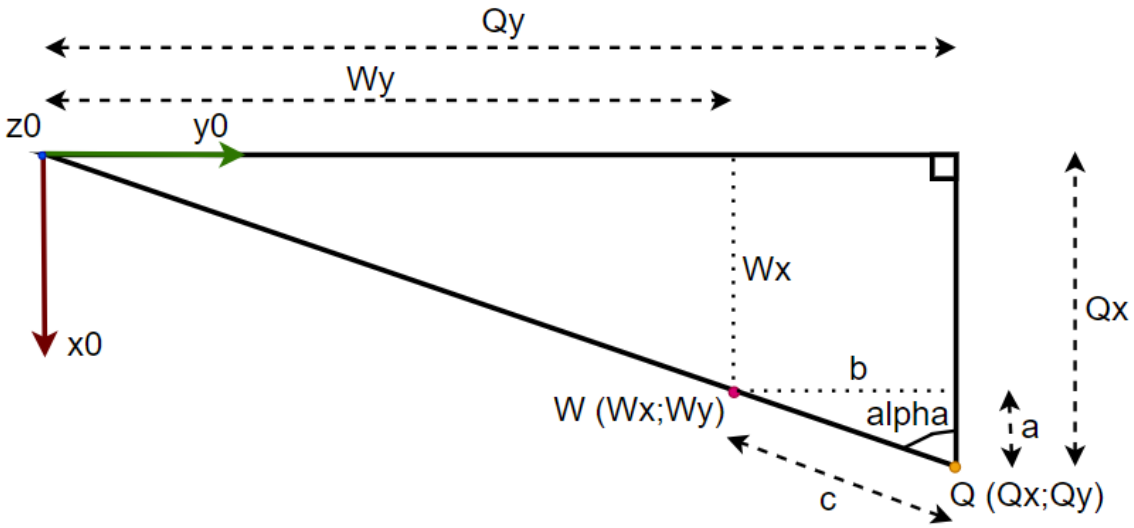
3.3.8 Hiệu chỉnh tọa độ điểm đến cho cánh tay robot

Vì cơ cấu chấp hành của cánh tay robot (tay gắp) bị phụ thuộc vào hai khớp 2 và 3 của cánh tay robot thông qua các thanh truyền động. Chính vì sự phụ thuộc này nên tay gắp luôn song song với mặt phẳng đặt cánh tay robot. Vì cấu tạo phụ thuộc trên của tay gắp nên việc đặt hệ tọa độ đầu cuối của cánh tay robot ở tay gắp là rất khó kiểm soát nên em sẽ đặt hệ tọa độ đầu cuối của cánh tay robot là ở điểm cuối của trục nối thứ 2 của cánh tay robot như hình bên dưới.



Hình 3-12. Hệ tọa độ đầu cuối của cánh tay robot

Khi đặt hệ tọa độ đầu cuối của cánh tay robot ở điểm cuối của trục nối thứ 2 của cánh tay robot thì ta cần hiệu chỉnh lại tọa độ điểm đến cho cánh tay robot sao cho tay gắp của cánh tay robot nằm đúng ở tọa độ tâm của vật thể. Cách tính toán sẽ dựa theo hình bên dưới.



Hình 3-13. Hình ảnh hiệu chỉnh cho cánh tay robot

Theo hình trên thì c sẽ là khoảng cánh hiệu chỉnh kiểm tra bằng thực nghiệm và ta sẽ cần biết tọa độ ở điểm W nên ta sẽ tiến hành các phép tính bên dưới.

$$alpha = 90° - \tan^{-1}\left(\frac{Qx}{Qy}\right)$$

$$a = c * \cos(alpha)$$

$$b = c * \sin(alpha)$$

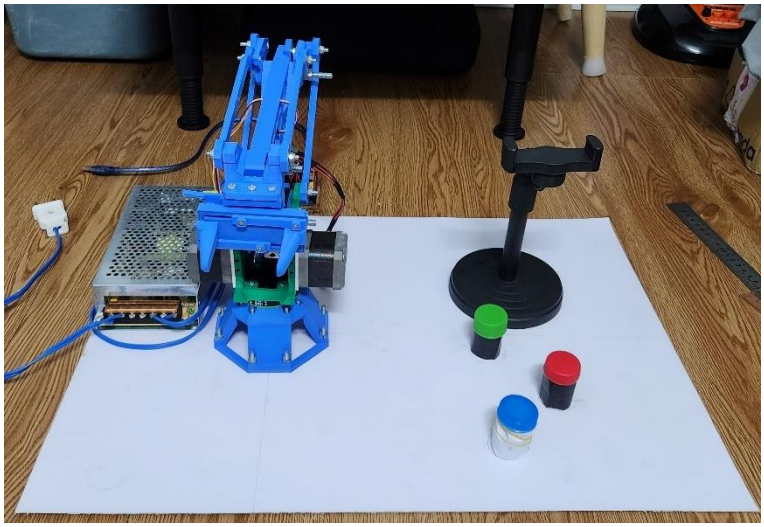
$$Wx = Qx - a$$

$$Wy = Qy - b$$

Matlab sẽ sử dụng Wx, Wy và Wz (chiều cao của vật thể, ta sẽ tự cho dựa vào vật thể cần được gắp cụ thể) để tính bài toán động học nghịch robot để tìm ra các góc xoay theta1, theta2 và theta3. Sau đó Matlab sẽ gửi các góc xoay theta1, theta2 và theta3 lên cổng Serial và vi điều khiển Arduino sẽ lấy dữ liệu các góc xoay này để điều khiển các động cơ bước đến đúng các góc xoay nhận được trên cổng Serial.

3.3.9 Mô hình cánh tay robot thực tế

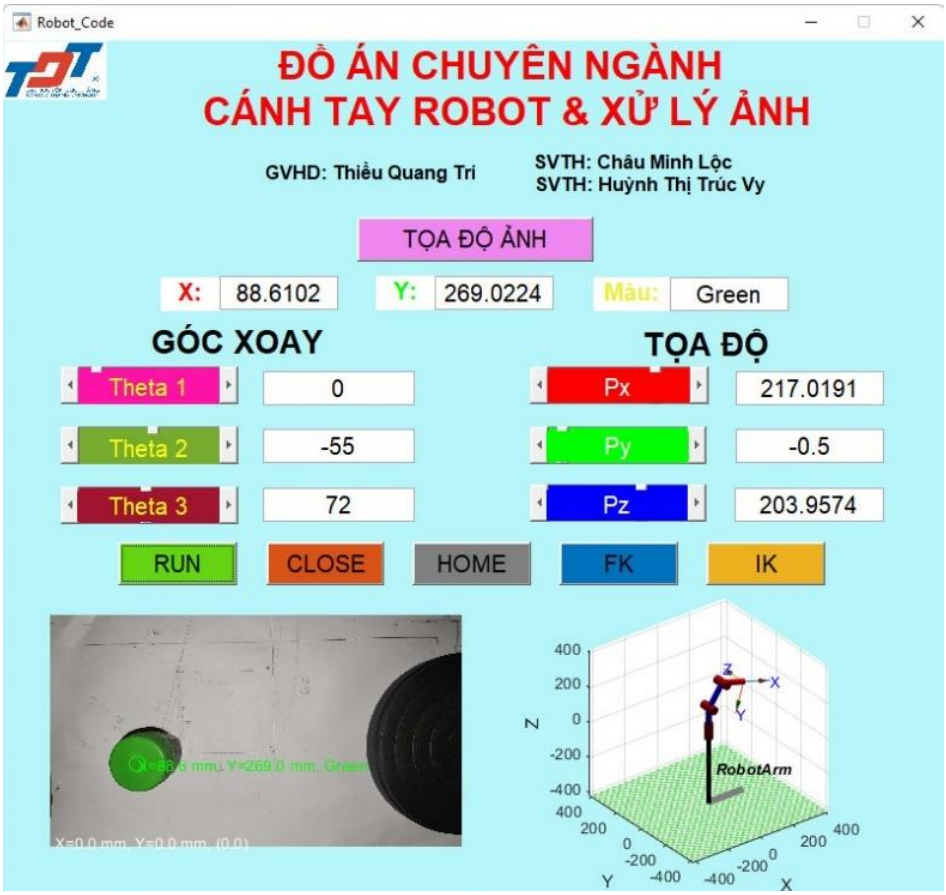
Dưới đây là hình ảnh làm mô hình thực tế.



Hình 3-14. Hình ảnh thi công mô hình cánh tay robot thực tế

3.4 Thiết kế giao diện người dùng GUIDE

Dưới đây là hình ảnh giao diện người dùng GUIDE trong Matlab. Ta có thể điều chỉnh góc xoay cũng như tọa độ đầu cuối của cánh tay robot tại đây. Người ra ta còn có thể xem được chuyển động mô phỏng của cánh tay robot so với cánh tay robot ngoài thực tế cũng như quan sát được các góc xoay và vị trí hiện tại của cánh tay robot. Cuối cùng, ta còn có thể theo dõi được tọa độ ảnh của đối tượng cần gắp.

