

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA



BÁO CÁO ĐỒ ÁN 1

**ĐỀ TÀI: TÌM HIỂU VỀ CẤU TRÚC VÀ CHUYỂN ĐỘNG CỦA
ROBOT DOG**

**(ANALYSIS OF THE MECHANICAL STRUCTURE AND
LOCOMOTION OF A DOG-LIKED QUADRUPED ROBOT)**

Giảng viên hướng dẫn: TS. Nguyễn Lê Dũng

Sinh viên thực hiện	MSSV
Trương Tuấn An	2210041
Võ Quế Long	2211910

Thành phố Hồ Chí Minh – 2025

PHÂN CÔNG NHIỆM VỤ

Chủ đề	Thành viên thực hiện	
	Trương Tuấn An	Võ Quế Long
Tổng quan lí thuyết	X	X
Chuẩn bị phần cứng	X	
Chuẩn bị phần mềm		X
Bài toán chuyển động		X
Bài toán tránh vật cản	X	
Tiến độ	100%	100%

MỤC LỤC

I. GIỚI THIỆU	1
1.1. Lý do chọn đề tài.....	1
1.2. Giới hạn của đề tài.....	1
II. TỔNG QUAN LÝ THUYẾT.....	2
2.1. Các loại robot di chuyển bằng chân.....	2
2.2. Phân loại cấu trúc chân robot	3
2.3. Số bậc tự do mỗi chân	5
2.4. Các kiểu bước đi	7
2.5. Động học & động lực học trong điều khiển.....	9
2.6. Bộ điều khiển.....	11
III. CHUẨN BỊ	14
3.1. Phần cứng	14
3.2. Phần mềm.....	21
IV. BÀI TOÁN CHUYỂN ĐỘNG	24
4.1. Tổng quan & mục tiêu	24
4.2. Điều khiển chuyển động của khớp sử dụng DC Motor.....	24
4.3. Điều khiển chuyển động của một chân robot.....	32
V. BÀI TOÁN TRÁNH VẬT CẢN	34
5.1. Tổng quan & mục tiêu	34
5.2. Xử lý dữ liệu cảm biến	34
5.3. Thuật toán tránh vật cơ bản.....	35
KẾT LUẬN	36
TÀI LIỆU THAM KHẢO	37

DANH MỤC TỪ VIẾT TẮT

Thuật ngữ	Viết tắt
Forward/Direct Kinematics	FK
Forward/Direct Dynamics	FD
Degree of freedom	DOF
Serial-Leg	S-L
Parallel-Leg	P-L
Hybrid-Leg	H-L
Homogeneous Transformation Matrix	HTM

I. GIỚI THIỆU

1.1. Lý do chọn đề tài

Trong những năm gần đây, Quadruped Robot đã thu hút sự quan tâm lớn trong nghiên cứu nhờ khả năng di chuyển linh hoạt, thích ứng với nhiều địa hình và tiềm năng ứng dụng rộng rãi trong công nghiệp, quân sự và đời sống. Đặc biệt, sự ra đời của robot Mini Cheetah năm 2018 đánh dấu bước tiến quan trọng trong thiết kế phần cứng và thuật toán điều khiển robot di động.

Hiện nay, robot bốn chân được sử dụng trong nhiều lĩnh vực: trinh sát, cứu hộ, vận chuyển hàng hóa trong quân sự; kiểm tra hạ tầng, làm việc trong môi trường nguy hiểm trong công nghiệp; hỗ trợ y tế, chăm sóc người khuyết tật và tương tác thông minh trong đời sống. Các nghiên cứu hiện tại tập trung vào tối ưu hóa phần cứng và phát triển thuật toán điều khiển giúp robot di chuyển mượt mà, thăng bằng tốt và vận hành ổn định trong điều kiện phức tạp.

Xuất phát từ mong muốn hiểu rõ cách thức vận hành của các hệ thống điều khiển hiện đại, nhóm đã chọn nghiên cứu và phát triển robot bốn chân. Đây là cơ hội để áp dụng kiến thức lý thuyết vào thực tế, đồng thời khám phá các giải pháp tối ưu trong lĩnh vực robot di động.

1.2. Giới hạn của đề tài

Lấy cảm hứng từ loài chó, nhóm đã phát triển đề tài “Dog Robot” nhằm mô phỏng khả năng di chuyển linh hoạt và thích ứng địa hình. Dù không thể leo tường hay bay, robot được tối ưu hóa để di chuyển hiệu quả, vượt qua địa hình phức tạp và hoạt động trong môi trường nguy hiểm như khu vực nhiễm phóng xạ, không gian chật hẹp hoặc địa hình gồ ghề. Với thiết kế phù hợp yêu cầu thực tế, Dog Robot hứa hẹn là công cụ hỗ trợ hiệu quả và mở ra nhiều ứng dụng tiềm năng.

II. TỔNG QUAN LÝ THUYẾT

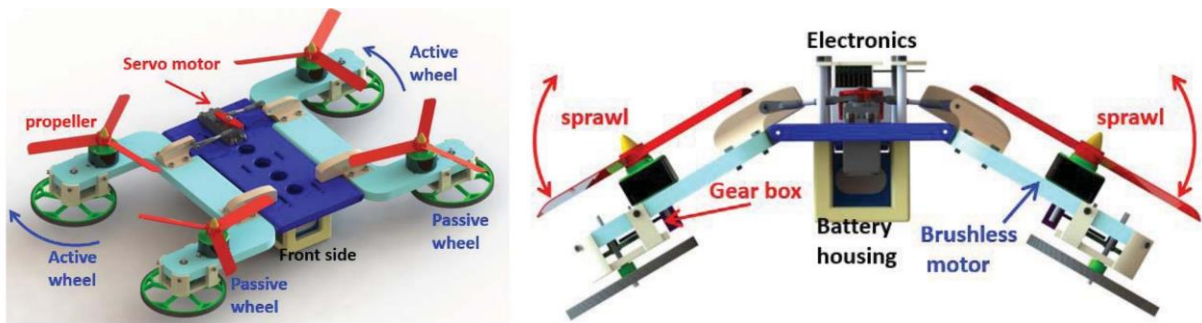
2.1. Các loại robot di chuyển bằng chân

2.1.1. Di chuyển bằng chân kết hợp

Là các robot sử dụng chân cùng với một hoặc nhiều phương thức di chuyển khác như bánh xe, bánh xích hoặc cơ cấu bay (cánh quạt), nhằm tăng cường khả năng thích nghi với nhiều loại địa hình khác nhau. Tuy nhiên, việc tích hợp nhiều cơ cấu chuyển động trên cùng một hệ thống đòi hỏi thiết kế cơ khí tinh vi và thuật toán điều khiển đa chế độ, dẫn đến độ phức tạp cao về cả phần cứng lẫn phần mềm.

Ví dụ:

- ANYmal C: robot bốn chân tích hợp bánh xe
- FSTAR: robot bốn chân tích hợp bánh xe và cánh quạt (Hình 2.1)



Hình 2.1. The mechanical design of the FSTAR robot

- Ngoài ra, để tránh công kênh về mặt cơ khí thì các robot côn trùng thường được kết hợp nhiều hơn một phương thức di chuyển.

Trong khuôn khổ đề án, nhóm tập trung nghiên cứu robot chỉ sử dụng chân để tạo ra chuyển động.

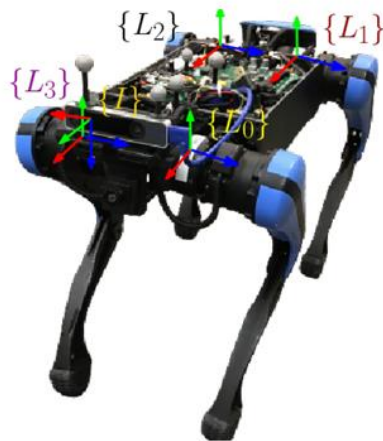
2.1.2. Di chuyển bằng chân

Có nhiều loại robot chỉ sử dụng chân để di chuyển, chúng được mô phỏng theo chuyển động của các loài động vật trên cạn như báo, chó, nhện, hoặc con người (Hình 2.2).



Hình 2.2. Atlas Robot from Boston Dynamics

Với mục tiêu đề án là mô phỏng chuyển động của loài chó, nhóm tập trung nghiên cứu các kiểu chuyển động của Quadruped Robot đã được nghiên cứu rất nhiều trong những năm gần đây. Bên dưới là hình ảnh một trong những dạng Quadruped Robot phổ biến nhất với khớp nối chân kiểu Serial-Leg:



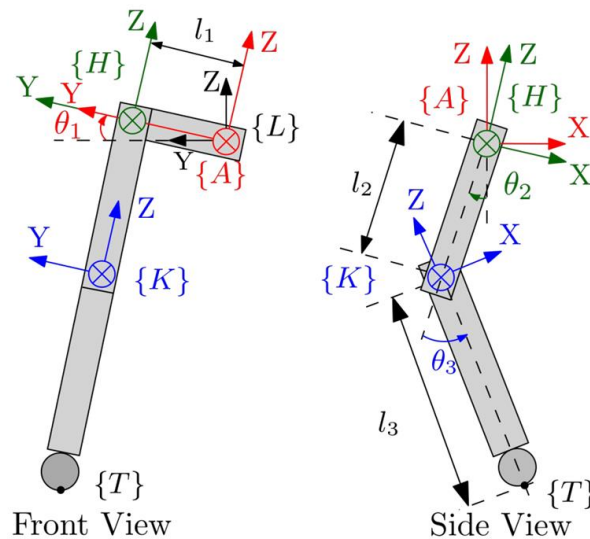
Hình 2.3. Quadruped Robot with Serial-Leg

2.2. Phân loại cấu trúc chân robot

Cấu trúc cơ khí của chân robot có ảnh hưởng lớn đến khả năng điều khiển, hiệu suất chuyển động, độ chính xác cũng như khả năng mở rộng trong thiết kế. Dựa trên nguyên lý cơ học, cấu trúc chân robot có thể được chia thành ba loại chính: Serial, Parallel và Hybrid.

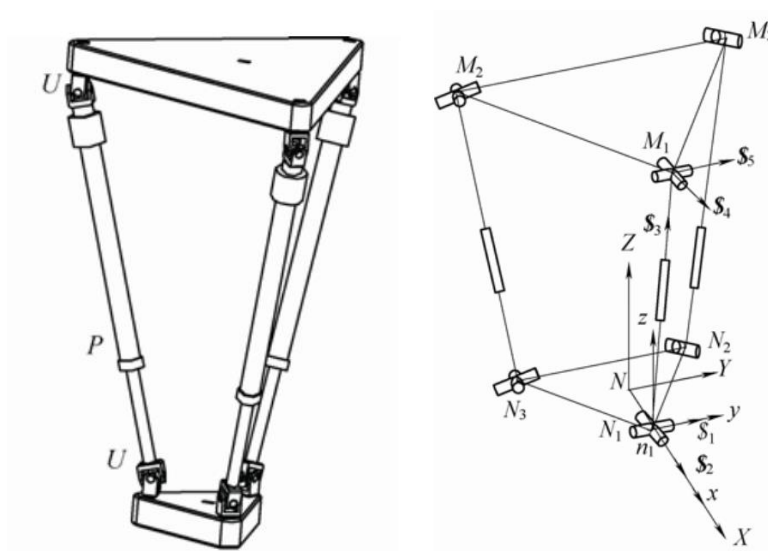
Serial: Là cấu trúc phổ biến nhất, trong đó các khớp (joint) được liên kết tuần tự từ gốc (base) – thường là khung thân robot – đến điểm cuối, thường là bàn chân (toe). Mỗi khớp điều khiển một DOF và ảnh hưởng đến toàn bộ các khâu phía sau trong chuỗi động

học. Hình bên dưới là một trong các cấu trúc chân nối tiếp với 3 motor được gắn lần lượt tại shoulder – hip – knee:



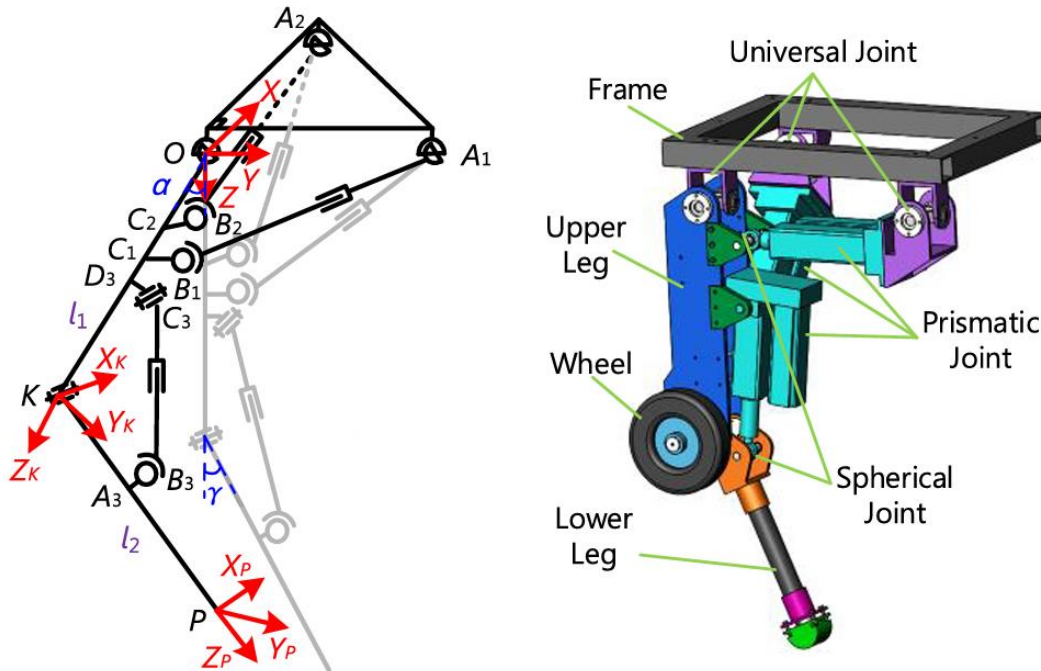
Hình 2.4. Both side of each Serial-Leg

Parallel: Trong cấu trúc song song, nhiều nhánh cơ cấu hoạt động đồng thời để điều khiển một điểm cuối duy nhất (toe), tạo nên nên hệ thống Kinematics khép kín. Loại này thường có cấu hình tương tự như Stewart platform (một dạng hexapod có khung đồng bộ với các chân), với độ cứng cao và độ chính xác chuyển động tốt hơn so với cấu trúc nối tiếp. Tuy nhiên cấu trúc này khá nặng về mặt cơ khí và thuật toán FK – IK phức tạp hơn so với Serial-Leg. Hình bên dưới là một trong những cấu trúc P-L phổ biến:



Hình 2.5. Mechanical model & branch coordinate system

Serial-Parallel Hybrid: Kết hợp ưu điểm của cả hai dạng trên, sử dụng Kinematics của S-L để điều khiển chính, kết hợp cơ cấu P-L để hỗ trợ truyền động (thường là dùng cấu trúc song song để điều khiển motor ở knee joint). Vì kết hợp cả hai cấu trúc S-L và P-L nên phần chân rất nặng về mặt cơ khí, thuật toán chuyển động cũng phức tạp hơn nhiều. Hình bên dưới là một trong những cấu trúc H-L đơn giản nhất:



Mechanical model & branch coordinate system

Sau cùng, nhóm quyết định chọn sử dụng cấu trúc chân không quá nặng về mặt cơ khí để tập trung vào phần điều khiển cũng như giúp dễ trong việc mô phỏng về sau là S-L.

