

ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

**NGHIÊN CỨU, THIẾT KẾ VÀ TỐI ƯU HỘP GIẢM TỐC
CYCLOID**

NGUYỄN HẢI ANH

anh.nh184342@sis.hust.edu.vn

NGUYỄN HUY HOÀNG

hoang.nh184456@sis.hust.edu.vn

PHẠM MINH ĐĂNG

dang.pm187420@sis.hust.edu.vn

Ngành Cơ Điện Tử

Chuyên ngành: Hệ thống sản xuất tự động

Giảng viên hướng dẫn: TS. Tào Ngọc Linh

Chữ ký của GVHD

Giảng viên phản biện: TS. Phùng Xuân Lan

Chữ ký của GVPB

Nhóm chuyên môn: Công nghệ chế tạo máy

Trường: Cơ khí

HÀ NỘI, 8/2023

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP
(NGÀNH CƠ ĐIỆN TỬ)

1. Thông tin về sinh viên

Họ và tên SV: **Nguyễn Hải Anh** Lớp: **CĐT 03 - K63** ĐT: 0376.550.159
Họ và tên SV: **Nguyễn Huy Hoàng** Lớp: **CĐT 03 - K63** ĐT: 0357.533.689
Họ và tên SV: **Phạm Minh Đăng** Lớp: **ME-NUT17** ĐT: 0392.067.528

Email (đại diện): anh.nh184342@sis.hust.edu.vn

Hệ đào tạo: Chính quy Chuyên ngành: Hệ thống sản xuất tự động

Đồ án tốt nghiệp được thực hiện tại: T-403

Thời gian làm ĐATN: 03/2023 đến 22/07/2023

2. Tên đề tài

NGHIÊN CỨU, THIẾT KẾ VÀ TỐI ƯU HỘP GIẢM TỐC CYCLOID

3. Các số liệu ban đầu:

- Dạng sản xuất : Sản xuất phục vụ cho ngành công nghiệp
- Điều kiện ứng dụng : Tỷ số truyền lớn, thiết kế nhỏ gọn.
- Vật liệu: Thép C45, nhựa POM, nhựa PLA

4. Nội dung thuyết minh và tính toán:

- Nghiên cứu thuật toán tối ưu cho hộp giảm tốc Cycloid
- Tính toán các thông số thiết kế hộp giảm tốc Cycloid
- Thiết lập bản vẽ các chi tiết trong hộp giảm tốc Cycloid
- Thiết lập hệ thống điều khiển động cơ bước

5. Các bản vẽ

- Bản vẽ lắp ráp: 1 bản
- Bản vẽ các chi tiết : 2 bản
- Bản vẽ sơ đồ đấu dây : 1 bản
- Sơ đồ tổng quan và thuật toán: 2 bản

*Hà Nội, ngày tháng năm 2023
Giảng viên hướng dẫn*

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP

(Dành cho Giáo viên hướng dẫn)

Tên đề tài: NGHIÊN CỨU, THIẾT KẾ VÀ TỐI ƯU HỘP GIẢM TỐC
CYCLOID

Họ và tên SV: Nguyễn Hải Anh Lớp: CDT 03 – K63

Họ và tên SV: Nguyễn Huy Hoàng Lớp: CDT 03 – K63

Họ và tên SV: Phạm Minh Đăng Lớp: ME-NUT17

Chuyên ngành: Hệ thống sản xuất tự động

Giáo viên hướng dẫn: TS. Tào Ngọc Linh

NỘI DUNG NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

I. Tác phong làm việc

.....

II. Những kết quả đạt được

.....
.....
.....
.....
.....
.....
.....

III. Hạn chế của đồ án

.....
.....
.....
.....
.....

IV. Kết luận

.....
.....

Đánh giá: điểm

Hà Nội, ngày tháng năm
2023

Giáo viên hướng dẫn
(Ký và ghi rõ họ tên)

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP

(Dành cho Giáo viên phản biện)

Tên đề tài: NGHIÊN CỨU, THIẾT KẾ VÀ TỐI ƯU HỘP GIẢM TỐC CYCLOID

Họ và tên SV: Nguyễn Hải Anh Lớp: CDT 03 – K63
Họ và tên SV: Nguyễn Huy Hoàng Lớp: CDT 03 – K63
Họ và tên SV: Phạm Minh Đăng Lớp: ME-NUT17

Chuyên ngành: Hệ thống sản xuất tự động

Giáo viên phản biện: TS. Phùng Xuân Lan

NỘI DUNG NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

I. Nh^ung k^{ết} qu^a đ^{ạt} đ^{ược}

II. Hạn chế của đồ án

.....
.....
.....

III. Kết luận

Danh sách đính kèm

Đánh giá: điểm

*Hà Nội, ngày tháng năm 2023
Giáo viên phản biện
(Ký và ghi rõ họ tên)*

LỜI CẢM ƠN

Trước hết, chúng em xin gửi lời cảm ơn chân thành đến thầy TS. Tào Ngọc Linh đã tận tình hướng dẫn và hỗ trợ trong suốt quá trình thực hiện đồ án tốt nghiệp. Nhờ sự chỉ dẫn và định hướng tận tâm từ thầy, chúng em đã có cơ hội tiếp cận và khám phá sâu hơn về đề tài của mình. Thầy đã truyền đạt kiến thức, kỹ năng và cung cấp những lời khuyên quý báu, giúp chúng em nắm vững kiến thức chuyên ngành và hoàn thành đồ án tốt nghiệp một cách thành công.

Chúng em cũng muốn bày tỏ lòng biết ơn sâu sắc đến toàn thể quý thầy cô trong NCM Công nghệ Chế tạo máy đã tạo môi trường học tập và nghiên cứu chất lượng. Những kiến thức và kỹ năng chúng em đã học được từ quý thầy cô không chỉ hỗ trợ chúng em trong quá trình làm đồ án tốt nghiệp mà còn sẽ trở thành nền tảng vững chắc cho sự phát triển sau này.

Đồng thời, chúng em xin gửi lời cảm ơn chân thành đến gia đình, người thân và bạn bè đã luôn ủng hộ, động viên và cung cấp sự giúp đỡ suốt quá trình chúng em thực hiện đồ án tốt nghiệp. Sự quan tâm, tình yêu thương và động viên của họ đã truyền sức mạnh và động lực mạnh mẽ cho chúng em vượt qua khó khăn và hoàn thành tốt công việc này.

Chúng em xin chân thành cảm ơn tất cả những người đã đồng hành và góp phần vào thành công của chúng em trong chặng đường đồ án tốt nghiệp này. Những kinh nghiệm, kiến thức và sự hỗ trợ của mọi người sẽ luôn được khắc sâu trong trái tim và có tác động vô cùng lớn đến sự phát triển và tiến bộ của chúng em trong tương lai.

TÓM TẮT NỘI DUNG ĐỒ ÁN

Hộp giảm tốc Cycloid được sử dụng rộng rãi trong các ngành công nghiệp. Chủ yếu là hộp giảm tốc hành tinh trong các ngành thiết bị trộn môi trường, trong dây chuyền sản xuất, v.v. Qua đó có thể thấy được tầm quan trọng của thiết bị này là không hề nhỏ. Hơn nữa đây còn là thiết bị trung gian không thể thiếu giữa động cơ và các bộ phận khác của máy trong các dây chuyền sản xuất nói riêng cũng như máy móc trong các ngành công nghiệp. Qua đó góp phần nâng cao năng suất và hiệu quả làm việc, góp phần không nhỏ vào công cuộc đổi mới đưa đất nước ngày càng phát triển. Bên cạnh các hộp giảm tốc truyền thống sử dụng răng thân khai, hộp giảm tốc hành tinh Cycloid là một sản phẩm mới xuất hiện trên thị trường trong thời gian gần đây do một số ít hãng nước ngoài độc quyền sản xuất. Với ưu điểm nổi trội hơn như tỷ số truyền lớn, kích thước nhỏ gọn, khả năng chịu tải cao và hiệu quả cao, nó ngày càng được sử dụng phổ biến trong các thiết bị máy móc của mọi ngành. Việc nghiên cứu, thiết kế, chế tạo hộp giảm tốc hành tinh Cycloid vẫn đang được quan tâm nhằm phát huy nội lực của ngành Cơ khí trong việc nghiên cứu, thiết kế, chế tạo một sản phẩm mới phục vụ sản xuất.

Với những lý do trên, sinh viên Kỹ thuật cơ điện tử, Đại Học Bách Khoa Hà Nội chúng em được giao nhiệm vụ đồ án tốt nghiệp như sau: “Nghiên cứu, thiết kế và tối ưu hộp giảm tốc Cycloid” bao gồm những nhiệm vụ sau đây:

- + Nghiên cứu thuật toán tối ưu cho hộp giảm tốc Cycloid
- + Tính toán các thông số để chế tạo hộp giảm tốc Cycloid
- + Thiết kế hộp giảm tốc Cycloid
- + Điều khiển tốc độ động cơ gắn hộp giảm tốc bằng PID

Trong quá trình nghiên cứu và chế tạo, chúng em đã gặp nhiều khó khăn, do đê tài mỏm, thời gian và kiến thức. Tuy nhiên sau một thời gian nghiên cứu, thiết kế và chế tạo, chúng em đã hoàn thành đê tài nhờ sự cố gắng của bản thân và đặc biệt là sự chỉ dẫn tận tình của giáo viên hướng dẫn là TS. Tào Ngọc Linh, cùng các thầy cô Bộ môn Công nghệ Chế tạo máy. Đến nay, chúng em đã hoàn thành được các nhiệm vụ của đồ án tốt nghiệp.

Một lần nữa chúng em xin cảm ơn!

MỤC LỤC

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI	1
1.1 Đặt vấn đề	1
1.1.1 Hộp giảm tốc là gì?	1
1.1.2 Truyền động Cycloid là gì?.....	1
1.1.3 Các loại hộp giảm tốc hiện nay.....	2
1.1.4 Tình hình chung	5
1.1.5 Kết luận	5
1.2 Tình hình nghiên cứu trong nước và trên thế giới.....	6
1.2.1 Tình hình trên thế giới.....	6
1.2.2 Tình hình trong nước:	8
1.3 Khái quát chung về các nghiên cứu của nhóm.....	9
1.3.1 Mục tiêu đề tài.....	10
1.3.2 Đối tượng nghiên cứu	11
1.3.3 Phương pháp nghiên cứu	11
CHƯƠNG 2. NGUYÊN LÝ, CƠ SỞ LÝ THUYẾT VÀ TÍNH TOÁN THIẾT KẾ MÁY.....	12
2.1 Nguyên lý hoạt động của Hộp giảm tốc, bộ truyền Cycloid.....	12
2.1.1 Nguyên lý hoạt động của Hộp giảm tốc.....	12
2.1.2 Nguyên lý hoạt động của bộ truyền Cycloid	13
2.2 Tính toán biên dạng đĩa Cycloid.....	14
2.2.1 Khái niệm	14
2.2.2 Thiết lập phương trình biên dạng đĩa Cycloid	17
2.2.3 Nhận xét	21
2.3 Chế tạo biên dạng đĩa Cycloid	22
2.3.1 Dùng dao phay đĩa với phương pháp chép hình	22
2.3.2 Dùng dao phay lăn	22
2.3.3 Gia công trên máy xoc bằng dao xoc định hình.....	23
2.3.4 Cắt răng trên các máy cắt hiện đại CNC	24
2.4 Lắp ráp hệ thống cơ khí	27
2.4.1 Khái niệm về lắp ráp hệ thống cơ khí:	27
2.4.2 Kỹ thuật lắp ráp	28
CHƯƠNG 3. TỔNG QUAN VỀ HỆ THỐNG ĐIỀU KHIỂN ĐỘNG CƠ BƯỚC CHO HỘP GIẢM TỐC	32

3.1	Tổng quan về Arduino.....	32
3.1.1	Giới thiệu về Arduino	32
3.1.2	Phần cứng của Arduino Board.....	32
3.1.3	Lập trình Arduino IDE.....	35
3.1.4	Thư viện lập trình trên Arduino IDE	36
3.1.5	Arduino MEGA 2560	38
3.2	Tổng quan về Động cơ bước NEMA 23	40
3.3	Tổng quan về C# - Visual Studio.....	43
3.3.1	Tổng quan về Visual Studio	43
3.3.2	Tổng quan về ngôn ngữ lập trình Python	49
3.3.3	Tổng quan về ngôn ngữ lập trình C#	51
3.4	Tổng quan về Driver A4988, Nguồn tần số, Encoder	52
3.4.1	A4988 Stepper Motor Driver.....	52
3.4.2	Nguồn tần số	55
3.4.3	Encoder AS5600	57
CHƯƠNG 4. CƠ SỞ LÝ THUYẾT, HÀM TÍNH TOÁN SỬ DỤNG TRONG THUẬT TOÁN VÀ TỐI UỐNG CHO HỘP GIẢM TỐC.....		59
4.1	Tối ưu các biến thiết kế và hàm số	59
4.1.1	Chức năng khách quan.....	59
4.1.2	Khối lượng của bộ giảm tốc	60
4.1.3	Tải trọng xuyên tâm khi quay ô trục.....	60
4.1.4	Ứng suất uốn tối đa của pin	60
4.2	Điều kiện hạn chế cho các biến	61
4.2.1	Hệ số biên độ ngắn.....	61
4.2.2	Biên dạng răng Cycloid	62
4.2.3	Đường kính tối đa của lỗ chốt hình trụ	62
4.2.4	Hệ số đường kính chốt.....	63
4.2.5	Độ bền uốn của bánh răng chốt	63
4.2.6	Cường độ tiếp xúc giữa chốt trụ và lỗ chốt trụ	63
4.2.7	Độ bền uốn của chốt trụ.....	63
4.2.8	Phương trình tối ưu thông số cuối cùng.....	63
4.3	Thuật toán tối ưu cho các biến thiết kế.....	64
4.3.1	Tổng quan về thuật toán Particle Swarm Optimization (PSO). 64	64
4.3.2	Nội dung của chương trình tối ưu.....	66

4.3.3	Kiểm nghiệm và tinh chỉnh lại kết quả tối ưu sau khi đã chạy thuật toán tối ưu	80
CHƯƠNG 5. THIẾT KẾ, CHẾ TẠO THỰC NGHIỆM HỘP GIẢM TỐC CYCLOID.....		88
5.1	Các thành phần cơ khí sau khi tinh chỉnh lại hộp giảm tốc Cycloid	88
5.1.1	Tổng quan về loại hộp giảm tốc cycloid mà nhóm thiết kế	88
5.1.2	Vỏ hộp giảm tốc Cycloid	89
5.1.3	Bánh răng Cycloid	90
5.1.4	Nắp dẫn truyền lực	91
5.1.5	Đệm lệch tâm	92
5.1.6	Nắp chặn vòng bi	93
5.1.7	Vòng bi tại bậc lệch tâm và tại nắp đậy dẫn truyền	94
5.2	Các thành phần điều khiển và giám sát hộp giảm tốc Cycloid.....	96
5.2.1	Bản vẽ điện.....	96
5.2.2	Lập trình điều khiển trên Arduino	97
5.3	Thuật toán điều khiển.....	100
5.3.1	Bộ điều khiển PID	102
5.3.2	Phương pháp Ziegler-Nichols	103
5.3.3	Tính toán các tham số của bộ điều khiển P	103
5.4	Giao diện điều khiển Winform	107
5.4.1	Tạo hình ảnh và giao diện cho Winform (Front End).....	108
5.4.2	Tạo các tính năng cho giao diện (Back End)	114
5.5	Kết quả thực tế	115
CHƯƠNG 6. KẾT LUẬN, ĐÁNH GIÁ, HƯỚNG PHÁT TRIỂN.....		117
6.1	Kết quả đạt được của đề tài	117
6.2	Ý nghĩa khoa học	117
6.3	Những vấn đề còn tồn tại.....	117
6.4	Hướng phát triển.....	118
TÀI LIỆU THAM KHẢO		119

DANH MỤC HÌNH VẼ

Hình 1.1 Một số hộp giảm tốc trên thị trường	1
Hình 1.2 Mô phỏng truyền động Cycloid	2
Hình 1.3 Hộp giảm tốc bánh răng trụ 1 cấp ZD	3
Hình 1.4 Hộp giảm tốc 2 cấp	3
Hình 1.5 Cấu tạo hộp giảm tốc bánh răng hành tinh	4
Hình 1.6 Hộp giảm tốc trực vít- bánh vít.....	4
Hình 1.7 Hộp giảm tốc Cyclo	5
Hình 1.8 Một số loại Động cơ – Hộp giảm tốc bánh răng con lăn của hãng Hap Dong.....	7
Hình 1.9 Một số loại Động cơ-Hộp giảm tốc bánh răng con lăn của hãng Sumiomo	8
Hình 1.10 Một số loại động cơ- Hộp giảm tốc bánh răng con lăn của hãng Centa	8
Hình 1.11 Các đĩa Cycloid do Trung tâm Tự động hóa – DHBKHN chế tạo.....	9
Hình 1.12 Động cơ- Hộp giảm tốc bánh răng con lăn do Nhà máy Cơ khí Mai Động sản xuất	9
Hình 2.1 Giải thích nguyên lý hoạt động của hộp giảm tốc	12
Hình 2.2 Cấu tạo cơ bản bên trong hộp giảm tốc	12
Hình 2.3 Mô tả nguyên lý làm việc của bộ truyền Cycloid.....	13
Hình 2.4 Các nhóm chính trong bộ truyền Cycloid.....	14
Hình 2.5 Khai triển của một hộp giảm tốc bánh răng con lăn	14
Hình 2.6 Sự tạo thành đường EpiCycloid.....	15
Hình 2.7 Sự tạo thành đường HypoCycloid	15
Hình 2.8 Sự tạo thành đường EpiCycloid kéo dài	16
Hình 2.9 Sự tạo thành biên dạng ăn khớp EpiCycloid với con lăn răng chốt	17
Hình 2.10 Xây dựng biên dạng đĩa Cycloid	19
Hình 2.11 Sơ đồ dao phay đĩa địa hình đặc biệt	22
Hình 2.12 Tính profin dao xọc bằng phương pháp giải tích.....	23
Hình 2.13 Máy phay đứng CNC	24
Hình 2.14 Một số loại máy CNC gia công biên dạng trực tiếp	25
Hình 2.15 Gia công profin răng đĩa Cycloid trên máy phay đứng CNC	25
Hình 2.16 Gia công profin răng đĩa Cycloid với dụng cụ có bán kính cho trước	26
Hình 2.17 Mô tả quá trình cắt biên dạng đĩa Cycloid trực tiếp	27
Hình 3.1 Mạch arduino Uno	33
Hình 3.2 Mạch Arduino Mega	34
Hình 3.3 Mạch arduino Nano	34
Hình 3.4 Phần mềm Arduino IDE	35

Hình 3.5 Giao diện của Arduino IDE	36
Hình 3.6 Giao diện thư viện của Arduino IDE	37
Hình 3.7 Sơ đồ chân arduino MEGA 2560.....	39
Hình 3.8 Động cơ bước Nema 23	40
Hình 3.9 Cấu tạo cơ bản StepMotor.....	40
Hình 3.10 Giao diện cơ bản của Visual Studio.....	44
Hình 3.11 Driver A4988	52
Hình 3.12 Sơ đồ chân Driver A4988	54
Hình 3.13 Nguồn tổ ong hiện nay trên thị trường	55
Hình 3.14 Cấu tạo nguồn tổ ong	55
Hình 3.15 Sơ đồ nguyên lý hoạt động nguồn tổ ong	56
Hình 4.1 Cấu tạo điển hình của một bánh răng trụ có 2 trục	61
Hình 4.2 Sơ đồ thuật toán tối ưu	66
Hình 4.3 Kết quả tối ưu sau khi chạy thuật toán	80
Hình 4.4 Kết quả trước tinh chỉnh bằng python.....	85
Hình 4.5 Kết quả sau tinh chỉnh bằng python.....	86
Hình 5.1 Bản vẽ chi tiết vỏ hộp giảm tốc.....	89
Hình 5.2 Bản vẽ chi tiết bánh răng Cycloid	90
Hình 5.3 Bản vẽ chi tiết nắp dẫn truyền lực.....	91
Hình 5.4 Bản vẽ chi tiết đệm lệch tâm	92
Hình 5.5 Bản vẽ chi tiết nắp chặn vòng bi	93
Hình 5.6 Tiêu chuẩn vòng bi	94
Hình 5.7 Phân loại vòng bi 6807	95
Hình 5.8 Sơ đồ đi dây.....	96
Hình 5.9 Sơ đồ thuật toán điều khiển	101
Hình 5.10 Sơ đồ nguyên lý hoạt động.....	101
Hình 5.11 Bộ PID truyền thống	102
Hình 5.12 Hệ ở biên giới ổn định.....	104
Hình 5.13 Đáp ứng của hệ với bộ điều khiển PID	104
Hình 5.14 Đáp ứng của hệ với bộ điều khiển PI	105
Hình 5.15 Đáp ứng của hệ với bộ điều khiển PID	106
Hình 5.16 Sơ đồ thuật toán điều khiển tốc độ động cơ	107
Hình 5.17 Giao diện chính	107
Hình 5.18 Mô phỏng hộp giảm tốc Cycloid trên SolidWorks	115
Hình 5.19 Gia công 1 số chi tiết bằng nhựa POM	115
Hình 5.20 Mô hình các thành phần hộp giảm tốc được in bằng máy in 3D	116
Hình 5.21 Chạy thử nghiệm hộp giảm tốc Cycloid	116

DANH MỤC BẢNG

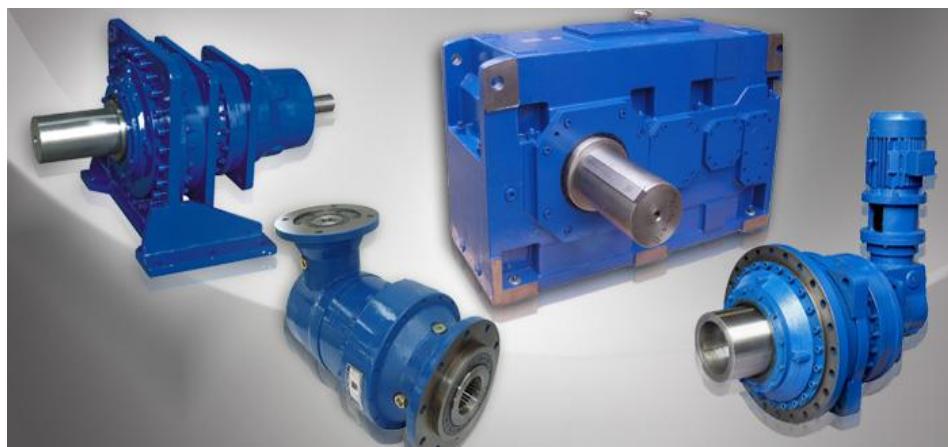
Bảng 2.1 Tỷ lệ khói lượng công việc	28
Bảng 3.1 Đặc tính kỹ thuật Arduino MEGA 2560	39
Bảng 3.2 Bảng thông số một số động cơ bước	43
Bảng 3.3 Bảng thông số một số loại động cơ bước	43
Bảng 3.4 Bảng lựa chọn chế độ Microstep	54
Bảng 4.1 Bảng giới hạn của các biến thiết kế	59
Bảng 4.2 Bảng thông số đầu vào của hộp giảm tốc	60
Bảng 4.3 Bảng số thiết kế của hộp giảm tốc	87
Bảng 5.1 Tính toán hệ số bằng phương pháp Ziegler - Nichols	103

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI

1.1 Đặt vấn đề

1.1.1 Hộp giảm tốc là gì?

Đúng như tên gọi của nó, hộp giảm tốc là một thiết bị dùng để giảm tốc độ các vòng quay. Đây là thiết bị trung gian giữa động cơ và các bộ phận khác của máy trong dây chuyền sản xuất với chức năng điều chỉnh tốc độ của động cơ điện cho phù hợp với yêu cầu.



Hình 1.1 Một số hộp giảm tốc trên thị trường

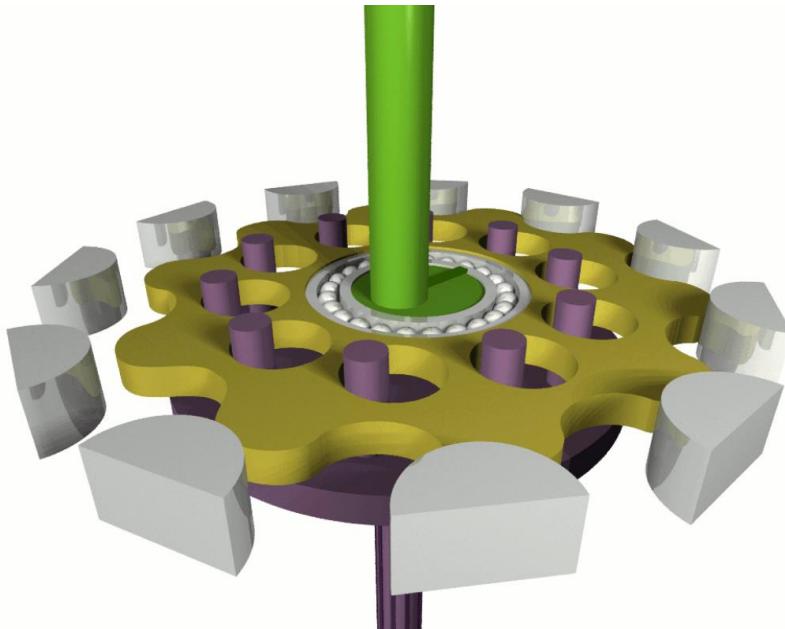
Động cơ điện thường có tốc độ quay vô cùng lớn, nhưng khi ứng dụng vào sản xuất thực tế thì nhiều trường hợp cần tốc độ quay nhỏ hơn nhiều. Lúc này việc chế tạo động cơ có công suất nhỏ cần chi phí rất cao, trong khi động cơ có công suất lớn lại nhỏ gọn, thiết kế đơn giản, chi phí thấp. Chính vì vậy, để làm giảm tốc độ động cơ sao cho phù hợp với yêu cầu của các máy móc thiết bị thì người ta đã tạo ra hộp giảm tốc. Hơn nữa, khi dùng hộp giảm tốc tải trọng của động cơ cũng được tăng lên rất nhiều.

1.1.2 Truyền động Cycloid là gì?

Bộ truyền động xyclon hay bộ giảm tốc độ xyclot là một cơ cấu để giảm tốc độ của trục đầu vào theo một tỷ lệ nhất định. Các bộ giảm tốc độ lốc xoáy có khả năng đạt tỷ lệ tương đối cao trong kích thước nhỏ gọn với phản ứng damped rất thấp.

Trục đầu vào truyền động một ổ trục lệch tâm, vòng bi này lần lượt truyền động đĩa cycloidal theo chuyển động lệch tâm, cycloidal. Chu vi của đĩa này được truyền tới một bánh răng vành cố định và có một loạt các chốt trục đầu ra hoặc con

lần được đặt xuyên qua mặt đĩa. Các chân trục đầu ra này trực tiếp điều khiển trục đầu ra khi đĩa quay vòng. Chuyển động xuyên tâm của đĩa không được dịch sang trục đầu ra.



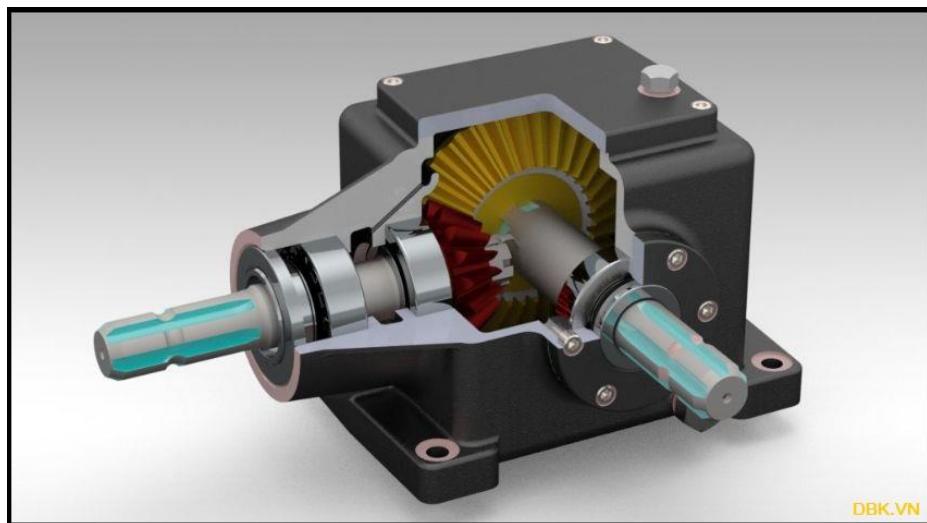
Hình 1.2 Mô phỏng truyền động Cycloid

1.1.3 Các loại hộp giảm tốc hiện nay

Thông thường có 2 cách phân loại hộp giảm tốc là phân loại theo cấp giảm tốc và phân loại theo cấu tạo.

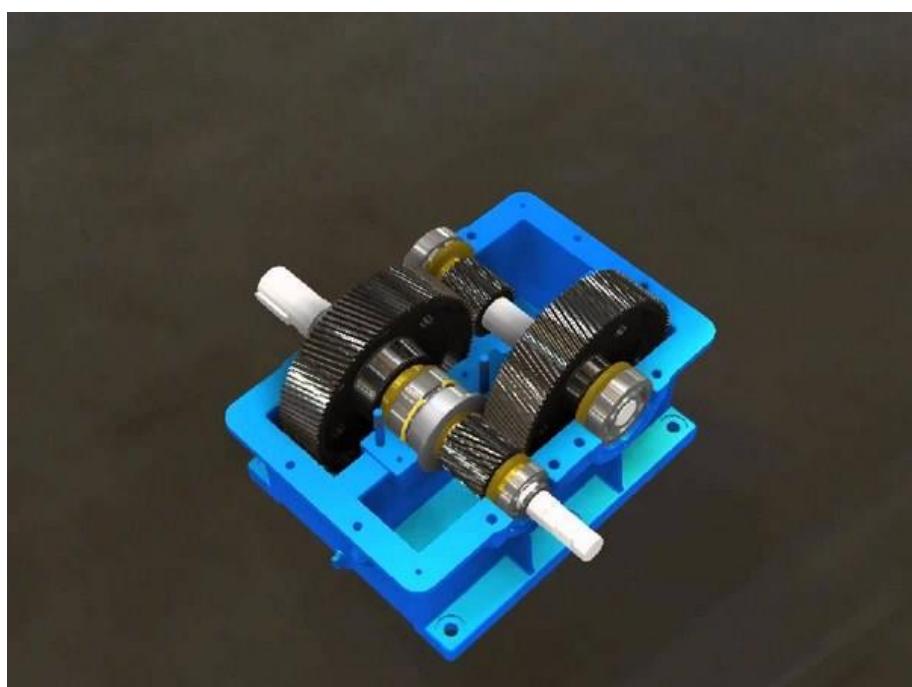
Phân loại theo cấp giảm tốc:

- Dạng giảm tốc mà đầu ra phù hợp với yêu cầu qua nhiều lần thay đổi tỷ số truyền động bằng cách thay đổi số lượng răng của các bánh răng người ta gọi là hộp giảm tốc nhiều cấp ngược lại khi thay đổi một lần số lượng bánh răng người ta gọi là hộp giảm tốc một cấp. Nếu phân theo cấp giảm tốc ta có rất nhiều loại hộp giảm tốc: 1 cấp, 2 cấp, 3 cấp, ...



DBK.VN

Hình 1.3 Hộp giảm tốc bánh răng trục 1 cấp ZD



Hình 1.4 Hộp giảm tốc 2 cấp

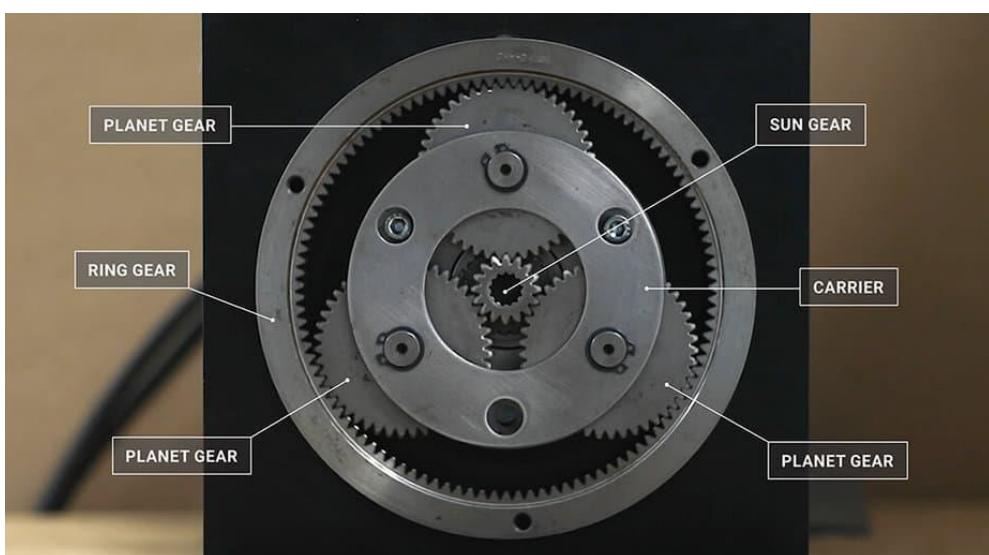
- Chúng ta có thể hiểu số cấp là số lần thay đổi tỉ số truyền động. Ví dụ, ta muốn tỉ số truyền động bằng 4, chỉ cần lắp phôi hợp 2 bánh răng với số lượng răng tương ứng với tỉ lệ truyền động này là 1:4. Hộp giảm tốc chỉ 1 lần truyền động, như vậy ta gọi là hộp giảm tốc loại 1 cấp.

- Tương tự như thế ta có hộp giảm tốc loại 2 cấp, 3 cấp, ... Thường thì khi chế tạo hộp giảm tốc, người ta thường chế tạo hộp nhiều cấp với tỉ số truyền mỗi cấp trong khoảng từ 3-5.

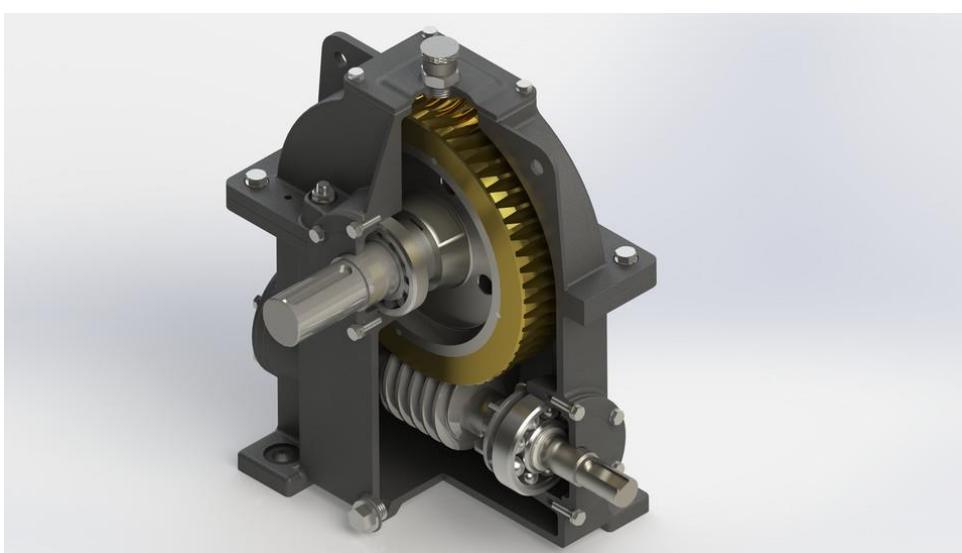
Phân loại theo cấu tạo:

Chủ yếu dựa vào hình dáng bên ngoài và thiết kế cấu tạo, người ta sẽ đưa ra tên gọi cho các loại hộp giảm tốc:

- Hộp giảm tốc bánh răng hành tinh
- Bánh răng nón, bánh răng thẳng
- Bánh răng côn- trụ
- Bánh răng- trục vít , trục vít- bánh răng
- Bánh răng trụ: khai triển, phân đôi, đồng trục
- Cyclo



Hình 1.5 Cấu tạo hộp giảm tốc bánh răng hành tinh



Hình 1.6 Hộp giảm tốc trục vít- bánh vít



Hình 1.7 Hộp giảm tốc Cyclo

1.1.4 Tình hình chung

Việc sáng tạo ra các loại hộp giảm tốc chất lượng cao có tỷ số truyền lớn, kích thước gọn nhẹ luôn là mục tiêu phấn đấu của nhiều công trình nghiên cứu khoa học kỹ thuật hàng thế kỷ nay. Trong những thập kỷ gần đây các yêu cầu này lại trở nên đa dạng hơn.

Các loại hộp giảm tốc bánh răng hành tinh truyền thống về danh nghĩa có thể đạt được tỉ số truyền cao, nhưng thực tế do sự hạn chế về hiệu suất nên nhiều trường hợp không đáp ứng được. Tỷ số truyền và hiệu suất thường không đồng thuận, nên thường phải cân đối hai chỉ tiêu này. Việc tính toán tỷ số truyền và hiệu suất cũng là 2 vấn đề cơ bản của các bộ truyền bánh răng hành tinh kiểu mới.

1.1.5 Kết luận

Do đặc điểm ăn khớp của loại bộ truyền Cycloid này không có khe hở cạnh răng nên làm việc êm, không gây va chạm khi đổi chiều quay. Cùng với khối lượng và kích thước nhỏ gọn nên được ứng dụng ngày càng nhiều trong các máy hiện đại, đặc biệt thích hợp để ứng dụng trong công nghệ Robot và các thiết bị y học.

- ❖ **Ưu điểm:**
- + Tỷ lệ truyền động cao: Hộp giảm tốc cycloid có thể đạt được tỷ lệ truyền động lớn, cho phép tăng hoặc giảm tốc độ của trực đầu ra một cách hiệu quả.
- + Khả năng chịu tải cao: Thiết kế cycloid với răng cưa hình tròn giúp tăng khả năng chịu tải và tăng tuổi thọ của hộp giảm tốc.

- + Kích thước nhỏ gọn: Hộp giảm tốc cycloid có kích thước nhỏ gọn và có thể tích chiếm ít không gian.
- + Khả năng chuyển động mịn: Cấu trúc đặc biệt của hộp giảm tốc cycloid giúp giảm độ rung và nhiễu trong quá trình truyền động, tạo ra chuyển động mịn hơn.
- + Độ chính xác cao: Hộp giảm tốc cycloid có độ chính xác cao, giúp đạt được đồng bộ chính xác giữa trục đầu vào và trục đầu ra.
- ❖ Nhược điểm:
 - + Chi phí cao: Hộp giảm tốc cycloid có chi phí sản xuất và lắp đặt cao hơn so với một số hệ thống truyền động khác.
 - + Hiệu suất truyền động không cao: Do các mô men truyền động trong hộp giảm tốc cycloid được chia thành nhiều hệ số truyền động nhỏ, hiệu suất của nó không cao như các hệ thống truyền động khác như hộp số hướng dẫn hoặc hộp số một cấp.
 - + Độ ôn cao: Một số hộp giảm tốc cycloid có tiếng ôn hoạt động lớn, đặc biệt khi hoạt động ở tốc độ cao.
 - + Bảo trì phức tạp: Do cấu trúc phức tạp và số lượng bộ phận nhiều, việc bảo trì và sửa chữa hộp giảm tốc cycloid có thể khó khăn và tốn kém.
 - + Tốc độ đáp ứng chậm: Do cấu trúc phức tạp, tốc độ đáp ứng của hộp giảm tốc cycloid có thể chậm hơn so với một số hệ thống truyền động khác.

1.2 Tình hình nghiên cứu trong nước và trên thế giới

1.2.1 Tình hình trên thế giới

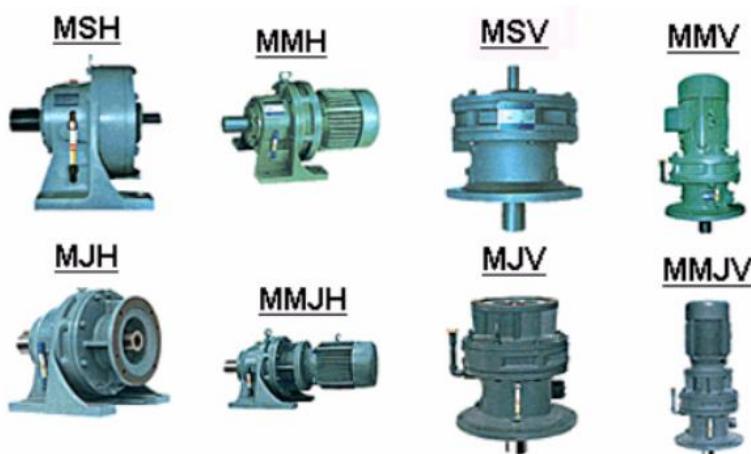
Hộp giảm tốc (Gear Box) là bộ phận quan trọng và cần thiết, nó có mặt trong hầu hết các loại thiết bị máy móc làm nhiệm vụ truyền công suất và đảm bảo tốc độ đầu ra theo yêu cầu. Tuy được đòi hỏi phải có kích thước nhỏ gọn, song vẫn đảm bảo độ chính xác động học, khả năng truyền công suất, làm việc ổn định và tuổi thọ theo yêu cầu. Sản phẩm hộp giảm tốc rất đa dạng về chủng loại kết cấu, công suất và tỉ số truyền. Hiện nay, trên thế giới cũng có rất nhiều hãng chế tạo hộp giảm tốc như: ASEA (Thụy Điển), NORD (Phần Lan), Falcone (Mỹ), Wattz (Đức), General Motor (Mỹ), MORIS (Anh), SUMITOMO (Nhật Bản)... Để tạo ra được các sản phẩm hộp giảm tốc có kích thước nhỏ gọn, hiệu suất cao, làm việc

êm, ổn định và tuổi thọ cao, bên cạnh việc nghiên cứu tối ưu hóa thiết kế, cải tiến công nghệ và thiết bị chế tạo hộp giảm tốc truyền thống sử dụng bánh răng thân khai, các hãng nước ngoài còn nghiên cứu chế tạo các hộp giảm tốc sử dụng các biên dạng răng mới, trong đó phải kể đến hộp giảm tốc hành tinh Cycloid do hãng SUMITOMO (Nhật Bản), DARALI DRIVER (Đài Loan)... chế tạo.

Tuy nhiên, với năng lực khoa học trong nước như hiện nay, việc tự nghiên cứu phát triển một hệ thống như vậy trong trường đại học, với sự tham gia của các nghiên cứu viên là sinh viên, học viên cao học..., cũng góp phần nâng cao năng lực của các nhà khoa học trong nước, góp phần tích cực vào quá trình đào tạo và phát triển nhân lực khoa học kỹ thuật.