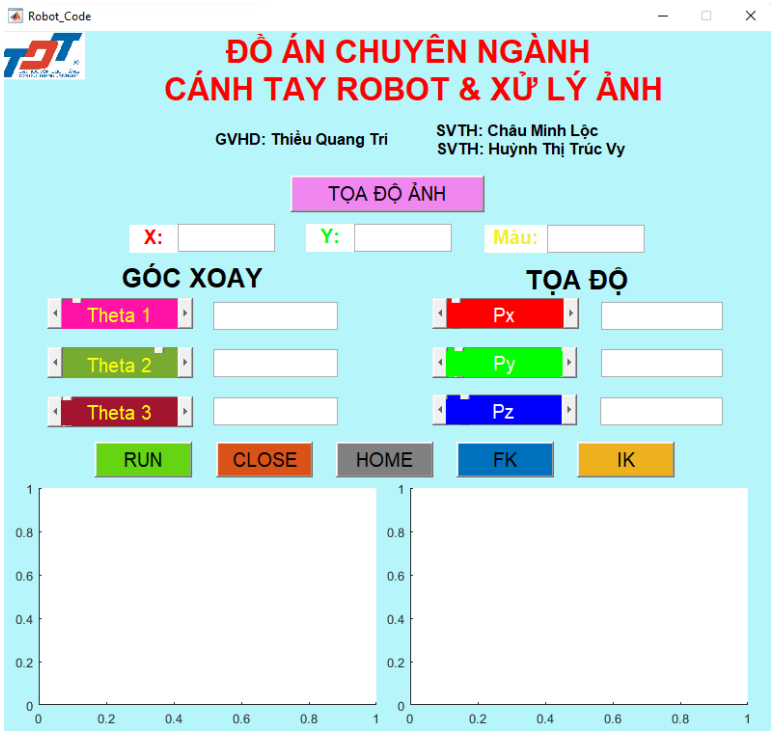


Hình 3-15. Giao diện người dùng GUIDE trong Matlab

CHƯƠNG 4. THỰC NGHIỆM

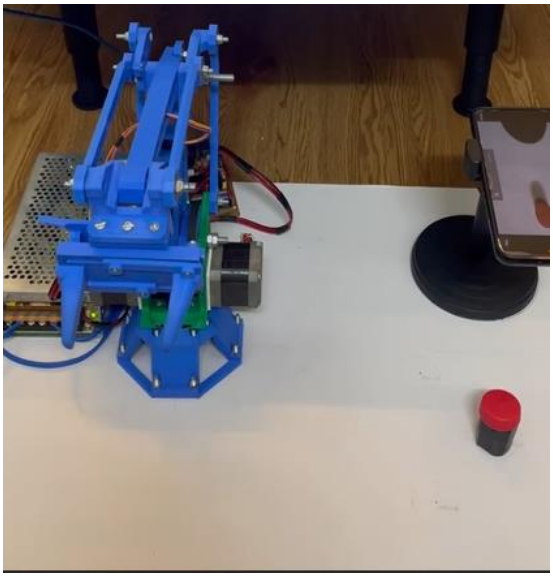
4.1 Tiến trình thực nghiệm

Bước 1: Ấn nút nhấn RUN trên giao diện người dùng GUIDE.



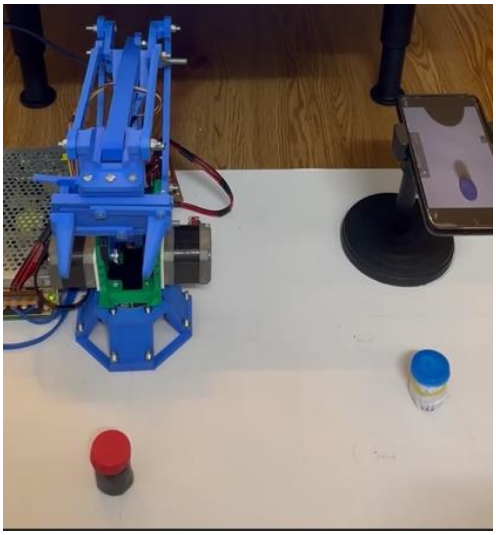
Hình 4-1. Bấm RUN trên giao diện người dùng GUIDE

Bước 2: Để vật màu đỏ vào vùng xử lý ảnh.



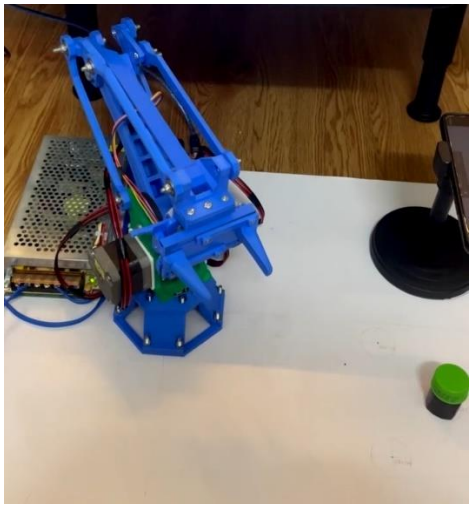
Hình 4-2. Đặt vật màu đỏ vào vùng xử lý ảnh

Bước 3: Sau khi cánh tay robot gắp vật để vào hộp màu đỏ và quay về vị trí ban đầu thì để vật màu xanh dương vào vùng xử lý ảnh.



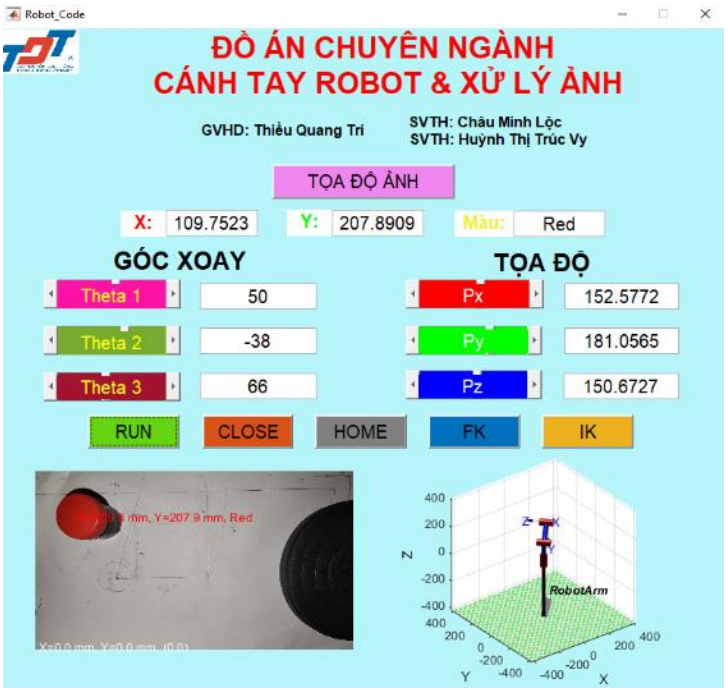
Hình 4-3. Đặt vật màu xanh dương vào vùng xử lý ảnh

Bước 4: Sau khi cánh tay robot gắp vật để vào hộp màu xanh dương và quay về vị trí ban đầu thì để vật màu xanh lá vào vùng xử lý ảnh.



Hình 4-4. Đặt vật màu xanh lá vào vùng xử lý ảnh

Bước 5: Sau khi cánh tay robot gắp vật để vào hộp màu xanh lá và quay về vị trí ban đầu thì ấn nút nhấn CLOSE trên giao diện người dùng GUIDE.



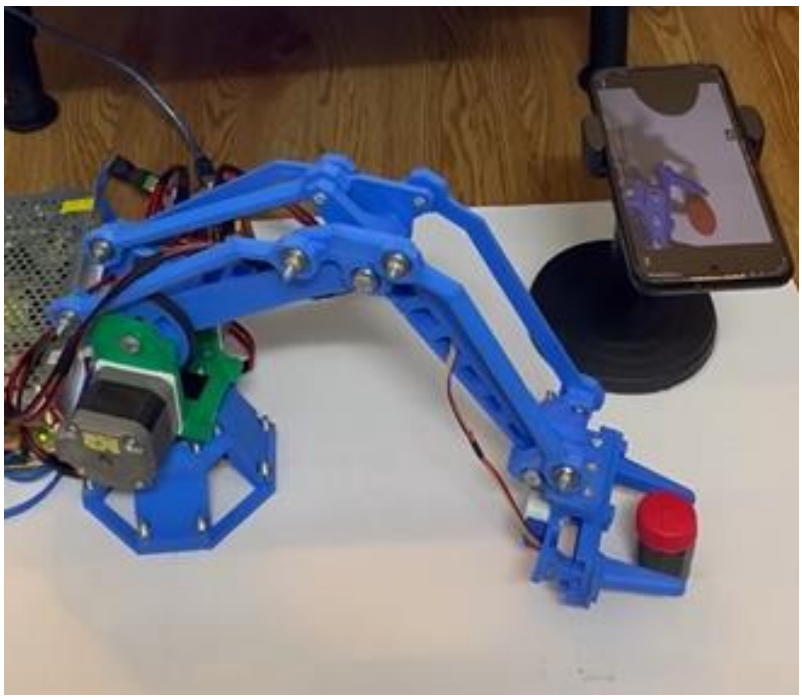
Hình 4-5. Bấm CLOSE trên giao diện người dùng GUIDE

