

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO



ĐỒ ÁN TỐT NGHIỆP

**NGHIÊN CỨU, CHẾ TẠO VÀ PHÁT TRIỂN
ROBOT 6 CHÂN TỰ ĐỘNG DI CHUYỂN
TRONG BẢN ĐỒ TRỰC TIẾP**

GVHD: TS. NGUYỄN VĂN THÁI
SVTH: NGUYỄN HUỲNH ANH TRUNG
MSSV: 15146112
SVTH: LÊ QUỐC CHỈ
MSSV: 15146013
SVTH: VŨ TRỌNG NHÂN
MSSV: 15146081

TP. Hồ Chí Minh, 10 tháng 7 năm 2019

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Giảng viên hướng dẫn: TS Nguyễn Văn Thái

Sinh viên thực hiện: Vũ Trọng Nhân MSSV:15146081

Lê Quốc Chi..... MSSV:15146013

Nguyễn Huỳnh Anh Trung MSSV:15146112

1. Tên đề tài:

Nghiên cứu, chế tạo và phát triển robot 6 chân tự động di chuyển trong bản đồ cho sẵn

2. Các số liệu, tài liệu ban đầu:

Servo 5521MG-180; Board control servo; Arduino mega; Pin 6000 mAh; LIDAR; nhựa PLA

3. Nội dung chính của đồ án:

Thiết kế một robot.....

Tạo app điều khiển trên Android

Tích hợp camera livestream về app

Quét map và điều khiển với LIDAR.....

4. Các sản phẩm dự kiến:

Robot AntPot hoàn chỉnh.....

App điều khiển trên Android có khả năng live stream, bản đồ được quét bởi LIDAR

5. Ngày giao đồ án:18/3/2019

6. Ngày nộp đồ án:11/7/2019

7. Ngôn ngữ trình bày: Bản báo cáo: Tiếng Anh ☐ Tiếng Việt ☒

Trình bày bảo vệ: Tiếng Anh ☐ Tiếng Việt ☒

TRƯỞNG KHOA

TRƯỞNG BỘ MÔN

GIẢNG VIÊN
HƯỚNG DẪN

(Ký, ghi rõ họ tên)

(Ký, ghi rõ họ tên)

(Ký, ghi rõ họ tên)

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Họ và tên Sinh viên:MSSV:

Họ và tên Sinh viên:MSSV:

Họ và tên Sinh viên:MSSV:

Ngành:.....

Tên đề tài:

Họ và tên Giáo viên hướng dẫn:

NHẬN XÉT

1. Về nội dung đề tài & khối lượng thực hiện:

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

2. Ưu điểm:

.....
.....

3. Khuyết điểm:

.....
.....

4. Đề nghị cho bảo vệ hay không?

.....

5. Đánh giá loại:.....

6. Điểm:(Bằng chữ:)

Tp. Hồ Chí Minh, ngày ... tháng ... năm 20...

Giáo viên hướng dẫn

(Ký & ghi rõ họ tên)

PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên:MSSV:

Họ và tên Sinh viên:MSSV:

Họ và tên Sinh viên:MSSV:

Ngành:

Tên đề tài:

Họ và tên Giáo viên phản biện:

NHẬN XÉT

1. Về nội dung đề tài & khối lượng thực hiện:

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

2. Ưu điểm:

.....
.....

3. Khuyết điểm:

.....
.....

4. Đề nghị cho bảo vệ hay không?

.....

5. Đánh giá loại:

6. Điểm:(Bằng chữ:)

Tp. Hồ Chí Minh, ngày ... tháng ... năm 20 ...

Giáo viên hướng dẫn

(Ký & ghi rõ họ tên)

LỜI CẢM ƠN

Đề tài “Nghiên cứu, chế tạo và phát triển robot 6 chân tự động di chuyển trong bản đồ trực tiếp” là nội dung nhóm chọn để nghiên cứu và làm đồ án tốt nghiệp sau bốn năm theo học chương trình đại học chuyên ngành Công nghệ kỹ thuật Cơ điện tử tại trường Đại học Sư phạm kỹ thuật Thành phố Hồ Chí Minh.

Để hoàn thành đề tài, lời cảm ơn đầu tiên chúng em xin được gửi đến giáo sư Kare Halvorsen đã chia sẻ code mẫu và kinh nghiệm thực hiện robot Hexapod, đó là nguồn tài liệu quý giúp đỡ chúng em rất nhiều trong quá trình thực hiện robot Hexapod.

Chúng em xin gửi lời cảm ơn tới TS. Nguyễn Văn Thái, THS. Phạm Bạch Dương đã góp ý và hướng dẫn chúng em trong quá trình hoàn thành đồ án này. Đồng thời xin gửi lời cảm ơn đến tập thể thầy cô cùng nhà trường đã truyền đạt cho chúng em rất nhiều kiến thức bổ ích trong quá trình bốn năm học để chúng em có được hiểu biết như ngày hôm nay.

Chúng em cũng xin cảm ơn anh Huỳnh Văn An - giám đốc công ty Goldeneye Technologies đã tạo giúp đỡ và tạo điều kiện thuận lợi cho chúng em rất nhiều trong suốt quá trình nghiên cứu đồ án.

Cảm ơn anh Trần Sơn Vũ đã đồng ý cho chúng em sử dụng code mẫu và hướng dẫn chúng em sử dụng LIDAR cho việc quét map và điều khiển robot.

Do trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên dự án cũng như bài báo cáo không thể tránh khỏi những thiếu sót, chúng em rất mong nhận được ý kiến đóng góp thầy, cô để chúng em rút kinh nghiệm, đó sẽ là hành trang tốt cho chúng em khi ra trường và đi làm.

Lời cuối cùng, chúng con cảm ơn ba mẹ và gia đình đã luôn nuôi nấng chúng con nên người và luôn là nguồn động viên cho chúng con những lúc khó khăn nhất để chúng con có được thành quả ngày hôm nay.

Nhóm xin chân thành cảm ơn!

TÓM TẮT

Đề tài “Nghiên cứu, chế tạo và phát triển robot 6 chân tự động di chuyển trong bản đồ trực tiếp” xây dựng một con robot Hexapod hoàn chỉnh, hoạt động linh hoạt và ổn định có khả năng điều khiển cả bằng tay và tự động, có khả năng vượt chướng ngại vật và nhận dạng môi trường xung quanh.

Chúng em thực hiện đề tài này nhằm tạo một công cụ bổ ích cho nền giáo dục, một loại robot có thể giúp người dùng, người học có thể có cơ hội tiếp cận với công nghệ robot. Đồng thời qua đó kiến tạo, khơi dậy niềm đam mê công nghệ của các bạn trẻ, ngoài ra còn có thể trau dồi các kiến thức đã học và áp dụng vào quá trình nghiên cứu sản phẩm này.

Nguyên lý hoạt động được dựa trên những phương trình động học thuận, động học nghịch như một cánh tay robot ba bậc tự do và áp dụng vào mỗi chân trong robot, lập trình bằng ngôn ngữ C++, điều khiển bằng Bluetooth.

Phần cứng bao gồm RC servo MG5221MG-180, board Arduino Mega 2560, Raspberry Pi 3, Raspberry Pi Zero, Camera Zero board control servo, mạch giảm áp, pin Li-po 5200mAh và 7000mAh, bộ điều khiển PS2, LIDAR, cảm biến HC-SR04. Mô phỏng trên Matlab và thiết kế trên phần mềm đồ họa Solidworks. Chúng em tiến hành gia công bằng công nghệ in 3D với vật liệu nhựa PLA, CNC lazer, chấn nhôm, CNC lazer mica (PMMA).

Qua nhiều phiên bản, nhóm chúng em đã chế tạo thành công robot Hexapod có khả năng di chuyển linh hoạt, đúng như đã mô phỏng trên Matlab. Robot có khả năng vượt được chướng ngại vật, cho phép tải nhẹ, có thể quan sát môi trường xung quanh bằng camera, quét map bằng LIDAR và tự động di chuyển tới điểm chỉ định.

MỤC LỤC

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP	i
PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN.....	ii
PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN	iii
LỜI CẢM ƠN	iv
TÓM TẮT	v
MỤC LỤC.....	vi
DANH SÁCH CÁC CHỮ VIẾT TẮT	viii
DANH SÁCH CÁC BẢNG BIỂU.....	ix
DANH SÁCH CÁC HÌNH ẢNH, BIỂU ĐỒ.....	x
CHƯƠNG 1. TỔNG QUAN	1
1.1. Đặt vấn đề.....	1
1.2. Khả năng ứng dụng	2
1.3. Tình hình nghiên cứu trong và ngoài nước	2
1.4. Lý do chọn đề tài.....	4
1.5. Mục tiêu và phương pháp nghiên cứu.....	4
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	6
2.1. Giới thiệu chung	6
2.2. Bài toán động học nghịch trong robot.....	6
2.3. Điều khiển thân robot.....	9
2.4. Điều khiển cách di chuyển của Robot.....	10
2.5. Tính ổn định của Hexapod	12
2.6. Giao tiếp Bluetooth với PS2.....	13
2.7. LIDAR. [11].....	17
CHƯƠNG 3. NỘI DUNG NGHIÊN CỨU	30
3.1. Mô phỏng trên Matlab.....	30
3.2. Thiết kế cơ khí.....	31

3.3. Thi công.....	45
3.4. Lưu đồ và giải thuật điều khiển cho di chuyển của Hexabod	48
3.5. Viết app điều khiển bằng Bluetooth kết nối đến HC06	54
3.6. Kết hợp chức năng quét map của LIDAR.....	61
CHƯƠNG 4. THỰC NGHIỆM.....	67
4.1. Kết quả về mặt hoạt động phần cứng.....	67
CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	73
5.1. KẾT LUẬN.....	73
5.2. HƯỚNG PHÁT TRIỂN ĐỀ TÀI.....	73
TÀI LIỆU THAM KHẢO	75

DANH SÁCH CÁC CHỮ VIẾT TẮT

CL: Coxa Length.

CPR: CenterPoint of Rotation.

EEPROM: Electrically Erasable Programmable Read-Only Memory.

FL: Femur Length.

LIDAR: Light Detection And Ranging^[3].

MARS: Multi Appendage Robotic System.

PS2: Play Station 2.

PWM: Pulse Width Modulation.

ROS: The Robot Operating System.

SLAM: Simultaneous Localization and Mapping.

SRAM: Static Random Access Memory.

TL: Tibia Length.

UART: Universal Asynchronous Receiver/Transmitter.

DANH SÁCH CÁC BẢNG BIỂU

Bảng 2-1. Data của các phím nhân PS2	15
Bảng 2-2. Gói dữ liệu các nút trong byte thứ 4.....	15
Bảng 2-3. Gói dữ liệu các nút trong byte thứ 5.....	16
Bảng 2-4. Bảng config sang chế độ gửi tín hiệu analog	16
Bảng 2-5. Bảng config sang chế độ gửi tín hiệu analog từ lực nhấn	16
Bảng 2-6. Tóm tắt thông số Arduino Mega 2560	21
Bảng 2-7. Bảng thông số mạch 32 servo controller	22
Bảng 2-8. Thông số UPEC.....	23
Bảng 2-9. Thông số Pin Li-po.....	24
Bảng 2-10. Thông số và khối lượng RC Servo	26
Bảng 2-11. Thông số PS2.....	27
Bảng 2-12. Thông số HC06	27
Bảng 2-13. Thông số và khối lượng RC Servo	28
Bảng 2-14. Thông số LIDAR.....	29
Bảng 3-1. Kết nối Arduino Mega 2560 với Raspberry Pi 3	42
Bảng 3-2. Kết nối công tắc hành trình vào Arduino Mega2560	42
Bảng 3-3. Kết nối HCRS-04 và FSR402 vào Arduino Mega2560	42
Bảng 3-4. Kết nối PS2 và HC06 vào Arduino Mega2560	43
Bảng 3-5. Kết nối 32 Servo Controller vào Arduino Mega2560	43
Bảng 3-6. Kết nối LIDAR vào Pi 3, Camera OV5647 vào Pi Zero.....	43
Bảng 3-7. Nguồn nuôi Driver 32 Servo, Pi 3, Pi Zero và cách kết nối.....	43
Bảng 3-8. Kết nối các Servo vào Controller	44

DANH SÁCH CÁC HÌNH ẢNH, BIỂU ĐỒ

Hình 1-1. Robot Atlas	2
Hình 1-2. Tripod Robot.....	2
Hình 1-1. Quadruple Robot.....	2
Hình 1-2. Hexpod Robot	2
Hình 1-3. Robot Lego	3
Hình 1-4. Robot Alpha 1E	3
Hình 1-5. Robot Nao	3
Hình 2-1. Chân loài chân khớp trong thực tế.....	6
Hình 2-2. Hình biểu diễn các khâu và khớp trong không gian tọa độ XYZ.....	7
Hình 2-3. Hình biểu diễn góc Coxa khi nhìn dọc theo phương Y từ trên xuống.....	7
Hình 2-4. Hình biểu diễn góc Femur và Tibia khi nhìn dọc theo phương Z	8
Hình 2-5. Các pha trong mỗi bước của Hexabod[10].....	10
Hình 2-6. Hình biểu diễn thứ tự các pha của mỗi chân trong một vòng bước[10]	11
Hình 2-7. Đa giác mà tọa độ trọng tâm nằm trong đó sẽ ổn định[10]	13
Hình 2-8. Chức năng các dây trong module PS2.....	14
Hình 2-9. RPLIDAR A1	17
Hình 2-10. Bản đồ trả về từ LIDAR	17
Hình 2-11. Máy in 3D	18
Hình 2-12. In 3D công nghệ FDM trên phần mềm Cura	19
Hình 2-13. Mô phỏng quá trình in theo lớp	20
Hình 2-14. Board Arduino Mega 2560	21
Hình 2-15. Sơ đồ tính năng của chân trong Board 32 Servo Controller.....	22
Hình 2-16. Mạch giảm áp UPEC 8,3V - 6V	23
Hình 2-17. Pin 7000mAh	24
Hình 2-18. Pin 5200mAh	24
Hình 2-19. Bên trong một RC servo	25
Hình 2-20. Servo 5521MG.....	25
Hình 2-21. PS2	26
Hình 2-22. HC06	26
Hình 2-23. Raspberry Pi 3.....	28
Hình 2-24. LIDAR	29
Hình 3-1. Lưu đồ trong việc mô phỏng hexapod trên Matlab	30
Hình 3-2. Kết quả mô phỏng sự di chuyển của Hexabod trên Matlab.....	30
Hình 3-3 Sơ đồ tổng quan kết nối cơ khí	31

Hình 3-4 Sơ đồ kết nối các module và tín hiệu.....	32
Hình 3-5. Hexapod VS1	33
Hình 3-6. Hexapod VS2 khung nhựa.....	33
Hình 3-7. Hexapod VS3 kết hợp đầu và đuôi	34
Hình 3-8. AntPot (Hexapod VS4).....	35
Hình 3-9. Thiết kế 3D phần đầu Hexapod	36
Hình 3-10. Thiết kế 3D phần thân Hexapod	37
Hình 3-11. Thiết kế 3D phần đuôi Hexapod.....	37
Hình 3-12. Thiết kế 3D phần chân Hexapod	38
Hình 3-13. Servo chuẩn bị lắp ráp cơ khí	40
Hình 3-14. Các bộ phận sau khi in, chuẩn bị lắp ráp	40
Hình 3-15. Bước 1.....	45
Hình 3-16. Bước 2.....	45
Hình 3-17. Bước 3.....	45
Hình 3-18. Bước 4.....	45
Hình 3-19. Bước 1.....	46
Hình 3-20. Bước 2.....	46
Hình 3-21. Bước 3.....	46
Hình 3-22. Bước 4.....	46
Hình 3-23. Bước 1.....	47
Hình 3-24. Bước 2.....	47
Hình 3-25. Bước 1.....	47
Hình 3-26. Bước 2.....	47
Hình 3-27. AntPot	48
Hình 3-28. Hình dáng của Gait trong giải thuật.....	48
Hình 3-29. Các hệ tọa độ trên Hexapod.....	50
Hình 3-30. Lưu đồ giải thuật 1 bước trong Gait	51
Hình 3-31. Lưu đồ trình tự chạy của Gait	52
Hình 3-32. Lưu đồ giải thuật vòng lặp chính	53
Hình 3-33. MIT kết nối bluetooth	55
Hình 3-34. MIT gửi thông tin nút khi nhấn nhả.....	55
Hình 3-35. MIT Joystick hướng theo tay kéo	56
Hình 3-36. MIT thả Joystick	56
Hình 3-37. Hệ tọa độ bên trong một khung Canvas.....	57
Hình 3-38. Code giải thuật giới hạn Joystick.....	58
Hình 3-39. App VS1	59

Hình 3-40. MIT kết nối WebViewer vào một link	59
Hình 3-41. MIT nút Change thay đổi đường link hai màn hình	60
Hình 3-42. App VS2 với màn hình và bố cục được xác định sơ bộ	60
Hình 3-43. App VS3 hoàn thiện.....	61
Hình 3-44 Footprint.....	62
Hình 3-45 Max_vel_x, min_vel_x	62
Hình 3-46 Yaw_goal_tolerance	62
Hình 3-47 Arg	63
Hình 4-1. Dùng USB Tester V3 để đo dòng trong Raspberry Pi.....	67
Hình 4-2. Đo tầm quét hiệu quả	70
Hình 4-3. Tải trọng tối đa mà Hexapod có thể giữ	71

CHƯƠNG 1. TỔNG QUAN

Robot Hexapod là một phương tiện cơ học đi trên sáu chân có tính linh hoạt cao trong việc di chuyển và được lấy cảm hứng từ phân ngành động vật sáu chân. Cùng với sự phát triển mạnh mẽ của các hệ thống Cơ-Điện tử, robot vượt địa hình ngày một được hoàn thiện và càng cho thấy lợi ích của nó trong quân sự, trong nghiên cứu, chúng thường được dùng để vận chuyển hàng hóa trên địa hình không bằng phẳng, can thiệp những khu vực, địa hình nguy hiểm, tìm kiếm cứu nạn, khám phá và lập bản đồ các môi trường chưa biết. Nhóm nghiên cứu đề tài này chủ yếu ứng dụng vào mục đích dân sự, hỗ trợ tìm kiếm cứu nạn, thám dò địa hình mà con người khó tiếp cận, hỗ trợ trong việc nghiên cứu, học tập. Trong đề tài này tập trung nghiên cứu vào robot sáu chân (Hexapod).

1.1. Đặt vấn đề

Trong “Chiến lược phát triển khoa học và công nghệ Việt Nam”, cơ điện tử là một trong những hướng công nghệ trọng điểm phục vụ phát triển kinh tế, xã hội. Và khi nhắc đến cơ điện tử, robot chính là sản phẩm đặc trưng của ngành này. Chúng là những bộ máy hoạt động đồng nhất dựa trên những bộ phận được điều khiển một cách phức tạp thông qua những thuật toán được đem mã hoá vào những vi điều khiển. Có nhiều kiểu robot và chúng em chia chúng thành nhóm robot theo cách thức di chuyển:

- Bằng cánh quạt như robot máy bay- Flycam
- Robot đi bằng bánh xe
- Robot có cánh như côn trùng hay chim
- Robot không chân- di chuyển bằng cách trườn như giun, rắn
- Robot đi bằng chân như động vật

Tuy có thật nhiều loại Robot, nhưng để ứng dụng vào học tập thì những robot di chuyển bốn hay sáu chân vẫn còn nhiều thiếu sót, về bốn chân, gần đây ta có robot Vorbal, mỗi chân hai khớp, với mã nguồn mở, tuy nhiên vẫn chưa đủ phức tạp để có thể thử thách kiến thức về động học do khá đơn giản.

1.2. Khả năng ứng dụng

Vì sự đòi hỏi cao về tri thức trong thiết kế và chế tạo, robot là một công cụ cực tốt để phục vụ trong việc học tập, nghiên cứu, tạo môi trường rộng rãi để áp dụng các kiến thức đã có, góp phần đưa hệ thống giáo dục bắt kịp với tiến độ phát triển công nghệ, đặc biệt là trong kỷ nguyên 4.0 ngày nay.

Ngoài ra, tính ứng dụng của Hexapod trở nên độc đáo bởi chính sự linh hoạt trong hình thức di chuyển, có thể di chuyển trên địa hình đa kết cấu. Hexapod là một trong các phương tiện lớn trong do thám không gian.

1.3. Tình hình nghiên cứu trong và ngoài nước



Hình 1-1. Robot Atlas

<https://www.bostondynamics.com/atlas>

[xem 10/07/2019]



Hình 1-2. Tripod Robot

Evan Ackerman, “Martian-Inspired Tripod Walking Robot Generates Its Own Gaits”,

<https://spectrum.ieee.org>

[xem 10/07/2019]



Hình 1-3. Quadruped Robot

<https://www.bostondynamics.com/atlas>

[xem 10/07/2019]



Hình 1-4. Hexapod Robot

<https://www.trossenrobotics.com/phantomx-ax-hexapod-mk1.aspx>

Các robot di chuyển bằng chân đã được nghiên cứu từ lâu, đều được lấy ý tưởng từ thực tế như dáng đi của con người, kiểu di chuyển của động vật bốn chân, đến kiểu di chuyển của động vật sáu, tám chân và tất cả đều có những thành công nhất định.

Robot Hexapod là một phương tiện cơ học đi trên sáu chân. Vì nó có thể ổn định tĩnh trên ba hoặc nhiều chân, một robot Hexapod có tính linh hoạt cao trong việc di chuyển. Nếu một chân bị vô hiệu hóa, robot vẫn có thể đi bộ. Hơn nữa, không phải tất cả chân của robot đều cần thiết cho sự ổn định, các chân khác được tự do tiếp cận các vị trí chân mới hoặc điều khiển tải trọng. Nhiều Hexapod robot được lấy cảm hứng từ phân ngành động vật sáu chân. Hiện nay trên thế giới đã có nhiều nhóm nghiên cứu và phát triển. Ở Việt Nam, robot di chuyển bằng chân cũng là đề tài được nhiều nhóm sinh viên thực hiện, là đề tài thích hợp phục vụ học tập.

Ở Việt Nam, những Robot phục vụ học tập đã có mặt trong các trường học:



Hình 1-5. Robot Lego



Hình 1-6. Robot Alpha 1E

<https://ubtrobot.com/pages/alpha>

[xem 10/07/2019]

Robot Lego tại lớp học Mindstorm nâng cao của Câu lạc bộ Robotics (tạm dịch Ngành học về robot) - IoT của trường ĐH Khoa học tự nhiên TP.HCM, hay Robot Alpha 1E trong chương trình Trại hè Công nghệ 2019 tại Học viện Sáng tạo Công nghệ TEKY. Robot còn có mặt trong các Lab của các trường đại học như Robot Nao của trường ĐH Khoa Học Tự Nhiên.



Hình 1-7. Robot Nao

Khi gõ từ khóa “Hexapod ở Việt Nam” hoặc “robot 6 chân ở Việt Nam” trên trang tìm kiếm Google, có rất ít kết quả liên quan đến đề tài này, đề tài Hexapod ở Việt Nam, chủ yếu được các bạn sinh viên nghiên cứu cho việc làm các dự án nhỏ, đồ án môn học, đồ án tốt nghiệp hay những ngày hội khoa học sáng tạo như: Robot dò tìm bom mìn của nhóm sinh viên Trường Đại học (ĐH) Bách khoa Đà Nẵng, gồm: Ngô Diên Bảo Triết, Lê Tự Duy Hoàng và Trần Văn Chính. Có vài kết quả về robot thương mại đơn giản phục vụ cho học tập nhưng là những mô hình đơn giản, hai DOF hoặc ba DOF lắp ghép bằng mica. Cũng có những cá nhân nghiên cứu, tìm hiểu về hexapod và đăng lên các diễn đàn hoặc đưa clip hoạt động lên Youtube.

