LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của bản thân, được xuất phát từ yêu cầu phát sinh trong quá trình làm luận văn. Các tài liệu liên quan có nguồn gốc rõ ràng, tuân thủ đúng nguyên tắc, kết quả trình bày trong luận văn là kết quả quá trình nghiên cứu trên mô hình thực tế

Tp. Hồ Chí Minh, tháng 06 năm 2019.

Tác giả luận văn

Tôn Thế Cường

LỜI CẢM ƠN

Mở đầu tôi xin gửi lời cảm ơn chân thành nhất đến giảng viên hướng dẫn, thầy **TS. Nguyễn Vĩnh Hảo** người đã luôn quan sát, định hướng và giúp đỡ tôi trong suốt quá trình thực hiện đề cương luận văn cao học. Cảm ơn câu lạc bộ nghiên cứu khoa học Pay It Forward đã tạo cơ hội cho tôi được tiếp xúc với những kiến thức cần thiết từ rất sớm.

Xin cảm ơn tất cả thầy cô trong bộ môn Điều Khiển Tự Động, các thầy cô trong khoa Điện Điện tử trường đại học Bách Khoa Tp.HCM đã giảng dạy, cung cấp cho tôi nhiều kiến thức bổ ích không chỉ trong quá trình nghiên cứu sau đại học mà còn cả quá trình học tập khi tôi còn là sinh viên.

Tôi cũng xin chân thành cảm ơn cha mẹ, các bạn bè trong khóa cao học Tự Động Hóa 2017 đã không ngừng khích lệ, giúp đỡ để tôi có thể hoàn thành tốt đề cương luận văn này.

Học viên thực hiện

TÔN THẾ CƯỜNG

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG

DANH MỤC TỪ VIẾT TẮT

TGU	Tohoku Gakuin University
ETH	Eidgenössische Technische Hochschule
CMU	Carnegie Mellon University
BallIP	Ball Inverted Pendulum
IMU	Inertial Measurement Unit
LQR	Linear Quadratic Regulator
SMC	Sliding Mode Control
HSMC	Hierarchical Sliding Mode Control
PID	Proportional-Integral-Derivative
SISO	Single input single output
SIMO	Single input multiple output
SMS	Sliding mode system
Ballbot	Nền tảng robot gắn trên bánh xe hình cầu

TÓM TẮT LUẬN VĂN

Việc tự động hóa kiểm soát cơ chế dịch chuyển trên mặt đất trong lĩnh vực robot đang thu hút sự quan tâm của các nhà nghiên cứu và thương mại trong thời gian gần đây vì những ứng dụng của chúng trong một số hoạt động như hướng dẫn, nhận diện khu vực, theo dõi đường đi, hỗ trợ và giải trí. Robot banh ra đời nhằm mục tiêu khắc phục những hạn chế của các nền tảng robot trước đây như không gian di chuyển và độ linh hoạt. Ballbot là nền tảng robot có độ linh hoạt cao khi chuyển hướng và di chuyển nhưng cũng đồng thời mang trong nó tính không ổn định vốn có, đặt ra rất nhiều thách thức cho những nhà nghiên cứu hệ thống điều khiển.

Luận văn thực hiện các công việc bao gồm thiết kế, thi công robot, điều khiển giữ thăng bằng cho robot một bánh. Hệ robot banh được mô hình hóa thành các phương trình chuyển động và thực hiện mô phỏng bằng Matlab/Simulink. Các bộ điều khiển giữ thăng bằng cho robot dự kiến được nghiên cứu trong đề tài bao gồm: bộ điều khiển LQR, bộ điều khiển chế độ trượt. Các bộ điều khiển sau khi được thiết kế bằng lý thuyết và mô phỏng sẽ được kiểm chứng bằng cách áp dụng vào điều khiển đối tượng thực.

Kết quả hiện tại đã xây dựng được mô hình toán và mô phỏng hệ thống. Mô hình ballbot thực tế cũng đã xây dựng xong nhằm kiểm chứng đáp ứng hệ thống qua các bộ điều khiển của mô hình thực. Thiết kế hai bộ điều khiển LQR và điều khiển chế độ trượt để ổn định hệ thống và kiểm soát vị trí robot. Bộ điều khiển tuyến tính LQR trong thực tế đã được kiểm chứng.

Chương 1. TỔNG QUAN

1.1. Giới thiệu chung

Robot banh là một robot trong đó sử dụng các khái niệm về sự ổn định động lực học đứng thẳng và cân bằng trên quả bóng. Điều này liên quan đến việc sử dụng lý thuyết điều khiển để di chuyển bóng và cân bằng thay vì dựa vào trọng lực. Kể từ khi ballbot đầu tiên xuất hiện tại đại học Carnegie Mellon (2005-2006) [1], đã có rất nhiều nghiên cứu liên quan đến nền tảng này và đạt được thành công nhất định như BalllP tại Tohoku Gakuin University [2], LEGO NXT Ballbot được phát triển bởi bởi Yorihisa Yamamoto [3] hay Rezero của ETH Zurich [4] .

Sự phát triển của robot đứng trên bóng yêu cầu xuất phát từ động lực học, cấu trúc của từng robot, thiết kế và thực hiện bộ điều khiển để ổn định robot. Mô hình ballbot khi phân tích trên một mặt cắt có thể được xem như con lắc ngược gắn trên quả bóng. Đây là nền tảng cổ điển thú vị và phổ biến để phát triển, thử nghiệm lý thuyết điều khiển do tính chất không ổn định vốn có của nó, thể hiện thách thức với các nhà nghiên cứu hệ thống điều khiển. Nguồn gốc của phương trình chuyển động hệ thống robot có thể được thực hiện bằng cách sử dụng phương trình Lagrange, với việc tiếp cận về một mô hình robot đơn giản, từ đây tuyến tính hóa phương trình chuyển động của hệ thống để áp dụng thuật toán điều khiển tuyến tính. Như ballbot LEGO NXT đã được xây dựng và hoàn thành với thuật toán điều khiển LQR. Robot này sử dụng cơ cấu lái bóng – chuột máy tính đơn giản, trong đó bóng được vận hành bởi bánh xe hướng về phía trực giao của quả bóng. Không dừng lại ở đó, một nền tảng ballbot khác được xây dựng và phổ biến hơn bao gồm ba bánh xe đặt lệch nhau góc 120^0 gắn trên bóng. Điều kiện cơ bản để có thể thực hiện và phát triển nền tảng này đó là robot phải có khả năng di chuyển linh hoạt nên trên bề mặt của bóng. Một thiết kế đặt biệt thuộc về phần cơ khí của của bánh xe được phát triển đó là omni wheel hay bánh xe đa hướng. Vấn đề cơ khí được giải quyết là điều kiện để các nhà nghiên cứu có thể phát triển những thuật toán điều khiển phức tạp hơn trong những năm gần đây. Các

bộ điều khiển phi tuyến đã và đang được áp dụng thể thay thế thuật toán LQR, PID kinh điển.

1.2. Đặt vấn đề

Con người từ lâu đã mơ ước về các trợ lý người máy cá nhân, tạo ra để thực hiện mọi nhiệm vụ của mình. Một trong những yêu cầu để có thể thực hiện điều đó là tạo ra các robot có kích thước và hình dạng giống với con người cùng với khả năng di chuyển linh hoạt. Nhiều robot hiện tại được thiết kế bao gồm ba hoặc bốn bánh xe, thiết lập bánh xe cơ sở đủ lớn đề robot có thể đứng cân bằng. Tuy nhiên hình dạng và kích thước của bánh xe sẽ làm giới hạn chiều cao của thân robot do bánh xe cơ sở sẽ phải nhỏ hơn đáng kế so với chiều cao. Đồng thời khi đó sự ổn định trọng tâm của thân robot là tĩnh hay bị động, nếu như có một sự thay đổi nhỏ ở tâm có thể làm cho robot mất ổn định. Điều này có thể khắc phục bằng cách hạ thấp trọng tâm của thân robot hay mở rộng diện tích tiếp xúc với bề mặt di chuyển, tuy nhiên đi kèm với nó là chi phí, khối lượng và hạn chế về tốc độ. Ngoài ra nếu không gian làm việc là có giới hạn về diện tích bề mặt tiếp xúc đòi hỏi chúng ta phải có giải pháp cho vấn đề này.

Một trong số những những giải pháp chính đáng đó chính là sự phát triển của dự án ballbot dựa trên động lực học ổn định. Đó là lý thuyết điều khiển được dùng để đảm bảo rằng robot có thể đứng thẳng mà không cần phải dựa vào sự ổn định tĩnh. Vấn đề ổn định động lực học tương tự đã được sử dụng trên robot hai bánh tự cân hằng Segway.

1.3. Quá trình phát triển

1.3.1. Sự xuất hiện của những robot banh đầu tiên

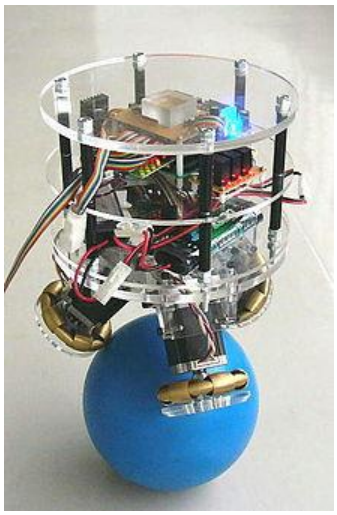
Dự án ballbot đầu tiên được phát triển năm 2005 tại đại học Carnegie Mellon Hoa Kì. Như đã đề cập, ballbot khá đơn giản bao gồm một quả bóng và thân robot có thể đứng thăng bằng trên thân bóng. Hơn nữa robot có thể di chuyển và lái bóng mà

không bị đổ. Phần thân robot hình trụ cao 1.5m, đường kính 400mm nặng 45kg có kích thước gần giống với con người (Hình 1.1).

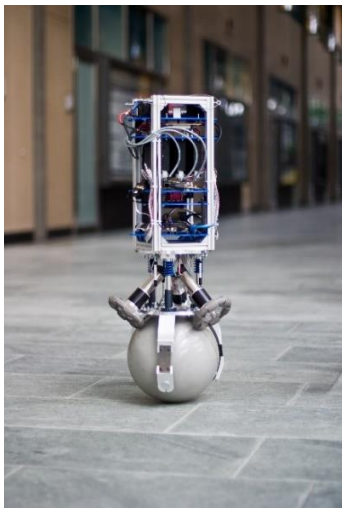
Tiếp theo sau đó là robot được phát triển tại đại học Tohoku Gakuin (TGU) Nhật Bản năm 2008, robot có hình dáng đơn giản hơn so với CMU. BallIP có chiều cao 500mm và nặng 11 kg, quả bóng bowling được phủ lớp cao su và có đường kính khoảng 200mm. Robot sử dụng cơ chế lái phức tạp hơn với ba động cơ đặt với nhau góc 120^0, ba bánh xe đa hướng gắn trên động cơ đặt tiếp xúc với bóng góc 45^0 (Hình 1.2)

Robot thứ ba phát triển tại đại học ETH, Zurich Thụy Sĩ, Rezero có hình dạng về cơ bản tương đối giống với robot của TGU cộng thêm cơ cấu hãm bóng cải tiến về cơ khí. Rezero có chiều cao 1m nặng 14.5 kg lái trên quả bóng nhôm rỗng dày 2mm và được phủ một lớp sơn dày 4mm (Hình 1.3).

Hình 1.1 Ballbot đầu tiên của CMU, 2005



Hình 1.2 BallIP của TGU, 2008



Hình 1.3 Rezero ETH, 2010

1.3.2. Những nghiên cứu trong hệ robot banh

- Bộ điều khiển mà CMU Ballbot sử dụng bao gồm hai vòng lặp trong đó vòng trong là bộ điều khiển PI với tính hiệu hồi tiếp trạng thái là tốc độ góc của bóng và vòng ngoài là bộ điều khiển toàn phương tuyến tính LQR [1].

- Ballbot của TGU sử dụng bộ điều khiển đơn giản vi phân tỉ lệ PD, các hệ số tỉ lệ và hệ số vi phân đều có được từ thực nghiệm [2].
- Một nền tảng được sử dụng phổ biến hơn và có cớ cấu truyền động tương tự CMU ballbot đó là bộ LEGO Mindstorms NXT kit. Mô hình toán của NXT ballbot được mô tả và tuyến tính hóa. Bộ điều khiển LQR được sử dụng, hiệu suất của robot được kiểm nghiệm thực tế [3].
- Ở trường đại học Bách Khoa cũng có những công trình nghiên cứu cho hệ ballbot, các thuật toán như LQR, PID, điều khiển mờ cũng đã được thực hiện

1.4. Giới thiệu luận văn

Luận văn được thực hiện với mục tiêu thiết kế và kiểm chứng bộ điều khiển chế độ trượt SMC cho mô hình ballbot nhằm giữ thăng bằng và kiểm soát vị trí của robot. Ngoài ra một bộ điều khiển toàn phương tuyến tính LQR cũng được sử dụng để có được cái nhìn tổng quát khi ta áp dụng hai bộ điều khiển tuyến tính và phi tuyến trên đối tượng thực. Thuật toán được nhúng trên kit SMT32F407 để điều khiển mô hình thật và so sánh kết quả từ các giải thuật trên. Các vấn đề cơ bản của luận văn gồm có:

- Mô hình hóa ballbot trên các mặt phẳng độc lập dựa trên phương trình Euler-Lagrange tham khảo từ mô hình Rezero của ETH Zurich.
- Tìm hiểu và thiết kế bộ điều khiển LQR, kiểm chứng trên mô hình thực tế.
- Tìm hiểu và thết kế bộ điều khiển trượt và điều khiển trượt phân cấp, kiểm chứng trên mô hình thực tế.
- Trên cơ sở kết quả thu được, phân tích đánh giá và rút ra kết luận về chất lượng bộ điều khiển vừa thiết kế.

Chương 2. MÔ TẢ TOÁN HỌC

Hiện tại theo giới hạn hiểu biết của tác giả, có ít nhất hai cách tiếp cận để xây dựng mô hình toán cho ballbot. Cơ học Lagrange là phương pháp được sử dụng phổ biến nhất và được xuất hiện cùng với robot banh đầu tiên của CMU [1]. Mô hình toán này xây dựng cho cơ cấu lái được vận hành bởi bánh xe hướng về phía trực giao của quả bóng. Cũng với phương trình Euler-Lagrange, mô hình toán cho Rezero của ETH Zurich [4] đã được xây dựng với cơ cấu ba bánh xe đặt lệch nhau góc 120^0. Phương pháp này phân tích hệ thống trên ba mặt phẳng riêng biệt, trong đó mô hình toán của hệ trên Ozy, Ozx được xây dựng tương tự nhau thể hiện góc nghiêng của thân robot và góc xoay của bóng. Và mô hình tương đương trên mặt phẳng Oyz còn lại dùng để xác định góc hướng của robot tự quay quanh trục thẳng đứng. Bên cạnh đó cơ học Newton cho mô hình ballbot cũng đã được các tác giả ở [8] đề xuất. Mặc dù cách tiếp cận này có phần phức tạp hơn nhưng với một số giả định, các phương trình động học ở [8] đưa ra có thể được rút gọn thành mô hình đơn giản mà CMU đã thực hiện. Để việc kiểm soát vị trí của ballbot có thể thực hiện đơn giản và chặt chẽ, mô phần mô tả toán học của hệ thống dưới đây sử dụng cơ học Lagrange trên các mặt phẳng (phương pháp do ETH đề xuất).

2.1. Phương trình Euler-Lagrange

Thay vì lực, đại lượng trung tâm của các định luật Newton, cơ học Lagrange sử dụng khái niệm năng lượng xác định trong hệ. Đại lượng trung tâm của cơ học Lagrange là Lagrangian, hàm tổng kết tính động lực của toàn bộ cơ hệ:

$$L(q, \dot{q}, t) \tag{2.1}$$

Trong đó q là tọa độ tổng quát của hệ thống và t là thời gian. Hàm Lagrange phi tương đối tính của hệ thống được xác định bằng tổng động năng T của hệ thống trừ đi thế năng U, được thể hiện trong phương trình:

$$L = T - U \tag{2.2}$$

Việc lựa chọn tọa độ q là tùy ý, với ràng buộc rằng chúng hoàn toàn phải xác định hệ thống. Tuy nhiên, sẽ là thuận lợi khi sử dụng tọa độ trực tiếp liên quan đến đại lượng đo lường như các góc nghiêng thân robot và góc quay của bánh xe.

Phương trình cơ sở chuyển động, các phương trình Euler Lagrange:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = F_i \qquad (2.3)$$

Trong đó F_i là lực tổng quát. Điều này dẫn đến phương trình Lagrange cho mỗi toạ độ q_i như sau :

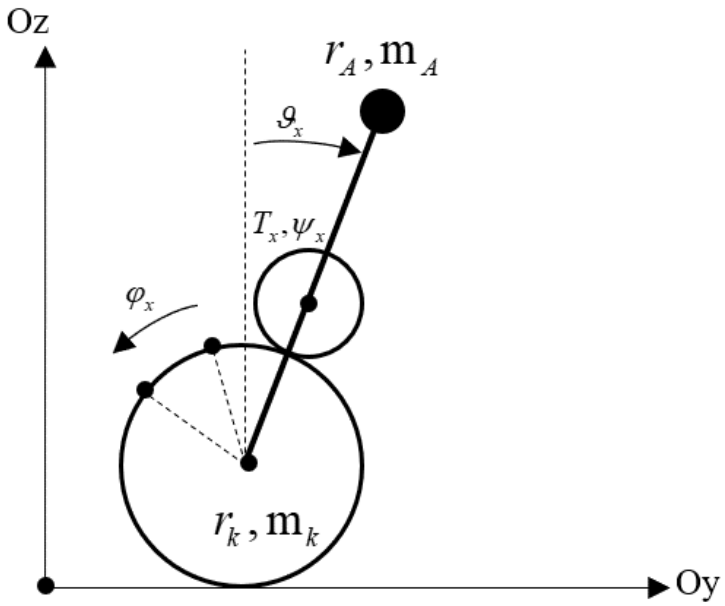
$$M(\mathrm{q},\dot{\mathrm{q}})\ddot{\mathrm{q}} + \mathrm{C}(\mathrm{q},\dot{\mathrm{q}}) + \mathrm{G}\,q = F \qquad (2.4)$$

M là ma trận khối lượng và quán tính, C là ma trận lực Coriolis và G là ma trận trọng lực. Phương trình này là phương trình chuyển động của hệ thống mà có thể là phi tuyến [4].

2.2. Cơ sở để xây dựng mô hình toán

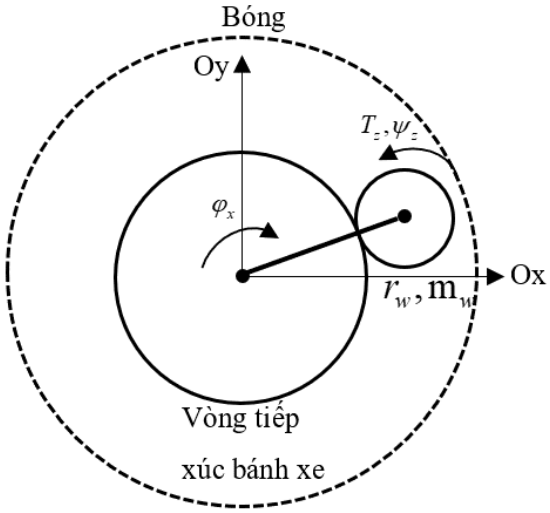
Mô hình toán cho hệ robot banh sau đây được xây dựng dựa trên việc tách rời robot trên ba mặt phẳng riêng biệt, tính toán động học trên các mặt phẳng đó và sử dụng phương trình Euler-Lagrange để tính ràng buộc, mô tả hệ thống.