Hãng Hap Dong của Hàn Quốc có hộp giảm tốc kiểu nằm và đứng với:

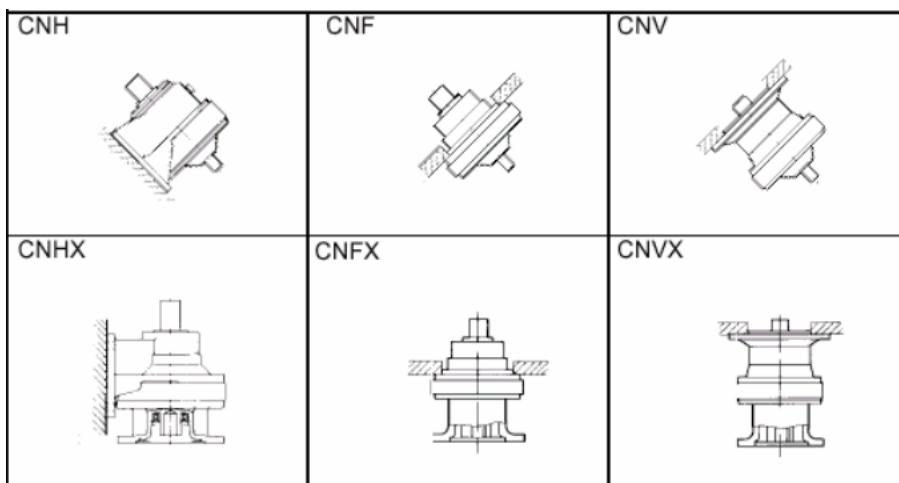
- Tỉ số truyền từ 11 đến 7569
- Công suất từ 0,2 đến 30 kW



Hình 1.8 Một số loại Động cơ – Hộp giảm tốc bánh răng con lăn của hãng Hap Dong

Hãng Sumitomo của Nhật thì ngoài các kiểu nằm và đứng còn có kiểu nằm nghiêng với:

- Hộp giảm tốc có thể chịu được sự quá tải đến 50%
- Tỉ số truyền một cấp từ 6 đến 119, hai cấp từ 102 đến 7569
- Công suất từ 0,03 kW đến 173 kW
- Momen xoắn tối hạn đạt tới 60700Nm



Hình 1.9 Một số loại Động cơ-Hộp giảm tốc bánh răng con lăn của hãng Sumiomo

Hãng Centa của Anh cũng sản xuất hộp giảm tốc Cycloid với nhiều chủng loại khác nhau:



Hình 1.10 Một số loại động cơ- Hộp giảm tốc bánh răng con lăn của hãng Centa

1.2.2 Tình hình trong nước:

Hiện nay, ở trong nước thị trường hộp giảm tốc chưa được nhiều đơn vị nghiên cứu, thiết kế và sản xuất quan tâm thích đáng. Có thể kể đến một vài đơn vị sản xuất hộp giảm tốc trong nước như: Công ty cơ khí Hà Nội, Công ty cơ khí Duyên Hải, Công ty cơ khí Trần Hưng Đạo, Công ty máy kéo và máy nông cụ Hà Tây, Công ty Diezen Sông Công ... Các công ty này thường chỉ sản xuất các hộp giảm tốc (sử dụng răng thân khai) đặc chủng để phục vụ cho một thiết bị cụ thể nào đó. Tuy nhiên có duy nhất công ty cơ khí Duyên Hải sản xuất hộp giảm tốc hai bậc GT-2B theo các gam công suất và tỷ số truyền khác nhau do Viện nghiên cứu máy (nay là Viện Nghiên cứu Cơ khí) thiết kế trước năm 1990 để bán ra thị trường. Trước thời gian thực hiện đề tài, tại trung tâm nghiên cứu kỹ thuật tự động hóa,

Đại học Bách Khoa Hà Nội, trên cơ sở vận dụng lý thuyết ăn khớp cơ bản, đã bước đầu nghiên cứu phần lý thuyết như các phương pháp tạo hình, xác định các dụng cụ cắt theo nguyên tắc bao hình, thiết kế hộp giảm tốc hành tinh Cycloid theo nguyên tắc modul hóa.

Trên thị trường nước ta thì khách hàng ưa chuộng các hộp giảm tốc của Đài Loan, Trung Quốc có giá thành mềm và của Nhật Bản, Mỹ vì có độ bền bỉ cao. Việc lựa chọn thiết bị cũng trở nên đơn giản hơn nhiều so với trước đây với các tiêu chí: Giá phải chăng, model đa dạng, chất lượng ổn định, chế độ bảo hành tốt.



Hình 1.11 Các đĩa Cycloid do Trung tâm Tự động hóa – DHBKHN chế tạo



Hình 1.12 Động cơ- Hộp giảm tốc bánh răng con lăn do Nhà máy Cơ khí Mai Động sản xuất

1.3 Khái quát chung về các nghiên cứu của nhóm

Nhóm nghiên cứu có thể tiến hành các hoạt động sau liên quan đến nghiên cứu, thiết kế và tối ưu hộp giảm tốc cycloid:

- Nghiên cứu tổng quan về hộp giảm tốc cycloid: Nhóm có thể thực hiện nghiên cứu chi tiết về cấu trúc, nguyên lý hoạt động và ứng dụng của hộp giảm tốc

cycloid. Điều này bao gồm tìm hiểu về các thành phần chính, cơ chế truyền động và các yếu tố ảnh hưởng đến hiệu suất và tính năng của hộp giảm tốc.

- Thiết kế và tối ưu hộp giảm tốc cycloid: Nhóm có thể tiến hành việc thiết kế hộp giảm tốc cycloid tối ưu dựa trên yêu cầu và mục tiêu cụ thể. Điều này bao gồm tìm kiếm và đánh giá các cấu trúc và thông số tối ưu để đạt được hiệu suất và tính năng tốt nhất. Nhóm có thể sử dụng các phương pháp tối ưu hóa như thuật toán di truyền, tìm kiếm cục bộ hoặc thuật toán tiến hóa để tìm kiếm các thông số tối ưu cho hộp giảm tốc cycloid.

- Đánh giá hiệu suất: Nhóm có thể tiến hành các thử nghiệm và đánh giá hiệu suất của hộp giảm tốc cycloid thiết kế và tối ưu hóa. Điều này bao gồm việc đo và phân tích các yếu tố như hiệu suất truyền động, tải trọng tối đa, độ ổn định và độ chính xác. Đánh giá hiệu suất cũng có thể bao gồm thử nghiệm trong điều kiện hoạt động thực tế để xác minh tính ứng dụng và đáng tin cậy của hộp giảm tốc cycloid.

- Nghiên cứu ứng dụng và phát triển: Nhóm có thể nghiên cứu và phát triển các ứng dụng mới của hộp giảm tốc cycloid trong các lĩnh vực như công nghiệp, tự động hóa, robot học, và máy móc chính xác. Điều này bao gồm việc áp dụng và tùy chỉnh hộp giảm tốc cycloid để đáp ứng các yêu cầu cụ thể trong các ứng dụng khác nhau.

Tổng quan chung về các nghiên cứu trên sẽ tập trung vào nghiên cứu chi tiết về cấu trúc, hoạt động và hiệu suất của hộp giảm tốc cycloid, cùng với việc thiết kế và tối ưu hóa để đạt được hiệu suất tốt nhất và đáp ứng yêu cầu ứng dụng cụ thể.

1.3.1 Mục tiêu đề tài

- Nghiên cứu thị trường hộp giảm tốc ở Việt Nam hiện nay và khả năng tối ưu hóa phát triển hộp giảm tốc.
- Tìm hiểu 1 số mẫu hộp giảm tốc bánh răng đã có trên thị trường từ đó tìm được những ưu nhược điểm của các loại hộp giảm tốc khác nhau từ đó đưa ra được lựa chọn phương án thiết kế tối ưu nhất phù hợp với nhu cầu của người sử dụng.
- Thiết kế, tối ưu chế tạo hộp giảm tốc.

- + Xây dựng thuật toán tối ưu cho hộp giảm tốc.
- + Thiết lập công thức tính toán, thiết kế cho các bộ phận chính của hộp giảm tốc.
- + Điều khiển tốc độ động cơ.
- + Xây dựng bản vẽ thiết kế cơ khí.
- Giải pháp để hộp giảm tốc Cycloid có thể sử dụng được rộng rãi và có thể thương mại hóa để bán trên thị trường Việt Nam.
- Chế tạo được mô hình để kiểm nghiệm các kết quả.

1.3.2 Đối tượng nghiên cứu

Trong phạm vi của đề tài, nhóm chỉ tập trung vào nghiên cứu, tối ưu, chế tạo hộp giảm tốc sử dụng bộ truyền Cycloid có những ưu điểm tỷ số truyền cao, hiệu suất đảm bảo...

1.3.3 Phương pháp nghiên cứu

- Nghiên cứu tài liệu.

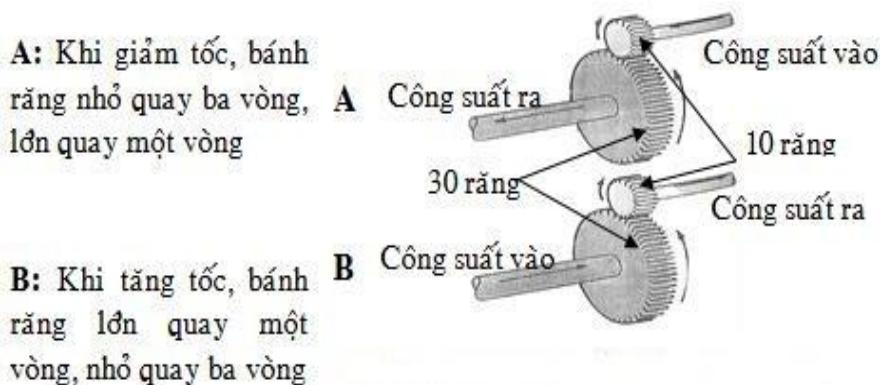
Tài liệu gồm :

- + Thông tin về các nhà sản xuất hộp giảm tốc, thông tin quảng cáo của các đại lý phân phối và trên mạng internet. Mạng internet là nơi cung cấp thông tin nhanh nhất, rẻ và rất thuận tiện.
- + Các sách lý thuyết cơ bản về các vấn đề liên quan như : công nghệ chế tạo máy, chi tiết máy, cơ sở lý thuyết...
- Phân tích số liệu, xây dựng công thức tính toán. Đây là phương pháp được dùng chủ yếu trong đề tài nhằm tìm ra các công thức cụ thể cho việc thiết kế một hộp giảm tốc.
- Phương pháp thực nghiệm .Đây là phương pháp kiểm nghiệm các công thức được lập ra và các phần không tính toán mà chọn theo kinh nghiệm.

CHƯƠNG 2. NGUYÊN LÝ, CƠ SỞ LÝ THUYẾT VÀ TÍNH TOÁN THIẾT KẾ MÁY

2.1 Nguyên lý hoạt động của Hộp giảm tốc, bộ truyền Cycloid

2.1.1 Nguyên lý hoạt động của Hộp giảm tốc



Hình 2.1 Giải thích nguyên lý hoạt động của hộp giảm tốc

- Thông thường hộp số giảm tốc là một hệ bánh răng ăn khớp với nhau theo đúng tỷ số truyền và momen quay đã thiết kế để lấy ra số vòng quay mà người sử dụng cần. Cũng có 1 số hộp giảm tốc không dùng hệ bánh răng thường mà dùng hệ bánh răng vi sai hoặc hệ bánh răng hành tinh. Với hộp số giảm tốc loại này thì kích thước sẽ nhỏ, gọn, chịu lực làm việc lớn.
- Tùy vào điều kiện làm việc và tính toán thì người ta sẽ thiết kế 1 hộp giảm tốc phù hợp với công việc. Khi người ta cần 1 số vòng quay trong 1 phút mà không có động cơ nào đáp ứng được thì người ta sẽ dùng đến hộp giảm tốc.

Cấu tạo cơ bản của hộp giảm tốc:

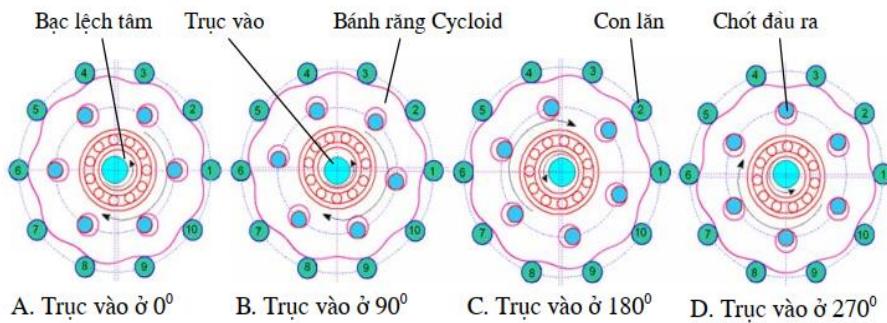


Hình 2.2 Cấu tạo cơ bản bên trong hộp giảm tốc

Bên trong hộp giảm tốc có cấu tạo cũng khá đơn giản, chúng gồm các bánh răng thẳng và nghiêng ăn khớp với nhau theo một tỷ số truyền nhất định. Khi có nguồn điện cấp vào, thiết bị này có thể tạo nên số vòng quay phù hợp với yêu cầu người sử dụng. Tùy vào điều kiện làm việc và tính toán thì người ta sẽ thiết kế một hộp giảm tốc phù hợp với công việc.

Hộp số giảm tốc dùng để giảm tốc độ vòng quay từ động cơ. Khi lắp ráp, một đầu hộp số giảm tốc được nối với động cơ (xích, đai, hoặc nối cứng), đầu còn lại của hộp giảm tốc được nối với tải.

2.1.2 Nguyên lý hoạt động của bộ truyền Cycloid



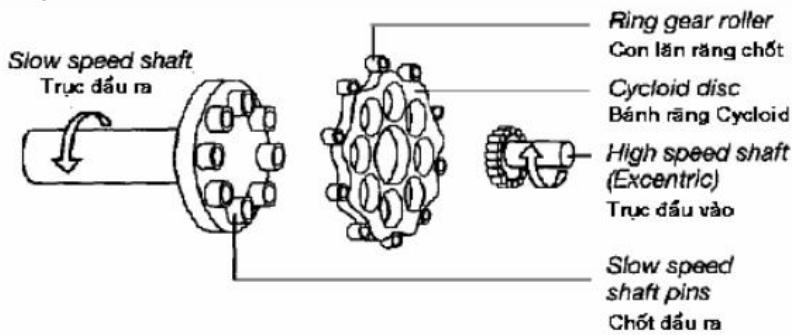
Hình 2.3 Mô tả nguyên lý làm việc của bộ truyền Cycloid

- Trục đầu vào được gắn lệch tâm với ổ lăn (thường là ổ lăn hình trụ), làm cho đĩa cycloidal lắc theo hình tròn. Đĩa cycloidal sẽ quay độc lập xung quanh ổ trực khi nó được đẩy vào bánh răng vành khuyên, tương tự như bánh răng hành tinh. Hướng quay của đĩa và đầu ra ngược với trục đầu vào.

- Số lượng chốt trên bánh răng lớn hơn số lượng chốt trên đĩa xích. Điều này làm cho đĩa cycloidal quay quanh ổ trực nhanh hơn so với trục đầu vào đang di chuyển xung quanh, tạo ra một vòng quay tổng thể theo hướng ngược lại với chuyển động quay của trục đầu vào.

- Đĩa cycloidal có các lỗ lớn hơn (bằng một lượng bằng độ lệch tâm của trục đầu vào) so với các chốt con lăn đầu ra đi bên trong chúng. Các chốt đầu ra sẽ di chuyển xung quanh trong các lỗ để đạt được chuyển động quay ổn định của trục đầu ra từ chuyển động lắc lư của đĩa hình tròn.

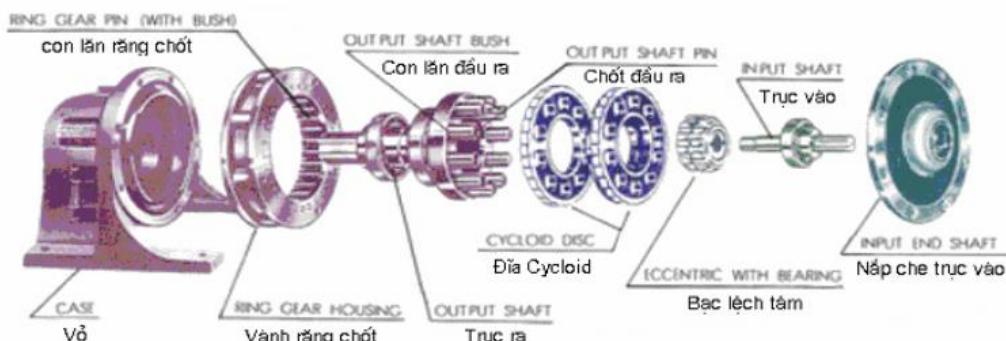
Cấu tạo cơ bản của bộ truyền Cycloid



Hình 2.4 Các nhóm chính trong bộ truyền Cycloid

Gồm 4 nhóm thành phần cơ bản:

- Trục đầu vào cùng với bạc lệch tâm và ô lăn.
- Các con lăn bánh răng chốt lắp trên vành răng chốt.
- Bánh răng Cycloid hay đĩa Cycloid (trong 1 bộ truyền có thể có 1,2 hoặc 3 đĩa Cycloid).
- Trục đầu ra.



Hình 2.5 Khai triển của một hộp giảm tốc bánh răng con lăn

2.2 Tính toán biên dạng đĩa Cycloid

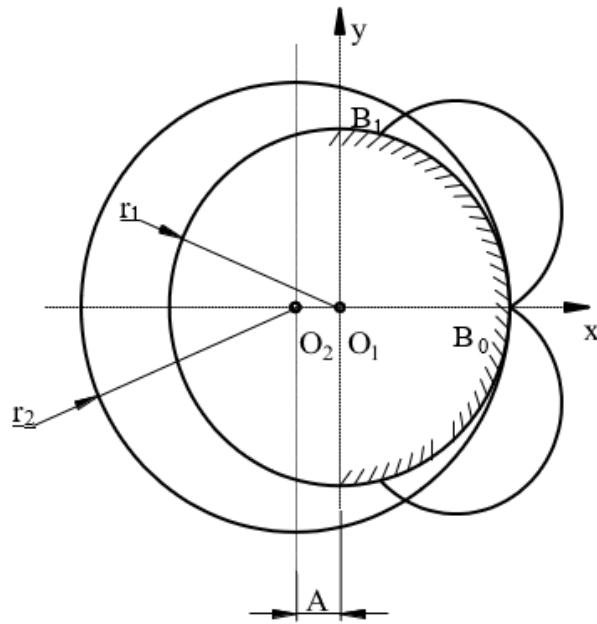
2.2.1 Khái niệm

Biên dạng Cycloid là quỹ tích của một điểm cố định trên một đường tròn khi đường tròn này lăn không trượt trên một đường thẳng hoặc đường tròn cố định khác.

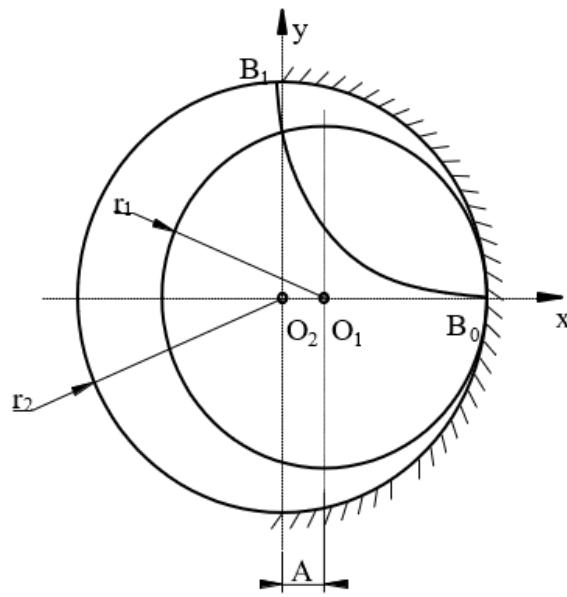
Trong các hình 2.6 và hình 2.7, có các vòng tròn bán kính r_1 và r_2 là các vòng tròn lăn (vòng tròn tâm tích). Khi lăn không trượt vòng tròn bán kính r_2 trên vòng tròn bán kính r_1 thì một điểm B nào đó cố định trên vòng tròn bán kính r_2 sẽ vẽ nên đường EpiCycloid B_0B_1 có phương trình ở dạng tham số:

$$\begin{cases} x_B = -A \cos \tau + r_2 \cdot \cos \frac{A}{r_2} \tau \\ y_B = -A \sin \tau + r_2 \cdot \sin \frac{A}{r_2} \tau \end{cases} \quad (2.2.1)$$

Với τ là tham số dịch chuyển.



Hình 2.6 Sự tạo thành đường EpiCycloid



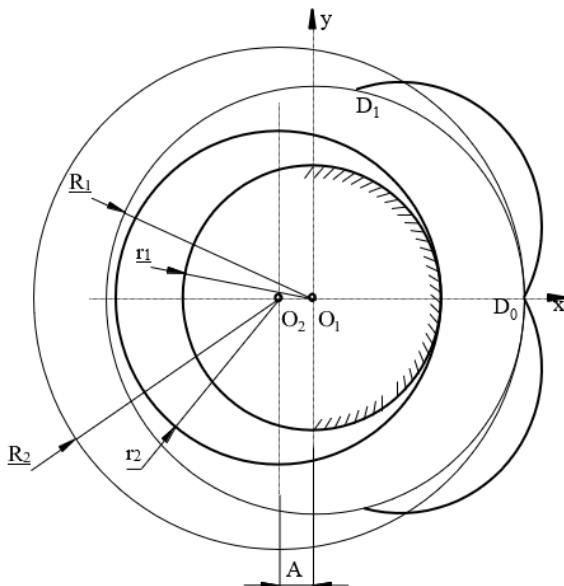
Hình 2.7 Sự tạo thành đường HypoCycloid

Khi vòng tròn bán kính r_1 lăn không trượt bên trong vòng tròn bán kính r_2 thì một điểm B nằm cố định trên đường tròn bán kính r_1 sẽ vẽ lên đường HypoCycloid B_0B_1 có phương trình ở dạng tham số:

$$\begin{cases} x_B = -A \cos \tau + r_1 \cdot \cos \frac{A}{r_2} \tau \\ y_B = -A \sin \tau + r_1 \cdot \sin \frac{A}{r_2} \tau \end{cases} \quad (2.2.2)$$

Nếu sử dụng một điểm D cũng gắn với đường tròn bán kính r_2 nhưng nằm ở bên ngoài thì điểm D sẽ vẽ lên đường EpiCycloid kéo dài D_0D_1 có phương trình dạng tham số:

$$\begin{cases} x_D = -A \cos \tau + R_2 \cdot \cos \frac{A}{r_2} \tau \\ y_D = -A \sin \tau + R_2 \cdot \sin \frac{A}{r_2} \tau \end{cases} \quad (2.2.3)$$



Hình 2.8 Sự tạo thành đường EpiCycloid kéo dài

Tương tự với trường hợp vòng tròn bán kính r_1 lăn không trượt bên trong vòng tròn bán kính r_2 sẽ thu được đường HypoCycloid kéo dài.

Nếu thay điểm D bằng con lăn có bán kính r_c thì khi 2 vòng tròn bán kính r_1 và r_2 lăn không trượt trên nhau, con lăn bán kính r_c sẽ tạo ra các đường bao cách đều đường EpiCycloid kéo dài $D'_0D'_1$ hoặc HypoCycloid kéo dài. Đó là biên dạng răng ăn khớp với con lăn răng chốt.

Phương trình đường bao cách đều đường EpiCycloid kéo dài $D'_0D'_1$:

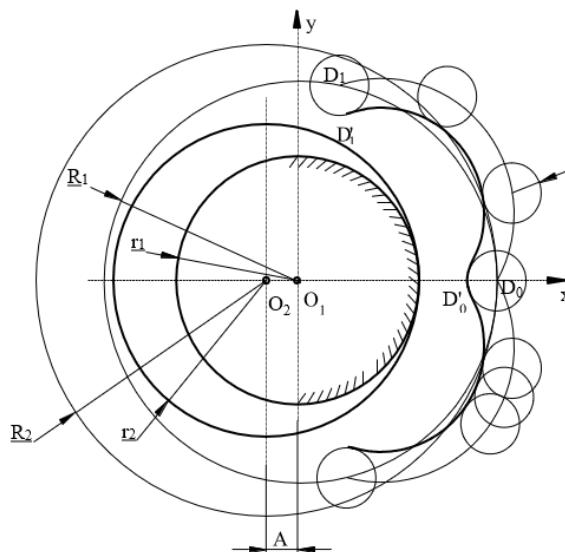
$$\begin{cases} x_{D'} = x_D - \frac{r_C y'_D}{\sqrt{x'_D{}^2 + y'_D{}^2}} \\ y_{D'} = y_D + \frac{r_C x'_D}{\sqrt{x'_D{}^2 + y'_D{}^2}} \end{cases} \quad (2.2.4)$$

Trong đó:

- x_D, y_D : tọa độ của điểm xuất phát trên đường EpiCycloid kéo dài, xác định theo công thức.
- x'_D, y'_D : đạo hàm bậc nhất của x_D và y_D , có:

$$\begin{cases} x'_D = A \sin \tau - R_2 \cdot \frac{A}{r_2} \cdot \sin \frac{A}{r_2} \tau \\ y'_D = -A \cos \tau + R_2 \cdot \frac{A}{r_2} \cdot \cos \frac{A}{r_2} \tau \end{cases} \quad (2.2.5)$$

Với r_C : bán kính con lăn răng chốt.



Hình 2.9 Sơ tạo thành biên dạng ăn khớp EpiCycloid với con lăn răng chốt

2.2.2 Thiết lập phương trình biên dạng đĩa Cycloid

Trong đồ án này chúng em tập trung đi vào nghiên cứu bộ truyền bánh răng con lăn với biên dạng đĩa Cycloid theo trường hợp ăn khớp EpiCycloid ngoại tâm tích.

2.2.2.1. Phương trình đường EpiCycloid kéo dài

Các vòng tròn bán kính r_1 và r_2 là các vòng tròn lăn (vòng tròn tâm tích) và cần thỏa mãn các điều kiện:

$$r_1 = A \cdot z_1 \quad , \quad r_2 = r_1 + A$$

Trong đó:

- z_1 là một số nguyên dương, chính là số răng đĩa Cycloid
- A là khoảng lệch tâm giữa hai vòng tròn r_1 và r_2 , $A = O_1O_2$.

Điểm B nằm cố định trên vòng tròn bán kính r_2 . Điểm D nằm cố định trên vòng tròn bán kính R_2 , vòng tròn R_2 luôn đồng tâm với vòng tròn r_2 và không chuyển động tương đối so với nhau.

Khi góc tạo bởi đường nối tâm hai vòng tròn r_1 và r_2 là O_1O_2 tạo với phương x một góc $\tau = 0^0$ thì điểm B trùng với điểm B_0 còn D trùng với điểm D_0 .

Khi vòng tròn bán kính r_2 lăn không trượt trên vòng tròn bán kính r_1 với đường nối tâm O_1O_2 tạo với phương x một góc τ thì điểm B trùng với B_1 có tọa độ (x_B, y_B) , điểm D trùng với D_1 có tọa độ (x_D, y_D) . Có chiều dài cung tròn mà r_2 đã lăn trên r_1 là:

$$S = \tau \cdot r_1 \quad (2.2.6)$$

Vòng tròn r_2 đã tự quay quanh tâm của nó một góc:

$$\phi = \angle O_1O_2B_1 = \frac{S}{r_2} = \frac{\tau \cdot r_1}{r_2} \quad (2.2.7)$$

Có tọa độ của điểm D_1 là:

$$\begin{cases} x_D = O_2D_1 \cos(\tau - \phi) - O_2O_1 \cos \tau \\ y_D = O_2D_1 \sin(\tau - \phi) - O_2O_1 \sin \tau \end{cases} \quad (2.2.8)$$

$$\text{thay : } O_2D_1 = R_2 \quad , \quad O_1O_2 = A \quad , \quad \phi = \frac{\tau \cdot r_1}{r_2}$$

có :

$$\begin{cases} x_D = R_2 \cos\left(1 - \frac{r_1}{r_2}\right)\tau - A \cos \tau \\ y_D = R_2 \sin\left(1 - \frac{r_1}{r_2}\right)\tau - A \sin \tau \end{cases} \quad (2.2.9)$$

với $A = r_2 - r_1$ có:

$$\begin{cases} x_D = R_2 \cos \frac{A}{r_2} \tau - A \cos \tau \\ y_D = R_2 \sin \frac{A}{r_2} \tau - A \sin \tau \end{cases} \quad (2.2.10)$$

Tiếp tục đổi biến , đặt:

$$\varphi = \frac{A}{r_2} \tau \quad (2.2.11)$$

Có :

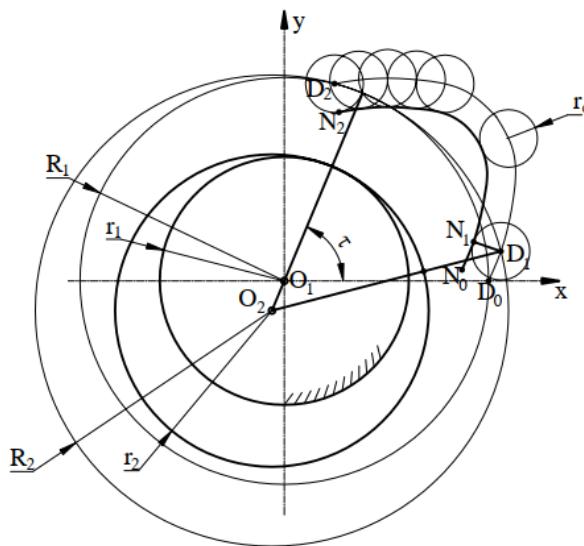
$$r_2 = (1 + z_1)A \Rightarrow \tau = (1 + z_1)\varphi \quad (2.2.12)$$

Từ đó suy ra:

$$\begin{cases} x_D = R_2 \cos \varphi - A \cos (1 + z_1)\varphi \\ y_D = R_2 \sin \varphi - A \sin (1 + z_1)\varphi \end{cases} \quad (2.2.13)$$

Để có được đường EpiCycloid kéo dài $D_0D_1D_2$ thì τ biến thiên từ 0 đến 2π , còn φ biến thiên từ 0 đến $2\pi/(1 + z_1)$, như vậy để có được đường EpiCycloid kéo dài đầy đủ thì τ biến thiên từ 0 đến $2\pi(1 + z_1)$, còn φ biến thiên từ 0 đến 2π .

2.2.2.2. Phương trình đường biên dạng đĩa Cycloid



Hình 2.10 Xây dựng biên dạng đĩa Cycloid

Thay điểm D bằng một vòng tròn bán kính r_c , đây chính là con lăn trong bộ truyền bánh răng con lăn. Khi τ biến thiên từ 0 đến 2π , quỹ đạo điểm D sẽ cho được đường EpiCycloid kéo dài $D_0D_1D_2$, khi đó đường bao họ các vòng tròn r_c sẽ tạo ra các đường bao cách đều, với đường bao phía trong là $N_0N_1N_2$. Các điểm N_0 , N_1 và N_2 được sinh ra tương ứng từ D_0 , D_1 , và D_2 . Gọi N có tọa độ (x, y) là một điểm trên đường bao cách đều tương ứng với vị trí của điểm D có tọa độ (x_D, y_D) trên đường EpiCycloid kéo dài, luôn có $DN = r_c$, do vậy có phương trình:

$$(x - x_D)^2 + (y - y_D)^2 = r_c^2 \quad (2.2.14)$$

Lấy đạo hàm theo x_D ta có:

$$x - x_D = -(y - y_D) \frac{dy_D}{dx_D} \quad (2.2.15)$$

Thay vào ta được:

$$(y - y_D)^2 = \frac{r_c^2}{1 + \left(\frac{dy_D}{dx_D}\right)^2} \quad (2.2.16)$$

Khai căn 2 vế:

$$y = y_D \pm \sqrt{\frac{r_c^2}{1 + \left(\frac{dy_D}{dx_D}\right)^2}} \quad (2.2.17)$$

Có:

$$x = x_D \mp \sqrt{\frac{r_c \cdot \frac{dy_D}{dx_D}}{1 + \left(\frac{dy_D}{dx_D}\right)^2}} \quad (2.2.18)$$

Trong đó (x_D, y_D) được xác định theo tính toán phụ thuộc τ hoặc phụ thuộc φ , dấu (-) ở phương trình tính y và (+) ở phương trình tính x biểu diễn đường bao ngoài, dấu (+) ở phương trình tính y và (-) ở phương trình tính x biểu diễn đường bao trong của họ vòng tròn bán kính r_c .

Lấy đạo hàm theo φ có:

$$\begin{cases} x'_D = -R_2 \sin \varphi + A(1 + z_1) \sin(1 + z_1) \varphi \\ y'_D = R_2 \cos \varphi - A(1 + z_1) \cos(1 + z_1) \varphi \end{cases} \quad (2.2.19)$$

$$\Rightarrow \begin{cases} x'_D = -y_D + Az_1 \sin(1 + z_1) \varphi \\ y'_D = x_D - Az_1 \cos(1 + z_1) \varphi \end{cases} \quad (2.2.20)$$

Mặt khác: $\frac{dy_D}{dx_D} = \frac{dy_D}{d\varphi} \cdot \frac{d\varphi}{dx_D} = \frac{\frac{dy_D}{d\varphi}}{\frac{dx_D}{d\varphi}} = \frac{y'_D}{x'_D}$

Ta có:

$$\begin{cases} x = x_D \mp \frac{r_c \cdot y'_D}{\sqrt{x'^D_2 + y'^D_2}} \\ y = y_D \pm \frac{r_c \cdot x'_D}{\sqrt{x'^D_2 + y'^D_2}} \end{cases} \quad (2.2.21)$$

Từ đó, khi cho φ biến thiên từ 0 đến 2π thì (x_D, y_D) sẽ vẽ lên đường EpiCycloid kéo dài đầy đủ, còn (x, y) sẽ vẽ lên đường bao họ vòng tròn bán kính r_c .

Như vậy phương trình biên dạng đĩa Cycloid cần dựng:

$$\begin{cases} x = x_D - \frac{r_c \cdot y'_D}{\sqrt{x'^D_2 + y'^D_2}} \\ y = y_D + \frac{r_c \cdot x'_D}{\sqrt{x'^D_2 + y'^D_2}} \end{cases} \quad (2.2.22)$$

2.2.3 Nhận xét

Từ lý thuyết xây dựng biên dạng đĩa Cycloid rút ra những nhận xét:

- Loại truyền động bánh răng này có đặc điểm là chỗ tiếp xúc với nhau đều nằm ngoài vùng tâm quay tức thời.

- Theo lý thuyết ăn khớp thì nếu điểm tiếp xúc nằm càng xa tâm tức thời thì vận tốc trượt giữa hai mặt răng càng lớn, do vậy hiệu suất càng thấp và càng chóng mòn. Tuy thuộc loại ăn khớp ngoài tâm tích nhưng ở truyền động bánh răng con lăn vận tốc trượt lại rất thấp bởi vì vận tốc góc tương đối là ω_{12} giữa đĩa Cycloid và các con lăn rất thấp:

$$\omega_{12} = |\omega_1 - \omega_2| = \omega_1 \left| 1 - \frac{\omega_2}{\omega_1} \right| = \omega_1 \left| \frac{z_2 - z_1}{z_2} \right|$$

Khi $z_2 - z_1 = 1$ thì: $\omega_{12} = \frac{\omega_1}{z_2}$, với z_2 là số con lăn.

- Một đặc điểm nữa của truyền động ăn khớp Cycloid nói trên, về nguyên lý, tất cả các con lăn đều đồng thời tiếp xúc mặt răng tương ứng, có tối đa một nửa số con lăn tham gia truyền lực cho nên khả năng truyền lực là rất lớn.

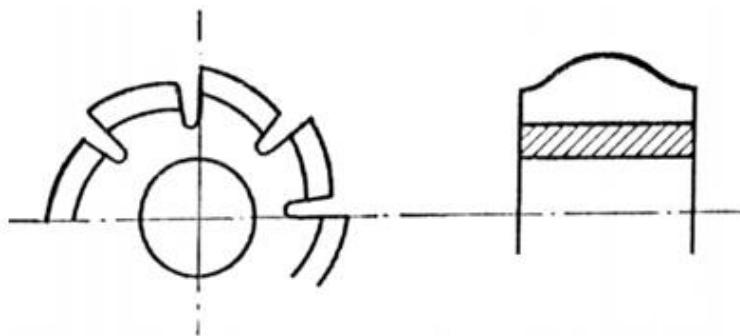
- Về mặt lý thuyết các con lăn sẽ đồng thời tiếp xúc với đĩa Cycloid nên bộ truyền bánh răng con lăn sẽ làm việc êm khi đảo chiều.

2.3 Chế tạo biên dạng đĩa Cycloid

Khi chế tạo đĩa Cycloid trong truyền động bánh răng với ăn khớp với con lăn có thể ứng dụng các phương pháp cắt bao hình. Trong trường hợp này, có thể giới thiệu một số phương pháp gia công sau:

2.3.1 Dùng dao phay đĩa với phương pháp chép hình

Trên hình 2.11, trình bày sơ đồ dao phay đĩa, biên dạng lưỡi cắt được xác định chính xác theo profin răng của đĩa Cycloid. Phương pháp này chỉ có thể chế tạo các profin ngoài của răng bánh răng. Thí dụ ứng dụng cho ăn khớp EpiCycloid đã thiết lập ở trên, các tọa độ của profin lưỡi cắt của dao phay sẽ nhận được từ phương trình.



Hình 2.11 Sơ đồ dao phay đĩa địa hình đặc biệt

Chế tạo biên dạng răng của đĩa Cycloid được thực hiện trên máy phay ngang. Để chia độ theo số răng sử dụng các đầu chia độ.

Phương pháp này được sử dụng để gia công thô răng khi gia công đơn chiếc. Gia công tinh và mài rà có thể tiến hành trên máy mài bằng phương pháp chạy rà.

2.3.2 Dùng dao phay lăn

Phương pháp cắt gọt biên dạng bằng dao phay lăn là một trong những phương pháp năng suất nhất, nhưng không đảm bảo độ chính xác cao của profin và độ bóng bề mặt.

Sau khi đã gia công bằng dao phay lăn, cũng như phương pháp chép hình, cần phải mài và chạy rà các răng của bánh răng. Dùng phương pháp phay lăn khi chế tạo các profin ngoài của răng bánh răng ăn khớp.

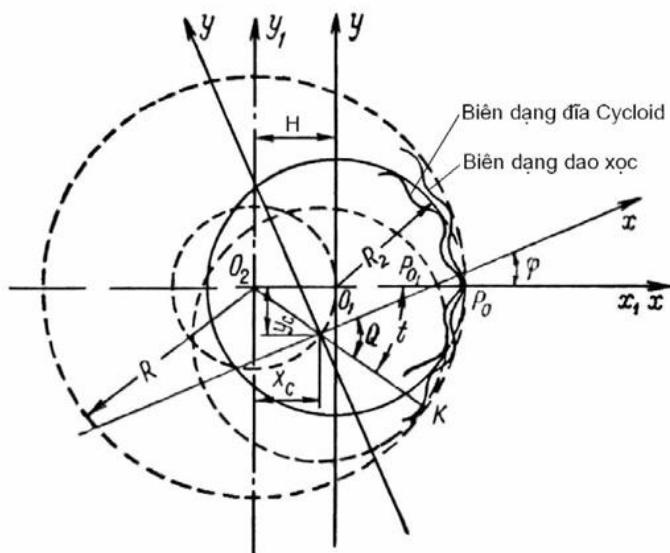
Sự đều đặn của độ cong profin răng ăn khớp Cycloid ngoại tâm tích và không bị gãy khúc tạo điều kiện thuận lợi để tạo profin bằng dao phay lăn.

Nên khi tạo biên dạng bằng phương pháp chép hình, profin của dụng cụ cắt trùng với profin của sản phẩm, thì khi gia công bằng phương pháp phay lăn, profin của dao phay lăn khác với profin răng của bánh răng được gia công. Profin của dụng cụ có thể xác định bằng đồ thị hoặc giải tích.

Khi đã biết đường biên của thanh răng khởi thủy, có thể tạo ra bề mặt hình xoắn ốc của lưỡi cắt răng của dao phay như một trực vít phù hợp với thanh răng đã cho. Sau khi nhận được bề mặt xoắn ốc, tiến đến chuyển sang hót lung và bố trí rãnh chia phoi của dao phay.

2.3.3 Gia công trên máy xọc bằng dao xọc định hình

Phương pháp này có thể được sử dụng để chế tạo cả răng trong lỗ răng ngoài của bánh răng có profin EpiCycloid và HypoCycloid. Dao xọc bản thân nó là dụng cụ nhiều lưỡi dao với các lưỡi dao răng định hình. Các lưỡi cắt cần có độ chính xác chế tạo cao.



Hình 2.12 Tính profin dao xọc bằng phương pháp giải tích

2.3.4 Cắt răng trên các máy cắt hiện đại CNC



Máy phay F4025-CNC



Máy phay F1015-CNC

Hình 2.13 Máy phay đứng CNC

Với các thế hệ máy CNC hiện nay, việc gia công profin răng đĩa Cycloid đã trở nên dễ dàng hơn nhờ khả năng điều khiển dụng cụ cắt đi theo quỹ đạo cho trước. Để gia công profin răng đĩa Cycloid có thể sử dụng các loại máy sau:

- Máy phay đứng
- Máy cắt biên dạng trực tiếp:
 - + Máy cắt bằng tia laser
 - + Máy cắt bằng tia nước áp suất cao
 - + Máy cắt dây
 - + Máy cắt bằng tia plasma
- ...



Máy cắt bằng tia nước áp suất cao



Máy cắt bằng tia Laser



Máy cắt bằng tia Plasma PAS35



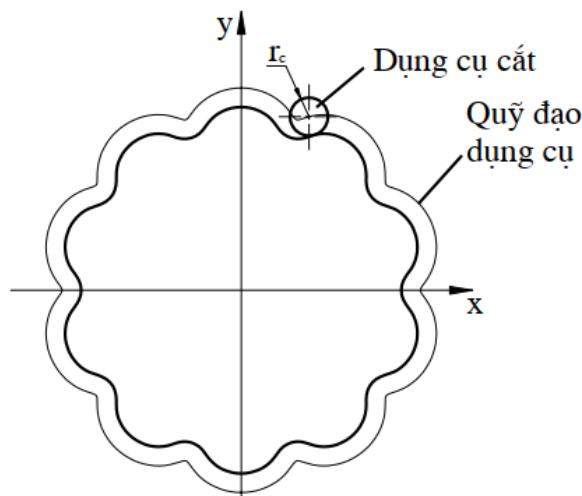
Sản phẩm cắt bằng tia Laser

Hình 2.14 Một số loại máy CNC gia công biên dạng trực tiếp

Để gia công trên các máy này cần tính toán được quỹ đạo của dụng cụ cắt chính là các đường EpiCycloid kéo dài hoặc chính là đường biên dạng của đĩa Cycloid. Nhờ sự phát triển của máy tính điện tử hiện nay thì việc tính toán các quỹ đạo này không còn là vấn đề khó khăn.

2.3.4.1. Gia công trên máy phay đứng – CNC

Dưới đây mô tả sơ đồ gia công biên dạng đĩa Cycloid trên máy phay đứng CNC. Dụng cụ cắt có thể là dao phay mặt đầu hoặc dao phay ngón có bán kính của dao là r_C chính là bán kính của con lăn sẽ ăn khớp với đĩa Cycloid cần chế tạo. Như vậy chỉ cần chạy dao theo quỹ đạo là đường EpiCycloid kéo dài được xác định ở trên, khi đó đường bao các vị trí của dụng cụ chính là biên dạng của đĩa Cycloid cần chế tạo. Phôi được gá đặt trên bàn máy. Dụng cụ cắt ở đây có thể là dao phay khi gia công tạo biên dạng, đá mài tròn khi mài bì mặt răng.



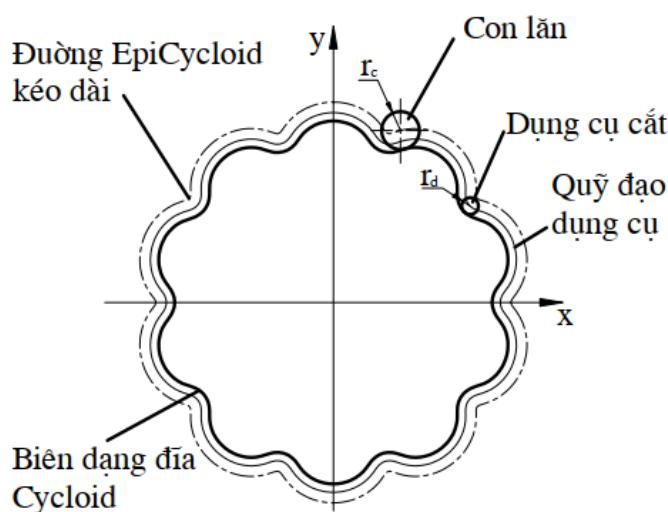
Hình 2.15 Gia công profin răng đĩa Cycloid trên máy phay đứng CNC

Để gia công theo phương pháp này đòi hỏi dụng cụ cắt có đường kính yêu cầu chính bằng đường kính con lăn sẽ ăn khớp với đĩa Cycloid, đây chính là một nhược điểm của phương pháp này do cần phải có những dụng cụ cắt khác nhau theo từng loại sản phẩm. Tuy nhiên nhược điểm này có thể khắc phục, nếu chỉ có dụng cụ cắt (dao phay, đá mài) có bán kính r_D nhỏ hơn bán kính con lăn r_C thì có thể tiến hành chạy dao theo quỹ đạo được xác định bởi phương trình:

$$\begin{cases} x = x_D - \frac{r_c \cdot y'_D}{\sqrt{x'^D_2 + y'^D_2}} \\ y = y_D + \frac{r_c \cdot x'_D}{\sqrt{x'^D_2 + y'^D_2}} \end{cases} \quad (2.3.1)$$

(x_D, y_D) là tọa độ điểm D tương ứng với vị trí (x, y) trên đường EpiCyloid kéo dài, xác định ở trên và các đạo hàm bậc nhất của chúng.