1.4. Lý do chọn đề tài.

Mảng robot di chuyển bằng chân là niềm đam mê chung của các thành viên trong nhóm. Là một dự án rất phù hợp với ngành cơ điện tử, sinh viên được áp dụng rất tốt các kiến thức chuyên ngành đã học được trên trường, đồng thời cũng cũng khá ít các dự án tương tự đã được thực hiện ở Việt Nam cho nên có rất ít tài liệu liên quan khiến dự án này vừa là niềm đam mê, vừa là thách thức mà chúng em muốn vượt qua.

Hiện nay, nhu cầu học tập và tìm hiểu công nghệ của nước ta rất cao, rất nhiều lớp học về robot đã được mở ra để đáp ứng được nhu cầu này và phục vụ cho nhu cầu đó thì robot là một công cụ không thể thiếu. Nhóm chúng em nghiên cứu và chế tạo ra robot 6 chân này phục vụ cho nhu cầu học tập đó của các em, giúp các em có sự hứng thú và có nhiều sự lựa chọn hơn cho quá trình học tập, nghiên cứu robot của mình.

1.5. Mục tiêu và phương pháp nghiên cứu.

Với dự án này chúng em nghiên cứu, mô phỏng trên Mathlab và tạo ra một robot Hexapod hoàn chỉnh có khả năng di chuyển, mô phỏng cách di chuyển của loài côn trùng chân khớp. Sử dụng các phương trình động học, truyền động để, thiết kế được bộ khung và chọn được động cơ phù hợp, ứng dụng công nghệ in 3D với vật liệu nhựa PLA trong việc chế tạo robot. Lập trình theo các giải thuật điều khiển đã tìm được. Điều khiển robot từ xa bằng các module điều khiển. Robot có thể quét được không gian xung quanh, xác định vị trí trong không gian và vẽ nên bản đồ gửi lên Web, người dùng có thể giao tiếp trực tiếp trên chính bản đồ gửi về, trực tiếp chọn trên màn hình để robot tự động đi tới vị trí được chuyển, dựa vào tín hiệu digital từ công tắc hành trình dưới mỗi chân để xác định điểm đặt chân, hỗ trợ việc di chuyển trên địa hình đa kết cấu.

Phương pháp nghiên cứu là tìm kiếm tài liệu trên các trang mạng trên Internet, nghiên cứu những thiết kế đã được các nhóm, các cá nhân phát triển trong và ngoài nước từ đó thiết kế ra một con robot cử động linh hoạt. Tập trung phân tích, tính toán,

chọn lựa và thực nghiệm các module và linh kiện để tìm thấy. Nghiên cứu và phát triển thuật toán trong code.

Nhóm đã thực hiện đề tài này trong hơn 10 tháng gồm bốn giai đoạn chính:

Giai đoạn 1:

- Tìm kiếm tài liệu

Giai đoạn 2:

- Nghiên cứu, lựa chọn và kiểm nghiệm các module, linh kiện phù hợp, xây dựng code điều khiển, lắp ráp một mô hình đơn giản. Mô phỏng trên Matlab
- Thiết kế phần khung xương cho robot đảm bảo các chức năng di chuyển cơ bản
- Dựa trên các thuật toán điều khiển, động học, và code mẫu, điều khiển từng khớp, từng chân và kết hợp các chân

Giai đoạn 3:

- Đánh giá khả năng hoạt động, độ bền, của thiết kế cũ, thiết kế lại khung của robot bằng vật liệu nhựa
- Tính toán, thiết kế khung bằng nhựa PLA, mua và gia công các chi tiết, lắp ráp thành một con robot hoàn chỉnh
- Hiệu chỉnh code

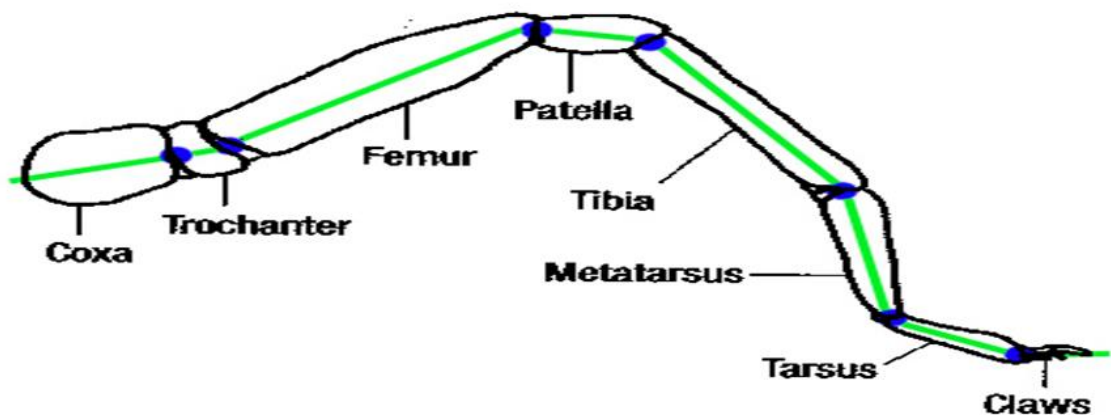
Giai đoạn 4:

- Thiết kế lại toàn bộ phần khung, vỏ robot, đảm bảo sự linh hoạt cho robot, giảm khối lượng, đảm bảo tính thẩm mỹ.
- Tính toán, chọn lại các module, nguồn phù hợp
- Hiệu chỉnh code, cải thiện khả năng di chuyển linh hoạt và giống với tự nhiên hơn
- Thiết kế app điều khiển
- Tích hợp module LIDAR, camera

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Giới thiệu chung

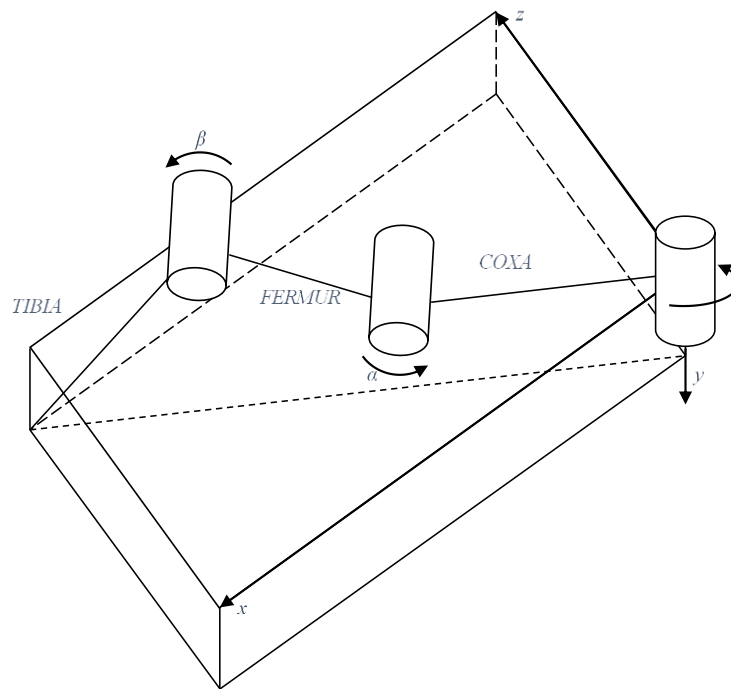
Để Hexapod có thể đi được, một số thuật toán cần phải làm việc cùng nhau để tạo thành bộ điều khiển hoàn chỉnh. Kết quả cuối cùng ở mọi khoảng thời gian là vị trí set-point cho mỗi servo. Mô hình bước cần phải được chọn, các quỹ đạo đã được tính toán và các ràng buộc vị trí các chân được cập nhật liên tục. Tùy thuộc vào vận tốc, các kiểu dáng khác nhau được chọn bởi một bộ điều khiển. Để thực thi mỗi kiểu dáng sẽ có một giai đoạn đứng và một giai đoạn xoay chân. Trong giai đoạn đứng là khi chân tiếp xúc mặt đất ở mọi thời điểm. Trong giai đoạn xoay chân quỹ đạo giữa hai vị trí đứng phải được tính toán đúng bởi bộ điều khiển. Do kích thước phần cứng như chiều dài chân, vị trí servo và chiều rộng cơ thể, một số ràng buộc nhất định sẽ hạn chế vị trí các chân. Các vị trí của mỗi chân cũng sẽ ảnh hưởng đến vị trí các chân còn lại trong không gian. Do sự giống nhau giữa một robot Hexapod và côn trùng chân khớp, rất nhiều cảm hứng có thể được lấy từ nó và sinh trắc học của chúng.



Hình 2-1. Chân loài chân khớp trong thực tế

2.2. Bài toán động học nghịch trong robot

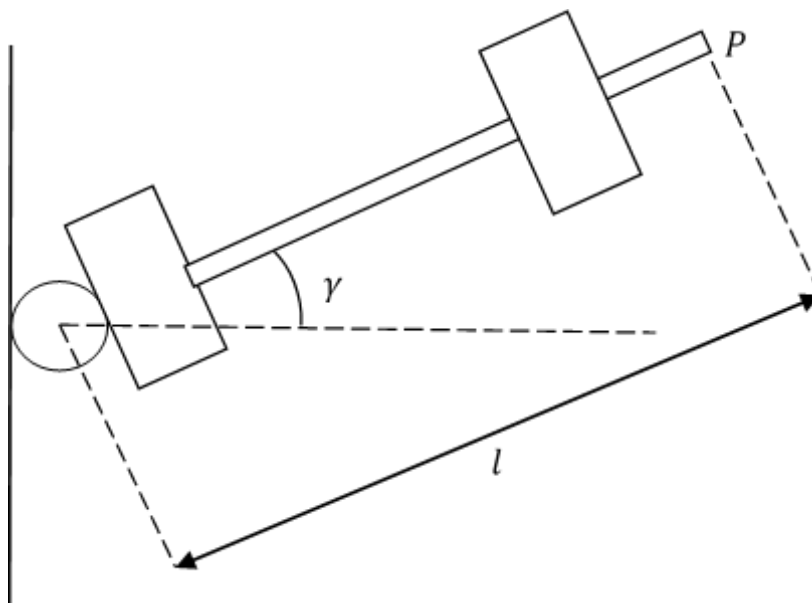
Động học nghịch là sử dụng các phương trình động học để xác định các tham số góc của mỗi khớp để có được vị trí mong muốn cho mỗi bộ phận của robot ^[4]. Tức là từ tọa độ P xác định trong không gian, với P là vị trí cuối cùng tại mỗi mũi chân của Hexapod, từ đó tính ra được các góc Coxa Femur và Tibia để điều khiển Servo, rồi điều khiển cả một hệ thống. Các thông số cần tính được diễn tả như cấu trúc bên dưới, bao gồm: ba khâu, ba khớp.



Hình 2-2. Hình biểu diễn các khâu và khớp trong không gian tọa độ XYZ.

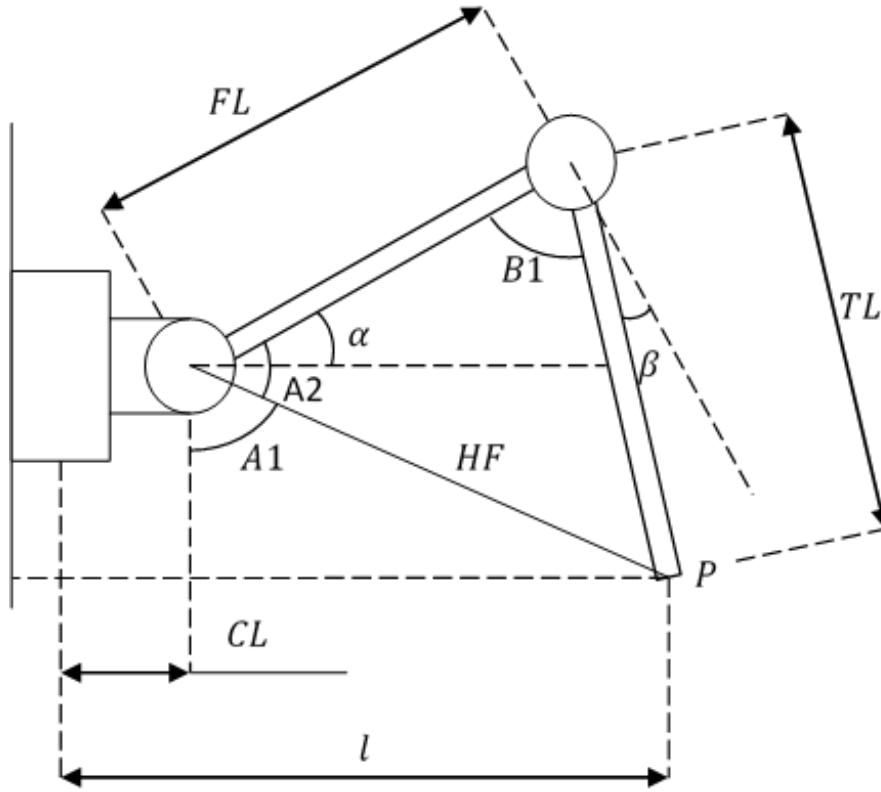
Các biến γ, α, β lần lượt là các Coxa, Femur và Tibia, là các góc hiện tại của mỗi Servo, mục tiêu chúng em hướng đến là xác định giá trị của các góc xoay Offset (tức là góc mà mỗi servo cần phải xoay thêm để đạt được góc xoay mong muốn) và code.

- Góc Coxa



Hình 2-3. Hình biểu diễn góc Coxa khi nhìn dọc theo phương Y từ trên xuống.

- Góc Femur và Tibia



Hình 2-4. Hình biểu diễn góc Femur và Tibia khi nhìn dọc theo phương Z

Gọi tọa độ của P là (x, y, z) trong không gian, gọi tắt Coxa Length, Femur Length và Tibia Length là CL, FL, TL .

Dựa vào kiến thức toán hình học, l có thể được tính bằng công thức sau:

$$l = \sqrt{x^2 + z^2} \quad (2-1)$$

$$HF = \sqrt{(l - CL)^2 + y^2} \quad (2-2)$$

$$A1 = \arctan2(y, l - CL) \quad (2-3)$$

$$A2 = \cos^{-1} \left(\frac{FL^2 + HF^2 - TL^2}{2 \cdot FL \cdot HF} \right) \quad (2-4)$$

$$B1 = \cos^{-1} \left(\frac{FL^2 + TL^2 - HF^2}{2 \cdot FL \cdot TL} \right) \quad (2-5)$$

Kết quả đạt được:

$$\alpha = 90^\circ - (A + A2) \quad (2-6)$$

$$\beta = 90^\circ - B1 \quad (2-7)$$

Do ban đầu, chúng em set góc sẵn có trong Servo là 90. Gọi γ', α', β' là các góc nhóm muốn hướng đến, công thức liên hệ giữa chúng và các góc Offset được thể hiện như sau:

$$\alpha' = \alpha + \alpha_{offset} \quad (2-8)$$

$$\beta' = \beta - \beta_{offset} \quad (2-9)$$

Vì xu hướng quay của hai góc α và β luôn ngược chiều, xuất hiện sự trái dấu trong phép tính. Đặc biệt đối với góc Coxa, do ở mỗi chân đều nằm ở phần góc phân tư trong hệ tọa độ khác nhau, từ đó có thể tìm được sự khác nhau về kết quả đối với mỗi chân với điều kiện sau:

$$\gamma = \arctan2(x, z) \quad (2-10)$$

Dựa vào các kết quả trên, thông số điều khiển Servo có thể được tính thông qua bộ chuyển đổi sang giá trị xung.

2.3. Điều khiển thân robot

Khi thân xoay hay tịnh tiến, do thân chính là gốc tọa độ các chân, trong khi các chân còn lại đứng yên. Vậy nên vị trí của các chân so với thân có sự thay đổi, tọa độ đó có thể tính bằng cách áp dụng phép tính ma trận xoay.

Gọi điểm P có tọa độ (x, y, z) được xác định trong không gian, lấy tâm thân làm gốc tọa độ. Các ma trận biểu diễn các phép quay quanh trục x, y, z một góc θ lần lượt là $R(x, \alpha), R(y, \beta), R(z, \gamma)$:

Ma trận quay quanh trục x^[9]:

$$R(x, \alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \quad (2-11)$$

Ma trận quay quanh trục y^[9]:

$$R(y, \beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \quad (2-12)$$

Ma trận quay quanh trục z ^[9]:

$$R(z, \gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2-13)$$

Chúng em tiến hành nhân các ma trận để có thể có ma trận tổng quát khi thân quay một góc bất kì trong không gian, với các góc α , β và γ là các góc được tạo bởi hình chiếu đường thẳng từ gốc tọa độ đến P lên mặt phẳng Oxy, Oyz, Oxz và các trục Ox, Oy, Oz, tính được góc tọa độ P' mới (x', y', z')

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} c\gamma.c\beta & -s\gamma.c\alpha + c\gamma.s\beta.s\alpha & s\gamma.s\alpha + c\gamma.s\beta.c\alpha \\ s\gamma.c\beta & c\gamma.c\alpha + s\gamma.s\beta.s\alpha & -c\gamma.s\alpha + s\gamma.s\beta.c\alpha \\ -s\beta & c\beta.s\alpha & c\beta.c\gamma \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (2-14)$$

Khi thân tịnh tiến, các tọa độ của chân đồng thời di dời 1 khoảng tương ứng **ngược lại** với hướng tịnh tiến của thân. Vậy có thể đơn giản tính P' (x', y', z') khi thân tịnh tiến một khoảng theo các hướng x, y, z tuần tự là $x1, y1, z1$

$$x' = x - x1 \quad (2-15)$$

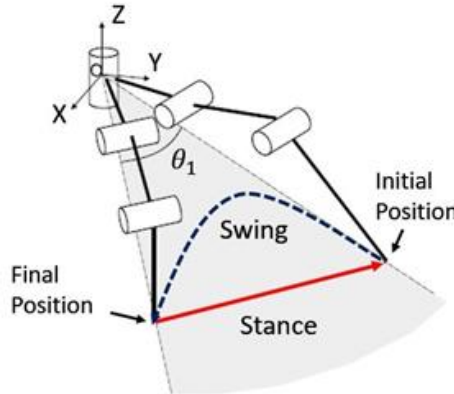
$$y' = y - y1 \quad (2-16)$$

$$z' = z - z1 \quad (2-17)$$

2.4. Điều khiển cách di chuyển của Robot

2.4.1. Phương thức di chuyển

Robot sẽ di chuyển bằng cách điều khiển từng cánh tay ba khớp (chân robot) theo một thứ tự mong muốn, thứ tự bước khác nhau tạo thành các kiểu dáng khác nhau, tăng thêm tính đa dạng. Tuy nhiên, dù có thứ tự khác nhau ra sao, các tọa độ mỗi chân luôn đi theo một quỹ đạo nhất định, quỹ đạo này gọi là Gait^[1]. Gait có hai pha cho hai trường hợp khi nâng chân và chân chạm đất.



Hình 2-5. Các pha trong mỗi bước của Hexabod^[10]

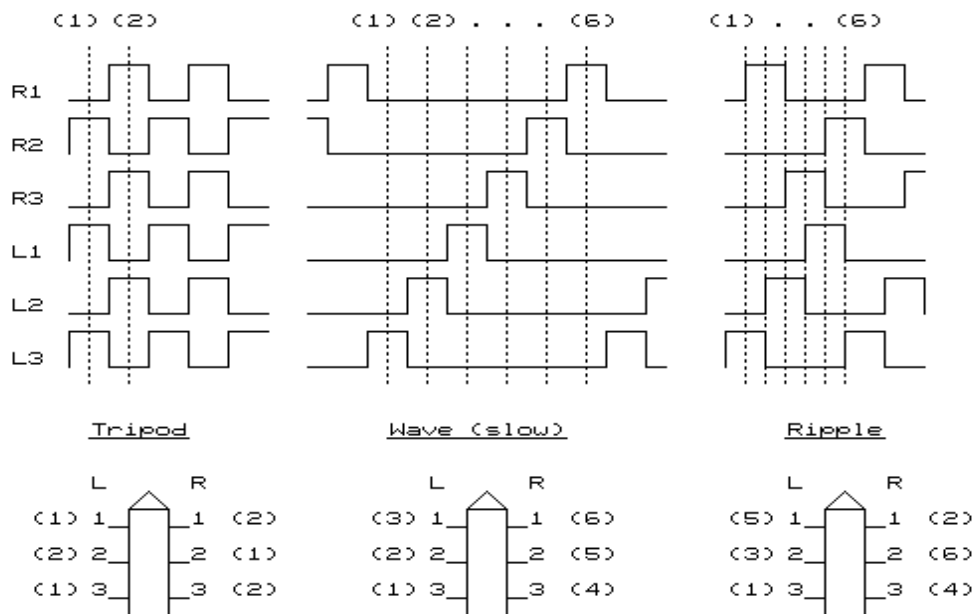
Trong giai đoạn Swing (nâng chân và quạt tới), chân di chuyển từ vị trí ban đầu đến vị trí cuối cùng trong không khí, được biểu thị bằng đường nét đứt. Mặt khác, trong giai đoạn Stance (chân chạm đất), bộ phận mũi chân tiếp xúc với mặt đất trong khi chân di chuyển từ vị trí ban đầu, di chuyển robot theo hướng ngược lại với mũi tên.

2.4.2. Các kiểu di chuyển

Phụ thuộc vào yêu cầu về tốc độ, tính ổn định, tiết kiệm năng lượng hay yêu cầu về địa hình thì ta có những sự lựa chọn khác nhau.

- Di chuyển liên tục: là kiểu di chuyển mà thân đồng thời tịnh tiến cùng với các chân

Có ba kiểu di chuyển phổ biến:



Hình 2-6.

Hình biểu diễn thứ tự các pha của mỗi chân trong một vòng bước^[10]

Với kiểu đi Tripod, sáu chân của robot được chia làm hai bộ (1), (2) thay phiên nhau bước.

Với kiểu wave, chỉ có một chân ở trong pha Swing, còn lại ở trong pha Stance. Rất chậm nhưng lại đỡ tốn năng lượng, hay dùng trong dò địa hình gồ ghề.

Với kiểu ripple, hai chân trong pha Swing, còn lại trong pha Stance, trung hòa hai cách trên.

Một trong số ba kiểu dáng di chuyển của ngành chân khớp, trong báo cáo này chúng em không đề cập đến kiểu Wave(slow) (lan truyền từng chân) và Ripple (hai chân chéo). Bởi vì để Robot di chuyển nhanh, mềm mại, tiết kiệm thời gian di chuyển, tạo được một mặt phẳng tiếp xúc ba điểm cân bằng thì kiểu dáng Tripod chiếm ưu thế nổi trội nhất.

- Kiểu di chuyển không liên tục: là kiểu mà sau khi tất cả các chân đã thực hiện hết các vòng bước thì thân mới tiến lên, đây là cách di chuyển thường thấy khi đi trên các địa hình dốc, thân robot chỉ tịnh tiến người về trước khi có đủ sáu chân chạm đất, khi độ cứng vững và tính bám là cao nhất. Đây là mục tiêu mà nhóm muốn hướng tới trong tương lai nhằm phục vụ ứng dụng vượt địa hình.