Hệ tọa độ được chọn để có thể có được mối liên quan trực tiếp đến đại lượng đo lường bao gồm encoder động cơ và IMU. Trên hai mặt phẳng Ozx và Ozy, hình ảnh của ballbot là như nhau và tương tự như hình ảnh của con lắc ngược bao gồm bóng (k), bánh xe đa hướng (w) và thân robot (A) :



Hình 2.1: Mô hình tương đương của robot trên mặt phẳng Oyz

Hình của ballbot chiếu theo phương Oz trên mặt phẳng Oxy được thể hiện như bánh xe xoay quanh thân bóng cố định và quanh trục Oz:



Hình 2.2: Mô hình tương đương của robot trên mặt phẳng Oxy

Để mô tả hệ thống, ta có các đại lượng: $\vartheta_{x,y,z}$ thể hiện góc lệch của thân robot, $\varphi_{x,y,z}$ là góc quay của bóng và $\psi_{x,x,z}$ là góc quay của bánh xe đa hướng. Vector vị trí của robot trên các mặt phẳng có thể được xác định bởi:

$$\vec{q}_{xy} = \begin{bmatrix} \varphi_z \\ \vartheta_z \end{bmatrix},\ \vec{q}_{yz} = \begin{bmatrix} \varphi_x \\ \vartheta_x \end{bmatrix},\ \vec{q}_{xz} = \begin{bmatrix} \varphi_y \\ \vartheta_y \end{bmatrix} \tag{2.5}$$

Với các ràng buộc về vị trí trong hệ tọa độ:

- Đối với mặt phẳng Oxz và tương tự Oyz:

$$Oxy: \begin{cases} x_k = \varphi_x r_k \\ x_{\mathrm{w}} = \varphi_x r_k + \sin(\vartheta_x).(\mathrm{r}_k + \mathrm{r}_{\mathrm{w}}) \\ x_A = \varphi_x r_k + \sin(\vartheta_x).\mathrm{l} \end{cases} Ozy: \begin{cases} y_k = \varphi_y r_k \\ y_{\mathrm{w}} = \varphi_y r_k + \sin(\vartheta_y).(\mathrm{r}_k + \mathrm{r}_{\mathrm{w}}) \\ y_A = \varphi_y r_k + \sin(\vartheta_y).\mathrm{l} \end{cases}$$

- Trong mặt phẳng Oxy, ràng buộc thể thông qua phương trình vận tốc:

$$Oxy: \begin{cases} \dot{\psi}_x = \dfrac{r_k}{r_{\mathrm{w}}}(\dot{\varphi}_x - \dot{\vartheta}_x) - \dot{\vartheta}_x \\ \dot{\psi}_y = \dfrac{r_k}{r_{\mathrm{w}}}(\dot{\varphi}_y - \dot{\vartheta}_y) - \dot{\vartheta}_y \quad (*) \\ \dot{\psi}_z = \dfrac{r_k}{r_{\mathrm{w}}}\sin(\alpha)(\dot{\varphi}_z - \dot{\vartheta}_z) \end{cases} \tag{2.6}$$

2.3. Xây dựng mô hình toán cho robot banh

Động lực học là cần thiết để xây dựng mô hình toán và cho phép mô phỏng. Đầu tiên việc giả định ban đầu để tạo ra một mô hình robot đơn giản trên một mặt phẳng trên đó động lực học có thể được biểu diễn. Sau đó, phương pháp Lagrange được sử dụng để lấy được các phương trình chuyển động. Những phương trình này được tuyến tính và được sử dụng để thiết kế bộ điều khiển. Sau cùng, hoàn chỉnh mô hình tuyến tính động lực học robot.

2.3.1. Ký hiệu và định nghĩa

Nguồn gốc của động lực học dựa vào một số tham số vật lý của robot. Tham số vật lý này được quy định như sau:

- r_k, r_w lần lượt là bán kính bóng và bán kính bánh xe đa hướng
- l là khoảng cách từ tâm khối lượng thân robot đến tâm của bóng
- m_k, m_w, m_A lần lượt là khối lượng của bóng, bánh xe và thân robot
- I_k, I_w, I_A lần lượt là momen quán tính của bóng, bánh xe và thân robot
- $\vartheta_{x,y,z}$ thể hiện góc lệch của thân robot theo các trục x, y, z
- $\varphi_{x,y,z}$ là góc quay của bóng theo các trục x, y, z
- $\psi_{x,x,z}$ là góc quay của bánh xe đa hướng theo các trục x, y, z

2.3.2. Phương trình phi tuyến của robot

Các phương trình chuyển động của mô hình robot trong một mặt phẳng sử dụng phương pháp Lagrange. Biểu diễn phương pháp Lagrange bắt đầu bằng cách xác định động năng và thế năng của thân robot, bóng và các động cơ, như sau:

- Năng lượng của bóng:
  - Trên mặt phẳng Oyz và tương tự Oxz, động năng của bóng bao gồm chuyển động tịnh tiến và chuyển động xoay.

$$T_{K,yz} = \frac{1}{2} m_k (r_k \dot{\varphi}_x)^2 + \frac{1}{2} I_K \dot{\varphi}_x^2 \qquad (2.7)$$

    Chọn gốc tọa độ tại tâm, thế năng của bóng $V_{k,yz} = 0$

  - Trên mặt phẳng Oxy năng lượng chỉ bao gồm động năng xoay quanh trục Oz

$$T_{k,xy} = \frac{1}{2} I_K \dot{\varphi}_z^2 \qquad (2.8)$$

- Năng lượng của bánh xe:

- Trên mặt Oyz và tương tự Oxy, động năng của bánh xe đa hướng bao gồm động năng tịnh tiến của bánh xe cùng với động năng tịnh tiến tự do của các khớp nối có hướng vuông góc và động năng xoay quanh trục động cơ.

$$T_{\mathrm{w,yz}} = \frac{1}{2} m_{\mathrm{w}} \left((\mathrm{r}_k \dot{\varphi}_x)^2 + 2(\mathrm{r}_k + \mathrm{r}_{\mathrm{w}}) \cos(\vartheta_x) \dot{\vartheta}_x (\mathrm{r}_k \dot{\varphi}_x) + (\mathrm{r}_k + \mathrm{r}_{\mathrm{w}})^2 \dot{\vartheta}_x^2 \right) + \frac{1}{2} I_{\mathrm{w}} \left(\frac{r_k}{r_{\mathrm{w}}} \left(\dot{\varphi}_x - \dot{\vartheta}_x \right) - \dot{\vartheta}_x \right)^2 \tag{2.9}$$

Thế năng của bánh xe:

$$V_{\mathrm{w,yz}} = m_{\mathrm{w}} g (\mathrm{r}_k + \mathrm{r}_{\mathrm{w}}) \cos(\vartheta_x) \tag{2.10}$$

- Trên mặt Oxy năng lượng là động năng xoay quanh trục Oz :

$$T_{\mathrm{w},xy} = \frac{1}{2} I_{\mathrm{w}} \dot{\psi}_z^2 \tag{2.11}$$

- Năng lượng của thân robot :
  - Trên mặt Oyz, Oxz động năng thân robot :

$$T_{\mathrm{A,yz}} = \frac{1}{2} m_A \left((\mathrm{r}_k \dot{\varphi}_x)^2 + 2l \cos(\vartheta_x) \dot{\vartheta}_x (\mathrm{r}_k \dot{\varphi}_x) + l^2 \dot{\vartheta}_x^2 \right) + \frac{1}{2} I_A \dot{\vartheta}_x^2 \tag{2.12}$$

Thế năng:

$$V_{\mathrm{A,yz}} = m_A g l \cos(\vartheta_x) \tag{2.13}$$

  - Trên mặt Oxy động năng xoay:

$$T_{\mathrm{A},xy} = \frac{1}{2} I_A \dot{\vartheta}_z^2 \tag{2.14}$$

Đại lượng Lagrange sẽ được xây dựng trong từng mặt phẳng với :

$$L = T_k + T_{\mathrm{w}} + T_A - V_k - V_{\mathrm{w}} - V_A \tag{2.15}$$

Phương trình Euler-Lagrange:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right)-\frac{\partial L}{\partial q_i}=F_i \tag{2.16}$$

Lực F tác động lên mô hình chính là momen xoắn T của động cơ, truyền trực tiếp đến góc xoay của bánh xe đa hướng.

Xét phương trình Lagrange trong mặt Oyz (tương tự với Oxz):

Lực F tác động lên góc quay của bánh xe đa hướng và góc lệch của thân robot lần lược là:

$$F_{yz1}=\begin{bmatrix}\frac{r_k}{r_w}T_x\\ -(\frac{r_k}{r_w}+1)\mathrm{T}_x\end{bmatrix};F_{yz2}=\begin{bmatrix}0\\ T_x\end{bmatrix}\Rightarrow F_{yz}=F_{yz1}+F_{yz2}=\begin{bmatrix}\frac{r_k}{r_w}T_x\\ -\frac{r_k}{r_w}\mathrm{T}_x\end{bmatrix} \tag{2.17}$$

Đại lượng Lagrange:

$$L_{yz}=T_{w,yz}+T_{A,yz}+T_{k,yz}-V_{w,yz}-V_{A,yz}-V_{k,yz} \tag{2.18}$$

Khi đó ta có phương trình Lagrange :

$$\frac{d}{dt}\left(\frac{\partial L_{yz}}{\partial \dot{q}_x}\right)-\frac{\partial L_{yz}}{\partial q_x}=F_{yz} \tag{2.19}$$

Kết quả đạo hàm trong phương trình chuyển động ta được phương trình :

$$M_x(\mathrm{q},\dot{\mathrm{q}})\ddot{\mathrm{q}}+\mathrm{C}_x(\mathrm{q},\dot{\mathrm{q}})+\mathrm{G}_x\,q=F_{yz}(*) \tag{2.20}$$

Trong đó ma trận khối lượng lượng và quán tính M_x, ma trận lực Coriolis C_x và ma trận trọng lực G_x lần lược là :

$$M_x = \begin{bmatrix} (m_k + m_A + m_w)r_k^2 + I_k + \left(\frac{r_k}{r_w}\right)^2 I_w & -\frac{r_k}{r_w^2}(r_k + r_w)I_w + \lambda r_k \cos\vartheta_x \\ -\frac{r_k}{r_w^2}(r_k + r_w)I_w + \lambda r_k \cos\vartheta_x & \frac{(r_k + r_w)^2}{r_w^2} I_w + I_A + m_A l^2 + m_w (r_k + r_w)^2 \end{bmatrix}$$

$$C_x = \begin{bmatrix} -r_k \lambda \sin(\vartheta_x \dot{\vartheta}_x^2) \\ 0 \end{bmatrix}; G_x = \begin{bmatrix} 0 \\ -g \sin(\vartheta_x \lambda) \end{bmatrix}; \{\lambda = l.m_A + (r_k + r_w)m_w\}$$

Xét phương trình Lagrange trong mặt phẳng Oxy:

Lực F tác động lên góc quay của bánh xe đa hướng và góc lệch của thân robot lần lược là:

$$F_{xy1} = \begin{bmatrix} \frac{r_k}{r_w}\sin(\alpha)T_z \\ -\frac{r_k}{r_w}\sin(\alpha)T_z \end{bmatrix}; F_{xy2} = \begin{bmatrix} 0 \\ T_z \end{bmatrix} \quad (2.21)$$

Ngoài ra lực ma sát T_f giữa bóng và mặt đất được xét đến khi khảo sát mô hình tự quay quanh Oz. Khi đó ta có :

$$F_{xy} = F_f + F_{xy1} + F_{xy2} = \begin{bmatrix} -T_f + \frac{r_k}{r_w}\sin(\alpha)T_z \\ \left(1 - \frac{r_k}{r_w}\sin(\alpha)\right)T_z \end{bmatrix} \quad (2.22)$$

T_f mang dấu âm do ma sát có xu hướng chống lại lực tác động F

Đại lượng Lagrange:

$$L_{xy} = T_{w,xy} + T_{A,xy} + T_{k,xy} - V_{w,xy} - V_{A,xy} - V_{k,xy} \quad (2.23)$$

Khi đó ta có phương trình Lagrange :

$$\begin{cases} \dfrac{d}{dt}\left(\dfrac{\partial L_{xy}}{\partial \dot{\varphi}_z}\right) - \dfrac{\partial L_{xy}}{\partial \varphi_z} = -T_f + \dfrac{r_k}{r_w}\sin(\alpha)T_z \\ \dfrac{d}{dt}\left(\dfrac{\partial L_{xy}}{\partial \dot{\vartheta}_z}\right) - \dfrac{\partial L_{xy}}{\partial \vartheta_z} = \left(1 - \dfrac{r_k}{r_w}\sin(\alpha)\right)T_z \end{cases} \quad (2.24)$$

Kết quả của đạo hàm ta được :

$$\ddot{\varphi}_z = -\frac{r_w^2 I_{A,xy} + r_k^2 I_{w,xy}\sin^2(\alpha)T_f + r_k r_w I_{A,xy}\sin(\alpha)T_z}{r_w^2 I_{A,xy} I_k + r_k^2 (I_{A,xy} + I_k) I_{w,xy}\sin^2(\alpha)} \quad (2.25)$$

$$\ddot{\vartheta}_z = -\frac{r_k \sin(\alpha)\left(r_k I_{w,xy}\sin(\alpha)T_f + r_w I_k T_z\right)}{r_w^2 I_{A,xy} I_k + r_k^2 (I_{A,xy} + I_k) I_{w,xy}\sin^2(\alpha)} \quad (2.26)$$

Hệ trục tọa độ của thân robot thực tế được xây dựng bằng cách ánh xạ hệ tọa độ gốc theo các phương z, x, y với góc xoay có thể đo được bằng cảm biến IMU.

Giả sử trục tọa độ gốc có tâm tại mặt đất là $O_I x_I y_I z_I$ và robot đã xoay lệch hướng so với vị trí ban đầu các góc $\vartheta_x, \vartheta_y, \vartheta_z$ khi đó :

- Hệ tọa độ $O_L x_L y_L z_L$ thể thiện góc xoay robot theo trục Oz có được bằng cách xác định tâm O_L là tâm quả bóng, giữ nguyên Oz ($O_L z_L$ trùng với $O_I z_I$) và xoay hệ trục quanh $O_I z_I$ một góc ϑ_z
- Hệ tọa độ $O_{A'} x_{A'} y_{A'} z_{A'}$ thể hiện lệch góc của robot theo trục Oy có được bằng cách giữ nguyên Oy ($O_L y_L$ trùng với $O_{A'} y_{A'}$) và xoay hệ trục quanh $O_L y_L$ một góc ϑ_y. Tâm $O_{A'}$ có thể trùng với tâm quả bóng.
- Sau cùng, hệ tọa độ sau của robot $O_A x_A y_A z_A$ thể hiện góc lệch của robot theo trục Ox có được bằng cách xác đinh tâm O_A là trọng tâm của thân robot, giữ nguyên trục Ox ($O_A x_A$ trùng với $O_{A'} x_{A'}$) và xoay hệ trục quanh $O_{A'} x_{A'}$ một góc ϑ_x

Hệ tọa độ này được dùng để xác định vị trí của robot khi di chuyển trong không gian.

2.3.3. Tuyến tính hóa

Để xây dựng bộ điều khiển tuyến tính cho ballbot, phương pháp thiết kế được sử dụng đòi hỏi phương trình trạng thái tuyến tính. Trước tiên tuyến tính hóa hệ thống trong mặt phẳng Oyz, mô hình trong mặt phẳng Oxz sẽ được xây dựng tương tự.

Xét hệ thống trên mặt phẳng Oyz có vector trạng thái:

$$x = \begin{bmatrix} \varphi_x & \vartheta_x & \dot{\varphi}_x & \dot{\vartheta}_x \end{bmatrix}^T \tag{2.27}$$

Từ (*) ta đã có phương trình phi tuyến của hệ thống với biến trạng thái q, biểu diễn trạng thái của hệ thống trên mặt phẳng Oyz được viết lại:

$$\dot{x} = f(\mathrm{x}) = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} \dot{\varphi}_x \\ \dot{\vartheta}_x \\ f_3 \\ f_4 \end{bmatrix} \text{ với } \begin{bmatrix} f_3 \\ f_4 \end{bmatrix} = M_x^{-1}(\mathrm{F}_x - (\mathrm{C}_x + \mathrm{G}_x)) \tag{2.28}$$

Trong đó $M_x = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}; C_x = \begin{bmatrix} C_{11} \\ C_{21} \end{bmatrix}; G_x = \begin{bmatrix} G_{11} \\ G_{21} \end{bmatrix}$

$$M_{11} = (m_k + m_A + m_w)\,r_k^2 + I_k + \left(\frac{r_k}{r_w}\right)^2 I_w$$

$$M_{12} = M_{21} = -\frac{r_k}{r_w^2}(r_k + r_w)\,I_w + (l.m_A + (r_k + r_w)\,m_w)\,r_k \cos\vartheta_x$$

$$M_{22} = \frac{(r_k + r_w)^2}{r_w^2} I_w + I_A + m_A l^2 + m_w (r_k + r_w)^2$$

$$C_{11} = -r_k (l.m_A + (r_k + r_w)\,m_w)\sin(\vartheta_x \dot{\vartheta}_x^2)$$

$$C_{21} = 0$$

$$G_{11} = 0$$

$$G_{21} = -g\sin(\vartheta_x (l.m_A + (r_k + r_w)\,m_w))$$

$$F_{11} = -F_{21} = (r_k / r_w)\,T_x$$

Khi đó, sử dụng ma trận Jacobi để xấp xỉ tuyến tính tại điểm làm việc x=0 ta được phương trình trạng thái viết ở dạng ma trận:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}; A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & A_{32} & 0 & A_{34} \\ 0 & A_{42} & 0 & 0 \end{bmatrix}; B = \begin{bmatrix} 0 \\ 0 \\ B_{31} \\ B_{41} \end{bmatrix}; \qquad (2.29)$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Chương 3. CƠ SỞ LÝ THUYẾT ĐIỀU KHIỂN

3.1. Nguyên lý hoạt động của xe một bánh

Mô hình robot cân bằng trên banh với mô tả toán học như ở chương 2 có thể xem tương tự như robot hai bánh xe hay con lắc ngược, tuy nhiên cơ cấu chuyển động của ballbot chỉ sử dụng một quả bóng để giữ thăng bằng và dịch chuyển. Ưu điểm này giúp cho việc chuyển hướng linh hoạt hơn nhưng đồng thời nó cũng đặt ra những đòi hỏi cao hơn cho bộ điều khiển trong việc giữ thăng bằng và di chuyển.

Về cơ bản, khi không điều khiển:

- Nếu robot thăng bằng thì góc nghiêng ϑ của thân robot so với trục thẳng đứng bằng không.
- Nếu robot nghiêng về phía trước (thẳng hướng động cơ và bánh xe thứ nhất) $\vartheta > 0$ theo quán tính bóng sẽ lăn về phía sau hai bánh xe sau sẽ xoay cùng chiều hướng ngược với hướng lăn của bóng dẫn đến robot sẽ ngã về trước (giả sử robot luôn gắn chặt với bóng). Như vậy ở trường hợp này ta cần điều khiển hai động cơ sau (2 và 3) để bóng lăn theo hướng ngược lại bên trên để robot thăng bằng trở lại.
- Tương tự như khi robot nghiêng về phía thẳng hướng của hai động cơ còn lại ($\pm 120^0$ so với góc đề cập bên trên) hai động cơ sau cũng cần phải được điều khiển tương tự để giữ robot thăng bằng.

Luận văn tập trung trình bày vào hai giải thuật điều khiển đó là LQR và điều khiển chế độ trượt. Các mục tiếp theo sẽ trình bày lý thuyết cho hai giải thuật này, làm tiền đề để tiếp cận xây dựng bộ điều khiển robot cân bằng.

3.2. Bộ điều khiển toàn phương tuyến tính LQR

Điều khiển toàn phương tuyến tính LQR (Linear Quadratic Regulator) là bộ điều khiển đơn giản và được đề xuất đầu tiên cho hệ robot banh.

Xét hê thống có biểu diễn trạng thái:

$$\dot{x} = Ax + Bu \tag{3.1}$$

Vấn đề đặt ra: xác định luật điều khiển hồi tiếp tất cả trạng thái:

$$u = -Kx \tag{3.2}$$

Để đưa vector trạng thái x về không sao cho cực tiểu hàm mục tiêu J:

$$J = \frac{1}{2}\int_0^\infty (x^T Qx + u^T Ru)dt \tag{3.3}$$

Trong đó :

Q là ma trận đối xứng xác định không âm, R là ma trận xác định dương.

