ỦY BAN NHÂN DÂN TP. HỒ CHÍ MINH

**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**NGUYỄN XUÂN HIẾU**

**NGHIÊN CỨU VÀ CHẾ TẠO HỆ THỐNG CÂN BẰNG TỰ ĐỘNG BALL AND PLATE**

**TÓM TẮT KHÓA LUẬN TỐT NGHIỆP**

**NGÀNH: CÔNG NGHỆ KỸ THUẬT ĐIỆN, ĐIỆN TỬ**

**tRÌNH ĐỘ ĐÀO TẠO: ĐẠI HỌC**

NGƯỜI HƯỚNG DẪN: THS. NGUYỄN HỮU PHÚC

**TP. HỒ CHÍ MINH, THÁNG 12 NĂM 2018**

**MỤC LỤC**

[1 Tổng quan về hệ thống Ball and Plate 3](#_Toc532485769)

[2. Nhiệm vụ của luận văn 3](#_Toc532485770)

[3. Phân tích hệ thống phần mềm 4](#_Toc532485771)

[4. Bộ điều khiển PID 8](#_Toc532485772)

[4.1 Giới thiệu bộ điều khiển PID: 8](#_Toc532485773)

[4.2 Phương pháp điều khiển PID Cascade 10](#_Toc532485774)

[4.3 Anti-windup cho vòng điều khiển PID 11](#_Toc532485775)

[5. Thiết kế cơ khí của hệ thống Ball And Plate 12](#_Toc532485776)

[6. Ứng dụng điều khiển hệ thống Ball and Plate. 13](#_Toc532485777)

[7. Kết quả thi công 16](#_Toc532485778)

[7.1 Đánh giá thuật toán xử lý ảnh từ Camera 16](#_Toc532485779)

[7.2 Đánh giá việc sử dụng bộ điều khiển PID 16](#_Toc532485780)

[7.3 Đánh giá hệ thống vật lý 16](#_Toc532485781)

[7.4 Đánh giá ứng dụng điều khiển hệ thống 16](#_Toc532485782)

[8. Kết luận 17](#_Toc532485783)

[9. Hướng phát triển 17](#_Toc532485784)

# Tổng quan về hệ thống Ball and Plate

Luận văn nghiên cứu về hệ thống Ball and Plate, là một hệ thống phi tuyến kinh điển, được mở rộng từ hệ thống Ball and Beam.

Hệ thống bao gồm một mặt phẳng (Plate) song song với mặt phẳng Oxz trong không gian 3 chiều Oxyz (với trục y vuông góc và ngược chiều với trọng lực của trái đất, tâm tọa độ đặt ở mặt đất). Hệ thống điều khiển góc nghiêng của mặt phẳng này nhằm mục đích đưa quả bóng (Ball) được thả lăn tự do trên mặt phẳng đến vị trí mong muốn.

Hệ thống điều khiển cần đầu vào là vị trí của quả bóng trên mặt phẳng theo tọa độ (coi mặt phẳng là không gian 2 chiều Oxy) và ngõ ra của hệ thống là góc nghiêng của mặt phẳng theo 2 góc (trong đó là góc nghiêng giữa mặt phẳng và trục Ox, là góc nghiêng giữa mặt phẳng và trục).

Vị trí của quả bóng trên mặt phẳng được tính toán bằng phương pháp xử lý ảnh. Hình ảnh được thu vào bằng Camera Raspberry Pi NoIR V2, được xử lý bằng máy tính nhúng Raspberry Pi 3 Model 3B. Vị trí này được truyền đến board điều khiển STM32F4 Discovery để xử lý và tính toán góc nghiêng của mặt phẳng, và điều khiển động cơ để tạo ra góc nghiêng này.

Ngoài ra trong luận văn này còn thiết kế một ứng dụng giúp cập nhật các thông số của hệ thống và vị trí điểm đặt của quả bóng, giúp thuận tiện hơn trong việc điều khiển hoạt động của hệ thống.

Mô hình hệ thống Ball and Plate được thể hiện ở hình sau :



Hình 2.1: Mô hình tổng quát hệ thống Ball and Plate

Hệ thống có thể được điều khiển từ các thiết bị thông mình như : điện thoại, laptop, pc, … thông qua mạng Wi-fi nội bộ.

Raspberry Pi 3 nhận lệnh điều khiển từ các thiết bị trên cũng thông qua mạng Wi-fi này, Raspbeery Pi 3 kết hợp với camera Raspberry Pi module NoIR để nhận dạng vị trí trái bóng và gửi xuống Kit STM32F4 nhờ giao tiếp UART, Kit này sẽ điều khiển hai động cơ Faulhaber 12V nhờ module mạch cầu H L298N.

# 2. Nhiệm vụ của luận văn

Giải quyết bài toán tìm bóng trên mặt phẳng bằng các phương pháp xử lý ảnh nhờ thư viện OpenCV. Đồng thời phải đáp ứng được yêu cầu xử lí ảnh ở tốc độ cao trên Raspberry Pi 3 Model B để mang lại đáp ứng tốt cho hệ thống.

Triển khai và kết hợp thành công bộ điều khiển PID - PID theo mô hình Cascade trên MCU STM32F411VET6.

Thiết kế ứng dụng điều khiển hệ thống bằng engine Unity.