2.3. Số bậc tự do mỗi chân

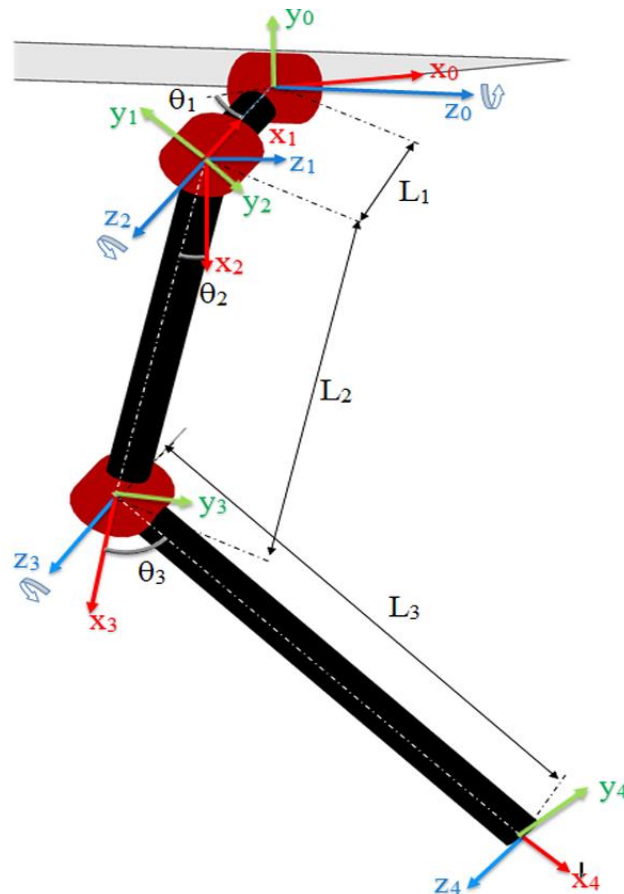
Thường sử dụng 2 – 3 joint tương ứng với 2 – 3 DOF cho mỗi chân của robot, dưới đây nhóm tiến hành phân tích ưu – nhược điểm của mỗi trường hợp.

2 DOF: Hai khớp được gắn motor lần lượt là hip joint và knee joint. Ưu điểm là điều khiển đơn giản, phương trình động học không quá phức tạp như ba bậc tự do. Tuy nhiên vì chuyển động quá đơn giản dẫn đến sự linh hoạt giảm, tốc độ di chuyển không cao, và vì không có shoulder joint nên việc thay đổi hướng di chuyển diễn ra chậm chạp (mặc dù 2 khớp mỗi chân vẫn có thể xoay góc).



Hình 2.6. Both side of an 2 DOF S-L

3 DOF: Nhóm đã chọn phương án này trong mô hình chuyển động. Ba bậc tự do tương ứng ở 3 khớp là shoulder joint, hip joint và knee joint giúp cho việc chuyển động trở nên mượt hơn, xoay góc nhanh hơn, di chuyển nhanh hơn và vượt chướng ngại vật tốt hơn so với chỉ dùng 2 DOF. Tuy nhiên việc dùng nhiều hơn mỗi chân 1 motor (tương ứng nhiều hơn 4 motor về toàn bộ hệ thống) khiến robot trở nên nặng nề, cần phải lưu ý điều này trong việc lựa chọn công suất của motor và nguồn để đáp ứng phần cơ khí và độ nặng toàn bộ robot.



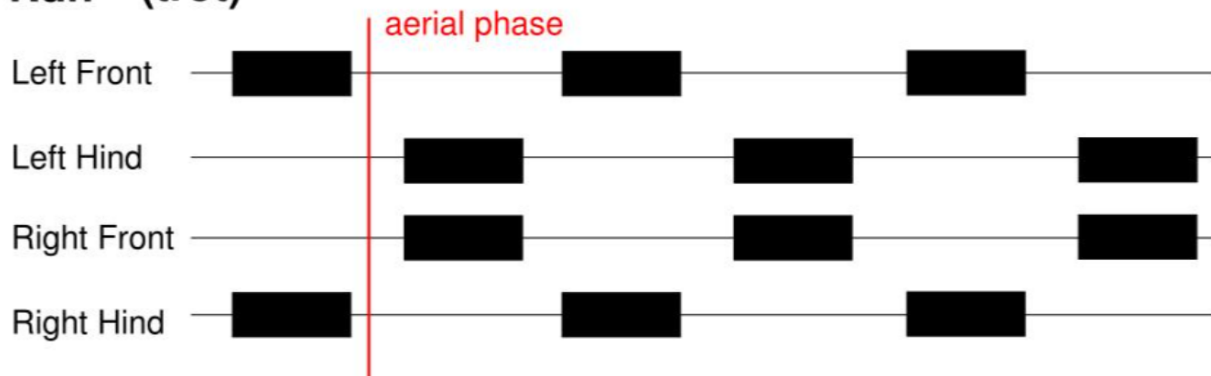
Hình 2.7. The Coordinate System of a 3 DOF S-L

2.4. Các kiểu bước đi

Quadruped Robot có nhiều kiểu di chuyển khác nhau nhưng thường thấy nhất là 5 gait được mô tả bên dưới bao gồm: Trot, Walk, Pace, Baund và Gallop.

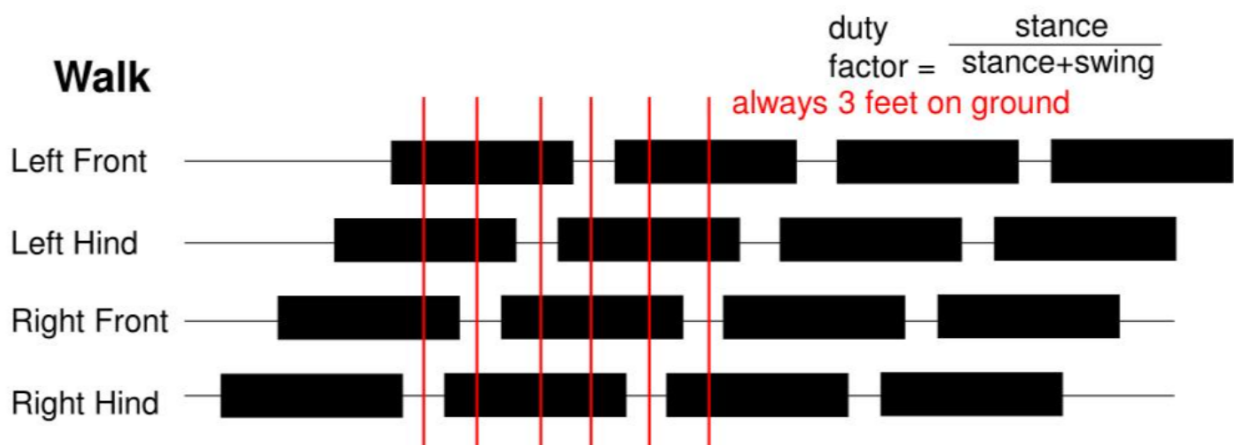
Trot gait (Một dạng của Run gait): Phổ biến nhất, nhóm chọn kiểu chuyển động này để thực hiện đồ án. Hai chân chéo nhau nhấc cùng lúc khiến robot di chuyển nhanh, tuy nhiên cần phải thực hiện bài toán cân bằng khá phức tạp.

Run (trot)



Hình 2.8. Trot bait table

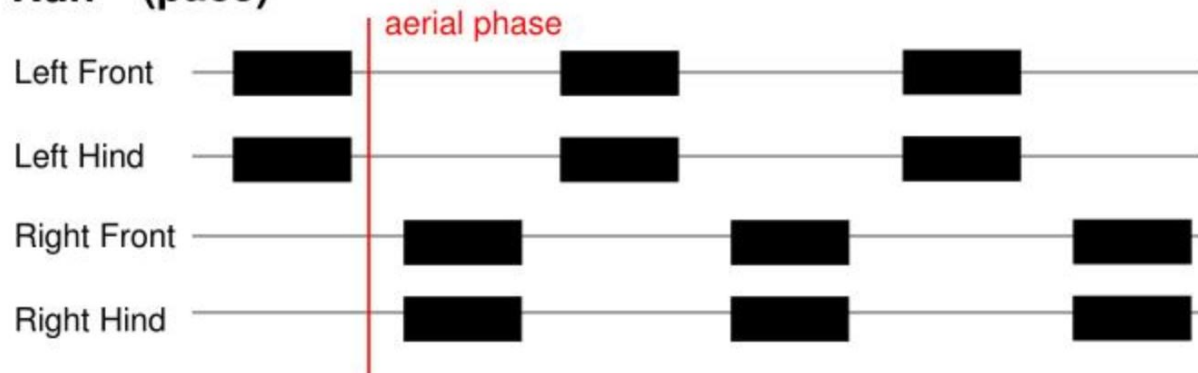
Walk gait: Mỗi lần nhấc 1 chân, 3 chân còn lại tiếp đất. Thường chọn gait table có 4 phase offset (0, 90°, 180°, 270°).



Walk gait table

Pace gait: Di chuyển hai chân cùng bên cùng lúc. Đặc điểm của kiểu di chuyển này là CoM (trọng tâm) lắc ngang lúc di chuyển nên cần giữ thăng bằng theo trục Y.

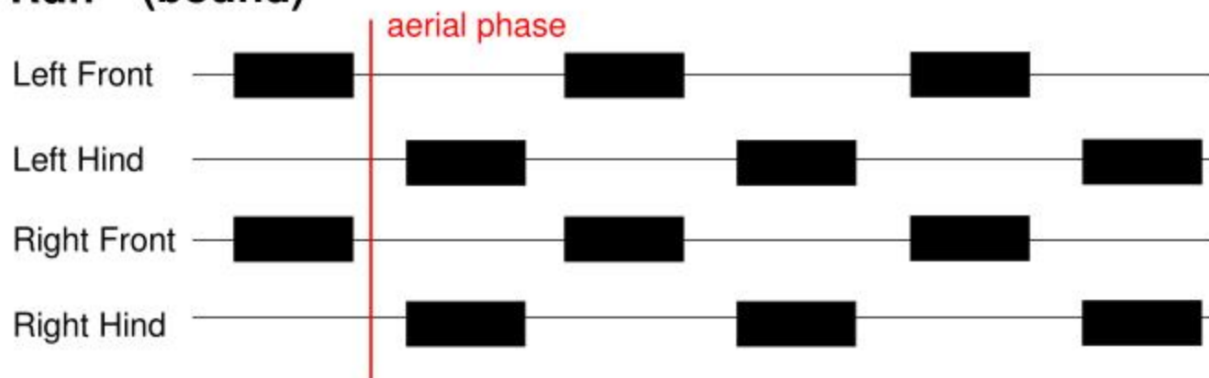
Run (pace)



Hình 2.9. Pace gait table

Bound gait: Hai chân trước nhấc lên trước cùng lúc, sau đó hai chân sau nhấc lên trước cùng lúc tạo ra di chuyển như nhảy bậc về phía trước.

Run (bound)



Hình 2.10. Bound gait table

Gallop gait: gait nhanh nhất đối với Quadruped Robot, có phase không chân nào tiếp đất.

Run (gallop)



Hình 2.11. Gallop gait table

2.5. Động học & động lực học trong điều khiển

Mục tiêu cuối cùng là sử dụng mô hình động lực học để điều khiển chuyển động của robot. Tuy nhiên, trong Đồ án 1 và cho đến khi hoàn thành việc lắp ráp cơ khí nhóm sẽ thực hiện mô phỏng và lập trình sử dụng mô hình động học (thuận/nghịch) cho các chân của robot. Lí do áp dụng mô hình động học là vì mô hình ít phức tạp hơn, tập trung điều khiển góc và vị trí để kiểm nghiệm phần điện – lập trình – cơ khí cho đến khi các cơ cấu hoạt động đủ tốt thì nhóm sẽ tiến hành chuyển từ Kinematics sang Dynamics để điều khiển thêm gia tốc và moment.

Mô hình FK-IK và FD-ID của mỗi chân robot được tham khảo từ một số bài báo khoa học (*TLTK [6], [7]*) có cấu trúc chân và chuyển động tương tự như cấu trúc chân và chuyển động mà nhóm đã chọn là Serial-Leg và Trot gait.

Trong phạm vi Đồ án 1, nhóm chỉ phân tích và xét đến mô hình Kinematics (*tương ứng Hình 2.7*):

- **Forward Kinematics:** Mục tiêu là xác định vị trí trong không gian của các joint trong không gian. Dựa vào các HTM ta có thể suy ra được tọa độ của joint tương ứng trong không gian, cụ thể tọa độ được xác định là vector tịnh tiến với ba phần tử đầu tiên của cột thứ tư của HTM tương ứng (ở đây ta chỉ quan tâm tọa độ của Toe, còn các thành phần khác là bổ sung để suy ra được tọa độ này). Bên dưới là các HTM tương ứng theo thứ tự nối tiếp với nhau và HTM của đỉnh (Toe) so với tọa độ gốc (Base).

