**BỘ GIÁO DỤC & ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT**

**TP. HỒ CHÍ MINH**

**KHOA ĐIỆN – ĐIỆN TỬ**

**-----------------⸙∆⸙-----------------**

****

**BÁO CÁO CUỐI KỲ**

**ĐIỀU KHIỂN THÔNG MINH**

**ĐỀ TÀI: THIẾT KẾ BỘ ĐIỀU KHIỂN PID**

**CHO HỆ PENDUBOT**

**GVHD: TS. Trần Đức Thiện**

**SVTH: Trần Văn Thể 20151567**

**Đoàn Quốc Việt 20151123**

**Nguyễn Phan Quang Phúc 20151151**

**TP. Hồ Chí Minh, ngày 30 tháng 05 năm 2023**

# LỜI CẢM ƠN

Nhóm em xin được bày tỏ lời cảm ơn chân thành đến thầy giáo **TS. Trần Đức Thiện** đã trực tiếp giúp đỡ, hướng dẫn nhóm em hoàn thành đề tài này. Trong suốt quá trình học tập và thực hiện đề tài em luôn được sự quan tâm, hướng dẫn và giúp đỡ tận tình của thầy đã trang bị kiến thức, kỹ năng cần thiết để nhóm thực hiện đề tài: “***Thiết kế bộ điều khiển PID cho hệ pendubot***”

Do trình độ nghiên cứu còn hạn chế và trong quá trình thực hiện đề tài không tránh khỏi những sai sót, hạn chế rất mong thầy bỏ qua. Và nhóm em rất mong nhận được sự quan tâm, góp ý của thầy để nhóm hoàn thiện tốt hơn bài báo cáo đề tài này cũng như rút kinh nghiệm cho bài báo cáo đồ án tốt nghiệp sắp tới.

Cuối cùng xin chúc thầy nhiều sức khoẻ, thành công trong công việc!

Nhóm em xin chân thành cảm ơn!

*TP Hồ Chí Minh, ngày 30 tháng 05 năm 2023*

Sinh viên thực hiện

Trần Văn Thể

Đoàn Quốc Việt

Nguyễn Phan Quang Phúc

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc136286530)

[Danh sách hình ảnh iii](#_Toc136286531)

[Danh sách bảng iv](#_Toc136286532)

[CHƯƠNG 1. TỔNG QUAN 1](#_Toc136286533)

[1.1 ĐẶT VẤN ĐỀ 1](#_Toc136286534)

[1.2 MỤC TIÊU 1](#_Toc136286535)

[1.3 GIỚI HẠN 2](#_Toc136286536)

[1.4 PHƯƠNG PHÁP NGHIÊN CỨU 2](#_Toc136286537)

[1.5 NỘI DUNG NGHIÊN CỨU 2](#_Toc136286538)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 4](#_Toc136286539)

[2.1 MÔ HÌNH TOÁN HỆ PENDOBOT 4](#_Toc136286540)

[2.2 TỔNG QUAN BỘ ĐIỀU PD MỜ 7](#_Toc136286541)

[CHƯƠNG 3. THIẾT KẾ VÀ MÔ PHỎNG BỘ ĐIỀU KHIỂN PD MỜ 9](#_Toc136286542)

[3.1 THIẾT KẾ BỘ ĐIỀU KHIỂN PD MỜ CHO HỆ PENDUBOT 9](#_Toc136286543)

[3.2 MÔ PHỎNG BỘ ĐIỀU KHIỂN FUZZY TRÊN MATLAB SIMULINK 11](#_Toc136286544)

[3.3 MÔ TẢ HỆ MÔ PHỎNG 19](#_Toc136286545)

[3.4 KẾT QUẢ MÔ PHỎNG 21](#_Toc136286546)

[CHƯƠNG 4. ÁP DỤNG GIẢI THUẬT DI TRUYỀN GA TÌM THÔNG SỐ TỐI ƯU CHO BỘ ĐIỀU KHIỂN PID 25](#_Toc136286547)

[4.1 TỔNG QUAN GIẢI THUẬT DI TRUYỀN 25](#_Toc136286548)

[4.2 ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN 25](#_Toc136286549)

[4.3 CẤU TRÚC CỦA GIẢI THUẬT DI TRUYỀN 26](#_Toc136286550)

[4.4 ÁP DỤNG GA TÌM THÔNG SỐ TỐI ƯU CHO BỘ ĐIỀU KHIỂN PID 27](#_Toc136286551)

[4.5 MÔ TẢ HỆ MÔ PHỎNG 29](#_Toc136286552)

[4.6 KẾT QUẢ MÔ PHỎNG 30](#_Toc136286553)

[CHƯƠNG 5. THIẾT KẾ BỘ ĐIỀU KHIỂN THẦN KINH XẤP XỈ BỘ ĐIỀU KHIỂN PID 34](#_Toc136286554)

[5.1 TỔNG QUAN MẠNG THẦN KINH 34](#_Toc136286555)

[5.2 THIẾT KẾ BỘ ĐIỀU KHIỂN MẠNG THẦN KINH CHO HỆ PENDUBOT 35](#_Toc136286556)

[5.2.1 Thu thập dữ liệu: 35](#_Toc136286557)

[5.2.2 Xây dựng mạng thần kinh lan truyền ngược 38](#_Toc136286558)

[5.3 MÔ TẢ HỆ MÔ PHỎNG 40](#_Toc136286559)

[5.4 KẾT QUẢ MÔ PHỎNG 40](#_Toc136286560)

[CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 45](#_Toc136286561)

[6.1 KẾT LUẬN 45](#_Toc136286562)

[6.2 HƯỚNG PHÁT TRIỂN 45](#_Toc136286563)

[TÀI LIỆU THAM KHẢO 46](#_Toc136286564)

# Danh sách hình ảnh

***Chương 2:***

[Hình 2. 1 Hệ pendubot 4](#_Toc136273039)

[Hình 2. 2 Mô hinh động cơ DC 6](#_Toc136273040)

Chương 3:

[Hình 3. 2 Ngõ vào của cho bộ Fuzzy 1 9](#_Toc136273044)

[Hình 3. 3 Ngõ vào cho bộ Fuzzy 2 9](#_Toc136273045)

[Hình 3. 4 Ngõ vào cho bộ Fuzzy 3 10](#_Toc136273046)

[Hình 3. 5 Ngõ vào cho bộ Fuzzy 4 10](#_Toc136273047)

[Hình 3. 6 Ngõ ra của bộ Fuzzy 1 10](#_Toc136273048)

[Hình 3. 7 Ngõ ra của bộ Fuzzy 2 10](#_Toc136273049)

[Hình 3. 8 Ngõ ra của bộ Fuzzy 3 11](#_Toc136273050)

[Hình 3. 9 Ngõ ra của bộ Fuzzy 4 11](#_Toc136273051)

[Hình 3. 10 Chọn khối Add 11](#_Toc136273052)

[Hình 3. 11 Chọn Khối Sum 12](#_Toc136273053)

[Hình 3. 12 Chọn khối Gain 12](#_Toc136273054)

[Hình 3. 13 Chọn khối Fuzzy 13](#_Toc136273055)

[Hình 3. 14 Chọn khối Subsystem 13](#_Toc136273056)

[Hình 3. 15 Mô phỏng phương trình vi phân cho hệ pendubot 13](#_Toc136273057)

[Hình 3. 16 Sơ đồ Simulink mô phỏng hệ pendubot dùng logic mờ 14](#_Toc136273058)

[Hình 3. 17 Khối PD\_Fuzzy1 15](#_Toc136273059)

[Hình 3. 18 Khối PD\_Fuzzy2 15](#_Toc136273060)

[Hình 3. 19 Chọn kiểu thiết kế Sugeno 16](#_Toc136273061)

[Hình 3. 20 Khai báo hàm liên thuộc ngõ vào 16](#_Toc136273062)

[Hình 3. 21 Khai báo hàm liên thuộc ngõ ra 17](#_Toc136273063)

[Hình 3. 22 Khai báo hàm liên thuộc ngõ vào 17](#_Toc136273064)

[Hình 3. 23 Khai báo hàm liên thuộc ngõ ra 18](#_Toc136273065)

[Hình 3. 24 Khai báo hàm liên thuộc ngõ vào 18](#_Toc136273066)

[Hình 3. 25 Khai báo hàm liên thuộc ngõ ra 19](#_Toc136273067)

[Hình 3. 26 Khai báo hàm liên thuộc ngõ ra 19](#_Toc136273068)

[Hình 3. 27 Khai báo hàm liên thuộc ngõ ra 20](#_Toc136273069)

[Hình 3. 28 Tạo các quy tắc giữ cân bằng hệ con lắc ngược 20](#_Toc136273070)

[Hình 3. 29 Thông số Laptop 21](#_Toc136273071)

[Hình 3. 30 Chọn chế độ mô phỏng Fixed – Step và thời gian lấy mẫu 0.01 21](#_Toc136273072)

[Hình 3. 31 Chọn Current Folder có chứa file Fis 22](#_Toc136273073)

[Hình 3. 32 Đường dẫn khối Fuzzy 1 22](#_Toc136273074)

[Hình 3. 33 Đường dẫn khối Fuzzy 2 22](#_Toc136273075)

[Hình 3. 34 Đường dẫn khối Fuzzy 3 23](#_Toc136273076)

[Hình 3. 35 Đường dẫn khối Fuzzy 4 23](#_Toc136273077)

[Hình 3. 36 Kết quả mô phỏng góc link 1 của bộ điều khiển PD mờ trong 10 giây (rad) 24](#_Toc136273078)

[Hình 3. 37 Kết quả mô phỏng góc link 2 của bộ điều khiển PD mờ trong 10 giây (rad) 24](#_Toc136273079)

[Hình 3. 38 Kết quả mô phỏng vận tốc link 1 của bộ điều khiển PD mờ trong 10 giây (rad/s) 25](#_Toc136273080)

[Hình 3. 39 Kết quả mô phỏng vận tốc link 2 của bộ điều khiển PD mờ trong 10 giây (rad/s) 25](#_Toc136273081)

[Hình 3. 40 Kết quả mô phỏng điện áp ngõ vào u (V) 26](#_Toc136273082)

Chương 4:

[Hình 4. 1 Lưu đồ giải thuật di truyền 28](#_Toc136273479)

[Hình 4.2 Thông số Laptop 32](#_Toc136273480)

[Hình 4.3 Chọn chế độ mô phỏng Fixed – Step và thời gian lấy mẫu 0.01 32](#_Toc136273481)

[Hình 4.4 Chọn thời gian mô phỏng 32](#_Toc136273482)

[Hình 4.5 Sự thay đổi hàm thích nghi qua các thế hệ 33](#_Toc136273483)

[Hình 4.6 Chương trình GA dừng lại sau khi đạt số thế hệ tối đa 33](#_Toc136273484)

[Hình 4.7 Thông số cần tìm sau khi chạy GA 34](#_Toc136273485)

[Hình 4.8 Kết quả mô phỏng hệ pendobot với thông số sau khi chạy GA 35](#_Toc136273486)

Chương 5:

[Hình 5. 1 Lưu đồ giải thuật quá trình xây dựng mạng thần kinh nhân tạo 36](#_Toc136275007)

[Hình 5. 2 Hệ thống Pendubot với bộ điều khiển PID 37](#_Toc136275008)

[Hình 5. 3 Thu thập dữ liệu cho 2 khối PID 38](#_Toc136275009)

[Hình 5. 4 Khối Random Number 1 38](#_Toc136275010)

[Hình 5. 5 Khối Saturation 1 39](#_Toc136275011)

[Hình 5. 6 Khối Random Number 2 39](#_Toc136275012)

[Hình 5. 7 Khối Saturation 2 40](#_Toc136275013)

[Hình 5. 8 Hàm S lưỡng cực 41](#_Toc136275014)

[Hình 5. 9 Hàm tuyến tính 41](#_Toc136275015)

[Hình 5. 10 Cấu trúc mạng thần kinh nhân tạo 41](#_Toc136275016)

[Hình 5. 11 Thông số Laptop 43](#_Toc136275017)

[Hình 5. 12 Chọn chế độ mô phỏng Fixed – Step và thời gian lấy mẫu 0.01 43](#_Toc136275018)

[Hình 5. 13 Chọn thời gian mô phỏng 43](#_Toc136275019)

[Hình 5. 14 Giao diện thể hiện các thông số của mạng thần kinh 44](#_Toc136275020)

[Hình 5. 15 Khối nơ-ron sau khi được huấn luyện 44](#_Toc136275021)

[Hình 5. 16 Tín hiệu đáp ứng của khối PID1 và 44](#_Toc136275022)

[Hình 5. 17 Tín hiệu đáp ứng của khối PID2 và 45](#_Toc136275023)

[Hình 5. 18 Tín hiệu ngõ ra q1 với bộ điều khiển PID và mạng thần kinh 46](#_Toc136275024)

[Hình 5. 19 Tín hiệu ngõ ra q2 với bộ điều khiển PID và mạng thần kinh 46](#_Toc136275025)

# Danh sách bảng

***Chương 2:***

[Bảng 2. 1 Thông số hệ pendubot 4](#_Toc136275533)

[Bảng 2. 2 Mô hình động cơ DC 6](#_Toc136275534)

[Bảng 2. 3 Thông số vật lý hệ xe pendubot 7](#_Toc136275535)

***Chương 5:***

[Bảng 5. 1 Đáp ứng tín hiệu ngõ ra điều khiển bởi mạng thần kinh 47](#_Toc136275567)

[Bảng 5. 2 Đáp ứng tín hiệu ngõ ra điều khiển bởi mạng thần kinh 47](#_Toc136275568)

# CHƯƠNG 1. TỔNG QUAN

## ĐẶT VẤN ĐỀ

Điều khiển thông minh là một lĩnh vực nghiên cứu tương đối mới ở Việt Nam thu hút được sự chú ý đáng kể trong suốt thời gian qua với sự tiến bộ của công nghệ. Điều khiển thông minh là một lĩnh vực giàu cơ hội và nhiều hướng đi đối với các ngành liên quan và một trong số đó là ngành điều khiển tự động để cải thiện hiệu suất tối ưu nhất cho các dây chuyền sản xuất.