4.2 Kết quả thực nghiệm

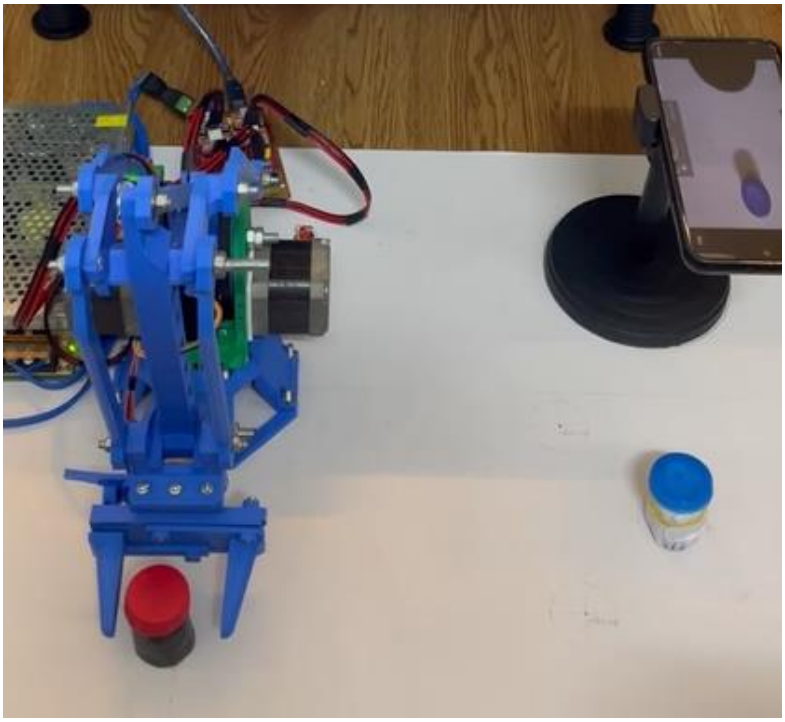
Sau khi ấn nút nhấn RUN trên giao diện người dùng GUIDE thì hệ thống bắt đầu hoạt động.

Sau khi để vật màu đỏ vào vùng xử lý ảnh thì Matlab xử lý ảnh và xuất lên giao diện người dùng GUIDE tọa độ tâm của vật thể.

Hình 4-6. Hiện tọa độ tâm của vật màu đỏ

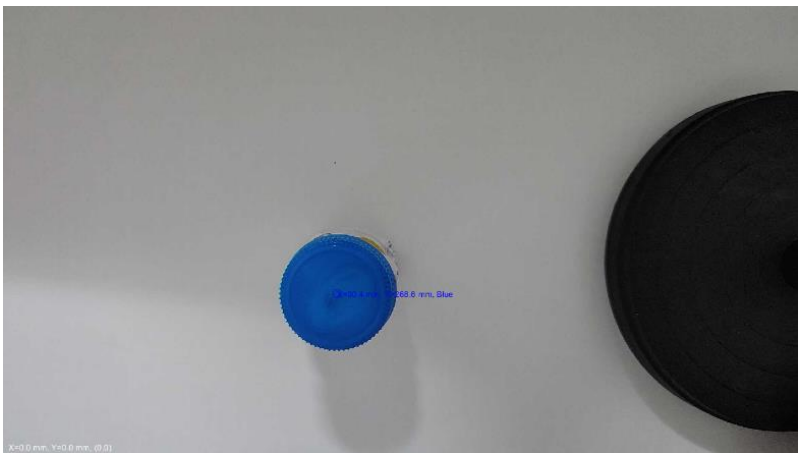


Hình 4-7. Cánh tay robot gắp vật màu đỏ

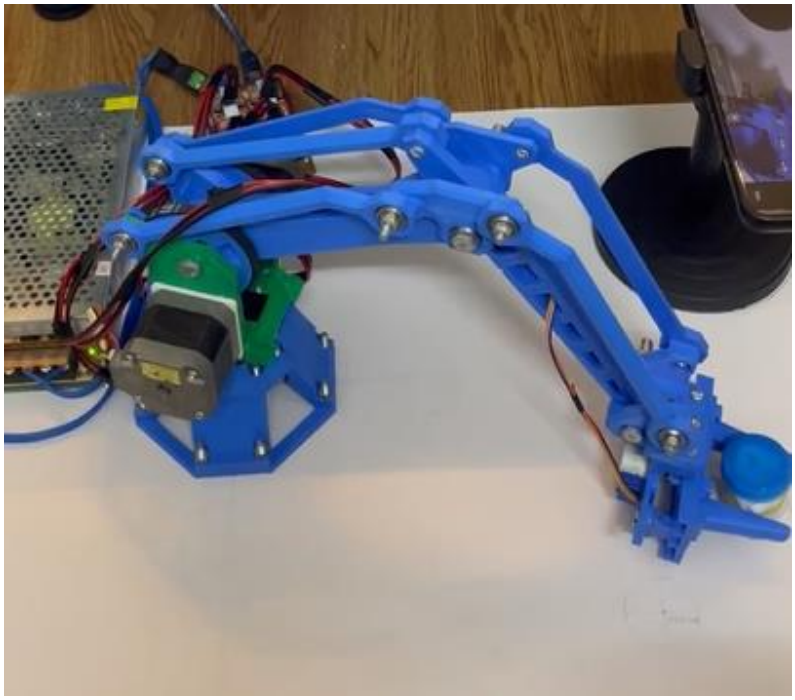


Hình 4-8. Cánh tay robot thả vật màu đỏ

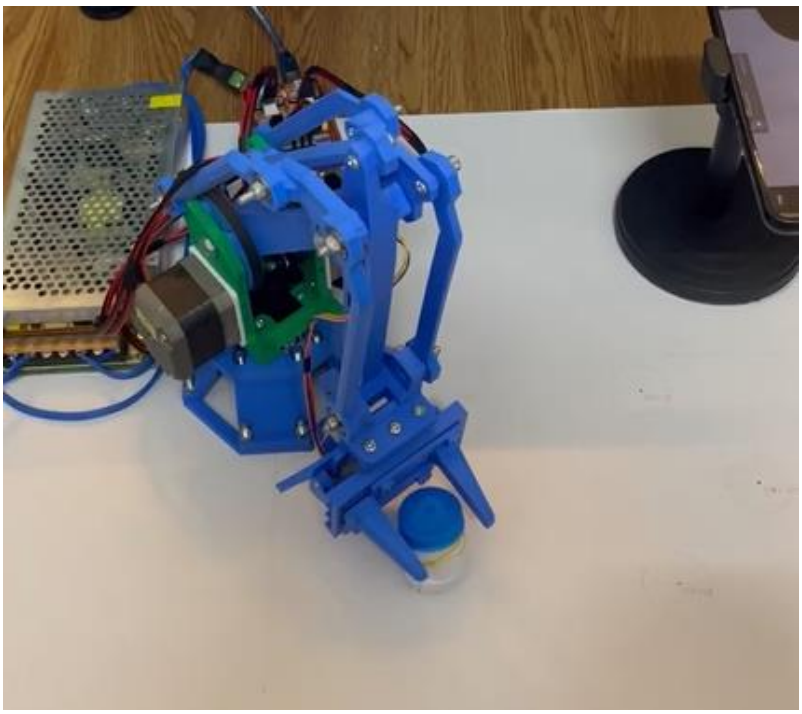
Sau khi cánh tay robot gắp vật để vào hộp màu đỏ và quay về vị trí ban đầu thì để vật màu xanh dương vào vùng xử lý ảnh.