Shoulder-Base joint:

$$T_0^1 = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 & -L_1 \cos(\theta_1) \\ \sin(\theta_1) & \cos(\theta_1) & 0 & -L_1 \sin(\theta_1) \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Hip-Shoudler joint:

$$T_1^2 = \begin{bmatrix} 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Knee-Hip joint:

$$T_2^3 = \begin{bmatrix} \cos(\theta_2) & -\sin(\theta_2) & 0 & L_2 \cos(\theta_2) \\ \sin(\theta_2) & \cos(\theta_2) & 0 & L_2 \sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Toe-Knee joint:

$$T_3^4 = \begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 & L_3 \cos(\theta_3) \\ \sin(\theta_3) & \cos(\theta_3) & 0 & L_3 \sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Toe-Base joint:

$$T_0^4 = T_0^1 T_1^2 T_2^3 T_3^4 = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}$$

Với các thành phần của ma trận KF ($m_{ij}, i = \overline{1,4}, j = \overline{1,4}$) như sau:

$m_{11} = \cos(\theta_2) \cos(\theta_3) \sin(\theta_1) - \sin(\theta_1) \sin(\theta_2) \sin(\theta_3)$
$m_{12} = -\cos(\theta_2) \sin(\theta_1) \sin(\theta_3) - \cos(\theta_3) \sin(\theta_1) \sin(\theta_2)$
$m_{13} = -\cos(\theta_1)$
$m_{14} = L_2 \cos(\theta_2) \sin(\theta_1) - L_1 \cos(\theta_1)$ $+ L_2 \cos(\theta_2) \cos(\theta_3) \sin(\theta_1)$ $- L_3 \sin(\theta_1) \sin(\theta_2) \sin(\theta_3)$
$m_{21} = \cos(\theta_1) \sin(\theta_2) \sin(\theta_3) - \cos(\theta_1) \cos(\theta_2) \cos(\theta_3)$
$m_{22} = \cos(\theta_1) \cos(\theta_2) \sin(\theta_3) + \cos(\theta_1) \cos(\theta_3) \sin(\theta_2)$
$m_{23} = -\sin(\theta_1)$
$m_{24} = L_3 \cos(\theta_1) \sin(\theta_2) \sin(\theta_3)$ $- L_2 \cos(\theta_1) \cos(\theta_2)$ $- L_3 \cos(\theta_1) \cos(\theta_2) \cos(\theta_3) - L_1 \sin(\theta_1)$
$m_{31} = \cos(\theta_2) \sin(\theta_3) + \cos(\theta_3) \sin(\theta_2)$
$m_{32} = \cos(\theta_2) \cos(\theta_3) - \sin(\theta_2) \sin(\theta_3)$
$m_{33} = 0$
$m_{34} = L_2 \sin(\theta_2) + L_3 \cos(\theta_2) \sin(\theta_3) + L_3 \cos(\theta_3) \sin(\theta_2)$
$m_{41} = 0$
$m_{42} = 0$
$m_{43} = 0$
$m_{44} = 1$

- Inverse Kinematics: Mục tiêu từ vị trí mong muốn suy ngược ra góc quay tương ứng ở mỗi khớp.

Shoulder joint:

$$\theta_1 = -atan2(-y_4, x_4) - atan2(\sqrt{x_4^2 + y_4^2 - L_1^2}, -L_1)$$

Hip joint:

$$\theta_2 = atan2\left(z_4, \sqrt{x_4^2 + y_4^2 - L_1^2}\right) - atan2(L_3 \sin(\theta_3), L_2 + L_3 \cos(\theta_3))$$

Knee joint:

$$\theta_3 = atan2(-\sqrt{1 - D^2}, D) \quad (\text{Legs for 1 and 3})$$

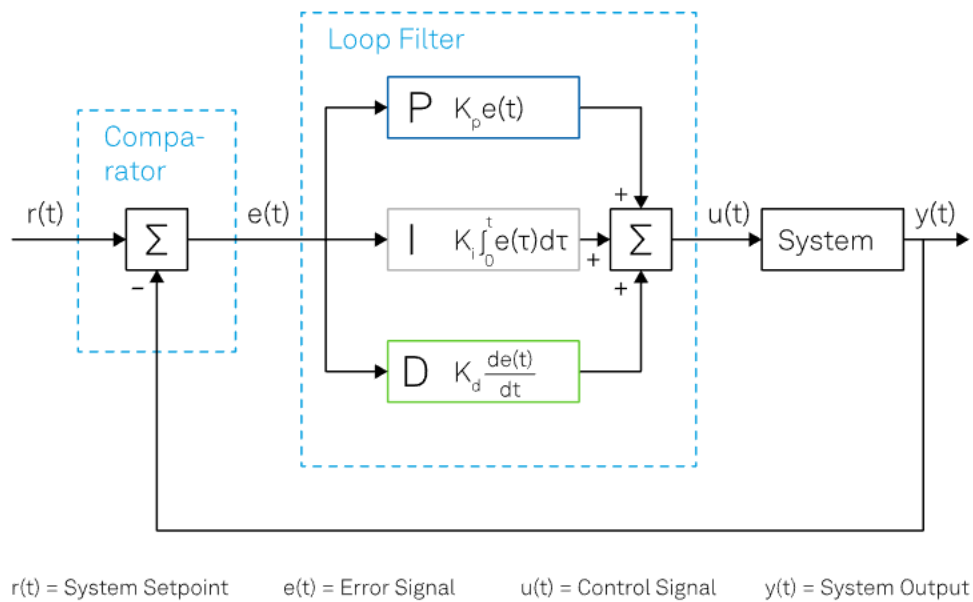
$$\theta_3 = atan2(\sqrt{1 - D^2}, D) \quad (\text{Legs for 2 and 4})$$

$$D = (x_4^2 + y_4^2 - L_1^2 + z_4^2 - L_2^2 - L_3^2) / (2 L_2 L_3)$$

2.6. Bộ điều khiển

Sử dụng một trong ba bộ điều khiển kinh điển: PID, Fuzzy hoặc SMC.

PID: Dành cho độ phức tạp thấp trong môi trường bằng phẳng, ít nhiễu. Sơ đồ khối của hệ thống sử dụng bộ điều khiển PID đơn giản như sau:

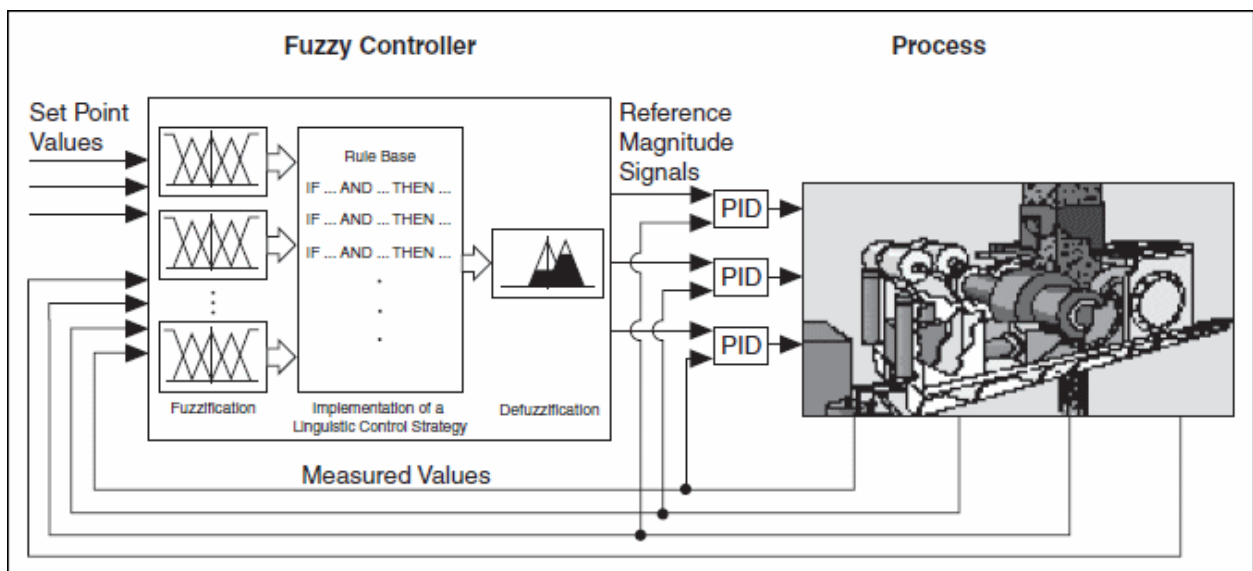


Hình 2.12. PID Block Diagram

Với:

- Input: sai số giữa giá trị đặt và giá trị hồi tiếp. Ở đây tùy giá trị mà ta quan tâm có thể là góc quay các joint hoặc vị trí điểm cuối. Tùy vào phương thức điều khiển là FK hay IK.
- Output: moment xoắn (torque) được tạo ra bởi áp khuếch đại thông qua driver tương ứng với DC Motor đang sử dụng bằng cách điều khiển dòng, PWM hoặc duty cycle.

Fuzzy: Dành cho địa hình gồ ghề hơn và có nhiều ít. Sơ đồ khối của hệ thống sử dụng bộ điều khiển PID Fuzzy như sau:



Hình 2.13. Fuzzy Block Diagram

Với:

- Input: Sai số và biến thiên sai số giữa giá trị đặt so với giá trị hồi tiếp. Lựa chọn giá trị là góc quay các joint hoặc vị trí điểm cuối tương tự bộ điều khiển PID thông thường.
- Output: torque tương tự như bộ điều khiển PID thông thường.
- Hệ mờ: Lựa chọn hệ mờ có luật phù hợp, với số luật lớn hơn 25 luật mỗi input để phù hợp với 3 giá trị đầu vào.

SMC: Dành cho địa hình phức tạp, nhiều nhiễu. Sơ đồ khối của hệ thống sử dụng bộ điều khiển trượt cần được xây dựng phức tạp hơn qua việc giải tìm mặt trượt và chọn các hệ số trượt nên ở đây sẽ không trình bày cụ thể. Tuy nhiên input và output có thể chọn giống như cách chọn của bộ điều khiển PID.

Hiện tại nhóm chưa chính thức chọn bộ điều khiển nào cho robot nhưng sẽ thử mô phỏng bộ điều khiển thông minh Fuzzy để điều khiển vị trí động cơ (chi tiết ở phần “*Bài toán chuyển động*”).

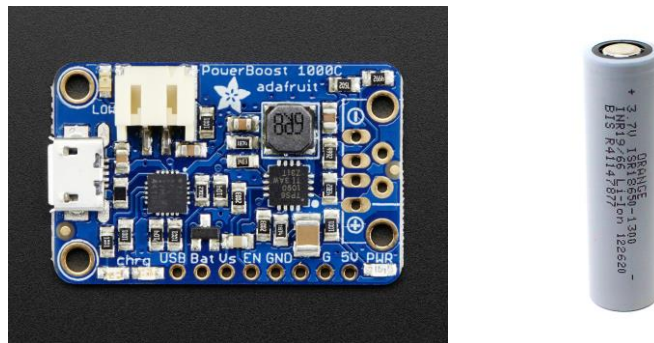
III. CHUẨN BỊ

3.1. Phần cứng

Trình bày tổng quan về các linh kiện chính được sử dụng để lắp ráp và vận hành robot. Bao gồm các phần cung cấp năng lượng, điều khiển chuyển động, cảm biến và bộ xử lý. Các linh kiện này giúp robot hoạt động ổn định và tương tác với môi trường.

3.1.1. Adafruit PowerBoost 1000C & 18650 Li-Ion Battery

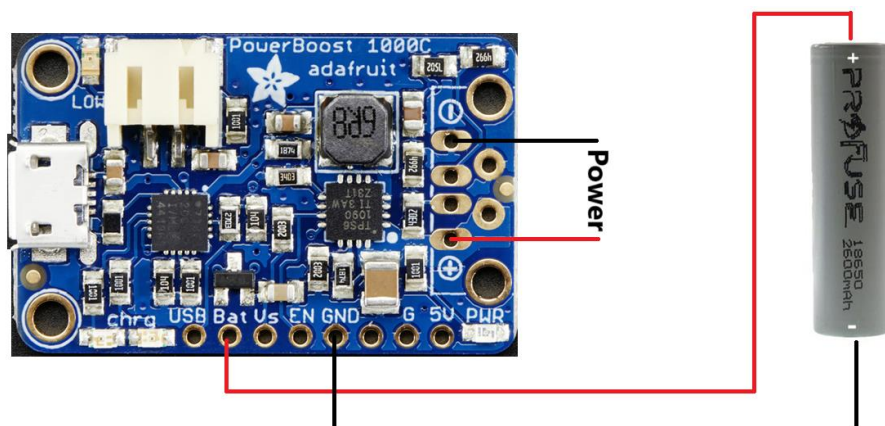
Cần 1 Adafruit PowerBoost 1000C và 1 18650 Li-Ion Battery



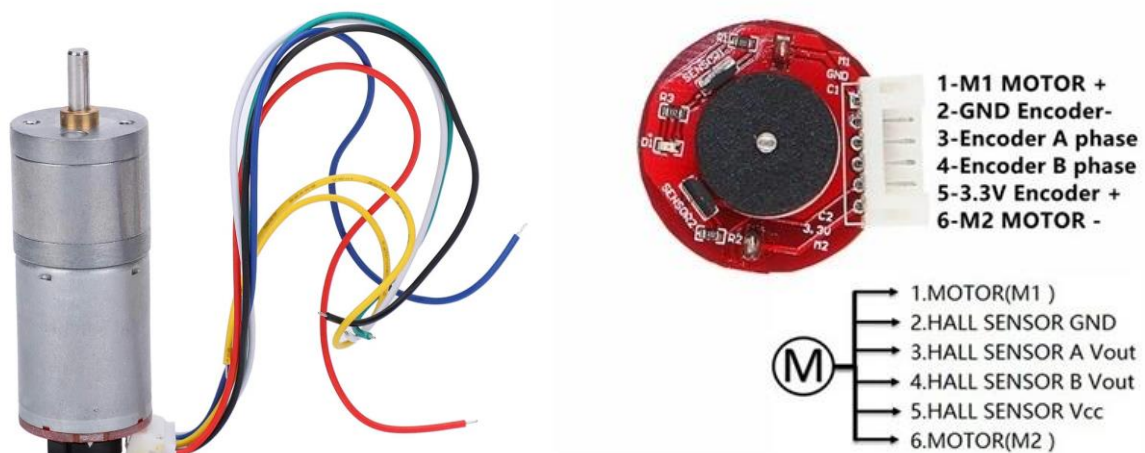
Chức năng: Cấp nguồn 5V (dòng tối đa 1A) liên tục, ổn định. Vừa sạc, vừa cấp nguồn thông qua USB hoặc Li-Ion Battery (18560 Battery)

Nguyên lý hoạt động: Khi không cắm sạc, pin 3.7V sẽ được tăng áp lên 5.2V để cấp điện cho thiết bị. Khi cắm sạc qua micro-USB, nguồn 5V sẽ vừa sạc pin vừa cấp điện cho thiết bị nhờ mạch chia tải. Hệ thống tự động chuyển đổi giữa nguồn USB và pin, đảm bảo nguồn ra luôn ổn định, không gián đoạn.

