

BỘ GIÁO DỤC & ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN



ĐỒ ÁN TỐT NGHIỆP

**ĐỀ TÀI: ROBOT TỰ ĐỘNG GẮP SẢN PHẨM
ỨNG DỤNG CÔNG NGHỆ XỬ LÝ ẢNH**

GVHD: TS. NGUYỄN VĂN THÁI

SVTH:

BÙI MINH QUANG

MSSV: 16151222

NGUYỄN QUỐC HỌC

MSSV: 16151167

Tp. Hồ Chí Minh tháng 08 năm 2020

Tp. HCM, ngày 10 tháng 08 năm 2020

NHIỆM VỤ ĐÒ ÁN TỐT NGHIỆP

Ho và tên sinh viên 1: Bùi Minh Quang MSSV: 16151222

MSSV: 16151222

Ho và tên sinh viên 2: Nguyễn Quốc Học MSSV: 16151167

Chuyên ngành: Công nghệ Kỹ thuật

và Tu đồng hóa

Hê đàò tao: Đ

Khóa: 2016 Lớp: 161511B

I. TÊN ĐỀ TÀI

ỨNG DỤNG CÔNG NGHỆ XỬ LÝ ẢNH

THE DING SONG NEAR THE BRIDGE

III. NHẬN VỊ

1. Cố gắng liên hệ

1. Các số liệu ban đầu:
(ghi những thông số, tập tài liệu tín hiệu, hình ảnh, ...).....
 2. Nội dung thực hiện:
 - Thiết kế và thi công mô hình 2 băng tải và cơ cấu gấp sản phẩm
 - Nghiên cứu về kit STM32F407G, động cơ step 42x42, module driver DRV8825
 - Nghiên cứu tìm hiểu các thuật toán xử lý ảnh và lập trình trên Python
 - Tạo app điều khiển trên máy tính sử dụng Python
 - Tích hợp camera livestream về app

III. NGÀY GIAO NHIỆM VỤ: 08/03/2020

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

V. HỌ VÀ TÊN CÁN BỘ HƯỚNG DẪN: TS. NGUYỄN VĂN THÁI

CÁN BỘ HƯỚNG DẪN

BỘ MÔN TỰ ĐÔNG ĐIỀU KHIỂN

Jd
TS. Nguyễn Văn Linh

LỊCH TRÌNH THỰC HIỆN ĐO ÁN TỐT NGHIỆP

Họ tên sinh viên 1: Bùi Minh Quang MSSV: 16151222
 Họ tên sinh viên 2: Nguyễn Quốc Học MSSV: 16151167
 Tên đề tài:

ROBOT TỰ ĐỘNG GẤP SẢN PHẨM ỨNG DỤNG CÔNG NGHỆ XỬ LÝ ẢNH

Tuần/ngày	Nội dung	Xác nhận GVHD
Tuần 1 Từ 03/03/2020 đến 08/03/2020	<ul style="list-style-type: none"> Gặp giáo viên hướng dẫn để chọn đề tài. Viết đề cương chi tiết 	
Tuần 2,3 Từ 11/03/2020 đến 22/03/2020	<ul style="list-style-type: none"> Định hình nhiệm vụ và nghiên cứu đề tài 	
Tuần 4 Từ 23/03/2020 đến 29/03/2020	<ul style="list-style-type: none"> Thiết kế sơ đồ khối và lưu đồ 	
Tuần 5, 6...9 Từ 30/03/2020 đến 03/05/2020	<ul style="list-style-type: none"> Nghiên cứu thiết kế phần cứng trên phần mềm Solidworks 	
Tuần 12 Từ 18/05/2020 đến 24/05/2020	<ul style="list-style-type: none"> Trao đổi với giáo viên hướng dẫn về bản thiết kế và tiến hành sửa thiết kế theo góp ý của giáo viên hướng dẫn 	
Tuần 13, 14 Từ 25/05/2020 đến 07/06/2020	<ul style="list-style-type: none"> Thi công phần đế và hai băng chuyền 	
Tuần 15, 16 Từ 08/06/2020 đến 21/06/2020	<ul style="list-style-type: none"> Thi công phần cơ cấu gấp của hệ thống 	
Tuần 17 Từ 22/06/2020 đến 28/06/2020	<ul style="list-style-type: none"> Thi công phần điện cho hệ thống 	
Tuần 18, 19 Từ 29/06/2020 đến 12/07/2020	<ul style="list-style-type: none"> Nghiên cứu viết chương trình xử lý ảnh trên phần mềm Python 	

	<ul style="list-style-type: none"> Tạo giao diện điều khiển tkinter trên Python 	
Tuần 20, 21 Từ 13/07/2020 đến 26/07/2020	<ul style="list-style-type: none"> Viết chương trình điều khiển hệ thống Truyền dữ liệu giao tiếp giữa máy tính và vi điều khiển 	
Tuần 22, 23 Từ 27/07/2020 đến 09/08/2020	<ul style="list-style-type: none"> Hoàn thiện sản phẩm Viết và hoàn chỉnh báo cáo 	

GV HƯỚNG DẪN
(Ký và ghi rõ họ và tên)

TS. Nguyễn Văn Hải

TRƯỜNG ĐH SPKT TP. HỒ CHÍ MINH
KHOA ĐIỆN-ĐIỆN TỬ
BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
ĐỘC LẬP - TỰ DO - HẠNH PHÚC
----oo----

Tp. HCM, ngày 10 tháng 08 năm 2020

LỜI CAM ĐOAN

Nhóm chúng em xin cam kết đề tài này là do nhóm tự thực hiện dựa vào một số tài liệu trước đó và không sao chép từ tài liệu hay công trình đã có trước đó.

Nhóm thực hiện đề tài

Bùi Minh Quang Nguyễn Quốc Học

LỜI CẢM ƠN

Trong thời gian làm đồ án tốt nghiệp, nhóm chúng em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô và bạn bè.

Nhóm chúng em xin gửi lời cảm ơn sâu sắc đến thầy Nguyễn Văn Thái. Thầy đã tận tình hướng dẫn, góp ý cho nhóm chúng em trong toàn bộ thời gian thực hiện đồ án để nhóm chúng em có thể hoàn thành đồ án tốt nhất.

Nhóm cũng xin gửi lời cảm ơn chân thành đến quý thầy cô khoa Điện – Điện tử, đặc biệt là quý thầy cô thuộc bộ môn Điều khiển tự động đã tận tình chỉ dạy những kiến thức từ cơ bản đến chuyên sâu để nhóm có kiến thức vững vàng để có thể tiến hành thực hiện và hoàn tất đồ án này.

Xin chân thành gửi lời cảm ơn đến các thầy cô trong hội đồng bảo vệ đã dành ra chút thời gian để xem bài luận văn tốt nghiệp này, giúp nhóm chúng em chỉ ra những mặt tích cực và hạn chế của đồ án.

Cuối cùng, nhóm xin gửi lời cảm ơn đến bậc cha mẹ, người thân đã động viên và giúp đỡ nhóm chúng em trong suốt con đường học tập cũng như quá trình nghiên cứu và thực hiện đồ án. Đồng thời cảm ơn các anh chị đi trước và bạn bè đã có những lời khuyên, lời góp ý chân thành để nhóm chúng em hoàn thiện hơn và hoàn thành đồ án đúng thời hạn.

Mặc dù đã cố gắng thực hiện đồ án một cách tốt nhất nhưng do còn hạn chế về mặt kiến thức nên nhóm chúng em còn nhiều thiếu sót về nội dung và hình thức. Nhóm chúng em hy vọng quý thầy cô thông cảm và rất mong nhận được những ý kiến quý báu từ quý thầy cô để nhóm chúng em có thể phát triển hơn.

Một lần nữa nhóm chúng em xin chân thành cảm ơn!

Nhóm thực hiện đề tài

Bùi Minh Quang

Nguyễn Quốc Học

MỤC LỤC

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP.....	III
LỊCH TRÌNH THỰC HIỆN ĐỒ ÁN TỐT NGHIỆP	IV
LỜI CAM ĐOAN	VI
LỜI CẢM ƠN	VII
MỤC LỤC	VIII
LIỆT KÊ HÌNH VẼ	X
LIỆT KÊ BẢNG	XIII
TÓM TẮT	XIVV
CHƯƠNG 1. TỔNG QUAN	1
1.1. ĐẶT VẤN ĐỀ	1
1.2. MỤC TIÊU	1
1.3. NỘI DUNG NGHIÊN CỨU	1
1.4. GIỚI HẠN	2
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	3
2.1. KIT STM32F407G DISCOVERY 1	3
2.1.1. Giới thiệu chung về dòng ARM Cortex - M.....	3
2.1.2. Tính năng, đặc điểm của KIT STM32F407G – DISC1.....	4
2.1.3. Chuẩn giao tiếp nối tiếp UART	6
2.2. ĐỘNG CƠ STEP MOTOR:	7
2.2.1. Động cơ bước là gì?	7
2.2.2. Ưu nhược điểm của động cơ bước:	8
2.2. VÀI NÉT VỀ XỬ LÝ ẢNH VÀ MỘT SỐ GIẢI THUẬT	8
2.2.1. Giới thiệu về xử lý ảnh	8
2.2.2. Ảnh số	9
2.2.3. Ảnh xám.....	9
2.2.4. Ảnh màu.....	10
2.2.5. Chuyển từ ảnh màu sang ảnh Gray (ảnh xám).....	10
2.2.6. Chuyển từ ảnh xám sang ảnh nhị phân	12
2.2.7. Lọc nhiễu ảnh (Opening and closing).....	12
2.2.8. Giải thuật phát hiện cạnh Canny	13
2.3 NGÔN NGỮ LẬP TRÌNH VÀ MỘT SỐ THƯ VIỆN SỬ DỤNG.....	14
2.3.1 Ngôn ngữ lập trình Python.....	14
2.3.2. Thư viện OpenCV	15
CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ	17
3.1. TÍNH TOÁN VÀ THIẾT KẾ PHẦN CƠ KHÍ	17
3.1.1. Thiết kế khung, băng tải	17
3.1.1.1. Yêu cầu thiết kế	17
3.1.1.2. Thiết kế băng tải.....	17
3.1.2. Thiết kế cơ cấu gấp sản phẩm.....	20

3.1.2.1. Yêu cầu thiết kế	20
3.1.2.2. Thiết kế cơ cấu gấp	20
3.1.3. Thiết kế sơ đồ khôi của hệ thống	23
3.1.4. Chọn thiết bị cho các khôi.	24
3.1.4.1. Khối xử lý trung tâm.....	24
3.1.4.2. Khối xử lý ảnh	25
3.1.4.3. Khối tín hiệu đầu vào (công tác hành trình)	26
3.1.4.3. Khối động lực	26
3.1.4.6. Con lăn	30
3.1.4.7. Van khí nén, van chân không, đầu tool.....	30
3.2. THIẾT KẾ MẠCH ĐIỀU KHIỂN CỦA HỆ THỐNG	33
3.3. THIẾT KẾ PHẦN MỀM	36
3.3.1. Thiết lập phần cứng cho vi xử lý trên phần mềm STM32CubeMX	37
3.3.2. Lập trình cho STM32 trên Keil uVision5	38
3.3.3. Thuật toán xử lý ảnh	39
3.3.3.1. Thuật toán xử lý ảnh phát hiện sản phẩm	39
3.3.3.2. Thuật toán xử lý ảnh phát hiện thùng đi vào	44
CHƯƠNG 4. THI CÔNG HỆ THỐNG	47
4.1. YÊU CẦU ĐIỀU KHIỂN	47
4.2. MÔ TẢ HOẠT ĐỘNG CỦA HỆ THỐNG	47
4.3. THI CÔNG HỆ THỐNG	47
4.3.1. Thi công cơ khí	47
4.3.2. Thi công mạch điều khiển.....	47
CHƯƠNG 5. KẾT QUẢ THỰC HIỆN	49
5.1. KẾT QUẢ THỰC HIỆN	49
5.1.1 Kết quả thi công cơ khí	49
5.1.1.1 Kết quả thi công băng chuyền.....	49
5.1.1.2 Kết quả thi hệ thống gấp và thả sản phẩm.	50
5.1.2. Kết quả thi công mạch điều khiển	51
5.1.3. Kết quả điều khiển và giám sát hệ thống.....	52
5.2. ĐÁNH GIÁ HOẠT ĐỘNG CỦA HỆ THỐNG	54
CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	55
6.1. KẾT LUẬN	55
6.1.1. Ưu điểm	55
6.1.2. Hạn chế	55
6.2. HƯỚNG PHÁT TRIỂN	55
TÀI LIỆU THAM KHẢO	56
PHỤ LỤC	57

LIỆT KÊ HÌNH VẼ

Hình	Trang
Hình 2.1. Kit STM32F407G – Disc1.....	3
Hình 2.2. Sơ đồ khói phần cứng STM32F407G-DISC1	5
Hình 2.3. Bố cục board STM32F407G-Disc1	6
Hình 2.4. Sơ đồ ghép nối STM32F407G-DISC1 máy tính	7
Hình 2.5. Cấu tạo động cơ bước	8
Hình 2.6. Các bước cơ bản trong xử lý ảnh	9
Hình 2.7. Chuyển ảnh RGB sang ảnh Gray với phương pháp Average	10
Hình 2.8. Chuyển ảnh RGB sang ảnh Gray với phương pháp Lightness	11
Hình 2.9. Chuyển ảnh RGB sang ảnh Gray với phương pháp Linear Luminance	11
Hình 2.10. Chuyển ảnh xám sang ảnh nhị phân	12
Hình 2.11. Sự khác nhau giữa Opening và Closing.....	12
Hình 2.12. Opening process.....	13
Hình 2.13. Closing process	13
Hình 2.14. Kết quả lệnh cv2.Canny	14
Hình 3.1. Bản vẽ thiết kế của hệ thống	17
Hình 3.2. Bản vẽ cơ khí con lăn.....	18
Hình 3.3. Bản vẽ gói đỡ con lăn	18
Hình 3.4. Thiết kế 3D của băng tải	19
Hình 3.5. Bản vẽ 2D thiết của băng tải	19
Hình 3.6. Cơ cấu gấp sản phẩm	20
Hình 3.7. Bản vẽ 2D cơ cấu gấp	21
Hình 3.8. Bản vẽ thiết kế bộ con lăn nhôm định hình 40x20 1	21
Hình 3.9. Bản vẽ thiết kế bộ con lăn nhôm định hình 40x20 2	22
Hình 3.10. Bản vẽ thiết kế bộ con lăn nhôm định hình 40x20 3	22
Hình 3.11. Sơ đồ khói của hệ thống.....	23
Hình 3.12. Bản vẽ cơ khí của board STM32F407G-DISC1.....	24

Hình 3.13. Webcam C720.....	25
Hình 3.14. Công tắc hành trình Endstop.....	26
Hình 3.15. Động cơ bước Nema17	27
Hình 3.16. Sơ đồ dây step motor	27
Hình 3.17. Sơ đồ kết nối chân Driver DRV8825.....	28
Hình 3.18. Bộ truyền Vit-me	28
Hình 3.19. Khớp nối vit-me 5-8mm	29
Hình 3.20. Dây đai GT2.....	29
Hình 3.21. Puly GT2.....	30
Hình 3.22. Ròng rọc 2GT20	30
Hình 3.23. Con lăn V-slot	30
Hình 3.24. Sơ đồ kết nối ống hơi	31
Hình 3.25. Van khí nén 3/2.....	31
Hình 3.26. Van chân không CV-10HS	32
Hình 3.27. Đầu tool hút.....	33
Hình 3.28. Mạch nguyên lý điều khiển relay.....	34
Hình 3.29. Mạch nguyên lý 1	35
Hình 3.30. Mạch nguyên lý 2	35
Hình 3.31. Thiết kế mạch PCB	36
Hình 3.32. Cài đặt GPIO cho các chân STM32.....	37
Hình 3.33. Cài đặt thông số truyền thông UART	37
Hình 3.34. Lưu đồ giải thuật cho vi điều khiển	38
Hình 3.35. Ảnh hệ thống chụp được từ camera	39
Hình 3.36. Lưu đồ xử lý ảnh sản phẩm.....	40
Hình 3.37. Ảnh đã được cắt và scale lại đúng tỉ lệ thực tế	41
Hình 3.38. Ảnh nhị phân với ngưỡng thresh được chọn trên thanh slider	41
Hình 3.39. Tâm panel đã được xác định tương đối chính xác	42
Hình 3.40. Vị trí của panel trong mô hình	42
Hình 3.41. Vùng xử lí được cắt	43

Hình 3.42. Lưu đồ xử lý ảnh phát hiện thùng.....	44
Hình 3.43. Ảnh vùng xử lý thùng trước và sau khi xử lý opening ảnh	45
Hình 3.44. Ảnh phát hiện biên của thùng đi vào	45
Hình 3.45. Vị trí thùng trong vùng xử lý ảnh và thực tế	46
Hình 5.1. Mô hình 2 băng tải vận chuyển sản phẩm và thùng	49
Hình 5.2. Hệ thống gấp và thả sản phẩm	50
Hình 5.3. Các cơ cấu di chuyển của cơ cấu gấp sản phẩm	51
Hình 5.4. Bảng điều khiển vật lý	51
Hình 5.5. Mạch điều khiển hệ thống.....	52
Hình 5.6. Hệ thống đang gấp vật bỏ vào thùng	53
Hình 5.7. Giao diện điều khiển hệ thống trên máy tính.....	54

LIỆT KÊ BẢNG

Bảng	Trang
Bảng 3.1. Thông số kĩ thuật của động cơ	27
Bảng 3.2. Thông số kĩ thuật của khớp nối 5-8mm	29

TÓM TẮT

Ngày nay với sự phát triển của khoa học công nghệ, nhất là trong lĩnh vực điều khiển và tự động, rất nhiều máy, hệ thống tự động thông minh ra đời, đã làm thay đổi về mọi mặt cuộc sống của con người. Cụ thể trong công nghiệp là các máy tự động, các trạm sản xuất linh hoạt, các nhà máy thông minh... Với những chiếc máy này việc sản xuất của con người đã trở nên thuận tiện hơn. Trong báo cáo này, nhóm muốn đề cập đến một trong những hệ thống tự động ứng dụng trong công nghiệp.

Đề tài “ROBOT TỰ ĐỘNG GẮP SẢN PHẨM ỨNG DỤNG CÔNG NGHỆ XỬ LÝ ẢNH” xây dựng một mô hình hoàn chỉnh, hoạt động linh hoạt và ổn định có khả năng điều khiển cả bằng tay và tự động.

Nhóm chúng em thực hiện đề tài này nhằm tạo ra một công đoạn trong dây chuyền sản xuất tự động, hạn chế sức người trong sản suất. Đồng thời qua đây cũng tạo nên một mô hình phục vụ giáo dục, khơi dậy niềm đam mê công nghệ của các bạn trẻ.

Nguyên lý hoạt động được dựa trên chuyển động 3 trực của máy in 3D, lập trình bằng ngôn ngữ C sử dụng phần mềm Keil uVision5 và lập trình xử lý ảnh bằng Python.

Phần cứng bao gồm động cơ step 42x42, board STM34F407G, Camera Logitech C270, Driver DVR8825, nguồn tổ ong 24V-12A. Thiết kế trên phần mềm đồ họa Solidworks. Nhóm chúng em tiến hành gia công bằng công nghệ in 3D với vật liệu nhựa PLA, CNC lazer mica, sử dụng nhôm hộp và nhôm định hình cho phần khung.

Sau một thời gian dài, nhóm chúng em đã chế tạo thành công mô hình có khả năng gấp chính xác sản phẩm bỏ vào thùng khi camera phát hiện có sản phẩm đến. Khi mà vật không khớp do bị lệch so với thùng thì cơ cấu xoay sẽ tự động xoay vật sao cho khớp với thùng nhờ quá trình xử lý ảnh bằng camera. Hệ thống sẽ dừng và báo lỗi khi có sản phẩm đi qua mà không được gấp bỏ vào thùng, khi ấy người dùng có thể điều khiển bằng tay để khắc phục sự cố. Ngoài ra hệ thống còn có chức năng quản lý số lượng sản phẩm cũng như số thùng đã được đóng gói.

Chương 1. TỔNG QUAN

1.1. ĐẶT VẤN ĐỀ

Trong những năm gần đây, cùng với sự phát triển của xã hội thì khoa học và kỹ thuật cũng phát triển với những bước tiến vượt bậc. Tất cả những phát minh được tạo ra nhằm phục vụ cho con người, giúp nâng cao chất lượng cuộc sống và mang lại cảm giác thoải mái hơn. Cùng với đó là sự phát triển vượt bậc trong lĩnh vực tự động hóa, những dây chuyền tự động trong sản xuất dần dần thay thế sức lao động con người, mang lại hiệu quả và năng suất cao. Chúng ta có thể kể đến những hệ thống tự động điển hình như các hệ thống phân loại sản phẩm, phân loại nông sản, hệ thống bãi giữ xe thông minh, nhà thông minh, thành phố thông minh...

Đề tài tập trung nghiên cứu và thiết kế một công đoạn trong dây chuyền tự động công nghiệp. Đó là khâu cuối cùng trong dây chuyền sản xuất sản phẩm tự động. Sau khi sản phẩm đi ra sẽ được đóng gói và được đưa vào kho. Cụ thể ở đây, nhóm chọn đối tượng là những tấm màn hình (panel) tivi.

1.2. MỤC TIÊU

Mục tiêu chính của đề tài là thiết kế ra một hệ thống có khả năng phát hiện sản phẩm đến và gấp bở vào thùng chứa. Hệ thống sử dụng camera Logitech C270 để lấy ảnh đầu vào, sau đó tiến hành xử lý bằng một số thuật toán xử lý ảnh để nhận diện vật sau đó truyền serial qua STM32F407G để điều khiển các động cơ step.

Khi nghiên cứu thực hiện đề tài, nhóm muốn áp dụng những kiến thức của bản thân về vi điều khiển, ngôn ngữ lập trình nhằm tạo ra những sản phẩm, thiết bị có khả năng ứng dụng trong thực tế.

Ngoài ra quá trình nghiên cứu thực hiện đề tài là cơ hội để nhóm tự củng cố và bổ sung những kiến thức đã được học ở trường, đồng thời phát huy tính sáng tạo, khả năng giải quyết một vấn đề theo yêu cầu đặt ra.

1.3. NỘI DUNG NGHIÊN CỨU

Đề tài bao gồm các khâu từ thiết kế đến khi xây dựng thành công mô hình, trình bày chi tiết các công đoạn tiến hành xây dựng mô hình từ quá trình tìm hiểu cách hoạt động

của hệ thống đến thiết kế cơ khí. Đưa ra phương án lựa chọn và kết nối thiết bị điều khiển, giám sát, thuật toán điều khiển hệ thống. Đề tài cũng nêu lên nhận xét, đánh giá kết quả đạt được và phân tích những hạn chế đưa ra hướng giải quyết và phát triển đề tài trong tương lai. Các nội dung trên được trình bày thành các chương:

CHƯƠNG 1. TỔNG QUAN

- **Chương 1: Tổng quan**

Trong chương này nêu ra được tình hình nghiên cứu hiện nay, lý do và mục tiêu chọn đề tài, đối tượng và phạm vi nghiên cứu của đề tài, phương pháp nghiên cứu và giới hạn của đề tài.

- **Chương 2: Cơ sở lý thuyết**

Trình bày tổng quan về cơ sở lý thuyết các đối tượng được sử dụng trong hệ thống.

- **Chương 3: Tính toán, thiết kế**

Từ yêu cầu đề tài thiết kế hệ thống, thiết kế về cơ khí, gia công. Lựa chọn thiết bị cho phần điện.

- **Chương 4: Thi công hệ thống**

Thi công phần cứng

Lưu đồ, chương trình điều khiển. Lập trình xử lý ảnh, thiết kế giao diện điều khiển và giám sát cho hệ thống.

- **Chương 5: Kết quả và nhận xét**

Kết quả tổng quan thi công phần cứng, phần mềm và đánh giá ưu nhược điểm của hệ thống sau khi thực hiện đề tài.

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

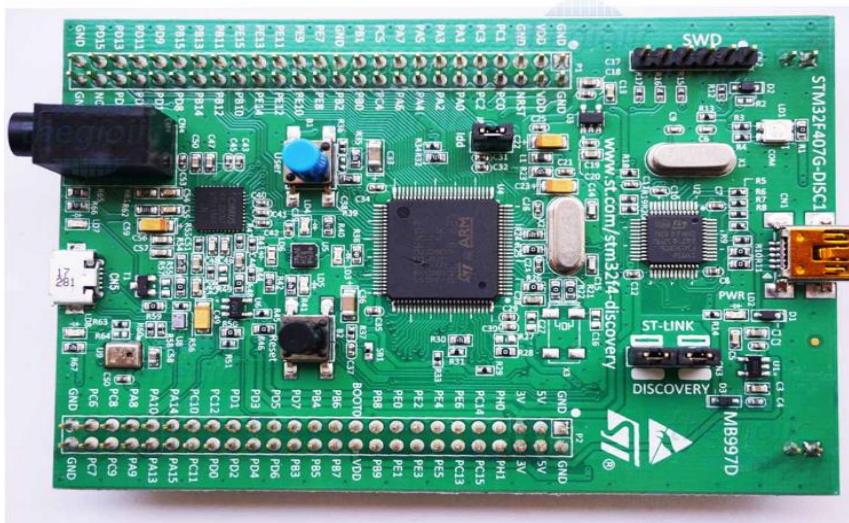
Đưa ra các kết luận về các vấn đề đã hoàn thành và chưa hoàn thành, các mặc định ché. Trình bày hướng phát triển của đề tài trong tương lai.

1.4. GIỚI HẠN

Để có được một hệ thống hoàn chỉnh, đẹp về hình thức và có độ ổn định, chính xác cao khi hoạt động đòi hỏi có sự kết hợp kiến thức của nhiều ngành khác nhau như cơ khí, điện, điện tử... Trong giới hạn kiến thức, thời gian cũng như vấn đề kinh phí trong mức cho phép, đề tài chỉ tập trung nghiên cứu thiết kế hệ thống ở mức độ mô hình. Nghiên cứu giải thuật lập trình và điều khiển giám sát, mà không đi sâu vào việc lựa chọn thiết bị cho tối ưu với mục đích sử dụng ngoài thực tế.

Chương 2. CƠ SỞ LÝ THUYẾT

2.1. KIT STM32F407G DISCOVERY 1



Hình 2.1. Kit STM32F407G – Disc1^[2]

2.1.1. Giới thiệu chung về dòng ARM Cortex - M

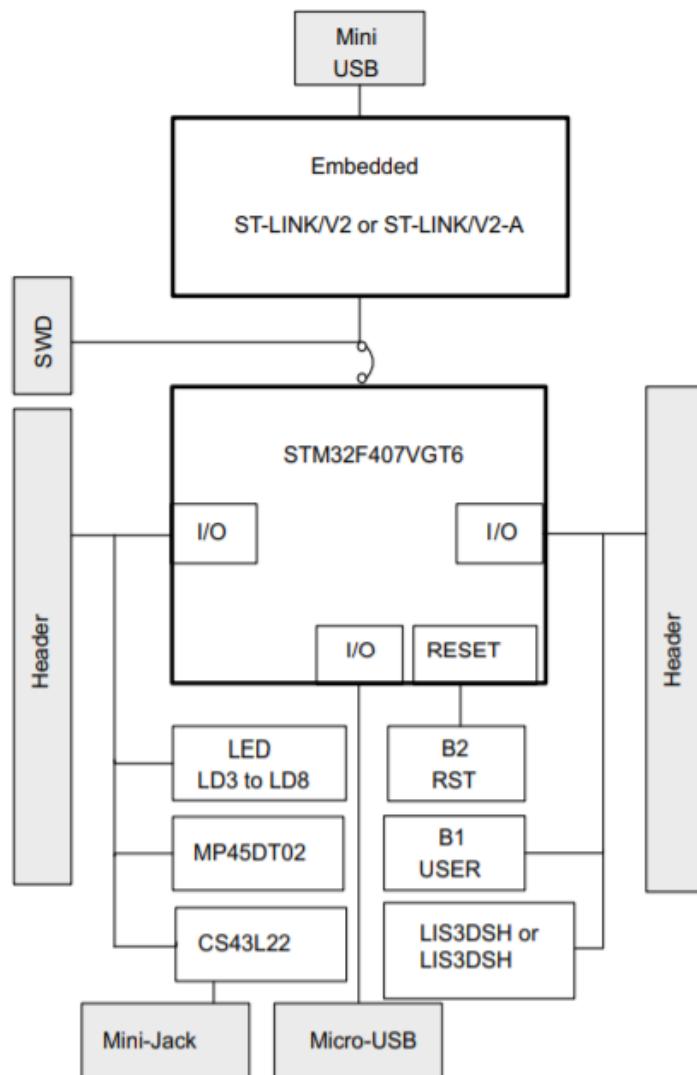
Dòng ARM Cortex là một bộ xử lý thế hệ mới đưa ra một kiến trúc chuẩn cho nhu cầu đa dạng về công nghệ. Không giống như các chip ARM khác, dòng Cortex là một lõi xử lý hoàn thiện, đưa ra một chuẩn CPU và kiến trúc hệ thống chung. Không giống với các kiến trúc ARM trước đó, dòng Cortex cho phép truy cập dữ liệu không xếp hàng (unaligned data, vì chip ARM là kiến trúc 32bit, do đó tất cả các dữ liệu hoặc mã chương trình đều được sắp xếp khít với vùng bộ nhớ là bội số của 4byte). Đặc điểm này cho phép sử dụng hiệu quả SRAM nội. Dòng Cortex còn hỗ trợ việc đặt và xoá các bit bên trong hai vùng 1Mbyte của bộ nhớ bằng phương pháp gọi là bit banding. Đặc điểm này cho phép truy cập hiệu quả tới các thanh ghi ngoại vi và các cờ được dùng trên bộ nhớ SRAM mà không cần một bộ xử lý luận lí (Boolean processor)^[1]. Dòng ARM Cortex có ba họ thiết kế bộ xử lý: Cortex-A, R và M. Bộ xử lý Cortex-A được sử dụng trong các thiết bị lớn hơn như BeagleBone, RaspberryPi, iPhone, iPad và thiết bị Android. Bộ xử lý Cortex-R được thiết kế cho các hệ thống nhúng thời gian thực có độ tin cậy cao như bộ điều khiển bồn nước nóng và màn hình hiển thị chuyến bay. Sê-ri Cortex-M dành cho các hệ thống nhúng, các ứng dụng của vi điều khiển và chi phí thấp. STM32 là dòng Sê-ri Cortex-M.^[2]

2.1.2. Tính năng, đặc điểm của KIT STM32F407G – DISC1

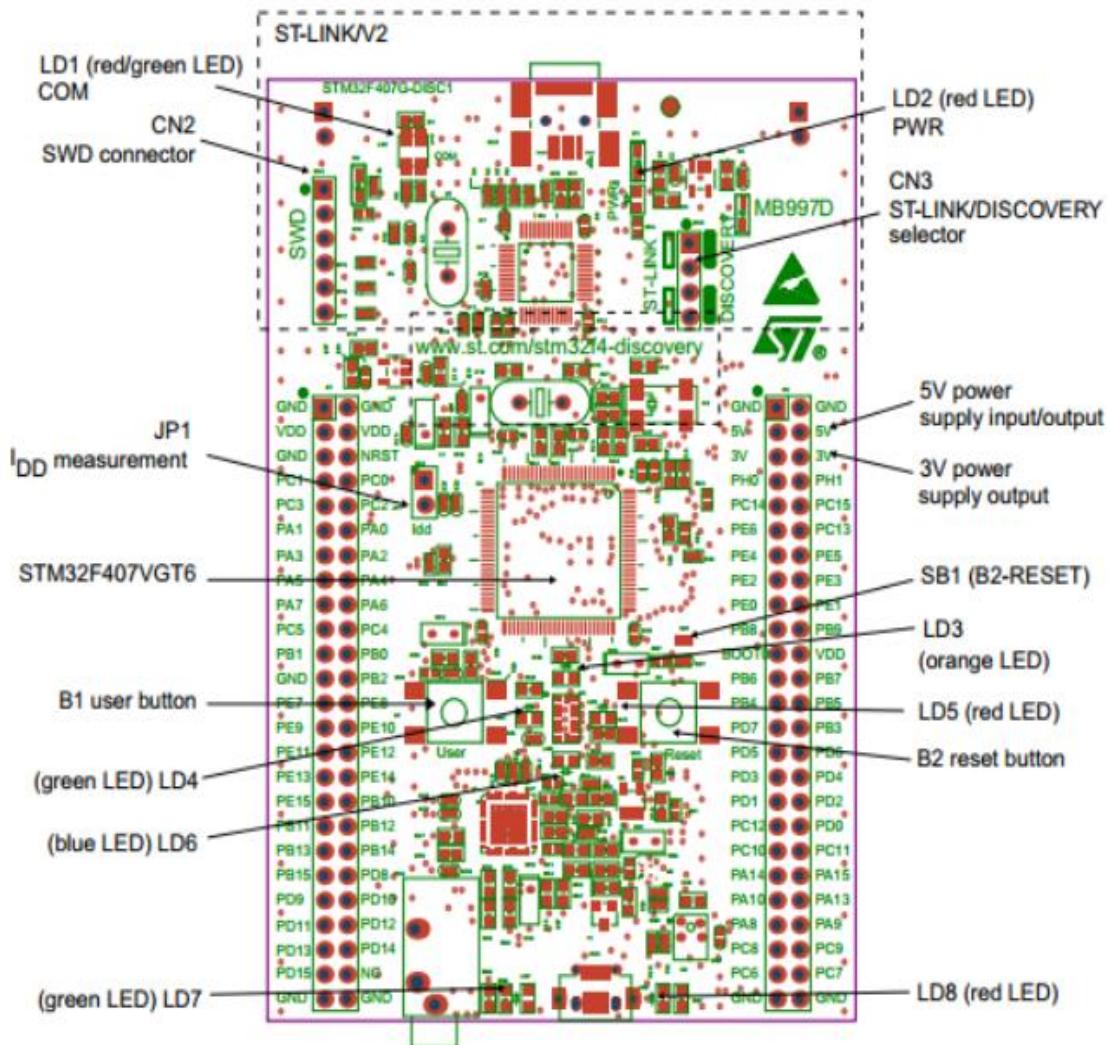
Board DISC1 có rất nhiều tính năng tích hợp để cho phép thử nghiệm mà không phải thêm bất kỳ chip hoặc board mới nào, dưới đây là những tính năng của bộ kit:

- ❖ MCU STM32F407VGT6 có lõi 32-bit ARM Cortex-M4F, Flash 1MB, RAM 192KB trong một gói LQFP100
- ❖ ST-LINK/V2 có chức năng lựa chọn chế độ hoạt động độc lập(có cổng SWD để lập trình và gỡ lỗi)
- ❖ Nguồn cấp: thông qua USB hoặc từ nguồn điện 5V bên ngoài
- ❖ Nguồn cấp ứng dụng bên ngoài: 3V và 5V LIS302DL, cảm biến chuyển động ST MEMS, bộ cảm biến góc 3 trực MP45DT02, bộ cảm biến âm thanh ST MEMS, micro kỹ thuật số đa hướng CS43L22, DAC âm thanh tích hợp trình điều khiển loa lớp D.
- ❖ Tám LED: Hai nút nhấn (người dùng và đặt lại)
 - LD1 (đỏ/xanh) để hiển thị giao tiếp USB
 - LD2 (màu đỏ) hiển thị nguồn 3.3 VDC
 - Bốn LED người dùng, LD3 (màu cam), LD4 (xanh lục), LD5 (đỏ) và LD6 (màu xanh).
 - 2 đèn LED USB OTG LD7 (xanh lá) VBus và LD8 (màu đỏ) thể hiện quá dòng,
- ❖ USB OTG FS với cổng micro-AB.
- ❖ Mở rộng các chân LQFP100 I/O để giúp kết nối nhanh chóng và cho phép bạn dễ dàng sử dụng và sửa lỗi.^[2]

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT



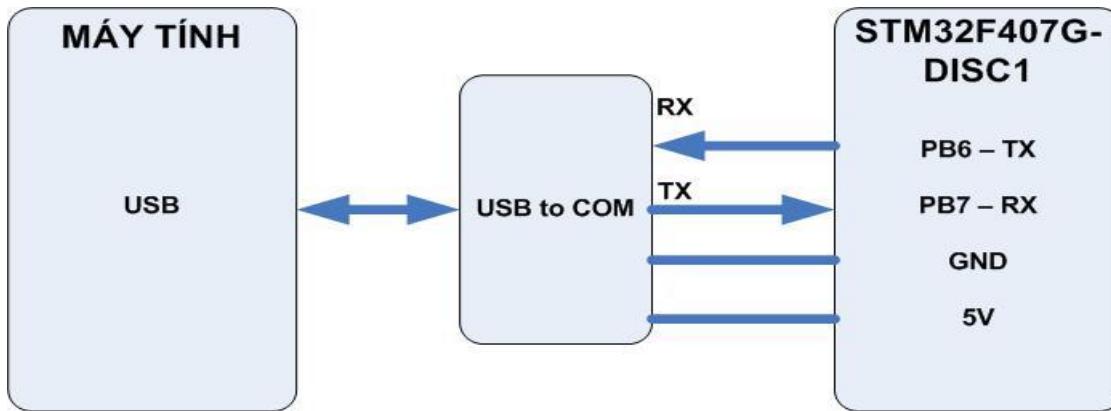
Hình 2.2. Sơ đồ khái niệm phần cứng STM32F407G-DISC1^[2]



Hình 2.3. Bố cục board STM32F407G-Disc1^[2]

2.1.3. Chuẩn giao tiếp nối tiếp UART

UART (Universal Asynchronous Receive/Transmit) là chuẩn giao tiếp truyền thông không đồng bộ. Đây là chuẩn giao tiếp phổ biến và dễ sử dụng, thường dùng trong giao tiếp giữa vđk với nhau hoặc với các thiết bị khác. Vì là giao tiếp không đồng bộ nên hai thiết bị phải được cài đặt thống nhất về khung truyền, tốc độ truyền. Ưu điểm của truyền thông nối tiếp là vi điều khiển có khả năng truyền nhận nhiều dữ liệu, tiết kiệm đường IO, nhưng nhược điểm là không được nhanh như truyền song song và dễ bị mất, lỗi dữ liệu.[3]



Hình 2.4. Sơ đồ ghép nối STM32F407G-DISC1 với máy tính

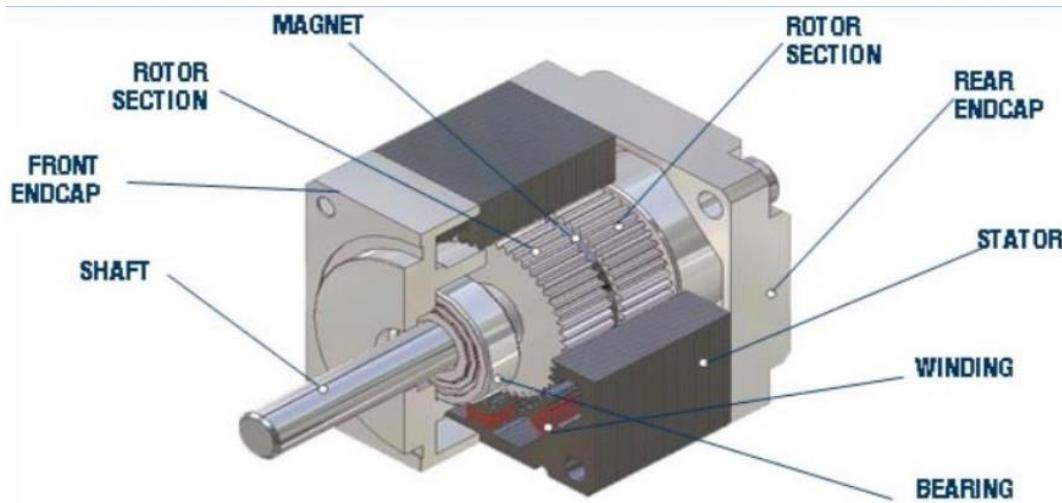
2.2. ĐỘNG CƠ STEP MOTOR:

2.2.1. Động cơ bước là gì?

Động cơ bước là một loại động cơ điện có nguyên lý và ứng dụng khác biệt với đa số các động cơ điện thông thường. Chúng thực chất là một động cơ đồng bộ dùng để biến đổi các tín hiệu điều khiển dưới dạng các xung điện rời rạc kế tiếp nhau thành các chuyển động góc quay hoặc các chuyển động của rôto có khả năng cố định rôto vào các vị trí cần thiết. [5]

Động cơ bước phổ biến trên thị trường được chia thành 3 loại chính là 2 pha, 3 pha và 5 pha tương ứng với góc bước cơ bản là 1.8 độ, 1.2 độ và 0.72 độ. Góc bước cơ bản càng nhỏ thì số lượng cực từ trên rôto càng lớn, ví dụ động cơ có góc bước 1.8 độ thì có 100 cặp cực từ. [5]

Cấu tạo của động cơ bước bao gồm rôto là một dãy các lá nam châm vĩnh cửu xếp chồng lên nhau, trên các lá này lại chia thành các cặp cực đối xứng nhau. Stato được cấu tạo từ sắt từ trên đó có chia các ranh để đặt cuộn dây (động cơ bước 2 pha có 4 cuộn dây). Khi cấp 1 xung điện vào cuộn dây trên stato thì rôto sẽ quay 1 góc bằng góc bước cơ bản.



Hình 2.5. Cấu tạo động cơ bước [5]

2.2.2. Ưu nhược điểm của động cơ bước:

Motor bước khi hoạt động có hiện tượng bị trượt bước do lực từ trên nam châm vĩnh cửu đã yếu nên cho vị trí không chính xác hoặc nguồn điện cấp vào không đủ (VD: động cơ bước có góc bước 1.8 độ nên cần 200 xung thì quay đủ 1 vòng tuy nhiên nếu có hiện tượng trượt bước thì cần nhiều hơn 200 xung mới đủ 1 vòng). [5]

Nhược điểm thứ 2 của động cơ bước đó là ồn và nóng lên khi hoạt động, điều này là hoàn toàn bình thường đối với động cơ bước vì bản thân nó được thiết kế để chịu được sức nóng như vậy. Đối với các driver điều khiển động cơ bước thế hệ mới nhất thì độ ồn và nóng của động cơ đã giảm đáng kể.

Ngược lại thì ưu điểm của động cơ bước đó chính là khả năng cung cấp moment xoắn cực lớn ở dải vận tốc thấp và trung bình, khá bền bỉ, thay thế dễ dàng và giá thành thấp.

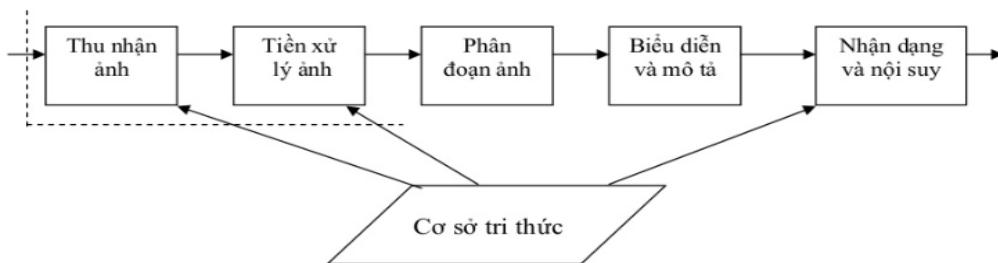
2.2. VÀI NÉT VỀ XỬ LÝ ẢNH VÀ MỘT SỐ GIẢI THUẬT

2.2.1. Giới thiệu về xử lý ảnh

Xử lý ảnh là một lĩnh vực mang tính khoa học và công nghệ. Nó là một ngành khoa học mới mẻ so với nhiều ngành khoa học khác nhưng tốc độ phát triển của nó rất nhanh, kích thích các trung tâm nghiên cứu, ứng dụng, đặc biệt là máy tính chuyên dụng riêng của nó.

Các phương pháp xử lý ảnh bắt đầu từ các ứng dụng chính: nâng cao chất lượng ảnh và phân tích ảnh. Về sau khi các phương pháp tri thức nhân tạo như mạng neural, các thuật toán xử lý hiện đại và cải tiến, công cụ ảnh ngày càng được áp dụng rộng rãi và thu nhiều kết quả khả quan.

Để dễ tưởng tượng, xét các bước cần thiết trong xử lý ảnh. Đầu tiên, ảnh tự nhiên từ thế giới ngoài được thu nhận qua các thiết bị thu (camera, máy chụp ảnh), sau đó được chuyển trực tiếp thành ảnh số tạo thuận lợi cho xử lý tiếp theo



Hình 2.6. Các bước cơ bản trong xử lý ảnh [12]

2.2.2. Ảnh số

Ảnh số là một tập hợp của nhiều điểm ảnh, hay còn gọi là pixel. Mỗi điểm ảnh biểu diễn một màu sắc nhất định (hay độ sáng với ảnh đen trắng) tại một điểm duy nhất, có thể xem một điểm ảnh giống như một chấm nhỏ trong một tấm ảnh màu. Bằng phương pháp đo lường và thống kê một lượng lớn các điểm ảnh, chúng ta hoàn toàn có thể tái cấu trúc các điểm ảnh này thành một ảnh mới gần giống với ảnh gốc. Có thể nói pixel gần giống như các phần tử có cấu trúc hạt trên một ảnh thông thường nhưng được sắp xếp theo từng hàng và cột và chứa các thông tin khác nhau.

Ảnh được biểu diễn dưới dạng một ma trận hai chiều với các pixel được xác định bởi cặp tọa độ (x, y), trong đó, giá trị của pixel tại tọa độ nhất định biểu diễn độ sáng (ảnh đen trắng) hay màu nhất định (ảnh màu).^[12]

Mỗi điểm ảnh tương ứng với một phần của một đối tượng vật lý trong thế giới ba chiều.

Có 2 dạng quan trọng trong ảnh số được dùng với nhiều mục đích khác nhau là ảnh màu và ảnh xám. Trong đó, ảnh màu được cấu trúc từ các pixel màu trong khi ảnh đen trắng được xây dựng từ các pixel có giá trị mức xám khác nhau.^[12]

2.2.3. Ảnh xám

Ảnh xám chỉ bao gồm 2 màu: màu đen và màu trắng. Người ta phân mức đen trắng đó thành L mức, nếu sử dụng số bít B=8 bit để mã hóa mức đen trắng (hay mức xám) thì L được xác định:

$$L=2^B \text{ (trong ví dụ của ta } L=2^8 = 256 \text{ mức)} \quad (2.1)$$

Nếu L bằng 2, B=1, nghĩa là chỉ có 2 mức: mức 0 và mức 1, còn gọi là ảnh nhị phân. Mức 1 ứng với màu sáng, còn mức 0 ứng với màu tối. Nếu L lớn hơn 2 ta có ảnh đa cấp xám,

Nói cách khác, với ảnh nhị phân mỗi điểm ảnh được mã hóa trên 1 bit, còn với ảnh 256 mức, mỗi điểm ảnh được mã hóa trên 8 bit, Như vậy, với ảnh đen trắng: nếu dùng 8 bit (1 byte) để biểu diễn mức xám, số các mức xám có thể biểu diễn được là 256. Mỗi mức xám được biểu diễn dưới dạng là một số nguyên nằm trong khoảng từ 0 đến 255, với mức 0 biểu diễn cho mức cường độ đen nhất và 255 biểu diễn cho mức cường độ sáng nhất.

Ảnh nhị phân khá đơn giản, các phần tử ảnh có thể coi như các phần tử logic. Ứng dụng chính của nó được dùng để phân biệt đối tượng ảnh với nền hay để phân biệt điểm biên với điểm khác.^[13]

2.2.4. Ảnh màu

Ảnh màu theo lý thuyết của Thomas là ảnh tổ hợp từ 3 màu cơ bản: đỏ (R), lục (G), lam (B) và thường thu nhận trên các dải băng tân khác nhau. Với ảnh màu, cách biểu diễn cũng tương tự như với ảnh đen trắng, chỉ khác là các số tại mỗi phân tử của ma trận biểu diễn cho ba màu riêng rẽ gồm: đỏ (red), lục (green) và lam (blue). Để biểu diễn cho một điểm ảnh màu cần 24 bit. 24 bit này được chia thành ba khoảng 8 bit. Mỗi màu cũng phân thành 7 cấp màu khác nhau (thường L=256). Mỗi khoảng này biểu diễn cho cường độ sáng của một trong các màu chính.

Do đó, để lưu trữ ảnh màu người ta có thể lưu trữ từng màu riêng biệt, mỗi màu lưu trữ như một ảnh đa cấp xám. Do đó, không gian nhớ dành cho một ảnh màu lớn gấp 3 lần một ảnh đa cấp xám cùng kích cỡ.^[13]

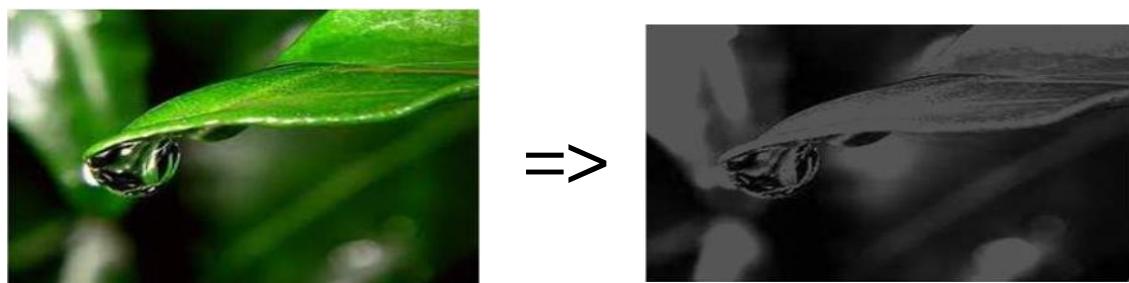
2.2.5. Chuyển từ ảnh màu sang ảnh Gray (ảnh xám)

❖ Phương pháp Average

Phương pháp trung bình là đơn giản nhất. Ta chỉ cần lấy trung bình của ba màu. Vì nó là hình ảnh RGB, nên có nghĩa là ta đã thêm R với G với B và sau đó chia cho 3 để có được hình ảnh thang độ xám mong muốn.

Nó được thực hiện theo cách này.

$$\text{Gray} = (R + G + B) / 3. \quad [12] \quad (2.2)$$



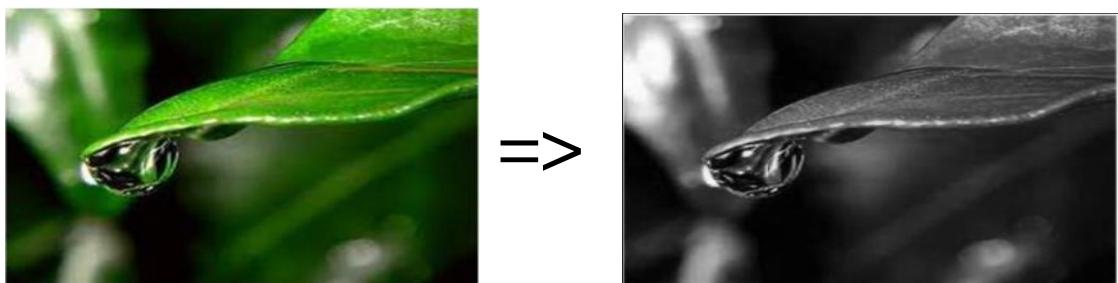
Hình 2.7. Chuyển ảnh RGB sang ảnh Gray với phương pháp Average ^[12]

Vì ba màu khác nhau có ba bước sóng khác nhau và có đóng góp riêng trong việc hình thành hình ảnh, vì vậy khi lấy trung bình tạo nên một ảnh xám khá tối và không như mong muốn.

❖ **Phương pháp Lightness:** phương pháp này có phương trình như sau:

$$\text{Gray} = (\text{Max}(R,G,B) + \text{Min}(R,G,B))/2. \quad (2.3)$$

Áp dụng vào ta được kết quả như sau:



Hình 2.8. Chuyển ảnh RGB sang ảnh Gray với phương pháp Lightness

Ảnh xám tạo ra từ phương pháp này đã chất lượng hơn so với phương pháp Average, ảnh trông dịu hơn có thể nhìn rõ đối tượng nhưng vẫn còn tối.

❖ **Phương pháp Linear Luminance**

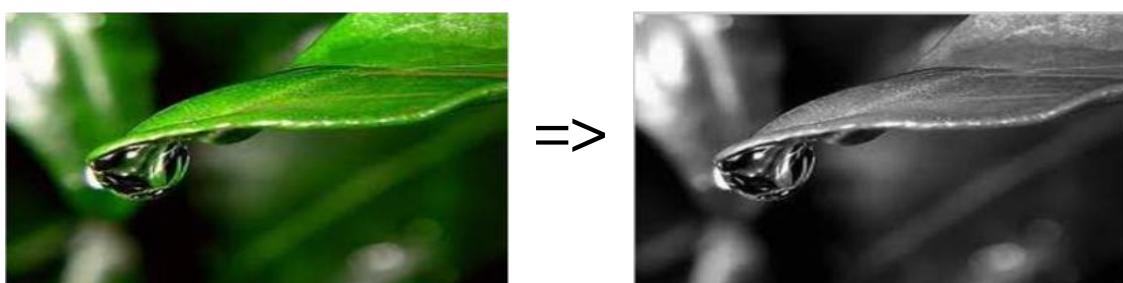
Ta đã thấy vấn đề xảy ra trong phương pháp Average và Lightness. Phương pháp trọng số là một giải pháp cho vấn đề đó. Vì màu đỏ có nhiều bước sóng hơn trong cả ba màu, và màu xanh lá cây là màu không chỉ có ít bước sóng hơn màu đỏ mà còn là màu mang lại hiệu ứng làm dịu mắt hơn.

Điều đó có nghĩa là chúng ta phải giảm sự đóng góp của màu đỏ và tăng sự đóng góp của màu xanh lá cây và đặt sự đóng góp của màu xanh ở giữa hai màu này.

Vì vậy, phương trình mới hình thành là:

$$\text{Gray} = ((0,3 * R) + (0,59 * G) + (0,11 * B)). \quad [12] \quad (2.4)$$

Áp dụng phương trình này vào hình ảnh, chúng ta có được điều này



Hình 2.9. Chuyển ảnh RGB sang ảnh Gray với phương pháp Luminance [12]

So với 2 kết quả của phương pháp trên ảnh ở phương pháp này sáng hơn. Chính vì vậy đây là phương pháp được ưu chuộng và sử dụng nhiều nhất.

2.2.6. Chuyển từ ảnh xám sang ảnh nhị phân

Ảnh nhị phân chủ yếu được dùng để phân biệt đối tượng ảnh với nền hay để phân biệt điểm biên với điểm khác. Bắt đầu với một ảnh xám và ta xác định một giá trị ngưỡng. Sau đó, đối với mỗi pixel của hình ảnh tỷ lệ xám, nếu giá trị của nó nhỏ hơn ngưỡng, thì ta gán cho nó giá trị 0 (màu đen), và lớn hơn ngưỡng ta gán cho nó giá trị 255 (màu trắng).



Hình 2.10. Chuyển ảnh xám sang ảnh nhị phân [13]

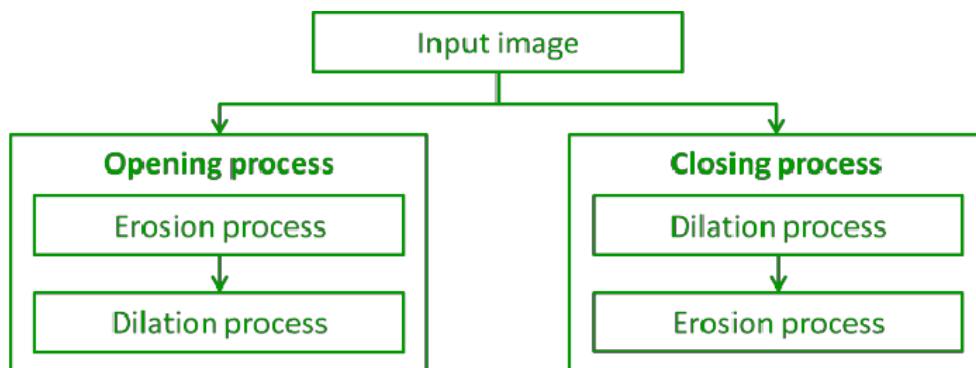
2.2.7. Lọc nhiễu ảnh (Opening and closing)

Opening và **Closing** là các thao tác kép được sử dụng trong xử lý hình ảnh kỹ thuật số để khôi phục hình ảnh bị nhiễu. **Opening** thường được sử dụng để hồi phục hoặc khôi phục ảnh gốc ở mức tối đa có thể. Tính năng **Closing** lại thường được sử dụng để làm mịn đường viền của hình ảnh bị méo và nối lại các điểm đứt gãy hép và các rãnh dài mỏng. **Closing** cũng được sử dụng để loại bỏ các lỗ nhỏ của hình ảnh thu được.

Sự kết hợp giữa **Opening** và **Closing** thường được sử dụng để làm sạch các hiện vật trong hình ảnh được phân đoạn trước khi sử dụng hình ảnh để phân tích kỹ thuật số. [14]

Opening: ảnh gốc → co ảnh (erosion) → giãn ảnh (dilation)

Closing: ảnh gốc → giãn ảnh (dilation) → co ảnh (erosion)



Hình 2.11. Sự khác nhau giữa Opening và Closing [14]

- ❖ Phép toán co ảnh (Erosion):

$$A \ominus B = \{z | (B)_z \subseteq A\} \quad [19] \quad (2.5)$$

- ❖ Phép toán giãn nở ảnh (Dilation):

$$A \oplus B = \{z | (\hat{B})_z \cap A \neq \emptyset\} \quad [19] \quad (2.6)$$

- ❖ Phép toán Openning:

$$A \circ B = (A \ominus B) \oplus B \quad [19] \quad (2.7)$$

- ❖ Phép toán Closing:

$$A \bullet B = (A \oplus B) \ominus B \quad [19] \quad (2.8)$$



Hình 2.12. Opening process [15]



Hình 2.13. Closing process [15]

2.2.8. Giải thuật phát hiện cạnh Canny

Trong hình ảnh, thường tồn tại các thành phần như: vùng trơn, góc / cạnh và nhiễu. Cạnh trong ảnh mang đặc trưng quan trọng, thường là thuộc đối tượng trong ảnh (object). Do đó, để phát hiện cạnh trong ảnh, giải thuật Canny là một trong những giải thuật phổ biến nhất trong xử lý ảnh.

Giải thuật phát hiện cạnh Canny gồm 4 bước chính sau:[16]

- **Giảm nhiễu:**

Vì tính năng phát hiện cạnh dễ bị nhiễu trong ảnh, nên bước đầu tiên là loại bỏ nhiễu trong ảnh bằng bộ lọc Gaussian 5x5

- **Tính Gradient và hướng gradient:**

Hình ảnh được làm mịn được lọc bằng bộ lọc Sobel theo cả hướng ngang và dọc để có được đạo hàm đầu tiên theo hướng ngang (G_x) và hướng dọc (G_y). Từ hai hình ảnh này, chúng ta có thể tìm thấy gradient cạnh và hướng cho mỗi pixel như sau:

$$\begin{aligned} Edge_Gradient (G) &= \sqrt{G_x^2 + G_y^2} \\ Angle (\theta) &= \tan^{-1} \left(\frac{G_y}{G_x} \right) \end{aligned} \quad (2.9)$$

Gradient hướng luôn vuông góc với các cạnh. Nó được gom thành một trong bốn góc đại diện cho các hướng dọc (90°), ngang (0°), chéo trái (135°) và chéo phải (45°).

- **Non-maximum Suppression** (viết tắt NMS):

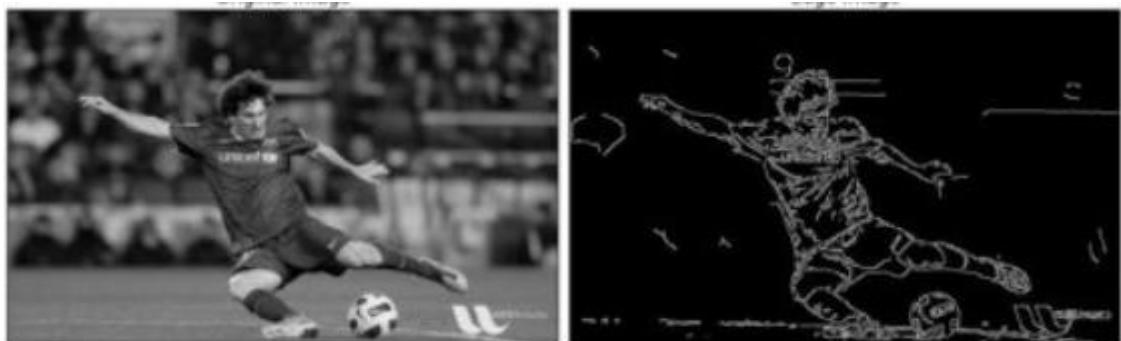
Sau khi nhận được độ lớn và hướng của gradient, quá trình quét toàn bộ hình ảnh được thực hiện để loại bỏ bất kỳ pixel không mong muốn nào có thể không tạo thành cạnh. Đối với điều này, tại mỗi pixel, pixel được kiểm tra xem độ lớn gradient của nó có phải là cực đại cục bộ trong vùng lân cận của nó theo hướng gradient hay không.

Kết quả nhận được là một hình ảnh nhị phân với "các cạnh mỏng".

- **Lọc ngưỡng:**

Ở bước này ta cần 2 giá trị ngưỡng `max_val` và `min_val`. Ta sẽ xét các pixel dương trên mặt nạ nhị phân kết quả của bước trước. Nếu giá trị gradient vượt ngưỡng `max_val` thì pixel đó chắc chắn là cạnh. Các pixel có độ lớn gradient nhỏ hơn ngưỡng `min_val` sẽ bị loại bỏ. Còn các pixel nằm trong khoảng 2 ngưỡng trên sẽ được xem xét rằng nó có nằm liền kề với những pixel được cho là "chắc chắn là cạnh" hay không. Nếu liền kề thì ta giữ, còn không liền kề bất cứ pixel cạnh nào thì ta loại. Sau bước này ta có thể áp dụng thêm bước hậu xử lý loại bỏ nhiễu nếu muốn.

Trong OpenCV, để dùng giải thuật Canny, ta đơn giản chỉ cần 1 lệnh `cv2.Canny`



Hình 2.14. Kết quả lệnh `cv2.Canny` ^[16]

2.3 NGÔN NGỮ LẬP TRÌNH VÀ MỘT SỐ THƯ VIỆN SỬ DỤNG

2.3.1 Ngôn ngữ lập trình Python

Python là một ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng, do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python được thiết kế với ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Python là ngôn ngữ có hình thức rất sáng sửa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu. Vào tháng 7

năm 2018, Van Rossum đã từ chức Leader trong cộng đồng ngôn ngữ Python sau 30 năm lãnh đạo.

Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động; do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, và Tcl. Python được phát triển trong một dự án mã mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

Ban đầu, Python được phát triển để chạy trên nền Unix. Nhưng rồi theo thời gian, Python dần dần phổ biến và mở rộng sang mọi hệ điều hành từ MS-DOS đến Mac OS, OS/2, Windows, Linux và các hệ điều hành khác thuộc họ Unix. Mặc dù sự phát triển của Python có sự đóng góp của rất nhiều cá nhân, nhưng Guido van Rossum hiện nay vẫn là tác giả chủ yếu của Python. Ông giữ vai trò chủ chốt trong việc quyết định hướng phát triển của Python.^[7]

Với cú pháp cực kì đơn giản và thanh lịch, Python là lựa chọn hoàn hảo cho những ai lần đầu tiên học lập trình. Python là ngôn ngữ có mã lệnh (source code hay đơn giản là code) không mấy phức tạp cả trường hợp bạn chưa biết gì về Python bạn cũng có thể suy đoán được ý nghĩa của từng dòng lệnh trong source code. Tuy nhiên, đây cũng là ngôn ngữ nổi tiếng về sự chặt chẽ, nhanh, mạnh và có mặt ở mọi hệ điều hành.

Được sử dụng trong lập trình từ những game đơn giản nhất, cho đến những thuật toán tìm kiếm phức tạp nhất, Python không hề danh là ngôn ngữ danh cho cả newbie lẫn hacker.

2.3.2. Thư viện OpenCV

Opencv là gì?