Hoàn thành và thử nghiệm thành công mô hình điều khiển Ball and Plate.

# 3. Phân tích hệ thống phần mềm

**3.1 Cấu trúc phần mềm trên Raspberry Pi 3**

Cấu trúc phần mềm trên Raspberry Pi 3 được thể hiện ở hình 1:



Hình 1: Cấu trúc phần mềm trên Raspberry Pi 3

Máy tính nhúng Raspberry Pi 3 được cài đặt hệ điều hành Raspbian dựa trên nhân Linux. Nhân hệ điều hành Linux hỗ trợ các device driver cho các ngoại vi trên CPU (GPIO, UART, SPI, I2C, HDMI, CSI). Nhờ đó, ta có thể dễ dàng phát triển các chương trình có liên quan đến phần cứng trên nền Linux bằng việc sử dụng các sytem call tương ứng.

Thư viện OpenCV có phiên bản được biên dịch để chạy trên nền nhân ARM-64bit của CPU BCM2837. Các hàm của OpenCV dễ sử dụng, hoạt động chính xác và được tối ưu hóa tốt.

Mã nguồn xử lý ảnh được tích hợp với OpenCV, và thực thi trên hệ điều hành Raspian. Mã nguồn sẽ tạo thêm 4 luồng để đảm nhận các công việc khác nhau nhằm tối ứu tài nguyên hệ thống:

* Luồng Capture Frame lấy ảnh BGR liên tục từ camera, với tốc độ 90 ảnh trên 1 giây.
* Luồng Pre-processing Frame chuyển ảnh lấy được từ luồng Capture Frame và chuyển sang ảnh HSV để nhận diện màu.
* Luồng Ball Position Detecting sẽ tìm trái bóng dựa vào màu và kích thước.
* Luồng Show Frame hiển thị vị trí trái bóng đã nhận dạng được, luồng này là tùy chọn, chỉ chạy để phục vụ mục đích debug.

Việc gửi và nhận dữ liệu được luồng chính đảm nhiệm.

Vậy khi hệ thống chạy thực tế, chỉ có 4 luồng được chạy là luồng chính, luồng Capture Frame, luồng Pre-processing Frame và luồng Ball Position Detecting.

Quan hệ giữa các luồng được thể hiện ở hình 2:



Hình 2: Tổng quan phần mềm xử lý ảnh trên Raspberry Pi 3

**3.2 Cấu trúc phần mềm trên Board STM32F4 Discovery**

Mã nguồn điều khiển trên Kit STM32F4 sẽ được tích hợp hai thư viện STM32F4 HAL và FreeRTOS để hỗ trợ cho việc triển khai bộ điều khiển PID lên hệ thống, được thể hiện ở hình 3:



Hình 3: Cấu trúc phần mềm trên Board STM32F4 Discovery

FreeRTOS cũng cung cấp các cơ chế đồng bộ như Linux và còn cung cấp thêm khả năng đồng bộ mới là task notification, giúp ta đồng bộ và bảo vệ các task nhanh hơn. Tuy nhiên, không giống như Linux, hệ điều hành FreeRTOS không hỗ trợ device driver, nên chúng ta sẽ dùng thư viện STM32F4 HAL.

Thư viện STM32F4 HAL là một thư viện phần cứng, hỗ trợ mạnh mẽ cho dòng MCU STM32F4. Thư viện này chứa các hàm giao tiếp phần cứng một cách nhanh chóng mà không cần nắm rõ các kiến thức về thanh ghi, giúp ta dễ dàng xây dựng một bộ device driver phục vụ riêng cho hệ thống.

Mã nguồn chia công việc thành các task. Mỗi task giao tiếp với nhau thông qua task notification, với task có ưu tiên thấp sẽ vào trạng thái Ready đợi task notification từ task có ưu tiên cao. Nên các task hoạt động theo kiểu Producer – Consumer. Vì vậy các vùng nhớ dùng chung giữa các task cũng không cần được bảo vệ.

Khi hệ thống chạy, mã nguồn sẽ tạo 5 task như sơ đồ 3.7:



Hình 4: Trình tự hoạt động của các task

Task setup chỉ chạy 1 lần duy nhất, để cài đặt các thông số ban đầu cho hệ thống.

Task Update UART Data dùng để cập nhật dữ liệu nhận được từ bộ UART.

Task Update Setpoint dùng để tính toán điểm đặt cho trái bóng.

Task Calculate Reference Encoder Value tính toán giá trị encoder tương ứng với điểm đặt vừa được tính để đưa động cơ đến vị trí góc mong muốn.

Task Control Motor dựa vào giá trị encoder vừa được tính để xuất xung PWM cho động cơ.

# 4. Bộ điều khiển PID

## 4.1 Giới thiệu bộ điều khiển PID:

Bộ điều khiển PID là bộ điều khiển dựa trên cơ chế hồi tiếp vòng kín, liên tục nhận về phản hồi từ các cảm biến đo trạng thái của hệ thống, từ đó tính toán sai số của hệ thống hiện tại so với trạng thái mong muốn. Bộ điều khiển sau đó đưa ra điều chỉnh dựa trên sai số của tín hiệu ngõ vào dựa vào các thông số về tỉ lệ, tích phân và đạo hàm.