Trong lĩnh vực kỹ thuật điều khiển, “*hệ pendubot*” là một trong những lĩnh vực được nghiên cứu phổ biến và đồng thời nó còn là hình mẫu cho việc kiểm soát ổn định hệ thống. Ngày nay, mặc dù đã có nhiều công trình nghiên cứu về hệ thống này nhưng các nhà khoa học vẫn không ngừng thí nghiệm trên hệ pendubot như một bước đệm để điều khiển ổn định cân bằng các hệ thống phức tạp hơn như robot thăng bằng, tàu vũ trụ, máy bay không người lái.

Và trong điều khiển thông minh có khá nhiều bộ điều khiển để ổn định hệ thống như bộ điều khiển kinh điển mờ, LQR, LQG, Sliding Mode, Fuzzy. Trong đó bộ điều khiển PID có khả năng thích ứng với các hệ phi tuyến khá tốt, cũng như tính trực quan của nó. Với hệ xe con lắc ngược có bản chất là phi tuyến, cùng với đó có khá nhiều chuyên gia trong lĩnh vực điều khiển thông minh đã thành công trong việc xây dựng hệ mờ cho hệ pendubot. Và đó là lí do nhóm em chọn đề tài **“Thiết kế bộ điều khiển PID cho hệ pendubot”.**

## MỤC TIÊU

Hệ pendubot sẽ dễ bị mất ổn định ngay cả khi nhiễu tác động là nhỏ nhất. Vì thế mục tiêu của đề tài là thiết kế bộ điều khiển PID điều khiển cho hệ đứng yên tại vị trí đặt. Kết hợp với đó nhóm sẽ sử dụng thêm giải thuật di truyền để tìm ra các thông số PID tối ưu nhất cho bộ điều khiển.

## GIỚI HẠN

Đề tài chỉ dừng lại ở việc mô phỏng, chưa thực hiện thực tế nên chưa điều chỉnh được thông số để phù hợp với yêu cầu. Ngoài ra hệ thống chỉ đáp ứng tốt khi 2 thanh không lệch quá so với phương thẳng đứng. Nếu lệch quá hệ thống sẽ mất cân bằng.

## PHƯƠNG PHÁP NGHIÊN CỨU

* Dựa trên các đề tài nghiên cứu có trên sách, báo, tạp chí khoa học. Từ đó nghiên cứu và phát triển đề tài. Khảo sát các mô hình hệ pendubot để hiểu nguyên lí hoạt động của hệ thống.
* Sử dụng Matlab/Simulink hoặc các phần mềm khác có thể mô phỏng được để chạy mô phỏng hệ thống.
* Nhóm sử dụng phương pháp thiết kế Fuzzy dựa vào kinh nghiệm chuyên gia để thiết kế hệ thống và sử dụng giải thuật GA để tìm thông số tối ưu cho bộ điều khiển PID, và sau cùng là thiết kế bộ điều khiển thần kinh xấp xỉ bộ điều khiển PID. Cùng đó với kinh nghiệm đã được học ở học phần lý thuyết “*Điều khiển thông minh*” để thiết kế có chọn lọc.

## NỘI DUNG NGHIÊN CỨU

* **Chương 2: Cơ sở lý thuyết**

Chương này trình bày lý thuyết về mô hình toán hệ pendubot và lý thuyết bộ điều khiển PID.

* **Chương 3: Thiết kế và mô phỏng bộ điều khiển PD mờ**

Chương này trình bày về các bước thiết kế bộ điều khiển mờ dựa vào kinh nghiệm chuyên gia và các bước thiết kế mô hình trên Simulink. Sau đó tiến hành mô phỏng hệ thống.

* **Chương 4: Sử dụng giải thuật di truyền tìm thông số tối ưu bộ điều khiển PID**

Chương này sẽ nói tổng quan về giải thuật di truyền GA và sau đó tiến hành lập trình giải thuật này trên Matlab để tìm thông số tối ưu cho bộ điều khiển PID. Sau đó tiến hành mô phỏng hệ thống.

* **Chương 5: Thiết kế bộ điều khiển thần kinh xấp xỉ bộ điều khiển mờ**

Chương này sẽ nói tổng quan về mạng thần kinh nơ-ron và cách thiết kế bộ điều khiển thần kinh xấp xỉ bộ điều khiển PID. Sau đó tiến hành mô phỏng hệ thống.

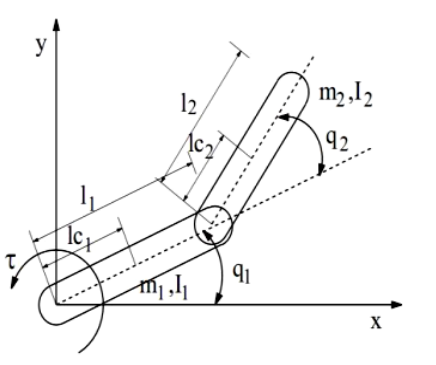
* **Chương 6: Kết luận và hướng phát triển**

Chương này sẽ nói về mục tiêu đã đạt được, kết quả điều khiển sử dụng bộ điều khiển PID, giải thuật di truyền GA và bộ điều khiển thần kinh. Nêu những vấn đề đề tài đã giải quyết được và hạn chế. Trình bày khả năng mở rộng đề tài và hướng phát triển.

# CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

## MÔ HÌNH TOÁN HỆ PENDOBOT

Mô hình hệ pendubot là mô hình có ngõ vào điều khiển ít hơn số bậc tự do có độ phi truyến cao và rất khổ để điểu khiển. Pendubot có cấu trúc cơ khí không quá phức tạp nên được nhiều nhà nghiên cứu sử dụng để kiểm tra giải phẩu điều khiển trong các phòng thí nghiêm. Sau đây là mô hình toán học của hệ:



Hình 2. Hệ pendubot

|  |  |
| --- | --- |
| Biến | Giải thích |
|  | Góc link 1 |
|  | Góc link 2 |
|  | Độ dài link 1 |
|  | Khoảng cách từ tọa độ động cơ tới tâm link1 |
|  | Độ dài link 2 |
|  | Khoảng cách từ tọa độ encoder của link 1 tới tâm link 2 |
|  | Khối lượng link 1 |
|  | Khối lượng link 2 |
|  | Mô men quá tính link 1 |
|  | Mô men quá tính link 2 |
|  | Gia tốc trọng trường |
|  | Mô ment trên link 1 |
|  | Mô men trên link 2 |

Bảng 2. Thông số hệ pendubot

Phương trình động học của hệ thống là:





Với:











Trong bài báo cáo này nhóm em điều khiển hệ Pendubot ở vị trí phía trên (Top)





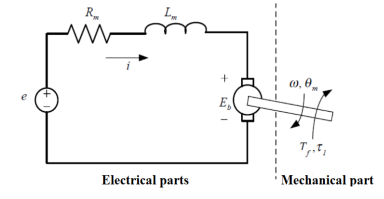




Trong thử nghiệm thực tế, kiểm soát đầu vào tín hiệu là điện áp áp dụng cho động cơ để tạo khoảnh khắc. Vì vậy, nó là cần thiết sang hệ thống tương đương thành một dạng mới mà đầu vào điều khiển tín hiệu là điện áp.

Từ (2.1) và hình 2.2, ngõ ra của động cơ có thể được tính như sau:





Bảng 2. Mô hình động cơ DC

Trong đó:

: moment của động cơ (Nm)

e: điện áp trên động cơ (V)

: vận tốc góc của động cơ (rad/s)

Từ (2.1) đến (2.12), qua một vài phép tính toán ta được:









Với:















|  |  |  |
| --- | --- | --- |
| Biến | Mô tả | Giá trị |
|  | Độ dài link 1 | 0.146(m) |
|  | Khoảng cách từ tọa độ động cơ tới tâm link1 | 0.055(m) |
|  | Độ dài link 2 | 0.253(m) |
|  | Khoảng cách từ tọa độ encoder của link 1 tới tâm link 2 | 0.149(m) |
|  | Khối lượng link 1 | 1.068(kg) |
|  | Khối lượng link 2 | 0.07(kg) |
|  | Mô men quá tính link 1 | 0.0189(kg.m2) |
|  | Mô men quá tính link 2 | 0.000409(kg.m2) |
|  | Gia tốc trọng trường | 9.80665(m/) |

Bảng 2. Thông số vật lý hệ xe pendubot

## TỔNG QUAN BỘ ĐIỀU KHIỂN PD MỜ

Hệ pendubot không ổn định nếu không được điều khiển, tức các khớp của hệ không điều khiển và toàn bộ hệ pendubot sẽ tạo với mặt đất một góc 90. Để giữ ổn định hệ thống này ta phải điều khiển động cơ duy nhất ở phần gốc của hệ theo cơ chế điều chỉnh liên tục để duy trì thẳng đứng con lắc.

Một nghiên cứu cho thấy 90% các bộ điều khiển được sử dụng hiện nay là bộ điều khiển PD. Tuy bộ điều khiển PD có khả năng điều khiển hệ thống với chất lượng quá độ tốt( đáp ứng nhanh, độ vọt lố thấp), nhưng bộ điều khiển PD không thể đảm bảo chất lượng điều khiển tại mọi điểm làm việc khi đối tượng điều khiển phi tuyến. Ta có thể hiểu PD mờ theoo nghĩa tín hiệu ra của bộ điều khiển tỷ lệ phi tuyến với tín hiệu vào, với tích phân tín hiệu vào và vi phân tín hiệu vào theo quy lật xác định bởi hệ quy tắc mờ.

**Trình tự thiết kế**

1. Xác định biến vào, biến ra (và biến trạng thái, nếu cần) của đối tượng.
2. Chuẩn hóa biến vào, biến ra về miền giá trị hay để sau này có thể lập trình dễ dàng bằng vi xử lí hoặc tích hợp vào các hệ PLC.
3. Định nghĩa các tập mờ trên tập cơ sở đã chuẩn hóa của các biến, và gán cho mỗi tập mờ một giá trị ngôn ngữ. Số lượng, vị trí và hình dạng của các tập mờ tùy thuộc vào từng ứng dụng cụ thể.
4. Gán các quan hệ giữa các tập mờ ở ngõ vào và ngõ ra, bước này xây dựng được hệ quy tắc mờ.
5. Chọn phương pháp suy diễn. Trong thực tế người ta thường chọn phương pháp duy diễn cục bộ nhằm đơn giản trong việc tính toán và áp dụng công thức hợp thành MAX-MIN hay MAX-PROD
6. Chọn phương pháp giải mờ. Trong điều khiển người ta thường chọn phương pháp giải mờ như trọng tâm, phương pháp trung bình trọng số.

Việc thiết kế bộ điều khiển mờ trực tiếp mang tính thử sai vì việc chọn số lượng, hình dạng, vị trí tập mờ, chọn hệ quy tắc đều phụ thuộc vào kinh nghiệm của người thiết kế. Ngay cả khi có thể chọn tập mờ và hệ quy tắc mờ từ quan hệ giữa ngõ vào và ngõ ra thì việc thiết kế cũng mang tính thử sai.

Tóm lại, bộ điều khiển mờ là bộ điều khiển dựa trên cở sở toán học là lý thuyết tập mờ và logic mờ. Bộ điều khiển mờ bắt chước sự xử lí thông tin và điều khiển của con người, vì vậy thích hợp để điều khiển những đối tượng phức tạp mà các phương pháp kinh điển không cho được kết quả mong muốn. Phương pháp thiết kế bộ điều khiển mờ mang nặng tính thử sai, vì vậy nếu người thiết kế không có kinh nghiệm thì rất khó có thể tổng hợp được bộ điều khiển mờ cho kết quả tốt.

# CHƯƠNG 3. THIẾT KẾ VÀ MÔ PHỎNG BỘ ĐIỀU KHIỂN PD MỜ

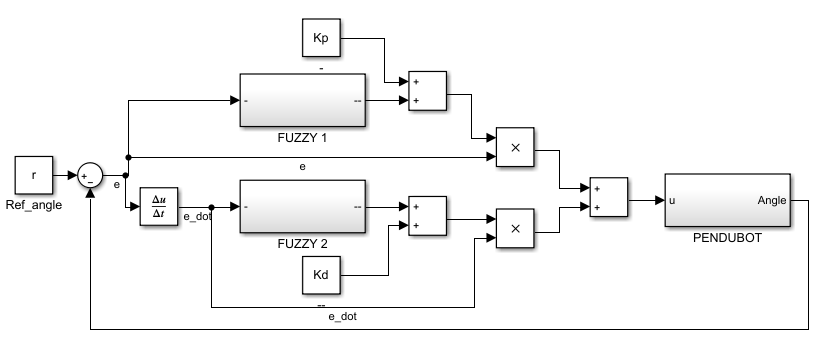
## THIẾT KẾ BỘ ĐIỀU KHIỂN PD MỜ CHO HỆ PENDUBOT

Áp dụng phương pháp thiết kế đã trình bày trong mục trước ta có trình tự thiết kế sau:

1. Chọn các biến vào và ra: Ta chọn bốn biến vào là , trong đó lần lượt là sai số của góc thanh 1 và thanh 2 tại vị trí cân bằng, là đạo hàm của  với = set-point , =set-point , biến ra là lần lượt là ngõ ra của bộ Fuzzy1, Fuzzy2, Fuzzy3, Fuzzy4.
2. Tín hiệu điều khiển là . Trong đó  là thông số của bộ điều khiển PD và là hệ số được chọn bởi kinh nghiệm và các giá trị được xác định như sau:

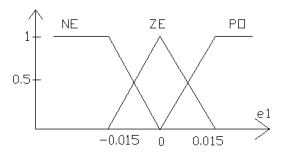


1. Chọn phương pháp suy diễn MAX-PROD.
2. Sơ đồ tổng quan của bộ điều khiển

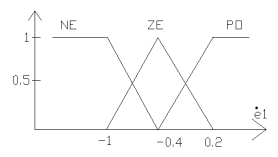


Hình 3. 1 Sơ đồ tổng quan của bộ điều khiển

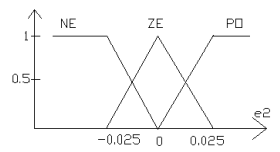
1. Các tập hợp đầu vào và đầu ra mờ được biểu thị như sau:



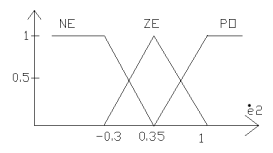
Hình 3. Ngõ vào của cho bộ Fuzzy 1



Hình 3. Ngõ vào cho bộ Fuzzy 2

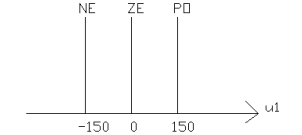


Hình 3. Ngõ vào cho bộ Fuzzy 3

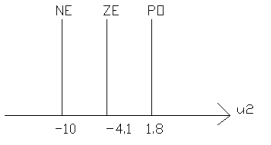


Hình 3. Ngõ vào cho bộ Fuzzy 4

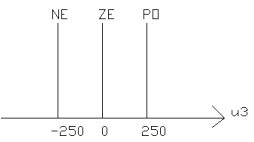
**Các tập mờ ngõ ra:**



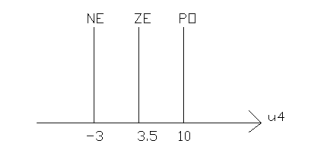
Hình 3. Ngõ ra của bộ Fuzzy 1



Hình 3. Ngõ ra của bộ Fuzzy 2



Hình 3. Ngõ ra của bộ Fuzzy 3



Hình 3. Ngõ ra của bộ Fuzzy 4

1. Hệ quy tắc mờ: Số luật mờ được sử dụng trong bộ điều khiển là 3 cho mỗi bộ điều khiển. Một số quy tắc được liệt kê dưới đây:

Nếu ngõ vào là NE thì ngõ ra là NE

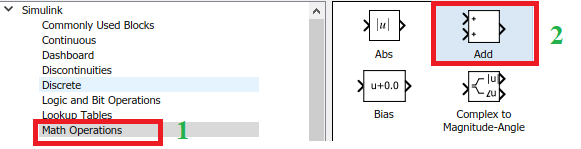
Nếu ngõ vào là ZE thì ngõ ra là ZE

Nếu ngõ vào là PO thì ngõ ra là PO

## MÔ PHỎNG BỘ ĐIỀU KHIỂN FUZZY TRÊN MATLAB SIMULINK

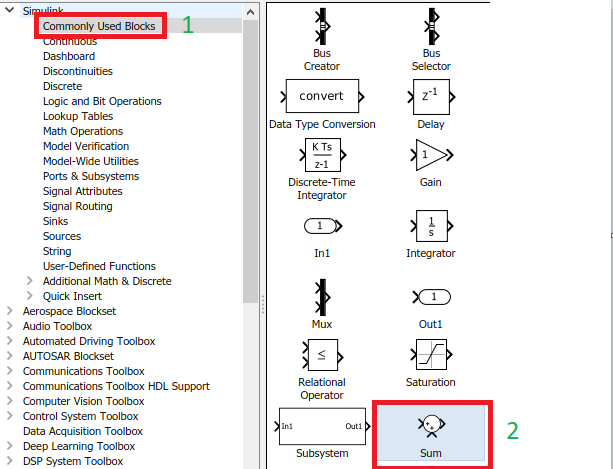
* Bước 1: Lấy các khối cần thiết để mô phỏng hệ thống.

Vào Math *Operations 🡺 Chọn khối Add* để tổng hợp tín hiệu đặt.



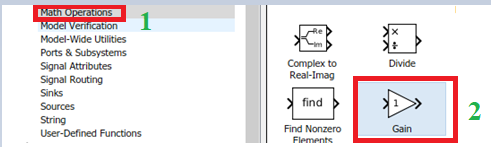
Hình 3. Chọn khối Add

Vào *Commonly🡺 Chọn Sum* để so sánh tín hiệu đặt và tín hiệu q hồi tiếp.



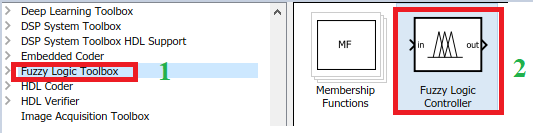
Hình 3. Chọn Khối Sum

Vào *Math operation🡺 chọn Gain* để khai báo các thông số của bộ điều khiển PD.



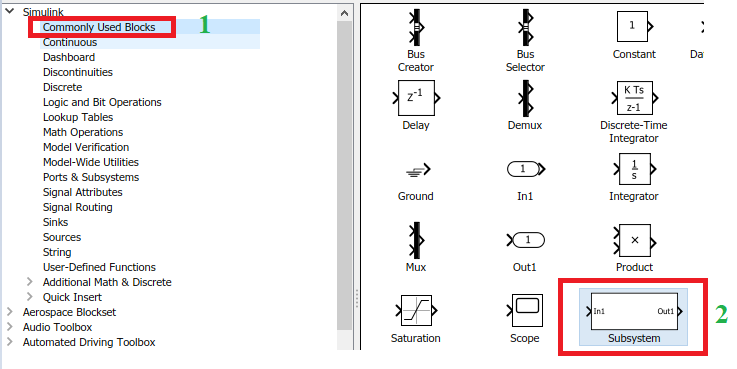
Hình 3. Chọn khối Gain

Vào Fuzzy Logic Toolbox🡺 Chọn Fuzzy Logic Controler để thêm bộ điều khiển Fuzzy.

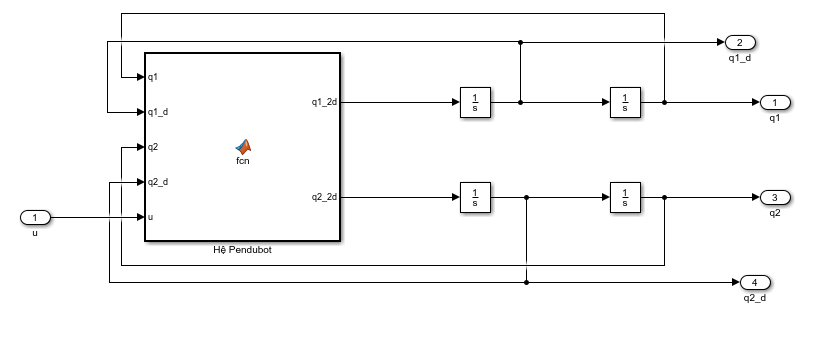


Hình 3. Chọn khối Fuzzy

Vào *Commonly Used Block🡺 Chọn Subsystem* để thiết kế hệ xe con lắc ngược.



Hình 3. Chọn khối Subsystem



Hình 3. Mô phỏng phương trình vi phân cho hệ pendubot

***Chương trình bên trong khối vi phân Pendubot***

**function** [q1\_2d,q2\_2d] = fcn(q1,q1\_d,q2,q2\_d,u)

m1=1.068;

m2=0.07;

l1=0.146;

l2=0.253;

lc1=0.055;

lc2=0.149;

I1=0.0189;

I2=0.000409;

Kb=0.0864;

Kt=0.0864;

Rm=3.7;

Cm=0.000415;

Jm=0.000427;

g=9.80665;

K1=Kt/Rm;

K2=((Kt\*Kb)/Rm)+Cm;

K3=Jm;

beta1=m1\*lc1^2+m2\*l1^2+I1;

beta2=m2\*lc2^2+I2;

beta3=m2\*l1\*lc2;

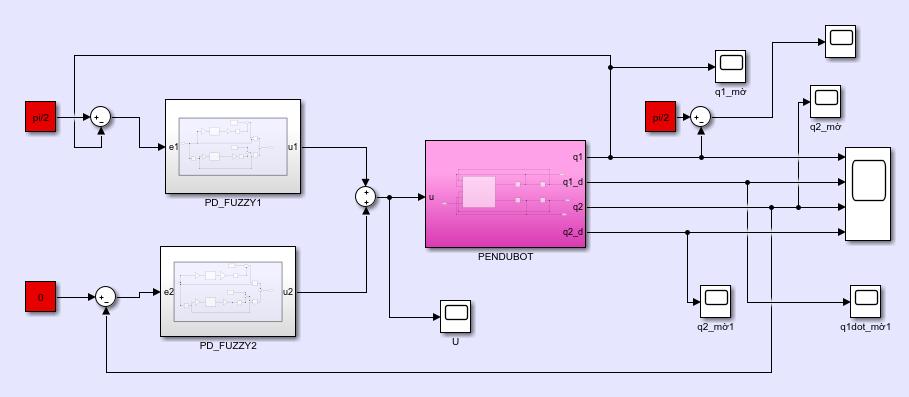
beta4=m1\*lc1+m2\*l1;

beta5=m2\*lc2;

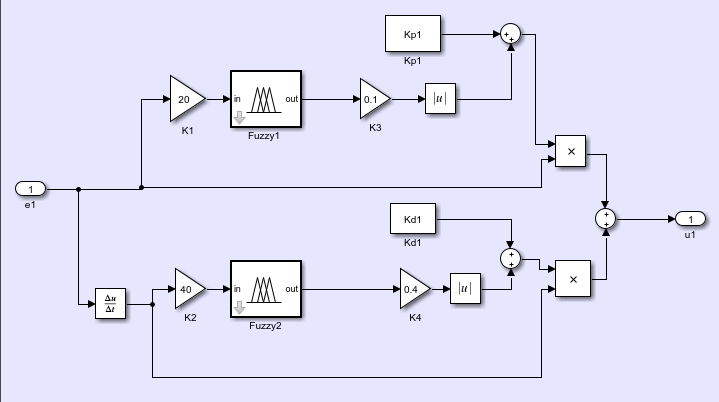
**q1\_2d**=1/(beta1\*beta2-beta3^2\*(cos(q2))^2+K3\*beta2)\*(beta2\*beta3\*sin(q2)\*(q1\_d+q2\_d)^2+beta3^2\*cos(q2)\*sin(q2)\*q1\_d^2-beta2\*beta4\*g\*cos(q1)+beta3\*beta5\*g\*cos(q2)\*cos(q1+q2)+beta2\*K1\*u-K2\*beta2\*q1\_d);

**q2\_2d**=-(beta5\*g\*cos(q1 + q2)\*(K3 + beta1 + beta3\*cos(q2)) - beta4\*g\*cos(q1)\*(beta2 + beta3\*cos(q2)) - K2\*q1\_d\*(beta2 + beta3\*cos(q2)) + beta3\*sin(q2)\*(q1\_d + q2\_d)^2\*(beta2 + beta3\*cos(q2)) + K3\*beta3\*q1\_d^2\*sin(q2) + beta3\*q1\_d^2\*sin(q2)\*(beta1 + beta3\*cos(q2)))/(- beta3^2\*cos(q2)^2 + K3\*beta2 + beta1\*beta2) - (K1\*u\*(beta2 + beta3\*cos(q2)))/(- beta3^2\*cos(q2)^2 + K3\*beta2 + beta1\*beta2);

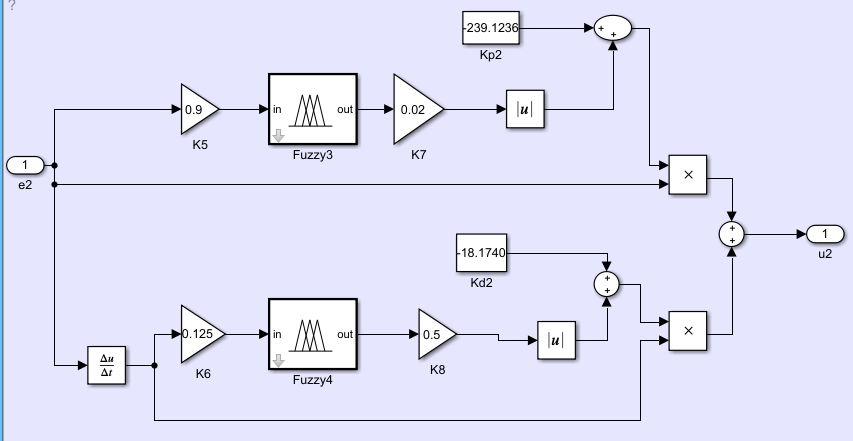
* Bước 3: Tiến hành kết nối các khối



Hình 3. Sơ đồ Simulink mô phỏng hệ pendubot dùng logic mờ



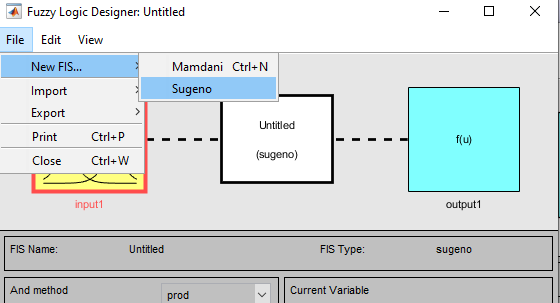
Hình 3. Khối PD\_Fuzzy1



Hình 3. Khối PD\_Fuzzy2

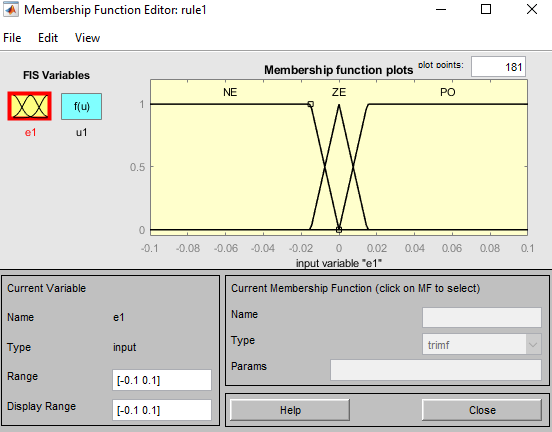
* Bước 4: Xây dựng bộ điều khiển mờ trên Fuzzy Toolbox