2.4.3. Điều khiển cho Hexapod quẹo phải trái

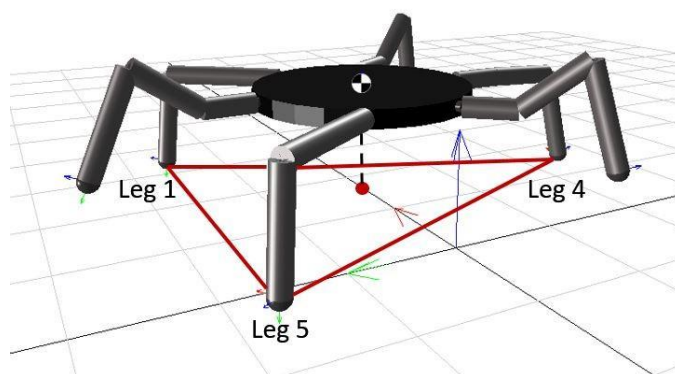
Để Hexapod có thể xoay, dáng đi đã sử dụng phải được sửa lại. Có một số phương pháp để điều khiển Hexapod xoay khá hữu hiệu. Phương pháp đầu tiên là thay đổi chiều dài mỗi bước ở hai bên, làm cho một bên chân di chuyển chậm hơn (bước đi ngắn hơn) sẽ khiến cho Hexapod xoay dần về phía đấy. Một phương pháp khác là giảm tần số xoay ở một bên thân để mất bớt một bước. Đối với việc điều khiển Hexapod cua gấp hay xoay quanh một điểm, ta thường kết hợp cả hai phương pháp. Ngoài ra ta cho chân bước lùi sẽ làm việc điều khiển đó được dễ dàng hơn. Một cách khác để Hexapod xoay tương tự như việc giảm chiều dài bước là xoay chân xung quanh trung tâm cơ thể. Xoay chân trên đất xung quanh trung tâm cơ thể chính sẽ khiến cho cơ thể có dáng vẻ như đang xoay. Để việc xoay được thực hiện, việc quan trọng phải đảm bảo vận tốc góc quay ở mỗi chân là bằng nhau và phải quay xung quanh cùng một điểm (trung tâm cơ thể). Khi một chân vượt quá xa khỏi vị trí, ta có thể đem trở về bằng giai đoạn xoay chân. Ta sử dụng phương tiện quay là hệ ma trận quay $R^{[1]}$.

$$R_z(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (2-18)$$

2.5. Tính ổn định của Hexapod

Độ ổn định của Hexapod được chia thành hai loại: ổn định tĩnh và ổn định động. Để được coi là ổn định tĩnh, Hexapod cần ổn định trong toàn bộ chu kỳ di chuyển, không cần thêm bất kỳ lực nào để cân bằng robot. Trong khi robot ổn định tĩnh, hình chiếu thẳng đứng tại toạ độ trọng tâm (COM) của nó nằm trong đa giác được hình thành từ các chân đang trong giai đoạn đẩy tiến. Trong trường hợp COM được đặt ở biên hoặc bên ngoài đa giác, robot sẽ ngã xuống trừ khi nó ổn định về mặt động lực,

tức là robot được cân bằng trong khi đi bộ do lực quán tính gây ra bởi chuyển động và không ổn định tĩnh khi dừng di chuyển.



Hình 2-7. Đa giác mà tọa độ trọng tâm nằm trong đó sẽ ổn định^[10]

2.6. Giao tiếp Bluetooth với PS2

2.6.1. Giới thiệu về chuẩn giao tiếp SPI

SPI (Serial Peripheral Bus) là một chuẩn truyền thông nối tiếp đồng bộ tốc độ cao (lên đến 10Mbps) do hãng Motorola phát triển. Đây là kiểu truyền thông Master-Slave, trong đó có một Master điều phối tất cả và nhiều Slaves được điều khiển bởi Master. SPI là một giao thức song công (full duplex) nghĩa là tại cùng một thời điểm quá trình truyền và nhận có thể xảy ra đồng thời. SPI đôi khi còn được gọi là giao thức “bốn dây” vì có bốn đường giao tiếp là SCK (Serial Clock), MISO (Master Input Slave Output), MOSI (Master Output Slave Input) và SS (Slave Select ^[6]).

SCK: Xung giữ nhịp cho giao tiếp SPI, vì SPI là chuẩn truyền đồng bộ nên cần một đường giữ nhịp, mỗi nhịp trên chân SCK báo 1bit dữ liệu đến hoặc đi. Sự tồn tại của chân SCK giúp quá trình truyền ít bị lỗi và vì thế tốc độ truyền của SPI có thể đạt rất cao. Xung nhịp chỉ được tạo ra bởi chip Master ^[8].

MISO– Master Input / Slave Output: nếu là chip Master thì đây là đường Input còn nếu là chip Slave thì MISO lại là Output ^[5].

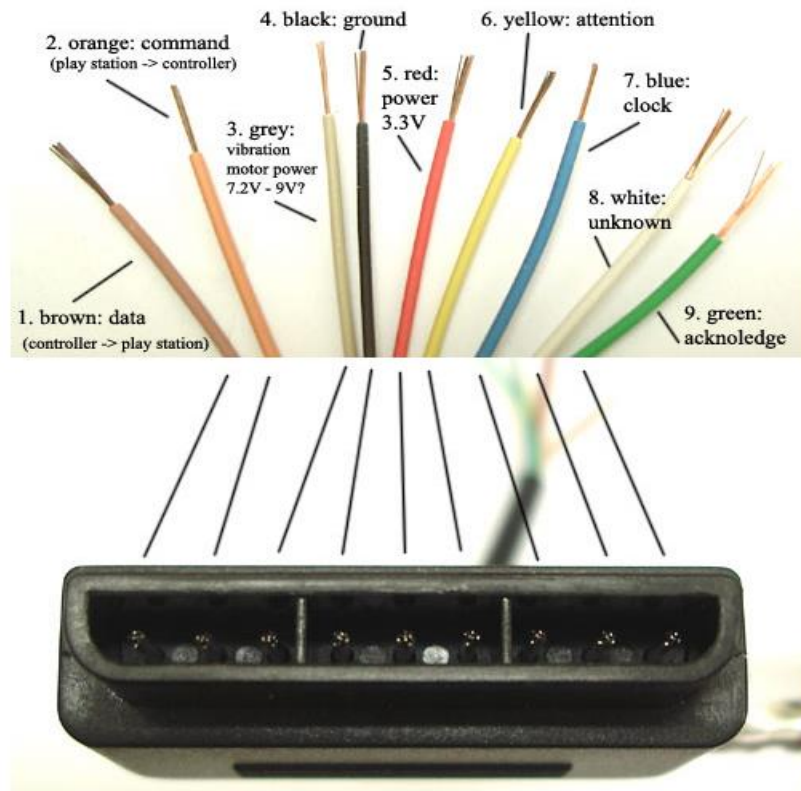
MOSI – Master Output / Slave Input: nếu là chip Master thì đây là đường Output còn nếu là chip Slave thì MOSI là Input ^[5].

SS – Slave Select: SS là đường chọn Slave cần giao tiếp, trên các chip Slave đường SS sẽ ở mức cao khi không làm việc ^[5].

Hoạt động: mỗi chip Master hay Slave có một thanh ghi dữ liệu 8 bits. Cứ mỗi xung nhịp do Master tạo ra trên đường giữ nhịp SCK, một bit trong thanh ghi dữ liệu

của Master được truyền qua Slave trên đường MOSI, đồng thời một bit trong thanh ghi dữ liệu của chip Slave cũng được truyền qua Master trên đường MISO. Do hai gói dữ liệu trên hai chip được gửi qua lại đồng thời nên quá trình truyền dữ liệu này được gọi là “song công”.

2.6.2. Giao tiếp giữa cần điều khiển PS2 với Vi điều khiển.



Hình 2-8. Chức năng các dây trong module PS2.

Ở đây cần điều khiển PS2 đóng vai trò là Slaves. Vi điều khiển là chip Master ^[5].

Các đầu vào đầu ra tương ứng của PS2 là:

- MISO: ⇔ dây 1. Brown (dây Data)
- MOSI: ⇔ dây 2. Orange (dây command)
- SS: ⇔ dây 6. Yellow (dây chọn slave)
- SCK: ⇔ dây 7. Blue (dây xung clock)

Một gói dữ liệu bao gồm 3byte header và thêm 2byte command bổ sung hoặc dữ liệu điều khiển.

3byte header:

- **0x01:** byte khởi đầu quá trình truyền nhận

- **0x42**: byte main polling command. Lệnh thăm dò chính, phụ thuộc vào cấu hình điều khiển, lệnh này có thể nhận được tất cả các tín hiệu số hoặc analog của các phím
- **0x00**: lệnh chỉ có chức năng đọc dữ liệu từ PS2

Sau đây là bảng data nhận được khi nhấn các phím PS2, một gói 5byte dữ liệu.

STT	Tên phím	Command					
		Header			Data		
		0x01	0x42	0x00	0x00	0x00	0x00
Byte#		1	2	3	4	5	6
1	Lên	0xFF	0x41	0x5A	0xF7	0xFF	0x00
2	Phải	0xFF	0x41	0x5A	0xFB	0xFF	0x00
3	Xuống	0xFF	0x41	0x5A	0xFD	0xFF	0x00
4	Trái	0xFF	0x41	0x5A	0xFE	0xFF	0x00
5	Tam giác	0xFF	0x41	0x5A	0xFF	0xF7	0x00
6	Tròn	0xFF	0x41	0x5A	0xFF	0xFB	0x00
7	Chéo	0xFF	0x41	0x5A	0xFF	0xFD	0x00
8	Vuông	0xFF	0x41	0x5A	0xFF	0xFE	0x00
9	L1	0xFF	0x41	0x5A	0xFF	0xDF	0x00
10	L2	0xFF	0x41	0x5A	0xFF	0x7F	0x00
11	R1	0xFF	0x41	0x5A	0xFF	0xEF	0x00
12	R2	0xFF	0x41	0x5A	0xFF	0xBF	0x00
13	Select	0xFF	0x41	0x5A	0x7F	0xFF	0x00
14	Start	0xFF	0x41	0x5A	0xEF	0xFF	0x00
15	L3	0xFF	0x41	0x5A	0xBF	0xFF	0x00
16	R3	0xFF	0x41	0x5A	0xDF	0xFF	0x00

Bảng 2-1. Data của các phím nhấn PS2

Ở các bảng sau đây, mỗi vị trí trong 8bit có một nút, khi nhấn, bit ở vị trí đó về 0

Hight Byte				Low Byte			
0	1	2	3	4	5	6	7
Select	L3	R3	Start	↑	→	↓	←

Bảng 2-2. Gói dữ liệu các nút trong byte thứ 4

Hight Byte				Low Byte			
7	6	5	4	3	2	1	0
L2	R2	L1	R1	△	○	×	□

Bảng 2-3. Gói dữ liệu các nút trong byte thứ 5

- **0x41:** Chế độ thiết bị: mức cao (**4**) cho biết chế độ (0x4 là digital, 0x7 là analog, 0xF là cấu hình/ thoát), mức thấp (**1**) là có bao nhiêu word 16bit theo sau header, mặc dù playstation không phải lúc nào cũng đợi tất cả các byte này (trong trường hợp các nút do chỉ có 2byte 4 và 5 nên chỉ có 1word).
- **0x5A:** Luôn là 0x5A, giá trị này xuất hiện ở một số nơi không có chức năng.

Nhận tín hiệu analog từ hai joystick và các nút nhấn.

Để gửi nhận được tín hiệu analog, cần cấu hình lại commend 0x44, có tác dụng chuyển đổi qua lại giữ hai tín hiệu analog và digital, chỉ có thể hoạt động khi dạng cấu hình là F3, tức là phải có 3word sau header (6 byte).

Byte#	1	2	3	4	5	6	7	8	9
Commend	0x01	44	00	01	03	00	00	00	00
Data	0xFF	F3	5A	00	00	00	00	00	00

Bảng 2-4. Bảng config sang chế độ gửi tín hiệu analog

- **01:** set analog mode
- **03:** khoá điều khiển để người dùng không thể chuyển lại digital bằng nút

Theo mặc định, các giá trị analog của lực nhấn các nút sẽ không được trả lại, để có thể kích hoạt chúng, cần có lệnh 0x4F, cũng như 0x44, chỉ có thể hoạt động khi cấu hình là F3.

Byte#	1	2	3	4	5	6	7	8	9
Commend	0x01	4F	00	FF	FF	03	00	00	00
Data	0xFF	F3	5A	00	00	00	00	00	5A

Bảng 2-5. Bảng config sang chế độ gửi tín hiệu analog từ lực nhấn

- **FF FF 03:** 18bit trong đó ứng 18byte trả về là 2byte trạng thái nút nhấn, 4byte giá trị analog của hai cần gạt (một cần gạt hai giá trị analog theo hai phương X và Y có khoảng giá trị 0 – 255, trạng thái đứng yên ban đầu là 125), 12byte giá trị analog ứng với các lực nhấn ở 10 hai nút

Sau khi thoát khỏi cấu hình bằng hàm 0x43, khi dùng hàm 0x42, ta có thể nhận về 18byte dữ liệu và nhận được tín hiệu analog.

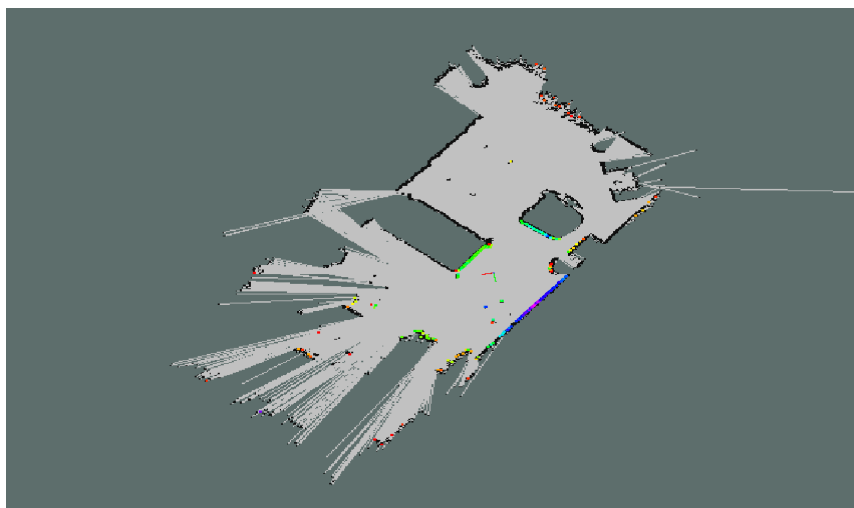
2.7. LIDAR.^[11]

RPLIDAR A1 dựa trên nguyên lý laser và sử dụng phần cứng xử lý và thu nhận tầm nhìn tốc độ cao do Hãng Slamtec phát triển. Hệ thống đo dữ liệu khoảng cách trong hơn 8000 lần mỗi giây.



Hình 2-9. RPLIDAR A1

LIDAR bắn tia laser đa hướng 360 độ, chạy theo chiều kim đồng hồ và quét môi trường xung quanh của nó và sau đó tạo ra một bản đồ phác thảo trong môi trường thực.



Hình 2-10. Bản đồ trả về từ LIDAR

Tỷ lệ lấy điểm mẫu của LIDAR trực tiếp quyết định xem robot có thể lập bản đồ nhanh và chính xác hay không. RPLIDAR cải thiện hệ thống thuật toán và thiết kế quang học bên trong để làm cho tốc độ mẫu lên tới 8000 lần với tần số 10Hz.

RPLIDAR có chi phí thấp phù hợp cho ứng dụng SLAM robot trong nhà.

2.7.1. Công nghệ in 3D

2.7.1.1. In 3D là gì?

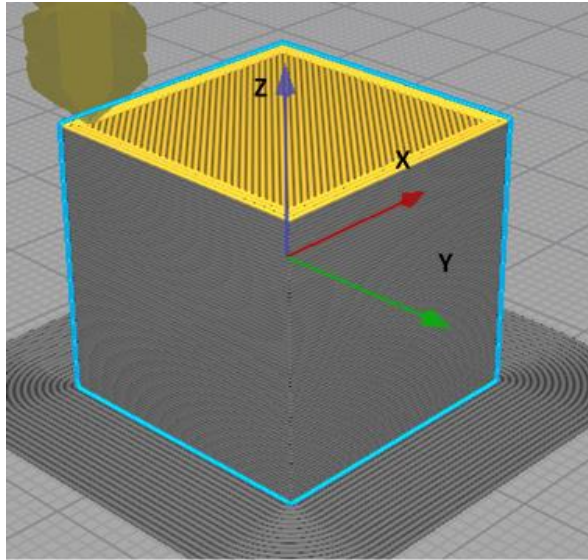
In 3D là một công nghệ tiên tiến cho phép bạn tạo một vật thể từ mô hình 3D, là công nghệ tạo mẫu nhanh với mục đích: tạo mẫu nhanh hơn và rẻ hơn. Ngày nay, công nghệ in 3D đã thay đổi rất nhiều: việc chế tạo rẻ hơn, dễ dàng hơn, công cụ hỗ trợ và cộng đồng rộng lớn khiến cho việc tiếp cận với công nghệ in 3D không còn khó, **kể cả với sinh viên.**



Hình 2-11. Máy in 3D

Hiện nay ở Việt Nam chúng ta dễ dàng tiếp cận với ba công nghệ in 3D chính: Công nghệ SLS, Công nghệ SLA, công nghệ FDM.

- Công nghệ FDM (Fused Deposition Modeling): Máy in 3D dùng công nghệ FDM xây dựng mẫu bằng cách đun nhựa nóng chảy rồi hóa rắn từng lớp tạo nên cấu trúc chi tiết dạng khối
- Cura là phần mềm xuất Gcode cho máy in 3D, là phần mềm mã nguồn mở của Ultimaker, là phần mềm hỗ trợ rất tốt cho máy in 3D và có cộng đồng sử dụng rộng lớn. chúng em chọn thông số theo những yêu cầu của chi tiết, lưu các file lại dưới dạng Gcode. Máy in sẽ in ra đúng chi tiết chúng ta mong muốn



Hình 2-12. In 3D công nghệ FDM trên phần mềm Cura

2.7.1.2. Vật liệu in 3D

Trong dự án này, chúng em sử dụng vật liệu PLA vì những ưu điểm của nó:

- Không độc
- Rẻ
- Dễ in

2.7.1.3. Đặc điểm của công nghệ in 3D FDM

Máy in 3D dùng công nghệ FDM xây dựng mẫu bằng cách đun nhựa nóng chảy rồi hoá rắn từng lớp tạo nên cấu trúc chi tiết dạng khối.

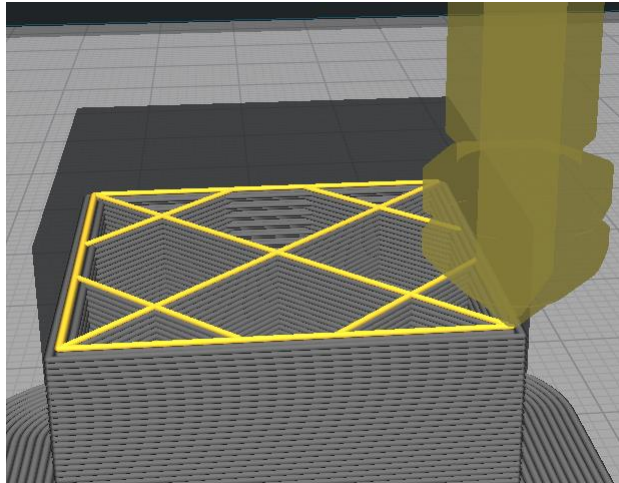
Vì đặc điểm của công nghệ in 3d FDM là theo lớp, nên cơ tính của chi tiết khác nhau theo phương tác động nên cần lưu ý chọn hướng in trước khi thiết kế

- Ưu điểm

- Dễ dàng thiết kế, gia công sản phẩm
- Chi phí thấp

- Nhược điểm

- Sản phẩm in ra có độ nhám lớn
- Thời gian in lâu
- Kích thước vật in được còn nhỏ



Hình 2-13. Mô phỏng quá trình in theo lớp

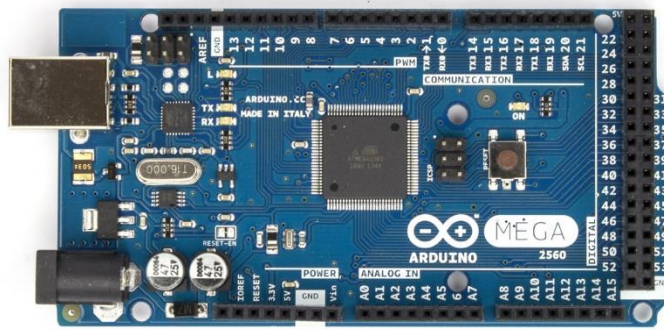
⇒ Với những đặc điểm và ưu điểm của công nghệ in 3D FDM và phần mềm CURA, nhóm quyết định chọn phương pháp gia công bằng công nghệ in 3D FDM để tạo phân vỏ cho robot của nhóm.

2.7.2. Lựa chọn thiết bị

2.7.2.1. Bộ điều khiển board Arduino Mega 2560

Arduino Mega 2560 là một vi điều khiển dựa trên nền ATmega 2560. Có 54 đầu vào/đầu ra số (trong đó có 15 đầu được sử dụng như đầu ra PWM), 16 đầu vào analog, 4 chân UARTs (cổng nối tiếp phần cứng), một 16 MHz dao động tinh thể, kết nối USB, một jack cắm điện, một đầu ICSP, và một nút reset. Chứa tất cả mọi thứ cần thiết để hỗ trợ các vi điều khiển, chỉ cần kết nối với máy tính bằng cáp USB hoặc sử dụng bộ chuyển đổi AC – DC hoặc pin. Arduino Mega tương thích với hầu hết các shield được thiết kế cho Arduino Duemilanove hoặc Diecimila.

Lý do đề tài chọn vi điều khiển Mega2560 vì bộ nhớ flash của MEGA rất lớn gấp 4 lần so với UNO (128kb) với vi điều khiển ATmega1280, ATmega328p, ... và những họ vi điều khiển khác. Rõ ràng, những dự án cần điều khiển nhiều loại động cơ và xử lý nhiều luồng dữ liệu song song (3 timer), nhiều ngắt hơn (6 cổng interrupt), ... có thể được phát triển dễ dàng với Arduino MEGA, chẳng hạn như: máy in 3d, Quadcopter, ...



Hình 2-14. Board Arduino Mega 2560

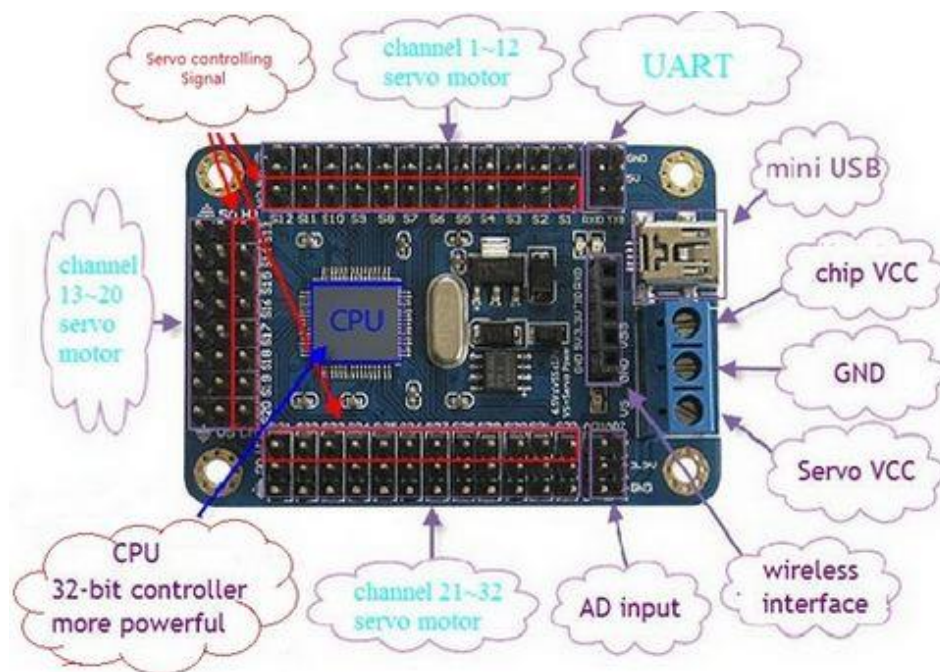
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Bảng 2-6. Tóm tắt thông số Arduino Mega 2560

2.7.2.2. Board 32 servo Controller.

Mạch điều khiển 32 RC Servo được sử dụng kết hợp với phần mềm trên máy tính qua cổng USB, tay cầm không dây PS2 hoặc kết nối với Vi điều khiển qua giao tiếp UART giúp bạn có thể dễ dàng điều khiển Mạch điều khiển 32 RC Servo có cách sử dụng và kết nối dễ dàng, phần mềm của mạch chạy trên hầu hết các hệ điều hành phổ biến hiện nay: Windows 7, Linux, MacOS, Android,...