Với r_D là bán kính dao, r_C là bán kính con lăn sẽ ăn khớp với đĩa Cycloid cần gia công.

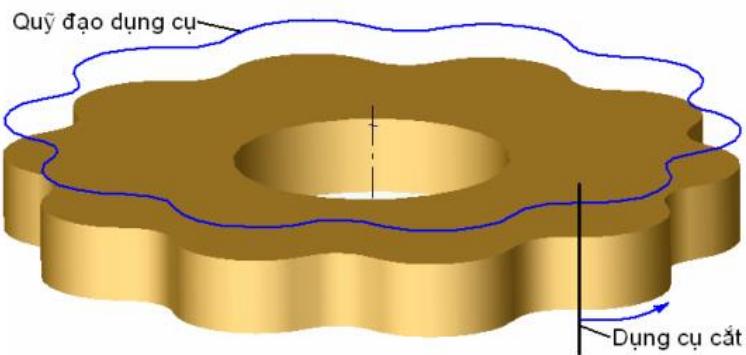


Hình 2.16 Gia công profin răng đĩa Cycloid với dụng cụ có bán kính cho trước

2.3.4.2. Gia công trên các máy cắt biên dạng trực tiếp – CNC

Với các loại máy này thực hiện di chuyển dụng cụ cắt đi theo quỹ đạo chính là biên dạng của đĩa Cycloid cần chế tạo. Do vậy việc gia công profin răng đĩa Cycloid rất đơn giản, việc chủ yếu là phải tính đủ chính xác đường biên dạng của sản phẩm để làm quỹ đạo chuyển động của dụng cụ cắt. Dụng cụ cắt ở đây có thể là:

- Dây (trong máy cắt dây)
- Tia laser (trong máy cắt laser)
- Tia nước áp suất cao (trong máy cắt bằng nước áp suất cao)
- Tia plasma (trong máy cắt bằng tia plasma)



Hình 2.17 Mô tả quá trình cắt biên dạng đĩa Cycloid trực tiếp

Tuy nhiên đối với các phương pháp này chỉ thích hợp với gia công thô bề mặt biên dạng đĩa Cycloid, để gia công tinh cần sử dụng phương pháp gia công khác để đạt độ bóng bề mặt yêu cầu.

2.4 Lắp ráp hệ thống cơ khí

2.4.1 Khái niệm về lắp ráp hệ thống cơ khí:

Quá trình sản xuất, chế tạo gồm nhiều quá trình hợp thành, lắp ráp là một quá trình cuối cùng thông qua sự kết nối một cách logic các chi tiết và các bộ phận để tạo ra sản phẩm. Theo VDI 2860 người ta định nghĩa quá trình lắp ráp như sau: “lắp ráp sự tổng hợp của tất cả các quá trình trong sản xuất để tạo nên những vật thể xác định”

Ngày nay do yêu cầu của sự phát triển sản xuất một số phái niêm trong lắp ráp được mở rộng, chẳng hạn như các quá trình Lắp ráp – và tháo dỡ tái sinh Hay Lắp ráp – tháo dỡ và kỹ thuật điều khiển. Ta có thể hình dung sự chuyển đổi của một số sản phẩm theo trình tự sau:

Sản phẩm - sản xuất - sử dụng - sự ra đời của sản phẩm mới đã được cải tiến.

Một quá trình lắp ráp phải thể hiện những yếu tố sau:

- + Đặc điểm của lắp ráp, tháo dỡ, cơ sở sản xuất thực hiện.
- + Các phương tiện cần thiết.
- + Các dự định về yêu cầu kỹ thuật, sự hợp lý của phương pháp lắp ráp (bằng tay, tự động hay phối hợp).

Chất lượng của lắp ráp phụ thuộc vào nhiều yếu tố, chẳng hạn phụ thuộc vào chất lượng của quá trình gia công cơ đối với chi tiết máy hay bộ phận lắp ráp. Việc lắp ráp chính là tạo nên sự ghép nối giữa các bề mặt của chi tiết bằng nhiều hình

thức khác nhau. Việc ghép nối còn được thực hiện bằng nguyên công bô xung cần thiết như: điều chỉnh, hiệu chỉnh, kiểm tra... Đó là lắp ráp bô xung bên cạnh lắp ráp cơ bản. Thường thời gian lắp ráp bô xung nhỏ hơn thời gian lắp ráp chung, nhưng cũng có trường hợp thời gian này chiếm tới 2/3 thời gian lắp ráp chung.

Ở hình trên trình bày cách sắp xếp và nội dung của quá trình lắp ráp.

Quá trình lắp ráp có những đặc trưng sau đây:

- + Thời gian lắp ráp chiếm một phần thời gian chế tạo, chẳng hạn trong ngành chế tạo máy và chế tạo ô tô nó chiếm từ 25% ÷ 50%, trong ngành điện tử, cơ khí chính xác chiếm từ 40 ÷ 70%.

- + Từ các chi tiết, bộ phận rời rạc được ghép nối, điều chỉnh, kiểm tra để tạo ra những sản phẩm với những chức năng sử dụng nhất định.

- + Muốn lắp ráp hiệu quả cần thực hiện những nguyên tắc có tính logic cao, sự chuẩn bị sẵn sàng về mặt tổ chức ở tất cả các lĩnh vực (sức lao động, đối tượng lao động, phương tiện lao động, thông tin lao động ...). Nếu sự chuẩn bị không tốt sẽ gây ra tổn thất của quá trình lắp ráp.

- + Quá trình lắp ráp là khâu cuối cùng tạo ra sản phẩm để có thể cung cấp sản phẩm nhanh cho thị trường. Điều đó có một ý nghĩa to lớn về mặt kinh tế - kỹ thuật đối với một cơ sở sản xuất, một xí nghiệp. Uy tín chất lượng và giá thành hợp lý sẽ bảo đảm lợi nhuận cao và lâu dài cho một xí nghiệp.

Quá trình lắp ráp có thể thực hiện bằng tay hay bằng máy hoặc là phối hợp.

2.4.2 Kỹ thuật lắp ráp

- Các phương pháp ghép nối

Bảng dưới đây trình bày các phương pháp ghép nối trong lắp ráp.

Thường tỷ lệ về khối lượng công việc giữa các phương pháp như sau:

Bằng bu long đai ốc khoảng 68%	Vòng halm khoảng 1%
Các phương pháp đặc biệt 2%	ke 29%

Bảng 2.1 Tỷ lệ khối lượng công việc

Ngược lại với quá trình gia công cơ, trong lắp ráp do tính đa dạng và tính tổng hợp có nhiều trường hợp công việc lắp ráp phải thực hiện bằng tay. Khi lắp ráp bằng tay người ta phải tạo chỗ làm việc phù hợp với điều kiện lao động.

- **Kết cấu của sản phẩm, bộ phận và chi tiết trong lắp ráp**

Qua phân tích người ta thấy rằng kết cấu của sản phẩm hay chi tiết ảnh hưởng đến 75% chi phí lắp ráp. Điều đó cho thấy ý nghĩa quan trọng của kết cấu sản phẩm đồng thời với việc xác định một hình thức lắp ráp thích hợp. Trong mối quan hệ này cần chú ý rằng hình thức lắp ráp chỉ là một trong rất nhiều quá trình tạo nên sản phẩm. Xuất phát từ nhiệm vụ tổng thể để có thể lựa chọn một hình thức lắp ráp hiệu quả. Tính hiệu quả được đánh giá bằng những phương pháp riêng.

Theo Hesse để có thể tránh được những sai sót cơ bản khi thực hiện các hình thức lắp ráp cần dựa theo 10 nguyên tắc dưới đây:

Số lượng chi tiết lắp ráp là tối thiểu, ít chủng loại, kích thước của một đơn vị lắp ráp ít chênh lệch nhiều.

Dễ dàng tìm được vị trí lắp, chặng hạn, mặt dẫn trượt nghiêng, chốt định vị, mặt dẫn hướng, mặt côn định tâm ...

Có găng thiết kế kết cấu, các chuyên động động học đơn giản.

Có găng tạo ra các mối lắp có thể tiến hành lắp tự động được.

Tránh cho dung sai dẫn trượt nhỏ, các chuỗi kích thước quá dài, có găng dùng phương pháp lắp lắn.

Cần xác định đúng chi tiết cơ sở (thường là thân máy).

Có găng thiết kế sản phẩm nhỏ gọn, không chiếm nhiều không gian nhà xưởng, nhẹ để dễ vận chuyển.

Có găng tạo các nhóm hay bộ phận ít liên quan đến nhau, dễ sửa chữa và kiểm tra.

Điều chỉnh đơn giản làm sao thực hiện được khâu bồi thường.

Các chi tiết có thể tiến hành lắp tự động. Các vị trí dùng lại hay chuyển động đủ cứng vững, tránh dùng những chi tiết kém cứng vững dễ biến dạng.

- **Tháo dỡ**

Do sự phát triển của sản xuất và sự khan hiếm dần nguồn tài nguyên thiên nhiên, sự ô nhiễm của môi trường sống, con người phải nghĩ tới việc tái sử dụng các sản phẩm đã sử dụng trong một mức độ cho phép.

Với khái niệm tháo dỡ có thể hiểu là ngược với quá trình lắp ráp hay sự tách rời các chi tiết, bộ phận ra khỏi một vật thể (sản phẩm) đã sử dụng. Có thể hình dung diễn biến đó theo một quá trình dưới đây:

Vật liệu → sản xuất → sản phẩm → sử dụng → tái sinh → vật liệu

Quá trình tháo dỡ và tái sinh có thể được thực hiện theo trình tự sau:

Tháo dỡ các bộ phận/ sản phẩm. Khi tháo dỡ cần phải :

- + Nắm vững kết cấu của sản phẩm.
- + Xác định các bộ phận, vị trí để tách, tháo ra.
- + Cách sắp xếp các chi tiết.

Rửa sạch các chi tiết cần thiết.

Kiểm tra phân loại các chi tiết theo từng nhóm:

- + Các chi tiết không cần sửa chữa nhưng vẫn dùng lại được
- + Các chi tiết có thể dùng lại được nhưng cần sửa chữa.
- + Các chi tiết không dùng lại được mà chỉ dùng để tái sinh.

Gia công các chi tiết phù hợp, hay có thể thay đổi một vài chi tiết mới.

Lắp ráp thành bộ phận sau đó kiểm tra.

Với các sản phẩm đã được sử dụng 10÷ 15 năm (thậm chí đã 30 năm) thì việc táo dỡ và tái sinh không mang tính hiệu quả. Những năm gần đây người ta nghiên cứu nguyên tắc lắp ráp và tháo dỡ theo kết cấu sản phẩm, hình dạng các bộ phận, kỹ thuật ghép nối, sự lựa chọn vật liệu.

Khi tháo dỡ cần tuân theo 3 nguyên tắc:

- + Giảm tối thiểu nguyên công không cần thiết.
 - + Rút ngắn quá trình tháo dỡ đến mức có thể.
 - + Tháo dỡ không phá huỷ.
 - + Tháo dỡ phá huỷ một phần.
- + Tháo dỡ phá huỷ hoàn toàn.

• *Lắp ráp hệ thống cơ khí*

Có nhiều phương pháp lắp ghép cơ khí. Lắp ghép bằng bu lông, hàn, lắp ghép bằng độ dôi, then... nhưng lắp ghép bằng buloong là được sử dụng nhiều nhất (chiếm 70% các mối ghép) vì những **ưu điểm** sau:

- + Cấu tạo đơn giản, mối ghép bảo đảm, kiểu ren đa dạng, dễ tháo lắp, có thể tháo lắp nhiều lần, có thể cố định vị trí bất kỳ của chi tiết, có thể chế tạo lực dọc trực lớn, giá thành tương đối thấp

Nhược điểm:

- + Tập trung ứng suất chân ren, làm giảm độ bền mỏi của ren
- + Lắp ráp hệ thống cơ khí của máy tiện thiết kế chủ yếu là lắp ráp bằng tay.
Độ chính xác lắp ráp khi hệ thống cơ khí bằng tay không cao.

CHƯƠNG 3. TỔNG QUAN VỀ HỆ THỐNG ĐIỀU KHIỂN ĐỘNG CƠ BUỚC CHO HỘP GIẢM TỐC

3.1 Tổng quan về Arduino

3.1.1 Giới thiệu về Arduino

Arduino là một nền tảng điều khiển mã nguồn mở cho phép người dùng tạo ra các dự án điện tử. Nó được thiết kế để dễ dàng sử dụng và học, với một ngôn ngữ lập trình C/C++ dễ hiểu và một số thư viện được tích hợp sẵn cho các tác vụ thông dụng như điều khiển các đầu ra và đầu vào, gửi và nhận dữ liệu qua các giao diện như UART, I2C và SPI.

Arduino có thể sử dụng trong rất nhiều dự án, từ các thiết bị điều khiển gia đình như bếp điện tử, đèn LED động và cảm biến nhiệt độ đến các dự án phức tạp hơn như robots, máy bay điều khiển tự động và hệ thống tưới cây tự động.

Arduino còn cung cấp một cộng đồng lớn của các nhà phát triển và nhà sản xuất, đảm bảo rằng các dự án có thể được hoàn thiện và cải tiến một cách dễ dàng. Ngoài ra, có rất nhiều tài liệu và hướng dẫn trực tuyến miễn phí dành cho người đê học và sử dụng Arduino một cách hiệu quả.

Từng bước, Arduino đang trở thành một công cụ quan trọng trong lĩnh vực điện tử và công nghệ. Nó đã giúp giảm thiểu sự phức tạp trong việc phát triển các dự án điện tử và cho phép người dùng tạo ra những thứ mới và sáng tạo.

Tổng kết, Arduino là một công cụ tuyệt vời cho những ai muốn tìm hiểu và phát triển các dự án điện tử. Nó cung cấp một nền tảng mạnh mẽ, dễ sử dụng và linh hoạt cho việc phát triển các dự án, và cung cấp một cộng đồng để hỗ trợ và chia sẻ kinh nghiệm.

3.1.2 Phần cứng của Arduino Board

Phần cứng của Arduino bao gồm một vi điều khiển chính, một bộ nhớ flash, một bộ chuyển đổi analog-digital (ADC), một số chân đầu ra và đầu vào, và một số chân trung tâm để kết nối với các thiết bị ngoài. Các chân đầu ra có thể được sử dụng để điều khiển các thiết bị như đèn LED, cảm biến, và đồng hồ. Các chân đầu vào có thể được sử dụng để đọc giá trị từ các cảm biến hoặc điều khiển từ bàn phím hoặc các nút bấm.

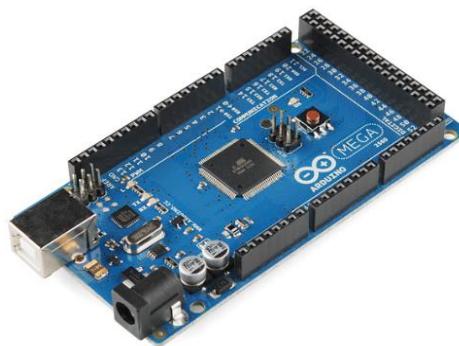
Vi điều khiển chính trong Arduino cung cấp những thao tác cơ bản cho việc xử lý dữ liệu và điều khiển các thiết bị. Bộ nhớ flash lưu trữ chương trình mà người dùng ghi vào Arduino, cho phép vi điều khiển thực hiện các tác vụ được yêu cầu. Bộ chuyển đổi ADC cho phép vi điều khiển đọc giá trị từ các cảm biến analog và chuyển chúng sang dạng số.

Dưới đây là một số loại board Arduino phổ biến:

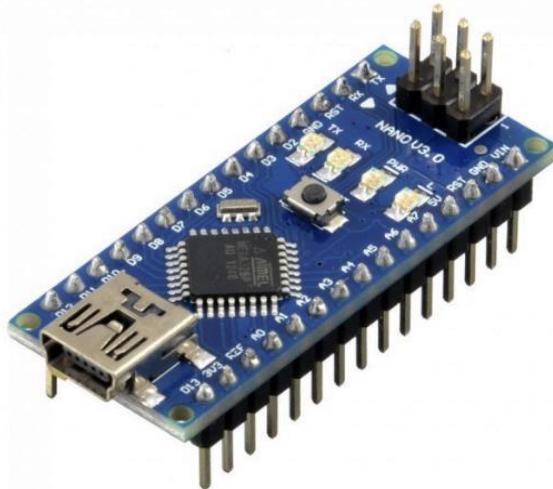
- + Arduino Uno
- + Arduino Mega
- + Arduino Nano
- + Arduino Pro Mini
- + Arduino Leonardo
- + Arduino Micro
- + Arduino Due
- + Arduino MKR1000
- + Arduino Zero
- + Arduino Esplora



Hình 3.1 Mạch arduino Uno



Hình 3.2 Mạch Arduino Mega



Hình 3.3 Mạch arduino Nano

Đây chỉ là một số trong rất nhiều loại board Arduino khác, mỗi loại có thể có một số tính năng và giải pháp khác nhau cho nhu cầu của người dùng.

Với mức giá hợp lý và khả năng tùy chỉnh cao, Arduino đang trở thành một trong những nền tảng điều khiển phổ biến nhất trên thế giới. Nó được sử dụng trong rất nhiều lĩnh vực, từ điều khiển giải trí đến điều khiển các hệ thống tự động hóa. Arduino cũng được sử dụng trong giáo dục và giảng dạy để giúp học sinh và sinh viên tìm hiểu về các kiến thức cơ bản về điều khiển và lập trình.

Tổng kết, Arduino là một nền tảng điều khiển phổ biến và linh hoạt, được thiết kế để hỗ trợ người dùng trong việc phát triển các dự án điều khiển và tự động hóa. Nó cung cấp một giao diện đơn giản và dễ sử dụng để kết nối với các thiết bị, và có một kho lớn tài nguyên và thư viện để hỗ trợ phát triển. Arduino được sử dụng rộng rãi trong nhiều lĩnh vực, từ giải trí đến giáo dục, và cung cấp một giải pháp tiết kiệm chi phí cho những ai muốn tìm hiểu về điều khiển và lập trình.

3.1.3 Lập trình Arduino IDE

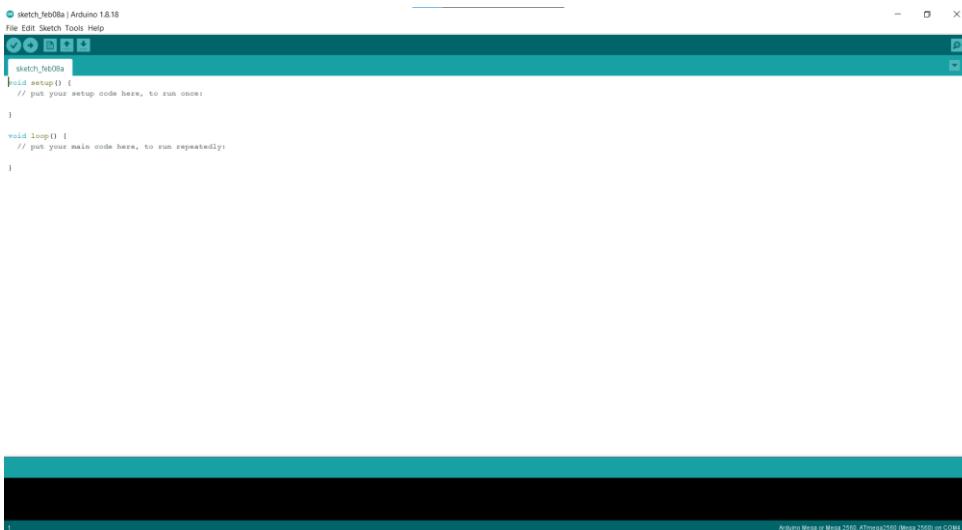
Arduino IDE là Arduino Integrated Development Environment (IDE) là một môi trường phát triển tích hợp để viết, chạy và gỡ lỗi mã cho các dự án Arduino. Nó cho phép người dùng gửi mã trực tiếp vào board Arduino và theo dõi các giá trị từ các cảm biến và thiết bị được kết nối. Arduino IDE cung cấp một giao diện đồ họa dễ sử dụng với các tính năng như lập trình, chẩn đoán lỗi, gửi mã và theo dõi dữ liệu. Nó hỗ trợ nhiều ngôn ngữ lập trình và cung cấp một cộng đồng lớn của các nhà phát triển và nhà phát triển phần mềm để hỗ trợ và chia sẻ kinh nghiệm.

Với Arduino IDE, bạn có thể dễ dàng viết mã cho các dự án của mình, chạy thử và sửa chữa các lỗi trong quá trình phát triển. Nó cung cấp một số công cụ tiện ích như gửi mã trực tiếp vào board Arduino, theo dõi các giá trị từ các cảm biến và thiết bị được kết nối và gỡ lỗi mã. Với Arduino IDE, cả người mới và chuyên gia có thể dễ dàng tạo ra các dự án đa dạng về điều khiển, đồ họa và thiết bị điện tử.



Hình 3.4 Phần mềm Arduino IDE

Ngoài ra, Arduino IDE còn hỗ trợ một số tính năng nâng cao như tích hợp với các thư viện, tạo các sketch (tập lệnh) mẫu, chẩn đoán lỗi trong mã và tự động hoàn thiện mã. Điều này giúp cho việc phát triển trở nên dễ dàng và nhanh hơn, đặc biệt là cho người mới bắt đầu.

*Hình 3.5 Giao diện của Arduino IDE*

Trong khi có rất nhiều công cụ phát triển khác có sẵn để hỗ trợ việc phát triển cho board Arduino, nhưng Arduino IDE vẫn là một trong những công cụ phổ biến nhất vì nó cung cấp một giao diện đồ họa dễ sử dụng và một cộng đồng lớn của nhà phát triển. Nếu bạn muốn bắt đầu phát triển cho board Arduino, hãy tải xuống và sử dụng Arduino IDE để tạo ra các dự án độc đáo và sáng tạo.

Tải về và cài đặt Arduino IDE rất dễ dàng, bạn có thể tải về trực tiếp từ trang web của Arduino hoặc từ các nguồn khác như Softonic hoặc SourceForge. Sau khi cài đặt, bạn có thể mở Arduino IDE và bắt đầu tạo mới hoặc mở các dự án đã có. Giao diện trực quan và dễ sử dụng sẽ giúp bạn dễ dàng hiểu và sử dụng các tính năng của công cụ này.

Arduino IDE còn cung cấp một số tính năng tùy chỉnh như chỉnh sửa giao diện, cài đặt các thư viện và tùy chỉnh các tùy chọn cho việc biên dịch và gửi mã đến board. Bạn có thể tùy chỉnh các tùy chọn này theo nhu cầu của dự án của mình.

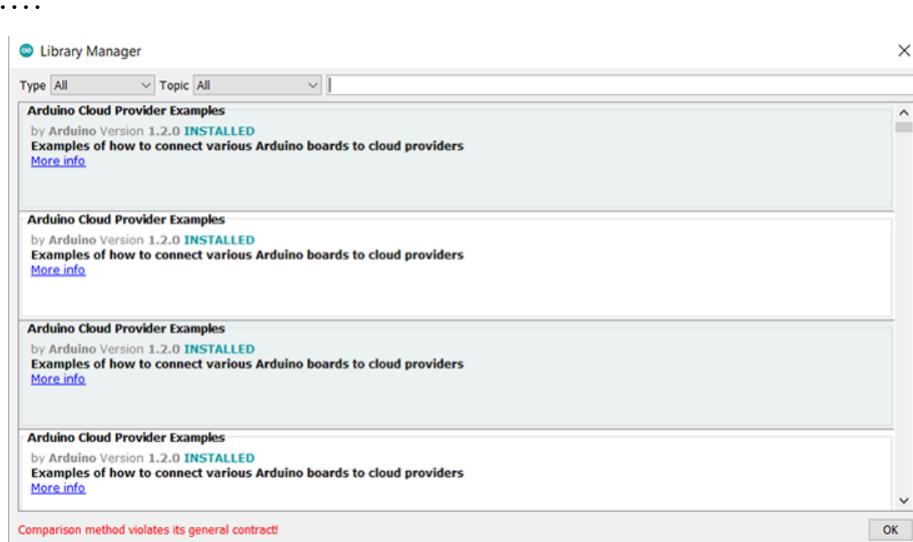
Tổng quát, Arduino IDE là một công cụ phát triển tuyệt vời để phát triển các dự án cho board Arduino. Nó cung cấp một giao diện đồ họa dễ sử dụng và nhiều tính năng mạnh mẽ để giúp bạn phát triển dự án một cách nhanh và dễ dàng. Hãy tải xuống và sử dụng nó ngay hôm nay để tạo ra các dự án sáng tạo và độc đáo của riêng bạn.

3.1.4 Thư viện lập trình trên Arduino IDE

Trong Arduino IDE, các thư viện là những tập hợp các hàm, biến và tập lệnh mà các lập trình viên có thể sử dụng để tăng tính năng của board và giảm thời gian

phát triển dự án. Một số ví dụ của các thư viện phổ biến trong Arduino IDE bao gồm:

- + Wire Library: Thư viện này giúp cho các lập trình viên sử dụng chuẩn giao tiếp I2C trên board.
 - + LiquidCrystal Library: Thư viện này giúp cho các lập trình viên sử dụng màn hình LCD.
 - + Servo Library: Thư viện này giúp cho các lập trình viên điều khiển động cơ servo.
 - + SPI Library: Thư viện này giúp cho các lập trình viên sử dụng chuẩn giao tiếp SPI trên board.
 - + SD Library: Thư viện này giúp cho các lập trình viên sử dụng thẻ nhớ SD trên board.
-



Hình 3.6 Giao diện thư viện của Arduino IDE

Để sử dụng một thư viện, bạn cần phải cài đặt nó trong Arduino IDE và gọi nó trong chương trình của mình bằng cách sử dụng từ khóa "include". Thực ra, các thư viện trên Arduino IDE có thể giúp cho các lập trình viên tăng tính linh hoạt và tối ưu hóa thời gian phát triển dự án của họ. Ngoài ra, các thư viện cũng giúp cho các lập trình viên giảm số lượng mã nguồn cần viết bằng cách sử dụng các hàm và biến đã được xây dựng sẵn.

Các thư viện trên Arduino IDE có thể được tìm kiếm và tải xuống từ trang web chính thức của Arduino hoặc từ các trang web khác. Để sử dụng một thư viện,

bạn cần phải cài đặt nó trong Arduino IDE và gọi nó trong chương trình của mình bằng cách sử dụng từ khóa "include".

Tóm lại, các thư viện trên Arduino IDE là một phần quan trọng của môi trường lập trình Arduino và có thể giúp cho các lập trình viên tăng tính linh hoạt và giảm thời gian phát triển dự án.

3.1.5 Arduino MEGA 2560

Arduino MEGA 2560

Các thành phần của board bao gồm nguồn cung cấp, vi điều khiển, driver điều khiển motor và motor. Thành phần khá quan trọng trong mạch điều khiển đó chính là vi điều khiển Arduino MEGA 2560.

Đây là một bo mạch được tích hợp nhiều tính năng nổi bật. Tính năng đầu tiên là thiết kế hệ thống I / O lớn với 16 bộ chuyển đổi tương tự và 54 bộ chuyển đổi digital hỗ trợ UART và các chế độ giao tiếp khác.

Thêm vào đó, Arduino Mega 2560 có sẵn RTC và các tính năng khác như bộ so sánh, timer, ngắt để điều khiển hoạt động, tiết kiệm điện năng và tốc độ nhanh hơn với xung thạch anh 16 Mhz.

Các tính năng khác bao gồm hỗ trợ JTAG để lập trình, gỡ lỗi và xử lý sự cố. Với bộ nhớ FLASH lớn và SRAM, bo này có thể xử lý chương trình hệ thống lớn một cách dễ dàng.

Nó cũng tương thích với các loại bo mạch khác nhau như tín hiệu mức cao (5V) hoặc tín hiệu mức thấp (3.3V) với chân nạp I / O. Brownout và watchdog giúp hệ thống đáng tin cậy và mạnh mẽ hơn. Nó hỗ trợ ICSP cũng như lập trình vi điều khiển USB với PC.

Arduino Mega 2560 là một sự thay thế của Arduino Mega cũ. Nó thường được sử dụng cho các dự án phức tạp.

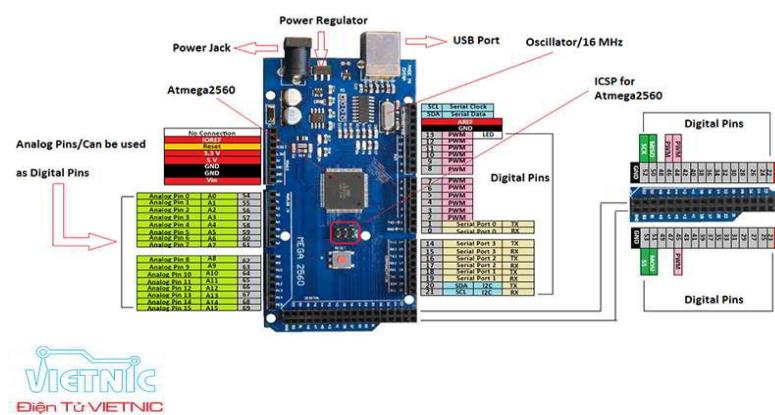
Các Đặc Điểm Kỹ Thuật:

Arduino Mega	Tính năng, đặc điểm
Ví điều khiển	AVR ATmega 2560 (8bit)
Nguồn cung cấp	7-12V (Bộ điều chỉnh sẵn có cho bộ điều khiển)
Số chân I/O số	54
Số chân I/O tương tự	16
Xung clock	16 MHz (nhà sản xuất cài đặt là 1MHz)
Bộ nhớ flash	128 KB
SRAM	8 KB
Giao tiếp	USB (Lập trình với ATmega 8), ICSP (lập trình), SPI, I2C và USART
Bộ Timer	2 (8bit) + 4 (16bit) = 6 Timer
PWM	12 (2-16 bit)
ADC	16 (10 bit)
USART	4
Ngắt thay đổi chân	24

Bảng 3.1 Đặc tính kỹ thuật Arduino MEGA 2560

- Sơ đồ chân Arduino MEGA 2560

Sơ đồ chân Arduino Mega2560



Hình 3.7 Sơ đồ chân arduino MEGA 2560

3.2 Tổng quan về Động cơ bước NEMA 23

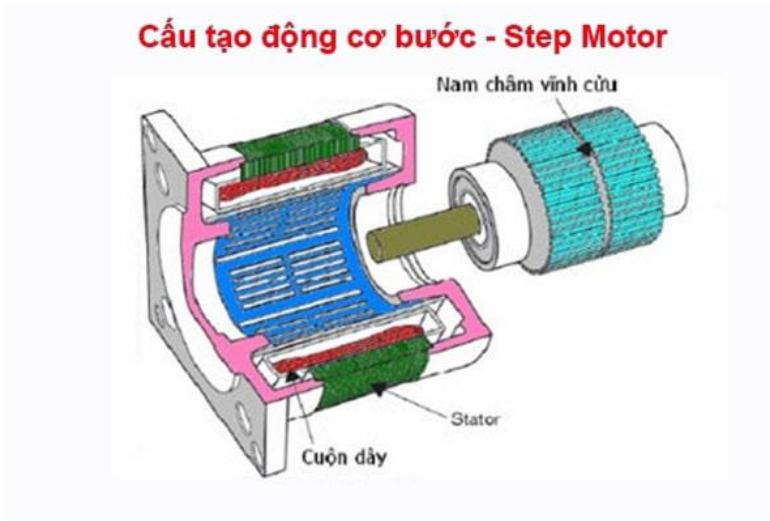


Hình 3.8 Động cơ bước Nema 23

- *Khái niệm chung :*

Động cơ bước hay còn gọi là step motor là động cơ điện DC không chổi than chia toàn bộ vòng quay thành một số bước bằng nhau. Vị trí của động cơ có thể được lệnh di chuyển và giữ ở một trong các bước này mà không cần bất kỳ cảm biến vị trí nào để phản hồi (bộ điều khiển vòng hở), miễn là động cơ có kích thước phù hợp với ứng dụng về mô men xoắn và tốc độ. Động cơ từ trờ chuyển mạch là động cơ bước rất lớn với số lượng cực giảm và thường được chuyển mạch vòng kín.

- *Cấu tạo và nguyên lý hoạt động của động cơ bước:*



Hình 3.9 Cấu tạo cơ bản StepMotor

StepMotor có cấu tạo như sau:

- 1 Rotor là một dãy các lá nam châm vĩnh cửu được xếp chồng lên nhau một cách cẩn thận. Trên các lá nam châm này lại chia thành các cặp cực xếp đối xứng nhau.

- Stato được tạo bằng sắt từ được chia thành các rãnh để đặt cuộn dây.

Cách hoạt động:

- Động cơ DC có chổi than quay liên tục khi điện áp DC được đặt vào các cực của chúng. Động cơ bước được biết đến với đặc tính chuyển đổi một chuỗi xung đầu vào (thường là sóng vuông) thành một số gia tốc được xác định chính xác ở vị trí quay của trục. Mỗi xung làm trục quay đi một góc cố định.

- Động cơ bước thực sự có nhiều nam châm điện "có răng" được bố trí như một stato xung quanh một rotor trung tâm, một miếng sắt hình bánh răng. Các nam châm điện được cung cấp năng lượng bởi mạch điều khiển bên ngoài hoặc bộ điều khiển vi mô. Để làm cho trục động cơ quay, đầu tiên, một nam châm điện được cung cấp năng lượng, nam châm này sẽ hút các răng của bánh răng bằng từ tính. Khi các răng của bánh răng thẳng hàng với nam châm điện đầu tiên, chúng sẽ lệch một chút so với nam châm điện tiếp theo. Điều này có nghĩa là khi nam châm điện tiếp theo được bật và nam châm điện đầu tiên bị tắt, bánh răng sẽ quay nhẹ để ăn khớp với nam châm tiếp theo. Từ đó quá trình được lặp lại. Mỗi phép quay một phần được gọi là một "bước", với một số nguyên của các bước thực hiện một vòng quay đầy đủ. Theo cách đó, động cơ có thể được quay theo một góc chính xác.

- Sự sắp xếp tròn của các nam châm điện được chia thành các nhóm, mỗi nhóm được gọi là một pha và có số lượng nam châm điện bằng nhau trong mỗi nhóm. Số lượng nhóm được chọn bởi nhà thiết kế động cơ bước. Các nam châm điện của mỗi nhóm được đặt xen kẽ với các nam châm điện của các nhóm khác để tạo thành một kiểu sắp xếp thống nhất. Ví dụ: nếu động cơ bước có hai nhóm được xác định là A hoặc B và có tổng cộng mười nam châm điện, thì kiểu phân nhóm sẽ là ABABABABAB.

- Các nam châm điện trong cùng một nhóm đều được cung cấp năng lượng cùng nhau. Do đó, động cơ bước có nhiều pha hơn thường có nhiều dây dẫn (hoặc dây dẫn) hơn để điều khiển động cơ.

- *Phân loại động cơ bước:*

Cách 1: Phân loại động cơ Step theo số pha động cơ

- Động cơ Step 2 pha tương ứng với góc bước 1.8 độ
- Động cơ Step 3 pha tương ứng với góc bước 1.2 độ
- Động cơ Step 5 pha tương ứng với góc bước 0.72 độ

Cách 2: Phân loại động cơ bước theo rotor

- Động cơ có rotor được tác dụng bằng dây quấn hoặc nam châm vĩnh cửu
- Động cơ thay đổi từ trở. Đây là loại động cơ có rotor không được tác động nhưng có phần tử cảm ứng

Cách 3: Phân loại theo cực của động cơ

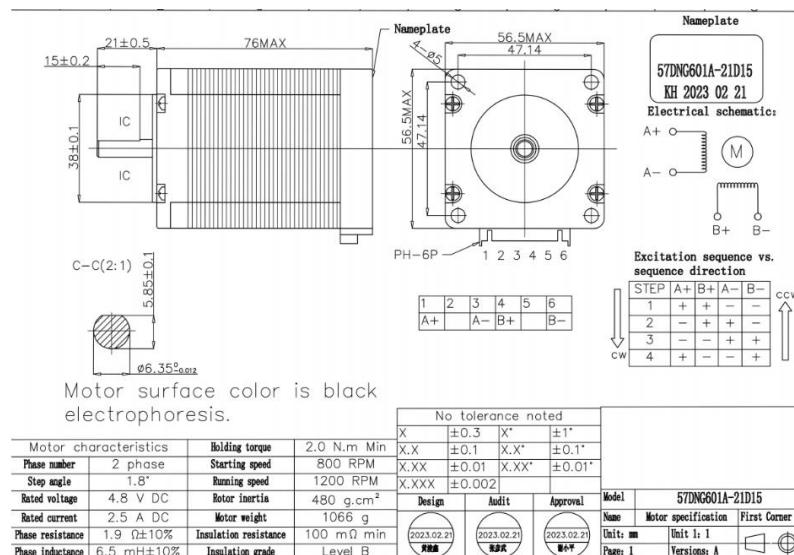
- Động cơ đơn cực
- Động cơ lưỡng cực

Ta có 1 vài thông số về động cơ bước Nema 23 :

- Góc bước $1,8^\circ$
- Chiều dài mặt bích $57 \times 57\text{mm}$
- Bước chính xác $\pm 5\%$
- Nhiệt độ môi trường - $20^\circ\text{C} \sim +50^\circ\text{C}$
- Lực hướng tâm tối đa 75N
- Lực dọc trục tối đa 15N

Model (No.)	Motor length L(mm)	Moment (N.m)	Current (A)	Resistance (Ω)	Moment of inertia (g.cm ²)	Weight (kg)
57BYGH41-204	41	0.6	2.0	1.2	150	0.47
57BYGH45-254	45	0.7	2.5	1.0	190	0.52
57BYGH51-254	51	1.0	2.5	1.2	230	0.59
57BYGH56-304	56	1.2	3.0	0.8	280	0.68
57BYGH64-304	64	1.4	3.0	1.5	380	0.85
57BYGH76-404	76	2.0	4.0	0.7	480	1.05
57BYGH84-404	82	2.3	4.2	1.0	560	1.15
57BYGH100- 424	100	2.6	4.2	0.8	680	1.4
57BYGH112- 424	112	3.0	4.2	1.4	800	1.5

Bảng 3.2 Bảng thông số một số động cơ bước

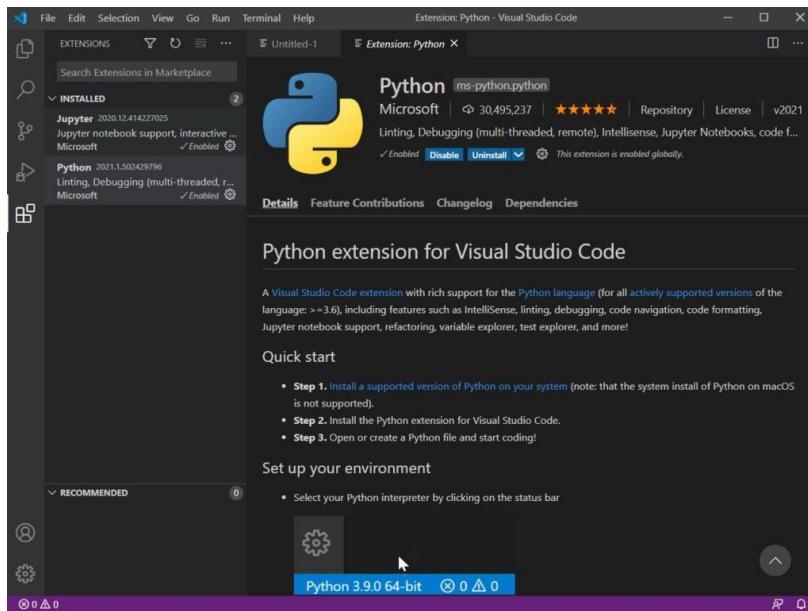


Bảng 3.3 Bảng thông số một số loại động cơ bước

3.3 Tổng quan về C# - Visual Studio

3.3.1 Tổng quan về Visual Studio

Microsoft Visual Studio là một môi trường phát triển tích hợp (IDE) từ Microsoft. Microsoft Visual Studio còn được gọi là "Trình soạn thảo mã nhiều người sử dụng nhất thế giới", được dùng để lập trình C++ và C# là chính. Nó được sử dụng để phát triển chương trình máy tính cho Microsoft Windows, cũng như các trang web, các ứng dụng web và các dịch vụ web. Visual Studio sử dụng nền tảng phát triển phần mềm của Microsoft như Windows API, Windows Forms, Windows Presentation Foundation, Windows Store và Microsoft Silverlight. Nó có thể sản xuất cả hai ngôn ngữ máy và mã số quản lý.



Hình 3.10 Giao diện cơ bản của Visual Studio

Visual Studio bao gồm một trình soạn thảo mã hỗ trợ IntelliSense cũng như cài tiền mã nguồn. Trình gõ lỗi tích hợp hoạt động cả về trình gõ lỗi mức độ mã nguồn và gõ lỗi mức độ máy. Công cụ tích hợp khác bao gồm một mẫu thiết kế các hình thức xây dựng giao diện ứng dụng, thiết kế web, thiết kế lớp và thiết kế gian đồ cơ sở dữ liệu. Nó chấp nhận các plug-in nâng cao các chức năng ở hầu hết các cấp bao gồm thêm hỗ trợ cho các hệ thống quản lý phiên bản (như Subversion) và bổ sung thêm bộ công cụ mới như biên tập và thiết kế trực quan cho các miền ngôn ngữ cụ thể hoặc bộ công cụ dành cho các khía cạnh khác trong quy trình phát triển phần mềm.

Visual Studio hỗ trợ nhiều ngôn ngữ lập trình khác nhau và cho phép trình biên tập mã và gõ lỗi để hỗ trợ (mức độ khác nhau) hầu như mọi ngôn ngữ lập trình. Các ngôn ngữ tích hợp gồm có C, C++ và C++/CLI (through qua Visual C++), VB.NET (through qua Visual Basic.NET), C# (through qua Visual C#) và F# (như của Visual Studio 2010). Hỗ trợ cho các ngôn ngữ khác như J++/J#, Python và Ruby thông qua dịch vụ cài đặt riêng rẽ. Nó cũng hỗ trợ XML/XSLT, HTML/XHTML, JavaScript và CSS.

Microsoft cung cấp phiên bản "Express" (đối với phiên bản Visual Studio 2013 trở về trước) và "Community" (đối với bản Visual Studio 2015 trở về sau) là phiên bản miễn phí của Visual Studio.

Tính năng

- Biên tập mã

Giống như bất kỳ IDE khác, nó bao gồm một trình soạn thảo mã hỗ trợ tô sáng cú pháp và hoàn thiện mã bằng cách sử dụng IntelliSense không chỉ cho các biến, hàm và các phương pháp mà còn các cấu trúc ngôn ngữ như vòng điều khiển hoặc truy vấn. IntelliSense được hỗ trợ kèm theo cho các ngôn ngữ như XML, Cascading Style Sheets và JavaScript khi phát triển các trang web và các ứng dụng web. Các đề xuất tự động hoàn chỉnh được xuất hiện trong một hộp danh sách phủ lén trên đỉnh của trình biên tập mã. Trong Visual Studio 2008 trở đi, nó có thể được tạm thời hiển thị suốt khi xem mã che khuất bởi nó. Các trình biên tập mã được sử dụng cho tất cả các ngôn ngữ được hỗ trợ.