Hình 5: Bộ PID liên tục

Sau các khâu tỉ lệ, đạo hàm và tích phân, hệ thống tính tổng kết quả các khâu trên, tổng này chính là tín hiệu điều khiển ngõ ra của bộ điều khiển PID:

|  |  |
| --- | --- |
|  | (2.1) |

Bộ điều khiển điều chỉnh tín hiệu sao cho sai số ngõ ra là nhỏ nhất. Đáp ứng của hệ thống điều khiển phụ thuộc vào các trọng số Kp, Ki, Kd.

Trong luận văn này, bộ điều khiển PID được sử dụng là bộ PID rời rạc với:

Khâu tỉ lệ:

|  |  |
| --- | --- |
|  | (2.2) |

Khâu vi phân: được tính bằng công thức sai phân lùi

|  |  |
| --- | --- |
|  | (2.3) |

Khâu tích phân: được tính bằng công thức tích phân hình thang

|  |  |
| --- | --- |
|  | (2.4) |

Vậy hàm truyền của bộ điều khiển PID như sau

|  |  |
| --- | --- |
|  | (2.5) |

Chuyển hàm truyền sang miền thời gian, ta có:

|  |  |
| --- | --- |
|  | (2.6) |

Dựa vào công thức trên, ta thấy rõ ràng hơn quan hệ của sai số và đáp ứng trong quá khứ đối với đáp ứng hiện tại của bộ PID.

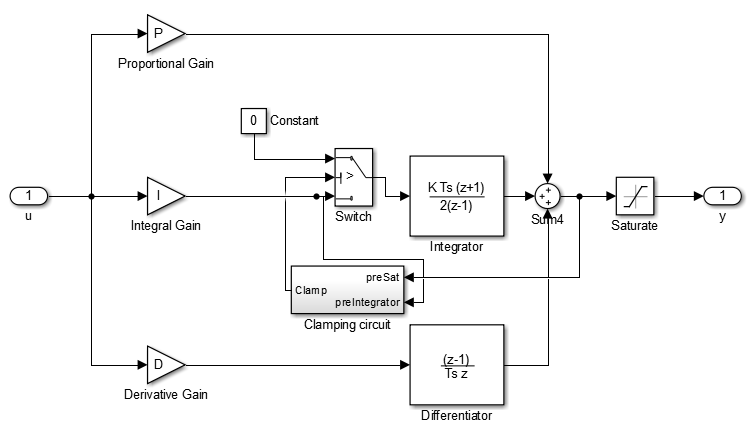
## 4.2 Phương pháp điều khiển PID Cascade

Bộ điều khiển PID thể hiện sự ưu việt khi sử dụng ở chế độ Cascade. Ở chế độ này, 2 bộ điều khiển PID có thể cùng nhau tính toán toán để đem lại đáp ứng tốt hơn cho hệ thống. 2 bộ điều khiển PID được sắp xếp sao cho ngõ ra của bộ điều khiển PID này là điểm đặt của bộ PID kia. Một bộ PID nhận tín hiệu từ bên ngoài (outer loop controller) và tạo ra điểm đặt cho bộ PID ở vòng trong (inner loop controller). Với PID Cascade, hệ thống sẽ có đáp ứng tốt hơn.

## 4.3 Anti-windup cho vòng điều khiển PID

Do tín hiệu của khâu tích phân được cộng dồn liên tục do chu kỳ điều khiển nhỏ (khoảng vài ms). Kết quả là tín hiệu ngõ ra của khâu tích phân cứ tăng lên mãi, đến một lúc sẽ vượt quá khả năng của hệ thống. Hiện tượng đó gọi là windup. Vì vậy với bộ điều khiển PID, ta cần phải thiết kế thêm khâu hiệu chỉnh để chống lại hiện tượng này, gọi là anti-windup

Với bộ điều khiển PID, cách đơn giản nhất để loại bỏ hiện tượng anti-windup là sử dụng phương pháp clamping, tức là ta sẽ chặn giới hạn trên và giới hạn dưới của giá trị ngõ ra của bô điều khiển PID, đồng thời tiến hành hiệu chỉnh thông số ngõ vào Ki khi có hiện tượng windup. Với mô hình hiệu chỉnh cho bộ điều khiển PID như sau:



Hình 6: Bộ PID rời rạc kết hợp với phương pháp clamping

Từ mô hình, ta thấy, khi ngõ ra của bộ điều khiển PID bão hòa, khâu hiệu chỉnh sẽ xét dấu của tích của thông số Ki và sai số ngõ vào. Nếu là bão hòa ở giới hạn trên và dấu của tích hai thông số là dương thì hiệu chỉnh Ki bằng 0, tương tự, khi bão hòa dưới và tích của hai thông số mang dấu âm thì cũng hiệu chỉnh Ki về 0.

**5. Thiết kế cơ khí của hệ thống Ball And Plate**

Mặt phẳng được sử dụng trong luận văn có kích thước 35 x 35cm, Phần mặt phẳng mà bóng hoạt động có màu trắng.

Camera xử lý ảnh và máy tính nhúng Raspberry Pi 3 được gắn cố định ở vị trí trung tâm của mặt phẳng và cao hơn mặt phẳng khoảng 80cm.