Vì phải điều khiển một lần 23 servo, đòi hỏi một lượng lớn chân PWM cần phải sử dụng. Mạch 32 Servo giúp tạo thêm không gian kết nối. Vi điều khiển Mega 2560 thì không đủ chân PWM



Hình 2-15. Sơ đồ tính năng của chân trong Board 32 Servo Controller

Điện áp sử dụng:	5VDC (cấp quá 5VDC sẽ làm cháy mạch).
Điện áp ngõ ra RC Servo:	5VDC.
CPU:	32bit
Hỗ trợ giao tiếp	USB (115200), tay cầm PS2, UART (4800, 9600, 19200, 38400, 57600, 115200)
Thời gian trễ:	1us
Tần số điều khiển:	50Hz

Bảng 2-7. Bảng thông số mạch 32 servo controller

2.7.2.3. Mạch giảm áp



Hình 2-16. Mạch giảm áp UPEC 8,3V - 6V

Đề tài sử dụng hai module giảm áp Ubec Ternigy vì Robot Kiến 25DOF cần một lượng Ample rất lớn để cung cấp cho tất cả servo digital JX5521. Mạch giảm áp thông thường phải mắc nhiều mạch song song lại để tăng ample nhưng rất cồng kềnh và phức tạp, tăng tải trọng robot.

Nên ta chọn Ubec nhỏ gọn, công suất cao, công tắc của Ubec ở chế độ 6V vì Servo hoạt động tốt ở mức điện áp 6V.

Đầu ra (Không đổi):	5v / 8A hoặc 6v / 8A
Đầu vào:	6v-12.6v (2-3cell Li-po)
Dòng không hoạt động:	60mA
Dãy hoạt động:	7.8 ~ 8.4v / 11.7v ~ 12.6v

Bảng 2-8. Thông số UPEC

2.7.2.4. Pin li-po 7000mAh

Pin Li-po là loại pin có thể sạc được nhiều lần, sử dụng chất điện phân dạng polymer khô. Pin Li-po với những ưu điểm vượt trội về tính năng và tuổi thọ nên đang được dùng trên đa số các thiết bị. Lý do đề tài chọn pin Li-po vì:

- Pin RC Li-po nhỏ, nhẹ và có thể làm ở mọi hình dáng kích thước
- Pin RC Li-po có dung lượng cao có nghĩa là nó chứa được nhiều năng lượng trong một gói pin nhỏ
- Pin RC Li-po có dòng xả cao để cung cấp năng lượng cho động cơ RC có đòi hỏi khắt khe nhất



Hình 2-17. Pin 7000mAh



Hình 2-18. Pin 5200mAh

Dung lượng	6000 mAh và 7000 mAh
Dòng xả	80C
Ngưỡng điện áp hoạt động	60mA
Ngưỡng hoạt động	7.8 ~ 8.4v (3cells)

Bảng 2-9. Thông số Pin Li-po

2.7.2.5. Động cơ RC Servo Digital RC JX5521

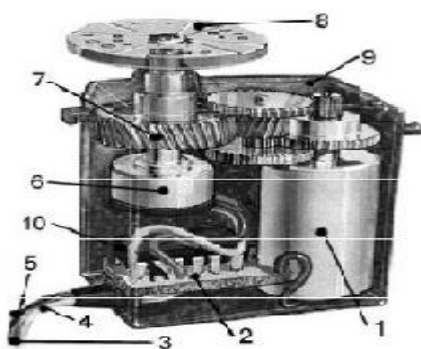
Có các loại động cơ thường được dùng để làm các mô hình robot là động cơ DC RC Servo, động cơ bước (Step motor) và Servo. Đề tài sẽ sử dụng động cơ RC Servo để điều khiển Robot vì động cơ RC Servo rẻ, có phản hồi tiếp trạng thái bằng biến trở được tích hợp ngay bên trong động cơ, việc điều khiển được đơn giản hóa chỉ cần duy nhất một chân phát tín hiệu PWM và mô hình này sẽ gần giống với việc sử dụng những động cơ Servo trong công nghiệp, dễ dàng ứng dụng thuật toán được xây dựng trong đề tài để phát triển công nghiệp.

Tại sao lại không sử dụng động cơ DC Servo và Step Servo cho dự án này? Lí do là quá đắt, muốn phát triển mô hình này lên thì cả DC Servo và Step Servo lại quá yếu hoặc quá chậm. Động cơ RC Servo sử dụng biến trở cho nên vị trí Home sẽ được đặt tại một mức điện trở cố định, điều này chỉ có ở những loại Servo đắt tiền với đĩa Encoder đặc biệt, còn Servo thường thì không. DC Servo và Step Motor thường sẽ

lấy vị trí ngay lúc cấp nguồn là vị trí Home như vậy muốn kiểm soát được tọa độ của cánh tay robot nói chung và trong trường hợp này là các chân của Robot nhện sáu chân ta phải sử dụng thêm các cảm biến hoặc công tắc hành trình,... Như vậy sẽ phải sử dụng nhiều hơn các chân vi điều khiển và gây sự cồng kềnh trong thiết kế.

Servo: phần hồi tiếp trạng thái thông dụng là encoder, nhưng với số lượng động cơ lớn (25 động cơ) đòi hỏi vi điều khiển phải đủ mạnh và yêu cầu tốc độ vi xử lý cao. Nếu không đáp ứng đủ thì dễ gây tình trạng treo vi điều khiển.

Cho nên RC Servo là phù hợp nhất với các mô hình robot di chuyển không bánh xe, cấu tạo bên trong của động cơ RC Servo như hình 3-28:



Hình 2-19. Bên trong một RC servo



Hình 2-20. Servo 5521MG

- Motor
- Electronics Board
- Positive Power Wire (Red)
- Signal Wire (Yellow or White)
- Negative or Ground Wire (Black)
- Potentiometer
- Output Shaft/Gear
- Servo Attachment Horn/Wheel/Arm
- Servo Case
- Integrated Control Chip

Trong hệ thống này, Servo đáp ứng của một các dãy xung số ổn định. Cụ thể hơn, mạch điều khiển là đáp ứng của một tín hiệu số các xung biến đổi từ 1ms – 2ms. Các xung này được gửi đi 50lần/giây. Chú ý rằng không phải xung một giây điều khiển servo mà là chiều dài của các xung biến đổi từ 1ms – 2ms. Các xung này được gửi đi 50lần/giây. Chú ý rằng không phải số xung trong một giây điều khiển servo mà là

chiều dài của các xung. Servo đòi hỏi khoảng 30 – 60 xung/giây. Nếu số này quá thấp, độ chính xác và công suất để duy trì servo sẽ giảm. Với độ dài xung 1ms, Servo được điều khiển quay theo một chiều (giả sử là chiều kim đồng hồ). Với độ dài xung 2ms, Servo quay theo chiều ngược lại.

Động cơ RC Servo Digital RC JX5521 là thích hợp nhất cho việc làm các loại robot vì nó luôn giữ lại trạng thái xung gần nhất, tránh trường hợp khung robot bị ỏ sập bất ngờ ở trạng thái không điều khiển, động cơ RC Servo Analog (Mg996,995,945,...) sẽ bị mất điều khiển khi dừng cấp xung đột ngột, dẫn đến việc khung robot bị sập bất ngờ.

Động cơ RC Servo Digital JX5521 có cấu tạo một trục xoay giống như Servo truyền thống giúp bạn dễ ứng dụng cho các thiết kế robot của mình, ngoài ra chất lượng của loại động cơ này rất tốt (tốt nhất trong các thử nghiệm hiện tại), động cơ có bánh răng kim loại, lực kéo mạnh, xoay êm, không rung, giữ vị trí tốt nhất, là một sự lựa chọn sáng giá cho thiết kế robot.

Trọng lượng sản phẩm	55.6g
Kích thước sản phẩm	40.5 * 20.2 * 44.2mm
Tốc độ	0.18sec / 60° tại 4.8VDC và 0.16sec / 60° tại 7.2VDC
Lực kéo	17.25Kg.cm tại 4.8VDC và 20.32kg.cm tại 7.2VDC
Điện áp hoạt động	4.8VDC đến 7.2VDC
Dòng điện tiêu thụ	>600mA
Chiều dài cáp	32cm

Bảng 2-10. Thông số và khối lượng RC Servo

2.7.2.1. Bộ điều khiển PS2 và HC06



Hình 2-21. PS2



Hình 2-22. HC06

- **PS2**

- Phạm vi bắt sóng Bluetooth lên đến 10m, không cần dùng dây dẫn. Tay cầm đạt độ nhạy cao, nút nhấn êm. Không có hiện tượng switch bounce của nút nhấn
- Tay cầm PS2 Wireless có 2 joystick khá linh hoạt giúp người dùng điều khiển cực kỳ chuẩn xác
- Tay cầm PS2 có bộ chuyển đổi tín hiệu kết nối phù hợp cho các bạn giao tiếp với vi điều khiển

Điện áp hoạt động	3.3V
Giao tiếp	Bluetooth
Khoảng cách tối đa	10m

Bảng 2-11. Thông số PS2

- **HC06**

Điện áp hoạt động	3.3VDC ~5VDC
Baudrate UART có thể chọn được	1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
Dải tần sóng hoạt động	Bluetooth 2.4GHz
Chip	CSR mainstream bluetooth- bluetooth V2.0 protocol standards.
Dòng điện khi hoạt động	khi Pairing 30 mA, sau khi pairing hoạt động truyền nhận bình thường 8 mA
Kích thước của module chính	28 mm x 15 mm x 2.35 mm

Bảng 2-12. Thông số HC06

Đề tài sử dụng HC06 vì robot được điều khiển bởi phần mềm trên điện thoại Android song song với việc điều khiển PS2. Người dùng có thể cài đặt phần mềm và kết nối Bluetooth.

2.7.2.2. Raspberry Pi 3



Hình 2-23. Raspberry Pi 3

CPU	Broadcom BCM2837B0 quad-core A53 (ARMv8) 64-bit @ 1.4GHz
GPU	Broadcom Videocore-IV
RAM	1GB LPDDR2 SDRAM
Networking	Gigabit Ethernet (via USB channel), 2.4GHz và 5GHz, Wi-Fi
Bluetooth	4.2, Low Energy (BLE)
Thẻ nhớ	Micro-SD
GPIO	40-pin GPIO header, populated
Ports	HDMI, 3.5mm analogue audio-video jack, 4x USB 2.0, Ethernet, Camera Serial Interface (CSI), Display Serial Interface (DSI)

Bảng 2-13. Thông số và khối lượng RC Servo

Lý do mạch Raspberry Pi 3 được sử dụng trong đề tài này vì:

- Raspberry Pi 3 Model B+ (Made in UK) là board mạch máy tính nhưng được sử dụng nhiều nhất hiện nay, ngoài việc sử dụng để hệ điều hành Linux hoặc Windows 10 IoT, máy còn có khả năng xuất tín hiệu ra bốn mươi chân GPIO giúp bạn có thể giao tiếp và điều khiển vô số các board mạch phần cứng khác để thực hiện vô số các ứng dụng khác nhau, máy có kích thước nhỏ gọn, giá thành phải chăng, cách sử dụng dễ dàng, chỉ cần cài hệ điều hành vào thẻ nhớ và cấp nguồn là có thể sử dụng
- Máy tính Raspberry Pi 3 Model B+ (Made in UK) có cộng đồng sử dụng rất lớn trên thế giới, đây chính là ưu điểm lớn nhất của Raspberry Pi, điều này giúp chúng em có thể tìm nguồn tài liệu cũng như hỗ trợ rất dễ dàng trên Google hoặc trang chủ Raspberry Pi. Đối với thuật toán dùng Robot ROS, SLAM LIDAR quét map cần thời gian xử lý tín hiệu nhanh, chính xác thì Raspberry Pi 3 là sự lựa chọn phù hợp

2.7.2.3. RPLIDAR A1



Hình 2-24. LIDAR

RPLIDAR A1 được sản xuất bởi hãng Slamtec được sử dụng cho các ứng dụng phát hiện vật cản, lập bản đồ bằng tia Laser trong xe, robot tự hành, hệ thống chống trộm, ..., cảm biến có độ ổn định và độ chính xác cao.

Cảm biến Laser Radar (LIDAR) RPLIDAR A1 sử dụng giao tiếp UART nên có thể dễ dàng giao tiếp với Vi điều khiển, Máy tính nhúng hoặc kết nối máy tính qua mạch chuyển USB-UART và phần mềm đi kèm, cảm biến có khả năng quét xa với khoảng cách lên đến 12m, tần số tối đa 10Hz với 8000 samples per time, phù hợp cho vô số các ứng dụng khác nhau.

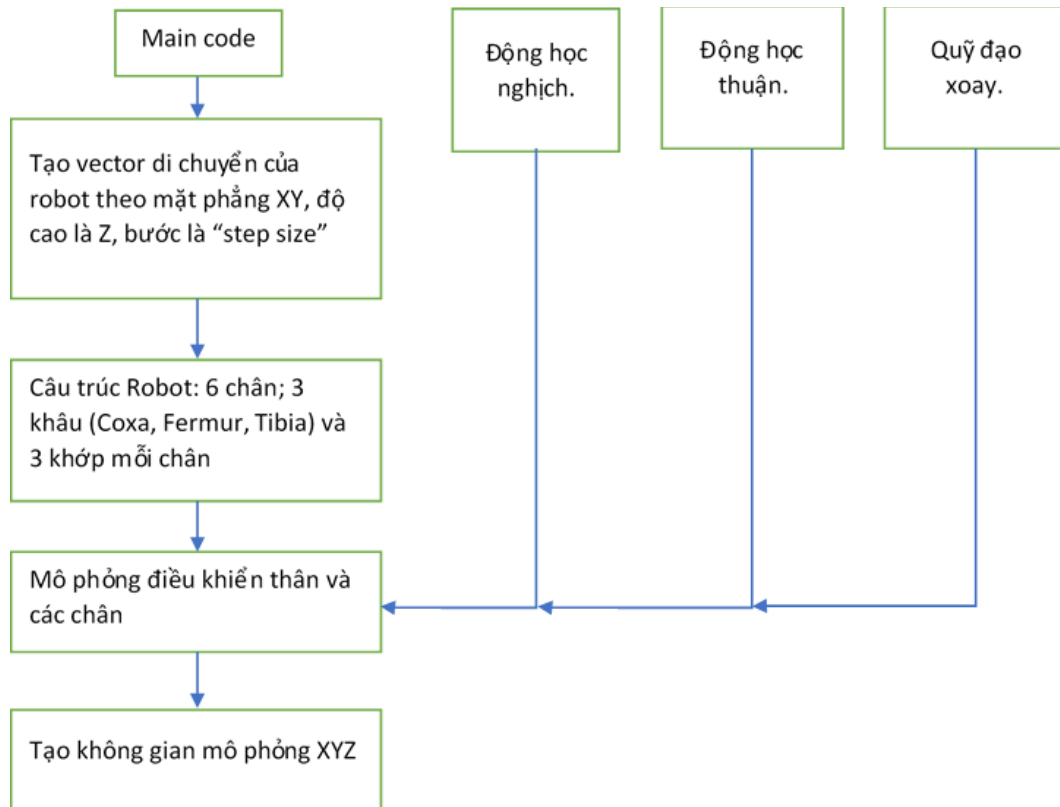
Điện áp hoạt động	5VDC
Chuẩn giao tiếp	UART
Phương pháp phát hiện vật cản	Laser
Khoảng cách phát hiện vật cản tối đa	12m
Góc quay	360°.
Tốc độ lấy mẫu tối đa	8000 Samples per time.
Tần số quét tối đa	10Hz
Kích thước	71 x 97mm

Bảng 2-14. Thông số LIDAR

CHƯƠNG 3. NỘI DUNG NGHIÊN CỨU

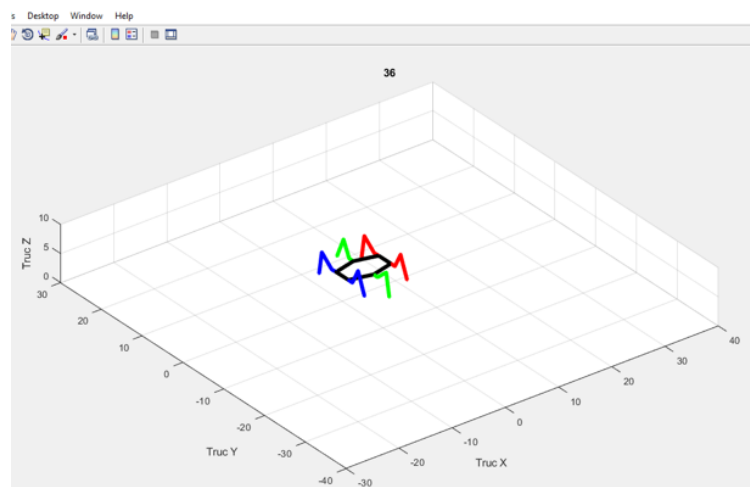
3.1. Mô phỏng trên Matlab

3.1.1. Lưu đồ



Hình 3-1. Lưu đồ trong việc mô phỏng hexapod trên Matlab

3.1.2. Kết quả mô phỏng

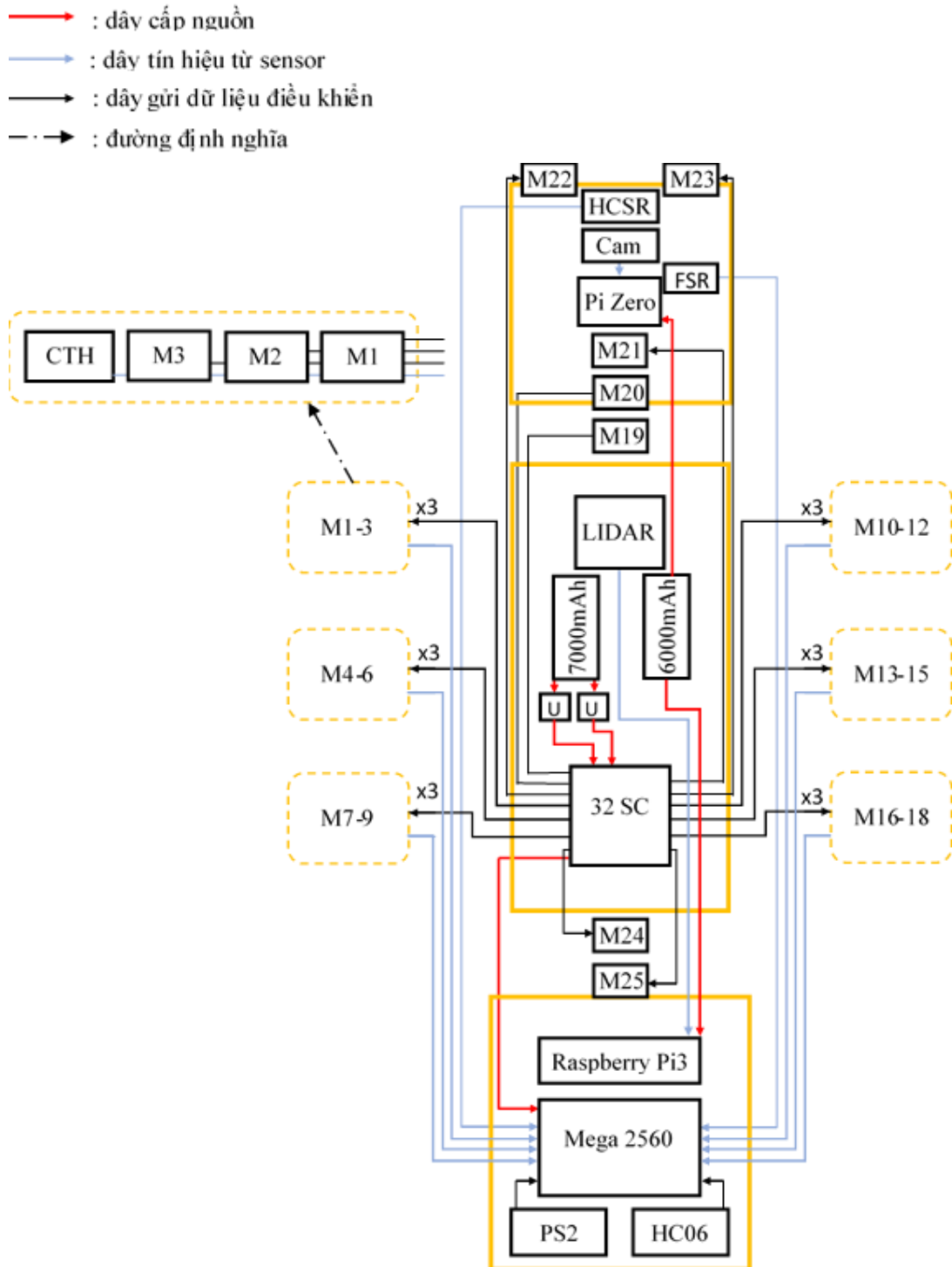


Hình 3-2. Kết quả mô phỏng sự di chuyển của Hexabod trên Matlab

3.2. Thiết kế cơ khí

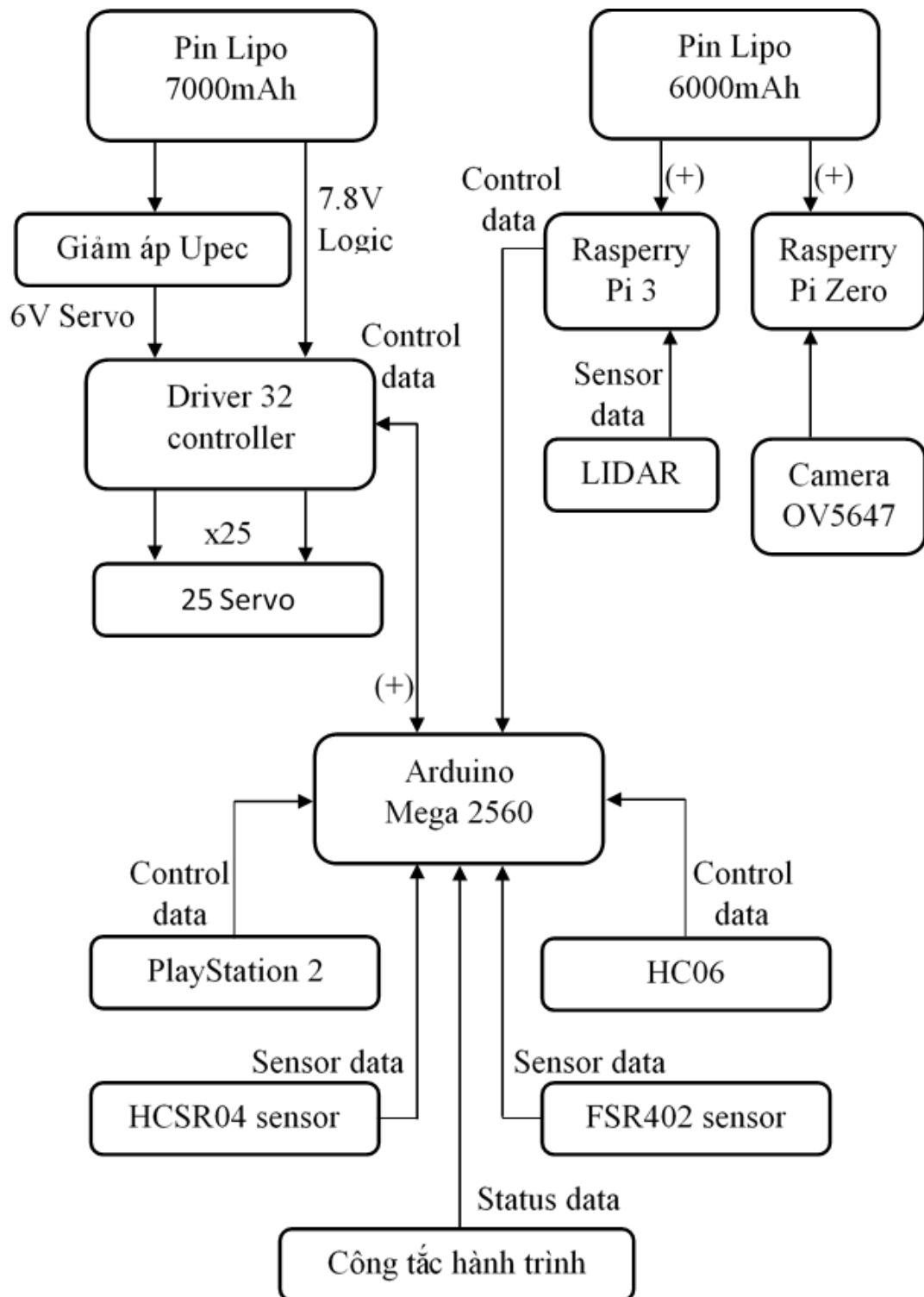
3.2.1. Lắp ráp, đi dây và kết nối các Module

3.2.1.1. Sơ đồ tổng quan kết nối cơ khí



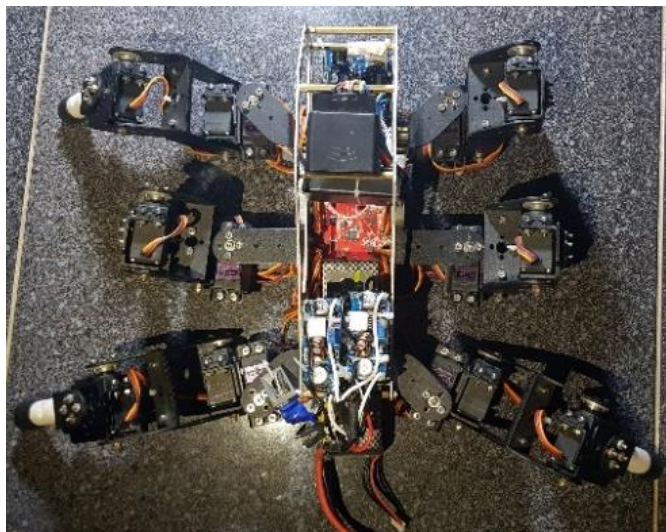
Hình 3-3 Sơ đồ tổng quan kết nối cơ khí

3.2.1.2. Sơ đồ tổng quan kết nối điện và tín hiệu



Hình 3-4 Sơ đồ kết nối các module và tín hiệu.

