**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA ĐIỆN - ĐIỆN TỬ**

**BÁO CÁO BÀI TẬP LỚN**

**MÔN HỌC: KỸ THUẬT ROBOT**

**THIẾT KẾ GIAO DIỆN MÔ PHỎNG, TÍNH TOÁN ĐỘNG HỌC**

**QUY HOẠCH QUỸ ĐẠO CHO CÁNH TAY ROBOT 4 BẬC TỰ DO**

**GVHD: TS. NGUYỄN HOÀNG GIÁP**

**LỚP: L01 – HK: 251**

|  |  |
| --- | --- |
| **Sinh viên thực hiện** | **Mã số sinh viên** |
| Lê Gia Uy | 2213895 |

**TP. HỒ CHÍ MINH, 2025**

**MỤC LỤC**

[CHƯƠNG I. GIỚI THIỆU ĐỀ TÀI 3](#_Toc215387794)

[1. Giới thiệu đề tài 3](#_Toc215387795)

[2. Mục tiêu của đề tài 4](#_Toc215387796)

[CHƯƠNG II. TÍNH TOÁN CÁC BÀI TOÁN LÝ THUYẾT 5](#_Toc215387797)

[1.. Bài toán động học thuận – Forward Kinematics (FK) 5](#_Toc215387798)

[2. Bài toán động học ngược - Inverse Kinematics (IK) 9](#_Toc215387799)

[3. Bài toán quy hoạch quỹ đạo 11](#_Toc215387800)

[CHƯƠNG III. THIẾT KẾ PHẦN MỀM VÀ KẾT QUẢ MÔ PHỎNG 14](#_Toc215387801)

[1. Thiết kết phần mềm: 14](#_Toc215387802)

[2. Thực hiện các thuật toán 15](#_Toc215387803)

[2.1. Xây dựng mô hình 3D. 15](#_Toc215387804)

[2.2. Xử lý Động học thuận 15](#_Toc215387805)

[2.3. Xử lý Động học ngược và Quy hoạch 15](#_Toc215387806)

[3. Kết quả mô phỏng 16](#_Toc215387807)

[3.1. Kiểm chứng động học thuận 16](#_Toc215387808)

[3.2. Kiểm chúng động học ngược 16](#_Toc215387809)

[3.3. Kiểm chứng quy hoạch quỹ đạo 17](#_Toc215387810)

[3.4. Kết luận chung 17](#_Toc215387811)

[CHƯƠNG IV. KẾT LUẬN 18](#_Toc215387812)

[1. Kết luận 18](#_Toc215387813)

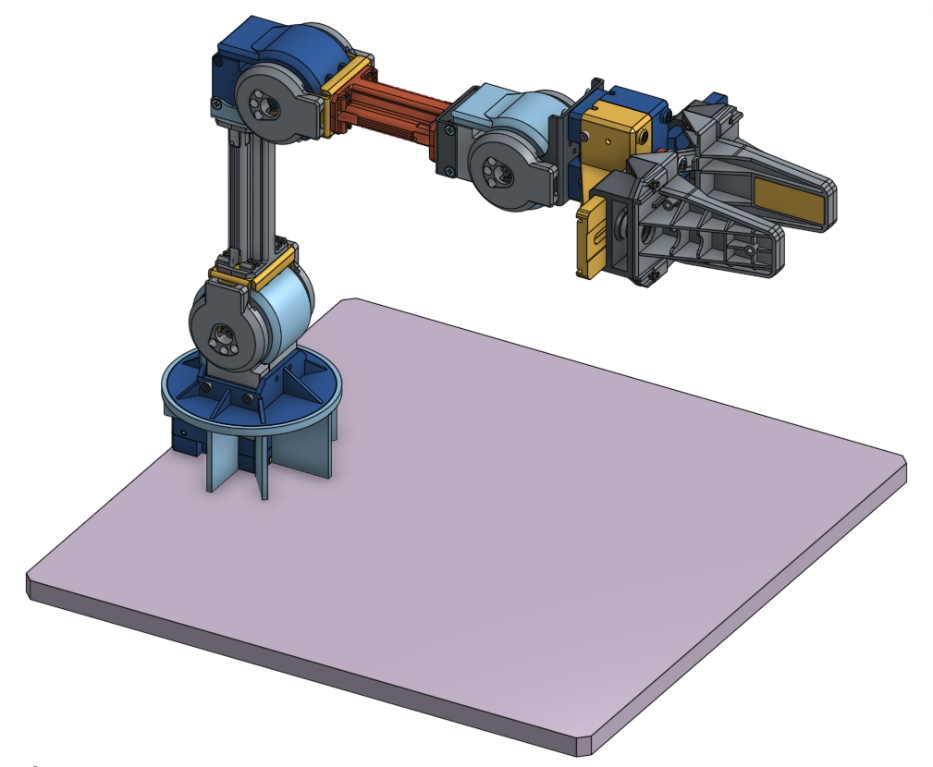
[2. Hạn chế 18](#_Toc215387814)