OpenCV (Open Source Computer Vision Library) là một thư viện mã nguồn mở hàng đầu, nó là miễn phí cho những ai bắt đầu tiếp cận với các học thuật. OpenCV được ứng dụng trong nhiều lĩnh vực như cho thị giác máy tính (computer vision) hay xử lý ảnh (image processing) và máy học (machine learning) và một số tính năng tăng tốc GPU trong hoạt động thời gian thực. OpenCV được phát hành theo giấy phép của BSD, vì vậy nó hoàn toàn miễn phí cho cả học thuật và thương mại. Thư viện được lập trình trên các ngôn ngữ cấp cao: C++, C, Python, hay Java và hỗ trợ trên các nền tảng Window, Linux, Mac OS, iOS và Android. OpenCV đã được tạo ra tại Intel vào năm 1999 bởi Gary Bradsky, và ra mắt vào năm 2000. OpenCV được thiết kế để tính toán hiệu quả và với sự tập trung nhiều vào các ứng dụng thời gian thực. Được viết bằng tối ưu hóa C/C++, thư viện có thể sử dụng lợi thế của xử lý đa lõi. Được sử dụng rộng rãi, phạm vi sử dụng từ nghệ thuật tương tác, cho đến lĩnh vực khai thác mỏ, bản đồ trên web hoặc công nghệ robot, nghiên cứu của sinh viên. Ở đề tài này thư

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

viện OpenCV được chạy trên ngôn ngữ Python. OpenCV được dùng làm thư viện chính để xử lý hình ảnh đầu vào và sau đó đi nhận dạng ảnh. [8]

Chức năng có trong thư viện OpenCV

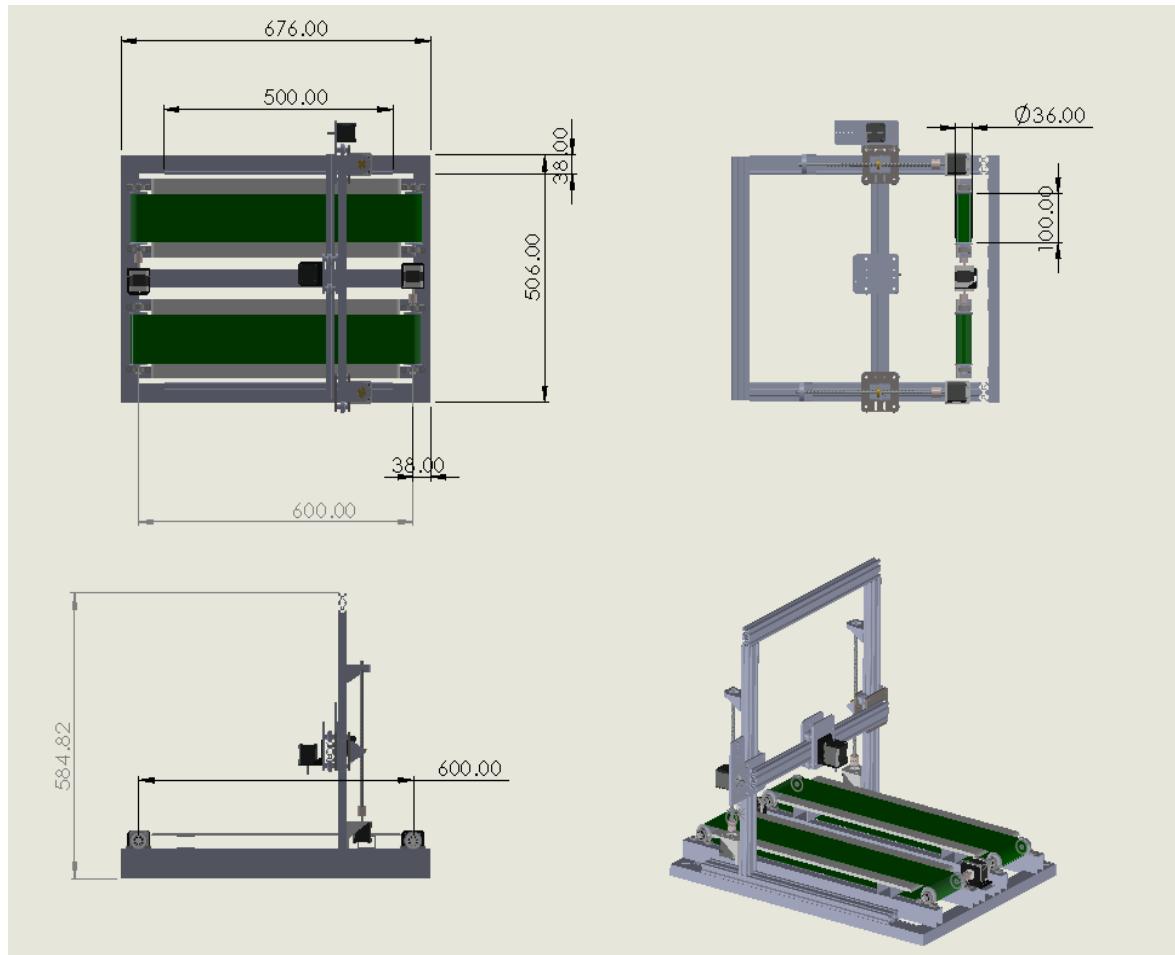
- Bộ công cụ hỗ trợ 2D và 3D
- Nhận diện khuôn mặt, cử chỉ
- Nhận dạng chuyển động, đối tượng, hành vi
- Tương tác giữa con người và máy tính
- Điều khiển robot

Các ứng dụng OpenCV

- Kiểm tra và giám sát tự động
- Robot và xe hơi tự lái
- Phân tích hình ảnh y tế
- Tìm kiếm và phục hồi hình ảnh/video

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

3.1. TÍNH TOÁN VÀ THIẾT KẾ PHẦN CƠ KHÍ



Hình 3.1. Bản vẽ thiết kế của hệ thống (mm)

3.1.1. Thiết kế khung, băng tải

3.1.1.1. Yêu cầu thiết kế

Với phần cơ khí gồm khung và các cơ cấu truyền động. Hệ thống băng tải được thiết kế tối ưu. Một số yêu cầu cho phần cơ khí của hệ thống bao gồm:

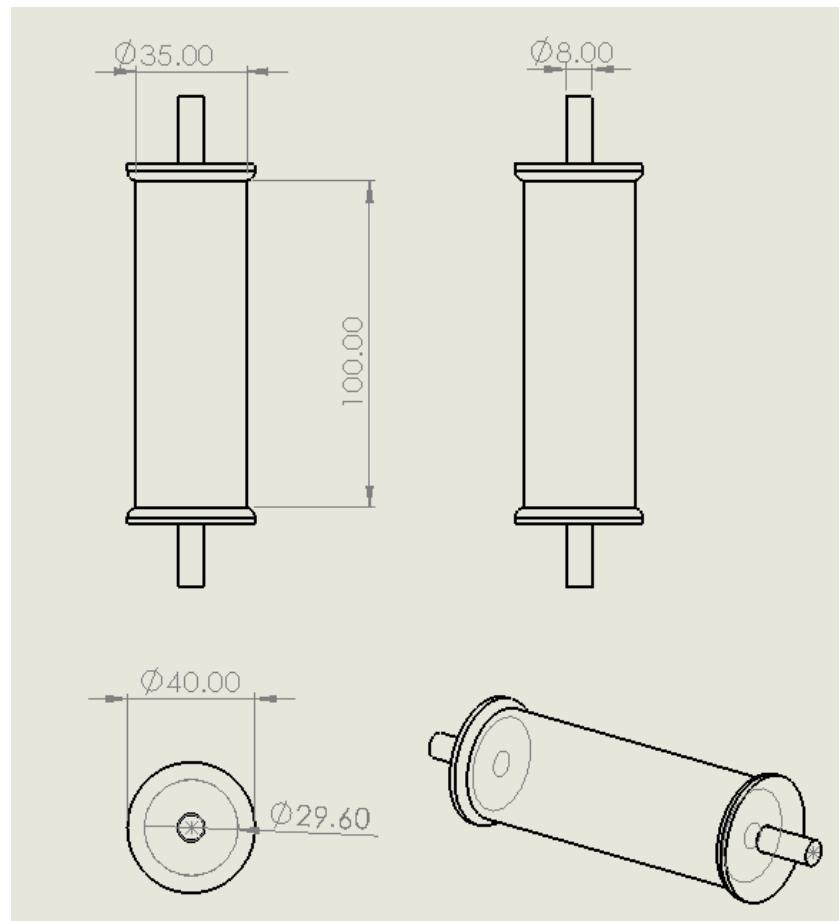
- Chắc chắn, đảm bảo không rung lắc khi hoạt động.
- Tiết diện của băng tải và khung phải tương thích, phù hợp.

3.1.1.2. Thiết kế băng tải

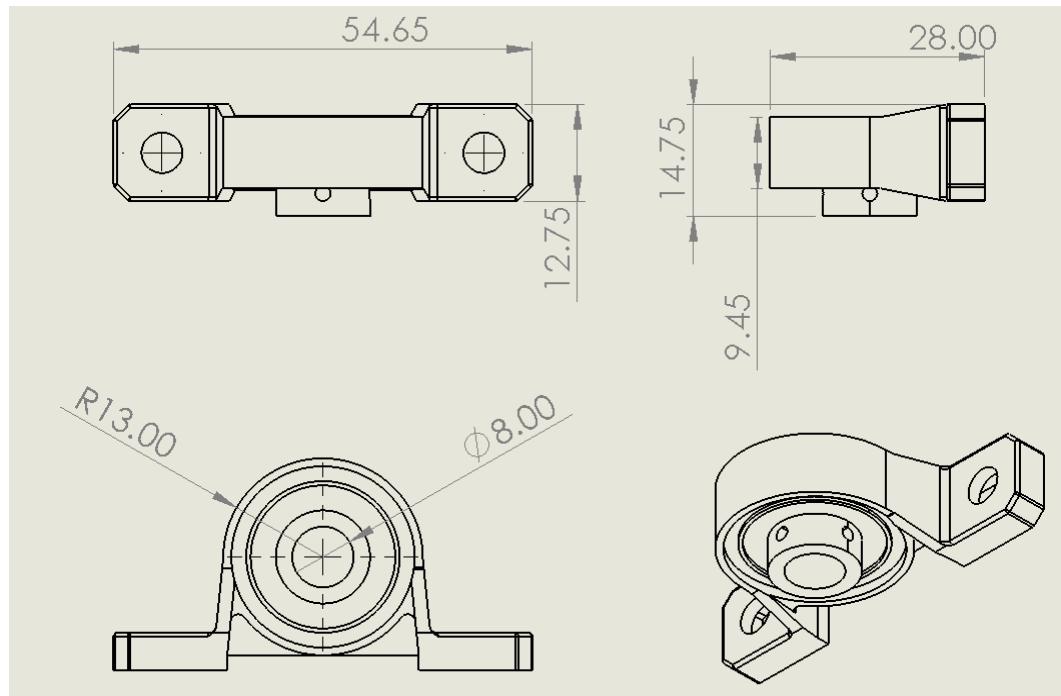
❖ Cấu tạo cơ khí của băng tải:

- Khung giá đỡ băng tải
- Gói đỡ, con lăn truyền động

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ



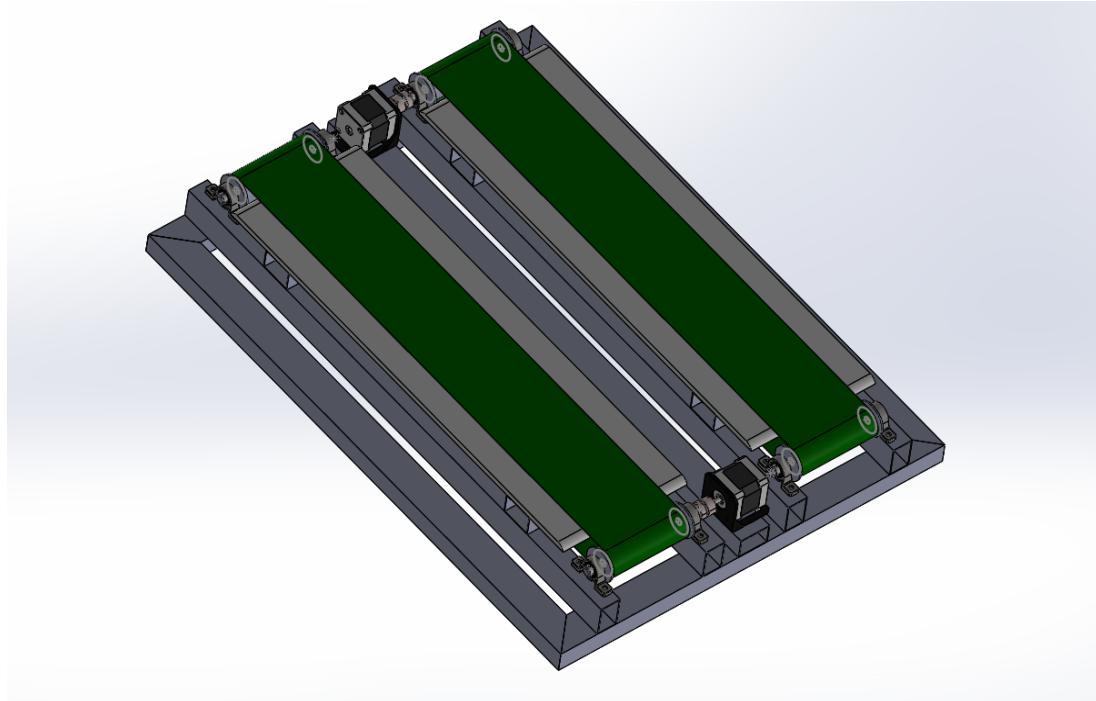
Hình 3.2. Bản vẽ cơ khí con lăn (mm)



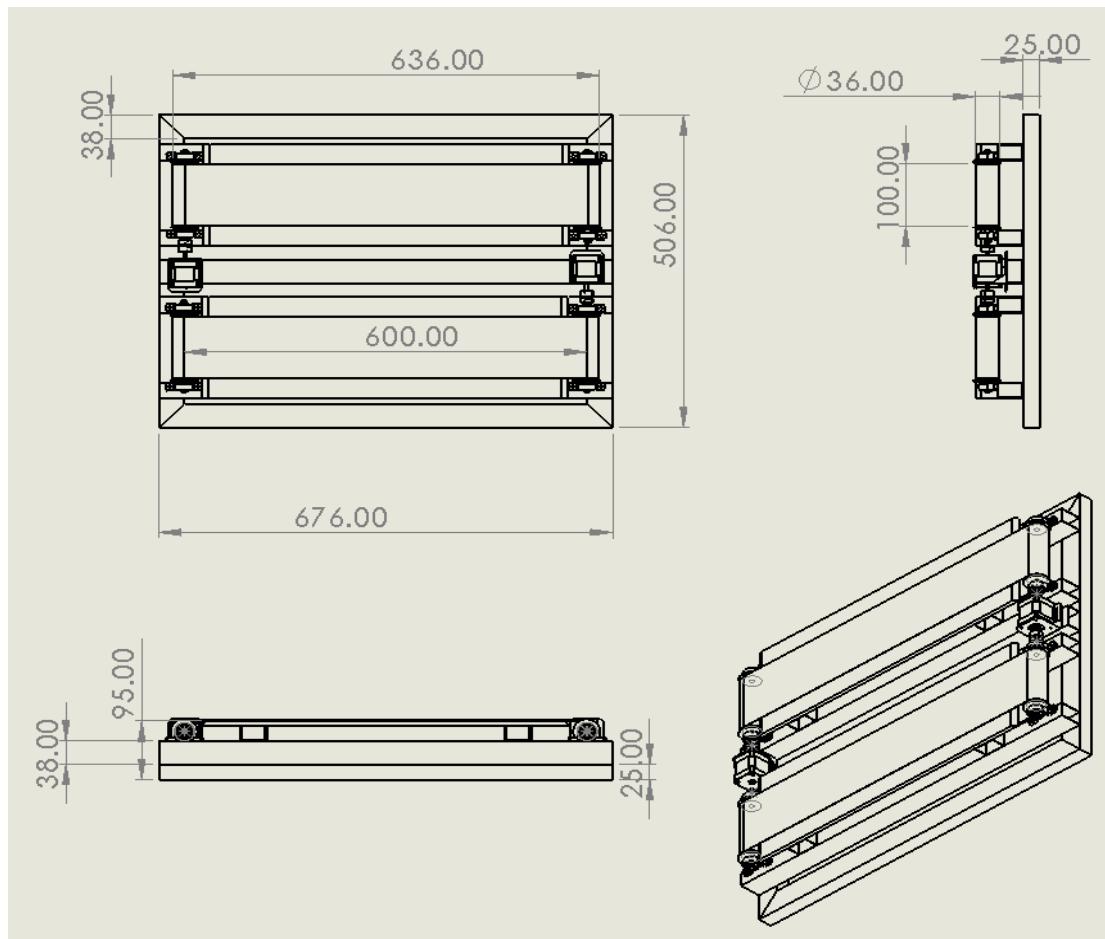
Hình 3.3. Bản vẽ gối đỡ con lăn (mm)

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

- ❖ Hình ảnh thiết kế 3D của băng tải:



Hình 3.4. Thiết kế 3D của băng tải



Hình 3.5. Bảng vẽ thiết kế 2D của băng tải (mm)

3.1.2. Thiết kế cơ cấu gấp sản phẩm

3.1.2.1. Yêu cầu thiết kế

Với tính xác về mặt xử lý, phần cơ cấu gấp sản phẩm cần phải đạt được những yêu cầu tất yếu sau:

- Cơ cấu chắc chắn, vững vàng
- Di chuyển mượt mà
- Đảm bảo độ chính xác cao

Để nhận biết được sản phẩm đi vào, nhóm quyết định chọn camera để xử lý thay vì chọn cảm biến truyền thống vì những lý do sau:

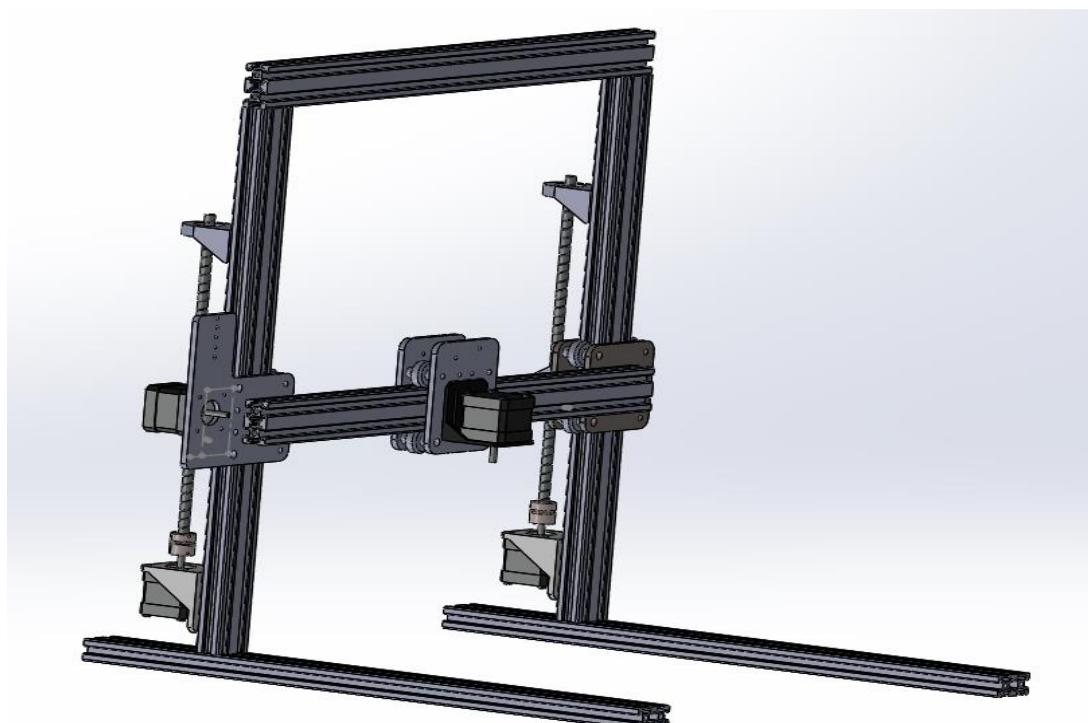
- Chỉ cần dùng 1 camera có thể thay thế nhiều cảm biến tiêm cận phát hiện vật.
- Công nghệ xử lý hình ảnh từ camera có thể phát hiện góc lệch của sản phẩm đi vào để xoay sản phẩm đúng hướng.

3.1.2.2. Thiết kế cơ cấu gấp

❖ Cấu tạo cơ cấu:

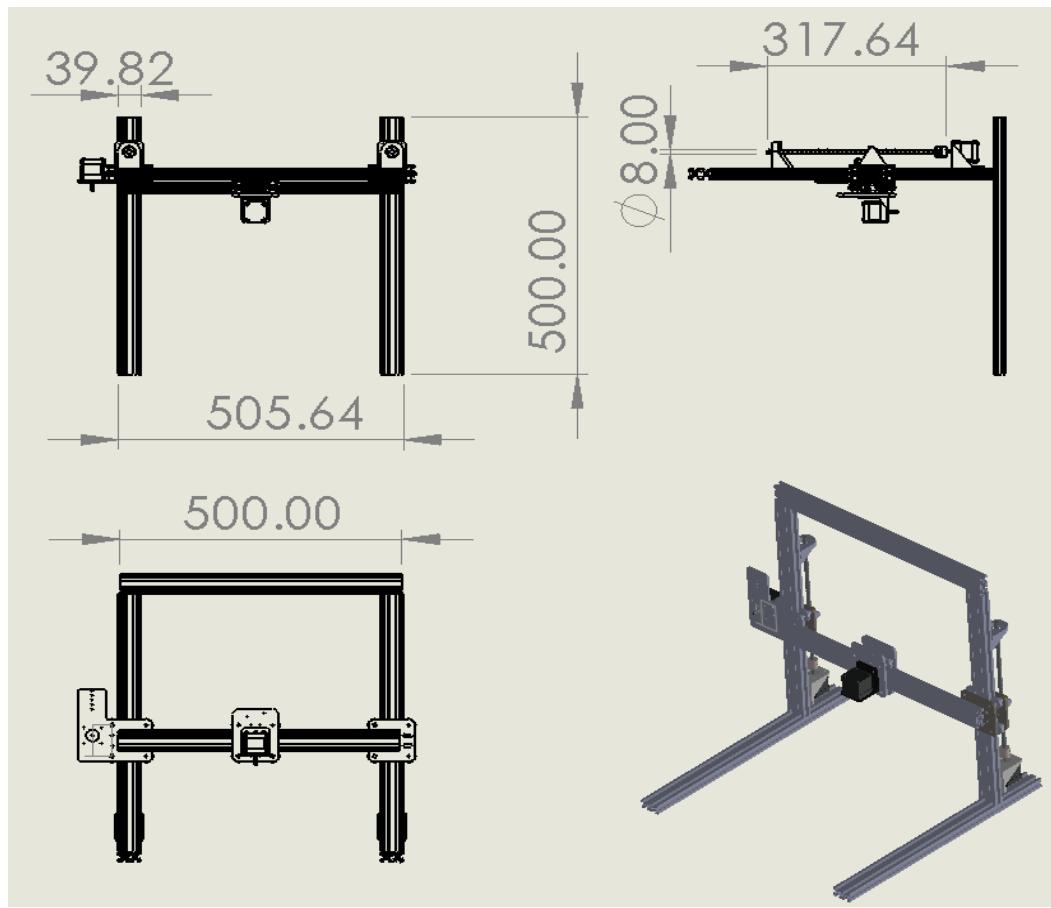
- Trục Z: truyền động qua vít-me
- Trục X: truyền động qua ròng rọc, puly và dây đai

❖ Hình ảnh thiết kế 3D:

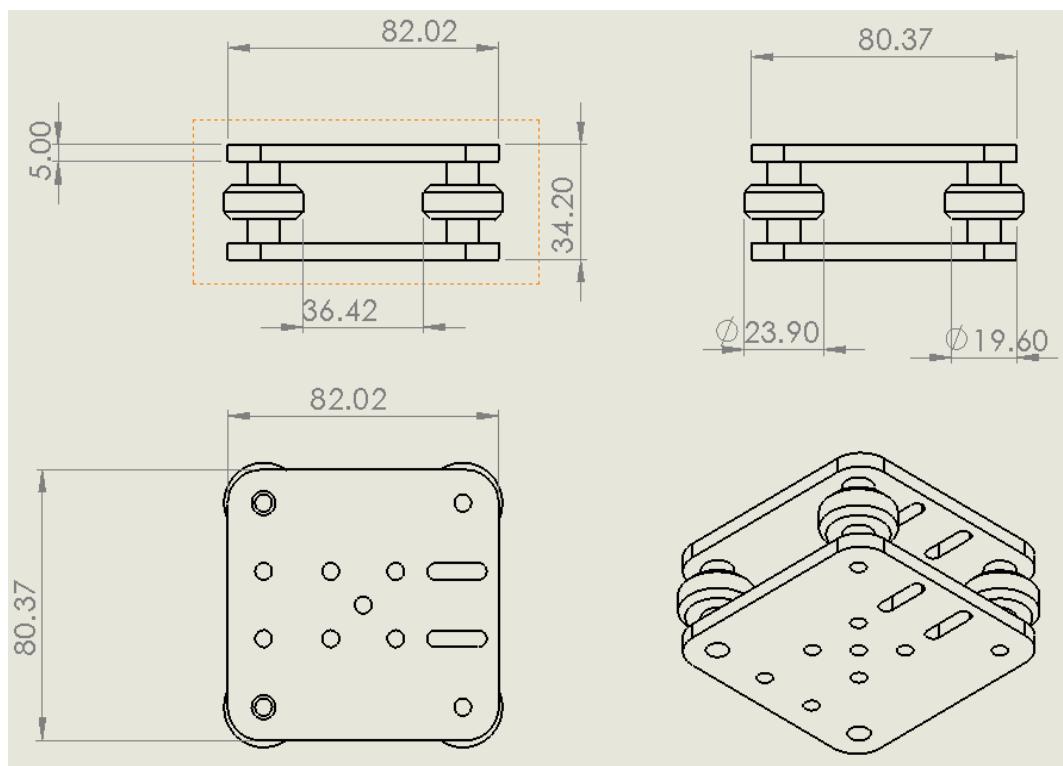


Hình 3.6. Cơ cấu gấp sản phẩm

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

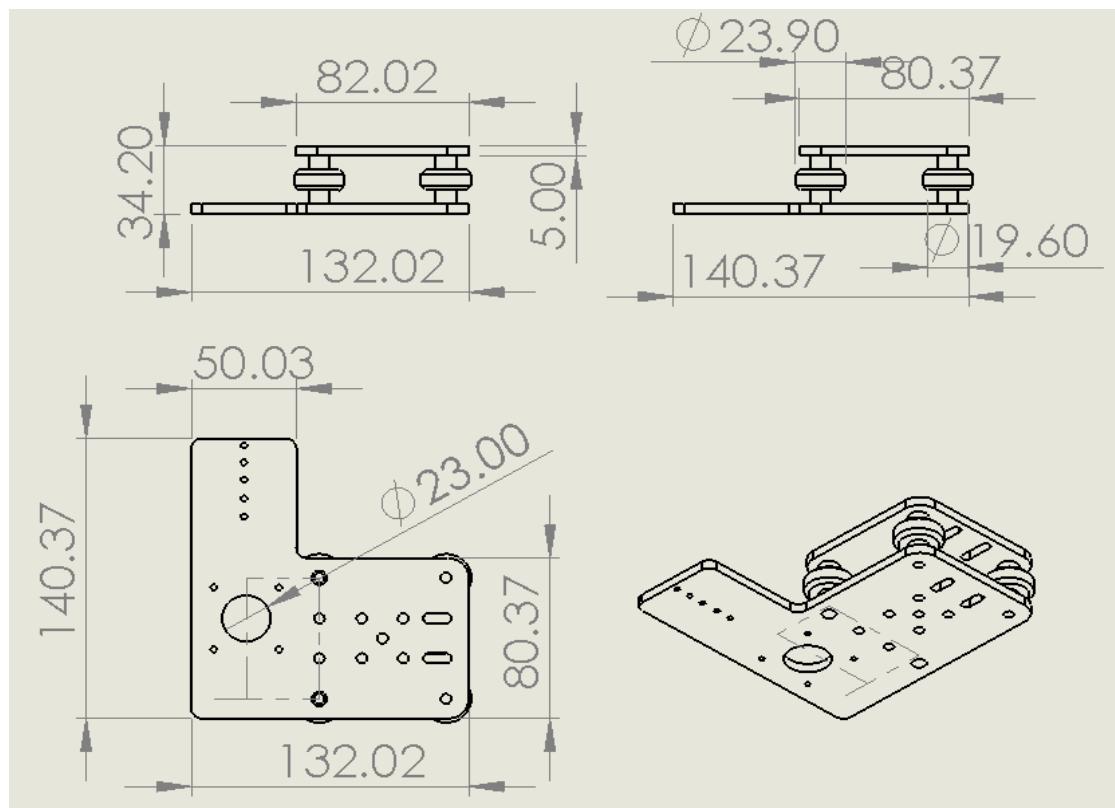


Hình 3.7. Bản vẽ 2D cơ cấu gấp (mm)

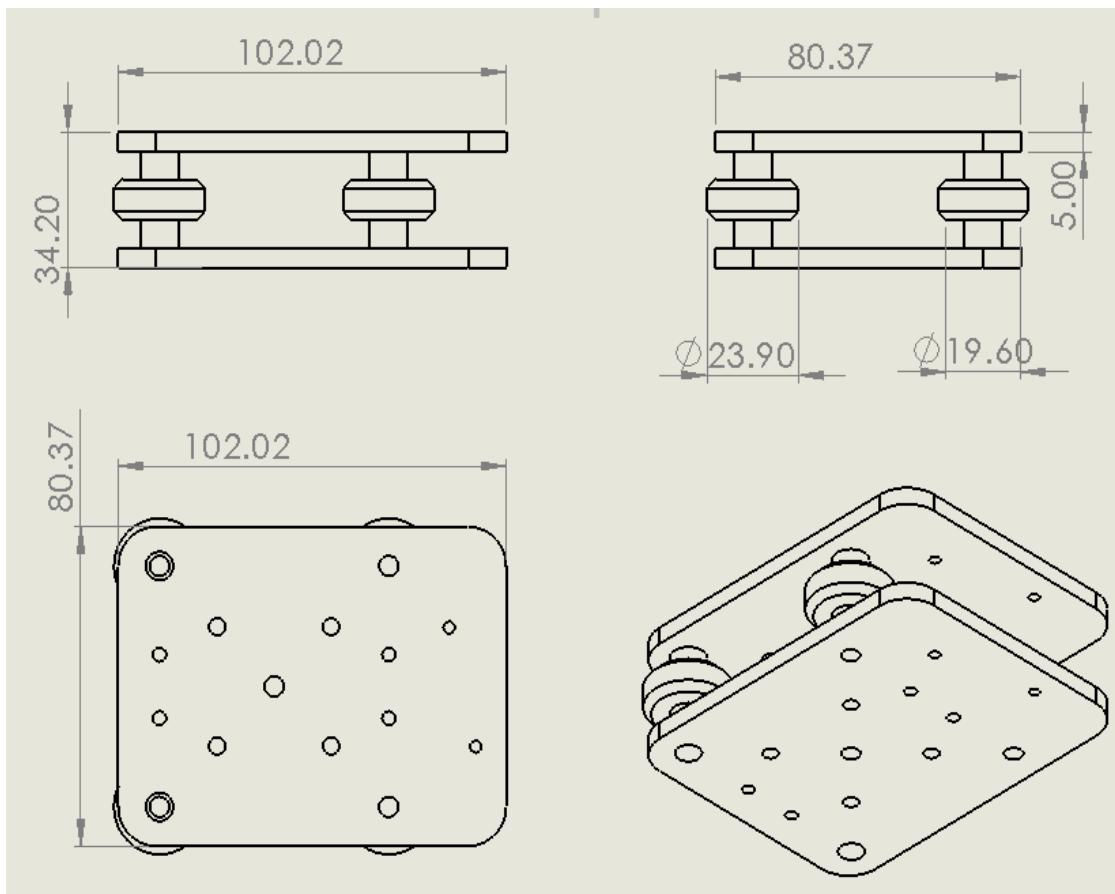


Hình 3.8. Bản vẽ thiết kế bộ con lăn nhôm định hình 40x20 1(mm)