Trong quá trình gần một năm nghiên cứu và phát triển dự án, robot đã được phát triển qua bốn phiên bản.



Hình 3-5. Hexapod VS1

Ở phiên bản đầu (VS1), nhằm kiểm tra khả năng di chuyển của Hexapod, chúng em chỉ dùng khớp nhôm làm khung, có thể dễ dàng mua ngoài thị trường, điều này giúp tiết kiệm thời gian phải bỏ ra, robot đi lại ổn, có thể quẹo phải trái nhưng do có khối lượng lớn, thời gian hoạt động được 15 phút.



Hình 3-6. Hexapod VS2 khung nhựa

Với Hexapod VS2 (phiên bản 2), chúng em chuyển qua sử dụng nhựa làm khung, dùng công nghệ in 3D để giảm tải trọng robot chịu phải. Sau đó chúng em tiến hành kiểm tra thử về dung lượng pin và thời gian hoạt động và có được kết quả rất khả quan, robot hoạt động được 30 phút, tải trọng ngoài lên đến 2kg. Ngoài ra, chúng em còn phát triển thêm phần code bên trong, thêm thuật toán ma trận xoay, robot đã có thể xoay thân tại chỗ mà chân không di chuyển khỏi đất.



Hình 3-7. Hexapod VS3 kết hợp đầu và đuôi

Nhưng nhìn chung, cả hai phiên bản đầu chỉ mới có chuyển động nhưng lại không có những chức năng mở rộng phục vụ cho nhu cầu phát triển mô hình học tập. Một phần nguyên do là không đủ không gian để có thể thêm vào các module. Một ý tưởng đặt ra, chúng em thiết kế thêm đầu và đuôi, lấy hình mẫu là một con kiến để thiết kế thay vì chỉ có sáu chân như ban đầu, điều này giúp có thêm không gian mà không tăng kích thước thân, và phiên bản 3 (VS3) được ra đời. Nhưng phiên bản này chỉ có chức năng kiểm tra thuật toán trong code, đảm bảo khi thêm hai bộ phận đầu đuôi sẽ không ảnh hưởng tới hoạt động của Hexapod. Chúng em sử dụng phần thiết kế có sẵn của Jeroen Janssen để xây dựng theo nhằm tiết kiệm thời gian, tăng lượng pin từ 3000mAh lên 6000mAh để cải thiện thời gian hoạt động. Sau quá trình xử lý, kết quả là robot có thể di chuyển ổn định với hai bộ phận thêm vào, robot có thể hoạt động trong 30 phút.

Sau VS3, chúng em tiến hành lựa chọn các module muốn sử dụng, từ đó thiết kế lại cho riêng mình một AntPot có thể chứa các module đó, phần chọn lựa thiết bị xem phần 3.4.1. Về chi tiết phần thiết kế, sẽ được nói rõ ở phần này.



Hình 3-8. AntPot (Hexapod VS4)

3.2.2. Thiết kế mô hình

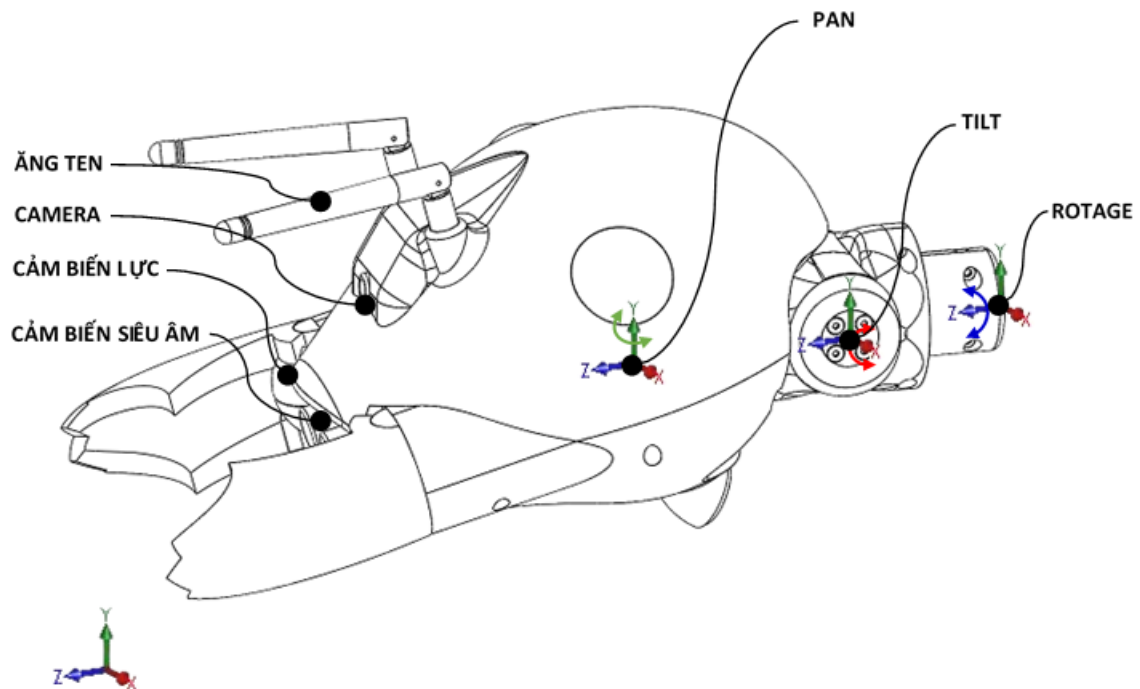
3.2.2.1. Phần Đầu

Nhóm muốn tạo một con robot có khả năng di chuyển đến những khu vực khuất tầm nhìn, hoặc cần do thám trước nên cần có Camera để quan sát và có thiết bị quét bản đồ.

Để camera có góc nhìn rộng, chúng em thiết kế phần đầu robot có ba bậc tự do giúp cho robot có khả năng xoay 180° ở trục Rotage, 40° ở trục Pan, 40° trục Tilt.

Phần đầu sử dụng bốn động cơ servo tương ứng với hai bậc tự do ở khớp Head Pan, Head Tilt và hai khớp ở răng, răng có góc đóng/mở 45° mỗi bên. Phần đầu được trang bị zero cam tại vị trí chính giữa hai râu có độ phân giải 5MP quay video 1080 30fps đi kèm với Pi zero. Cảm biến lực tích hợp vào càng cho phép đo được lực kẹp. Cảm biến siêu âm được gắn ở vị trí miệng dùng để né vật cản dưới tầm quét của Lidar. Ăngten được sử dụng để tăng tầm sử dụng PS2.

Phần đầu được chia thành 12 chi tiết gồm chi tiết đầu, hai chi tiết mắt trái- phải, năm chi tiết răng- bánh răng, ba chi tiết cổ, tất cả được lắp ghép với nhau bằng ốc và đai ốc.

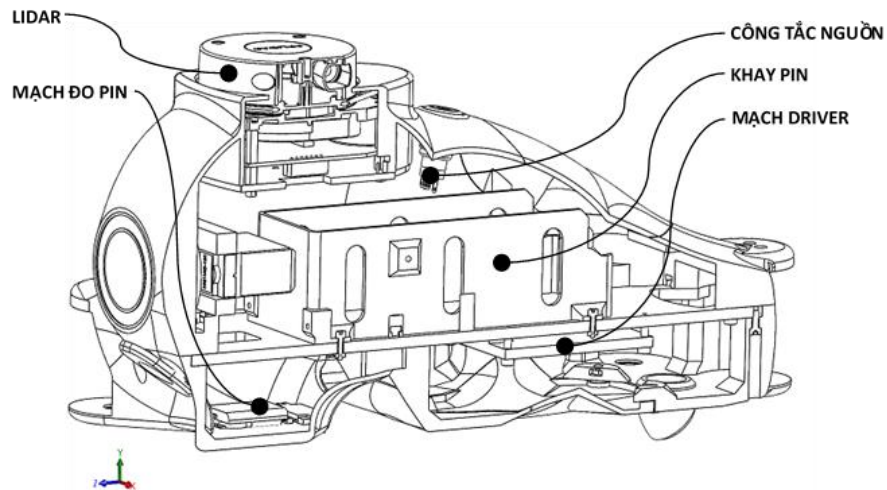


Hình 3-9. Thiết kế 3D phần đầu Hexapod

3.2.2.2. Phần thân

Thiết kế thân gồm Lidar, bộ nguồn gồm hai viên pin 7000mAh, 6000mAh, hai mạch báo pin. Để đảm bảo cho tính linh hoạt của phần đầu, động cơ trục Rotage của phần đầu được đặt trong thân và được đỡ bằng bạc đạn 35BD5220. Phần thân cho phép khớp γ ở mỗi chân quay 50° , cho phép khớp Pan của phần đuôi quay 40°

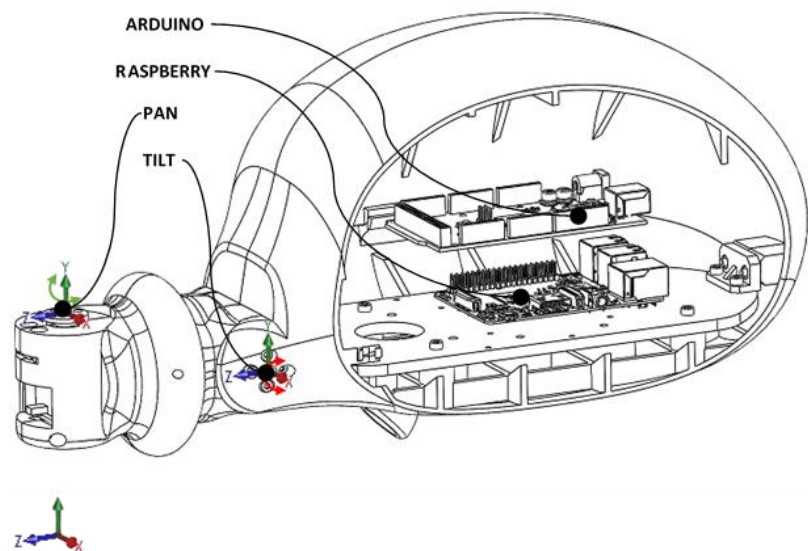
Là nơi gắn kết sáu chân, phần đầu và đuôi, cần phải đảm bảo độ vững chắc, nhưng vì hạn chế kích thước của bàn in của máy in 3D nên chúng em phải tách phần thân thành bốn phần và kết nối với nhau bằng mica ở chính giữa. Mica cũng làm nền cho khay pin và mạch.



Hình 3-10. Thiết kế 3D phần thân Hexapod

3.2.2.1. Phần đuôi

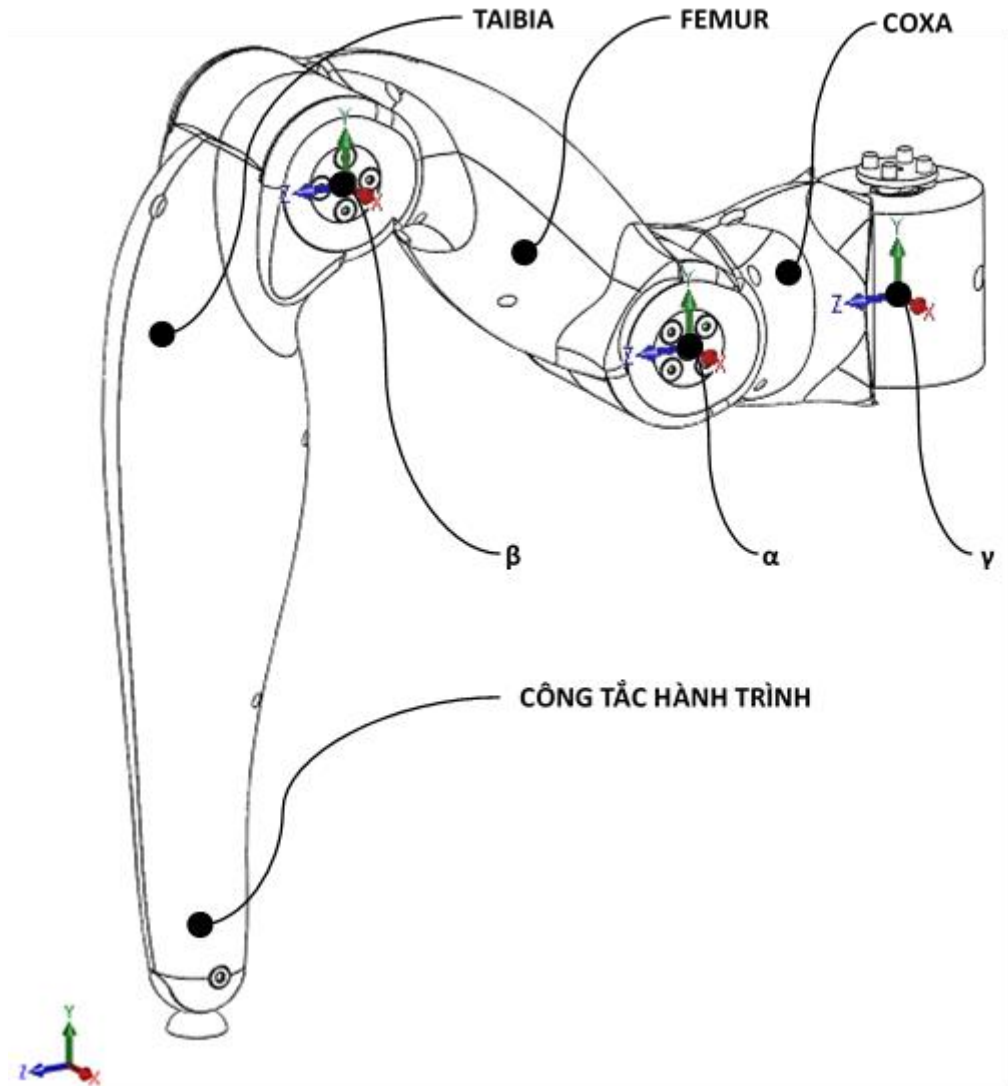
Phần đuôi sử dụng 2 động cơ servo tương ứng với 2 bậc tự do ở đuôi Tail Pan cho phép quay 40° , Tail Tilt cho phép quay 40° , bên trong được thiết kế là nơi chứa Arduino mega 2560 và Raspberry Pi3, phần đuôi có kích thước lớn hơn bản in cũng được chia thành 5 phần và kết nối với nhau bằng mica đảm bảo phần đuôi có kết cấu vững chắc và tính thẩm mỹ cao. Phần đuôi trang bị 6 con led tương ứng với 6 chân, khi chân chạm đất led sẽ sáng.



Hình 3-11. Thiết kế 3D phần đuôi Hexapod

3.2.2.2. Phần chân

Thiết kế chân gồm ba động cơ servo 5521MG tương ứng với 3 bậc tự do với kích thước khâu: Coxa 75mm, Femur 113.5mm, Taibia 221.6mm. Chân có công tắc hành trình để làm thuật toán điều khiển vượt địa hình. Các servo được giấu kín và dây dẫn được bọc trong dây lưới đảm bảo tính thẩm mỹ. Chân được thiết kế với độ dày vỏ 2mm và gân 3m đảm bảo độ cứng vững và độ bền của chi tiết, phần đỉnh ngón chân có lớp cao su chống trượt vì vật liệu PLA có độ ma sát kém với các mặt phẳng nhẵn như gạch men, bàn, ...



Hình 3-12. Thiết kế 3D phần chân Hexapod

3.2.3. Gia công

3.2.3.1. Cài đặt thông số máy in

Sợi nhựa sử dụng là PLA đường kính 1,75mm, đầu phun 0,4mm.

3.2.3.1.1. Quality

- **Layer height: 0,28mm**
- **Initial layer line width: 120%**

3.2.3.1.2. Shell

- **Wall thickness: 1.2mm**
- **Top/bottom thickness: 0.8m**
- **Optimize wall printing order:** tối ưu hóa số vị trí rút nhựa và quãng đường di chuyển

3.2.3.1.3. Infill

- **Infil density: 15%**

3.2.3.1.4. Material

- **Default printing temperature: 205°C**
- **Flow: 105%**
- **Enable retraction: Chọn**
- **Retraction extra prime amount: 0,06mm³**

3.2.3.1.5. Speed

- **Print speed: 80mm/s**
- **Wall speed: 40mm/s**
- **Top/bottom speed: 50mm/s**
- **Travel speed: 100mm/s**
- **Print Acceleration: 2000mm/s²**

3.2.3.1.6. Cooling

- **Regular fan speed at height: 0,3mm**

3.2.3.1.7. Support

- **Support overhang angle: 70°**

3.2.3.1.8. Build plate adhesion

- **Build plate adhesion type: Brim**

3.2.3.2. Chuẩn bị

- Động cơ.
25 động cơ servo 5521MG.



Hình 3-13. Servo chuẩn bị lắp ráp cơ khí

- Vỏ



Hình 3-14. Các bộ phận sau khi in, chuẩn bị lắp ráp

- Linh kiện điện tử:

STT	Tên linh kiện	Số lượng
1	Ăng ten	2
2	Arduino mega 2560	1
3	Cảm biến lực FSR402	1
4	Cảm biến siêu âm HC-SR04	1
5	Camera Pi	1
6	Led	6
7	Lidar	1
8	Mạch điều khiển servo	1
9	Mạch đo pin	2
10	Mạch giảm áp 5V-3A	1
11	Mạch giảm áp UBEC	2
12	Nút nguồn	1
13	Pin 6000mah	1
14	Pin 7000mah	1
15	Raspberri pi3	1
16	Raspberri zero	1

- Linh kiện cơ khí:

STT	Tên linh kiện	Số lượng
1	Bạc đạn B683zz	1
2	Bạc đạn B684zz	4
3	Bạc đạn FL6x12x4	21
4	Công tắc hành trình	6
5	Đai ốc M3	23
6	Đai ốc M4	4
7	Lót cao su	6
8	Ốc M2 lục giác đầu trụ 15mm	12
9	Ốc M2.5 lục giác đầu trụ 15mm	4
10	Ốc M3 lục giác đầu bằng 5mm	65
11	Ốc M3 lục giác đầu trụ 10mm	60
12	Ốc M3 lục giác đầu trụ 12mm	1
13	Ốc M3 lục giác đầu trụ 20mm	8

14	Ốc M3 lục giác đầu trụ 8mm	100
15	Ốc M4 lục giác đầu trụ 10mm	40
16	Đai ốc M3	234
17	Đai ốc M4	40

3.2.4. Bảng địa chỉ kết nối

Arduino Mega 2560	Raspberry Pi 3
35	29
37	31
39	33
41	37
43	36
45	32
GND	GND

Bảng 3-1. Kết nối Arduino Mega 2560 với Raspberry Pi 3

Aruino Mega2560	Công tắc hành trình	Led
34	SW1	D1
36	SW2	D2
38	SW3	D3
40	SW4	D4
42	SW5	D5
44	SW6	D6

Bảng 3-2. Kết nối công tắc hành trình vào Arduino Mega2560

Arduino Mega2560	HCRS-04	FSR402
VCC	VCC	VCC
GND	GND	GND
20	TRIGGER	
21	ECHO	
A2		Analog

Bảng 3-3. Kết nối HCRS-04 và FSR402 vào Arduino Mega2560

Arduino Mega 2560	PlayStation 2	HC06
VCC	VCC	VCC

GND	GND	GND
10	SEL	
11	CMD	
12	CLK	
13	DAT	
0		RX
1		TX

Bảng 3-4. Kết nối PS2 và HC06 vào Arduino Mega2560

Mega2560	32 Servo Torobot
VIN	VCC
GND	GND
19	RX
18	TX

Bảng 3-5. Kết nối 32 Servo Controller vào Arduino Mega2560

	Raspberry Pi 3	Raspberry Pi Zero
LIDAR A1	Port 1	
Camera OV5647		Jack Camera

Bảng 3-6. Kết nối LIDAR vào Pi 3, Camera OV5647 vào Pi Zero

Pin	Driver32 Torobot	Raspberry Pi 3	Raspberry Pi Zero
Li-po 6000mah		Cổng USB	Cổng USB
Li-po 7000mah	2 x Ubec15A		

Bảng 3-7. Nguồn nuôi Driver 32 Servo, Pi 3, Pi Zero và cách kết nối

Chân phải sau	
Coxa	0
Femur	1
Tabia	2
Chân phải giữa	
Coxa	4
Femur	5
Tabia	6
Chân phải trước	
Coxa	8
Femur	9
Tabia	10
Chân trái sau	
Coxa	31
Femur	30
Tabia	29
Chân trái giữa	
Coxa	27
Femur	26
Tabia	25
Chân trái trước	
Coxa	23
Femur	16
Tabia	21
HeadRotate	17
HeadPan	12
Đầu và đuôi	
HeadTilt	11
MandibleLeft	22
MandibleRight	13
AbdomenPan	3
AbbdomenTile	28

Bảng 3-8. Kết nối các Servo vào Controller

3.3. Thi công

3.3.1. Lắp chân



Hình 3-15. Bước 1



Hình 3-16. Bước 2



Hình 3-17. Bước 3



Hình 3-18. Bước 4

3.3.2. Lắp thân



Hình 3-19. Bước 1



Hình 3-20. Bước 2



Hình 3-21. Bước 3



Hình 3-22. Bước 4

3.3.3. Lắp đầu



Hình 3-23. Bước 1



Hình 3-24. Bước 2

3.3.4. Lắp đuôi



Hình 3-25. Bước 1



Hình 3-26. Bước 2

3.3.5. Hoàn thiện

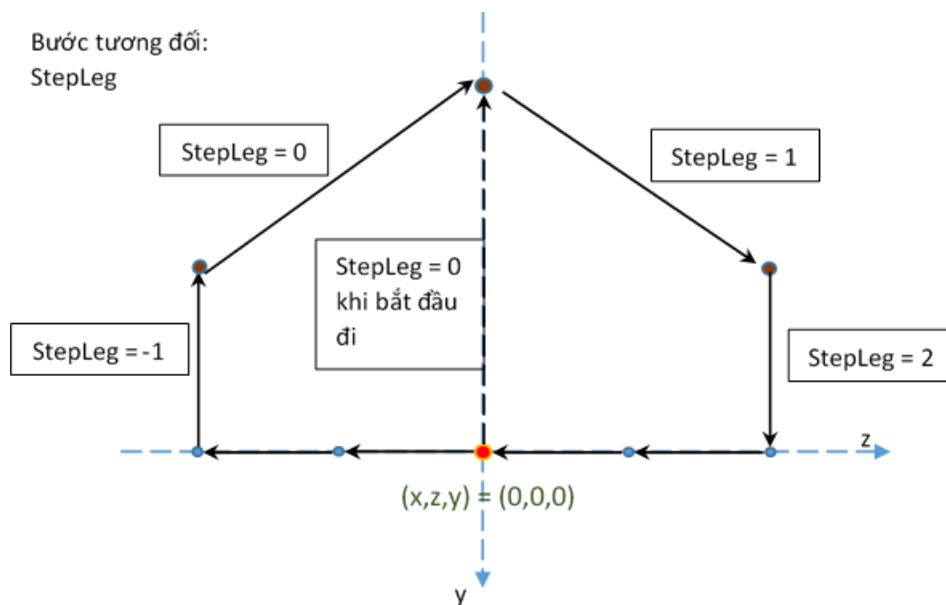


Hình 3-27. AntPot

3.4. Lưu đồ và giải thuật điều khiển cho di chuyển của Hexabod

3.4.1. Thuật toán dành cho dáng đi

Ở Chương II phần 4.1 có nói qua, Hexapod sẽ được lập trình để có thể thực một dáng đi (Gait) chủ đạo, từ đó thực hiện các chức năng di chuyển tiến lùi trái phải, xoay thân. Điều đặc biệt phải lưu ý đó là ***hướng của dáng đi này phải cùng hướng với hướng di chuyển của thân*** và di chuyển ngược chiều khi chạm đất, điều này giúp robot có thể tiến về trước.