Đấu dây: Nguồn sạc sử dụng Li-Ion Battery. Nguồn cấp được dùng cho toàn bộ các thành phần khác

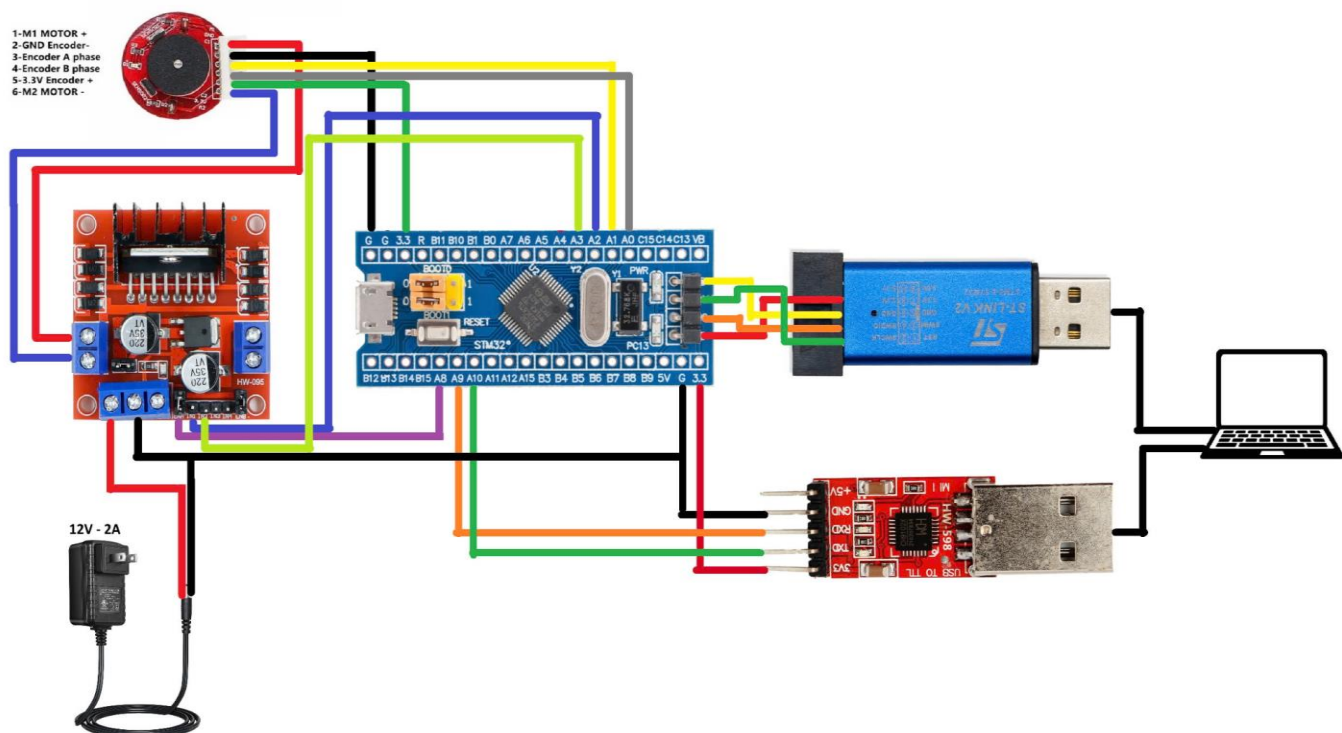


3.1.2. DC Motor: Cần 12 Motor, mỗi ba motor tương ứng một chân của robot.

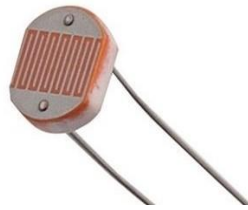


Chức năng: Điều khiển vị trí mỗi khớp robot.

Đầu dây: Thực hiện đấu nối 1 motor để làm quen với các bài toán điều khiển tốc độ và vị trí.

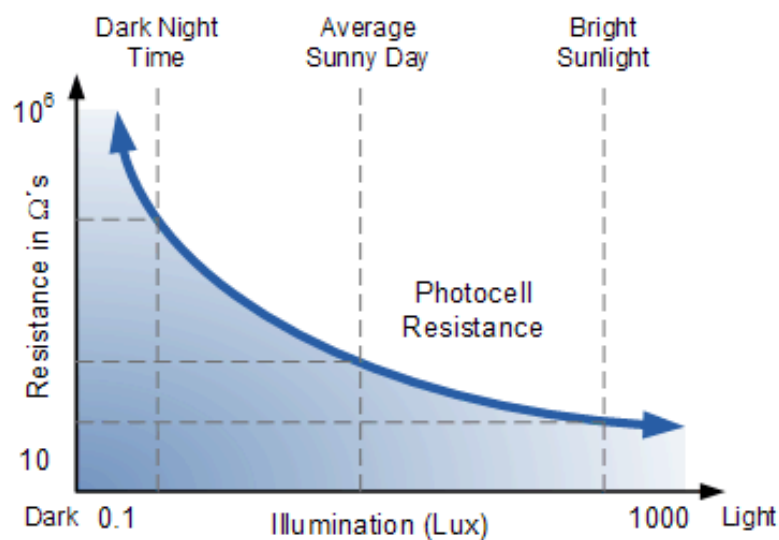
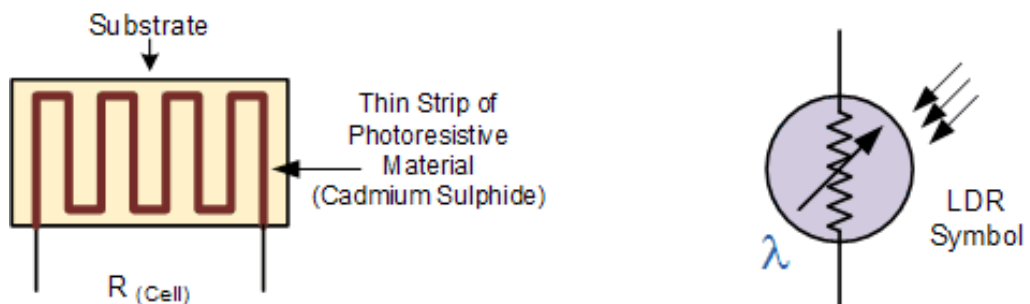


3.1.3. LDR



LDR (Light - Dependent Resistor) là một thiết bị điện tử được sử dụng cho các ứng dụng cảm biến cường độ ánh sáng. LDR có điện trở tương đối thấp trong ánh sáng và khi môi trường xung quanh tối hơn, điện trở của LDR tăng đáng kể.

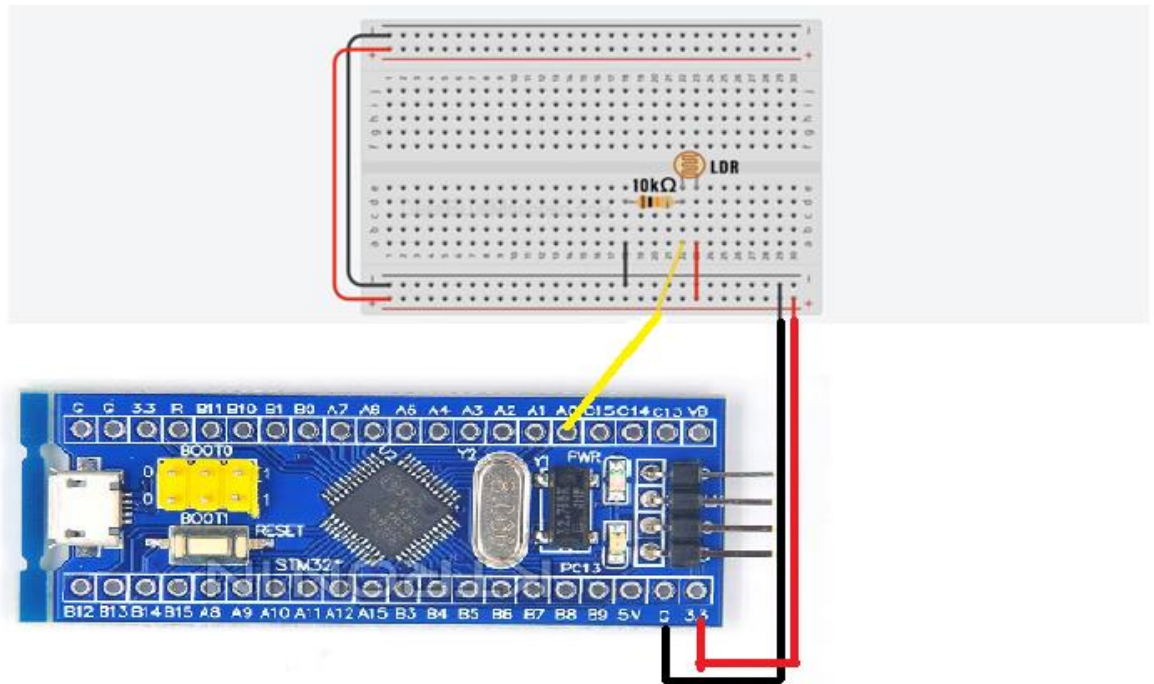
LDR có thể có điện trở là $2k\Omega$ vào ban ngày, $4k\Omega$ trong ánh sáng phòng và lên tới $200k\Omega$ - $1M\Omega$ trong bóng tối.



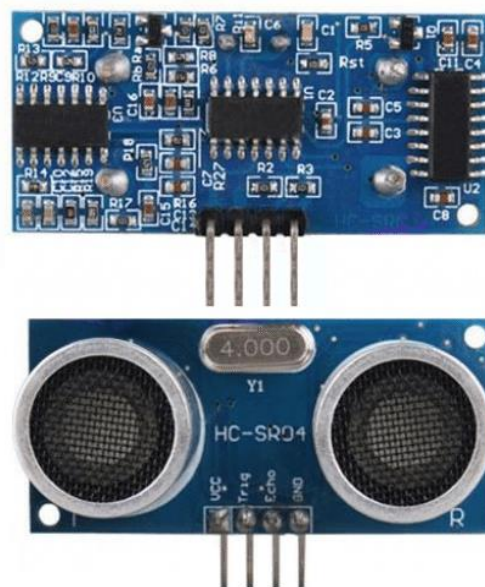
Chức năng: Phát hiện ánh sáng để di chuyển về phía nguồn sáng.

Nguyên lý hoạt động: Khi ánh sáng chiếu vào bề mặt LDR, điện trở của nó giảm, cho phép dòng điện chạy qua mạnh hơn. Ngược lại, khi ánh sáng yếu hoặc tối, điện trở của LDR tăng lên, hạn chế dòng điện. Điều này giúp vi điều khiển đọc giá trị điện áp thay đổi và suy ra mức độ sáng của môi trường.

Đầu dây:



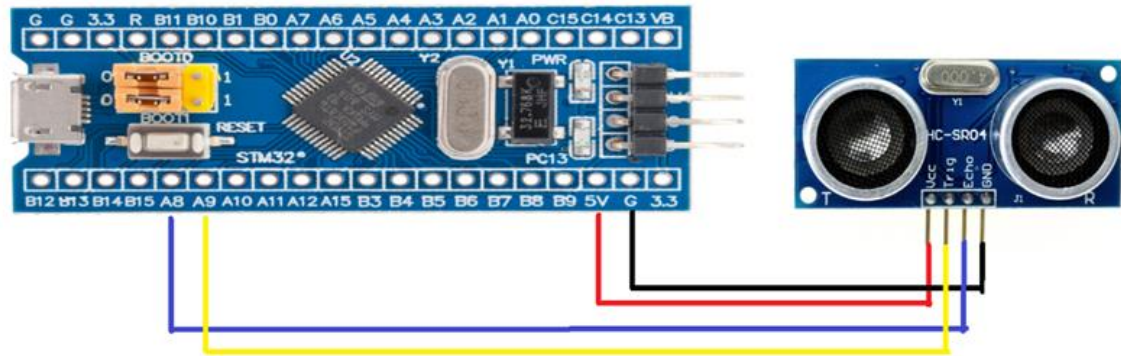
3.1.4. HC-SR04



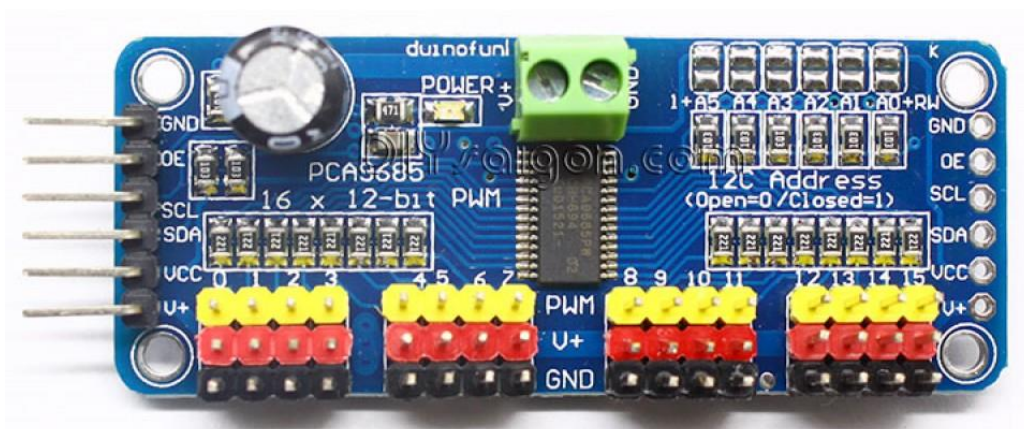
Chức năng: Đo khoảng cách.