Hệ thống này được thiết kế với 2 motor Faulhaber đảm nhận nhiệm vụ điều khiển góc nghiêng của mặt phẳng theo 2 trục x và y. Cơ cấu điều khiển được thể hiện ở hình 3.12:

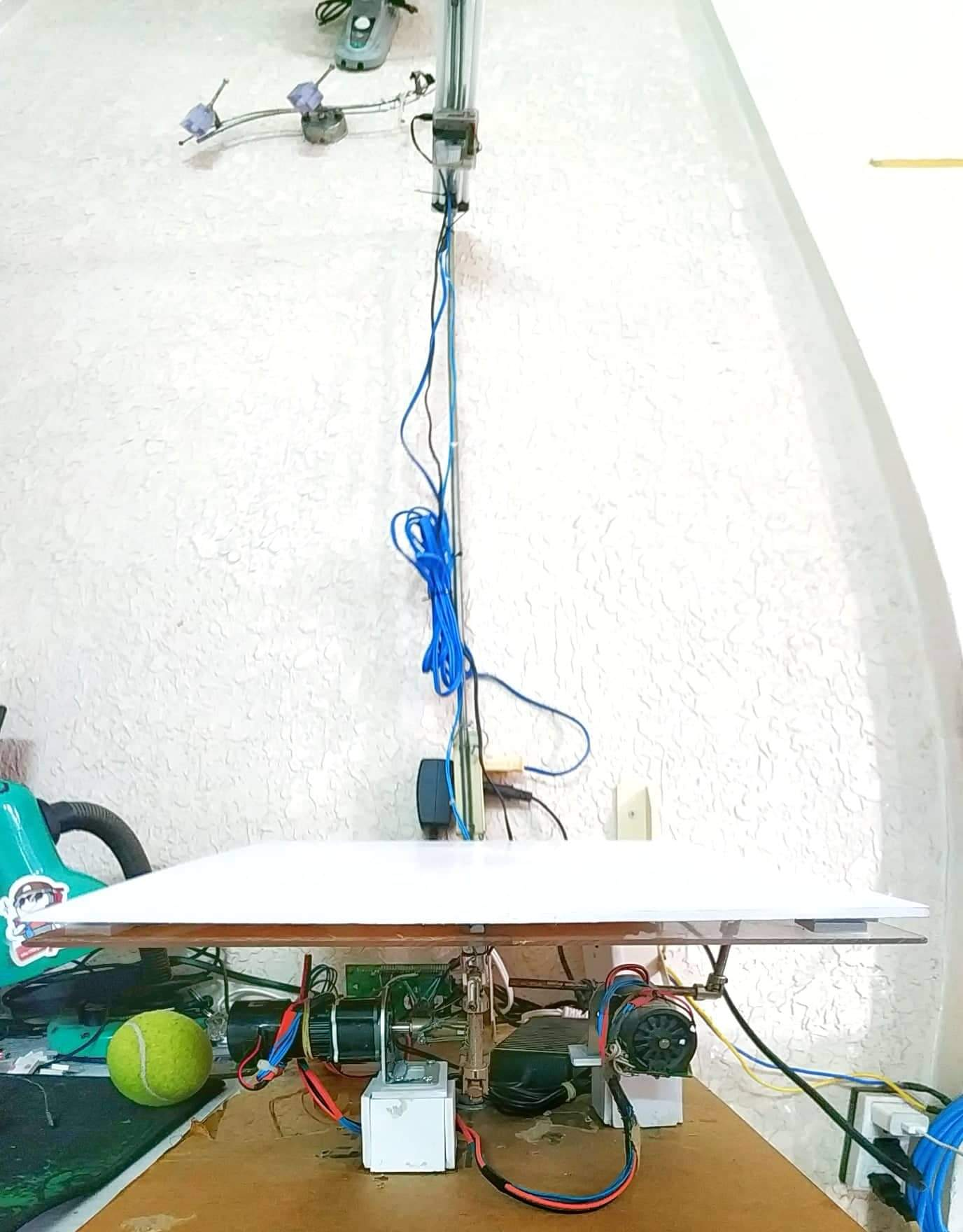


Hình 7: Mô hình cơ khí hệ thống Ball and Plate

Trong cơ cấu này, trục động cơ được lắp với một cánh tay đòn để tác động lên mặt phẳng.Như vậy, 2 trục xoay của động cơ được lắp vuông góc với trục mà động cơ đó điều khiển.

Bóng được sử dụng trong luận văn là bóng Tennis màu vàng có đường kính khoảng 6.8 cm.

