

MỤC LỤC

1.	CHƯƠNG I	1
1.1	Đặt vấn đề	2
1.2	Các công trình nghiên cứu có liên quan	5
1.2.1	nBot [1]	5
1.2.2	JOE – A mobile Inverted Pendulum [2]	6
1.2.3	Một số dạng robot hai bánh tự cân bằng của các hãng sản xuất	7
1.2.4	Các báo cáo nghiên cứu khoa học có liên quan	8
1.3	Nội dung luận văn thạc sĩ	8
2.	CHƯƠNG II	10
2.1	Mô hình hóa robot hai bánh tự cân bằng trên địa hình phẳng	11
2.2	Mô hình hóa robot hai bánh tự cân bằng trên địa hình phẳng trong Matlab Simulink	16
2.3	Mô phỏng hệ thống	17
2.4	Kết luận	17
3.	CHƯƠNG III	18
3.2	Cơ sở lý thuyết bộ lọc Kalman	19
3.2.1	Giới thiệu về bộ lọc Kalman	19
3.2.2	Quá trình ước lượng:	20
3.2.3	Bản chất xác suất của bộ lọc	22
3.2.4	Thuật toán Kalman rời rạc:	22
3.3	Lý thuyết điều khiển trượt	24
3.3.1	Thiết kế điều khiển	26

3.3.2	Sự tồn tại nghiệm vòng kín	28
3.3.3	Định lý 1: Sự tồn tại chế độ trượt	28
3.3.4	Định lý 2:.....	29
3.3.5	Định lý 3: Chuyển động trượt	29
4.	CHƯƠNG IV	31
4.1	Phương pháp điều khiển toàn phương tuyến tính – LQR.....	32
4.1.1	Tuyến tính hóa hệ thống	32
4.1.2	Khảo sát tính điều khiển được và tính quan sát được của hệ thống:	34
4.1.3	Hàm chỉ tiêu chất lượng điều khiển	34
4.1.4	Sơ đồ mô phỏng	35
4.1.5	Kết quả mô phỏng.....	36
4.1.6	Điều khiển dùng LQR PI cho khâu vị trí.....	40
4.1.7	Kết quả mô phỏng LQR PI.....	41
4.1.8	Kết luận	41
4.2	Phương pháp điều khiển PID thích nghi mô hình tham chiếu	42
4.2.1	Đặt vấn đề.....	42
4.2.2	Cấu trúc bộ điều khiển PID cho robot hai bánh tự cân bằng	42
4.2.3	Bộ điều khiển PID với thông số cố định	43
4.2.4	Bộ điều khiển PID thích nghi mô hình tham chiếu cho robot hai bánh tự cân bằng	52
4.2.5	Kết luận	62
5.	CHƯƠNG V	63
5.1	Thiết kế mô hình robot hai bánh tự cân bằng.....	64
5.1.1	Thiết kế cơ khí	64

5.1.2	Cấu trúc điều khiển phần cứng	66
5.2	Bộ lọc Kalman cho thành phần IMU	73
5.2.1	Thực hiện bộ lọc Kalman	73
5.2.2	Kết quả thực nghiệm.....	77
5.2.3	Kết luận	79
5.3	Bộ điều khiển nhúng robot hai bánh tự cân bằng	79
5.3.1	Giới thiệu.....	79
5.3.2	Bộ điều khiển LQR PI cho khâu vị trí	81
5.3.3	Bộ điều khiển PID thích nghi mô hình tham chiếu.....	95
6.	CHƯƠNG VI.....	106
6.1	Kết quả đạt được	107
6.2	Một số hạn chế	108
6.3	Hướng phát triển	109

CHƯƠNG 1

TỔNG QUAN

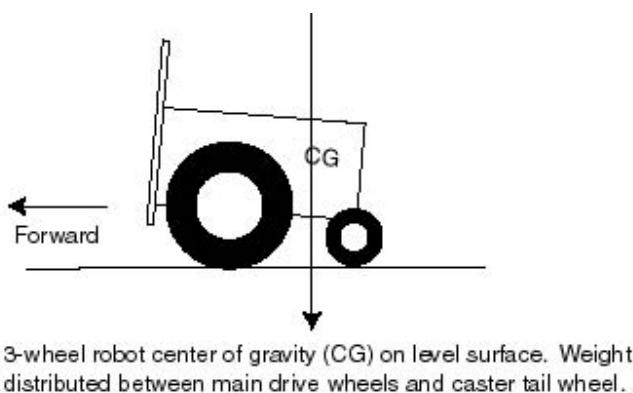
1.1 Đặt vấn đề

Trong ngành tự động hóa – điều khiển tự động nói chung và điều khiển học nói riêng, mô hình con lắc ngược là một trong những đối tượng nghiên cứu điển hình và đặc thù bởi đặc tính động không ổn định của mô hình nên việc điều khiển được đối tượng này trên thực tế đặt ra như một thử thách.

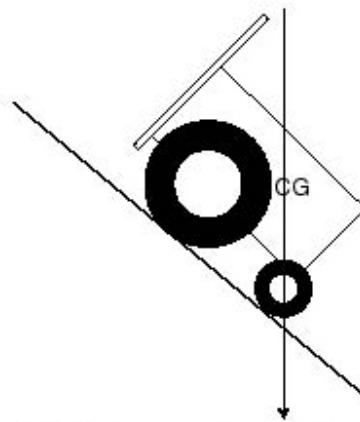
Kết quả nghiên cứu mô hình con lắc ngược cơ bản, ví dụ như mô hình xe-con lắc, con lắc ngược quay... có thể ứng dụng và kế thừa sang các mô hình tương tự khác nhưng có tính ứng dụng thực tiễn hơn, chẳng hạn như mô hình tên lửa, mô hình xe hai bánh tự cân bằng...

Như vậy, để cân đối giữa tính cơ bản với tính ứng dụng thực tiễn nhưng vẫn đảm bảo quy mô nghiên cứu nằm trong khả năng cho phép, robot hai bánh tự cân bằng được chọn làm xuất phát điểm cho ý tưởng về đề tài nghiên cứu.

Robot hai bánh tự cân bằng được xem như cầu nối kinh nghiệm giữa mô hình con lắc ngược với robot hai chân và robot giống người. Đây là dạng robot có hai bánh đồng trục, do đó khắc phục được những nhược điểm vốn có của các robot hai hoặc ba bánh kinh điển. Các robot hai hoặc ba bánh kinh điển, theo đó có cấu tạo gồm bánh dẫn động và bánh tự do (hay bất kì cái gì khác) để đỡ trọng lượng robot. Nếu trọng lượng được đặt nhiều vào bánh lái thì robot sẽ không ổn định và dễ bị ngã, còn nếu đặt vào nhiều bánh đuôi thì hai bánh chính sẽ mất khả năng bám. Nhiều thiết kế robot có thể di chuyển tốt trên địa hình phẳng nhưng không thể di chuyển lên xuống trên địa hình lồi lõm hoặc mặt phẳng nghiêng. Khi di chuyển lên đồi, trọng lượng robot dồn vào đuôi xe làm mất khả năng bám và trượt ngã.

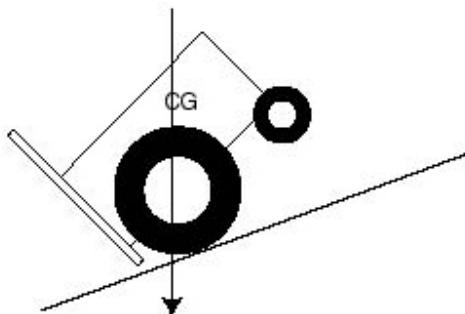


Hình 1.1 - Robot dạng 3 bánh xe di chuyển trên địa hình bằng phẳng trọng lượng được chia đều cho bánh lái và bánh dẫn nhỏ.



3-wheel robot on upward ramp. Center of gravity now situated over rear wheel, causing main wheels to lose traction.

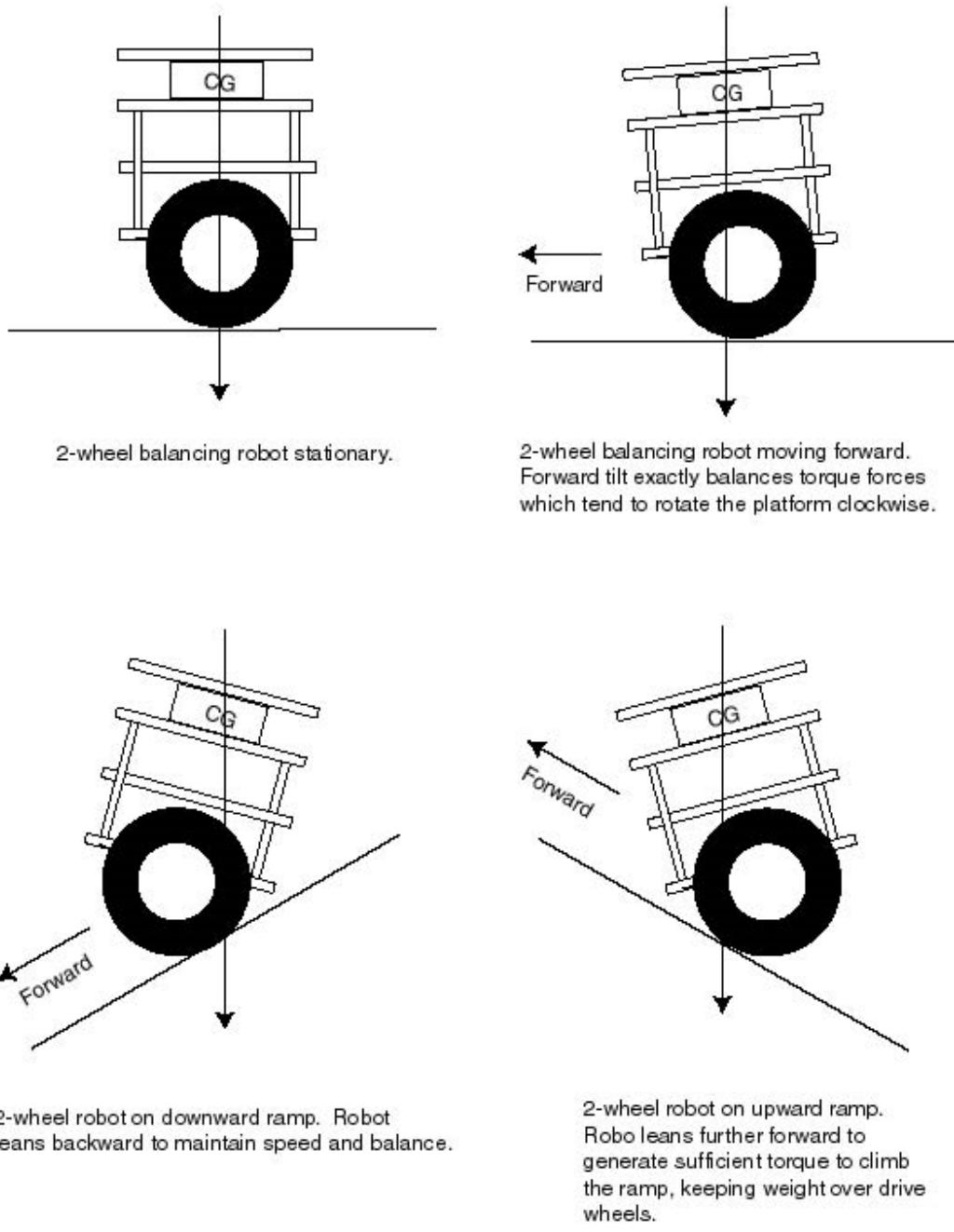
Hình 1.2 - Robot dạng 3 bánh xe khi lên dốc, trọng lượng dồn vào bánh trước khiến lực ma sát giùp xe bám trên mặt đường không được đảm bảo.



3-wheel robot on downward ramp. Center of gravity now shifts ahead of main drive wheel axle, causing robot to tip over forward.

Hình 1.3 - Robot dạng 3 bánh xe khi xuống dốc, trọng lực dồn vào bánh sau khiến xe có thể bị lật úp.

Ngược lại, các robot dạng hai bánh đồng trực lại thăng bằng rất linh động khi di chuyển trên địa hình phức tạp, mặc dù bản thân robot là một hệ thống không ổn định. Khi robot di chuyển trên địa hình dốc, nó tự động nghiêng ra trước và giữ cho trọng lượng dồn về hai bánh chính. Tương tự, khi di chuyển xuống dốc, nó nghiêng ra sau và giữ trọng tâm rơi vào bánh chính. Vì vậy, không bao giờ có hiện tượng trọng tâm xe rơi ngoài vùng đỡ bánh xe để có thể gây ra lật úp.



Hình 1.4 - Robot 2 bánh di chuyển trên các địa hình khác nhau theo hướng bảo toàn sự thăng bằng

Chính vì những ưu điểm về tính thăng bằng và di chuyển linh hoạt như trên, robot hai bánh tự cân bằng nhận được sự quan tâm từ nhiều nhà nghiên cứu và các hãng sản xuất robot trên thế giới.

1.2 Các công trình nghiên cứu có liên quan

1.2.1 nBot [1]



Hình 1.5 - nBot

Robot nBot do David P.Anderson chế tạo. Nguyên tắc điều khiển nBot như sau: các bánh xe sẽ chạy theo hướng mà phần trên robot sắp ngã, nếu bánh xe có thể được lái theo cách giữ vững trọng tâm robot thì robot sẽ được giữ cân bằng.

Quá trình điều khiển sử dụng tín hiệu từ hai cảm biến: cảm biến góc nghiêng của thân robot so với phương của trọng lực và encoder gắn ở bánh xe để đo vị trí robot. Tín hiệu này hình thành nên 4 biến: góc nghiêng thân robot, vận tốc góc nghiêng, vị trí robot và vận tốc robot; 4 biến này được tính toán thành điện áp điều khiển động cơ cho robot.

1.2.2 JOE – A mobile Inverted Pendulum [2]



Hình 1.1 - JOE

JOE do phòng thí nghiệm điện tử công nghiệp của viện Công nghệ Liên bang Lausanne, Thụy Sĩ tạo ra vào năm 2002. Robot JOE cao 65cm, nặng 12kg, tốc độ tối đa khoảng 1.5m/s, có khả năng leo dốc nghiêng đến 30° .

Nguồn điện cấp là nguồn pin 32V dung lượng 1.8Ah. Hình dạng của nó gồm hai bánh xe đồng trục, mỗi bánh gắn với một động cơ DC, robot này có thể chuyển động xoay theo hình chữ U. Hệ thống điều khiển gồm hai bộ điều khiển “không gian trạng thái” (state space) tách rời nhau, kiểm soát động cơ để giữ cân bằng cho hệ thống. Thông tin trạng thái được cung cấp bởi hai encoder quang và hai cảm biến: gia tốc góc và con quay hồi chuyển (gyro). JOE được điều khiển bởi một bộ điều khiển từ xa RC. Bộ điều khiển trung tâm và xử lý tín hiệu là một board xử lý tín hiệu số (DSP) phát triển bởi chính nhóm và của viện Federal, kết hợp với FPGA của XILINC.

1.2.3 Một số dạng robot hai bánh tự cân bằng của các hãng sản xuất



Hình 1.6 - Xe Segway



Hình 1.7 - Winglet của TOYOTA



Hình 1.8 - NXTway-GS của LEGO MINDSTORMS

1.2.4 Các báo cáo nghiên cứu khoa học có liên quan

Các báo cáo nghiên cứu khoa học về robot hai bánh tự cân bằng tập trung vào những vấn đề sau:

- Mô hình hóa hệ thống robot hai bánh tự cân bằng [1] [2] [3] [22][17]
- Điều khiển robot hai bánh tự cân bằng sử dụng các phương pháp điều khiển tuyến tính [1] [2] [3] [22]
- Điều khiển robot hai bánh tự cân bằng sử dụng các phương pháp điều khiển phi tuyến [6] [7][10][12]
- Điều khiển robot hai bánh tự cân bằng sử dụng các phương pháp điều khiển phi tuyến kết hợp với giải thuật điều khiển thông minh [8][9][11][13][18][19][20]

1.3 Nội dung luận văn thạc sĩ

Luận văn có cấu trúc gồm 6 phần chính như sau:

Chương I: Giới thiệu tổng quan về mô hình robot 2 bánh tự cân bằng, tình hình nghiên cứu hiện nay của thế giới về đề tài này.

Chương II: Mô tả mô hình toán học của robot 2 bánh tự cân bằng trên địa hình phẳng. Mô hình hóa và khảo sát đặc tính động học của đối tượng trên Matlab simulink.

Chương III: Cơ sở lý thuyết, kiến thức cơ sở và các kiến thức nền tảng để điều khiển robot.

Chương IV: Xây dựng giải thuật điều khiển robot, mô phỏng đáp ứng của hệ thống để kiểm tra tính phù hợp của giải thuật điều khiển.

Chương V: Hiện thực hóa robot 2 bánh tự cân bằng về kết cấu cơ khí, điện tử, thành phần cảm biến, hệ thống điều khiển nhúng. So sánh kết quả đạt được giữa mô phỏng và thực nghiệm.

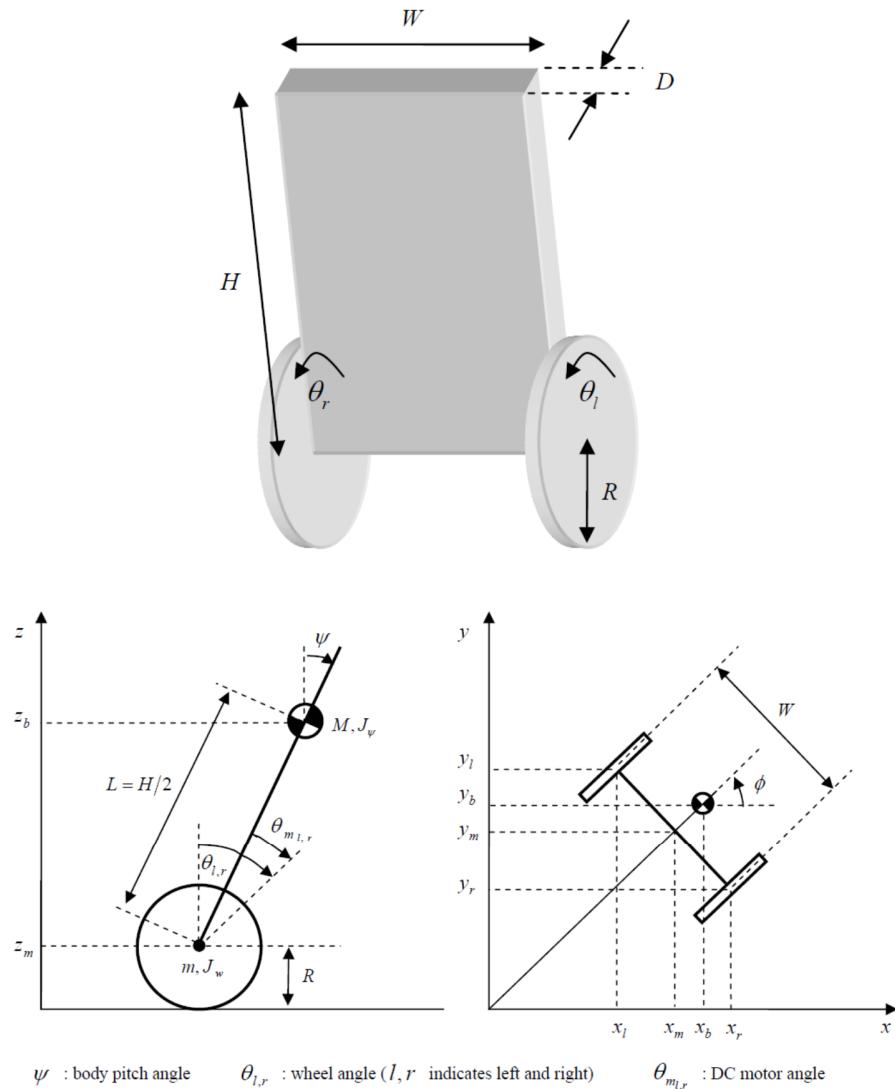
Chương VI: Ưu – khuyết điểm của các giải thuật đã thực hiện trong luận văn, đề xuất hướng nghiên cứu tiếp theo để hoàn thiện và mở rộng đề tài.

CHƯƠNG II

ĐẶC TÍNH ĐỘNG LỰC HỌC

2.1 Mô hình hóa robot hai bánh tự cân bằng trên địa hình phẳng

Xây dựng hệ phương trình trạng thái mô tả hệ thống robot hai bánh tự cân bằng.



Hình 2.1 - Mô hình robot hai bánh tự cân bằng trong mặt phẳng

Kí hiệu	Ý nghĩa
$m - [kg]$	Khối lượng bánh xe
$M - [kg]$	Khối lượng robot

$R - [m]$	Bán kính bánh xe
$W - [m]$	Chiều rộng robot
$D - [m]$	Chiều sâu robot
$H - [m]$	Chiều cao robot
$L - [m]$	Khoảng cách từ trọng tâm robot đến trục bánh xe
f_w	Hệ số ma sát giữa bánh xe và mặt phẳng di chuyển
f_m	Hệ số ma sát giữa robot và động cơ DC
$J_m - [kg \cdot m^2]$	Moment quán tính của động cơ DC
$R_m - [Ohm]$	Điện trở động cơ DC
$Kb - [V \sec / rad]$	Hệ số EMF của động cơ DC
$Kt - [Nm / A]$	Moment xoắn của động cơ DC
n	Tỉ số giảm tốc
$g - [m / s^2]$	Gia tốc trọng trường
$\theta - [rad]$	Góc trung bình của bánh trái và phải
$\theta_{l,r} - [rad]$	Góc của bánh trái và phải
$\psi - [rad]$	Góc nghiêng của phần thân robot
$\phi - [rad]$	Góc xoay của robot
$x_l, y_l, z_l - [m]$	Tọa độ bánh trái
$x_r, y_r, z_r - [m]$	Tọa độ bánh phải
$x_m, y_m, z_m - [m]$	Tọa độ trung bình
$F_\theta, F_\psi, F_\phi - [Nm]$	Moment phát động theo các phương khác nhau
$F_{l,r} - [Nm]$	Moment phát động của động cơ bánh trái, phải
$i_l, i_r - [A]$	Dòng điện động cơ bánh trái, phải
$v_l, v_r - [V]$	Điện áp động cơ bánh trái, phải

Sử dụng phương pháp Euler-Lagrange để xây dựng mô hình động học. Giả sử tại thời điểm $t = 0$, robot di chuyển theo chiều dương trục x, ta có các phương trình sau:

$$\begin{bmatrix} \theta \\ \phi \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(\theta_l + \theta_r) \\ \frac{R}{W}(\theta_l - \theta_r) \end{bmatrix} \quad [2.1]$$

$$\begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = \begin{bmatrix} \int \dot{x}_m dt \\ \int \dot{y}_m dt \\ R \end{bmatrix} \quad [2.2]$$

$$\text{và } \begin{bmatrix} \dot{x}_m \\ \dot{y}_m \end{bmatrix} = \begin{bmatrix} R\dot{\theta} \cos \phi \\ R\dot{\theta} \sin \phi \end{bmatrix} \quad [2.3]$$

$$\begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} = \begin{pmatrix} x_m - \frac{W}{2} \sin \phi \\ y_m + \frac{W}{2} \cos \phi \\ z_m \end{pmatrix} \quad [2.4]$$

$$\text{và } \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = \begin{pmatrix} x_m + \frac{W}{2} \sin \phi \\ y_m - \frac{W}{2} \cos \phi \\ z_m \end{pmatrix} \quad [2.5]$$

$$\begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} = \begin{bmatrix} x_m + L \sin \psi \cos \phi \\ y_m + L \sin \psi \sin \phi \\ z_m + L \cos \psi \end{bmatrix} \quad [2.6]$$

Phương trình động năng của chuyển động tịnh tiến:

$$T_1 = \frac{1}{2} m (\dot{x}_l^2 + \dot{y}_l^2 + \dot{z}_l^2) + \frac{1}{2} m (\dot{x}_r^2 + \dot{y}_r^2 + \dot{z}_r^2) + \frac{1}{2} M (\dot{x}_b^2 + \dot{y}_b^2 + \dot{z}_b^2) \quad [2.7]$$

Phương trình động năng của chuyển động quay

$$T_2 = \frac{1}{2} J_w \dot{\theta}_l^2 + \frac{1}{2} J_w \dot{\theta}_r^2 + \frac{1}{2} J_\psi \dot{\psi}^2 + \frac{1}{2} J_\phi \dot{\phi}^2 + \frac{1}{2} n^2 J_m (\dot{\theta}_l - \dot{\psi})^2 + \frac{1}{2} n^2 J_m (\dot{\theta}_r - \dot{\psi})^2 \quad [2.8]$$

Với:

$$\frac{1}{2}n^2J_m(\dot{\theta}_l - \psi)^2; \frac{1}{2}n^2J_m(\dot{\theta}_r - \psi)^2 \quad [2.9]$$

là động năng quay của phần ứng động cơ trái và phải.

Phương trình thế năng:

$$U = mgz_l + mgz_r + Mgz_b \quad [2.10]$$

Phương trình Lagrange

$$L = T_1 + T_2 - U \quad [2.11]$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = F_\theta \quad [2.12]$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\psi}}\right) - \frac{\partial L}{\partial \psi} = F_\psi \quad [2.13]$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\phi}}\right) - \frac{\partial L}{\partial \phi} = F_\phi \quad [2.14]$$

Lấy đạo hàm L theo các biến θ, ψ, ϕ ta được:

$$[(2m+M)R^2 + 2J_w + 2n^2J_m]\ddot{\theta} + (MLR\cos\psi - 2n^2J_m)\ddot{\psi} - MLR\dot{\psi}^2 \sin\psi = F_\theta \quad [2.15]$$

$$(MLR\cos\psi - 2n^2J_m)\ddot{\theta} + (ML^2 + J_\psi + 2n^2J_m)\ddot{\psi} - MgL\sin\psi - ML^2\dot{\phi}^2 \sin\psi \cos\psi = F_\psi \quad [2.16]$$

$$\left[\frac{1}{2}mW^2 + J_\phi + \frac{W^2}{2R^2}(J_w + n^2J_m) + ML^2 \sin^2\psi\right]\ddot{\phi} + 2ML^2\dot{\psi}\dot{\phi} \sin\psi \cos\psi = F_\phi \quad [2.17]$$

Momen động lực do động cơ DC sinh ra:

$$\begin{bmatrix} F_\theta \\ F_\psi \\ F_\phi \end{bmatrix} = \begin{bmatrix} F_l + F_r \\ F_\psi \\ \frac{W}{2R}(F_r - F_l) \end{bmatrix} \quad [2.18]$$

Và:

$$\begin{aligned} F_l &= nK_t i_l + f_m (\dot{\psi} - \dot{\theta}_l) - f_w \dot{\theta}_l \\ F_r &= nK_t i_r + f_m (\dot{\psi} - \dot{\theta}_r) - f_w \dot{\theta}_r \\ F_\psi &= -nK_t i_l - nK_t i_r - f_m (\dot{\psi} - \dot{\theta}_l) - f_m (\dot{\psi} - \dot{\theta}_r) \end{aligned} \quad [2.19]$$

Sử dụng phương pháp PWM để điều khiển động cơ nên chuyển từ dòng điện sang điện áp động cơ:

$$L_m \dot{i}_{l,r} = v_{l,r} + K_b (\dot{\psi} - \dot{\theta}_{l,r}) - R_m i_{l,r} \quad [2.20]$$

Xem điện cảm phản ứng tương đối nhỏ (gần bằng 0), có thể bỏ qua, suy ra:

$$i_{l,r} = \frac{v_{l,r} + K_b (\dot{\psi} - \dot{\theta}_{l,r})}{R_m} \quad [2.21]$$

Từ đó, các moment lực sinh ra:

$$F_\theta = \alpha(v_l + v_r) - 2(\beta + f_w)\dot{\theta} + 2\beta\dot{\psi} \quad [2.22]$$

$$F_\psi = -\alpha(v_l + v_r) + 2\beta\dot{\theta} - 2\beta\dot{\psi} \quad [2.23]$$

$$\text{với } \alpha = \frac{nK_t}{R_m} \text{ và } \beta = \frac{nK_t K_b}{R_m} + f_m \quad [2.24]$$

$$F_\phi = \frac{W}{2R}\alpha(v_r - v_l) - \frac{W^2}{2R^2}(\beta + f_w)\dot{\phi} \quad [2.25]$$

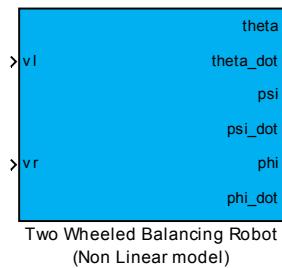
Thu được phương trình động lực học mô tả chuyển động của robot như sau:

$$\begin{aligned} &[(2m+M)R^2 + 2J_w + 2n^2 J_m]\ddot{\theta} + (MLR \cos\psi - 2n^2 J_m)\ddot{\psi} - MLR\dot{\psi}^2 \sin\psi \\ &= \alpha(v_l + v_r) - 2(\beta + f_w)\dot{\theta} + 2\beta\dot{\psi} \end{aligned} \quad [2.26]$$

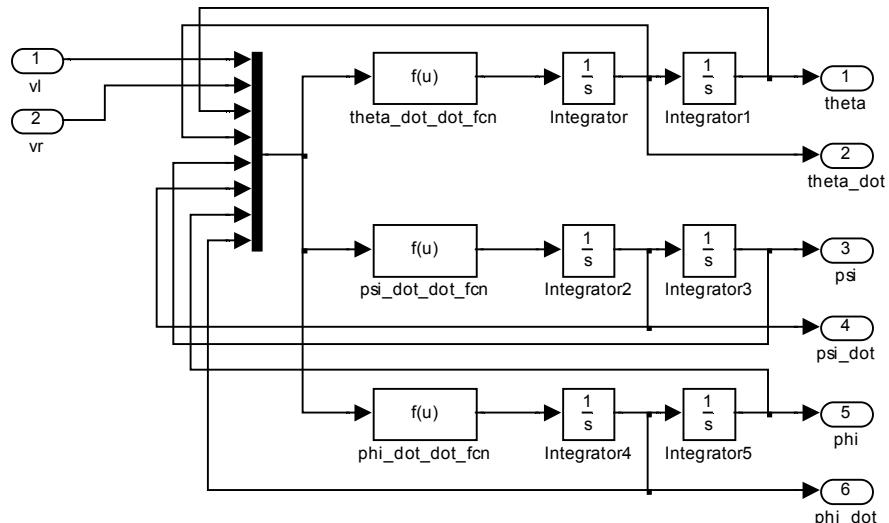
$$\begin{aligned} & \left(MLR\cos\psi - 2n^2J_m\right)\ddot{\theta} + \left(ML^2 + J_\psi + 2n^2J_m\right)\ddot{\psi} - MgL\sin\psi - ML^2\dot{\phi}^2\sin\psi\cos\psi \\ & = -\alpha(v_l + v_r) + 2\beta\dot{\theta} - 2\beta\dot{\psi} \end{aligned} \quad [2.27]$$

$$\begin{aligned} & \left[\frac{1}{2}mW^2 + J_\phi + \frac{W^2}{2R^2}(J_w + n^2J_m) + ML^2\sin^2\psi\right]\ddot{\phi} + 2ML^2\dot{\psi}\dot{\phi}\sin\psi\cos\psi \\ & = \frac{W}{2R}\alpha(v_r - v_l) - \frac{W^2}{2R^2}(\beta + f_w)\dot{\phi} \end{aligned} \quad [2.28]$$

2.2 Mô hình hóa robot hai bánh tự cân bằng trên địa hình phẳng trong Matlab Simulink



Hình 2.2 - Mô hình phi tuyến của robot hai bánh tự cân bằng trong Matlab Simulink



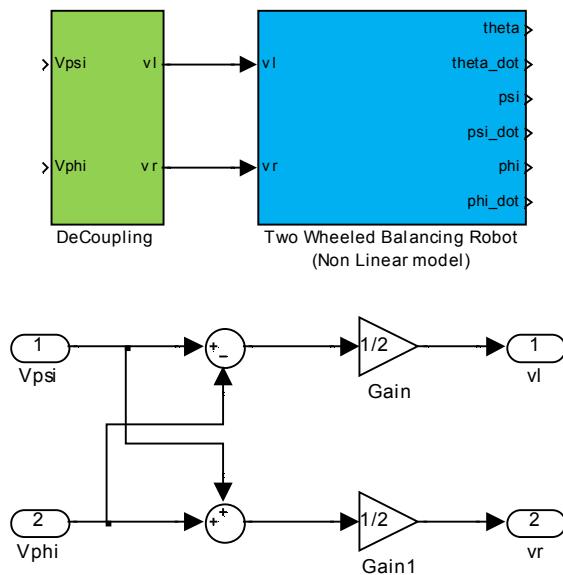
Hình 2.3 - Bên trong khối “Two Wheeled Balancing Robot (Non-Linear Model)”

Phương trình động lực học của robot như trên thể hiện mối quan hệ giữa giá trị điện áp điều khiển hai động cơ với độ nghiêng, vị trí, vận tốc của hệ robot, giá trị điện áp

hai động cơ v_l, v_r tác động lên các thông số đó dưới dạng tổng $v_l + v_r$, còn với góc xoay, giá trị điện áp hai động cơ v_l, v_r tác động lên thông số này dưới dạng hiệu $v_r - v_l$. Khi đó, tách bài toán hệ robot thành hai bài toán nhỏ hơn với hai tín hiệu điều khiển V_ψ và V_ϕ

$$\begin{cases} V_\psi = (v_l + v_r) \\ V_\phi = (v_r - v_l) \end{cases} \Rightarrow \begin{cases} v_l = \frac{1}{2}(V_\psi - V_\phi) \\ v_r = \frac{1}{2}(V_\psi + V_\phi) \end{cases} \quad [2.29]$$

Khôi thực hiện chức năng này gọi là khôi phân tách (decoupling)



Hình 2.4 - Bên trong khối “DeCoupling”

2.3 Mô phỏng hệ thống

2.4 Kết luận

CHƯƠNG III

CƠ SỞ LÝ THUYẾT

Chương này trình bày lý thuyết cơ bản về bộ lọc Kalman và điều khiển trượt cho hệ phi tuyến, làm cơ sở cho quá trình lọc tín hiệu thu được từ các cảm biến và thuật giải điều khiển robot hai bánh tự cân bằng trong chương tiếp theo.

3.2 Cơ sở lý thuyết bộ lọc Kalman

3.2.1 Giới thiệu về bộ lọc Kalman

Vào năm 1960, R.E.Kalman cho xuất bản nghiên cứu của mình, đưa ra giải pháp đệ qui để rời rạc hóa dữ liệu trong bộ lọc tuyến tính. Kể từ đó, việc giải quyết các bài toán kỹ thuật số, một lĩnh vực rất rộng lớn, đã trở nên dễ dàng hơn rất nhiều. Bộ lọc Kalman được mở rộng ra nghiên cứu và ứng dụng, đặc biệt là trong lĩnh vực tự động hay hỗ trợ việc tự định vị.

Bộ lọc Kalman thu thập và kết hợp linh động các tín hiệu từ các cảm biến thành phần. Mỗi khi mẫu thông kê nhiễu trên các cảm biến được xác nhận, bộ lọc Kalman sẽ cho ước lượng giá trị tối ưu (do đã loại được nhiễu), và có độ phân bố ổn định. Trong đề tài này tín hiệu đưa vào bộ lọc được lấy từ hai cảm biến: cảm biến gia tốc (accelerometers) sẽ cho ra giá trị góc đo và cảm biến vận tốc góc (rate gyro), và tín hiệu ngõ ra của bộ lọc là tín hiệu đã được xử lí lẫn nhau trong bộ lọc; dựa vào mối quan hệ: vận tốc góc = đạo hàm của góc.

Bộ lọc Kalman là thuật toán xử lí dữ liệu hồi quy tối ưu. Nó tối ưu đối với chi tiết cụ thể trong bất kì tiêu chuẩn có nghĩa nào. Bộ lọc Kalman tập hợp tất cả thông tin được cung cấp tới nó, xử lí các giá trị sẵn có, ước lượng các giá trị quan tâm, sử dụng các hiểu biết động học, thiết bị giá trị và hệ thống, để mô tả số liệu thống kê của hệ thống nhiễu và những thông tin bất kì về điều kiện ban đầu của các giá trị cần ước lượng.

Đối với bộ lọc Kalman, thuật ngữ “lọc” không có ý nghĩa như các bộ lọc khác. Đây là một giải thuật tính toán và ước lượng thống kê tối ưu tất cả các thông tin ngõ vào được cung cấp tới nó để có được một giá trị ra đáng tin cậy nhất cho việc xử lý tiếp theo. Do vậy lọc Kalman có thể sử dụng để loại bỏ các tín hiệu nhiễu mà được mô hình là những tín hiệu nhiễu trắng trên tất cả dải thông mà nó nhận được từ ngõ vào, dựa trên các thông kê trước đó và chuẩn trực lại giá trị ước lượng bằng các giá trị đo hiện tại với độ lệch pha gần như không tồn tại và có độ lợi tối thiểu xấp xỉ là 0 đối với những tín hiệu ngõ vào không đáng tin cậy. Mặc dù phải tốn khá nhiều thời gian xử lý lệnh, nhưng với tốc độ hiện tại của các vi điều khiển thời gian thực làm việc tính toán ước lượng tối ưu của bộ lọc này trở nên đơn giản và đáng tin cậy rất nhiều. Nhờ có cơ chế tự cập nhật các giá trị cơ sở (bias) tại mỗi thời điểm tính toán, cũng như xác định sai lệch của kết quả đo

trước với kết quả đo sau nên giá trị đo luôn được ổn định, chính xác, gần như không bị sai số về độ lợi và độ lệch pha của các tín hiệu. Hơn thế, được xây dựng bởi hàm trạng thái, do vậy bộ lọc Kalman có thể kết hợp không chỉ hai tín hiệu từ hai cảm biến, mà có thể kết hợp được nhiều cảm biến đo ở những dải tần khác nhau của cùng một giá trị đại lượng vật lý. Chính vì điều này, làm bộ lọc Kalman trở nên phổ dụng hơn tất cả những bộ lọc khác trong việc xử lý tín hiệu chính xác của các cảm biến tọa độ, như cảm biến la bàn, GPS, góc, gyro...