Nguyên lý hoạt động: Phát ra một xung âm thanh, chờ đợi âm phản xạ quay trở lại và nếu đo đúng thời gian, bạn có thể xác định được có vật thể nào phía trước hay không và khoảng cách đến vật thể đó là

Đầu dây:



3.1.5. PCA9685



Input Voltage: 2.3V to 5.5V (typically powered by 3.3V or 5V)

Number of Channels: 16 channels

PWM Resolution: 12-bit (4096 steps)

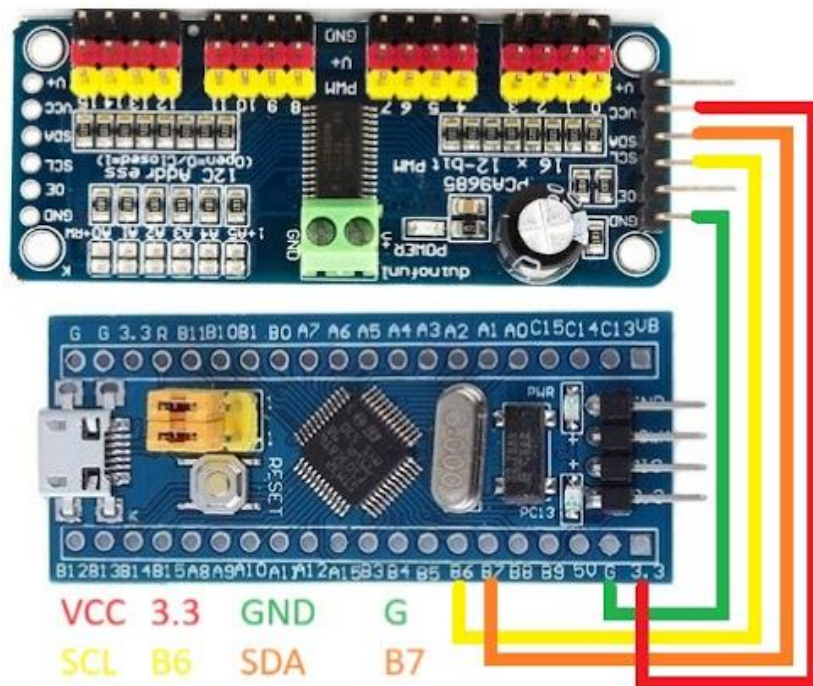
Output Frequency: Programmable from approximately 24Hz to 1526Hz

Control Interface: I2C (Inter-Integrated Circuit)

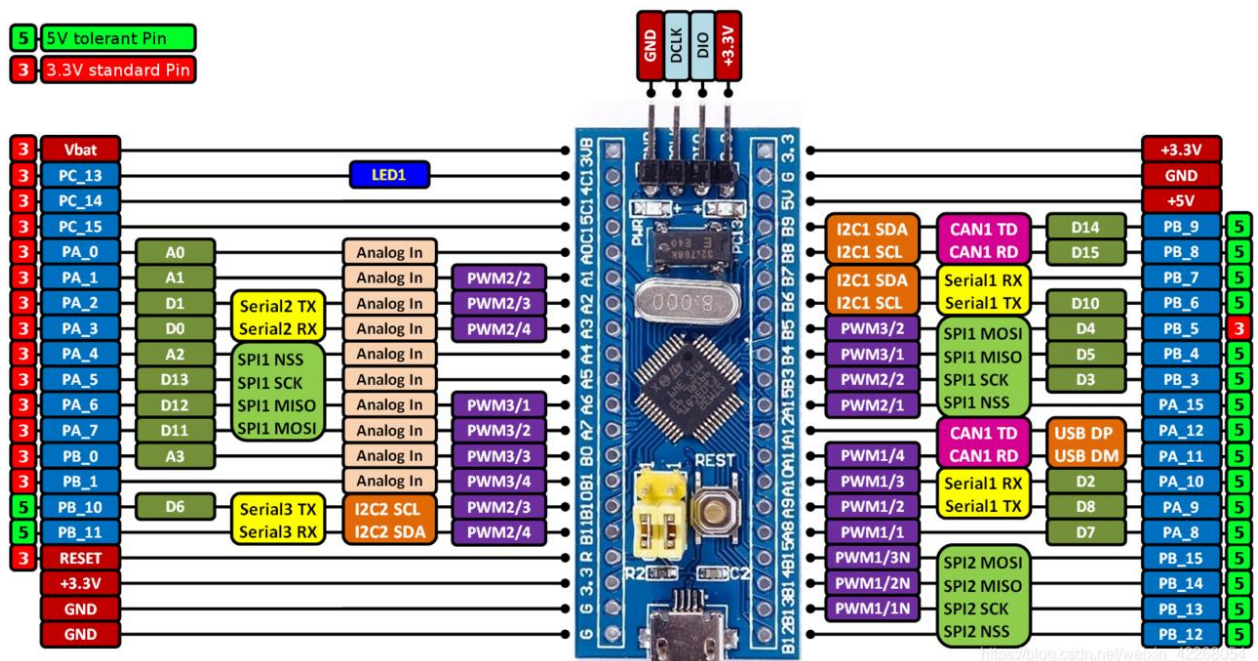
Chức năng: Có thể điều khiển dễ dàng 16 động cơ servo thông qua trình điều khiển

Nguyên lý hoạt động: Điều khiển 16 động cơ servo bằng cách sử dụng chân SCL và SDA giao tiếp I²C

Đầu dây:



3.1.6. KIT STM32 Blue Pill



GPIO	ADC	Timer	USART	SPI	I2C
37	10-bit (12 channels)	3 timer 16-bit & 1 timer 32-bit	3	2	2

3.1.7. Động cơ BLDC: Động cơ DC không chổi than, còn được gọi là động cơ chuyển mạch điện tử, là một loại động cơ đồng bộ hoạt động bằng nguồn điện một chiều.



Chức năng: Mỗi chân robot chó thường có 2 đến 3 bậc tự và mỗi bậc tự do được điều khiển bởi 1 động cơ.

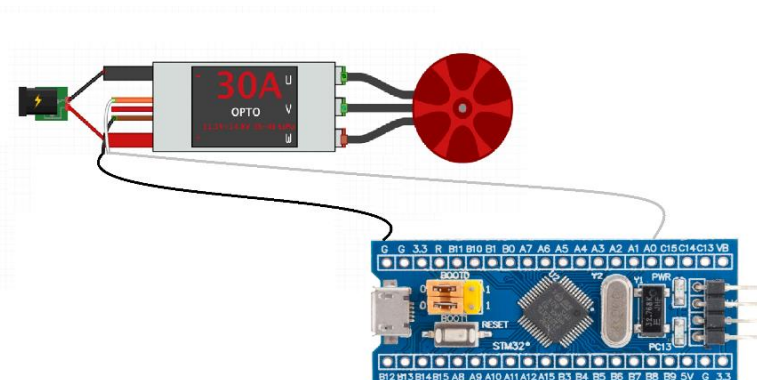
Nguyên lý hoạt động: Sử dụng công nghệ chuyển mạch điện tử để điều khiển tốc độ và mô-men xoắn của động cơ. Bộ điều khiển điều chỉnh pha và biên độ của xung dòng điện DC để đạt được khả năng kiểm soát chính xác.

3.1.8. ESC 30A: Mỗi động cơ BLDC sẽ nối với một ESC riêng

Chức năng: ESC 30A được sử dụng để điều khiển động cơ không chổi than.

Nguyên lý hoạt động: ESC nhận tín hiệu điều khiển dạng PWM từ vi điều khiển, rồi chuyển đổi thành các xung điện áp 3 pha để kích hoạt các cuộn dây động cơ theo trình tự phù hợp với vị trí rotor. Nó xác định vị trí rotor dựa vào cảm biến Hall hoặc cảm biến sensorless để kích pha chính xác.

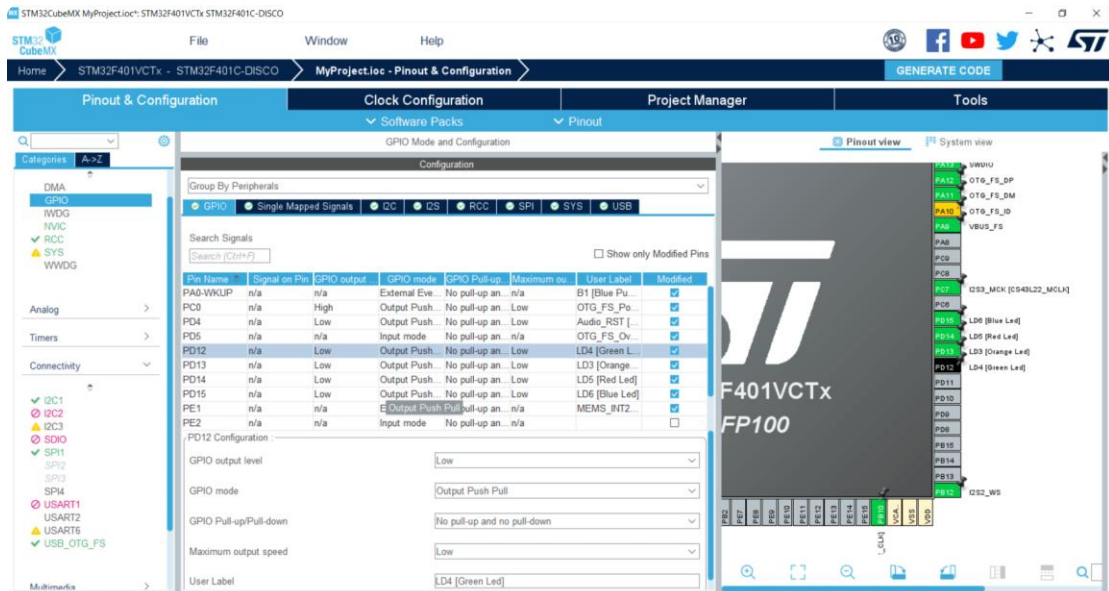
Đấu dây



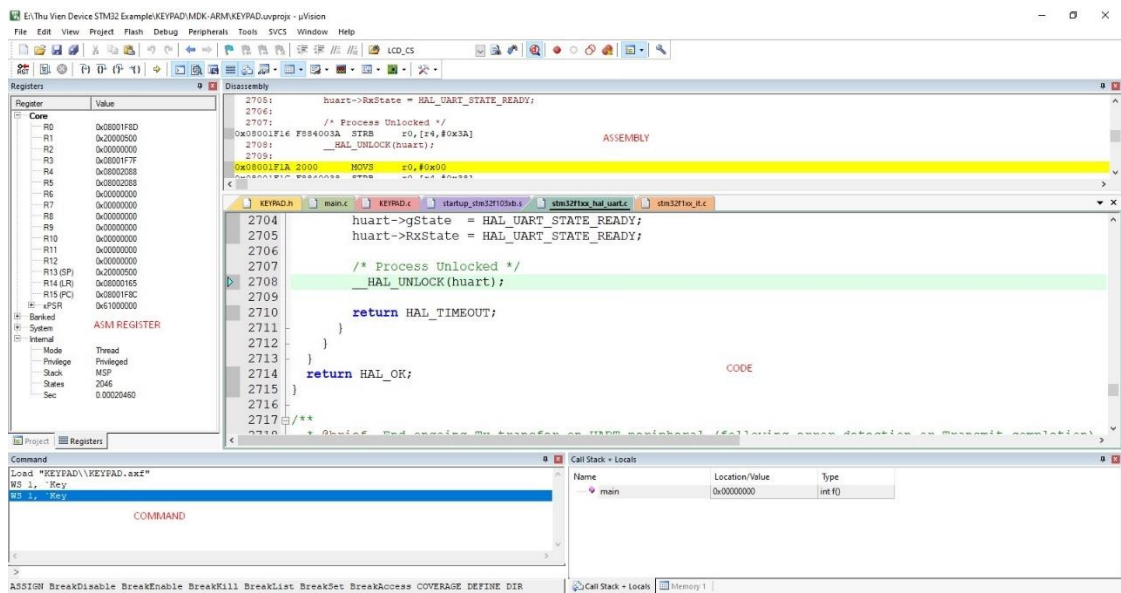
3.2. Phần mềm

Lập trình nhúng (STM32/ESP32):

- STM32: sử dụng phần mềm STM32CubeMX và Keil C để lập trình

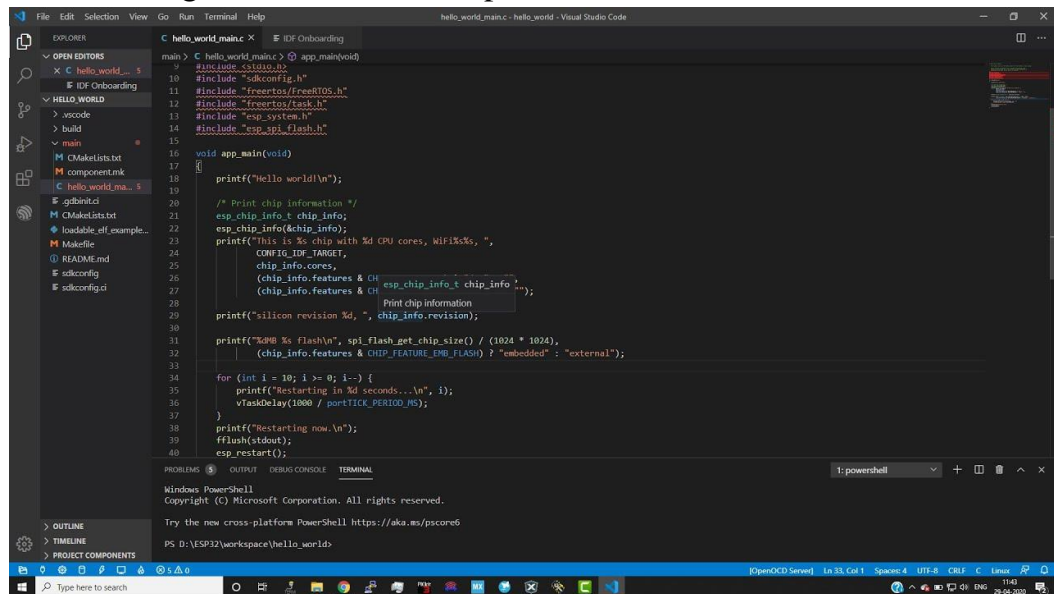


Hình 3.1. STM32CubeMX Interface



Hình 3.2. Keil C Interface

- ESP32: sử dụng framework IDF để lập trình trên VS Code



Hình 3.3. ESP IDF Interface

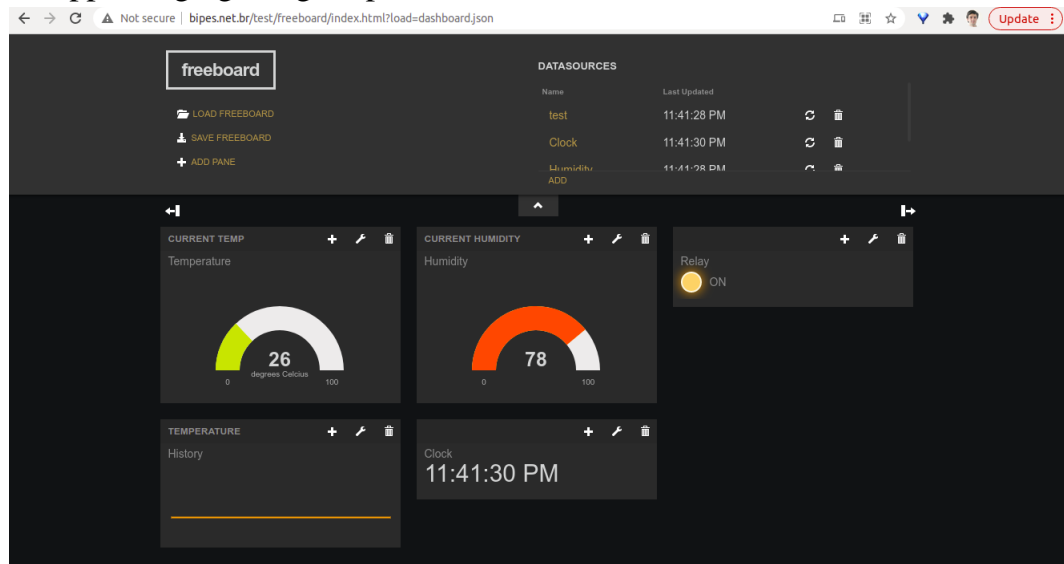
Lập trình app (IOT):