Do Q đối xứng và xác định không âm, ta luôn tìm được ma trận C sao cho :

$$Q = C^T C \tag{3.4}$$

Giả sử (A,B) điều khiển được, luật điều khiển tối ưu được xác định bởi phương trình (3.2) với :

$$K = R^{-1}B^T P \tag{3.5}$$

P là ma trận (nxn) đối xứng, xác định dương là nghiệm của phương trình Riccati :

$$PA + A^T P + Q - PBR^{-1}B^T P = 0 \tag{3.6}$$

Nghiệm phương trình Riccati :

$$H = \begin{bmatrix} A & BR^{-1}B^T \\ -Q & A^T \end{bmatrix} \tag{3.7}$$

Ma trận H (2n x 2n) có tính chất như sau :

- H có n trị riêng với phần thực dương và n trị riêng với phần thực âm

- Nếu λ và $\begin{bmatrix} p_1 \\ p_2 \end{bmatrix}$ là trị riêng và vector trị riêng bên phải của H, thì $-\lambda$ và $\begin{bmatrix} p_2^T & -p_1^T \end{bmatrix}$ là trị riêng và vector riêng bên trái của H.

$$H\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \lambda\begin{bmatrix} p_1 \\ p_2 \end{bmatrix}$$

$$\begin{bmatrix} p_2^T & -p_1^T \end{bmatrix} H = -\lambda\begin{bmatrix} p_2^T & -p_1^T \end{bmatrix}$$

Gọi Λ là ma trận đường chéo (n x n) chứa các trị riêng với phần thực âm của H và X là ma trận (2n x 2n) các vector riêng tương ứng. Phân hoạch X:

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{3.9}$$

Với X_1 và X_2 là các ma trận (n x n). Ta có :

$$\begin{bmatrix} A & BR^{-1}B^T \\ -Q & A^T \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}\Lambda \tag{3.10}$$

Nghiệm của phương trình Riccati (3.6) được cho bởi :

$$P = X_2 X_1^{-1} \tag{3.11}$$

Chứng minh: từ (3.10) ta có :

$$\begin{aligned} & AX_1 - BR^{-1}B^T X_2 = X_1\Lambda \\ \Leftrightarrow\ & A - BR^{-1}B^T X_2 X_1^{-1} = X_1\Lambda X_1^{-1} \\ \Leftrightarrow\ & A - BR^{-1}B^T P = X_1\Lambda X_1^{-1} \\ \Leftrightarrow\ & PA - PBR^{-1}B^T P = PX_1\Lambda X_1^{-1} = X_2\Lambda X_1^{-1} \end{aligned} \tag{3.12}$$

Mặt khác ta còn có :

$$
\begin{aligned}
&-QX_1 - A^T X_2 = X_2\Lambda \\
&\Leftrightarrow -Q - A^T X_2 X_1^{-1} = X_2\Lambda X_1^{-1} \\
&\Leftrightarrow -Q - A^T P = X_2\Lambda X_1^{-1}
\end{aligned}
\tag{3.13}
$$

Từ (3.12) và (3.13) ta có được phương trình ở (3.6):

$$PA + A^T P + Q - PBR^{-1}B^T P = 0$$

Cũng từ (3.6), (3.5) và luật điều khiển (3.2) ta có :

$$
\begin{aligned}
&PA + A^T P + Q - PBR^{-1}B^T P = 0 \\
&\Leftrightarrow PA + A^T P + Q - PBK = 0 \\
&\Leftrightarrow P\left[A - BK\right] + \left[A - BK\right]^T P + Q - K^T B^T P = 0
\end{aligned}
$$

P là nghiệm phương trình Lyapunov và P đối xứng, xác định dương.

Ma trận trạng thái của hệ kín được cho bởi:

$$A - BK = A - BR^{-1}B^T P = X_1\Lambda X_1^{-1} \tag{3.14}$$

(3.14) là phân hoạch trị riêng của ma trận trạng thái hệ thống vòng kín A – BK có các trị riêng là các phần tử của Λ => hệ kín ổn định.

3.3. Bộ điều khiên chế độ trượt

Điều khiển chế độ trượt SMC là một kĩ thuật điều khiển phi tuyến có đặc tính kĩ thuật đáng chú ý về tính chính xác, mạnh mẽ, dễ điều chỉnh và thực hiện.

Các hệ thống SMS được thiết kế để điều khiển trạng thái hệ thống lên một bề mặt cụ thể trong không gian trạng thái gọi là mặt trượt. Một khi đã đạt đến bề mặt trượt, điều khiển chế độ trượt giữ trạng thái trên vùng lân cận gần của bề mặt trượt. Do đó điều khiển chế độ trượt là thiết kế bộ điều khiển gồm có hai phần. Phần đầu tiên liên quan đến việc thiết kế bề mặt trượt để chuyển động trượt đáp ứng các thông số kỹ thuật

thiết kế. Thứ hai là việc lựa chọn luật điều khiển có thể thực hiện được trên hệ thống thực đồng thời đảm bảo tính chất của mặt trượt [6].

Có hai ưu điểm chính của điều khiển trượt, trước tiên là tính linh động, trạng thái của hệ thống có thể được điều chỉnh bởi việc lựa chọn mặt trượt. Thứ hai là đáp ứng vòng kín giúp cho hệ thống không quá nhạy cảm với các thành phần không chắc chắn của mô hình.

Trên quan điểm thực tế, SMC là giải pháp cho phép điều khiển quá trình phi tuyến, có nhiễu và sự không chắc chắn trong giới hạn cho phép của mô hình. Dưới đây là ứng dụng của SMC cho việc điều khiển cân bằng ballbot.

3.3.1. Lý thuyết điều khiển trượt cơ bản

Xét hệ thống phi tuyến một ngõ vào một ngõ ra (SISO):

$$\begin{cases} \dot{x} = f(\mathrm{x},\mathrm{t}) + \mathrm{g}(\mathrm{x},\mathrm{t})\,\mathrm{u} \\ y = h(\mathrm{x},\mathrm{t}) \end{cases} \tag{3.15}$$

Trong đó y, u biểu diễn ngõ ra vô hướng và biến ngõ vào, $x \in R^n$ biểu thị vector trạng thái của hệ thống.

Mục đích của việc điều khiển là làm cho biến đầu ra y bám theo y_{des} mong muốn. Có nghĩa là sai lệch ngõ ra e = y - y_{des} có xu hướng tiến về vùng lân cận nhỏ bằng 0 sau một khoảng thời gian chấp nhận được.

Như đã đề cập, điều khiển chế độ trượt SMC bao gồm hai giai đoạn:

- Định nghĩa mặt trượt:

  Giai đoạn đầu tiên là định nghĩa hàm vô hướng nhất định của trạng thái hệ thống:

$$\sigma(\mathrm{x}) = \mathrm{R}^n \rightarrow R \tag{3.16}$$

Thông thường mặt trượt phụ thuộc vào sai lệch ngõ ra e_y cùng với số lượng nhất định các đạo hàm của nó: $\sigma = \sigma(e, \dot{e}, \ddot{e}, ..., e^{(k)})$

Hàm σ nên được chọn sao cho σ có xu hướng tiến về không, dẫn đến phương trinh vi phân ổn định với bất kì hàm sai lệch ngõ ra $e_y(t)$ nào.

Các hàm vô hướng được lựa chọn điển hình nhất:

$$\begin{aligned} \sigma &= \dot{e} + c_0 e \\ \sigma &= \ddot{e} + c_1 \dot{e} + c_0 \\ \sigma &= e^{(k)} + \sum_{i=0}^{k-1} c_i e \end{aligned} \qquad (3.17)$$

Theo quan điểm hình học, phương trình $\sigma = 0$ định nghĩa bề mặt trong không gian sai lệch, được gọi là bề mặt trượt. Các quỹ đạo của hệ thống điều khiển ràng buộc lên bề mặt trượt, theo đó ngõ ra hệ thống đáp ứng được thông số kĩ thuật thiết kế.

Sự lựa chọn tham số dương c_i gần như là tùy ý và xác định cực duy nhất dẫn đến kết quả "giảm động lực" của hệ thống khi trượt. c_i được chọn cần đảm bảo đa thức đặt trưng của phương trình vi phân sau Hurwitz (tất cả các nghiệm có phần thực âm):

$$e^{(k)} + c_{k-1} e^{(k-1)} + ... + c_1 \dot{e} + c_0 e = 0 \qquad (3.18)$$

Trái lại, số nguyên k lại rất quan trọng, nó phải bằng n -1, với n là bậc tương đối của hệ thống giữa ngõ ra và đầu vào.

Một cách tường minh, hàm vô hướng S (σ) được định nghĩa:

$$S = e^{(n-1)} + c_{n-2} e^{(n-2)} + ... + c_1 \dot{e} + c_0 e$$

Từ phương trình mô tả hệ thống (3.15) và $e_i = y_i - r_i$ ta được:

$$S = (x_n + c_{n-2} x_{n-1} + ... + c_1 x_2 + c_0 x_1) - (r^{(n-1)} + c_{n-2} r^{(n-2)} + ... + c_1 \dot{r} + a_0 r) \qquad (3.19)$$

- Thiết kế ngõ vào điều khiển:

Vấn đề đặt ra là xác định luật điều khiển u để đưa quỹ đạo pha của hệ thống về mặt trượt và duy trì trên mặt trượt một cách bền vững đối với mọi biến động của f(x) và g(x).

Đạo hàm (3.19) theo thời gian ta có:

$$\dot{S} = f(x) + g(x)u + c_{n-2}(x_n - r^{(n-1)}) + ... + c_1(x_3 - \ddot{r}) + c_0(x_2 - \dot{r}) - r^{(n)} \quad (3.20)$$

Có thể chọn u sao cho $\dot{S} = -K\,\mathrm{sgn}(S)$ (3.21)

Trong đó K là một hằng số dương chọn trước. Luật điều khiển được xác định:

$$u = -[f(x) + c_{n-2}(x_n - r^{(n-1)}) + ... + c_1(x_3 - \ddot{r}) + c_0(x_2 - \dot{r}) - r^{(n)} + K\,\mathrm{sgn}(S)]g^{-1}(x) \quad (3.22)$$

Tính bền vững của luật điều khiển: Trong điều kiện có sai số mô hình, luật điều khiển (3.22) luôn đưa quỹ đạo pha của hệ thống về mặt trượt S = 0 nếu điều kiện sau được thỏa mãn:

- Nếu S > 0 thì $\dot{S} < 0$
- Nếu S < 0 thì $\dot{S} > 0$
- Nếu S = 0 thì $\dot{S} = 0$

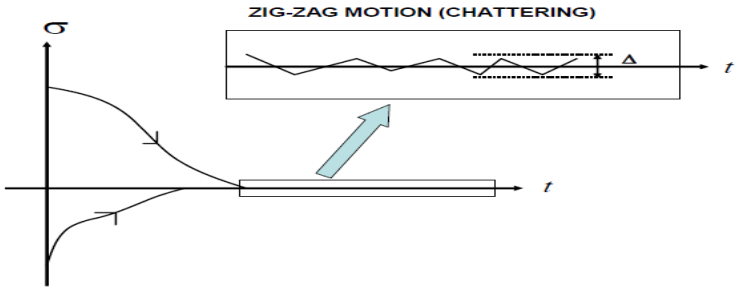
Việc lựa chọn mặt trượt cần thỏa mãn hai điều kiện sau [5]:

- S không phụ thuộc tường minh vào u nhưng $\dot{S}$ phụ thuộc tường minh vào u
- Phương trình vi phân (3.18) Hurwitz để nghiệm để $e \to 0$ khi $t \to \infty$

Một số cách tiếp cận dựa trên phương pháp kiểm soát chế độ trượt [6]:

- Điều khiển chế độ trượt tiêu chuẩn:

  Điều khiển không liên tục $u = -K\,sign(\sigma)$



Hình 3.1: Biến σ theo thời gian [6]

Khi đó: $u = \begin{cases} -K & \sigma > 0 \\ K & \sigma < 0 \end{cases}$ K là hằng số dương đủ lớn.

Khi ở trạng thái ổn định, biến điều khiển u sẽ trở lại tần số rất cao (về lý thuyết xem như vô hạn). Để giải quyết vấn đề trên (hiện tượng Chattering) các kỹ thuật điều khiển chế độ trượt được đề xuất, hàm dấu sign được thay thế bằng một hàm xấp xỉ liên tục khác:

SAT $u = -Ksat(\sigma, \varepsilon) \equiv -\mathrm{K}\dfrac{\sigma}{|\sigma| + \varepsilon} \quad \varepsilon > 0, \varepsilon \approx 0$

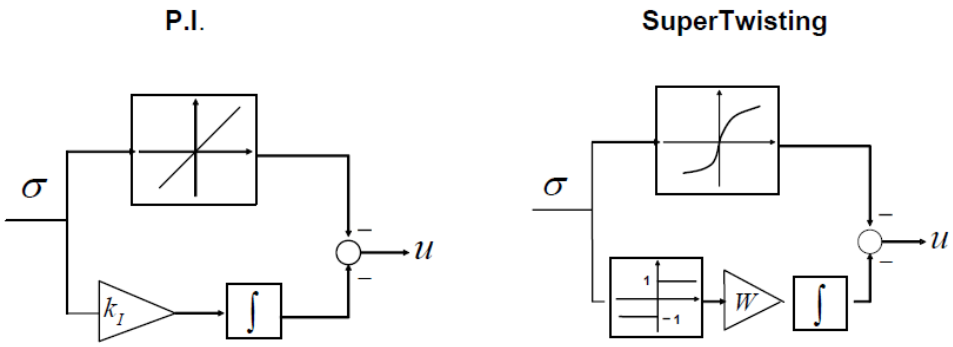
TANH $u = -K\tanh(\sigma / \varepsilon) \quad \varepsilon > 0, \varepsilon \approx 0$

- Điều khiển chế độ trượt bậc cao:

Thuật toán điều khiển chế độ trượt bậc hai là một giải pháp thay thế mạnh mẽ giúp giải quyết các vấn đề bên trên mà không ảnh hưởng đến tinh chất của bộ điều khiển trượt: (2-SMC "SupperTwisting")

$$u = -\lambda\sqrt{|\sigma|}sgn(\sigma) + \mathrm{w} \qquad \lambda = \sqrt{K} \quad \mathrm{w} = 1.1K$$
$$\mathrm{w} = -\mathrm{W}\,\mathrm{sgn}(\sigma)$$

K là hằng số dương được lấy đủ lớn.



Hình 3.2: Sơ đồ mặt trượt cổ điển (trái) và Super_Twisting (phải) [6]

3.3.2. Điều khiển chế độ trượt phân cấp

Lý thuyết điều khiển trượt cơ bản có thể giải quyết được hệ thống SISO một ngõ vào một ngõ ra. Tuy nhiên để ý phương trình mô tả hệ robot banh với vector trạng thái $x = \left[\varphi_x \quad \dot{\varphi}_x \quad \vartheta_x \quad \dot{\vartheta}_x \right]^T$:

$$\begin{cases} \dot{x}_1 = \mathrm{x}_2 \\ \dot{x}_2 = f_2(\mathrm{x}_3, \mathrm{x}_4) + g_2(\mathrm{x}_3)\,\mathrm{u} \\ \dot{x}_3 = \mathrm{x}_4 \\ \dot{x}_4 = f_4(\mathrm{x}_3, \mathrm{x}_4) + g_4(\mathrm{x}_3)\,\mathrm{u} \end{cases} \tag{3.23}$$

Ta thấy hệ thống có một ngõ vào điều khiển u nhưng cần điều khiển nhiều hơn một ngõ ra trạng thái bao gồm góc nghiêng ϑ và vị trí ϕ (under-actuated system).

Để giải quyết bài toán trên, phương pháp điều khiển chế độ trượt phân cấp HSMC được đưa ra cho hệ thống SIMO. Hệ thống được phân cấp thành các hệ con, và mặt trượt cho mỗi hệ con được xác định. Mặt trượt của hệ con đầu tiên được chọn làm bề mặt trượt của lớp đầu tiên, nó dùng để xây dựng bề mặt trượt của lớp thứ hai với bề mặt trượt của hệ thống con thứ hai. Cứ như vậy cho đến khi tất cả các mặt trượt đều được nhắc đến [7].

Xét hệ thống SIMO được biểu diễn bởi phương trình trạng thái:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f_1 + b_1 u \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = f_2 + b_2 u \\ \vdots \\ \dot{x}_{2n-1} = x_{2n} \\ \dot{x}_{2n} = f_n + b_n u \end{cases} \quad (3.24)$$

Trong đó $x = [x_1, x_2, ..., x_n]^T$ là vector trạng thái của hệ thống, f_i, b_i là các hàm phi tuyến, u là tín hiệu điều khiển. Với n khác nhau ta có các thống khác nhau như hệ con lắc, con lắc ngược đôi, ba …. Một cách tổng quát, phương trình trạng thái của hệ thống có dạng bên trên có thể viết lại :

$$\begin{cases} \dot{x}_{2i-1} = x_{2i} \\ \dot{x}_{2i} = f_i + b_i u \end{cases} \quad (3.25)$$

Mặt trượt tổng quát cho các hệ con :

$s_i = c_i e_i + \dot{e}_i$ trong đó e_i là các sai lệch khi điều khiển trượt bám hoặc

$s_i = c_i x_{2i-1} + x_{2i}$ c_i xác định dương (3.26)

khi điều khiển chế độ trượt ổn định hóa.

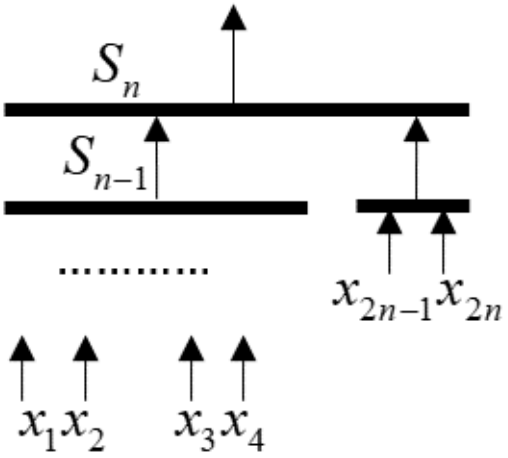
Đạo hàm (3.26) theo thời gian ta có:

$$\dot{s}_i = c_i \dot{x}_{2i-1} + \dot{x}_{2i} = c_i \dot{x}_{2i} + f_i + b_i \quad (3.27)$$

Trên mặt trượt $\dot{s}_i = 0$ xác định được tín hiệu điều khiển tương đương cho các hệ con :

$$u_{eqi} = -(\dot{x}_{2i} + f_i) b_i^{-1} \quad (3.28)$$

Điều khiển chế độ trượt phân cấp có thể được thiết kế như sau : mặt trượt của một hệ con được chọn như là mặt trượt lớp đầu tiên S_1. Sau đó S_1 được sử dùng để xây dựng mặt trượt lớp thứ hai S_2 với mặt trượt của hệ con thứ hai. Tiếp tục cho đến khi ta đạt được tất cả các mặt trượt [7] :



Hình 3.3: Cấu trúc phân lớp các mặt trượt

Với cách thiết kế trên, mặt trượt ở lớp thứ i sẽ chứa thông tin của mặt trượt lớp trước đó cùng với mặt trượt của hệ con hiện tại. Một cách tổng quát:

$$S_i = a_{i-1} S_{i-1} + s_i \tag{3.29}$$

Với a_0, $S_0 = 0$ và a_i là hằng số cần thiết kế.

Hay $S_i = \sum_{r=1}^{i} \left(\prod_{j=r}^{i} a_j \right) s_r$ với $a_i = 1$ (3.40)

Tương tự, luật điều khiển của lớp i cần phải chứa thông tin của các lớp trước đó cùng với luật điều khiển của hệ thống con hiện tại.

$$u_i = u_{i-1} + u_{eqi} + u_{swi} \tag{3.41}$$

$$\Rightarrow u_n = \sum_{l=1}^{n-1} u_{swl} + \sum_{l=1}^{n-1} u_{eql} + u_{swn} \tag{3.42}$$

Với $u_0 = 0$ và u_{swi} mang ý nghĩa là luật chuyển đổi của mặt trượt.