Các trình biên tập mã Visual Studio cũng hỗ trợ cài đặt dấu trang trong mã để điều hướng nhanh chóng. Hỗ trợ điều hướng khác bao gồm thu hẹp các khối mã lệnh và tìm kiếm gia tăng, ngoài việc tìm kiếm văn bản thông thường và tìm kiếm Biểu thức chính quy. Các trình biên tập mã cũng bao gồm một bìa kẹp đa mục và một danh sách công việc. Các trình biên tập mã hỗ trợ lưu lại các đoạn mã được lặp đi lặp lại nhằm chèn vào mã nguồn sử dụng về sau. Một công cụ quản lý cho đoạn mã được xây dựng là tốt. Những công cụ này nổi lên như các cửa sổ trôi nổi có thể được thiết lập để tự động ẩn khi không sử dụng hoặc neo đậu đến các cạnh của màn hình. Các trình biên tập mã Visual Studio cũng hỗ trợ cài tiến mã nguồn bao gồm tham số sắp xếp lại, biến và phương pháp đổi tên, khai thác và đóng gói giao diện các lớp thành viên bên trong những trạng thái giữa những thứ khác.

Visual Studio có tính năng biên dịch nền (còn gọi là biên dịch gia tăng). Như mã đang được viết, Visual Studio biên dịch nó trong nền để cung cấp thông tin phản hồi về cú pháp và biên dịch lỗi, được đánh dấu bằng một gạch dưới gợn sóng màu đỏ. Biên dịch nền không tạo ra mã thực thi, vì nó đòi hỏi một trình biên dịch khác hơn là để sử dụng tạo ra mã thực thi. Biên dịch nền ban đầu được giới thiệu với Microsoft Visual Basic nhưng bây giờ đã được mở rộng cho tất cả các ngôn ngữ.

- Trình gỡ lỗi

Visual Studio có một trình gỡ lỗi hoạt động vừa là một trình gỡ lỗi cấp mã nguồn và là một trình gỡ lỗi cấp máy. Nó hoạt động với cả hai mã quản lý cũng như ngôn ngữ máy và có thể được sử dụng để gỡ lỗi các ứng dụng được viết bằng các ngôn ngữ được hỗ trợ bởi Visual Studio. Ngoài ra, nó cũng có thể đính kèm theo quy trình hoạt động và theo dõi và gỡ lỗi những quy trình. Nếu mã nguồn cho quá trình hoạt động có sẵn, nó sẽ hiển thị các mã như nó đang được chạy. Nếu mã nguồn không có sẵn, nó có thể hiển thị các tháo gỡ. Các Visual Studio debugger cũng có thể tạo bối cảnh nhớ cũng như tải chúng sau để gỡ lỗi. Các chương trình đa luồng cao cấp cũng được hỗ trợ. Trình gỡ lỗi có thể được cấu hình sẽ được đưa ra khi một ứng dụng đang chạy ngoài Visual Studio bị treo môi trường.

Trình gỡ lỗi cho phép thiết lập các breakpoint (mà cho phép thực thi được tạm thời dừng lại tại một vị trí nhất định) và watch (trong đó giám sát các giá trị của biến là việc thực hiện tiến bộ). Breakpoint có thể có điều kiện, nghĩa là chúng được kích hoạt khi điều kiện được đáp ứng. Mã có thể được biểu diễn, tức là chạy một dòng (của mã nguồn) tại một thời điểm. Nó có hoặc là bước sang các chức năng để gỡ lỗi bên trong nó, hoặc là nhảy qua nó, tức là, việc thực hiện các chức năng không có sẵn để kiểm tra thủ công. Trình gỡ lỗi hỗ trợ Edit and Continue, nghĩa là, nó cho phép mã được chỉnh sửa khi nó đang được sửa lỗi (chỉ có 32 bit, không được hỗ trợ trong 64 bit). Khi gỡ lỗi, nếu con trỏ chuột di chuyển lên bất kỳ biến, giá trị hiện tại của nó được hiển thị trong phần chú giải ("chú thích dữ liệu"), nơi mà nó cũng có thể được thay đổi nếu muốn. Trong quá trình viết mã, các trình gỡ lỗi của Visual Studio cho phép một số chức năng được gọi ra bằng tay từ cửa sổ công cụ Immediate. Các thông số cho phương thức được cung cấp tại các cửa sổ Immediate.

- Thiết kế

Windows Forms Designer: được sử dụng để xây dựng GUI sử dụng Windows Forms; bố trí có thể được xây dựng bằng các nút điều khiển bên trong hoặc khóa chúng vào bên cạnh mẫu. Điều khiển trình bày dữ liệu (như hộp văn bản, hộp danh sách, vv) có thể được liên kết với các nguồn dữ liệu như cơ sở dữ liệu hoặc truy vấn. Các điều khiển dữ liệu ràng buộc có thể được tạo ra bằng cách rê các mục từ cửa sổ nguồn dữ liệu lên bề mặt thiết kế. Các giao diện người dùng được liên kết

với mã sử dụng một mô hình lập trình hướng sự kiện. Nhà thiết kế tạo ra bằng C thăng hay VB.NET cho ứng dụng.

WPF Designer: có tên mã là Cider,[được giới thiệu trong Visual Studio 2008. Giống như Windows Forms Designer, hỗ trợ kéo và thả ẩn dụ. Sử dụng tương tác người-máy nhằm mục tiêu theo Windows Presentation Foundation. Nó hỗ trợ các chức năng WPF bao gồm kết nối dữ liệu và tự động hóa bố trí quản lý. Nó tạo ra mã XAML cho giao diện người dùng. Các tập tin XAML được tạo ra là tương thích với Microsoft Expression Design, sản phẩm thiết kế theo định hướng. Các mã XAML được liên kết với mã đang sử dụng một mô hình code-behind.

Web designer/development: Visual Studio cũng bao gồm một trình soạn thảo và thiết kế trang web cho phép các trang web được thiết kế bằng cách kéo và thả các đối tượng. Nó được sử dụng để phát triển các ứng dụng ASP.NET và hỗ trợ HTML, CSS và JavaScript. Nó sử dụng mô hình code-behind để liên kết với mã ASP.NET. Từ Visual Studio 2008 trở đi, công cụ bố trí được sử dụng bởi các nhà thiết kế web được chia sẻ với Microsoft Expression Web. Ngoài ra ASP.NET MVC Framework hỗ trợ cho công nghệ MVC là tải xuống riêng biệt và dự án ASP.NET Dynamic Data có sẵn từ Microsoft.

Class designer: Các lớp thiết kế được dùng để biên soạn và chỉnh sửa các lớp (bao gồm cả các thành viên và truy cập của chúng) sử dụng mô hình UML. Các lớp thiết kế có thể tạo ra mã phác thảo C thăng và VB.NET cho các lớp và cá phương thức. Nó cũng có thể tạo ra sơ đồ lớp từ các lớp viết tay.

Data designer: Thiết kế dữ liệu có thể được sử dụng để chỉnh sửa đồ họa giản đồ cơ sở dữ liệu bao gồm các bảng, khóa chính, khóa ngoại và các ràng buộc. Nó cũng có thể được sử dụng để thiết kế các truy vấn từ các giao diện đồ họa.

Mapping designer: Từ Visual Studio 2008 trở đi, thiết kế ánh xạ được dùng bởi Language Integrated Query để thiết kế các ánh xạ giữa các giản đồ cơ sở dữ liệu và các lớp để đóng gói dữ liệu. Các giải pháp mới từ cách tiếp cận ORM, ADO.NET Entity Framework sẽ thay thế và cải thiện các công nghệ cũ.

- Các công cụ khác

Open Tabs Browser: được sử dụng để liệt kê tất cả thẻ đang mở và chuyển đổi giữa chúng. Được viền dẫn bằng cách sử dụng CTRL+TAB.

Properties Editor: được sử dụng để chỉnh sửa các thuộc tính trong một cửa sổ giao diện bên trong Visual Studio. Nó liệt kê tất cả các thuộc tính có sẵn (gồm chỉ đọc và những thuộc tính có thể được thiết lập) cho tất cả các đối tượng bao gồm các lớp, biểu mẫu, trang web và các hạng mục khác.

Object Browser: là một không gian tên và trình duyệt lớp thư viện cho Microsoft .NET. Nó có thể được sử dụng để duyệt các không gian tên (được sắp xếp theo thứ bậc) trong Assembly (CLI). Các hệ thống phân cấp có thể hoặc không có thể phản ánh các tổ chức trong hệ thống tập tin.

Solution Explorer: theo cách nói trong Visual Studio, một giải pháp là một tập hợp các tập tin mã và các nguồn khác được sử dụng để xây dựng một ứng dụng. Các tập tin trong một giải pháp được sắp xếp theo thứ bậc, mà có thể có hoặc không thể phản ánh các tổ chức trong hệ thống tập tin. Solution Explorer được sử dụng để quản lý và duyệt các tập tin trong một giải pháp.

Team Explorer: được sử dụng để tích hợp các khả năng của Team Foundation Server, Revision Control System và là cơ sở cho môi trường CodePlex đối với dự án mã nguồn mở. Ngoài việc kiểm soát nguồn nó cung cấp khả năng xem và quản lý các công việc riêng lẻ (bao gồm cả lỗi, nhiệm vụ và các tài liệu khác) và để duyệt thống kê TFS. Nó được bao gồm như là một phần của một cài đặt TFS và cũng có sẵn để tải xuống cho Visual Studio. Team Explorer cũng có sẵn như là một môi trường độc lập duy nhất để truy cập các dịch vụ TFS.

Data Explorer: được sử dụng để quản lý cơ sở dữ liệu trên Microsoft SQL Server. Nó cho phép tạo ra và sửa đổi các bảng cơ sở dữ liệu (hoặc bằng cách ban hành các lệnh T-SQL hoặc bằng cách sử dụng các thiết kế dữ liệu). Nó cũng có thể được sử dụng để tạo các truy vấn và các thủ tục lưu trữ trong T-SQL hoặc trong Managed code thông qua SQL CLR. Có sẵn gỡ lỗi và hỗ trợ IntelliSense.

Server Explorer: công cụ được sử dụng để quản lý các kết nối cơ sở dữ liệu trên một máy tính truy cập được. Nó cũng được sử dụng để duyệt chạy Windows Services, quay thực hiện, Windows Event Log và hàng đợi tin nhắn và sử dụng chúng như một nguồn dữ liệu.

Dotfuscator Software Services Community Edition: Visual Studio bao gồm một phiên bản light của sản phẩm PreEmptive Solutions' Dotfuscator cho mã gây

rồi và giảm kích thước ứng dụng. Khởi đầu với Visual Studio 2010, phiên bản này của Dotfuscator sẽ bao gồm khả năng Runtime Intelligence cho phép tác giả thu thập cách sử dụng của người dùng cuối, hiệu suất, tính ổn định và các thông tin từ các ứng dụng của họ chạy trong sản xuất.

Text Generation Framework: Visual Studio bao gồm một khung tạo văn bản đầy đủ được gọi là Text Template Transformation Toolkit T4 cho phép Visual Studio tạo ra tập tin văn bản từ các mẫu hoặc trong IDE hoặc thông qua mã.

ASP.NET Web Site Administration Tool: công cụ quản trị trang web ASP.NET cho phép cấu hình các trang web ASP.NET.

Visual Studio Tools for Office: Công cụ Visual Studio cho Office là một SDK và một add-in cho Visual Studio bao gồm các công cụ để phát triển cho các bộ Microsoft Office. Trước đây (với Visual Studio.NET 2003 và Visual Studio 2005) đó là một SKU riêng biệt mà chỉ hỗ trợ Visual C# Visual Basic.NET hoặc đã được đưa vào Team Suite. Với Visual Studio 2008, nó không còn là một SKU riêng biệt nhưng lại kèm trong các phiên bản chuyên nghiệp và cao hơn. Một thời gian chạy riêng biệt được yêu cầu khi triển khai các giải pháp VSTO.

3.3.2 Tổng quan về ngôn ngữ lập trình Python

Python là một ngôn ngữ lập trình cao cấp, được phát triển bởi Guido van Rossum và ra mắt lần đầu vào năm 1991. Python được thiết kế để đơn giản, dễ đọc và dễ viết, với mục tiêu tạo ra một ngôn ngữ lập trình có cú pháp sạch sẽ và mã nguồn ngắn gọn.

Dưới đây là một số điểm tổng quan về Python:

- Đơn giản và dễ học: Python có cú pháp đơn giản và dễ hiểu, giúp cho người mới học lập trình dễ dàng tiếp cận. Nó cung cấp một cú pháp rõ ràng và có thể đọc được, gần giống với ngôn ngữ tự nhiên.

- Đa năng: Python là một ngôn ngữ lập trình đa năng, có thể được sử dụng trong nhiều lĩnh vực khác nhau. Nó hỗ trợ cả lập trình hướng đối tượng (OOP) và lập trình cấu trúc, và có thể sử dụng cho phát triển web, phân tích dữ liệu, trí tuệ nhân tạo, tự động hóa, và nhiều ứng dụng khác.

- Thư viện và hệ sinh thái phong phú: Python có một cộng đồng phát triển mạnh mẽ và hệ sinh thái thư viện phong phú. Có hàng ngàn thư viện và công cụ

sẵn có để giúp việc phát triển ứng dụng dễ dàng và nhanh chóng. Ví dụ: NumPy, Pandas, Matplotlib, TensorFlow, Django, Flask, và nhiều thư viện khác.

- Hỗ trợ cho nhiều nền tảng: Python có sẵn cho các nền tảng khác nhau như Windows, macOS, Linux và cũng có thể chạy trên các thiết bị di động. Điều này làm cho nó trở thành một ngôn ngữ phổ biến và dễ dàng sử dụng trên nhiều hệ điều hành và môi trường.

- Cộng đồng lớn: Python có một cộng đồng lập trình viên rộng lớn và tích cực. Cộng đồng này cung cấp sự hỗ trợ, tài liệu phong phú, và các dự án mã nguồn mở đa dạng, giúp người dùng tận dụng tối đa tiềm năng của Python.

- Tính năng bảo mật: Python có các tính năng bảo mật tích hợp, bao gồm quản lý bộ nhớ tự động và kiểm tra kiểu dữ liệu, giúp ngăn chặn các lỗ hổng bảo mật phổ biến như tràn bộ đệm và xâm nhập dữ liệu.

- Hỗ trợ cho học tập và giảng dạy: Python được sử dụng rộng rãi trong giáo dục và học tập. Nó cung cấp một cú pháp đơn giản và môi trường phát triển dễ sử dụng, giúp sinh viên và người học mới dễ dàng tiếp cận lập trình.

Python là một ngôn ngữ lập trình mạnh mẽ và linh hoạt, đã trở thành một trong những ngôn ngữ phổ biến nhất trên thế giới. Với sự đa dụng, cộng đồng mạnh mẽ và sự dễ học, Python đã trở thành lựa chọn hàng đầu cho nhiều nhà phát triển và dự án phần mềm.

Mục tiêu của việc phát triển Python

Mục tiêu chính của việc phát triển Python là cung cấp một ngôn ngữ lập trình mạnh mẽ, đơn giản và dễ sử dụng cho cộng đồng lập trình viên. Dưới đây là một số mục tiêu cụ thể của việc phát triển Python:

- Đơn giản và dễ học: Python được thiết kế để có cú pháp rõ ràng và dễ đọc, giúp người mới học lập trình dễ dàng tiếp cận và nắm bắt. Mục tiêu là tạo ra một ngôn ngữ lập trình có cú pháp sạch sẽ và dễ viết.

- Đa năng và linh hoạt: Python hướng đến việc hỗ trợ nhiều lĩnh vực và ứng dụng khác nhau. Ngôn ngữ này không chỉ hỗ trợ lập trình hướng đối tượng (OOP) và lập trình cấu trúc, mà còn có thể sử dụng cho phát triển web, trí tuệ nhân tạo, phân tích dữ liệu, tự động hóa và nhiều lĩnh vực khác.

- Hiệu suất và tốc độ: Mặc dù Python là một ngôn ngữ thông dịch, nhưng các cải tiến liên tục được thực hiện để cải thiện hiệu suất và tốc độ thực thi. Python cung cấp các công cụ và thư viện tối ưu để đạt được hiệu suất tốt trong các ứng dụng yêu cầu cao.

- Hệ sinh thái phong phú: Python có một cộng đồng phát triển mạnh mẽ và hệ sinh thái thư viện phong phú. Mục tiêu là tiếp tục phát triển và hỗ trợ các thư viện, công cụ và framework cho nhiều mục đích khác nhau, từ phân tích dữ liệu và machine learning đến web development và đồ họa.

- Hỗ trợ đa nền tảng: Python được phát triển để chạy trên nhiều nền tảng khác nhau, bao gồm Windows, macOS và các hệ điều hành Linux. Điều này giúp đảm bảo tính di động và sẵn sàng sử dụng trên các môi trường phát triển khác nhau.

- Sự thân thiện và cộng đồng: Python có một cộng đồng lập trình viên rộng lớn và hỗ trợ tích cực. Mục tiêu là duy trì một môi trường thân thiện, chia sẻ kiến thức và hỗ trợ lẫn nhau. Cộng đồng Python chú trọng vào việc phát triển và tạo ra các dự án mã nguồn mở để giúp cộng đồng lập trình viên và người dùng cuối.

Tổng quan, mục tiêu của việc phát triển Python là tạo ra một ngôn ngữ lập trình mạnh mẽ, dễ học và đa năng, đồng thời hỗ trợ cộng đồng lập trình viên và cung cấp một hệ sinh thái phong phú để thúc đẩy sự phát triển và ứng dụng của Python trong các lĩnh vực khác nhau.

3.3.3 Tổng quan về ngôn ngữ lập trình C#

C# (C Sharp) là một ngôn ngữ lập trình phổ biến và mạnh mẽ được phát triển bởi Microsoft. C# thuộc họ ngôn ngữ lập trình dựa trên Java và C++, và nó thường được sử dụng để phát triển ứng dụng máy tính, ứng dụng web, ứng dụng di động, và nhiều loại phần mềm khác trên nền tảng Windows. Dưới đây là một tổng quan về C#:

Cú pháp và Tính năng: C# kế thừa cú pháp từ ngôn ngữ C/C++, nhưng cũng có sự ảnh hưởng từ Java. C# hỗ trợ lập trình hướng đối tượng, cùng với nhiều tính năng mạnh mẽ như quản lý bộ nhớ tự động (garbage collection), đa luồng (multithreading), xử lý ngoại lệ (exception handling), và lập trình LINQ (Language Integrated Query).

Phát triển Ứng dụng Đa dạng: C# có thể được sử dụng để phát triển nhiều loại ứng dụng khác nhau, bao gồm:

- Ứng dụng máy tính (desktop applications): Sử dụng Windows Presentation Foundation (WPF) hoặc Windows Forms.

- Ứng dụng web: Sử dụng ASP.NET và ASP.NET Core để phát triển ứng dụng web.

- Ứng dụng di động: Sử dụng Xamarin hoặc Blazor để phát triển ứng dụng di động trên nhiều nền tảng.

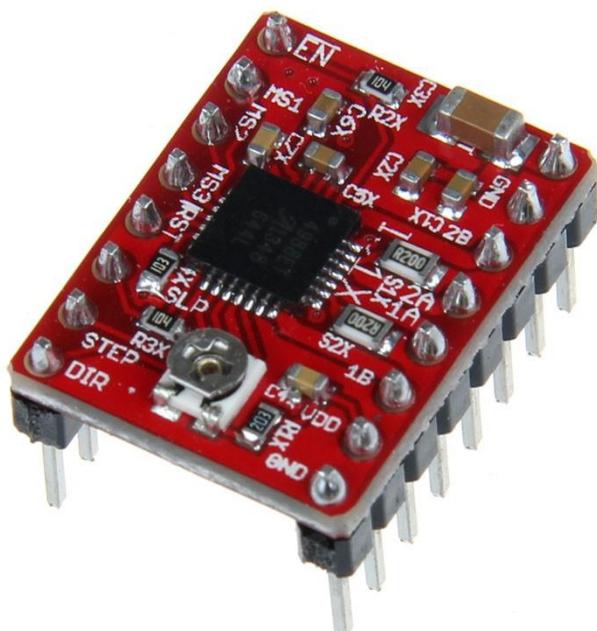
- Phần mềm dịch vụ: Sử dụng .NET Core để xây dựng các dịch vụ web, API, và ứng dụng dựa trên máy chủ.

Môi trường Phát triển: C# thường được phát triển bằng việc sử dụng môi trường phát triển tích hợp (IDE) của Microsoft, chủ yếu là Visual Studio. Visual Studio cung cấp nhiều công cụ hỗ trợ phát triển, gỡ lỗi và quản lý mã nguồn.

3.4 Tổng quan về Driver A4988, Nguồn tổ ong, Encoder

3.4.1 A4988 Stepper Motor Driver

A4988 là một loại chip điều khiển động cơ bước đơn giản và mạnh mẽ. Nó được sử dụng rộng rãi trong các ứng dụng Arduino. Với những chức năng như điều khiển dòng động cơ, điều chỉnh tốc độ, và chế độ hoạt động microstep.



Hình 3.11 Driver A4988

Một số tính năng chính của A4988 bao gồm:

- Điện áp làm việc từ 8V đến 35V, giúp tương thích với nhiều nguồn cung cấp điện khác nhau.

- Dòng làm việc lên đến 4A, cho phép điều chỉnh dòng điện đầu ra cho động cơ phù hợp với yêu cầu cụ thể.

- Hỗ trợ chế độ vận hành bán hình chữ nhật và chế độ vận hành bán hình tam giác, có thể tùy chỉnh thông qua các tín hiệu đầu vào.

- Tích hợp bảo vệ quá dòng, quá nhiệt và quá áp, giúp đảm bảo sự an toàn hoạt động của hệ thống.

- Giao thức điều khiển số bước và chiều quay rất đơn giản.

- 5 cấp điều chỉnh bước: 1; 1/2; 1/4; 1/8; và 1/16 bước.

- Có chức năng bảo vệ ngắn mạch, bảo vệ quá nhiệt, bảo vệ tụt áp và chống dòng ngược.

A4988 là một loại chip điều khiển động cơ bước đơn giản và mạnh mẽ. Nó được sử dụng rộng rãi trong các ứng dụng Arduino. Với những chức năng như điều khiển dòng động cơ, điều chỉnh tốc độ, và chế độ hoạt động microstep.

Các mô-đun này có một số chức năng an toàn như sau:

- Bảo vệ quá dòng.

- Bảo vệ sụt áp.

- Bảo vệ quá nhiệt.

Thông số kỹ thuật:

Dưới đây là một số tính năng và thông số kỹ thuật quan trọng của chip A4988:

- Điện áp hoạt động: Tính từ 8V đến 35V.

- Dòng điều khiển động cơ: Tối đa 2A

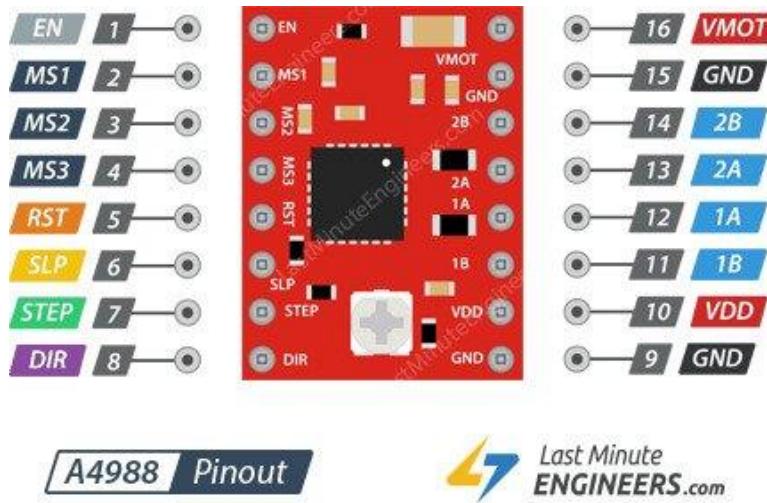
- Chế độ hoạt động microstep: Hỗ trợ từ full step đến 1/16 step

- Chức năng tắt nguồn tự động (Automatic Power-Down): Giúp tiết kiệm năng lượng khi động cơ không hoạt động.

- Chức năng bảo vệ quá nhiệt (Thermal Shutdown): Ngắt kết nối nguồn khi nhiệt độ vượt quá mức cho phép.

- Chức năng giảm dòng động cơ (Current Decay Mode): Điều chỉnh cách giảm dòng điện khi động cơ không hoạt động để giảm tiếng ồn và tiết kiệm năng lượng.

Sơ đồ chân Driver điều khiển động cơ bước A4988



A4988 Pinout



Hình 3.12 Sơ đồ chân Driver A4988

Dưới đây là bảng mô tả cách lựa chọn chế độ microstep bằng các chân microstep selection trên Driver A4988:

MS1 Pin	MS2 Pin	MS3 Pin	Chế độ microstep
LOW	LOW	LOW	Full step
HIGH	LOW	LOW	Half step
LOW	HIGH	LOW	1/4 step
HIGH	HIGH	LOW	1/8 step
HIGH	HIGH	HIGH	1/16 step

Bảng 3.4 Bảng lựa chọn chế độ Microstep

Bằng cách cấu hình các chân microstep selection theo ý muốn, ta có thể điều chỉnh độ phân giải và độ chính xác của động cơ bước khi hoạt động.

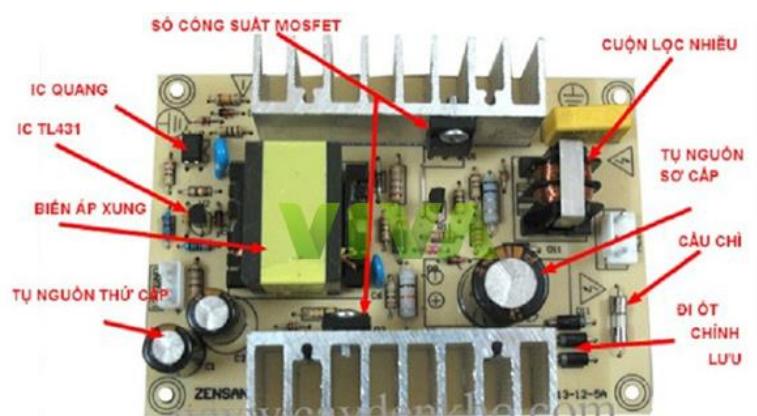
3.4.2 Nguồn tổ ong



Hình 3.13 Nguồn tổ ong hiện nay trên thị trường

3.4.2.1. Cấu tạo

Nguồn xung là bộ mạch có công dụng để biến đổi nguồn điện từ xoay chiều sang nguồn điện một chiều. Bằng chế độ dao động xung tạo bằng mạch điện tử kết hợp với một biến áp xung. Trên thực tế, có rất nhiều nguồn tuyển tính cổ điển sử dụng biến áp sắt từ giúp hạ áp, sau đó dùng chỉnh lưu kết hợp với IC nguồn tuyển tính. Nhiệm vụ chính là để tạo ra các cấp điện áp một chiều tùy theo mục đích sử dụng như: 3.3V, 5V, 6V...



Hình 3.14 Cấu tạo nguồn tổ ong

- Có rất nhiều nguồn tổ ong với các kích thước và linh kiện khác nhau. Tuy nhiên, một board cơ bản sẽ có những linh kiện sau:

+ Biến áp xung: đây là bộ phận chính trong một bo mạch xung. Được cấu tạo từ các cuộn dây quấn trên một lõi từ. Cấu tạo của nó, khá giống với các loại máy biến áp thông thường. Song, nguồn tần số sử dụng lõi ferit còn biến áp truyền thông sử dụng lõi thép kỹ thuật điện.

+ Cầu chì: bộ phận dùng để bảo vệ mạch nguồn bị ngắn mạch.

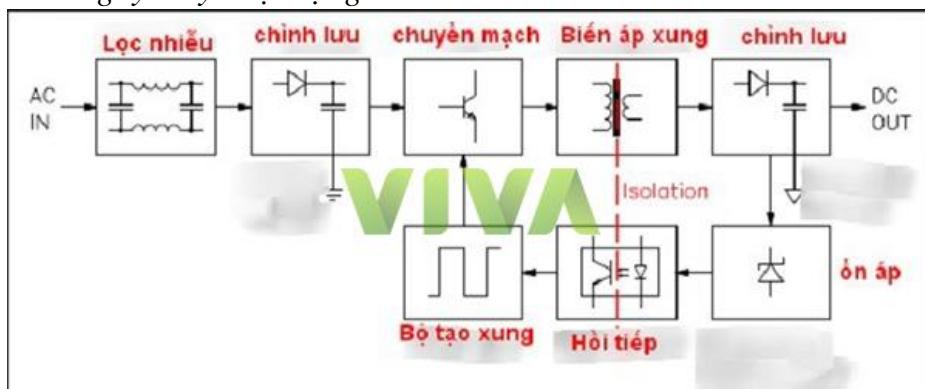
+ Sò công suất: đây là chất bán dẫn có tác dụng như một công tắc chuyển mạch. Nó có thể là transistor, mosfet, IC tích hợp, IGBT. Nhiệm vụ chính là đóng/cắt điện từ chân (+) của tụ lọc sơ cấp vào cuộn dây sơ cấp của biến áp xung rồi cho xuống mass.

+ Cuộn chấn nhiễu, tụ lọc sơ cấp, diode chỉnh lưu: các linh kiện này trong nguồn xung dùng để biến đổi điện áp xoay chiều 220V thành điện áp một chiều tích. Và trữ trên tụ lọc sơ cấp để cung cấp năng lượng cho cuộn sơ cấp của máy biến áp xung.

+ Tụ lọc nguồn thứ cấp: đây là bộ phận dùng để tích trữ năng lượng điện từ cuộn thứ cấp của biến áp xung để cấp cho tải tiêu thụ.

+ IC quang và IC TL431: linh kiện này sẽ tạo ra một điện áp cố định để không chế điện áp ra bên thứ cấp ổn định theo mong muốn.

3.4.2.2. Nguyên lý hoạt động



Hình 3.15 Sơ đồ nguyên lý hoạt động nguồn tần số

Khi công tắc điện mở, nguồn điện sẽ được đi qua nguồn xung. Khi đó, cuộn sơ cấp của biến áp được đóng/cắt điện liên tục bằng sò công suất sẽ xuất hiện từ trường biến thiên. Dẫn đến cuộn thứ cấp của biến áp cũng xuất hiện một điện áp ra. Điện áp này được chỉnh lưu qua một vài diode rồi đưa ra tụ lọc (tụ điện) thứ

cấp để san phẳng điện áp. Tiếp theo, các tụ IC quang và IC TL431 sẽ không chế dao động đóng cắt điện vào cuộn sơ cấp của biến áp xung sao cho điện áp ra bên thứ cấp đạt yêu cầu.

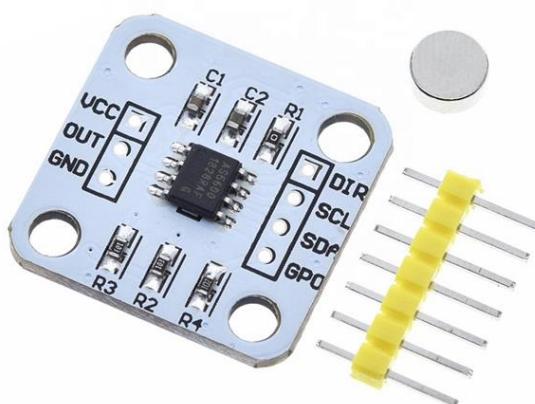
3.4.3 Encoder AS5600

Module cảm biến góc quay bằng từ trường AS5600 I2C là một mạch đo góc bằng cảm ứng từ trường không tiếp xúc với độ chính xác cao, nhiều chế độ đầu ra: Analog/PWM dễ dàng sử dụng khi kết nối với vi điều khiển.

Chức năng chân ra:

- VCC: 3.3V
- GND: GND
- Out: Ngõ ra điện áp PWM / Analog
- DIR: Hướng quay (mặt đất = Giá trị tăng theo chiều kim đồng hồ; sau đó VCC = Giá trị giảm theo chiều kim đồng hồ)

- SCL: chân giao tiếp I2C
- SDA: chân truyền dữ liệu I2C
- GPO: Lựa chọn chế độ
- Kích thước: 23 x 23mm
- Trọng lượng: 2.2g



Hình 3.20. Mạch AS5600

Nguyên lý hoạt động của encoder AS5600 như sau:

- Hall Effect Sensors: AS5600 sử dụng các cảm biến Hall Effect để phát hiện mức độ từ từ tính tạo ra bởi một nam châm cố định gắn trên cơ cấu quay (như động cơ bước). Khi nam châm quay, mức độ từ từ tính sẽ thay đổi theo góc quay.

- Chuyển đổi Analog Signal: AS5600 đọc dữ liệu từ các cảm biến Hall Effect và chuyển đổi chúng thành tín hiệu analog. Điều này có thể là một tín hiệu điện áp biểu thị mức độ từ từ tính hoặc một tín hiệu dòng tương tự.

- Chuyển đổi số hóa: Tín hiệu analog được chuyển đổi thành tín hiệu số bằng bộ chuyển đổi ADC (Analog-to-Digital Converter) có sẵn trên chip AS5600. Điều này cho phép dữ liệu từ cảm biến được biểu diễn dưới dạng giá trị số.

- Xử lý và Đọc Góc Quay: Giá trị số biểu diễn góc quay hiện tại của cơ cấu quay (động cơ bước) có thể được đọc từ AS5600 bằng giao diện truyền thông như I2C hoặc SPI. Phần mềm điều khiển có thể đọc giá trị này và sử dụng nó để theo dõi và điều khiển vị trí hoặc góc quay của động cơ bước.

- Phản hồi Đóng (Feedback): Dữ liệu góc quay từ encoder AS5600 có thể được sử dụng như một hình thức phản hồi đóng (feedback) trong hệ thống điều khiển động cơ bước. Bằng cách so sánh góc quay thực tế với góc quay mục tiêu, hệ thống điều khiển có thể điều chỉnh dòng điện đưa vào động cơ để đạt được vị trí hoặc góc quay mong muốn.

CHƯƠNG 4. CƠ SỞ LÝ THUYẾT, HÀM TÍNH TOÁN SỬ DỤNG TRONG THUẬT TOÁN VÀ TỐI UU CHO HỘP GIẢM TỐC

4.1 Tối ưu các biến thiết kế và hàm số

Một vecto bao gồm năm biến thiết kế được biểu thị bằng:

$$X = (D_z, d'_z, B, K_1, d'_w)$$

Trong đó:

$K_1 = \frac{r_b}{R_z}$ với $R_z = \frac{D_z}{2}$ và r_b : bán kính lõi tâm của bánh răng

D_z : là đường kính của vòng tròn phân phối pin

d'_z : là đường kính của pin

B : độ dày của bánh răng cycloid

d'_w : là đường kính của chốt đầu ra

Giới hạn cho các biến thiết kế:

Min	Biến	Max	Đơn vị
0.45	K_1	0.8	N/A
18	D_z	36	mm
3	d'_z	8	mm
4	B	12	mm
3	d'_w	8	mm

Bảng 4.1 Bảng giới hạn của các biến thiết kế

Ta cần thiết kế một hộp giảm tốc có tỷ số truyền cao , nhỏ gọn và hiệu suất cao nên ở đây để đơn giản hơn trong quá trình tính toán chúng ta sẽ ưu tiên kích thước D_z là kích thước để đảm bảo cho hộp giảm tốc cycloid có kích thước nhỏ.

Ngoài ra ta có hai kích thước d'_z : là đường kính của pin và d'_w : là đường kính của chốt đầu ra để thuận tiện cho tính toán cùnh như gia công ta sẽ cho $d'_z = d'_w$.

4.1.1 Chức năng khách quan

Để thuận tiện công suất đầu vào, tốc độ đầu vào, tỷ số truyền, số chốt đầu ra và vật liệu GCr15 được chọn cho các chốt và pin

Theo các cơ sở này, các mục tiêu phụ được trình bày như sau.

Công suất đầu vào, (W)	12
Tốc độ đầu vào (vòng/phút)	60
Tỷ số truyền, u	9
Số chốt đầu ra ,Z _w	6
Ứng suất trượt tối đa của vật liệu, (σ_{HP})	850
Ứng suất uốn tối đa của vật liệu, (σ_{FP})	150
Đường kính ngoài của ổ vòng bi nhỏ, D ₁ (mm)	18
Độ dày của vòng bi nhỏ, b (mm)	4

Bảng 4.2 Bảng thông số đầu vào của hộp giảm tốc

4.1.2 Khối lượng của bộ giảm tốc

Kích thước xuyên tâm bị ảnh hưởng bởi đường kính của vòng tròn phân phối pin (D_z) và d'_z đường kính của pin. Kích thước trực bị ảnh hưởng bởi độ dày của bánh răng cycloid (B) và khoảng cách giữa hai bánh răng cycloid. Do đó, có thể được thể hiện là :

$$\min f_1(X) = \frac{\pi}{4} (D_z + d'_z + 2\Delta_1)^2 \times (2B + \delta) \quad (4.1.1)$$

Do $\delta = 0$ và $\Delta_1 = 0$ vì trong phần tính toán này chúng ta sẽ sử dụng phiên bản hộp giảm tốc cycloid một cấp và không sử dụng vòng bi cho các con lăn cũng như chốt đầu ra.

4.1.3 Tải trọng xuyên tâm khi quay ổ trực

Tuổi thọ của ổ trực cánh tay quay phần lớn phụ thuộc vào tải trọng hướng tâm của nó. Nó ảnh hưởng hơn nữa đến tuổi thọ của bộ giảm tốc. Do đó, cần xem xét giảm thiểu tải trọng xuyên tâm của ổ trực cánh tay quay. Nó có thể được thể hiện như sau:

$$\min f_2(X) = \frac{2.6 T_g Z_b}{K_1 D_z Z_g} \quad (4.1.2)$$

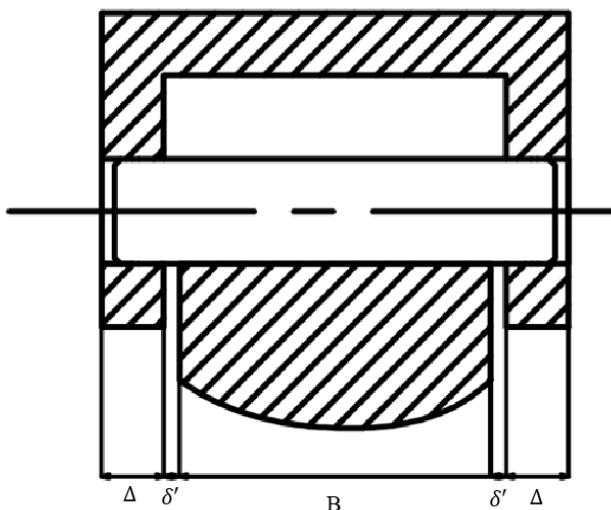
4.1.4 Ứng suất uốn tối đa của pin

Gãy pin là một trong những dạng thất bại chính của bộ giảm tốc cycloid này. Do đó, giảm thiểu ứng suất uốn tối đa của pin là đối tượng phụ thứ ba. Đối với

bánh răng có hai điểm tựa (trong Hình 4.1), ứng suất uốn của chân được hiển thị như sau :

$$\sigma_F \approx \frac{M_w \max}{0.1 d_z'^3} = \frac{44 L_1 L_2 T_v}{L K_1 Z_g D_z d_z'^3} \quad (4.1.3)$$

Trong đó :



Hình 4.1 Cấu tạo điểm hình của một bánh răng trụ có 2 trục

$$+ L_1 = L_2 = 0.5B + \delta' + 0.5\Delta = 0.5B + 2$$

$$+ L = L_1 + L_2 = B + 2\delta' + \Delta = B + 4$$

+ Δ : Là độ dày của thành bên của vỏ bánh răng.

+ δ' : Là khoảng cách giữa bánh răng cycloid và mặt bên trong của vỏ

Mục tiêu phụ thứ ba được trình bày dưới dạng :

$$\min f_3(X) = \frac{44 L_1 L_2 T_v}{L K_1 Z_g D_z d_z'^3} \quad (4.1.4)$$

4.2 Điều kiện hạn chế cho các biến

4.2.1 Hệ số biên độ ngắn

Nếu hệ số biên độ ngắn lớn hơn bán kính tối thiểu của cấu hình răng lý thuyết bánh răng cycloid thì sẽ bị giảm, nó sẽ làm giảm bán kính của pin, nghĩa là cường độ tiếp xúc của bánh răng cycloid và Gear Pin tăng. Theo phương trình (4.1.4), ứng suất uốn của chân sẽ tăng lên khi giảm hệ số biên độ ngắn. Nó gợi ý rằng phạm vi ràng buộc của hệ số biên độ ngắn là [0,45, 0,8].

Do đó, phương trình ràng buộc có thể được xác định bởi:

$$g_1(X) = 0.45 - K_1 \leq 0$$

$$g_2(X) = K_1 - 0.8 \leq 0$$

Hay

$$0.45 \leq K_1 \leq 0.8 \quad (4.2.1)$$

4.2.2 Biên dạng răng Cycloid

Để ngăn biên dạng răng cycloid khỏi bị cắt xén và góc nhọn, tỷ lệ đường kính của chốt với đường kính của vòng tròn phân bố bánh răng chốt phải nhỏ hơn hệ số tối thiểu của bán kính cong biên dạng răng lý thuyết (α_{min}). Theo đó, hạn chế này có thể được xác định bởi:

$$g_3(X) = \frac{d'_z}{D_z} - \alpha_{min} \leq 0$$

$$\text{Tại } \alpha_{min} = (1 + K_1)^2 / (1 + K_1 + Z_g K_1)$$

Hay

$$g_3(X) = \frac{d'_z}{D_z} - \frac{(1 + K_1)^2}{(1 + K_1 + Z_g K_1)} \leq 0 \quad (4.2.2)$$

4.2.3 Đường kính tối đa của lỗ chốt hình trụ

Để đảm bảo độ bền của bánh răng xích, phải có độ dày nhất định (T) giữa hai lỗ chốt đầu ra liền kề và nói chung, $T = 0.03D_z$. Như vậy, đường kính lớn nhất của lỗ chốt hình trụ thỏa mãn các ràng buộc sau:

$$g_4(X) = 2T - D_w + d_{sk} + D_1 \leq 0 \quad (4.2.3)$$

Ở đây Z_w là số pin đầu ra. D_w là đường kính của vòng tròn phân bố lỗ pin đầu ra, có thể được xác định bởi:

$$D_w = \frac{d_{fc} + D_1}{2}$$

Trong đó:

- d_{fc} : Đường kính của vòng chân răng của bánh răng cycloid.

- D_1 : Đường kính của lỗ tâm bánh răng cycloid, cũng là đường kính ngoài của ố đỡ tay quay.

- d_{sk} : là đường kính của lỗ chốt hình trụ, có thể được xác định bởi:

$$d_{sk} = d_w + 2e = d'_w + 2$$

Và với d'_w là đường kính ngoài của chốt đầu ra.

4.2.4 Hệ số đường kính chốt

Để đảm bảo độ bền của vỏ bánh răng chốt và tránh hỏng biên dạng của bánh răng chốt, giá trị của hệ số đường kính chốt K_2 phải nằm trong khoảng 1,25–4. Do đó, điều kiện ràng buộc này có thể được thể hiện như:

$$\begin{aligned} g_5(X) &= 1.25 - K_2 \leq 0 \\ g_6(X) &= K_2 - 4 \leq 0 \end{aligned}$$

Hay

$$1.25 \leq K_2 \leq 4 \quad (4.2.4)$$

$$\text{Tại đó: } K_2 = \frac{D_z}{d'_z} \cdot \sin \frac{\pi}{Z_b}$$

4.2.5 Độ bền uốn của bánh răng chốt

Theo phương trình (4.1.3), ràng buộc như sau :

$$g_7(X) = \frac{44L_1L_2T_v}{LK_1Z_gD_zd'^3_z} - \sigma_{FP} \leq 0 \quad (4.2.5)$$

Ở đây ta có σ_{FP} là ứng suất uốn cho phép.

4.2.6 Cường độ tiếp xúc giữa chốt trụ và lõi chốt trụ

Theo ông Rao, ràng buộc này như sau:

$$g_8(X) = 0.0949 \sqrt{\frac{10K_1T_vD_z}{Z_wD_wB(r_w^2Z_b + \frac{D_zK_1}{2}r_w)}} - \sigma_{HP} \leq 0 \quad (4.2.6)$$

Ở đây r_w là bán kính của lõi pin, có thể được xác định bởi

$$r_w = \frac{d'_w}{2}$$

Ở đây ta có σ_{HP} là ứng suất uốn cho phép.