*Chọn thiết kế theo kiểu* Sugeno

**

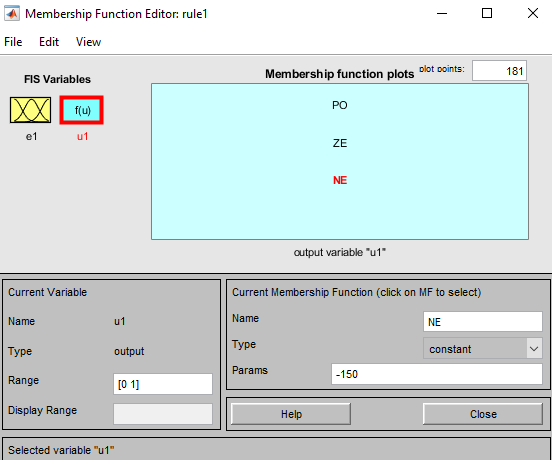
Hình 3. Chọn kiểu thiết kế Sugeno

*Khai báo hàm liên thuộc ngõ vào* 



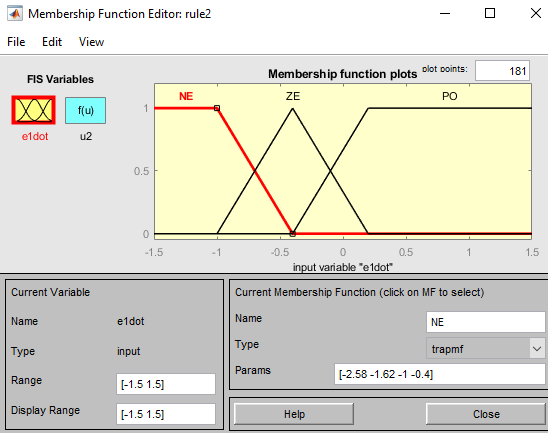
Hình 3. Khai báo hàm liên thuộc ngõ vào

*Khai báo hàm liên thuộc ngõ ra* 



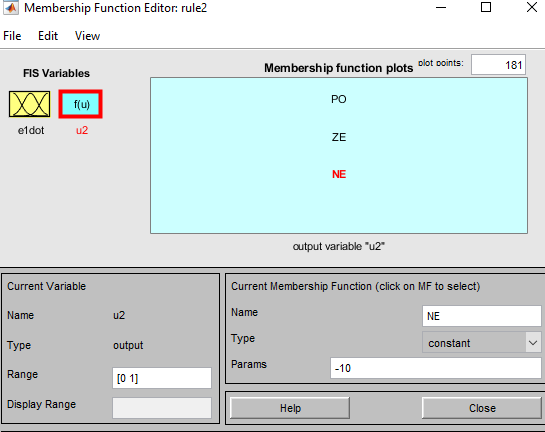
Hình 3. Khai báo hàm liên thuộc ngõ ra

*Khai báo hàm liên thuộc ngõ vào* 



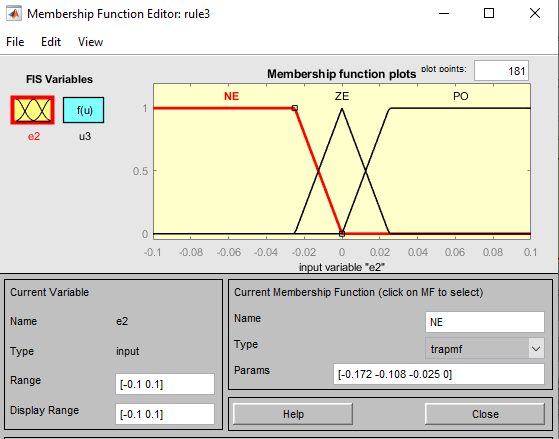
Hình 3. Khai báo hàm liên thuộc ngõ vào

*Khai báo hàm liên thuộc ngõ ra* 



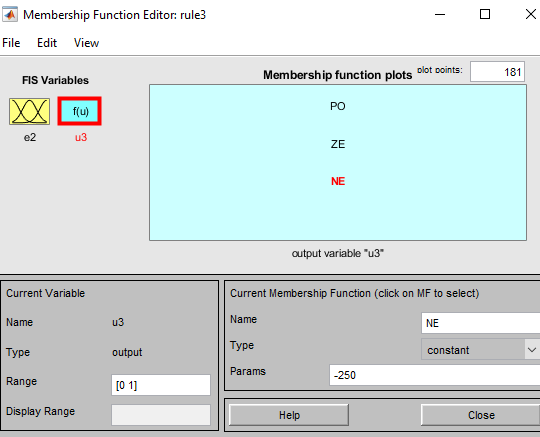
Hình 3. Khai báo hàm liên thuộc ngõ ra

*Khai báo hàm liên thuộc ngõ vào* 

**

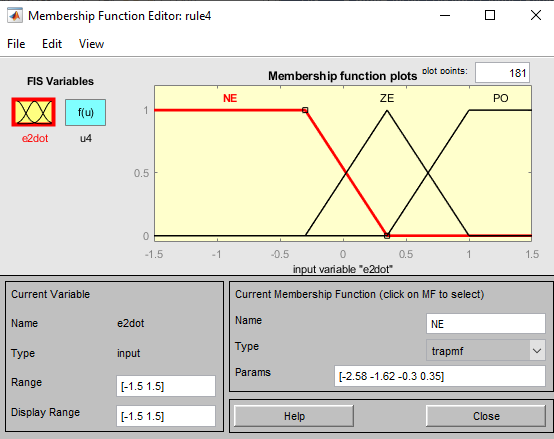
Hình 3. Khai báo hàm liên thuộc ngõ vào

*Khai báo hàm liên thuộc ngõ ra* 



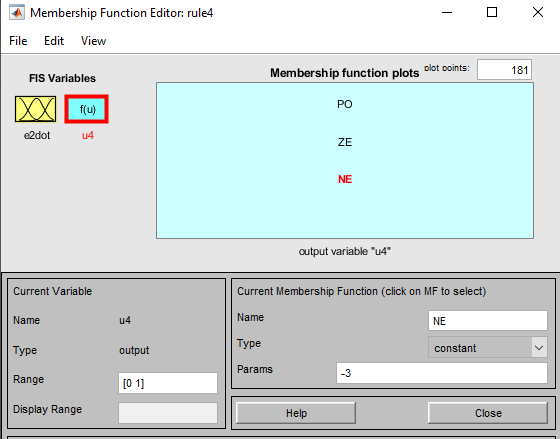
Hình 3. Khai báo hàm liên thuộc ngõ ra

*Khai báo hàm liên thuộc ngõ vào* 



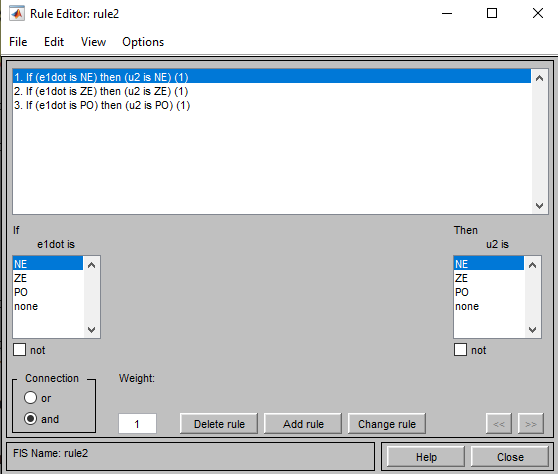
Hình 3. Khai báo hàm liên thuộc ngõ ra

*Khai báo hàm liên thuộc ngõ vào* 



Hình 3. Khai báo hàm liên thuộc ngõ ra

*Vào Rule để tạo 3 luật mờ cho từng bộ Fuzzy, mỗi bộ 3 luật như nhau*

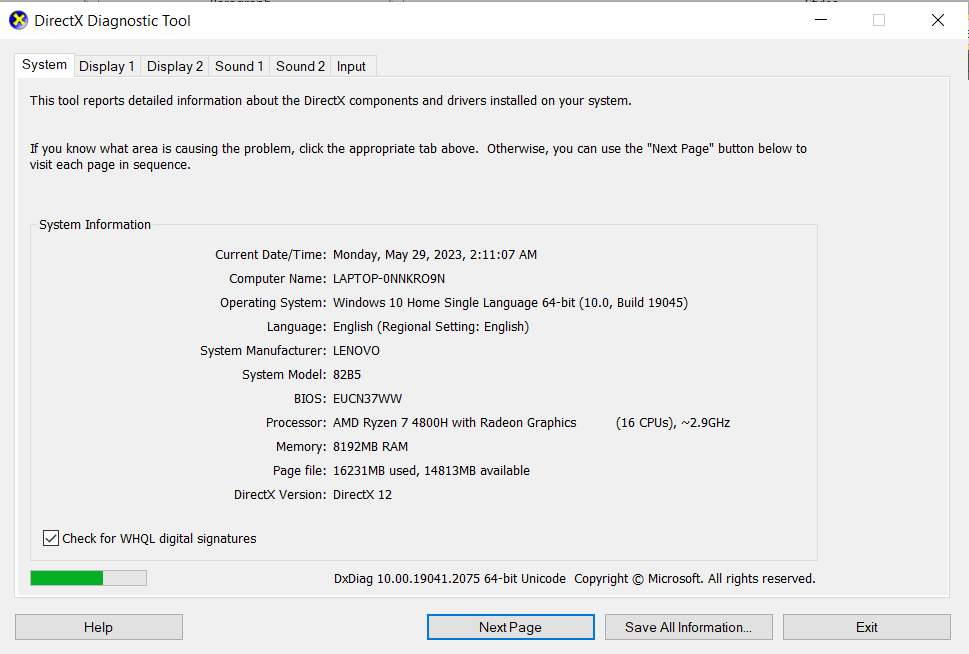


Hình 3. Tạo các quy tắc giữ cân bằng hệ con lắc ngược

Tương tự cho 3 bộ Fuzzy còn lại

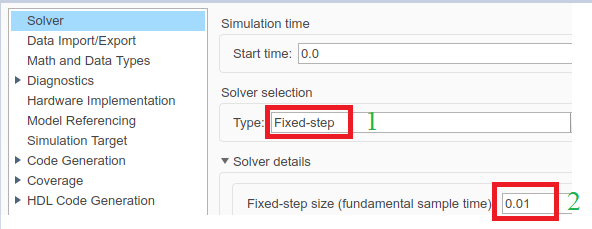
## MÔ TẢ HỆ MÔ PHỎNG

* Nhóm sử dụng Laptop Lenovo Legion 5, core ryzen 7, phiên bản MATLAB R2022b. Thời gian mô phỏng 10 giây. Thời gian lấy mẫu là 0.01.

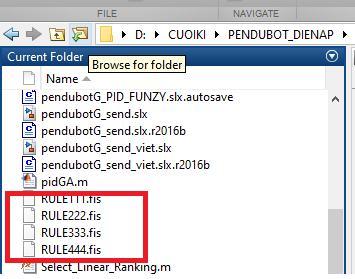


Hình 3. Thông số Laptop

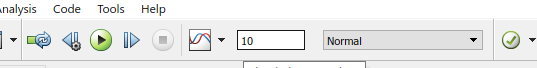
* Chế độ mô phỏng Fixed – Step và thời gian lấy mẫu 0.01.



Hình 3. Chọn chế độ mô phỏng Fixed – Step và thời gian lấy mẫu 0.01

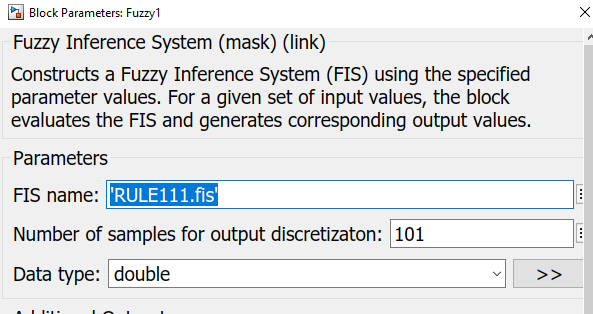
 

Hình 3. Chọn Current Folder có chứa file Fis



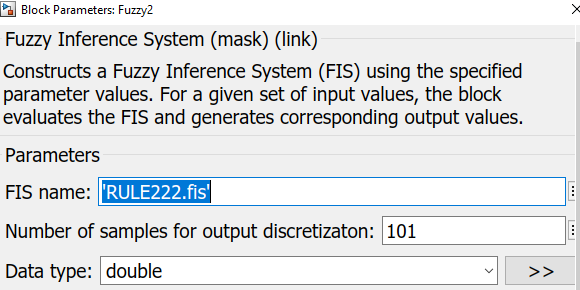
Hình 3.31 Chọn thời gian mô phỏng

* Khai báo đường dẫn file Fis trong khối Fuzzy Logic Controller:



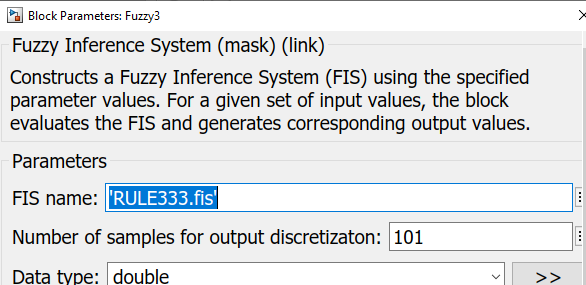
Hình 3. Đường dẫn khối Fuzzy 1

* Khối Fuzzy2



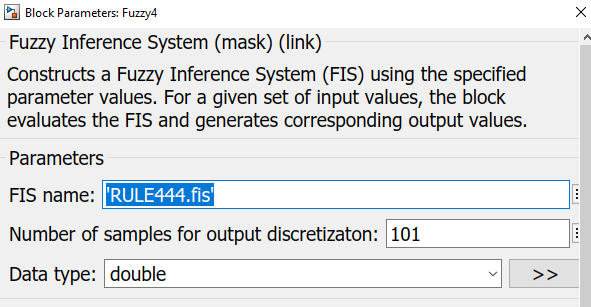
Hình 3. Đường dẫn khối Fuzzy 2

* Khối Fuzzy3



Hình 3. Đường dẫn khối Fuzzy 3

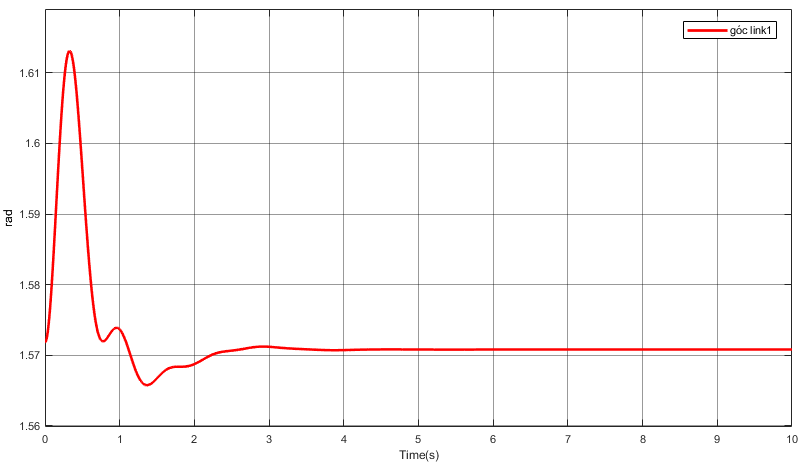
* Khối Fuzzy4



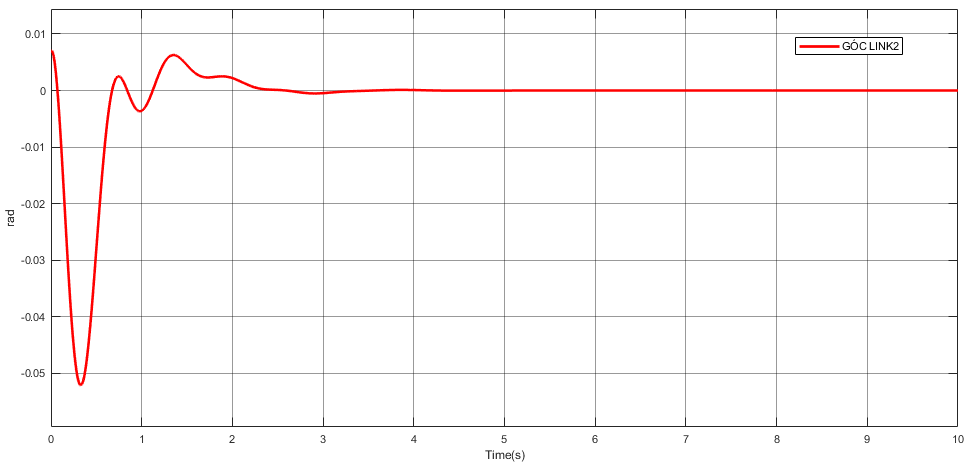
Hình 3. Đường dẫn khối Fuzzy 4

## KẾT QUẢ MÔ PHỎNG

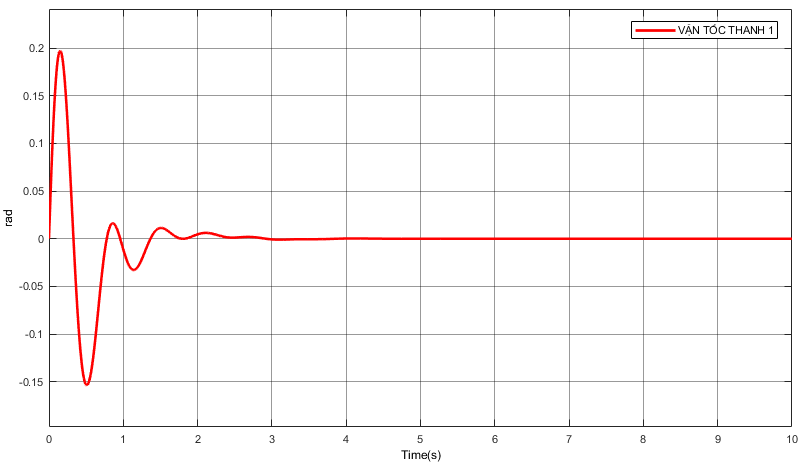
Dưới đây là kết quả mô phỏng hệ pendubot ở vị trí Top:



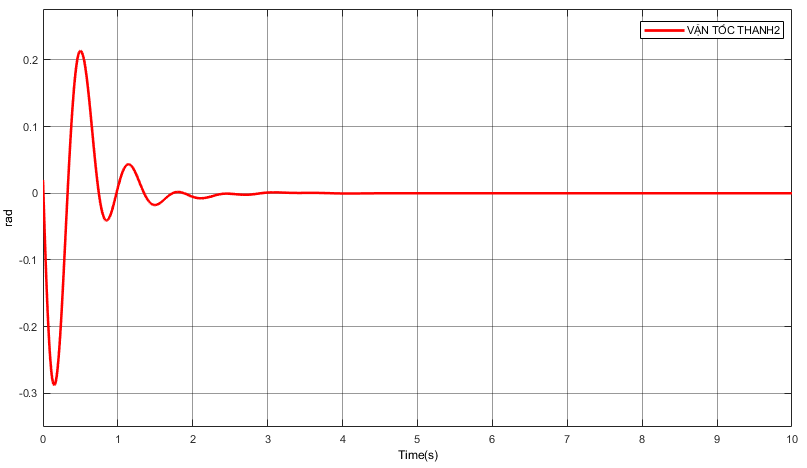
Hình 3. Kết quả mô phỏng góc link 1 của bộ điều khiển PD mờ trong 10 giây (rad)



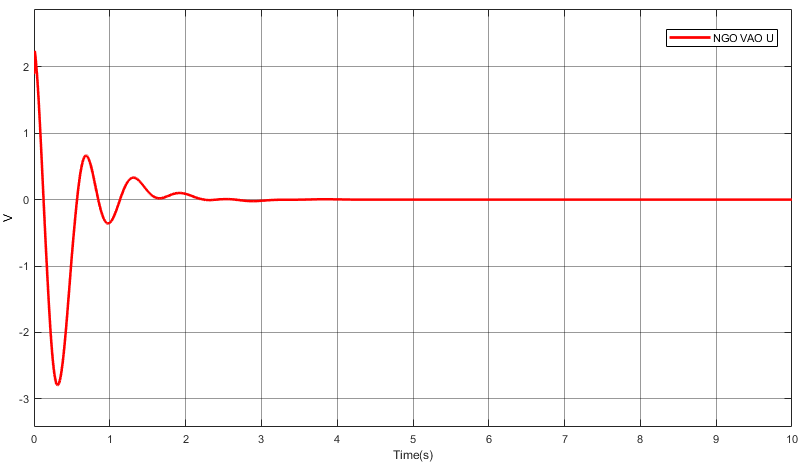
Hình 3. Kết quả mô phỏng góc link 2 của bộ điều khiển PD mờ trong 10 giây (rad)



Hình 3. Kết quả mô phỏng vận tốc link 1 của bộ điều khiển PD mờ trong 10 giây (rad/s)



Hình 3. Kết quả mô phỏng vận tốc link 2 của bộ điều khiển PD mờ trong 10 giây (rad/s)



Hình 3. Kết quả mô phỏng điện áp ngõ vào u (V)

**Nhận xét:** Tại vị trí ban đầu đã đặt cho thanh 1 và thanh 2 như ở trên theo hình 3.30 sau khoảng 3s link1 ổn định tại vịtrí 1.571(rad). Và trong quá trình điều khiển thanh 1 giao động trong khoảng từ 1.56 -1.62 (rad). Theo hình Hình 3.32, thanh 2 giao động trong khoảng từ -0.051 – 0.0062(rad). Và sau 3.1 giây thanh 2 ổn định và cách đặt vị trí thanh 1 theo phương thẳng đứng hướng lên và vị trí thanh 2 theo phương của thanh 1. Dựa vào kết quả mô phỏng, hệ thống đã ổn định được với bộ điềukhiển PD mờ ở vị trí Top.

Ta thấy hệ thống thanh 1 bắt đầu dao động thì chiều của thanh 1 ngược so với thanh 2 giữ ổn định cho thanh 1 không rơi khỏi vị trí cân bằng

Khi hệ thống ổn định thì tín hiệu điện áp u trả về giá trị 0.

*Như vậy bộ điều khiển PD Fuzzy đã được thiết kế thành công.*

# CHƯƠNG 4. ÁP DỤNG GIẢI THUẬT DI TRUYỀN GA TÌM THÔNG SỐ TỐI ƯU CHO BỘ ĐIỀU KHIỂN PID

## TỔNG QUAN GIẢI THUẬT DI TRUYỀN

Giải thuật di truyền (Gientic Algorithm – GA) được Holland đưa ra vào năm 1975 là giải thuật tìm kiếm lời giải tối ưu trên nguyên tắc phỏng theo quá trình tiến hóa và quy luật di truyền của sinh vật trong tự nhiên. Bản chất toán học của GA là giải thuật tìm kiếm theo xác suất.

Quá trình tiến hóa thể hiện tính tối ưu chỗ, thế hệ sau bao giờ cũng tốt hơn (phát triển hơn, hoàn thiện hơn, thế hệ trước. Tiến hóa tự nhiên được duy trì nhờ hai quá trình cơ bản sinh sản và chọn lọc tự nhiên, các thế hệ mới luôn được sản sinh ra để bổ sung, thay thế thế hệ cũ. Có thể nào phát triển hơn, thích ứng hơn với môi trường sẽ có nhiều khả năng tồn tại và phát triển, có thể nào không thích ứng được với môi trường sẽ bị đào thải. Sự thay đổi của môi trường là động lực của quá trình tiến hóa. Ngược lại, tiết hóa cũng tác động trở lại góp phần làm thay đổi môi trường.

Di truyền và đột biến là hai cơ chế có vai trò quan trọng như nhau trong tiến trình tiến hóa, dù rằng hiện tượng đột biến diễn ra với xác suất nhỏ hơn nhiều so với di truyền. GA nói riêng và các giải thuật tiến hóa nói chung tuy có điểm khác biệt nhưng đều mô phỏng ba quá trình cơ bản của tiến hóa tự nhiên: chọn lọc tự nhiên, lai ghép (sinh sản) và đột biến.

GA là phương pháp tối ưu hóa ngẫu nhiên tổng quát dùng để giải các bài toán tìm kiếm. GA có một số khác biệt so với giải thuật tối ưu hóa và tìm kiếm thông thường. Thứ nhất, GA làm việc với quần thể gồm nhiều NST (tập hợp nhiều lời giải), tìm kiếm nhiều điểm cực trị cùng một lúc. GA là phương pháp tìm kiếm lời giải tối ưu toàn cục. Thứ hai, GA làm việc với chuỗi ký hiệu (chuỗi NST) chứ không làm việc với các thông số. Thứ ba, GA chỉ cần đánh giá hàm mục tiêu để định hướng quá trình tìm kiếm mà không cần bất cứ thông tin gì khác.

## ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

GA là thuật toán tổng quát có khả năng giải quyết thành công các bài toán ở nhiều lĩnh vực, đặc biệt là những bài toán khó giải quyết bằng các phương pháp khác. Những năm gần đây, các ứng dụng của GA tăng mạnh trong nhiều lĩnh vực như: kỹ thuật (thiết kế động cơ, thiết kế máy bay...), tối ưu hóa, hoạch định hoạt động robot, hệ thống phân loại, hệ thống học, nhận dạng mẫu, huấn luyện mạng thần kinh, chỉnh định hệ mờ...

Lưu đồ GA dùng để giải bài toán tối ưu trong điều khiển được trình bày ở hình 30. Để áp dụng GA giải bài toán tối ưu trước hết phải mã hóa (encode) lời giải của bài toán thành chuỗi NST. Mỗi NST đại diện cho một cá thể trong quần thể. Qua quá trình chọn lọc tự nhiên những cá thể thích nghi nhất với môi trường sống (có độ thích nghi cao) mới tồn tại và có cơ hội sinh sản để tạo ra thế hệ con có xu hướng thích nghi với môi trường sống tốt hơn thế hệ bố mẹ. Cá thể nào có độ thích nghi càng cao thì càng có nhiều cơ hội để tồn tại và bắt cặp với một cá thể khác để sinh ra thế hệ con. Nhờ quá trình lại ghép mà các đặc tính tốt của thế hệ trước được truyền lại cho thế hệ sau.

## CẤU TRÚC CỦA GIẢI THUẬT DI TRUYỀN



Hình 4. Lưu đồ giải thuật di truyền

**Các thông số khởi tạo cho giải thuật di truyền**

* *Max\_generation* (Số lượng thế hệ tối đa): sau khi lai tạo và phân tích, khi thế hệ hiện tại vượt qua số lượng thế hệ tối đa thì chương trình sẽ dừng lại và đưa ra kết quả.
* *Max\_stall\_generation* (Số cá thể giống nhau): khi số lượng cá thể giống nhau bằng giá trị đặt thì chương trình dừng.
* *Epsilon* (Ngưỡng): khi giá trị hàm thích nghi J nhỏ hơn ngưỡng thì chương trình dừng lại.
* *Pop\_size* (Số lượng cá thể cha, mẹ trong quần thể): khởi tạo số cá thể cha mẹ cho quần thể. Kích thước quần thể là một trong những tùy chọn quan trọng nhất và có thể đóng vai trò quyết định trong nhiều ứng dụng. Nếu kích thước quần thể quá nhỏ, GA có thể hội tụ quá nhanh; ngược lại nếu kích thước quần thể quá lớn, GA sẽ hội tụ chậm và có thể lãng phí tài nguyên máy tính.
* *Npar* (Số lượng nhiễm sắc thể trong quần thể): cần tìm bao thông số cho hệ thống thì npar bằng bấy nhiêu.
* *Range* (Khoảng giá trị của các nhiễm sắc thể): khai báo khoảng giá trị cho phép của thông số cần tìm.
* *Dec* (Số các số nguyên*)*: khai báo số các số nguyên của các thông số cần tìm.
* *Sig (*Số chữ số thập phân): khai báo số các số thập phân có nghĩa sau dấu phẩy của các thông số cần tìm.
* *Crossover* (Xác xuất lai ghép): là tỉ lệ thế hệ con sau khi lai ghép giống với bố mẹ.
* *Mutation* (Xác xuất đột biến): vai trò đột biến là khôi phục chất liệu di truyền bị mất hoặc chưa khai phá để ngăn hiện tượng GA hội tụ sớm. Tuy nhiên không phải đột biến nào cũng có lợi.

## ÁP DỤNG GA TÌM THÔNG SỐ TỐI ƯU CHO BỘ ĐIỀU KHIỂN PID

1. **Chương trình**

Ở đây nhóm em chạy 100 thế hệ, pop\_size=20, npar =6, Crossover=0.6, Mutation=0.4.

**Phần code**

clc;

clear all

rand ('state', sum(100\*clock));

max\_generation=100; % có 250 lần chạy trong quá trình

max\_stall\_generation=100; % 50 thế hệ giống nhau thì dừng

epsilon=0.0001; % J chuẩn

pop\_size=20; % số lượng cá thể cha mẹ trong quần thể

npar = 6; % có 5 nhiễm sắc thể trong 1 cá thể

range= [ 0 0 0 0 0 0 ;...