Luật điều khiển trên có nguồn gốc từ lý thuyết ổn định Lyapunov. Hàm Lyapunov của lớp thứ i được xác định bởi :

$$V_i(\mathrm{t}) = \mathrm{S}_i^2 / 2 \tag{3.43}$$

Đạo hàm V_i theo thời gian ta có:

$$\dot{V}_i(\mathrm{t}) = \mathrm{S}_i\, \dot{S}_i = S_i(a_{i-1}\dot{S}_{i-1} + \dot{s}_i) = \mathrm{S}_i \sum_{r=1}^{i}\left(\prod_{j=r}^{i} a_j\right)\dot{s}_r \tag{3.44}$$

Khai triển $\dot{V}_i(\mathrm{t})$ từ (3.27), (3.28), (3.41), (3.44) ta có được dạng tổng quát :

$$\dot{V}_i(\mathrm{t}) = S_i\left\{\sum_{l=1}^{i}\left[\sum_{\substack{r=1\\ r\neq l}}^{i}\left(\prod_{j=r}^{i} a_j\right)b_r\right]u_{eql} + \sum_{l=1}^{i}\left[\sum_{r=1}^{i}\left(\prod_{j=r}^{i} a_j\right)b_r\right]u_{swl}\right\} \tag{3.45}$$

Khi hàm V xác định như trên, để ý thấy với

$$\dot{\mathrm{S}}_i = -k_i S_i - \eta_i\, \mathrm{sgn}(\mathrm{S}_i) \tag{3.46}$$

$$\Rightarrow \dot{V}_i(\mathrm{t}) = \mathrm{S}_i(-k_i S_i - \eta_i\, \mathrm{sgn}(\mathrm{S}_i)) = -\mathrm{k}_i \mathrm{S}_i^2 - \eta_i S_i\, \mathrm{sgn}(\mathrm{S}_i) \tag{3.47}$$

Nếu k và η là các hằng số dương thì $\dot{V}_i(\mathrm{t}) = -\mathrm{k}_i \mathrm{S}_i^2 - \eta_i \left|S_i\right| < 0$ nếu $S_i \neq 0$

Theo bổ đề Barbalat $\dot{V}_i(\mathrm{t}), \mathrm{S}_i$ sẽ có xu hướng tiến về không theo thời gian, hệ thống ổn định với bất kì hàm sai lệch ngõ ra nào.

Từ (3.45), (3.47) luật chuyển đổi mặt trượt được xác định:

$$u_{swi} = -\sum_{l=1}^{i-1} u_{swl} - \frac{\sum_{l=1}^{i}\left[\sum_{\substack{r=1 \\ r\neq l}}^{i}\left(\prod_{j=r}^{i} a_j\right) b_r\right] u_{eql} - \left(k_i S_i + \eta_i sgn S_i\right)}{\sum_{r=1}^{i}\left(\prod_{j=r}^{i} a_j\right) b_r} \qquad (3.48)$$

Như vậy từ (3.41), (3.48) luật điều khiển chế độ trượt phân cấp có được:

$$u_n = \frac{\sum_{r=1}^{n}\left(\prod_{j=r}^{n} a_j\right) b_r u_{eqr} - \left(k_n S_n + \eta_n sign S_n\right)}{\sum_{r=1}^{n}\left(\prod_{j=r}^{n} a_j\right) b_r} \qquad (3.49)$$

Chỉ luật điều khiển chuyển đổi ở lớp cuối cùng (n) là thực, nó đã bao gồm các lớp khác. Ở giai đoạn quá độ, bất kì trạng thái của hệ thống con nào lệch lệch khỏi mặt trượt của chính nó, luật điều khiển chuyển đổi của lớp cuối cùng sẽ lái nó quay trở lại mặt trượt. Dẫn đến khi ở giai đoạn trượt, trạng thái của hệ thống sẽ trượt trên mặt trượt của lớp cuối cùng. Trạng thái của hệ thống con sẽ được giữ để trượt trên mặt trượt của chính nó [7].

Chương 4. XÂY DỰNG THUẬT TOÁN ĐIỀU KHIỂN

Từ cơ sở mô hình toán xây dựng ở chương 2 và lý thuyết điều khiển được trình bày ở chương 3, thiết kế bộ điều khiển cho hệ thống với mục tiêu giữ cho ballbot cân bằng (góc nghiêng) và bám theo giá trị đặt (vị trí). Chương 4 trình bày phần xây dựng các bộ điều khiển cho ballbot bao gồm: LQR và điều khiển chế độ trượt.

Thông số mô hình:

Khối lượng của bóng	m_k	5 kg
Khối lượng của bánh xe đa hướng	m_w	0.35 kg
Khối lượng của thân robot	m_A	2.6 kg
Bán kính bóng	r_k	0.12 m
Bán kính bánh xe đa hướng	r_w	0.05 m
Bán kính thân robot	r_A	0.1 m
Khoảng cách từ tâm bóng đến trọng tâm robot	l	0.2 m
Gia tốc trọng trường	g	9.81 m/s^2
Góc nghiêng của bánh xe so với bóng	α	pi/4 radian
Tỉ số truyền của hộp số	i	33
Momen xoắn của động cơ	I_m	$3.33*10^{-6}$ kgm^2

Bảng 4.1: Thông số mô hình ballbot

4.1. Thiết kế bộ điều khiển LQR

4.1.1. Mô hình không gian trạng thái tuyến tính của robot

Vector trạng thái của hệ thống không xét đến thành phần tự quay quanh trục z :

$$x = \begin{bmatrix} \vartheta_x & \dot{\vartheta}_x & \vartheta_y & \dot{\vartheta}_x & \varphi_x & \dot{\varphi}_x & \varphi_y & \dot{\varphi}_y \end{bmatrix} \quad (4.1)$$

Khi đó ma trận trạng thái của hệ phương trình sau khi tuyến tính hóa:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ A_{x42} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & A_{y42} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ A_{x32} & A_{x34} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & A_{y32} & A_{y34} & 0 & 0 & 0 & 0 \end{bmatrix}; B = \begin{bmatrix} 0 & 0 \\ B_{x41} & 0 \\ 0 & 0 \\ 0 & B_{y41} \\ 0 & 0 \\ B_{x31} & 0 \\ 0 & 0 \\ 0 & B_{y31} \end{bmatrix} \quad (4.2)$$

C là ma trận đơn vị 8x8 và D = 0

Giá trị của $A_{x32}, A_{x34}, A_{x42}, B_{x31}, B_{x41}$ và tương tự cho $A_{y32}, A_{y34}, A_{y42}, B_{y31}, B_{y41}$ được tính toán với thông số thực tế của mô hình lần lượt là :

$$\begin{cases} A_{x32} = A_{y32} = 2.5508 \\ A_{x34} = A_{y34} = 0 \\ A_{x42} = A_{y42} = 21.9394 \end{cases}; \begin{cases} B_{x31} = B_{y31} = 45.2048 \\ B_{x41} = B_{y41} = -20.4078 \end{cases}$$

Ngõ ra hệ thống là vector trạng thái x, đầu vào là hệ thống thực tế là momen xoắn T_1, T_2, T_3. Trong đó T thực tế sẽ được chuyển đổi từ T_x, T_y trong tính toán theo công thức:

$$\begin{cases} T_1 = \frac{1}{3}\left(T_z + \frac{2}{\cos(\alpha)}\left(T_x \cos(\beta) - \mathrm{T}_y \sin(\beta)\right)\right) \\ T_2 = \frac{1}{3}\left(T_z + \frac{1}{\cos(\alpha)}\left(\sin(\beta)(-\sqrt{3}\,\mathrm{T}_x + \mathrm{T}_y) - \cos(\beta)(\mathrm{T}_x + \sqrt{3}\,\mathrm{T}_y)\right)\right) \\ T_3 = \frac{1}{3}\left(T_z + \frac{1}{\cos(\alpha)}\left(\sin(\beta)(\sqrt{3}\,\mathrm{T}_x + \mathrm{T}_y) + \cos(\beta)(-\mathrm{T}_x + \sqrt{3}\,\mathrm{T}_y)\right)\right) \end{cases} \quad (4.3)$$

Với β là góc lệch của T_1 so với T_x (radian)

Tính hiệu điều khiển u là điện áp cấp cho ba động cơ.

4.1.2. Tính toán độ lợi điều khiển

Đối tượng tuyến tính rời rạc được mô tả bởi phương trình trạng thái :

$$x(\mathrm{k}+1) = \mathrm{A}_d\, x(\mathrm{k}) + \mathrm{B}_d\, u(\mathrm{k}) \quad (4.4)$$

Trong đó : $x(\mathrm{k}) = [\mathrm{x}_1(\mathrm{k}), \mathrm{x}_2(\mathrm{k}), ..., \mathrm{x}_n(\mathrm{k})]^T$ là vector trạng thái

$\mathrm{u}(\mathrm{k}) = [\mathrm{u}_1(\mathrm{k}), \mathrm{u}_2(\mathrm{k}), ..., \mathrm{u}_m(\mathrm{k})]^T$ là vector tính hiệu điều khiển

Rời rạc hóa mô hình với thời gian lấy mẫu 5ms ta được ma trận trạng thái rời rạc :

$$A_d = \begin{bmatrix} 1 & 0.005 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.1097 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.005 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1097 & 1 & 0 & 0 & 0 & 0 \\ 3.189.10^{-5} & 5.314.10^{-8} & 0 & 0 & 1 & 0.005 & 0 & 0 \\ 0.01276 & 3.189.10^{-5} & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 3.189.10^{-5} & 5.314.10^{-8} & 0 & 0 & 1 & 0.005 \\ 0 & 0 & 0.01276 & 3.189.10^{-5} & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$B_d = \begin{bmatrix} -0.0002551 & 0 \\ -0.102 & 0 \\ 0 & -0.0002551 \\ 0 & -0.102 \\ 0.0005651 & 0 \\ 0.226 & 0 \\ 0 & 0.0005651 \\ 0 & 0.226 \end{bmatrix}$$

Bài toán đặt ra là tìm tín hiệu điều khiển u(k) điều chỉnh hệ thống từ trạng thái ban đầu x(0) = x_0 bất kì về trạng thái cuối x(N) = 0 sao cho tối thiểu hàm chỉ tiêu chất lượng dạng toàn phương :

$$J(\mathrm{u}) = \frac{1}{2}\sum_{k=0}^{N-1}\left[x^T(\mathrm{k}).\mathrm{Q}.\mathrm{x}(\mathrm{k}) + \mathrm{u}^T(\mathrm{k}).\mathrm{R}.\mathrm{u}(\mathrm{k}) \right] \qquad (4.5)$$

Ma trận Q và R được chọn là:

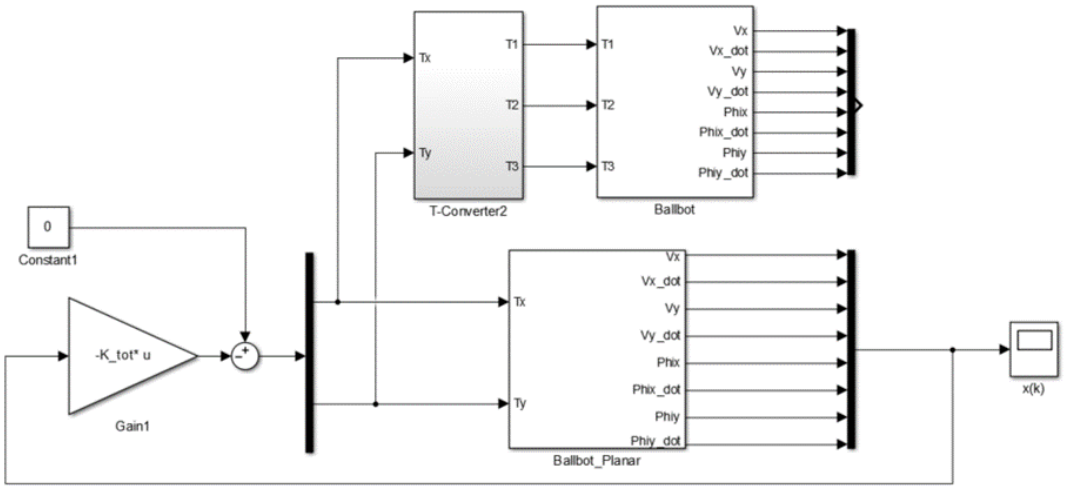
$$Q = \begin{bmatrix} 5000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 \end{bmatrix}$$

$$R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$$

Độ lợi K tính được:

$$K = \begin{bmatrix} -41.5370 & -8.334 & 0 & 0 & -2.9813 & -2.3239 & 0 & 0 \\ 0 & 0 & -39.8666 & -7.9533 & 0 & 0 & -2.6166 & -2.1572 \end{bmatrix}$$

4.1.3. *Sơ đồ mô phỏng*

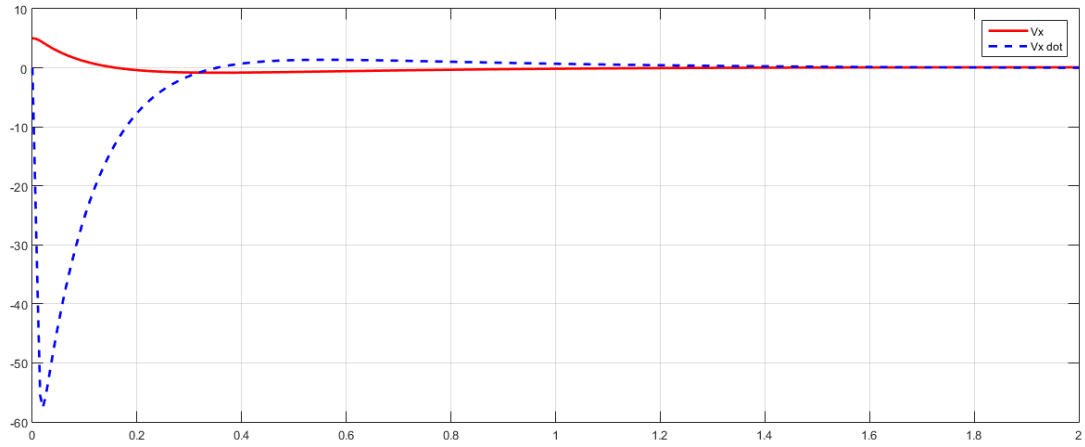


Hình 4.1: Sơ đồ điều khiển LQR

Trong đó khối T-Converter được dùng để chuyển đổi tín hiệu điều khiển tham khảo trong hai mặt phẳng x, y thành ba tín hiệu điều khiển thực xuất cho ba động cơ.

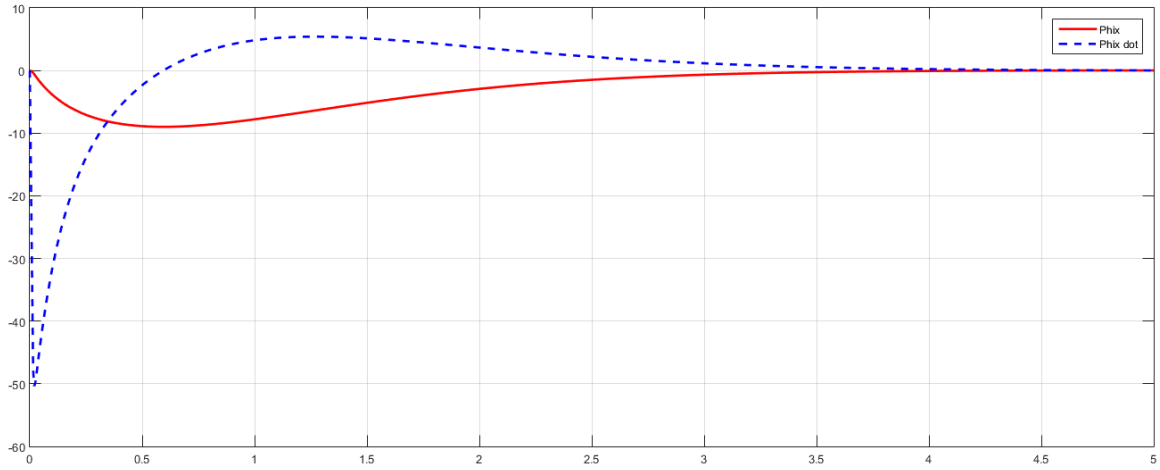
4.1.4. *Kết quả mô phỏng*

Kết quả mô phỏng góc nghiêng của thân robot và vận tốc với góc đặt ban đầu là 5^0 trong mặt phẳng Oyz trong hai giây đầu tiên:



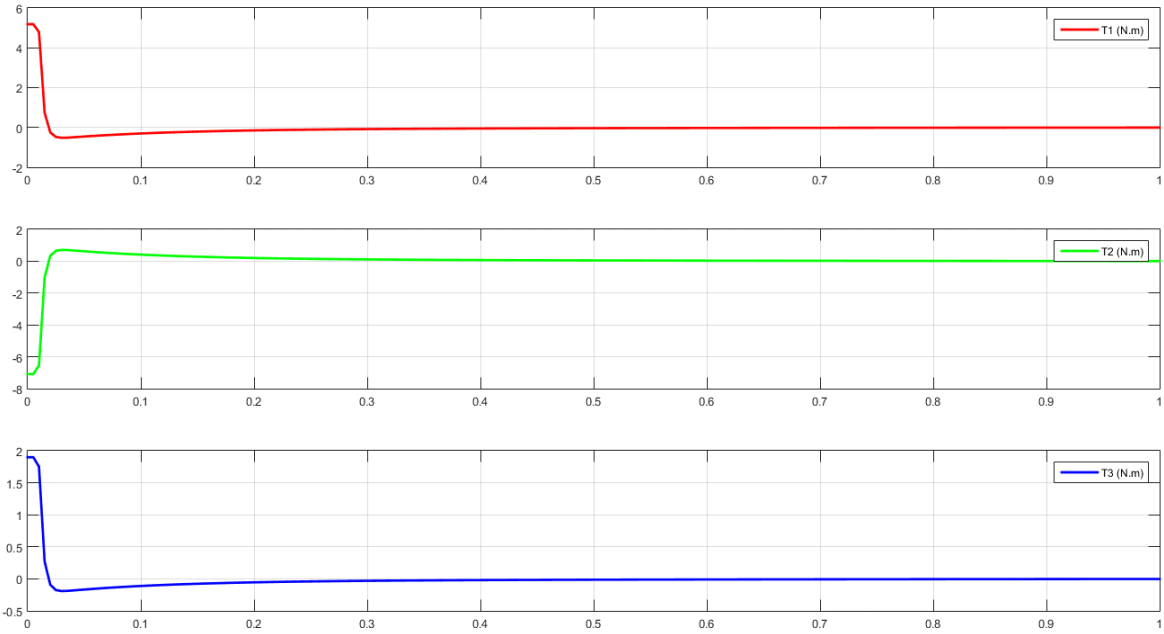
Hình 4.2: Góc nghiêng và vận tốc thân robot với bộ điều khiển LQR

Góc xoay của bóng và vận tốc trong mặt phẳng Oyz trong 5 giây đầu:



Hình 4.3: Góc lệch và vận tốc của bóng với bộ điều khiển LQR

Tín hiệu điều khiển cho ba động cơ trong giây đầu tiên:



Hình 4.4: Tín hiệu điều khiển cho ba động cơ với bộ điều khiển LQR

4.2. Thiết kế bộ điều khiển trượt

4.2.1. Điều khiển trượt cơ bản

Với bộ điều khiển trượt cơ bản ta chỉ có thể kiểm soát được góc nghiêng của robot. Điều này chỉ tồn tại khi mô phỏng, trong thực tế nếu không kiểm soát được vị trí, hay robot luôn chuyển động theo một hướng thì sẽ không giữ được thăng bằng.