# CHƯƠNG I. GIỚI THIỆU ĐỀ TÀI

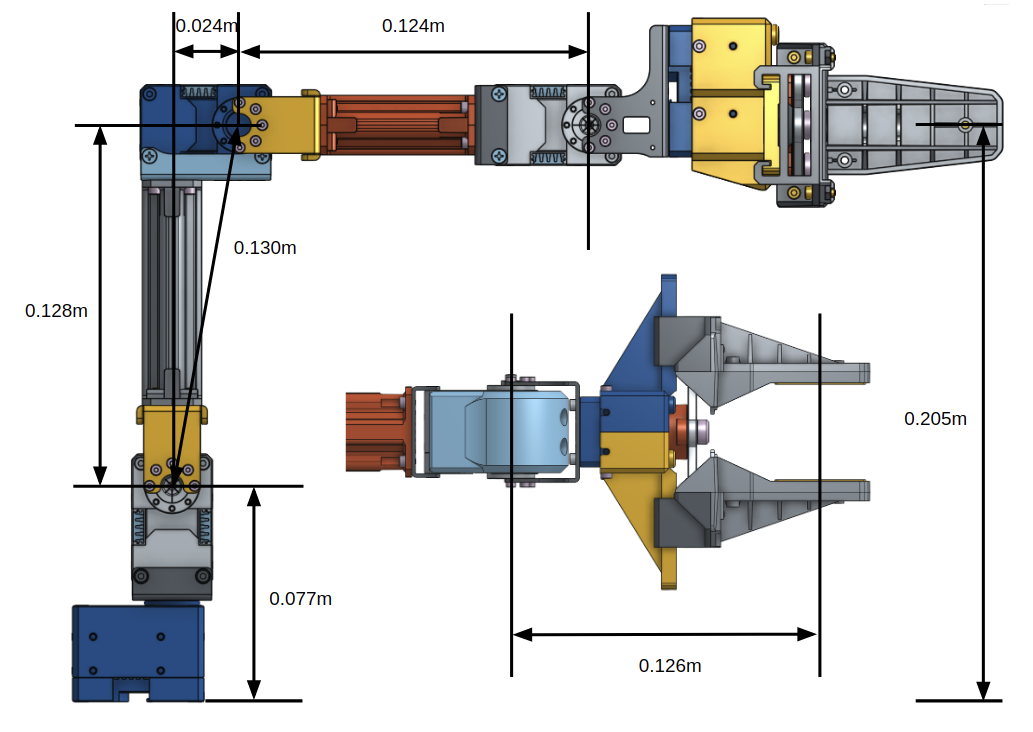
## 1. Giới thiệu đề tài

Trong kỷ nguyên công nghiệp 4.0, cánh tay robot đóng vai trò then chốt trong dây chuyền tự động hóa sản xuất. Việc nắm vững cơ sở lý thuyết về động học (Kinematics) và quy hoạch quỹ đạo (Trajectory Planning) là nền tảng bắt buộc đối với kỹ sư Robot.

Tuy nhiên, việc tính toán thủ công các phương trình động học thường phức tạp và khó hình dung trực quan. Do đó, việc xây dựng một phần mềm mô phỏng giúp kiểm chứng lý thuyết, trực quan hóa không gian làm việc và kiểm tra quỹ đạo chuyển động trước khi áp dụng lên robot thực là vô cùng cần thiết.

Đề tài này tập trung nghiên cứu mô hình robot OpenManipulator-X (4-DOF), sử dụng công cụ MATLAB App Designer để xây dựng một giao diện điều khiển toàn diện, tích hợp tính toán động học và quy hoạch quỹ đạo thời gian thực.