- Android app: Blynk



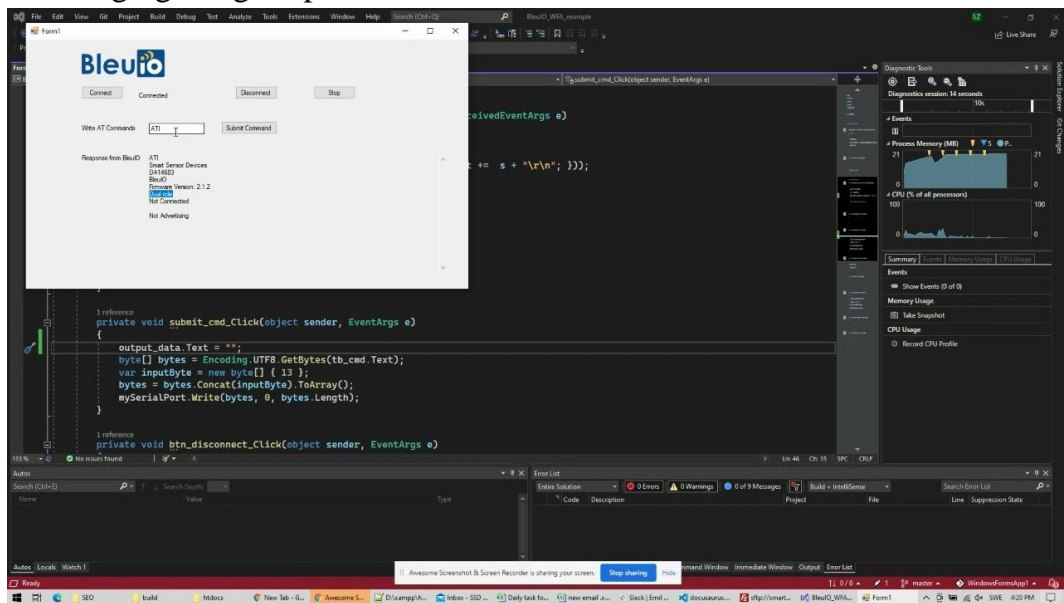
Hình 3.4. Blynk Interface

- Web app: dùng ngôn ngữ lập trình HTML



Hình 3.5. HTML for IOT App

- GUI: dùng ngôn ngữ lập trình C#



Hình 3.6. Winforms C# Interface

Mô phỏng mô hình toán của bộ điều khiển: Sử dụng MATLAB.

Mô phỏng chuyển động 3D: Sử dụng CoppeliaSim Edu.

Platform building: Sử dụng SolidWorks để tạo bản vẽ 3D chi tiết sau khi mô phỏng thành công.

IV. BÀI TOÁN CHUYỂN ĐỘNG

4.1. Tổng quan & mục tiêu

Để điều khiển những chuyển động đơn giản của robot cần điều khiển vị trí của 12 động cơ cùng lúc. Mỗi 3 motor tương ứng một chân của robot, được lắp lần lượt ở các khớp vai – hông – gối (shoulder – hip – knee). Ở đây ta chọn loại chân để thực hiện chuyển động là Serial-Leg, Trot gait, 3 DOF.

Mục tiêu: Cần điều khiển để đáp ứng các chuyển động đơn giản: *Đứng - Ngồi, Tiến - Lùi và Quay trái - Quay phải*.

4.2. Điều khiển chuyển động của khớp sử dụng DC Motor

Tiến hành triển khai phần cứng sử dụng một động cơ DC điều khiển vị trí để hiểu được chế hoạt động của thuật toán điều chế độ rộng xung (PWM) và cảm biến encoder. Sử dụng bộ điều khiển PI Fuzzy cho các ứng dụng điều khiển vị trí.

4.2.1. Mô phỏng MATLAB Simulink

Thiết kế bộ điều khiển PI Fuzzy:

Do đối tượng không có khâu tích phân lý tưởng nên dùng bộ điều khiển PI mờ.

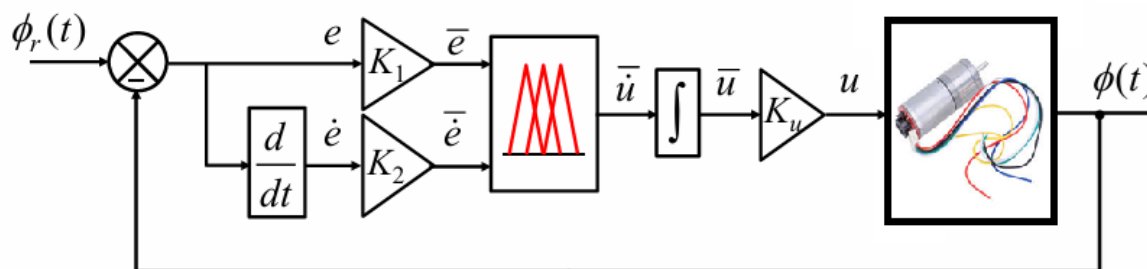
Tín hiệu đặt: $0 \leq \phi_r \leq 2\pi$ (rad)

Các biến vào/ra của bộ điều khiển PI Fuzzy cơ bản:

Hai biến vào: sai số và tốc độ biến thiên sai số.

Một biến ra: biến thiên moment điều khiển.

Sơ đồ khối hệ thống điều khiển:



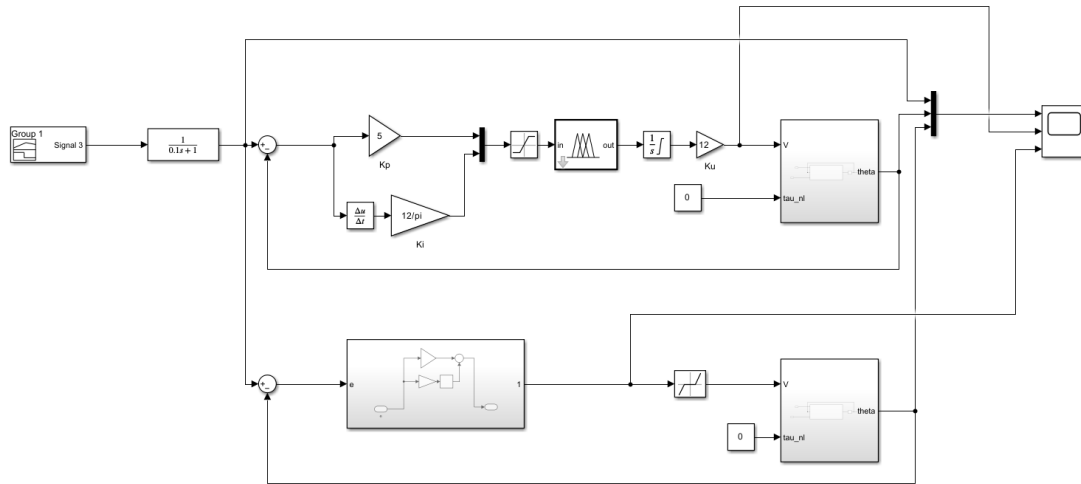
Chuẩn hóa biến vào/ ra của bộ điều khiển:

$$\text{Sai số: } e(t) = \phi_r(t) - \phi(t) \quad -2\pi \leq e \leq 2\pi \text{ (rad)} \Rightarrow K_1 = \frac{1}{2\pi}$$

$$\text{Biên thiên sai số: } \frac{\pi}{6} \leq \dot{e} \leq \frac{\pi}{6} \text{ (rad/s)} \Rightarrow K_2 = \frac{6}{\pi}$$

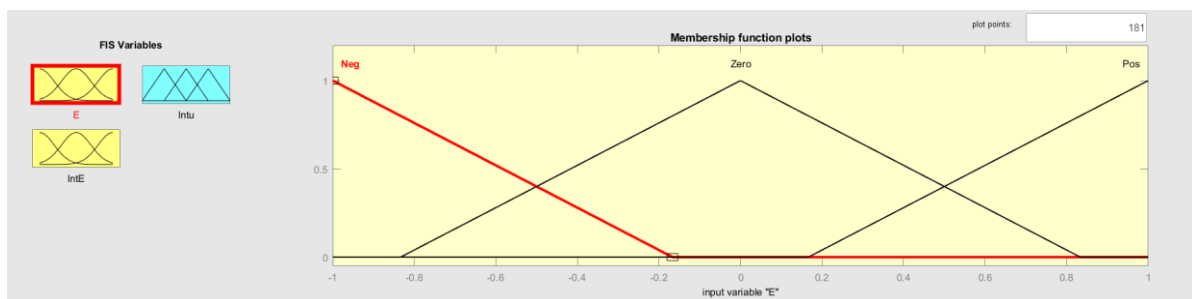
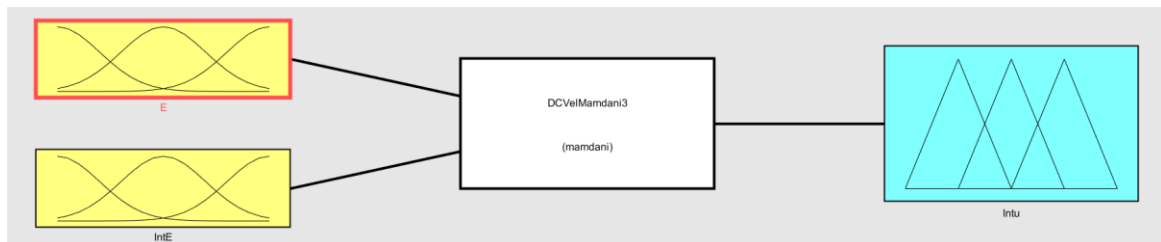
$$\text{Mômen điều khiển: } -12 \leq u \leq 12 \text{ (Nm)} \Rightarrow K_u = 12$$

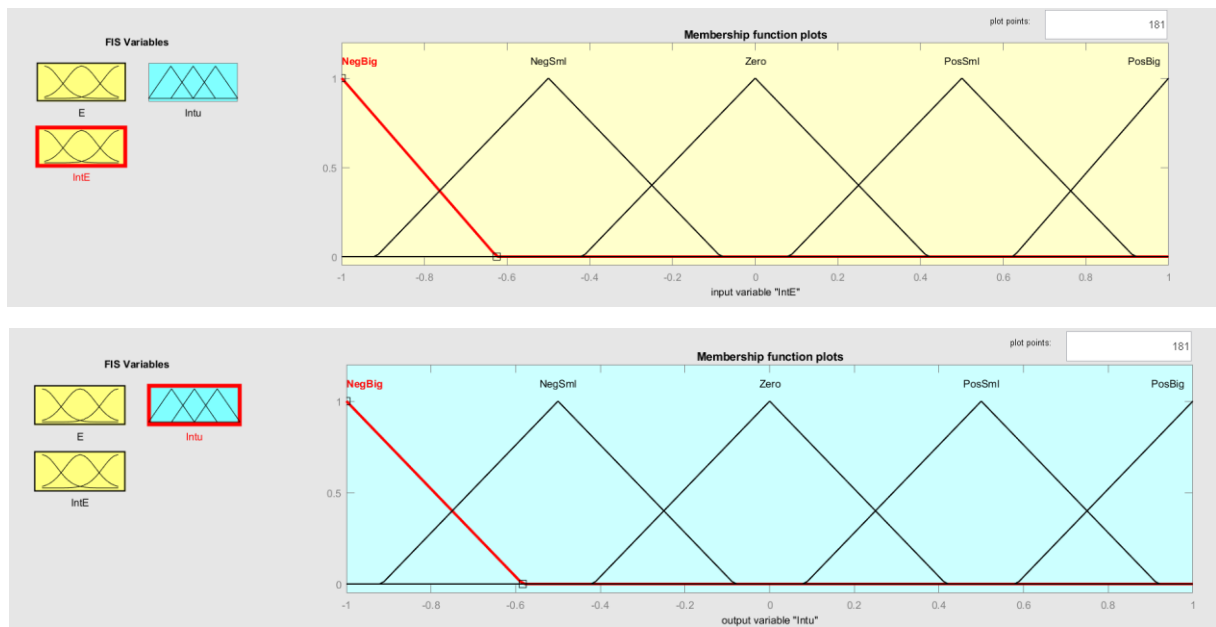
(Khi mô phỏng và tiến hành thực nghiệm các hệ số cần tinh chỉnh)



Hình 4.1. Whole System

Membership function:



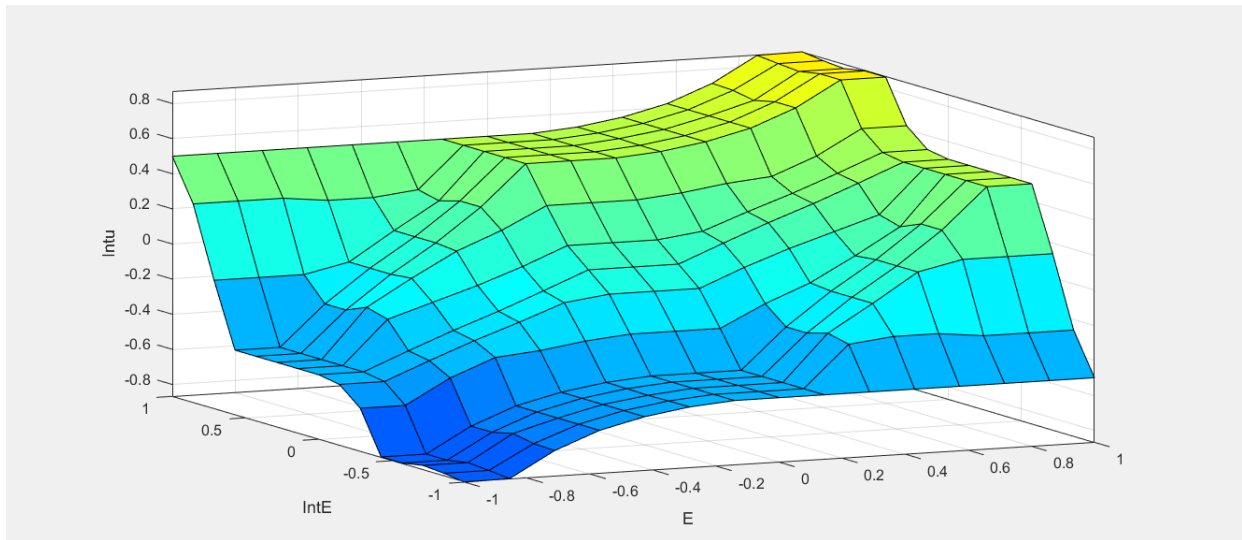


Hình 4.2. Membership function

<ol style="list-style-type: none"> 1. If (E is Neg) and (IntE is NegBig) then (u is NegBig) (1) 2. If (E is Zero) and (IntE is NegBig) then (u is NegSml) (1) 3. If (E is Pos) and (IntE is NegBig) then (u is NegSml) (1) 4. If (E is Neg) and (IntE is NegSml) then (u is NegBig) (1) 5. If (E is Zero) and (IntE is NegSml) then (u is NegSml) (1) 6. If (E is Pos) and (IntE is NegSml) then (u is PosSml) (1) 7. If (E is Neg) and (IntE is Zero) then (u is NegSml) (1) 8. If (E is Zero) and (IntE is Zero) then (u is Zero) (1) 9. If (E is Pos) and (IntE is Zero) then (u is PosSml) (1) 10. If (E is Neg) and (IntE is PosSml) then (u is NegSml) (1) 11. If (E is Zero) and (IntE is PosSml) then (u is PosSml) (1) 12. If (E is Pos) and (IntE is PosSml) then (u is PosBig) (1) 13. If (E is Neg) and (IntE is PosBig) then (u is PosSml) (1) 14. If (E is Zero) and (IntE is PosBig) then (u is PosSml) (1) 15. If (E is Pos) and (IntE is PosBig) then (u is PosBig) (1) 		
If	and	
E is		IntE is
<div>Neg</div> <div>Zero</div> <div>Pos</div> <div>none</div>		<div>NegBig</div> <div>NegSml</div> <div>Zero</div> <div>PosSml</div> <div>PosBig</div> <div>none</div>
<input type="checkbox"/> not		<input type="checkbox"/> not

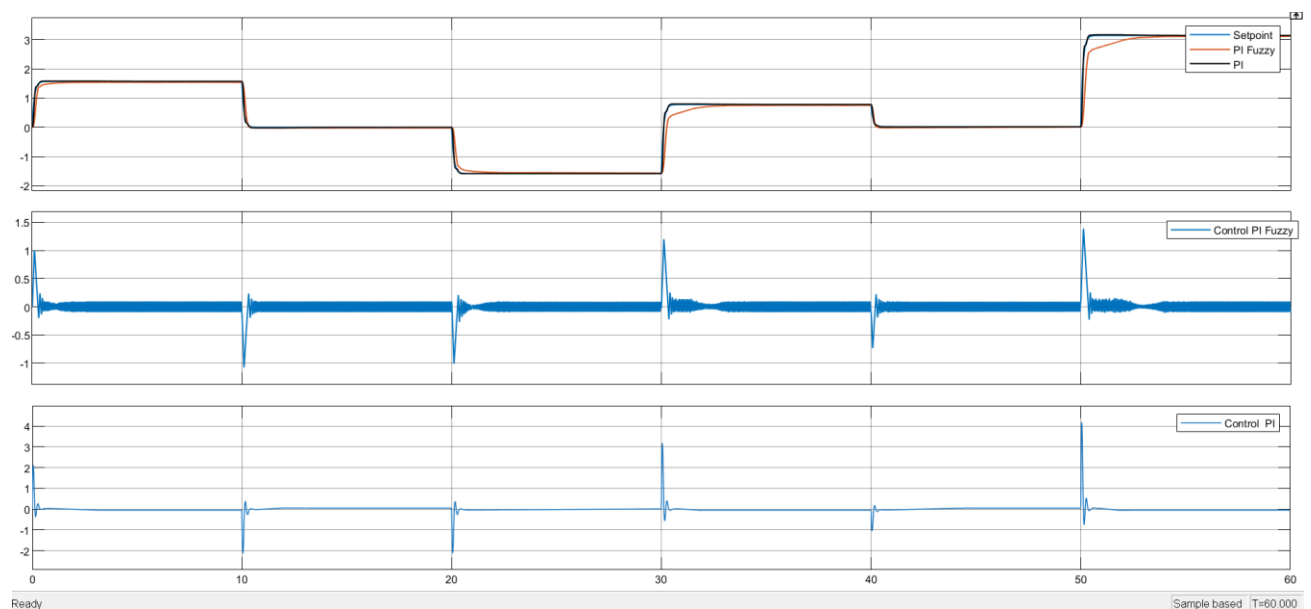
Hình 4.3. Fuzzy rules

Hệ qui tắc mờ điều khiển được đưa ra dựa vào kinh nghiệm cùng với phương pháp giải mờ trọng tâm dẫn đến mặt điều khiển như sau:



Hình 4.4. Fuzzy surface

Kết quả mô phỏng:

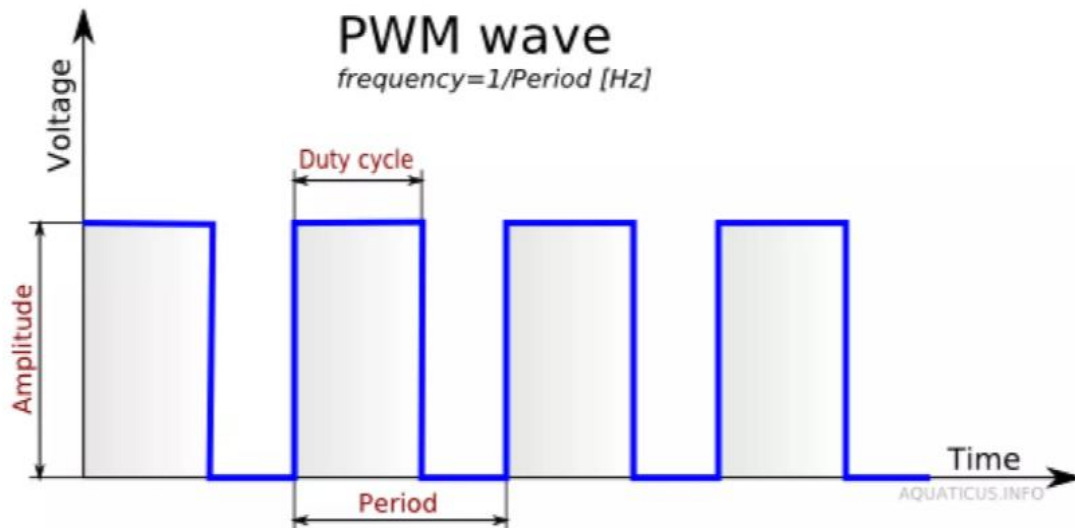


Hình 4.5. PI & PI Fuzzy response

Nhận xét: Kết quả cho thấy bộ điều khiển PI đáp ứng tốt hơn bộ điều khiển thông minh PI Fuzzy trong điều kiện lý tưởng.

4.2.2. Cấu hình và lập trình nhúng cho motor

Sử dụng phương pháp điều chế độ rộng xung PWM để điều khiển torque gián tiếp thông qua áp ngõ ra:

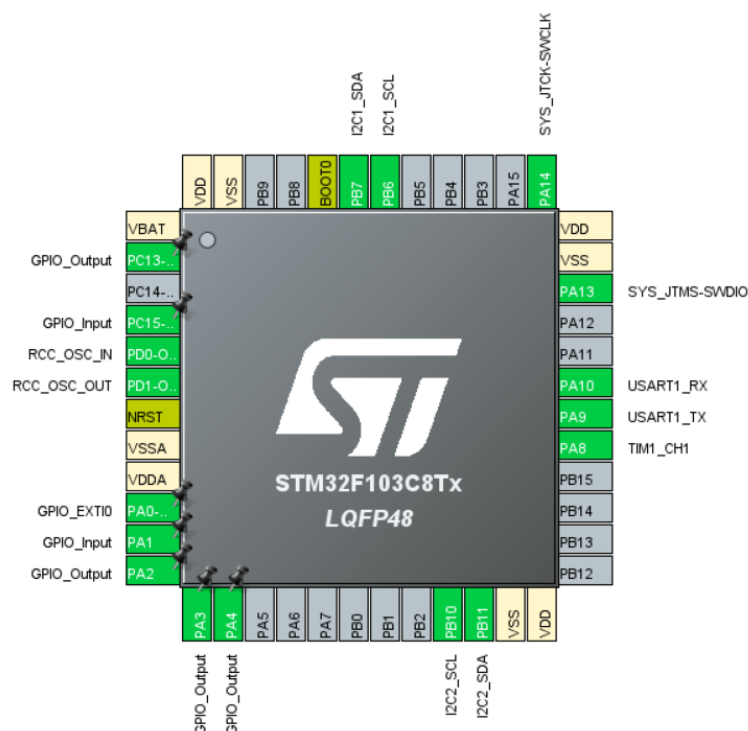


Hình 4.6. PWM Method

Giá trị điện áp trung bình trên tải: $U_t = U_{\max} \frac{T_{\text{on}}}{T_{\text{off}}} = \text{Amplitude} * \frac{\text{Duty Cycle}}{\text{Perio}}$

Ta thấy giá trị điện áp trên tải phụ thuộc vào tỉ số $\gamma = \frac{\text{Duty Cycle}}{\text{Perio}}$ do đó ứng với mỗi tần số xung, ta có thể điều chỉnh Duty cycle để điều chỉnh điện áp. Đối với vi điều khiển STM32F103C8T6 sử dụng thạch anh 72MHz, ta có các timer 16 bit hỗ trợ chế độ PWM với độ phân giải cao.

Cấu hình



TIM1 Mode and Configuration

Mode	
Slave Mode	Disable
Trigger Source	Disable
Clock Source	Internal Clock
Channel1	PWM Generation CH1

Configuration	
Reset Configuration	
✓ NVIC Settings	✓ DMA Settings
✓ Parameter Settings	✓ User Constants

Configure the below parameters :

i

✓ Counter Settings

Prescaler (PSC - 16 bits value)	0
Counter Mode	Up
Counter Period (AutoReload Regi...	65535
Internal Clock Division (CKD)	No Division
Repetition Counter (RCR - 8 bits v...	0
auto-reload preload	Disable

✓ Trigger Output (TRGO) Parameters

Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection	Reset (UG bit from TIMx_EGR)

✓ Break And Dead Time management - BR...

BRK State	Disable
BRK Polarity	High

Hình 4.7. Timer 1: tạo xung PWM qua chân PA8

TIM2 Mode and Configuration

Mode	
Slave Mode	Disable
Trigger Source	Disable
Clock Source	Internal Clock

Configuration	
Reset Configuration	
✓ Parameter Settings	✓ User Constants
✓ NVIC Settings	✓ DMA Settings

Configure the below parameters :

i

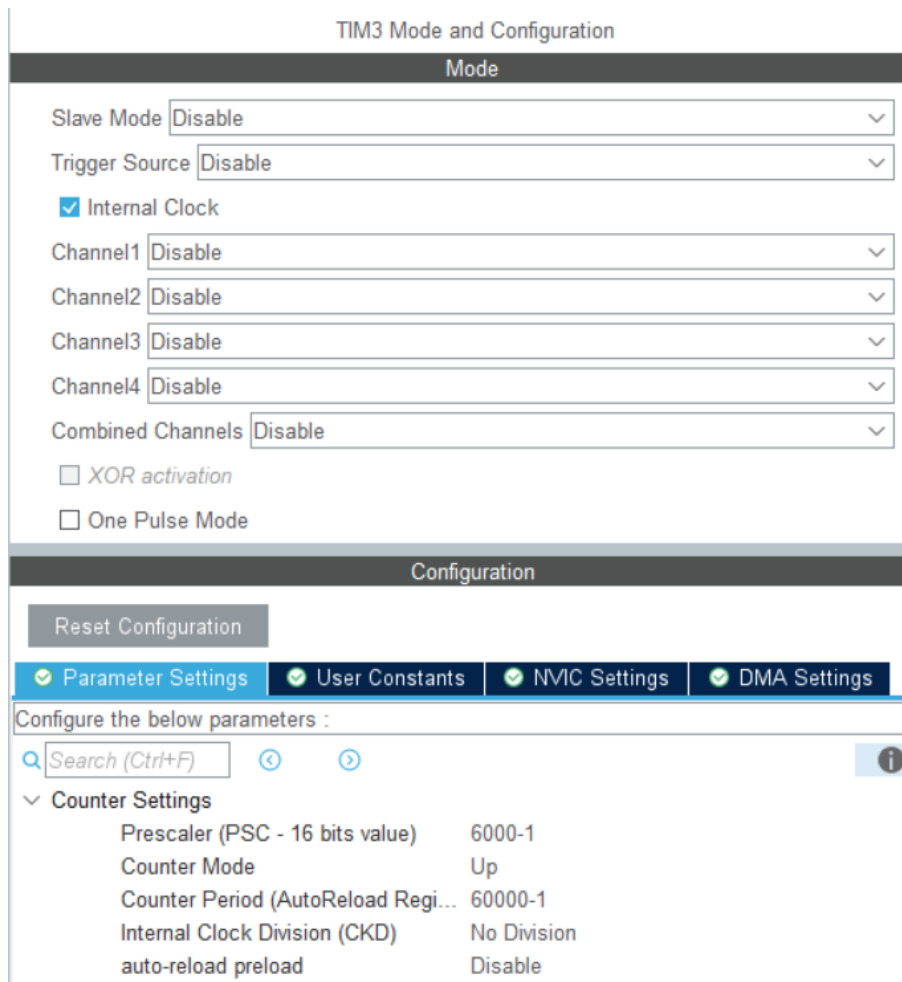
✓ Counter Settings

Prescaler (PSC - 16 bits value)	72-1
Counter Mode	Up
Counter Period (AutoReload Regis...	10000-1
Internal Clock Division (CKD)	No Division
auto-reload preload	Disable

✓ Trigger Output (TRGO) Parameters

Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
Trigger Event Selection	Reset (UG bit from TIMx_EGR)

Hình 4.8. Timer 2: đọc số xung encoder



Hình 4.9. Timer 3: định thời để thay đổi setpoint

Lập trình firmware

Thiết lập quỹ đạo tham chiếu

```
void Active_Inactive_Status_Tracking ()
{
    Timer3_Delay_Flag = 0;
    count ++;

    if (count == 1)
    {
        setpoint = 0;
    }

    if (count == 2)
    {
        setpoint = 50;
    }

    if (count == 3)
    {
```

```

        setpoint = 100;
    }

    if (count == 4)
    {
        setpoint = 150;
        count = 0;
    }
}