Hình 4-9. Hiện tọa độ tâm của vật màu xanh dương

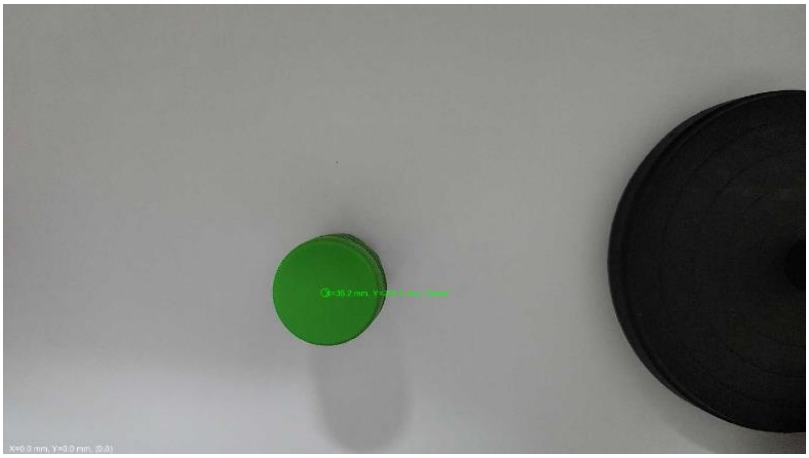


Hình 4-10. Cánh tay robot gắp vật màu xanh dương

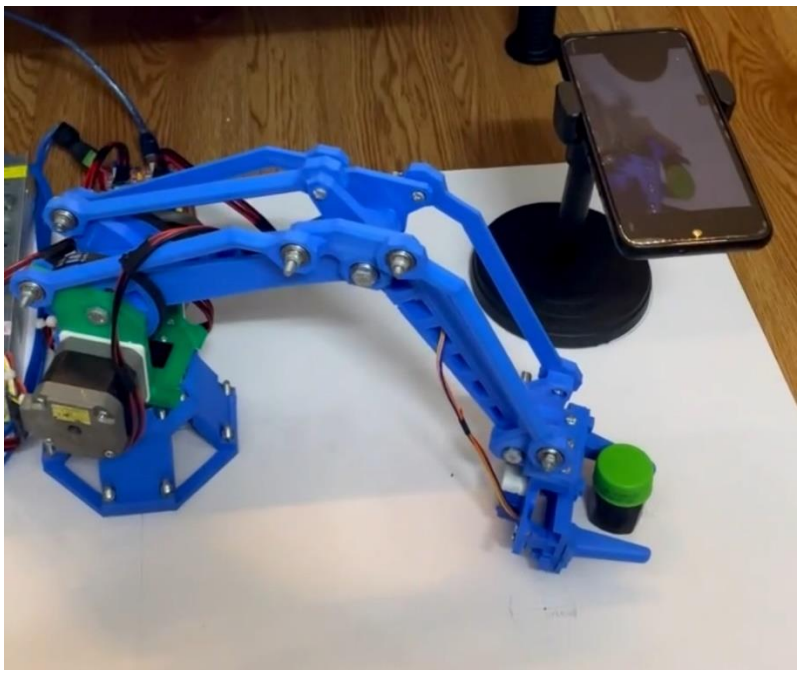


Hình 4-11. Cánh tay robot thả vật màu xanh dương

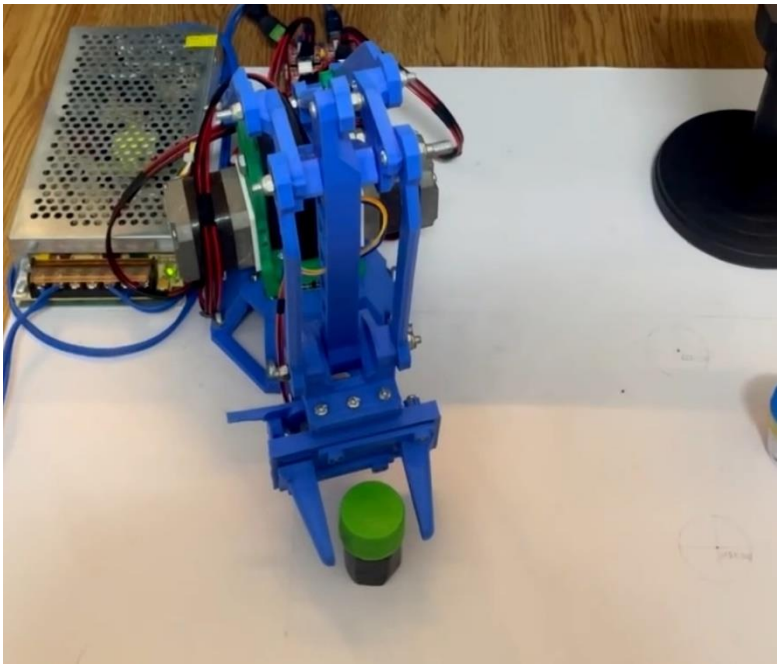
Sau khi cánh tay robot gắp vật để vào hộp màu xanh dương và quay về vị trí ban đầu thì để vật màu xanh lá vào vùng xử lý ảnh.



Hình 4-12. Hiện tọa độ tâm của vật màu xanh lá



Hình 4-13. Cánh tay robot gắp vật màu xanh lá



Hình 4-14. Cánh tay robot thả vật màu xanh lá

Sau khi cánh tay robot gắp vật để vào hộp màu xanh lá và quay về vị trí ban đầu thì ấn nút nhấn CLOSE trên giao diện người dùng GUIDE.

4.3 Kết luận thực nghiệm

Kết quả thực nghiệm đạt được phù hợp với các yêu cầu dự kiến đưa ra ban đầu ở chương 1 báo cáo.

CHƯƠNG 5. KẾT LUẬN

5.1 Ưu điểm

- Cánh tay robot được điều khiển bằng động cơ bước thông qua điều khiển vi bước từ driver A4988 nên góc xoay có sai số rất nhỏ.
- Mọi thông tin về góc xoay, tọa độ điểm cuối của cánh tay robot và tọa độ ảnh đều được hiển thị trên giao diện người dùng GUIDE của Matlab.
- Có thể quan sát chuyển động của cánh tay robot từ xa thông qua công cụ mô phỏng Simulink của Matlab.

5.2 Nhược điểm

- Sự mô phỏng trên Simulink và thực tế có độ trễ nhất định.
- Việc giao tiếp giữa phần mềm Matlab trong máy tính và vi điều khiển Arduino Uno thông qua cổng Serial chỉ dùng một đường truyền baudrate 9600 nên việc gửi và nhận giữ Matlab và vi điều khiển cần tạo độ trễ để không bị nhận nhầm dữ liệu không mong muốn.

5.3 Hướng phát triển

Trong đồ án sau em sẽ thay đổi vi điều khiển bằng PLC để mô phỏng mô hình cánh tay robot dùng cho quá trình sản xuất công nghiệp. Đồng thời tìm các cách giao tiếp khác để giảm bộ trễ cũng như tăng độ chính xác hơn về thời gian của mô phỏng và mô hình thực tế.

TÀI LIỆU THAM KHẢO

Tiếng Việt

Nguồn tổ ong 12V-10A:

https://nguonled.vn/nguon-tong-12v10a-lon

Tiếng Anh

Vi điều khiển Arduino Uno:

https://docs.arduino.cc/hardware/uno-rev3

Shield mở rộng CNC V3 cho Arduino Uno:

https://blog.protoneer.co.nz/arduino-cnc-shield/arduino-cnc-shield-schematics/

Module driver A4988:

https://www.alldatasheet.com/datasheet-pdf/pdf/338780/ALLEGRO/A4988.html

Động cơ bước NEMA 17 size 42 1.8°:

https://www.alldatasheet.com/datasheet-pdf/pdf/1572377/ETC/NEMA17.html

Động cơ bước 28byj-48 12V:

https://www.alldatasheet.com/datasheet-pdf/pdf/1245086/ETC1/28BYJ48.html

Công tắc hành trình V-151-1C25:

https://www.alldatasheet.com/datasheet-pdf/pdf/826528/OMRON/V-151-1C25.html

Mô hình cánh tay robot:

https://grabcad.com/library/robot-arm-community-version-cad-3d-printed-robotic-arm-1