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ



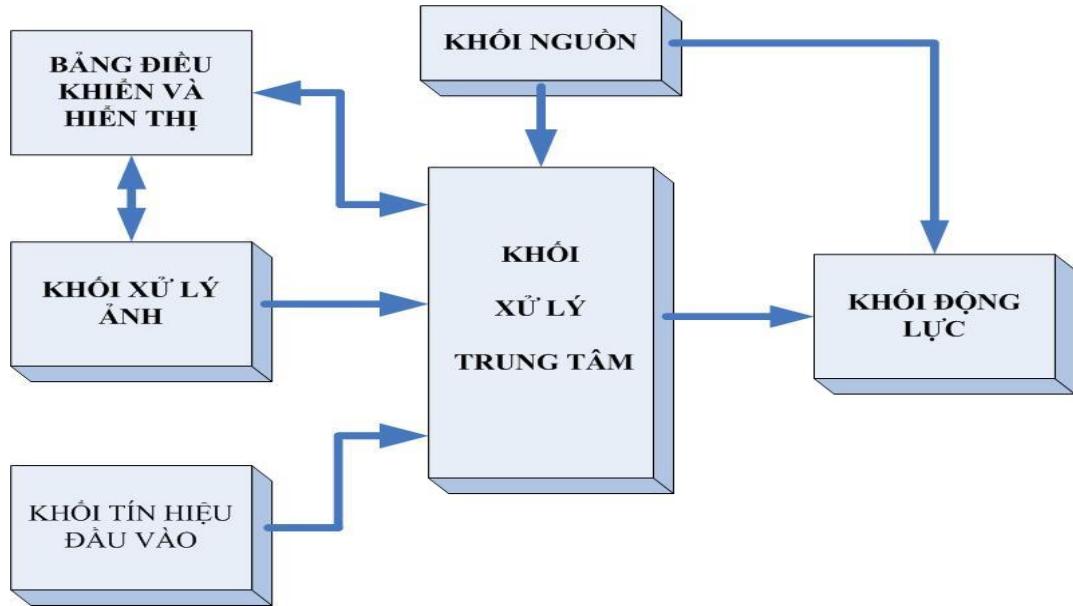
Hình 3.9. Bản vẽ thiết kế bộ con lăn nhôm định hình 40x20 2 (mm)



Hình 3.10. Bản vẽ thiết kế bộ con lăn nhôm định hình 40x20 3 (mm)

3.1.3. Thiết kế sơ đồ khói của hệ thống

Hệ thống có sơ đồ khói như sau:



Hình 3.11. Sơ đồ khói của hệ thống

❖ Chức năng của từng khói:

-**Bảng điều khiển và hiển thị:**

- Cho phép người sử dụng tác động vào hệ thống để vận hành như mở, tắt hệ thống.
- Giám sát tình trạng hoạt động của hệ thống
- Quản lý số lượng sản phẩm
- Nhận các tín hiệu cài đặt từ người vận hành.
- Báo động khi có lỗi và gửi tín hiệu dừng hệ thống về khói điều khiển

-**Khối xử lý ảnh:** Xử lý thông tin sản phẩm và gửi tín hiệu về vi điều khiển

- Khối tín hiệu đầu vào** (công tắc hành trình): gửi tín hiệu vào khói điều khiển để giới hạn phạm vi hoạt động của trục X và trục Z

-**Khối xử lý trung tâm:**

- Chứa chương trình điều khiển.
- Xử lý các tín hiệu ngõ vào và xuất tín hiệu điều khiển đến các khói ngõ ra của hệ thống

-**Khối nguồn:** cung cấp nguồn cho vi điều khiển và mạch động lực

-**Khối động lực:**

- Băng tải: vận chuyển sản phẩm
- Cơ cấu gấp: gấp sản phẩm từ băng tải 1 bỏ vào thùng trên băng tải 2

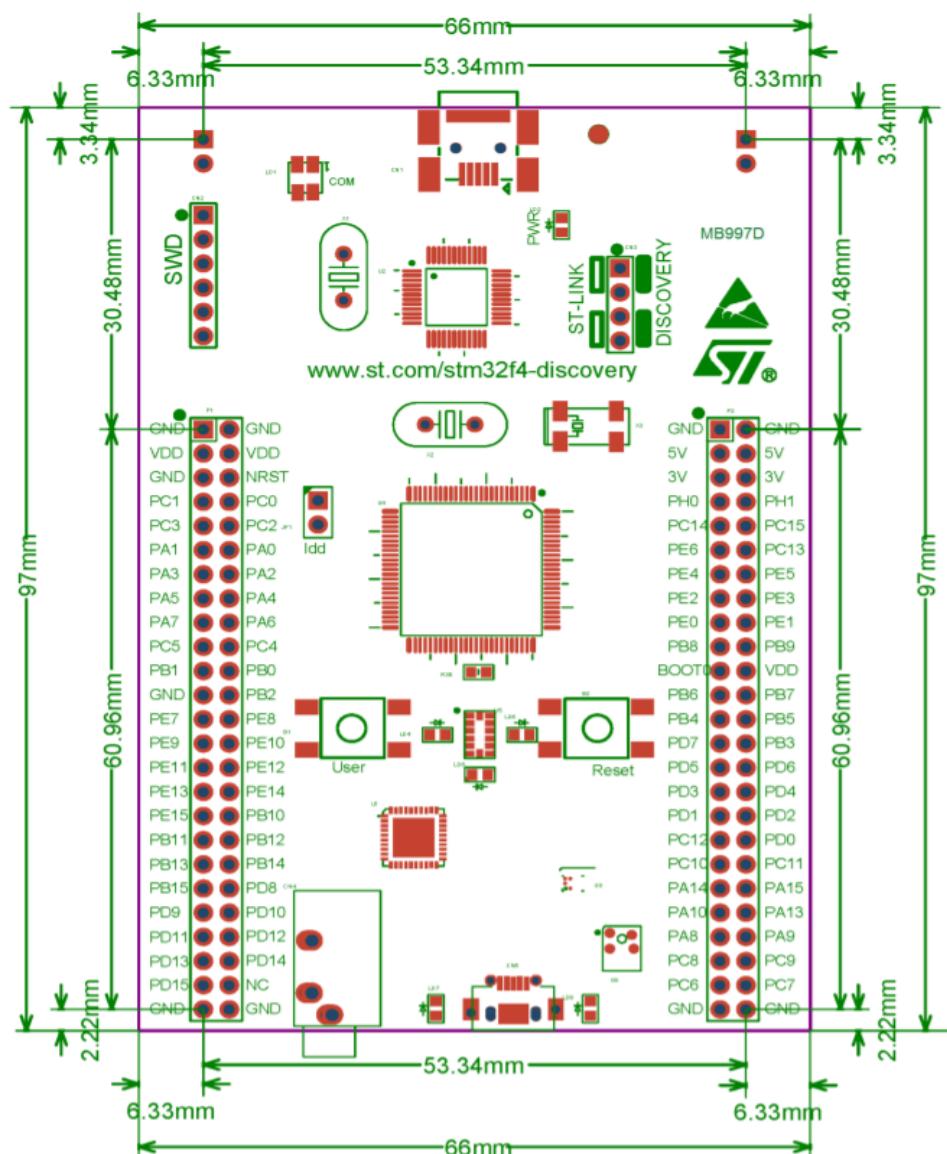
3.1.4. Chọn thiết bị cho các khối.

3.1.4.1. Khối xử lý trung tâm

❖ STM32F407G-DISC1

STM32F407G-Disc1 được chọn làm đơn vị xử lý trung tâm cho hệ thống. Thay vì chọn bộ điều khiển PLC với giá thành khá cao thì vi điều khiển là một lựa chọn tối ưu nhất với giá tiền rẻ hơn rất nhiều. Và so về cấu hình trong các dòng vi điều khiển thì STM32 với con chip vi điều khiển 32 bit lõi Arm Cortex mạnh mẽ là một đối thủ vượt trội hơn rất nhiều với cùng một mức giá.

Và với yêu cầu điều khiển của đề tài, việc sử dụng Arduino hay các dòng vi điều khiển khác thì không đủ tài nguyên để sử dụng.



Hình 3.12. Bản vẽ cơ khí của board STM32F407G-DISC1 [2]

3.1.4.2. Khối xử lý ảnh

❖ Camera: Webcam Logitech C270

Webcam Logitech C270 giúp quay video với độ phân giải cao và sắc nét, phân giải video lên đến 1280 x 720 pixel. Ở tốc độ 30 khung hình/giây, chất lượng video trở nên mượt mà, trong khi hình ảnh rõ ràng, đầy màu sắc và có độ tương phản. Cảm biến hình ảnh với công nghệ RightLight cung cấp hình ảnh đạt chất lượng ngay cả khi quay trong bối cảnh mờ tối, C270 sẽ điều chỉnh với điều kiện ánh sáng để tạo ra hình ảnh có độ tương phản, tươi sáng hơn. Micrô tích hợp với công nghệ RightSound phục vụ mọi nhu cầu về chat voice. Webcam C270 nhỏ gọn, linh hoạt và có thể điều chỉnh góc quay dễ dàng giao tiếp với máy tính qua chuẩn giao tiếp USB với thiết lập đơn giản [4].



Hình 3.13. Webcam C720 [4]

❖ Thông số kỹ thuật:[4]

- Độ phân giải tối đa: 720p/30fps
- Loại tiêu cự: Lấy nét cố định
- Công nghệ thấu kính: tiêu chuẩn
- Micrô tích hợp: đơn âm
- Phạm vi quan sát: 60°
- Kẹp phổ dụng phù hợp với máy tính xách tay, LCD hoặc màn hình
- Độ dài dây cáp: 1.5m

❖ Phần mềm: Python 3.7 và thư viện OpenCV

3.1.4.3. Khối tín hiệu đầu vào (công tắc hành trình)

Module công tắc hành trình (Endstop CNC, Printer 3D)

Mạch công tắc hành trình được thiết kế gọn nhẹ bao gồm các thành phần như LED, điện trở và 1 công tắc hành trình. Mạch được thiết kế để dễ dàng kết nối với vi điều khiển. Công tắc hành trình máy in 3D được sử dụng trong các thiết kế máy in 3D để xác định điểm của hành trình trực, công tắc được thiết kế dễ sử dụng với dây cắm đi kèm, đèn báo kích hoạt. Ngoài việc sử dụng cho máy in 3D công tắc có thể được sử dụng cho nhiều mục đích khác nhau. Nguồn cung cấp từ 3-12VDC.^[6]

❖ **Ưu điểm:** Dễ cài đặt, phỏ biến hơn.

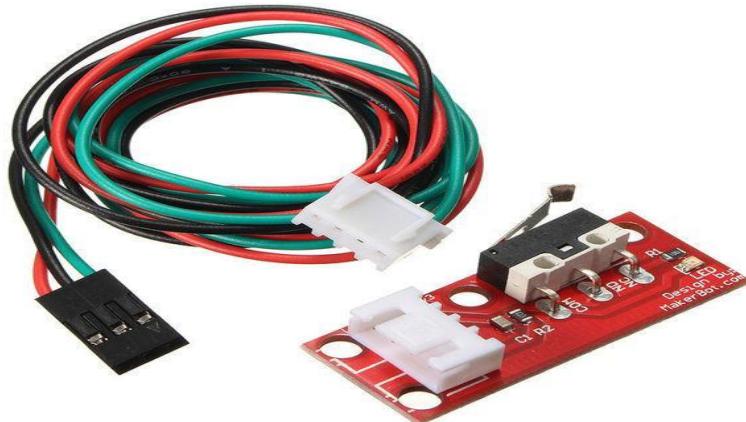
Nhược điểm: dễ hao mòn các bộ phận cơ khí, tuổi thọ ngắn hơn.

❖ **Đặc điểm nổi bật:**

- Là phương án Endstop rẻ nhất
- Đơn giản, không cần thêm mạch xử lý tín hiệu
- Độ tin cậy cao (có thể lên tới 1 triệu lần đóng/cắt)

❖ Đầu dây tín hiệu: ^[6]

- Dây đỏ: VCC
- Dây đen: GND
- Dây xanh: SIGNAL



Hình 3.14. Công tắc hành trình Endstop ^[6]

3.1.4.3. Khối động lực

❖ **Băng tải:** băng tải PVP

Băng tải sử dụng để vận chuyển sản phẩm bằng động cơ được điều khiển bằng STM32F407G thông qua module driver DRV8825.

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

Hệ thống có 2 băng tải, băng tải một vận chuyển sản phẩm, băng tải 2 vận chuyển các thùng chứa sản phẩm.

❖ **Động cơ:** Động Cơ Bước Nema17

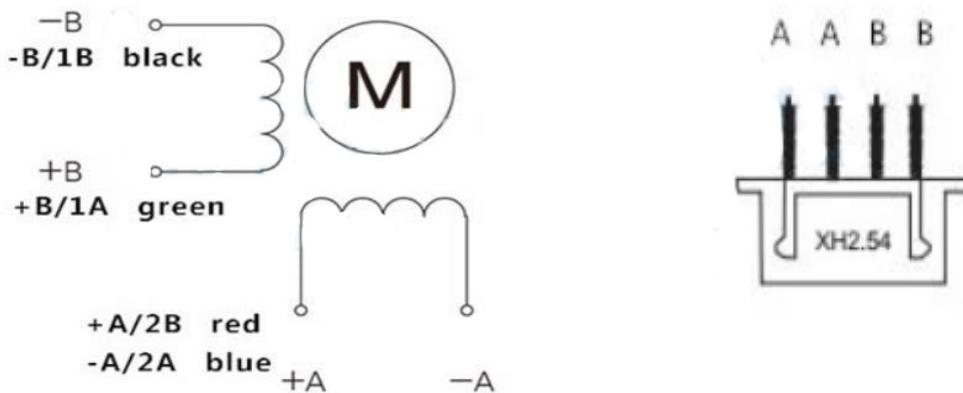


Hình 3.15. Động cơ bước Nema17 [6]

Thông số kỹ thuật [6]	
Góc bước	1.8 độ
Số pha	2 phases
Mô-men xoắn	0,45nm/64ozin
Điện trở	2.1Ω/pha
Dòng điện định mức	1.7 A
Độ tự cảm	2.5mH/pha
Điện áp	12 – 24VDC
Công suất	6.6W
Khối lượng	297g
Số dây	4 dây
Đường kính trục	5 mm

Bảng 3.1. Thông số kỹ thuật của động cơ

Sơ đồ hệ thống dây điện:



Hình 3.16. Sơ đồ dây step motor [6]

❖ **Module Driver DRV8825**

Driver động cơ bước DRV8825 với đầy đủ các tính năng của một driver chuyên nghiệp: điều chỉnh dòng giới hạn, vi bước (1/32 bước), bảo vệ quá dòng, quá nhiệt, v.v...

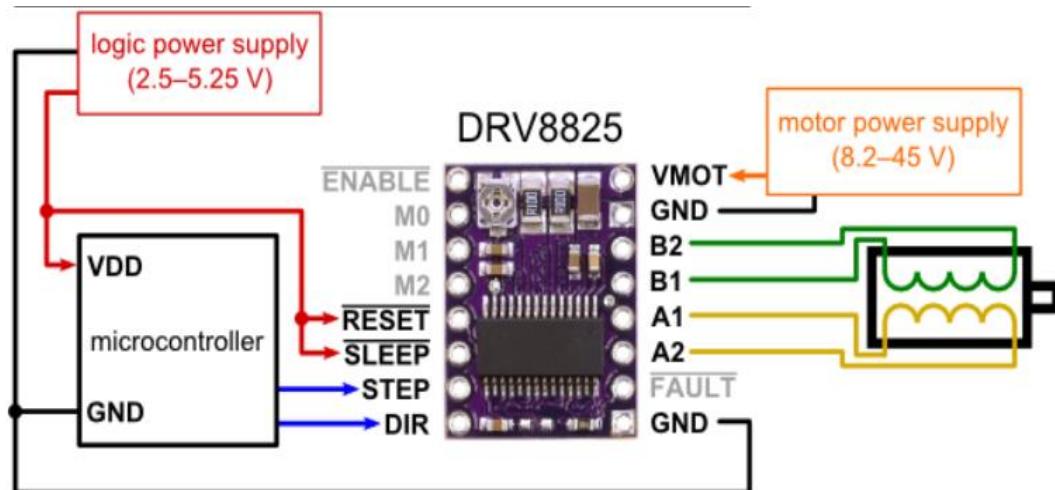
Hoạt động ở dải điện áp cao từ 8.2V đến 45V và có thể đạt được xấp xỉ 1,8A trên mỗi pha mà không cần tản nhiệt. Driver có các chân ra và bề mặt gần như

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

đồng nhất với module A4988 vì vậy nó có thể dùng thay thế cho board đó trong nhiều ứng dụng khác nhau. [7]

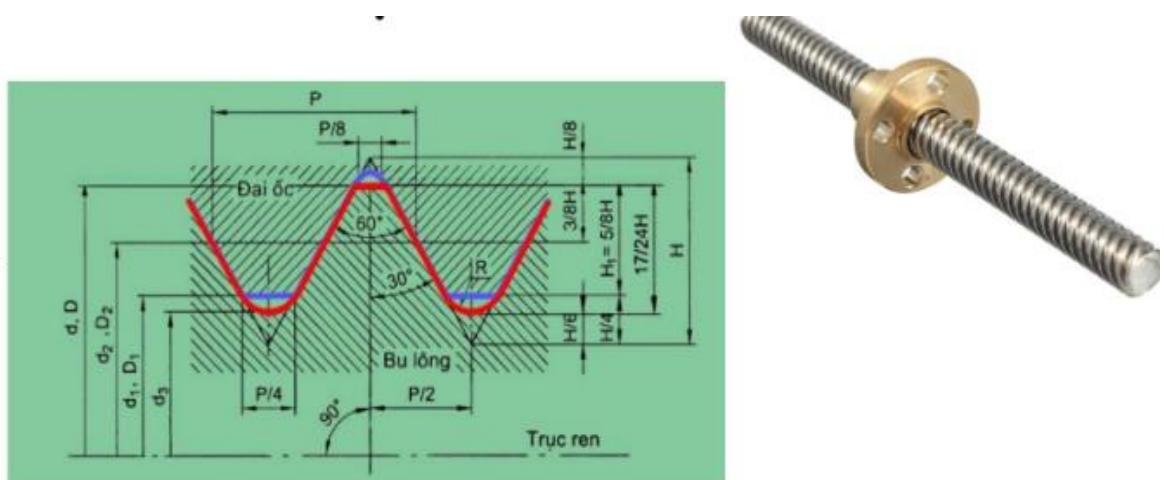
Thông số kỹ thuật:

- Điện áp cung cấp: 8.2~45VDC
- Dòng trung bình (RMS): 1.5A, dòng đỉnh (Peak) lên đến 2.5A.
- 6 độ phân giải bước khác nhau: full, half step, 1/4 step, 1/8 step, 1/16 step, 1/32 step.
- Điện áp điều khiển: 3.3V và 5V.
- Tự động shutdown khi quá nhiệt, quá dòng
- Bảo vệ ngắn mạch và bảo vệ quá tải.
- Mạch 4 lớp, 2 lớp phủ đồng giúp cải thiện khả năng tản nhiệt. [6]



Hình 3.17. Sơ đồ kết nối chân Driver DRV8825 [6]

❖ Vit-me:



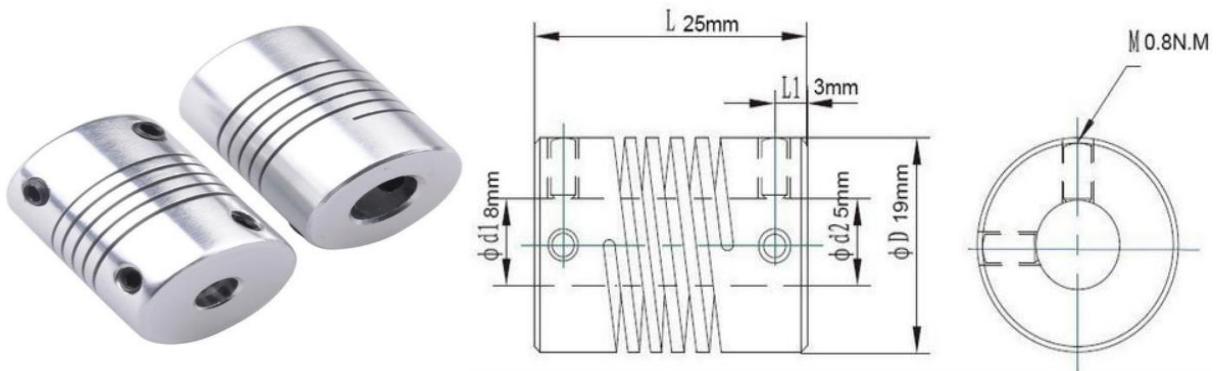
Hình 3.18. Bộ truyền Vit-me [6]

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

Thông số bộ truyền Vit-me: [6]

- Loại ren: ren hình thnag
- Bước ren trực vít: $p = 2\text{mm}$
- Số vòng ren bánh vít: 4 vòng, bước xoắn 8mm
- Chiều dài trực vít: $L = 30\text{cm}$
- Đường kính trực vít: 8mm
- Đường kính đai ốc: 8mm

❖ **Khớp nối động cơ:** Khớp nối vit-me 5-8mm.



Hình 3.19. Khớp nối vit-me 5-8mm [6]

Thông số kỹ thuật:

Đường kính ngoài D(mm)	Đường kính trực d1(mm)	Đường kính trực d1(mm)	Chiều dài (mm)	Vật liệu
19	8	5	25	Hợp kim nhôm

Bảng 3.2. Thông số kỹ thuật của khớp nối 5-8mm

❖ **Dây đai:** Dây đai GT2 [6]



- Bước răng: 2mm
- Chiều rộng: 6mm

Hình 3.20. Dây đai GT2 [6]

❖ **Puly:** Puly GT2 [6]



- Bước răng: 2mm
- Số răng: 20
- Trục: 5mm

Hình 3.21. Puly GT2 [6]

❖ **Ròng rọc:** Ròng rọc 2GT20 [6]



- Số răng: 20
- Bước răng: 2mm
- Chiều dài rãnh: 6.5mm
- Trục: 5mm

Hình 3.22. Ròng rọc 2GT20 [6]

3.1.4.6. Con lăn

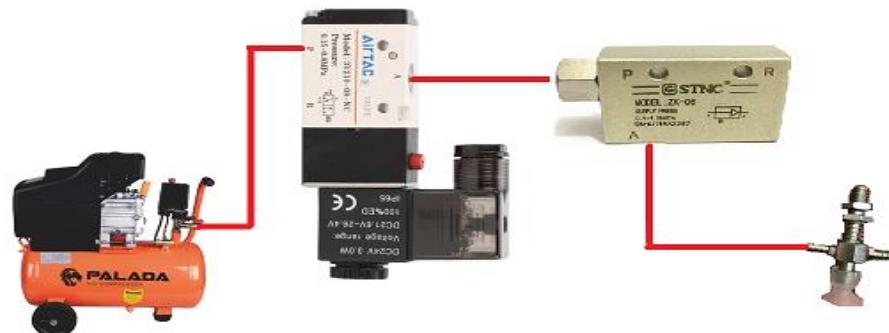
Chọn con lăn V-slot dùng cho nhôm định hình 20x20 hoặc 20x40



Hình 3.23. Con lăn V-slot [6]

3.1.4.7. Van khí nén, van chân không, đầu tool

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ



Hình 3.24. Sơ đồ kết nối ống hơi.

❖ **Van khí nén:** Van điện từ khí nén AIRTAC 4V210-08

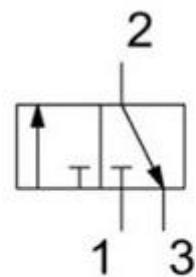


Hình 3.25. Van khí nén 3/2 [9]

Van điện từ khí nén AIRTAC 4V210-08 là loại van khí nén 3/2 có 3 cổng 2 vị trí và 1 đầu coil điện, được điều khiển bằng điện, thường được dùng để điều khiển xi lanh khí nén. [9]

Thông số kỹ thuật: [9]

- Kích thước công: 1/4".(ren 13).
- Kích thước công xả: 1/4" (ren 13).
- Áp suất hoạt động: 0.15 – 0.8 MPa.
- Loại van hơi 3 cửa 2 vị trí. (1 Đầu Coil Điện)
- Nhiệt độ hoạt động: -20~70°C.
- Điện áp: 24VDC



- Cổng (1) là cổng đưa áp suất vào
- Cổng (2) là cổng áp suất đi ra
- Cổng (3) là cổng xả

Van thường đóng (NC)

Van khí nén 3/2 thường đóng tiếng anh Normal Close (NC)

Trang thái chưa được cấp điện vào đầu coil thì (P) và (A) không thông nhau, (A) và (R) thông nhau.

Khi đầu coil được cấp điện thì van đảo chiều (P) và (A) lúc này sẽ thông với nhau, (A) và (R) lúc này sẽ không thông nhau nữa.

❖ Van chân không: CONVUM CV 10-HS

Van thực hiện việc hút hơi chân không, điều tiết và cùi nối, vận chuyển hơi.

- Lưu chất: Khí nén
- Nhiệt độ từ: 0°C ~ 60 °C
- Áp suất: 0.1 – 0.6 Mpa

Van hút chân không ZK là thiết bị chuyên dùng trong các hệ thống, máy móc khí nén phục vụ cho các nhà máy: đóng gói và chế biến thực phẩm và nông, lâm sản, dệt may công nghiệp, lắp ráp linh kiện điện tử và các chi tiết máy, sản xuất robot, luyện kim, cơ khí chế tạo máy ...



Hình 3.26. Van chân không CV-10HS [10]

❖ **Đầu tool hút:**

Đầu tool của robot được thiết kế theo yêu cầu sử dụng. Băng tải vận chuyển những vật phẩm mỏng, nhẹ, nên đầu tool được thiết kế theo dạng đầu hút, dùng khí nén.



Hình 3.27. Đầu tool hút

3.2. THIẾT KẾ MẠCH ĐIỀU KHIỂN CỦA HỆ THỐNG

Mạch điều khiển được nhóm thiết kế có thể đáp ứng những yêu cầu sau:

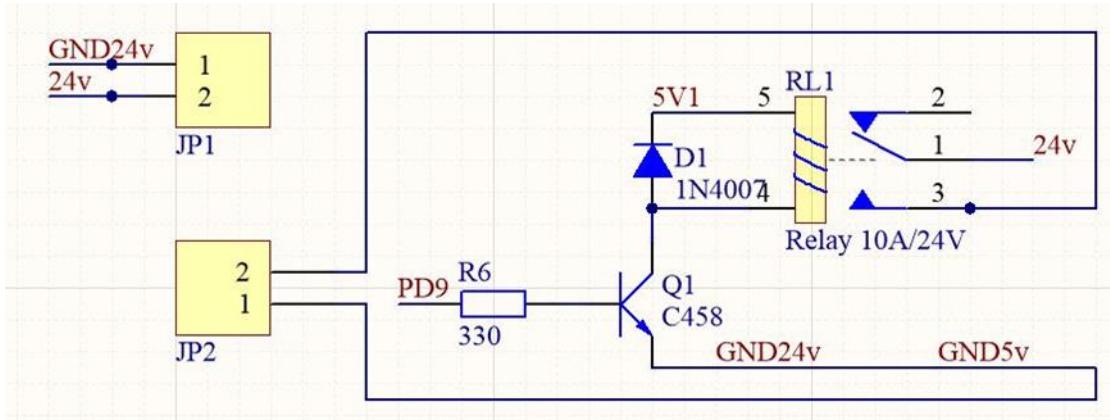
❖ **Về mặt phần cứng:**

- Có thể nối nguồn vào mạch động lực dễ dàng.
- Có thể tháo lắp module điều khiển.
- Có thể tháo lắp module driver điều khiển động cơ.
- Có thể nối dây stepper motor dễ dàng.
- Có thể kết nối điều khiển các thiết bị ngoại vi như van khí nén, đèn báo.
- Có thể kết nối nút nhấn, công tắc hành trình dễ dàng.

Để có thể đáp ứng những yêu cầu trên, nhóm quyết định:

- Chọn hàng rào cái để kết nối module điều khiển và driver stepper motor với mạch điện.
- Chọn hàng rào đặc để cắm dây động cơ, nút nhấn, công tắc hành trình.
- Chọn domino 2 chân thẳng để nối nguồn mạch động lực, van khí nén, đèn báo.

- ❖ **Về mặt kỹ thuật điều khiển:** Vì điều khiển sử dụng điện áp 3V DC, để có thể điều khiển được các thiết bị ngoại vi sử dụng điện áp 24V cần sử dụng relay 24V trung gian để kích mạch động lực. Nhóm đã thiết kế mạch điều khiển relay như sau:

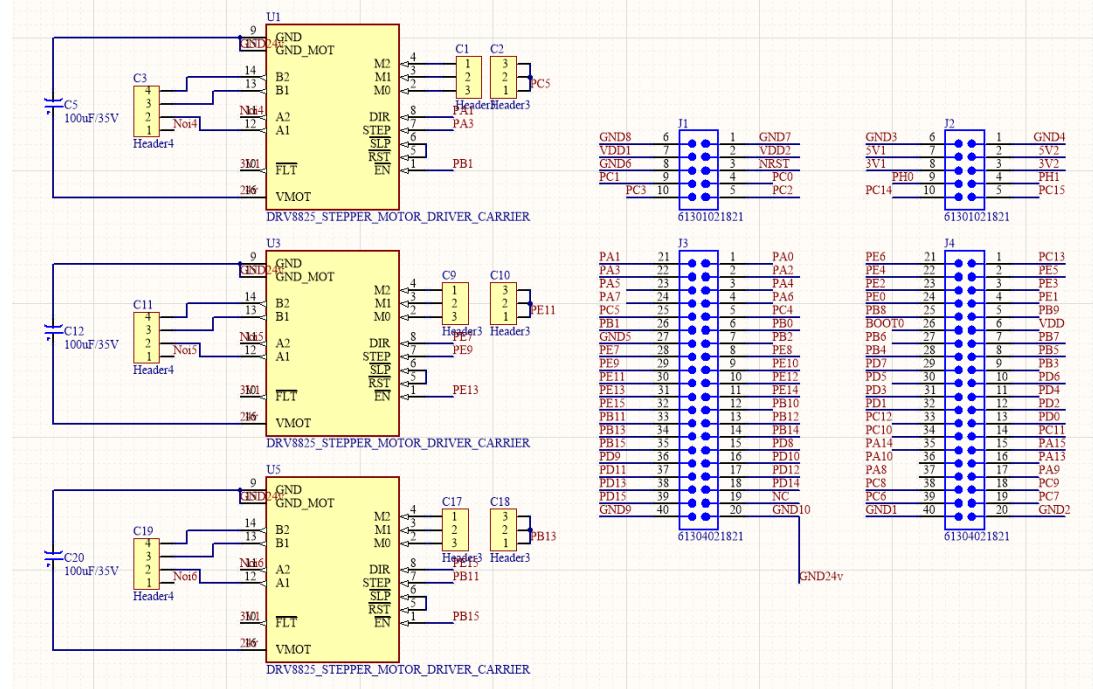


Hình 3.28. Mạch nguyên lý điều khiển relay

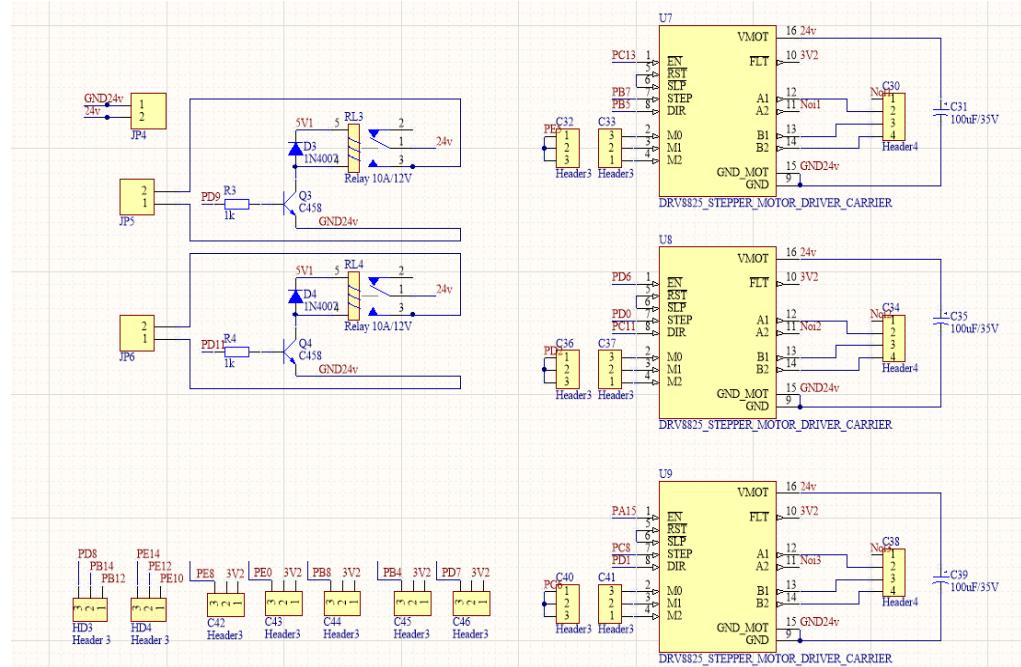
- Mạch nhận tín hiệu từ ngõ ra STM32 và kích hoạt transistor Q1 hoạt động ở chế độ bão hòa, từ đó thông mạch cấp nguồn 5V cho cuộn coid relay. Điện trở R6 được chọn để Q1 hoạt động ở chế độ bão hòa là $330\ \Omega$.
- Khi cuộn coil được dẫn, relay thông mạch tiếp điểm 1, 3 từ đó đưa 24V ra chân 2 của domino JP2 để cấp nguồn cho mạch động lực.
- Cuộn coil relay mắc song song với diod ngược để dẫn dòng quấn tính được tạo ra từ từ trường của cuộn dây trước đó, bảo vệ transistor không bị đánh thủng mối nối CE.