Hình 1.1 Robot OpenMANIPULATOR-X[[1]](#footnote-1)

Hình 1.2. Kích thước của Robot[[2]](#footnote-2)

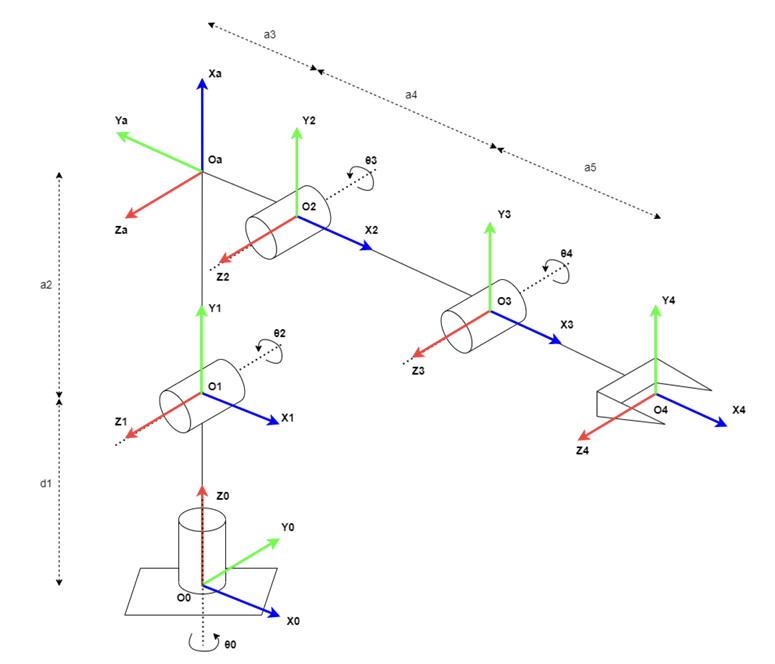
## 2. Mục tiêu của đề tài

* Xây dựng mô hình toán học: Thiết lập bảng tham số Denavit-Hartenberg (D-H), xây dựng phương trình Động học thuận (FK) và Động học nghịch (IK) cho robot 4 bậc tự do.
* Giải quyết bài toán thuận và động học nghịch:
* Quy hoạch quỹ đạo (Trajectory Planning): Áp dụng phương pháp LSPB (Linear Segment with Parabolic Blend) để tạo profie vận tốc hình thang, giúp robot di chuyển mượt mà, giảm rung động.
* Thiết kế giao diện (GUI): Xây dựng phần mềm trực quan trên MATLAB cho phép người dùng nhập liệu, điều khiển và giám sát các thông số (Vị trí, Vận tốc, Gia tốc) qua đồ thị.

# CHƯƠNG II. TÍNH TOÁN CÁC BÀI TOÁN LÝ THUYẾT

## 1.. Bài toán động học thuận – Forward Kinematics (FK)

Bước 1: Đặt các hệ trục tọa độ

Dựa theo các quy tắc Denavit-Hartenberg (D-H), hình 1.1 và hình 1.2, xác định được các hệ trục toạ độ của Robot 

Hình 2.1. Đặt các hệ trục tọa độ

Với các thông số: d1 = 77 mm, a2 = 128 mm, a3 = 24 mm, a4 = 124 mm, a5 = 126 mm

Bước 2: Xây dựng bảng DH

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **θi** | **di** | **ai** | **αi** |
| **1** | θ1 | d1 | 0 |  |
| **2** | θ2 + | 0 | a2 | 0 |
| **a** |  | 0 | a3 | 0 |
| **3** | θ3 | 0 | a4 | 0 |
| **4** | θ4 | 0 | a5 | 0 |

Với các thông số: d1 = 77 mm, a2 = 128 mm, a3 = 24 mm, a4 = 124 mm, a5 = 126 mm

Bước 3: Tính các ma trận biến đổi thuần nhất giữa các khớp và từ gốc tọa độ đến EE

Dựa trên bảng tham số D-H đã thiết lập, ma trận biến đổi thuần nhất tổng quát từ hệ quy chiếu thứ i về hệ quy chiếu thứ i-1 được xác định theo công thức Denavit-Hartenberg:

Áp dụng các tham số từ bảng D-H vào công thức tổng quát, ta xác định được các ma trận biến đổi cho từng khớp như sau:

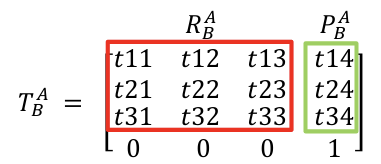
Ma trận biến đổi từ khớp 1 về gốc tọa độ:

  Ma trận biến đổi từ khớp 2 về khớp 1

Ma trận biến đổi từ khớp trung gian về khớp 2

Ma trận biến đổi từ khớp 3 về khớp trung gian

Ma trận biến đổi từ khớp 4 về khớp 3 trung gian

Từ các ma trận biến đổi giữa các khớp, ta tính được ma trận biến đổi từ EE đến gốc tọa độ. từ ma trận tìm được hướng và vị trí của EE

Bước 4: Công thức tính giá trị tọa độ P= [Px Py Pz]; Roll, Pitch, Yaw

Trong bài toán Động lực thuận, ta sẽ cần xác định vị trí và hướng (thể hiện qua các góc biến đổi Roll, Pitch, Yall) của khâu End Effector so với khâu Base cố định:

Từ ma trận biến đổi , tìm được công thức tính vị trí:

Từ ma trận xoay

Tìm được công thức tính Roll, Pitch, Yaw:

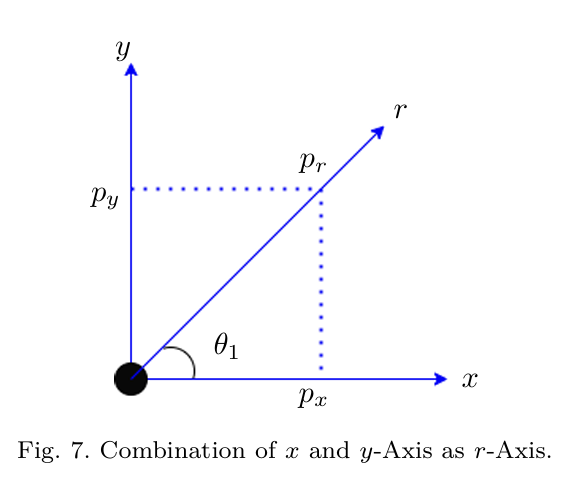
Nếu

Nếu

## 2. Bài toán động học ngược - Inverse Kinematics (IK)

Bài toán động học ngược yêu cầu tìm bộ biến khớp = [] khi biết trước:

* Vị trí điểm tác động cuối: = .
* Góc Pitch của khâu cuối so với mặt phẳng ngang: (radian).

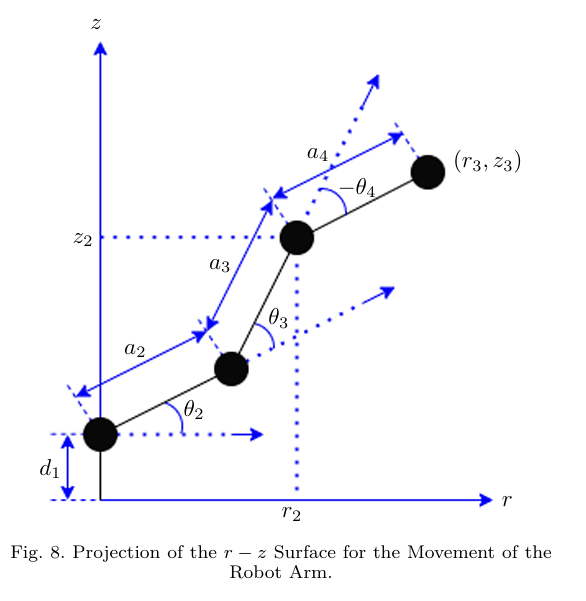
Chiếu điểm xuống mặt phẳng . Góc quay là góc tạo bởi vector vị trí P với trục

Hình 2.2. Gộp trục x y thành trục r[[3]](#footnote-3)

Robot OpenMANIPULATOR-X có 4 bậc tự do, trong đó 3 khớp cuối (Khớp 2, 3, 4) cùng di chuyển trong một mặt phẳng đứng. Nếu ta cố gắng giải trực tiếp từ gốc đến điểm cuối (End-Effector), phương trình sẽ rất phức tạp vì nó phụ thuộc vào cả 4 góc cùng lúc.

Nhận thấy rằng vị trí của điểm cuối phụ thuộc vào vị trí khớp 4 cộng thêm độ dài khâu cuối theo một góc nghiêng Nếu biết điểm cuối và góc nghiêng, ta có thể lùi lại để tìm vị trí của Khớp 4 (gọi là Tâm cổ tay - Wrist Center, viết tắt là WC).

Khi đã biết vị trí WC, bài toán trở thành tìm , để đưa cánh tay chạm vào WC. Lúc này khâu cuối , không còn tham gia vào bài toán tam giác nữa, giúp việc tính toán đơn giản hơn rất nhiều.