1000 1000 1000 1000 1000 1000]; % giá trị K1 K2 K3 K4 K5 nằm trong khoảng 0-100

dec= [3 3 3 3 3 3]; % có 2 chữ số nguyên

sig= [4 4 4 4 4 4]; % có 5 chữ số thập phân

cross\_prob = 0.6; % xác xuất lai ghép

mutate\_prob = 0.4; % xác xuất đột biến

elitism = 1; % luôn giữ lại giá trị tốt nhất trong khi lai tạo

rho=0.02; % trọng số quyết định cái nào quan trọng với J hơn

par=Init(pop\_size,npar,range); % khai tạo 35 cá thể cha me đầu tiên

Terminal=0; % khởi động các giá trị đầu tiên trước khi chạy GA

generation = 0; % khởi động các giá trị đầu tiên trước khi chạy GA

stall\_generation=0; % khởi động các giá trị đầu tiên trước khi chạy GA

for pop\_index=1:pop\_size,

Kp1=par(pop\_index,1)-1000;

Ki1=par(pop\_index,2) -1000;

Kd1=par(pop\_index,3) -1000;

Kp2=par(pop\_index,4) -1000;

Ki2=par(pop\_index,5) -1000;

Kd2=par(pop\_index,6) -1000;

sim('pendubotG'); % bắt đầu mô phỏng

if length(e1)>9500 % lai ghép thiệt

Kp1

Ki1

Kd1

Kp2

Ki2

Kd2

J=e1'\*e1+ e2'\*e2;

fitness(pop\_index)=1/(J+eps); % tìm cực tiểu hàm thích nghi

else

J= J ^100;

fitness(pop\_index)=1/(J+eps);

end

end;

[bestfit0,bestchrom]=max(fitness);

Kp1=par(bestchrom,1)-1000; %các cá thể cha mẹ đầu tiên

Ki1=par(bestchrom,2) -1000; %các cá thể cha mẹ đầu tiên

Kd1=par(bestchrom,3) -1000; %các cá thể cha mẹ đầu tiên

Kp2=par(bestchrom,4) -1000; %các cá thể cha mẹ đầu tiên

Ki2=par(bestchrom,5) -1000; %các cá thể cha mẹ đầu tiên

Kd2=par(bestchrom,6)-1000; %các cá thể cha mẹ đầu tiên

J0=1/bestfit0+0.001

while ~Terminal

generation = generation+1;

disp(['generation #' num2str(generation) ' of maximum ' num2str(max\_generation)]);

pop=Encode\_Decimal\_Unsigned(par,sig,dec); % mã hóa thập phân (NST của các cha mẹ)

parent=Select\_Linear\_Ranking(pop,fitness,0.2,elitism,bestchrom);

child=Cross\_Twopoint(parent,cross\_prob,elitism,bestchrom);% lai ghép 2 điểm

pop=Mutate\_Uniform(child,mutate\_prob,elitism,bestchrom);% đột biến theo dạng phân bố đều

par=Decode\_Decimal\_Unsigned(pop,sig,dec);% giải mã kết quả về lại NST

for pop\_index=1:pop\_size, % đánh giá lại độ thích nghi các cá thể sau tiến hóa

Kp1=par(bestchrom,1)-1000; % quy đổi giá trị NST về Kp1

Ki1=par(bestchrom,2)-1000; % quy đổi giá trị NST về Ki1

Kd1=par(bestchrom,3)-1000; % quy đổi giá trị NST về Kd1

Kp2=par(bestchrom,4)-1000; % quy đổi giá trị NST về Kp2

Ki2=par(bestchrom,5)-1000; % quy đổi giá trị NST về Ki2

Kd2=par(bestchrom,6)-1000; % quy đổi giá trị NST về Kd2

sim('pendubotG'); % tiến hành chạy mô phỏng để kiểm tra

if length(e1)>9500

Kp1

Ki1

Kd1

Kp2

Ki2

Kd2

J=e1'\*e1+ e2'\*e2;

fitness(pop\_index)=1/(J+eps); % tìm cực tiểu hàm thích nghi

else

J= J ^100;

fitness(pop\_index)=1/(J+eps);

end

end;

[bestfit(generation),bestchrom]=max(fitness);

if generation == max\_generation % điều kiện dừng

Terminal = 1;

elseif generation>1,

if abs(bestfit(generation)-bestfit(generation-1))<epsilon,

stall\_generation=stall\_generation+1;

if stall\_generation == max\_stall\_generation, Terminal = 1;end

else

stall\_generation=0;

end;

end;

end

plot(1./bestfit) % vẽ đồ thị hàm thích nghi

Kp1=par(bestchrom,1) -1000 % hiển thị NST Kp1

Ki1=par(bestchrom,2) -1000 % hiển thị NST Ki1

Kd1=par(bestchrom,3) -1000 % hiển thị NST Kd1

Kp2=par(bestchrom,4) -1000 % hiển thị NST Kp2

Ki2=par(bestchrom,5) -1000 % hiển thị NST Ki2

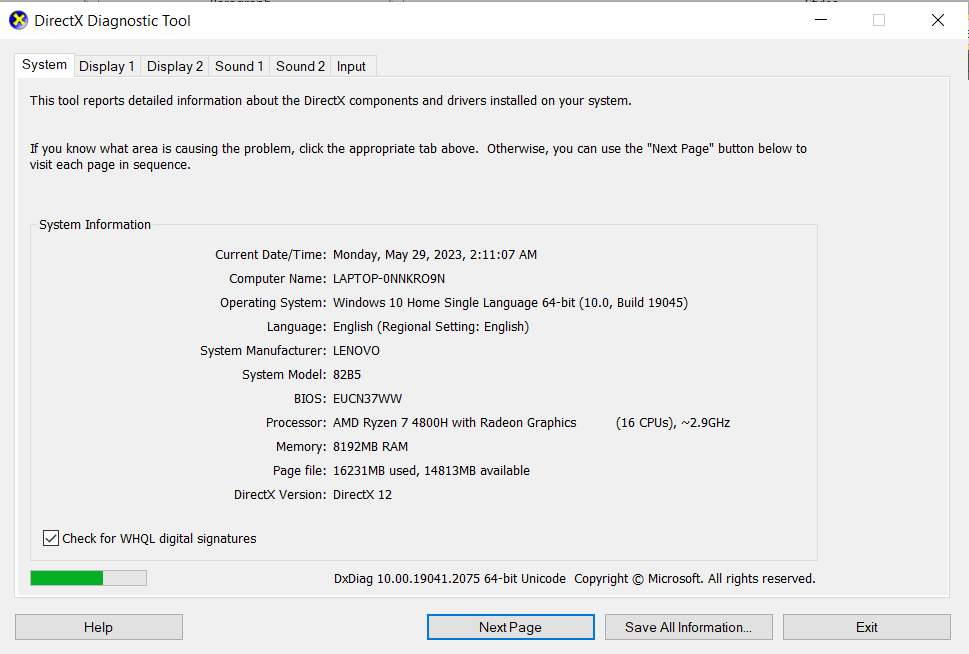
Kd2=par(bestchrom,5)-1000 % hiển thị NST Kd2

J=1/bestfit(end) % hàm tiêu chuẩn tương ứng với cá thể con tốt nhất này

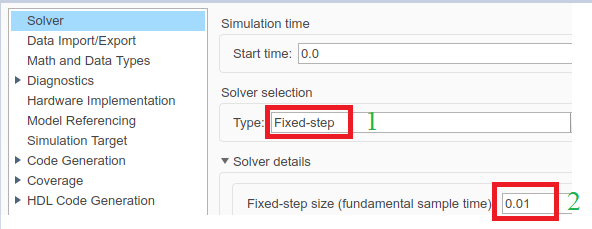
sim('pendubotG'); %tien hành mô phỏng lại

## MÔ TẢ HỆ MÔ PHỎNG

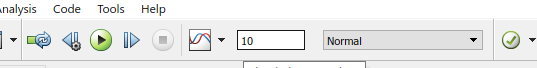
* Nhóm sử dụng Laptop Lenovo Legion 5, core ryzen 7, phiên bản MATLAB R2022b. Thời gian mô phỏng 10 giây. Thời gian lấy mẫu là 0.01.



Hình 4. Thông số Laptop



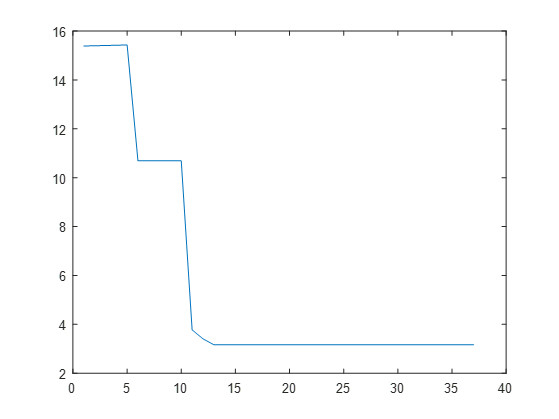
Hình 4. Chọn chế độ mô phỏng Fixed – Step và thời gian lấy mẫu 0.01

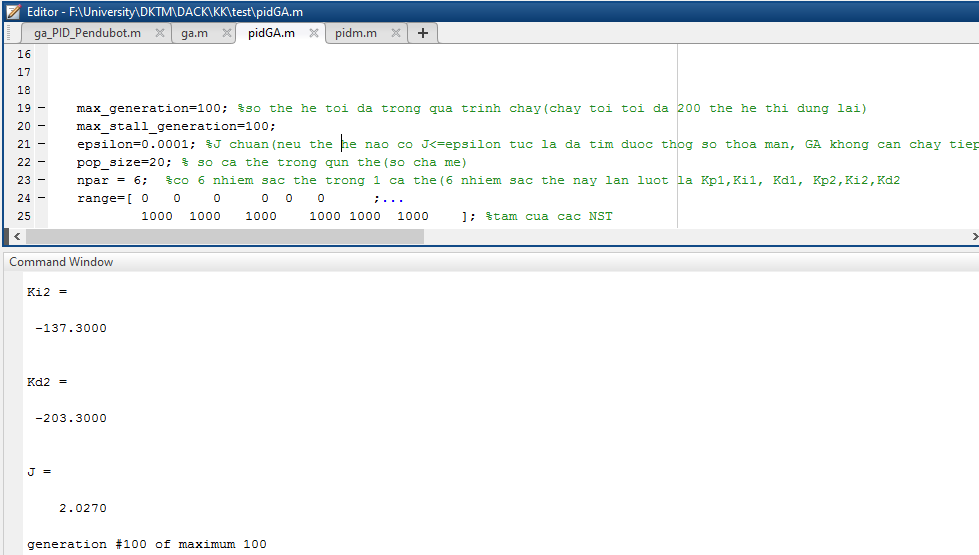


Hình 4. Chọn thời gian mô phỏng

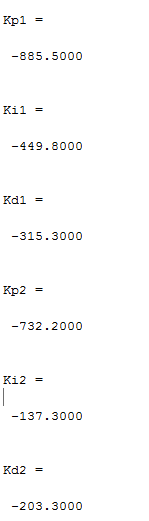
## KẾT QUẢ MÔ PHỎNG

Dưới đây là kết quả sau khi chạy mô phỏng:

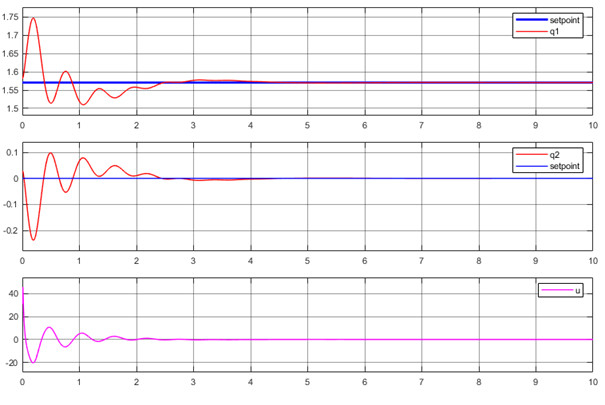


Hình 4. Sự thay đổi hàm thích nghi qua các thế hệ

Hình 4. Chương trình GA dừng lại sau khi đạt số thế hệ tối đa



Hình 4. Thông số cần tìm sau khi chạy GA



Hình 4. Kết quả mô phỏng hệ pendobot với thông số sau khi chạy GA

Kết quả mô phỏng cho thấy ban đầu khi cho  và đặt giá trị q1 mong muốn là  thì khi bắt đầu mô phỏng  sẽ tăng lên và góc và giảm xuống dao động tắt dần,  sẽ giảm xuống sau đó tăng lên và cũng dao động tắt dần, sau khoảng thời gian tầm 2.5(giây) thì  đạt được vị trí đặt  và góc  cũng về 0, tức là cân bằng so với phương thẳng đứng.

Khi ta tiếp tục cho  về vị trí ban đầu 0(m) ở thời gian 5(giây) thì đáp ứng hệ thống vẫn rất tốt, hệ thống vẫn giữ được trạng thái cân bằng ở vị trí đặt. So với thông số cũ thì thông số mới của bộ điều khiển PID GA tìm được đáp ứng nhanh hơn bộ điều khiển mờ., nhưng tín hiệu điều khiển không tốt bằng bộ điều khiển mờ, xuất hiện nhiều dao động.

Nhưng tổng quan thông số mới này đáp ứng khá tốt về ngõ ra vị trí và góc lệch con lắc hầu như giống bộ điều khiển fuzzy. *Như vậy giải thuật GA đã tìm được thông số tối ưu của bộ điều khiển PID thành công.*

# CHƯƠNG 5. THIẾT KẾ BỘ ĐIỀU KHIỂN THẦN KINH XẤP XỈ BỘ ĐIỀU KHIỂN PID

## TỔNG QUAN MẠNG THẦN KINH

Bộ điều khiển mạng thần kinh được tạo ra là dựa vào cơ chế truyền nhận và xử lý tín hiệu của bộ não con người và nó hoạt động như là một mạng tính toán phân bố song song. Việc xây dựng bộ điều khiển thần kinh chính là tạo ra một mạng thần kinh nhân tạo bắt chước quá trình làm việc của bộ não con người. Trái với các mô hình tính toán thông thường, hầu hết các mạng thần kinh phải được huấn luyện trước khi sử dụng. Từ nguyên lý trên, nhóm sẽ trình bày các bước thiết kế mạng thần kinh sao chép bộ điều khiển mờ để điều khiển cho hệ pendubot.



Hình 5. Lưu đồ giải thuật quá trình xây dựng mạng thần kinh nhân tạo

Việc thiết kế bộ điều khiển mạng thần kinh nhân tạo mang tính thử sai vì việc chọn số lớp ẩn, tốc độ học, thu thập dữ liệu… đều phụ thuộc vào kinh nghiệm của người thiết kế.