Hình 3-28. Hình dáng của Gait trong giải thuật

Để có thể thực hiện được kiểu dáng đi đó, ta phải cho tọa độ chân thực hiện một tổ hợp các tọa độ hình thành nên Gait, càng nhiều tọa độ, bước đi càng mịn, nhưng bù lại thời gian xử lý hết lại lâu, trong báo cáo này sẽ lấy một dáng đi có tám tọa độ gồm năm tọa độ chạm đất, ba tọa độ nâng chân để giải thích về giải thuật

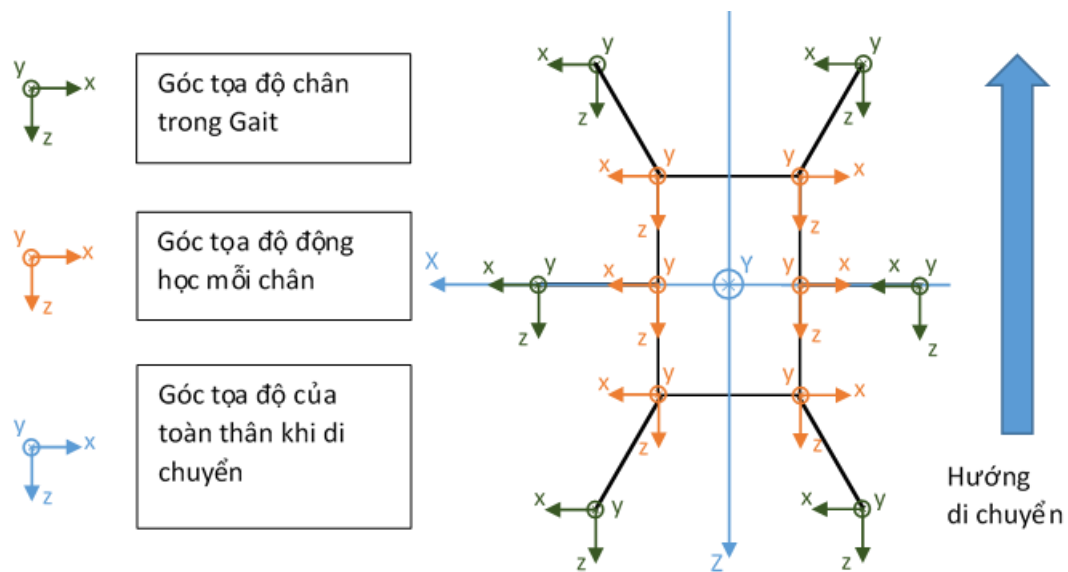
Khi thực hiện Gait, để trông như các chân hoạt động cùng lúc, ta phải cho từng chân thực hiện di dời một khoản tọa độ theo hình dáng Gait như trên, mỗi lần di dời vậy là một bước trong Gait, khi thực hiện xong hết sáu chân mới bắt đầu qua bước tiếp theo

Bước tuyệt đối: ở mỗi Gait này, do các chân của Hexapod hoạt động nâng chân tuần tự, nên mỗi chân đều có số thứ tự nâng chân của mình, đó gọi là bước tuyệt đối (hay còn có thể gọi là bước cơ sở), thứ tự này được tính từ vị trí đầu tiên của chân khi khai báo trong code và đếm tăng dần theo hướng di chuyển của Gait (hướng mũi tên trong Hình 3-33). Và bước tương đối sẽ bằng số bước lúc bấy giờ của Gait trừ cho bước tuyệt đối. Ta cho khi bước tương đối bằng 0 ($\text{StepLeg}=0$) là tại lúc chân đang được nâng cao nhất trong Gait, từ đây ta có thể tiến hành code bằng cách đặt tọa độ cho chân ứng với từng trường hợp StepLeg , gọi đó làm hàm Gait, có lưu đồ như Hình 3-35.

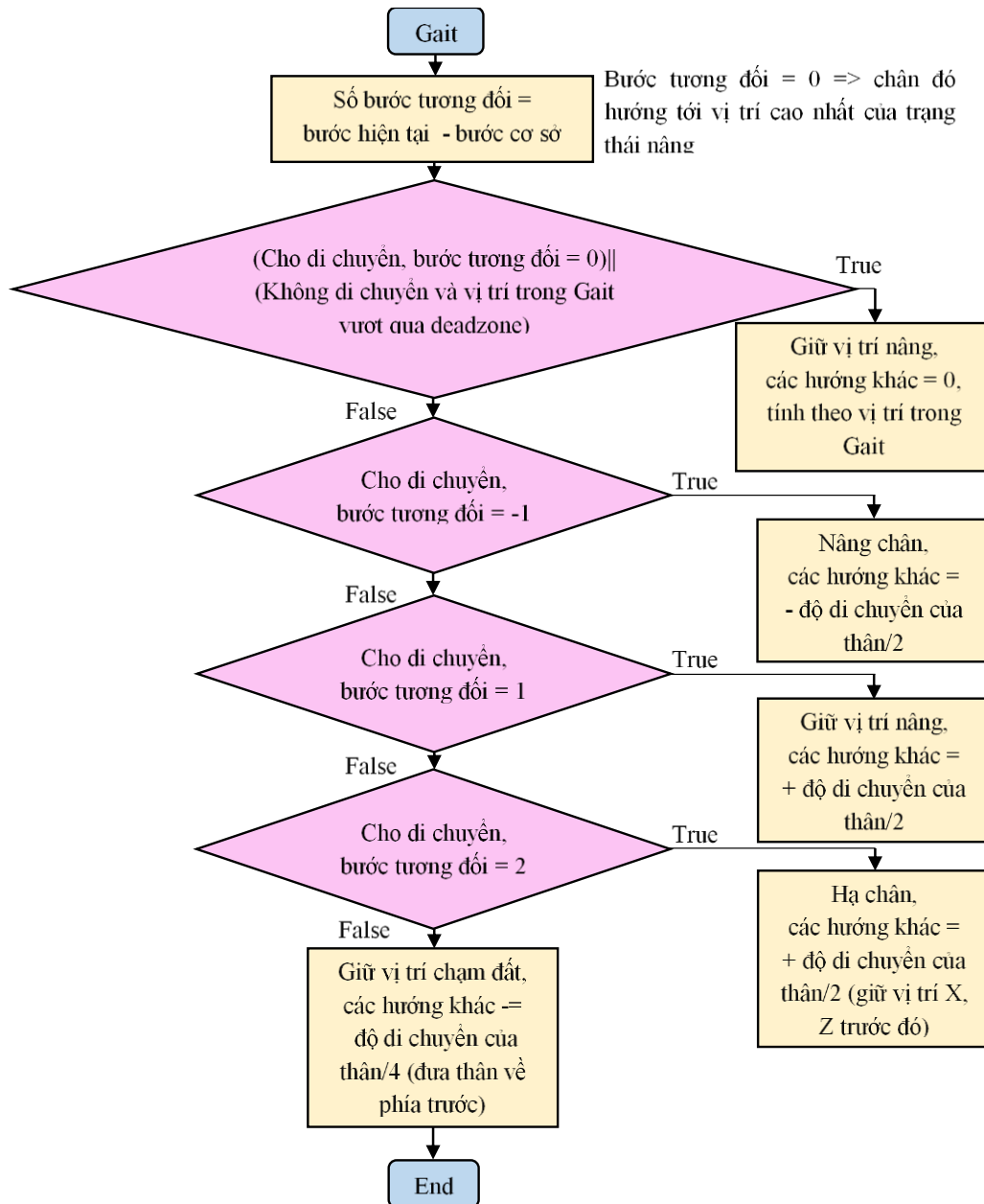
Bên cạnh đó phải cho hàm Gait trên chạy xuyên suốt 8 bước của Gait và trả về 1 khi đi hết số bước để Hexapod di chuyển liên tục. Deadzone là khoản mà ở đó tọa độ ko thay đổi, do điều khiển bằng PS2, joystick xuất ra tín hiệu analog trong khoản 0-255, dùng deadzone để giảm bớt độ nhạy, tránh việc va chạm ngoài ý muốn. Ta có lưu đồ ở Hình 3-36.

Từ đây có thể cho hàm chạy vòng lặp để có thể điều khiển Hexapod di chuyển theo hướng ta cần, có lưu đồ như Hình 3-37.

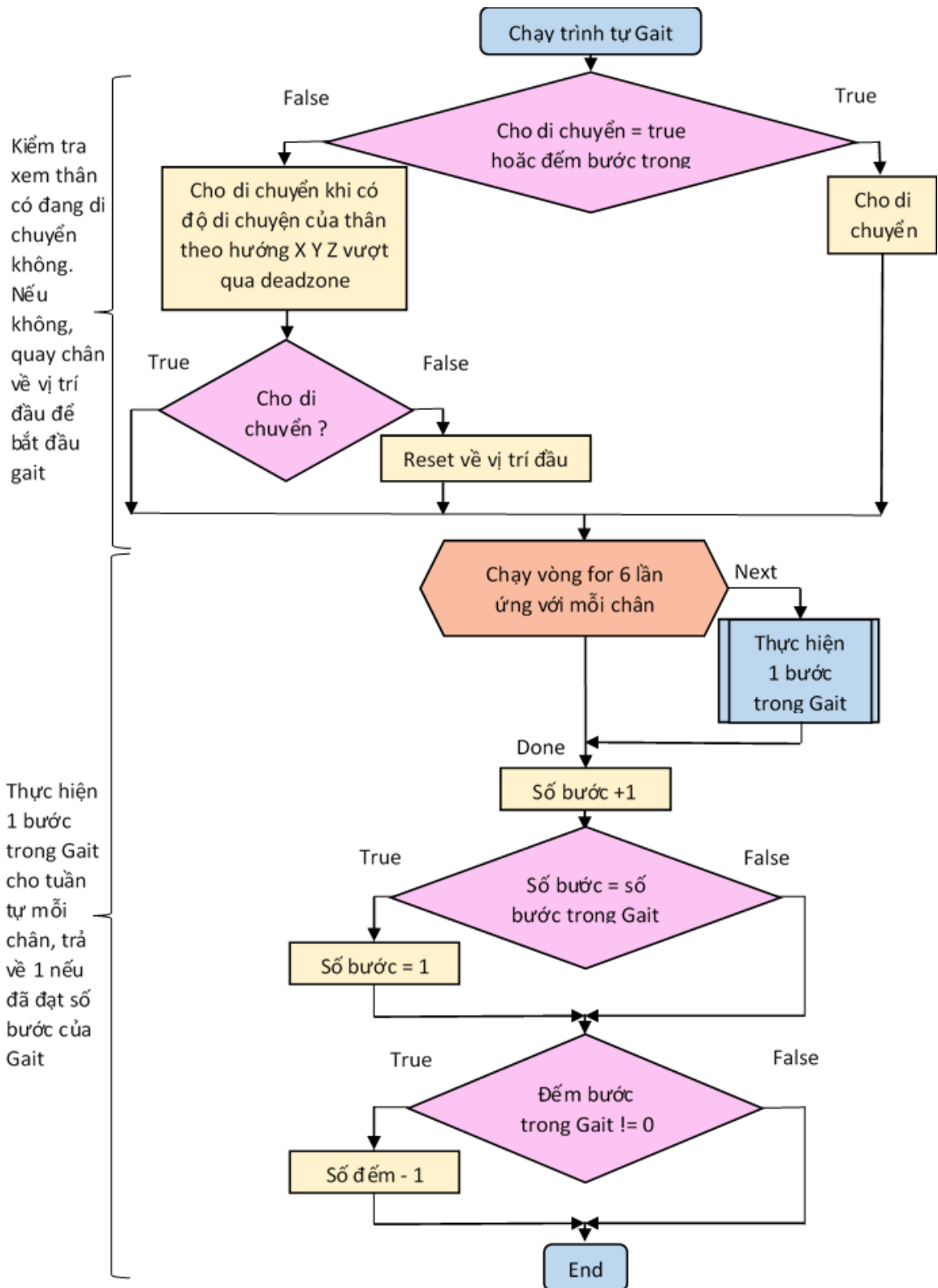
Trước khi xem phần code, chúng ta cần phải xác định trước hệ tọa độ trong toàn thân robot để dễ dàng đối chiếu:



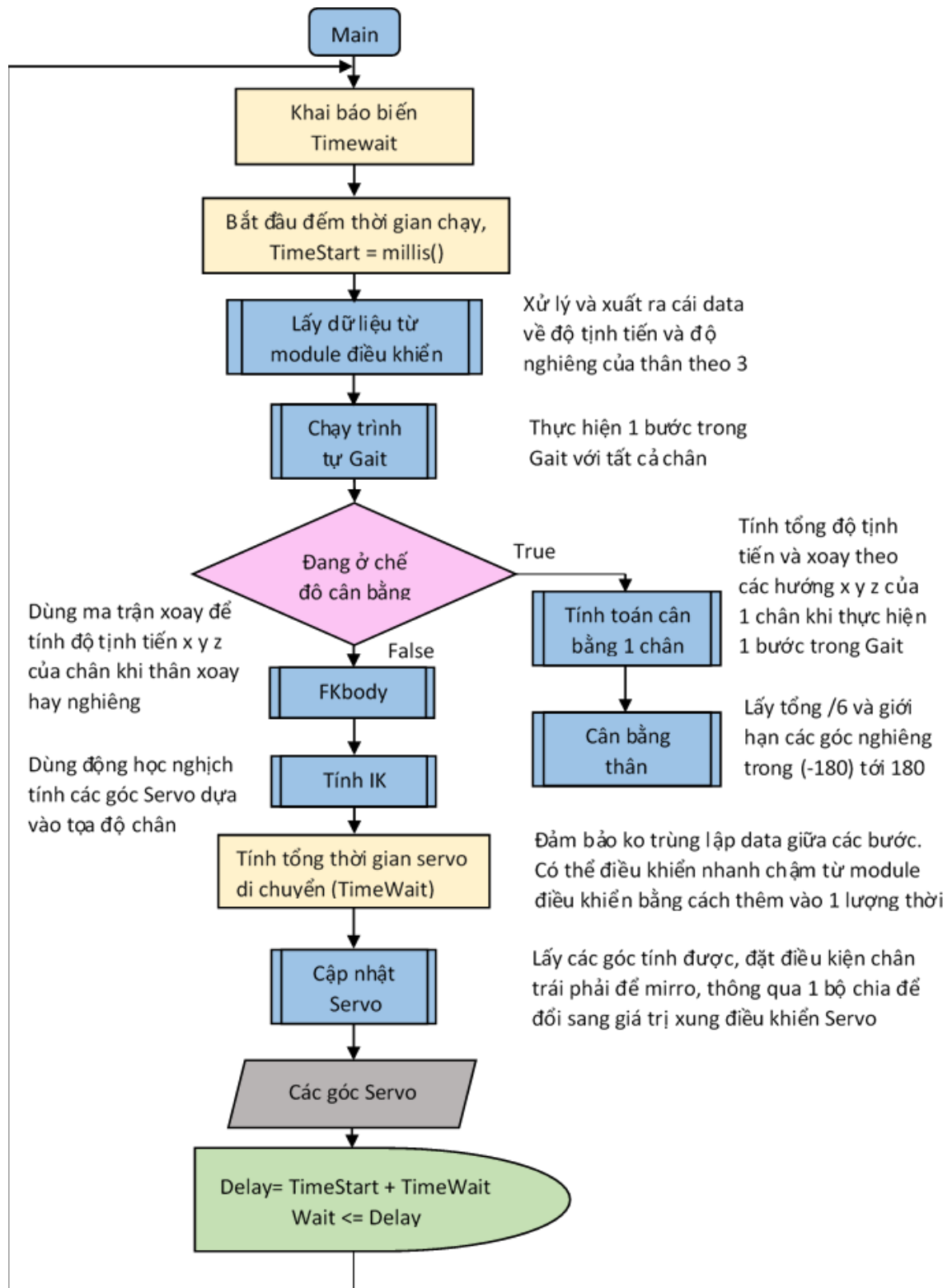
Hình 3-29. Các hệ tọa độ trên trên Hexapod



Hình 3-30. Lưu đồ giải thuật 1 bước trong Gait



Hình 3-31. Lưu đồ trình tự chạy của Gait



Hình 3-32. Lưu đồ giải thuật vòng lặp chính

3.5. Viết app điều khiển bằng Bluetooth kết nối đến HC06

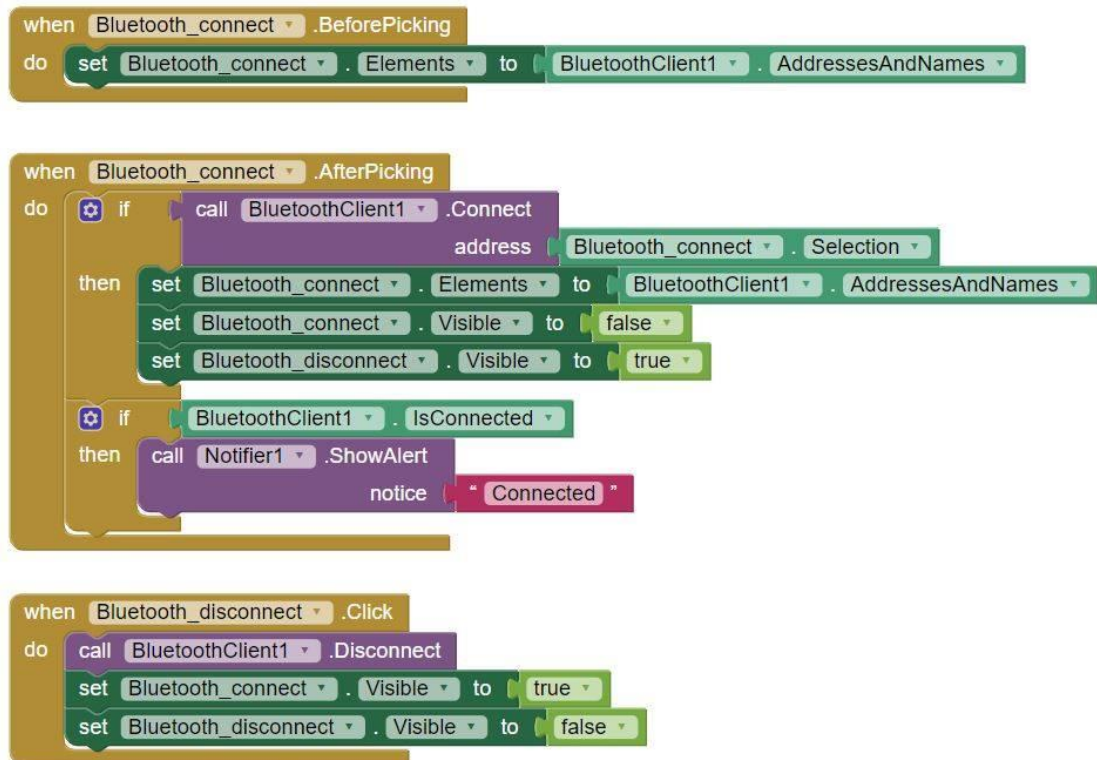
3.5.1. Giới thiệu công cụ thực hiện: MIT App Inventor

Đây là một trang Web cung cấp công cụ giúp ích cho việc tạo một app có thể sử dụng trên điện thoại, thực hiện các chức năng đơn giản. Vì là một trang Web, người dùng không phải tải về để sử dụng, thay vào đó là một tài khoản Gmail. Nhóm chọn sử dụng công cụ này bởi tính đơn giản trong quá trình tạo app, các câu lệnh có thể tạo ra chỉ với những thao tác kéo thả các block.

3.5.2. Quá trình thực hiện

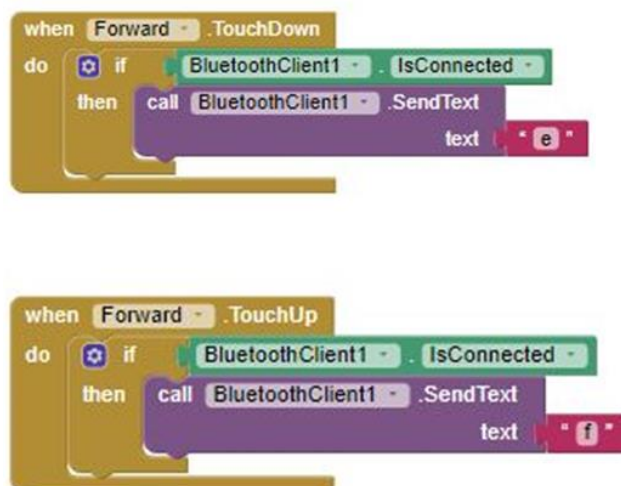
Mục tiêu ban đầu chúng em đề ra đối với app này là phải gửi được tín hiệu điều khiển thông qua đường truyền bluetooth, đảm bảo việc thực hiện không bị gián đoạn. Vậy nên chúng em tiến hành thử nghiệm kết nối và gửi dữ liệu các nút và 1 joystick bằng phiên bản đầu. Ở phiên bản này, app có thể kết nối với HC06 và gửi được tín hiệu dưới dạng ký tự ứng với mỗi nút nhấn và tín hiệu analog từ joy trong khoảng từ 0-255 thông qua cổng Serial.

- Kết nối bluetooth bằng ListPicker và Button, điều kiện ban đầu là điện thoại phải bật kết nối bluetooth, máy chứa các địa chỉ ghép đôi và khả dụng xung quanh, ListPicker BeforePicking giúp hiện ra màn hình danh sách các địa chỉ đẩy ra màn hình giao diện, sau đó AfterPicking giúp kết nối vào địa chỉ được chọn. Sau cùng là Button giúp ngắt kết nối khi không cần dùng nữa. Hai nút này sẽ luân phiên xuất hiện trên giao diện nút nhấn còn lại được kích hoạt thành công.