Introduction to Robotics: Mechanics and Control (3rd Edition) - John J. Craig

PHỤ LỤC 1: CHƯƠNG TRÌNH VI ĐIỀU KHIỂN ARDUINO UNO

Robot_Arm.ino:

```
const int StepX = 2;
const int DirX = 5;
const int StepY = 3;
const int DirY = 6;
const int StepZ = 4;
const int DirZ = 7;
const int StepA = 12;
const int DirA = 13;
const int ENABLE = 8;
const int LSX = 9;
const int LSY = 10;
const int LSZ = 11;
const int LSA = A0;
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

float Theta1, Theta2, Theta3;
float Theta1M, Theta2M, Theta3M, Color, SerialData;
float Theta1R, Theta2R, Theta3R;
float Theta1T, Theta2T, Theta3T;
float HomeX, HomeY, HomeZ, HomeA;
float Angle = 0.5;
float AngleA = 10.0;
float HieuChinh = 35.0;
float Distance = 30.0;
int wait = 20;
int count = 0, objectRemoved, tieptucchay;
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

void HamGripper(float Theta, int Status) {
  digitalWrite(DirA, Status);
  for (int x = 0; x < (Theta / 0.35) * 17; x++) {
    digitalWrite(StepA, 1);
    delayMicroseconds(500);
    digitalWrite(StepA, 0);
    delayMicroseconds(500);
  }
}

void HamMove(int Step, int Dir, float Theta, int Status) {
  digitalWrite(Dir, Status);
  for (int x = 0; x < (Theta / 0.1125) * 4.5; x++) {
```

```
    digitalWrite(Step, 1);
    delayMicroseconds(500);
    digitalWrite(Step, 0);
    delayMicroseconds(500);
  }
}

void HamDelay(uint32_t TimeNow, uint32_t TimeDelay) {
  uint32_t Time = millis();
  while (Time - TimeNow < TimeDelay)
    Time = millis();
}

void HOME() {
  while (digitalRead(LSX) == 1)
    HamMove(StepX, DirX, Angle, 0);
  while (digitalRead(LSY) == 1 and digitalRead(LSZ) == 1) {
    HamMove(StepY, DirY, Angle, 0);
    HamMove(StepZ, DirZ, Angle, 0);
  }
  while (digitalRead(LSY) == 1)
    HamMove(StepY, DirY, Angle, 0);
  while (digitalRead(LSZ) == 1)
    HamMove(StepZ, DirZ, Angle, 0);
}

void HOME_END() {
  while (digitalRead(LSY) == 1 and digitalRead(LSZ) == 1) {
    HamMove(StepY, DirY, Angle, 0);
    HamMove(StepZ, DirZ, Angle, 0);
  }
  while (digitalRead(LSY) == 1)
    HamMove(StepY, DirY, Angle, 0);
  while (digitalRead(LSZ) == 1)
    HamMove(StepZ, DirZ, Angle, 0);
  while (digitalRead(LSX) == 1)
    HamMove(StepX, DirX, Angle, 0);
}

void HomeGripper() {
  float Angle_Gripper;
  // while (digitalRead(LSA) == 1) {
  //   HamGripper(AngleA, 1);
  //   Serial.println("HomeGripper");
  //   HamDelay(millis(), wait);
  // }
```

```
  Angle_Gripper = (5.0 * 360.0) / 45.0;
  for (HomeA = AngleA; HomeA <= Angle_Gripper; HomeA += AngleA) {
    HamGripper(AngleA, 0);
    Serial.println("Gripper");
    HamDelay(millis(), wait);
  }
}

void HamTheta1() {
  Theta1R = Theta1M;
  if (round(Theta1R) == 0.0) {
    Theta1 = Theta1M;
    Serial.println("STEPX");
    Serial.println(Theta1);
    HamDelay(millis(), wait);
  } else {
    Theta1 = 0.0;
    for (HomeX = Angle; HomeX <= Theta1R; HomeX += Angle) {
      HamMove(StepX, DirX, Angle, 1);
      Theta1 = Theta1 + Angle;
      Serial.println("STEPX");
      Serial.println(Theta1);
      HamDelay(millis(), wait);
    }
  }
}

void HamTheta2() {
  Theta2R = abs(-130.0 - Theta2M);
  if (Theta2R == 0.0) {
    Theta2 = -130.0;
    Theta3 = 114.0;
    Serial.println("STEPY");
    Serial.println(Theta2);
    Serial.println("STEPZ");
    Serial.println(Theta3);
    HamDelay(millis(), wait);
  } else if (Theta2R > 0.0 && Theta2R <= 79.0) {
    for (HomeY = Angle; HomeY <= Theta2R; HomeY += Angle) {
      HamMove(StepY, DirY, Angle, 1);
      Theta2 = -130.0 + HomeY;
      Theta3 = 114.0 - HomeY;
      Serial.println("STEPY");
      Serial.println(Theta2);
      Serial.println("STEPZ");
      Serial.println(Theta3);
```

```
    HamDelay(millis(), wait);
   }
  } else {
   for (HomeY = Angle; HomeY <= 79.0; HomeY += Angle) {
    HamMove(StepY, DirY, Angle, 1);
    Theta2 = -130.0 + HomeY;
    Theta3 = 114.0 - HomeY;
    Serial.println("STEPY");
    Serial.println(Theta2);
    Serial.println("STEPZ");
    Serial.println(Theta3);
    HamDelay(millis(), wait);
   }
   Theta3R = abs(Theta2R - 79.0);
   for (HomeZ = Angle; HomeZ <= Theta3R; HomeZ += Angle) {
    HamMove(StepZ, DirZ, Angle, 1);
    Theta3 = 35.0 + HomeZ;
    Serial.println("STEPZ");
    Serial.println(Theta3);
    HamDelay(millis(), wait);
   }
   Theta2R = abs(Theta2R - 79.0);
   for (HomeY = Angle; HomeY <= Theta2R; HomeY += Angle) {
    HamMove(StepY, DirY, Angle, 1);
    Theta2 = -51.0 + HomeY;
    Theta3 = Theta3 - Angle;
    Serial.println("STEPY");
    Serial.println(Theta2);
    Serial.println("STEPZ");
    Serial.println(Theta3);
    HamDelay(millis(), wait);
   }
  }
}

void HamTheta3() {
  Theta3R = abs(Theta3M - Theta3);
  if (Theta3R == 0.0) {
   Theta3 = Theta3M;
   Serial.println("STEPZ");
   Serial.println(Theta3);
   HamDelay(millis(), wait);
  } else {
   for (HomeZ = Angle; HomeZ <= Theta3R; HomeZ += Angle) {
    HamMove(StepZ, DirZ, Angle, 1);
    Theta3 = Theta3 + Angle;
```

```
      Serial.println("STEPZ");
      Serial.println(Theta3);
      HamDelay(millis(), wait);
    }
  }
}

void HamTheta4(float Distance, int Status) {
  float Angle_Gripper;
  Angle_Gripper = (Distance * 360.0) / 45.0;
  for (HomeA = AngleA; HomeA <= Angle_Gripper; HomeA += AngleA) {
    HamGripper(AngleA, Status);
    Serial.println("Gripper");
    HamDelay(millis(), wait);
  }
}

void HamHieuChinh() {
  // for (HomeY = Angle; HomeY <= HieuChinh; HomeY += Angle) {
  //   HamMove(StepY, DirY, Angle, 0);
  //   Theta2 = Theta2M - HomeY;
  //   Theta3 = Theta3 + Angle;
  //   Serial.println("STEPY");
  //   Serial.println(Theta2);
  //   Serial.println("STEPZ");
  //   Serial.println(Theta3);
  //   HamDelay(millis(), wait);
  // }
  // for (HomeZ = Angle; HomeZ <= HieuChinh; HomeZ += Angle) {
  //   HamMove(StepZ, DirZ, Angle, 1);
  //   Theta3 = Theta3 - Angle;
  //   Serial.println("STEPZ");
  //   Serial.println(Theta3);
  //   HamDelay(millis(), wait);
  // }
  for (HomeY = Angle; HomeY <= HieuChinh; HomeY += Angle) {
    HamMove(StepY, DirY, Angle, 1);
    Theta2 = Theta2 + Angle;
    Theta3 = Theta3 - Angle;
    Serial.println("STEPY");
    Serial.println(Theta2);
    Serial.println("STEPZ");
    Serial.println(Theta3);
    HamDelay(millis(), wait);
  }
}
```

```
void HamRed() {
  Theta1M = 0.0;
  Theta2M = -40.0;
  Theta3M = 140.0;
  HamTheta2();
  HamTheta3();
  HamTheta1();
}

void HamGreen() {
  Theta1M = 20.0;
  Theta2M = -40.0;
  Theta3M = 140.0;
  HamTheta2();
  HamTheta3();
  HamTheta1();
}

void HamBlue() {
  Theta1M = 45.0;
  Theta2M = -40.0;
  Theta3M = 140.0;
  HamTheta2();
  HamTheta3();
  HamTheta1();
}

void HamRemoved() {
  objectRemoved = Serial.parseFloat();
  HamDelay(millis(), wait);
  objectRemoved = round(objectRemoved);
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

void setup() {
  pinMode(StepX, OUTPUT);
  pinMode(DirX, OUTPUT);
  pinMode(StepY, OUTPUT);
  pinMode(DirY, OUTPUT);
  pinMode(StepZ, OUTPUT);
  pinMode(DirZ, OUTPUT);
  pinMode(StepA, OUTPUT);
  pinMode(DirA, OUTPUT);
  pinMode(ENABLE, OUTPUT);
```

```
  pinMode(LSX, INPUT_PULLUP);
  pinMode(LSY, INPUT_PULLUP);
  pinMode(LSZ, INPUT_PULLUP);
  pinMode(LSA, INPUT_PULLUP);
  digitalWrite(ENABLE, 0);
  Serial.begin(9600);

  HOME();
  Serial.println("HOME");
}

void loop() {
  if (Serial.available() > 0) {

    // while (count == 2) {

    //   tieptucchay = Serial.parseFloat();
    //   HamDelay(millis(), wait);
    //   tieptucchay = round(tieptucchay);
    //   if (tieptucchay == 222.0) {
    //     count = 0;
    //     Serial.println("HOME");
    //     HamDelay(millis(), wait);
    //   delay(5000);
    //   break;
    //   }
    //   Serial.println("WAIT");
    //   HamDelay(millis(), wait);
    // }


    while (count == 0) {
      Serial.println("WAIT");
      HamDelay(millis(), wait);
      tieptucchay = Serial.parseFloat();
      HamDelay(millis(), wait);
      tieptucchay = round(tieptucchay);
      if (tieptucchay == 222.0) {
        Serial.println("HOME");
        HamDelay(millis(), wait);
        // Thoát khỏi vòng lặp và tiếp tục với code sau đó
      }

      Theta1M = Serial.parseFloat();
      HamDelay(millis(), wait);
```

```
  Theta2M = Serial.parseFloat();
  HamDelay(millis(), wait);
  Theta3M = Serial.parseFloat();
  HamDelay(millis(), wait);
  Color = Serial.parseFloat();
  HamDelay(millis(), wait);
  SerialData = Serial.parseFloat();
  HamDelay(millis(), wait);

  Theta1M = round(Theta1M);
  Theta2M = round(Theta2M);
  Theta3M = round(Theta3M);
  Color = round(Color);
  SerialData = round(SerialData);

  if (SerialData == 555.0)
    count = 1;
}

while (count == 1) {
  HamTheta2();
  HamRemoved();
  if (objectRemoved == 999.0) {
    HOME();
 count=0;
    break;
  }
  HamTheta3();
  HamRemoved();
  if (objectRemoved == 999.0) {
    HOME();
 count=0;
    break;
  }
  HamTheta1();
  HamRemoved();
  if (objectRemoved == 999.0) {
    HOME();
 count=0;
    break;
  }

  HomeGripper();
  // HamRemoved();
//   if (objectRemoved == 999.0) {
//     HOME();
```

```
//  count=0;
//     break;
//   }
  HamHieuChinh();
//   HamRemoved();
//   if (objectRemoved == 999.0) {
//     HOME();
//  count=0;
//     break;
//   }
  HamTheta4(Distance, 1);
//   HamRemoved();
//   if (objectRemoved == 999.0) {
//     HOME();
//  count=0;
//     break;
//   }
  HOME();
//   HamRemoved();
//   if (objectRemoved == 999.0) {
//     HOME();
//  count=0;
//     break;
//   }
  Serial.println("HOME_NEW");
  HamDelay(millis(), wait);

  if (Color == 1.0)
    HamRed();
  else if (Color == 2.0)
    HamGreen();
  else
    HamBlue();
  // HamRemoved();
//   if (objectRemoved == 999.0) {
//     HOME();
//  count=0;
//     break;
//   }

  HamTheta4(Distance, 0);
  // HamRemoved();
//   if (objectRemoved == 999.0) {
//     HOME();
//  count=0;
//     break;
```

```
   //   }
     HOME_END();
     HamRemoved();
     if (objectRemoved == 999.0) {
       HOME();
    count=0;
       break;
     }
     Serial.println("END");
     HamDelay(millis(), wait);
     Serial.println("HOME");
     HamDelay(millis(), wait);
     count = 0;
     delay(8000);
    }
  }
}
```