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

Tổng hợp những yêu cầu trên nhóm đã thiết kế mạch nguyên lý như sau:



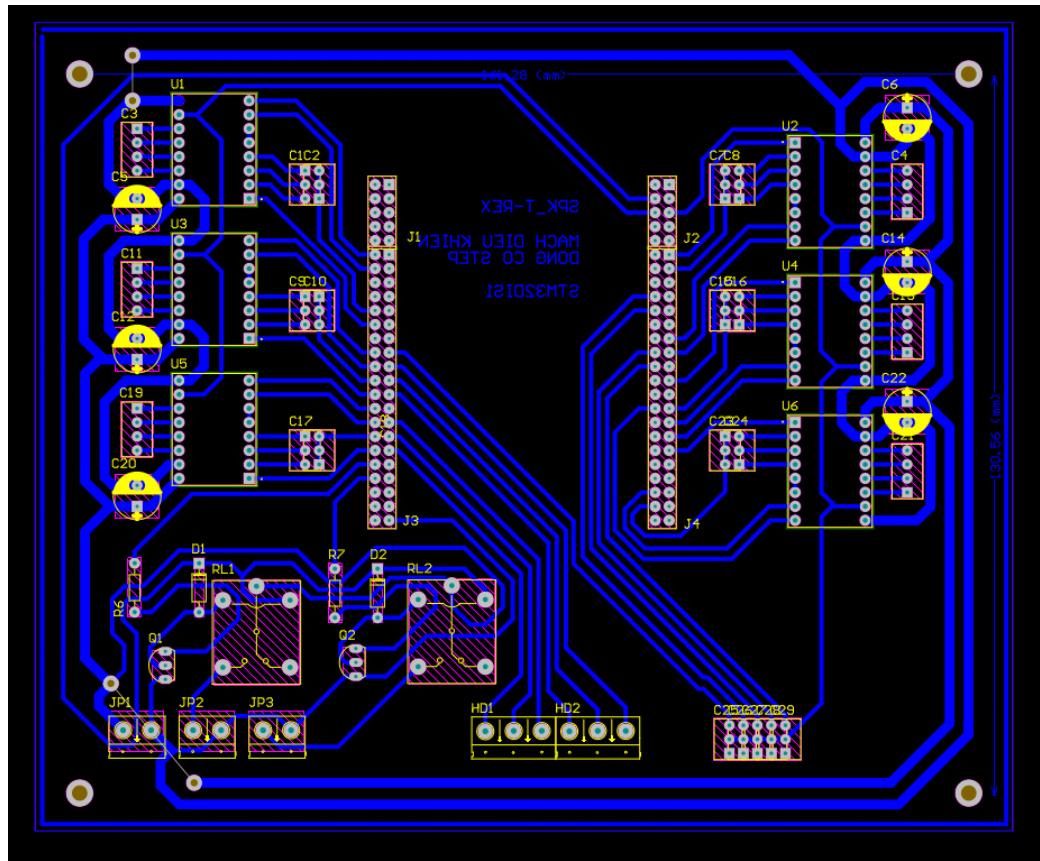
Hình 3.29. Mạch nguyên lý 1



Hình 3.30. Mạch nguyên lý 2

- Mạch gồm vi điều khiển STM32F4 điều khiển 6 driver DRV8825, 2 relay, lấy tín hiệu đầu vào từ các hàng rào đặc được nối với công tắc hành trình và nút nhấn.
- Nguồn mạch động lực 24V đặc lấy từ domino 2 chân JP4.
- Nguồn mạch điều khiển được lấy từ cáp mini USB kết nối STM32F4 với máy tính.

- Các hàng rào cái đôi J1, J2, J3, J4 được đặt tên theo tên các chân của kit STM32F401DIC1 để hoạt động chính xác khi cắm kít vào board mạch.
- Các khối U1,3,5,7,8,9 là sơ đồ nguyên lý của DRV8825. Các chân EN', STEP, DIR được nối vào ngõ ra vi xử lý để điều khiển. Chân VMOT, FLT' nối nguồn 24V, 3V. Chân GND, GND_MOT được nối chung và nối với GND của vi xử lý để đồng bộ điện áp. Các chân M0, M1, M2 được nối với hàng rào đực để thuận tiện thay đổi vi bước điều khiển động cơ.
- Ngoài ra còn có 2 mạch kích relay đã trình bày ở trên.



Hình 3.31. Thiết kế mạch PCB

3.3. THIẾT KẾ PHẦN MỀM

Phần mềm nhóm sử dụng trong đề tài bao gồm: STM32CubeMX, Keil uVision5, Python 3.7.6.

Những công việc cần thực hiện:

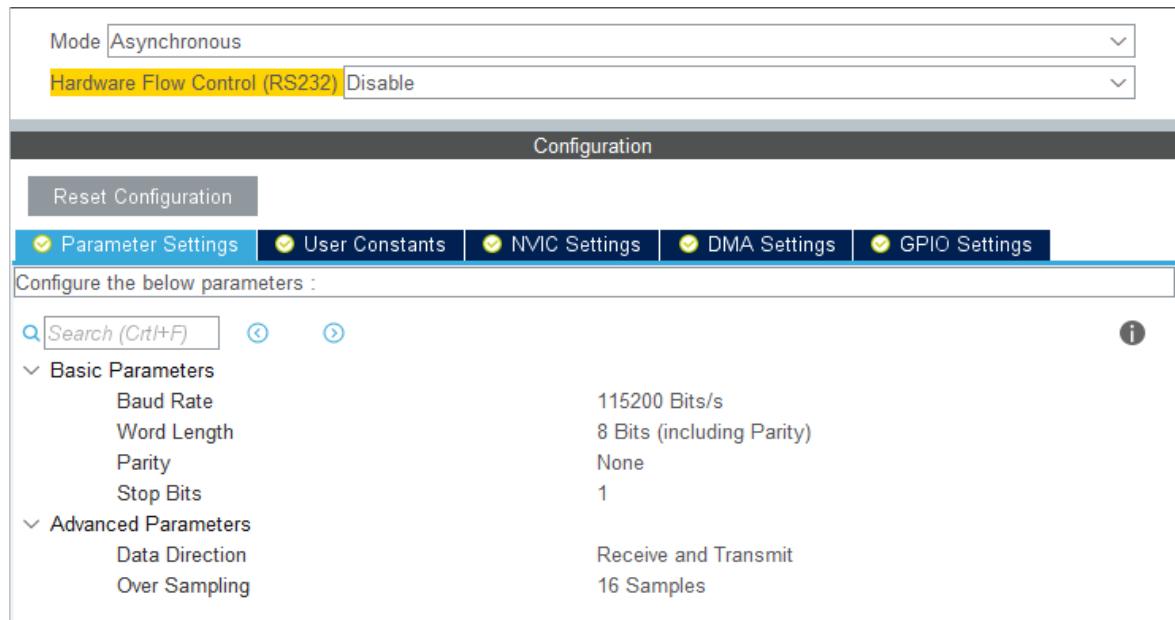
- Thiết lập phần cùn cho vi xử lý trên phần mềm STM32CubeMX.
- Lập trình cho STM32 trên Keil uVision5.
- Lập trình giao diện và xử lý ảnh trên Python.

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

3.3.1. Thiết lập phần cứng cho vi xử lý trên phần mềm STM32CubeMX

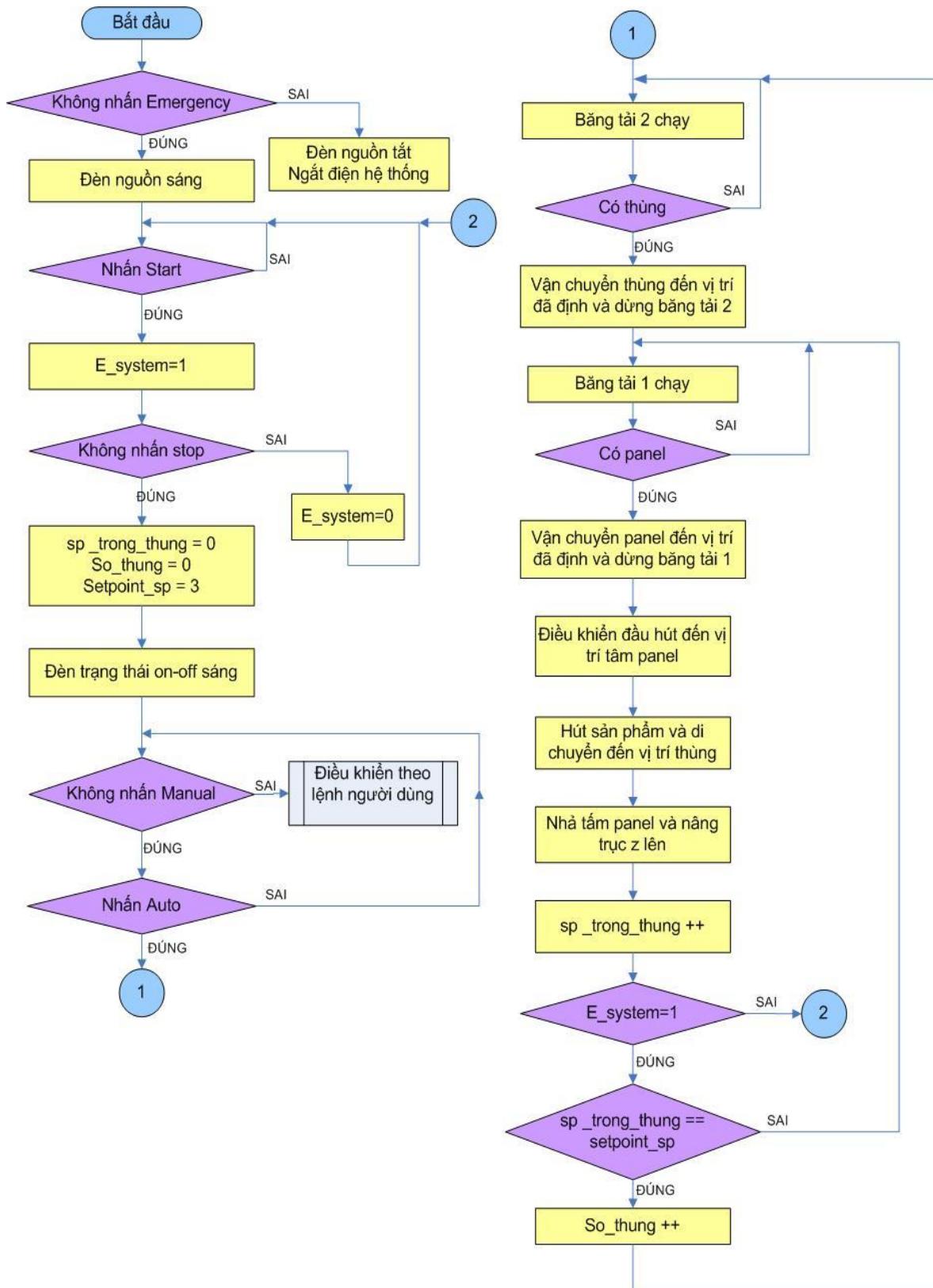
Pin Name	Signal on Pin	GPIO output...	GPIO mode	GPIO Pull-up...	Maximum ou...	User Label	Modified
PA0-WKUP	n/a	n/a	External Eve...	No pull-up an...	n/a	B1 [Blue Pu...	✓
PA1	n/a	Low	Output Push...	No pull-up an...	Low	DIR1	✓
PA3	n/a	Low	Output Push...	No pull-up an...	Low	STEP1	✓
PA15	n/a	Low	Output Push...	No pull-up an...	Low	EN6	✓
PB1	n/a	Low	Output Push...	No pull-up an...	Low	EN1	✓
PB2	n/a	n/a	Input mode	No pull-up an...	n/a	BOOT1	✓
PB4	n/a	n/a	Input mode	Pull-down	n/a	ENDSTOP4	✓
PB5	n/a	Low	Output Push...	No pull-up an...	Low	DIR4	✓
PB7	n/a	Low	Output Push...	No pull-up an...	Low	STEP4	✓
PB8	n/a	n/a	Input mode	Pull-down	n/a	ENDSTOP3	✓
PB11	n/a	Low	Output Push...	No pull-up an...	Low	STEP3	✓
PB13	n/a	Low	Output Push...	No pull-up an...	Low	MODSTEP3	✓
PB15	n/a	Low	Output Push...	No pull-up an...	Low	EN3	✓
PC0	n/a	High	Output Push...	No pull-up an...	Low	OTG_FS_Po...	✓
PC5	n/a	Low	Output Push...	No pull-up an...	Low	MODSTEP1	✓
PC6	n/a	Low	Output Push...	No pull-up an...	Low	MODSTEP6	✓
PC8	n/a	Low	Output Push...	No pull-up an...	Low	STEP6	✓
PC11	n/a	Low	Output Push...	No pull-up an...	Low	DIR5	✓
PC13-ANTI_...	n/a	Low	Output Push...	No pull-up an...	Low	EN4	✓
PD0	n/a	Low	Output Push...	No pull-up an...	Low	STEP5	✓
PD1	n/a	Low	Output Push...	No pull-up an...	Low	DIR6	✓
PD2	n/a	Low	Output Push...	No pull-up an...	Low	MODSTEP5	✓
PD4	n/a	Low	Output Push...	No pull-up an...	Low	Audio_RST [...]	✓
PD5	n/a	n/a	Input mode	No pull-up an...	n/a	OTG_FS_Ov...	✓
PD6	n/a	Low	Output Push...	No pull-up an...	Low	EN5	✓
PD7	n/a	Low	Output Push...	Pull-down	Low	ENDSTOP5	✓
PD10	n/a	Low	Output Push...	No pull-up an...	Low	Relay1	✓
PD11	n/a	Low	Output Push...	Pull-up	Low	Relay2	✓
PD12	n/a	Low	Output Push...	No pull-up an...	Low		□
PD13	n/a	Low	Output Push...	No pull-up an...	Low		□
PD14	n/a	Low	Output Push...	No pull-up an...	Low		□
PD15	n/a	Low	Output Push...	No pull-up an...	Low		□
PE0	n/a	n/a	Input mode	Pull-down	n/a	ENDSTOP2	✓
PE1	n/a	n/a	External Eve...	No pull-up an...	n/a	MEMS_INT2...	✓
PE3	n/a	Low	Output Push...	No pull-up an...	Low	CS_I2C/SPI ...	✓
PE5	n/a	Low	Output Push...	No pull-up an...	Low	MODSTEP4	✓
PE7	n/a	Low	Output Push...	No pull-up an...	Low	DIR2	✓
PE8	n/a	n/a	Input mode	Pull-down	n/a	ENDSTOP1	✓
PE9	n/a	Low	Output Push...	No pull-up an...	Low	STEP2	✓
PE11	n/a	Low	Output Push...	No pull-up an...	Low	MODSTEP2	✓
PE13	n/a	Low	Output Push...	No pull-up an...	Low	EN2	✓
PE15	n/a	Low	Output Push...	No pull-up an...	Low	DIR3	✓

Hình 3.32. Cài đặt GPIO cho các chân STM32



Hình 3.33. Cài đặt thông số truyền thông UART

3.3.2. Lập trình cho STM32 trên Keil uVision5



Hình 3.34. Lưu đồ giải thuật cho vi điều khiển.

3.3.3. Thuật toán xử lý ảnh

Quá trình xử lý ảnh gồm 2 phần riêng biệt: xử lý phát hiện sản phẩm đi vào và xử lý phát hiện thùng đi vào. Quá trình xử lý ảnh phát hiện sản phẩm sẽ trả về giá trị tọa độ x, y của sản phẩm để gửi tín hiệu xuống vi điều khiển để thực hiện gấp sản phẩm. Quá trình xử lý ảnh phát hiện thùng chỉ phát hiện tọa độ y của thùng đi vào. Do đặc tính thùng có đồ bóng và phản chiếu ánh sáng nên việc xác định các cạnh góc khuất là rất khó đối với nhóm hiện tại.

3.3.3.1. Thuật toán xử lý ảnh phát hiện sản phẩm

Khi bắt đầu chạy giao diện điều khiển, mặc định không có sản phẩm đi vào. Biến `co_san_pham = 0`.

Phần mềm tiến hành lấy hình ảnh từ camera. Thư viện OpenCV hỗ trợ lấy video từ camera bằng lệnh `cv2.VideoCapture()`. Từ video, ta lấy từng hình ảnh để xử lý bằng cách dùng lệnh `cv2.imread()`.

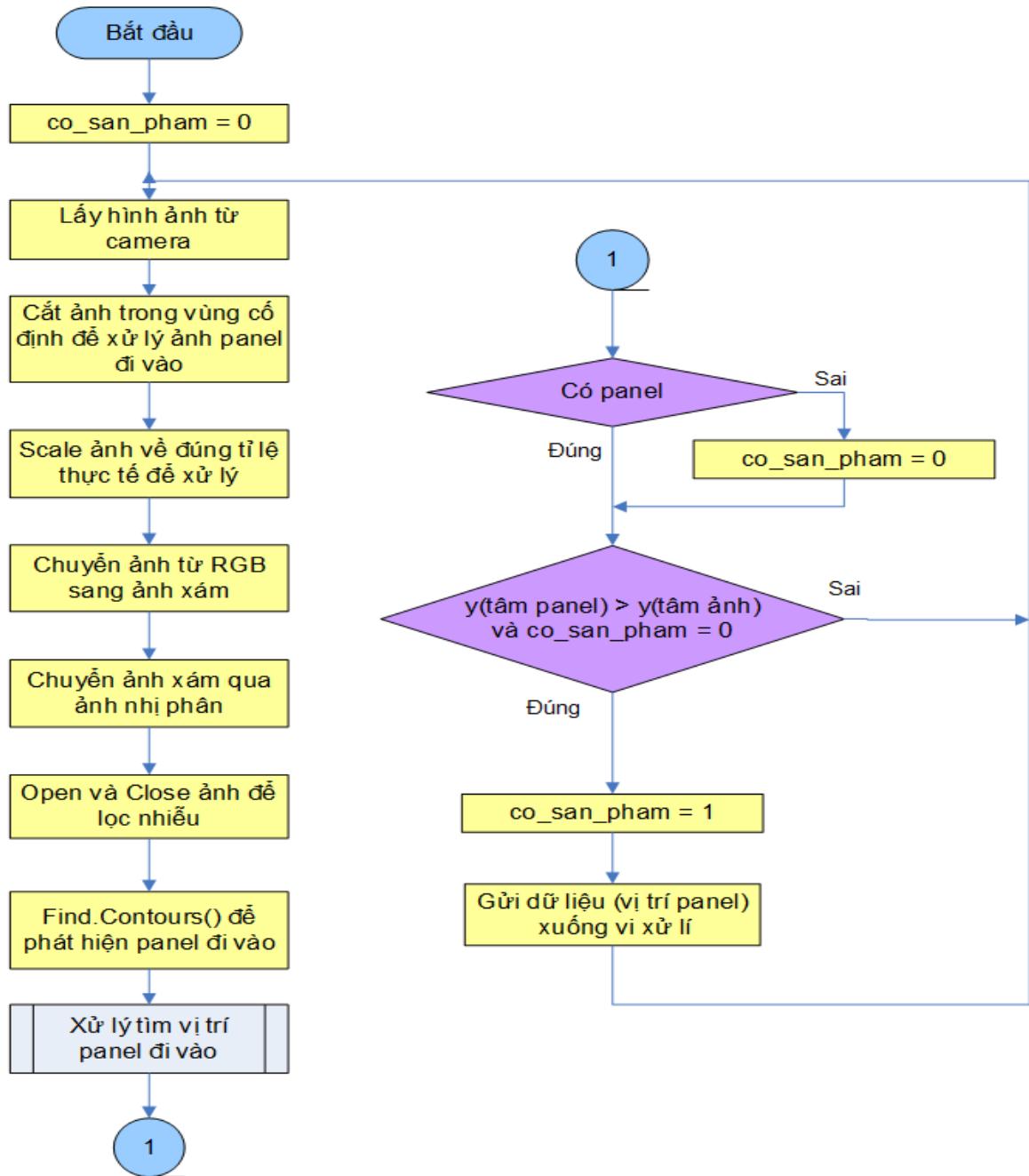
Vì hình ảnh thu được từ camera có quá nhiều thông tin, nhóm dùng cách cắt ảnh trong một vùng cố định để xử lý. Vùng xử lý vật được khoanh bằng 4 chấm đỏ. Ta có thể chọn 4 chấm này trong giao diện điều khiển.

Tuy nhiên để xử lý vùng này cần lưu thành dạng ảnh hình chữ nhật và scale ảnh về đúng tỉ lệ thật là 165x101 mm. Ảnh sau khi chuyển đổi có kích thước 430x263. Thư viện OpenCV hỗ trợ cắt và scale ảnh bằng lệnh `cv2.getPerspectiveTransform()` và `cv2.warpPerspective()`



Hình 3.35. Ảnh hệ thống chụp được từ camera

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ



Hình 3.36. Lưu đồ xử lý ảnh sản phẩm



Hình 3.37. Ảnh đã được cắt và scale lại đúng tỉ lệ thực tế

Sau khi đã có ảnh đúng tỉ lệ thực tế, ta tiến hành xử lý tìm vị trí:

Đầu tiên chuyển ảnh màu sang gray. Có 3 phương pháp để chuyển ảnh màu sang ảnh xám: lightness, average và linear luminance. Nhóm quyết định chọn phương pháp linear luminance vì những ưu điểm đã nêu trong phần cơ sở lý thuyết.

$$\text{img_gray} = ((0.3 * R) + (0.59 * G) + (0.11 * B))$$

Trong chương trình dùng lệnh cv2.cvtColor(image,
cv2.COLOR_BGR2GRAY)

Hàm này mặc định lần lượt 3 trong số là 0.3, 0.59, 0.11

Tiếp theo ta chuyển ảnh xám sang ảnh nhị phân bằng cách xét ngưỡng thresh. Hàm sử dụng trong Python là cv2.threshold(img_gray, thresh, 255, cv2.THRESH_BINARY)

Tham số thresh được lấy từ thanh slider trong giao diện thiết kế để phù hợp với từng điều kiện ánh sáng khác nhau. Mặc định trong chương trình bằng 0.



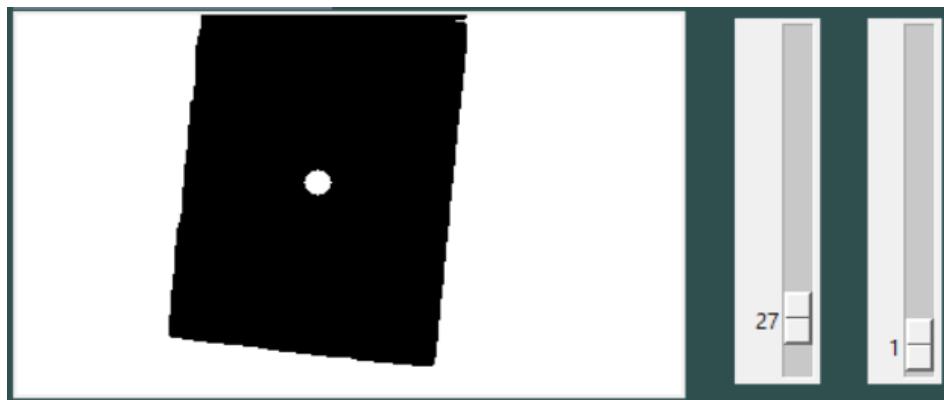
Hình 3.38. Ảnh nhị phân với ngưỡng thresh được chọn trên thanh slider

Sau đó tiến hành lọc nhiễu bằng phương pháp open và close.

Trong Python sử dụng hàm cv2.morphologyEx(thresh_img,
cv2.MORPH_OPEN, kernel) với kernel là mặt nạ đĩa dùng để so dán ảnh. Kích thước đĩa được thay đổi bằng thanh slider trong giao diện để khử nhiễu với nhiều kích thước khác nhau.

CHƯƠNG 3. TÍNH TOÁN VÀ THIẾT KẾ

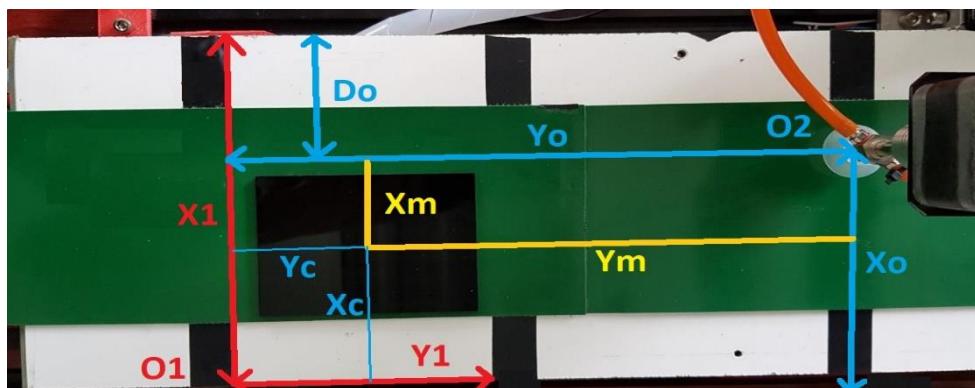
Sau khi khử nhiễu, ảnh chỉ còn 1 khối đặt đồng nhất màu đen chính là ảnh tấm panel sản phẩm đã qua xử lý. Để xác định tâm panel, thư viện OpenCV hỗ trợ hàm cv2.findContours() xác định tâm khối đặt. Hàm trả về tọa độ tâm trong ảnh.



Hình 3.39. Tâm panel đã được xác định tương đối chính xác

Sau khi tìm được vị trí panel trong ảnh, ta cần tính toán để tìm ra vị trí thật của panel để tiến hành gấp vật.

Để xác định vị trí panel đi vào, nhóm sử dụng phương pháp hình học như sau:



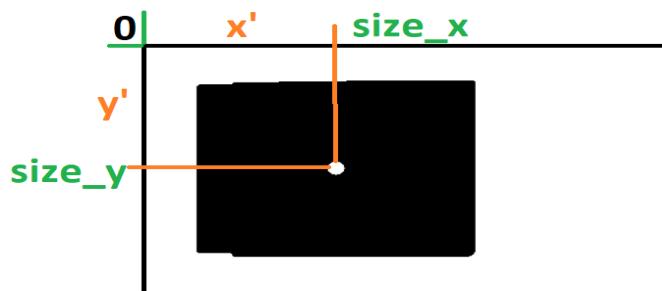
Hình 3.40. Vị trí của panel trong mô hình

Thực tế các thông số Yo, Xo, Y1, X1, Yc, Xc, Do, Ym đã được xác định và không thay đổi:

- $Yo = 195 \text{ mm}$
- $Do = 55 \text{ mm}$
- $Xo = 110 \text{ mm}$
- $X1 = 165 \text{ mm}$
- $Y1 = 101 \text{ mm}$
- $Yc = Y1/2 = 50 \text{ mm}$

$$\Rightarrow Y_m = Y_o - Y_c = 195 - 50 = 145 \text{ mm}$$

Ảnh vùng xử lý được cắt như sau:



Hình 3.41. Vùng xử lí được cắt

Kích thước ảnh trong vùng xử lý đã được đặt trước là $\text{size_x}=430$, $\text{size_y}=263$.

Vị trí x' , y' của tâm panel trong ảnh được truy xuất từ hàm `findContours` của OpenCV.

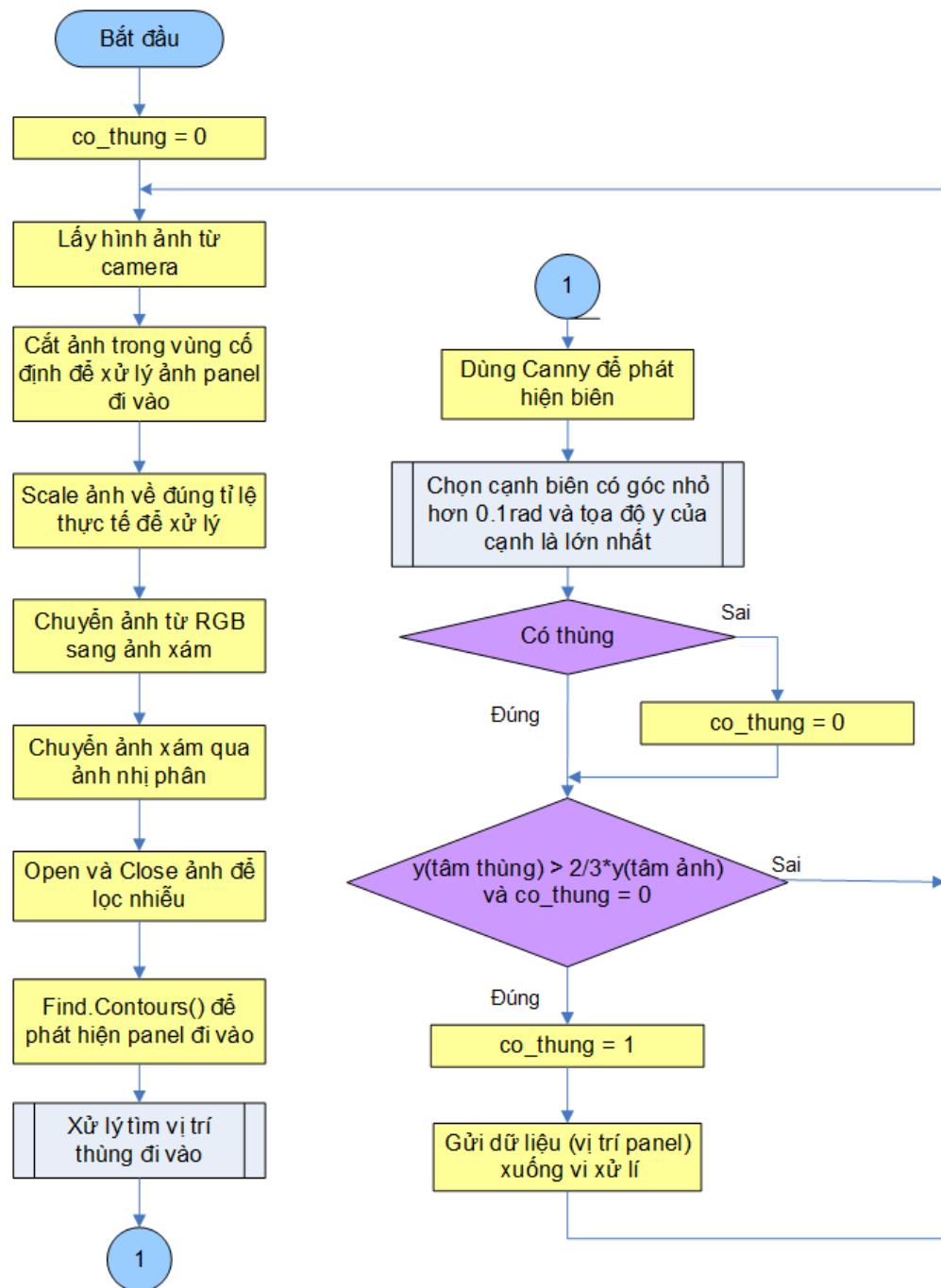
$$\text{Ta có: } X_c = x' * X_1 / \text{size_x}$$

$$\Rightarrow X_m = X_o - X_c = 110 - x' * X_1 / \text{size_x}$$

Như vậy ta đã xử lý xong và tìm được vị trí của sản phẩm đi vào.

Tuy nhiên phần mềm lấy ảnh từ camera liên tục nên mỗi lần xử lý ảnh lại phát hiện một sản phẩm mới. Để khắc phục điều này, nhóm giới hạn điều kiện để phát hiện một sản phẩm mới là tâm sản phẩm phải đi qua nửa ảnh và chưa có sản phẩm mới nào (biến `co_san_pham = 0`). Sau đó set biến `co_san_pham = 1`. Chỉ khi nào không còn vật trong ảnh mới reset biến `co_san_pham = 0`.