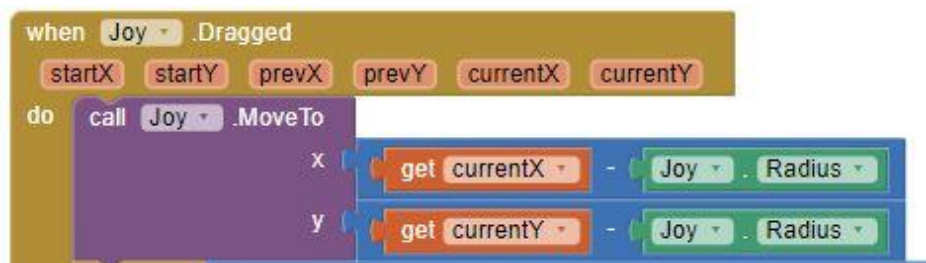
Hình 3-33. MIT kết nối bluetooth

- Gửi dữ liệu qua đường truyền bluetooth, khi kết nối bluetooth thành công, nếu nhấn vào một nút chức năng (ngoại trừ joystick), app sẽ gửi một ký tự được thiết lập, khi nhả nút cũng sẽ gửi một ký tự khác, dùng trong điều khiển khi nhấn giữ. Sau đây là ví dụ của nút nhấn tiến:



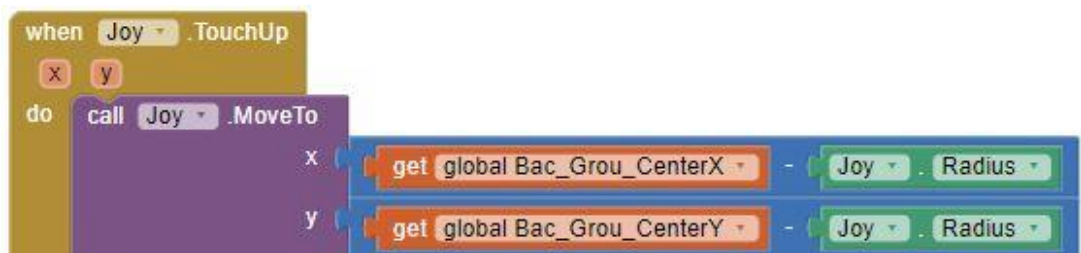
Hình 3-34. MIT gửi thông tin nút khi nhấn nhả

- Xử lý dữ liệu khi kéo joystick: để có thể sử dụng chức năng như một joystick, chúng em sử dụng màn hình Canvas và thao tác kéo thả một hình tròn thay cho nút quay. Mặt khác, có hai vấn đề cần phải giải quyết, vị trí trên Canvas và dữ liệu. Về vị trí, khi kéo thì hình tròn đó sẽ theo tay mình, thả ra thì hình tròn quay về vị trí đầu và kéo Joystick sẽ không vượt qua khỏi bản kích đường tròn ngoài, vấn đề này chúng em chủ yếu dùng lệnh Move to để đi tới tọa độ em cần. Về dữ liệu, cần phải xử lý tọa độ của Joystick để chuyển thành giá trị 0-255 rồi gửi trả dữ liệu cho HC06.
 - Để có thể theo vị trí hiện tại của ngón tay, tại điều kiện Dragged chúng em dùng lệnh Move to current, current là tọa độ mà ngón tay chạm vào màn hình Canvas



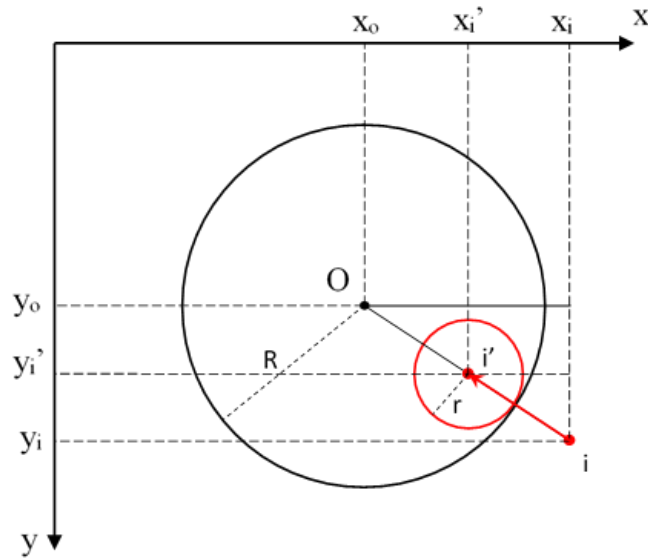
Hình 3-35. MIT Joystick hướng theo tay kéo

- Để có thể khi thả tay, hình tròn quay về vị trí ban đầu, điều kiện TouchUp dành cho việc khi thả tay ra khỏi Canvas giúp chúng em xử lý vấn đề này



Hình 3-36. MIT thả Joystick

- Để Joystick không vượt ra khỏi khung hình tròn ở nền (Background Joystick), chúng em đã phải tìm một công thức chung để có thể di dời tọa độ đúng với khung giới hạn. Hình 3-29 sau thể hiện tọa độ thực của Joystick bên trong Canvas:



Hình 3-37. Hệ tọa độ bên trong một khung Canvas

Gọi tọa độ Background Joystick trong Canvas là (x_0, y_0) , vị trí ngón tay hiện tại là (x_i, y_i) và vị trí Joystick chúng em muốn là (x_i, y_i') , bán kính Background là R , bán kính Joystick là r .

Nếu cứ dùng tọa độ này tính, chúng em phải phân trường hợp phụ thuộc vào từng góc phần tư tọa độ góc O , điều này quá mất thời gian xử lý. Vậy nên em đưa ra một ý tưởng, tịnh tiến hệ tọa độ về vị trí tọa độ của Background Joystick, dựa vào đó tính tọa độ i' . Em tính được các tọa độ sau khi tịnh tiến là $i(x_i - x_0, y_i - y_0)$, $i'(x_i' - x_0, y_i' - y_0)$

Ở vấn đề này điều kiện ban đầu đưa ra là nếu Joystick bị kéo ra khỏi giới hạn của Background, tức là khoảng cách đến tâm Background phải lớn hơn khoảng cách tọa độ giới hạn, đặt A là khoảng cách từ điểm nhấn đến tâm Background, điều kiện đó được đề ra như sau:

$$A = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} \quad (3-1)$$

$$A > R - r \quad 3-2$$

Nếu không thỏa điều kiện này, Joystick chỉ cần di chuyển tới vị trí chạm tay trên Canvas. Nếu thỏa, phải thực hiện thuật toán di dời tọa độ. Sự liên hệ giữa hai tọa độ trên, chúng em dùng đến định lý Talet trong tam giác để tính:

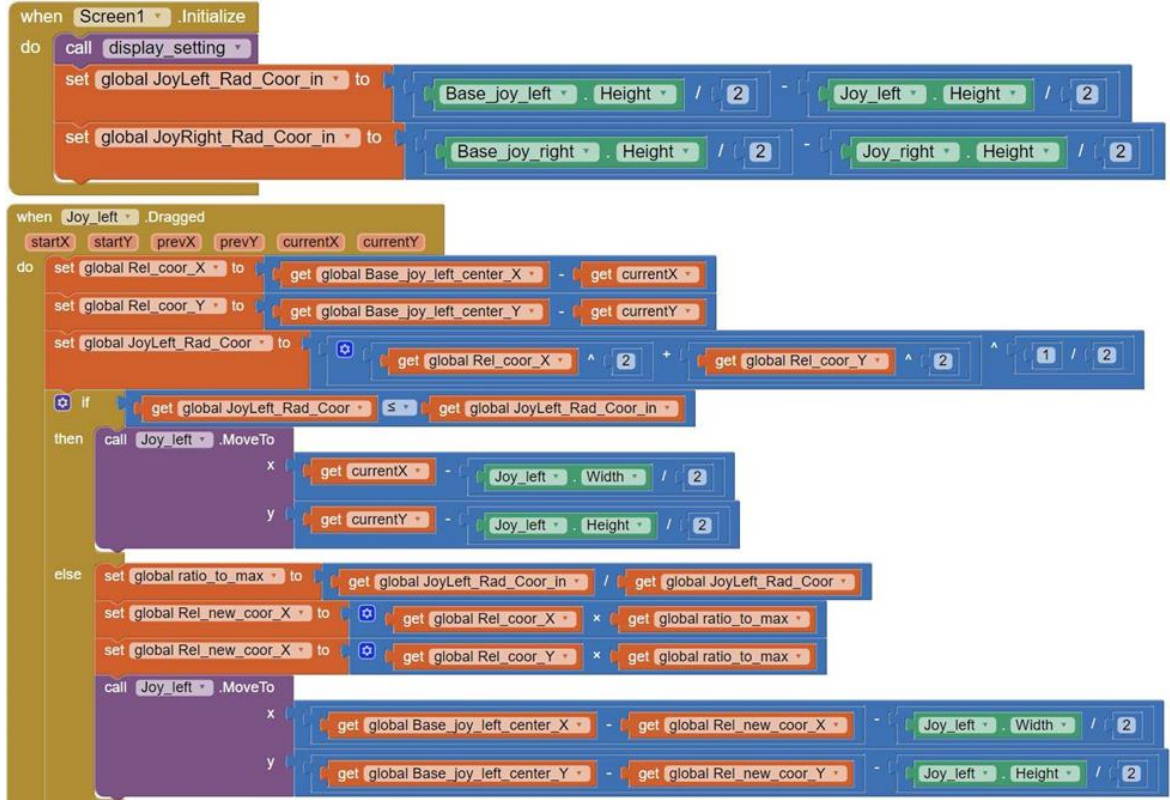
$$\frac{x_i - x_0}{x_i' - x_0} = \frac{y_i - y_0}{y_i' - y_0} = \frac{A}{R - r} \quad (3-3)$$

Từ phương trình trên, tọa độ i' có thể được tính bằng công thức sau:

$$x'_i = \frac{(x_i - x_o)(R - r)}{A} + x_o \quad (3-4)$$

$$y'_i = \frac{(y_i - y_o)(R - r)}{A} + y_o \quad (3-5)$$

Chúng em tiến hành chuyển thuật toán trên vào lập trình app:



Hình 3-38. Code giải thuật giới hạn Joystick

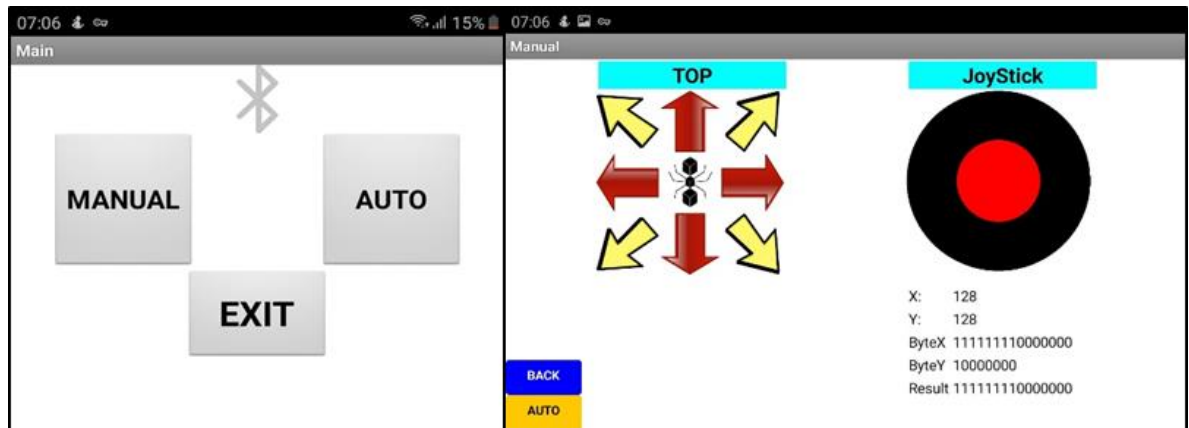
- Vấn đề cuối cùng là đổi giá trị tọa độ sang giá trị trong khoản 0-255, chúng em đưa tọa độ Joystick so với vị trí Background Joystick qua một bộ chuyển đổi, gọi giá trị dữ liệu của Joystick theo 2 hướng x, y là d_x, d_y bộ chuyển đổi này được tính theo công thức sau:

$$dx = (x_i - x_o) \frac{255}{2R - 2r} \quad (3-6)$$

Do dữ liệu theo trục y ngược với hệ tọa độ trong Canvas, nên công thức tính dy có sự thay đổi so với công thức tính dx:

$$dy = 255 - (y_i - y_o) \frac{255}{2R - 2r} \quad (3-7)$$

3.5.3. Kết quả



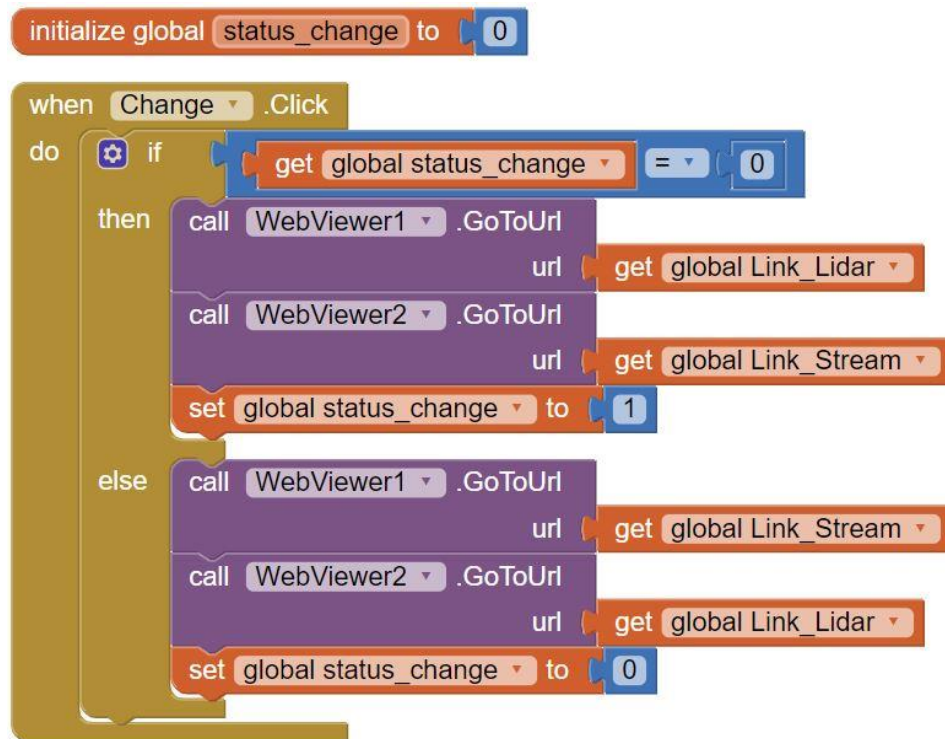
Hình 3-39. App VS1

Mục tiêu thứ hai, chúng em muốn phần mềm này phải kết nối được với mạng nhằm có được thông tin trả về từ trang html mà nhóm dùng, ngoài ra ngay trong phiên bản này, chúng em bắt đầu thiết kế bố cục cho giao diện phần mềm. Chúng em muốn app có đầy đủ những chức năng giống như một PS2 để tạo cảm giác điều khiển tương tự mà trước đó nhóm từng làm, nên em thiết kế bố cục cái nút, hình ảnh ban đầu như một PS2, bên cạnh đó thêm vào hai màn hình có thể kết nối mạng. Em tiến hành thử nghiệm hai màn hình bằng cách cho kết nối với link Youtube, một trang phổ biến, và một nút change để có thể chuyển đường link giữa hai màn hình. Chúng em sử dụng thuộc tính WebView để có thể hiện ra được trang Web muốn hướng đến, dùng lệnh GoToUrl để kết nối:

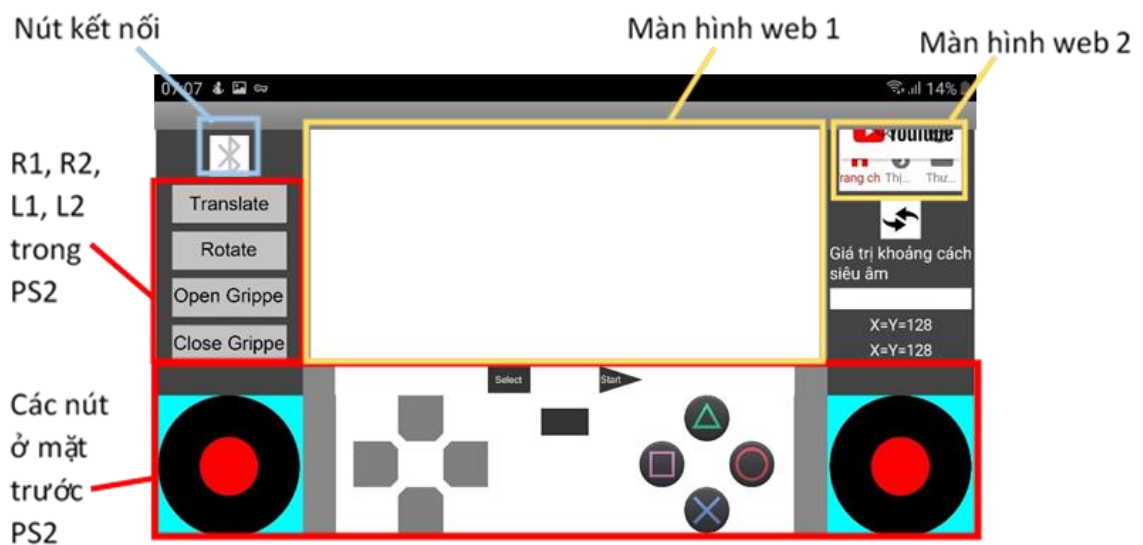


Hình 3-40. MIT kết nối WebView vào một link

Chúng em thiết kế nút Change để có thể thay đổi đường link giữa hai màn hình:



Hình 3-41. MIT nút Change thay đổi đường link hai màn hình



Hình 3-42. App VS2 với màn hình và bố cục được xác định sơ bộ

Cuối cùng, với app VS3, chúng em hoàn thiện giao diện, tiến hành chỉnh sửa ảnh nền, thay đổi màu sắc, cải thiện độ thuận mắt. Loại bỏ các chức năng không cần thiết ví dụ như khung thể hiện khoảng cách siêu âm, các nút định hướng. Tăng kích cỡ khung lướt Web để dễ dàng thao tác qua mạng, bổ sung khung Test để đặt vào đường Link đến html.



Hình 3-43. App VS3 hoàn thiện

3.6. Kết hợp chức năng quét map của LIDAR

3.6.1. Giới thiệu công cụ thực hiện: ROS và SLAM

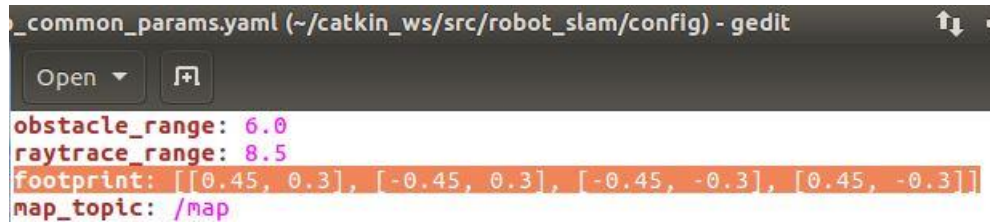
Đây là một môi trường linh hoạt cho việc viết các phần mềm robot. Nó tập hợp các công cụ, thư viện và tiêu chuẩn chung để giúp người dùng có thể dễ dàng vận hành 1 chức năng, thực hiện một hành vi phức tạp nào đó trên nhiều loại nền tảng robot khác nhau.

SLAM là hệ thống sử dụng thông tin ảnh thu được từ camera để tái tạo môi trường bên ngoài, bằng cách đưa thông tin môi trường vào một map (2D hoặc 3D). Từ đó, thiết bị (robot, camera, xe) có thể định vị (localization) đang ở đâu, trạng thái, tư thế của nó trong map để tự động thiết lập đường đi (path planning) trong môi trường hiện tại. Ở đề tài này, chúng em không dùng camera mà thay vào đó là dùng thiết bị ngoại vi là LIDAR, bắn tia laser để quét và tạo map 2D.

Điều khiển tự động thiết bị robot chia làm 3 vấn đề chính: định vị (localization), tái tạo môi trường (mapping) và hoạch định đường đi (path planning). Trong đó SLAM giúp việc định vị và tái tạo môi trường được xảy ra cùng một lúc.

3.6.2. Những cân chỉnh phù hợp với kích thước Ant-Pod

- Footprint: tọa độ của 4 góc quanh tâm LIDAR, tạo thành khoản trống biểu thị kích thước Ant-Pod



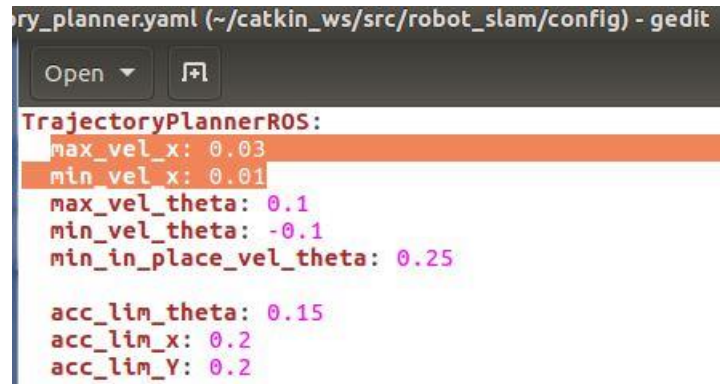
```

_common_params.yaml (~/.catkin_ws/src/robot_slam/config) - gedit
Open ▾ [icon]
obstacle_range: 6.0
raytrace_range: 8.5
footprint: [[0.45, 0.3], [-0.45, 0.3], [-0.45, -0.3], [0.45, -0.3]]
map_topic: /map

```

Hình 3-44 Footprint

- Max_vel_x, min_vel_x: tốc độ max và tốc độ min của Ant-Pod



```

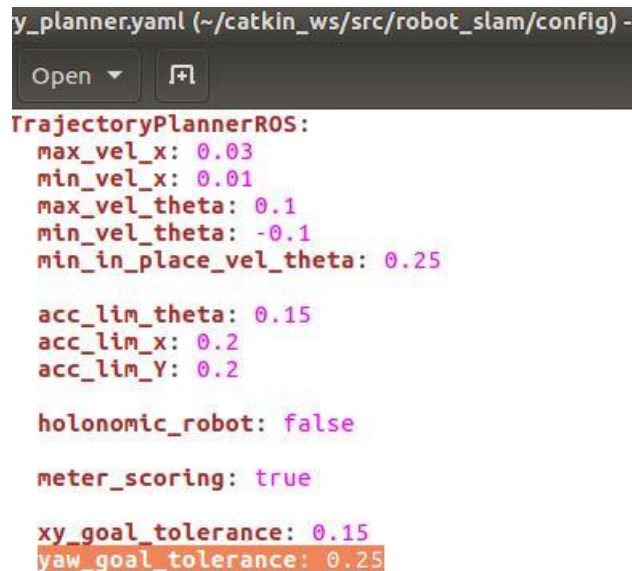
trajectory_planner.yaml (~/.catkin_ws/src/robot_slam/config) - gedit
Open ▾ [icon]
TrajectoryPlannerROS:
  max_vel_x: 0.03
  min_vel_x: 0.01
  max_vel_theta: 0.1
  min_vel_theta: -0.1
  min_in_place_vel_theta: 0.25

  acc_lim_theta: 0.15
  acc_lim_x: 0.2
  acc_lim_y: 0.2

```

Hình 3-45 Max_vel_x, min_vel_x

- Yaw_goal_tolerance: phương sai khi xoay



```

trajectory_planner.yaml (~/.catkin_ws/src/robot_slam/config) - gedit
Open ▾ [icon]
TrajectoryPlannerROS:
  max_vel_x: 0.03
  min_vel_x: 0.01
  max_vel_theta: 0.1
  min_vel_theta: -0.1
  min_in_place_vel_theta: 0.25

  acc_lim_theta: 0.15
  acc_lim_x: 0.2
  acc_lim_y: 0.2

  holonomic_robot: false

  meter_scoring: true

  xy_goal_tolerance: 0.15
  yaw_goal_tolerance: 0.25

```

Hình 3-46 Yaw_goal_tolerance

- Arg: chứa chuỗi các kích thước tương đối của LIDAR trên bản đồ:
 - 0 0 0,2: vị trí của x y z theo đơn vị met, do để LIDAR cao 0,2 met so với đất

- 3,14 0 0: độ xoay yaw pitch roll: do đặt LIDAR trong thiết kế ngược chiều trước sau



Hình 3-47 Arg

3.6.3. Nguyên lý giao tiếp giữa arduino và Raspberry Pi 3:

RPLIDAR A1 sau khi chạy tất cả các thuật toán để phát hiện vật cản và xây dựng lên bản đồ thì Raspberry Pi 3 sẽ tạo ra 2 biến liên tục thay đổi, đó chính là **Góc** (đơn vị độ) và **Hướng đi**. Sau đó Raspberry Pi 3 sẽ giao tiếp gửi dữ liệu 2 biến đó đến Arduino mega 2560. Sau đó Arduino mega sẽ quyết định đến mục tiêu di chuyển của robot theo 9 hướng cơ bản.