PHỤ LỤC 2: CHƯƠNG TRÌNH MATLAB

Robot_Code.m:

```
function varargout = Robot_Code(varargin)
% ROBOT_CODE MATLAB code for Robot_Code.fig
%      ROBOT_CODE, by itself, creates a new ROBOT_CODE or raises the existing
%      singleton*.
%
%      H = ROBOT_CODE returns the handle to a new ROBOT_CODE or the handle to
%      the existing singleton*.
%
%      ROBOT_CODE('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in ROBOT_CODE.M with the given input arguments.
%
%      ROBOT_CODE('Property','Value',...) creates a new ROBOT_CODE or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Robot_Code_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Robot_Code_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Robot_Code

% Last Modified by GUIDE v2.5 25-Nov-2023 03:22:04

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
   'gui_Singleton',  gui_Singleton, ...
   'gui_OpeningFcn', @Robot_Code_OpeningFcn, ...
   'gui_OutputFcn',  @Robot_Code_OutputFcn, ...
   'gui_LayoutFcn',  [] , ...
   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
   [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
   gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Robot_Code is made visible.
function Robot_Code_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% varargin   command line arguments to Robot_Code (see VARARGIN)

% Choose default command line output for Robot_Code
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% --- Outputs from this function are returned to the command line.
function varargout = Robot_Code_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
axes(handles.axes0);
imshow('E:\DoAnChuyenNganh\MoPhong\TDTU.jpg');

% --- Executes on button press in pushbutton0.
function pushbutton0_Callback(hObject, eventdata, handles)
url = 'http://192.168.35.135:8080/shot.jpg';

paper_width_mm = 594;
paper_height_mm = 840;

% Kích thước thực của ảnh (250.5 x 140 mm)
CR = 280; %280
CD = 155;  %155

% Kích thước ảnh trên màn hình (1920x1080 pixels)
image_width_pixels = 1920;
image_height_pixels = 1080;

% Tính tỷ lệ giữa kích thước thực và kích thước trên màn hình
width_ratio = CR / image_width_pixels;
height_ratio = CD / image_height_pixels;
a = 175; % Khoảng cách từ tâm robot tới khung ảnh theo chìu X
b = 320; % Khoảng cách từ tâm robot tới khung ảnh theo chìu Y

while true
   try
      % Đọc hình ảnh từ URL
      imagen = webread(url);

      % Hiển thị kích thước hình ảnh
      % disp(['Kích thước hình ảnh: ' num2str(size(imagen, 2)) 'x' num2str(size(imagen, 1)) ' pixels']); %2 là
lấy chiều rộng,1 là chiều cao ảnh

      % Xử lý hình ảnh, xác định ngưỡng màu
      red = imagen(:,:,1);
      green = imagen(:,:,2);
      blue = imagen(:,:,3);

      % Tính toán sự khác biệt giữa các kênh màu
      diff_rg = red - green - blue;
```

```
        diff_gb = green - blue - red;
        diff_br = blue - green - red;

        % Điều chỉnh ngưỡng màu cho mỗi kênh (trong dải từ 0-255)
        threshold_red = 5;  % Ngưỡng màu cho kênh đỏ
        threshold_green = 10;  % Ngưỡng màu cho kênh xanh lá cây
        threshold_blue = 5;  % Ngưỡng màu cho kênh xanh dương

        % Tạo mask cho từng màu dựa trên ngưỡng
        red_mask = diff_rg > threshold_red;
        green_mask = diff_gb > threshold_green;
        blue_mask = diff_br > threshold_blue;

        % Reset các nhãn
        label_red = zeros(size(red_mask));
        label_green = zeros(size(green_mask));
        label_blue = zeros(size(blue_mask));

        % Sử dụng bwlabel để đánh dấu các đối tượng trong ảnh cho từng màu
        [label_red, num_red] = bwlabel(red_mask, 8);
        [label_green, num_green] = bwlabel(green_mask, 8);
        [label_blue, num_blue] = bwlabel(blue_mask, 8);

        % Sử dụng regionprops để truy xuất thuộc tính của các đối tượng cho từng màu
        stats_red = regionprops(label_red, 'Area', 'Centroid', 'Eccentricity');
        stats_green = regionprops(label_green, 'Area', 'Centroid', 'Eccentricity');
        stats_blue = regionprops(label_blue, 'Area', 'Centroid', 'Eccentricity');

        min_area_threshold = 1000;
        max_eccentricity = 0.45; % ngưỡng gần giống hình tròn

        % Hiển thị hình ảnh
        axes(handles.axes1);
        imshow(imagen);
        hold on

        % Đặt vị trí tọa độ (0, 0) trên hình ảnh
        text(20, size(imagen, 1) - 20, ['X=0.0 mm, Y=0.0 mm, (0,0)'], 'Color', 'w');

    % Khởi tạo biến ngoài các vòng lặp
X_robot_red = 0.0;
Y_robot_red = 0.0;
Color_red = NaN;

X_robot_green = 0.0;
Y_robot_green = 0.0;
Color_green = NaN;

X_robot_blue = 0.0;
Y_robot_blue = 0.0;
Color_blue = NaN;

% ...

% Đối tượng màu đỏ
found_red_object = false;
for i = 1:num_red
```

```
    if stats_red(i).Area > min_area_threshold && stats_red(i).Eccentricity < max_eccentricity
        centroid = stats_red(i).Centroid;
        x_mm = centroid(1) * width_ratio;
        y_mm = CD - centroid(2) * height_ratio;
        X_robot_red = a - x_mm;
        Y_robot_red = b - y_mm;
        Color_red = 1.0;
        plot(centroid(1), centroid(2), 'ro', 'MarkerSize', 10);
        text(centroid(1), centroid(2), ['X=', num2str(X_robot_red, '%.1f'), ' mm, Y=', num2str(Y_robot_red,
'%.1f'), ' mm, Red'], 'Color', 'r');
        found_red_object = true;
    end
end

% ...

% Đối tượng màu xanh lá cây
found_green_object = false;
for i = 1:num_green
    if stats_green(i).Area > min_area_threshold && stats_green(i).Eccentricity < max_eccentricity
        centroid = stats_green(i).Centroid;
        x_mm = centroid(1) * width_ratio;
        y_mm = CD - centroid(2) * height_ratio;
        X_robot_green = a - x_mm;
        Y_robot_green = b - y_mm;
        Color_green = 2.0;
        plot(centroid(1), centroid(2), 'go', 'MarkerSize', 10);
        text(centroid(1), centroid(2), ['X=', num2str(X_robot_green, '%.1f'), ' mm, Y=', num2str(Y_robot_green,
'%.1f'), ' mm, Green'], 'Color', 'g');
        found_green_object = true;
    end
end

% ...

% Đối tượng màu xanh dương
found_blue_object = false;
for i = 1:num_blue
    if stats_blue(i).Area > min_area_threshold && stats_blue(i).Eccentricity < max_eccentricity
        centroid = stats_blue(i).Centroid;
        x_mm = centroid(1) * width_ratio;
        y_mm = CD - centroid(2) * height_ratio;
        X_robot_blue = a - x_mm;
        Y_robot_blue = b - y_mm;
        Color_blue = 3.0;
        plot(centroid(1), centroid(2), 'bo', 'MarkerSize', 10);
        text(centroid(1), centroid(2), ['X=', num2str(X_robot_blue, '%.1f'), ' mm, Y=', num2str(Y_robot_blue,
'%.1f'), ' mm, Blue'], 'Color', 'b');
        found_blue_object = true;
    end
end

% ...

% Sau tất cả các phát hiện màu, xác định giá trị cuối cùng
if found_red_object
    X_robot = X_robot_red;
```

```
    Y_robot = Y_robot_red;
    Color = Color_red;
elseif found_green_object
    X_robot = X_robot_green;
    Y_robot = Y_robot_green;
    Color = Color_green;
elseif found_blue_object
    X_robot = X_robot_blue;
    Y_robot = Y_robot_blue;
    Color = Color_blue;
else
    % Không có đối tượng được phát hiện
    X_robot = 0.0;
    Y_robot = 0.0;
    Color = NaN;
end

hold off
drawnow;  % Cập nhật hiển thị

    catch
        disp('Lỗi trong quá trình xử lý hình ảnh.');
    end

    set(handles.edit7,'string',num2str(X_robot));
    set(handles.edit8,'string',num2str(Y_robot));

    if (Color==1.0)
        set(handles.edit9,'string','Red');
    elseif (Color==2.0)
        set(handles.edit9,'string','Green');
    elseif (Color==3.0)
        set(handles.edit9,'string','Blue');
    else
        set(handles.edit9,'string','Error');