4.2.7 Độ bền uốn của chốt trụ

Theo ông Rao, ràng buộc này như sau:

$$g_9(X) = \frac{96T_v(1.5B + \delta)}{Z_wR_wd'^3_w} - \sigma_{FP} \leq 0 \quad (4.2.9)$$

4.2.8 Phương trình tối ưu thông số cuối cùng

Như ta biết sự quan trọng trong chất lượng của một loại hộp giảm tốc chính là độ bền và độ chịu tải do đó ở đây ta chọn hệ số cân đối giữa các phương trình tối ưu như sau : 0,2 : 0,5 : 0,3

20%: Cho khối lượng hộp giảm tốc

50%: Cho tải trọng xuyên tâm

30%: Cho ứng suất uốn của pin

$$\text{Hàm } F = 2F_1 + 5F_2 + 3F_3$$

Với :

$$F_1 = \sqrt[3]{f_1}$$

$$F_2 = \sqrt{\frac{1}{f_2}}$$

$$F_3 = \sqrt[4]{\frac{1}{f_3}}$$

Từ đó ta có:

$$f_1(X) = \frac{\pi}{4} (D_z + d'_z + 2\Delta_1)^2 \times (2B + \delta)$$

$$f_2(X) = \frac{2.6 T_g Z_b}{K_1 D_z Z_g}$$

$$f_3(X) = \frac{44 L_1 L_2 T_v}{L K_1 Z_g D_z d'^3_z}$$

Cùng với hàm số tối ưu và các biến tối ưu là các giá trị của các thông số khác mà ta đã biết trước hay giả định trước để thuận tiện trong quá trình chạy thuật toán cùngh như tính toán các giá trị của các biến.

4.3 Thuật toán tối ưu cho các biến thiết kế

4.3.1 Tổng quan về thuật toán Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) là một thuật toán tối ưu hóa được lấy cảm hứng từ hành vi di cư tập hợp của đàn chim và các loài động vật khác. Thuật toán này giải quyết các bài toán tối ưu bằng cách mô phỏng sự tương tác và hợp tác giữa các "hạt" (particles) trong một không gian tìm kiếm.

Cơ bản, thuật toán PSO bao gồm các bước sau:

- + Khởi tạo không gian tìm kiếm: Định nghĩa không gian tìm kiếm bằng cách xác định các giới hạn và phạm vi của các biến tối ưu hóa.

- + Khởi tạo quần thể hạt: Tạo ra một quần thể gồm nhiều hạt (particles), mỗi hạt đại diện cho một giải pháp ứng viên.

+ Đánh giá hạt: Đánh giá hiệu suất của mỗi hạt bằng cách tính toán giá trị mục tiêu (fitness value) dựa trên hàm mục tiêu của bài toán tối ưu.

+ Cập nhật vị trí và vận tốc: Mỗi hạt di chuyển trong không gian tìm kiếm dựa trên vị trí hiện tại và vận tốc của nó. Vận tốc được điều chỉnh bằng cách tính toán các thành phần chính của PSO, bao gồm vị trí tốt nhất của hạt (Pbest) và vị trí tốt nhất của quần thể (Gbest).

+ Lặp lại quá trình: Quá trình cập nhật vị trí và vận tốc được lặp lại cho đến khi đạt được tiêu chí dừng, chẳng hạn như đạt đến số lần lặp tối đa hoặc đạt được giá trị mục tiêu mong muốn.

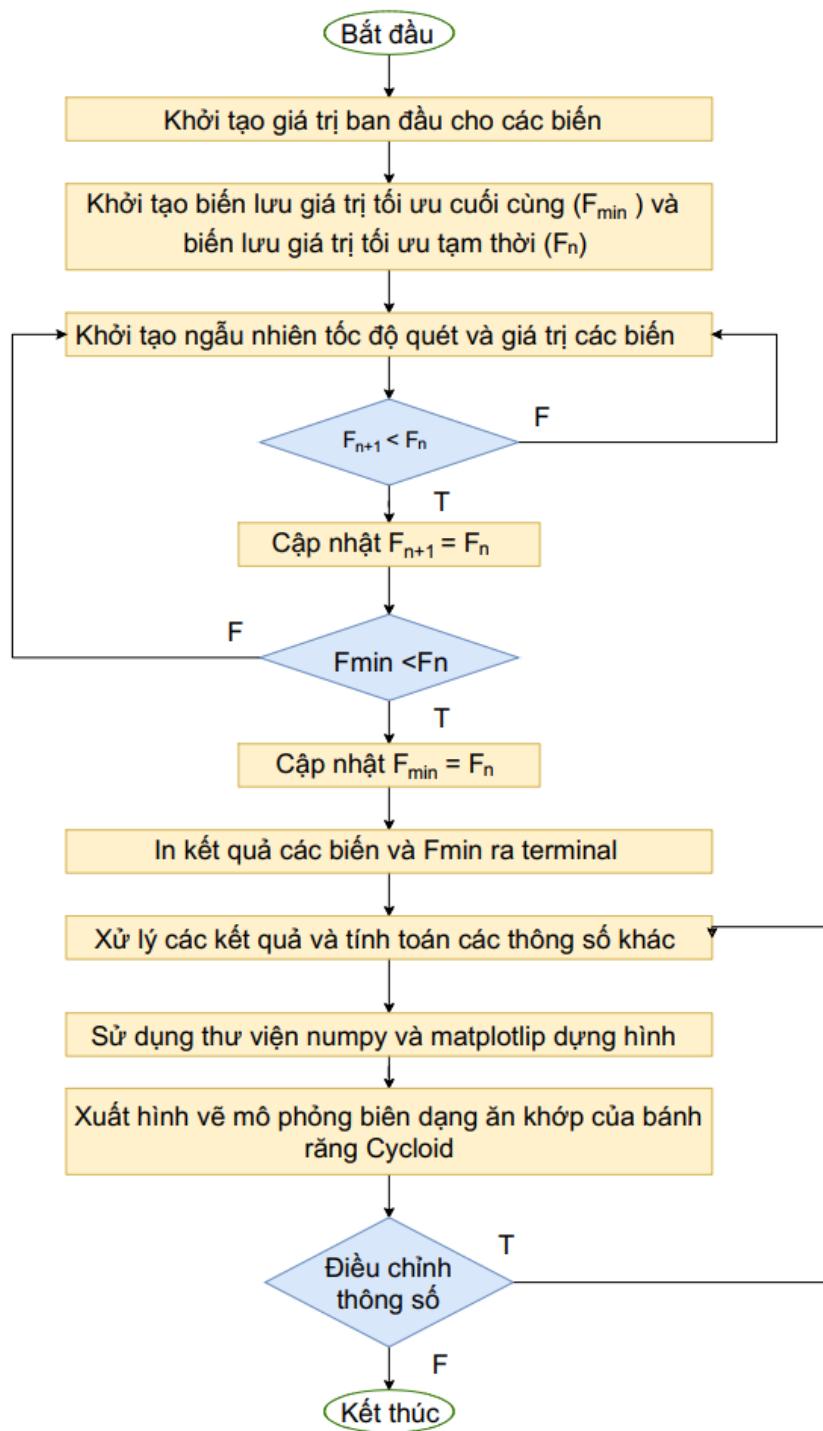
+ Trả về giải pháp tốt nhất: Sau khi thuật toán PSO kết thúc, giải pháp tốt nhất được xác định bằng cách chọn hạt có giá trị mục tiêu tốt nhất (Pbest hoặc Gbest) trong quần thể.

Thuật toán PSO kết hợp sự khám phá không gian tìm kiếm thông qua việc di chuyển ngẫu nhiên và sự khai thác thông qua việc học hỏi từ vị trí tốt nhất hiện tại. Điều này giúp PSO tìm ra các giải pháp tiềm năng và hội tụ đến kết quả tối ưu.

PSO đã được áp dụng rộng rãi trong nhiều lĩnh vực, bao gồm tối ưu hóa hàm, mạng nơ-ron nhân tạo, tối ưu hóa các tham số, và các bài toán tối ưu phức tạp khác. Đặc điểm của PSO là dễ hiểu, cài đặt đơn giản và có khả năng tìm kiếm nhanh chóng trong không gian tìm kiếm lớn.

4.3.2 Nội dung của chương trình tối ưu

Sơ đồ thuật toán



Hình 4.2 Sơ đồ thuật toán tối ưu

4.3.2.1. Code tối ưu

```
import numpy as np
class Particle:
```

```

def __init__(self, x_min, x_max, y_min, y_max, z_min, z_max, t_min,
t_max):
    self.position = np.array([np.random.uniform(x_min, x_max),
                             np.random.uniform(y_min, y_max),
                             np.random.uniform(z_min, z_max),
                             np.random.uniform(t_min, t_max)])
    self.velocity = np.array([np.random.uniform(-1, 1),
                             np.random.uniform(-1, 1),
                             np.random.uniform(-1, 1),
                             np.random.uniform(-1, 1)])
    self.best_position = self.position.copy()
    self.best_fitness = float('inf')
def update_velocity(self, global_best_position, inertia_weight,
cognitive_weight, social_weight):
    self.velocity = (inertia_weight * self.velocity + cognitive_weight
* np.random.rand() * (self.best_position - self.position) + social_weight
* np.random.rand() * (global_best_position - self.position))
    def update_position(self, x_min, x_max, y_min, y_max, z_min, z_max,
t_min, t_max):
        self.position = np.clip(self.position + self.velocity, [x_min,
y_min, z_min, t_min], [x_max, y_max, z_max, t_max])
    def evaluate_fitness(self):
        x, y, z, t = self.position
        f1 = self.F1(y, z, t)
        f2 = self.F2(x, t)
        f3 = self.F3(x, y, z, t)
        fitness = 2 * f1 + 5 * f2 + 3 * f3
        return fitness
    def F1(self, y, z, t):
        f1 = pow(((y + t) * (y + t) * (z) * (np.pi/2)), 1/3)
        mi = pow(((3 + 18) * (3 + 18) * (4) * (np.pi/2)), 1/3)
        ma = pow(((8 + 36) * (8 + 36) * (12) * (np.pi/2)), 1/3)
        return (f1 - mi) / (ma - mi)
    def F2(self, x, t):
        f2 = pow(598 / (9 * x * t), 1/2)
        mi = pow(598 / (9 * 0.8 * 36), 1/2)
        ma = pow(598 / (9 * 0.45 * 18), 1/2)
        return (f2 - mi) / (ma - mi)
    def F3(self, x, y, z, t):
        f3 = pow((44 * 41.81 * ((0.5 * z + 2) * 0.5)) / (9 * x * y**3 *
t), 1/4)
        ma = pow((44 * 41.81 * ((0.5 * 12 + 2) * 0.5)) / (9 * 0.45 * 3**3
* 18), 1/4)
        mi = pow((44 * 41.81 * ((0.5 * 4 + 2) * 0.5)) / (9 * 0.8 * 8**3 *
36), 1/4)
        return (f3 - mi) / (ma - mi)
def particle_swarm_optimization(population_size, x_min, x_max, y_min,
y_max, z_min, z_max, t_min, t_max,
num_iterations, inertia_weight,
cognitive_weight, social_weight):

```

```

particles = [Particle(x_min, x_max, y_min, y_max, z_min, z_max,
t_min, t_max) for _ in range(population_size)]
global_best_fitness = float('inf')
global_best_position = None
for iteration in range(num_iterations):
    for particle in particles:
        fitness = particle.evaluate_fitness()
        if fitness < particle.best_fitness:
            particle.best_fitness = fitness
            particle.best_position = particle.position.copy()
        if fitness < global_best_fitness:
            global_best_fitness = fitness
            global_best_position = particle.position.copy()
        particle.update_velocity(global_best_position,
inertia_weight, cognitive_weight, social_weight)
        particle.update_position(x_min, x_max, y_min, y_max, z_min,
z_max, t_min, t_max)
    return global_best_fitness, global_best_position
# Example usage
population_size = 100
x_min, x_max = 0.45, 0.8
y_min, y_max = 3.0, 8.0
z_min, z_max = 4.0, 12.0
t_min, t_max = 18.0, 36.0
num_iterations = 100
inertia_weight = 0.8
cognitive_weight = 1.5
social_weight = 1.5
def particle_export():
    best_fitness, best_position =
particle_swarm_optimization(population_size, x_min, x_max, y_min, y_max,
z_min, z_max, t_min, t_max, num_iterations,
inertia_weight, cognitive_weight, social_weight)
    print("best_fitness", best_fitness)
    print("best_position", best_position)
    return best_fitness, best_position

```

4.3.2.2. Code Draw 2D

```

import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from matplotlib.widgets import Slider, Button, RadioButtons
from func.Code import particle_export

best_fitness, best_position = particle_export()
Rm = 3
Rd = ((best_position[3]-best_position[1]-2) - 6)/2
rd = best_position[1]/2

```

```

d = best_position[1]
D_b = best_position[3]

fig, ax = plt.subplots(figsize=(6,6))
plt.subplots_adjust(left=0.15, bottom=0.35)

ax.set_aspect('equal')
plt.xlim(-1.4*40,1.4*40)
plt.ylim(-1.4*40,1.4*40)
#plt.grid()
t = np.linspace(0, 2*np.pi, 4096)
# set thông số ban đầu cho model
delta = 1
mode_fig = 1
pin_fig = 0
cycloid_fig = 1

## draw pin
num_pins = 50
pins = [ax.plot([], [], 'k-')[0] for n in range(num_pins)]
def draw_pin_init():
    for p in pins:
        p.set_data([0], [0])

def pin_update(n,d,D,phi):
    global mode_fig
    for i in range(int(n)):
        if mode_fig == 1:
            x = (d/2*np.sin(t)+ D/2*np.cos(2*i*np.pi/n))
            y = (d/2*np.cos(t) + D/2*np.sin(2*i*np.pi/n))
        if mode_fig == 0:
            x = (d/2*np.sin(t)+ D/2*np.cos(2*i*np.pi/n))*np.cos(phi/(n))
- (d/2*np.cos(t) + D/2*np.sin(2*i*np.pi/n))*np.sin(phi/(n))
            y = (d/2*np.sin(t)+ D/2*np.cos(2*i*np.pi/n))*np.sin(phi/(n))
+ (d/2*np.cos(t) + D/2*np.sin(2*i*np.pi/n))*np.cos(phi/(n))
        pins[i].set_data(x,y)

## draw inner_pin
num_inner_pins = 10
inner_pins = [ax.plot([], [], 'g-')[0] for n in range(num_inner_pins)]
def draw_inner_pin_init():
    for p in inner_pins:
        p.set_data([0], [0])

def inner_pin_update(n,N,rd,Rd,phi):
    global mode_fig
    for i in range(int(n)):
        if mode_fig == 1:

```

```

        x = (rd*np.sin(t)+ Rd*np.cos(2*i*np.pi/n))*np.cos(-phi/(N-1)-
3*np.pi/(N-1)-np.pi/3) - (rd*np.cos(t) + Rd*np.sin(2*i*np.pi/n))*np.sin(-
phi/(N-1)+ 3*np.pi/(N-1)-np.pi/3)
        y = (rd*np.sin(t)+ Rd*np.cos(2*i*np.pi/n))*np.sin(-phi/(N-1)-
3*np.pi/(N-1)-np.pi/3) + (rd*np.cos(t) + Rd*np.sin(2*i*np.pi/n))*np.cos(-
phi/(N-1)+ 3*np.pi/(N-1)-np.pi/3)
    if mode_fig == 0:
        x = (rd*np.sin(t)+ Rd*np.cos(2*i*np.pi/n))*np.cos(
3*np.pi/(N-1)-np.pi/3) - (rd*np.cos(t) + Rd*np.sin(2*i*np.pi/n))*np.sin(
3*np.pi/(N-1)-np.pi/3)
        y = (rd*np.sin(t)+ Rd*np.cos(2*i*np.pi/n))*np.sin(
3*np.pi/(N-1)-np.pi/3) + (rd*np.cos(t) + Rd*np.sin(2*i*np.pi/n))*np.cos(
3*np.pi/(N-1)-np.pi/3)
    inner_pins[i].set_data(x,y)

## draw drive_pin
d0, = ax.plot([0], [0], 'k-', lw=2)
def drive_pin_update(r):
    x = r*np.sin(t)
    y = r*np.cos(t)
    d0.set_data(x,y)

#inner circleA:
num_inner_circlesA = 10
inner_circlesA = [ax.plot([], [], 'r-')[0] for n in
range(num_inner_circlesA)]
def draw_inner_circleA_init():
    for p in inner_circlesA:
        p.set_data([0], [0])

def update_inner_circleA(e,n,N,rd,Rd, phi):
    global mode_fig
    for i in range(int(n)):
        if mode_fig == 1:
            x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(-phi/(N-
1)+ 3*np.pi/(N-1)-np.pi/3) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin(-phi/(N-1)+ 3*np.pi/(N-
1)-np.pi/3) + e*np.cos(phi)
            y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(-phi/(N-
1)+ 3*np.pi/(N-1)-np.pi/3) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos(-phi/(N-1)+ 3*np.pi/(N-
1)-np.pi/3) + e*np.sin(phi)
        if mode_fig == 0:
            x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(
3*np.pi/(N-1)-np.pi/3) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin( 3*np.pi/(N-1)-np.pi/3)
+ e*np.cos(phi)

```

```

y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(
3*np.pi/(N-1)-np.pi/3) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos( 3*np.pi/(N-1)-np.pi/3)
+ e*np.sin(phi)
inner_circlesA[i].set_data(x,y)

#inner circleB:
num_inner_circlesB = 10
inner_circlesB = [ax.plot([], [], 'b-')[0] for n in
range(num_inner_circlesB)]
def draw_inner_circleB_init():
    for p in inner_circlesB:
        p.set_data([0], [0])

def update_inner_circleB(e,n,N,rd,Rd, phi):
    global mode_fig
    global cycloid_fig
    for i in range(int(n)):
        if mode_fig == 1:
            if cycloid_fig == 3:
                x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(-
phi/(N-1)+ 3*np.pi/(N-1) - np.pi/(3)) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin(-phi/(N-1)+ 3*np.pi/(N-
1) - np.pi/(3)) - e*np.cos(phi-np.pi/3)
                y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(-
phi/(N-1)+ 3*np.pi/(N-1) - np.pi/(3)) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos(-phi/(N-1)+ 3*np.pi/(N-
1) - np.pi/(3)) - e*np.sin(phi-np.pi/3)
            if cycloid_fig == 2:
                x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(-
phi/(N-1) + 3*np.pi/(N-1) - np.pi/(3)) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin(-phi/(N-1) + 3*np.pi/(N-
1) - np.pi/(3)) - e*np.cos(phi)
                y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(-
phi/(N-1) + 3*np.pi/(N-1) - np.pi/(3)) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos(-phi/(N-1) + 3*np.pi/(N-
1) - np.pi/(3)) - e*np.sin(phi)
            if mode_fig == 0:
                if cycloid_fig == 3:
                    x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(
3*np.pi/(N-1) - np.pi/(3)) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin( 3*np.pi/(N-1)-
np.pi/(3)) - e*np.cos(phi-np.pi/3)
                    y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(
3*np.pi/(N-1) - np.pi/(3)) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos( 3*np.pi/(N-1)-
np.pi/(3)) - e*np.sin(phi-np.pi/3)
                if cycloid_fig == 2:

```

```

        x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(
3*np.pi/(N-1) - np.pi/(3)) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin( 3*np.pi/(N-1) -
np.pi/(3)) - e*np.cos(phi)
        y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(
3*np.pi/(N-1) - np.pi/(3)) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos( 3*np.pi/(N-1) -
np.pi/(3)) - e*np.sin(phi)
    inner_circlesB[i].set_data(x,y)

#inner circleC:
num_inner_circlesC = 10
inner_circlesC = [ax.plot([], [], 'g-')[0] for n in
range(num_inner_circlesC)]
def draw_inner_circleC_init():
    for p in inner_circlesC:
        p.set_data([0], [0])

def update_inner_circleC(e,n,N,rd,Rd, phi):
    global mode_fig
    for i in range(int(n)):
        if mode_fig == 1:
            x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(-phi/(N-
1) + 3*np.pi/(N-1) - np.pi/(3)) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin(-phi/(N-1) + 3*np.pi/(N-
1)- np.pi/(3)) - e*np.cos(phi+np.pi/3)
            y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(-phi/(N-
1) + 3*np.pi/(N-1)- np.pi/(3)) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos(-phi/(N-1) + 3*np.pi/(N-
1)- np.pi/(3)) - e*np.sin(phi+np.pi/3)
        if mode_fig == 0:
            x = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.cos(
3*np.pi/(N-1) - np.pi/(3)) -
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.sin( 3*np.pi/(N-1)-
np.pi/(3)) - e*np.cos(phi+np.pi/3)
            y = ((rd+e)*np.cos(t)+Rd*np.cos(2*i*np.pi/n))*np.sin(
3*np.pi/(N-1) - np.pi/(3)) +
((rd+e)*np.sin(t)+Rd*np.sin(2*i*np.pi/n))*np.cos( 3*np.pi/(N-1)-
np.pi/(3)) - e*np.sin(phi+np.pi/3)
        inner_circlesC[i].set_data(x,y)

##inner pinA:
inner_pinA, = ax.plot([0],[0],'r-')
dotA, = ax.plot([0],[0], 'ro', ms=5)
def update_inner_pinA(e,Rm, phi):
    x = (Rm+e)*np.cos(t)+e*np.cos(phi)
    y = (Rm+e)*np.sin(t)+e*np.sin(phi)
    inner_pinA.set_data(x,y)

```

```

x1 = (Rm+e)*np.cos(phi)+e*np.cos(phi)
y1 = (Rm+e)*np.sin(phi)+e*np.sin(phi)
dotA.set_data(x1, y1)

##inner pinB:
inner_pinB, = ax.plot([0],[0], 'b-')
dotB, = ax.plot([0],[0], 'bo', ms=5)
def update_inner_pinB(e,Rm, phi):
    global cycloid_fig
    if cycloid_fig==3:
        x = (Rm+e)*np.cos(t)-e*np.cos(phi-np.pi/3)
        y = (Rm+e)*np.sin(t)-e*np.sin(phi-np.pi/3)
    if cycloid_fig==2:
        x = (Rm+e)*np.cos(t)-e*np.cos(phi)
        y = (Rm+e)*np.sin(t)-e*np.sin(phi)
    inner_pinB.set_data(x,y)
    if cycloid_fig==3:
        x1 = (Rm+e)*np.cos(phi+2*np.pi/3)-e*np.cos(phi-np.pi/3)
        y1 = (Rm+e)*np.sin(phi+2*np.pi/3)-e*np.sin(phi-np.pi/3)
    if cycloid_fig==2:
        x1 = (Rm+e)*np.cos(phi+np.pi)-e*np.cos(phi)
        y1 = (Rm+e)*np.sin(phi+np.pi)-e*np.sin(phi)
    dotB.set_data(x1, y1)

##inner pinC:
inner_pinC, = ax.plot([0],[0], 'g-')
dotC, = ax.plot([0],[0], 'go', ms=5)
def update_inner_pinC(e,Rm, phi):
    x = (Rm+e)*np.cos(t)-e*np.cos(phi+np.pi/3)
    y = (Rm+e)*np.sin(t)-e*np.sin(phi+np.pi/3)
    inner_pinC.set_data(x,y)

    x1 = (Rm+e)*np.cos(phi-2*np.pi/3)-e*np.cos(phi+np.pi/3)
    y1 = (Rm+e)*np.sin(phi-2*np.pi/3)-e*np.sin(phi+np.pi/3)
    dotC.set_data(x1, y1)

##ehypocycloidA:
ehypocycloidA, = ax.plot([0], [0], 'r-')
edotA, = ax.plot([0], [0], 'ro', ms=5)
def update_ehypocycloidA(e,n,D,d, phis):
    global mode_fig
    RD=D/2
    rd=d/2
    rc = (n-1)*(RD/n)
    rm = (RD/n)
    xa = (rc+rm)*np.cos(t)-e*np.cos((rc+rm)/rm*t)

```

```

ya = (rc+rm)*np.sin(t)-e*np.sin((rc+rm)/rm*t)

dxa = (rc+rm)*(-np.sin(t)+(e/rm)*np.sin((rc+rm)/rm*t))
dfa = (rc+rm)*(np.cos(t)-(e/rm)*np.cos((rc+rm)/rm*t))
if mode_fig == 1:
    x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dfa))*np.cos(-phis/(n-1))-(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.sin(-phis/(n-1)) +
    e*np.cos(phis)
    y = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dfa))*np.sin(-phis/(n-1))+(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.cos(-phis/(n-1)) +
    e*np.sin(phis)
if mode_fig == 0:
    x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dfa)) + e*np.cos(phis)
    y = (ya + rd/np.sqrt(dxa**2 + dya**2)*dxa) + e*np.sin(phis)

ehypocycloidA.set_data(x,y)
edotA.set_data(x[0], y[0])

##ehypocycloidB:
ehypocycloidB, = ax.plot([0],[0], 'b-')
edotB, = ax.plot([0],[0], 'bo', ms=5)
def update_ehypocycloidB(e,n,D,d, phis):
    global mode_fig
    global cycloid_fig
    RD=D/2
    rd=d/2
    rc = (n-1)*(RD/n)
    rm = (RD/n)
    xa = (rc+rm)*np.cos(t)+e*np.cos((rc+rm)/rm*t)
    ya = (rc+rm)*np.sin(t)+e*np.sin((rc+rm)/rm*t)

    dxa = (rc+rm)*(-np.sin(t)-(e/rm)*np.sin((rc+rm)/rm*t))
    dya = (rc+rm)*(np.cos(t)+(e/rm)*np.cos((rc+rm)/rm*t))
    if mode_fig == 1:
        if cycloid_fig == 3:
            x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dfa))*np.cos(-
            phis/(n-1) + np.pi/3/(n-1))-(ya + rd/np.sqrt(dxa**2 +
            dya**2)*dxa)*np.sin(-phis/(n-1) + np.pi/3/(n-1)) - e*np.cos(phis-
            np.pi/3)
            y = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dfa))*np.sin(-
            phis/(n-1) + np.pi/3/(n-1))+(ya + rd/np.sqrt(dxa**2 +
            dya**2)*dxa)*np.cos(-phis/(n-1) + np.pi/3/(n-1)) - e*np.sin(phis-
            np.pi/3)
        if cycloid_fig == 2:
            x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dfa))*np.cos(-
            phis/(n-1))-(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.sin(-phis/(n-
            1)) - e*np.cos(phis)

```

```

y = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya))*np.sin(-
phis/(n-1))+(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.cos(-phis/(n-
1)) - e*np.sin(phis)

if mode_fig == 0:
    if cycloid_fig == 3:
        x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya))*np.cos(
np.pi/3/(n-1))-(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.sin( np.pi/3/(n-
1)) - e*np.cos(phis-np.pi/3)
        y = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya))*np.sin(
np.pi/3/(n-1))+(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.cos( np.pi/3/(n-
1)) - e*np.sin(phis-np.pi/3)
        if cycloid_fig == 2:
            x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya)) -
e*np.cos(phis)
            y = (ya + rd/np.sqrt(dxa**2 + dya**2)*dxa) - e*np.sin(phis)

ehypocycloidB.set_data(x,y)
edotB.set_data(x[0], y[0])

##ehypocycloidC:
ehypocycloidC, = ax.plot([0],[0], 'g-')
edotC, = ax.plot([0],[0], 'go', ms=5)
def update_ehypocycloidC(e,n,D,d, phis):
    global mode_fig
    RD=D/2
    rd=d/2
    rc = (n-1)*(RD/n)
    rm = (RD/n)
    xa = (rc+rm)*np.cos(t)+e*np.cos((rc+rm)/rm*t)
    ya = (rc+rm)*np.sin(t)+e*np.sin((rc+rm)/rm*t)

    dxa = (rc+rm)*(-np.sin(t)-(e/rm)*np.sin((rc+rm)/rm*t))
    dya = (rc+rm)*(np.cos(t)+(e/rm)*np.cos((rc+rm)/rm*t))

    if mode_fig == 1:
        x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya))*np.cos(-phis/(n-1)-
np.pi/3/(n-1))-(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.sin(-phis/(n-
1) - np.pi/3/(n-1)) - e*np.cos(phis+np.pi/3)
        y = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya))*np.sin(-phis/(n-1)-
np.pi/3/(n-1))+(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.cos(-phis/(n-
1) - np.pi/3/(n-1)) - e*np.sin(phis+np.pi/3)
        if mode_fig == 0:
            x = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya))*np.cos( -
np.pi/3/(n-1))-(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.sin( -
np.pi/3/(n-1)) - e*np.cos(phis+np.pi/3)
            y = (xa + rd/np.sqrt(dxa**2 + dya**2)*(-dya))*np.sin( -
np.pi/3/(n-1))+(ya + rd/np.sqrt(dxa**2 + dya**2)*dxa)*np.cos( -
np.pi/3/(n-1)) - e*np.sin(phis+np.pi/3)

```

```

ehypocycloidC.set_data(x,y)
edotC.set_data(x[0], y[0])

##ehypocycloid_Pin:
ehypocycloid_Pin, = ax.plot([0],[0], 'k-')
edot_Pin, = ax.plot([0],[0], 'ko', ms=5)
def update_ehypocycloid_Pin(e,n,D,d, phis):
    global mode_fig
    RD=D/2
    rd=d/2
    rc = (n)*(RD/(n+1))
    rm = (RD/(n+1))
    xa = (rc+rm)*np.cos(t)+e*np.cos((rc+rm)/rm*t)
    ya = (rc+rm)*np.sin(t)+e*np.sin((rc+rm)/rm*t)

    dxa = (rc+rm)*(-np.sin(t)-(e/rm)*np.sin((rc+rm)/rm*t))
    dyda = (rc+rm)*(np.cos(t)+(e/rm)*np.cos((rc+rm)/rm*t))

    if mode_fig == 1:
        x = (xa + (rd-e)/np.sqrt(dxa**2 + dyda**2)*(-dyda))*np.cos(
        np.pi/(n)) - (ya + (rd-e)/np.sqrt(dxa**2 + dyda**2)*dxa)*np.sin(
        np.pi/(n))
        y = (xa + (rd-e)/np.sqrt(dxa**2 + dyda**2)*(-dyda))*np.sin(
        np.pi/(n)) + (ya + (rd-e)/np.sqrt(dxa**2 + dyda**2)*dxa)*np.cos(
        np.pi/(n))
    if mode_fig == 0:
        x = (xa + (rd-e)/np.sqrt(dxa**2 + dyda**2)*(-dyda))*np.cos(phis/(n)
        + np.pi/(n))-(ya + (rd-e)/np.sqrt(dxa**2 + dyda**2)*dxa)*np.sin(phis/(n) +
        np.pi/(n))
        y = (xa + (rd-e)/np.sqrt(dxa**2 + dyda**2)*(-dyda))*np.sin(phis/(n)
        + np.pi/(n))+(ya + (rd-e)/np.sqrt(dxa**2 + dyda**2)*dxa)*np.cos(phis/(n) +
        np.pi/(n))

    ehypocycloid_Pin.set_data(x,y)
    edot_Pin.set_data(x[0], y[0])

axcolor = 'lightgoldenrodyellow'

ax_fm = plt.axes([0.25, 0.27, 0.5, 0.02], facecolor=axcolor)
ax_Rm = plt.axes([0.25, 0.24, 0.5, 0.02], facecolor=axcolor)
ax_n = plt.axes([0.25, 0.21, 0.5, 0.02], facecolor=axcolor)
ax_Rd = plt.axes([0.25, 0.18, 0.5, 0.02], facecolor=axcolor)
ax_rd = plt.axes([0.25, 0.15, 0.5, 0.02], facecolor=axcolor)
ax_e = plt.axes([0.25, 0.12, 0.5, 0.02], facecolor=axcolor)
ax_N = plt.axes([0.25, 0.09, 0.5, 0.02], facecolor=axcolor)
ax_d = plt.axes([0.25, 0.06, 0.5, 0.02], facecolor=axcolor)
ax_D = plt.axes([0.25, 0.03, 0.5, 0.02], facecolor=axcolor)

```

```

# thanh trượt điều chỉnh giá trị
sli_fm = Slider(ax_fm, 'fm', -100, 100, valinit=100, valstep=delta*20)
sli_Rm = Slider(ax_Rm, 'Rm', 1, 20, valinit=Rm, valstep=delta/10)
sli_n = Slider(ax_n, 'n', 2, 10, valinit=6, valstep=delta)
sli_Rd = Slider(ax_Rd, 'Rd', 1, 40, valinit=Rd, valstep=delta/10)
sli_rd = Slider(ax_rd, 'rd', 1, 10, valinit=rd, valstep=delta/10)
sli_e = Slider(ax_e, 'e', 0.1, 10, valinit=1, valstep=delta/10)
sli_N = Slider(ax_N, 'N', 2, 40, valinit=10, valstep=delta)
sli_d = Slider(ax_d, 'd', 2, 20, valinit=d, valstep=delta/10)
sli_D = Slider(ax_D, 'D', 5, 150, valinit=D_b, valstep=delta/10)

def update(val):
    sfm = sli_Rm.val
    sRm = sli_Rm.val
    sRd = sli_Rd.val
    sn = sli_n.val
    srd = sli_rd.val
    se = sli_e.val
    sN = sli_N.val
    sd = sli_d.val
    sD = sli_D.val
    ax.set_xlim(-1.4*0.5*sD,1.4*0.5*sD)
    ax.set_ylim(-1.4*0.5*sD,1.4*0.5*sD)

sli_fm.on_changed(update)
sli_Rm.on_changed(update)
sli_Rd.on_changed(update)
sli_n.on_changed(update)
sli_rd.on_changed(update)
sli_e.on_changed(update)
sli_N.on_changed(update)
sli_d.on_changed(update)
sli_D.on_changed(update)

# Khởi tạo các nút bấm chọn và chọn giá trị ban đầu
resetax = plt.axes([0.85, 0.001, 0.1, 0.04])
button = Button(resetax, 'Reset', color=axcolor, hovercolor='0.975')
rax = plt.axes([0.025, 0.65, 0.15, 0.15], facecolor=axcolor)
radio = RadioButtons(rax, ('Mode 1', 'Mode 2'), active=1)
rax_1 = plt.axes([0.025, 0.45, 0.15, 0.15], facecolor=axcolor)
radio_1 = RadioButtons(rax_1, ('Pin', 'NonePin'), active=0)
rax_2 = plt.axes([0.025, 0.25, 0.15, 0.15], facecolor=axcolor)
radio_2 = RadioButtons(rax_2, ('Single', 'Dual', 'Triple'), active=0)

def reset(event):
    sli_fm.reset()
    sli_Rm.reset()
    sli_n.reset()
    sli_rd.reset()

```

```

sli_Rd.reset()
sli_e.reset()
sli_N.reset()
sli_d.reset()
sli_D.reset()

button.on_clicked(reset)

def modefunc(label):
    global mode_fig
    if label == 'Mode 1':
        mode_fig = 0
    if label == 'Mode 2':
        mode_fig = 1
radio.on_clicked(modefunc)

def pin_modefunc(label):
    global pin_fig
    if label == 'Pin':
        pin_fig = 0
    if label == 'NonePin':
        pin_fig = 1
radio_1.on_clicked(pin_modefunc)

def cycloid_modefunc(label):
    global cycloid_fig
    if label == 'Single':
        cycloid_fig = 1
    if label == 'Dual':
        cycloid_fig = 2
    if label == 'Triple':
        cycloid_fig = 3
radio_2.on_clicked(cycloid_modefunc)

def animate(frame):
    global pin_fig
    global cycloid_fig
    sfm = sli_fm.val
    sRm = sli_Rm.val
    sRd = sli_Rd.val
    sn = sli_n.val
    srd = sli_rd.val
    se = sli_e.val
    sN = sli_N.val
    sd = sli_d.val
    sD = sli_D.val
    frame = frame+1
    if sfm == 0:
        phi = 0
    if sfm != 0:

```

```

    phi = 2*np.pi*frame/sfm
    draw_pin_init()
    draw_inner_pin_init()
    draw_inner_circleA_init()
    draw_inner_circleB_init()
    draw_inner_circleC_init()
    #ehypocycloid_Pin_init()
    if pin_fig == 0:
        pin_update(sN,sd,sD,phi)
        ehypocycloid_Pin.set_visible(0)
        edot_Pin.set_visible(0)
    if pin_fig == 1:
        ehypocycloid_Pin.set_visible(1)
        edot_Pin.set_visible(1)
        update_ehypocycloid_Pin(se,sN,sD,sd, phi)
    inner_pin_update(sn,sN,srd,sRd,phi)
    #inner_pin_update(sn,sN,srd,sRd,0)
    drive_pin_update(sRm)

    inner_pinB.set_visible(1)
    dotB.set_visible(1)
    inner_pinC.set_visible(1)
    dotC.set_visible(1)
    ehypocycloidB.set_visible(1)
    ehypocycloidC.set_visible(1)
    edotB.set_visible(1)
    edotC.set_visible(1)
    if cycloid_fig == 3:
        update_inner_pinA(se,sRm, phi)
        update_inner_pinB(se,sRm, phi)
        update_inner_pinC(se,sRm, phi)
        update_inner_circleA(se,sn,sN,srd,sRd, phi)
        update_inner_circleB(se,sn,sN,srd,sRd, phi)
        update_inner_circleC(se,sn,sN,srd,sRd, phi)
        update_ehypocycloidA(se,sN,sD,sd, phi)
        update_ehypocycloidB(se,sN,sD,sd, phi)
        update_ehypocycloidC(se,sN,sD,sd, phi)
    if cycloid_fig == 2:
        update_inner_pinA(se,sRm, phi)
        update_inner_pinB(se,sRm, phi)
        update_inner_circleA(se,sn,sN,srd,sRd, phi)
        update_inner_circleB(se,sn,sN,srd,sRd, phi)
        update_ehypocycloidA(se,sN,sD,sd, phi)
        update_ehypocycloidB(se,sN,sD,sd, phi)
        edotC.set_visible(0)
        inner_pinC.set_visible(0)
        dotC.set_visible(0)
        ehypocycloidC.set_visible(0)
        edotC.set_visible(0)
    if cycloid_fig == 1:

```

```

update_inner_pinA(se,sRm, phi)
update_inner_circleA(se,sn,sN,srd,sRd, phi)
update_ehypocycloidA(se,sN,sD, sd, phi)
inner_pinB.set_visible(0)
dotB.set_visible(0)
inner_pinC.set_visible(0)
dotC.set_visible(0)
ehypocycloidB.set_visible(0)
ehypocycloidC.set_visible(0)
edotB.set_visible(0)
edotC.set_visible(0)

fig.canvas.draw_idle()

ani = animation.FuncAnimation(fig, animate, frames=sli_fm.val*(sli_N.val-1), interval=150)
dpi=100
plt.text(0, 4.8, f'Best fitness : {best_fitness}', transform=plt.gca().transAxes, fontsize=12)
plt.text(0, 4.4, f'Best position: {best_position}', transform=plt.gca().transAxes, fontsize=12)
plt.show()

```

4.3.3 Kiểm nghiệm và tinh chỉnh lại kết quả tối ưu sau khi đã chạy thuật toán tối ưu

4.3.3.1. Kết quả sau khi chạy thuật toán

Thông qua các hàm f_1 , f_2 , f_3 và giới hạn các biến được xác định trước ta tìm ra được các kết quả như sau:

```

PROBLEMS      OUTPUT      DEBUG CONSOLE      TERMINAL
● PS C:\Users\Mr.Anh\Desktop\code_HGT> python .\Code.py
Best Fitness: 0.9372036734477599
Best Position: [ 0.85 8. 4. 36. ]
○ PS C:\Users\Mr.Anh\Desktop\code_HGT>

```

Hình 4.3 Kết quả tối ưu sau khi chạy thuật toán

Kết quả tốt nhất cho hàm:

$$F_{\min} = 0.9372036734477599$$

Với vector bao gồm năm biến thiết kế được biểu thị bằng:

$$X = (D_z, d'_z, B, K_1, d'_w).$$

Trong đó:

- $K_1 = 0.85$
- $D_z = 36 \text{ mm}$
- $d'_z = 8 \text{ mm}$
- $B = 4 \text{ mm}$
- $d'w = d'_z = 8 \text{ mm}$

Sau quá trình phân tích, chúng ta đã nhận thấy rằng tất cả các kết quả đạt được đều nằm trong khoảng giá trị tìm kiếm mà chúng ta đã định trước. Trong trường hợp này, chúng ta đã chọn giới hạn dài giá trị dựa trên kích thước của hộp giảm tốc cycloid mà chúng ta đã thiết kế, để đảm bảo sự phù hợp với kích thước của động cơ. Việc giới hạn như vậy là cần thiết để đảm bảo tính đồng nhất và sự tương thích giữa các thành phần trong hệ thống.

Do đó, quyết định chọn kích thước $D_z = 36\text{mm}$ làm giá trị biên chuẩn để tiến hành điều chỉnh các biến số còn lại, nhằm đáp ứng các điều kiện biên và đáp ứng các yêu cầu của các thiết bị tiêu chuẩn, giúp việc chế tạo hộp giảm tốc cycloid trở nên dễ dàng hơn.

Qua việc lựa chọn kích thước biên chuẩn này, chúng ta sẽ tập trung vào việc tinh chỉnh các yếu tố khác nhằm đảm bảo sự tương thích và hoạt động ổn định của hộp giảm tốc cycloid trong quá trình sử dụng.

Dựa trên các công thức liên quan và thực tế các bộ phận tiêu chuẩn đi kèm với hộp giảm tốc cycloid mà ta thiết kế ta xác định được các biến số.

4.3.3.2. Kiểm nghiệm và tinh chỉnh kết quả để chế tạo

+ Theo mục 4.2.1 ta có hệ số biên độ ngắn cần thỏa mãn:

$$0.45 \leq K_1 \leq 0.8 \quad (4.3.1)$$

Tuy nhiên như đã nói ở trên ta sẽ ưu tiên chọn $D_z = 36\text{mm}$ làm kích thước chuẩn mà vòng bi là một thiết bị tiêu chuẩn với mã 6701zz ta có kích thước $12 \times 18 \times 4 \text{ mm}$ do đó $r_b = 9$ đã được xác định từ trước.

$$K_1 = \frac{r_b}{R_z} = \frac{9}{18} = 0.5$$

Thỏa mãn bất phương trình (4.2.1)

+ Theo mục 4.2.2 ta có :

$$g_3(X) = \frac{d'z}{D_z} - \frac{(1+K_1)^2}{(1+K_1+Z_g K_1)} \leq 0$$

$$g_3(X) = \frac{8}{36} - \frac{(1 + 0.5)^2}{(1 + 0.5 + 9 * 0.5)} = -\frac{11}{72} \leq 0 \quad (4.3.2)$$

Thỏa mãn điều kiện của bất phương trình (4.2.2)

+ Theo mục 4.2.3 ta có :

$$\begin{aligned} g_4(X) &= 2 * T - D_w + d_{sk} + D_1 \leq 0 \\ g_4(X) &= 2 * 1.08 - 24 + 10 + 18 = \frac{154}{25} \leq 0 \end{aligned} \quad (4.3.3)$$

Với :

$$T = 0.03D_z = 1.08 \text{ mm}$$

$$D_w = \frac{d_{fc} + D_1}{2} = \frac{30 + 18}{2} = 24 \text{ mm}$$

$$d_{sk} = d_w + 2e = d'_w + 2 = 10 \text{ mm}$$

Tại đây ta thấy với kích thước chốt đầu ra càng lớn sẽ càng bền, để không bị gãy tuy nhiên lại làm cho lỗ chốt trên bánh răng to lên và làm cho thành của chúng mỏng đi gây ra nứt gãy. Vì vậy giải pháp đưa ra để tránh gãy chốt và tránh hỏng bánh răng cycloid là chúng ta sẽ tăng số lượng chốt đầu ra lên và giảm kích thước chốt đầu ra xuống.