3.2.2 Quá trình ước lượng:

Vấn đề chung của bộ lọc Kalman nhằm ước lượng biến trạng thái $x \in R^n$ của quá trình điều khiển rời rạc được điều chỉnh bởi các phương trình tuyến tính ngẫu nhiên khác nhau. Phương trình không gian trạng thái của bộ lọc:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \quad [3.1]$$

Với giá trị $z \in R^m$ là:

$$z_k = Hx_k + v_k \quad [3.2]$$

Biến ngẫu nhiên w_k, v_k đặc trưng cho nhiều quá trình và nhiều đo của hệ. Chúng độc lập với nhau, tần suất phân bố thông thường:

$$p(w) \sim N(0, Q) \quad [3.3]$$

$$p(v) \sim N(0, R) \quad [3.4]$$

Trên thực tế, ma trận tương quan nhiều quá trình Q và ma trận tương quan nhiều đo R có thể thay đổi sau mỗi bước thời gian hay giá trị, tuy nhiên để đơn giản, trong hầu hết các trường hợp Q và R được xem là hằng số.

Ma trận vuông A trong phương trình [3.1] thể hiện mối quan hệ của các biến trạng thái ở thời điểm k-1 với thời điểm hiện tại k. Thực ra trên thực tế ma trận A thay đổi sau mỗi bước thời gian, nhưng ở đây ma trận A xem như hằng số. Ma trận B thể hiện mối liên hệ tín hiệu điều khiển $u \in R^L$ đối với biến trạng thái x. Ma trận H trong phương trình [3.2] thể hiện mối liên hệ giữa biến trạng thái với tín hiệu ra z, H cũng được xem là hằng số.

Những tính toán căn bản của bộ lọc:

Định nghĩa:

$\hat{x}_k^- = E\{x_k | y_1, y_2, \dots, y_{k-1}\}$ là giá trị ước lượng của x_k trước khi ta xử lý giá trị đo tại thời điểm k.

$\hat{x}_k^+ = E\{x_k | y_1, y_2, \dots, y_k\}$ là giá trị ước lượng của x_k sau khi ta xử lý giá trị đo tại thời điểm k.

$\hat{x}_k \in R^n$ là giá trị ước lượng trạng thái sau tại bước k có được sau khi so sánh với giá trị đo z_k . Và chúng ta có sai số ước lượng trạng thái trước và sau:

$$\begin{cases} e_k^- \equiv x_k - \hat{x}_k^- \\ e_k \equiv x_k - \hat{x}_k \end{cases} \quad [3.5]$$

Tương quan sai số ước lượng trước “priori”:

$$P_k^- = E\{e_k^- e_k^{-T}\} \quad [3.6]$$

Tương quan sai số ước lượng sau “posteriori”:

$$P_k = E\{e_k e_k^T\} \quad [3.7]$$

Khi lấy đạo hàm phương trình bộ lọc Kalman, với mục đích tìm một phương trình để tính toán trạng thái ước lượng posterior \hat{x}_k thể hiện sự tương quan giữa giá trị ước lượng priori \hat{x}_k^- và độ sai lệch giữa giá trị đo thực z_k và giá trị đo ước lượng $H\hat{x}_k^-$:

$$\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-) \quad [3.8]$$

Ma trận K trong ... là ma trận độ lợi hay hệ số trộn để tối thiểu hóa phương trình tương quan sai số posteriori. Biểu thức tính K để tối thiểu hóa phương trình ... như sau:

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} = \frac{P_k^- H^T}{(H P_k^- H^T + R)} \quad [3.9]$$

Từ đó thấy rằng tương quan sai số giá trị đo lường R tiến tới 0, khi đó:

$$\lim_{R_k \rightarrow 0} K_k = H^{-1}$$

Mặt khác, tương quan sai số ước lượng priori của P_k^- tiến đến 0, khi đó:

$$\lim_{P_k^- \rightarrow 0} K_k = 0 \quad [3.10]$$

Một cách nghĩ khác về giá trị hiệu chỉnh bù bởi K là nếu ma trận tương quan sai số giá trị đo lường R tiến tới 0 thì giá trị đo được z_k sẽ có độ tin cậy càng cao, trong khi giá trị ước lượng $H\hat{x}_k^-$ sẽ có độ tin cậy càng thấp. Mặt khác, nếu tương quan sai số ước lượng priori P_k^- tiến tới 0 thì z_k sẽ không đáng tin mà giá trị ước lượng $H\hat{x}_k^-$ sẽ càng đáng tin.

3.2.3 Bản chất xác suất của bộ lọc

Sự điều chỉnh cho x_k trong ... đã xác định bản chất ước lượng priori \hat{x}_k^- với điều kiện tất cả các giá trị đo z_k đều có nghĩa (luật phân bố Bayer). Điều đó cho thấy bộ lọc Kalman duy trì hai thời điểm đầu tiên của sự phân bố trạng thái:

$$\begin{aligned} E[x_k] &= \hat{x}_k \\ E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] &= P_k \end{aligned} \quad [3.11]$$

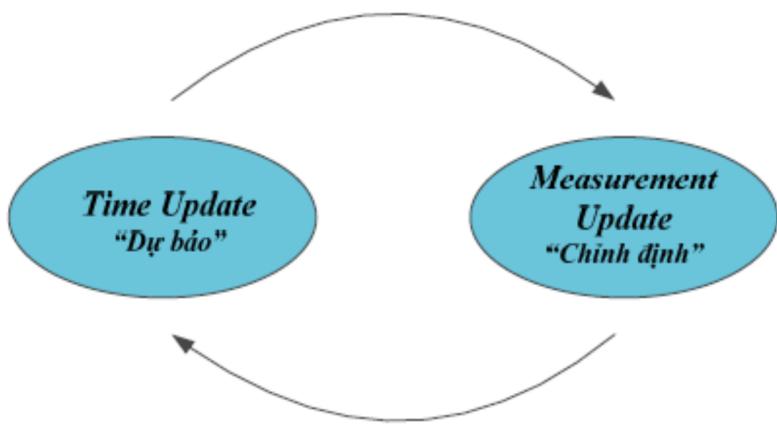
Phương trình ước lượng trạng thái posteriori phản ánh giá trị trung bình của phân bố trạng thái. Tương quan sai số ước lượng trạng thái posteriori phản ánh sự thay đổi của phân bố trạng thái. Ngoài ra ta còn có:

$$p(x_k | z_k) \sim N\left(E[x_k], E\left[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T\right]\right) = N(\hat{x}_k, P_k) \quad [3.12]$$

3.2.4 Thuật toán Kalman rời rạc:

Bộ lọc Kalman ước lượng tiến trình bằng cách sử dụng dạng điều khiển hồi tiếp: bộ lọc ước lượng các trạng thái của quá trình tại một vài thời điểm và sau đó chứa tín hiệu hồi tiếp trong các dạng của giá trị đo lường. Do đó, phương trình bộ lọc Kalman chia làm hai nhóm: *phương trình cập nhật thời gian* và *phương trình cập nhật giá trị đo lường*. *Phương trình cập nhật thời gian* chịu trách nhiệm cho việc dự báo trước (về mặt thời gian) của trạng thái hiện tại và ước lượng sai số tương quan để chứa vào bộ ước lượng trước priori cho bước thời gian tiếp theo. *Phương trình cập nhật giá trị đo lường* chịu trách nhiệm cập nhật cho tín hiệu hồi tiếp, nghĩa là cập nhật giá trị mới vào giá trị ước lượng priori để tạo tín hiệu ước lượng sau posteriori tốt hơn.

Phương trình cập nhật thời gian cũng có thể được coi là phương trình dự đoán. Trong khi đó *phương trình cập nhật giá trị đo lường* thì được xem như là phương trình hiệu chỉnh. Vì vậy, thuật toán ước lượng cuối cùng đều giống nhau ở thuật toán dự đoán và hiệu chỉnh để giải quyết vấn đề số học như hình vẽ dưới đây:



Hình 3.1 - Quy trình thực hiện của bộ lọc Kalman

Phương trình cập nhật thời gian cho bộ lọc Kalman rời rạc:

$$\begin{aligned}\hat{x}_k^- &= A\hat{x}_{k-1}^- + Bu_{k-1} \\ P_k^- &= AP_{k-1}^-A^T + Q\end{aligned}\quad [3.13]$$

Phương trình cập nhật giá trị đo lường cho bộ lọc Kalman rời rạc:

$$\begin{aligned}K_k &= P_k^- H^T \left(HP_k^- H^T + R \right)^{-1} \\ \hat{x}_k &= \hat{x}_k^- + K \left(z_k - H\hat{x}_k^- \right) \\ P_k &= (I - K_k H) P_k^-\end{aligned}\quad [3.14]$$

Nhiệm vụ đầu tiên trong suốt quá trình cập nhật giá trị đo lường là tính toán độ lợi Kalman, K_k . Bước tiếp theo là xử lý giá trị đo thực được chứa trong z_k . Sau đó, tính trạng thái ước lượng sau posteriori bằng cách kết hợp giá trị đo được theo công thức \hat{x}_k ở trên. Bước cuối cùng là tính giá trị sai số ước lượng tương quan posteriori vào P_k .

Sau mỗi chu trình tính toán của bộ lọc Kalman, các giá trị được cập nhật theo cặp, tiến trình được lặp lại với ước lượng posteriori của trạng thái trước dùng để dự đoán ước lượng priori mới. Trạng thái đệ quy tự nhiên là một trong những điểm đặc trưng của bộ lọc Kalman, nó thay thế điều kiện đệ quy ước lượng hiện tại cho giá trị đã qua.

Trong điều kiện thực hiện thực tế của bộ lọc, giá trị nhiễu tương quan R thường được dùng làm giá trị ưu tiên để tính toán cho bộ lọc. Trên thực tế, việc đo các giá trị ma trận R là rất phổ biến bởi vì chúng ta có thể đo quy trình theo nhiều cách vì vậy mà thường lấy mẫu giá trị để đưa ra khuynh hướng thay đổi của giá trị nhiễu.

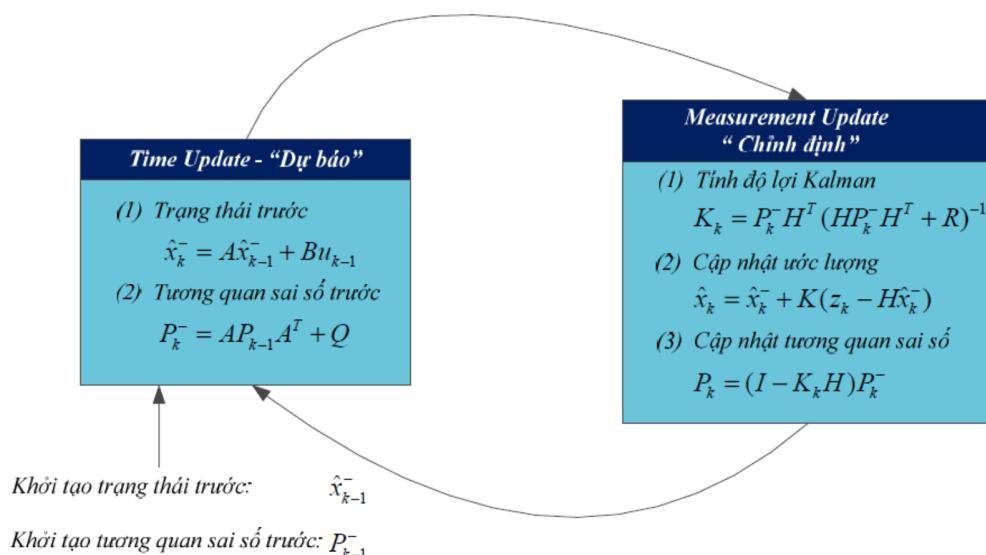
Sự xác định rõ tương quan nhiễu quá trình Q thường rất khó bởi vì điều

điền hình là chúng ta không có khả năng quan sát trực tiếp tiến trình mà chúng ta đang ước lượng. Đôi khi sự liên hệ tới những quy trình mẫu đơn giản có thể đưa ra những giá trị chấp nhận được nếu một mẫu xen vào không chắc chắn đủ với tiến trình thông qua sự lựa chọn Q . Chắc chắn trong trường hợp này, mẫu đó sẽ hi vọng rằng giá trị tiến trình là đáng tin cậy.

Trong những trường hợp khác, dù muôn hay không chúng ta đều có cái chuẩn hợp lí cho việc lựa các thông số, thường thì chất lượng bộ lọc sẽ tốt hơn nhiều lần khi có chứa sự hiệu chỉnh các tham số Q và R . Sự hiệu chỉnh thường được thực hiện gián tiếp, thường thì với sự giúp đỡ của một bộ lọc Kalman khác trong quy trình chung, liên hệ như một hệ thống đồng nhất.

Với điều kiện Q và R là các hằng số thực, cả hai cho phép ước lượng sai số tương quan P_k và độ lợi Kalman K_k sẽ ổn định nhanh chóng và sau đó trở thành hằng số.

Trong điều kiện luận văn, thông số Q và R được hiệu chỉnh dựa vào quá trình thử sai để dự đoán khuynh hướng hiệu chỉnh của hệ thống và tìm ra bộ thông số phù hợp nhất.

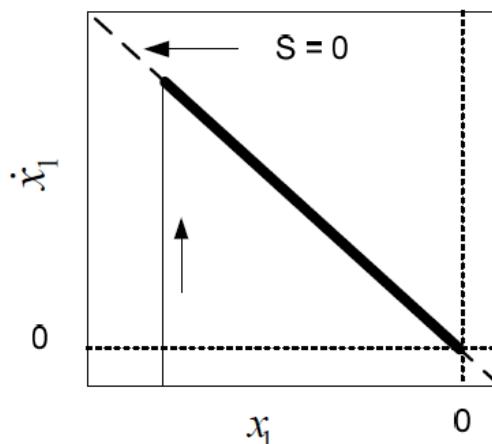


Hình 3.2 - Tổng quan chu trình thực hiện bộ lọc Kalman hoàn chỉnh

3.3 Lý thuyết điều khiển trượt

Trong lý thuyết điều khiển, điều khiển trượt là một dạng điều khiển thay đổi cấu trúc. Đây là một phương pháp điều khiển phi tuyến làm thay đổi đặc tính động học của hệ phi tuyến bằng cách sử dụng tín hiệu điều khiển đóng – cắt tần số cao. Luật điều khiển hồi tiếp trạng thái không phải là hàm liên tục theo thời gian. Nó chuyển từ cấu trúc liên tục từ trạng thái này sang trạng thái khác dựa vào vị trí hiện hành trong không gian trạng

thái. Do đó điều khiển trượt là một phương pháp điều khiển thay đổi cấu trúc. Nhiều bộ điều khiển được thiết kế sao cho các quỹ đạo luôn tiến đến một điều kiện chuyển đổi. Và quỹ đạo cuối cùng sẽ không tồn tại một cách trọn vẹn bên trong một cấu trúc điều khiển. Mà quỹ đạo cuối cùng sẽ trượt trên các biên của các cấu trúc điều khiển. Chuyển động của hệ thống khi nó trượt dọc theo các đường biên này được gọi là chế độ trượt và quỹ tích các đường biên này được gọi là bề mặt trượt - siêu diện S. Hình 3.13 là một ví dụ cho thấy quỹ đạo của một hệ thống ở chế độ điều khiển trượt. Bề mặt trượt được mô tả bởi phương trình $s = 0$ và chế độ trượt dọc theo mặt cong bắt đầu sau khoảng thời gian hữu hạn khi các quỹ đạo của hệ thống tiến đến bề mặt trượt. Trong lý thuyết điều khiển hiện đại, bất cứ hệ thống cấu trúc biến đổi, giống như hệ thống trong chế độ trượt, có thể được xem như là trường hợp đặc biệt của hệ thống động học lai – hybrid.



Hình 3.3 - Quỹ đạo mặt phẳng pha của một hệ thống được ổn định hóa bằng bộ điều khiển trượt. Sau giai đoạn đầu đạt đến chế độ trượt, các trạng thái của hệ thống “trượt” dọc theo đường thẳng $s = 0$.

Hãy hình dung, điều khiển trượt sử dụng độ lợi rất lớn để cưỡng bức các quỹ đạo của hệ thống động học để trượt theo không gian con trượt bị hạn chế. Các quỹ đạo từ chế độ trượt giảm bậc này có nhiều tính chất mong muốn (ví dụ như hệ thống trượt tự nhiên dọc theo nó cho đến khi trở về trạng thái dừng tại vị trí cân bằng mong muốn). Đặc điểm chính của điều khiển trượt là tính bền vững của nó. Bởi vì sự điều khiển có thể đơn giản như là một sự chuyển đổi giữa hai trạng thái (on/off hoặc forward/reverse). Nó không nhất thiết phải chính xác và cũng không nhạy với sự thay đổi của các tham số đi vào trong kênh điều khiển.Thêm vào đó, bởi vì luật điều khiển không phải là hàm liên tục, do đó có thể đạt được chế độ trượt trong khoảng thời gian hữu hạn (tức là tốt hơn đặc tính tiệm cận). Dưới một số điều kiện thông thường nào đó, để đạt được sự tối ưu cầu sử dụng bộ điều khiển bang – bang. Do vậy, điều khiển trượt miêu

tả bộ điều khiển tối ưu cho tập hợp rộng các hệ thống động học Một ứng dụng của bộ điều khiển trượt là điều khiển truyền động điện được vận hành bằng bộ biến đổi công suất sử dụng chế độ đóng – cắt. Do chế độ làm việc gián đoạn của các bộ biến đổi công suất đó, một bộ điều khiển trượt ở chế độ gián đoạn là sự lựa chọn tự nhiên thay cho bộ điều khiển liên tục và do đó có thể áp dụng bộ điều chế độ rộng xung hay một kỹ thuật tương tự có sử dụng tín hiệu liên tục cho một ngõ ra chỉ có thể mang các trạng thái rời rạc.

3.3.1 Thiết kế điều khiển

Xét hệ phi tuyến được mô tả bằng phương trình

$$\dot{x}(t) = f(x, t) + B(x, t)u(t)$$

$x(t) \triangleq \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_{n-1}(t) \\ x_n(t) \end{bmatrix} \in \mathbb{R}^n$ là vector trạng thái n chiều và

$u(t) \triangleq \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_{m-1}(t) \\ u_m(t) \end{bmatrix} \in \mathbb{R}^m$ là vector đầu vào m chiều được dùng cho hồi tiếp trạng thái.

Hàm $f: \mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^n$ và $B: \mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^{n \times m}$ giả sử là liên tục và đủ trơn sao cho thỏa mãn định lí Picard–Lindelöf để đảm bảo nghiệm $x(t)$ trong phương trình (3.64) tồn tại và duy nhất.

Bài toán đặt ra là thiết kế luật hồi tiếp trạng thái $u(t)$ (tức là ánh xạ từ trạng thái hiện tại $x(t)$ tại thời điểm t sang tín hiệu đầu vào $u(t)$ để ổn định hóa hệ thống động học trong phương trình (3.64) quanh gốc $\underline{x} = [0, 0, \dots, 0]^T$. Có nghĩa là dưới luật điều khiển này, bất kỳ khi nào khi hệ thống đi ra khỏi gốc nó sẽ quay trở về gốc. Ví dụ, thành phần x_i của vector trạng thái x có thể là sai lệch một ngõ ra nào đó so với một tín hiệu đã biết (thí dụ như tín hiệu sin mong muốn). Nếu tín hiệu u có thể đảm bảo rằng trạng thái x_i nhanh chóng trở về trạng thái $x_i = 0$. khi đó ngõ ra sẽ bám theo tín hiệu sin mong muốn. Trong điều khiển trượt, người thiết kế biết

rằng hệ thống vận hành theo mong muốn (tức là nó có điểm cân bằng ổn định) miễn sao nó bị ràng buộc trên bề mặt trượt. Điều khiển trượt cưỡng bức các quỹ đạo của hệ thống đi vào không gian con này và duy trì chúng ở đó sao cho các quỹ đạo trượt dọc theo bề mặt trượt. Không gian con giảm bậc này được gọi là bề mặt (mặt cong) trượt và khi hồi tiếp vòng kín cưỡng bức các quỹ đạo trượt dọc theo nó, đây được xem như là chế độ trượt của hệ thống vòng kín. Các quỹ đạo dọc theo không gian con này có thể được so sánh như các quỹ đạo dọc theo các vectơ riêng của hệ tuyến tính bất biến. Tuy nhiên, chế độ trượt bị cưỡng bức bằng cách giảm trường vector có độ lợi hồi tiếp cao. Giống như viên bi lăn dọc theo thanh ray, các quỹ đạo bị hạn chế về chế độ trượt. Việc thiết kế điều khiển trượt bao gồm:

- Chọn một siêu diện (bề mặt trượt) S sao cho quỹ đạo của hệ thống thể hiện những đặc tính mong muốn khi bị hạn chế trên bề mặt này.
- Tìm độ lợi hồi tiếp sao cho quỹ đạo của hệ thống giao nhau và vẫn duy trì trên bề mặt trượt này.

Bởi vì luật điều khiển trượt là hàm không liên tục, do đó nó có thể điều khiển các quỹ đạo về chế độ trượt trong khoảng thời gian hữu hạn (tức là sự ổn định của bề mặt trượt tốt hơn ổn định tiệm cận). Tuy nhiên, một khi các quỹ đạo tiến đến bề mặt trượt, hệ thống sẽ có những đặc tính của chế độ trượt (thí dụ như gốc $x = 0$ có thể ổn định tiệm cận trên bề mặt này).

Người thiết kế chọn hàm chuyển đổi $\sigma : \mathbb{R}^n \mapsto \mathbb{R}^m$ thể hiện “khoảng cách” của các trạng thái x đến bề mặt trượt.

- Trạng thái x ở bên ngoài mặt trượt này có $\sigma(x) \neq 0$.
- Trạng thái x ở trên mặt trượt này có $\sigma(x) = 0$.

Luật điều khiển trượt chuyển đổi từ trạng thái này sang trạng thái kia dựa vào dấu của khoảng cách này. Vì thế điều khiển trượt hoạt động giống như áp suất (stiff pressure) luôn đẩy về hướng của chế độ trượt khi $\sigma(x) = 0$. Quỹ đạo $x(t)$ mong muốn sẽ tiến dần đến bề mặt trượt. Do luật điều khiển không liên tục, các quỹ đạo nào tiến về mặt trượt trong khoảng thời gian hữu hạn. Một khi quỹ đạo tiến đến bề mặt trượt, nó sẽ trượt trên đó và có thể sẽ tiến về gốc $x = 0$.

Bề mặt trượt có $n \times m$ chiều, trong đó n là số trạng thái trong x và m là số tín hiệu đầu vào (tín hiệu điều khiển) trong u . Với mỗi chỉ mục của tín hiệu điều khiển $1 \leq k \leq m$ có $n \times 1$ bề mặt trượt cho bởi phương trình.

$$\{x \in \mathbb{R}^n : \sigma_k(x) = 0\}$$

Phản quan trong việc thiết kế hệ thống điều khiển cấu trúc biến là chọn luật điều khiển chế độ trượt (tức là bề mặt trượt được cho bởi phương trình $\sigma(x) = 0$)

Để đưa các trạng thái hệ thống x thỏa mãn $\sigma(x) = 0$ ta phải:

- Đảm bảo rằng hệ thống có khả năng tiến đến bề mặt $\sigma(x) = 0$ từ bất kỳ điều kiện đầu nào.

Khi đạt được chế độ trượt $\sigma(x) = 0$ tác động điều khiển có khả năng duy trì hệ thống ở chế độ $\sigma(x) = 0$.

3.3.2 Sự tồn tại nghiệm vòng kín

Chú ý rằng, luật điều khiển là không liên tục, do đó không liên tục Lipschitz cục bộ vì thế sự tồn tại và duy nhất nghiệm của hệ kín không được đảm bảo theo định lý Picard–Lindelöf. Vì vậy nghiệm phải được hiểu trong ngữ cảnh Filippov. Nói một cách tổng quát, hệ kín di chuyển dọc theo mặt $\sigma(x) = 0$ được xấp xỉ bởi hàm trơn $\sigma(x) = 0$. Tuy nhiên đặc tính trơn này có thể không thực hiện được. Tương tự, phương pháp điều chế độ rộng xung tốc độ cao hoặc điều chế sigma – delta tạo ra các ngõ ra được giả sử chỉ có hai trạng thái. Các vấn đề phức tạp này có thể được tránh đi bằng cách sử dụng phương pháp thiết kế bộ điều khiển phi tuyến khác. Trong một số trường hợp, thiết kế điều khiển trượt có thể được xấp xỉ bằng việc thiết kế bộ điều khiển liên tục khác

3.3.3 Định lý 1: Sự tồn tại chế độ trượt

Xét hàm Lyapunov:

$$V(\sigma(x)) = \frac{1}{2} \sigma^T(x) \sigma(x) = \frac{1}{2} \|\sigma(x)\|_2^2 \quad [3.15]$$

Đối với hệ thống cho bởi phương trình [2.1] và bề mặt trượt cho bởi phương trình [2.2], điều kiện đủ để tồn tại chế độ trượt là:

$$\underbrace{\sigma^T}_{\frac{dV}{dt}} \underbrace{\dot{\sigma}}_{\frac{d\sigma}{dt}} < 0 \quad [3.16]$$

Trong lân cận bờ mặt trượt cho bởi $\sigma(x) = 0$. Nói một cách tổng quát (đối với trường hợp tín hiệu điều khiển là đại lượng vô hướng khi $m = 0$), để đạt được $\sigma^T \dot{\sigma} < 0$, luật điều khiển hồi tiếp $u(x)$ được chọn sao cho σ và $\dot{\sigma}$ có dấu ngược nhau. Có nghĩa là:

- $u(x)$ làm cho $\dot{\sigma}(x)$ âm khi $\sigma(x)$ dương.
- $u(x)$ làm cho $\dot{\sigma}(x)$ dương khi $\sigma(x)$ âm.

$$\dot{\sigma} = \frac{\partial \sigma}{\partial x} \dot{x} = \frac{\partial \sigma}{\partial x} (f(x, t) + B(x, t)u) \quad [3.17]$$

Và do đó $u(x)$ có ảnh hưởng rất lớn đến $\dot{\sigma}$.

3.3.4 Định lý 2:

Đối với hệ thống cho bởi phương trình (1) và bờ mặt trượt cho bởi phương trình (2), không gian con mà mặt cong trượt $\{x \in \mathbb{R}^n : \sigma(x) = 0\}$ có thể được đạt đến, được cho bởi phương trình:

$$\{x \in \mathbb{R}^n : \sigma^T(x)\dot{\sigma}(x) < 0\} \quad [3.18]$$

Có nghĩa là khi toàn bộ các điều kiện đầu đến từ không gian này, hàm Lyapunov $V(\sigma)$ và các quỹ đạo x đảm bảo chắc chắn sẽ di chuyển về bờ mặt trượt $\sigma(x) = 0$. Hơn nữa, nếu điều kiện từ định lý 1 thỏa mãn, chế độ trượt sẽ đi vào vùng mà đạo hàm V bị chặn bởi 0 trong khoảng thời gian hữu hạn. Do đó chế độ trượt sẽ đạt được trong khoảng thời gian hữu hạn.

3.3.5 Định lý 3: Chuyển động trượt

Đặt $\frac{\partial \sigma}{\partial x} B(x, t)$ là không suy biến. Có nghĩa là hệ thống có thể hoàn toàn điều khiển được để đảm bảo rằng luôn có một tín hiệu điều khiển có thể dịch chuyển các quỹ

đạo di chuyển về phía mặt trượt gần hơn. Khi đó, một khi xảy ra chế độ trượt $\sigma(x) = 0$, hệ thống sẽ vẫn còn nằm trong chế độ trượt đó. Dọc theo các quỹ đạo trượt $\sigma(x)$, bằng hằng số và do đó các quỹ đạo trượt được mô tả bằng phương trình vi phân:

$$\dot{\sigma} = 0$$

Nếu trạng thái cân bằng x là ổn định đối với phương trình vi phân này, khi đó hệ thống sẽ trượt dọc theo bề mặt trượt để về trạng thái cân bằng.

Luật điều khiển tương đương trong chế độ trượt có thể tìm được bằng cách giải phương trình $\dot{\sigma}(x) = 0$ cho luật điều khiển tương đương $u(x)$. Đó là:

$$\frac{\partial \sigma}{\partial x} (f(x, t) + B(x, t)u) = 0 \quad [3.19]$$

Do đó, điều khiển tương đương:

$$u = -(\frac{\partial \sigma}{\partial x} B(x, t))^{-1} \frac{\partial \sigma}{\partial x} f(x, t) \quad [3.20]$$

Có nghĩa là, mặc dù tín hiệu điều khiển u là không liên tục, sự chuyển đổi nhanh chóng qua chế độ trượt nơi $\sigma(x) = 0$ cưỡng bức hệ thống hoạt động như là nó được điều khiển bởi một tín hiệu liên tục.

Tương tự, các quỹ đạo hệ thống ở chế độ trượt:

$$\dot{x} = f(x, t) - B(x, t)(\frac{\partial \sigma}{\partial x} B(x, t))^{-1} \frac{\partial \sigma}{\partial x} f(x, t) = f(x, t)(I - B(x, t)(\frac{\partial \sigma}{\partial x} B(x, t))^{-1} \frac{\partial \sigma}{\partial x}) \quad [3.21]$$

Kết quả hệ thống cân bằng phương trình vi phân chế độ trượt:

$$\dot{\sigma}(x) = 0$$

Từ định lý này, chuyển động trượt là bất biến (không nhạy) đối với nhiễu dù nhỏ đi vào hệ thống. Có nghĩa là khi tín hiệu điều khiển đủ lớn để bảo đảm $\sigma^T \dot{\sigma} < 0$ và $\dot{\sigma}$ bị chặn đều, chế độ trượt sẽ được duy trì như thế là không có nhiễu. Tính chất bất biến của điều khiển trượt đối với nhiễu và độ không chắc chắn của mô hình là đặc điểm thu hút nhất. Do đó bộ điều khiển trượt rất bền vững.

CHƯƠNG IV

GIẢI THUẬT ĐIỀU KHIỂN

Trong chương này trình bày 3 thuật toán để điều khiển robot hai bánh tự cân bằng: LQR PI, PID thích nghi mô hình tham chiếu và trượt PI.

4.1 Phương pháp điều khiển toàn phương tuyến tính – LQR

4.1.1 Tuyến tính hóa hệ thống

Tuyến tính hóa phương trình chuyển động của hệ robot tại điểm làm việc ($\psi \rightarrow 0, \sin \psi \rightarrow \psi, \cos \psi \rightarrow 1$, bỏ qua các thành phần bậc 2 trong phương trình), ta được hệ phương trình tuyến tính hóa như sau:

$$[(2m+M)R^2 + 2J_w + 2n^2J_m]\ddot{\theta} + (MLR - 2n^2J_m)\ddot{\psi} = \alpha(v_l + v_r) - 2(\beta + f_w)\dot{\theta} + 2\beta\dot{\psi} \quad [4.1]$$

$$(MLR - 2n^2J_m)\ddot{\theta} + (ML^2 + J_\psi + 2n^2J_m)\ddot{\psi} - MgL\psi = -\alpha(v_l + v_r) + 2\beta\dot{\theta} - 2\beta\dot{\psi} \quad [4.2]$$

$$\left[\frac{1}{2}mW^2 + J_\phi + \frac{W^2}{2R^2}(J_w + n^2J_m) \right] \ddot{\phi} = \frac{W}{2R}\alpha(v_r - v_l) - \frac{W^2}{2R^2}(\beta + f_w)\dot{\phi} \quad [4.3]$$

Suy ra hệ phương trình trạng thái tuyến tính hóa của robot:

$$\begin{aligned} \dot{x}_1 &= A_1x_1 + B_1u \\ \dot{x}_2 &= A_2x_2 + B_2u \end{aligned} \quad [4.4]$$

Trong đó:

$$x_1 = \begin{bmatrix} \theta \\ \dot{\theta} \\ \psi \\ \dot{\psi} \end{bmatrix}, \quad x_2 = \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix}, \quad u = \begin{bmatrix} v_l \\ v_r \end{bmatrix} \quad [4.5]$$

Trong đó:

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-2[(\beta + f_w)E(2,2) + \beta E(1,2)]}{\det(E)} & \frac{-gMLE(1,2)}{\det(E)} & \frac{2\beta[E(2,2) + E(1,2)]}{\det(E)} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{2[(\beta + f_w)E(1,2) + \beta E(1,1)]}{\det(E)} & \frac{gMLE(1,1)}{\det(E)} & \frac{-2\beta[E(1,1) + E(1,2)]}{\det(E)} \end{bmatrix} \quad [4.6]$$

$$B_1 = \begin{bmatrix} 0 & 0 \\ \frac{\alpha [E(2,2) + E(1,2)]}{\det(E)} & \frac{\alpha [E(2,2) + E(1,2)]}{\det(E)} \\ 0 & 0 \\ \frac{-\alpha [E(1,1) + E(1,2)]}{\det(E)} & \frac{-\alpha [E(1,1) + E(1,2)]}{\det(E)} \end{bmatrix} \quad [4.7]$$

$$A_2 = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{J}{I} \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 & 0 \\ -\frac{K}{I} & \frac{K}{I} \end{bmatrix} \quad [4.8]$$

$$E = \begin{bmatrix} (2m+M)R^2 + 2J_w + 2n^2J_m & MLR - 2n^2J_m \\ MLR - 2n^2J_m & ML^2 + J_\psi + 2n^2J_m \end{bmatrix} \quad [4.9]$$

$$I = \frac{1}{2}mW^2 + J_\phi + \frac{W^2}{2R^2}(J_w + n^2J_m) \quad [4.10]$$

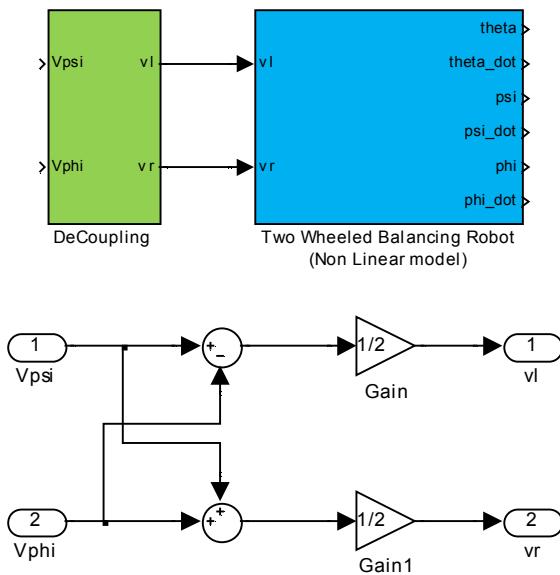
$$J = \frac{W^2}{2R^2}(\beta + f_w) \quad [4.11]$$

$$K = \frac{W}{2R}\alpha \quad [4.12]$$

Phương trình thể hiện mối quan hệ giữa giá trị điện áp điều khiển hai động cơ; với độ nghiêng, vị trí, vận tốc của hệ robot, giá trị điện áp hai động cơ v_l, v_r tác động lên các thông số đó dưới dạng tổng $v_l + v_r$, còn với góc xoay, giá trị điện áp hai động cơ v_l, v_r tác động lên thông số này dưới dạng hiệu $v_r - v_l$. Khi đó, tách bài toán hệ robot thành hai bài toán nhỏ hơn với hai tín hiệu điều khiển V_ψ và V_ϕ

$$\begin{cases} V_\psi = (v_l + v_r) \\ V_\phi = (v_r - v_l) \end{cases} \Rightarrow \begin{cases} v_l = \frac{1}{2}(V_\psi - V_\phi) \\ v_r = \frac{1}{2}(V_\psi + V_\phi) \end{cases} \quad [4.13]$$

Khối thực hiện chức năng này gọi là khối phân tách (decoupling)



Hình 4.1 - Bên trong khối “DeCoupling”

4.1.2 Khảo sát tính điều khiển được và tính quan sát được của hệ thống:

$$\text{Ma trận điều khiển } M_c = [B, AB, A^2B, A^3B]$$

$$\text{Ma trận quan sát } M_o = [C, CA, CA^2, CA^3]^T$$

Kiểm tra, ta được $\text{Rank}(M_c) = \text{Rank}(M_o) = 4$

Suy ra hệ thống điều khiển được và quan sát được.