    end
end

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
clc
s = serialport('COM4',9600);

theta1=0.0;
theta2=-130.0;
theta3=114.0;
SpeedX=0;
SpeedY=0;
SpeedZ=0;
count = 0;
SerialData = 0.0;
objectRemoved = 999.0;
KhoangCach=90.0;
ChieuCao=120.0;

X_robot = 0.0;
Y_robot = 0.0;
```

```
url = 'http://192.168.35.135:8080/shot.jpg';

paper_width_mm = 594;
paper_height_mm = 840;

% Kích thước thực của ảnh (250.5 x 140 mm)
CR = 280; %250.5
CD = 155;  %140

% Kích thước ảnh trên màn hình (1920x1080 pixels)
image_width_pixels = 1920;
image_height_pixels = 1080;

% Tính tỷ lệ giữa kích thước thực và kích thước trên màn hình
width_ratio = CR / image_width_pixels;
height_ratio = CD / image_height_pixels;
a = 175; % Khoảng cách từ tâm robot tới khung ảnh theo chìu X
b = 320; % Khoảng cách từ tâm robot tới khung ảnh theo chìu Y

while true
   try
      % Đọc hình ảnh từ URL
      imagen = webread(url);

      % Hiển thị kích thước hình ảnh
      % disp(['Kích thước hình ảnh: ' num2str(size(imagen, 2)) 'x' num2str(size(imagen, 1)) ' pixels']); %2 là
lấy chiều rộng,1 là chiều cao ảnh

      % Xử lý hình ảnh, xác định ngưỡng màu
      red = imagen(:,:,1);
      green = imagen(:,:,2);
      blue = imagen(:,:,3);

      % Tính toán sự khác biệt giữa các kênh màu
      diff_rg = red - green - blue;
      diff_gb = green - blue - red;
      diff_br = blue - green - red;

      % Điều chỉnh ngưỡng màu cho mỗi kênh (trong dải từ 0-255)
      threshold_red = 5;  % Ngưỡng màu cho kênh đỏ
      threshold_green = 10;  % Ngưỡng màu cho kênh xanh lá cây
      threshold_blue = 5;  % Ngưỡng màu cho kênh xanh dương

      % Tạo mask cho từng màu dựa trên ngưỡng
      red_mask = diff_rg > threshold_red;
      green_mask = diff_gb > threshold_green;
      blue_mask = diff_br > threshold_blue;

      % Reset các nhãn
      label_red = zeros(size(red_mask));
      label_green = zeros(size(green_mask));
      label_blue = zeros(size(blue_mask));

      % Sử dụng bwlabel để đánh dấu các đối tượng trong ảnh cho từng màu
      [label_red, num_red] = bwlabel(red_mask, 8);
      [label_green, num_green] = bwlabel(green_mask, 8);
```

```
    [label_blue, num_blue] = bwlabel(blue_mask, 8);

    % Sử dụng regionprops để truy xuất thuộc tính của các đối tượng cho từng màu
    stats_red = regionprops(label_red, 'Area', 'Centroid', 'Eccentricity');
    stats_green = regionprops(label_green, 'Area', 'Centroid', 'Eccentricity');
    stats_blue = regionprops(label_blue, 'Area', 'Centroid', 'Eccentricity');

    min_area_threshold = 1000;
    max_eccentricity = 0.45; % ngưỡng gần giống hình tròn

    % Hiển thị hình ảnh
    axes(handles.axes1);
    imshow(imagen);
    hold on

    % Đặt vị trí tọa độ (0, 0) trên hình ảnh
    text(20, size(imagen, 1) - 20, ['X=0.0 mm, Y=0.0 mm, (0,0)'], 'Color', 'w');

    % Khởi tạo biến ngoài các vòng lặp
X_robot_red = 0.0;
Y_robot_red = 0.0;
Color_red = NaN;

X_robot_green = 0.0;
Y_robot_green = 0.0;
Color_green = NaN;

X_robot_blue = 0.0;
Y_robot_blue = 0.0;
Color_blue = NaN;

% ...

% Đối tượng màu đỏ
found_red_object = false;
for i = 1:num_red
  if stats_red(i).Area > min_area_threshold && stats_red(i).Eccentricity < max_eccentricity
    centroid = stats_red(i).Centroid;
    x_mm = centroid(1) * width_ratio;
    y_mm = CD - centroid(2) * height_ratio;
    X_robot_red = a - x_mm;
    Y_robot_red = b - y_mm;
    Color_red = 1.0;
    plot(centroid(1), centroid(2), 'ro', 'MarkerSize', 10);
    text(centroid(1), centroid(2), ['X=', num2str(X_robot_red, '%.1f'), ' mm, Y=', num2str(Y_robot_red,
'%.1f'), ' mm, Red'], 'Color', 'r');
    found_red_object = true;
  end
end

% ...

% Đối tượng màu xanh lá cây
found_green_object = false;
for i = 1:num_green
  if stats_green(i).Area > min_area_threshold && stats_green(i).Eccentricity < max_eccentricity
    centroid = stats_green(i).Centroid;
```

```
        x_mm = centroid(1) * width_ratio;
        y_mm = CD - centroid(2) * height_ratio;
        X_robot_green = a - x_mm;
        Y_robot_green = b - y_mm;
        Color_green = 2.0;
        plot(centroid(1), centroid(2), 'go', 'MarkerSize', 10);
        text(centroid(1), centroid(2), ['X=', num2str(X_robot_green, '%.1f'), ' mm, Y=', num2str(Y_robot_green,
'%.1f'), ' mm, Green'], 'Color', 'g');
        found_green_object = true;
    end
end

% ...

% Đối tượng màu xanh dương
found_blue_object = false;
for i = 1:num_blue
    if stats_blue(i).Area > min_area_threshold && stats_blue(i).Eccentricity < max_eccentricity
        centroid = stats_blue(i).Centroid;
        x_mm = centroid(1) * width_ratio;
        y_mm = CD - centroid(2) * height_ratio;
        X_robot_blue = a - x_mm;
        Y_robot_blue = b - y_mm;
        Color_blue = 3.0;
        plot(centroid(1), centroid(2), 'bo', 'MarkerSize', 10);
        text(centroid(1), centroid(2), ['X=', num2str(X_robot_blue, '%.1f'), ' mm, Y=', num2str(Y_robot_blue,
'%.1f'), ' mm, Blue'], 'Color', 'b');
        found_blue_object = true;
    end
end

% ...

% Sau tất cả các phát hiện màu, xác định giá trị cuối cùng
if found_red_object
    X_robot = X_robot_red;
    Y_robot = Y_robot_red;
    Color = Color_red;
elseif found_green_object
    X_robot = X_robot_green;
    Y_robot = Y_robot_green;
    Color = Color_green;
elseif found_blue_object
    X_robot = X_robot_blue;
    Y_robot = Y_robot_blue;
    Color = Color_blue;
else
    % Không có đối tượng được phát hiện
    X_robot = 0.0;
    Y_robot = 0.0;
    Color = NaN;
end

hold off
drawnow;  % Cập nhật hiển thị

    catch
```

```
    disp('Lỗi trong quá trình xử lý hình ảnh.');
end

alpha_robot = 90-atand(X_robot/Y_robot);
a_robot = KhoangCach*cosd(alpha_robot);
b_robot = KhoangCach*sind(alpha_robot);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%

Px = X_robot-a_robot;
Py = Y_robot-b_robot;
Pz = ChieuCao;

d1=130.2;
a2=140.01;
a3=140.0;

r1 = sqrt(Px^2+Py^2);
r2 = Pz - d1;
r3 = sqrt(r1^2+r2^2);

phi1 = acosd((a3^2-a2^2-r3^2)/(-2.0*a2*r3));
phi2 = atand(r2/r1);
phi3 = acosd((r3^2-a2^2-a3^2)/(-2.0*a2*a3));

X_Object = atand(Py/Px);
Y_Object = -(phi1 + phi2);
Z_Object = 180.0 - phi3;

set(handles.edit7,'string',num2str(X_robot));
set(handles.edit8,'string',num2str(Y_robot));

if (Color==1.0)
    set(handles.edit9,'string','Red');
elseif (Color==2.0)
    set(handles.edit9,'string','Green');
elseif (Color==3.0)
    set(handles.edit9,'string','Blue');
else
    set(handles.edit9,'string','Error');
end

if (X_robot == 0.0 && Y_robot == 0.0)
    count = 1;
    fprintf(s, '%.2f\n', objectRemoved);
    pause(0.01);
    data = fscanf(s, '%s')
else
    count = 0;
end

while (count==0)
    data = fscanf(s, '%s')
```

```
if (strcmp(data,'HOME')||strcmp(data,'ENDHOME'))
    SerialData = 555.0;
    fprintf(s, '%.2f\n', X_Object);
    pause(0.01);
    fprintf(s, '%.2f\n', Y_Object);
    pause(0.01);
    fprintf(s, '%.2f\n', Z_Object);
    pause(0.01);
    fprintf(s, '%.2f\n', Color);
    pause(0.01);
    fprintf(s, '%.2f\n', SerialData);
    pause(2);
    count=1;
end


if strcmp(data,'WAIT')
    fprintf(s, '%.2f\n', 222.0);
    pause(0.01);
    count=1;
end

if strcmp(data,'HOME_NEW')
    theta1=0.0;
    theta2=-130.0;
    theta3=114.0;
    FK(theta1, theta2, theta3, handles);
    SetTheta123(handles, theta1, theta2, theta3);
    count=1;
end

if strcmp(data,'STEPX')
    data=fscanf(s, '%s')
    SpeedX=SpeedX+1;
    if (SpeedX==20)
        theta1=str2double(data);
        FK(theta1, theta2, theta3, handles);
        SetTheta123(handles, theta1, theta2, theta3);
        SpeedX=0;
        count=1;
    end