Quy đổi tọa độ không gian 3D sang mặt phẳng 2D (r-z). Bước này tương ứng với việc gộp trục x và y thành trục bán kính r

Hình 2.3. Mặt phẳng r - z[[4]](#footnote-4)

Tính lùi về tâm cổ tay (, ) Quan sát Hình 8, ta thấy điểm (, ) cách điểm (, ) một đoạn bằng độ dài theo hướng góc.

Giải bài toán tam giác để tìm góc Xét tam giác được tạo bởi ba điểm: Gốc tọa độ (Khớp 2), Điểm nối khớp 3, và Tâm cổ tay (, )

Hai cạnh bên là và .

Cạnh huyền là khoảng cách từ gốc đến (, ) có độ dài

Từ đó tính được góc

Tính góc được xác định là góc tạo bởi khâu và trục hoành. Dựa vào hình học, góc này là hiệu của góc nâng vector vị trí (, ) và góc lệch của tam giác:

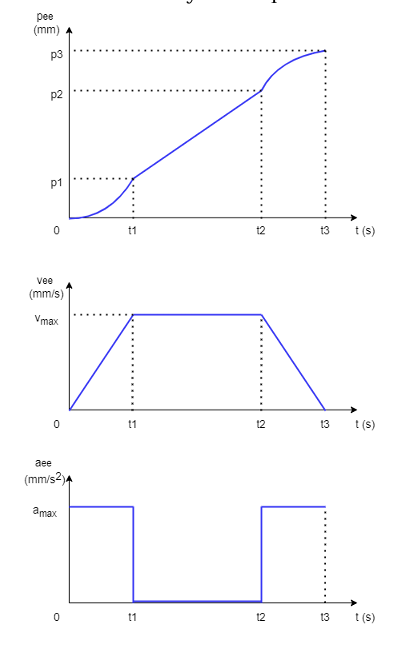
Góc Dựa vào hình vẽ, tổng các góc khớp bằng góc hướng cuối cùng $\phi$4:

Việc sử dụng các biến trung gian (, ) như trong Hình 8 giúp tách biệt bài toán thành hai phần: phần định vị trí cổ tay (phụ thuộc , , ) và phần định hướng khâu cuối (phụ thuộc ) giúp đơn giản hóa hệ phương trình động học ngược.

## 3. Bài toán quy hoạch quỹ đạo

Để đảm bảo robot di chuyển mượt mà từ điểm đầu đến điểm cuối, tránh hiện tượng rung giật do thay đổi vận tốc đột ngột, ta sử dụng phương pháp LSPB cho các khớp

Đặc điểm của phương pháp này là tạo ra biên dạng vận tốc hình thang (như Hình minh họa), bao gồm 3 giai đoạn chuyển động:

* Giai đoạn Tăng tốc: Vận tốc tăng tuyến tính với gia tốc không đổi . Quỹ đạo vị trí có dạng Parabol.
* Giai đoạn Vận tốc đều: Vận tốc giữ nguyên hằng số Quỹ đạo vị trí là đường thẳng (Linear).
* Giai đoạn Giảm tốc:Vận tốc giảm tuyến tính về 0 với gia tốc âm Quỹ đạo vị trí có dạng Parabol.

Hình 2.4. Đồ thị vận tốc hình thang

Bước 1: Giải bài toán ngược tỉm ra các bộ số = [] và Tính góc lệch giữa vị trí đích và vị trí hiện tại cho từng khớp k:

Bước 2: Dựa trên vận tốc giới hạn và gia tốc giới hạn tính toán thời gian tối thiểu cần thiết cho mỗi khớp để hoàn thành quãng đường :

Bước 3: Tìm ra thời gian của khớp "chậm nhất" (khớp cần nhiều thời gian nhất). Sau đó, ép tất cả các khớp còn lại phải di chuyển chậm lại để cùng hoàn thành trong thời gian . Vận tốc đồng bộ mới của từng khớp được tính lại:

Bước 4: Sinh quỹ đạo theo thời gian thực Tại mỗi thời điểm $t$, vị trí $q(t)$ và vận tốc $v(t)$ được tính theo 3 giai đoạn (gọi $t\_1$ là thời điểm kết thúc tăng tốc):

Giai đoạn 1 (Tăng tốc):

Giai đoạn 2 (Vận tốc đều): với

Giai đoạn 3 (Giảm tốc):

# CHƯƠNG III. THIẾT KẾ PHẦN MỀM VÀ KẾT QUẢ MÔ PHỎNG

## 1. Thiết kết phần mềm:

Phần mềm mô phỏng được xây dựng trên môi trường MATLAB App Designer. Đây là môi trường lập trình giao diện người dùng (GUI) hiện đại, hỗ trợ mạnh mẽ các công cụ toán học và đồ họa 3D, phù hợp cho việc mô phỏng cánh tay robot.