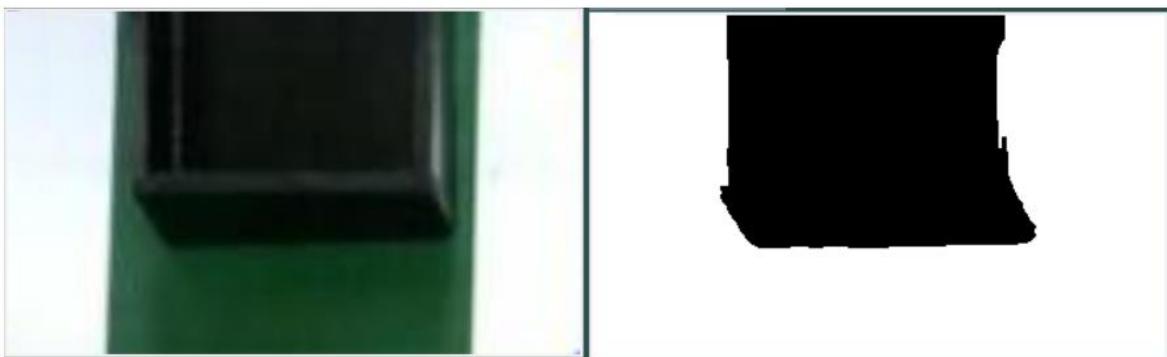
3.3.3.2. Thuật toán xử lý ảnh phát hiện thùng đi vào



Hình 3.42. Lưu đồ xử lý ảnh phát hiện thùng

Quá trình phát hiện thùng đi vào tương tự như quá trình phát hiện sản phẩm đi vào.

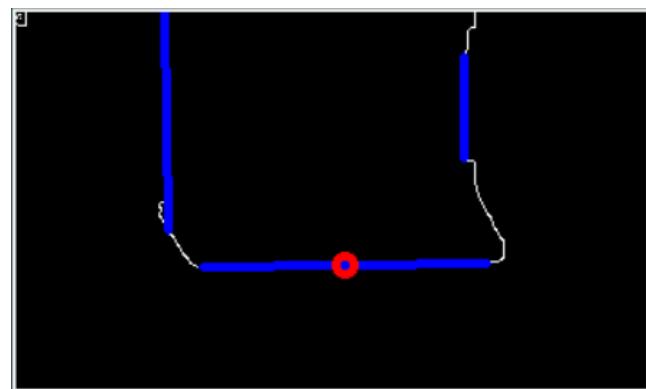
Kết quả như hình 3.43.



Hình 3.43. Ảnh vùng xử lý thùng trước và sau khi xử lý opening ảnh.

Ta thấy thùng có độ nổi khối và bóng nên việc dùng tâm của khối phát hiện được trong ảnh nhị phân để xác định vị trí thùng là không chính xác. Vì vậy nhóm quét định dùng phương pháp phát hiện cạnh biên để phát hiện thùng đi vào.

Có 2 phương pháp phát hiện biên phổ biến là Canny và Sobel. Nhóm quyết định chọn phương pháp Canny vì chất lượng đường biên khá mượt so với dùng phương pháp Sobel. Kết quả thu được như hình 4.13.



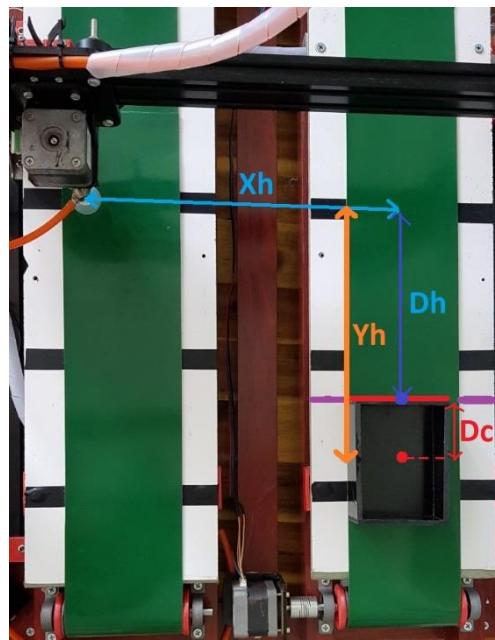
Hình 3.44. Ảnh phát hiện biên của thùng đi vào

Trong các đường biên phát hiện được, ta thấy đường có khả năng phát hiện được cao hơn là đường ngang dưới cùng.

Nếu vị trí y đường ngang này lớn hơn $2/3$ kích thước ảnh trong vùng xử lý, sản phẩm mới sẽ được phát hiện.

Giải pháp không ché không cho xác nhận sản phẩm mới đi vào nhiều lần hoàn toàn giống như phần xử lý ảnh phát hiện sản phẩm.

Vị trí thùng đi vào được tính toán như sau:



Hình 3.45. Vị trí thùng trong vùng xử lý ảnh và thực tế.

$$Xh = 265$$

$$Dh = 177$$

$$Dc = 53$$

$$\Rightarrow Yh = Dh + Dc = 230$$

Như vậy ta có vị trí thùng đi vào luôn được thiết lập là $Xh=265$ và $Yh=230$

Chương 4. THI CÔNG HỆ THỐNG

4.1. YÊU CẦU ĐIỀU KHIỂN

Hệ thống được thiết kế với các yêu cầu sau:

- Hai băng tải hoạt động tốt và có thể điều khiển được vị trí.
- Nhận diện được vật và tiến hành xử lý ảnh thông qua hình ảnh được ghi lại nhờ Webcam Logitech.
- Khi vật đến vị trí quy định, băng tải sẽ dừng và cơ cấu di chuyển vật hoạt động gấp vật đến vị trí thùng.
- Khi trong thùng đã đủ số lượng sản phẩm, băng tải thùng sẽ di chuyển đưa thùng đến khâu lưu kho.
- Hệ thống có thể chỉnh sửa và nâng cấp chương trình

4.2. MÔ TẢ HOẠT ĐỘNG CỦA HỆ THỐNG

Hệ thống được giám sát và điều khiển bởi một bảng điều khiển vật lý và một bảng điều khiển trên máy tính.

Bảng điều khiển vật lý có đèn báo màu vàng để báo nguồn mạch động lực và đèn báo màu xanh để báo hiệu hệ thống đang hoạt động. Nút nhấn khẩn cấp để ngắt mạch động lực ngay lập tức khi nhấn, đèn xanh tắt, mạch điều khiển vẫn hoạt động. Khi nhấn START hệ thống hoạt động, nhấn STOP hệ thống dừng lại.

Bảng điều khiển trên máy tính có nút START, STOP như bảng điều khiển vật lý. Chế độ MANUAL dùng để điều khiển vị trí các trục x, z, van khí nén và băng chuyền. Chế độ AUTO cho phép hệ thống tự động đưa thùng và sản phẩm vào, xử lý ảnh và gấp chính xác sản phẩm để thả vào thùng. Ta có thể cài đặt các thông số xử lý ảnh, số sản phẩm mỗi thùng. Ngoài ra còn hiển thị hình ảnh thu được từ camera và hình ảnh đã xử lý.

4.3. THI CÔNG HỆ THỐNG

Trong quá trình thi công và hoàn thiện hệ thống, nhóm đã thực hiện 2 công việc chính là thi công phần cơ khí và phần mềm.

4.3.1. Thi công cơ khí

Phần cơ khí của hệ thống gồm 2 phần chính: hệ thống đưa sản phẩm và thùng vào/ra và hệ thống gấp và thả sản phẩm.

4.3.2. Thi công mạch điều khiển

Việc thi công mạch điều khiển bao gồm các bước sau:

CHƯƠNG 4. THI CÔNG HỆ THỐNG

- Xuất mạch in PCB ra file pdf và in.
- ũi board đồng.
- Khoan lỗ để cắm linh kiện.
- Hàn chân linh kiện vào mạch.
- Tráng nhựa thông để bảo vệ mạch.

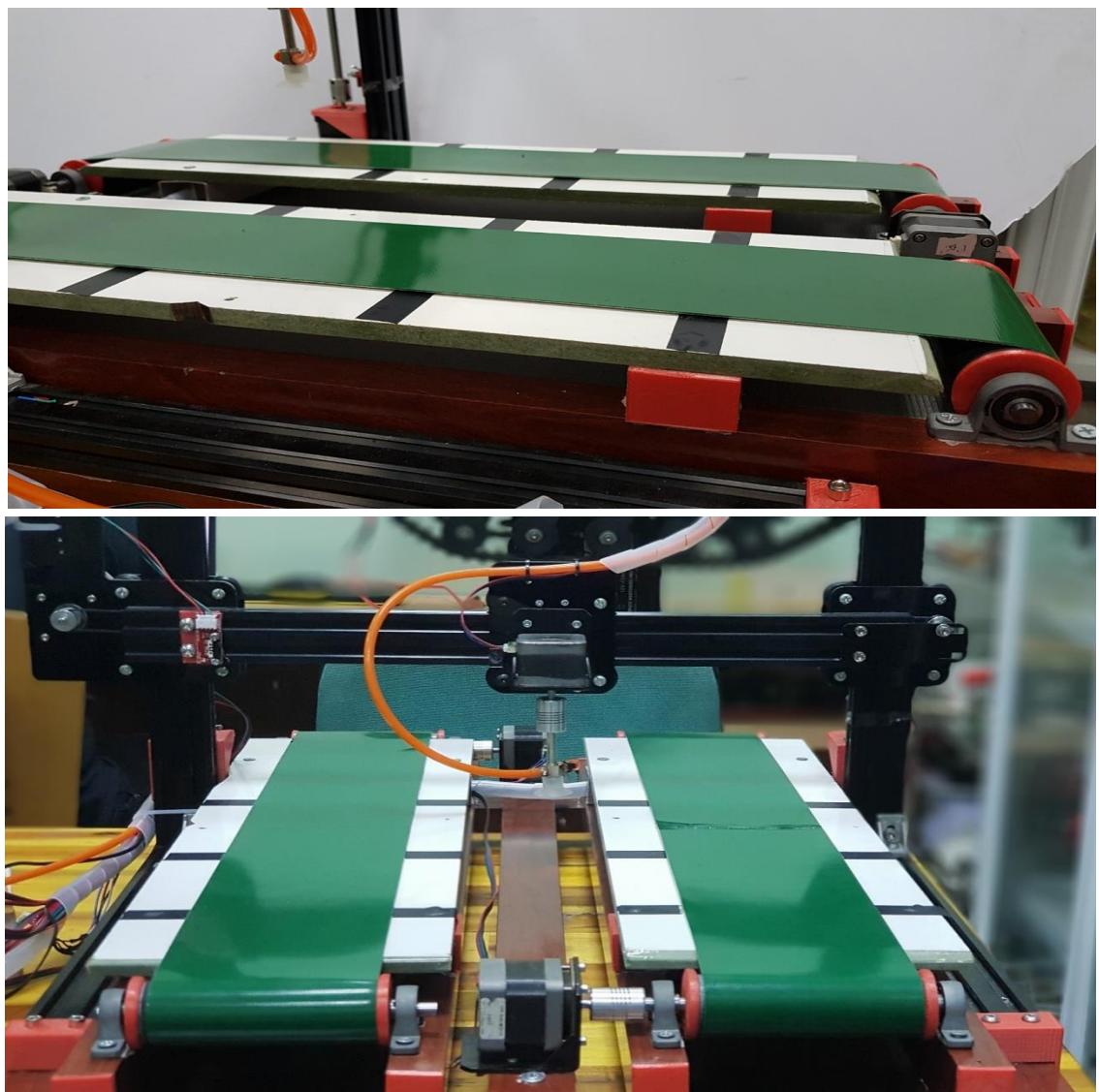
Chương 5. KẾT QUẢ THỰC HIỆN

5.1. KẾT QUẢ THỰC HIỆN

5.1.1 Kết quả thi công cơ khí

5.1.1.1 Kết quả thi công băng chuyền

Nhóm đã hoàn thành 2 mô hình băng tải gắn trên khung để chắc chắn. Tốc độ băng tải có thể thay đổi được và tốc độ tối đa là 630 mm/s. Băng chuyền có khả năng chịu tải 0.5 kg và có thể thay đổi momen xoắn bằng cách vặn biến trở của driver DRV8825. Băng tải được thiết kế có thể điều khiển vị trí chính xác đến 0.01 mm.



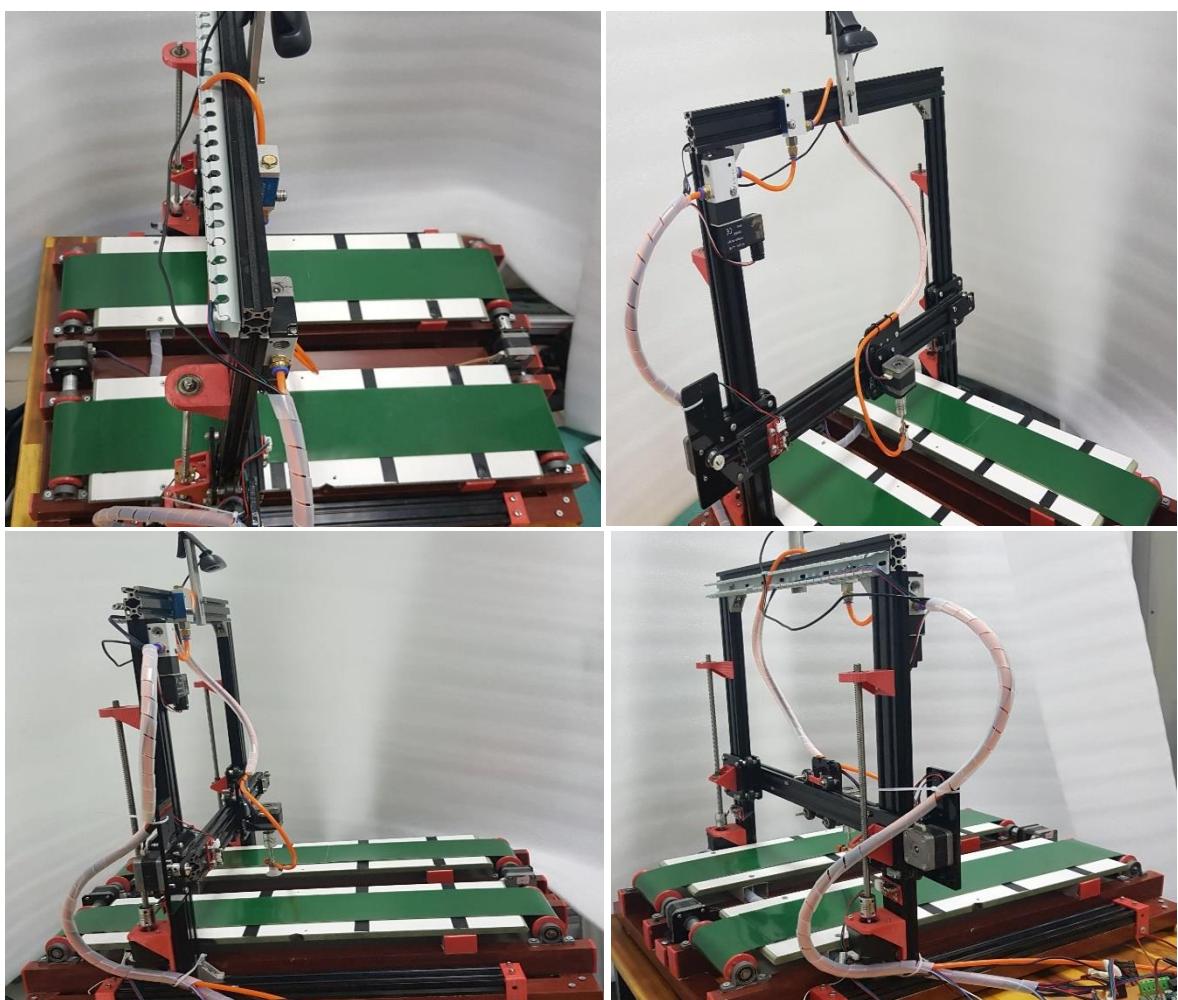
Hình 5.1. Mô hình 2 băng tải vận chuyển sản phẩm và thùng

5.1.1.2 Kết quả thi hệ thống gấp và thả sản phẩm.

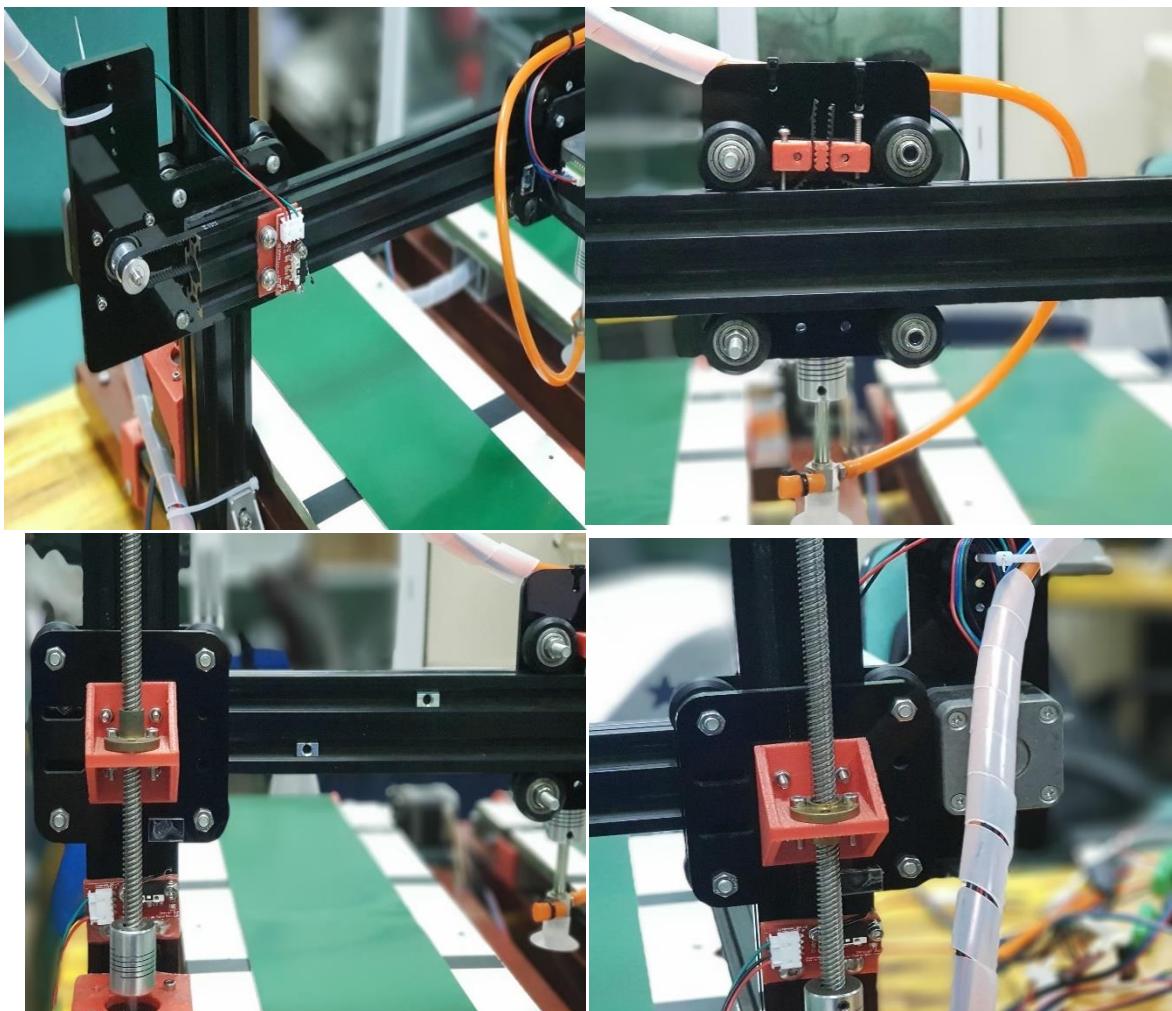
Hệ thống gấp, thả được điều khiển bởi 2 trục X, Z và hệ thống hút chân không dùng khí nén.

Trục X, Z được thi công và lắp ghép chính xác để có thể di chuyển linh hoạt và độ chính xác cao. Tốc độ tối đa của trục X là 200 mm/s, tốc độ tối đa của trục z là 40 mm/s.

Hệ thống hút chân không có thể gấp và thả vật dễ dàng, tốc độ đáp ứng tức thời.



Hình 5.2. Hệ thống gấp và thả sản phẩm



Hình 5.3. Các cơ cấu di chuyển của cơ cấu gấp sản phẩm

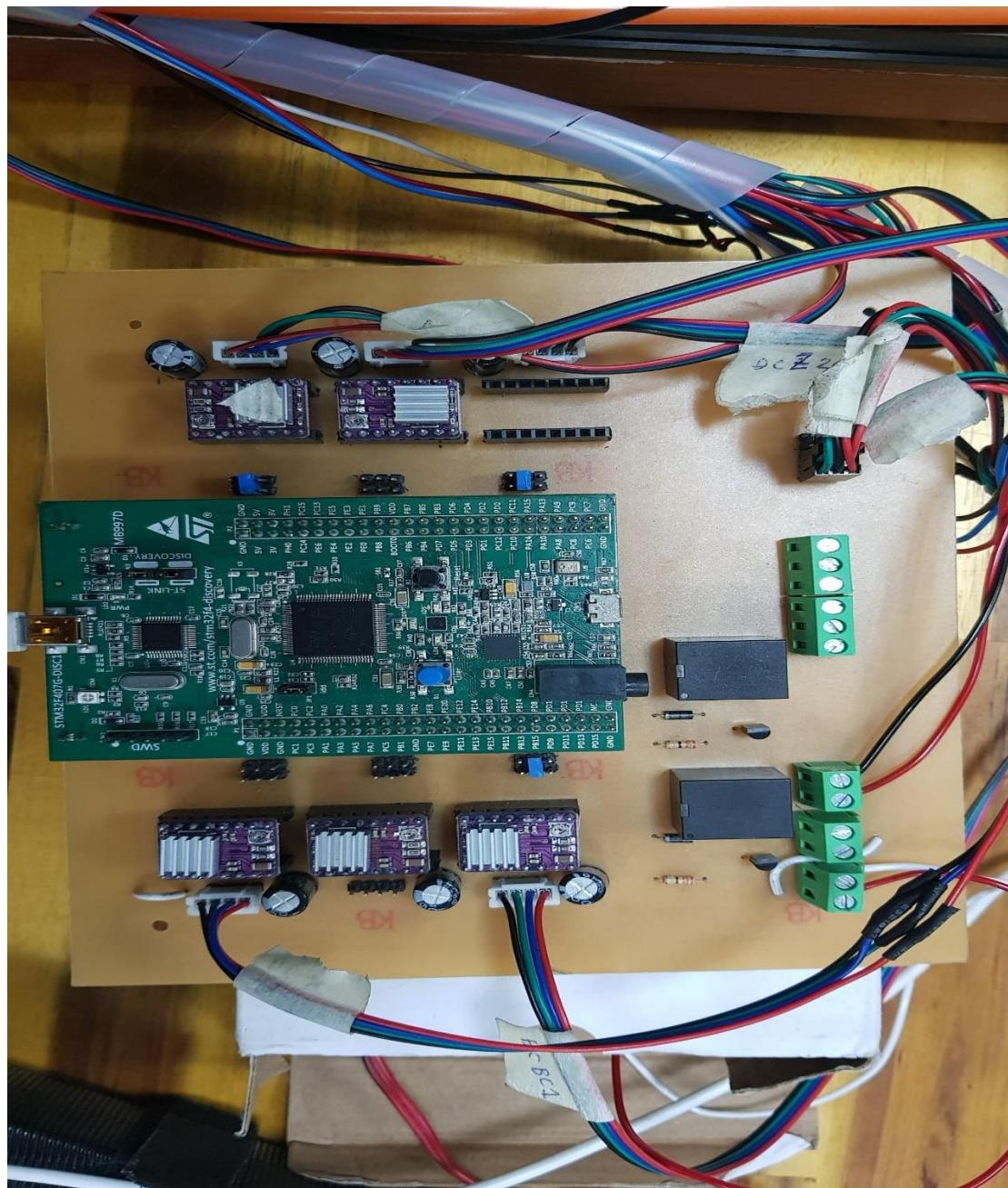
5.1.2. Kết quả thi công mạch điều khiển

Mạch điều khiển được bảo vệ bởi bảng điều khiển có nút nhấn và đèn.



Hình 5.4. Bảng điều khiển vật lý

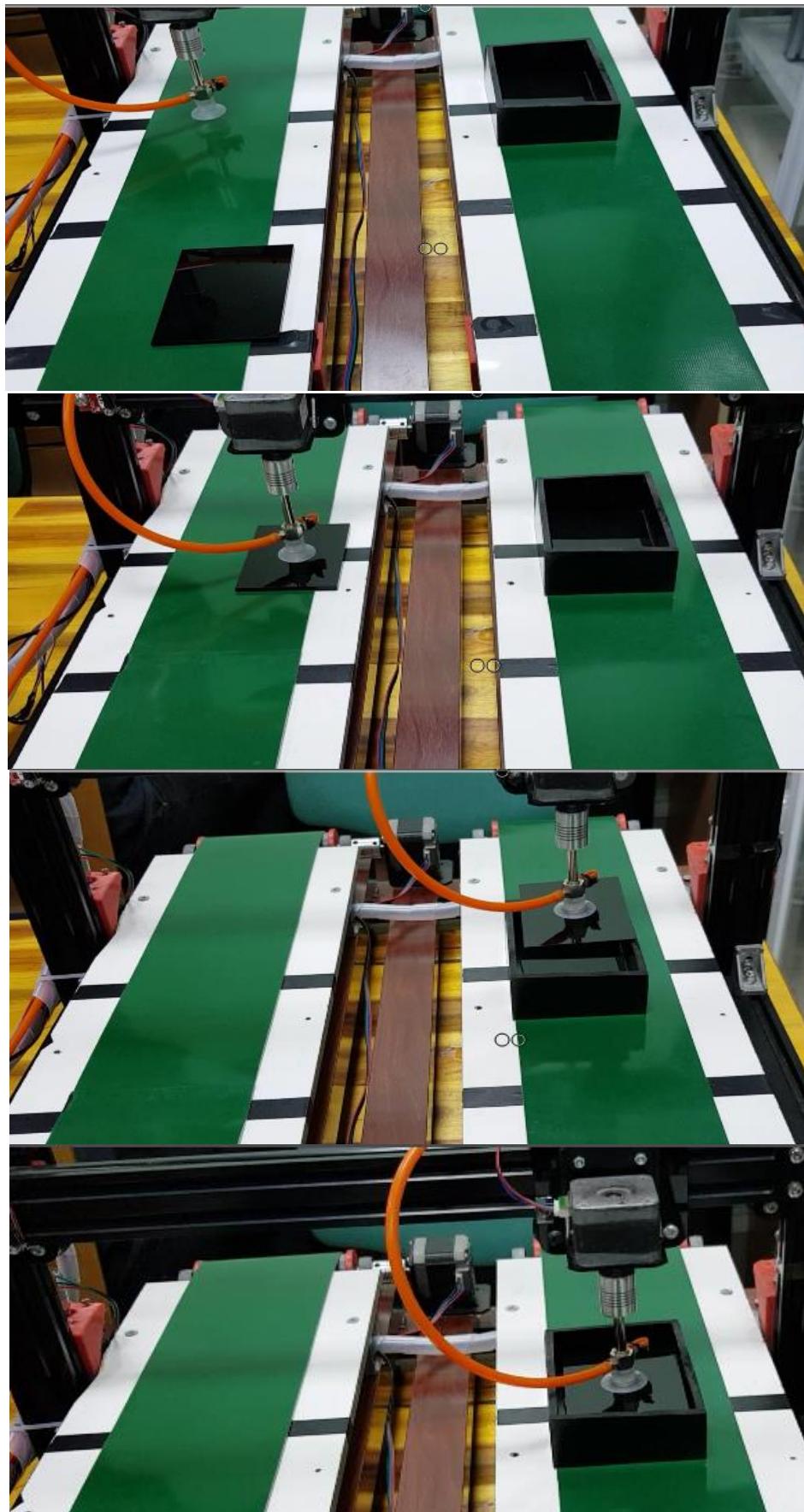
Mạch điều khiển có thể điều khiển 6 động cơ step cùng lúc, 1 van khí nén 3/2 và 1 đèn 24V.



Hình 5.5. Mạch điều khiển hệ thống

5.1.3. Kết quả điều khiển và giám sát hệ thống

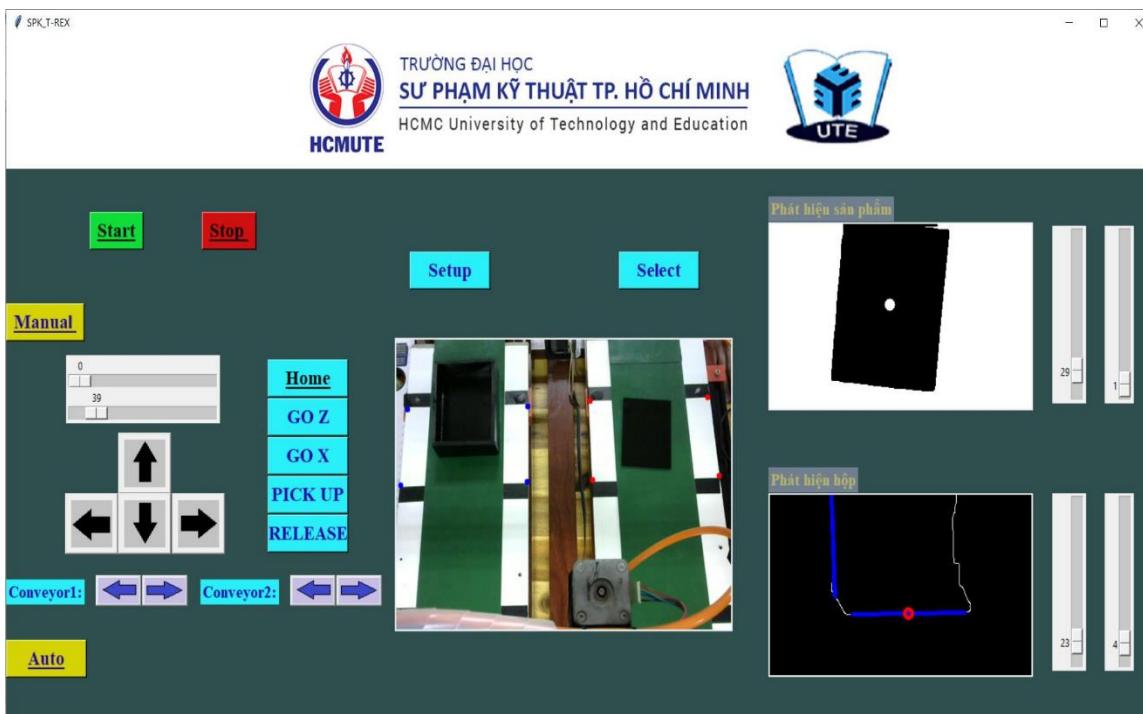
Hệ thống điều khiển hoạt động chính xác và có thể gấp vật thả vào thùng với sai lệch nhỏ



Hình 5.6. Hệ thống đang gấp vật bỏ vào thùng

CHƯƠNG 5. KẾT QUẢ THỰC HIỆN

Bảng điều khiển dễ sử dụng và hoạt động chính xác lệnh người dùng



Hình 5.7. Giao diện điều khiển hệ thống trên máy tính
Video hoạt động của hệ thống.

[Link youtube: https://youtu.be/inw7zRvi9oA](https://youtu.be/inw7zRvi9oA)

5.2. ĐÁNH GIÁ HOẠT ĐỘNG CỦA HỆ THỐNG

Hệ thống hoạt động tốt và có độ chính xác cao, có thể phát triển và áp dụng vào dây chuyền sản xuất.

Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. KẾT LUẬN

6.1.1. Ưu điểm

Sau thời gian nghiên cứu và thực hiện đề tài “ROBOT TỰ ĐỘNG GẮP SẢN PHẨM ỦNG DỤNG CÔNG NGHỆ XỬ LÝ ẢNH”, nhóm đã ứng dụng thành công những kiến thức đã học và đạt được những yêu cầu mà đề tài đã đặt ra ban đầu về xử lí. Thiết kế và điều khiển thành công hệ thống với 2 chế độ hoạt động đáp ứng được đa dạng nhu cầu trong thực tế. Giao diện điều khiển cung cấp đầy đủ tính năng quản lí, giám sát và điều khiển hệ thống. Giao diện được thiết kế đơn giản, dễ sử dụng và than thiện với người dùng. Có thể kết hợp với các hệ thống khác để xây dựng một dây chuyền liên tục từ đó mang lại khả năng mở rộng quy mô hoặc phát triển hệ thống.