    if (str2double(data)==round(X_Object))
        theta1=str2double(data);
        FK(theta1, theta2, theta3, handles);
        SetTheta123(handles, theta1, theta2, theta3);
        SpeedX=0;
        count=1;
    end
end

if strcmp(data,'STEPY')
    data=fscanf(s, '%s')
    SpeedY=SpeedY+1;
    if (SpeedY==20)
        theta2=str2double(data);
        FK(theta1, theta2, theta3, handles);
        SetTheta123(handles, theta1, theta2, theta3);
```

```
            SpeedY=0;
            count=1;
        end
        if (str2double(data)==round(Y_Object))
            theta2=str2double(data);
            FK(theta1, theta2, theta3, handles);
            SetTheta123(handles, theta1, theta2, theta3);
            SpeedX=0;
            count=1;
        end
    end

    if strcmp(data,'STEPZ')
        data=fscanf(s, '%s')
        SpeedZ=SpeedZ+1;
        if (SpeedZ==20)
            theta3=str2double(data);
            FK(theta1, theta2, theta3, handles);
            SetTheta123(handles, theta1, theta2, theta3);
            SpeedZ=0;
            count=1;
        end
        if (str2double(data)==round(Z_Object))
            theta3=str2double(data);
            FK(theta1, theta2, theta3, handles);
            SetTheta123(handles, theta1, theta2, theta3);
            SpeedX=0;
            count=1;
        end
    end

    if strcmp(data,'END')
        theta1=0.0;
        theta2=-130.0;
        theta3=114.0;
        FK(theta1, theta2, theta3, handles);
        SetTheta123(handles, theta1, theta2, theta3);
        count = 1;
        SerialData = 0.0;
        pause(10);
    end


    if strcmp(data,'Gripper')
        count=1;
    end
  end
end



% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% clc;
% clear;
close all;
```

```
% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
Home(handles);


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
theta1=str2double(get(handles.edit1,'string'));
theta2=str2double(get(handles.edit2,'string'));
theta3=str2double(get(handles.edit3,'string'));

FK(theta1, theta2, theta3, handles);
SetTheta123(handles, theta1, theta2, theta3);


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
Px=str2double(get(handles.edit4,'string'));
Py=str2double(get(handles.edit5,'string'));
Pz=str2double(get(handles.edit6,'string'));

IK(Px, Py, Pz, handles);
SetPxyz(handles, Px, Py, Pz);


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
SliderFK(handles);

% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider2_Callback(hObject, eventdata, handles)
SliderFK(handles);

% --- Executes during object creation, after setting all properties.
function slider2_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider3_Callback(hObject, eventdata, handles)
SliderFK(handles);

% --- Executes during object creation, after setting all properties.
function slider3_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes on slider movement.
function slider4_Callback(hObject, eventdata, handles)
SliderIK(handles);

% --- Executes during object creation, after setting all properties.
function slider4_CreateFcn(hObject, eventdata, ~)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider5_Callback(hObject, eventdata, handles)
SliderIK(handles);

% --- Executes during object creation, after setting all properties.
function slider5_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider6_Callback(hObject, eventdata, handles)
SliderIK(handles);

% --- Executes during object creation, after setting all properties.
function slider6_CreateFcn(hObject, eventdata, handles)
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


function edit1_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit2_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit3_Callback(hObject, eventdata, handles)
function edit3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit4_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit5_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit6_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit7_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit8_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit9_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function SetPxyz(handles, Px, Py, Pz)
set(handles.slider4,'value',Px);
set(handles.slider5,'value',Py);
set(handles.slider6,'value',Pz);
set(handles.edit4,'string',num2str(Px));
set(handles.edit5,'string',num2str(Py));
set(handles.edit6,'string',num2str(Pz));
```

```
function SetTheta123(handles, theta1, theta2, theta3)
set(handles.slider1,'value',theta1);
set(handles.slider2,'value',theta2);
set(handles.slider3,'value',theta3);
set(handles.edit1,'string',num2str(theta1));
set(handles.edit2,'string',num2str(theta2));
set(handles.edit3,'string',num2str(theta3));

theta1T=theta1*pi/180.0;
theta2T=theta2*pi/180.0;
theta3T=theta3*pi/180.0;

L(1) = Link([theta1T,130.2,2.83,-pi/2,0.0]);
L(2) = Link([theta2T,-1.25,140.01,0.0]);
L(3) = Link([theta3T,0.75,140,0.0]);

axes(handles.axes2);
RobotArm = SerialLink(L);
RobotArm.name = 'RobotArm';
RobotArm.plot([theta1T, theta2T, theta3T]);

ModelName = 'RobotArm_MoPhong';

theta1S=-theta1-90.0;
theta2S=theta2;
theta3S=theta3;

set_param([ModelName '/Slider Gain1'],'Gain',num2str(theta1S));
set_param([ModelName '/Slider Gain2'],'Gain',num2str(theta2S));
set_param([ModelName '/Slider Gain3'],'Gain',num2str(theta3S));


function FK(theta1, theta2, theta3, handles)
a1=2.83;      alpha1=-90.0;    d1=130.2;
a2=140.01;    alpha2=0.0;      d2=-1.25;
a3=140.0;     alpha3=0.0;      d3=0.75;

T10=[
   cosd(theta1)   -cosd(alpha1)*sind(theta1)    sind(alpha1)*sind(theta1)   a1*cosd(theta1)
   sind(theta1)    cosd(alpha1)*cosd(theta1)   -sind(alpha1)*cosd(theta1)   a1*sind(theta1)
   0.0             sind(alpha1)                cosd(alpha1)                d1
   0.0               0.0                       0.0                 1.0
   ];

T21=[
   cosd(theta2)   -cosd(alpha2)*sind(theta2)    sind(alpha2)*sind(theta2)   a2*cosd(theta2)
   sind(theta2)    cosd(alpha2)*cosd(theta2)   -sind(alpha2)*cosd(theta2)   a2*sind(theta2)
   0.0             sind(alpha2)                cosd(alpha2)                d2
   0.0               0.0                       0.0                 1.0
   ];

T32=[
   cosd(theta3)   -cosd(alpha3)*sind(theta3)    sind(alpha3)*sind(theta3)   a3*cosd(theta3)
   sind(theta3)    cosd(alpha3)*cosd(theta3)   -sind(alpha3)*cosd(theta3)   a3*sind(theta3)
   0.0             sind(alpha3)                cosd(alpha3)                d3
   0.0               0.0                       0.0                 1.0
   ];
```

```
T=T10*T21*T32;

Px=T(1,4);
Py=T(2,4);
Pz=T(3,4);

SetPxyz(handles, Px, Py, Pz);


function IK(Px, Py, Pz, handles)
d1=130.2;
a2=140.01;
a3=140.0;

r1 = sqrt(Px^2+Py^2);
r2 = Pz - d1;
r3 = sqrt(r1^2+r2^2);

phi1 = acosd((a3^2-a2^2-r3^2)/(-2.0*a2*r3));
phi2 = atand(r2/r1);
phi3 = acosd((r3^2-a2^2-a3^2)/(-2.0*a2*a3));

theta1 = atand(Py/Px);
theta2 = -(phi1 + phi2);
theta3 = 180.0 - phi3;
SetTheta123(handles, theta1, theta2, theta3);


function SliderFK(handles)
theta1=get(handles.slider1,'value');
theta2=get(handles.slider2,'value');
theta3=get(handles.slider3,'value');

FK(theta1, theta2, theta3, handles);
SetTheta123(handles, theta1, theta2, theta3);


function SliderIK(handles)
Px=get(handles.slider4,'value');
Py=get(handles.slider5,'value');
Pz=get(handles.slider6,'value');

IK(Px, Py, Pz, handles);
SetPxyz(handles, Px, Py, Pz);


function Home(handles)
theta1=0.0;
theta2=-130.0;
theta3=114.0;

FK(theta1, theta2, theta3, handles);
SetTheta123(handles, theta1, theta2, theta3);
```