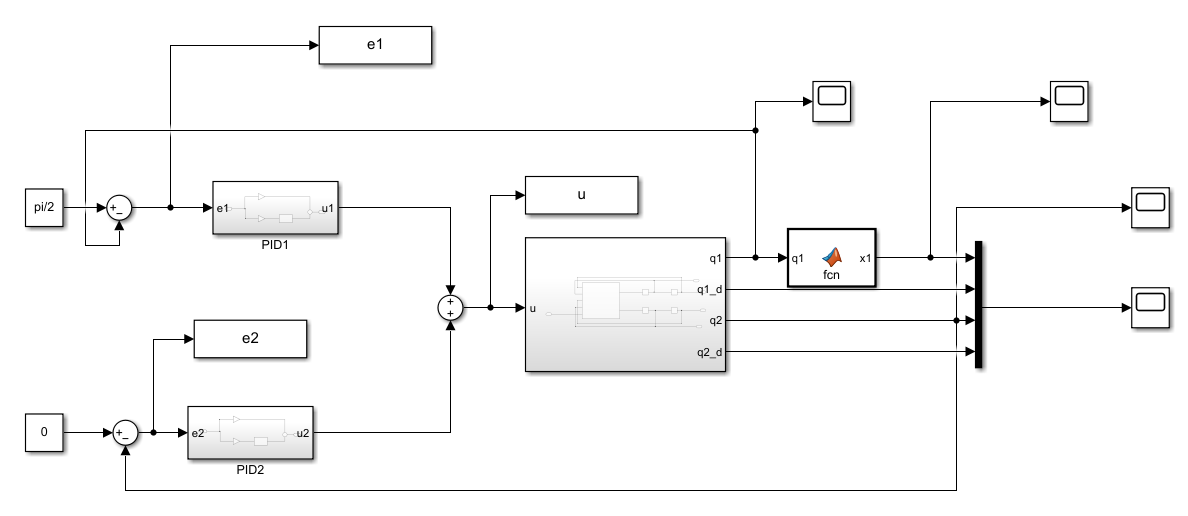
Tóm lại, bộ điều khiển mạng thần kinh là bộ điều khiển dựa trên cơ sở toán học, ở đây sử dụng thuật toán Levenberg-Marquardt (L-M) cho mô hình mạng thần kinh lan truyền ngược. Mạng thần kinh sẽ sao chép và bắt chước quá trình xử lý thông tin, điều khiển của bộ khiển khiển mờ. Trong quá trình sao chép, có thể bộ điều khiển mạng thần kinh chỉ cho ra đáp ứng tương đối so với bộ điều khiển mờ, vì quá trình thử sai sẽ tùy vào kinh nghiệm của người thiết kế mà có thể cho ra bộ điều khiển mạng thần kinh tốt hay không.

## THIẾT KẾ BỘ ĐIỀU KHIỂN MẠNG THẦN KINH CHO HỆ PENDUBOT

Áp dụng phương pháp thiết kế đã trình bày trong mục trước ta có trình tự thiết kế sau:

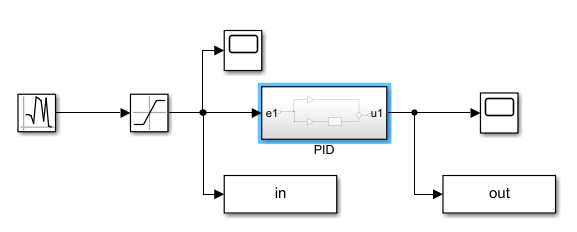
### Thu thập dữ liệu:

Dựa vào mô hình của bộ điều khiển PID cho hệ thống:



Hình 5. Hệ thống Pendubot với bộ điều khiển PID

Thu thập dữ liệu để huấn luyện mạng thần kinh cho 2 khối PID:

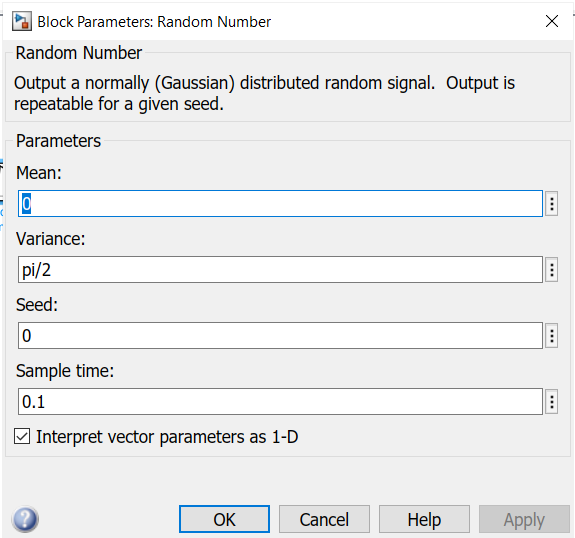


Hình 5. Thu thập dữ liệu cho 2 khối PID

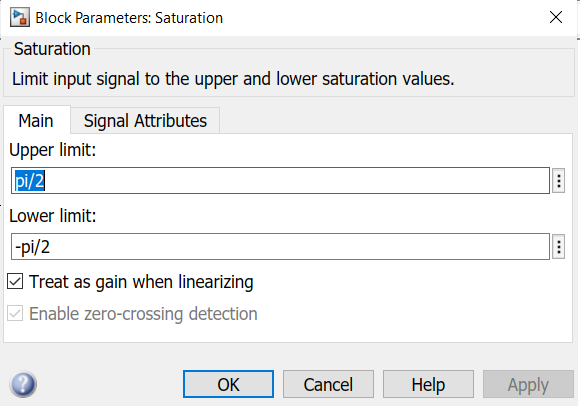
* Với tín hiệu ngõ vào là in
* Tín hiệu ngõ ra out

Do các khối thu thập tín hiệu là như nhau, chỉ khác ở khối tín hiệu Random Number và Saturation, ta lần lượt thay từng bộ PID vào để huấn luyện và thiết lập môi trường như sau:

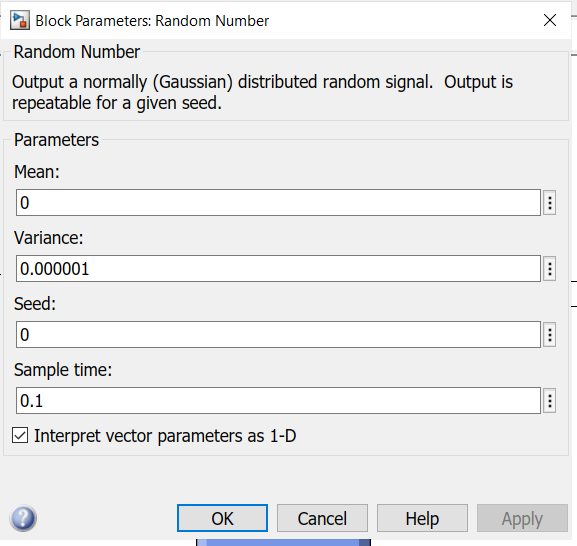
* Thời gian thu thập dữ liệu là 1000 giây. Chế độ Fixed-Step, thời gian lấy mẫu 0.01 giây, không dùng chức năng Single Simulation Output.



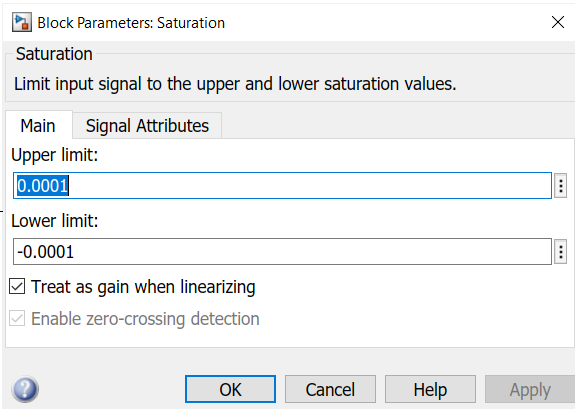
Hình 5. Khối Random Number 1



Hình 5. Khối Saturation 1



Hình 5. Khối Random Number 2



Hình 5. Khối Saturation 2

### Xây dựng mạng thần kinh lan truyền ngược

Sau khi thu thập và xử lý dữ liệu đã xong, ta tiến hành xây dựng mạng thần kinh nhân tạp với cấu trúc:

● Mạng có 4 ngõ vào và 1 ngõ ra.

● Có 50 lớp ẩn, các tế bào thần kinh có hàm kích hoạt dạng S lượng cực (tansig).

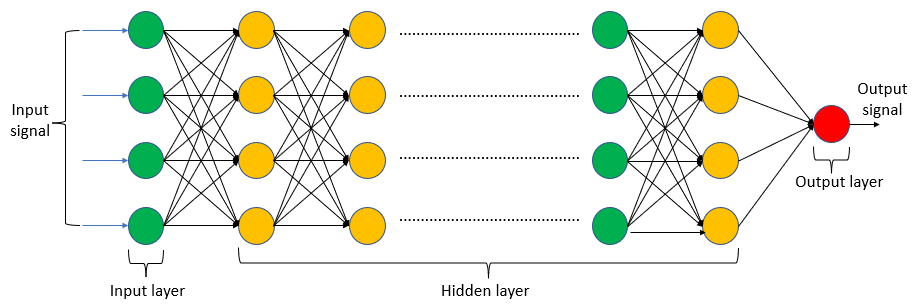


Hình 5. Hàm S lưỡng cực



Hình 5. Hàm tuyến tính

● Lớp ra gồm 1 tế bào thần kinh có hàm kích hoạt tuyến tính (purelin).



Hình 5. Cấu trúc mạng thần kinh nhân tạo

**Luật cập nhật trọng số:**

* Thuật toán sử dụng ở đây là Levenberg-Marquardt (L-M). Thuật toán này được thiết kế để có thể đạt được tốc độ huấn luyện bậc hai của mạng mà không cần tính toán ma trận Hessian, tốc độ hội tụ cũng nhanh hơn thuật toán suy giảm độ dốc. Thuật toán L-M dựa trên triển khai bậc 2 của khai triển Taylor:



* Trong đó:

: là khoảng cách lân cận trong khai triển.

: là vector gradient của E theo W.



là ma trận đạo hàm bậc 2.



* Tại điểm cần tìm và xác định dương.
* Giả sử:





* Khi đó:



* Để tránh bước dịch chuyển quá lớn:



**Chương trình xây dựng một mạng thần kinh lan truyền ngược:**

% programmed by Doan Quoc Viet

ngovao = in’; % dữ liệu đầu vào

ngora = out’; % dữ liệu đầu ra

net=newff([minmax(ngovao)],[40 1],{'tansig' 'purelin'},'trainlm');

net.trainparam.epochs=60000; % số vòng lặp

net.trainparam.lr= 0.0001; % tốc độ học

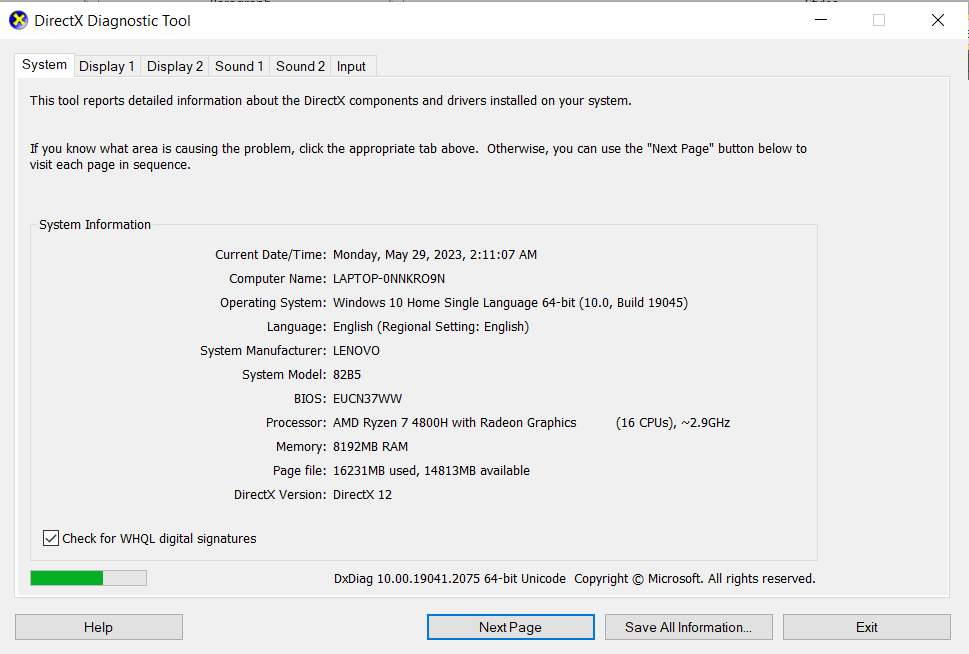
net.trainparam.goal = 1e-7; % sai số

net = train(net,[ngovao],[ngora]); % huấn luyện mạng

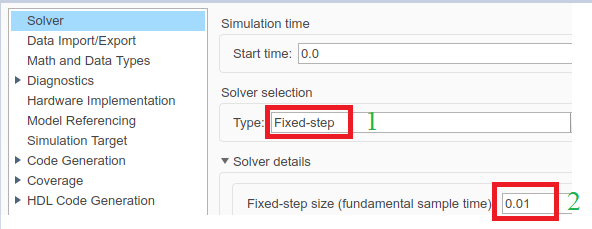
gensim(net,0.1); % tạo khối nơ-rơn

## MÔ TẢ HỆ MÔ PHỎNG

* Nhóm sử dụng Laptop Lenovo Legion 5, core ryzen 7, phiên bản MATLAB R2022b. Thời gian mô phỏng 20 giây.



Hình 5. Thông số Laptop



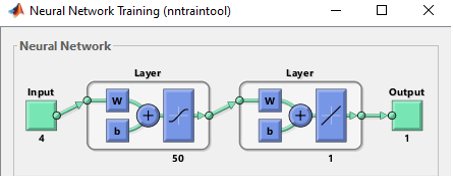
Hình 5. Chọn chế độ mô phỏng Fixed – Step và thời gian lấy mẫu 0.01



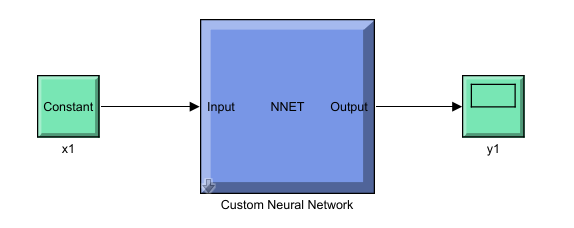
Hình 5. Chọn thời gian mô phỏng

## KẾT QUẢ MÔ PHỎNG

Sau khi quá trình huấn luyện mạng kết thúc, một giao diện thể hiện các thông số của mạng thần kinh và một khối nơ ron được tạo ra, khối này chính là quá trình sao chép khối mờ tạo ra.