Giao diện chương trình được chia thành 4 khu vực chức năng chính:

* Khu vực Hiển thị 3D (Main Visualization):
* Sử dụng UIAxes để vẽ mô hình robot 4 bậc tự do theo thời gian thực.
* Hiển thị không gian làm việc (Workspace) và quỹ đạo di chuyển (Path Tracing). Các hệ trục tọa độ gắn liền với từng khớp được hiển thị trực quan.
* Bảng điều khiển Động học thuận (FORWARD Panel):
* Gồm 4 thanh trượt (Slider) và ô nhập liệu tương ứng với 4 khớp [].
* Nút RESET để đưa robot về vị trí mặc định và nút CLEAR để xóa đường vẽ quỹ đạo.
* Bảng điều khiển Động học ngược & Quy hoạch (INVERSE & PLANNING Panel):
* Cho phép nhập tọa độ mong muốn và góc hướng Pitch ()
* Cho phép nhập các thông số giới hạn động học: Vận tốc tối đa gia tốc tối đa
* Nút INVERSE: Tính toán và đưa robot đến đích ngay lập tức.
* Nút PLANNING: Kích hoạt giải thuật LSPB để robot di chuyển mượt mà.
* Khu vực đồ thị giám sát: Gồm 8 đồ thị nhỏ hiển thị Vị trí và Vận tốc của từng khớp theo thời gian thực khi chạy chế độ Planning.

## 2. Thực hiện các thuật toán

### 2.1. Xây dựng mô hình 3D.

Xây dựng mô hình 3D Mô hình robot được xây dựng từ các khối cơ bản (Hình trụ - Cylinder, Hình hộp - Box) trong hàm Init\_Geometry\_Data. Vị trí các khối này được cập nhật liên tục thông qua hàm Draw\_Robot, sử dụng các ma trận biến đổi thuần nhất đã tính toán ở Chương II.

### 2.2. Xử lý Động học thuận

Khi người dùng thay đổi giá trị trên thanh trượt (Slider), chương trình sẽ:

* Đọc giá trị góc khớp.
* Tính toán chuỗi ma trận
* Nhân các ma trận để tìm (End-Effector).
* Cập nhật vị trí hiển thị và các góc Euler lên giao diện.

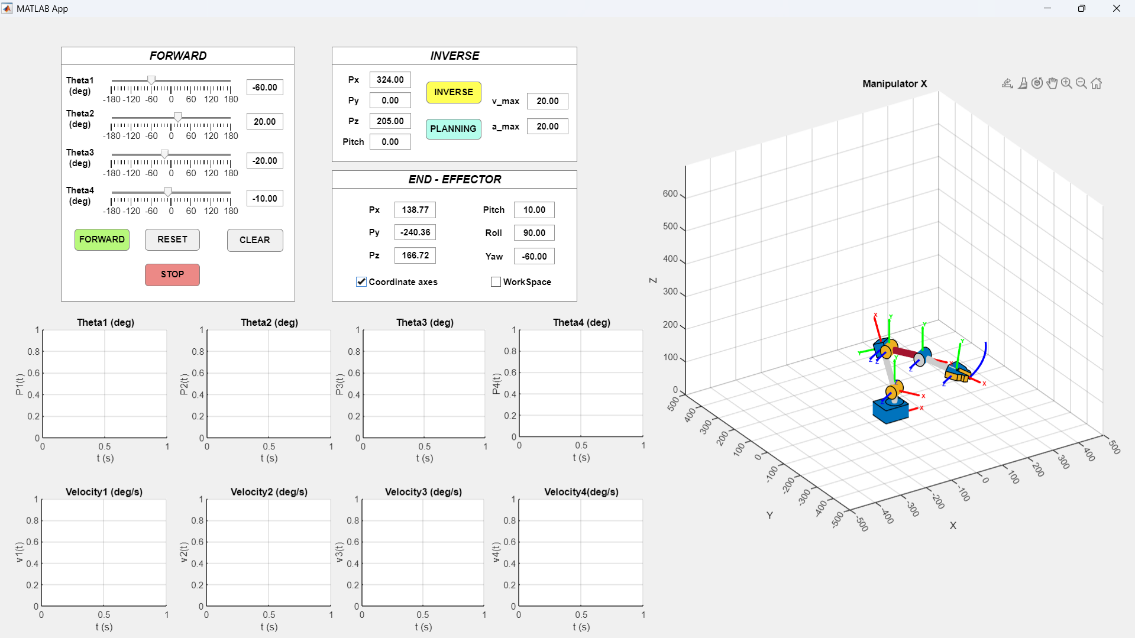
### 2.3. Xử lý Động học ngược và Quy hoạch