4.1.3 Hàm chỉ tiêu chất lượng điều khiển

Mục tiêu điều khiển là tìm vector điều khiển tối ưu U thỏa mãn chỉ tiêu chất lượng J đạt cực tiểu

$$J = \int_0^\infty (x^T Q x + u^T R u) dt \quad [4.14]$$

Chọn Q_1, R_1 cho phương trình trạng thái thứ nhất, để xác định giá trị K_1 hồi tiếp cho khâu điều khiển giữ thẳng bằng cho Robot và giá trị K_2 hồi tiếp cho khâu điều khiển góc xoay robot. Thông qua quá trình thử sai, trọng số ma trận Q, R được chọn sao cho ứng với điều kiện góc nghiêng ban đầu là 0.3 (rad), giá trị điện áp v_l, v_r không quá 24V, thời gian xác lập góc nghiêng, vị trí dưới 3s, vị trí và góc nghiêng vọt lồ ít... như sau:

$$Q_1 = \begin{bmatrix} 10^3 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10^4 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ và } R_1 = \begin{bmatrix} 10^2 & 0 \\ 0 & 10^2 \end{bmatrix} \quad [4.15]$$

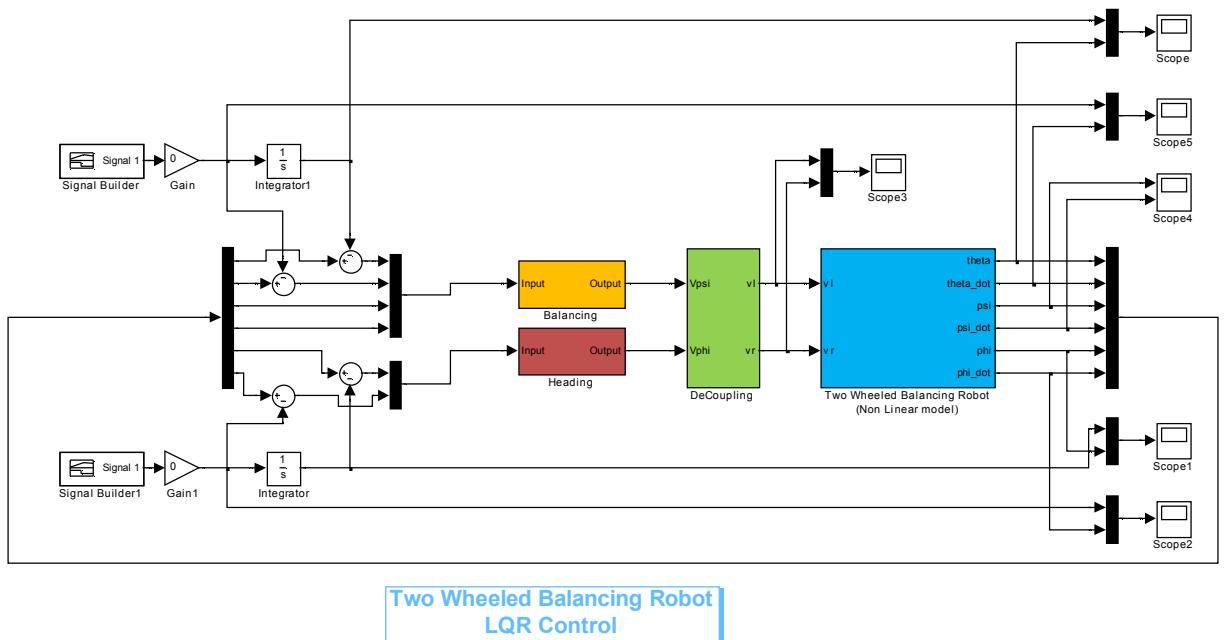
$$Q_2 = \begin{bmatrix} 10^4 & 0 \\ 0 & 10 \end{bmatrix} \text{ và } R_2 = \begin{bmatrix} 10^2 & 0 \\ 0 & 10^2 \end{bmatrix} \quad [4.16]$$

Xác định được vector hồi tiếp:

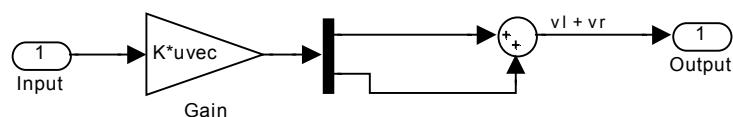
$$K_1 = \begin{bmatrix} -2.2361 & -2.8402 & -702.2036 & -95.7018 \\ -2.2361 & -2.8402 & -702.2036 & -95.7018 \end{bmatrix} \quad [4.17]$$

$$K_2 = \begin{bmatrix} -7.0711 & -5.4685 \\ 7.0711 & 5.4685 \end{bmatrix} \quad [4.18]$$

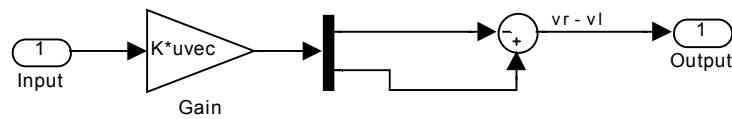
4.1.4 Sơ đồ mô phỏng



Hình 4.2 - Sơ đồ mô phỏng hệ thống điều khiển bằng phương pháp LQR



Hình 4.3 - Bên trong khối “Balancing”

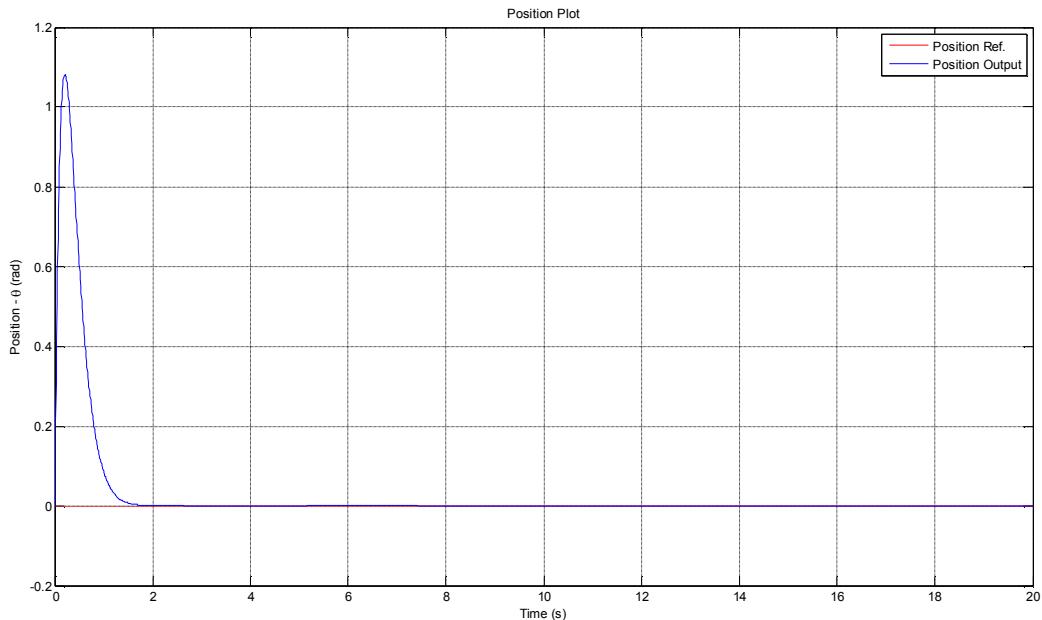


Hình 4.4 - Bên trong khối “Heading”

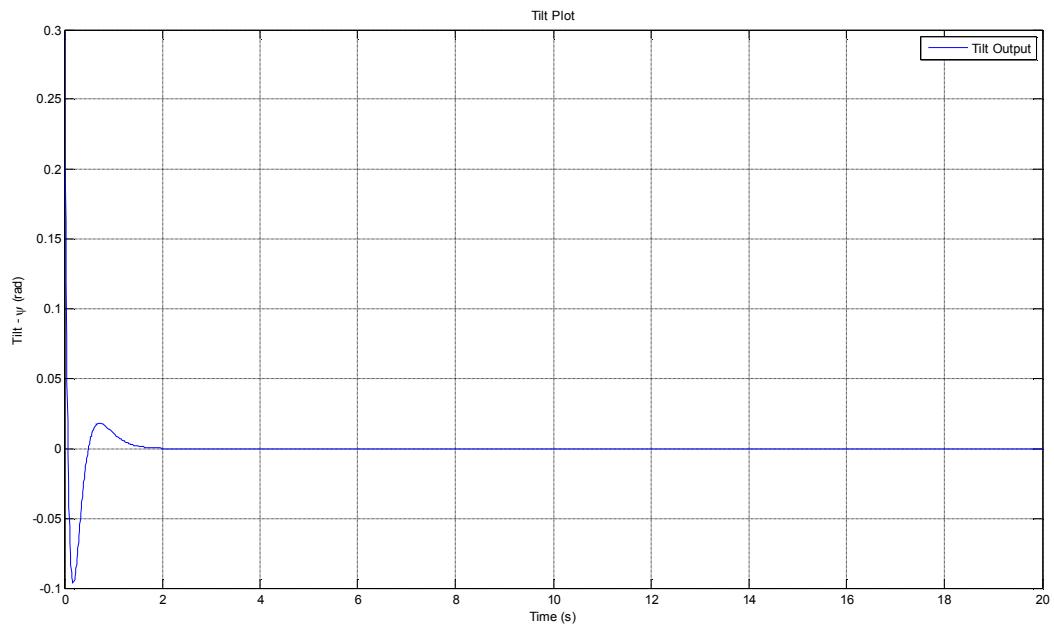
Như đã trình bày ở trên, hệ số độ lợi hồi tiếp tính theo phương pháp LQR là K_1, K_2 sử dụng tương ứng trong khối “Balancing” (điều chỉnh góc nghiêng - cân bằng, vị trí) và “Heading” (điều chỉnh góc xoay).

4.1.5 Kết quả mô phỏng

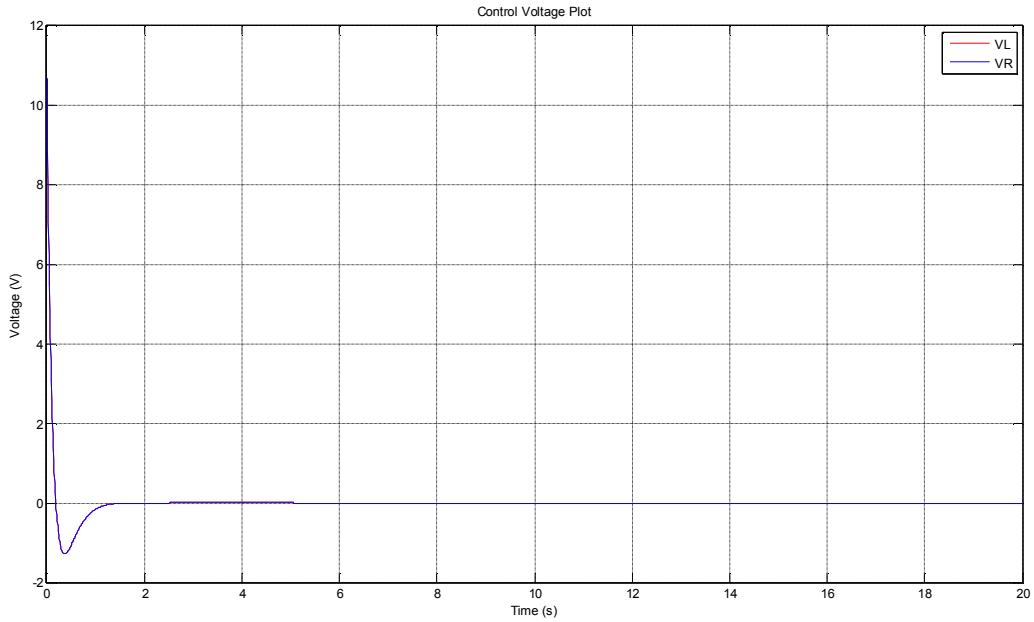
a. Đáp ứng với góc nghiêng ban đầu



Hình 4.5 - Đáp ứng ngõ ra vị trí (góc θ) ứng với điều kiện góc nghiêng ban đầu $\psi_0 = 0.3$

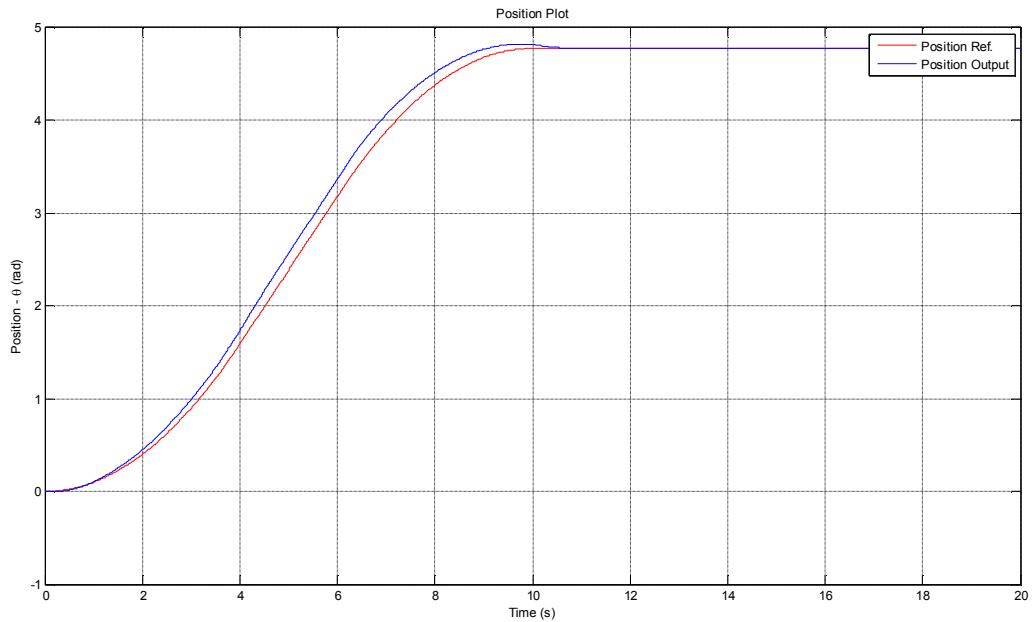


Hình 4.6 - Đáp ứng góc nghiêng (góc ψ) ứng với điều kiện góc nghiêng ban đầu $\psi_0 = 0.3$

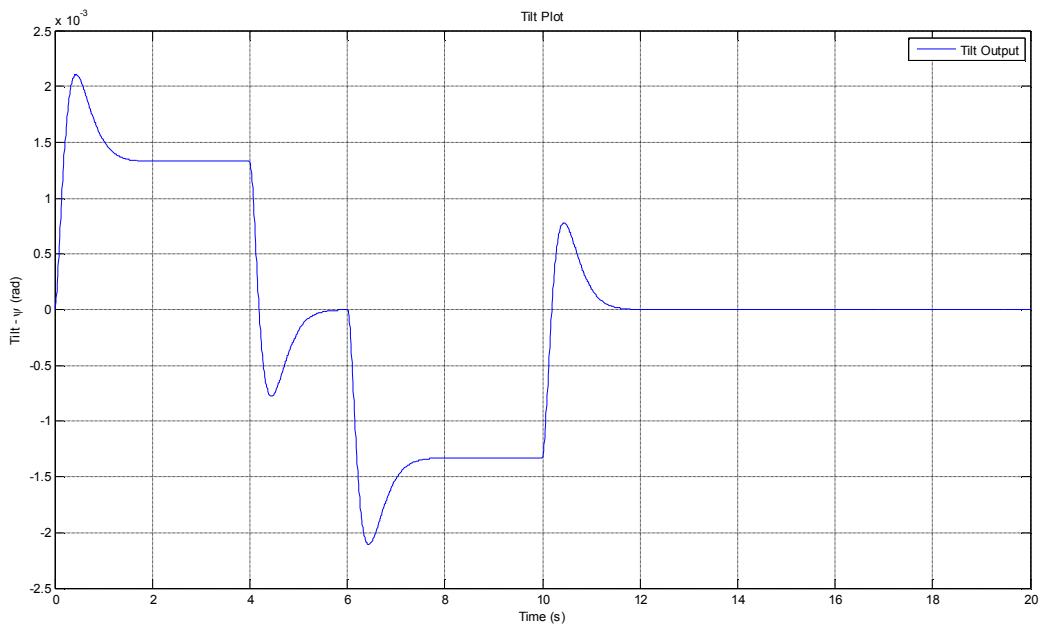


Hình 4.7 - Tín hiệu điều khiển động cơ trái (VL) và phải (VR) ứng với điều kiện góc nghiêng ban đầu $\psi_0 = 0.3$

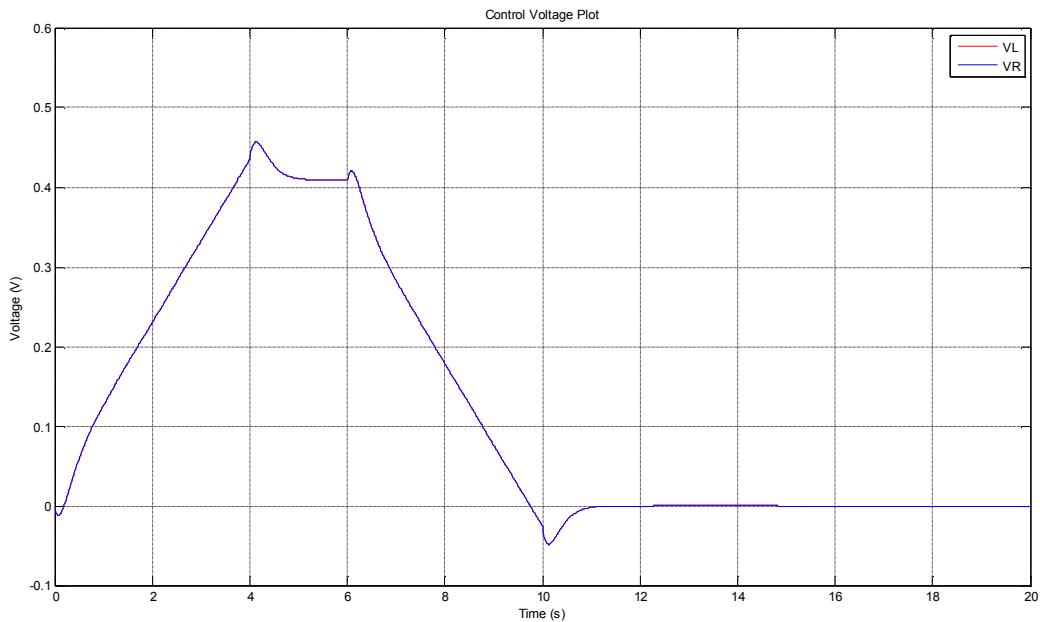
b. Đáp ứng với tín hiệu đặt vị trí



Hình 4.8 - Đáp ứng ngõ ra vị trí (góc θ) ứng với tín hiệu đặt vị trí

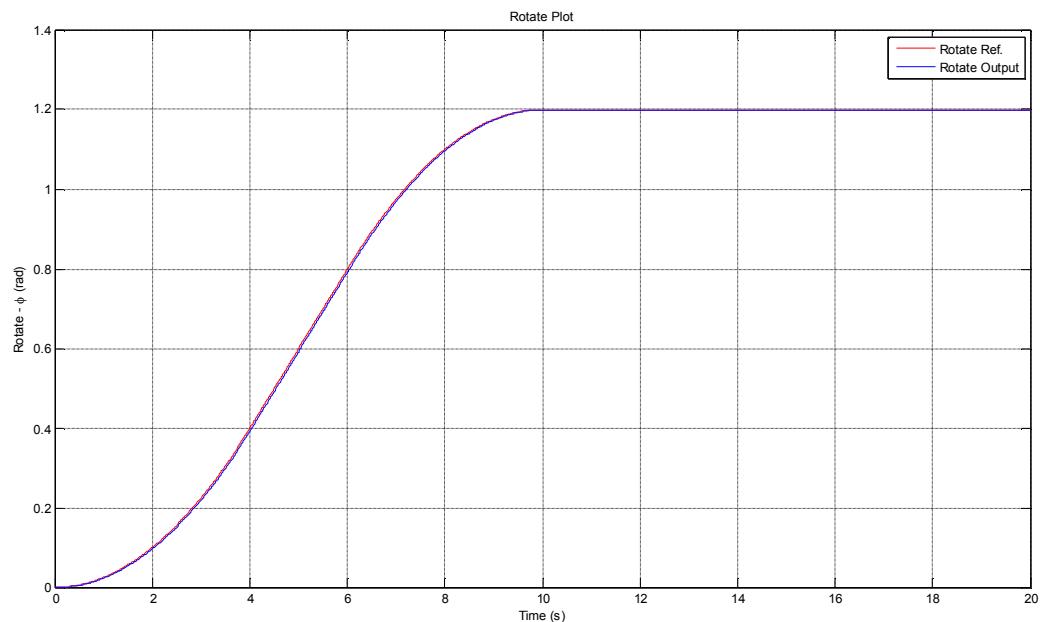


Hình 4.9 - Đáp ứng góc nghiêng (góc ψ) ứng với tín hiệu đặt vị trí

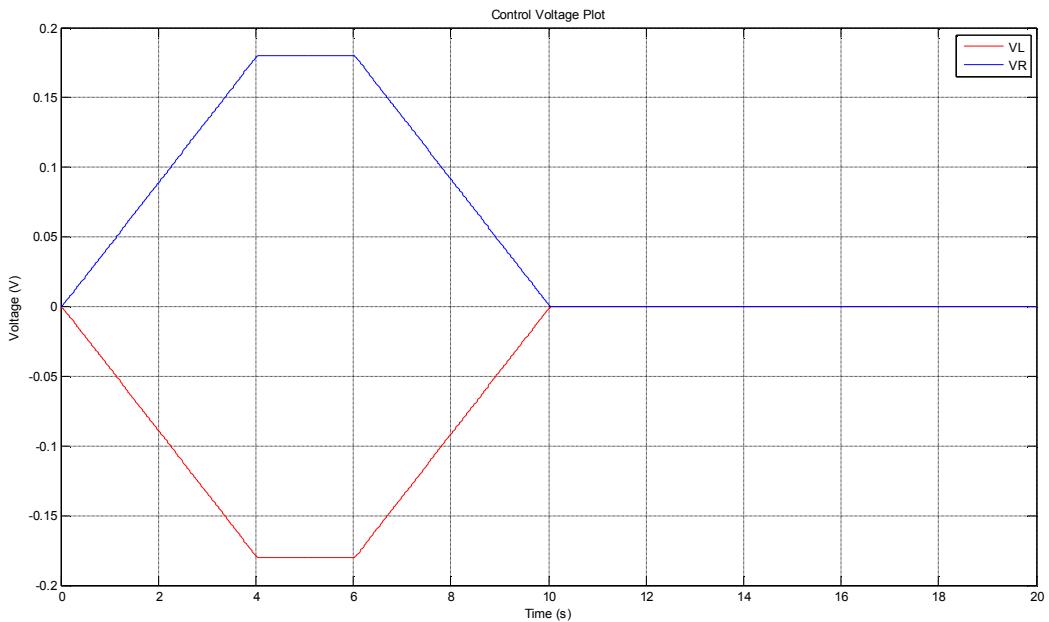


Hình 4.10 - Tín hiệu điều khiển động cơ trái (VL) và phải (VR) ứng với tín hiệu đặt vị trí.

c. Đáp ứng với tín hiệu đặt góc xoay



Hình 4.11 - Đáp ứng ngõ ra góc xoay ($\text{góc } \phi$) ứng với tín hiệu đặt góc xoay



Hình 4.12 - Tín hiệu điều khiển động cơ trái (VL) và phải (VR) ứng với tín hiệu đặt góc xoay

4.1.6 Điều khiển dùng LQR PI cho khâu vị trí

Để giảm sai số xác lập của khâu điều khiển vị trí, thêm vào khâu tích phân vị trí cho hệ thống, khi đó, phương trình trạng thái mới của khâu giữ cân bằng như sau:

$$A_{tp} = \begin{bmatrix} A_1 & 0 \\ C_1(1,:) & 0 \end{bmatrix} \text{ và } B_{tp} = \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \quad [4.19]$$

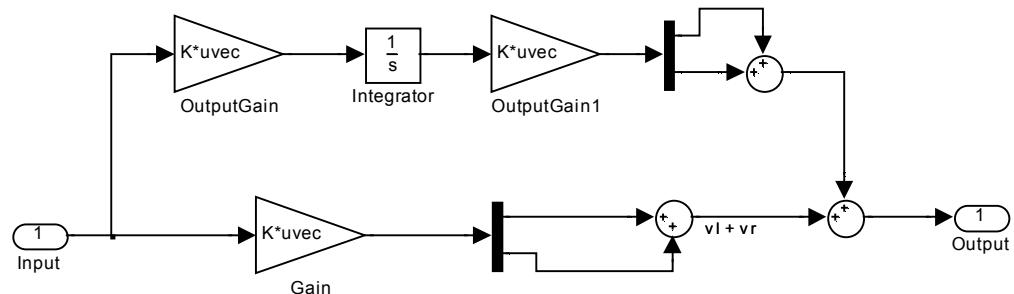
Chọn các trọng số của ma trận Q cho khâu giữ cân bằng như sau (ma trận R không đổi):

$$Q_{tp} = \begin{bmatrix} 1e3 & 0 & 0 & 0 & 0 \\ 0 & 1e1 & 0 & 0 & 0 \\ 0 & 0 & 1e4 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1e-4 \end{bmatrix} \quad [4.20]$$

Xác định được vector hồi tiếp:

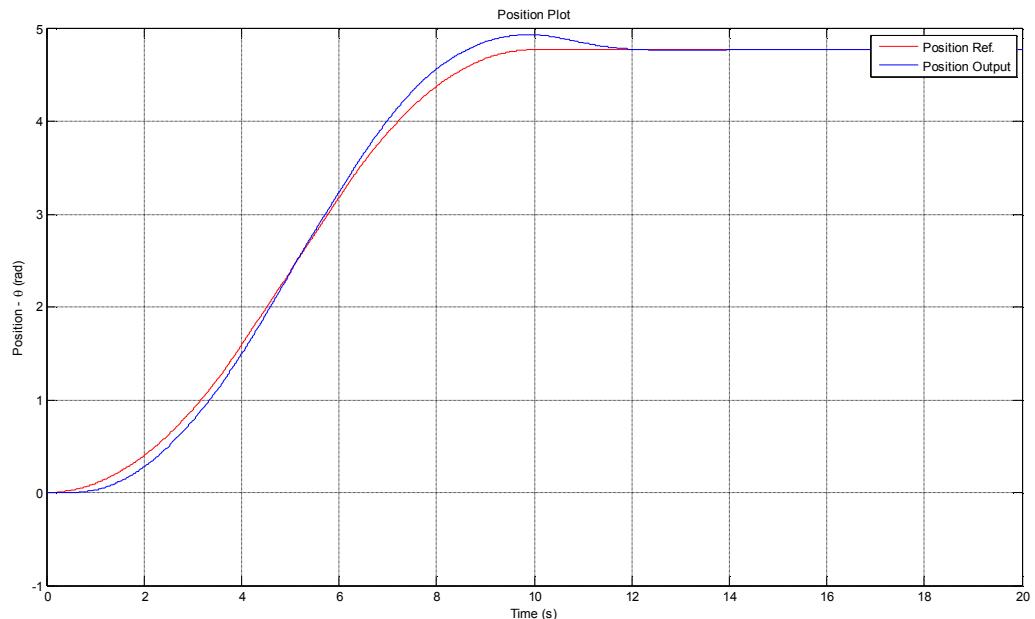
$$K_{tp} = \begin{bmatrix} -2.2369 & -2.8407 & -702.2339 & -95.7057 & -0.0007 \\ -2.2369 & -2.8407 & -702.2339 & -95.7057 & -0.0007 \end{bmatrix} \quad [4.21]$$

Sơ đồ simulink mô phỏng trong khối Balance Control như sau:



Hình 4.13 - Khối “Balance Control” điều khiển dùng LQR PI

4.1.7 Kết quả mô phỏng LQR PI



Hình 4.14 - Đáp ứng ngõ ra vị trí theo tín hiệu đặt vị trí

4.1.8 Kết luận

Chất lượng đáp ứng giữa bộ điều khiển LQR và LQR có thêm khâu tích phân vị trí là gần như nhau, tuy nhiên, đáp ứng vị trí của hệ thống khi có khâu tích phân tốt hơn, sai số nhỏ hơn.

4.2 Phương pháp điều khiển PID thích nghi mô hình tham chiếu

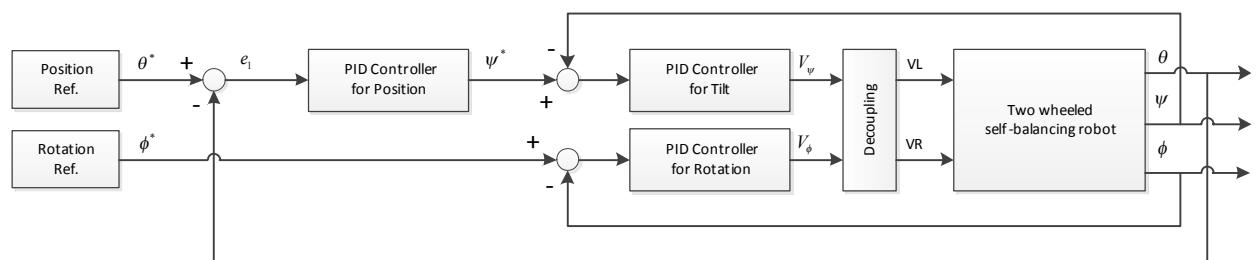
4.2.1 Đặt vấn đề

Các bộ điều khiển PID được sử dụng rất nhiều để điều khiển robot hai bánh tự cân bằng. Tuy nhiên, khi điều kiện làm việc thay đổi (như hệ số ma sát trên địa hình di chuyển khác nhau, độ dốc của địa hình robot đang di chuyển...) hay thông số của hệ thống thay đổi (như khối lượng robot, vị trí trọng tâm robot...) thì thông số bộ điều khiển PID cần được thay đổi cho phù hợp để cải thiện đáp ứng của hệ thống. Trong phạm vi của tiểu luận này, sự **thay đổi hệ số ma sát giữa bánh xe và mặt đường** được sử dụng làm ví dụ điển hình về sự thay đổi thông số trong đối tượng cần điều khiển.

4.2.2 Cấu trúc bộ điều khiển PID cho robot hai bánh tự cân bằng

Ở đây, ba bộ PID được sử dụng để điều khiển robot hai bánh tự cân bằng, bao gồm:

- Bộ PID điều khiển góc nghiêng (ψ)
- Bộ PID điều khiển vận vị trí (θ)
- Bộ PID điều khiển góc xoay (ϕ)



Hình 4.15 - Cấu trúc bộ điều khiển PID cho hệ robot hai bánh tự cân bằng

Hàm truyền đạt bộ điều khiển PID liên tục:

$$G_{PID}(s) = \frac{U(s)}{E(s)} = K_P + \frac{K_I}{s} + K_D \left(\frac{s}{1 + \tau s} \right) \quad [4.22]$$

Rời rạc hóa bộ điều khiển bằng cách biến đổi từ miền s sang z, với:

$$s \Leftrightarrow \frac{2}{T} \left(\frac{1-z^{-1}}{1+z^{-1}} \right) \quad [4.23]$$

Tín hiệu điều khiển sau khi rời rạc có dạng như sau:

$$u(n) = K_P e(n) + K_I v(n) + K_D w(n) \quad [4.24]$$

Với $v(n)$ là lượng tích phân, và $w(n)$ là lượng vi phân, được tính như sau:

$$\frac{V(s)}{E(s)} = \frac{1}{s} \quad [4.25]$$

$$\frac{V(z)}{E(z)} = \frac{T}{2} \left(\frac{1+z^{-1}}{1-z^{-1}} \right) \Rightarrow v(n) = v(n-1) + \frac{T}{2} [e(n) + e(n-1)] \quad [4.26]$$

$$\frac{W(s)}{E(s)} = \frac{s}{1+\tau s} \quad [4.27]$$

$$\frac{W(z)}{E(z)} = \frac{2(1-z^{-1})}{(2\tau+T)-(2\tau-T)z^{-1}} \quad [4.28]$$

$$\text{Với: } \alpha_0 = 2\tau + T; \alpha_1 = 2\tau - T \text{ và } p(n) = \frac{\alpha_0}{2} w(n) \quad [4.29]$$

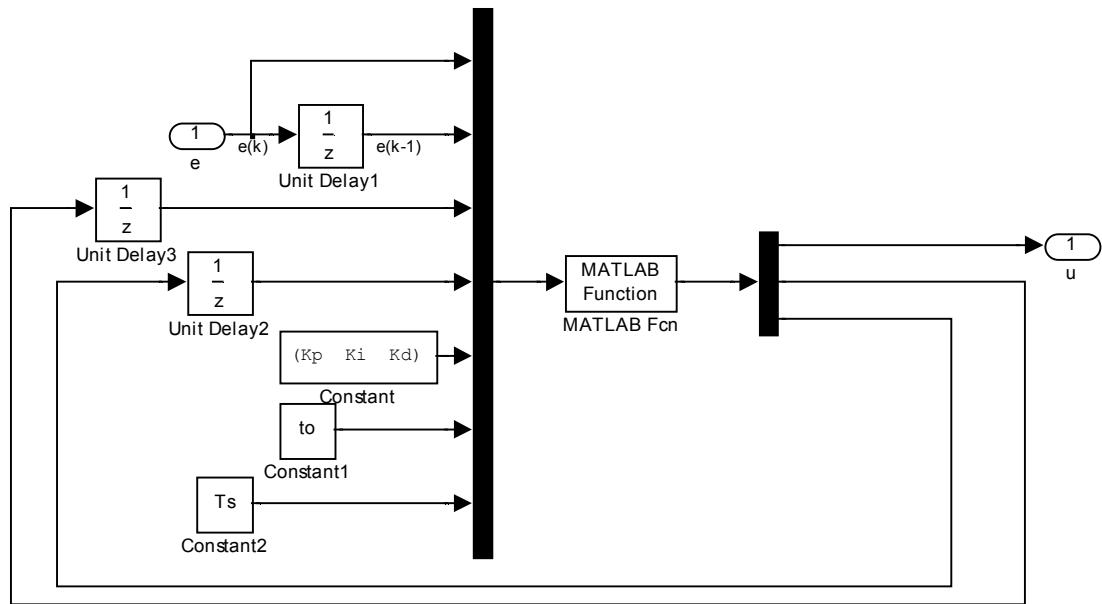
$$\Rightarrow \frac{\alpha_0}{2} w(n) = \frac{\alpha_1}{2} w(n-1) + e(n) - e(n-1) \Rightarrow p(n) = \frac{\alpha_1}{\alpha_0} p(n-1) + e(n) - e(n-1) \quad [4.30]$$

$$K_D w(n) = \left(\frac{2K_D}{\alpha_0} \right) p(n) \quad [4.31]$$

4.2.3 Bộ điều khiển PID với thông số cố định

Bộ điều khiển PID với thông số KP, KI, KD cố định như trên, hệ thống chỉ làm việc tốt trong điều kiện hệ số KP, KI, KD đã được chỉnh định tối ưu và trong quá trình làm việc, các thông số trong mô hình không đổi.

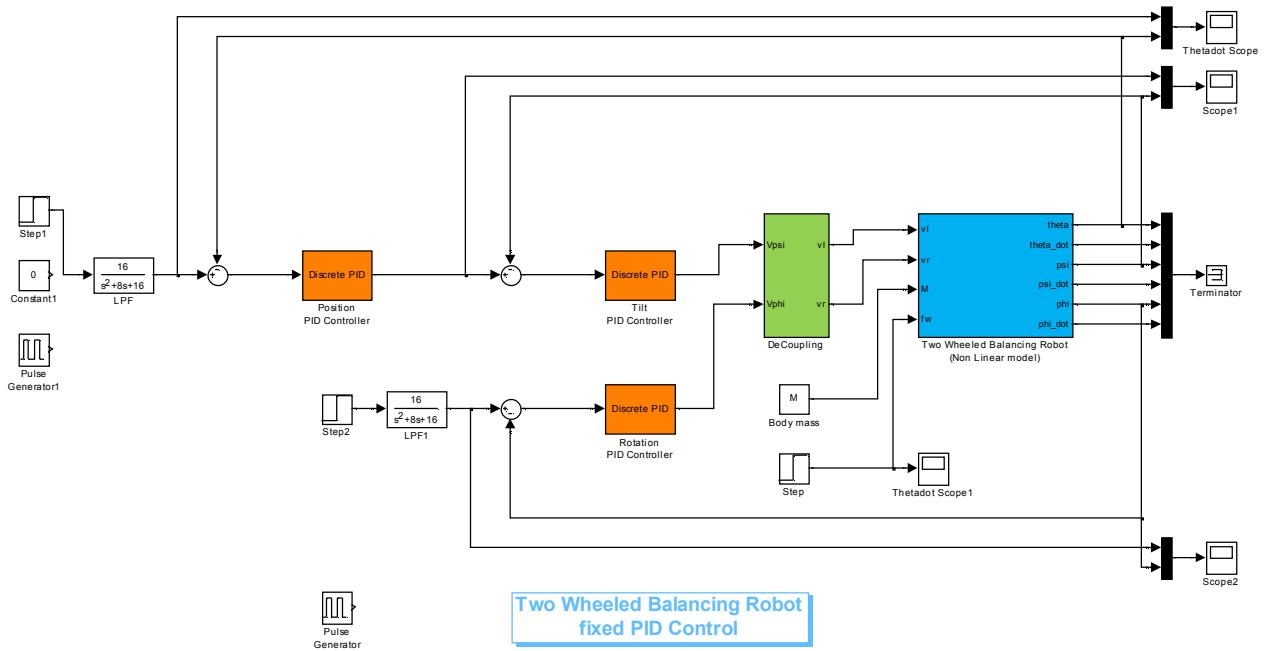
Cấu trúc bộ điều khiển PID rời rạc với hệ số cố định sử dụng để điều khiển góc nghiêng, vị trí và góc xoay là như nhau và được hiện thực như sau trong Matlab Simulink.