6.1.2. Hạn chế

- ❖ Thi công phần cứng chỉ dừng lại ở mức độ mô hình.
- ❖ Lựa chọn thiết bị chưa thực sự tối ưu đối với yêu cầu của hệ thống.
- ❖ Thời gian đáp ứng chưa thực sự nhanh.

6.2. HƯỚNG PHÁT TRIỂN

- ❖ Nâng cao giải thuật xử lý ảnh để mô hình hoàn hiện hơn, có thể xử lí nhanh với độ chính xác cao
- ❖ Phát triển mô hình thành hệ thống lớn có thể áp dụng phục vụ các công việc trong công nghiệp. Tích hợp hệ thống vào các hệ thống điều khiển chung của nhà máy, xí nghiệp tạo nên một dây chuyền sản xuất tự động

TÀI LIỆU THAM KHẢO

Kênh youtube tham khảo:

Nguyễn Văn Thái: Image Processing

Link: <https://www.youtube.com/channel/UCiqTWB69N8-DniwukPbH1Ow>

Trang web tham khảo:

- [1] <http://kitstm32f4.blogspot.com/2015/11/21gioi-thieu-chung-ve-dong-arm-cortex.html>
 - [2] https://www.st.com/resource/en/user_manual/dm00039084-discovery-kit-with-stm32f407vg-mcu-stmicroelectronics.pdf
 - [3] <https://vidieukhien.xyz/2018/03/22/bai-7-stm32f4-usart/>
 - [4] <https://www.logitech.com/vi-vn/product/hd-webcam-c270>
 - [6] <https://www.thegioiic.com/>
 - [7] [https://vi.wikipedia.org/wiki/Python_\(ng%C3%B4n_ng%C3%A1ng%BB%AF_1%E1%BA%ADp_tr%C3%ACnh%ACnh\)#cite_note-24](https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%C3%A1ng%BB%AF_1%E1%BA%ADp_tr%C3%ACnh%ACnh)#cite_note-24)
 - [8] <https://www.stdio.vn/article/xu-ly-anh-voi-opencv-cac-phep-toan-hinh-thai-hoc-Nljcg>
 - [9] <http://vandientuchinhhang.com/van-dien-tu-khi-nen-airtac-3v210-08--van-khi-nen-3-2--ren-13mm--sp674.html>
 - [10] <http://vattucn.com/may-tron/thiet-bi-do-khi/van-chan-khong-comvum-cv-10hs-detail.html>
 - [12] https://www.tutorialspoint.com/dip/grayscale_to_rgb_conversion.htm#:~:text=Average%20method%20is%20the%20most,get%20your%20desired%20grayscale%20image.
 - [13] <https://techutorialsx.com/2019/04/13/python-opencv-converting-image-to-black-and-white/>
 - [14] <https://www.geeksforgeeks.org/difference-between-opening-and-closing-in-digital-image-processing/>
 - [15] https://docs.opencv.org/trunk/d9/d61/tutorial_py_morphological_ops.html
 - [16] https://docs.opencv.org/master/da/d22/tutorial_py_canny.html
- Giáo trình tham khảo:**
- [17] Nguyễn Thanh Hải, “Giáo Trình Xử lý ảnh”, Nhà xuất bản Đại Học Quốc Gia, Thành phố Hồ Chí Minh, 2014.
 - [18] “Giáo trình xử lý ảnh”, Học viện bưu chính viễn thông
 - [19] Rafael C. Gonzalez, Richard E. Woods - Digital Image Processing (2008)

PHỤ LỤC

1. Chương trình giao diện điều khiển và xử lý ảnh.

```
from PIL import Image
from PIL import ImageTk
from tkinter import messagebox
import tkinter
import cv2
import numpy as np
import math
import imutils
import string
import serial
import time
ser = serial.Serial(
    port='COM8',\
    baudrate=115200,\ 
    parity=serial.PARITY_NONE,\ 
    stopbits=serial.STOPBITS_ONE,\ 
    bytesize=serial.EIGHTBITS,\ 
    timeout=0)
# chon vung xu ly anh
enable_chon_vung_xla=0
count_chon_vung_xla=
setup=0
dcx=30
dcy=30
diem_vung_xla=[(370,102), (598,98), (375,232),(617,227),(20,114), (251,108),
(8,241),(248,232)]
# kich thuoc khung xla
kty_chuan = 101
ktx_chuan = 165
xchuan = 350
ychuan = int(xchuan * kty_chuan / ktx_chuan)
# kich thuoc chuan vung xla hop
kty_chuan_hop = 98
ktx_chuan_hop = 164
xchuan_hop = 350
```

```

ychuan_hop = int(xchuan_hop * kty_chuan_hop / ktx_chuan_hop)
# bien tam
co_hop_moi=0
co_sp_moi=0
da_gap_sp=0
# di chuyen lien tuc manual
chedo_manual=0
cho_phep_di_chuyen_lien_tuc_manual=0
manualbc1l=0
manualbc1r=0
manualbc2l=0
manualbc2r=0
# che do auto
chedo_auto=0
gui_tin_hieu_co_sp_moi_xuong_vxl=0
vitrisp=57
vitrihop=265
kich_thuoc_chuoi_gui=60

def nothing(x):
    pass
def dishes(r):
    if r==0:
        r=1
    matrixx=np.ones(((2*r-1),(2*r-1)), np.uint8)

    for x in range(2*r-1):
        # print(x)
        for y in range(2*r-1):

            if (pow(x-r+1,2) + pow(y-r+1,2)) <= pow((r-1),2) +1:
                matrixx[x, y] = 0

            else:
                matrixx[x, y] = 1

    # print(matrixx)

```

```

return matrixx

def tao_chuoi_gui(a,b):
    global kich_thuoc_chuoi_gui
    if(b==None):
        c = f"!{a}@"
    else:
        c=f"!{a}@{b}%"
    if(len(c)<=kich_thuoc_chuoi_gui):
        d="0"*(kich_thuoc_chuoi_gui-len(c))
        c=f"{c}{d}"
    return c

    elif(len(c)>kich_thuoc_chuoi_gui):
        return None

def tao_chuoi_gui3(a,b,d):
    global kich_thuoc_chuoi_gui
    if(b==None):
        c = f"!{a}@"
    else:
        c=f"!{a}@{b}%{d}^"
    if(len(c)<=kich_thuoc_chuoi_gui):
        d="0"*(kich_thuoc_chuoi_gui-len(c))
        c=f"{c}{d}"
    return c

    elif(len(c)>kich_thuoc_chuoi_gui):
        return None

def tao_chuoi_gui4(a,b,d,e):
    global kich_thuoc_chuoi_gui
    if(b==None):
        c = f"!{a}@"
    else:
        c=f"!{a}@{b}%{d}^{e}&"
    if(len(c)<=kich_thuoc_chuoi_gui):
        d="0"*(kich_thuoc_chuoi_gui-len(c))
        c=f"{c}{d}"
    return c

```

```

elif(len(c)>kich_thuoc_chuoi_gui):
    return None
def tim_giao_diem(diem1,diem2):
    r1=diem1[0]
    theta1=diem1[1]
    r2 = diem2[0]
    theta2 = diem2[1]
    if(theta1==theta2):
        return None
    else:
        if(theta1==math.pi/2):
            a1=0
            b1=1
            c1=r1
        elif(theta1==0):
            a1=1
            b1=0
            c1=r1
        else:
            a1=-np.tan(math.pi-theta1)
            b1=1
            c1=r1/np.sin(theta1)
#
        if (theta2 == math.pi / 2):
            a2 = 0
            b2 = 1
            c2 = r2
        elif (theta2 == 0):
            a2 = 1
            b2 = 0
            c2 = r2
        else:
            a2=-np.tan(math.pi-theta2)
            b2=1
            c2=r2/np.sin(theta2)

dx=c1*b2-c2*b1

```

```

dy=a1*c2-a2*c1
d=a1*b2-a2*b1

x=dx/d
y=dy/d
gg = (int(x), int(y))
return gg

class videoStream:
    def __init__(self):
        self.root = tkinter.Tk()
        self.root.title('SPK_T-REX')
        self.root.geometry("1800x900")
        self.root.configure(bg="dark slate gray")
        self.panel = tkinter.Label(self.root)
        self.panel.pack()
        self.panel2 = tkinter.Label(self.root)
        self.panel2.pack()
        self.panel3 = tkinter.Label(self.root)
        self.panel3.pack()
        self.panel4 = tkinter.Label(self.root)
        self.panel4.pack()
        bard = Image.open("logo.png")
        img = ImageTk.PhotoImage(bard)
        lb = tkinter.Label(self.root, image=img, width=1600, height=150,
                           bg='WHITE')
        lb.image = img
        lb.place(x=0, y=0)
        self.btnStart = tkinter.Button(self.root, text="Start", font=("Times New
Roman", 17, "bold", "underline"),
                                       fg="#000000",bg="#12de3b",command=self.func_start)
        self.btnStart.place(x=112, y=204)
        self.btnStop = tkinter.Button(self.root, text="Stop ", font=("Times New
Roman", 17, "bold", "underline"),
                                       fg="#000000",bg="#d11111",command=self.func_stop)
        self.btnStop.place(x=262, y=204)

```

```

        self.btnManual = tkinter.Button(self.root, text="Manual ", font=("Times New
Roman", 17, "bold", "underline"),
                                    fg="#2207a8",
bg="#d6d304",command=self.func_manual)
self.btnManual.place(x=0, y=310)

        self.btnHome = tkinter.Button(self.root, text="Home", font=("Times New
Roman", 17, "bold", "underline"),
                                    fg="#000000", bg="#2af0fa", width=7,
command=self.func_home)
self.btnHome.place(x=350, y=375)
        self.btnGo1 = tkinter.Button(self.root, text="GO Z", font=("Times New
Roman", 17, "bold"),
                                    fg="#2207a8",
bg="#2af0fa",width=7,command=self.func_goz)
self.btnGo1.place(x=350, y=420)
        self.btnGo2 = tkinter.Button(self.root, text="GO X", font=("Times New
Roman", 17, "bold"),
                                    fg="#2207a8",
bg="#2af0fa",width=7,command=self.func_gox)
self.btnGo2.place(x=350, y=465)
        self.btn11 = tkinter.Button(self.root, text="Setup", font=("Times New Roman",
17, "bold"),
                                    fg="#2207a8", bg="#2af0fa", width=7,
command=self.func_btn11)
self.btn11.place(x=540, y=250)

        self.btn13 = tkinter.Button(self.root, text="Select", font=("Times New
Roman", 17, "bold"),
                                    fg="#2207a8", bg="#2af0fa", width=7,
command=self.func_btn13)
self.btn13.place(x=820, y=250)
bard1 = Image.open("up.png")
img1 = ImageTk.PhotoImage(bard1)
        self.btnUp = tkinter.Button(self.root, image = img1,relief=tkinter.RAISED,
height = 65,width = 65,command=self.func_btnUp)
self.btnUp.place(x=150, y=460)

```

```

bard2 = Image.open("down.png")
img2 = ImageTk.PhotoImage(bard2)
self.btnExit = tkinter.Button(self.root, image=img2, width=65,
height=65,command=self.func_btnDwn)
self.btnExit.place(x=150, y=530)
bard3 = Image.open("left.png")
img3 = ImageTk.PhotoImage(bard3)
self.btnExitLeft = tkinter.Button(self.root, image=img3, width=65,
height=65,command=self.func_btnLeft)
self.btnExitLeft.place(x=79, y=530)
bard4 = Image.open("right.png")
img4 = ImageTk.PhotoImage(bard4)
self.btnExitRight = tkinter.Button(self.root, image=img4, width=65,
height=65,command=self.func_btnRight)
self.btnExitRight.place(x=222, y=530)
self.btnExitPickup = tkinter.Button(self.root, text="PICK UP", font=("Times New
Roman", 17, "bold"),
fg="#2207a8", bg="#2af0fa",
width=7,command=self.func_pickup)
self.btnExitPickup.place(x=350, y=510)
self.btnExitPickup = tkinter.Button(self.root, text="RELEASE", font=("Times
New Roman", 17, "bold"),
fg="#2207a8", bg="#2af0fa",
width=7,command=self.func_release)
self.btnExitPickup.place(x=350, y=555)
self.btnExitAuto = tkinter.Button(self.root, text="Auto", font=("Times New
Roman", 17, "bold","underline"),
fg="#2207a8", bg="#d6d304", width =
7,command=self.func_auto)
self.btnExitAuto.place(x=0, y=700)
self.btnExitSetsokinh = tkinter.Button(self.root, text="Set", font=("Times New
Roman", 17, "bold", "underline"),
fg="#2207a8", bg="#d6d304", width=7,
command=self.func_slSetsokinh)
self.btnExitSetsokinh.place(x=350, y=700)
self.slz = tkinter.Scale(self.root, from_=0, to=180, length=200, resolution=1,
orient=tkinter.HORIZONTAL)

```

```

    self.slz.pack()
    self.slz.place(x=80, y=370)
    self.slx = tkinter.Scale(self.root, from_=0, to=300, length=200, resolution=1,
orient=tkinter.HORIZONTAL)
        self.slx.pack()
        self.slx.place(x=80, y=407)
    self.sl11 = tkinter.Scale(self.root, from_=255, to=0, length=200, resolution=1,
orient=tkinter.VERTICAL)
        self.sl11.pack()
        self.sl11.place(x=1400, y=220)
    self.sl12 = tkinter.Scale(self.root, from_=50, to=0, length=200, resolution=1,
orient=tkinter.VERTICAL)
        self.sl12.pack()
        self.sl12.place(x=1470, y=220)
    self.slhop_thresh = tkinter.Scale(self.root, from_=255, to=0, length=200,
resolution=1, orient=tkinter.VERTICAL)
        self.slhop_thresh.pack()
        self.slhop_thresh.place(x=1400, y=530)
    self.slhop_open = tkinter.Scale(self.root, from_=50, to=0, length=200,
resolution=1,
                                orient=tkinter.VERTICAL)
        self.slhop_open.pack()
        self.slhop_open.place(x=1470, y=530)
    self.sl_sokinh = tkinter.Scale(self.root, from_=0, to=20, length=100,
resolution=1,
                                orient=tkinter.HORIZONTAL)
        self.sl_sokinh.pack()
        self.sl_sokinh.place(x=220, y=700)
    self.lb1=tkinter.Label(self.root, text="Conveyor1:", font=("Times New
Roman", 15, "bold"), fg="#2207a8",
                                bg="#2af0fa").place(x=0, y=630)
    bard5 = Image.open("left1.png")
    img5 = ImageTk.PhotoImage(bard5)
    self.btnC11 = tkinter.Button(self.root, image=img5, width=55,
height=30,command=self.func_butC11).place(x=120, y=625)
    bard6 = Image.open("right1.png")
    img6 = ImageTk.PhotoImage(bard6)

```

```

    self.btnC1r = tkinter.Button(self.root, image=img6, width=55,
height=30,command=self.func_butC1r).place(x=182, y=625)

    self.lb2 = tkinter.Label(self.root, text="Conveyor2:", font=("Times New
Roman", 15, "bold"), fg="#2207a8",
                           bg="#2af0fa").place(x=260, y=630)
    self.lb3 = tkinter.Label(self.root, text="Phát hiện sản phẩm", font=("Times
New Roman", 15, "bold"), fg="dark khaki",
                           bg="slate gray").place(x=1020, y=186)
    self.lb3 = tkinter.Label(self.root, text="Phát hiện thùng", font=("Times New
Roman", 15, "bold"),
                           fg="dark khaki",
                           bg="slate gray").place(x=1020, y=500)
    self.btnC2l = tkinter.Button(self.root, image=img5, width=55,
height=30,command=self.func_butC2l).place(x=380, y=625)
    self.btnC2r = tkinter.Button(self.root, image=img6, width=55,
height=30,command=self.func_butC2r).place(x=442, y=625)

    self.key = cv2.waitKey(1)
    self.camera = cv2.VideoCapture(1)
    self.camera1()
    self.camera2()
    self.likecamera()
    self.root.mainloop()

def func_slSetsokinh(self):
    a=self.sl_sokinh.get()
    choui=tao_chuoi_gui("setsokinh",a)
    ser.write(choui.encode())

def func_start(self):
    choui=tao_chuoi_gui("start",None)
    ser.write(choui.encode())

def func_stop(self):
    choui=tao_chuoi_gui("stop",None)
    ser.write(choui.encode())
    global chedo_auto, chedo_manual
    chedo_manual = 0
    chedo_auto = 0

def func_manual(self):

```

```

chuoi=tao_chuoi_gui("manual",None)
ser.write(chuoi.encode())
global chedo_auto,chedo_manual
chedo_manual=1
chedo_auto = 0

def func_auto(self):
    chuoi=tao_chuoi_gui("auto",None)
    ser.write(chuoi.encode())
    global chedo_auto, chedo_manual
    chedo_manual = 0
    chedo_auto = 1

def func_home(self):
    chuoi=tao_chuoi_gui("home",None)
    ser.write(chuoi.encode())
    self.slx.set(0)
    self.slz.set(0)

def func_goz(self):
    giatrix=self.slz.get()
    chuoi=tao_chuoi_gui("vitriz",giatrix)
    ser.write(chuoi.encode())

def func_gox(self):
    giatrix=self.slx.get()
    chuoi=tao_chuoi_gui("vitrix",giatrix)
    ser.write(chuoi.encode())

def func_pickup(self):
    chuoi=tao_chuoi_gui("vanhut",1)
    ser.write(chuoi.encode())

def func_release(self):
    chuoi=tao_chuoi_gui("vanhut",0)
    ser.write(chuoi.encode())

def func_butC1l(self):
    global chedo_manual,manualbc11
    if(chedo_manual==1):
        manualbc11=1-manualbc11
        if(manualbc11==1):
            chuoi = tao_chuoi_gui("manualbc1", 1)
            ser.write(chuoi.encode())

```

```

if (manualbc1l == 0):
    chuoi = tao_chuoi_gui("manualbc1", 0)
    ser.write(chuoi.encode())

def func_butC1r(self):
    global chedo_manual,manualbc1r
    if(chedo_manual==1):
        manualbc1r=1-manualbc1r
    if(manualbc1r==1):
        chuoi = tao_chuoi_gui("manualbc1", 2)
        ser.write(chuoi.encode())
    if (manualbc1r == 0):
        chuoi = tao_chuoi_gui("manualbc1", 0)
        ser.write(chuoi.encode())

def func_butC2l(self):
    global chedo_manual,manualbc2l
    if(chedo_manual==1):
        manualbc2l=1-manualbc2l
    if(manualbc2l==1):
        chuoi = tao_chuoi_gui("manualbc2", 1)
        ser.write(chuoi.encode())
    if (manualbc2l == 0):
        chuoi = tao_chuoi_gui("manualbc2", 0)
        ser.write(chuoi.encode())

def func_butC2r(self):
    global chedo_manual,manualbc2r
    if(chedo_manual==1):
        manualbc2r=1-manualbc2r
    if(manualbc2r==1):
        chuoi = tao_chuoi_gui("manualbc2", 2)
        ser.write(chuoi.encode())
    if (manualbc2r == 0):
        chuoi = tao_chuoi_gui("manualbc2", 0)
        ser.write(chuoi.encode())

def lientuczlen(self):
    global cho_phep_di_chuyen_lien_tuc_manual
    while (cho_phep_di_chuyen_lien_tuc_manual==1):
        giatriz = self.slz.get()

```

```

giatriz = giatriz + 1
self.slz.set(giatriz)
chuoi = tao_chuoi_gui("vitriz", giatriz)
ser.write(chuoi.encode())
time.sleep(1)
if(self.btnUp['relief']==tkinter.RAISED):
    cho_phep_di_chuyen_lien_tuc_manual=0
def func_btnUp(self):
    giatriz=self.slz.get()
    giatriz=giatriz+1
    self.slz.set(giatriz)
    chuoi=tao_chuoi_gui("vitriz",giatriz)
    ser.write(chuoi.encode())
def func_btnDwn(self):
    giatriz=self.slz.get()
    giatriz=giatriz-1
    self.slz.set(giatriz)
    chuoi=tao_chuoi_gui("vitriz",giatriz)
    ser.write(chuoi.encode())
def func_btnLeft(self):
    giatrix=self.slx.get()
    giatrix=giatrix-1
    self.slx.set(giatrix)
    chuoi=tao_chuoi_gui("vitrix",giatrix)
    ser.write(chuoi.encode())
def func_btnRight(self):
    giatrix=self.slx.get()
    giatrix=giatrix+1
    self.slx.set(giatrix)
    chuoi=tao_chuoi_gui("vitrix",giatrix)
    ser.write(chuoi.encode())
def func_btn11(self):
    global setup
    setup=1
def func_btn12(self):
    global setup
    setup = 0

```

```

def func_btn13(self):
    global enable_chon_vung_xla
    enable_chon_vung_xla=1
def scales(self,frame,scaleeee):
    x=int(frame.width*scaleeee)
    y=int(frame.height*scaleeee)
    frame2 = frame.resize((x, y), Image.ANTIALIAS)
    return frame2
def click_chon_vung_xu_ly_anh(self,e):
    global enable_chon_vung_xla,count_chon_vung_xla
    global diem_vung_xla
    global dcx, dcy, scale
    if(enable_chon_vung_xla==1):
        if(count_chon_vung_xla<8):
            diem_vung_xla[count_chon_vung_xla]=(int(e.x / scale),int(e.y / scale))
            print("left "+str(count_chon_vung_xla)+" "+
str(diem_vung_xla[count_chon_vung_xla]))
            count_chon_vung_xla += 1
        if(count_chon_vung_xla==8):
            enable_chon_vung_xla=0
            count_chon_vung_xla=0
def likecamera(self):
    global ser,gui_tin_hieu_co_sp_moi_xuong_vxl,vitrisp,vitrihop
    self.panel3.after(15,self.likecamera)
def camera1(self):
    global diem_vung_xla
    global co_sp_moi,da_gap_sp
    global chedo_auto,chedo_manual
    _, frame = self.camera.read()
    g = frame.copy()
    point1=diem_vung_xla[0]
    point2=diem_vung_xla[1]
    point3=diem_vung_xla[2]
    point4=diem_vung_xla[3]
    global xchuan,ychuan
    pst1 = np.float32([point1, point2, point3, point4])
    pst2 = np.float32([[0, 0], [xchuan, 0], [0, ychuan], [xchuan, ychuan]])

```

```

vung_xla = cv2.getPerspectiveTransform(pst1, pst2)
vung_xla_scaled = cv2.warpPerspective(frame, vung_xla, (xchuan, ychuan))
frame=vung_xla_scaled
thpoint = self.sl11.get()
bk = 4
bk2 = self.sl12.get()
kernel = dishes(bk)
kernel2 = dishes(bk2)
imggray= cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
_, imgthreshold = cv2.threshold(imggray, thpoint, 255,
cv2.THRESH_BINARY)
closing = cv2.morphologyEx(imgthreshold, cv2.MORPH_CLOSE, kernel)
openimg = cv2.morphologyEx(closing, cv2.MORPH_OPEN, kernel)
# for n in range(10):
closing = cv2.morphologyEx(openimg, cv2.MORPH_CLOSE, kernel2)
openimg = cv2.morphologyEx(closing, cv2.MORPH_OPEN, kernel2)
edgedetect = cv2.Canny(openimg, thpoint, 150, apertureSize=3)
g[:, :, 0] = 0
g[:, :, 2] = 0
frame_hop = g
point5 = diem_vung_xla[4]
point6 = diem_vung_xla[5]
point7 = diem_vung_xla[6]
point8 = diem_vung_xla[7]
global xchuan_hop, ychuan_hop
pst1_hop = np.float32([point5, point6, point7, point8])
pst2_hop = np.float32([[0, 0], [xchuan_hop, 0], [0, ychuan_hop], [xchuan_hop,
ychuan_hop]])
vung_xla_hop = cv2.getPerspectiveTransform(pst1_hop, pst2_hop)
vung_xla_scaled_hop = cv2.warpPerspective(frame_hop, vung_xla_hop,
(xchuan_hop, ychuan_hop))
frame_hop = vung_xla_scaled_hop
thpoint_hop = self.slhop_thresh.get()
bk2_hop = self.slhop_open.get()
kernel2_hop = dishes(bk2_hop)
imggray_hop = cv2.cvtColor(frame_hop, cv2.COLOR_BGR2GRAY)
_, imgthreshold_hop = cv2.threshold(imggray_hop, thpoint_hop, 255,

```

```

cv2.THRESH_BINARY)
closing_hop = cv2.morphologyEx(imgthreshhold_hop, cv2.MORPH_CLOSE,
kernel2_hop)
openimg_hop = cv2.morphologyEx(closing_hop, cv2.MORPH_OPEN,
kernel2_hop)
edgedetect_hop = cv2.Canny(openimg_hop, thpoint_hop, 150, apertureSize=3)
minnn = self.slz.get()
maxgap = self.slx.get()
lines = cv2.HoughLinesP(edgedetect, 1, np.pi / 180, 50,
minLineLength=minnn, maxLineGap=maxgap)
dachuyen = cv2.cvtColor(edgedetect, cv2.COLOR_GRAY2RGB)
countline = 0
if (str(lines) != "None"):
    for bb in lines:
        countline += 1
        minnn=10
        maxgap=50
        lines_hop = cv2.HoughLinesP(edgedetect_hop, 1, np.pi / 180, 50,
minLineLength=minnn, maxLineGap=maxgap)
        dachuyen_hop = cv2.cvtColor(edgedetect_hop, cv2.COLOR_GRAY2RGB)
        countline_hop = 0
        if (str(lines_hop) != "None"):
            for bb in lines_hop:
                countline_hop = countline_hop + 1
                global co_hop_moi
                center_canh_duoil_hop=(0,0)
                if (str(lines_hop) != "None"):
                    for line_hop in lines_hop:
                        x1, y1, x2, y2 = line_hop[0]
                        cv2.line(dachuyen_hop, (x1, y1), (x2, y2), (255, 0, 0), 3)
                        countcanhngang_hop=0
                        if(x2!=x1):
                            if(abs((y2-y1)/(x2-x1))<0.2):
                                countcanhngang_hop+=1
                                center_canh_duoil_hop=(int((x2+x1)/2),int((y2+y1)/2))
                                cv2.circle(dachuyen_hop, center_canh_duoil_hop, 5, (0, 0, 255),
thickness=3)

```

```

if (co_hop_moi==0) :
    if center_canh_duo_hop[1]>(ychuan_hop*2/3) and
center_canh_duo_hop[1]<(ychuan_hop*2/3+10):
        co_hop_moi=1
        goc_bang_chuyen2 = int((145+97/3+105/2) * 800 / 261)
        if(chedo_auto==1):
            chuo_hop = tao_chuo_gui("tudonggapbc2", goc_bang_chuyen2)
            ser.write(chuo_hop.encode())
        iv = np.invert(openimg)
        cnts = cv2.findContours(iv.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
        cnts = imutils.grab_contours(cnts)
        count_contour = 0
        for a in cnts:
            count_contour = count_contour + 1
            global gui_tin_hieu_co_sp_moi_xuong_vxl
            global vitrisp,vitrihop
            global ktx_chuan,kyt_chuan
            if (count_contour != 0):
                if (bk2 >= 1):
                    for c in cnts:
                        M = cv2.moments(c)
                        if (M["m00"] != 0):
                            cX = int(M["m10"] / M["m00"])
                            cY = int(M["m01"] / M["m00"])
                            cv2.drawContours(openimg, [c], -1, (0, 255, 0), 2)
                            cv2.circle(openimg, (cX, cY), 7, (255, 255, 255), -1)
                            if(co_sp_moi==0):
                                if (count_contour == 1):
                                    if (cY >=int(ychuan/2)):
                                        co_sp_moi=1
                                        gui_tin_hieu_co_sp_moi_xuong_vxl=1
                                        goc_bang_chuyen = int(191 * 500 / 160)
                                        vitrisp=int((xchuan-cX)*ktx_chuan/xchuan)-55
                                        if(vitrisp<0):
                                            messagebox.showinfo("Cảnh báo","Vị trí kính nhỏ hơn 0")

```

```

chuoi=tao_chuoi_gui4("stop",goc_bang_chuyen,vitrisp,vitrihop,)
    ser.write(chuoi.encode())
    if (chedo_auto == 1):

chuoi=tao_chuoi_gui4("tudonggap",goc_bang_chuyen,vitrisp,vitrihop,)
    ser.write(chuoi.encode())
elif (count_contour == 0):
    co_sp_moi=0
    iv_hop = np.invert(openimg_hop)
    cnts_hop = cv2.findContours(iv_hop.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    cnts_hop = imutils.grab_contours(cnts_hop)
    count_contour_hop = 0
    for a in cnts_hop:
        count_contour_hop = count_contour_hop + 1
    global ktx_chuan_hop, kty_chuan_hop
    if (count_contour_hop == 0):
        co_hop_moi = 0
        frame=openimg
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        frame = Image.fromarray(frame)
        frame=self.scales(frame,1)
        frame = ImageTk.PhotoImage(frame)
        self.panel.configure(image=frame)
        self.panel.image = frame
        self.panel.after(30, self.camera1)
        self.panel.place(x=1020, y=216)
        frame4 = cv2.cvtColor(dachuyen_hop, cv2.COLOR_BGR2RGB)
        frame4 = Image.fromarray(frame4)
        frame4 = self.scales(frame4, 1)
        frame4 = ImageTk.PhotoImage(frame4)
        self.panel4.configure(image=frame4)
        self.panel4.image = frame4
        self.panel4.place(x=1020, y=530)
def camera2(self):
    global setup
    global dcx,dcy

```

```

global scale
global diem_vung_xla
scale=0.7
_, frame2 = self.camera.read()
if (setup==1):
    for n in range(8):
        if(n<4):
            cv2.circle(frame2, diem_vung_xla[n], radius=5,color= (0, 0,
255),thickness=-1)
        else:
            cv2.circle(frame2, diem_vung_xla[n], radius=5, color=(255, 0, 0),
thickness=-1)
frame3 = cv2.cvtColor(frame2, cv2.COLOR_BGR2RGB)
frame3 = Image.fromarray(frame3)
frame3=sself.scales(frame3,scale)
frame3 = ImageTk.PhotoImage(frame3)
self.panel2.configure(image=frame3)
self.panel2.image = frame3
self.panel2.bind('<Button-1>', self.click_chon_vung_xu_ly_anh)
self.panel2.after(30, self.camera2)
self.panel2.place(x=520, y=350)
if __name__ == '__main__':
    objVideo = videoStream()

```

2. Chương trình vi xử lý.

```

#include "main.h"
#include "cmsis_os.h"
#include "string.h"
#include "stdlib.h"
#include "stdio.h"
#include "dwt_delay.h"
I2C_HandleTypeDef hi2c1;
I2C_HandleTypeDef hi2c3;
I2S_HandleTypeDef hi2s3;
SPI_HandleTypeDef hspi1;
TIM_HandleTypeDef htim9;
UART_HandleTypeDef huart3;
osThreadId_t defaultTaskHandle;

```

```

const osThreadAttr_t defaultTask_attributes = {
    .name = "defaultTask",
    .priority = (osPriority_t) osPriorityNormal,
    .stack_size = 128 * 4
};

osThreadId_t myTask02Handle;
const osThreadAttr_t myTask02_attributes = {
    .name = "myTask02",
    .priority = (osPriority_t) osPriorityNormal,
    .stack_size = 128 * 4
};

osThreadId_t myTask03Handle;
const osThreadAttr_t myTask03_attributes = {
    .name = "myTask03",
    .priority = (osPriority_t) osPriorityNormal,
    .stack_size = 128 * 4
};

osThreadId_t myTask04Handle;
const osThreadAttr_t myTask04_attributes = {
    .name = "myTask04",
    .priority = (osPriority_t) osPriorityNormal,
    .stack_size = 128 * 4
};

osThreadId_t myTask05Handle;
const osThreadAttr_t myTask05_attributes = {
    .name = "myTask05",
    .priority = (osPriority_t) osPriorityNormal,
    .stack_size = 128 * 4
};

osThreadId_t myTask06Handle;
const osThreadAttr_t myTask06_attributes = {
    .name = "myTask06",
    .priority = (osPriority_t) osPriorityNormal,
    .stack_size = 128 * 4
};

osThreadId_t myTask07Handle;
const osThreadAttr_t myTask07_attributes = {
    .name = "myTask07",