Khi nhấn nút PLANNING, quy trình xử lý diễn ra như sau:

* Input: Nhận tọa độ đích và giới hạn
* Inverse Kinematics: Sử dụng phương pháp hình học để tính ra bộ góc khớp đích = []. Kiểm tra điều kiện vùng làm việc (nếu điểm đích nằm ngoài tầm với, hệ thống sẽ báo lỗi).
* Synchronization (Đồng bộ hóa): Tính toán thời gian di chuyển chung dựa trên khớp cần nhiều thời gian nhất.
* Trajectory Generation: Sử dụng vòng lặp để tính toán vị trí góc theo profile hình thang vận tốc (LSPB) cho từng bước thời gian
* Animation: Cập nhật hình ảnh 3D và vẽ đồ thị vận tốc, vị trí.

## 3. Kết quả mô phỏng

### 3.1. Kiểm chứng động học thuận

Thiết lập: Đặt các góc khớp lần lượt là , Nhấn FORWARD Robot sẽ di chuyển và cập nhật và các góc Euler

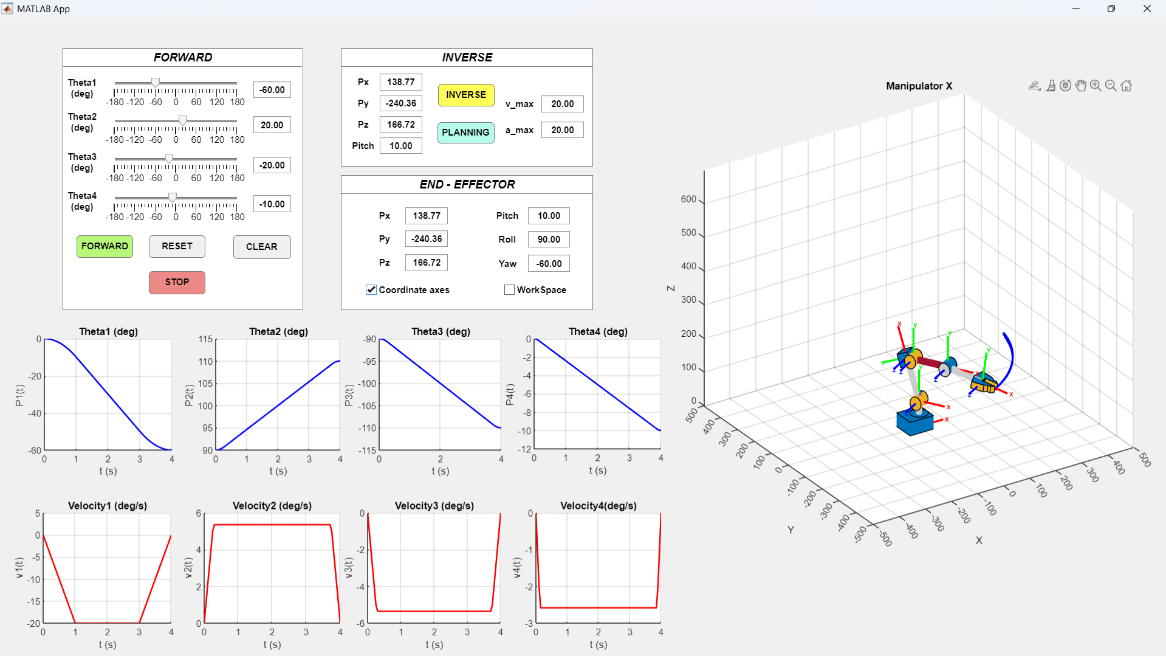
Hình 3.1. Kiểm chứng động học thuận

### 3.2. Kiểm chúng động học ngược

 Thiết lập: Từ vị trí ở hình 3.1.ss đặt các vị trí lần lượt 324, 0, 205 và góc Pitch = 0 là tọa độ ban đầu của robot. Nhấn INVERSE Robot sẽ quay về vị trí ban đầu.