Hình 4.16 - Cấu trúc bên trong bộ điều khiển PID rời rạc với thông số cố định.

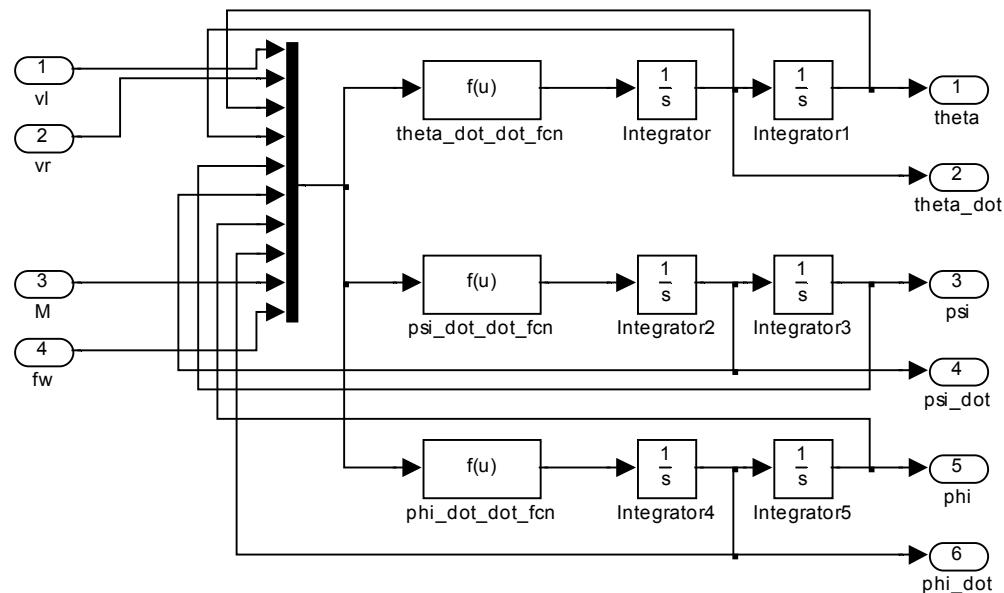
Khối hàm trong khối “MATLAB Fcn” gọi hàm “fixedDiscretePID_fcn_new” thực hiện trong m-file “fixedDiscretePID_fcn_new.m”

Hệ thống điều khiển robot hai bánh cân bằng sử dụng bộ PID rời rạc với thông số cố định hiện thực như sau:



Hình 4.17 - Robot hai bánh tự cân bằng sử dụng 3 bộ điều khiển PID cố định

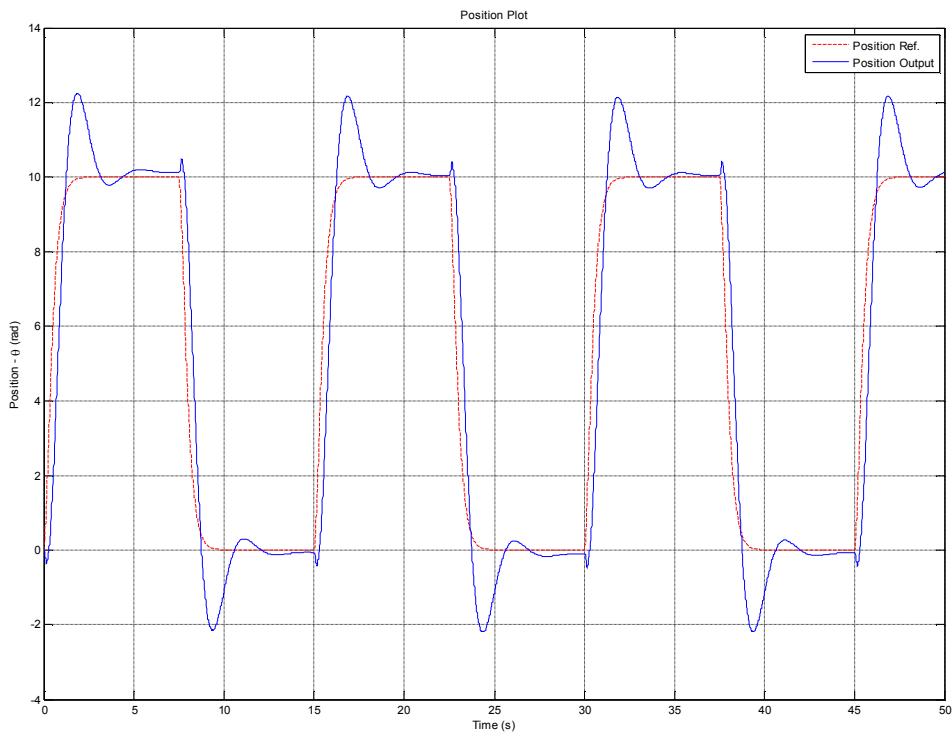
Trong đó, mô hình robot hai bánh tự cân bằng trong trường hợp này được bổ sung thêm 2 ngõ vào là khối lượng thân robot (M) và hệ số ma sát giữa bánh xe và mặt phẳng di chuyển (fw).



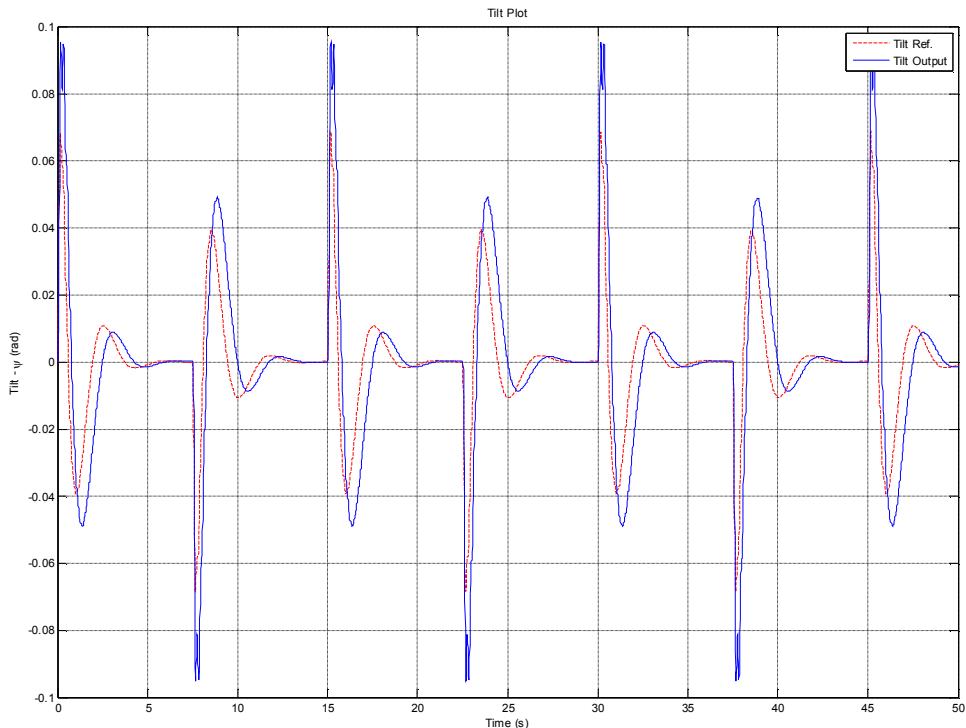
Hình 4.18 - Mô hình robot hai bánh tự cân bằng đã bổ sung
2 tín hiệu vào là M và fw

a. Trường hợp thông số mô hình cố định

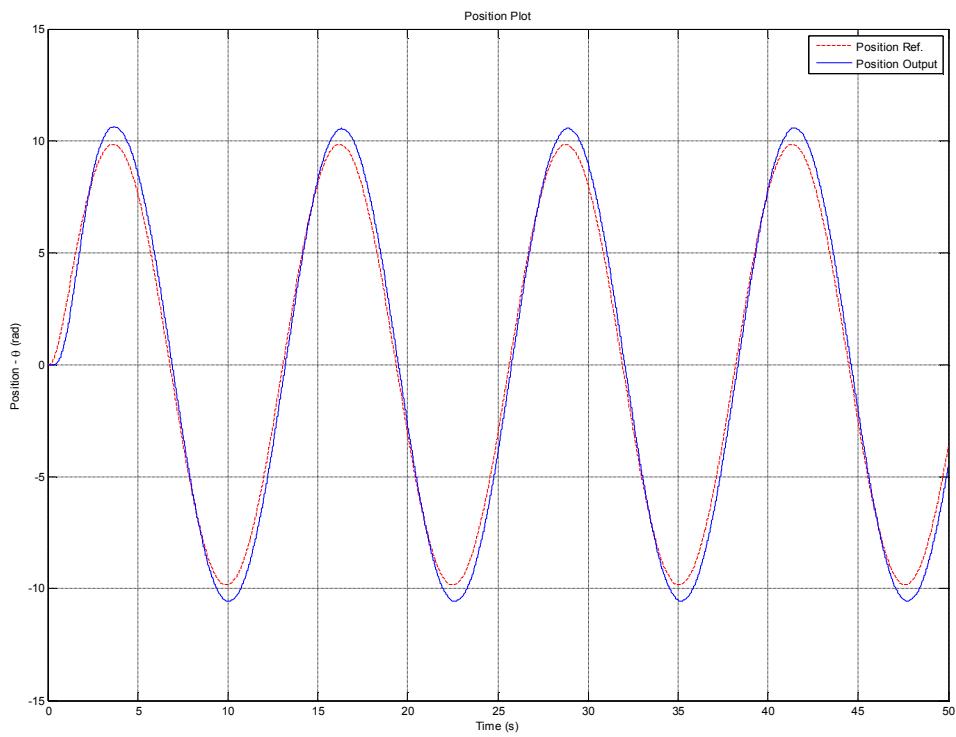
Trong trường hợp thông số M và fw cố định, bộ PID cố định sẽ điều khiển hệ thống hoạt động tốt ứng với các tín hiệu vào là xung vuông và sóng sine như sau:



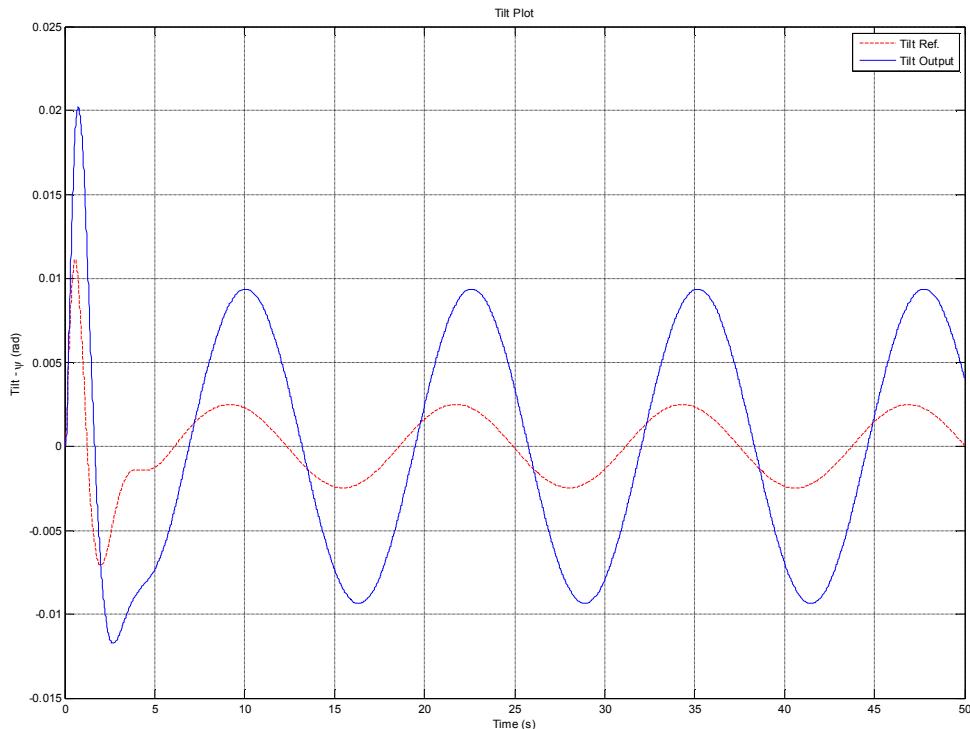
Hình 4.19 - Đáp ứng ngõ ra vị trí với tín hiệu đặt vị trí là xung vuông



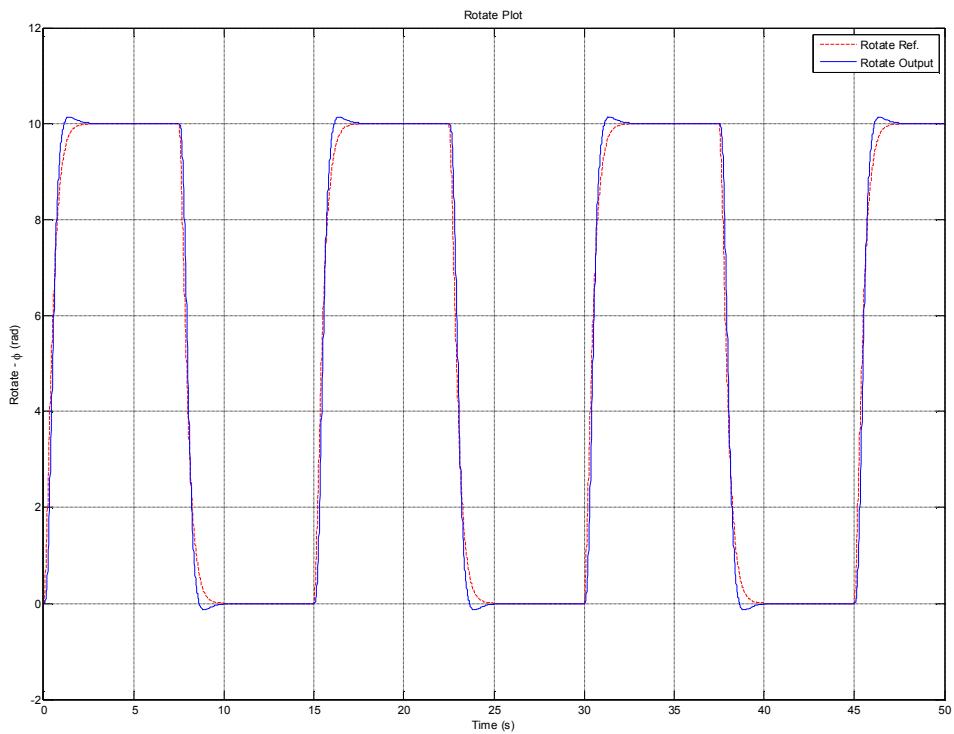
Hình 4.20 - Đáp ứng ngõ ra góc nghiêng với tín hiệu đặt vị trí là xung vuông



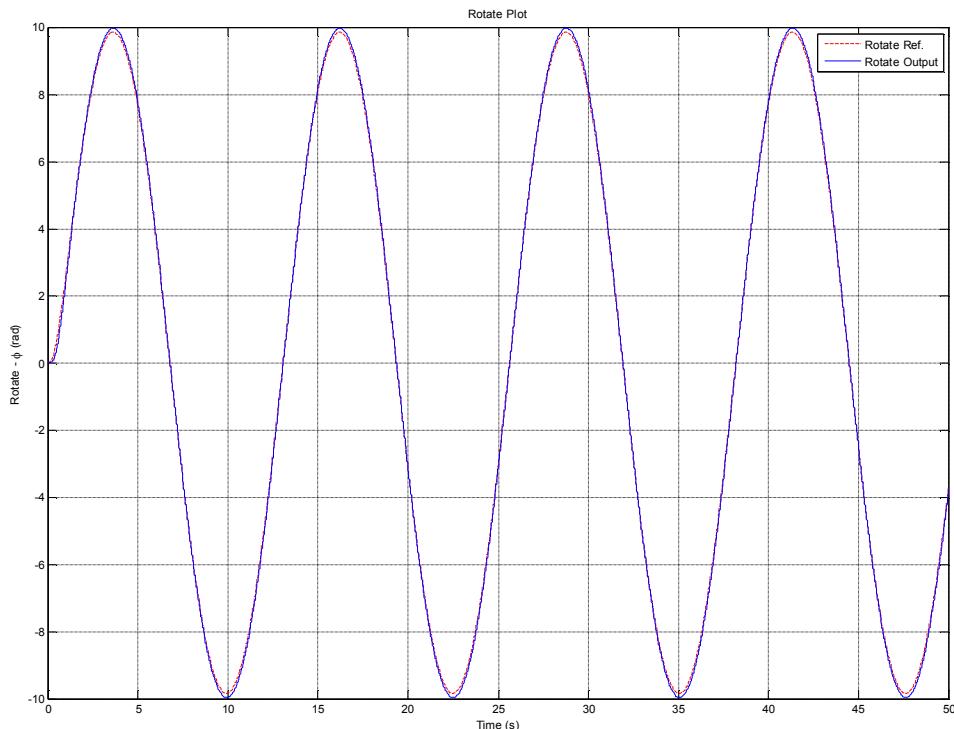
Hình 4.21 - Đáp ứng ngõ ra vị trí với tín hiệu đặt vị trí là sóng sin



Hình 4.22 - Đáp ứng ngõ ra góc nghiêng với tín hiệu đặt vị trí là sóng sin



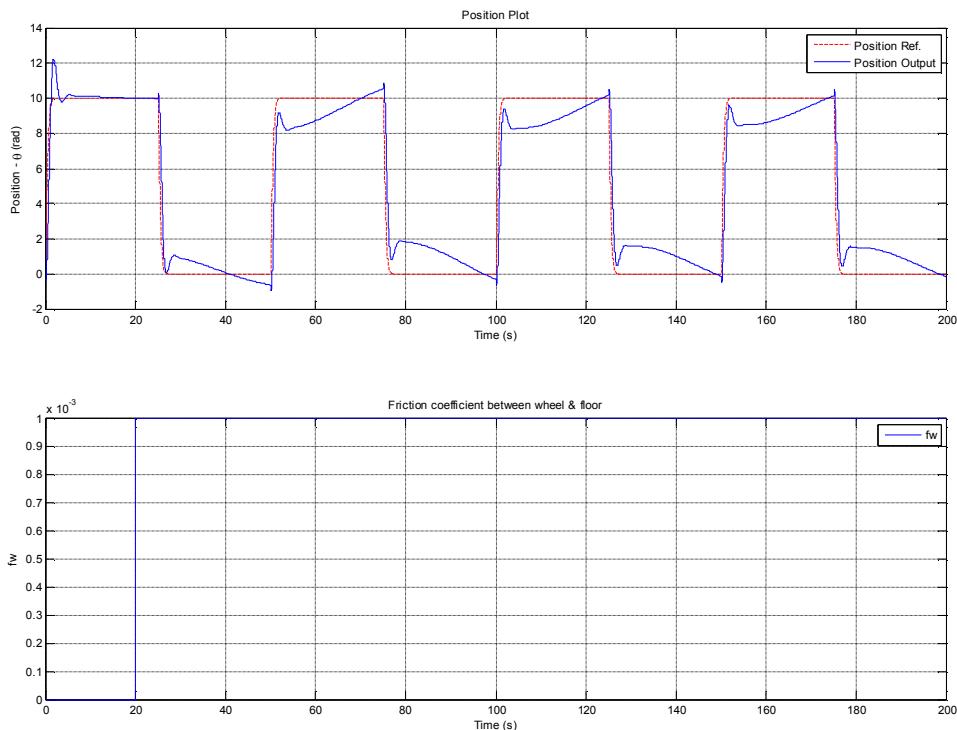
Hình 4.23 - Đáp ứng ngõ ra góc xoay với tín hiệu đặt góc xoay là sóng vuông



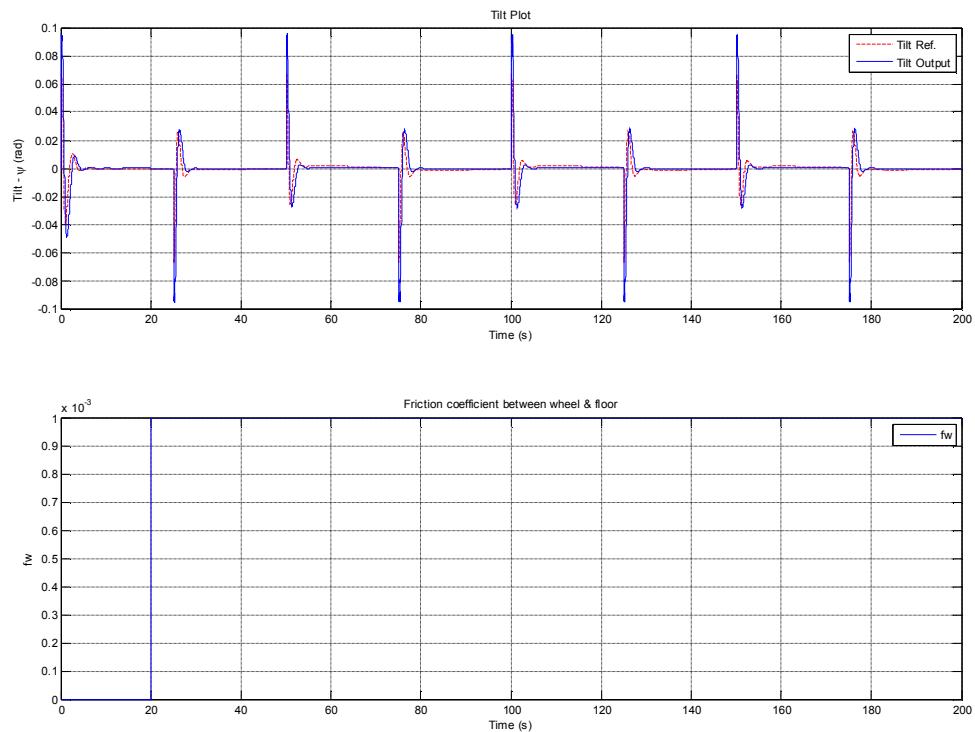
Hình 4.24 - Đáp ứng ngõ ra góc xoay với tín hiệu đặt góc xoay là sóng sin

b. Trường hợp có sự thay đổi thông số mô hình

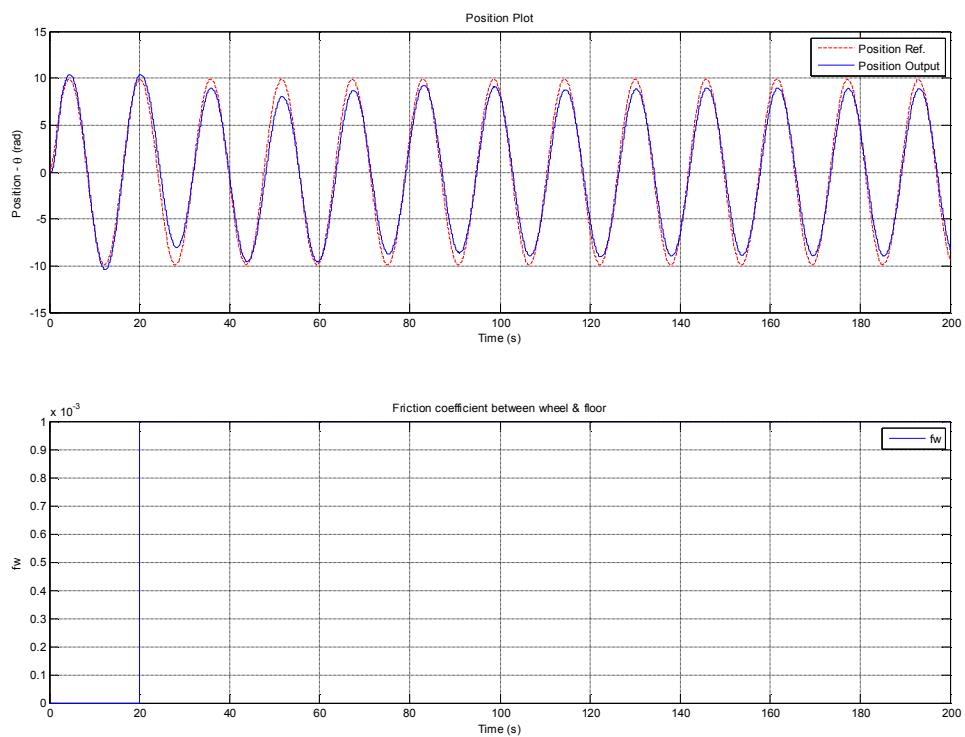
Giả sử hệ số ma sát fw giữa bánh xe và mặt phẳng di chuyển có sự thay đổi trong phạm vi $[0 \quad 0.001]$, trong trường hợp này, lấy ví dụ hệ số fw ở thời điểm 20s thay đổi từ 0 lên 0.001 thì đáp ứng ngõ ra của hệ thống như sau:



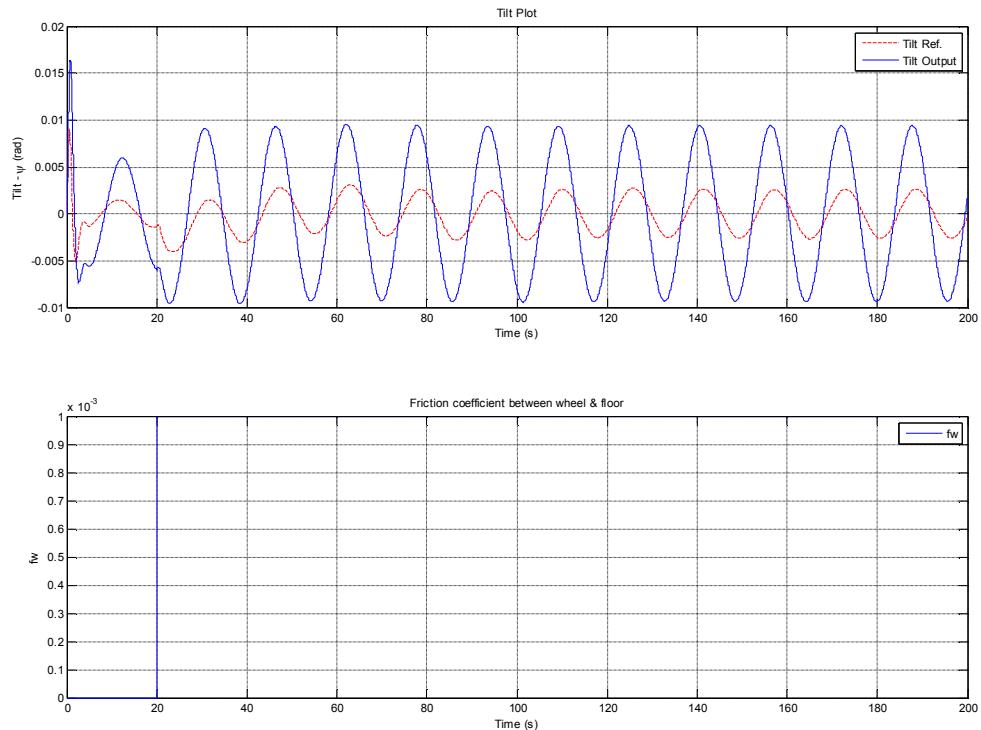
Hình 4.25 - Đáp ứng ngõ ra vị trí trong trường hợp tín hiệu đặt vị trí là xung vuông



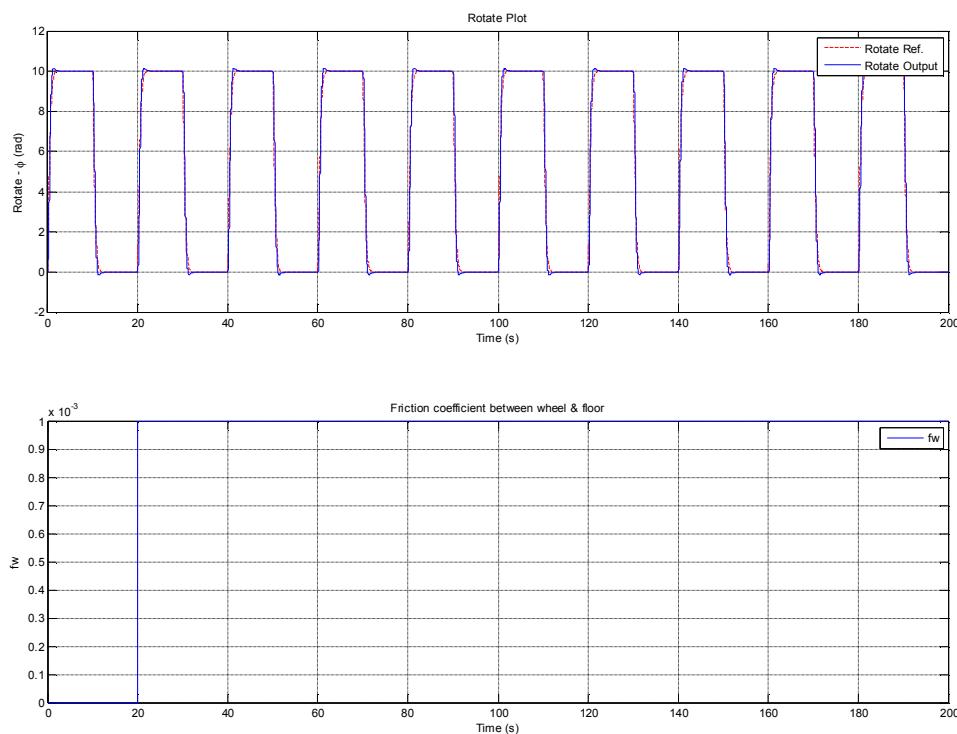
Hình 4.26 - Đáp ứng ngõ ra góc nghiêng trong trường hợp tín hiệu đặt vị trí là xung vuông



Hình 4.27 - Đáp ứng ngõ ra vị trí trong trường hợp tín hiệu đặt vị trí là sóng sine



Hình 4.28 - Đáp ứng ngõ ra góc nghiêng trong trường hợp tín hiệu đặt vị trí
là sóng sine



Hình 4.29 - Đáp ứng ngõ ra góc xoay trong trường hợp tín hiệu đặt góc xoay là xung vuông

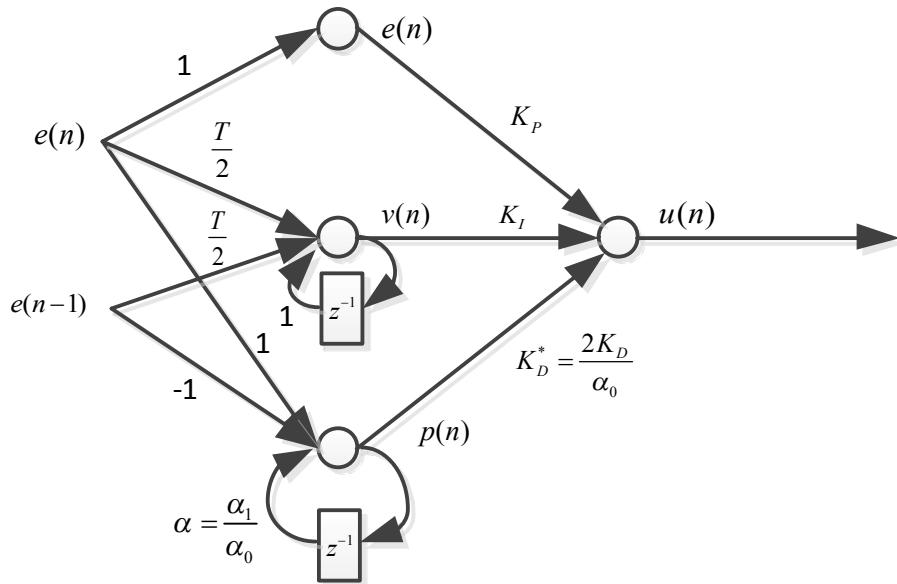
Có thể thấy, đáp ứng ngõ ra vị trí của hệ thống bị ảnh hưởng một cách rõ rệt, ngõ ra vị trí của hệ thống không thể tiếp tục đáp ứng theo tín hiệu đặt như ban đầu kể từ thời điểm hệ số fw thay đổi. Còn các bộ PID điều khiển góc nghiêng thì vẫn đảm bảo ngõ ra hệ thống bám tốt theo tín hiệu đặt.

Như vậy trong trường hợp có sự thay đổi hệ số ma sát fw trong mô hình thì bộ PID cố định điều khiển vị trí không thể đảm bảo được chất lượng điều khiển. Phần tiếp theo sẽ trình bày cách thiết kế bộ PID thích nghi cho khâu điều khiển vị trí robot hai bánh tự cân bằng.

4.2.4 Bộ điều khiển PID thích nghi mô hình tham chiếu cho robot hai bánh tự cân bằng

a. Cấu trúc bộ điều khiển PID thích nghi

Cấu trúc bộ PID rời rạc có thể được biểu diễn dưới dạng cấu trúc mạng neuron bao gồm 1 lớp ẩn và 1 lớp ra như sau:



Hình 4.30 - Cấu trúc mạng neuron của bộ điều khiển PID rì rạc

Như vậy, để bộ điều khiển PID hoạt động tốt trong điều kiện thông số mô hình thay đổi thì các thông số trong bộ điều khiển PID cũng phải thay đổi theo phù hợp.

Quá trình thay đổi các thông số PID trong trường hợp này thực hiện thông qua quá trình cập nhật các hệ số K_p , K_i , K_d bằng cách lan truyền ngược sai số tương tự như quá trình huấn luyện mạng neuron. Việc cập nhật hệ số K_p , K_i , K_d trong trường hợp này chính là cập nhật các trọng số ở lớp ngõ ra; còn các trọng số ngõ vào được giữ cố định.

b. Giải thuật cập nhật các thông số cho bộ điều khiển PID vị trí

Ngõ ra của mạng neuron được tính như sau:

$$e(n) = O_1(n) \quad [4.32]$$

$$v(n) = O_2(n) = O_2(n-1) + \frac{T}{2} [e(n) + e(n-1)] \quad [4.33]$$

$$p(n) = O_3(n) = \alpha O_3(n-1) + e(n) - e(n-1) \quad [4.34]$$

$$u(n) = K_p O_1(n) + K_i O_2(n) + K_d^* O_3(n) \quad [4.35]$$

$$\text{Với: } \alpha = \frac{\alpha_1}{\alpha_0}, K_d^* = \frac{2K_d}{\alpha_0} \quad [4.36]$$

Giải thuật cập nhật được thực hiện trên cơ sở tối ưu hóa hàm mục tiêu và hàm mục tiêu chính là sai số giữa tín hiệu mong muốn và tín hiệu đáp ứng của robot.

Tín hiệu mong muốn ở đây là tín hiệu ngõ ra của mô hình tham chiếu bậc 2 ứng với tín hiệu đặt vận tốc. Mô hình bậc 2 được chọn theo đặc tính đáp ứng như thời gian tăng trưởng, sai số xác lập, độ vọt lồ, đặc tính dao động...

Tín hiệu ngõ ra là ngõ ra vị trí của robot.

Như vậy, hàm mục tiêu cho bộ điều khiển PID điều khiển vị trí:

$$I(n) = \frac{1}{2} e_2^2(n) = \frac{1}{2} (\theta_{mref}(n) - \theta(n))^2 \quad [4.37]$$

Thông số K_P, K_I, K_D của bộ điều khiển được cập nhật online thông qua phương pháp Gradient biến thiên (thuật toán suy giảm độ dốc) và luật cập nhật được thực hiện như sau:

$$\begin{aligned} K_P(n+1) &= K_P(n) - \eta_{KP} \frac{\partial I(n)}{\partial K_P(n)} \\ K_I(n+1) &= K_I(n) - \eta_{KI} \frac{\partial I(n)}{\partial K_I(n)} \\ K_D(n+1) &= K_D(n) - \eta_{KD} \frac{\partial I(n)}{\partial K_D(n)} \end{aligned} \quad [4.38]$$

$\eta_{KP}, \eta_{KI}, \eta_{KD}$ là tốc độ cập nhật.