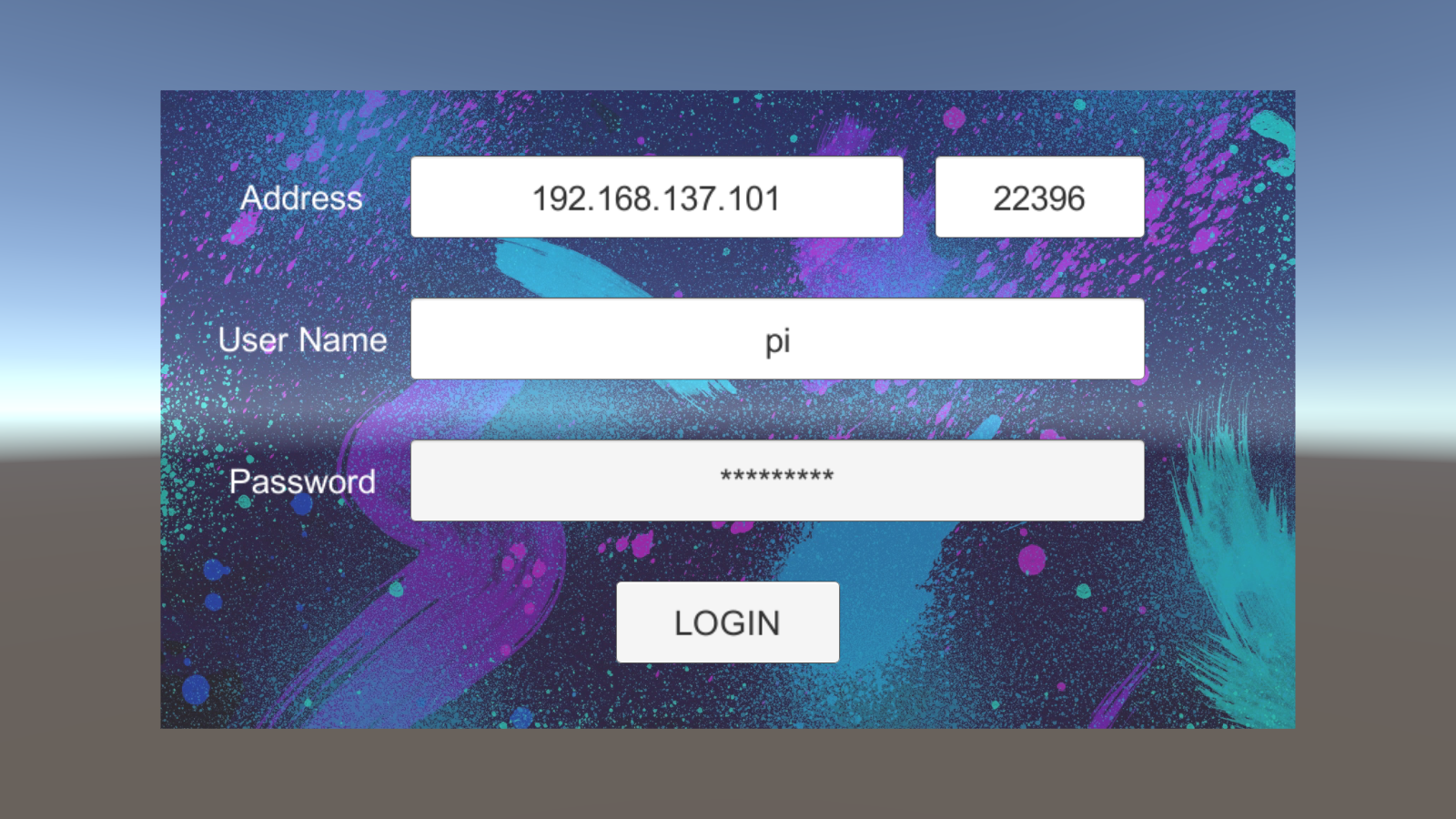
Mô hình thực tế của hệ thống thể hiện ở hình 3.13:



Hình 8: Hình ảnh thực tế

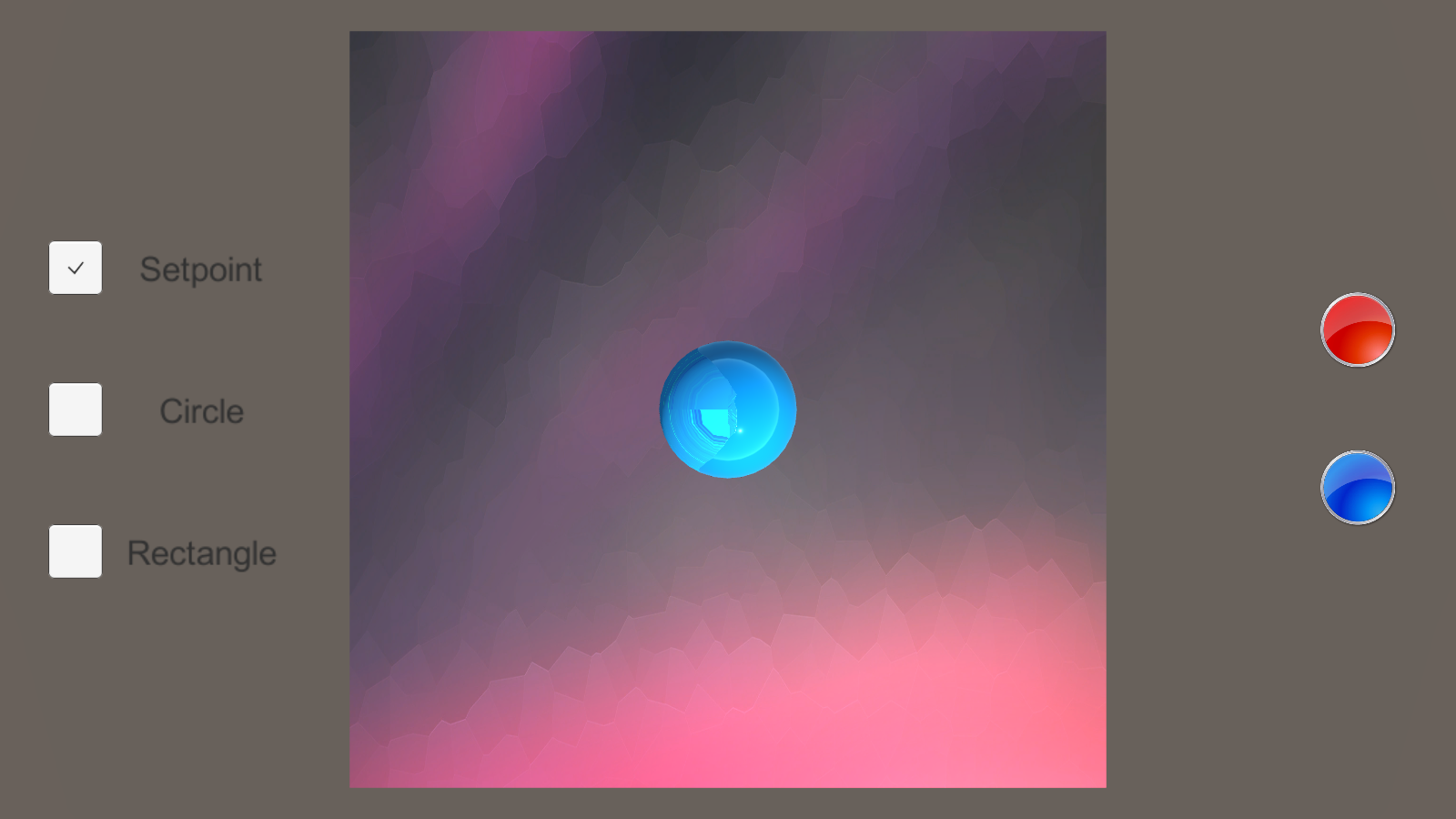
# 6. Ứng dụng điều khiển hệ thống Ball and Plate.

Để thuận tiện cho việc điều khiển hệ thống, luận văn thiết kế một ứng dụng điều khiển. Ứng dụng này giúp ta dễ dàng thay đổi điểm đặt của hệ thống, từ đó có thể điều khiển các chế độ hoạt động khác như điều khiển banh chạy theo quỹ đạo hình tròn, hình tứ giác, … Đồng thời ứng dụng còn hỗ trở thay đổi các thông số Kp, Ki, Kd của cả hai vòng điều khiển, giúp việc tìm ra thông số tối ưu cho hệ thống trở nên thuận tiện hơn.