```

```

.priority = (osPriority_t) osPriorityNormal,
.stack_size = 128 * 4
};

osThreadId_t myTask08Handle;
const osThreadAttr_t myTask08_attributes = {
    .name = "myTask08",
    .priority = (osPriority_t) osPriorityNormal,
    .stack_size = 128 * 4
};

//Khai bao bien
int dcx=1;
int dcz1=0;
int dcz2=5;
int dchut=2;
int dcbc1=3;
int dcbc2=4;

//auto

uint8_t sokinh_hop=0;
uint8_t sokinh_tong=0;
uint8_t sohop=0;
uint8_t chieu_cao_hop=40;
uint8_t do_day_day_hop=4;
uint8_t do_day_kinh=3;
uint8_t set_sokinh_hop=3;
uint8_t co_hop=0;
int64_t vitrisp=0;
int64_t vitrihop=265;
int64_t vitriz_hop=0;
int64_t vitriz_sp=0;
uint8_t vitrisp_chuoi[20]="";
uint8_t vitrihop_chuoi[20]="";
uint8_t tudonggap=0;
uint8_t tudonggapbc2=0;
uint8_t ten_chuoi[20]++;
uint8_t giatri[20]++;
uint8_t send_buf[60]++;
uint8_t receive_buf[60]++;
//start/stop
uint8_t Enable_system=0;
uint8_t nut_start=0;

```

```

uint8_t nut_stop=0;
//Che do dieu khien
uint8_t che_do_auto=0;
uint8_t che_do_manual=0;
//TRUCZ
uint8_t homez1=0;
uint8_t homez2=0;
int64_t vitriz=0;
int64_t vitriz_tam=0;
uint16_t Z1=0;
uint16_t Z2=0;
uint8_t endstop_z1;
uint8_t endstop_z2;
//TRUCX
int64_t vitrix=0;
int64_t vitrix_tam=0;
uint16_t X=0;
uint8_t endstop_x;
//BANGCHUYEN;
uint8_t enable_dk_vitri_bc1=0;
uint8_t enable_dk_vitri_bc2=0;
int64_t vitribc1=0;
int64_t vitribc2=0;
int64_t vitribc1_tam=0;
int64_t vitribc2_tam=0;
uint8_t enable_bc1=0;
uint8_t enable_bc2=0;
//manualbc2;
int64_t manual_bc1=0;
int64_t manual_bc2=0;
//manual vanhut
int64_t vanhut=0;
// cac ham khoi tao cua phan mem CubeMX
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_I2C1_Init(void);
static void MX_I2S3_Init(void);
static void MX_SPI1_Init(void);
static void MX_I2C3_Init(void);
static void MX_TIM9_Init(void);
static void MX_USART3_UART_Init(void);
void StartDefaultTask(void *argument);
void TRUCZ1(void *argument);
void TRUCZ2(void *argument);

```

```

void TRUCX(void *argument);
void DAUHUT(void *argument);
void BANGCHUYEN1(void *argument);
void BANGCHUYEN2(void *argument);
void KT NGO VAO(void *argument);
/* USER CODE BEGIN PFP */

//void dkdc(int dc, int64_t goc, int delay,int modstep);
//void delayms(uint32_t dl);

int main(void)
{
    //khai bao phan cung
    HAL_Init();
    SystemClock_Config();
    MX_GPIO_Init();
    MX_I2C1_Init();
    MX_I2S3_Init();
    MX_SPI1_Init();
    MX_I2C3_Init();
    MX_TIM9_Init();
    MX_USART3_UART_Init();

    HAL_UART_Receive_IT(&huart3,receive_buf,60);
    HAL_GPIO_WritePin(MODSTEP1_GPIO_Port,MODSTEP1_Pin,1);
    HAL_GPIO_WritePin(MODSTEP2_GPIO_Port,MODSTEP2_Pin,1);
    HAL_GPIO_WritePin(MODSTEP3_GPIO_Port,MODSTEP3_Pin,1);
    HAL_GPIO_WritePin(MODSTEP4_GPIO_Port,MODSTEP4_Pin,1);
    HAL_GPIO_WritePin(MODSTEP5_GPIO_Port,MODSTEP5_Pin,1);
    HAL_GPIO_WritePin(MODSTEP6_GPIO_Port,MODSTEP6_Pin,1);
    HAL_GPIO_WritePin(Relay2_GPIO_Port,Relay2_Pin,1);
    //Ham khai tao chay da nham cua Cubemx
    osKernelInitialize();
    /* Create the thread(s) */
    defaultTaskHandle = osThreadNew(StartDefaultTask, NULL, &defaultTask_attributes);
    myTask02Handle = osThreadNew(TRUCZ1, NULL, &myTask02_attributes);
    myTask03Handle = osThreadNew(TRUCZ2, NULL, &myTask03_attributes);
    myTask04Handle = osThreadNew(TRUCX, NULL, &myTask04_attributes);
    myTask05Handle = osThreadNew(DAUHUT, NULL, &myTask05_attributes);
    myTask06Handle = osThreadNew(BANGCHUYEN1, NULL, &myTask06_attributes);
    myTask07Handle = osThreadNew(BANGCHUYEN2, NULL, &myTask07_attributes);
    myTask08Handle = osThreadNew(KT NGO VAO, NULL, &myTask08_attributes);
    osKernelStart();
}

```

```

while (1)
{
    //Chuong chay trong cac task rieng
}
}

void SystemClock_Config(void)
{
    RCC_OscInitTypeDef RCC_OscInitStruct = {0};
    RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
    RCC_PeriphCLKInitTypeDef PeriphClkInitStruct = {0};
    __HAL_RCC_PWR_CLK_ENABLE();
    __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
    RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
    RCC_OscInitStruct.HSEState = RCC_HSE_ON;
    RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
    RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
    RCC_OscInitStruct.PLL.PLLM = 4;
    RCC_OscInitStruct.PLL.PLLN = 168;
    RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
    RCC_OscInitStruct.PLL.PLLQ = 7;
    if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
    {
        Error_Handler();
    }
    RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                                |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
    RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
    RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
    RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
    RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV4;

    if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK)
    {
        Error_Handler();
    }
    PeriphClkInitStruct.PeriphClockSelection = RCC_PERIPHCLK_I2S;
    PeriphClkInitStruct.PLLI2S.PLLI2SN = 50;
    PeriphClkInitStruct.PLLI2S.PLLI2SR = 2;
    if (HAL_RCCE_PeriphCLKConfig(&PeriphClkInitStruct) != HAL_OK)
    {
        Error_Handler();
    }
}

static void MX_I2C1_Init(void)

```

```

{
    hi2c1.Instance = I2C1;
    hi2c1.Init.ClockSpeed = 100000;
    hi2c1.Init.DutyCycle = I2C_DUTYCYCLE_2;
    hi2c1.Init.OwnAddress1 = 0;
    hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
    hi2c1.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
    hi2c1.Init.OwnAddress2 = 0;
    hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
    hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
    if (HAL_I2C_Init(&hi2c1) != HAL_OK)
    {
        Error_Handler();
    }
}

static void MX_I2C3_Init(void)
{
    hi2c3.Instance = I2C3;
    hi2c3.Init.ClockSpeed = 100000;
    hi2c3.Init.DutyCycle = I2C_DUTYCYCLE_2;
    hi2c3.Init.OwnAddress1 = 0;
    hi2c3.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
    hi2c3.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
    hi2c3.Init.OwnAddress2 = 0;
    hi2c3.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
    hi2c3.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
    if (HAL_I2C_Init(&hi2c3) != HAL_OK)
    {
        Error_Handler();
    }
}

static void MX_I2S3_Init(void)
{
    hi2s3.Instance = SPI3;
    hi2s3.Init.Mode = I2S_MODE_MASTER_TX;
    hi2s3.Init.Standard = I2S_STANDARD_PHILIPS;
    hi2s3.Init.DataFormat = I2S_DATAFORMAT_16B;
    hi2s3.Init.MCLKOutput = I2S_MCLKOUTPUT_ENABLE;
    hi2s3.Init.AudioFreq = I2S_AUDIOFREQ_48K;
    hi2s3.Init.CPOL = I2S_CPOL_LOW;
    hi2s3.Init.ClockSource = I2S_CLOCK_PLL;
    hi2s3.Init.FullDuplexMode = I2S_FULLDUPLEXMODE_DISABLE;
    if (HAL_I2S_Init(&hi2s3) != HAL_OK)

```

```

{
    Error_Handler();
}
}

static void MX_SPI1_Init(void)
{
    hspi1.Instance = SPI1;
    hspi1.Init.Mode = SPI_MODE_MASTER;
    hspi1.Init.Direction = SPI_DIRECTION_2LINES;
    hspi1.Init.DataSize = SPI_DATASIZE_8BIT;
    hspi1.Init.CLKPolarity = SPI_POLARITY_LOW;
    hspi1.Init.CLKPhase = SPI_PHASE_1EDGE;
    hspi1.Init.NSS = SPI_NSS_SOFT;
    hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_2;
    hspi1.Init.FirstBit = SPI_FIRSTBIT_MSB;
    hspi1.Init.TIMode = SPI_TIMODE_DISABLE;
    hspi1.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
    hspi1.Init.CRCPolynomial = 10;
    if (HAL_SPI_Init(&hspi1) != HAL_OK)
    {
        Error_Handler();
    }
}

static void MX_TIM9_Init(void)
{
    TIM_ClockConfigTypeDef sClockSourceConfig = {0};
    htim9.Instance = TIM9;
    htim9.Init.Prescaler = 2;
    htim9.Init.CounterMode = TIM_COUNTERMODE_UP;
    htim9.Init.Period = 41;
    htim9.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
    htim9.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
    if (HAL_TIM_Base_Init(&htim9) != HAL_OK)
    {
        Error_Handler();
    }
    sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
    if (HAL_TIM_ConfigClockSource(&htim9, &sClockSourceConfig) != HAL_OK)
    {
        Error_Handler();
    }
}

static void MX_USART3_UART_Init(void)

```

```

{
    huart3.Instance = USART3;
    huart3.Init.BaudRate = 115200;
    huart3.Init.WordLength = UART_WORDLENGTH_8B;
    huart3.Init.StopBits = UART_STOPBITS_1;
    huart3.Init.Parity = UART_PARITY_NONE;
    huart3.Init.Mode = UART_MODE_TX_RX;
    huart3.Init.HwFlowCtl = UART_HWCONTROL_NONE;
    huart3.Init.OverSampling = UART_OVERSAMPLING_16;
    if (HAL_UART_Init(&huart3) != HAL_OK)
    {
        Error_Handler();
    }
}

static void MX_GPIO_Init(void)
{
    GPIO_InitTypeDef GPIO_InitStruct = {0};
    __HAL_RCC_GPIOE_CLK_ENABLE();
    __HAL_RCC_GPIOC_CLK_ENABLE();
    __HAL_RCC_GPIOH_CLK_ENABLE();
    __HAL_RCC_GPIOA_CLK_ENABLE();
    __HAL_RCC_GPIOB_CLK_ENABLE();
    __HAL_RCC_GPIOD_CLK_ENABLE();
    HAL_GPIO_WritePin(GPIOE, CS_I2C_SPI_Pin|MODSTEP4_Pin|DIR2_Pin|STEP2_Pin
                    |MODSTEP2_Pin|EN2_Pin|DIR3_Pin, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOC, EN4_Pin|MODSTEP1_Pin|MODSTEP6_Pin|STEP6_Pin
                    |DIR5_Pin, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(OTG_FS_PowerSwitchOn_GPIO_Port, OTG_FS_PowerSwitchOn_Pin,
                     GPIO_PIN_SET);
    HAL_GPIO_WritePin(GPIOA, DIR1_Pin|STEP1_Pin|EN6_Pin, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOB, EN1_Pin|STEP3_Pin|MODSTEP3_Pin|EN3_Pin
                    |DIR4_Pin|STEP4_Pin, GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOD, Relay1_Pin|Relay2_Pin|GPIO_PIN_12|GPIO_PIN_13
                    |GPIO_PIN_14|GPIO_PIN_15|STEP5_Pin|DIR6_Pin
                    |MODSTEP5_Pin|Audio_RST_Pin|EN5_Pin|ENDSTOP5_Pin,
                     GPIO_PIN_RESET);
    GPIO_InitStruct.Pin = CS_I2C_SPI_Pin|MODSTEP4_Pin|DIR2_Pin|STEP2_Pin
                        |MODSTEP2_Pin|EN2_Pin|DIR3_Pin;
    GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
    HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);
    GPIO_InitStruct.Pin =
EN4_Pin|OTG_FS_PowerSwitchOn_Pin|MODSTEP1_Pin|MODSTEP6_Pin

```

```

|STEP6_Pin|DIR5_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
GPIO_InitStruct.Pin = PDM_OUT_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
GPIO_InitStruct.Alternate = GPIO_AF5_SPI2;
HAL_GPIO_Init(PDM_OUT_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = B1_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_EVT_RISING;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(B1_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = DIR1_Pin|STEP1_Pin|EN6_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
GPIO_InitStruct.Pin = EN1_Pin|STEP3_Pin|MODSTEP3_Pin|EN3_Pin
|DIR4_Pin|STEP4_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
GPIO_InitStruct.Pin = BOOT1_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(BOOT1_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = ENDSTOP1_Pin|ENDSTOP2_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_PULLDOWN;
HAL_GPIO_Init(GPIOE, &GPIO_InitStruct);
GPIO_InitStruct.Pin = CLK_IN_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
GPIO_InitStruct.Alternate = GPIO_AF5_SPI2;
HAL_GPIO_Init(CLK_IN_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = Relay1_Pin|GPIO_PIN_12|GPIO_PIN_13|GPIO_PIN_14
|GPIO_PIN_15|STEP5_Pin|DIR6_Pin|MODSTEP5_Pin
|Audio_RST_Pin|EN5_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;

```

```

GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(GPIOD, &GPIO_InitStruct);
GPIO_InitStruct.Pin = Relay2_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_PULLUP;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(Relay2_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = VBUS_FS_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(VBUS_FS_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = OTG_FS_ID_Pin|OTG_FS_DM_Pin|OTG_FS_DP_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
GPIO_InitStruct.Alternate = GPIO_AF10_OTG_FS;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
GPIO_InitStruct.Pin = OTG_FS_OverCurrent_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(OTG_FS_OverCurrent_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = ENDSTOP5_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
GPIO_InitStruct.Pull = GPIO_PULLDOWN;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
HAL_GPIO_Init(ENDSTOP5_GPIO_Port, &GPIO_InitStruct);
GPIO_InitStruct.Pin = ENDSTOP4_Pin|ENDSTOP3_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
GPIO_InitStruct.Pull = GPIO_PULLDOWN;
HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
GPIO_InitStruct.Pin = MEMS_INT2_Pin;
GPIO_InitStruct.Mode = GPIO_MODE_EVT_RISING;
GPIO_InitStruct.Pull = GPIO_NOPULL;
HAL_GPIO_Init(MEMS_INT2_GPIO_Port, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */
//ham dieu khien dong co
void dkdc(int dc, int64_t goc, int delayus,int modstep)
{
    uint32_t xung;
    switch (dc)

```

```

{
    case 1: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_1,0);//ENABLE STEP1
        break;
    case 2: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_13,0);//ENABLE STEP2
        break;
    case 3: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_15,0);//ENABLE STEP3
        break;
    case 4: HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,0);//ENABLE STEP4
        break;
    case 5: HAL_GPIO_WritePin(GPIOD,GPIO_PIN_6,0);//ENABLE STEP5
        break;
    case 6: HAL_GPIO_WritePin(EN6_GPIO_Port,EN6_Pin,0);//ENABLE STEP6
        break;
}
if ( goc > 0 )
{
    switch (dc)
    {
        case 1: HAL_GPIO_WritePin(GPIOA,GPIO_PIN_1,1);//DIR1
        break;
        case 2: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_7,1);//DIR2
        break;
        case 3: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_15,1);//DIR3
        break;
        case 4: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_5,1);//DIR4
        break;
        case 5: HAL_GPIO_WritePin(GPIOC,GPIO_PIN_11,1);//DIR5
        break;
        case 6: HAL_GPIO_WritePin(DIR6_GPIO_Port,DIR6_Pin,1);//DIR6
        break;
    }
}
else
{
    switch (dc)
    {
        case 1: HAL_GPIO_WritePin(GPIOA,GPIO_PIN_1,0);//DIR1
        break;
        case 2: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_7,0);//DIR2
        break;
        case 3: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_15,0);//DIR3
        break;
        case 4: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_5,0);//DIR4
        break;
    }
}

```

```

        case 5: HAL_GPIO_WritePin(GPIOC,GPIO_PIN_11,0);//DIR5
            break;
        case 6: HAL_GPIO_WritePin(DIR6_GPIO_Port,DIR6_Pin,0);//DIR6
            break;
    }
}

if (goc < 0) goc=-goc;
xung= 200*(goc)*modstep/360;

for (int i=0;i<xung;i++)
{
    if(Enable_system==0) break;
    switch (dc)
    {
        case 1: HAL_GPIO_WritePin(GPIOA,GPIO_PIN_3,1);//1
            break;
        case 2: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_9,1);//2
            break;
        case 3: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_11,1);//3
            break;
        case 4: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_7,1);//4
            break;
        case 5: HAL_GPIO_WritePin(GPIOD,GPIO_PIN_0,1);//5
            break;
        case 6: HAL_GPIO_WritePin(STEP6_GPIO_Port,STEP6_Pin,1);//6
            break;
    }
    osDelay(delayus);
    switch (dc)
    {
        case 1: HAL_GPIO_WritePin(GPIOA,GPIO_PIN_3,0);//1
            break;
        case 2: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_9,0);//2
            break;
        case 3: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_11,0);//3
            break;
        case 4: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_7,0)//4
            break;
        case 5: HAL_GPIO_WritePin(GPIOD,GPIO_PIN_0,0)//5
            break;
        case 6: HAL_GPIO_WritePin(STEP6_GPIO_Port,STEP6_Pin,0)//6
            break;
    }
}

```

```

switch (dc)
{
    case 1: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_1,0);//ENABLE STEP1
        break;
    case 2: HAL_GPIO_WritePin(GPIOE,GPIO_PIN_13,0);//ENABLE STEP2
        break;
    case 3: HAL_GPIO_WritePin(GPIOB,GPIO_PIN_15,1);//ENABLE STEP3
        break;
    case 4: HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,1);//ENABLE STEP4
        break;
    case 5: HAL_GPIO_WritePin(GPIOD,GPIO_PIN_6,1);//ENABLE STEP5
        break;
    case 6: HAL_GPIO_WritePin(EN6_GPIO_Port,EN6_Pin,1);//ENABLE STEP6
        break;
}

//ham so sanh chuoi uint8
uint8_t so_sanh_chuoi(uint8_t* chuoi1,char* chuoi2)
{
    if(strcmp((char*)chuoi1,chuoi2)==0)
        return 1;
    else return 0;
}

//copy chuoi uint8
void copybetween(uint8_t* srcchuoi,uint8_t* deschuoi,char kitu1,char kitu2)
{
    int s1,s2;
    for(int m=0;m<strlen((char*)srcchuoi);m++)
    {
        if(srcchuoi[m]==(uint8_t)kitu1)
        {
            s1=m;
            s2=m+1;
        }
        else if(srcchuoi[m]==(uint8_t)kitu2)
        {
            s2=m;
            break;
        }
    }
    for(int x=0;x<20;x++)
    {

```

```

        deschuoi[x]=0;
    }
    if((s2-s1)>1)
    {
        for(int n=0;n<(s2-s1-1);n++)
        {
            deschuoi[n]=srcchuoi[s1+n+1];
        }
    }

}

//ham xu ly tin hieu nhan tu may tinh de thuc thi tac vu
void xulychuoinhan(uint8_t* chuo)
{
    copybetween(chuo,ten_chuo,'!','@');

    copybetween(chuo,giatri,'@','%');

    if(so_sanh_chuo(ten_chuo,"start"))
    {
        Enable_system=1;
    }
    else if(so_sanh_chuo(ten_chuo,"stop"))
    {
        Enable_system=0;
    }
    else if(so_sanh_chuo(ten_chuo,"auto"))
    {
        che_do_auto=1;
        che_do_manual=0;
        enable_bc1=1;
        enable_bc2=1;
    }
    else if(so_sanh_chuo(ten_chuo,"manual"))
    {
        che_do_auto=0;
        che_do_manual=1;
    }
    else if(so_sanh_chuo(ten_chuo,"vitrix"))
    {
        vitrix=atoi((char*)giatri);
    }
    else if(so_sanh_chuo(ten_chuo,"vitriz"))

```

```

{
    vitriz=atoi((char*)giatri);
}
else if(so_sanh_chuoi(ten_chuoi,"vitribc1"))
{
    vitribc1=atoi((char*)giatri);
    enable_bc1=0;
    enable_dk_vitri_bc1=1;
}
else if(so_sanh_chuoi(ten_chuoi,"tudonggap"))
{
    vitribc1=atoi((char*)giatri);
    copybetween(chuoi,vitrisp_chuoi,'%', '^');
    copybetween(chuoi,vitrihop_chuoi,'^','&');
    vitrisp=atoi((char*)vitrisp_chuoi);
    vitrihop=atoi((char*)vitrihop_chuoi);

    enable_bc1=0;
    tudonggap=1;
    vitribc1_tam=0;
//
//    vitrix=vitrisp;
//    chuaden_vitrix_sp=1;

//
//    vitribc1=atoi((char*)giatri);
//    enable_bc1=0;
//    enable_dk_vitri_bc1=1;
}
else if(so_sanh_chuoi(ten_chuoi,"tudonggapbc2"))
{
    vitribc2=atoi((char*)giatri);

    enable_bc2=0;
    tudonggapbc2=1;
    vitribc2_tam=0;
//
//    vitrix=vitrisp;
//    chuaden_vitrix_sp=1;
//    vitribc1=atoi((char*)giatri);
//    enable_bc1=0;
//    enable_dk_vitri_bc1=1;
}

```

```

else if(so_sanh_chuoi(ten_chuoi,"manualbc1"))
{
    manual_bc1=atoi((char*)giatri);
}
else if(so_sanh_chuoi(ten_chuoi,"manualbc2"))
{
    manual_bc2=atoi((char*)giatri);

}

else if(so_sanh_chuoi(ten_chuoi,"vanhut"))
{
    vanhut=atoi((char*)giatri);
}
else if(so_sanh_chuoi(ten_chuoi,"setsokinh"))
{
    set_sokinh_hop=atoi((char*)giatri);
}
}

//dieu khien vi tri truc z1
void dkdc_vitriz2()
{
    int64_t goc;
    goc=(vitriz-vitrix_tam)*360/8;
    dkdc(dcz2,goc,1,1);
}

//dieu khien vi tri truc x
void dkdc_vitrix()
{
    int64_t goc;
    goc=(vitrix-vitrix_tam)*360/40;
    dkdc(dcx,goc,1,1);

}

//dieu vi tri truc x o che do tu dong
void dkvitri_auto_sp_x(int64_t vtx)
{
    int64_t goc;
    goc=(vtx-vitrix_tam)*360/40;
    dkdc(dcx,goc,1,1);
    vitrix_tam=vtx;
    vitrix=vtx;
}

```

```

//dieu khien bang chuyen 1 o che do tu dong
void dkdc_vitribc1()
{
    int64_t goc;
    goc=(vitribc1-vitribc1_tam);
    dkdc(dcbc1,-goc,1,4);
}

//dieu khien bang chuyen 2 o che do tu dong
void dkdc_vitribc2()
{
    int64_t goc;
    goc=(vitribc2-vitribc2_tam);
    dkdc(dcbc2,goc,1,4);
}

//ham dieu khien vi tri chung cua 3 truc
void dkvitri_auto_sp(int64_t vtx,int64_t vtz,int64_t vtbc1)
{

    vitribc1=vtbc1;
    vitriz=vtz;
    osDelay(300);
    vitrix=vtx;
    //    while((((vitrix==vitrix_tam)&&(vitriz==vitriz_tam))&&(vitribc1==vitribc1_tam)))
    //    {
    //        osDelay(1);
    //    }
    //aa++;
    for(;;)
    {
        if(vitrix==vitrix_tam)
        {
            if(vitriz==vitriz_tam)
            {
                if(vitribc1==vitribc1_tam)
                {
                    //return 1;

                    break;
                }
            }
        }
    }

    osDelay(1);
}

```

```

        }
        //aa++;
    }

//ham ngat khi nhan tin hieu uart tu may tinh
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
    if(huart->Instance==huart3.Instance)
    {
        xulychuoinhan(receive_buf);
        HAL_UART_Transmit(&huart3,receive_buf,strlen((char*)receive_buf),1);
        HAL_UART_Receive_IT(&huart3,receive_buf,60);
    }
}

//tac vu dieu khien chinh, chi hoat dong che do auto
void StartDefaultTask(void *argument)
{
    for(;;)
    {
        if(che_do_auto)
        {
            if(tudonggap==1)
            {
                dkvitri_auto_sp(vitrisp,10,vitribc1);
                dkvitri_auto_sp(vitrisp,do_day_kinh,vitribc1);
                HAL_GPIO_WritePin(Relay2_GPIO_Port,Relay2_Pin,0);
                dkvitri_auto_sp(vitrisp,do_day_kinh+2,vitribc1);
                dkvitri_auto_sp(170,50,vitribc1);
                dkvitri_auto_sp_x(vitrihop);

                dkvitri_auto_sp(vitrihop,do_day_day_hop+do_day_kinh*(sokinh_hop+1),vitribc1);
                HAL_GPIO_WritePin(Relay2_GPIO_Port,Relay2_Pin,1);
                dkvitri_auto_sp(vitrihop,50,vitribc1);

                sprintf((char*)send_buf,"!dagapxong@%d^%d&",sokinh_hop,sohop);

                HAL_UART_Transmit(&huart3,(uint8_t*)"dagapxong",strlen("dagapxong"),1);
                sokinh_hop++;
                sokinh_tong++;
                dkvitri_auto_sp(120,10,vitribc1);
                while(co_hop==0)
                {
                    osDelay(1);
                }
            }
        }
    }
}

```

```

        dkvitri_auto_sp_x(vitrisp);
        enable_bc1=1;
        tudonggap=0;
    }
}

}

//tac vu dieu khien truc z2
void TRUCZ2(void *argument)
{
    for(;;)
    {
        if(Enable_system)
        {
            if(che_do_manual)
            {
                if(homez2==1)
                {
                    if((endstop_z2==0)&&(endstop_z1==0))
                    {
                        dkdc(dcz2,-2,1,1);
                    }
                    else
                    {
                        HAL_GPIO_WritePin(EN5_GPIO_Port,EN5_Pin,1);
                        vitriz=0;
                        vitriz_tam=0;
                        homez2=0;
                        for(int a=0;a<20;a++)
                        {
                            ten_chuoi[a]=0;
                        }
                    }
                }
                if(vitriz_tam!=vitriz)
                {
                    dkdc_vitriz2();
                    vitriz_tam=vitriz;
                }
            }
            else if(che_do_auto)
        }
    }
}

```

```

    {
        if(tudonggap==1)
        {
            if(vitrix_tam!=vitrix)
            {
                dkdc_vitrix2();
                vitrix_tam=vitrix;
            }
        }
    }

//tac vu dieu khien truc x
void TRUCX(void *argument)
{
    vitrix_tam=vitrix;
    for(;;)
    {
        if(Enable_system)
        {
            if(che_do_manual)
            {
                if(so_sanh_chuoi(ten_chuoi,"home"))
                {
                    if(endstop_x==0)
                    {
                        dkdc(dcx,-2,1,1);
                    }
                    else
                    {
                        vitrix=0;
                        vitrix_tam=0;
                        homez1=1;
                        homez2=1;
                        for(int a=0;a<20;a++)
                        {
                            ten_chuoi[a]=0;
                        }
                    }
                }
                if(vitrix_tam!=vitrix)
                {

```

```

        dkdc_vitrix();
        vitrix_tam=vitrix;
    }
}

else if(che_do_auto)
{
    if(tudonggap==1)
    {
        if(vitrix_tam!=vitrix)
        {
            dkdc_vitrix();
            vitrix_tam=vitrix;
        }
    }
}

//tac vu dieu khien bang chuyen1
void BANGCHUYEN1(void *argument)
{
    for(;;)
    {
        if(Enable_system)
        {
            if(che_do_manual)
            {
                if(manual_bc1==1)//left
                {
                    dkdc(dcbc1,-2,1,4);
                }
                if(manual_bc1==2)//right
                {
                    dkdc(dcbc1,2,1,4);
                }
            }
            else if(che_do_auto)
            {
                if(enable_bc1==1)
                {
                    dkdc(dcbc1,-2,1,4);
                }
            }
        }
    }
}

```

```

        if(tudonggap==1)
        {
            if(vitribc1!=vitribc1_tam)
            {
                dkdc_vitribc1();
                vitribc1_tam=vitribc1;
            }
        }

        HAL_GPIO_WritePin(EN3_GPIO_Port,EN3_Pin,0);
    }
}

//tac vu dieu khien bang chuyen 2
void BANGCHUYEN2(void *argument)
{
    dkdc(dcbc2,2,1,4);
    for(;)
    {
        if(Enable_system)
        {
            if(che_do_manual)
            {
                if(manual_bc2==1)//left
                {
                    dkdc(dcbc2,2,1,4);
                }
                if(manual_bc2==2)//right
                {
                    dkdc(dcbc2,-2,1,4);
                }
            }
            else if(che_do_auto)
            {
                if(enable_bc2==1)
                {
                    dkdc(dcbc2,2,1,4);
                }
                if(tudonggapbc2==1)
                {
                    if(vitribc2!=vitribc2_tam)

```

```

        {
            dkdc_vitribc2();
            vitribc2_tam=vitribc2;
            co_hop=1;
        }
        if(sokinh_hop==set_sokinh_hop)
        {
            sohop++;
            sokinh_hop=0;
            enable_bc2=1;
            tudonggapbc2=0;
            co_hop=0;
        }
    }
}

//tac vu kiem tra ngo vao
void KT NGO VAO(void *argument)
{
    for(;;)
    {
        endstop_x=HAL_GPIO_ReadPin(ENDSTOP1_GPIO_Port,ENDSTOP1_Pin);
        endstop_z1=HAL_GPIO_ReadPin(ENDSTOP2_GPIO_Port,ENDSTOP2_Pin);
        endstop_z2=HAL_GPIO_ReadPin(ENDSTOP3_GPIO_Port,ENDSTOP3_Pin);
        nut_start=HAL_GPIO_ReadPin(ENDSTOP4_GPIO_Port,ENDSTOP4_Pin);
        nut_stop=HAL_GPIO_ReadPin(ENDSTOP5_GPIO_Port,ENDSTOP5_Pin);
        if(nut_stop)
        {
            Enable_system=0;
        }
        else if(nut_start)
        {
            Enable_system=1;
        }
        if(Enable_system==0)
        {
            sokinh_hop=0;
            sokinh_tong=0;
            sohop=0;
            co_hop=0;
            vitrix=0;
            vitrix_tam=0;
        }
    }
}

```

```
    vitriz=0;
    vitriz_tam=0;
    vitribc1=0;
    vitribc1_tam=0;
    vitribc2=0;
    vitribc2_tam=0;
    tudonggap=0;
    tudonggapbc2=0;
    che_do_manual=0;
    che_do_auto=0;
}
}
}
```

Biên Bản Chính Sửa Đồ Án Tốt Nghiệp

Tp. HCM ngày 24 tháng 08 năm 2020

Tên đề tài:

Họ và Tên sinh viên 1: Bùi Minh Quang
Họ và Tên sinh viên 1: Nguyễn Quốc Học
GVHD:

MSSV: 16151222
MSSV: 16151167

STT	Nội dung yêu cầu chỉnh sửa	Nội dung đã chỉnh sửa
1	Nêu ưu điểm của hệ thống gấp sản phẩm dung xử lý ảnh so với trường hợp sử dụng cảm biến.	Bổ sung vào mục “3.1.2. Thiết kế cơ cấu gấp sản phẩm” trang 18.
2	Tên đề tài nên được chỉnh sửa lại.	Tên đề tài được đổi thành “Robot tự động gấp sản phẩm ứng dụng công nghệ xử lý ảnh”.
3	Thông nhất các đại từ được sử dụng trong đề tài: Nhóm thực hiện, tôi, chúng em.	<ul style="list-style-type: none">- Từ “nhóm tôi” đổi thành “nhóm chúng em” ở trang vi.- Từ “tôi” đổi thành “nhóm” ở trang 1.- Từ “chúng em” đổi thành “nhóm chúng em” ở trang vii, xiii.
4	Phản “4.4 Thi công phần mềm” ở “Chương 4: Thi công hệ thống” nên chuyển sang “Chương 3: Tính toán và thiết kế”.	Đổi thành “3.3. Thiết kế phần mềm” ở trang 36.

Xác nhận GVHD

Xác nhận Bộ môn


TS. Nguyễn Văn Lán



TS. Trần Vi Đô