```

Bộ điều khiển PI Fuzzy

```

void PID_Calculation ()
{
    PID_Calculation_Flag = 0;

    //=====BEGIN MATLAB=====//
    error = setpoint - pulse;
    error = error / 150;
    if (error > 1.0f) error = 1.0f;
    if (error < -1.0f) error = -1.0f;

    integral = integral + (error * Delta_t);
    integral = integral / 1.5;

    derivative = (error - pre_error) / Delta_t;
    derivative = derivative / 1.5;
    if (derivative > 1.0f) derivative = 1.0f;
    if (derivative < -1.0f) derivative = -1.0f;

    //FUZZY FUNCTION BLOCK
    u_dot = fuzzy_inference(error, derivative);

    //INTEGRAL BLOCK
    u = u + (u_dot * Delta_t);

    //SATURATION BLOCK
    if (u > 1)
    {
        u = 1;
    }
    if (u < -1)
    {
        u = -1;
    }

    if (fabs(setpoint - pulse) <= 2)
    {

```

```

u = 0;
}

//GAIN BLOCK
duty_cycle_output = u*100;

//SATURATION BLOCK
if (duty_cycle_output > 100)
{
    duty_cycle_output = 100;
}

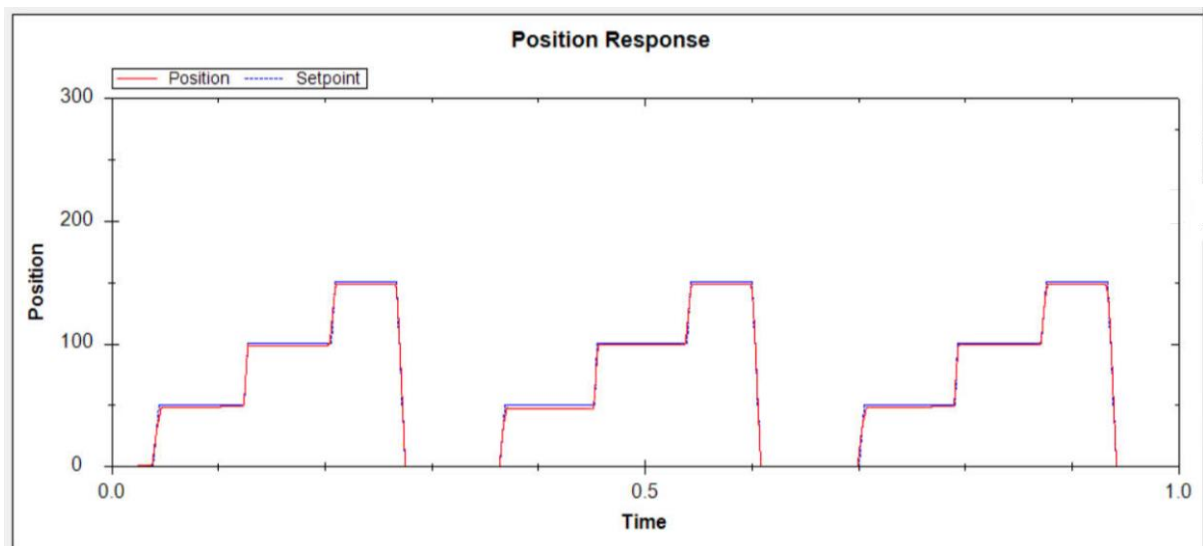
if (duty_cycle_output < -100)
{
    duty_cycle_output = -100;
}

//DEAD ZONE
if (fabs(duty_cycle_output) < 5) duty_cycle_output = 0;

//=====END MATLAB=====//
pre_error = error;

```

Kết quả



4.3. Điều khiển chuyển động của một chân robot

4.3.1. Điều khiển vị trí của điểm cuối (toe)

Để điều khiển vị trí của điểm cuối ta dùng phương trình động học thuận như đã trình bày ở (2.5 – *Forward Kinematics*). Vị trí trong không gian của điểm cuối so với trục tọa

độ gốc (base) được xác định thông qua ma trận HTM T_0^4 . Vị trí của Toe trong không gian

3D sẽ là vector tịnh tiến $p = \begin{bmatrix} m_{14} \\ m_{23} \\ m_{34} \end{bmatrix}$ tương ứng với vector $\begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix}$.

Như vậy, với bảng thông số đã được trình bày ở (2.5 – *Forward Kinematics*) ta có thể suy ra vector vị trí p của Toe trong không gian với input là giá trị góc của 3 khớp tương

ứng $\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$.

4.3.2. Điều khiển góc từng khớp (joint)

Để điều khiển góc từng khớp cần sử dụng phương trình IK như đã được trình bày ở (2.5 – *Inverse Kinematics*). Góc ở đây đóng vai trò là output với các input là vị trí của điểm

cuối (Toe) so với gốc (Base) $\begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix}$. Từ vector tọa độ ta có thể suy ra góc tương ứng của 3

khớp $\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$ bằng cách giải hệ phương trình ba ẩn số:

$$\theta_1 = -atan2(-y_4, x_4) - atan2(\sqrt{x_4^2 + y_4^2 - L_1^2}, -L_1)$$

$$\theta_2 = atan2\left(z_4, \sqrt{x_4^2 + y_4^2 - L_1^2}\right) - atan2(L_3 \sin(\theta_3), L_2 + L_3 \cos(\theta_3))$$

$$\theta_3 = atan2(-\sqrt{1 - D^2}, D) \quad (\text{Legs for 1 and 3})$$

$$\theta_3 = atan2(\sqrt{1 - D^2}, D) \quad (\text{Legs for 2 and 4})$$

$$D = (x_4^2 + y_4^2 - L_1^2 + z_4^2 - L_2^2 - L_3^2) / (2 L_2 L_3)$$

V. BÀI TOÁN TRÁNH VẬT CẢN

5.1. Tổng quan & mục tiêu

Để robot phát hiện và tránh vật cản đơn giản cần dùng một cảm biến để đo khoảng cách.

Mục tiêu đo được khoảng cách vật cách robot 5cm thì robot sẽ dừng.

5.2. Xử lý dữ liệu cảm biến

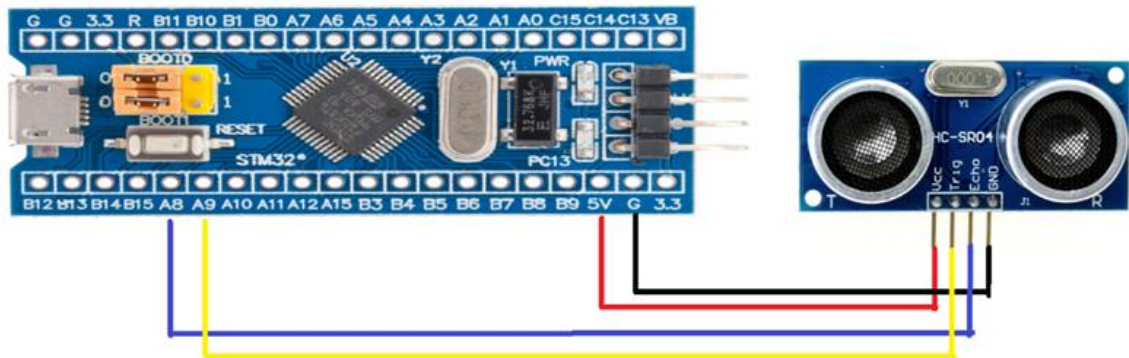
Cấu hình:

PA9 - GPIO Output

PA8 - GPIO Input

Timer1 - Internal Clock

Kết nối cảm biến siêu âm



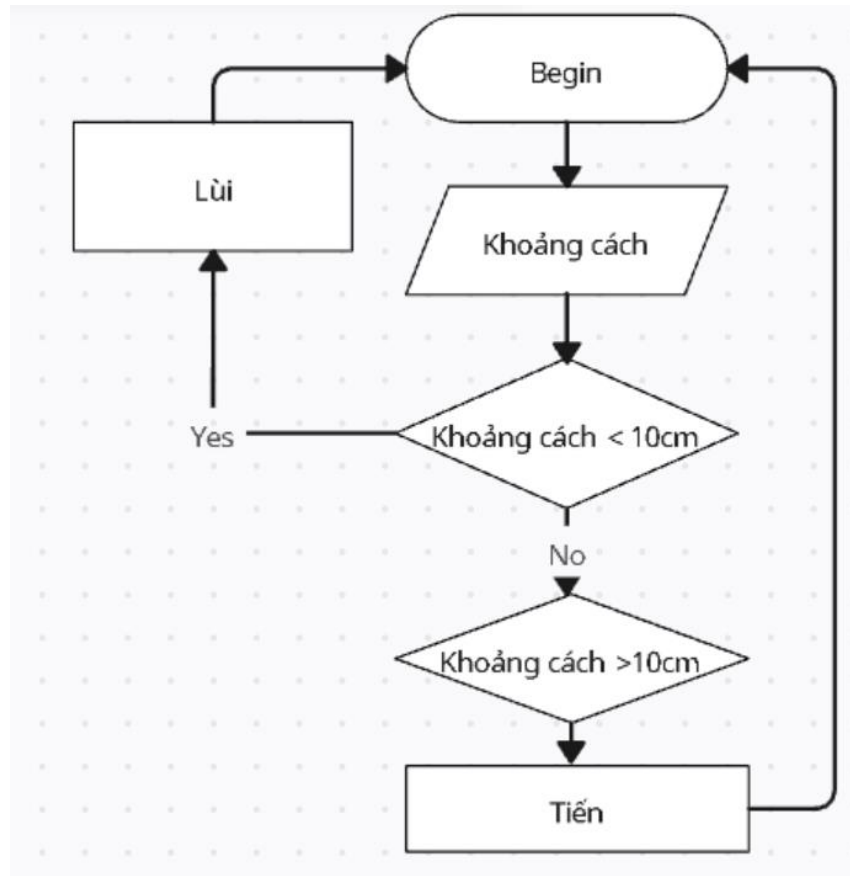
VCC: Nối với nguồn 5V

Trig: Chân kích hoạt, do STM32 điều khiển để gửi xung tín hiệu đến cảm biến.

Echo: Chân nhận tín hiệu phản hồi, nối với chân input của STM32 để nhận xung phản hồi từ vật cản.

GND: Nối đất.

5.3. Thuật toán tránh vật cơ bản



Cảm biến siêu âm được kích hoạt bằng cách gửi một xung đến chân Trig, làm cho cảm biến phát ra sóng siêu âm. Sau khi sóng phản xạ khỏi vật cản, nó sẽ quay trở lại chân Echo, tại đó ta có thể tính được thời gian tín hiệu phản hồi.

Khoảng cách đến vật cản được tính bằng công thức sau dựa trên thời gian tín hiệu phản hồi:

$$\text{Distance} = \frac{\text{Timer} \times \text{Speed of Sound}}{2}$$

Speed of Sound: tốc độ âm thanh trong không khí, khoảng 343 m/s
Sau khi đã xây dựng một robot tránh vật cản đơn giản, có thể cải tiến dự án bằng cách thêm các cảm biến bổ sung bên trái và bên phải để điều hướng chính xác hơn. Áp dụng các thuật toán điều khiển nâng cao hơn, lập kế hoạch đường đi.

KẾT LUẬN

Trong khuôn khổ Đồ án 1, nhóm đã nghiên cứu tổng quan về cấu trúc phần cứng và phương pháp điều khiển của robot bốn chân với cấu hình chân nối tiếp (Serial-Leg). Các nội dung trọng tâm bao gồm phân tích động học thuận và nghịch, mô hình hóa chuyển động của từng chân, xây dựng thuật toán tránh vật cản sử dụng cảm biến siêu âm, và tiến hành mô phỏng bộ điều khiển PI Fuzzy trên nền tảng MATLAB. Quá trình thực hiện không chỉ giúp nhóm củng cố kiến thức lý thuyết mà còn tạo điều kiện tiếp cận thực tế thông qua việc triển khai các thuật toán điều khiển trên phần cứng.

Trong thời gian tới, nhóm định hướng tiếp tục hoàn thiện phần cứng robot và thực hiện kiểm thử toàn bộ hệ thống trong môi trường thực tế. Đồng thời, nhóm sẽ tập trung tối ưu hóa các thuật toán điều khiển, mở rộng khả năng vận hành của robot trên nhiều loại địa hình khác nhau. Việc tích hợp thêm các cảm biến nâng cao, công cụ định hướng và hệ thống quan sát như camera sẽ được xem xét nhằm tăng cường khả năng thích nghi, xử lý tình huống phức tạp và mở rộng phạm vi ứng dụng trong các lĩnh vực như công nghiệp, cứu hộ và giám sát môi trường. Đây sẽ là tiền đề quan trọng cho các giai đoạn phát triển tiếp theo, hướng đến xây dựng một hệ thống robot bốn chân thông minh, ổn định và có tính ứng dụng cao.

TÀI LIỆU THAM KHẢO

- [1] Fukuoka, Y., Kimura, H., & Cohen, A.H. (2003). *Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concept. The International Journal of Robotics Research*, 22(3-4), 187-202. <https://journals.sagepub.com/doi/10.117>
- [2] Yasar, C. F. (2025). *DC Motor Position with Fuzzy Logic Control* the method that applies fuzzy logic to improve response (Version 1.0.0) [MATLAB code]. MathWorks File Exchange. <https://www.mathworks.com/matlabcentral/fileexchange/167991-dc-motor-position-with-fuzzy-logic-control>
- [3] Md, H.R., Md, F.A.M., Saadia, B.A., & Mohammad, S. (2022). *Kinematics analysis of a quadruped robot: Simulation and Evaluation*. https://www.researchgate.net/publication/361466674_Kinematics_analysis_of_a_quadruped_robot_Simulation_and_Evaluation
- [4] Zhao, J., Liu, K., Zhao, F., & Sun, Z. (2019). *Design and Kinematic Analysis on a Novel Serial-Parallel Hybrid Leg for Quadruped Robot*. https://link.springer.com/chapter/10.1007/978-3-030-27535-8_39
- [5] Niu, J., Wang, H., Jiang, J., & Chen, L. *Kinematic Analysis of a Serial-Parallel Hybrid Mechanism and Its Application to a Wheel-Legged Robo*. <https://ieeexplore.ieee.org/abstract/document/9115045>
- [6] Burgul, C., Lee, W., Geneva, P., & Huang, G. *Online Determination of Legged Kinematics*.
- [7] Kalyoncu, M. (2017). *Inverse Kinematic Analysis of a Quadruped Robot*. https://www.researchgate.net/publication/322594373_Inverse_Kinematic_Analysis_of_a_Quadruped_Robot