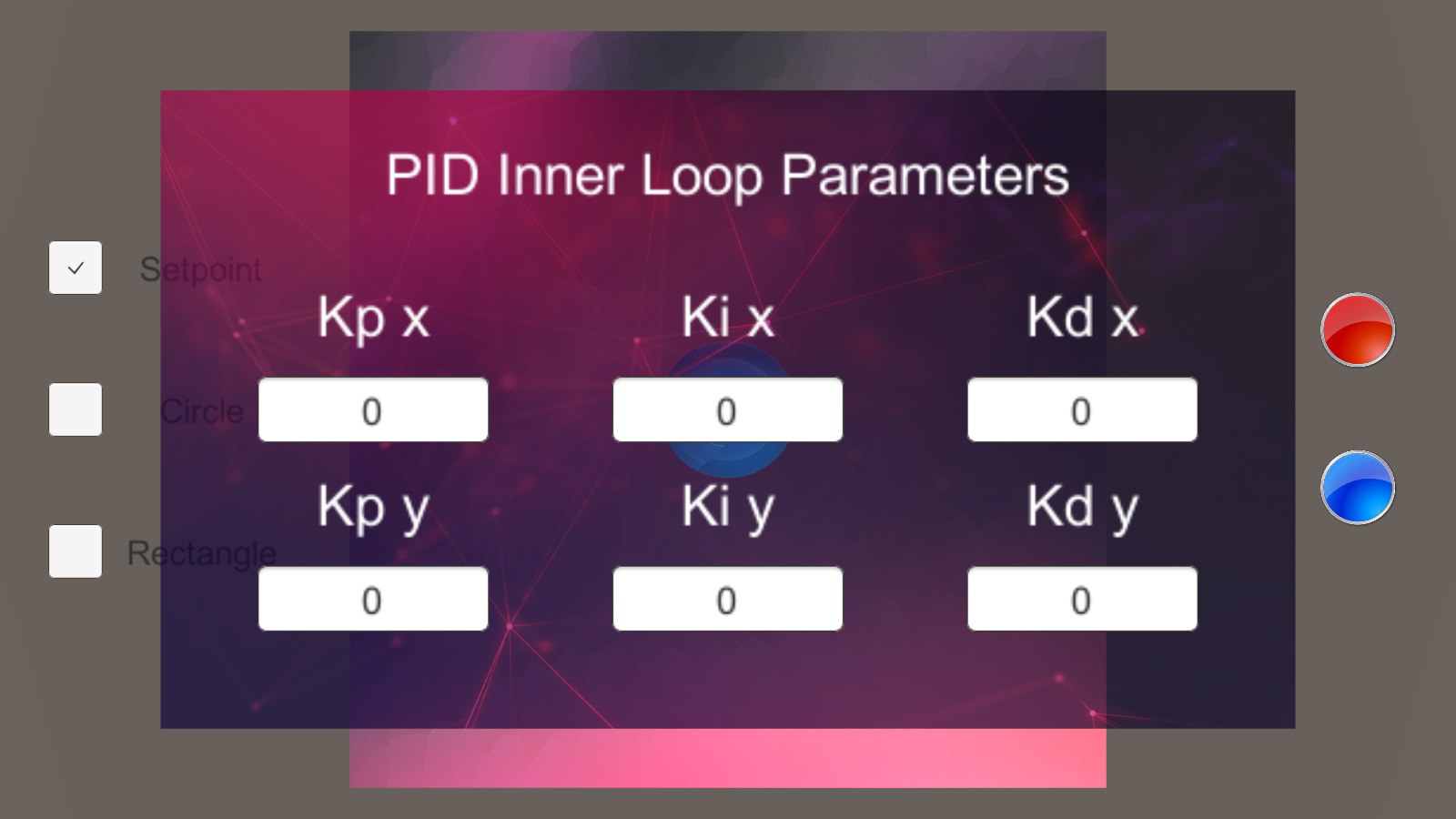
Hình 9 : Giao diện đăng nhập

Để đăng nhập vào hệ thống, cần phải nhập đúng địa chỉ IP hiện tại của hệ thống trong mạng. Với user name mặc định là ‘pi’, password là ‘raspberry’, cổng mặc định là 22396.

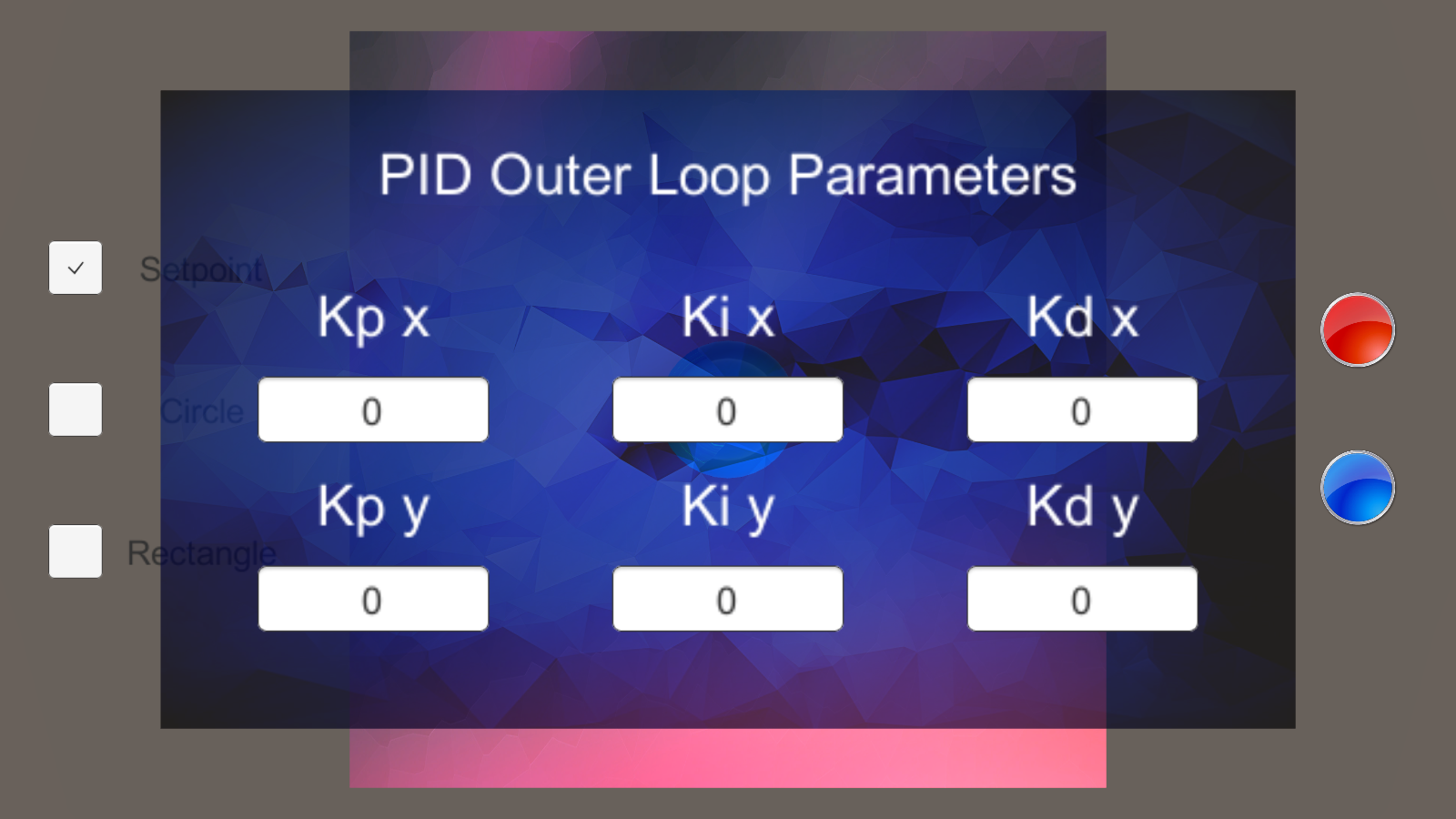


Hình 10: Giao diện chính

Giao diện chính hiển thị trái banh và mặt phẳng theo hướng từ trên nhìn xuống. Bên trái là 3 chế độ điều khiển : điểm đặt, chạy theo quỹ đạo tròn và chạy theo quỹ đạo tứ giác. Bên phải với 3 nút nhấn : nút đỏ để hiển thị bảng điều khiển cho bộ PID vòng trong, nút xanh để hiện thị bảng điều khiển cho bộ PID vòng ngoài.