Ta có thể tính được:

$$\begin{aligned} \frac{\partial I(n)}{\partial K_P(n)} &= \frac{\partial I(n)}{\partial \theta(n)} \frac{\partial \theta(n)}{\psi^*(n)} \frac{\psi^*(n)}{\partial K_P(n)} \\ \frac{\partial I(n)}{\partial K_I(n)} &= \frac{\partial I(n)}{\partial \theta(n)} \frac{\partial \theta(n)}{\psi^*(n)} \frac{\psi^*(n)}{\partial K_I(n)} \\ \frac{\partial I(n)}{\partial K_D(n)} &= \frac{\partial I(n)}{\partial \theta(n)} \frac{\partial \theta(n)}{\psi^*(n)} \frac{\psi^*(n)}{\partial K_D(n)} \end{aligned} \quad [4.39]$$

Trong trường hợp là bộ PID điều khiển vị trí, ngõ ra bộ PID $u(n)$ chính là $\psi^*(n)$

Suy ra:

$$\begin{aligned}\frac{\partial I(n)}{\partial K_P(n)} &= -e_2(n) J_P(n) O_1(n) \\ \frac{\partial I(n)}{\partial K_I(n)} &= -e_2(n) J_P(n) O_2(n) \\ \frac{\partial I(n)}{\partial K_D(n)} &= -e_2(n) J_P(n) O_3(n)\end{aligned}\quad [4.40]$$

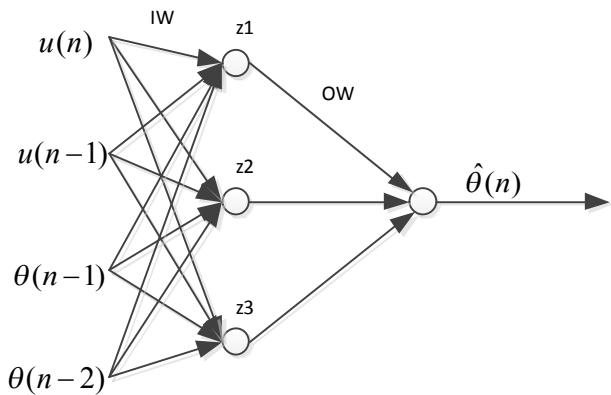
Trong biểu thức trên, giá trị của $J_P(n) = \frac{\partial \theta(n)}{u(n)}$ (Jacobian biểu diễn quan hệ giữa ngõ vào $\psi^*(n)$ và ngõ ra $\theta(n)$ của hệ thống) không thể tính tường minh, do không thể xác định được giá trị chính xác của mô hình robot. Do vậy, trong trường hợp chu kì lấy mẫu đủ nhỏ, $\frac{\partial \theta(n)}{u(n)}$ được xấp xỉ bằng cách:

$$J_P = \frac{\partial \theta(n)}{u(n)} \approx \frac{\Delta \theta(n)}{\Delta u(n)} = \frac{\theta(n) - \theta(n-1)}{u(n) - u(n-1)} \quad [4.41]$$

Thông thường, cách xấp xỉ như trên không hoàn toàn chính xác. Để tránh điều này, một mạng neuron được sử dụng để nhận dạng online đặc tính vào/ra của đối tượng (trong trường hợp này, cần nhận dạng quan hệ giữa ngõ vào $u(n)$ hay $\psi^*(n)$ và ngõ ra $\theta(n)$ của robot); từ đó biểu thức J_P được xấp xỉ bằng $\frac{\partial \hat{\theta}(n)}{u(n)}$, trong đó, $\hat{\theta}(n)$ là ngõ ra xấp xỉ của mạng neuron nhận dạng đối tượng.

Mạng neuron nhận dạng được xây dựng từ vector hồi quy gồm 4 ngõ vào là $[u(n) \ u(n-1) \ \theta(n-1) \ \theta(n-2)]^T$, 1 lớp ẩn gồm 3 neuron và 1 lớp ra, hàm kích hoạt là tuyến tính, các trọng số trong mạng neuron được cập nhật online mỗi chu kì lấy mẫu thông qua giải thuật lan truyền ngược.

Cấu trúc mạng neuron nhận dạng đối tượng như sau:



Hình 4.31 - Cấu trúc mạng neuron nhận dạng đối tượng

Trong đó, ma trận trọng số ngõ vào là IW, OW là ma trận trọng số ngõ ra.

Quy ước:

$$IW(n) = \begin{bmatrix} IW_{11}(n) & IW_{12}(n) & IW_{13}(n) & IW_{14}(n) \\ IW_{21}(n) & IW_{22}(n) & IW_{23}(n) & IW_{24}(n) \\ IW_{31}(n) & IW_{32}(n) & IW_{33}(n) & IW_{34}(n) \end{bmatrix} \quad [4.42]$$

$$OW(n) = [OW_1(n) \quad OW_2(n) \quad OW_3(n)] \quad [4.43]$$

Vector ngõ vào:

$$X(n) = [u(n) \quad u(n-1) \quad \theta(n-1) \quad \theta(n-2)]^T \quad [4.44]$$

Vector ngõ ra lớp ẩn:

$$Z(n) = [z_1(n) \quad z_2(n) \quad z_3(n)]^T \quad [4.45]$$

$$Z(n) = IW(n)X(n) \quad [4.46]$$

Ngõ ra mạng neuron:

$$\hat{\theta}(n) = OW(n)Z(n) \quad [4.47]$$

Hàm mục tiêu huấn luyện mạng neuron nhận dạng:

$$I_I(n) = \frac{1}{2}e_3^2(n) = \frac{1}{2}[\theta(n) - \hat{\theta}(n)]^2 \quad [4.48]$$

Tương tự, ma trận trọng số được cập nhật theo phương pháp gradient biến thiên, kết quả luật cập nhật như sau:

$$OW(n+1) = OW(n) + \eta e_3(n) Z^T(n) \quad [4.49]$$

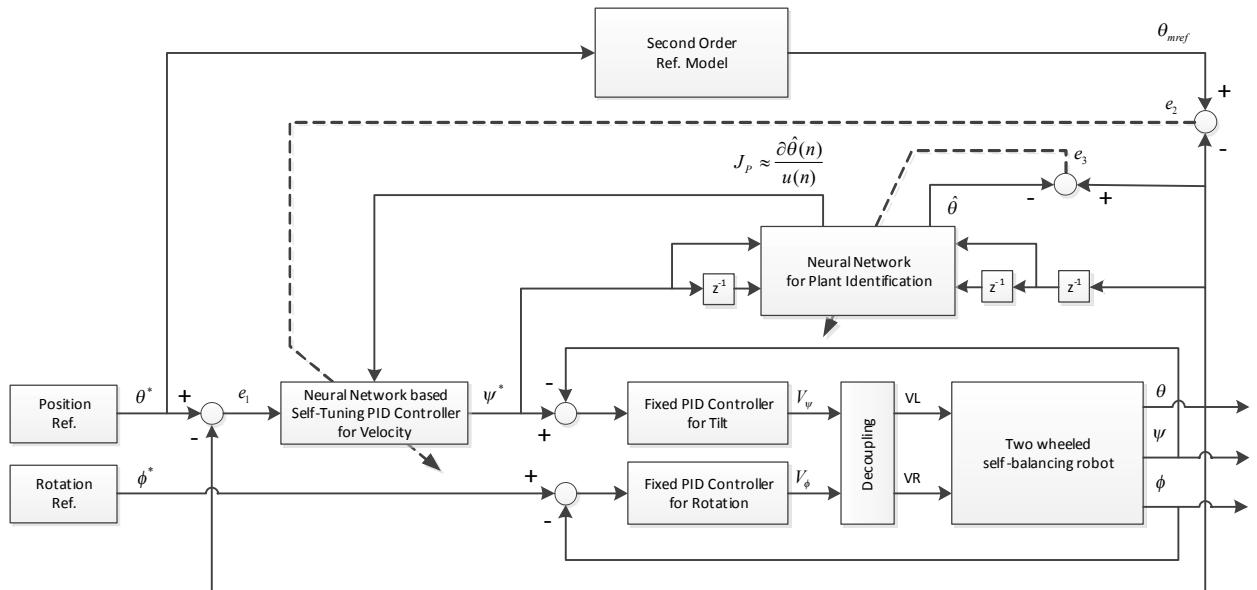
$$IW(n+1) = IW(n) + \eta e_3(n) OW^T(n) X^T(n) \quad [4.50]$$

Khi đó, lượng J_p được tính xấp xỉ như sau:

$$J_p(n) \approx \frac{\partial \hat{\theta}(n)}{u(n)} = OW_1(n)IW_{11}(n) + OW_2(n)IW_{21}(n) + OW_3(n)IW_{31}(n) \quad [4.51]$$

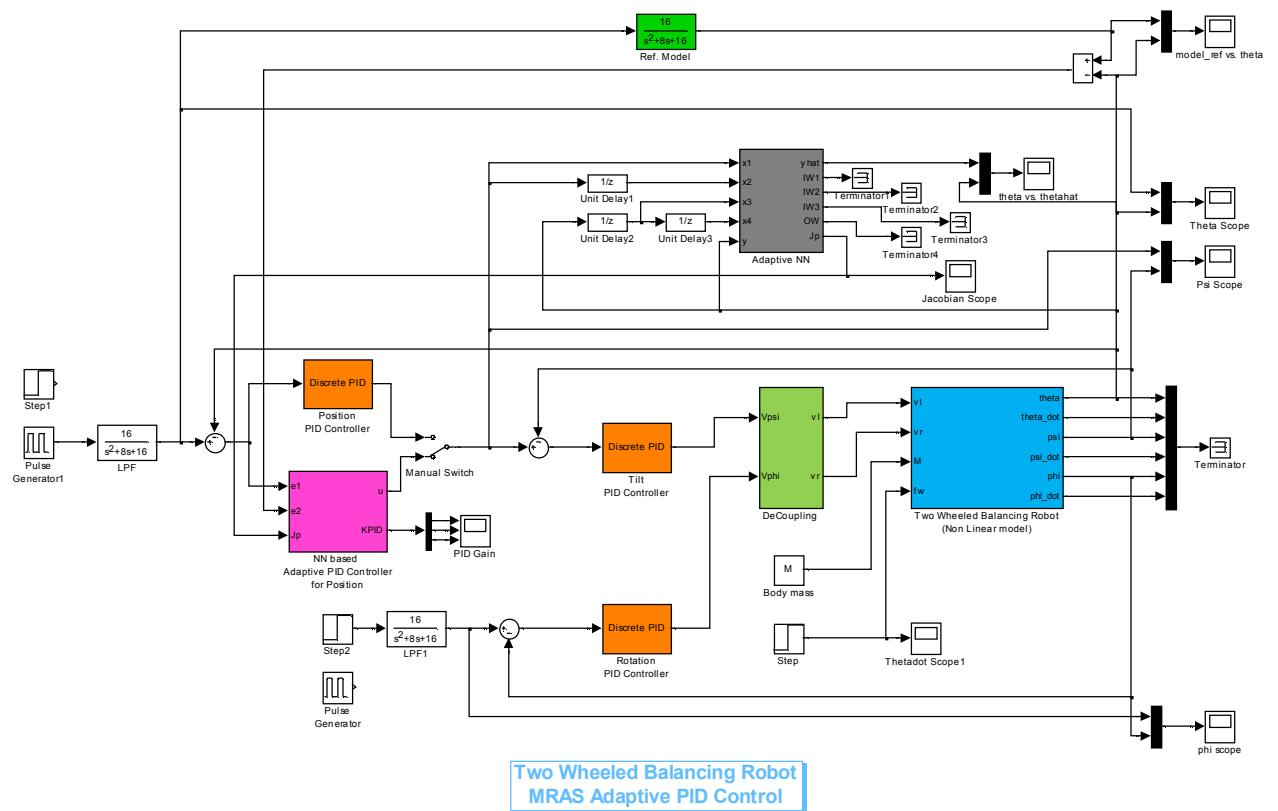
c. Hiện thực hóa bộ điều khiển PID thích nghi cho khâu vị trí

Sơ đồ điều khiển tổng quát của hệ thống được thiết kế như sau:



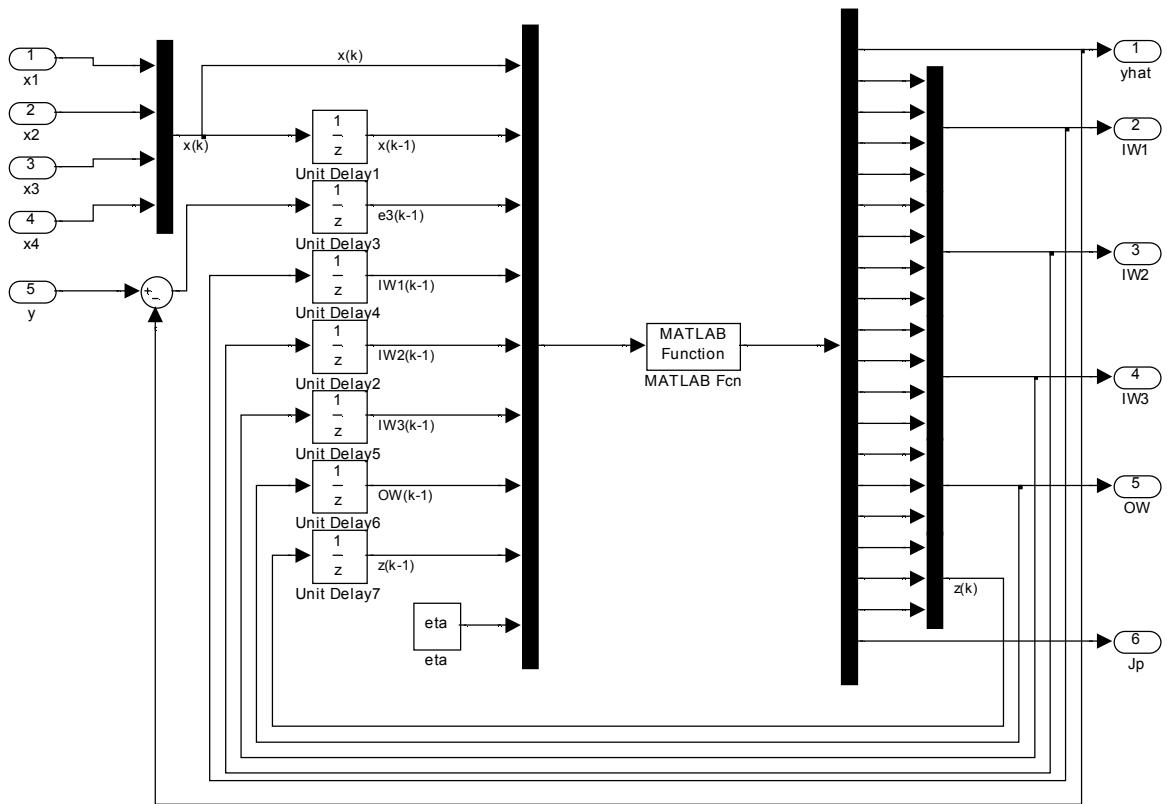
Hình 4.32 - Tổng quát bộ điều khiển PID thích nghi mô hình tham chiếu cho khâu vị trí dựa trên cấu trúc mạng neuron

Hệ thống trên được hiện thực trong Matlab Simulink như sau:



Hình 4.33 - Bộ điều khiển PID thích nghi mô hình tham chiếu cho khâu vị trí
dựa trên cấu trúc mạng neuron trong Matlab Simulink

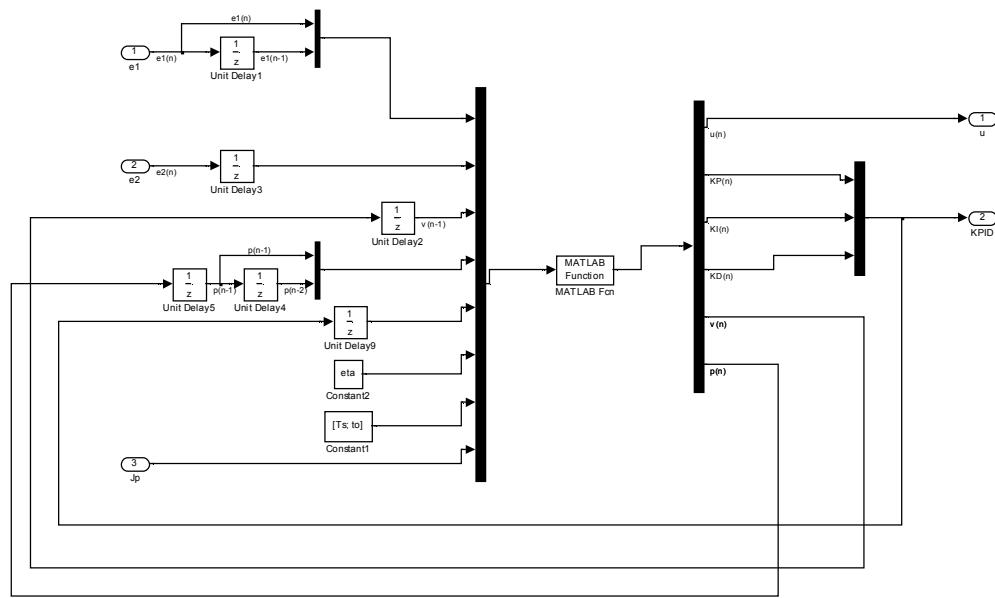
Khối “Adaptive NN” là mạng neuron nhận dạng đặc tính hệ thống có quan hệ vào-rõa là ψ^* và θ để xấp xỉ lượng J_p



Hình 4.34 - Khối “Adaptive NN”

Khối “MATLAB Fcn” gọi hàm viết trong m-file “adaptiveNNwithoutBias_fcn.m”

Khối “NN based Adaptive PID Controller for Position” là khối PID tự chỉnh cho khâu vị trí, sao cho sai lệch giữa ngõ ra mô hình tham chiếu bậc hai θ_{mref} và ngõ ra vị trí của robot θ tối thiểu.

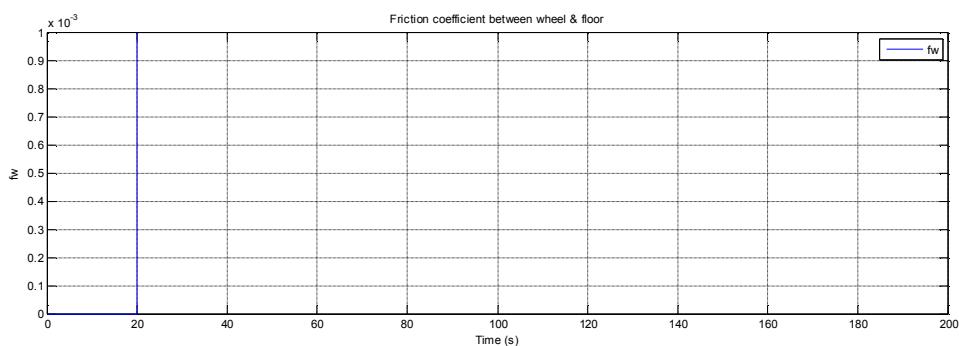
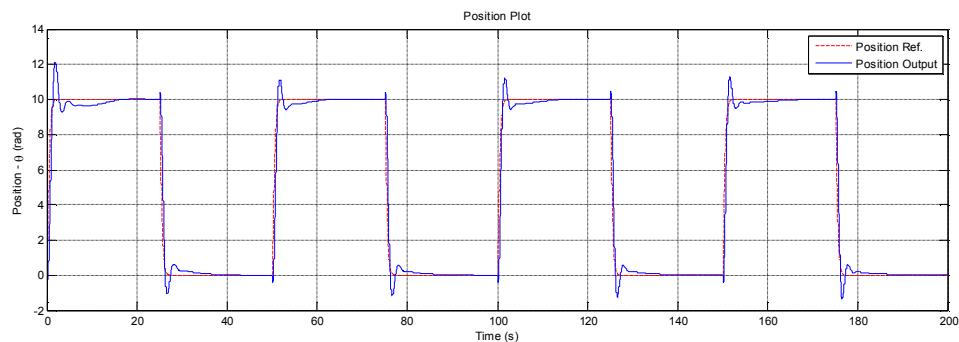


Hình 4.1 Khối “NN based Adaptive PID Controller for Position”

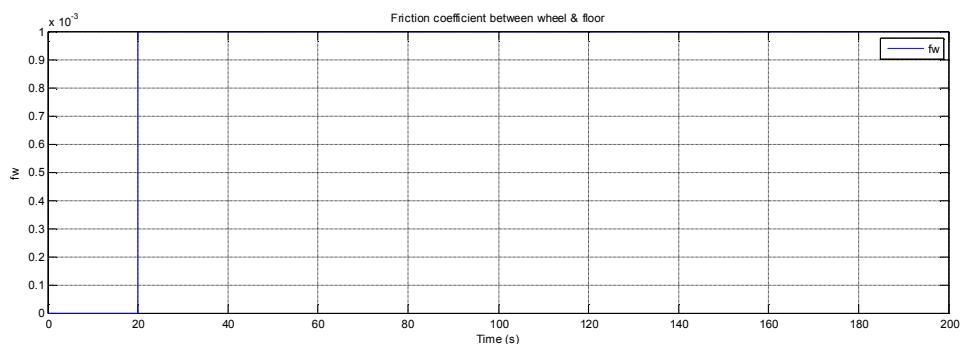
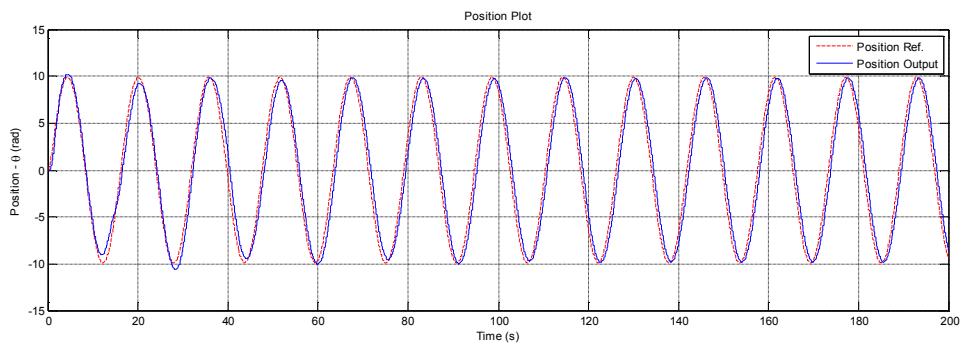
Khối “MATLAB Fcn” gọi hàm viết trong m-file “STPID_fcn_final.m”

b. Kết quả mô phỏng

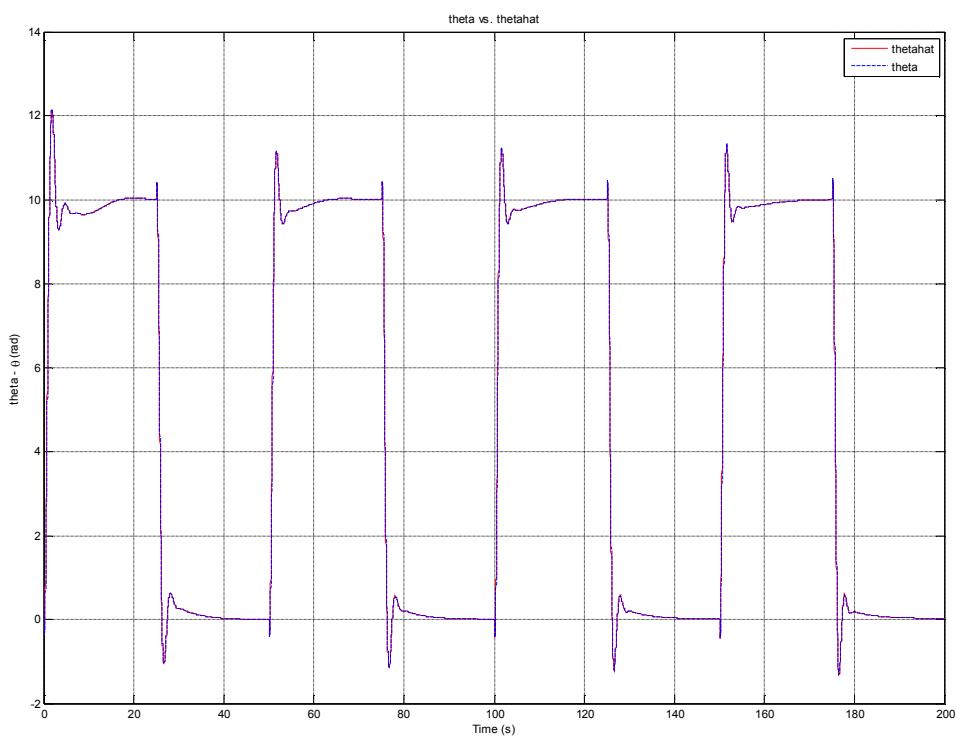
Kết quả điều khiển ứng với các tín hiệu đặt xung vuông, sóng sine như sau:



Hình 4.35 - Ngõ ra vị trí ứng với tín hiệu đặt vị trí là xung vuông

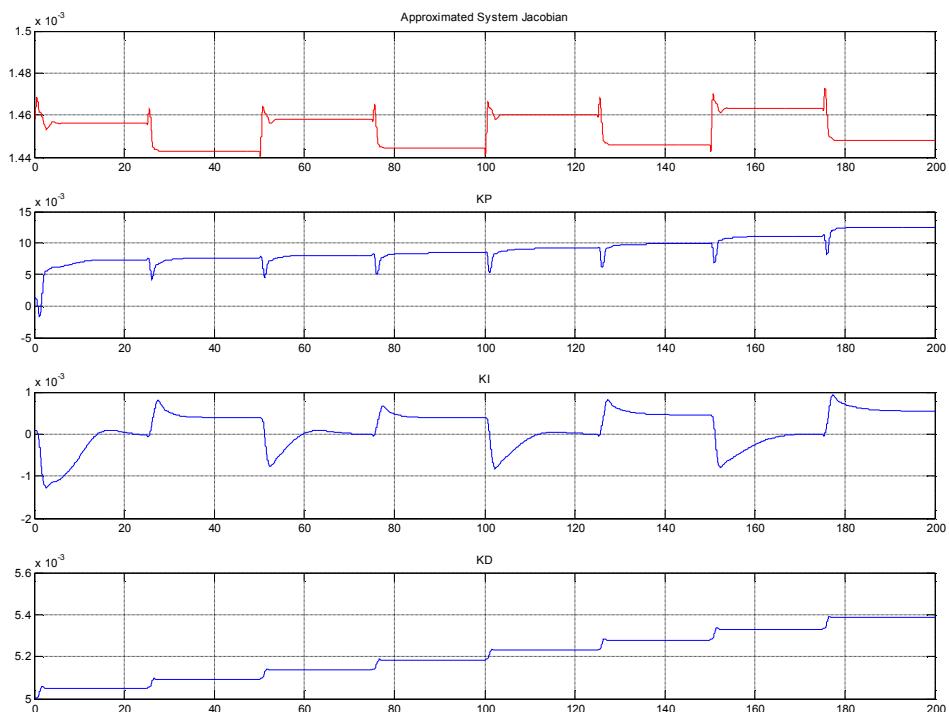


Hình 4.36 - Ngõ ra vị trí ứng với tín hiệu đặt vị trí là sóng sine



Hình 4.37 - Ngõ ra vị trí của robot và ngõ ra vị trí xấp xỉ của mạng neuron nhận dạng hệ thống

Ngõ ra vận tốc xấp xỉ của mạng neuron từ thời điểm fw thay đổi có sai lệch chút ít so với ngõ ra vận tốc của robot, nhưng sau đó nhanh chóng bám trở lại để tiến tới sai số xấp xỉ bằng 0.



Hình 4.38 - Jacobian xấp xỉ và các hệ số của bộ PID thích nghi mô hình tham chiếu cho khâu vị trí

4.2.5 Kết luận

Đáp ứng ngõ ra vị trí điều khiển bởi bộ điều khiển PID thích nghi mô hình tham chiếu đã được cải thiện tốt so với bộ điều khiển PID với thông số cố định. Tuy nhiên, việc chọn lựa các giá trị khởi tạo cho thông số bộ PID, tốc độ cập nhật các thông số là rất quan trọng, quyết định đến sự hội tụ các thông số và sự hoạt động ổn định của hệ thống, nếu tốc độ cập nhật quá nhanh, bộ điều khiển có thể đáp ứng nhanh chóng với sự thay đổi nhưng đồng thời cũng nhạy với nhiễu và rất dễ mất ổn định.

CHƯƠNG V

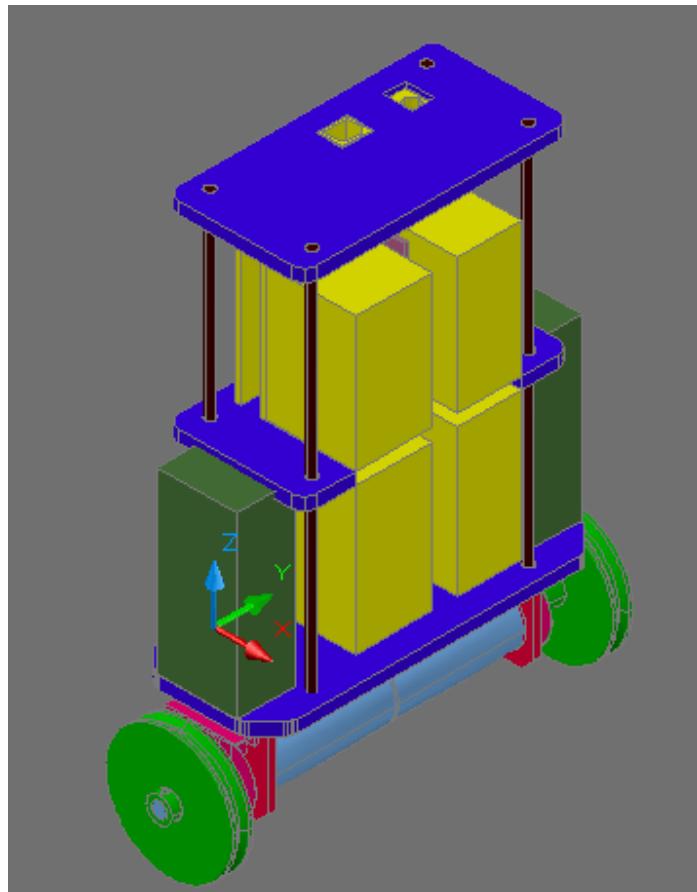
BỘ ĐIỀU KHIỂN NHÚNG ROBOT HAI BÁNH TỰ CÂN BẰNG

Chương này trình bày thiết kế mô hình cơ khí của robot hai bánh tự cân bằng, thành phần cảm biến sử dụng trên robot, cách thiết kế bộ lọc Kalman để xử lý tín hiệu nhận được từ các cảm biến. Thuật toán điều khiển bao gồm LQR PI và PID thích nghi mô hình tham chiếu được nhúng lên DSP TMS320F28335 của Texas Instrument. Chương này cũng trình bày kết quả điều khiển robot ứng với các giải thuật khác nhau trên địa hình phẳng hay dốc nghiêng đến 10 độ.

5.1 Thiết kế mô hình robot hai bánh tự cân bằng

5.1.1 Thiết kế cơ khí

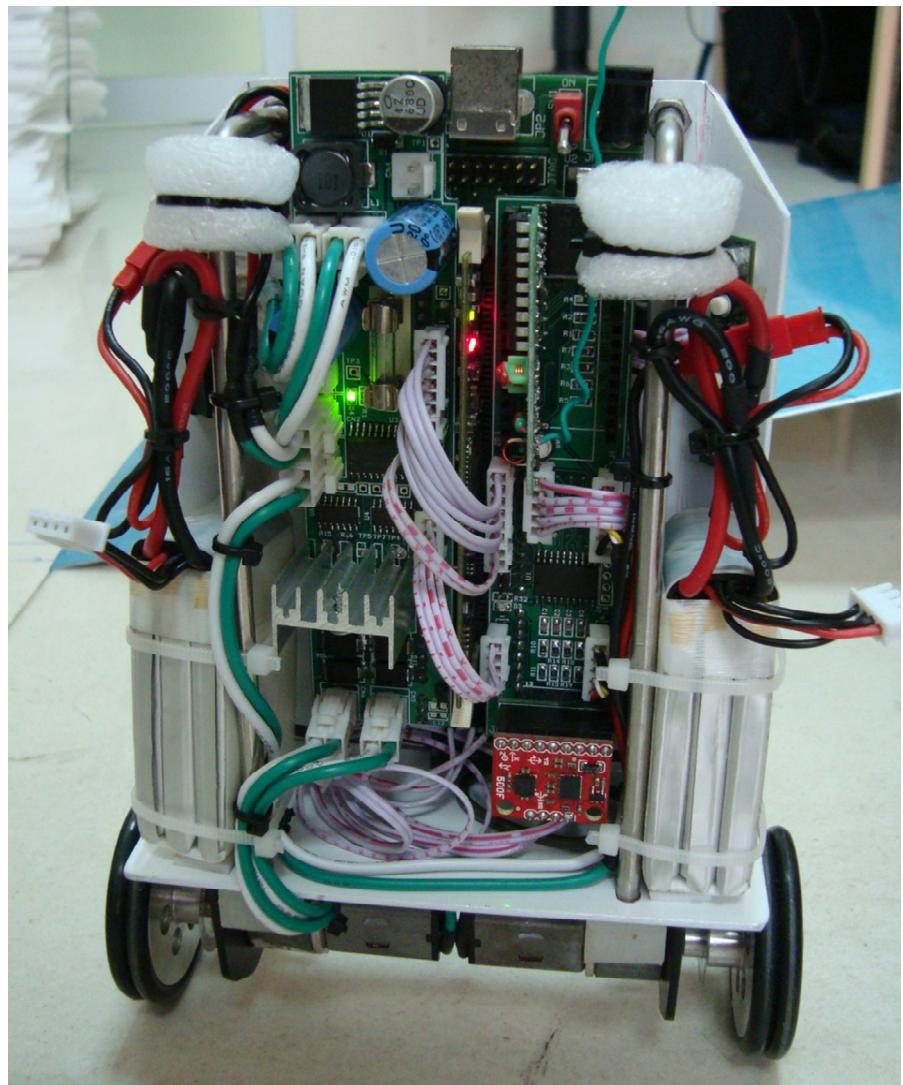
Bao gồm 2 động cơ DC 24V/110mA, tỉ số truyền 1:30, trực tiếp gắn đồng trục lên khung sườn robot. Khung sườn robot sử dụng nhôm tấm và ti inox để gắn kết và cố định các chi tiết lên thân robot.



Hình 5.1 - Bản vẽ 3D mô hình robot hai bánh tự cân bằng.

Thông số mô hình robot hai bánh tự cân bằng như sau:

Kí hiệu	Ý nghĩa	Giá trị
$m - [kg]$	Khối lượng bánh xe	0.044
$M - [kg]$	Khối lượng robot	0.920
$R - [m]$	Bán kính bánh xe	0.0267
$W - [m]$	Chiều rộng robot	0.145
$D - [m]$	Chiều sâu robot	0.055
$H - [m]$	Chiều cao robot	0.185
$L - [m]$	Khoảng cách từ trọng tâm robot đến trục bánh xe	0.1
f_w	Hệ số ma sát giữa bánh xe và mặt phẳng di chuyển	0
f_m	Hệ số ma sát giữa robot và động cơ DC	0.0022
$J_m - [kg \cdot m^2]$	Moment quán tính của động cơ DC	1.4762×10^{-5}
$R_m - [Ohm]$	Điện trở động cơ DC	40
$Kb - [V \sec / rad]$	Hệ số EMF của động cơ DC	0.0218
$Kt - [Nm / A]$	Moment xoắn của động cơ DC	0.0197
n	Tỉ số giảm tốc	30
$g - [m / s^2]$	Gia tốc trọng trường	9.81
$\theta - [rad]$	Góc trung bình của bánh trái và phải	
$\theta_{l,r} - [rad]$	Góc của bánh trái và phải	
$\psi - [rad]$	Góc nghiêng của phần thân robot	
$\phi - [rad]$	Góc xoay của robot	
$x_l, y_l, z_l - [m]$	Tọa độ bánh trái	
$x_r, y_r, z_r - [m]$	Tọa độ bánh phải	
$x_m, y_m, z_m - [m]$	Tọa độ trung bình	
$F_\theta, F_\psi, F_\phi - [Nm]$	Moment phát động theo các phương khác nhau	
$F_{l,r} - [Nm]$	Moment phát động của động cơ bánh trái, phải	
$i_l, i_r - [A]$	Dòng điện động cơ bánh trái, phải	
$v_l, v_r - [V]$	Điện áp động cơ bánh trái, phải	

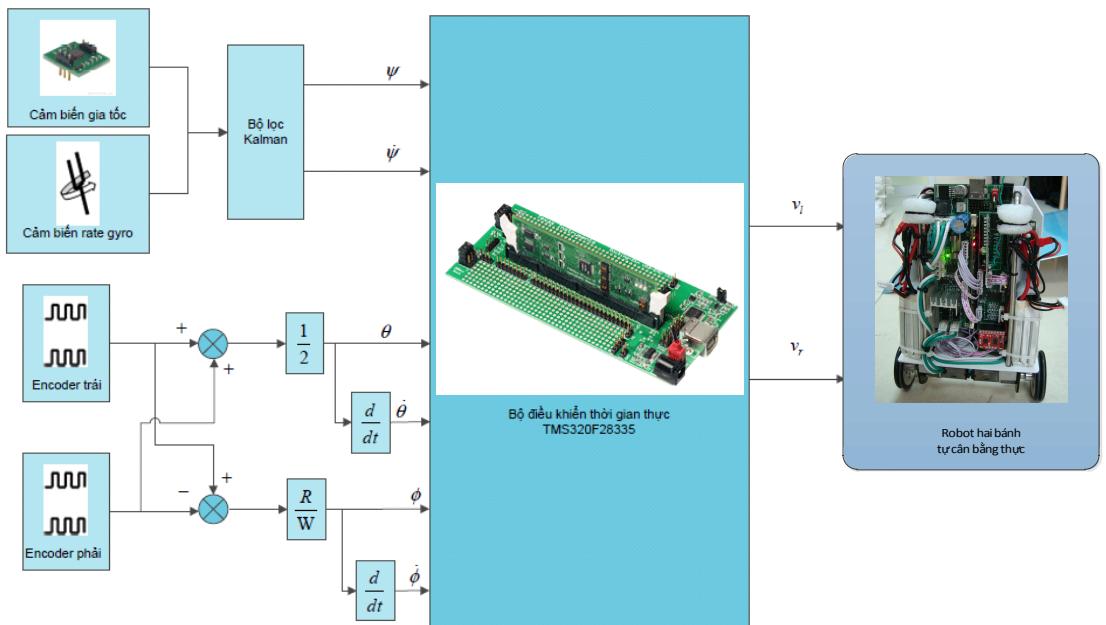


Hình 5.2 - Mô hình robot trên thực tế

5.1.2 Cấu trúc điều khiển phần cứng

Nguồn cung cấp sử dụng pin Lithium-polimer 11.1V/1500mA (x2)

Robot sử dụng chip DSP TMS320F28335 của Texas Instrument dưới dạng KIT thí nghiệm Delfino C28335 làm trung tâm thu thập dữ liệu và điều khiển mọi hoạt động của robot. Cảm biến sử dụng bao gồm: cảm biến gia tốc (accelerometer) kết hợp cảm biến con quay hồi chuyển (gyroscope) để tính toán, ước lượng giá trị góc nghiêng và vận tốc góc nghiêng thân robot, encoder 100 xung/vòng gắn trực tiếp trên bánh xe robot để đo vị trí và vận tốc di chuyển của robot. Một module bluetooth hoặc module thu tín hiệu RF được sử dụng để thu thập dữ liệu và điều khiển trong quá trình hoạt động của robot.