4.2.1.1. Phương trình mô tả hệ thống

Phương trình mô tả hệ thống được viết lại với vector trạng thái $x = \begin{bmatrix} \varphi_x & \dot{\varphi}_x & \vartheta_x & \dot{\vartheta}_x \end{bmatrix}^T$ và ngõ vào u = Tx:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \dfrac{2.3051\sin(x_3)\dot{x}_3^2 + 85.01545u + 36.2312u\cos(x_3) - 61.609\cos(x_3)\sin(x_3) + 40.0377\sin(x_3)}{\cos(x_3) - 0.7693\cos^2(x_3) + 2.3967} \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \dfrac{0.5\sin(x_3)\dot{x}_3^2 - 19.2152u + 72.7425\sin(x_3) - 0.7693\cos(x_3)\sin(x_3)\dot{x}_3^2 - 36.2312\cos(x_3)u}{\cos(x_3) - 0.7693\cos^2(x_3) + 2.3967} \end{cases} \tag{4.6}$$

Đặt

$$\begin{cases} f_1 = x_2 \\ f_2 = \dfrac{2.3051\sin(x_3)x_4^2 - 61.609\cos(x_3)\sin(x_3) + 40.0377\sin(x_3)}{\cos(x_3) - 0.7693\cos^2(x_3) + 2.3967} \\ g_2 = \dfrac{85.01545 + 36.2312\cos(x_3)}{\cos(x_3) - 0.7693\cos^2(x_3) + 2.3967} \\ f_3 = x_4 \\ f_4 = \dfrac{\left[0.5\sin(x_3) - 0.7693\cos(x_3)\sin(x_3)\right]x_4^2 + 72.7425\sin(x_3)}{\cos(x_3) - 0.7693\cos^2(x_3) + 2.3967} \\ g_4 = \dfrac{-19.2152 - 36.2312\cos(x_3)}{\cos(x_3) - 0.7693\cos^2(x_3) + 2.3967} \end{cases} \tag{4.7}$$

Ta có phương trình: $\begin{cases} \dot{x}_1 = f_1(\mathrm{x}_2) \\ \dot{x}_2 = f_2(\mathrm{x}_3, \mathrm{x}_4) + g_2(\mathrm{x}_3)\,\mathrm{u} \\ \dot{x}_3 = f_3(\mathrm{x}_3) \\ \dot{x}_4 = f_4(\mathrm{x}_3, \mathrm{x}_4) + g_4(\mathrm{x}_3)\,\mathrm{u} \end{cases}$ (4.8)

4.2.1.2. Thiết kế luật điều khiển

Mặt trượt $S = x_4 - \varphi(\mathrm{x}_3)$, trên mặt trượt $\mathrm{S} = 0 \Rightarrow x_4 = \varphi(\mathrm{x}_3)$

Khi đó (4.8) được viết lại: $\dot{\mathrm{x}}_3 = \varphi(\mathrm{x}_3)$ (4.9)

Chọn φ sao cho (4.9) có 0 là điểm cân bằng ổn định tiệm cận [5]:

$$\varphi(\mathrm{x}_3) = -0.5\,\mathrm{x}_3 \Rightarrow \dot{\mathrm{x}}_3 = -0.5\,\mathrm{x}_3 = \mathrm{f}(\mathrm{x}_3) \quad (4.10)$$

Ta có tại $\mathrm{x}_3 = 0$: $\begin{cases} f(\mathrm{x}_3) = 0 \\ \dot{f}(\mathrm{x}_3) = -0.5 < 0 \end{cases}$

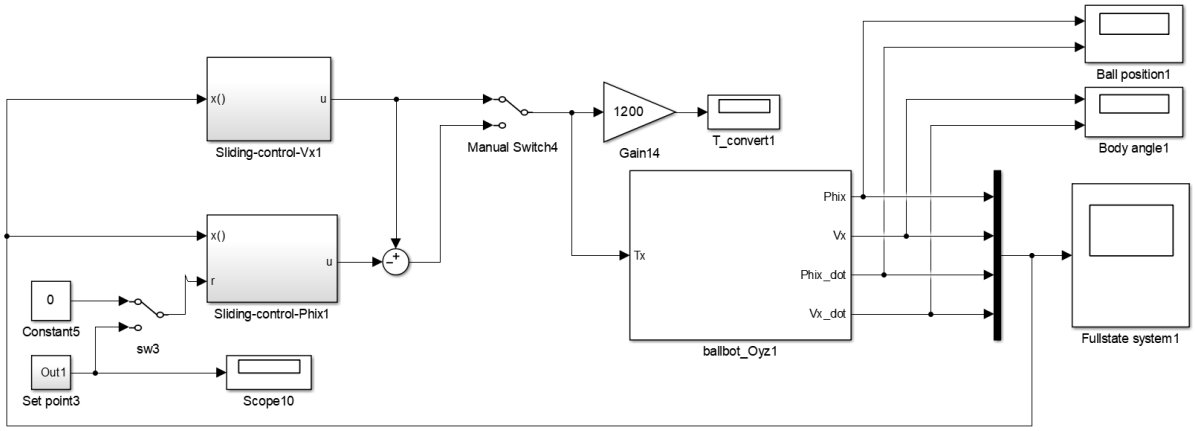
Mặt trượt $S = x_4 + 0.5\,\mathrm{x}_3$ (4.11)

$$\dot{\mathrm{S}} = \dot{\mathrm{x}}_4 + \dot{x}_3 = f_4(\mathrm{x}_3, \mathrm{x}_4) + \mathrm{g}_4(\mathrm{x}_3)\,\mathrm{u} + \mathrm{x}_4 = -\mathrm{Ksign(S)} \quad (4.12)$$

$$\Rightarrow u = \mathrm{g}_4^{-1}(\mathrm{x}_3)\left[-Ksign(\mathrm{S}) - \mathrm{x}_4 - \mathrm{f}_4(\mathrm{x}_3, \mathrm{x}_4)\right] \quad (4.13)$$

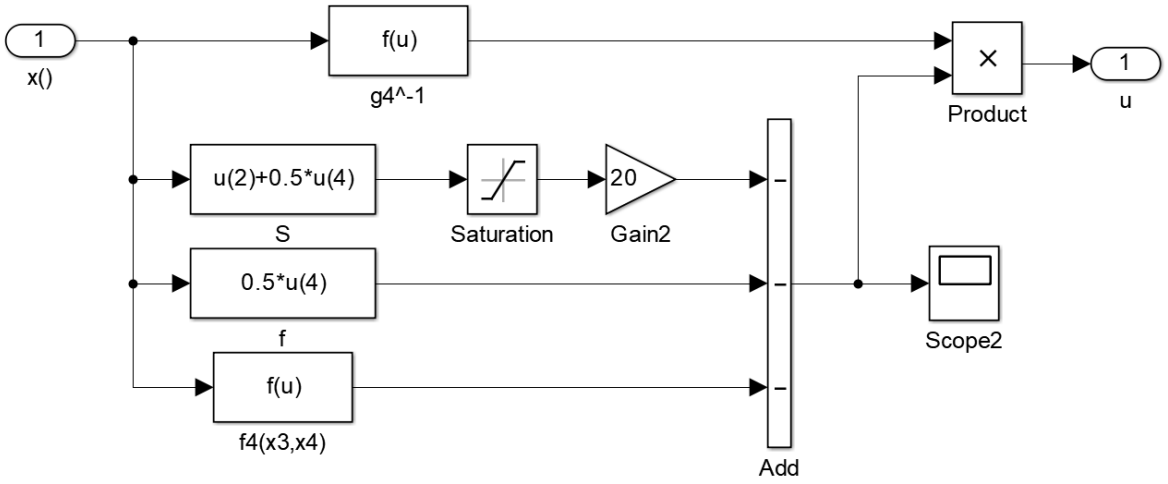
4.2.1.3. Mô phỏng hệ thống

Sơ đồ mô phỏng hệ thống điều khiển với một mặt trượt xét mặt Oyz:



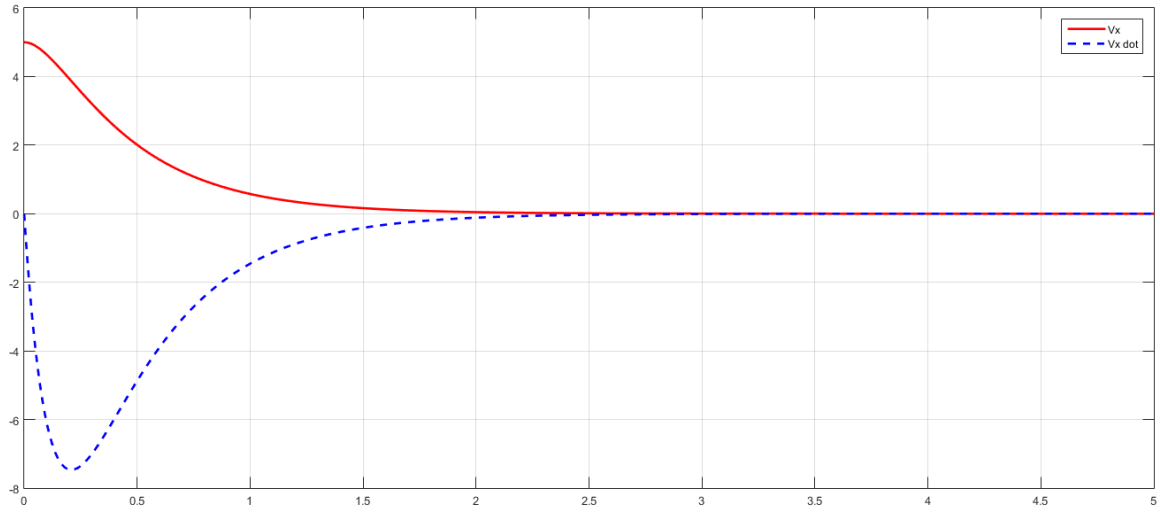
Hình 4.5: Sơ đồ điều khiển hệ thống với một mặt trượt

Khối sliding control:



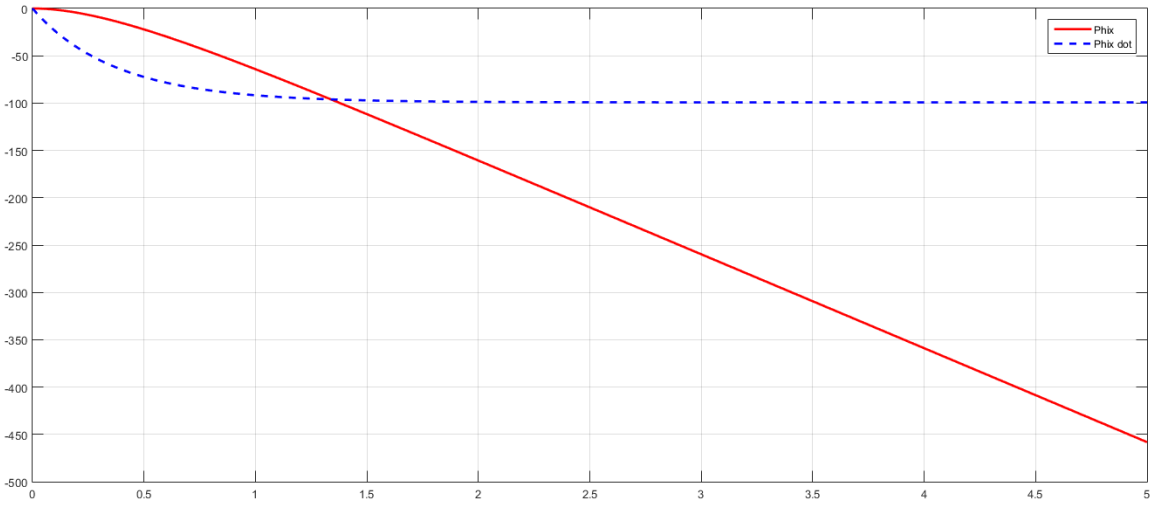
Hình 4.6: Luật điều khiển với một mặt trượt

Kết quả mô phỏng góc thân robot và vận tốc trong 5 giây đầu tiên, góc đặt 5^0:



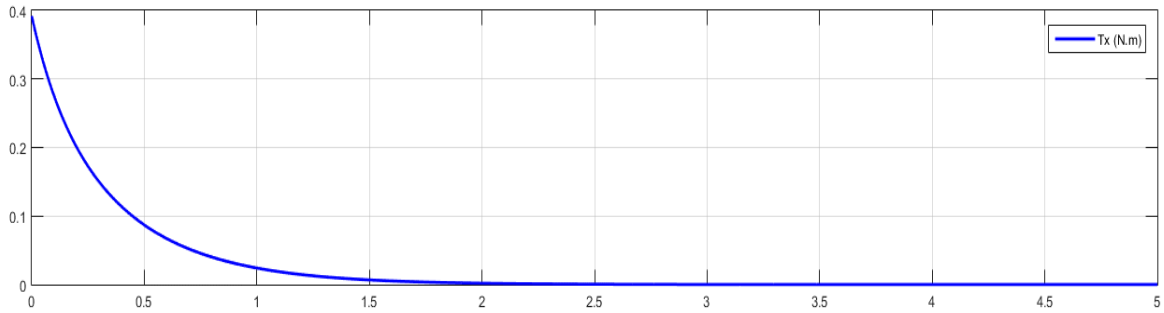
Hình 4.7: Góc nghiêng và vận tốc thân robot khi điều khiển với một mặt trượt

Kết quả mô phỏng góc xoay của bóng trong 5 giây đầu tiên:



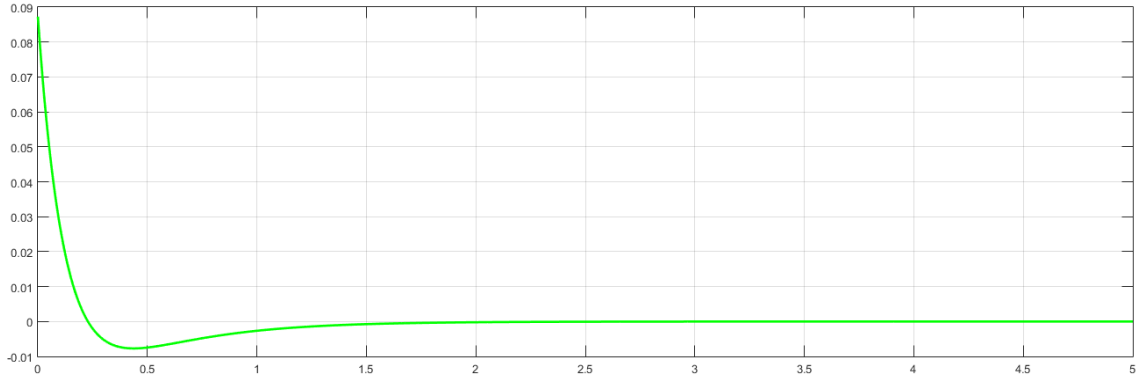
Hình 4.8: Góc xoay và vận tốc của bóng khi điều khiển với một mặt trượt

Tín hiệu điều khiển:



Hình 4.9: Tín hiệu điều khiển khi điều khiển với một mặt trượt

Mặt trượt S:



Hình 4.10: Mặt trượt S

Ta thấy mặc dù tín hiệu điều khiển có hội tụ về không khi thời gian $t \to \infty$ và hàm định nghĩa sai lệch của trạng thái hệ thống ϑ (mặt trượt S) hội tụ về không, tuy nhiên vị trí của robot vẫn chưa được kiểm soát $\phi \to \infty$ khi $t \to \infty$.

Tương tự như khi thiết kế mặt trượt để ổn định hai trạng thái vị trí của robot, đáp ứng của hệ thống sẽ cho kết quả không tốt khi ta thả trạng thái góc nghiêng. Lúc này đòi hỏi một cơ chế chuyển đổi để đảm bảo tín hiệu điều khiển có thể ổn định hóa cùng lúc hai mặt trượt.

4.2.2. Điều khiển chế độ trượt phân cấp cho mô hình ballbot

4.2.2.1. Phương trình mô tả hệ thống

Xét phương trình trạng thái của hệ thống đã được thể hiện ở 4.2.1.1

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f_2(x_3, x_4) + g_2(x_3)u \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = f_4(x_3, x_4) + g_4(x_3)u \end{cases} \text{ Vector trạng thái } x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \text{ với } \begin{cases} x_1 = \phi_x \\ x_2 = \dot{\phi}_x \\ x_3 = \vartheta_x \\ x_4 = \dot{\vartheta}_x \end{cases} \quad (4.14)$$

Mặt trượt cần thiết kế để kiểm soát sai lệch $e_1 = \varphi_x - \varphi_d$ và $e_2 = \vartheta_x$ (4.15)

φ_d thể hiện vị trí đặt cần đạt được.

4.2.2.2. Thiết kế luật điều khiển

Mặt trượt cho hai hệ thống con được thiết kế:

$$\begin{cases} s_1 = c_1(\varphi_x - \varphi_d) + (\dot{\varphi}_x - \dot{\varphi}_d) \\ s_2 = c_2\vartheta_x + \dot{\vartheta}_x \end{cases} \quad (4.16)$$

Luật điều khiển tương đương của mỗi hệ con:

$$\begin{cases} u_{eq1} = [\ddot{\varphi}_x - c_1(\dot{\varphi}_x - \dot{\varphi}_d) - f_2]g_2^{-1} \\ u_{eq2} = [-c_2\dot{\vartheta}_x - f_4]g_4^{-1} \end{cases} \quad (4.17)$$

Định nghĩa mặt trượt ở lớp đầu tiên: $S_1 = s_1$

Luật điều khiển trượt và hàm Lyapunov được xác định:

$$\begin{cases} u_1 = u_{eq1} + u_{sw1} \\ V_1 = 0.5S_1^2 \end{cases} \quad (4.18)$$

Đạo hàm V_1 theo thời gian ta có:

$$\dot{V}_1 = S_1 \dot{S}_1 \tag{4.19}$$

với $\dot{S}_1 = -k_1 S_1 - \eta_1 \operatorname{sgn}(S_1)$ (4.20)

k_1, η_1 là hằng số dương cần thiết kế.

Luật SMC cho lớp đầu tiên xác định: $u_1 = u_{eq1} + \dot{S}_1 / g2$ (4.19)

Mặt trượt lớp thứ hai được thiết kế: $S_2 = \alpha S_1 + s_2$, α là hằng số cần thiết kế.

Luật điều khiển trượt và hàm Lyapunov cho lớp thứ hai:

$$\begin{cases} u_2 = u_1 + u_{eq2} + u_{sw2} \\ V_2 = 0.5 S_2^2 \end{cases} \tag{4.20}$$

Đạo hàm V_2 theo thời gian:

$$\dot{V}_2 = S_2 \dot{S}_2 \tag{4.21}$$

với $\dot{S}_2 = -k_2 S_2 - \eta_2 \operatorname{sgn}(S_2)$ (4.22)

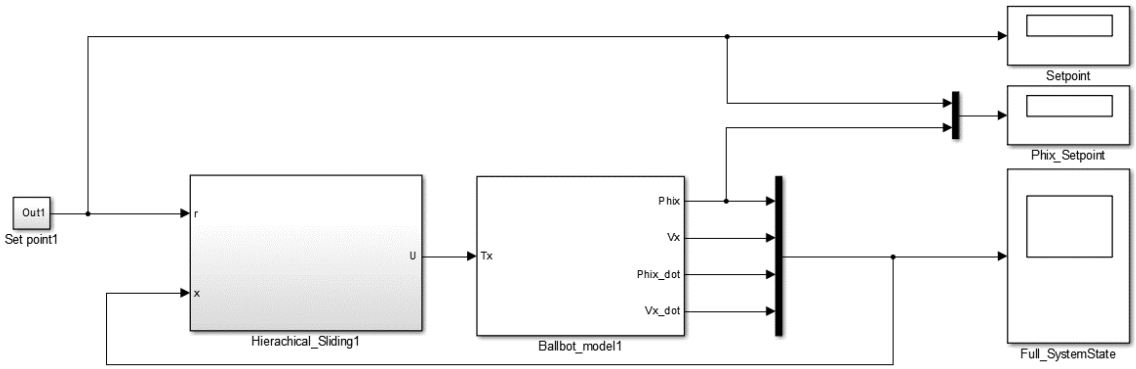
k_2, η_2 là hằng số dương cần thiết kế.