Hình 11: Giao diện cài đặt bộ PID vòng trong



Hình 12: Giao diện cài đặt bộ PID vòng ngoài

# 7. Kết quả thi công

## 7.1 Đánh giá thuật toán xử lý ảnh từ Camera

Việc xử lý hình ảnh trên camera với kĩ thuật đa luồng đem lại tốc độ xử lý cao và tận dụng được tài nguyên hệ thống trên Raspberry Pi 3.

Tốc độ xử lý luôn được đảm bảo 90 khung hình trên giây, đủ để hệ thống đáp ứng nhanh và chính xác.

## 7.2 Đánh giá việc sử dụng bộ điều khiển PID

Đối với chế độ điểm đặt, bóng được giữ ở vị trí tương đối so với điểm đặt, tuy nhiên vẫn còn bị trượt và chưa ổn định hoàn toàn.

Đối với chế độ quỹ đạo tròn, trái banh có chạy theo quỹ đạo tròn nhưng chưa bám sát vào quỹ đạo đặt, sai số điều khiển còn cao.

Đối với chế độ tứ giác, trái banh được điều khiển tới các góc với vọt lố lớn, chưa ổn định.

Nhìn chung thì hệ thống có thể cân bằng tốt trái banh trên mặt phẳng, nhưng chưa điều khiển chính xác được vị trí trái banh.

## 7.3 Đánh giá hệ thống vật lý

Hệ thống vật lý trong bài luận văn chưa được gia công tốt, dễ rung lắc, làm ảnh hưởng đến việc điều khiển trái banh.

Các trục quay và tay nâng chưa được gia công chính xác gây ra sai số, các khớp nối giữa tay quay và mặt phẳng còn nhiều rung lắc.

## 7.4 Đánh giá ứng dụng điều khiển hệ thống

Ứng dụng cơ bản đã điều khiển được hệ thống theo mong muốn của người điều khiển. Tuy nhiên ứng dụng hoạt động chưa ổn định, thỉnh thoảng bị treo.

# 8. Kết luận

Thử nghiệm thành công bộ điều khiển PID vào việc điều khiển động cơ và góc nghiêng của mặt phẳng

Kết hợp thành công bộ điều khiển PID-PID theo mô hình Cascade.

Áp dụng thành công các kĩ thuật lập trình đa luồng để tối ưu hóa việc xử lí ảnh tốc độ cao trên Raspberry Pi 3.

Áp dụng thành công kĩ thuật lập trình Multitask trên hệ điều hành FreeRTOS để tạo ra phần mềm điều khiển hệ thống trên STM32F4.

Hoàn thành và thử nghiệm thành công mô hình điều khiển Ball And Plate.

Áp dụng thành công các kĩ thuật xử lý ảnh trên OpenCV để giải quyết bài toán tìm bóng trên mặt phẳng.

Phát triển thành công ứng dụng điều khiển hệ thống Ball and Plate dựa trên phần mềm Unity.

# 9. Hướng phát triển

Cải thiện hệ thống cơ khí.

Sử dụng hệ thống máy tính nhúng có sức mạnh lớn hơn và camera có tốc độ quay phim cao hơn để nâng tốc độ nhận dạng ảnh, giúp đáp ứng hệ thống tốt hơn.

Sử dụng bộ điều khiển phi tuyến để tính toán với mô hình chính xác hơn.

Tối ưu ứng dụng điều khiển để trải nghiệm của người điều khiển tốt hơn.

**TÀI LIỆU THAM KHẢO**

[[1]](http://coffeebrain.org/wikicontrol/images/d/d6/Ball_and_Plate_CCAC2017.pdf) Nguyễn Thị Phương Hà, Huỳnh Thái Hoàng (2011) , *Lý Thuyết Điều Khiển Tự Động*, Nhà xuất bản Đại học Quốc gia Tp. Hồ Chí Minh.

[[2]](http://folk.ntnu.no/skoge/prost/proceedings/process-control-slovakia-2017/data/papers/027.pdf) Mark Mitchell, Jeffrey Oldham, Alex Samuel (2001), *Advanced Linux Programming,* composed in Bembo and MCPdigital by New Riders Publishing.

[[3]](https://www.iasj.net/iasj?func=fulltext&aId=125507) Richard Barry (2016), *Mastering the FreeRTOS Real Time Kernel*.

[4] IBM Knowledge Center, *Using poll() instead of select().*

[5] N3K EN (2016), *Unity Mobile Game Tutorial.*

[6] *OpenCV documentation.*