Do đó chốt đầu ra chúng ta sẽ chọn:

$$d_w = 3 \text{ mm}$$

+ Theo mục 4.2.4 ta có :

$$1.25 \leq K_2 \leq 4 \quad (4.3.4)$$

Tại đó:

$$\begin{aligned} K_2 &= \frac{D_z}{d'_z} \cdot \sin \frac{\pi}{Z_b} \\ 1.25 &\leq \frac{36}{4} * \sin \frac{\pi}{10} \leq 4 \end{aligned}$$

$$1.25 \leq 2.781152949 \leq 4$$

Thỏa mãn bất phương trình (4.2.4)

+ Theo mục 4.2.5 ta có độ bền uốn của chốt :

$$\begin{aligned} g_7(X) &= \frac{44L_1L_2T_v}{LK_1Z_gD_zd'^3_z} - \sigma_{FP} \leq 0 \\ \frac{44 * 4 * 4 * 41.82}{8 * 0.5 * 9 * 36 * 4^3} - 150 * 10^{-3} &\leq 0 \end{aligned} \quad (4.3.5)$$

$$-149.6450463 * 10^{-3} \leq 0$$

Ở đây ta có:

$$\sigma_{FP} = 150 \text{ Mpa}$$

$$L_1 = L_2 = 0.5B + 2 = 4$$

$$T_v = 41.82$$

$$L = 8$$

$$Z_g = 9$$

$$D_z = 36$$

$$d'_z = 4$$

$$K_1 = 0.5$$

Thỏa mãn độ bền uốn của chốt bánh răng.

+ Theo mục 4.2.6 cường độ tiếp xúc giữa chốt đầu ra và lỗ chốt :

$$g_8(X) = 0.0949 \sqrt{\frac{10K_1T_vD_z}{Z_wD_wB\left(r_w^2Z_b + \frac{D_zK_1}{2}r_w\right)}} - \sigma_{HP} \leq 0 \quad (4.3.6)$$

$$g_8(X) = 0.0949 \sqrt{\frac{10 * 0.5 * 41.82 * 36}{6 * 24 * 4\left(1.5^2 * 10 + \frac{36 * 0.5}{2} * 1.5\right)}} - 850 * 10^{-3} \leq 0$$

$$-0.8049526507 \leq 0$$

Với :

$$\sigma_{HP} = 850 \text{ Mpa}$$

$$L_1 = L_2 = 0.5B + 2 = 4$$

$$T_v = 41.82$$

$$r_w = 1.5$$

$$D_z = 36$$

$$d'_z = 4$$

$$Z_b = 10$$

$$K_1 = 0.5$$

Như vậy với các thông số trên thì cường độ tiếp xúc giữa chốt đầu ra và lỗ chốt thỏa mãn điều kiện .

+ Theo mục 4.2.7 chốt đầu ra phải thỏa mãn độ bền uốn

$$g_9(X) = \frac{96T_v(0.5B + \delta)}{Z_w R_w d_w'^3} - \sigma_{FP} \leq 0 \quad (4.3.7)$$

$$\frac{96 * 41.81(0.5 * 4 + 0)}{6 * 12 * 3^3} - 150 * 10^{-3} \leq 0$$

$3.9793827160493827 \leq 0$ (Không thỏa mãn)

Với :

$$\sigma_{FP} = 150 \text{ MPa}$$

$$B = 4 \text{ mm}$$

$$T_v = 41.82 \text{ Nm}$$

$$d_w = 3$$

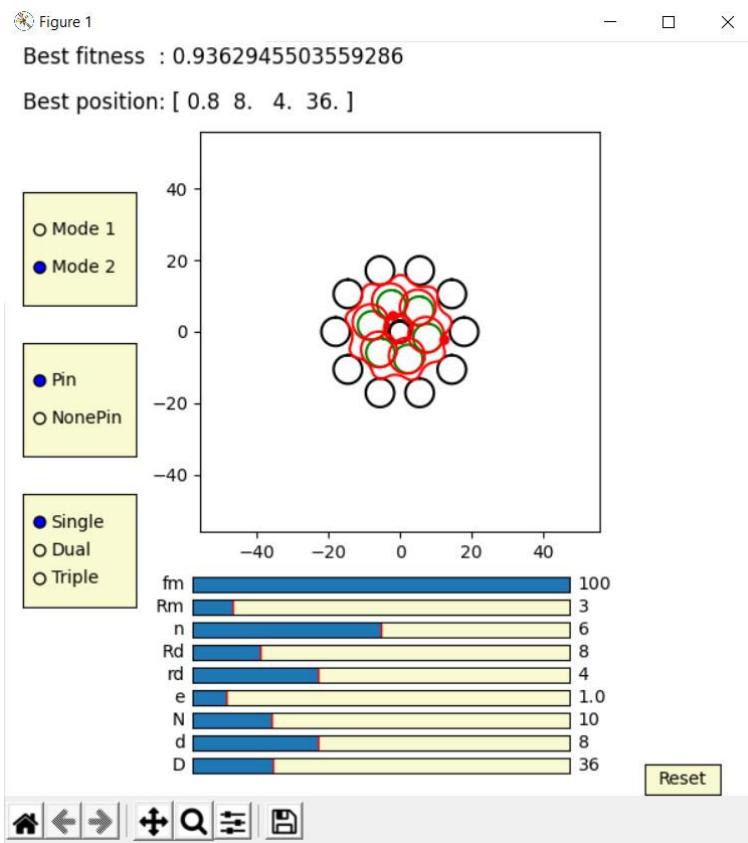
$$D_z = 36$$

$$d'_z = 4$$

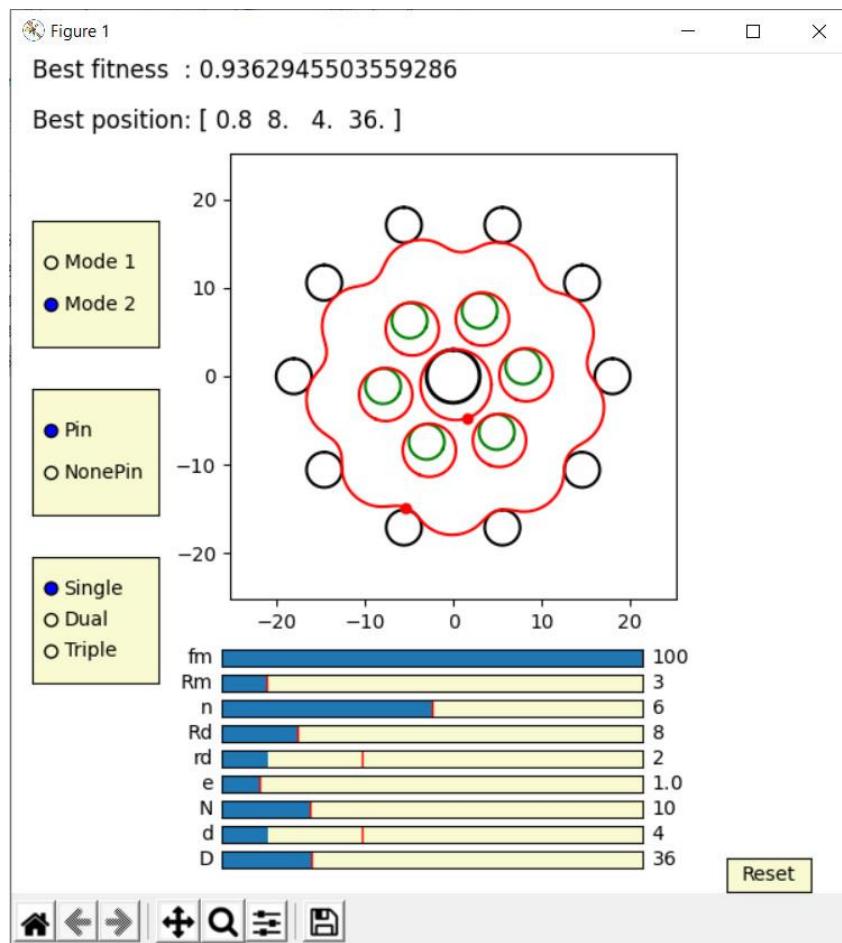
$$Z_b = 10$$

$$K_1 = 0.5$$

Do chúng ta đã chỉnh lại $d_w = 3 \text{ mm}$ nên tại đây nó đã không thỏa mãn được điều kiện độ bền uốn. Như đã nói ở trên ta sẽ tăng số lượng chốt đầu ra và giảm kích thước đường kính chốt đầu ra.



Hình 4.4 Kết quả trước tinh chỉnh bằng python



Hình 4.5 Kết quả sau tinh chỉnh bằng python

Sau khi kiểm tra và tinh chỉnh kết quả tối ưu chúng ta được một bảng thông số thiết kế như sau:

STT	Thông số	Kí hiệu	Giá trị	Đơn vị
1	Độ lệch tâm	e	1	N/A
2	Hệ số biên độ ngắn	K_1	0.5	N/A
3	Bán kính lỗ tâm bánh răng	r_b	9	mm
4	Bán kính của vòng tròn phân bố pin	R_z	18	mm
5	Số pin	Z_b	10	Cái
6	Số răng của bánh răng cycloid	Z_g	9	Cái
7	Mômen xoắn đầu ra	T_v	41.81	Nm
8	Mômen cản bánh răng cycloid	T_g	23	Nm
9	Đường kính vòng tròn phân phối pin	D_z	36	mm
10	Đường kính của chốt	d'_z	4	mm
11	Độ dày của bánh răng cycloid	B	10	mm
12	Đường kính của chốt đầu ra	d_w	3	mm
13	Độ dày của vòng bi nhỏ	b	4	mm
14	Độ dày thành bên của hộp giảm tốc	Δ	2	mm
15	Khoảng cách giữa bánh răng cycloid và mặt bên trong của vỏ	δ'	1	mm
16	Độ dày giữa hai lỗ chốt đầu ra liền kề	T	1.08	mm
17	Đường kính của vòng tròn phân phối lỗ pin đầu ra	D_w	24	mm
18	Đường kính của vòng chân răng của bánh răng cycloid	d_{fc}	30	mm
19	Đường kính của lỗ tâm bánh răng cycloid	D_1	10	mm
20	Đường kính của lỗ chốt	d_{sk}	6	mm
23	Ứng suất tiếp xúc cho phép	σ_{HP}	850	Mpa
24	Ứng suất uốn cho phép	σ_{FP}	150	Mpa
25	Bán kính của lỗ pin	r_w	2	mm
26	Tốc độ quay của ô lăn nhỏ	n	60	v/min
27	Momen giữ tối đa của động cơ nema 23	M	1.26	Nm
28	Số chốt đầu ra	Z_w	6	Cái

Bảng 4.3 Bảng số thiết kế của hộp giảm tốc

CHƯƠNG 5. THIẾT KẾ, CHẾ TẠO THỰC NGHIỆM HỘP GIẢM TỐC CYCLOID

5.1 Các thành phần cơ khí sau khi tinh chỉnh lại hộp giảm tốc Cycloid

5.1.1 Tổng quan về loại hộp giảm tốc cycloid mà nhóm thiết kế

Mặc dù hộp giảm tốc cycloid có nhiều ưu điểm, như tỷ số giảm tốc cao, khả năng chịu tải tốt và độ ổn định cao, nhưng nó cũng có một số nhược điểm:

- Độ chính xác gia công: Vì cấu trúc phức tạp và yêu cầu độ chính xác cao trong quá trình gia công, hộp giảm tốc cycloid thường đòi hỏi quy trình gia công phức tạp và công nghệ chính xác, từ đó làm tăng chi phí sản xuất và thời gian gia công.

- Độ ổn định: Một số mô hình hộp giảm tốc cycloid có thể gặp vấn đề về độ ổn định, đặc biệt trong các ứng dụng yêu cầu độ chính xác cao và độ tin cậy. Các vấn đề có thể bao gồm rung động, tiếng ồn và sự mất cân bằng trong quá trình hoạt động. Sau khi kiểm tra và tinh chỉnh kết quả tối ưu chúng ta được một bảng thông số thiết kế như sau.

- Kích thước và trọng lượng: Hộp giảm tốc cycloid thường có kích thước lớn và trọng lượng nặng hơn so với các loại hộp giảm tốc khác. Điều này có thể ảnh hưởng đến việc thiết kế và lắp đặt trong các ứng dụng có yêu cầu không gian hạn chế.

- Giá thành: Do tính phức tạp và yêu cầu gia công chính xác cao, hộp giảm tốc cycloid có xu hướng đắt hơn các loại hộp số khác. Điều này có thể làm tăng chi phí ban đầu và ảnh hưởng đến tính khả thi kinh tế của dự án.

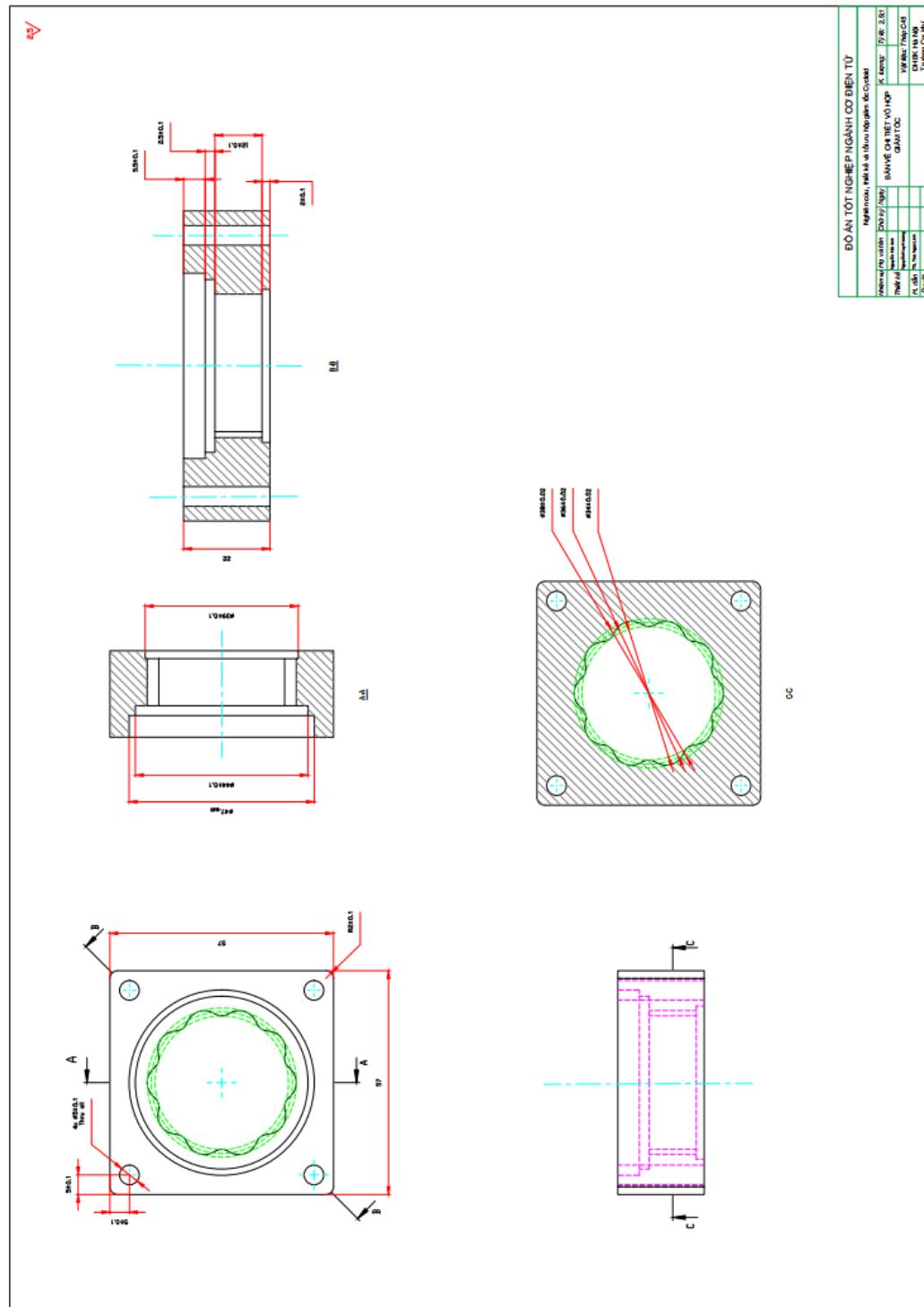
Do chưa có kinh nghiệm trong việc thiết kế cũng như trình độ chế tạo, gia công, lắp ráp hộp giảm tốc Cycloid với độ chính xác cao với nhiều lần thất bại. Chúng em quyết định chọn model đơn giản hơn để thực hiện thiết kế và chế tạo hộp giảm tốc Cycloid, Cycloid loại non pin (non-pin cycloid) là một dạng hộp giảm tốc cycloid trong đó không sử dụng các chốt hay chốt bi như các mô hình cycloid truyền thống. Thay vào đó, nó sử dụng một cấu trúc bên trong khác gọi là "gearless" để thực hiện chuyển động giảm tốc.

Trong hộp giảm tốc cycloid loại non pin, thay vì sử dụng các ròng rọc và chốt bi truyền thống, nó sử dụng một bộ truyền động có cấu trúc xoắn ốc và bánh

răng để tạo ra chuyển động giảm tốc. Cấu trúc này giúp giảm ma sát và tiếng ồn, đồng thời cung cấp hiệu suất và độ tin cậy cao hơn.

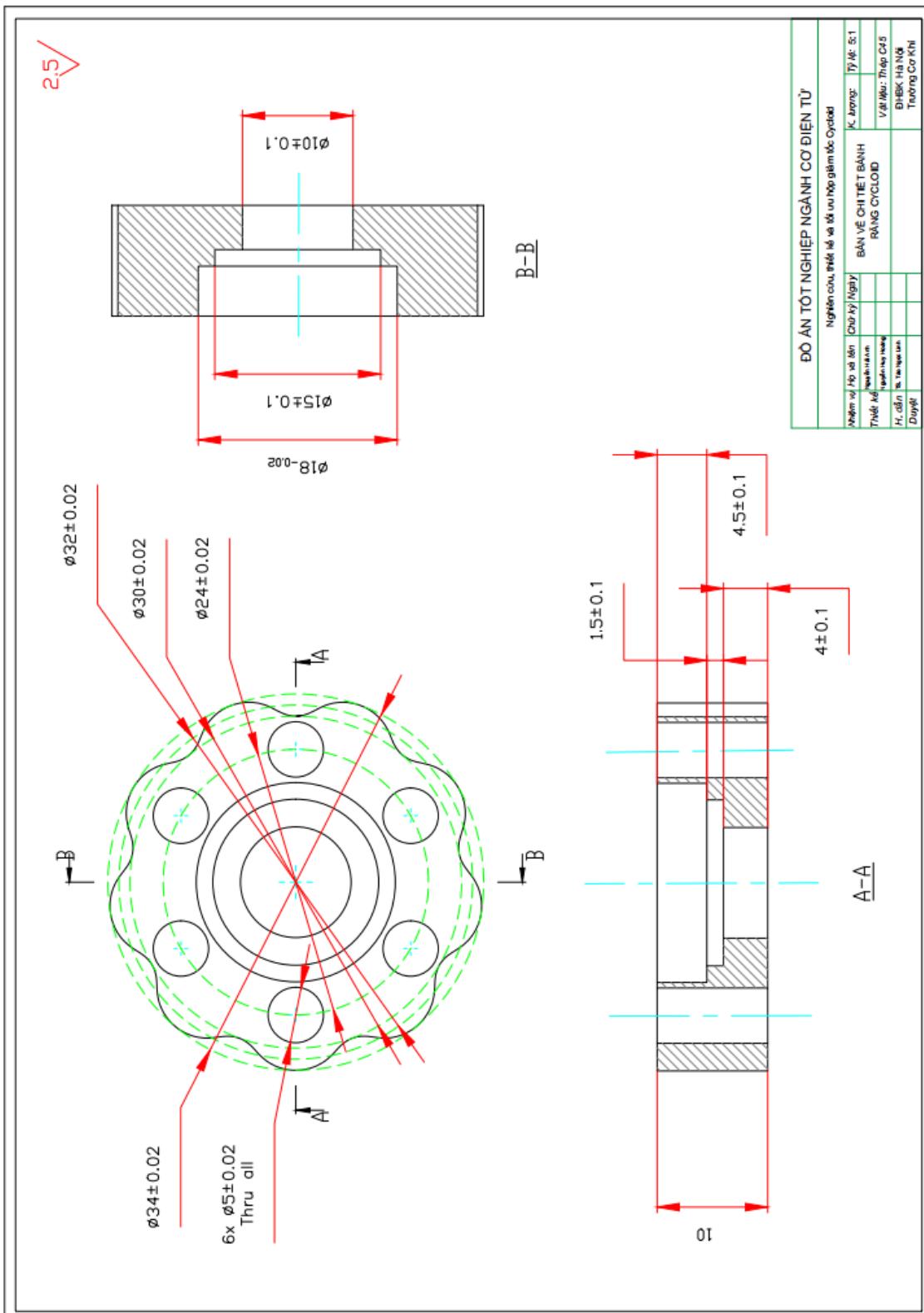
Hộp giảm tốc cycloid loại non pin thường được sử dụng trong các ứng dụng đòi hỏi độ chính xác và độ tin cậy cao, chẳng hạn như robot công nghiệp, máy tiện CNC, hệ thống truyền động trong công nghiệp và tự động hóa.

5.1.2 Vỏ hộp giảm tốc Cycloid



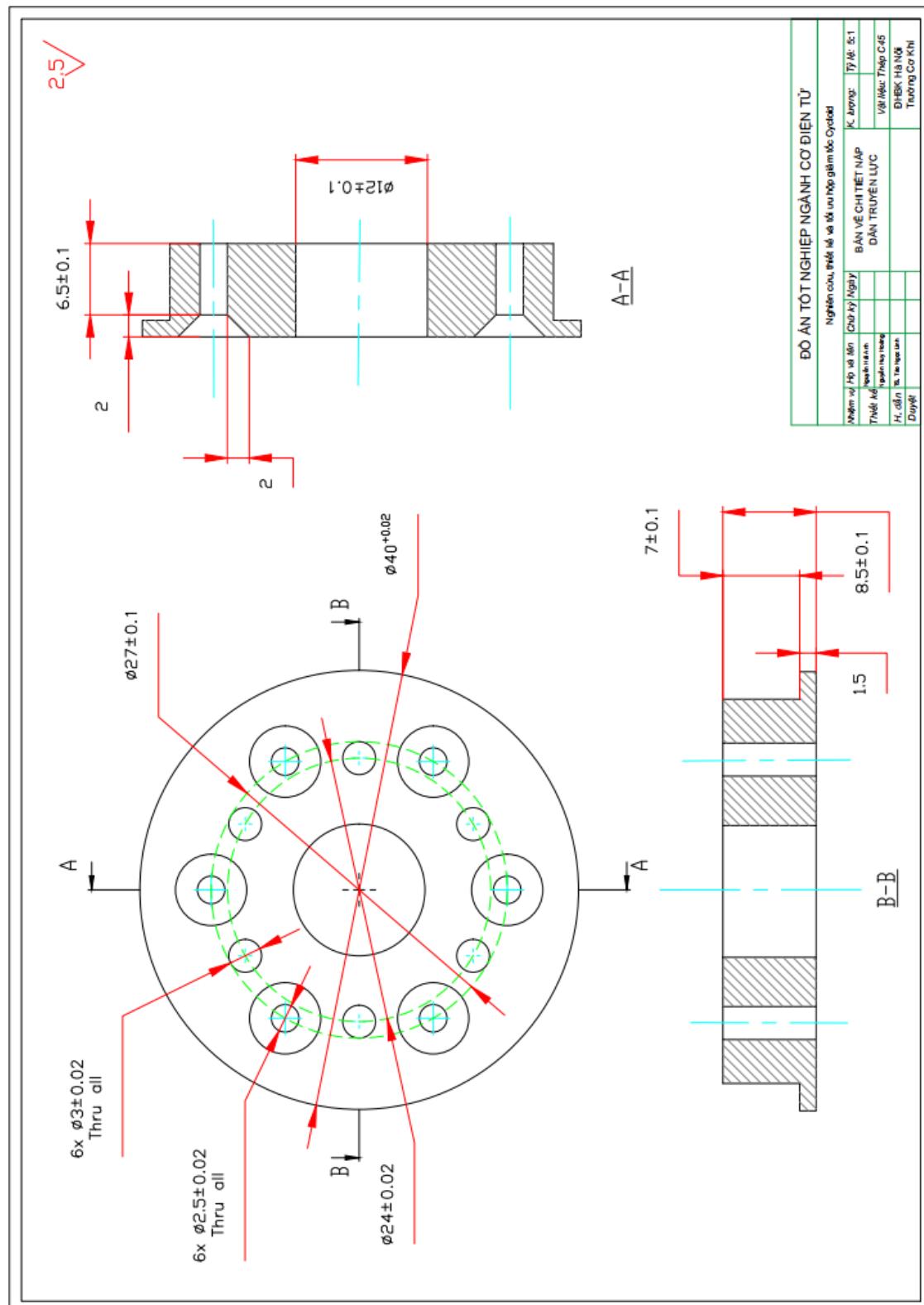
Hình 5.1 Bản vẽ chi tiết vỏ hộp giảm tốc

5.1.3 Bánh răng Cycloid



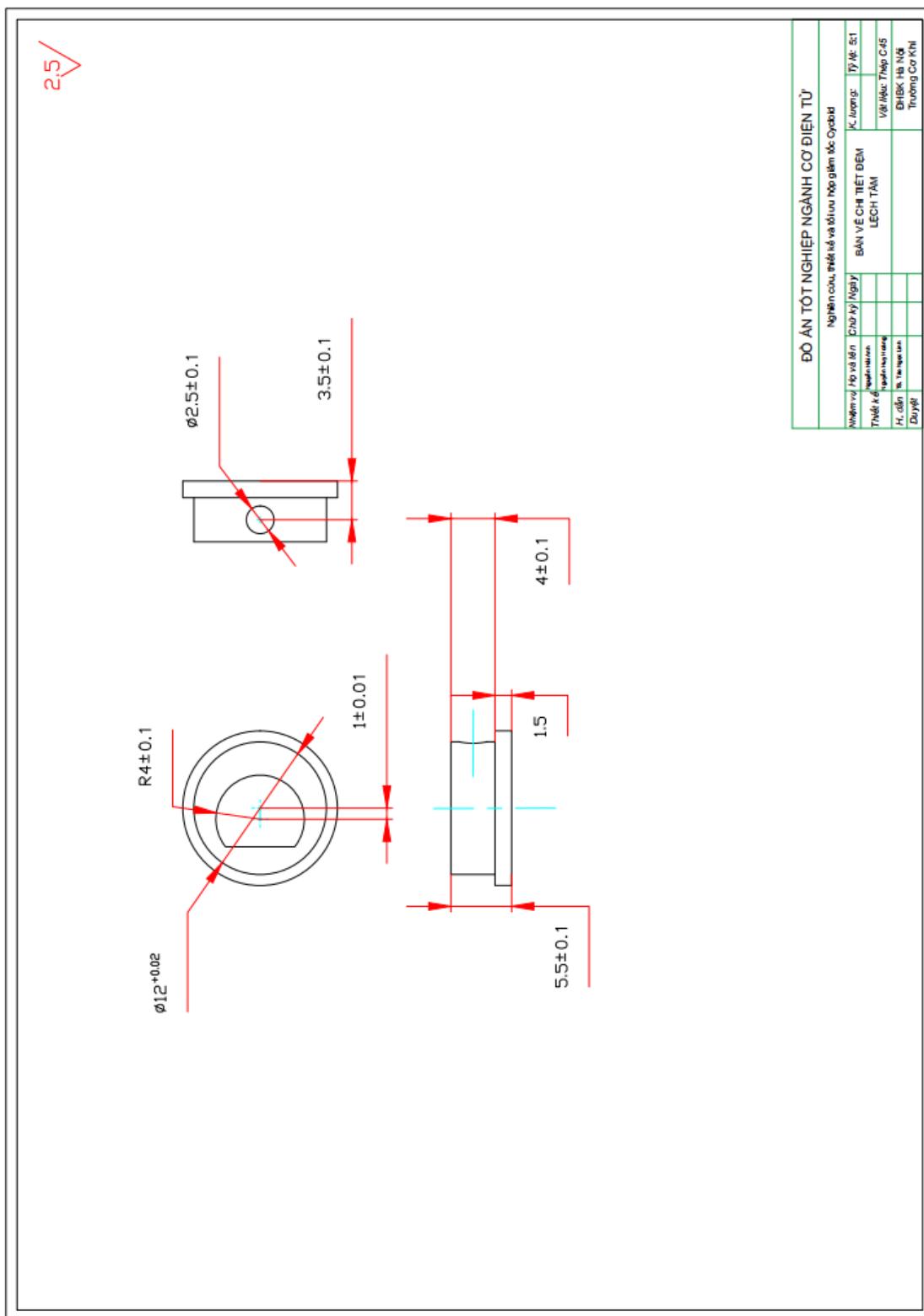
Hình 5.2 Bản vẽ chi tiết bánh răng Cycloid

5.1.4 Nắp dẫn truyền lực



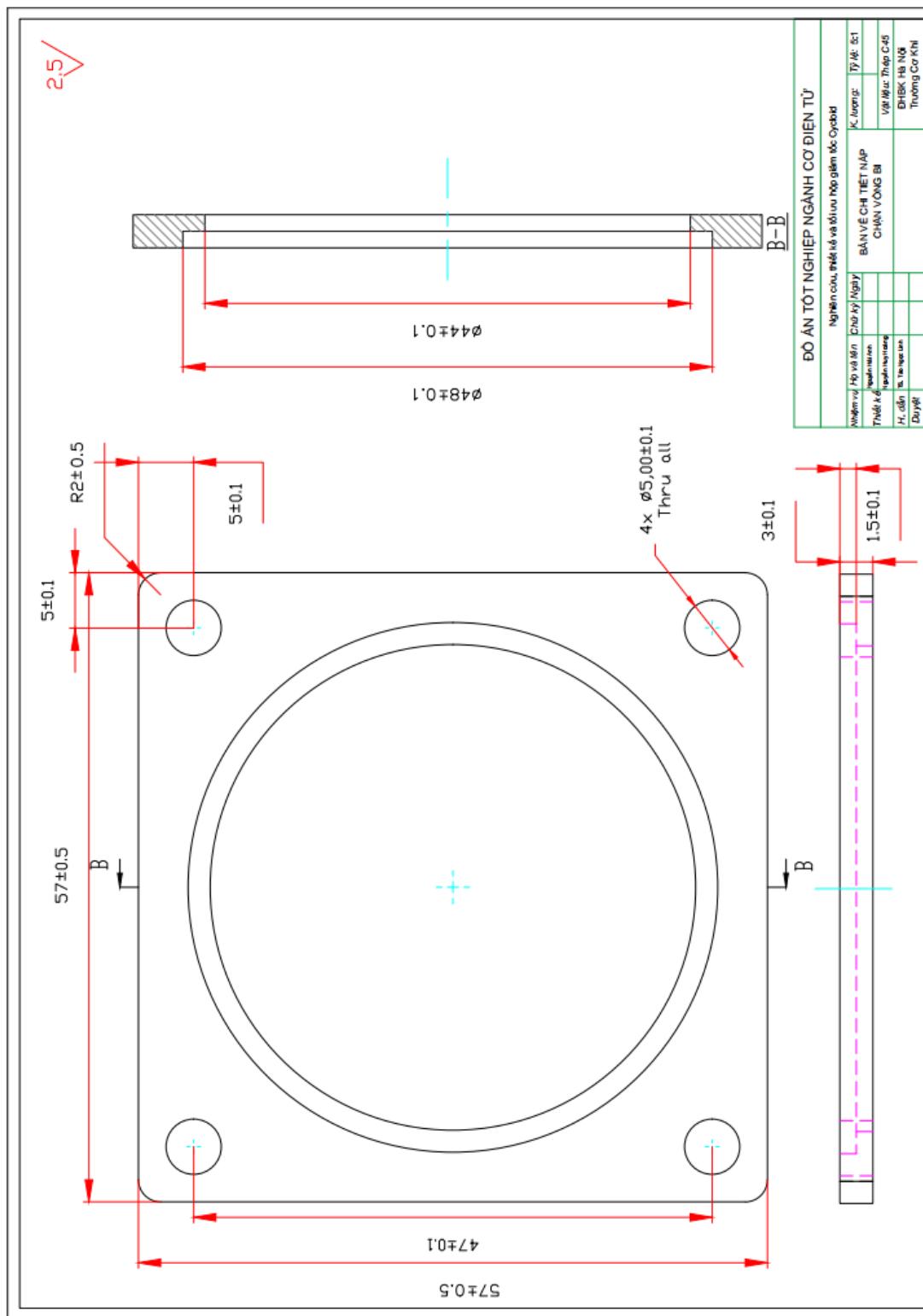
Hình 5.3 Bản vẽ chi tiết nắp dẫn truyền lực

5.1.5 Đệm lệch tâm



Hình 5.4 Bản vẽ chi tiết đệm lệch tâm

5.1.6 Nắp chặn vòng bi



Hình 5.5 Bản vẽ chi tiết nắp chặn vòng bi

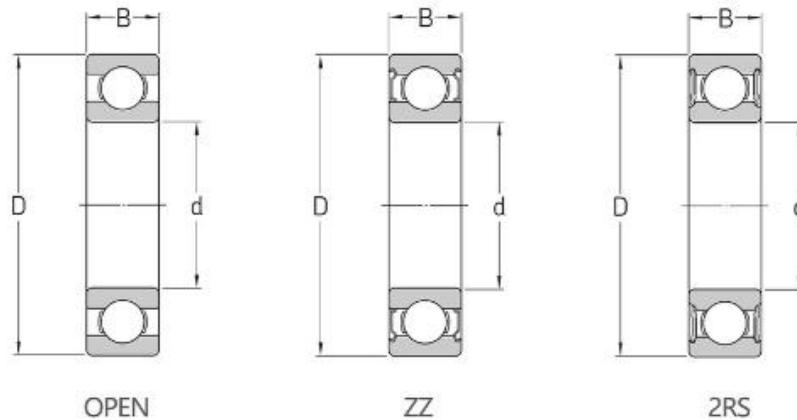
5.1.7 Vòng bi tại bạc lêch tâm và tại nắp đậy dẫn truyền

5.1.7.1. Vòng bi 6701zz:

Lý do chọn vòng bi 6701zz:

- Vòng bi 6701ZZ là vòng bi rãnh sâu với các tóm rộng (chỉ mặt đóng ZZ).
- Có một miếng chắn kim loại hai mặt để ngăn chặn bụi và các tạp chất vào bên trong vòng bi, giúp bảo vệ và kéo dài tuổi thọ của vòng bi.
- Vòng bi 6701 đây là loại vòng bi thông dụng được sử dụng trong nhiều ngành công nghiệp, bao gồm cả hộp giảm tốc, máy móc công nghiệp, ô tô, và các thiết bị điện tử

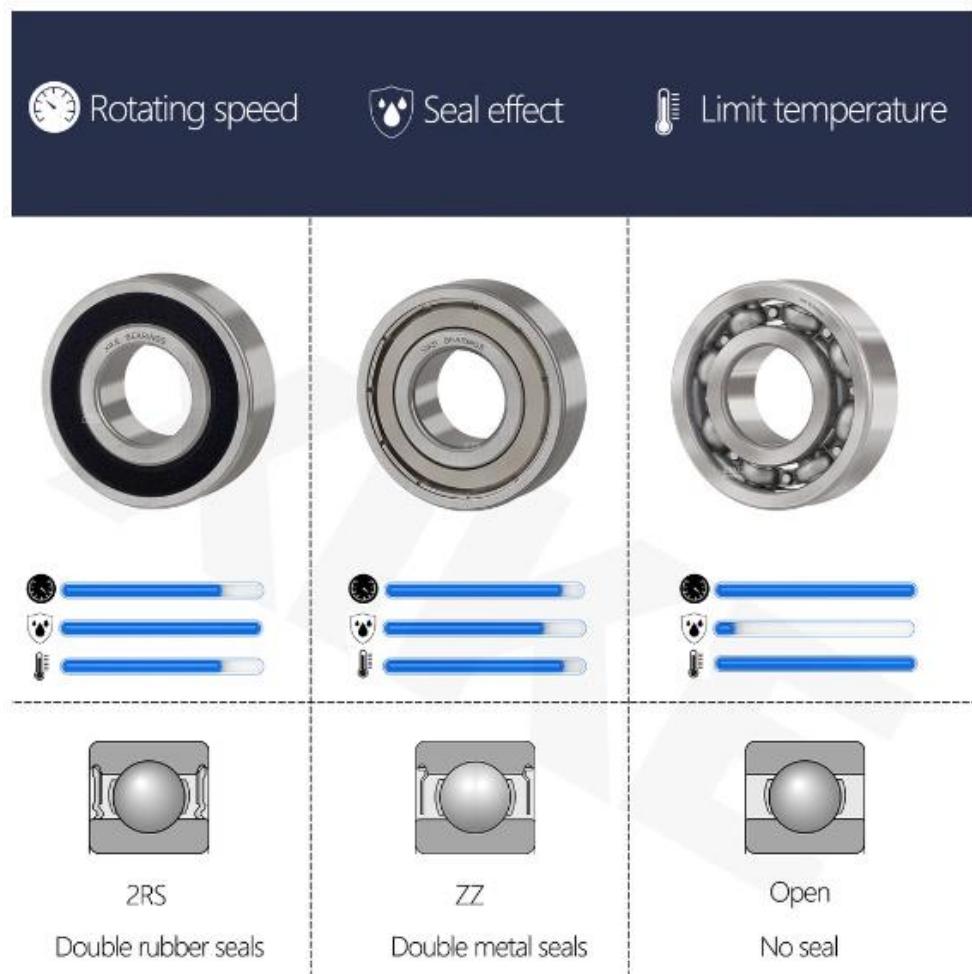
XIKE Model	Main dimensions / mm			Seal types	Material
	d	D	B		
6900	10	22	6	Open / ZZ / 2RS	Chrome steel
6901	12	24	6	Open / ZZ / 2RS	Chrome steel
6902	15	28	7	Open / ZZ / 2RS	Chrome steel
6903	17	30	7	Open / ZZ / 2RS	Chrome steel
6904	20	37	9	Open / ZZ / 2RS	Chrome steel
6905	25	42	9	Open / ZZ / 2RS	Chrome steel
6906	30	47	9	Open / ZZ / 2RS	Chrome steel
6907	35	55	10	Open / ZZ / 2RS	Chrome steel
6908	40	62	12	Open / ZZ / 2RS	Chrome steel
6909	45	68	12	Open / ZZ / 2RS	Chrome steel
6910	50	72	12	Open / ZZ / 2RS	Chrome steel



Hình 5.6 Tiêu chuẩn vòng bi

5.1.7.2. Vòng bi 6807zz:

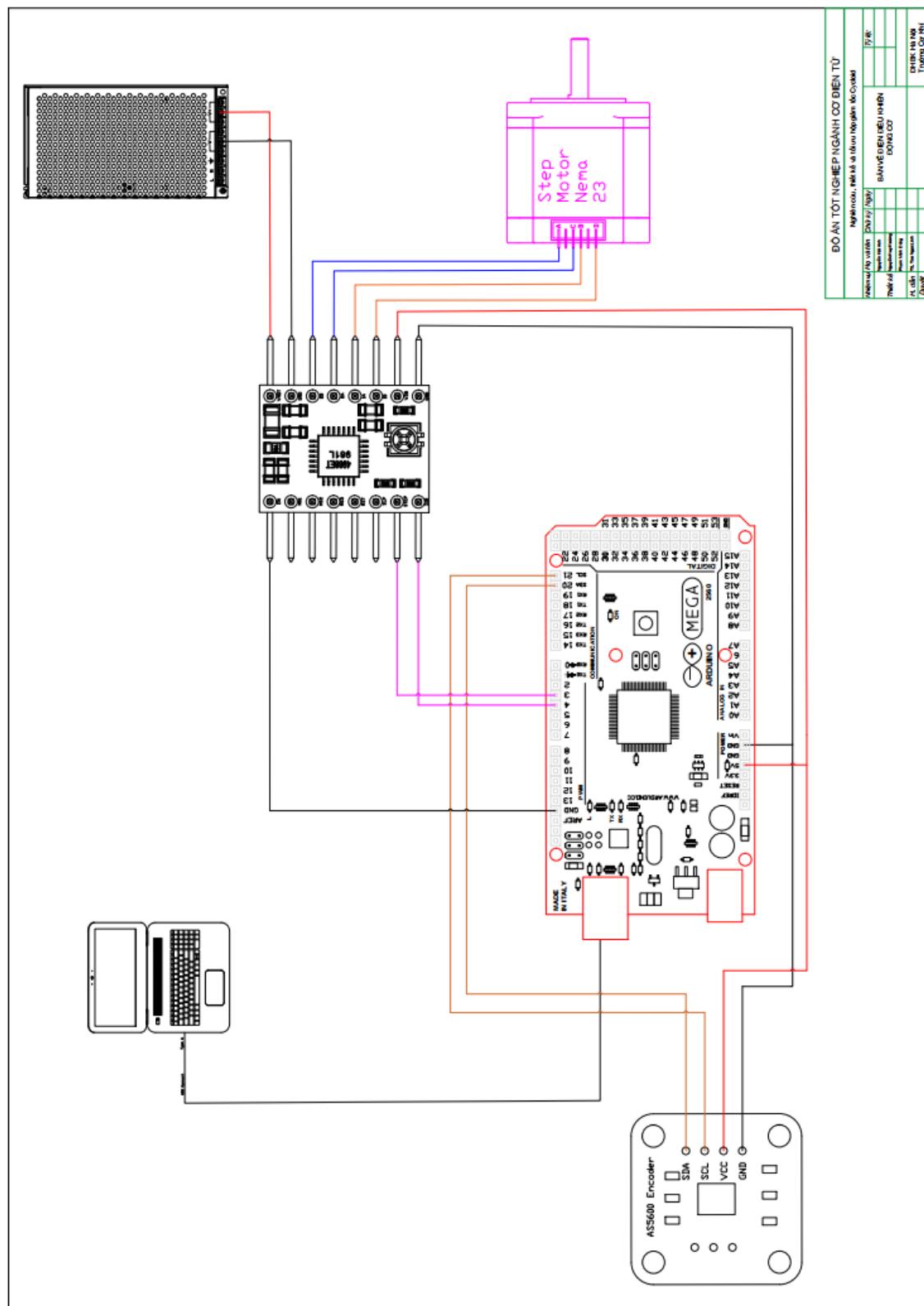
Vòng bi có miếng chặn bụi kim loại cả hai mặt (Sealed Bearing): Một phiên bản của vòng bi 6807 có thể có miếng chặn bụi kim loại cả hai mặt, giúp bảo vệ bên trong vòng bi khỏi bụi, nước và các tạp chất bên ngoài.