Hình 5.3 - Cấu trúc điều khiển phần cứng của robot

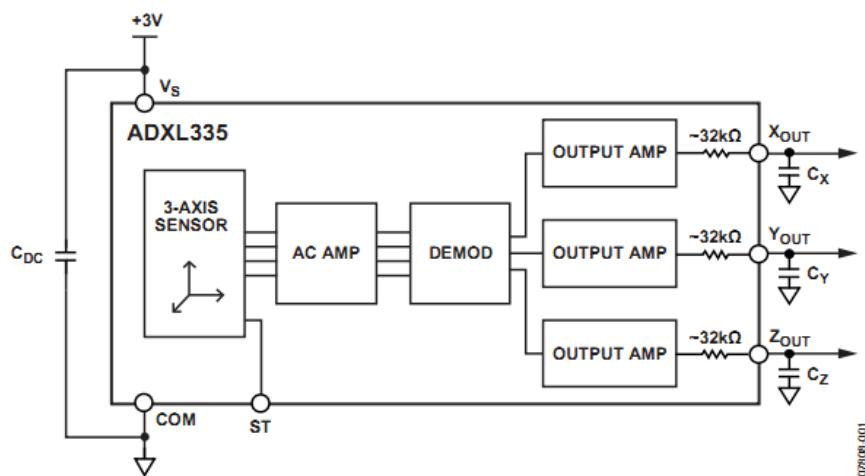
a. *DSP TMS320F28335*

- Hoạt động ở tần số 150Mhz (6.67ns/chu kỳ máy).
- Hỗ trợ hoạt động tính toán số thực single precision 32bit FPU.
- 16 x 16 Dual MAC, 16 x 16 và 32 x 32 MAC.
- Bus có kiến trúc Harvard.
- 6 kênh điều khiển DMA cho các ngoại vi: ADC, McBSP, ePWM, XINTF và SARAM.
- Bộ nhớ: 256 x 16KB Flash, 34KB x 16 SARAM.
- Boot ROM 8K x 16.
- 18 kênh PWM trong đó có 6 kênh PWM có độ phân giải cao 150ps.
- 8 Timer 32 bits.
- Hỗ trợ các chuẩn giao tiếp: 2xCAN, 3xSCI (UART), 2xMcBSP, 1xSPI, 1xI2C.
- 2 bộ giao tiếp ENCODER 32 bit.

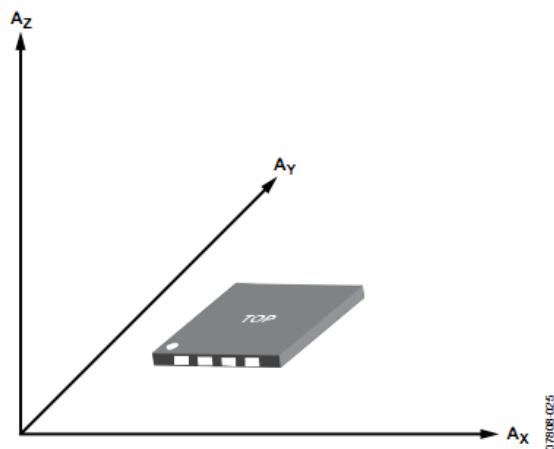
- 2 bộ ADC (16 kênh), tốc độ chuyển đổi 80ns.

b. Cảm biến gia tốc (accelerometer)

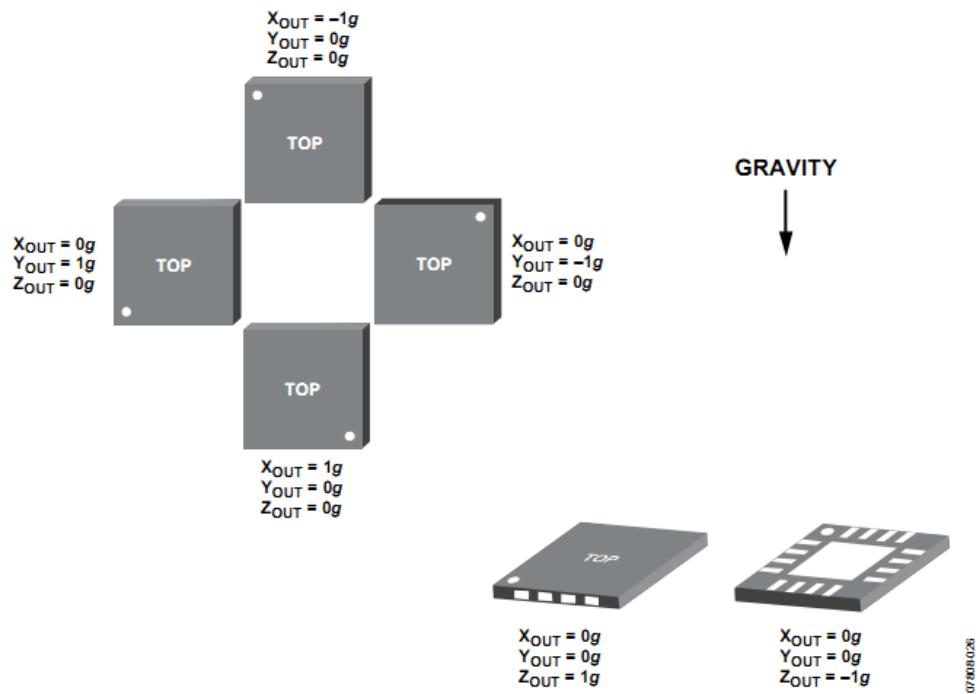
Sử dụng cảm biến gia tốc ADXL335 của hãng Analog Device là loại cảm biến gia tốc 3 trục công suất thấp, thích hợp cho các ứng dụng di động, có tầm đo +/-3g, ngõ ra là tín hiệu điện áp analog, hoạt động ở mức điện áp 3.3V hoàn toàn tương thích với mức điện áp của vi điều khiển trung tâm. Cảm biến gia tốc được sử dụng để đo gia tốc tĩnh, độ nghiêng của hệ robot.



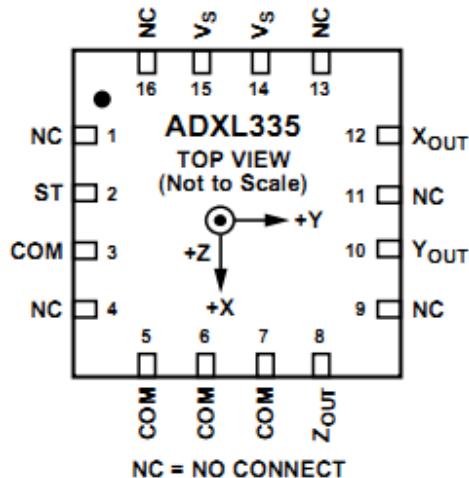
Hình 5.4 - Sơ đồ khối cảm biến gia tốc ADXL335



Hình 5.5 - Phương chiều các trục gia tốc tương ứng



Hình 5.6 - Đáp ứng giá trị gia tốc ngõ ra tương ứng đối với hướng của trọng lực



Hình 5.7 - Sơ đồ chân ngõ ra trên cảm biến gia tốc

- Giá trị điện áp ngõ ra Ax, Ay tại điểm 0g là: 1.65V.
- Giá trị điện áp ngõ ra Az tại điểm 0g là: 1.8V.
- Độ nhạy của cảm biến theo các phương x, y, z là: 300mV/g.

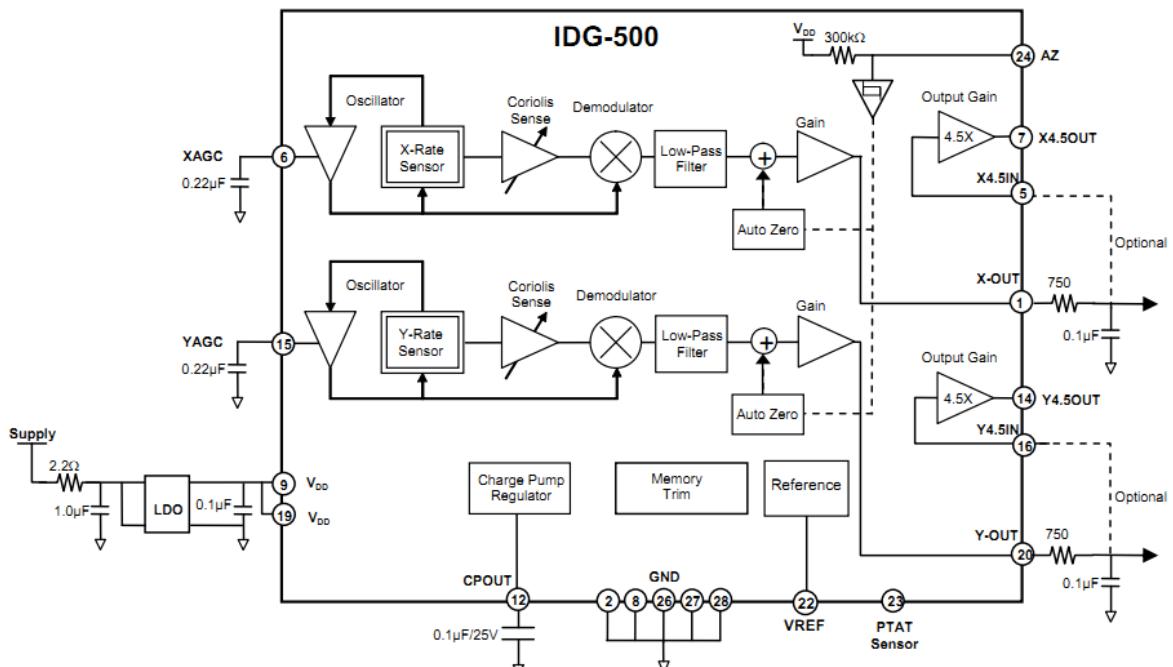
Dựa vào cảm biến gia tốc ADXL335, có thể đo được gia tốc của robot theo 3 phương Ax, A y , A z và dựa vào gia tốc có thể xác định được độ nghiêng dựa vào công thức sau:

$$\psi = \arctan\left(\frac{A_y}{A_z}\right) \quad [5.1]$$

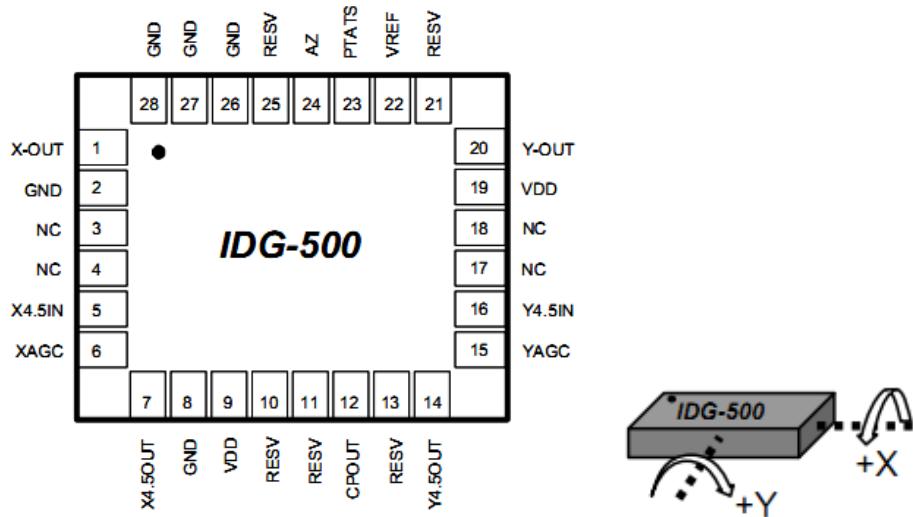
c. Cảm biến con quay hồi chuyển (gyroscope)

IDG500 là cảm biến con quay hồi chuyển 2 trục (X và Y), độ phân giải $\pm 500^\circ / s$ (ngõ ra thường) hoặc $\pm 110^\circ / s$ trên ngõ ra khuếch đại tích hợp, điện áp ngõ ra dạng analog tương ứng có độ phân giải là $2mV^\circ / s$ hoặc $9.1mV^\circ / s$, tích hợp sẵn bộ khuếch đại và bộ lọc thông thấp, hoạt động ở điện áp 3V. Con quay hồi chuyển được sử dụng để đo vận tốc góc nghiêng của robot. Sự kết hợp 2 cảm biến: gia tốc và con quay hồi chuyển để đo 2 thông số: góc nghiêng tĩnh và vận tốc góc nghiêng của robot.

Trong trường hợp chỉ sử dụng một loại cảm biến gia tốc để đo độ nghiêng sau đó lấy vi phân để đo gia tốc nghiêng, do đáp ứng của cảm biến gia tốc là có giới hạn, nên làm như vậy sẽ dẫn đến tình trạng thông số đo được bị sai lệch lớn.



Hình 5.8 - Sơ đồ khối cảm biến con quay hồi chuyển IDG500



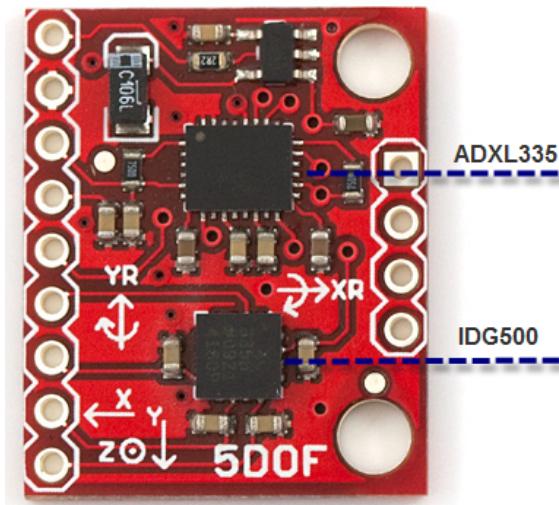
Hình 5.9 - Sơ đồ chân và đáp ứng góc quay trên cảm biến con quay hồi chuyển
Biểu thức mô tả mối quan hệ giữa điện áp ngõ ra của gyro và gia tốc quay:

$$V_{out} = 1.35V + SoA * \dot{\psi} \quad [5.2]$$

SoA [$V /^o /s$] là độ nhạy của cảm biến gia tốc, độ nhạy là $2mV /^o /s$, nếu sử dụng ngõ ra thường (hệ số khuếch đại là 1), và độ nhạy là $9.1mV /^o /s$ nếu sử dụng ngõ ra khuếch đại với hệ số là 4.

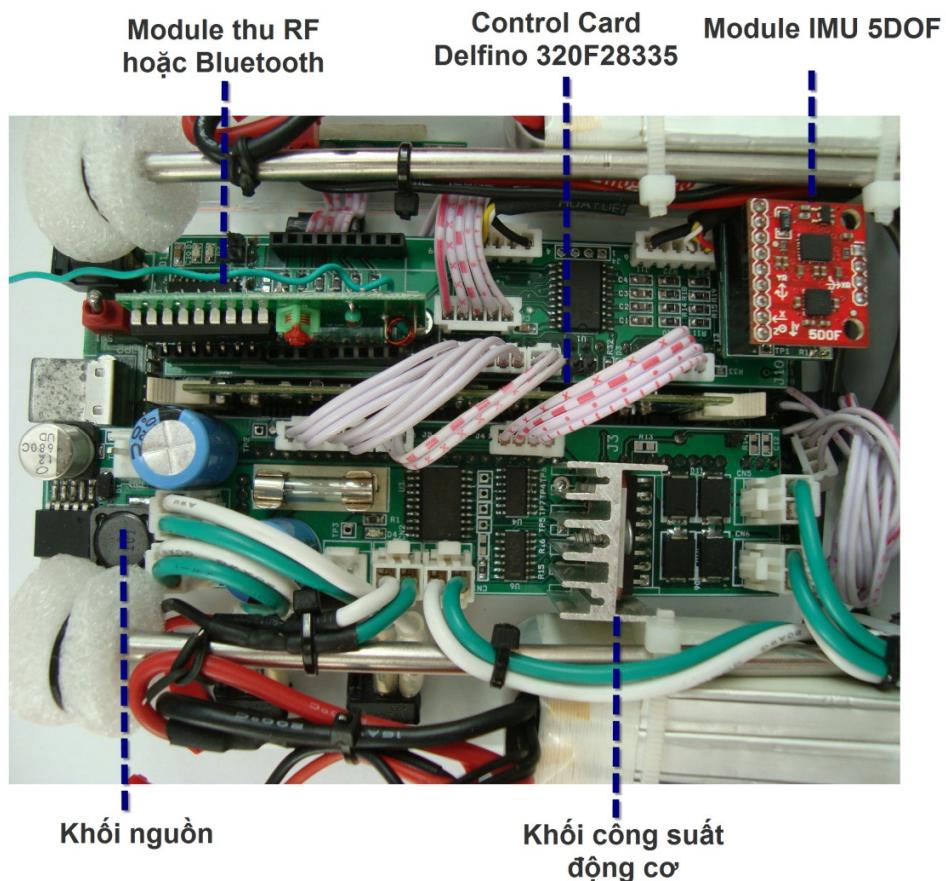
d. Module IMU 5 bậc tự do (5DOF)

Kết hợp cảm biến gia tốc ADXL335 và cảm biến con quay hồi chuyển IDG500 trên một PCB.



Hình 5.10 - Module IMU 5DOF

e. Mạch điều khiển chính

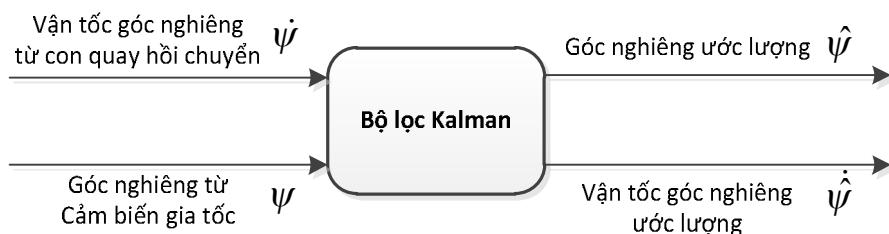


Hình 5.11 - Mạch điều khiển chính trên robot

5.2 Bộ lọc Kalman cho thành phần IMU

5.2.1 Thực hiện bộ lọc Kalman

Luận văn sử dụng hai cảm biến: cảm biến gia tốc (accelerometer) và cảm biến con quay hồi chuyển (gyroscope) để đo góc nghiêng và vận tốc góc nghiêng. Tuy nhiên, vấn đề đặt ra là cần phải kết hợp thông tin từ hai cảm biến để xác định chính xác góc nghiêng thực của hệ robot loại bỏ được ảnh hưởng của nhiễu đo và nhiễu quá trình. Để giải quyết vấn đề này, giải thuật lọc Kalman được sử dụng, với mục đích ước lượng giá trị góc nghiêng của hệ robot từ hai loại cảm biến trên và loại bỏ được ảnh hưởng của nhiễu. Bộ lọc Kalman được khảo sát với mô hình 3 biến trạng thái như sau:



Hình 4.2 Mô hình bộ lọc Kalman với 3 biến trạng thái.

Với mô hình này, bộ lọc sử dụng 2 biến ngõ vào là vận tốc góc nghiêng từ cảm biến con quay hồi chuyển và góc nghiêng từ cảm biến gia tốc; 2 biến ngõ ra là góc nghiêng ước lượng và vận tốc góc nghiêng ước lượng.

Mô hình bộ lọc Kalman 3 biến được xây dựng như sau:

Đặt:

x_1 là vận tốc góc đo được từ cảm biến gyro.

x_2 là giá trị góc nghiêng đo được từ cảm biến gia tốc.

x_3 là giá trị bias – phân cực của gyro.

$w_{1,2}$ là nhiễu quá trình, v là nhiễu do.

Ta có, phương trình toán học bộ lọc Kalman 3 biến như sau:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = w_1(t) \\ \dot{x}_3 = w_2(t) \end{cases} \Rightarrow \begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} w(t) \\ z = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + v(t) \end{cases}$$

Rời rạc hệ phương trình trên, ta được:

$$\begin{aligned} \dot{x}_1 &= \frac{x_1(k+1) - x_1(k)}{T} & \frac{x_1(k+1) - x_1(k)}{T} &= x_2(k) \\ \dot{x}_2 &= \frac{x_2(k+1) - x_2(k)}{T} & \frac{x_2(k+1) - x_2(k)}{T} &= w_1(k) \Rightarrow \begin{cases} x_1(k+1) = x_1(k) + T x_2(k) \\ x_2(k+1) = x_2(k+1) + T w_1(k) \end{cases} \\ \dot{x}_3 &= \frac{x_3(k+1) - x_3(k)}{T} & \frac{x_3(k+1) - x_3(k)}{T} &= w_2(k) \\ &&& \begin{cases} x_3(k+1) = x_3(k) + T w_2(k) \end{cases} \end{aligned}$$

Và:

$$\begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} v(k)$$

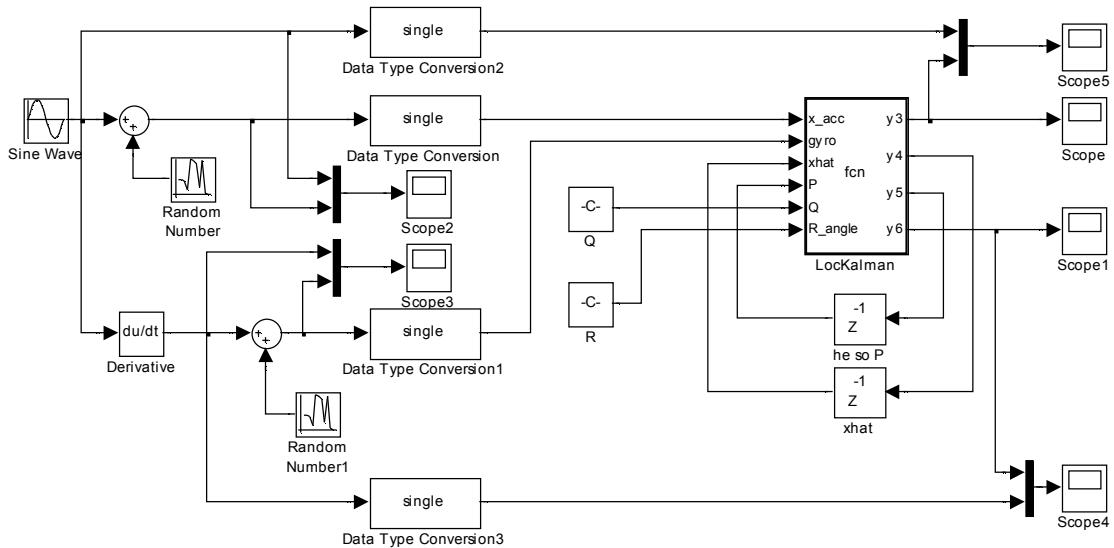
Hệ phương trình rời rạc:

$$\begin{cases} \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \end{bmatrix} = \begin{bmatrix} 1 & T & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} 0 \\ w_1(k) \\ w_2(k) \end{bmatrix} \\ \begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} v(k) \end{cases}$$

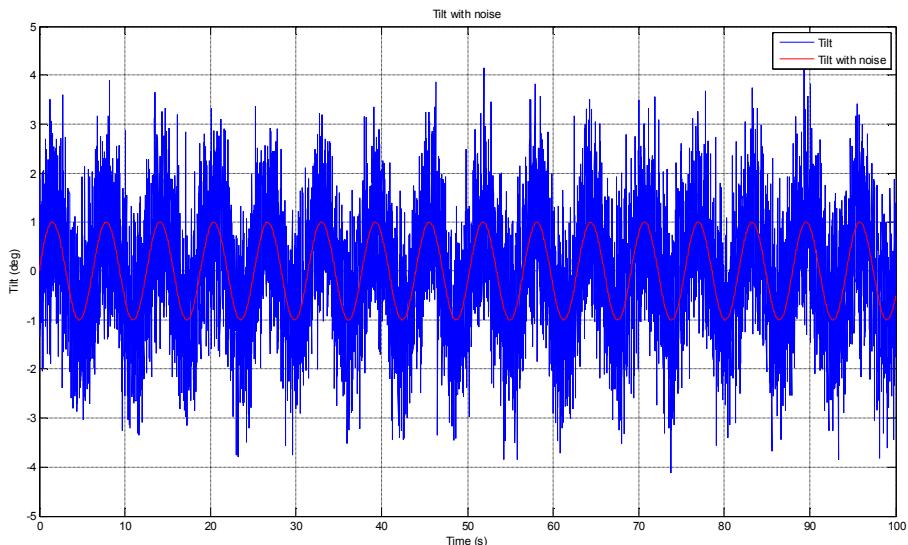
Phương trình cập nhật giá trị ước lượng:

$$\hat{x}_k = \hat{x}_k^- + K \left(\begin{bmatrix} Angle_dot_measured \\ Angle_measured \end{bmatrix} - \begin{bmatrix} Angle_dot_estimated \\ Angle_estimated \end{bmatrix} \right)$$

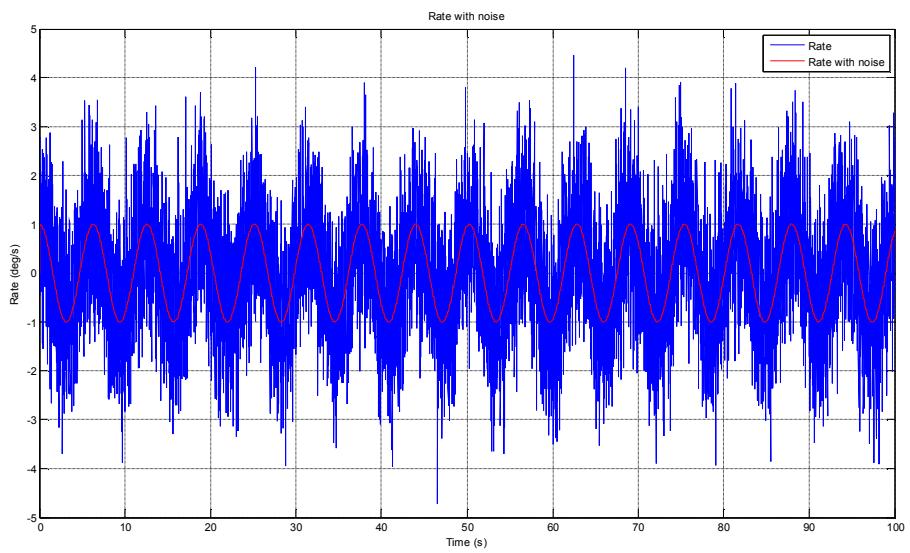
Dùng Matlab kiểm chứng tính đúng đắn của mô hình bộ lọc Kalman:



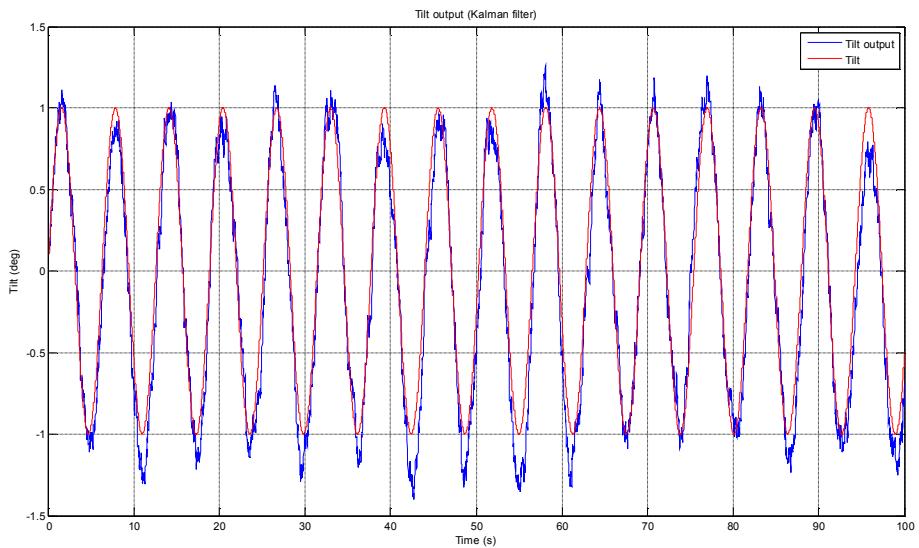
Hình 4.3 Kiểm chứng mô hình bộ lọc Kalman trong Matlab simulink.



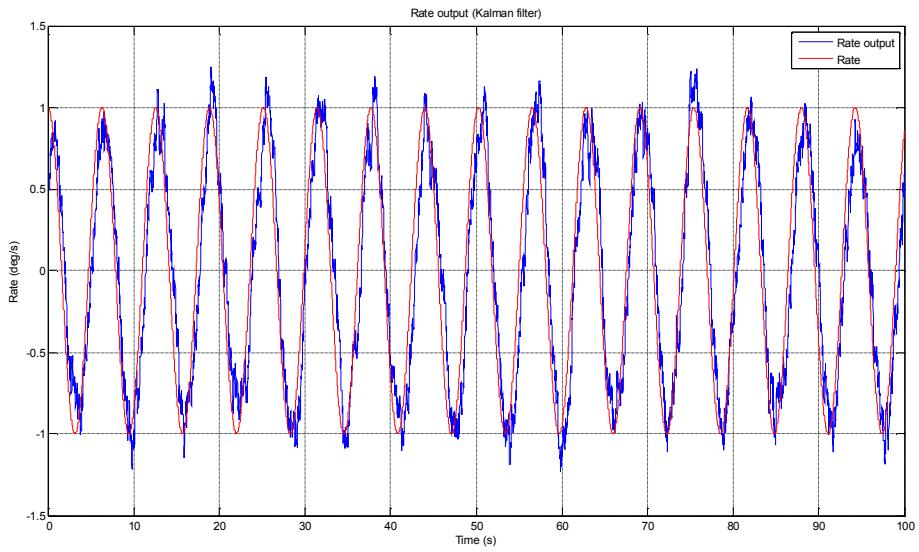
Hình 4.4 Tín hiệu là góc nghiêng và góc nghiêng có nhiễu tác động



Hình 4.5 Tín hiệu là vận tốc góc nghiêng và vận tốc góc nghiêng có nhiễu tác động



Hình 4.6 Góc nghiêng sau khi qua bộ lọc Kalman và góc nghiêng không nhiễu ban đầu



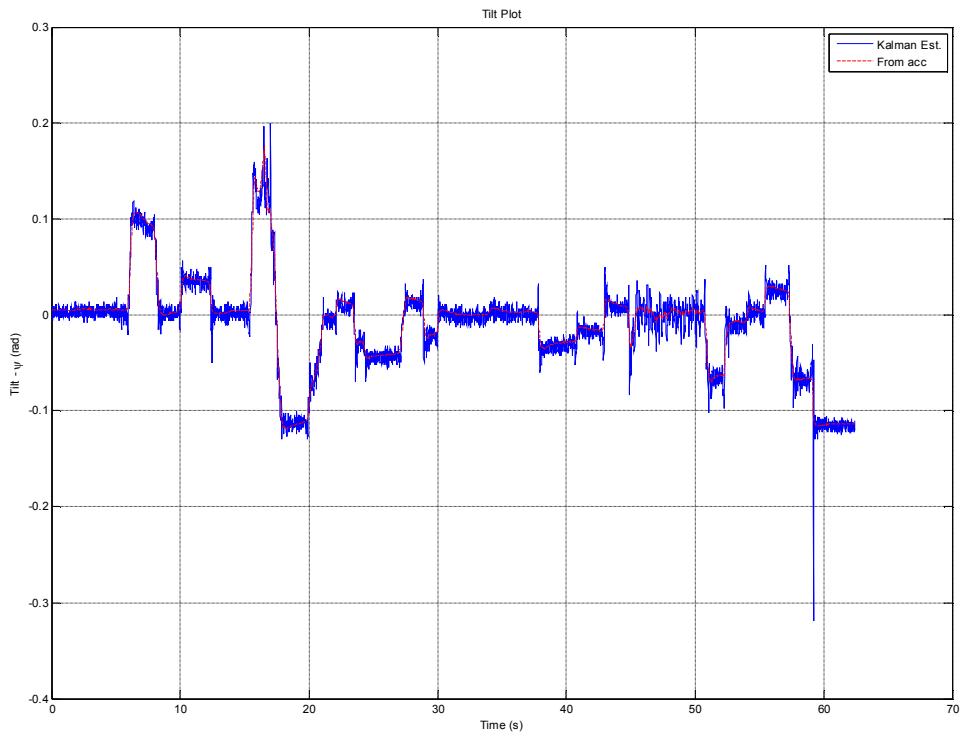
Hình 4.7 Vận tốc góc nghiêng sau khi qua bộ lọc Kalman và vận tốc góc nghiêng không nhiễu ban đầu

5.2.2 Kết quả thực nghiệm

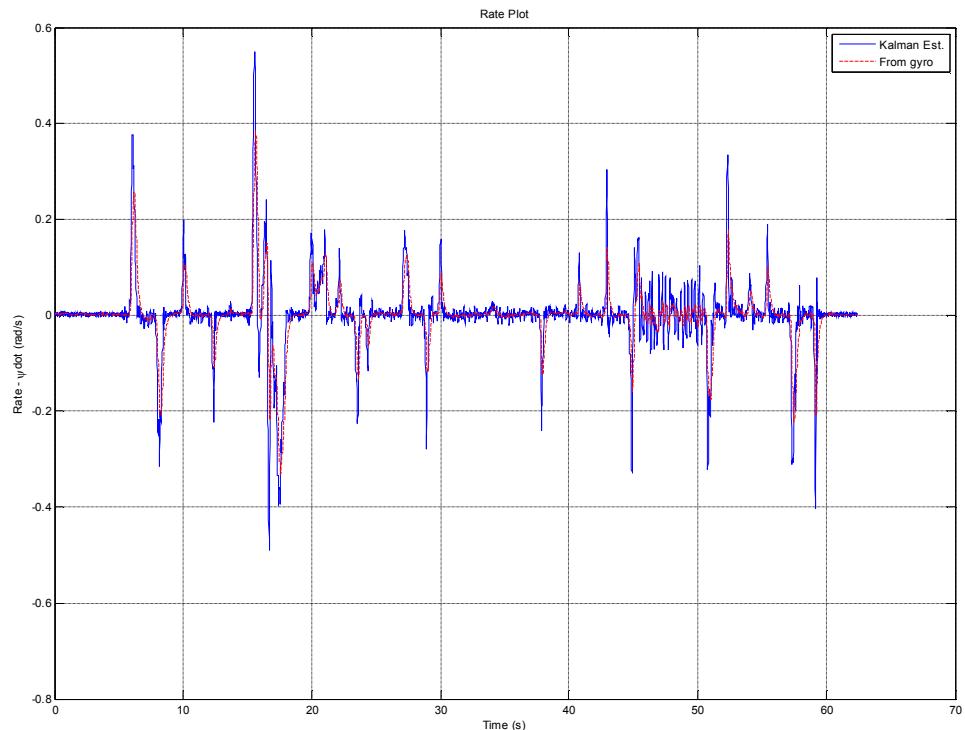
Áp dụng mô hình bộ lọc Kalman trên cho module IMU để ước lượng lượng góc nghiêng và vận tốc góc nghiêng thân robot trên thực tế.