Luật SMC được xác định bằng:

$$u = u_2 = \frac{\alpha g_2 u_{eq1} + g_4 u_{eq2} - (k_2 S_2 + \eta_2 \operatorname{sgn}(S_2))}{\alpha g_2 + g_4} \tag{4.23}$$

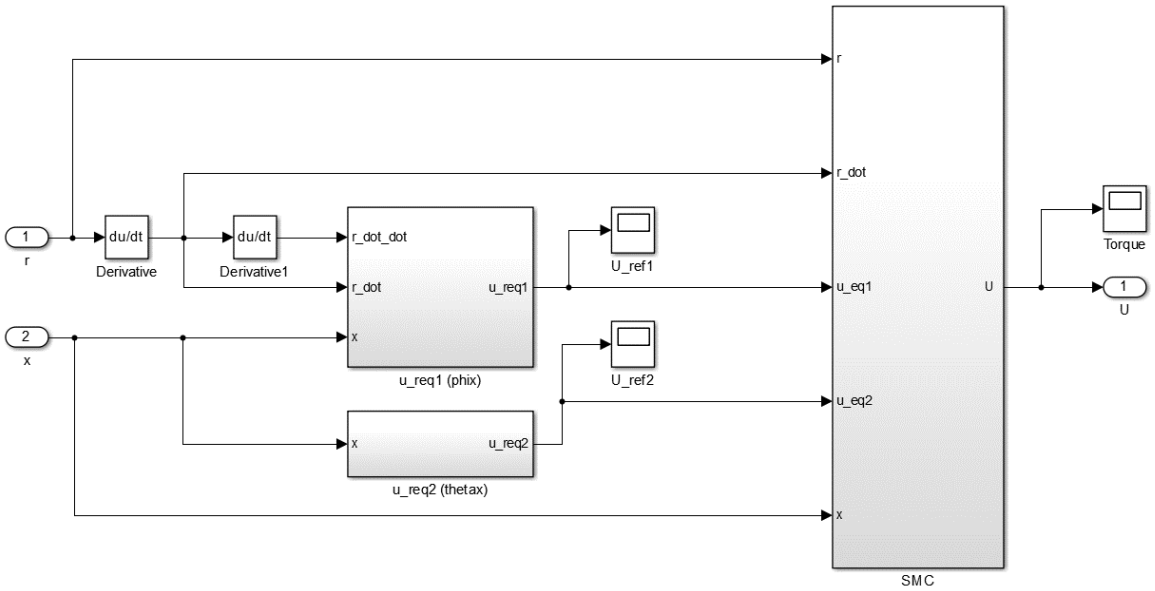
4.2.2.3. Mô phỏng hệ thống

Mô phỏng hệ thống điều khiển trượt phân cấp cho ballbot trên mặt phẳng Oyz, mặt Oxz được xây dựng tương tự:



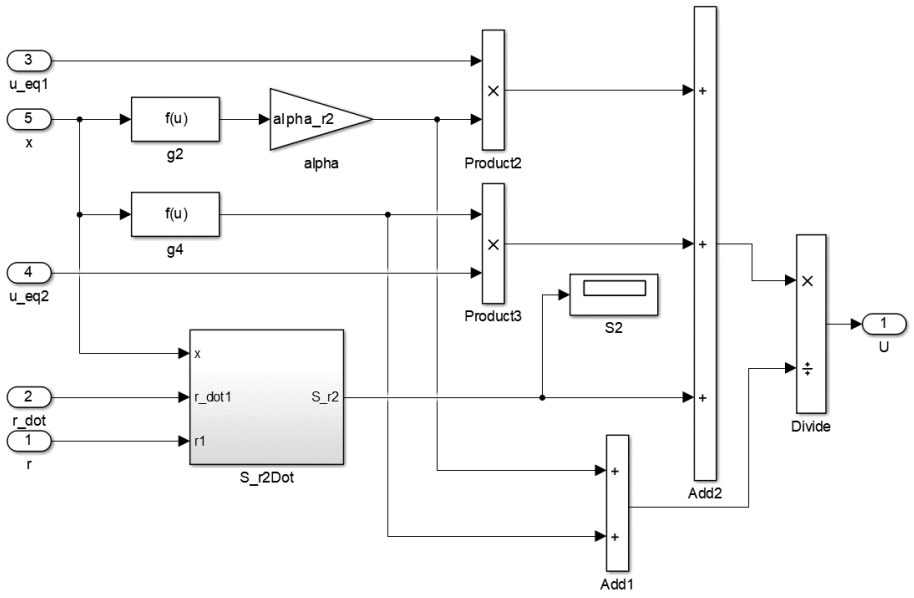
Hình 4.11: Mô hình điều khiển chế độ trượt phân cấp trực tiếp

Khối điều khiển trượt phân cấp:



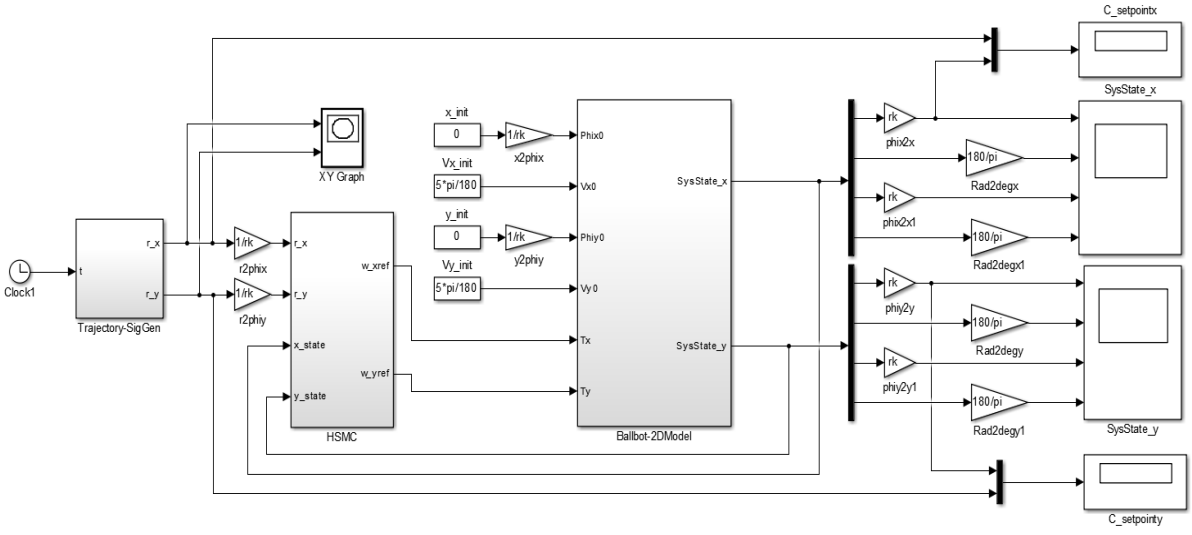
Hình 4.12: Khối điều khiển trượt phân cấp

Cơ chế chuyển đổi mặt trượt:



Hình 4.13: Cơ chế chuyển đổi mặt trượt

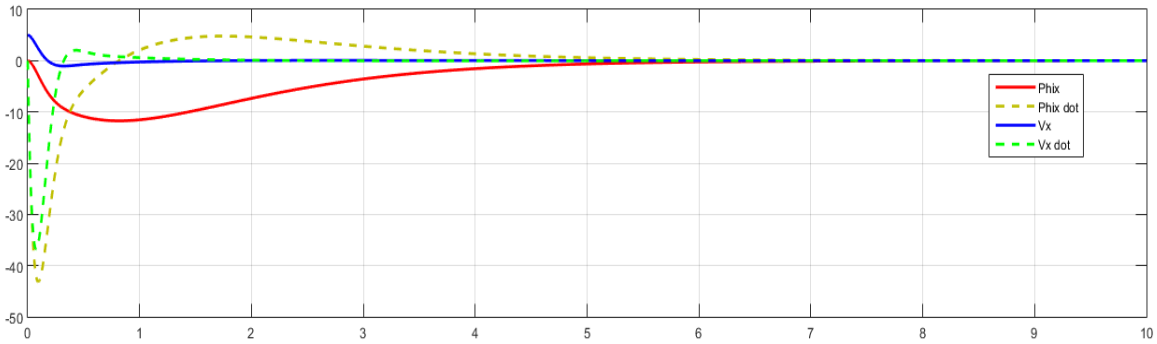
Mô hình hệ thống với bộ điều khiển chế độ trượt phân cấp bám theo quỹ đạo:



Hình 4.14: Mô hình hệ thống điều khiển chế độ trượt bám theo quỹ đạo

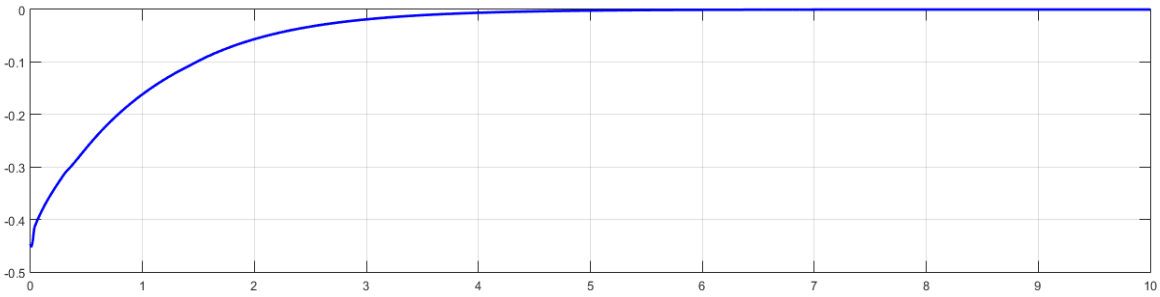
4.2.2.4. Kết quả mô phỏng

Trường hợp: Điều khiển trượt ổn định hóa hệ thống với góc đặt ban đầu 5^0 mô phỏng trong 10 giây đầu tiên:



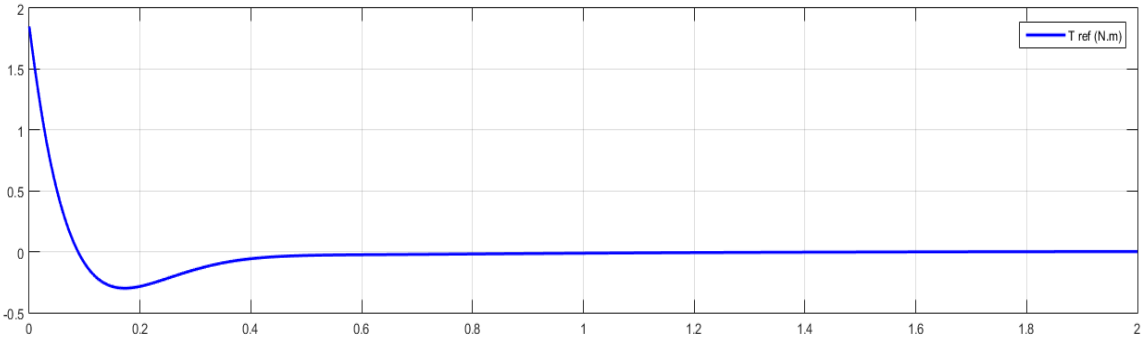
Hình 4.15: Trạng thái của hệ thống điều khiển chế độ trượt phân cấp

Mặt trượt lớp cuối cùng S2:



Hình 4.16: Mặt trượt lớp cuối S2 chế độ ổn định hóa

Tín hiệu điều khiển u ở chế độ trượt ổn định hóa ở 2 giây đầu tiên :

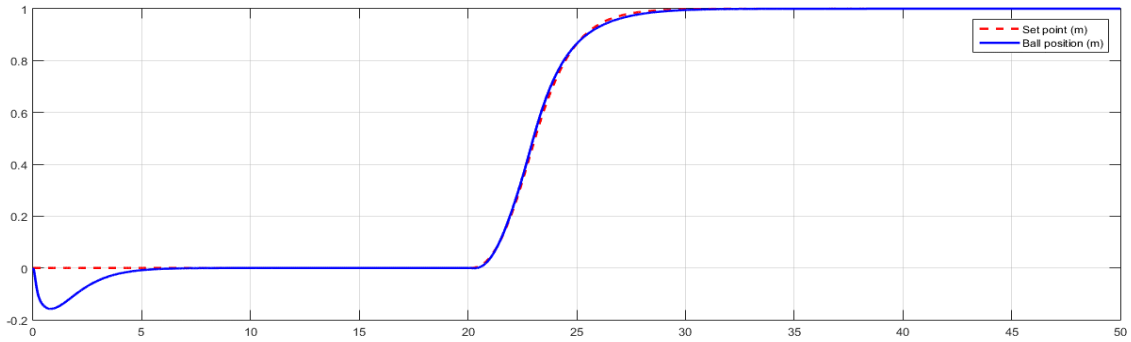


Hình 4.17: Tín hiệu điều khiển chế độ trượt phân cấp ổn định hóa

Ta thấy, ngõ ra điều khiển u đã ổn định được tất cả các trạng thái của hệ thống và có xu hướng tiến về không khi $t \to \infty$. Trạng thái của hệ thống đã trượt trên mặt phẳng của riêng của nó để khi quan sát mặt trượt ở lớp cuối $S2 \to 0$ khi $t \to \infty$. Ngõ ra bộ điều khiển u cũng đồng thời tiến về không khi $t \to \infty$.

Trường hợp: Điều khiển trượt bám (điều khiển vị trí của bóng). Tại giây thứ 20 đặt một tính hiệu là hàm dốc và quan sát đáp ứng của hệ thống.

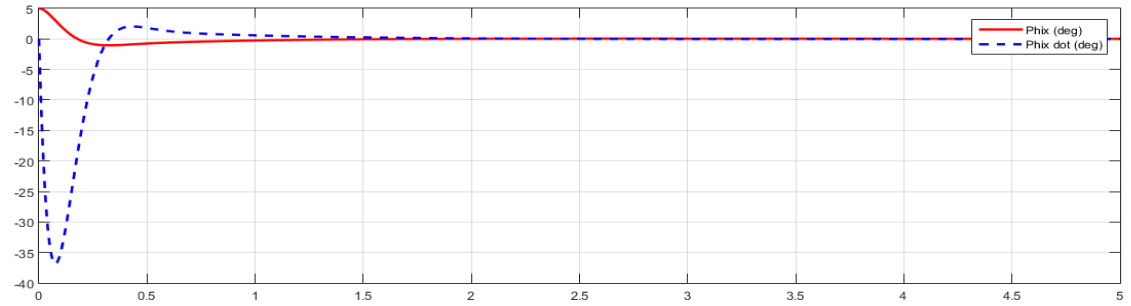
Tín hiệu đặt và đáp ứng góc xoay của bóng:



Hình 4.18: Đáp ứng vị trí của hệ thống điều khiển chế độ trượt phân cấp

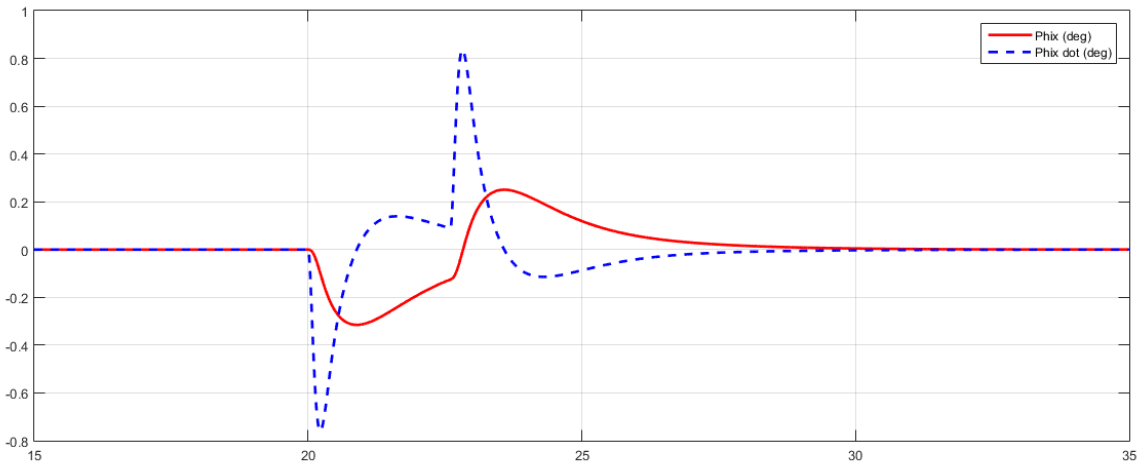
Giai đoạn đầu tiên hệ thống cần ổn định hóa với góc đặt thân robot ban đầu là 5^0 do đó mà đáp ứng vị trí tại những giây đầu tiên có sai lệch. Tại giây thứ 20 với tín hiệu đặt là hàm dốc đáp ứng vị trí của robot bám tốt theo tín hiệu đặt. Hàm dốc được chọn do hệ có quán tính lớn.

Đáp ứng góc nghiêng của hệ thống thời điểm ổn định hệ thống:



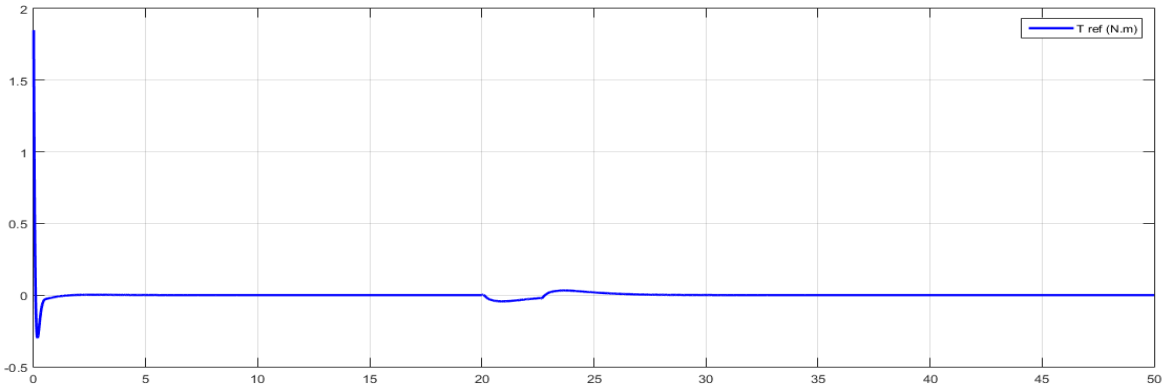
Hình 4.19: Đáp ứng góc nghiêng của hệ thống tại thời điểm ổn định hóa

Đáp ứng góc nghiêng của hệ thống thời điểm có tín hiệu đặt (giây 15 -> 35)



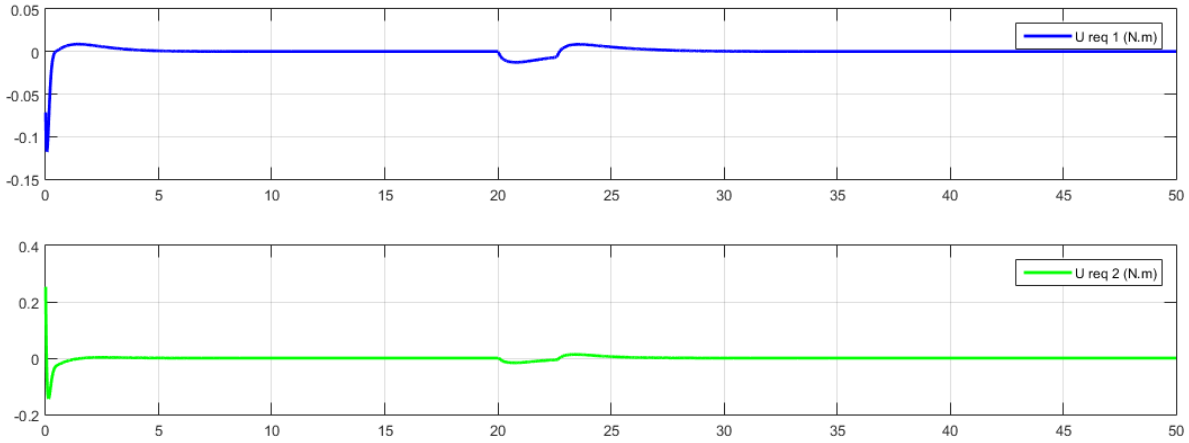
Hình 4.20: Đáp ứng góc nghiêng của hệ thống khi có tín hiệu đặt

Ngõ vào điều khiển u :