Hình 5.7 Phân loại vòng bi 6807

5.2 Các thành phần điều khiển và giám sát hộp giảm tốc Cycloid

5.2.1 Bản vẽ điện



Hình 5.8 Sơ đồ đi dây

5.2.2 Lập trình điều khiển trên Arduino

5.2.2.1. Khai báo thư viện và các giá trị biến

```
#include <Wire.h>
#include <TimerOne.h>
#include <PID_v1.h>
#include <SimpleKalmanFilter.h>
```

5.2.2.2. Khai báo chân cảm

```
#define stepsPerRevolution 70
#define pulse 3
#define dir 4
```

5.2.2.3. Khai báo giá trị biến và hằng số

```
int magnetStatus = 0;

int lowbyte;
word highbyte;
int rawAngle;
float degAngle;

int quadrantNumber, previousquadrantNumber;
int kanj = 0;
float numberofTurns = 0;
float correctedAngle = 0;
float startAngle = 0;
float totalAngle = 0;
float old_angle = 0;
float previoustotalAngle = 0;
unsigned int count = 50;
int speed = 0;
int flag = 0;
double kp = 02 , ki = 05 , kd = 0 ,input = 0, output = 0, setpoint = 0;
unsigned long time;
```

5.2.2.4. Khai báo hàm PID

```
PID myPID(&input, &output, &setpoint, kp, ki, kd, REVERSE);
SimpleKalmanFilter bo_loc(2,2,0.001);
```

5.2.2.5. Thiết lập (Setup)

```
void setup()
{
    Serial.begin(115200);
    Wire.begin();
    Wire.setWireTimeout(3000,true);
    pinMode(dir, OUTPUT);
    pinMode(pulse, OUTPUT);
    pinMode(50, OUTPUT);
    Wire.setClock(800000);
    checkMagnetPresence();
    ReadRawAngle();
```

```

digitalWrite(dir,1);
startAngle = degAngle;
Timer1.initialize(10000);
Timer1.attachInterrupt(Control_Stepper);

myPID.SetMode(AUTOMATIC);
myPID.SetSampleTime(100);
myPID.SetOutputLimits(500, 10000);
setpoint = 0;
time = millis();
}

```

5.2.2.6. Vòng lặp (Loop)

```

void loop()
{
    if ( (unsigned long) (millis() - time) > 49)
    {
        ReadRawAngle();
        correctAngle();
        checkQuadrant();
        input = bo_loc.updateEstimate((totalAngle-old_angle)*20/6);
        old_angle=totalAngle;
        totalAngle =0;

        if(input < 0.5) input = 0;

        myPID.Compute();
        Timer1.setPeriod(output);
        Serial.println("vt="+String(input));
        time = millis();
    }
    if (Serial.available() > 0)
    {
        String inputString = Serial.readStringUntil('\n');
        String setpoint1String = String(inputString[0])
+String(inputString[1]) +String(inputString[2]);
        setpoint = setpoint1String.toInt();
        String kpString = String(inputString[3]) +String(inputString[4]);
        kp = kpString.toInt();
        String kiString = String(inputString[5]) +String(inputString[6]);
        ki = kiString.toInt()/100;
        String kdString = String(inputString[7]) +String(inputString[8]);
        kd = kdString.toInt()/100;
        //Serial.println("vt=124");

    }
}
void Control_Stepper(void)
{

```

```

if (flag == 1)
{
    digitalWrite(pulse,1);
    flag = 0;
}
else
{
    digitalWrite(pulse,0);
    flag = 1;
}

void ReadRawAngle()
{
    Wire.beginTransmission(0x36);
    Wire.write(0x0D);
    Wire.endTransmission();
    Wire.requestFrom(0x36, 1);

    while(Wire.available() == 0);
    lowbyte = Wire.read();

    Wire.beginTransmission(0x36);
    Wire.write(0x0C);
    Wire.endTransmission();
    Wire.requestFrom(0x36, 1);

    while(Wire.available() == 0);
    highbyte = Wire.read();
    highbyte = highbyte << 8;
    rawAngle = highbyte | lowbyte;
    degAngle = rawAngle * 0.087890625;
}

void correctAngle()
{
    correctedAngle = degAngle - startAngle;
    if(correctedAngle < 0)
    {
        correctedAngle = correctedAngle + 360;
    }
}

void checkQuadrant()
{
    if(correctedAngle >= 0 && correctedAngle <=90)
    {
        quadrantNumber = 1;
    }
}

```

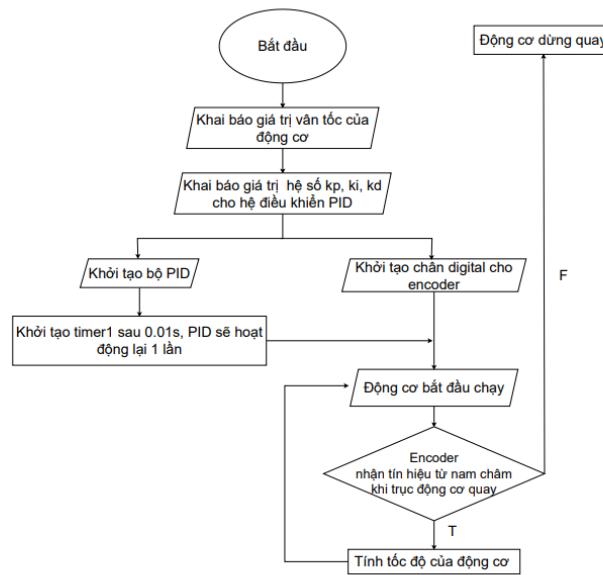
```

if(correctedAngle > 90 && correctedAngle <=180)
{
    quadrantNumber = 2;
}
if(correctedAngle > 180 && correctedAngle <=270)
{
    quadrantNumber = 3;
}
if(correctedAngle > 270 && correctedAngle <360)
{
    quadrantNumber = 4;
}
if(quadrantNumber != previousquadrantNumber
{
    if(quadrantNumber == 1 && previousquadrantNumber == 4)
    {
        numberofTurns++;
    }
    if(quadrantNumber == 4 && previousquadrantNumber == 1)
    {
        numberofTurns--;
    }
    previousquadrantNumber = quadrantNumber;
}
totalAngle = (numberofTurns*360) + correctedAngle
}
void checkMagnetPresence()
{
    while((magnetStatus & 32) != 32)
    {
        magnetStatus = 0;
        Wire.beginTransmission(0x36);
        Wire.write(0x0B);
        Wire.endTransmission();
        Wire.requestFrom(0x36, 1);
        while(Wire.available() == 0);
        magnetStatus = Wire.read();
    }
}

```

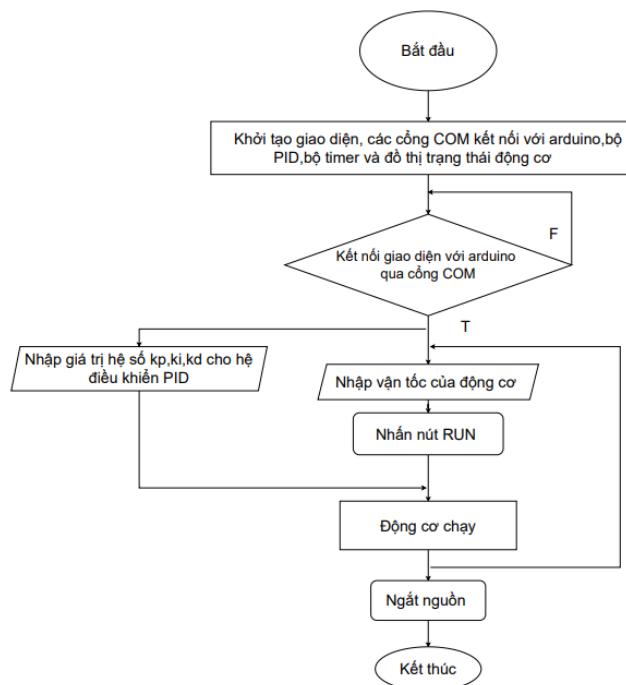
5.3 Thuật toán điều khiển

SƠ ĐỒ THUẬT TOÁN ĐIỀU KHIỂN



Hình 5.9 Sơ đồ thuật toán điều khiển

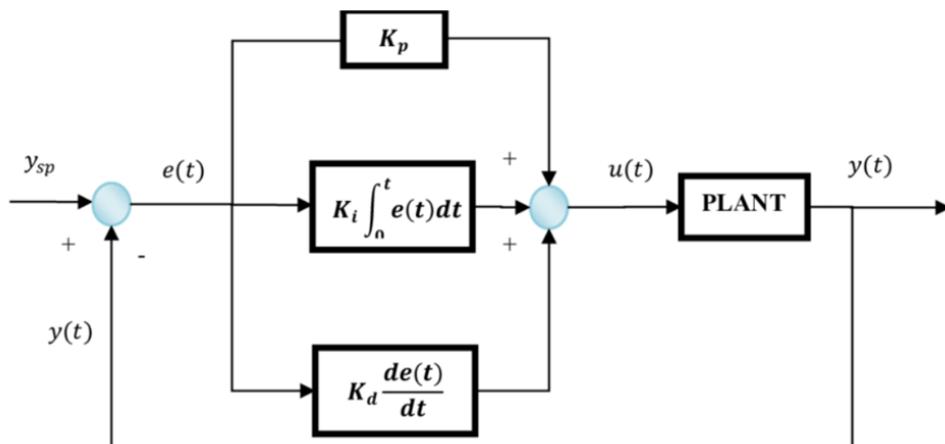
SƠ ĐỒ NGUYÊN LÝ HOẠT ĐỘNG



Hình 5.10 Sơ đồ nguyên lý hoạt động

5.3.1 Bộ điều khiển PID

PID (Proportional Integral Derivative) là một cơ chế phản hồi vòng điều khiển được sử dụng rộng rãi trong các hệ thống điều khiển công nghiệp. Bộ điều khiển PID được sử dụng nhiều nhất trong các hệ thống điều khiển vòng kín (có tín hiệu phản hồi). Bộ điều khiển PID sẽ tính toán giá trị sai số là hiệu số giữa giá trị đo thông số biến đổi và giá trị đặt mong muốn. Bộ điều khiển sẽ thực hiện giảm tối đa sai số bằng cách điều chỉnh giá trị điều khiển đầu vào. Để đạt được kết quả tốt nhất, các thông số PID sử dụng trong tính toán phải điều chỉnh theo tính chất của hệ thống - trong khi kiểu điều khiển là giống nhau, các thông số phải phụ thuộc vào đặc thù của hệ thống.



Hình 5.11 Bộ PID truyền thống

Trong đó:

- P (Proportional): là phương pháp điều chỉnh tỉ lệ, giúp tạo ra tín hiệu điều chỉnh tỉ lệ với sai lệch đầu vào theo thời gian lấy mẫu.

- I (Integral): là tích phân của sai lệch theo thời gian lấy mẫu. Điều khiển tích phân là phương pháp điều chỉnh để tạo ra các tín hiệu điều chỉnh sao cho độ sai lệch giảm về 0. Từ đó cho ta biết tổng sai số tức thời theo thời gian hay sai số tích lũy trong quá khứ. Khi thời gian càng nhỏ thể hiện tác động điều chỉnh tích phân càng mạnh, tương ứng với độ lệch càng nhỏ.

- D (Derivative): là vi phân của sai lệch. Điều khiển vi phân tạo ra tín hiệu điều chỉnh sao cho tỉ lệ với tốc độ thay đổi sai lệch đầu vào. Thời gian càng lớn thì

phạm vi điều chỉnh vi phân càng mạnh, tương ứng với bộ điều chỉnh đáp ứng với thay đổi đầu vào càng nhanh.

5.3.2 Phương pháp Ziegler-Nichols

Thông thường việc chọn các thông số P, I, D được xác định bằng thực nghiệm dựa vào đáp ứng xung của hệ thống. Ziegler – Nichols đưa ra phương pháp chọn tham số PID cho mô hình quán tính bậc nhất có trễ. Ở đây ta xấp xỉ hàm truyền của động cơ để dùng phương pháp này, tuy không hoàn toàn chính xác nhưng có thể cho đáp ứng tương đối tốt.

Phương pháp này đòi hỏi phải tính được giá trị giới hạn của của khâu tỉ lệ K_{gh} và chu kì giới hạn của hệ kín T_{gh} . Sau đó tìm các thông số khác theo bảng sau:

Bộ điều khiển	K_p	T_I	T_D
P	$0.5 \cdot K_{gh}$	-	-
PI	$0.45 \cdot K_{gh}$	$0.83 \cdot T_{gh}$	-
PID	$0.6 \cdot K_{gh}$	$0.5 \cdot T_{gh}$	$0.125 \cdot T_{gh}$

Bảng 5.1 Tính toán hệ số bằng phương pháp Ziegler - Nichols

Để tìm được K_{gh} và T_{gh} , ban đầu ta chỉnh K_i , K_d bằng 0 sau đó tăng từ từ K_p để hệ thống ở biên giới ổn định (dao động với biên độ và chu kì không đổi), tại đây ta xác định được K_{gh} và T_{gh} sau đó tính các thông số khác tùy theo bộ điều khiển như bảng trên.

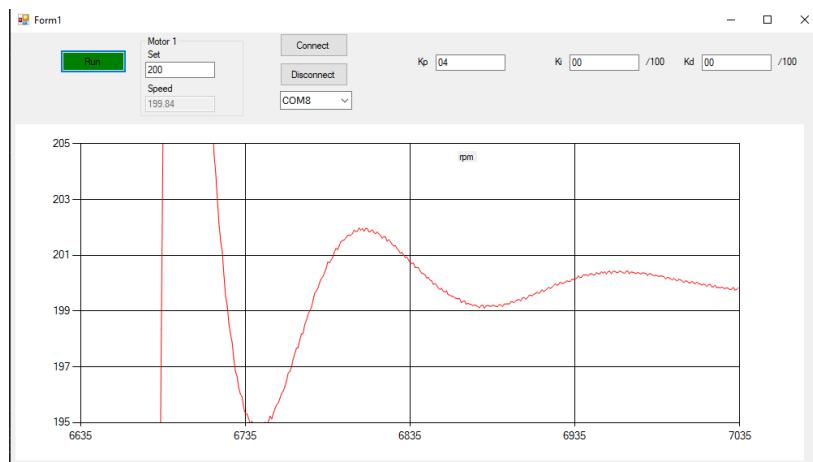
$$K_i = K_p / T_I$$

$$K_d = K_p * T_D$$

Để thuận tiện trong quá trình điều chỉnh và quan sát đáp ứng, nhóm đã xây dựng chương trình viết bằng Visual Studio giao tiếp với mạch điều khiển lập trình bằng Matlab/Simulink.

5.3.3 Tính toán các tham số của bộ điều khiển P

Tăng dần K_p , tại biên giới ổn định ta có được đồ thị vận tốc



Hình 5.12 Hệ ở biên giới ổn định

Khi vận tốc động cơ ở biên giới ổn định này, ta có được

$$+ K_{gh} = 4$$

$$+ T_{gh} = 0.2(s)$$

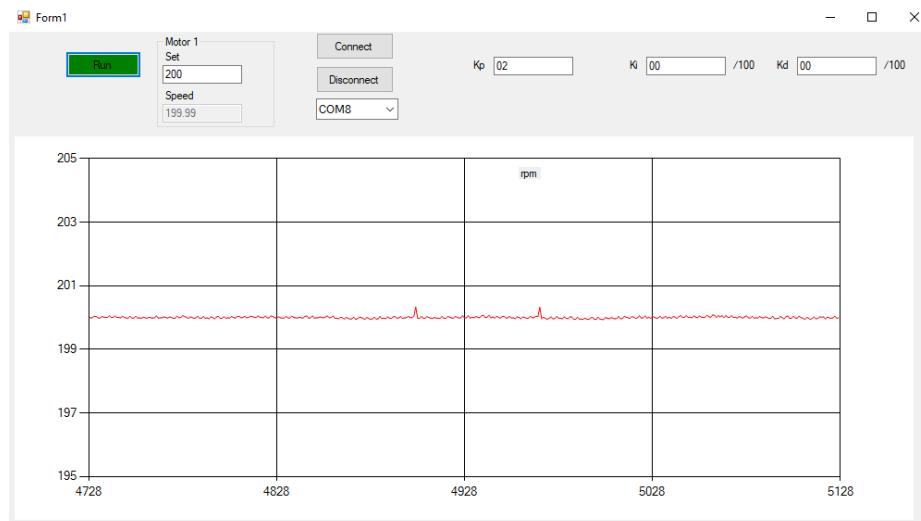
Vậy

$$+ K_p = 2$$

$$+ K_i = 0$$

$$+ K_d = 0$$

Đáp ứng vận tốc của động cơ lúc này



Hình 5.13 Đáp ứng của hệ với bộ điều khiển PID

5.3.3.1. Tính tham số của bộ điều khiển PI só:

Khi vận tốc động cơ ở biên giới ổn định này, ta có được

$$+ K_{gh} = 4$$

$$+ T_{gh} = 0.2(s)$$

Vậy

$$+ K_p = 1.8$$

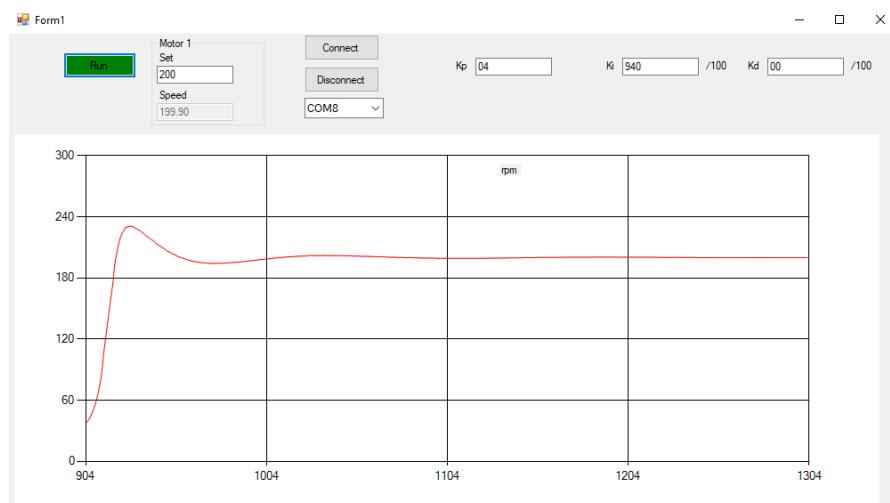
$$+ T_i = 0.17$$

$$+ T_d = 0$$

$$+ K_i = 9.4$$

$$+ K_d = 0$$

Đáp ứng vận tốc của động cơ lúc này:



Hình 5.14 Đáp ứng của hệ với bộ điều khiển PI

5.3.3.2. Tính toán tham số của bộ điều khiển PID số :

Khi vận tốc động cơ ở biên giới ổn định này, ta có được

$$+ K_{gh} = 4$$

$$+ T_{gh} = 0.2(s)$$

Vậy

$$+ K_p = 2.4$$

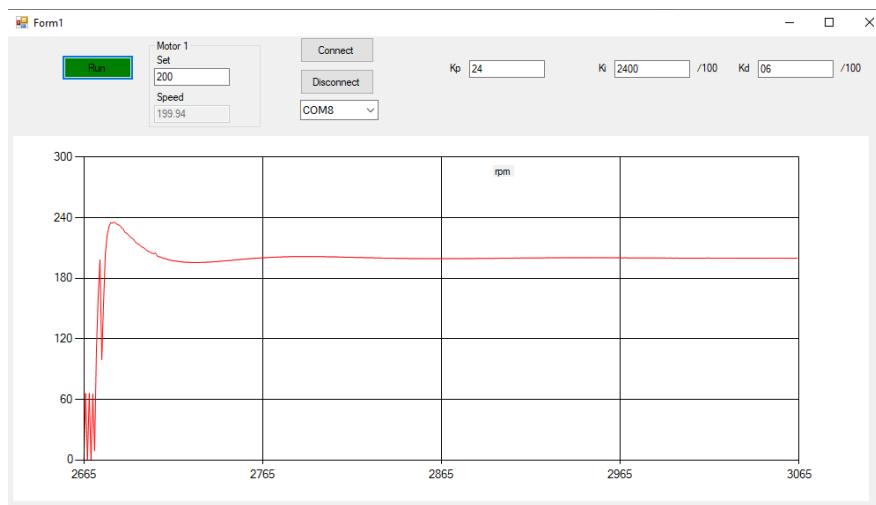
$$+ T_i = 0.1$$

$$+ T_d = 0.025$$

$$+ K_i = 24$$

$$+ K_d = 0.06$$

Đáp ứng vận tốc của động cơ lúc này:



Hình 5.15 Đáp ứng của hệ với bộ điều khiển PID

5.3.3.3. Nhận xét

Từ đồ thị của phương pháp điều chỉnh hệ số **Ziegler-Nichols**, ta rút ra được những nhận xét như sau:

- Bộ điều khiển tỉ lệ có độ vọt lồ nhỏ, sai lệch ít, không sử dụng được trong trường hợp điều khiển tốc độ động cơ

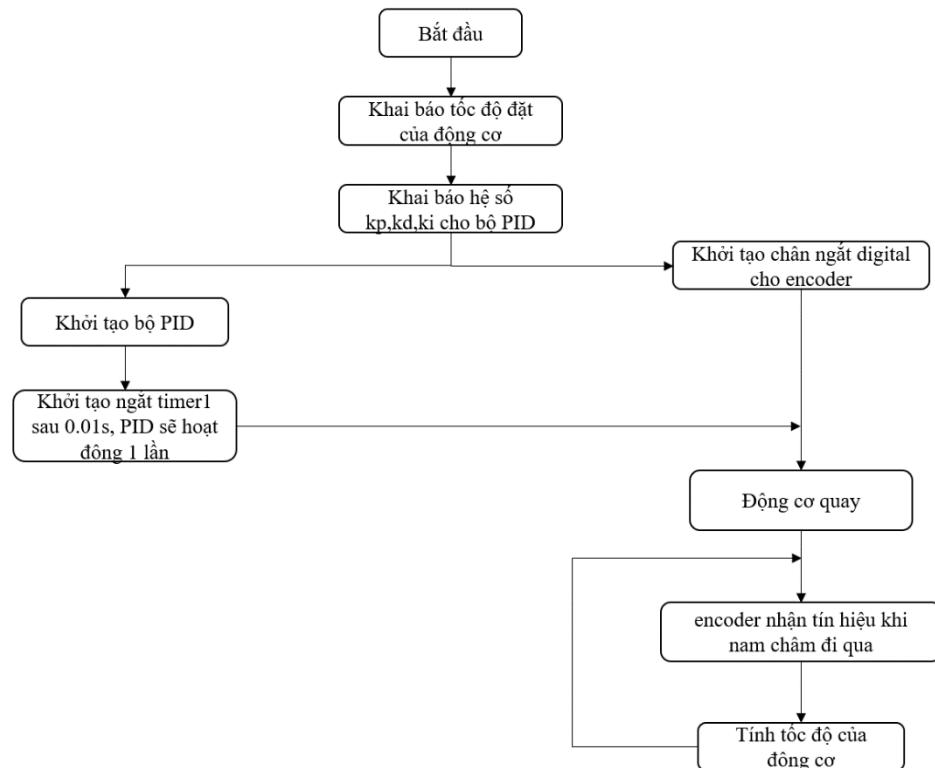
- 2 bộ điều khiển còn lại (PI, PID) đều giúp ổn định đối tượng một cách nhanh chóng, độ vọt lồ nhỏ có thể chấp nhận được. Tuy nhiên mỗi bộ điều khiển có những ưu nhược điểm nhất định.

- Với bộ điều khiển tỉ lệ (P), hệ đáp ứng rất nhanh, tuy nhiên sai số xác lập lại cực kì lớn (40 vòng / giây).

- Với bộ điều khiển tỉ lệ - tích phân (PI), hệ đi vào ổn định một cách rất nhanh chóng, sai số xác lập đạt giá trị nhỏ nhất (< 4 vòng / giây), tuy nhiên ở khâu quá độ, tốc độ động cơ có biên độ rất lớn, độ vọt lồ vẫn còn lớn. Ưu điểm của bộ điều khiển tỉ lệ - tích phân (PI) với hệ động cơ bước là sai số nhỏ, đáp ứng nhanh, đầu ra của hệ bám giá trị đặt cho trước.

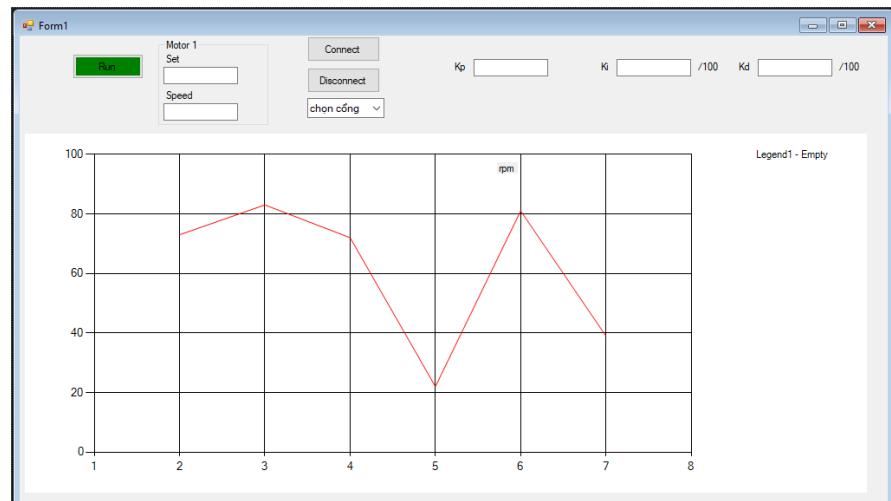
- Với bộ điều khiển tỉ lệ - tích phân – vi phân (PID), hệ đi vào ổn định cũng rất nhanh, tuy nhiên sai số xác lập vẫn còn lớn (< 8 vòng/ giây, lớn hơn so với bộ điều khiển PI), khâu quá độ diễn ra lâu hơn, hệ đáp ứng chậm hơn bộ điều khiển PI số.

- ⇒ Như vậy, nên chọn bộ điều khiển PI làm bộ điều khiển cho động cơ bước.
- ⇒ Phương pháp **Ziegler-Nichols** điều khiển khá tối ưu.



Hình 5.16 Sơ đồ thuật toán điều khiển tốc độ động cơ

5.4 Giao diện điều khiển Winform



Hình 5.17 Giao diện chính

5.4.1 Tạo hình ảnh và giao diện cho Winform (Front End)

```
1  #pragma once
2
3
4  namespace ArduinoPIDmotorspeed {
5      #include <math.h>
6      #define PI 3.14159265
7      int flag = 0;
8      int t = 0;
9      int set = 0;
10
11     using namespace System;
12     using namespace System::ComponentModel;
13     using namespace System::Collections;
14     using namespace System::Windows::Forms;
15     using namespace System::Data;
16     using namespace System::Drawing;
17     using namespace System::IO::Ports;
18     using namespace System::Threading;
19
20     /// <summary>
21     /// Summary for Form1
22     ///
23     /// WARNING: If you change the name of this class, you will need to change the
24     ///          'Resource File Name' property for the managed resource compiler tool
25     ///          associated with all .resx files this class depends on. Otherwise,
26     ///          the designers will not be able to interact properly with localized
27     ///          resources associated with this form.
28     /// </summary>
29     public ref class Form1 : public System::Windows::Forms::Form
30     {
31         public:
32             Form1(void)
33             {
34                 InitializeComponent();
35                 // TODO: Add the constructor code here
36                 //
37             }
38
39         protected:
40             /// <summary>
41             /// Clean up any resources being used.
42             /// </summary>
43             ~Form1()
```

```
44     {
45         if (components)
46         {
47             delete components;
48         }
49     }
50     private: System::IO::Ports::SerialPort^ serialPort1;
51     private: System::Windows::Forms::Button^ button2;
52
53     private: System::Windows::Forms::TextBox^ textBox1;
54
55     private: System::Windows::Forms::Button^ button1;
56
57     private: System::Windows::Forms::Label^ label1;
58
59
60
61     private: System::Windows::Forms::Timer^ timer1;
62
63
64
65     protected:
66     private: System::ComponentModel::.IContainer^ components;
67
68     private:
69         /// <summary>
70         /// Required designer variable.
71         String^ mStr;
72         String^ speed;
73     private: System::Windows::Forms::Label^ label5;
74     private: System::Windows::Forms::GroupBox^ groupBox1;
75     private: System::Windows::Forms::Label^ label6;
76     private: System::Windows::Forms::TextBox^ textBox5;
77     private: System::Windows::Forms::Label^ label7;
78
79
80
81
82
83
84
85
90
91
92
93     private: System::Windows::Forms::ComboBox^ comboBox1;
94     private: System::Windows::Forms::Button^ button6;
95
96     private: System::Windows::Forms::DataVisualization::Charting::Chart^ chart1;
97         int i,j;
98
99
100    private: System::Windows::Forms::Timer^ timer2;
101    private: System::Windows::Forms::TextBox^ textBox2;
102    private: System::Windows::Forms::TextBox^ textBox3;
103    private: System::Windows::Forms::TextBox^ textBox4;
104    private: System::Windows::Forms::Label^ label2;
105    private: System::Windows::Forms::Label^ label3;
106    private: System::Windows::Forms::Label^ label4;
107    private: System::Windows::Forms::Label^ label8;
108    private: System::Windows::Forms::Label^ label9;
109    private: System::Windows::Forms::ContextMenuStrip^ contextMenuStrip1;
110
111
112
113
```

```

128
129
130
131
132 #pragma region Windows Form Designer generated code
133 /// <summary>
134 /// Required method for Designer support - do not modify
135 /// the contents of this method with the code editor.
136 /// </summary>
137 void InitializeComponent(void)
138 {
139     this->components = (gcnew System::ComponentModel::Container());
140     System::Windows::Forms::DataVisualization::Charting::ChartArea^ chartArea2 = (gcnew System::Windows::Forms::DataVisualization::Charting::ChartArea());
141     System::Windows::Forms::DataVisualization::Charting::Legend^ legend2 = (gcnew System::Windows::Forms::DataVisualization::Charting::Legend());
142     System::Windows::Forms::DataVisualization::Charting::Series^ series2 = (gcnew System::Windows::Forms::DataVisualization::Charting::Series());
143     this->serialPort1 = (gcnew System::IO::Ports::SerialPort(this->components));
144     this->button2 = (gcnew System::Windows::Forms::Button());
145     this->textBox1 = (gcnew System::Windows::Forms::TextBox());
146     this->button1 = (gcnew System::Windows::Forms::Button());
147     this->label1 = (gcnew System::Windows::Forms::Label());
148     this->timer1 = (gcnew System::Windows::Forms::Timer(this->components));
149     this->label5 = (gcnew System::Windows::Forms::Label());
150     this->groupBox1 = (gcnew System::Windows::Forms::GroupBox());
151     this->label7 = (gcnew System::Windows::Forms::Label());
152     this->textBox5 = (gcnew System::Windows::Forms::TextBox());
153     this->label11 = (gcnew System::Windows::Forms::Label());
154     this->comboBox1 = (gcnew System::Windows::Forms::ComboBox());
155     this->button6 = (gcnew System::Windows::Forms::Button());
156     this->chart1 = (gcnew System::Windows::Forms::DataVisualization::Charting::Chart());
157     this->contextMenuStrip1 = (gcnew System::Windows::Forms::ContextMenuStrip(this->components));
158     this->timer2 = (gcnew System::Windows::Forms::Timer(this->components));
159     this->textBox2 = (gcnew System::Windows::Forms::TextBox());
160     this->textBox3 = (gcnew System::Windows::Forms::TextBox());
161     this->textBox4 = (gcnew System::Windows::Forms::TextBox());
162     this->label12 = (gcnew System::Windows::Forms::Label());
163     this->label13 = (gcnew System::Windows::Forms::Label());
164     this->label14 = (gcnew System::Windows::Forms::Label());
165     this->label18 = (gcnew System::Windows::Forms::Label());
166     this->label19 = (gcnew System::Windows::Forms::Label());
167     this->groupBox1->SuspendLayout();
168     (cli:>safe_cast<System::ComponentModel::ISupportInitialize^>(this->chart1))->BeginInit();
169     this->SuspendLayout();
170     //
171     // serialPort1
172     //
173     this->serialPort1->DataReceived += gcnew System::IO::Ports::SerialDataReceivedEventHandler(this, &Form1::serialPort1_DataReceived_1);
174     //
175     //
176     // button2
177     //
178     this->button2->BackColor = System::Drawing::Color::Green;
179     this->button2->Location = System::Drawing::Point(81, 26);
180     this->button2->Margin = System::Windows::Forms::Padding(4);
181     this->button2->Name = L"button2";
182     this->button2->Size = System::Drawing::Size(109, 36);
183     this->button2->TabIndex = 0;
184     this->button2->Text = L"Run";
185     this->button2->UseVisualStyleBackColor = false;
186     this->button2->Click += gcnew System::EventHandler(this, &Form1::button2_Click_1);
187     //
188     // textBox1
189     //
190     this->textBox1->Location = System::Drawing::Point(8, 39);
191     this->textBox1->Margin = System::Windows::Forms::Padding(4);
192     this->textBox1->Name = L"textBox1";
193     this->textBox1->Size = System::Drawing::Size(113, 22);
194     this->textBox1->TabIndex = 2;
195     this->textBox1->TextChanged += gcnew System::EventHandler(this, &Form1::textBox1_TextChanged);
196     //
197     // button1
198     //
199     this->button1->Location = System::Drawing::Point(442, 1);
200     this->button1->Margin = System::Windows::Forms::Padding(4);
201     this->button1->Name = L"button1";
202     this->button1->Size = System::Drawing::Size(112, 36);
203     this->button1->TabIndex = 4;
204     this->button1->Text = L"Connect";
205     this->button1->UseVisualStyleBackColor = true;
206     this->button1->Click += gcnew System::EventHandler(this, &Form1::button1_Click_1);
207     //
208     // label1
209     //
210     this->label1->AutoSize = true;
211     this->label1->Location = System::Drawing::Point(8, 20);
212     this->label1->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);

```

```

212     this->label1->Name = L"label1";
213     this->label1->Size = System::Drawing::Size(29, 17);
214     this->label1->TabIndex = 6;
215     this->label1->Text = L"Set";
216     //
217     // timer1
218     //
219     this->timer1->Tick += gcnew System::EventHandler(this, &Form1::timer1_Tick_1);
220     //
221     // labels
222     //
223     this->label5->AutoSize = true;
224     this->label5->Location = System::Drawing::Point(554, 36);
225     this->label5->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
226     this->label5->Name = L"label5";
227     this->label5->Size = System::Drawing::Size(0, 17);
228     this->label5->TabIndex = 12;
229     this->label5->Click += gcnew System::EventHandler(this, &Form1::label5_Click);
230     //
231     // groupBox1
232     //
233     this->groupBox1->Controls->Add(this->label7);
234     this->groupBox1->Controls->Add(this->textBox5);
235     this->groupBox1->Controls->Add(this->label1);
236     this->groupBox1->Controls->Add(this->textBox1);
237     this->groupBox1->Location = System::Drawing::Point(214, 5);
238     this->groupBox1->Margin = System::Windows::Forms::Padding(4);
239     this->groupBox1->Name = L"groupBox1";
240     this->groupBox1->Padding = System::Windows::Forms::Padding(4);
241     this->groupBox1->Size = System::Drawing::Size(169, 123);
242     this->groupBox1->TabIndex = 13;
243     this->groupBox1->TabStop = false;
244     this->groupBox1->Text = L"Motor 1";
245     this->groupBox1->Enter += gcnew System::EventHandler(this, &Form1::groupBox1_Enter);
246     //
247     // label7
248     //
249     this->label7->AutoSize = true;
250     this->label7->Location = System::Drawing::Point(8, 71);
251     this->label7->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
252     this->label7->Name = L"label7";
253     this->label7->Size = System::Drawing::Size(49, 17);

254     this->label7->TabIndex = 14;
255     this->label7->Text = L"Speed";
256     this->label7->Click += gcnew System::EventHandler(this, &Form1::label7_Click);
257     //
258     // textBox5
259     //
260     this->textBox5->Enabled = false;
261     this->textBox5->Location = System::Drawing::Point(8, 91);
262     this->textBox5->Margin = System::Windows::Forms::Padding(4);
263     this->textBox5->Name = L"textBox5";
264     this->textBox5->Size = System::Drawing::Size(113, 22);
265     this->textBox5->TabIndex = 13;
266     this->textBox5->TextChanged += gcnew System::EventHandler(this, &Form1::textBox5_TextChanged);
267     //
268     // label6
269     //
270     this->label6->AutoSize = true;
271     this->label6->Location = System::Drawing::Point(735, 180);
272     this->label6->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
273     this->label6->Name = L"label6";
274     this->label6->Size = System::Drawing::Size(32, 17);
275     this->label6->TabIndex = 14;
276     this->label6->Text = L"ppm";
277     this->label6->Click += gcnew System::EventHandler(this, &Form1::label6_Click);
278     //
279     // comboBox1
280     //
281     this->comboBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 9, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,
282         static_cast<System::Byte>(0)));
283     this->comboBox1->FormattingEnabled = true;
284     this->comboBox1->Items->AddRange(gcnew cli::array< System::Object^ >{30} {
285         L"COM1", L"COM2", L"COM3", L"COM4", L"COM5", L"COM6",
286         L"COM7", L"COM8", L"COM9", L"COM10", L"COM11", L"COM12", L"COM13", L"COM14", L"COM15", L"COM16", L"COM17", L"COM18", L"COM19",
287         L"COM20", L"COM21", L"COM22", L"COM23", L"COM24", L"COM25", L"COM26", L"COM27", L"COM28", L"COM29", L"COM30"
288     });
289     this->comboBox1->Location = System::Drawing::Point(442, 88);
290     this->comboBox1->Margin = System::Windows::Forms::Padding(3, 2, 3, 2);
291     this->comboBox1->Name = L"comboBox1";
292     this->comboBox1->Size = System::Drawing::Size(117, 26);
293     this->comboBox1->TabIndex = 3;
294     this->comboBox1->Text = L"chọn cổng";
295     this->comboBox1->SelectedIndexChanged += gcnew System::EventHandler(this, &Form1::comboBox1_SelectedIndexChanged);

```

```

296   // button6
297   //
298   this->button6->Location = System::Drawing::Point(442, 46);
299   this->button6->Margin = System::Windows::Forms::Padding(4);
300   this->button6->Name = L"button6";
301   this->button6->Size = System::Drawing::Size(112, 36);
302   this->button6->TabIndex = 18;
303   this->button6->Text = L"Disconnect";
304   this->button6->UseVisualStyleBackColor = true;
305   this->button6->Click += gcnew System::EventHandler(this, &Form1::button6_Click);
306   //
307   // chart1
308   //
309   chartArea2->Name = L"ChartArea1";
310   this->chart1->ChartAreas->Add(chartArea2);
311   legend2->Name = L"Legend1";
312   this->chart1->Legends->Add(legend2);
313   this->chart1->Location = System::Drawing::Point(8, 139);
314   this->chart1->Margin = System::Windows::Forms::Padding(4);
315   this->chart1->Name = L"chart1";
316   series2->ChartArea = L"ChartArea1";
317   series2->ChartType = System::Windows::Forms::DataVisualization::Charting::SeriesChartType::Line;
318   series2->Color = System::Drawing::Color::Red;
319   series2->IsVisibleInLegend = false;
320   series2->Legend = L"Legend1";
321   series2->MarkerColor = System::Drawing::Color::White;
322   series2->Name = L"Series1";
323   this->chart1->Series->Add(series2);
324   this->chart1->Size = System::Drawing::Size(1296, 509);
325   this->chart1->TabIndex = 11;
326   this->chart1->Text = L"chart1";
327   this->chart1->Click += gcnew System::EventHandler(this, &Form1::chart1_Click);
328   //
329   // contextMenuStrip1
330   //
331   this->contextMenuStrip1->ImageScalingSize = System::Drawing::Size(20, 20);
332   this->contextMenuStrip1->Name = L"contextMenuStrip1";
333   this->contextMenuStrip1->Size = System::Drawing::Size(61, 4);
334   //
335   // timer2
336   //
337   this->timer2->Enabled = true;
338   this->timer2->Interval = 10;
339   this->timer2->Tick += gcnew System::EventHandler(this, &Form1::timer2_Tick);
340   //
341   // textBox2
342   //
343   this->textBox2->Location = System::Drawing::Point(698, 33);
344   this->textBox2->Margin = System::Windows::Forms::Padding(4);
345   this->textBox2->Name = L"textBox2";
346   this->textBox2->Size = System::Drawing::Size(113, 22);
347   this->textBox2->TabIndex = 19;
348   //
349   // textBox3
350   //
351   this->textBox3->Location = System::Drawing::Point(919, 33);
352   this->textBox3->Margin = System::Windows::Forms::Padding(4);
353   this->textBox3->Name = L"textBox3";
354   this->textBox3->Size = System::Drawing::Size(113, 22);
355   this->textBox3->TabIndex = 20;
356   //
357   // textBox4
358   //
359   this->textBox4->Location = System::Drawing::Point(1136, 33);
360   this->textBox4->Margin = System::Windows::Forms::Padding(4);
361   this->textBox4->Name = L"textBox4";
362   this->textBox4->Size = System::Drawing::Size(113, 22);
363   this->textBox4->TabIndex = 21;
364   //
365   // label2
366   //
367   this->label2->AutoSize = true;
368   this->label2->Location = System::Drawing::Point(665, 36);
369   this->label2->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
370   this->label2->Name = L"label2";
371   this->label2->Size = System::Drawing::Size(25, 17);
372   this->label2->TabIndex = 22;
373   this->label2->Text = L"Kp";
374   //
375   // label3
376   //
377   this->label3->AutoSize = true;
378   this->label3->Location = System::Drawing::Point(891, 36);
379

```

```

380     this->label3->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
381     this->label3->Name = L"label3";
382     this->label3->Size = System::Drawing::Size(20, 17);
383     this->label3->TabIndex = 23;
384     this->label3->Text = L"Ki";
385     //
386     // label4
387     //
388     this->label4->AutoSize = true;
389     this->label4->Location = System::Drawing::Point(1103, 36);
390     this->label4->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
391     this->label4->Name = L"label4";
392     this->label4->Size = System::Drawing::Size(25, 17);
393     this->label4->TabIndex = 24;
394     this->label4->Text = L"Kd";
395     //
396     // label8
397     //
398     this->label8->AutoSize = true;
399     this->label8->Location = System::Drawing::Point(1040, 36);
400     this->label8->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
401     this->label8->Name = L"label8";
402     this->label8->Size = System::Drawing::Size(36, 17);
403     this->label8->TabIndex = 25;
404     this->label8->Text = L"/100";
405     this->label8->Click += gcnew System::EventHandler(this, &Form1::label8_Click);
406     //
407     // label9
408     //
409     this->label9->AutoSize = true;
410     this->label9->Location = System::Drawing::Point(1257, 36);
411     this->label9->Margin = System::Windows::Forms::Padding(4, 0, 4, 0);
412     this->label9->Name = L"label9";
413     this->label9->Size = System::Drawing::Size(36, 17);
414     this->label9->TabIndex = 26;
415     this->label9->Text = L"/100";
416     //
417     // Form1
418     //
419     this->AutoScaleDimensions = System::Drawing::SizeF(8, 16);
420     this->AutoSizeMode = System::Windows::Forms::AutoSizeMode::Font;
421     this->ClientSize = System::Drawing::Size(1336, 661);
422     this->Controls->Add(this->label9);
423     this->Controls->Add(this->label8);
424     this->Controls->Add(this->label4);
425     this->Controls->Add(this->label3);
426     this->Controls->Add(this->label2);
427     this->Controls->Add(this->textBox4);
428     this->Controls->Add(this->textBox3);
429     this->Controls->Add(this->textBox2);
430     this->Controls->Add(this->button6);
431     this->Controls->Add(this->comboBox1);
432     this->Controls->Add(this->label15);
433     this->Controls->Add(this->label16);
434     this->Controls->Add(this->groupBox1);
435     this->Controls->Add(this->chart1);
436     this->Controls->Add(this->button1);
437     this->Controls->Add(this->button2);
438     this->Margin = System::Windows::Forms::Padding(4);
439     this->Name = L"Form1";
440     this->Text = L"Form1";
441     this->Load += gcnew System::EventHandler(this, &Form1::Form1_Load_1);
442     this->groupBox1->ResumeLayout(false);
443     this->groupBox1->PerformLayout();
444     (cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this->chart1))->EndInit();
445     this->ResumeLayout(false);
446     this->PerformLayout();
447
448 ...