Sử dụng mô hình bộ lọc Kalman 3 biến trạng thái, chu kì lấy mẫu 10ms, ma trận các trọng số được chọn thử sai trong quá trình thực nghiệm để cho kết quả tốt nhất như sau:

$$Q = \begin{bmatrix} 1e-3 & 0 & 0 \\ 0 & 9e-2 & 0 \\ 0 & 0 & 1e-6 \end{bmatrix} \text{ và } R = \begin{bmatrix} 20 & 0 \\ 0 & 20 \end{bmatrix}$$



Hình 5.12 - Ngõ ra góc nghiêng từ cảm biến gia tốc và ngõ ra góc nghiêng ước lượng từ bộ lọc Kalman



Hình 5.13 - Ngõ ra vận tốc góc nghiêng từ cảm biến con quay hồi chuyển và ngõ ra vận tốc góc nghiêng ước lượng từ bộ lọc Kalman

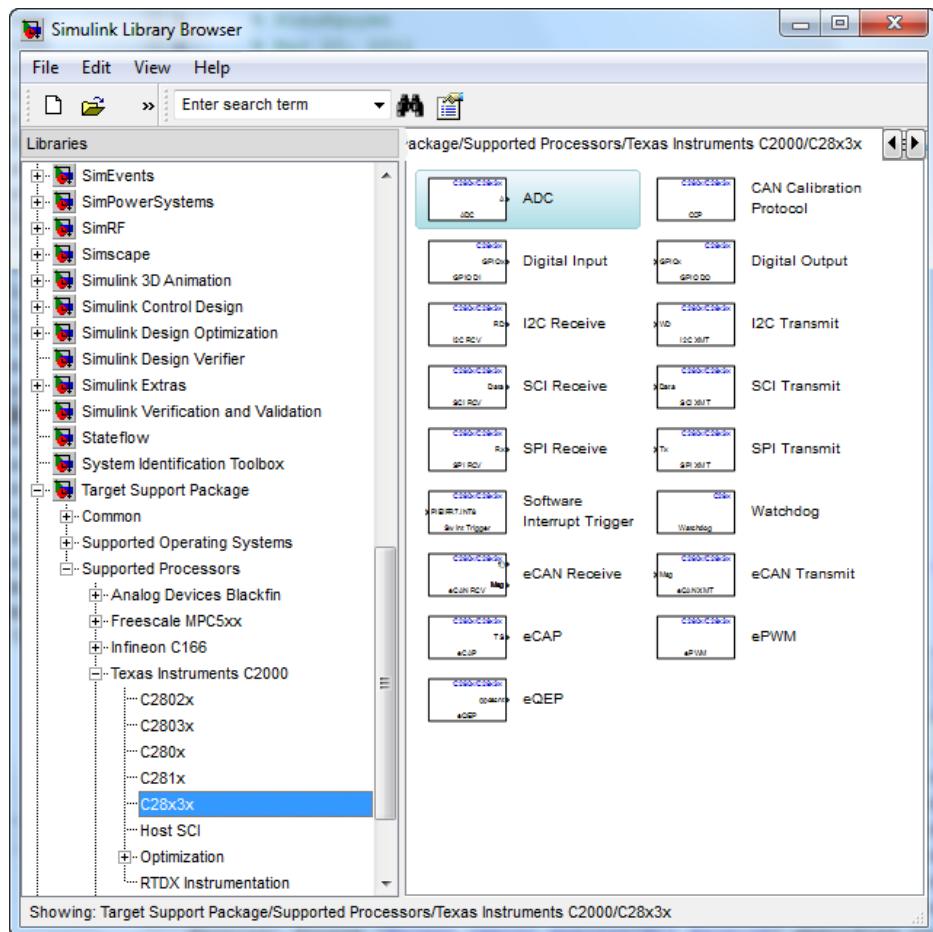
5.2.3 Kết luận

Tín hiệu góc nghiêng nhận được từ cảm biến gia tốc bị nhiễu rất nhiều, trong khi đó, nếu tích phân tín hiệu vận tốc góc thu được từ cảm biến con quay hồi chuyển để lấy giá trị góc nghiêng thì giá trị này bị trôi xa so với giá trị thực tế. Sau khi sử dụng bộ lọc Kalman 3 biến trạng thái như đã trình bày thì tín hiệu góc nghiêng và vận tốc góc nghiêng ước lượng loại bỏ nhiễu đáng kể và loại bỏ trôi hoàn toàn.

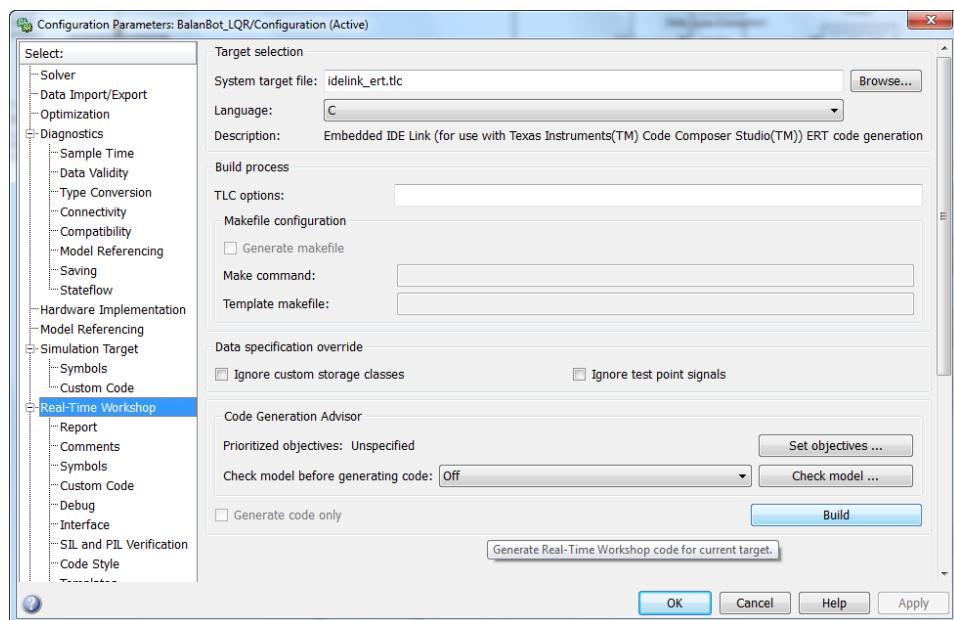
5.3 Bộ điều khiển nhúng robot hai bánh tự cân bằng

5.3.1 Giới thiệu

Quá trình điều khiển robot được thực hiện trên vi điều khiển DSP TMS320F28335 với sự hỗ trợ của thư viện Target Support Package và Real-Time Workshop của Matlab 2010b – hỗ trợ cho các vi điều khiển họ TI C2000. Bộ điều khiển được thiết kế trên Simulink trong miềon rời rạc, sau đó các ứng dụng được Embedded IDE Link biên dịch sang mã ngôn ngữ C tương thích với thư viện hỗ trợ bởi Code Composer, sau đó tiếp tục được Code Composer biên dịch thành mã máy nhúng MCU DSP TMS320F28335.



Hình 5.14 - Thư viện hỗ trợ trong Target Support Package cho MCU DSP C2000

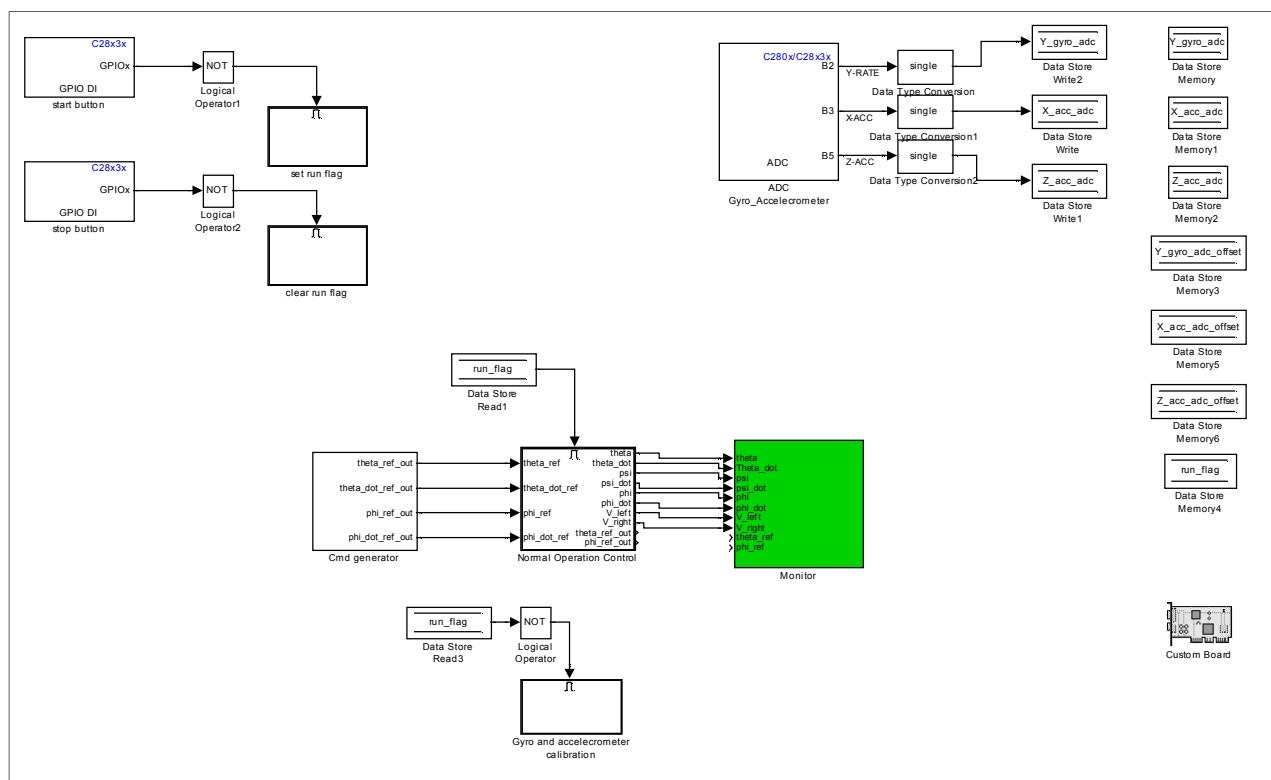


Hình 5.15 - Lựa chọn System Target File để Embedded IDE Link biên dịch chương trình từ Simulink sang ngôn ngữ C được Code Composer Studio hỗ trợ

Dữ liệu thu thập được để đánh giá đáp ứng của robot, chất lượng của bộ điều khiển và quá trình điều khiển hoạt động di chuyển của robot được chuyển qua giao tiếp SCI của MCU và chuyển tiếp qua module bluetooth để truyền nhận dữ liệu không dây đến module bluetooth trên máy tính để quá trình vận hành được linh hoạt.

5.3.2 Bộ điều khiển LQR PI cho khâu vị trí

a. Sơ đồ các khối

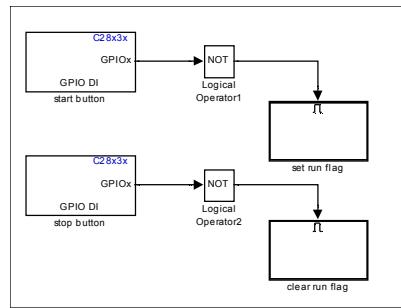


Hình 5.16 - Bộ điều khiển LQR PI khâu vị trí cho robot hai bánh tự cân bằng

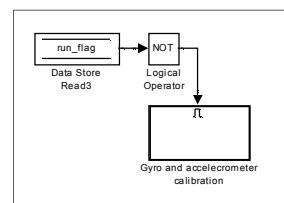
Bộ điều khiển bao gồm các khối sau:

- Khối nhận tín hiệu bắt đầu/tạm ngưng từ nút nhấn start/stop.
- Khối tự động cân chỉnh IMU: trong thời gian robot tạm nhung hoạt động, khối này tự động lấy mẫu và thiết lập giá trị tham chiếu (offset) cho cảm biến gia tốc và cảm biến con quay hồi chuyển.

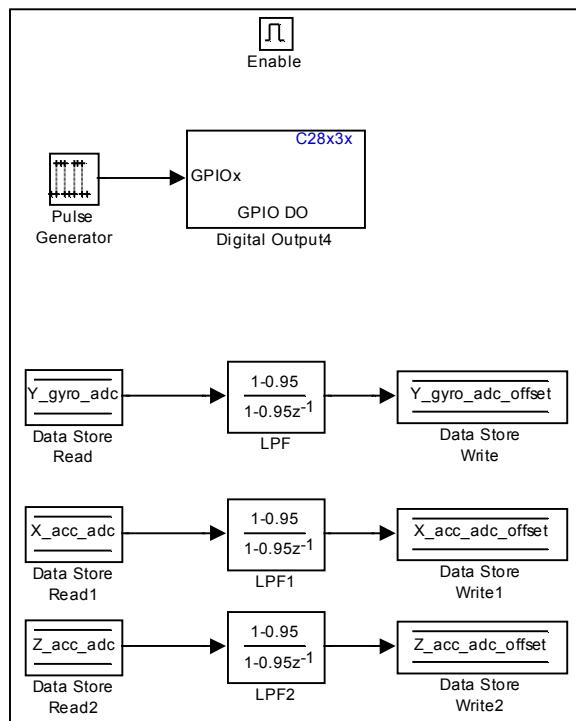
- Khối thu thập và chia sẻ dữ liệu từ cảm biến trên IMU: liên tục lấy mẫu, cập nhật giá trị ADC từ các cảm biến trên IMU.
- Khối vận hành chính của robot: giữ thăng bằng và di chuyển theo lệnh điều khiển từ remote RF thông qua IO trên MCU.



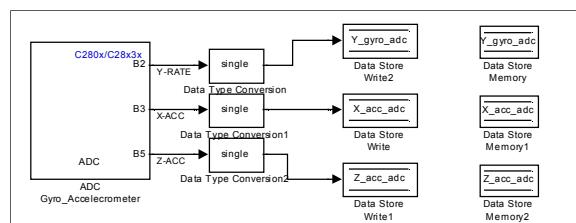
Hình 5.17 - Khối nhận tín hiệu bắt đầu/tạm ngưng



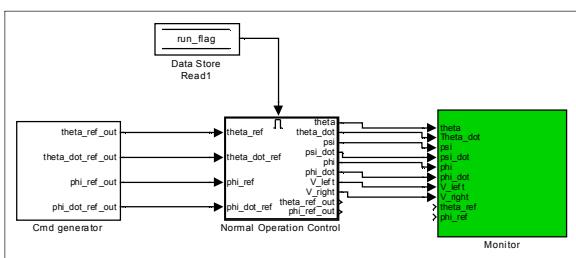
Hình 5.18 - Khối tự động cân chỉnh IMU



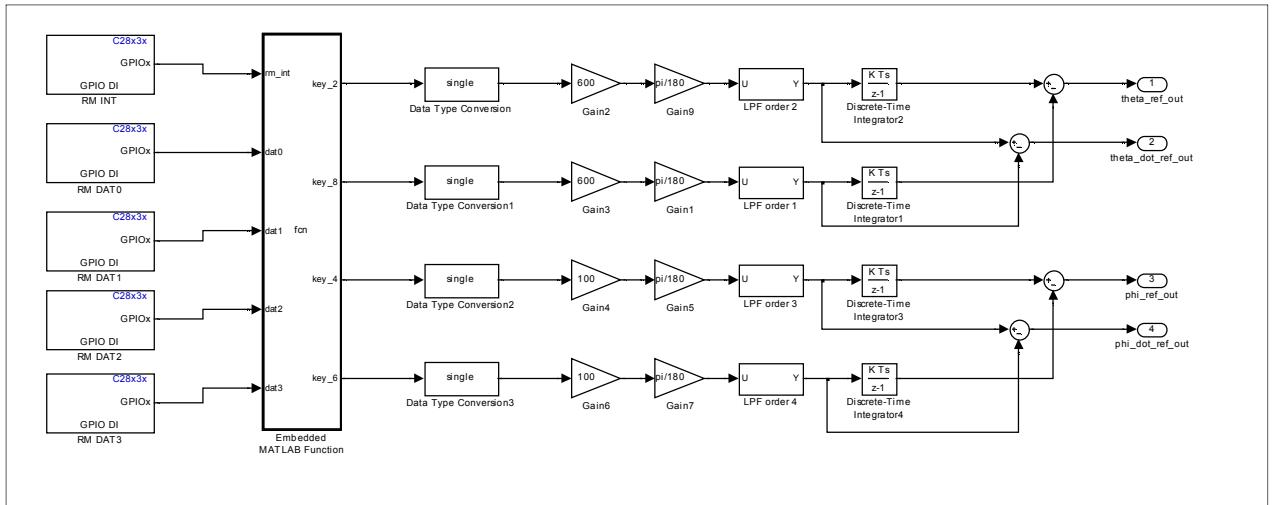
Hình 5.19 - Bên trong khối tự động cân chỉnh IMU



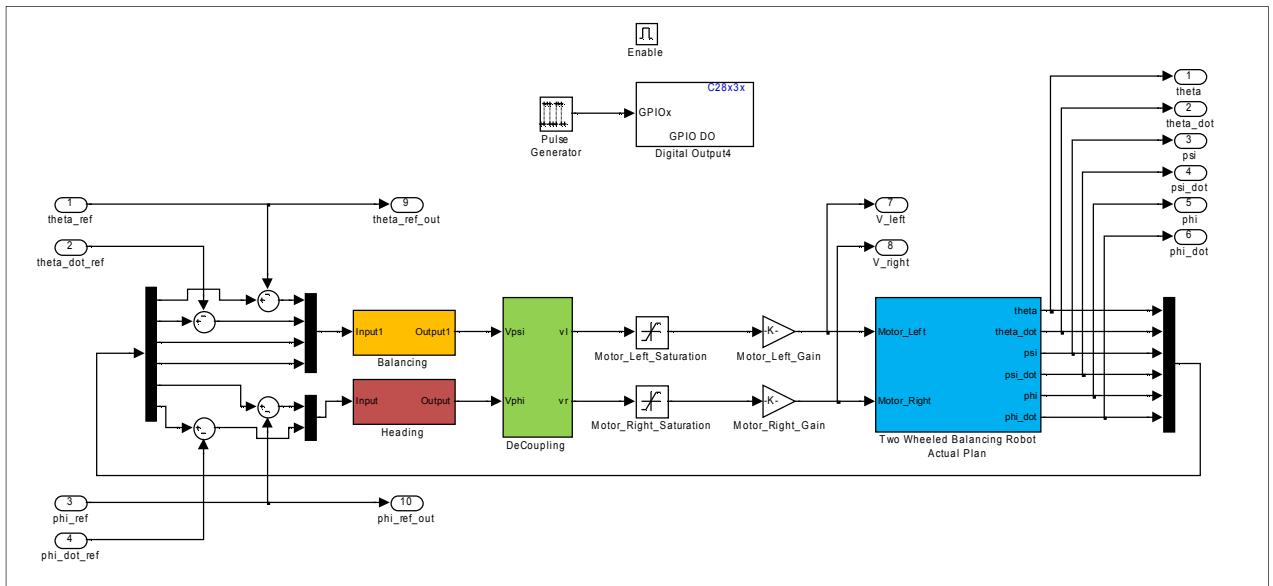
Hình 5.20 - Khối thu thập và chia sẻ dữ liệu từ cảm biến trên IMU



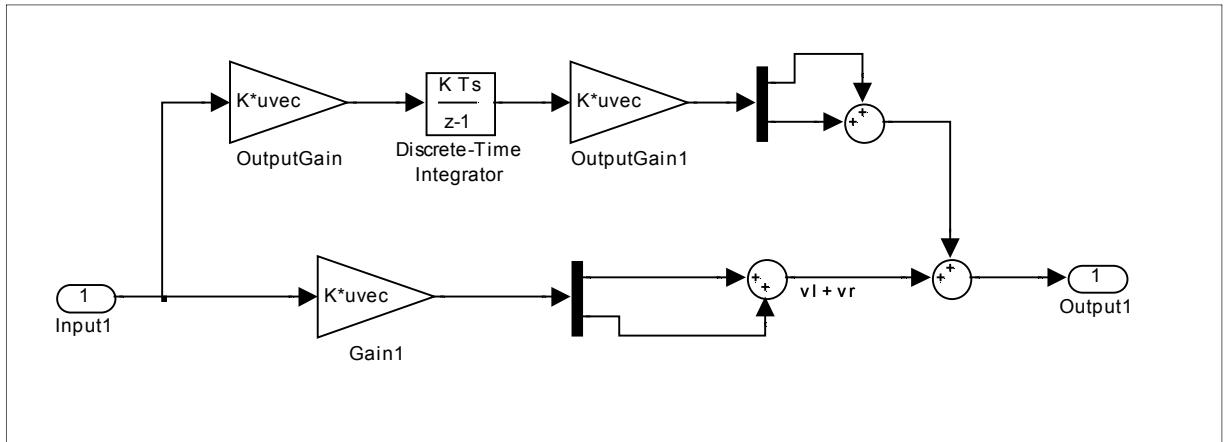
Hình 5.21 - Khối vận hành chính của robot



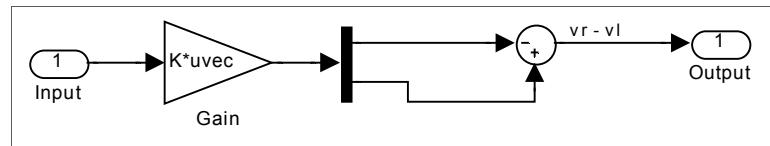
Hình 5.22 - Bên trong khối nhận lệnh từ remote RF “Cmd Generator”



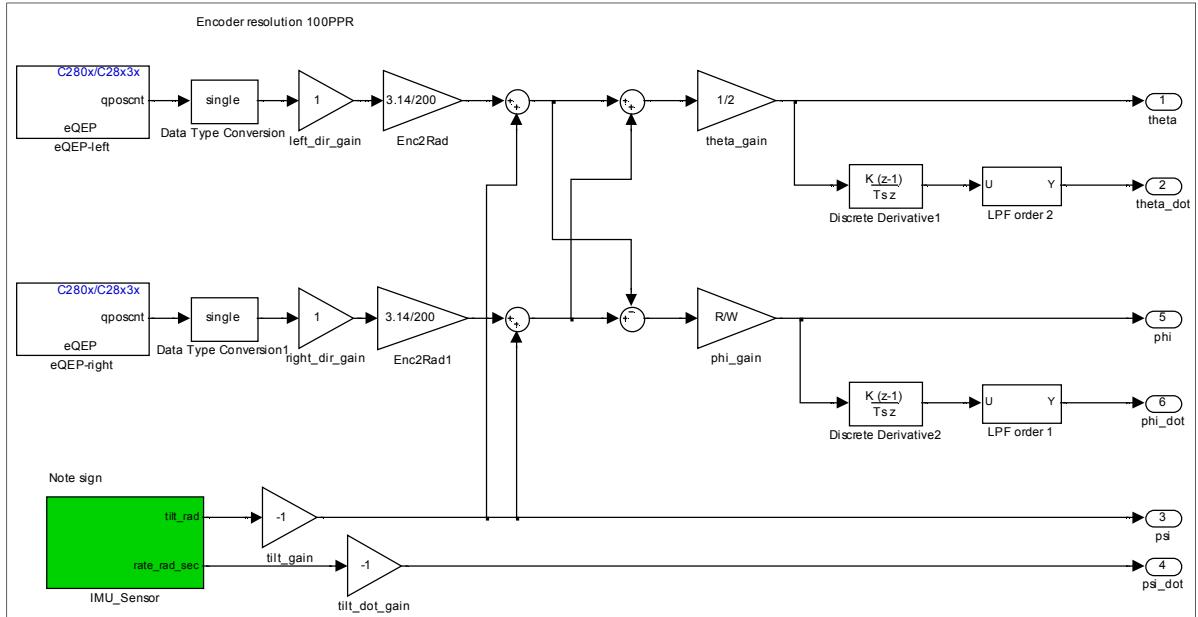
Hình 5.23 - Bên trong khối “Normal Operation Control” – Giữ thăng bằng và di chuyển robot



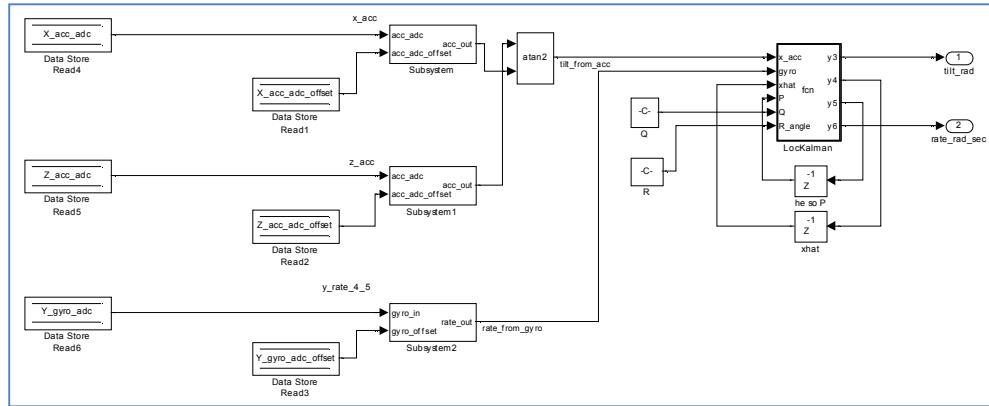
Hình 5.24 - Bên trong khối điều khiển giữ thăng bằng và vị trí “Balancing” – sử dụng LQR PI khâu vị trí



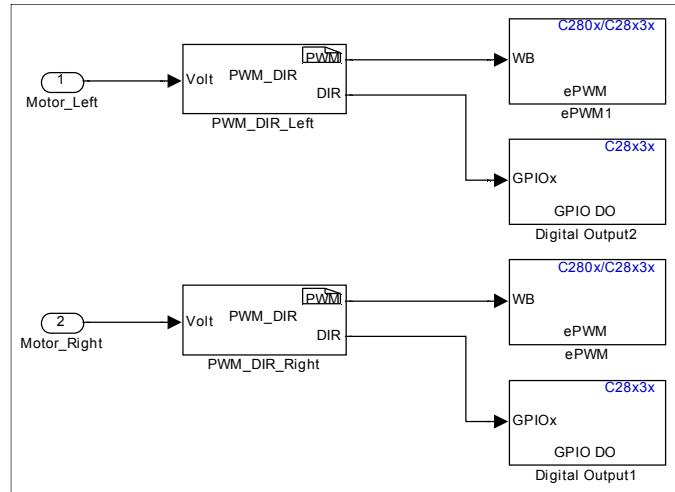
Hình 5.25 - Bên trong khối điều khiển góc xoay robot “Heading” – sử dụng LQR



Hình 5.26 - Bên trong khối nhận tín hiệu hồi tiếp từ cảm biến IMU và encoder.



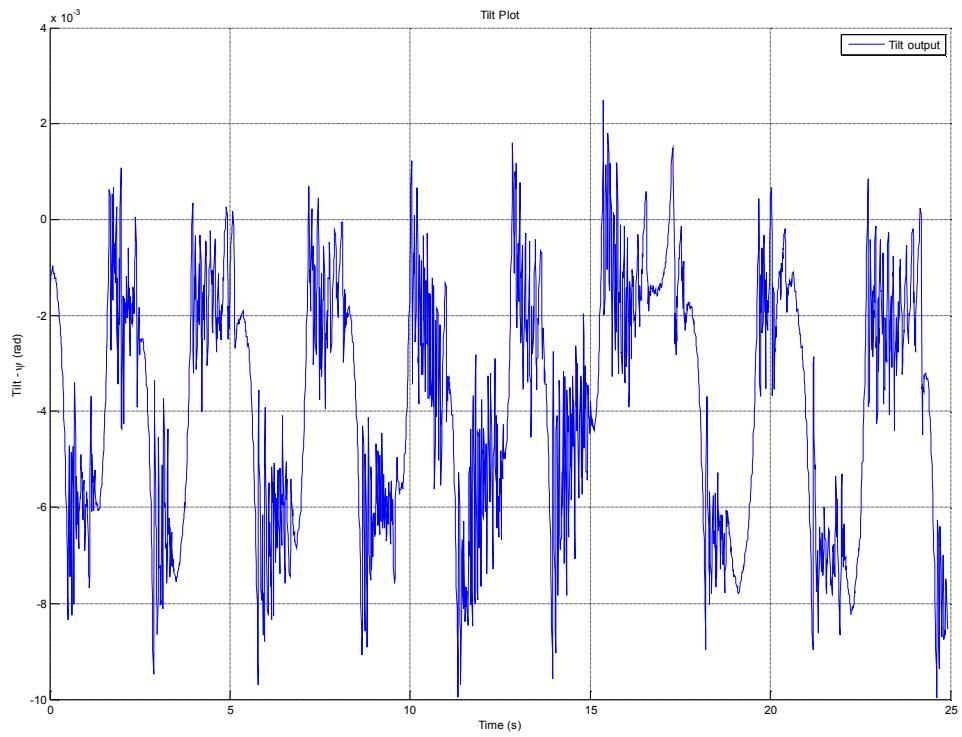
Hình 5.27 - Khối lọc Kalman cho IMU



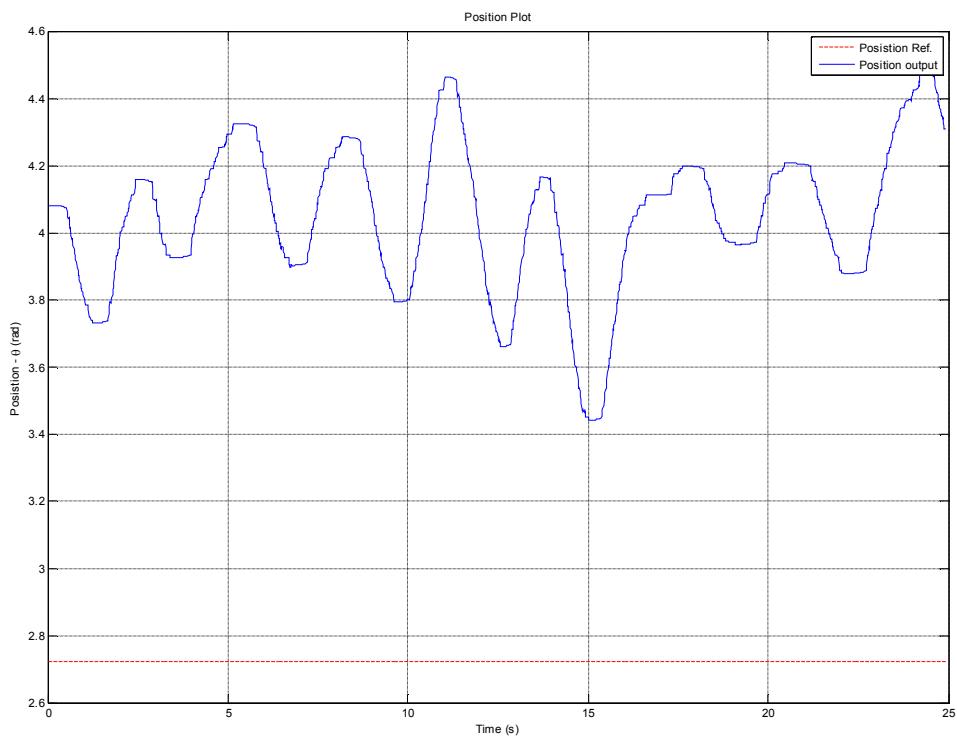
Hình 5.28 - Bên trong khối điều khiển động cơ bánh trái/phải

b. Kết quả thực nghiệm bộ điều khiển LQR PI cho robot hai bánh tự cân bằng

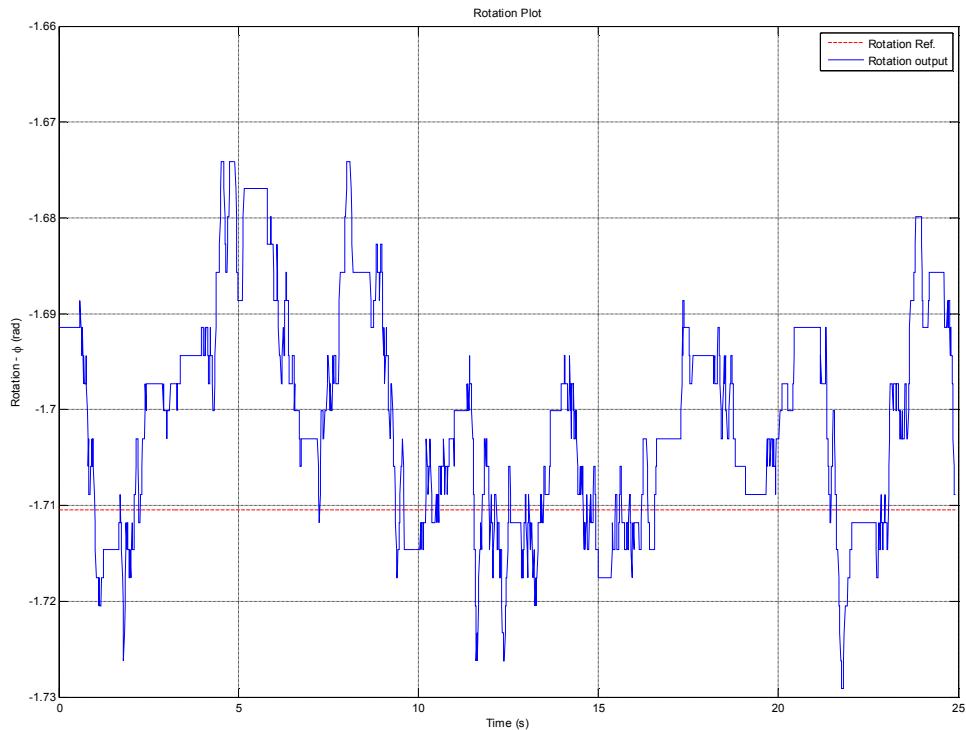
- Khi robot giữ thăng bằng trên địa hình phẳng:



Hình 5.29 - Ngõ ra góc nghiêng trong khi giữ thăng bằng trên địa hình phẳng, sai số vào khoảng ± 0.01 (rad)

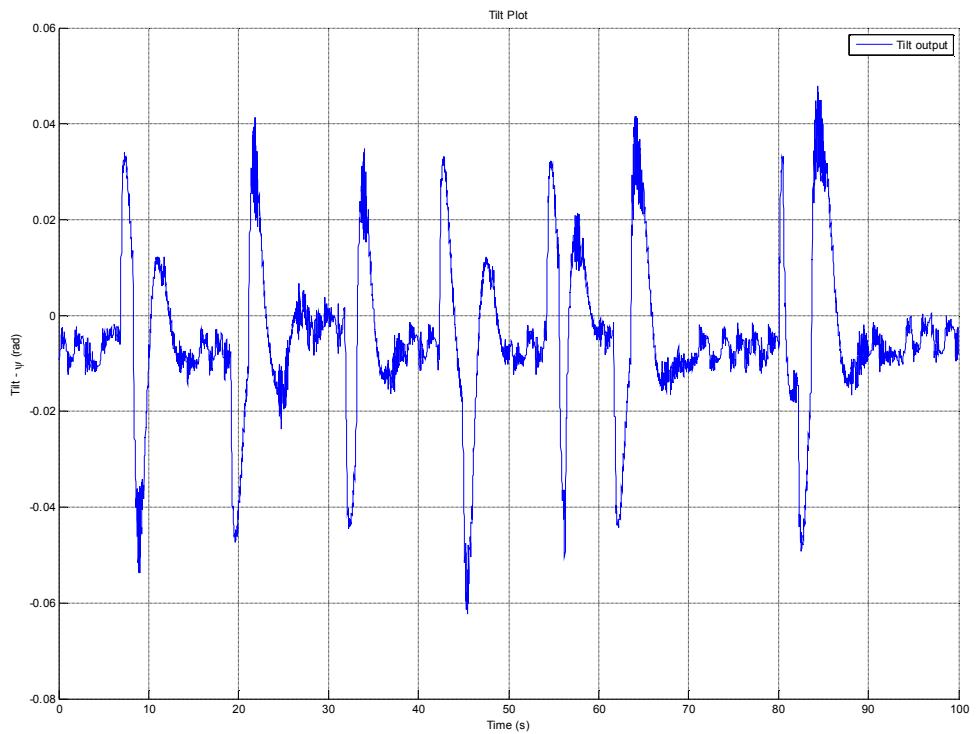


Hình 5.30 - Ngõ ra vị trí trong khi giữ thăng bằng trên địa hình phẳng, sai số vị trí trung bình khoảng 4 (rad)

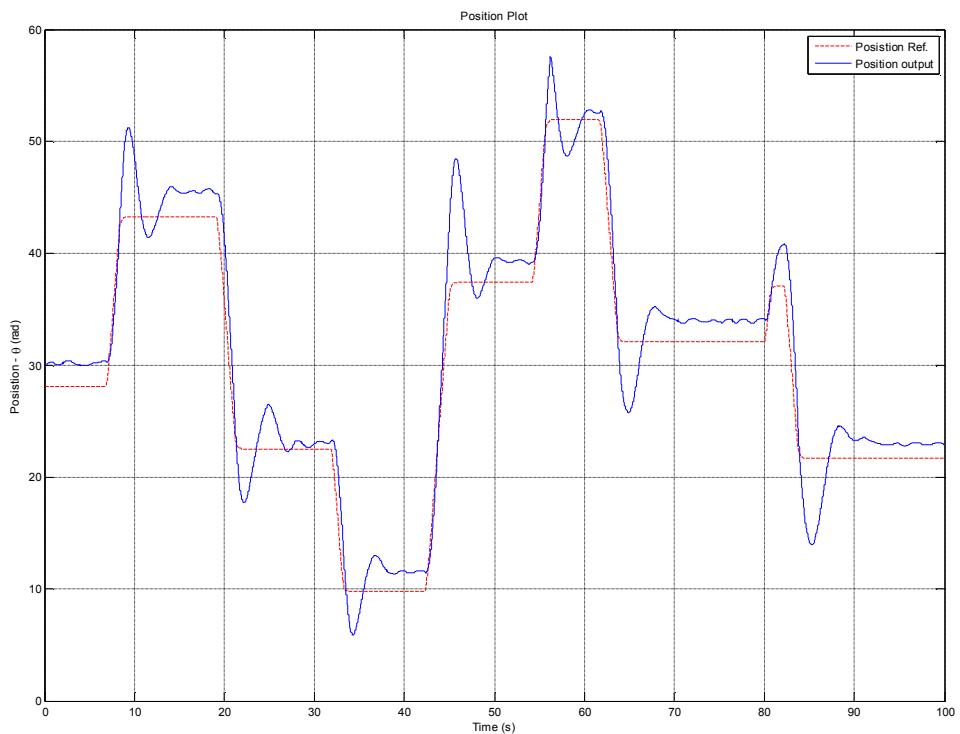


Hình 5.31 - Ngõ ra góc xoay trong khi giữ thăng bằng trên địa hình phẳng, sai số trung bình khoảng 0.02 (rad)

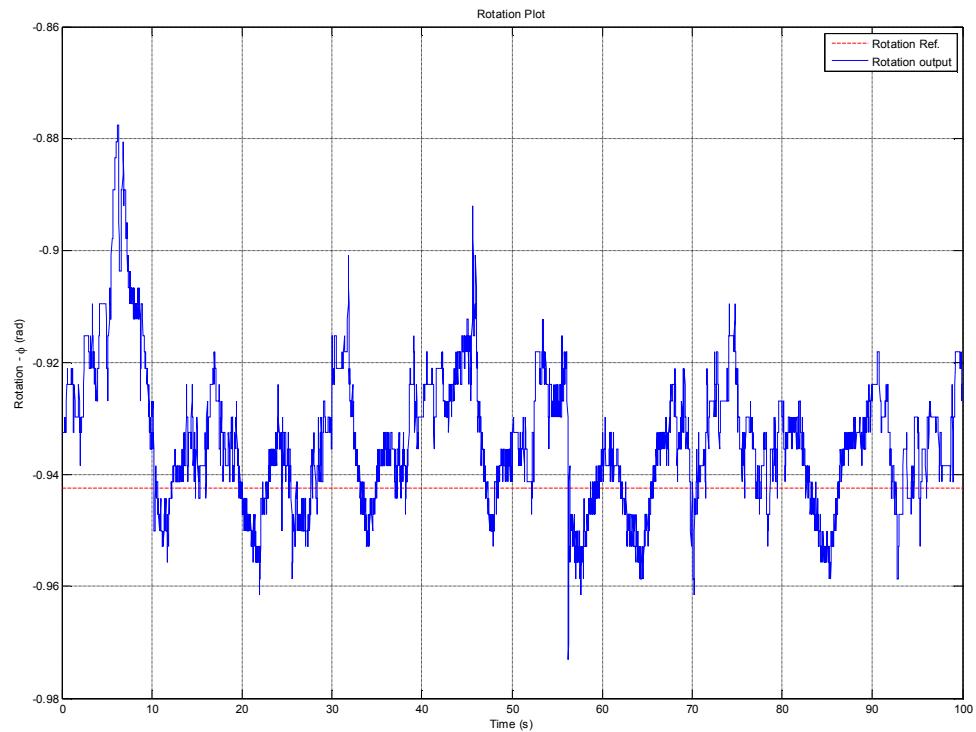
- Khi robot di chuyển trên địa hình phẳng:



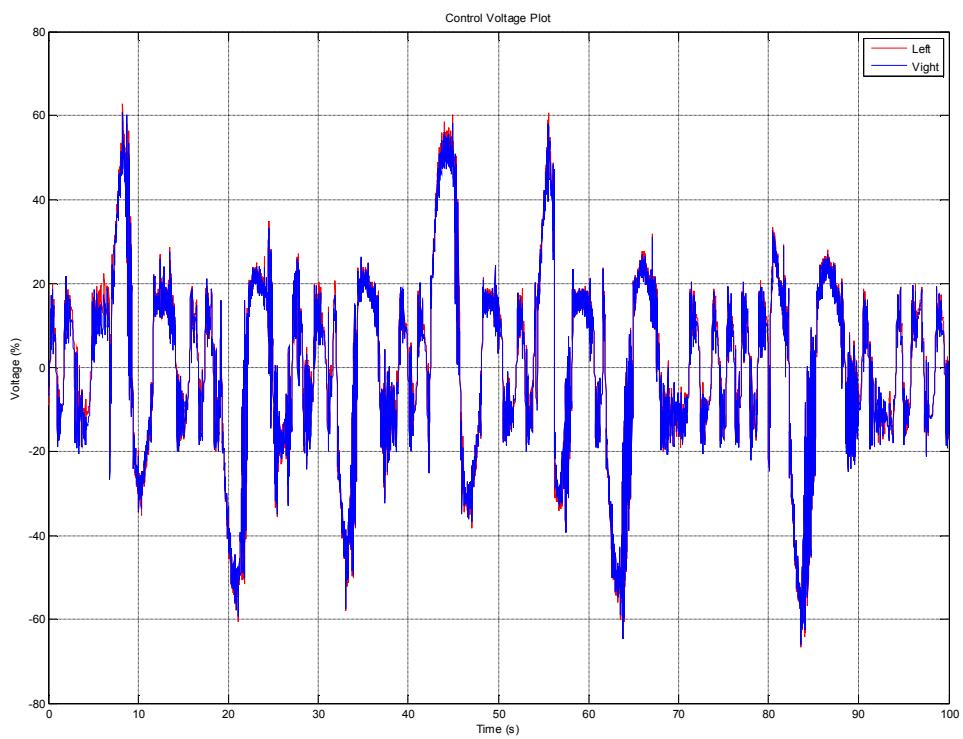
Hình 5.32 - Ngõ ra góc nghiêng khi robot di chuyển trên địa hình phẳng với tín hiệu đặt vị trí và vận tốc từ remote.