Hình 3.2. Kiểm chứng động học ngược

### 3.3. Kiểm chứng quy hoạch quỹ đạo

Thiết lập: = 20 , = 20 Di chuyển từ vị trí ban đầu đến vị trí ở hình 3.1. Có lần lượt bằng 138.77, -240.36, 166.72 và 10 độ. Nhấn PLANNING Robot sẽ di chuyển đến các ở (3.1) và vẽ các đồ thị vị trí và vận tốc của các khớp

Hình 3.3. Kiểm chứng quy hoạch quỹ đạo

### 3.4. Kết luận chung

Phần mềm đã đáp ứng tốt các yêu cầu của đề tài:

* Mô phỏng chính xác động học của robot 4 bậc tự do.
* Giải quyết thành công bài toán động học ngược với độ chính xác cao.
* Thực hiện quy hoạch quỹ đạo trơn mượt, đảm bảo các ràng buộc về vận tốc và gia tốc, giúp robot vận hành ổn định.

# CHƯƠNG IV. KẾT LUẬN

## 1. Kết luận

Hoàn thành các mục tiêu đề ra:

* Mô hình hóa: Xây dựng chính xác bảng tham số D-H cho robot OpenMANIPULATOR-X (4 bậc tự do).
* Thuật toán: Giải quyết bài toán Động học thuận (FK) và Động học ngược (IK) bằng phương pháp hình học. Quy hoạch quỹ đạo thành công bằng phương pháp LSPB có đồng bộ hóa thời gian.
* Phần mềm: Thiết kế hoàn thiện giao diện GUI trên MATLAB App Designer, cho phép mô phỏng, điều khiển và giám sát các thông số động học trực quan.

## 2. Hạn chế

* Chưa xét đến bài toán động lực học (lực, quán tính, ma sát).
* Hiệu năng phần mềm chưa được tối ưu hóa triệt để, tốc độ phản hồi giao diện còn độ trễ (lag) khi tính toán các quỹ đạo phức tạp hoặc chạy trên cấu hình máy tính thấp.

TÀI LIỆU THAM KHẢO

[1] TS. Nguyễn Hoàng Giáp, *"Bài giảng Kỹ thuật Robot"*, Khoa Điện - Điện tử, Trường Đại học Bách Khoa - ĐHQG TP.HCM, 2025.

[2] ROBOTIS, *“OpenMANIPULATOR-X Manual”,* Tham khảo từ [[link]](https://emanual-robotis-com.translate.goog/docs/en/platform/openmanipulator_x/overview/?_x_tr_sl=en&_x_tr_tl=vi&_x_tr_hl=vi&_x_tr_pto=tc)

[3] H. Z. Ting, M. H. M. Zaman, M. F. Ibrahim, and A. M. Moubark, "Kinematic Analysis for Trajectory Planning of Open-Source 4-DoF Robot Arm", *International Journal of Advanced Computer Science and Applications (IJACSA)*

[4] The MathWorks Inc., "MATLAB App Designer Documentation," Natick, Massachusetts, Tham khảo từ: [[link]](https://www.mathworks.com/help/matlab/app-designer.html)

1. ROBOTIS, *“OpenMANIPULATOR-X Manual”,* Tham khảo từ [[link]](https://emanual-robotis-com.translate.goog/docs/en/platform/openmanipulator_x/overview/?_x_tr_sl=en&_x_tr_tl=vi&_x_tr_hl=vi&_x_tr_pto=tc) [↑](#footnote-ref-1)
2. ROBOTIS, *“OpenMANIPULATOR-X Manual”,* Tham khảo từ [[link]](https://emanual-robotis-com.translate.goog/docs/en/platform/openmanipulator_x/overview/?_x_tr_sl=en&_x_tr_tl=vi&_x_tr_hl=vi&_x_tr_pto=tc) [↑](#footnote-ref-2)
3. H. Z. Ting, M. H. M. Zaman, M. F. Ibrahim, and A. M. Moubark, "Kinematic Analysis for Trajectory Planning of Open-Source 4-DoF Robot Arm," *International Journal of Advanced Computer Science and Applications (IJACSA)* [↑](#footnote-ref-3)
4. H. Z. Ting, M. H. M. Zaman, M. F. Ibrahim, and A. M. Moubark, "Kinematic Analysis for Trajectory Planning of Open-Source 4-DoF Robot Arm," *International Journal of Advanced Computer Science and Applications (IJACSA)* [↑](#footnote-ref-4)