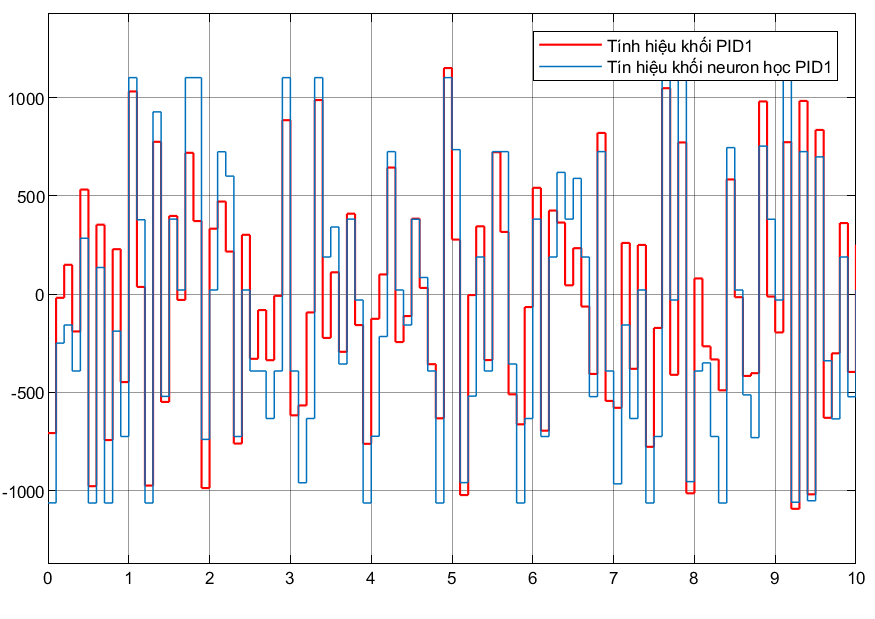


Hình 5. Giao diện thể hiện các thông số của mạng thần kinh



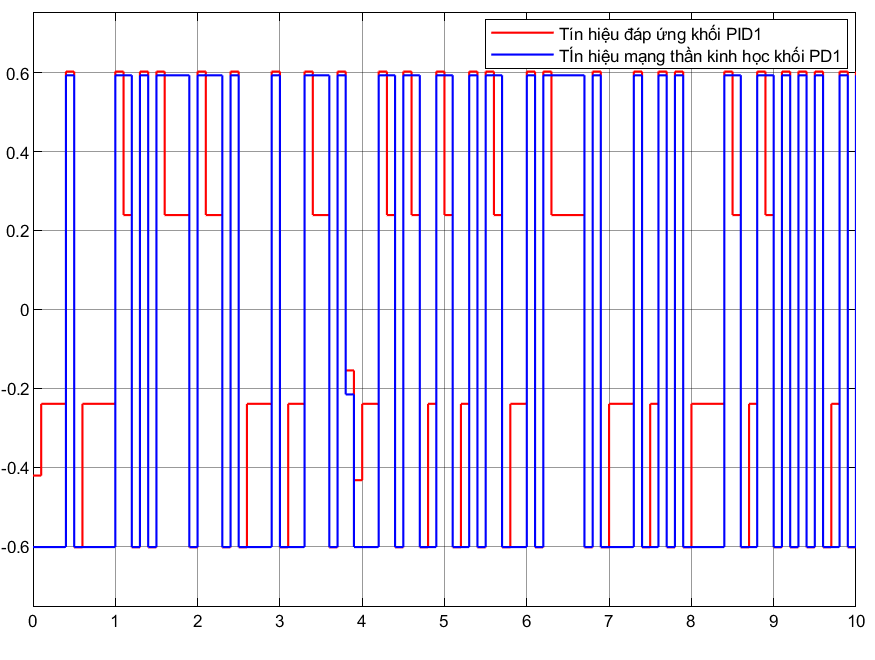
Hình 5. Khối nơ-ron sau khi được huấn luyện

Đáp ứng tín hiệu điều khiển của 2 mạng thần kinh huấn luyện đối chiếu với đáp ứng của 2 khối PID của hệ thống:



Hình 5. Tín hiệu đáp ứng của khối PID1 và

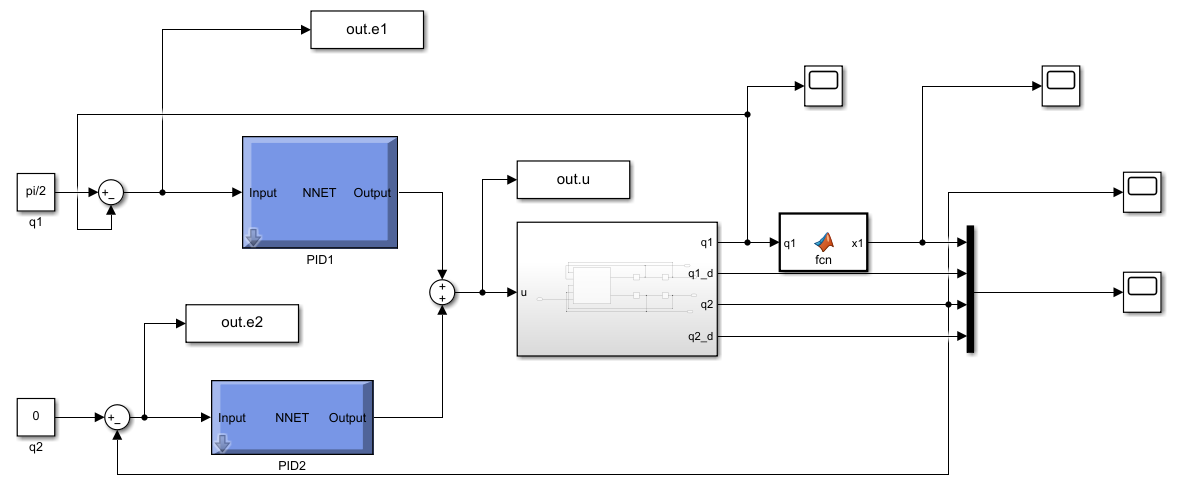
mạng thần kinh trong môi trường thu thập dữ liệu



Hình 5. Tín hiệu đáp ứng của khối PID2 và

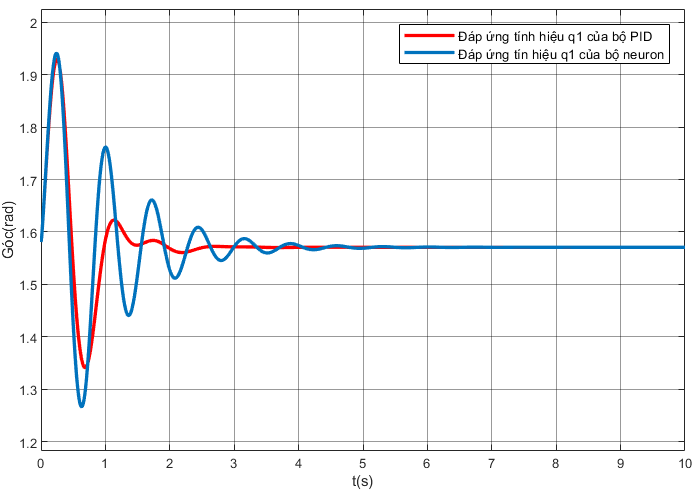
mạng thần kinh trong môi trường thu thập dữ liệu

Sau khi huấn luyện xong, ta đem khối nơ ron sau khi được huấn luyện thay thế cho bộ điều PID và xem đáp ứng ngõ, từ đó đánh giá chất lượng của khối nơ ron, nếu chưa đạt yêu cầu thì tiếp tục huấn luyện lại cho khối nơ ron.

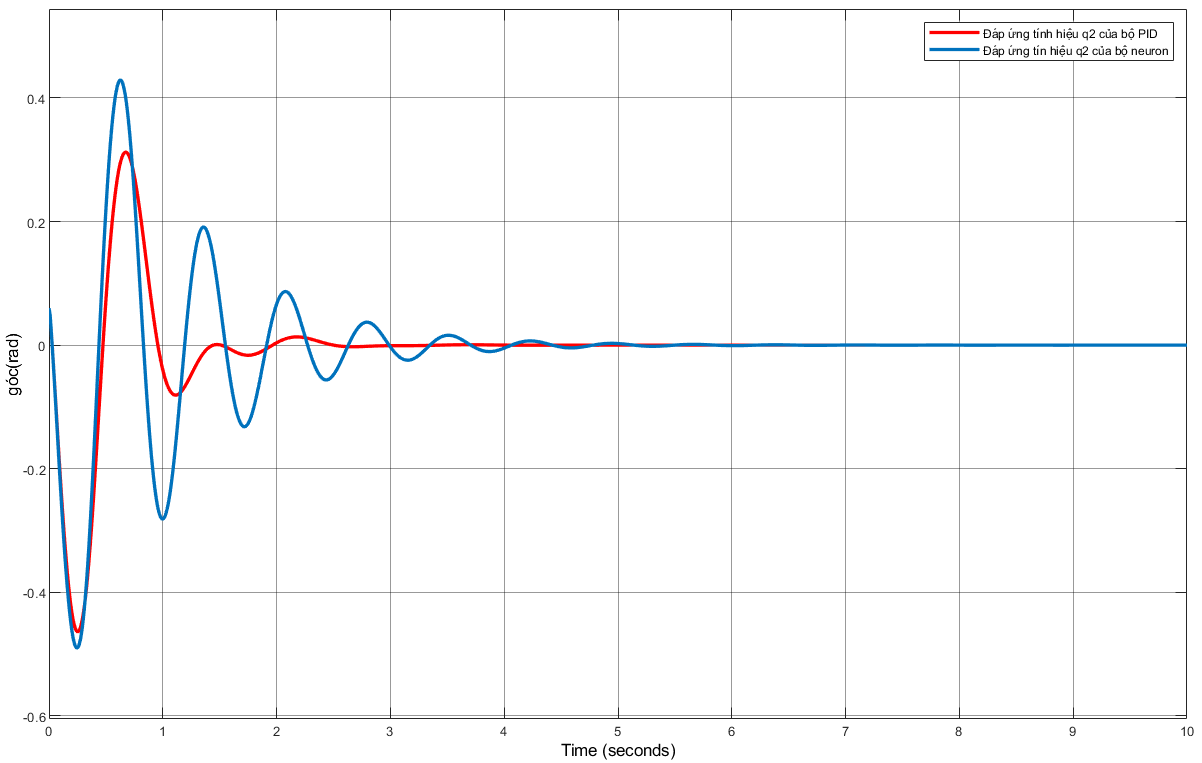


*Hình 5.17 Sơ đồ Simulink sau khi thay thế bộ khiển mờ bằng khối nơ ron đã huấn luyện*

Dưới đây là kết quả mô phỏng hệ pendubot dùng bộ khiển nơ ron thay cho khối PID:



Hình 5. Tín hiệu ngõ ra q1 với bộ điều khiển PID và mạng thần kinh



Hình 5. Tín hiệu ngõ ra q2 với bộ điều khiển PID và mạng thần kinh

Dựa vào kết quả trên, ban đầu ta cho các tín hiệu đặt vàlần lượt là và 0 với trạng thái ban đầu của hệ thống là , ,,. Qua quan sát đáp ứng ngõ ra , ta thấy:

|  |  |
| --- | --- |
| POT(%) | 23.48 |
|  | 0 |
|  | 0.494 |
|  | 2.2 |

Bảng 5. Đáp ứng tín hiệu ngõ ra điều khiển bởi mạng thần kinh

|  |  |
| --- | --- |
| POT(%) | 566.6 |
|  | 0 |
|  | 0.037 |
|  | 5.11 |

Bảng 5. Đáp ứng tín hiệu ngõ ra điều khiển bởi mạng thần kinh

Do đó ta có nhận xét:

* Biên đồ dao động của ngõ ra hệ thống rộng do có độ vọt lố lớn.
* Hệ thống có thời gian lên ngắn, tuy nhiên thời gian xác lập khá dài (xấp xỉ 5.11 s cho hệ thống xác lập ở vị trí top.
* Sai số xác lập cho 2 link của hệ thống bằng 0
* Khi thiết kế dùng phương pháp thử sai để chọn số lớp ẩn, tốc độ học, thuật toán … nên còn tùy vào kinh nghiệm của người thiết kế mạng nơ ron mà có thể xây dựng khối nơ ron đáp ứng tốt hơn. Mô hình mạng thần kinh đã học tương đối tốt những đặc tính của bộ điều khiển PD cho hệ pendubot

# CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## KẾT LUẬN

Sau khi thực hiện xong đề tài này nhóm đã thiết kế được bộ điều khiển PD mờ cân bằng cho hệ pendubot. Thiết kế được mô hình simulink mô phỏng hệ thống. Điều khiển được hệ cân bằng ở vị trí Top. Chất lượng điều khiển khá tốt, tín hiệu điều khiển ổn định. Cùng với đó nhóm đã thành công trong việc sử dụng giải thuật di truyền GA để tìm thông số tiền xử lí và hậu xử lí cho bộ PID, chất lượng bộ điều khiển mà GA tìm ra đáp ứng rất tốt khá tương đồng thông số ban đầu. Cuối cùng nhóm cũng đã thiết kế thành công bộ điều khiển thần kinh xấp xỉ bộ điều khiển PID cho kết quả ngoài mong đợi, bộ điều khiển thần kinh đã học được cách xử lí dữ liệu của bộ điều khiển PID, đáp ứng tốt về góc lệch của 2 thanh trên hệ pendubot và tín hiệu điều khiển tương đồng với bộ điều khiển PID.

## HƯỚNG PHÁT TRIỂN

Kế thừa những ưu điểm và cải tiến những hạn chế, nhóm đưa ra một số đề xuất, hướng phát triển để tối ưu hệ thống:

* Tối ưu giải thuật di truyền GA để tìm thông số giúp bộ điều khiển đáp ứng tốt hơn ở mọi vị trí đặt.
* Thiết kế mô hình thật để khảo sát chất lượng hệ thống so với mô phỏng, từ đó đưa ra nhận xét.

# TÀI LIỆU THAM KHẢO

[1]. Tran Hoang Chinh, Nguyen Minh Tam, Nguyen Van Dong Hai (2017) Ứng dụng điều khiển PID- Mờ cho hệ Pendubot.

[2]. Huỳnh Thái Hoàng (2012)*. Hệ thống điều khiển thông minh,* Nhà xuất bản Đại học quốc gia, TP. Hồ Chí Minh.