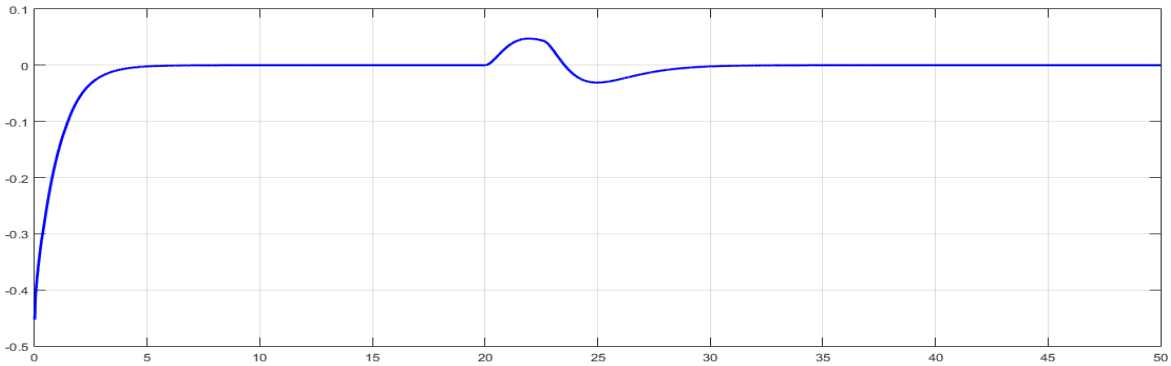
Hình 4.21: Tín hiệu điều khiển chế độ trượt phân cấp bám

Tín hiệu điều khiển của từng mặt trượt:



Hình 4.22: Tín hiệu điều khiển của từng mặt trượt

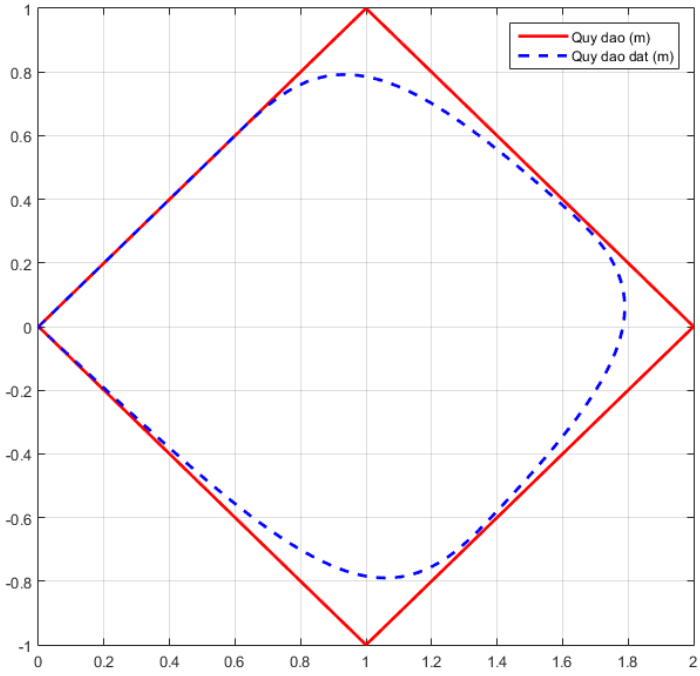
Mặt trượt lớp cuối của hệ thống:



Hình 4.23: Mặt trượt lớp cuối S2 chế trượt bám

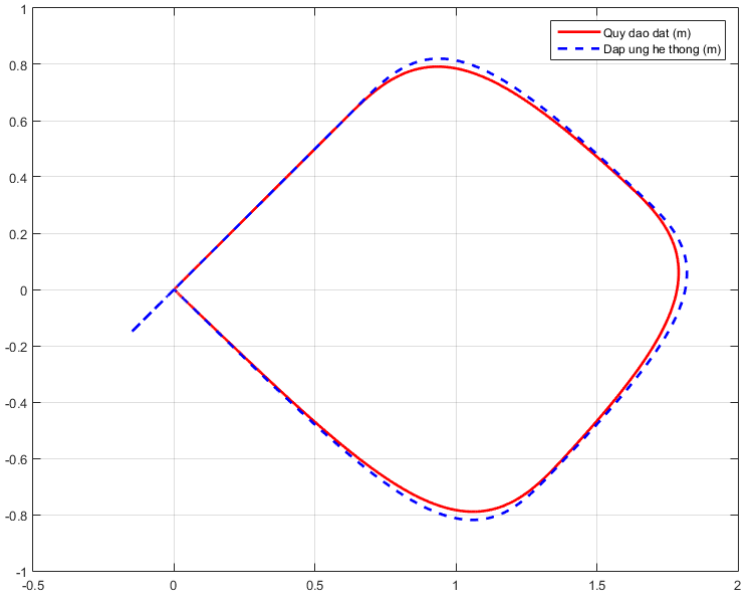
Ta thấy, ngõ ra điều khiển u đã ổn định được tất cả các trạng thái của hệ thống. Kể cả khi có tín hiệu đặt vị trí, góc nghiêng của thân robot vẫn được ổn định cân bằng. Mặt trượt ở lớp cuối $S2 \to 0$ khi $t \to \infty$. Ngõ ra bộ điều khiển u và các tín hiệu điều khiển của hệ con cũng được ổn định, tiến về không khi $t \to \infty$.

Trường hợp: Tín hiệu đặt cho hai mặt phẳng Ozx và Ozy được quy hoạch thành quỹ đạo. Với quỹ đạo hình vuông, sau khi được làm trễ:



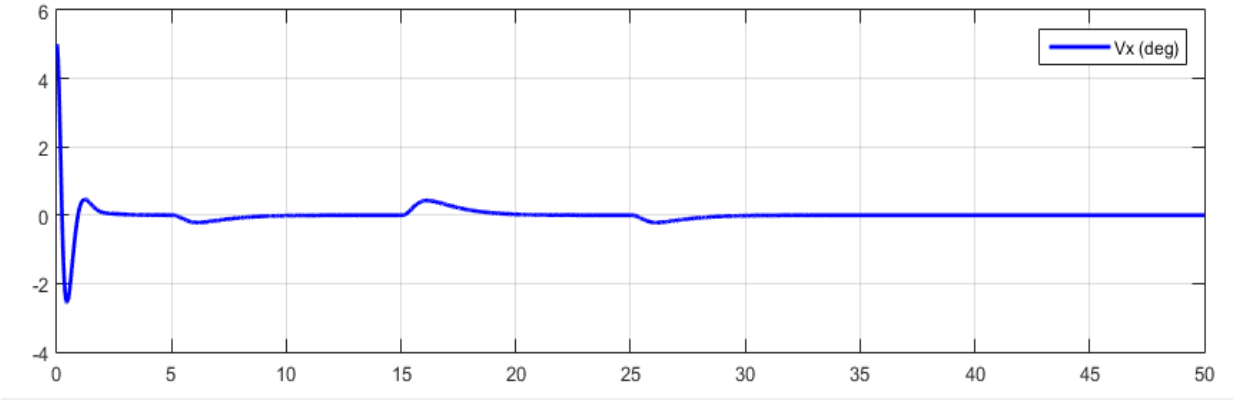
Hình 4.24: Quỹ đạo quy hoạch và quỹ đạo đặt hình vuông

Đáp ứng của hệ thống với quỹ đạo đặt hình vuông đã được làm trễ :

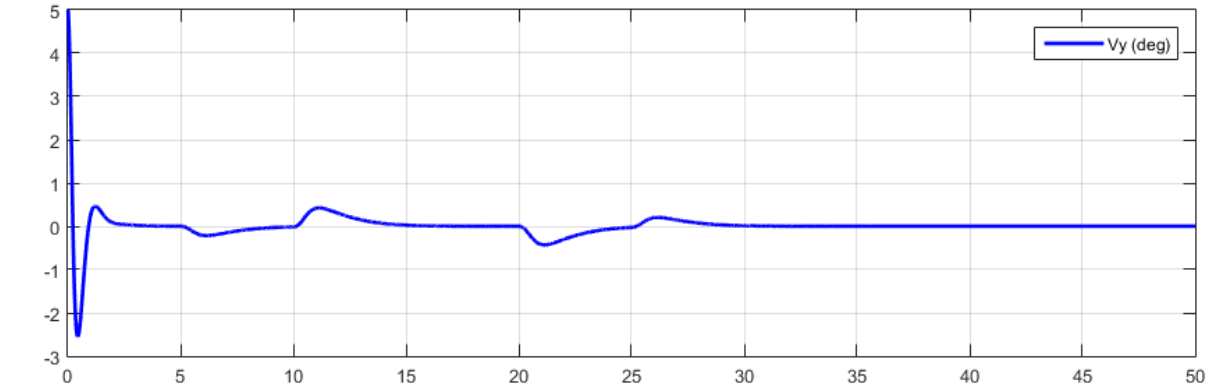


Hình 4.25: Đáp ứng của hệ thống với quỹ đạo đặt hình vuông

Đáp ứng góc nghiêng của hệ thống:



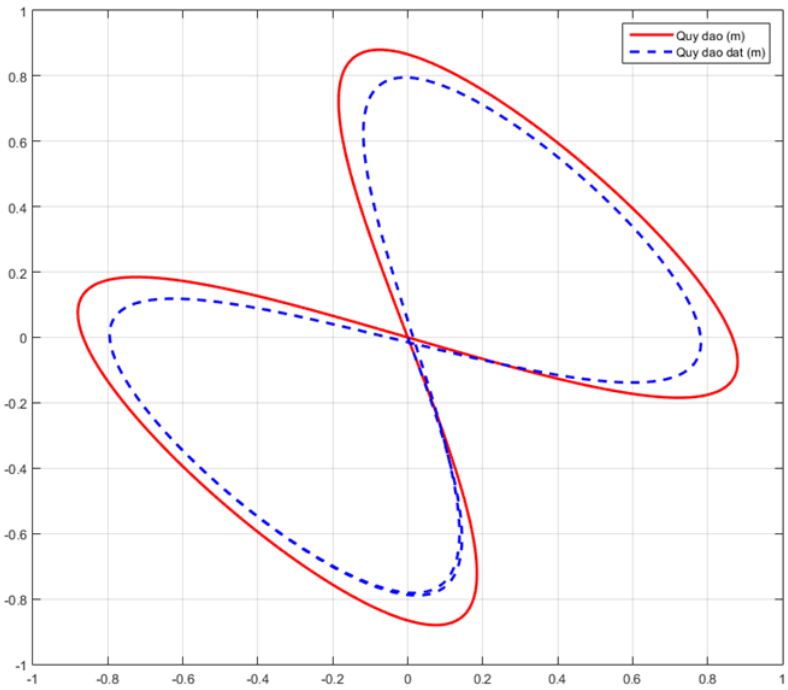
Hình 4.26: Đáp ứng góc nghiêng theo trục x với quỹ đạo đặt hình vuông



Hình 4.27: Đáp ứng góc nghiêng theo trục y với quỹ đạo đặt hình vuông

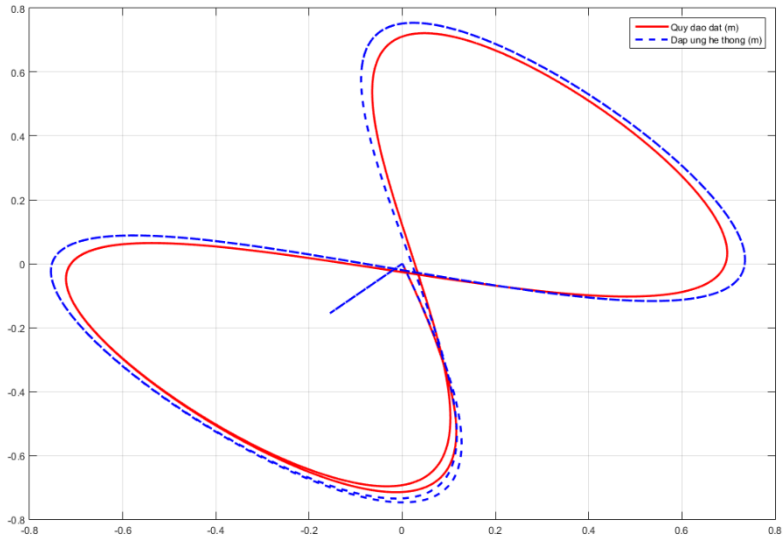
Ở 5 giây đầu tiên, tín hiệu đặt bằng không để robot ổn định với góc nghiêng ban đầu 5^0 theo các trục. Thời gian tiếp theo, với quỹ đạo đặt là hình vuông sau khi đã được làm trễ, đáp ứng góc nghiêng của hệ trong quá trình robot trong quá trình di chuyển là nhỏ hơn $\pm 2^0$, quỹ đạo của ballbot bám tốt theo quỹ đạo đặt.

Trường hợp: Quỹ đạo đặt là đường số 8 sau khi được làm trễ:



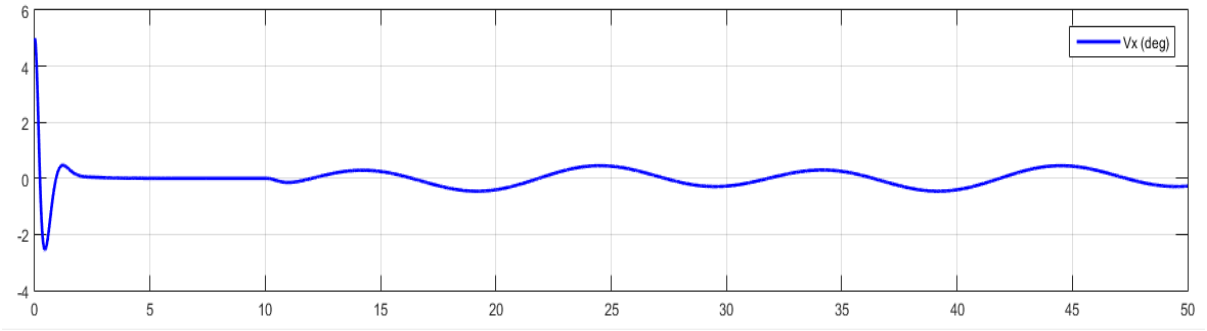
Hình 4.28: Quỹ đạo quy hoạch và quỹ đạo đặt hình số 8

Đáp ứng của hệ thống với quỹ đạo đặt hình số 8 đã được làm trễ :

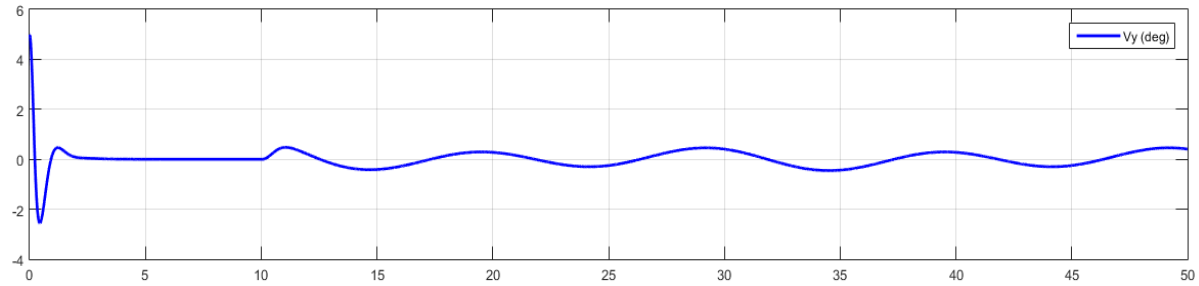


Hình 4.29: Đáp ứng của hệ thống với quỹ đạo đặt hình số 8

Đáp ứng góc nghiêng của hệ thống:



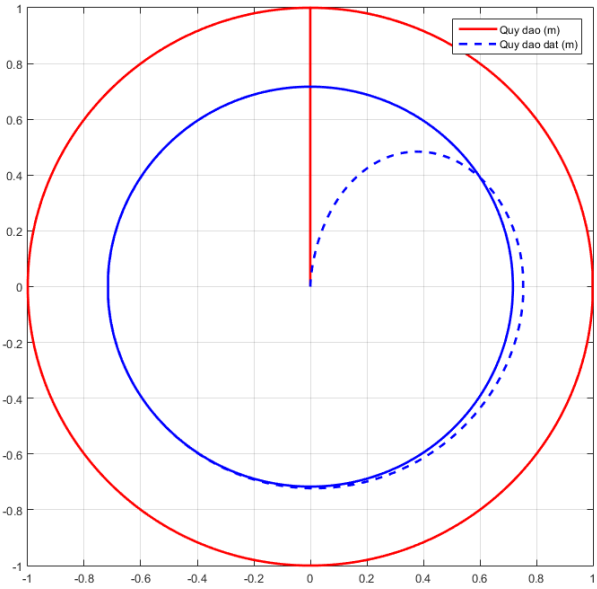
Hình 4.30: Đáp ứng góc nghiêng theo trục x với quỹ đạo đặt hình số 8



Hình 4.31: Đáp ứng góc nghiêng theo trục y với quỹ đạo đặt hình số 8

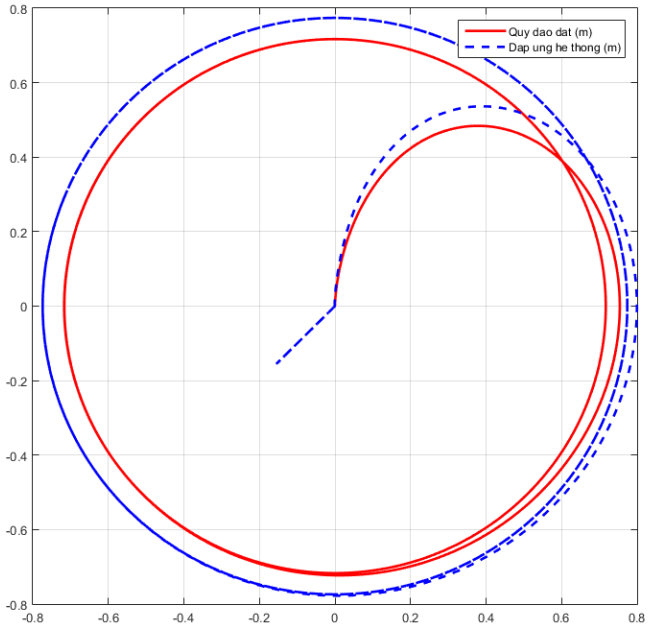
Với quá trình đặt quỹ đạo tương tự như bên trên ở giây thứ 10^0 hệ thống bắt đầu bám theo giá trị đặt. Khi quỹ đạo là hình số 8 góc nghiêng của robot có lớn hơn trường hợp hình vuông và sai lệch này tồn tại trong suốt thời gian di chuyển vì ballbot luôn chuyển hướng đối với quỹ đạo này.

Trường hợp: Quỹ đạo đặt là hình tròn sau khi được làm trễ:



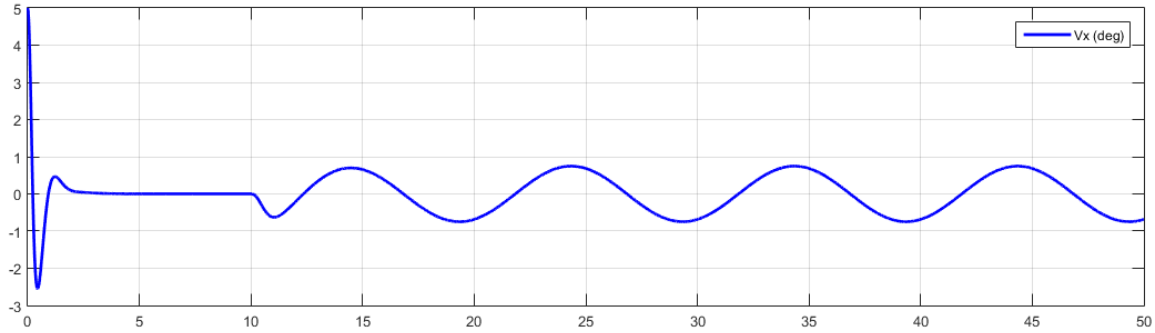
Hình 4.32: Quỹ đạo quy hoạch và quỹ đạo đặt hình tròn

Đáp ứng của hệ thống với quỹ đạo đặt hình tròn đã được làm trễ:

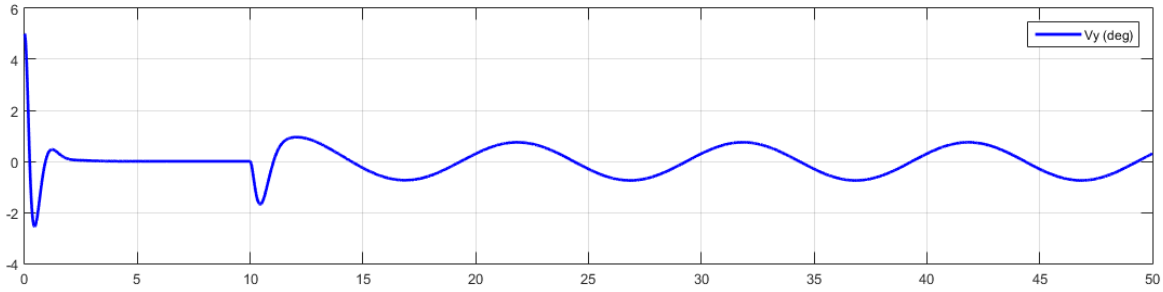


Hình 4.33: Đáp ứng của hệ thống với quỹ đạo đặt hình tròn

Đáp ứng góc nghiêng của hệ thống:



Hình 4.34: Đáp ứng góc nghiêng theo trục x với quỹ đạo đặt hình tròn



Hình 4.35: Đáp ứng góc nghiêng theo trục y với quỹ đạo đặt hình tròn

Ta thấy với quỹ đạo đặt là hình tròn ngoài việc robot phải chuyển hướng liên tục còn có lực *** tác động làm cho góc nghiêng của hệ thống giao động lớn nhất trong ba trường hợp của quỹ đạo.