- Hướng 1 : đi thẳng
- Hướng 2 : đi lùi
- Hướng 3 : Xoay trái tại chỗ
- Hướng 4 : Xoay Phải tại chỗ
- Hướng 5 : đi thẳng kết hợp xoay trái
- Hướng 6 : đi thẳng kết hợp xoay phải
- Hướng 7 : đi lùi kết hợp xoay trái
- Hướng 8: đi lùi kết hợp xoay phải
- Hướng 9: Đứng yên

Khi robot di chuyển đến vị trí mới so với vị trí cũ. Góc robot sẽ bị thay đổi, ta sẽ nhận được biến của Góc.

- Raspberry Pi 3 sẽ giao tiếp với arduino mega qua các dây tín hiệu.
- Giao thức truyền từ Raspberry Pi 3 đến mega là song song (parallel). Các chân vật lý được kết nối với nhau thông qua bản sau:

Raspberry Pi 3	Arduino Mega	Tên biến
Chân 29	Chân 35	PosAngle
Chân 31	Chân 37	NegAngle
Chân 33	Chân 39	PosLinear
Chân 37	Chân 41	NegLinear
Chân 36	Chân 43	ComWrite
Chân 32	Chân 45	ComRead
Chân GND	Chân GND	

Các tín hiệu từ raspberry pi3 sẽ là tín hiệu đầu vào input cho mega. Ta sẽ có các qui định cho các hướng như sau :

- Bước 1: Chân 45 được kích lên mức cao 5v. sẽ cho phép đọc các thông tin từ Raspberry Pi3
- Bước 2: Chân 43 của mega sẽ mở đầu nhận gói data bằng cách kích mức thấp 0v

`digitalWrite(ComWrite,0);`

- Bước 3: Hướng và góc được lựa chọn theo data bằng cách kích mức cao các chân mega

Ví Dụ :

- Hướng 1: Đi thẳng

`digitalRead(PosAngle,1);`

`digitalRead(NegAngle,1);`

`digitalRead(PosLinear,1);`

`digitalRead(NegLinear,0);`

`RobotMoveForward(); // robot đi thẳng`

- Hướng 2: Đi lùi

`digitalRead(PosAngle,1);`

`digitalRead(NegAngle,1);`

`digitalRead(PosLinear,0);`

```
digitalRead(NegLinear,1);
RobotMoveBackward(); // robot đi lùi
.....
```

Các hướng còn lại sẽ thay đổi theo cách kích mức cao hoặc thấp của chân digital theo mã nhị phân

- Bước 4: Xác nhận Robot đã kết thúc di chuyển theo hướng đã chọn

Ta sẽ kết thúc 1 lần nhận data bằng cách kích mức cao chân 43 của mega

```
digitalWrite(ComWrite,1);
```

3.6.4. Các bước để khởi chạy Rviz trong nền Ubuntu :

ifconfig

////////// dùng làm share screen, của NoVNC

Tab 1

```
x11vnc -forever -display :0
```

Tab 2

```
cd ~/Desktop/noVNC-1.1.0/ && ./utils/launch.sh
```

//sua host trong page tu ubuntu thành IP

//////////

Tab 3

```
ssh ros@IP //kết nối ip
```

```
password: 12345678
```

```
cd catkin_ws/                đến không gian làm việc
```

```
source devel/setup.bash      chạy setup bên trong devel
```

```
roslaunch robot_slam rplidar.launch    khởi chạy chương trình robot slam
```

Tab 4

ssh ros@IP // kết nối ip

password: 12345678

sudo -s //cấp quyền chạy chương trình

password: 12345678

cd catkin_ws/ // đến không gian làm việc

source devel/setup.bash //chạy setup bên trong devel

roslaunch robot_slam driver // hiện lên thông tin gửi và trả giữa raspberry và arduino

Tab 5

cd ~/catkin_ws/ // không gian làm việc

source devel/setup.bash //chạy setup bên trong devel

cd src/robot_slam/scripts // chạy đến script

./client.sh //chạy đến client

Click 2D Nav Goal

Fullscreen (F11)

CHƯƠNG 4. THỰC NGHIỆM

4.1. Kết quả về mặt hoạt động phần cứng

Chúng em thực hiện cho HexaPod chạy thử ngoài thực tế và tiến hành đo đạc, thử nghiệm này được diễn ra trong môi trường bằng phẳng, nhiệt độ phòng, các chương ngại không quá thấp để LIDAR có thể phát hiện như tường, các thùng Carton.

4.1.1. Thời gian hoạt động

4.1.1.1. Raspberry Pi 3 và Raspberry Pi Zero

Raspberry đóng vai trò quan trọng khi phải truyền tải thông tin vị trí của Hexapod trong bản đồ, giúp ta nhận biết môi trường xung quanh nên thời gian hoạt động thực tế của của Raspberry rất quan trọng.

- Tiêu chí đánh giá:
 - So sánh thời gian hoạt động liên tục thực tế của hai Raspi với thời gian tính toán
 - Sạc xả 5 lần trong thời gian 2 giờ 25 phút

Chúng em tiến hành đo dòng sử dụng trong Raspberry Pi bằng USB Tester V3 được cắm trực tiếp vào cổng USB, đầu ra nối ra LIDAR. Từ kết quả đo được, do chúng em chọn nguồn cấp cho Pi là 5200mAh, thời gian Raspberry hoạt động được dựa trên lý thuyết được tính bằng công thức sau:

$$T = \frac{6000}{0,61 \cdot 1000} \approx 10(h) \quad (4-1)$$



Hình 4-1. Dùng USB Tester V3 để đo dòng trong Raspberry Pi

Trong thực tế, do không thể xả hết lượng pin nhằm đảm bảo về mặt tuổi thọ pin, một viên pin khi sạc đầy có áp là 8.15V và ngưỡng hoạt động từ 7.6-8.15V nên thời gian tính toán ra đạt 2 giờ 25 phút thì phải sạc lại, giá trị điện áp bị giảm theo số lần sạc- khoảng 300 lần (8.15V là giá trị đã giảm qua nhiều lần sử dụng), nếu là pin chưa qua sử dụng, ngưỡng trên sẽ là 8.4 ứng với 2 cell.

4.1.1.2. Các Servo

Do LIDAR và mạch điều khiển 25 servo được cấp nguồn với hai nguồn pin khác nhau, nên thời gian hoạt động của các servo so với Raspberry có sự khác nhau, chúng em cho Hexapod được bật nguồn liên tục, chia các trường hợp hoạt động như: không cho di chuyển, di chuyển liên tục và di chuyển với tải tại càng, bấm thời gian từ khi bật nguồn cho tới khi mạch báo pin báo yếu pin

- Tiêu chí đánh giá:
 - Sạc, xả 5 lần
 - Đo thời gian mà robot còn hoạt động ổn định từ lúc sạc đầy pin
- Chúng em thu được thời gian hoạt động sau:
 - Khi không hoạt động: sau khoản 45 phút thì có dấu hiệu robot bị đổ, các servo tại chân mất điện
 - Khi di chuyển liên tục: sau 18 phút thì chân di chuyển không còn ổn định, bước đi không còn đều như ban đầu, một vài góc tại các khớp bị lệch so với các chân còn lại
 - Khi có tải: tải là một viên tạ nặng 200g được kẹp bởi càng của Hexapod, chỉ di chuyển được 14 phút. Sau đó thân có xu hướng đổ về trước.

4.1.1.3. Tốc độ di chuyển và sự ổn định của robot khi di chuyển

• Tốc độ tối thiểu

Để đo được tốc độ tối thiểu của Hexapod, chúng em cho robot giảm tốc hết mức từ bộ điều khiển rồi tăng dần cho tới khi Hexapod có thể di chuyển với dáng đi ổn định, cho đi trong khoảng cách 60cm, thu được kết quả Hexapod di chuyển trong 1 phút 10s, vậy tốc độ tối thiểu của Hexapod được tính:

$$v_{min} = \frac{60}{(1 \cdot 60 + 10)} = 0.85 \text{ (cm/s)} \quad (4-2)$$

- **Tốc độ tối đa mà robot hoạt động ổn định**

Tương tự như tốc độ tối thiểu, với tốc độ tối đa, chúng em đẩy tốc độ lên cao nhất từ bộ điều khiển (trong lập trình, giá trị vận tốc cao nhất là 12 cm/s khi đi thẳng), rồi giảm dần cho đến khi bước di chuyển được ổn định. Robot đi được trong vòng 8 giây, tốc độ cao nhất có thể được tính:

$$v_{max} = \frac{60}{8} = 7.5 \text{ (cm/s)} \quad (4-3)$$

Tuy nhiên, khuyến cáo không sử dụng robot cao hơn tốc độ tối đa cho phép vì Hexapod sẽ dậm chân quá mạnh, điều này sẽ ảnh hưởng rất lớn tới kết cấu cơ khí.

- **Tốc độ hiệu quả (v_{rq} : required speed)**

Đây là tốc độ cao nhất khi Hexapod thực hiện chức năng quét map, khi vượt qua mức tốc độ này, hình ảnh map thu được sẽ giật, việc này không ảnh hưởng đến quá trình quét map, nhưng khi di chuyển tự động, việc này sẽ khiến Hexapod dừng giữa đoạn đường thay vì đi về hướng chỉ định. Không như cách đo trên, nhưng lần này chúng em cho robot đi tự động trước, giảm tốc dần cho tới khi việc di chuyển không bị dừng lại đột ngột, chúng em giữ tốc độ đó và bắt đầu tính vận tốc như cách trên, kết quả thu được là robot đi được 60cm trong 26 giây, vận tốc hiệu quả có thể được tính bằng phép tính:

$$v_{rq} = \frac{60}{26} = 2,3 \text{ (cm/s)} \quad (4-4)$$

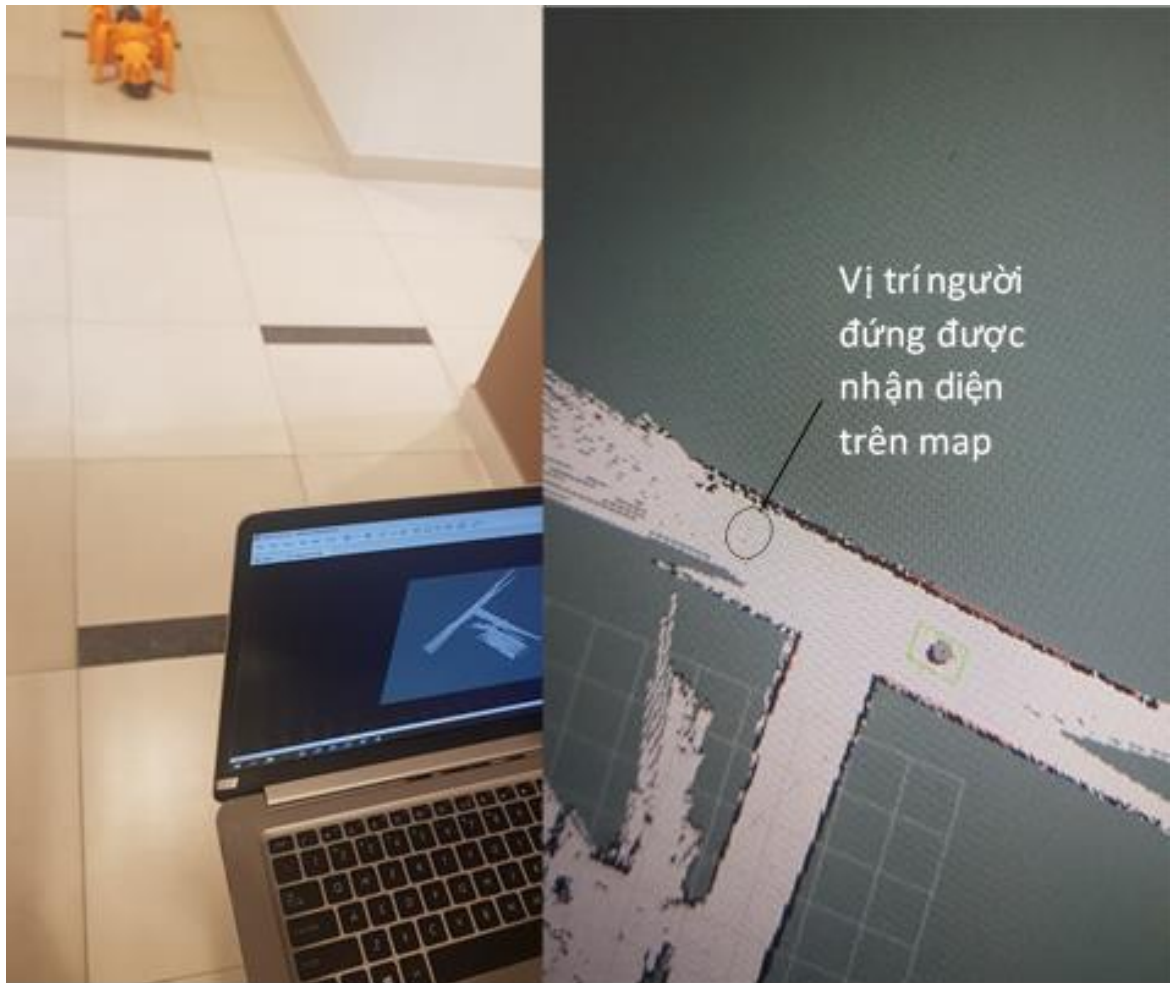
4.1.1.4. Tầm quét của LIDAR

4.1.1.4.1. Tầm quét hiệu quả

Theo thông tin của nhà sản xuất, LIDAR có đề khoảng cách tối đa quét được là 12m, chúng em tiến hành thử nghiệm xem ở khoảng cách bao nhiêu thì trên map mà LIDAR quét, người dùng có thể nhận biết rằng ở đó có người- tầm quét hiệu quả. Để giải quyết yêu cầu này, chúng em tiến hành đặt Hexapod trong một hành lang dài, đặt cho LIDAR quét map và theo dõi qua màn hình, cho một thành viên đi lùi dần cho đến khi điểm nhận biết không còn ổn định

- Tiêu chí đánh giá
 - Khởi động lại LIDAR 5 lần
 - Môi trường không có vật trong suốt

- Sau khi đo thì tầm quét hiệu quả được xác định: 340cm



Hình 4-2. Đo tầm quét hiệu quả

4.1.1.4.2. Tầm quét tối thiểu

LIDAR sẽ có một khoảng cách nào đó đủ gần để khi gửi thông tin lên map, dữ liệu sẽ bỏ qua vật cản này, tức nằm trong khoảng này thì map không hiển thị được. Cách xác định cũng tương tự như trên, chỉ tiến gần và đưa vật cản về phía bộ phận quét của LIDAR (tầng chứa Lazer) cho đến khi map không hiển thị được vật thì ngừng.

- Tiêu chí đánh giá
 - Sử dụng vật không trong suốt như sách, bìa cartong làm vật chắn
 - Khởi động lại LIDAR 5 lần
- Số liệu của tầm quét tối thiểu được đo và cho kết quả là: 15cm

4.1.1.5. Tải trọng

Robot không thiết kế để tải vật trên thân, nên chúng em chỉ kiểm nghiệm khả năng nâng vật tại càng.

Để xác định được tải trọng, chúng em cho Hexapod kẹp vật và giữ trong 5 giây, sau đó tăng dần tải trọng lên, kết quả thu được, robot có thể kẹp một vật nặng 500g, vượt tải trọng này, đầu của robot không còn giữ được vị trí.



Hình 4-3. Tải trọng tối đa mà Hexapod có thể giữ

4.1.1.6. Độ hiệu quả của các chức năng

- Hoạt động các Servo

Hexapod có thể di chuyển ổn định trong những điều kiện nêu trên. Robot có thể tiến, lùi, xoay trái, phải, có thể tự xoay quanh các hệ trục tọa độ, tịnh tiến thân trong không gian. Nhưng còn khá ồn trong di chuyển, tiếng ồn từ khi các servo quay và khi chân chạm đất, do khi thực hiện cho chân đi tới một tọa độ, chân chỉ đi thẳng đến đấy mà không hề giảm tốc khi đến gần.

- Điều khiển bằng PS2

Việc điều khiển không bị gián đoạn, robot hoạt động trơn tru. Nhưng vì chưa xử lý hoàn toàn được vấn đề Deadzone trong tín hiệu analog, nên Hexapod rất dễ bị lệch phương đứng nếu vô tình chạm phải hai Joystick. Chức năng khá nhiều, đòi hỏi người sử dụng phải bỏ nhiều thời gian để có thể làm quen với việc điều khiển.

- Điều khiển bằng phần mềm Android qua Bluetooth

Do phần mềm tạo app còn đơn giản, các chức năng câu lệnh còn hạn chế khiến các thành phần trong giao diện phần mềm không thể đặt đè chồng lên nhau được, làm cho màn hình kết nối mạng bé, khó nhìn. Về điều khiển, phần mềm gửi không nhanh bằng PS2, các Joystick khi kéo còn giật do đòi hỏi phải qua một quá trình xử lý, gây mất thời gian, làm robot bị delay hơn so với PS2.

- LIDAR quét map

Chức năng này hoạt động ổn định, luôn trả về các thông tin về môi trường xung quanh một cách nhanh chóng, giúp người dùng nhận biết được cả vị trí Hexabod và môi trường chỉ thông qua một màn hình máy tính.

- Đi tự động

Chức năng này Hexapod thực hiện vẫn còn chưa được ổn định. Hạn chế lớn nhất trong trường hợp Hexapod đi được là mỗi khi chọn lại một điểm khác, ta bắt buộc phải nhấn lại nút 2D trong phần mềm mới có thể chọn tiếp.

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. KẾT LUẬN

Sau một thời gian nghiên cứu và tìm hiểu, nhóm chúng đã rút ra được:

- Nghiên cứu lí thuyết và tính toán bài toán cho robot sáu chân. Đây là cơ sở quan trọng nhất cho việc điều khiển chuyển động cho robot
- Rút ra được những khó khăn mà con người lấy cảm hứng từ thiên nhiên để xây dựng một con robot có khả năng hoạt động như một loài côn trùng
- Nghiên cứu các loại dáng đi của robot và đưa ra mô hình hình học cho các loại dáng đi. Dựa vào động học thuận của thân và động học nghịch của chân, tính toán được vị trí đặt chân cho robot để có bước đi mượt và tránh bước nhảy
- Thiết kế thành công phần cơ khí robot sáu chân có phần đầu và bụng như loài kiến
- Robot hoạt động đúng bởi tác lệnh từ tay game Play Station, Phần Mềm điều khiển qua điện thoại android, tự động di chuyển trong bản đồ từ LIDAR, gấp một vật thể có trọng lượng nhỏ hơn 500g, có thể trực tiếp xem môi trường bên ngoài thông qua camera
- Robot chưa có thiết kế tối ưu

5.2. HƯỚNG PHÁT TRIỂN ĐỀ TÀI

- Tạo môi trường lập trình thân thiện hơn với học sinh, sinh viên. Người dùng có thể tùy chỉnh dễ dàng các động tác, dáng đi, những module được tích hợp tùy thuộc vào nhu cầu đề ra
- Tạo một giao diện mô phỏng ứng với hoạt động mỗi chân giúp người nghiên cứu dễ dàng nắm bắt được thuật toán
- Hướng tới thiết kế khuôn nhằm giảm khối lượng ban đầu, tăng tải trọng
- Việc điều khiển bằng app còn gập do sử dụng đường truyền Bluetooth, phát triển lên điều khiển bằng wifi hoặc sóng LORA, giúp ta có thể điều khiển Hexapod ở nhiều nơi hơn
- Tích hợp Ai, chuyển đổi ngôn ngữ C thành ngôn ngữ python để robot thông minh hơn. Có khả năng giao tiếp trò chuyện với con người, dạy trẻ em học chữ. Phát hiện người lạ trong nhà và phát ra âm thanh cảnh báo.
- Nghiên cứu những loại động cơ brushless giúp robot có tính bật nhảy cao. Lựa chọn nguồn năng lượng cao, giúp robot hoạt động lâu hơn. Trong hoạt động tìm kiếm cứu nạn động đất, phần đầu và bụng robot sẽ sử dụng những động cơ khỏe

hơn, giúp robot gấp những vật nặng như đá đất, robot lượn lách trong không gian hẹp hơn, phần bụng sẽ mang nước, lương thực cho nạn nhân bị kẹt.

TÀI LIỆU THAM KHẢO

- [1] Dan Thilderkvist and Sebastian Svensson (2015), “Motion Control of Hexapod Robot Using Model-Based Design”, *Printed in Sweden by Media-Tryck*, pp. 17-19, pp. 42.
- [2] Fredrik Persson and Mattias Lindström (2010), “The Memec Hexapod Robot a demonstration platform”, pp. 9.
- [3] NOAA (2013), “LIDAR—Light Detection and Ranging—is a remote sensing method used to examine the surface of the Earth”
- [4] Paul, Richard (1981), “*Robot manipulators: mathematics, programming, and control : the computer control of robot manipulators*”. MIT Press, Cambridge, MA. ISBN 978-0-262-16082-7.
- [5] Sunil93 (2013), “*Interfacing PS2 controller with AVR -Bit Bang*”, pp. 4.
- [6] Nguyễn Văn Hân (2017), “Giao thức truyền dữ liệu nối tiếp”, trang 4
- [7] Trần Quốc Hùng (2012), “Giáo trình Dung sai - Kỹ thuật đo”, ”, *nhà xuất bản ĐHQG TP HCM*.
- [8] Tăng Quang Khải và Nguyễn Tuấn Anh (2014), “Tìm hiểu giao diện SPI”, Hà Nội, trang 8-9.
- [9] PGS.TS. Nguyễn Trường Thịnh (2014), “Giáo trình kỹ thuật robot”, *nhà xuất bản ĐHQG TP HCM*.
- [10] Canberk Suat Gurel, “A project log for Hexapod Modelling, Path Planning and Control”, [https://hackaday.io/project/ 29/06/2017](https://hackaday.io/project/290620-hexapod-modelling-path-planning-and-control). [Internet]. [10/07/2019]
- [11] <http://www.slamtec.com/en/lidar/a1>. [Internet]. [xem 10/07/2019]
- [12] <https://store.arduino.cc/usa/mega-2560-r3>. [Internet]. [xem 10/07/2019]