```

5.4.2 Tạo các tính năng cho giao diện (Back End)

```

449 [ #pragma endregion
450 [ private: System::Void Form1_Load_1(System::Object^ sender, System::EventArgs^ e) {
451 }
452 }
453 ]
454 [ private: System::Void button2_Click_1(System::Object^ sender, System::EventArgs^ e) {
455     serialPort1->Write(textBox1->Text); //start motor
456     serialPort1->Write(textBox2->Text);
457     serialPort1->Write(textBox3->Text);
458     serialPort1->WriteLine(textBox4->Text);
459 }
460 ]
461 [ private: System::Void timer1_Tick_1(System::Object^ sender, System::EventArgs^ e) {
462     String^ length;
463     length=mStr->Length.ToString();
464     if(System::Convert::ToInt32(length)>3){
465         if (mStr->Substring(0, 3) == "vt=") {
466             speed = mStr->Substring(3, System::Convert::ToInt32(length) - 4);
467             textBox5->Text = speed;
468             //print motor speed into Chart
469             this->chart1->Series["Series1"]->Points->AddXY(i, System::Convert::ToDouble(speed));
470             i++;
471             this->chart1->ChartAreas["ChartArea1"]->AxisX->Minimum = i - 400; //shift x-axis
472             this->chart1->ChartAreas["ChartArea1"]->AxisY->Maximum = 300; //adjust y-axis max value
473             this->chart1->ChartAreas["ChartArea1"]->AxisY->Minimum = 0;
474         }
475     }
476 ]
477 [ private: System::Void button1_Click_1(System::Object^ sender, System::EventArgs^ e) {
478     serialPort1->PortName = comboBox1->Text;
479     serialPort1->BaudRate = 115200;
480
481     serialPort1->Open();
482     timer1->Start();
483     mStr = "0";
484     i = 300;
485     //serialPort1->WriteLine("vs_set_speed 1" + textBox1->Text); //send set_speed1 to Arduino
486
487     //serialPort1->WriteLine("vs_set_speed 2" +textBox6->Text); //send set_speed2 to Arduino
488
489     String^ setpoint1 = textBox1->Text;
490 }
491 ]
492 [ private: System::Void serialPort1_DataReceived_1(System::Object^ sender, System::IO::Ports::SerialDataReceivedEventArgs^ e)
493 {
494     mStr=serialPort1->ReadLine();
495 }
496 ]
497 [ private: System::Void button4_Click(System::Object^ sender, System::EventArgs^ e) {
498 }
499 ]
500 [ private: System::Void label11_Click(System::Object^ sender, System::EventArgs^ e) {
501 }
502 ]
503 [ private: System::Void label5_Click(System::Object^ sender, System::EventArgs^ e) {
504 }
505 ]
506 [ private: System::Void groupBox1_Enter(System::Object^ sender, System::EventArgs^ e) {
507 }
508 ]
509 [ private: System::Void textBox5_TextChanged(System::Object^ sender, System::EventArgs^ e) {
510 }
511 ]
512 [ private: System::Void chart1_Click(System::Object^ sender, System::EventArgs^ e) {
513 }
514 ]
515
516
517 [ private: System::Void button6_Click(System::Object^ sender, System::EventArgs^ e) {
518     serialPort1->Close();
519 }
520 ]
521 [ private: System::Void textBox1_TextChanged(System::Object^ sender, System::EventArgs^ e) {
522 }
523 ]
524 [ private: System::Void textBox10_TextChanged(System::Object^ sender, System::EventArgs^ e) {
525 }
526 ]
527 [ private: System::Void label6_Click(System::Object^ sender, System::EventArgs^ e) {
528 }
529 ]
530 [ private: System::Void textBox6_TextChanged(System::Object^ sender, System::EventArgs^ e) {
531 }
532 ]

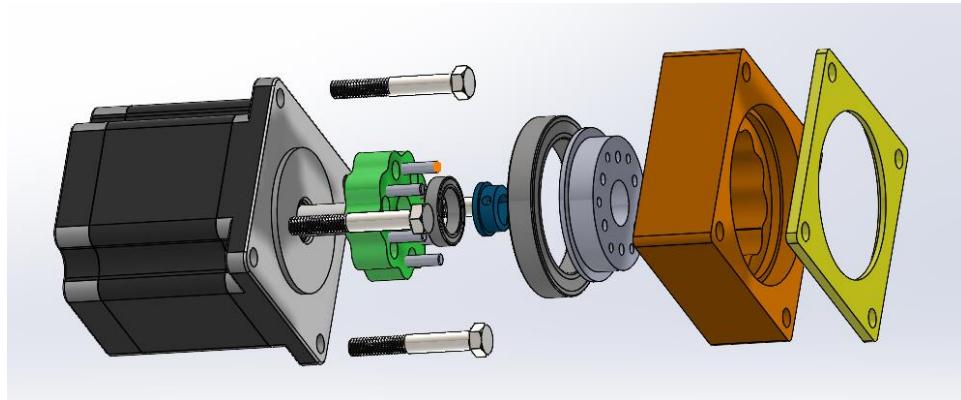
```

```

533     }
534     }
535     }
536     }
537     }
538     }
539     }
540     }
541     }
542     }
543     }
544     }
545     flag = 1;
546     }
547     }
548     flag = 0;
549     }
550     }
551     if (flag == 1)
552     {
553         set = 180 + 90 * sin(0.01 * t);
554         serialPort1->WriteLine(set + "\r");
555         t++;
556     }
557     }
558     }
559     }
560     }
561     };
562     }
563     }
564

```

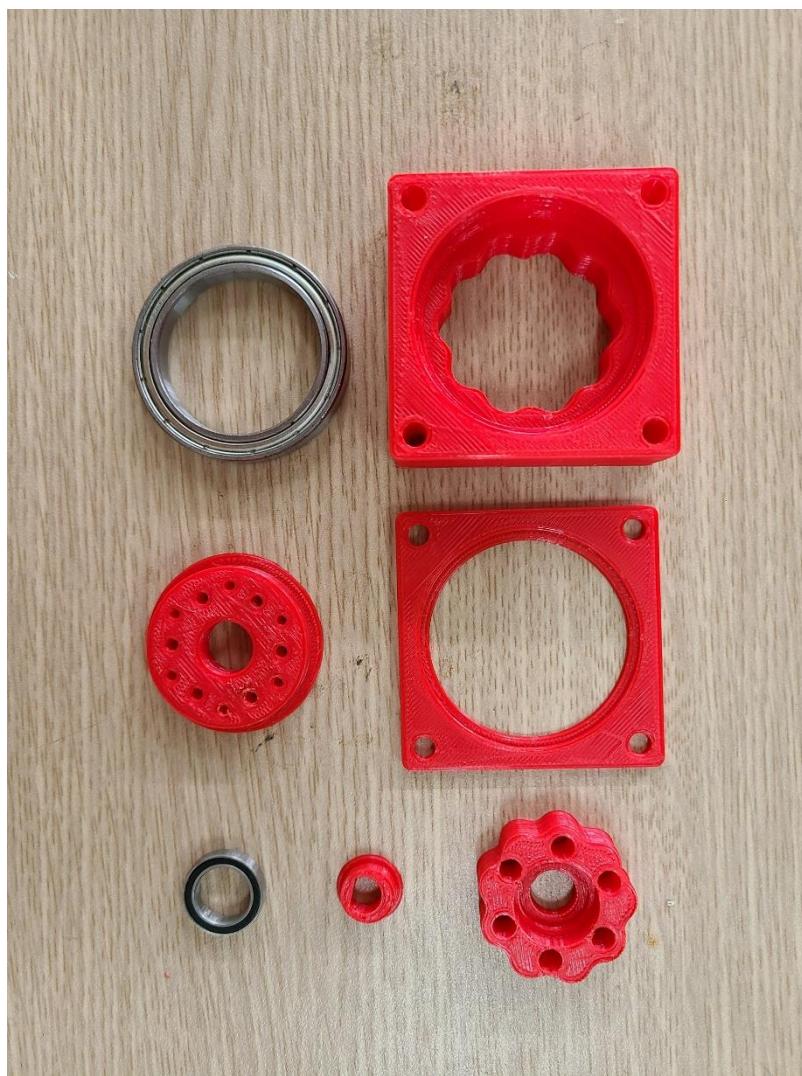
5.5 Kết quả thực tế



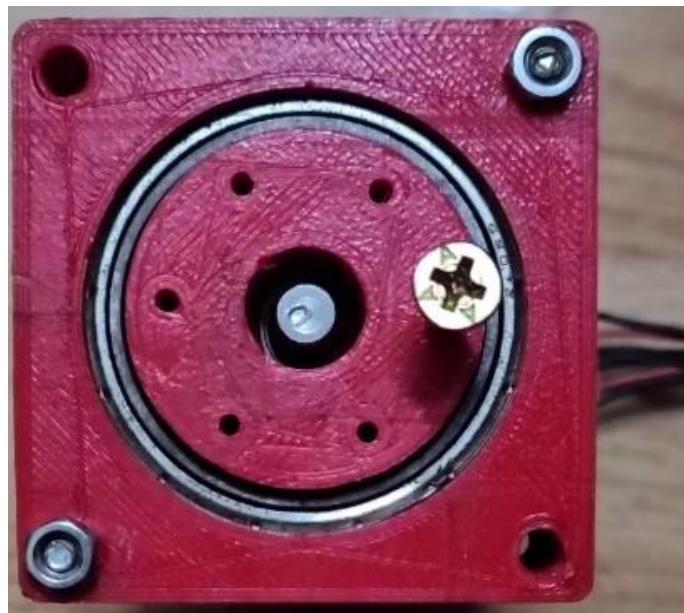
Hình 5.18 Mô phỏng hộp giảm tốc Cycloid trên SolidWorks



Hình 5.19 Gia công 1 số chi tiết bằng nhựa POM



Hình 5.20 Mô hình các thành phần hộp giảm tốc được in bằng máy in 3D



Hình 5.21 Chạy thử nghiệm hộp giảm tốc Cycloid

CHƯƠNG 6. KẾT LUẬN, ĐÁNH GIÁ, HƯỚNG PHÁT TRIỂN

6.1 Kết quả đạt được của đề tài

Hộp giảm tốc Cycloid của nhóm đã đáp ứng được cơ bản các mục tiêu đặt ra ban đầu về tỷ số truyền, hiệu suất, thông số ...

- **Tỷ số truyền (Gear Ratio):** Hộp giảm tốc Cycloid đáp ứng tỷ số truyền đã thiết kế ban đầu để đạt được mục tiêu vận tốc hoặc lực cần thiết cho ứng dụng cụ thể.

- **Hiệu suất:** Hiệu suất của hộp giảm tốc Cycloid cao, giúp giảm thiểu tổn thất năng lượng trong quá trình truyền động và làm tăng hiệu quả của hệ thống.

- **Độ tin cậy:** Hộp giảm tốc Cycloid có độ tin cậy cao để đảm bảo hoạt động ổn định trong thời gian dài và giảm thiểu sự cố.

- **Khả năng tải và tuổi thọ:** Hộp giảm tốc Cycloid có khả năng chịu tải cao và tuổi thọ dài để đáp ứng yêu cầu của ứng dụng.

- **Kích thước và trọng lượng:** Hộp giảm tốc Cycloid của nhóm thiết kế có kích thước và trọng lượng nhỏ hơn so với các hộp giảm tốc truyền thống, đó là một ưu điểm lớn vì giúp tiết kiệm không gian và trọng lượng của hệ thống.

- **Khả năng tùy chỉnh:** Hộp giảm tốc Cycloid có khả năng tùy chỉnh để phù hợp với các yêu cầu cụ thể của ứng dụng, điều này sẽ giúp tối ưu hóa hiệu quả của hệ thống.

6.2 Ý nghĩa khoa học

- Đã áp dụng được những kiến thức sức bền vật liệu, tự động hóa, hệ thống dẫn động cơ khí, dung sai lắp ghép, máy công cụ, ... vào việc tính toán thiết kế và chế tạo trong một đề tài nghiên cứu thực thụ.

- Đã hoàn thiện được thiết kế, chế tạo, lắp ráp một hộp giảm tốc hoàn chỉnh.

- Áp dụng được những kiến thức liên ngành vào việc hoàn thành một đề tài mang giá trị thực tiễn cao.

6.3 Những vấn đề còn tồn tại

- Hộp giảm tốc khi vận hành vẫn còn khá ồn, thi thoảng xảy ra hiện tượng trượt răng.

- Thuật toán chưa được tối ưu.
- Trình độ gia công, lắp ráp còn thấp.
- Bộ thông số PID vẫn chưa đc tối ưu , đôi khi vẫn còn sự vọt lô.

6.4 Hướng phát triển

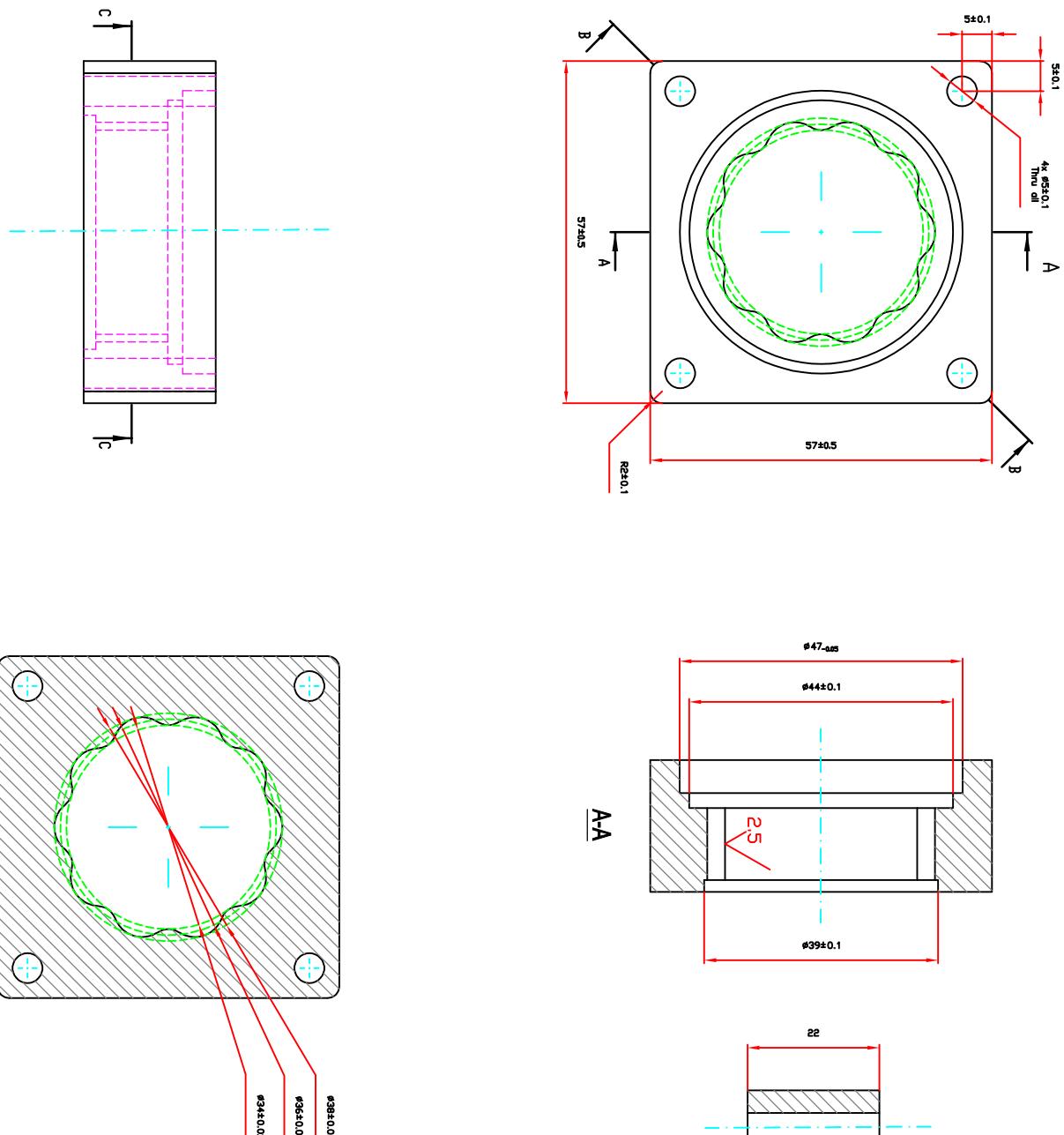
- Với những nghiên cứu hiện tại là nền tảng cũng như tiền đề để xây dựng và phát triển thành công đề tài ở mức độ cao hơn. Hoàn thiện hệ thống một cách tổng quát và hoạt động ổn định hơn trong nhiều môi trường.

- Để tăng tính ổn định cho hộp giảm tốc ta có thể ta có thể sử dụng những máy gia công có độ chính xác cao hơn.

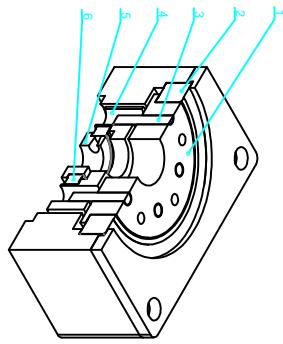
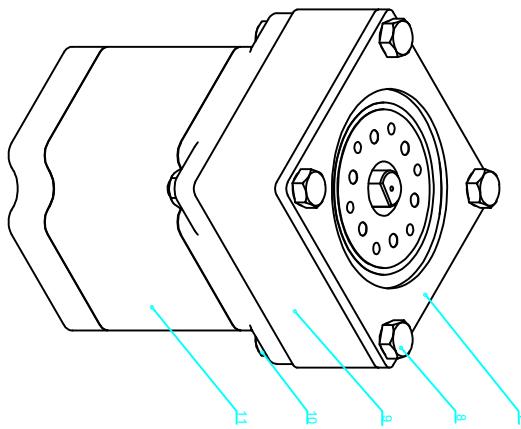
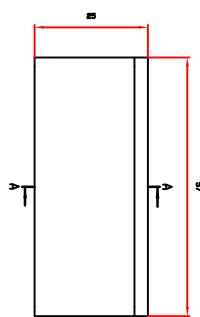
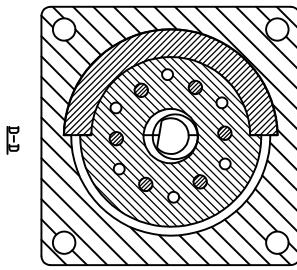
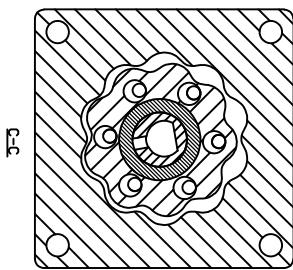
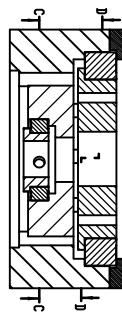
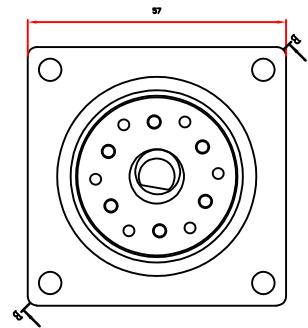
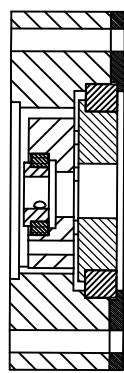
- Áp dụng thêm nhiều phương pháp tính toán khác để tối ưu hóa bộ thông số cho ổn định hơn.

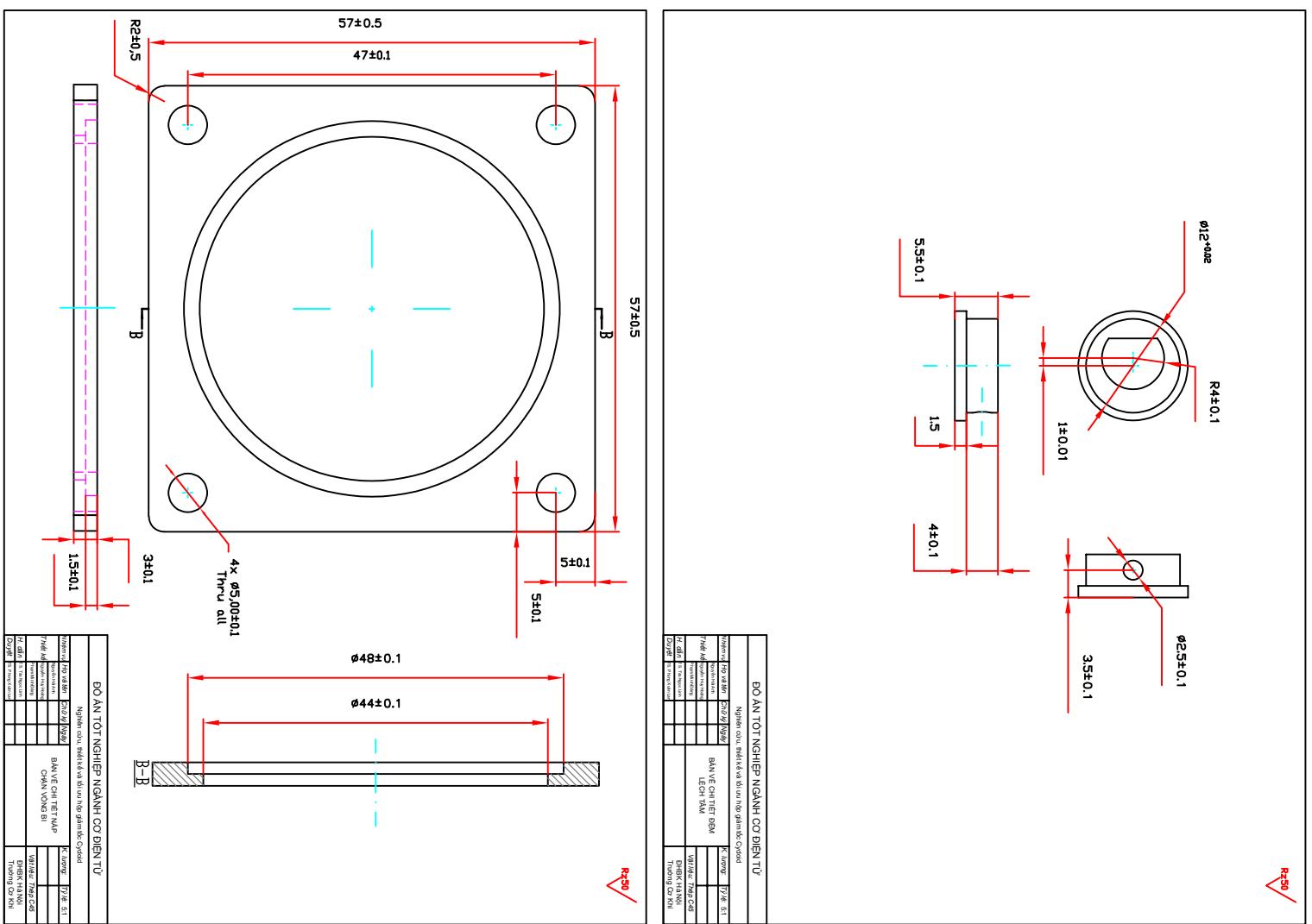
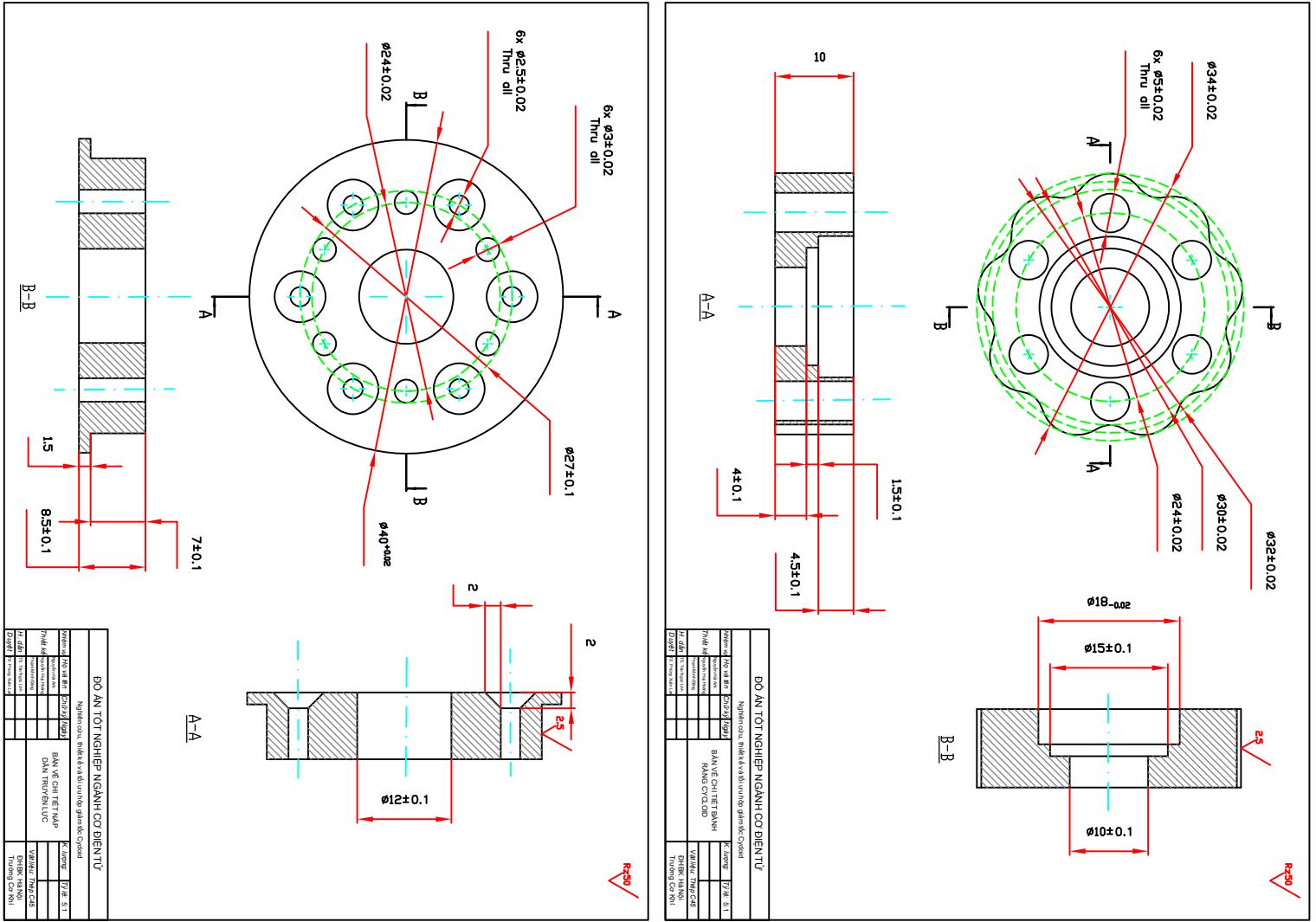
TÀI LIỆU THAM KHẢO

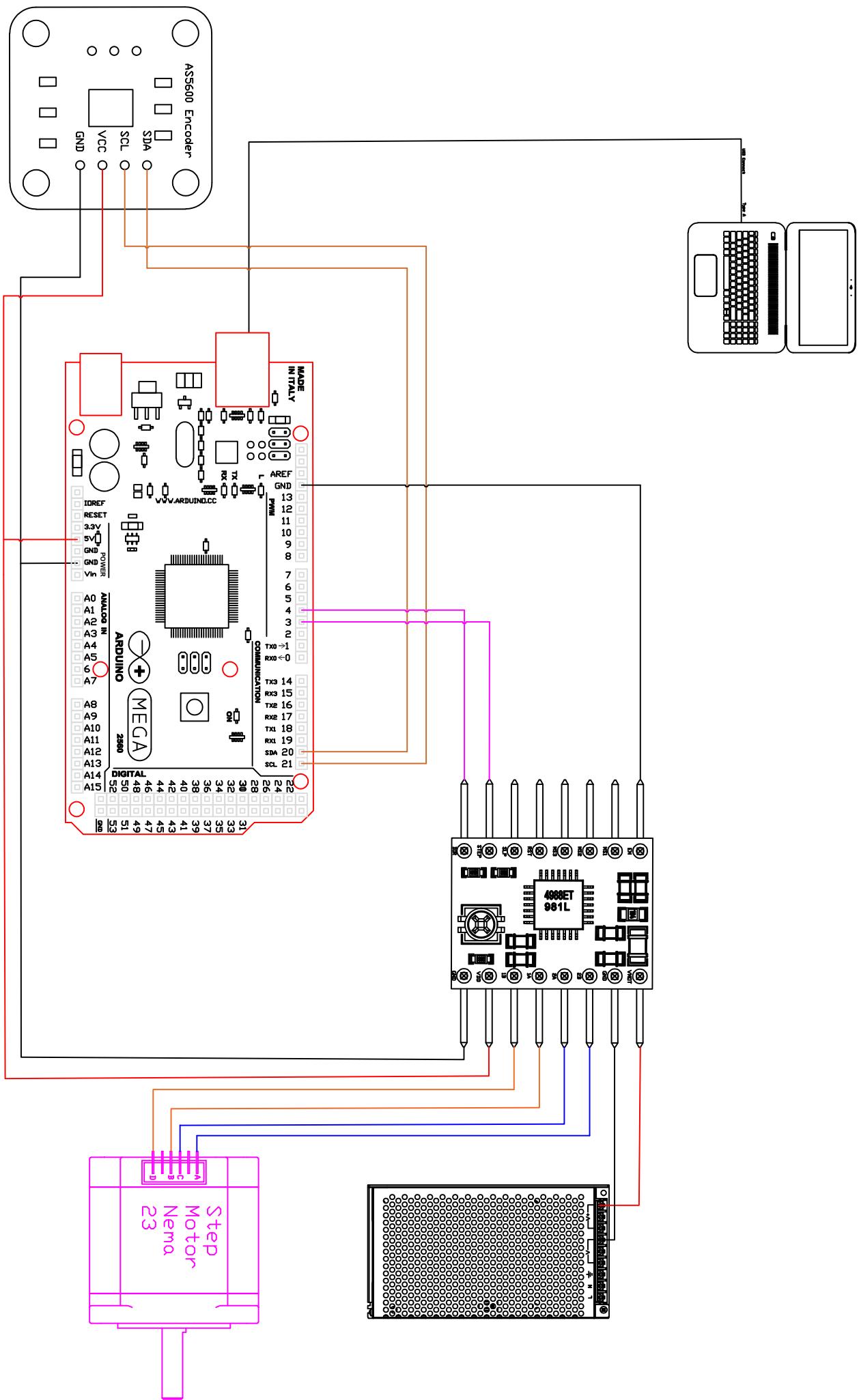
- [1] Cycloid Drive, https://en.wikipedia.org/wiki/Cycloidal_drive.
- [2] Yaliang Wang, Qijing Qian, Guoda Chen, Shousong Jin and Yong Chen. *Multi-objective optimization design of cycloid pin gear planetary reducer.*
- [3] Wang J, Luo SM and Su DY. *Multi-objective optimal design of cycloid speed reducer based on genetic algorithm.* Mech Mach Theory 2016; 102: 135–148.
- [4] Rao ZG. *Designing of mechanism for planetary gearing.* Beijing, China: National Defense Industry Press, 1994 (in Chinese).
- [5] Trịnh Chất (2001), *Cơ sở thiết kế máy và chi tiết máy*, Nhà xuất bản Khoa học và Kỹ thuật, Hà Nội.
- [6] Trịnh Chất, Lê Văn Uyển (2002), *Tính toán thiết kế hệ dẫn động cơ khí – Tập 1,2*, Nhà xuất bản giáo dục, Hà Nội.
- [7] Wang YN, Wu LH and Yuan XF. *Multi-objective selfadaptive differential evolution with elitist archive and crowding entropy-based diversity measure.* Soft Comput 2010; 14: 193–209.
- [8] <https://tapchicokhi.com.vn/>
- [9] Xin Li ,Weidong He, Lixing Li, Linda C. Schmidt. *A New Cycloid Drive With High-Load Capacity and High Efficiency*
- [10] Chen BK, Fang TT, Li CY, et al. *Gear geometry of cycloid drives.* Sci China Ser E 2008; 51: 598–610.
- [11] Thuật toán PSO, https://vi.wikipedia.org/wiki/Tối_ưu_bầy_đàn.
- [12] Lê Trung Kiên, Xây dựng phần mềm tự động điều chỉnh tham số các bộ điều khiển PID, Luận văn Thạc sĩ (2007), file_goc_779005.pdf (tailieuhoctap.vn) .
- [13] TS. Huỳnh Thái Hoàng,Lý thuyết điều khiển tự động, 2016.
- [14] P.I.D - SPEED & POSITION CONTROL, <http://arduino.vn/result/5401-pid-speed-position-control>.



ĐO ÁN TỐT NGHIỆP NGÀNH CƠ ĐIỆN TỬ					
Nghiên cứu, thiết kế và tối ưu hóa giàm tốc Cycloid					
Nghiem ky	Hoa van ten	Chu ky N(gay)	BAN VẼ CHI TIẾT VỎ HỘP	K lượng:	Dy #: 2.5.1
Thiet ke	Nguyen Van Khoa		GIAM TOC		
Thiet ke	Tran Mai Trong			Vat lieu: Thach-C45	
H. oan	T.S. Hoang Van Lam			DRBK-HA-N01	
Duyet	TS Phan Xuan Lai			Truong Co Khi	

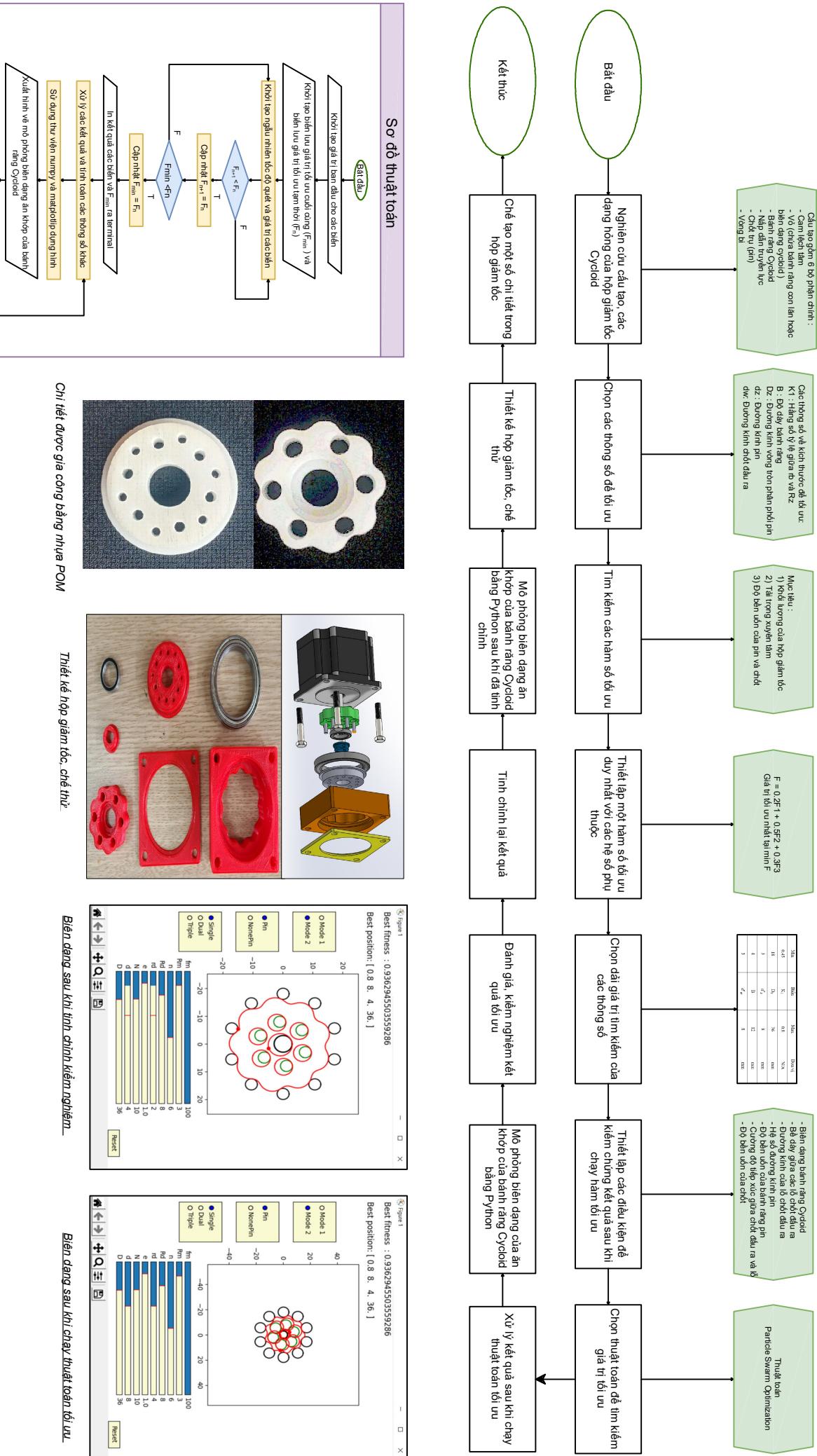




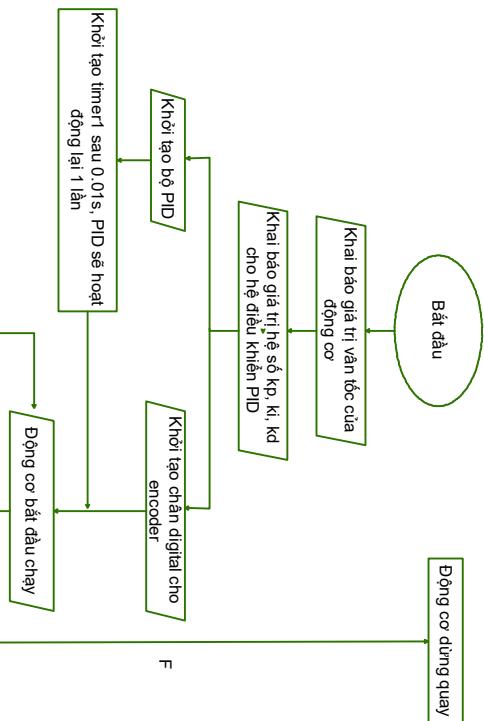


ĐO ÁN TỐT NGHIỆP NGÀNH CƠ ĐIỆN TỬ			
Nhóm	Họ và tên	Chú ý	Nghệ thuật, thi đấu, giải thưởng
Thứ tự tập tin đính kèm	Nguyễn Văn Anh		
Hỗ trợ	Trịnh Minh Phong		
Hỗ trợ	S. Bùi Ngọc Linh		
Duyệt	T. S. Phan Xuân Lai		
			ĐHQG Hà Nội Trường Cơ Kỹ

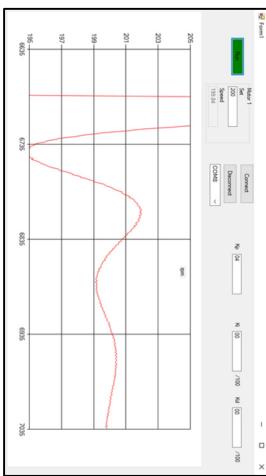
SƠ ĐỒ TỔNG QUAN QUÁ TRÌNH NGHIÊN CỨU, TỐI ƯU VÀ CHẾ TẠO HỘP GIẢM TỐC CYCLOID



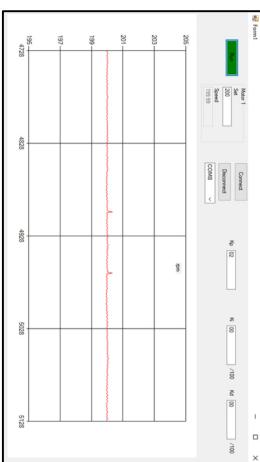
SƠ ĐỒ THUẬT TOÁN ĐIỀU KHIỂN



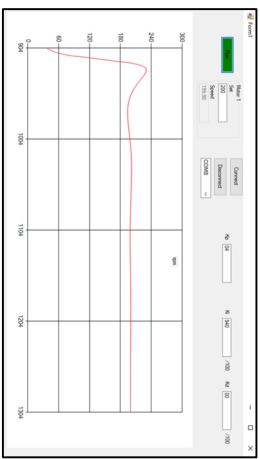
Hệ biên dạng ôn định



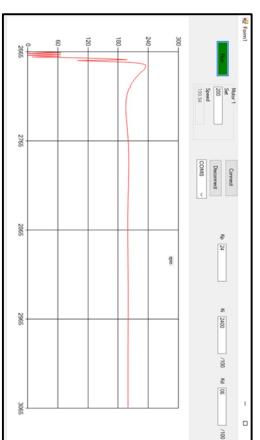
二
四



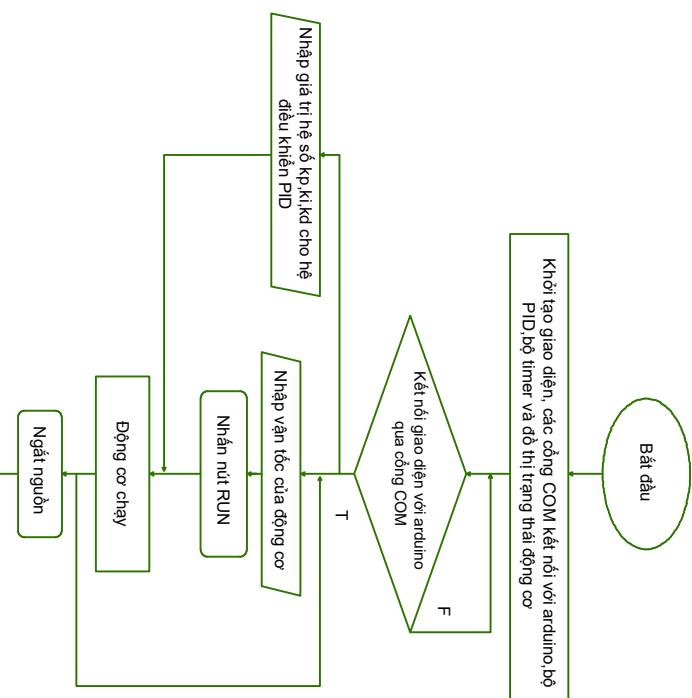
工
業
P



HEPID



SƠ ĐỒ NGUYÊN LÝ HOẠT ĐỘNG



ĐỒ ÁN TỐT NGHIỆP NGÀNH CƠ ĐIỆN TỬ			
Nghiên cứu thiết kế và tối ưu hóa hệ thống giám sát Cybersecurity			
Niệm vụ	Họ và tên	Giới tính	Nơi sinh
Thí điểm	Nguyễn Văn Anh	Đực	Đà Nẵng
Thí điểm	Nguyễn Văn Hùng	Đực	Đà Nẵng
Thí điểm	Nguyễn Thị Kim	Female	Đà Nẵng
Thí điểm	Trần Minh Đức	Đực	Đà Nẵng
Thí điểm	TS. Trần Ngọc Linh	Female	Đà Nẵng
Duyệt	BS Phan Văn Anh	Đực	Đà Nẵng

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP
(Dành cho Giáo viên hướng dẫn)

Tên đề tài: NGHIÊN CỨU, THIẾT KẾ VÀ TỐI ƯU HỘP GIẢM TỐC CYCLOID

Họ và tên SV: Nguyễn Hải Anh

Lớp: CDT 03 – K63

Họ và tên SV: Nguyễn Huy Hoàng

Lớp: CDT 03 – K63

Họ và tên SV: Phạm Minh Đăng

Lớp: ME-NUT17

Chuyên ngành: Hệ thống sản xuất tự động

Giáo viên hướng dẫn: TS. Tào Ngọc Linh

NỘI DUNG NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

I. Tác phong làm việc

.....Sát sao làm việc nghiêm túc, đúng theo yêu cầu của đề án
.....đủ mực.....

II. Những kết quả đạt được

.....Mô hình hóa công thức.....
.....Những tính toán chính xác, đúng và đủ, thuộc về cơ học chất
.....sai số trong lầm trên để áp dụng trong những bài phân tích
.....về.....các vấn đề khác.....
.....
.....
.....

III. Hạn chế của đồ án

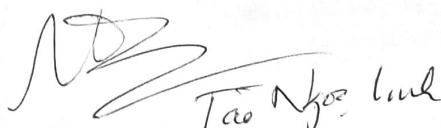
.....Lộn xộn!...số sai số trong quá trình phân tích.....
.....tính toán sai số, sai số, sai số.....
.....
.....
.....

IV. Kết luận

.....Đúng ý cho bài vở.....
.....

Đánh giá: ..10.... điểm

Hà Nội, ngày 16 tháng 8 năm 2023
Giáo viên hướng dẫn
(Ký và ghi rõ họ tên)


Tào Ngọc Linh

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP
(Dành cho Giáo viên phản biện)

Tên đề tài: NGHIÊN CỨU, THIẾT KẾ VÀ TỐI ƯU HỘP GIẢM TỐC
CYCLOID

Họ và tên SV: Nguyễn Hải Anh

Lớp: CDT 03 – K63

Họ và tên SV: Nguyễn Huy Hoàng

Lớp: CDT 03 – K63

Họ và tên SV: Phạm Minh Đăng

Lớp: ME-NUT17

Chuyên ngành: Hệ thống sản xuất tự động

Giáo viên phản biện: TS. Phùng Xuân Lan

NỘI DUNG NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

I. Những kết quả đạt được

Nhóm sinh nên đã thiết kế và chế tạo được mô hình...
hiệp giao thông cycloid.
Chế tạo thành công, ban đầu đã có được thời gian để đánh...
giá. Khi hoạt động quá nhanh, hộp giảm tốc...

II. Hạn chế của đồ án

Các chi tiết được chế tạo từ máy in 3D có ảnh hưởng...
nhất định, bề mặt không đều, không chính xác...
Bên xe có thay đổi nhanh, làm mờ ảnh hưởng đến hiệu...
chức năng lượng...
Bộ sưu tập liệu mà nó chế tạo ra.

III. Kết luận

Đóng góp, hình ảnh và lý do, báo cáo, đề án TNTP...
mô hình thành công nghiệp.

Đánh giá: điểm

Hà Nội, ngày 16 tháng 8 năm 2023
Giáo viên phản biện
(Ký và ghi rõ họ tên)


Phùng Xuân Lan