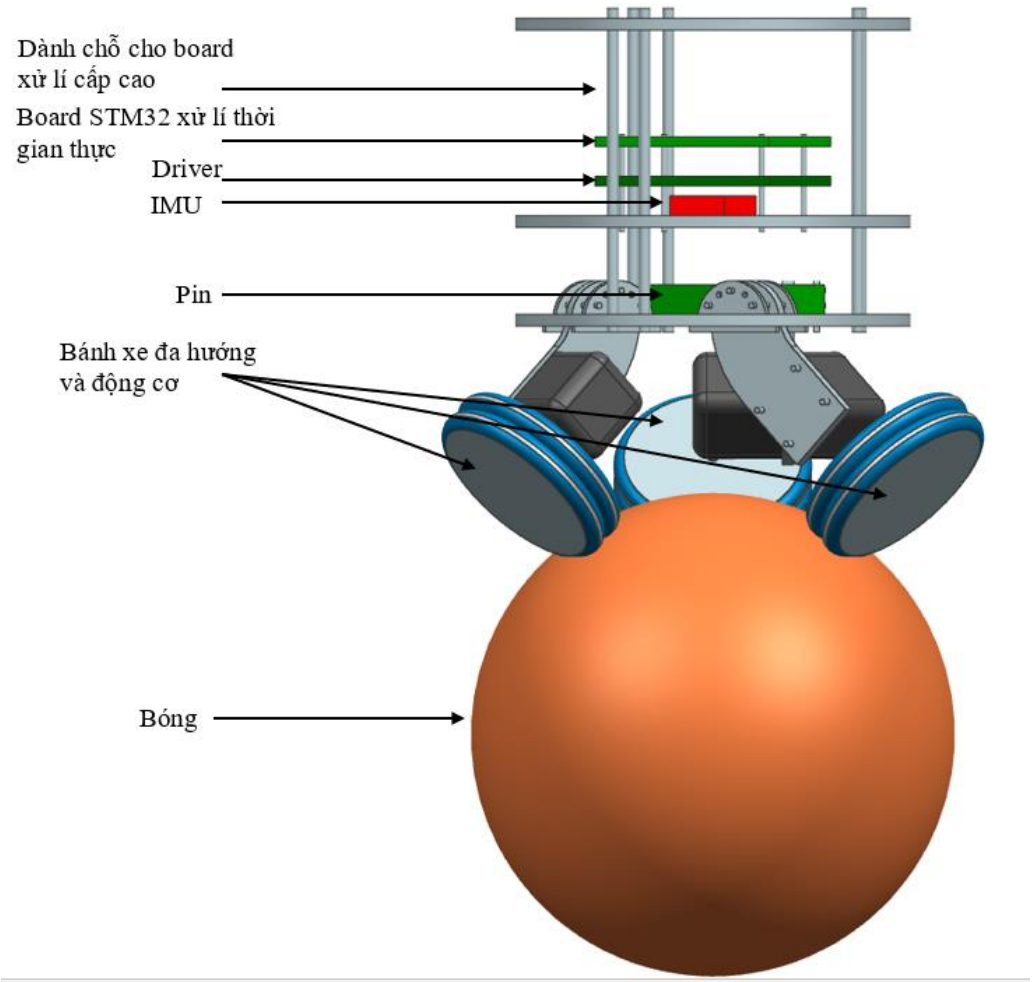
Chương 5. THỰC NGHIỆM

5.1. Mô hình robot một bánh

Mô hình ballbot được thiết kế đơn giản bao gồm phần thân robot là các tấm mica để đặt board mạch và cố định động cơ. Các tấm này được gắn với với nhau bằng bulong và trục vít. Bóng được chọn là quả bóng rổ size 7 đường kính 24 cm.

Cơ cấu lái gồm ba động cơ đặt lệch nhau góc 120^{o}, bánh xe để lái được chọn là bánh xe đa hướng có đường kính 10 cm.

Bản vẽ mô hình ballbot:

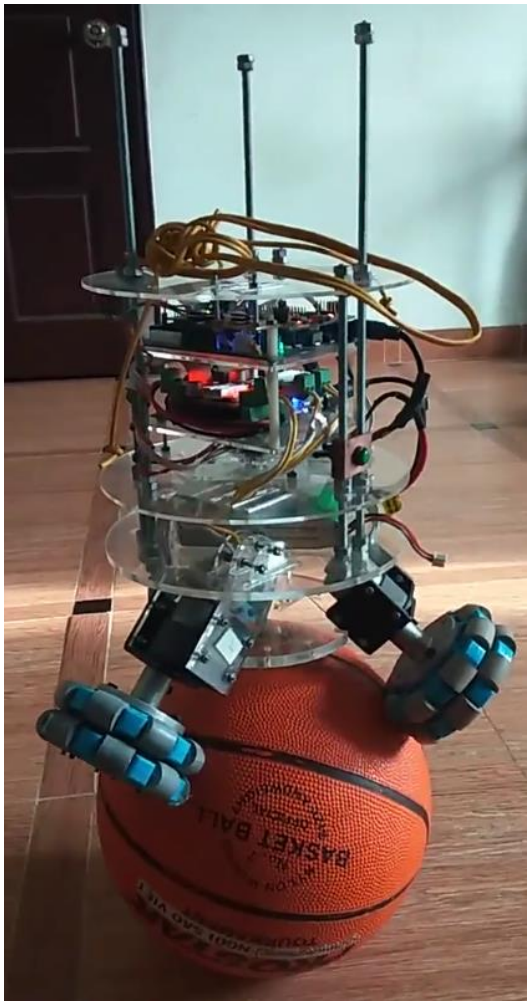


Hình 5.1 Bản vẽ mô hình ballbot

Phần điện cho ballbot được thiết kế bao gồm pin được đặt ở lớp dưới cùng của robot bao gồm pin nguồn cho động cơ và pin nguồn cho mạch điều khiển. Tiếp theo đó, IMU được đặt ở vị trí tương đối gần tâm của robot. Lớp trên là phần mạch điện, board driver được đặt dưới cùng, gần sát pin và động cơ. Bên trên là STM32F407 board xử lý thời gian thực. Mạch đệm cho STM32 có phần mở rộng và cổng giao tiếp với các board xử lý cấp cao gần với lớp người dùng.

5.2. Mô hình thực tế robot một bánh

Mô hình ballbot đã thi công thực tế:



Hình 5.2: Mô hình ballbot đã thi công

Board điều khiển thời gian thực được chọn là STM32F4-discovery board. 32-bit ARM cortex M4, với FPU core, DSP core, DMA core, 1MB flash, 192KB RAM, tần số xung nhịp lớn nhất 168MHz, tốc độ tính toán 1.25 DMIPS/MHz. Board có nhiệm vụ đọc giá trị từ cảm biến (IMU và encoder), xuất xung PWM điều khiển động cơ servo, thực hiện và xử lý thuật toán.

Driver cho động cơ là VNH2SP30 evaluation board (ST chip thiết kế cho các ứng dụng automotive) dải điện áp hoạt động từ 5.5 – 16v, dòng tối đa 30A, dòng liên tục 14A, trở kháng nội MOSFET: 19mΩ, tần số PWM tối đa 20kHz, bảo vệ quá dòng và quá áp.

Động cơ được chọn là servo GA37, 12V có giảm tốc 1:33, 11x33 = 363 xung / vòng, đường kính trục 6mm.

Bánh xe đa hướng sử dụng có đường kính 10cm, 8 bánh vệ tinh và trục nối. Bánh xe đa hướng được lắp theo cặp để có thể lăn trên bóng tốt hơn.

Cảm biến góc nghiêng, phần quan trọng nhất của robot, các cảm biến đã được thử nghiệm sử dụng bao gồm:

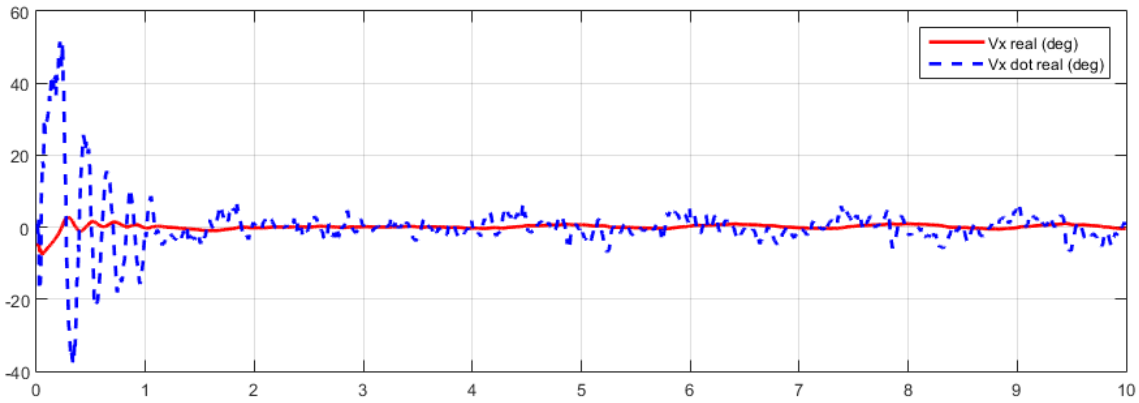
- GY951 IMU cho kết quả ước lượng góc nghiêng khá tốt tuy nhiên tốc độ cập nhật chậm và không đạt yêu cầu (20ms trong khi tốc độ điều khiển là 5ms).
- Giải thuật ước lượng cũng đã được xây dựng cho MPU6050 cho kết quả khá tốt, đạt tốc độ cập nhật cần thiết nhưng cảm biến này không ổn định và có những điều kiện làm sai giải thuật ước lượng như sốc từ bên ngoài.
- ADIS là giải pháp cho kết quả tốt nhất, tốc độ cập nhật cao và góc ước lượng có sai số chấp nhận được.

5.3. *Kết quả điều khiển robot thực tế*

5.3.1. *Đáp ứng của hệ thống với bộ điều khiển LQR*

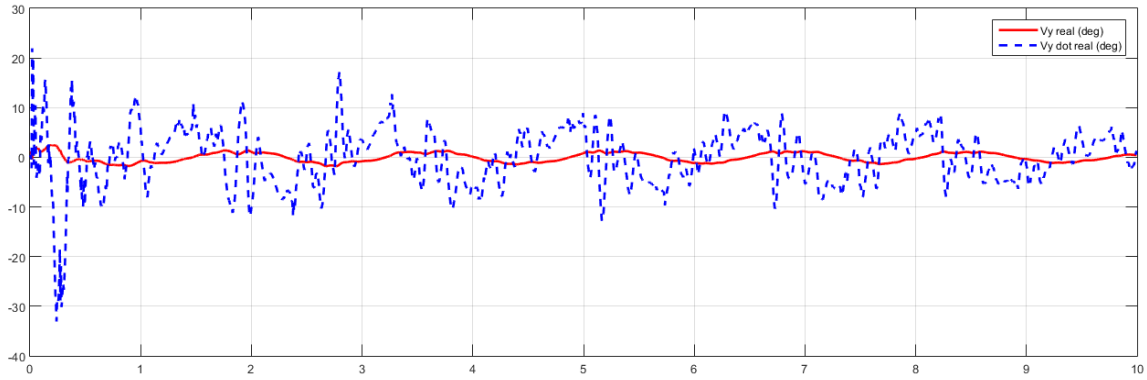
Chạy thử nghiệm thuật toán LQR được thiết kế ở 4.1.2, trên mô hình thực.

Đáp ứng góc lệch thân robot theo trục x (xét trong mặt phẳng Oyz):



Hình 5.3: Đáp ứng góc lệch thân robot theo trục x và vận tốc

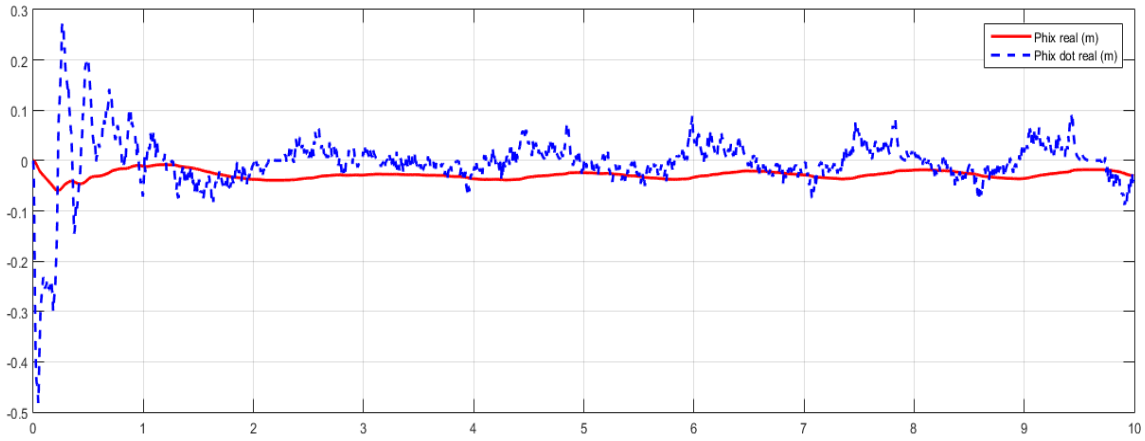
Đáp ứng góc lệch thân robot theo trục y (xét trong mặt phẳng Oxz):



Hình 5.4: Đáp ứng góc lệch thân robot theo trục y và vận tốc

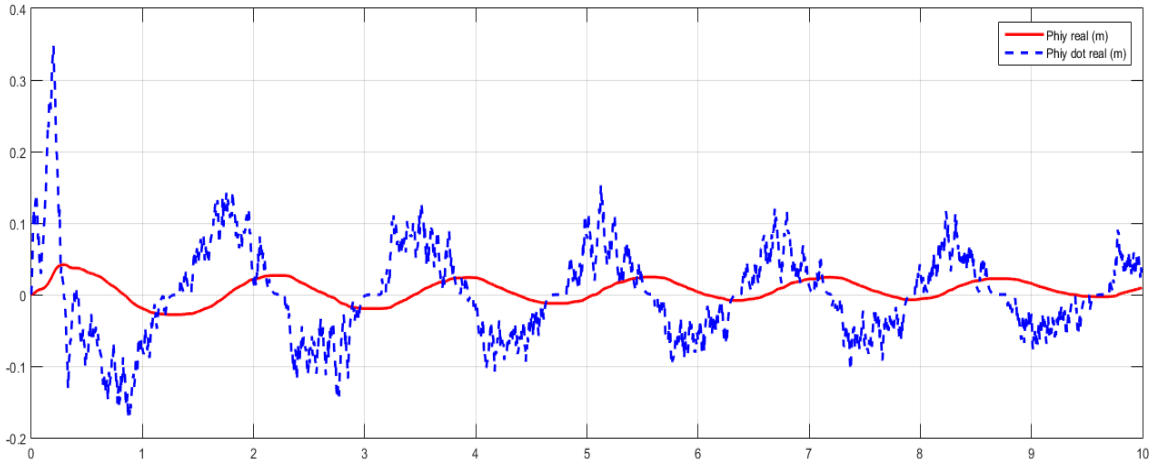
Trong mô hình thực tế, robot không đứng yên mà luôn có giao động nhỏ. Giao động này có thể xuất phát từ độ rơ của động cơ và bánh xe đa hướng gây ra góc lệch, hoặc quá trình calibrate robot cũng không hoàn toàn chính xác góc 0^0.

Vị trí robot quy đổi từ ngõ ra góc xoay của bóng theo trục x:



Hình 5.5: Đáp ứng vị trí theo trục x và vận tốc với điều khiển LQR

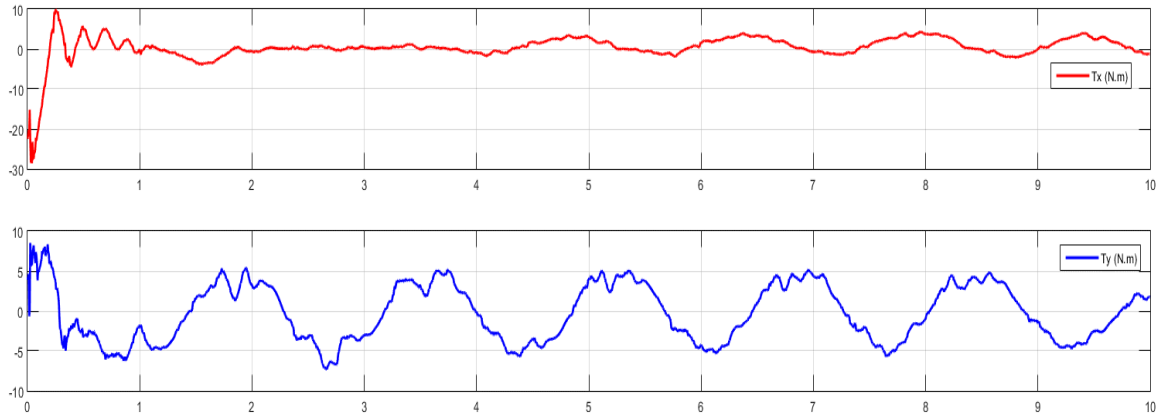
Vị trí robot quy đổi từ ngõ ra góc xoay của bóng theo trục y:



Hình 5.6: Đáp ứng vị trí theo trục y và vận tốc với điều khiển LQR

Ta thấy được sự tương quan giữa góc lệch thân robot với vị trí của bóng. Khi thân robot lệch về phía trước (xét trục y) bóng được điều khiển đề lăn về phía sau để có thể giữ được thăng bằng.

Ngõ ra bộ điều khiển:



Hình 5.7: Ngõ ra bộ điều khiển LQR

TÀI LIỆU THAM KHẢO

[1] T.B Lauwers, G.A. Kantor and R.L. Hollis, "A Dynamically Stable Single-Wheeled Mobile Robot with inverse Mouse-Ball Drive", Carnegie Mellon University, USA 2006

[2] M. Kumagai and T. Ochiai, "Development of a robot balancing on a ball", Control, Automation and System, International Conference, Oct. 2008 pp. 433-438

[3] Yorihisa Yamamoto, "NXT Ballbot Model-Based Design-Control of self-balancing robot on a ball built with LEGO Mindstorms NXT", Apr 2009

[4] Peter Fankhauser and Corsin Gwerder, "Modeling and Control of Ballbot", ETH Zurich, Switzerland, TBP, 2010

[5] Dương Hoài Nghĩa, "Hệ thống điều khiển đa biến", Nhà xuất bản Đại Học Quốc Gia Tp.HCM

[6] Cagliari University – Electrical electronic Engineering department, "A quick introduction to sliding mode control and its applications"

[7] Dianwei Qian, Jianqiang Yi and Dongbin Zao, "Hierachical sliding mode control for a class of SIMO under-actuated systems"

[8] Ching-Chih Tsai, Ming-Han Juang, Cheng-Kai Chan, Ching-Wen Liao and Siang-Jyun Chan, "Self-Balancing and position control using multiple loop approach for Ball robot"