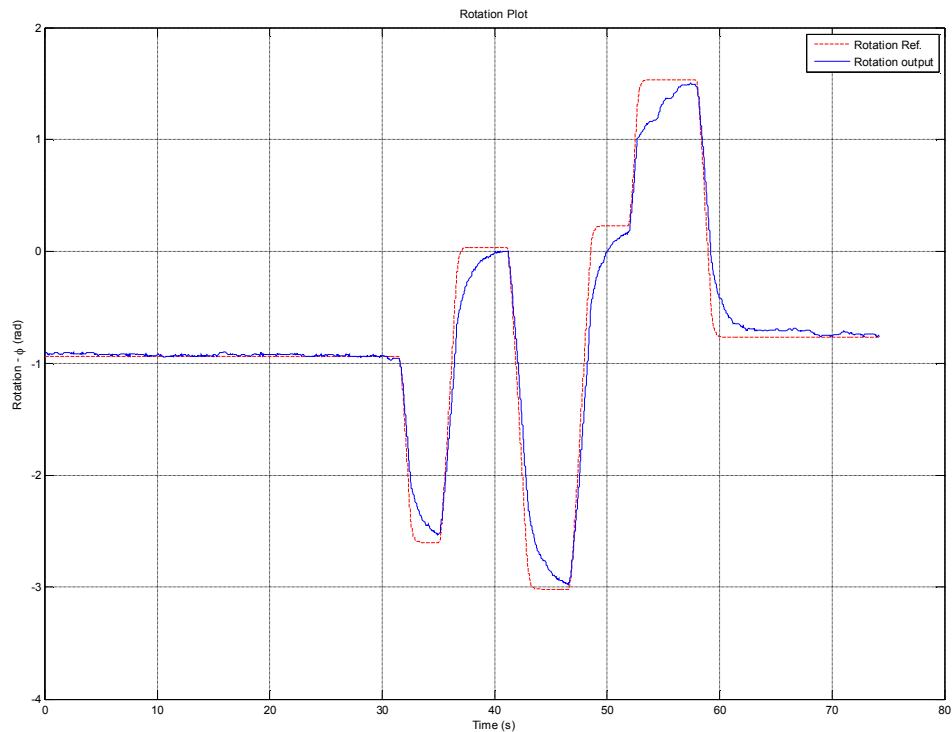
Hình 5.33 - Ngõ ra vị trí khi robot di chuyển trên địa hình phẳng với tín hiệu đặt vị trí và vận tốc từ remote, sai số khoảng 4 (rad)



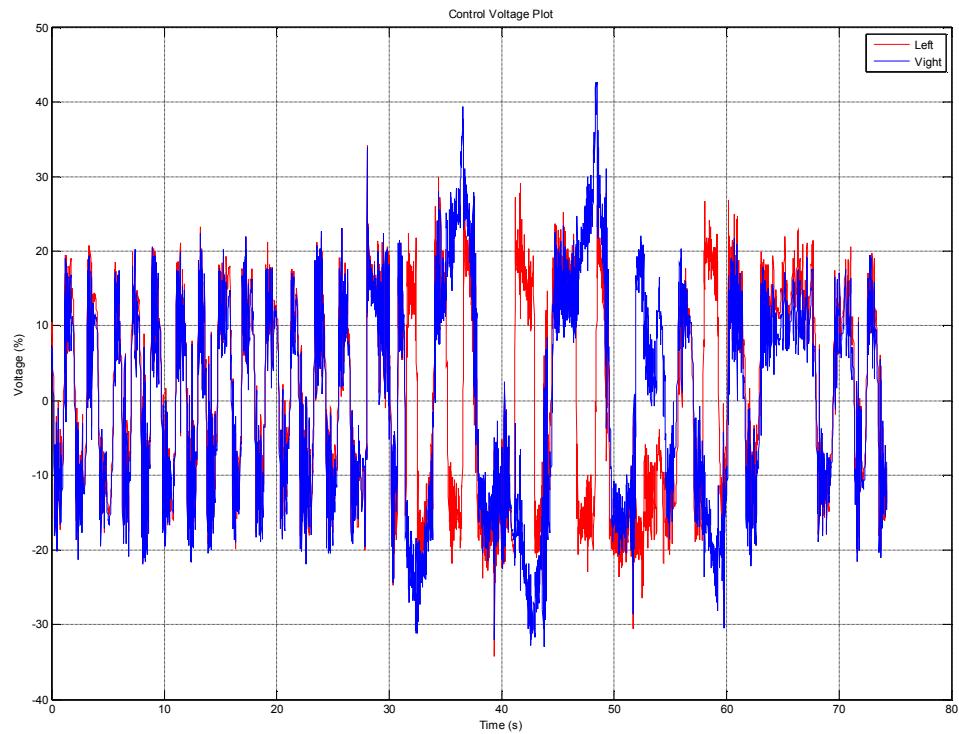
Hình 5.34 - Ngõ ra góc xoay khi robot di chuyển trên địa hình phẳng với tín hiệu đặt vị trí và vận tốc từ remote.



Hình 5.35 - Phần trặc điện áp điều khiển (24V) cho động cơ trái/phải khi robot di chuyển trên mặt phẳng

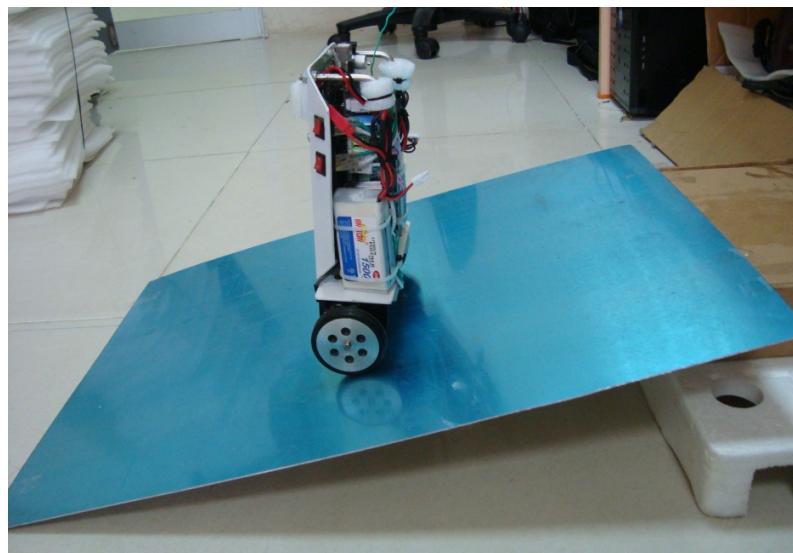


Hình 5.36 - Ngõ ra góc xoay khi robot di chuyển trên địa hình phẳng với tín hiệu đặt góc xoay từ remote.

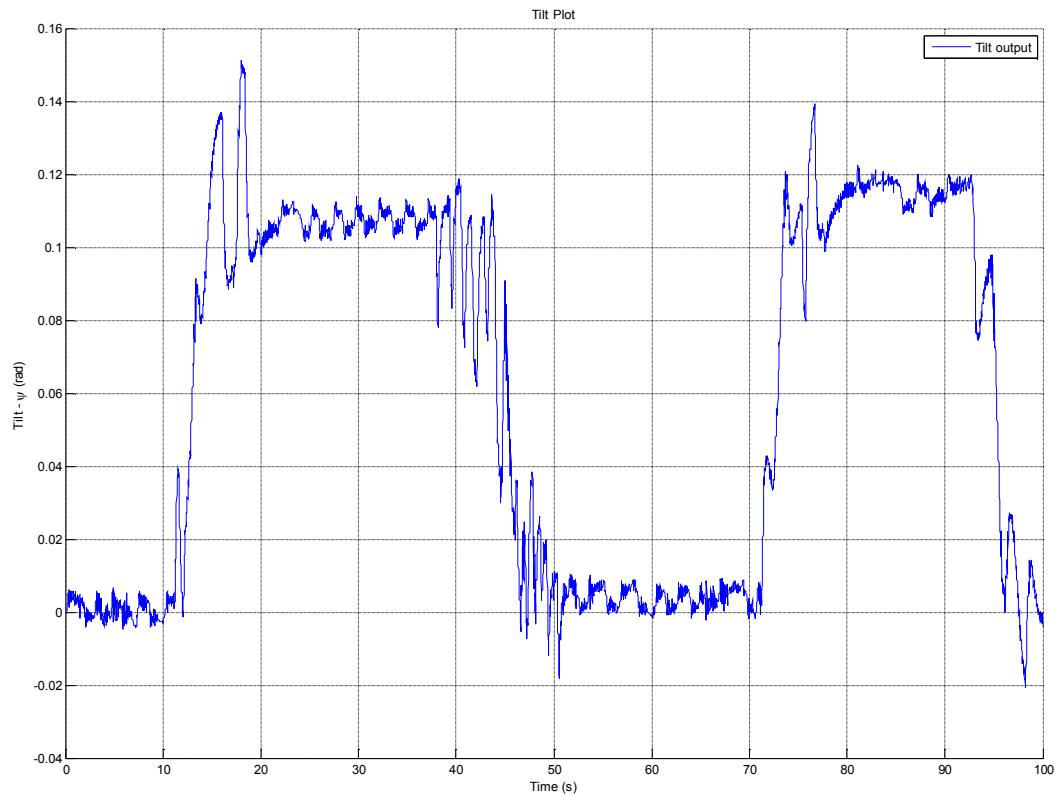


Hình 5.37 - Phần trặc điện áp điều khiển (24V) cho động cơ trái/phải khi robot di chuyển trên mặt phẳng với tín hiệu đặt là góc xoay

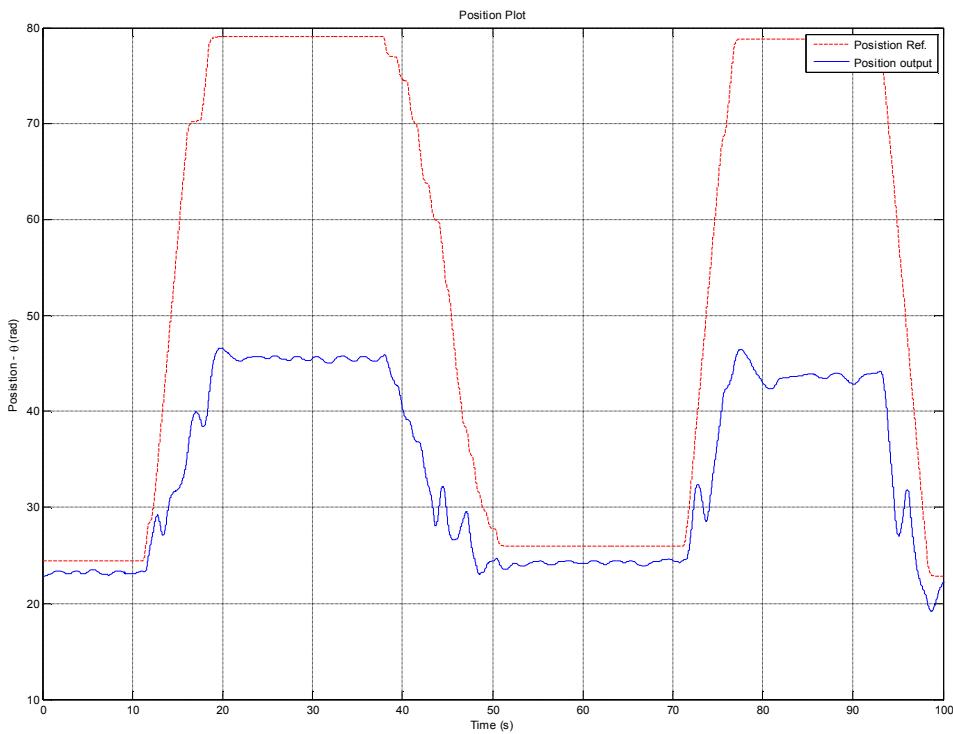
- Khi robot di chuyển trên địa hình dốc nghiêng 12.5° :



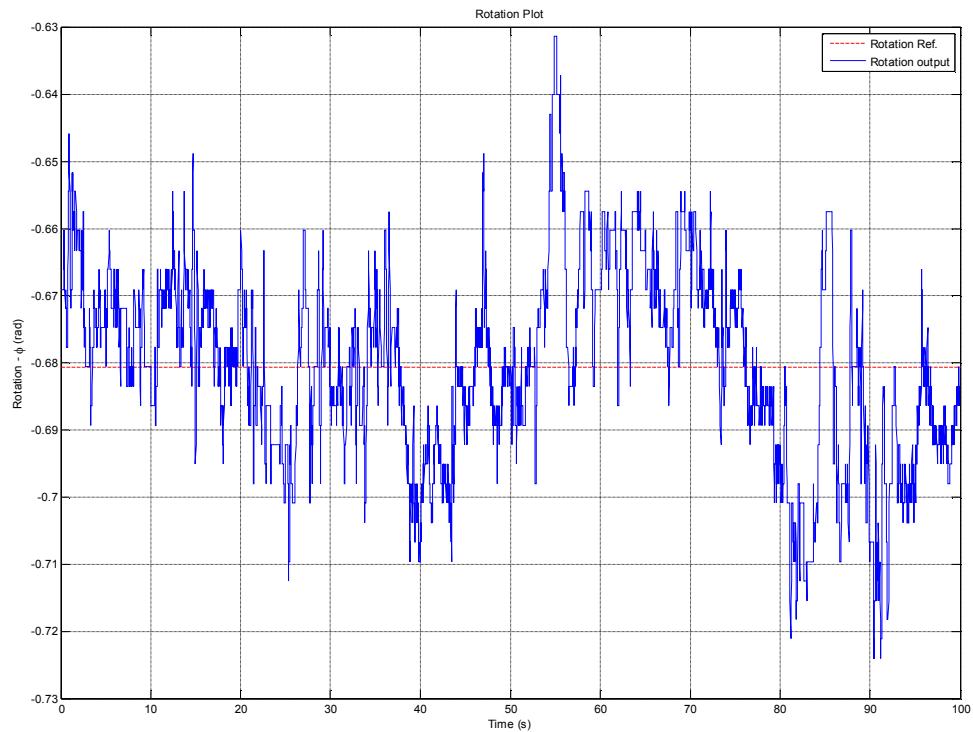
Hình 5.38 - Robot đang di chuyển trên dốc nghiêng 12.5 độ.



Hình 5.39 - Ngõ ra góc nghiêng khi robot di chuyển trên địa hình dốc nghiêng 12.5° , sai số khoảng 0.11 (rad)



Hình 5.40 - Ngõ ra vị trí khi robot di chuyển trên địa hình dốc nghiêng 12.5^0 , sai số khoảng 35 (rad)



Hình 5.41 - Ngõ ra góc xoay khi robot di chuyển trên địa hình dốc nghiêng 12.5° , sai số khoảng 0.02 (rad)

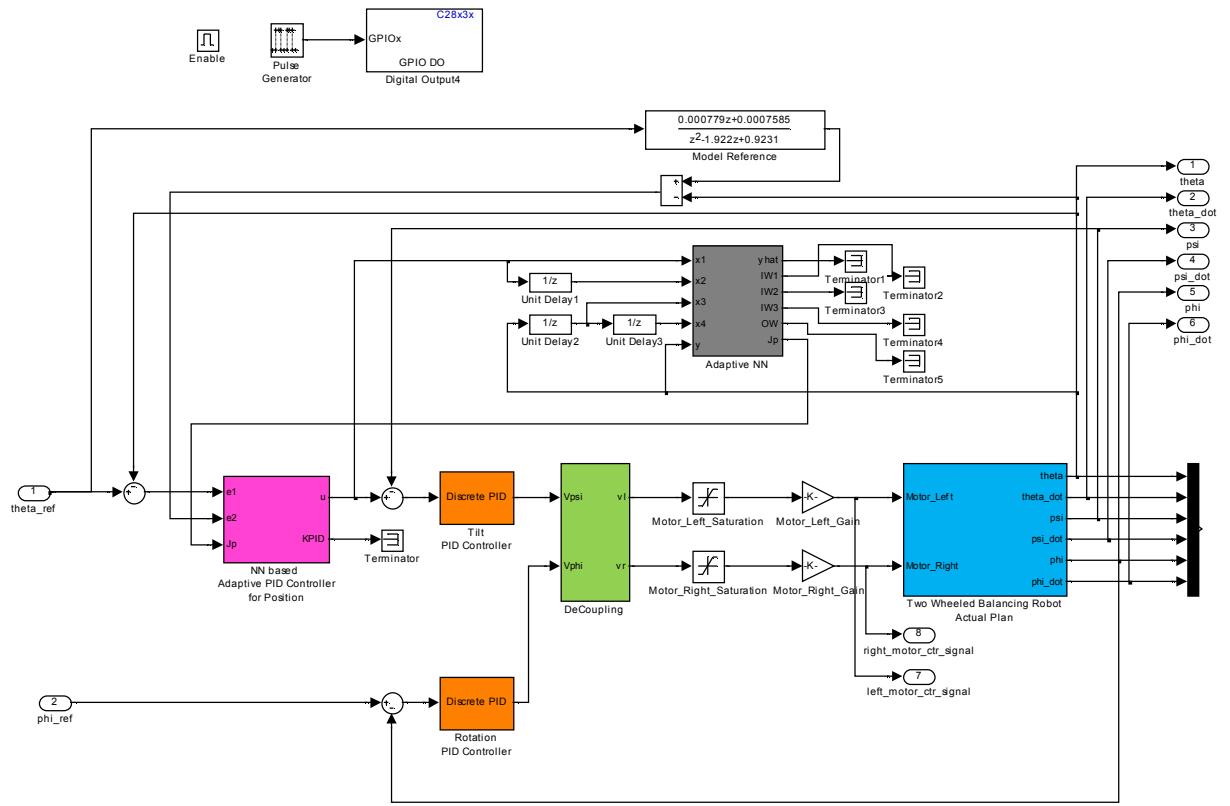
c. Nhận xét

Với LQR PI, robot đã giữ được thăng bằng và di chuyển được trên cả địa hình phẳng và dốc nghiêng đến 12.5 độ. Sai số góc nghiêng khi robot giữ thăng bằng trên mặt phẳng có thể lý giải là do robot phải nghiêng theo hướng dốc nghiêng một độ nhất định để trọng tâm robot lúc này đi qua mặt tiếp xúc giữa bánh xe và mặt phẳng. Tuy nhiên, sai số vị trí vẫn còn quá lớn, mặc dù đáp ứng vị trí tương đối tốt (có vọt lố nhưng không quá dao động), là do trọng số ảnh hưởng của vị trí trong ma trận Q và thông số khâu PI vị trí chưa tinh chỉnh hợp lý. Quá trình thực nghiệm cho thấy, thông số trên chỉ chọn trong phạm vi giới hạn, vì nếu quá lớn dễ khiến hệ thống mất ổn định.

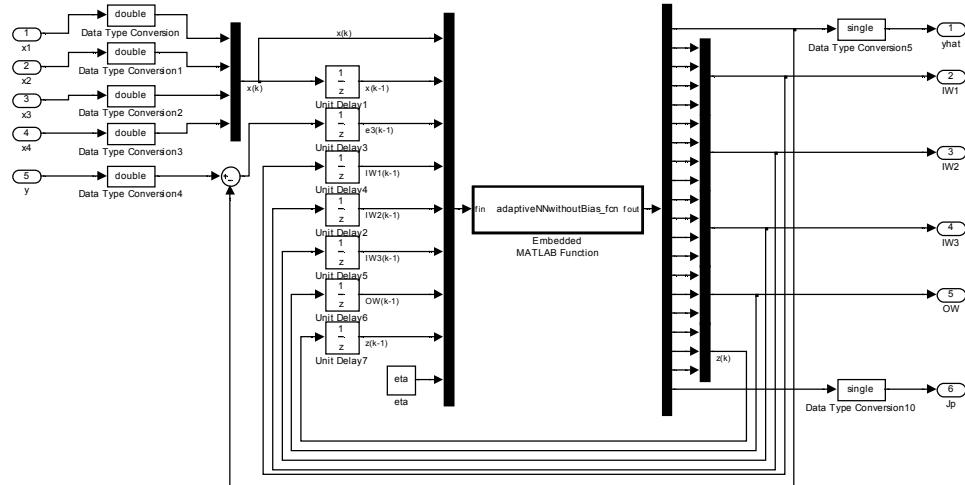
5.3.3 Bộ điều khiển PID thích nghi mô hình tham chiếu

a. Sơ đồ các khối

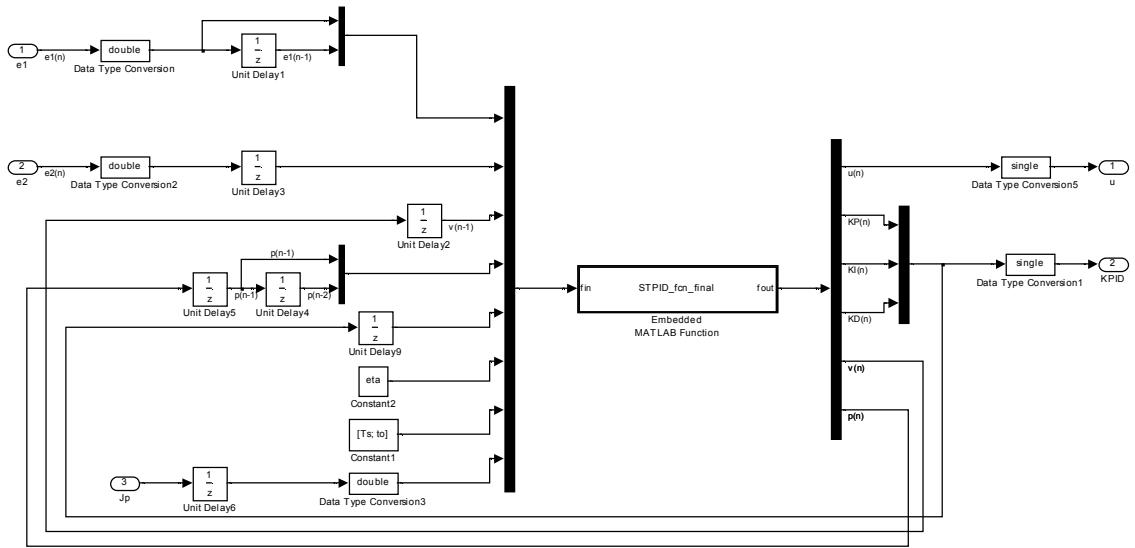
Sơ đồ các khối trong điều khiển robot sử dụng bộ PID thích nghi mô hình tham chiếu so với LQR PI chỉ khác ở khâu điều khiển di chuyển và giữ thăng bằng “Normal Operation Control” như sau:



Hình 5.42 - Bên trong khối điều khiển giữ thăng bằng và di chuyển đổi với bộ điều khiển PID thích nghi mô hình tham chiếu



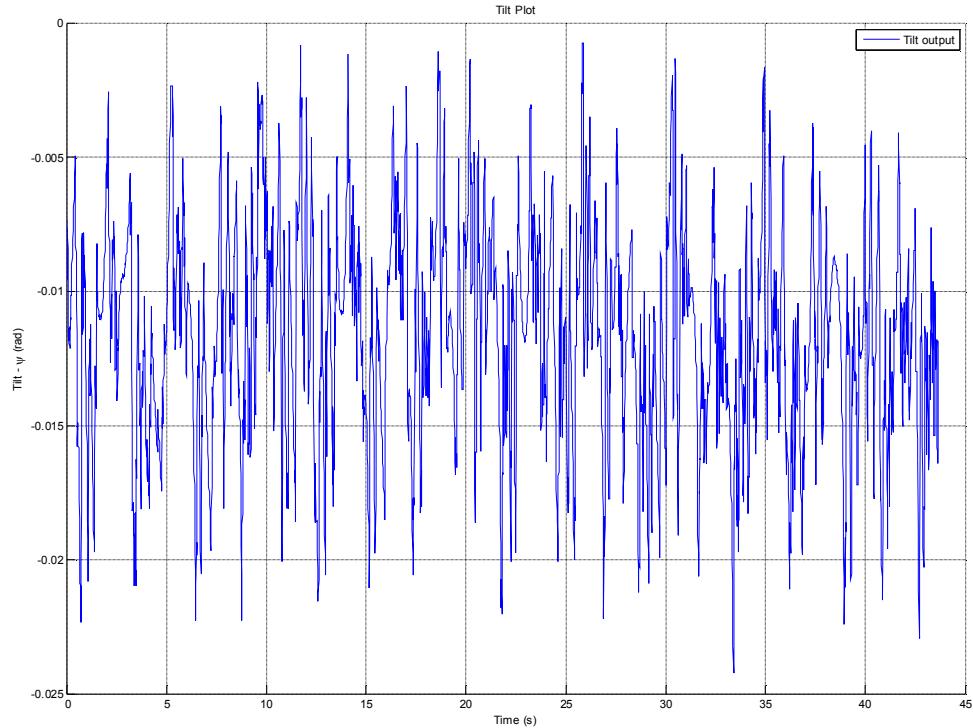
Hình 5.43 - Bên trong khối neuron nhận dạng mô hình đổi tượng sử dụng khối Embedded Matlab function



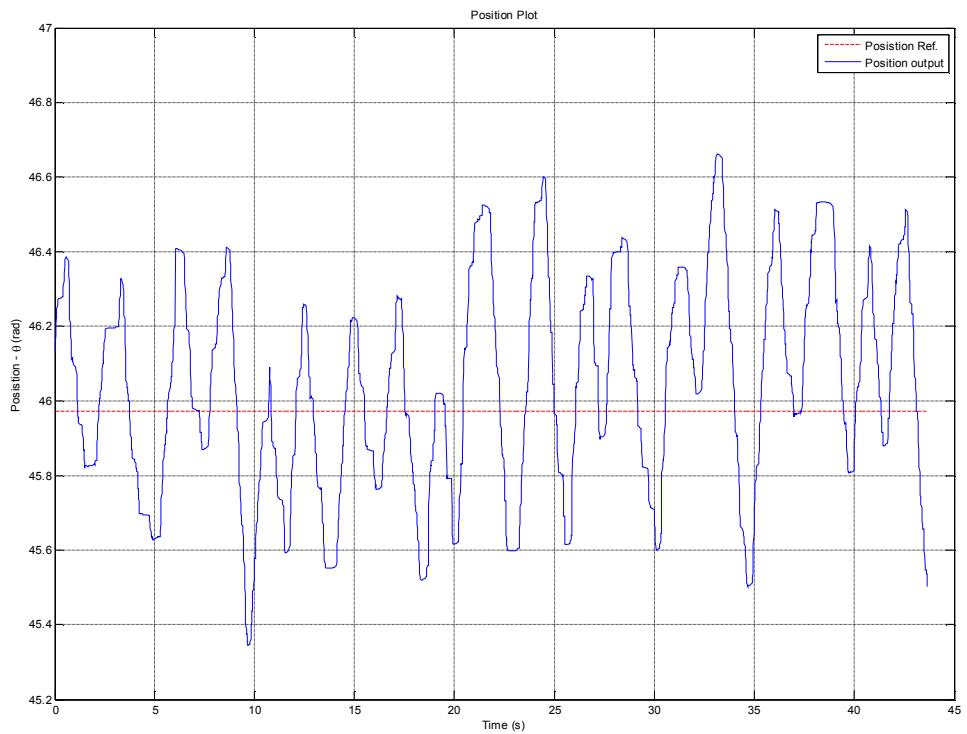
Hình 5.44 - Bên trong khối PID thích nghi cho khâu vị trí cấu trúc mạng neuron.

b. Kết quả thực nghiệm

- Khi robot giữ thăng bằng trên địa hình phẳng:

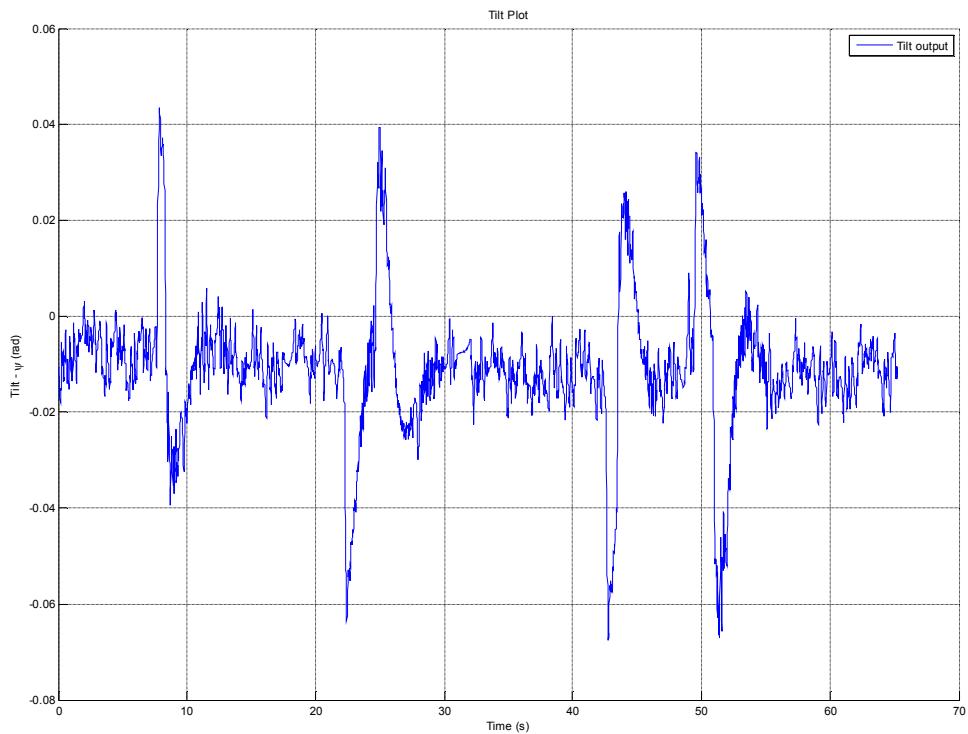


Hình 5.45 - Ngõ ra góc nghiêng trong khi giữ thăng bằng trên địa hình phẳng, sai số vào khoảng ± 0.012 (rad)

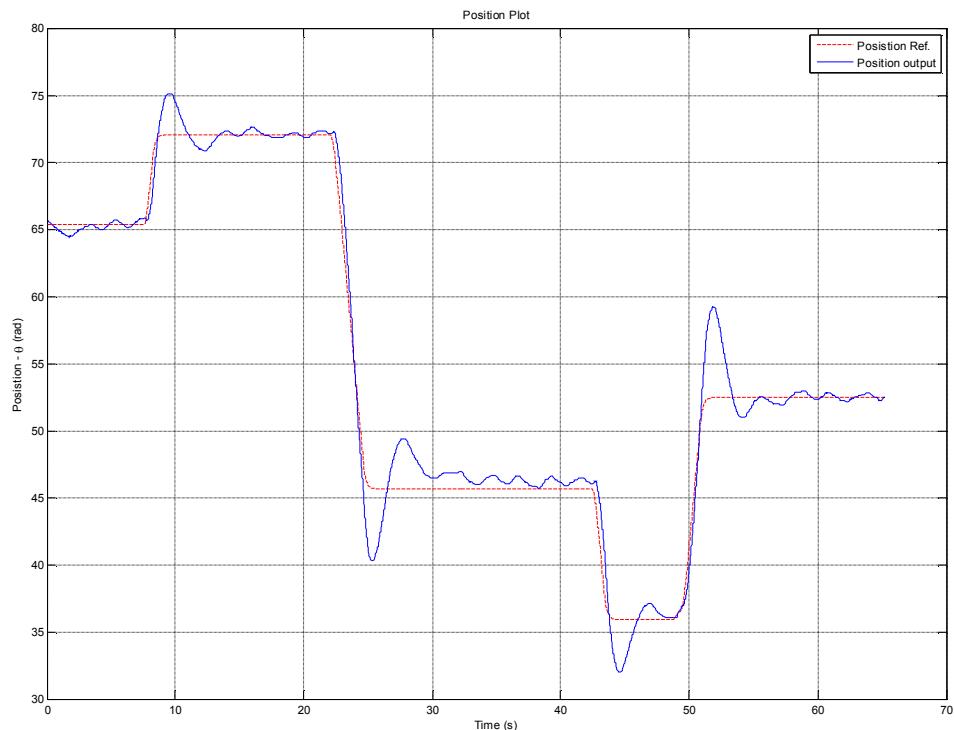


Hình 5.46 - Ngõ ra vị trí trong khi giữ thăng bằng trên địa hình phẳng, sai số vị trí trung bình khoảng 0.6 (rad)

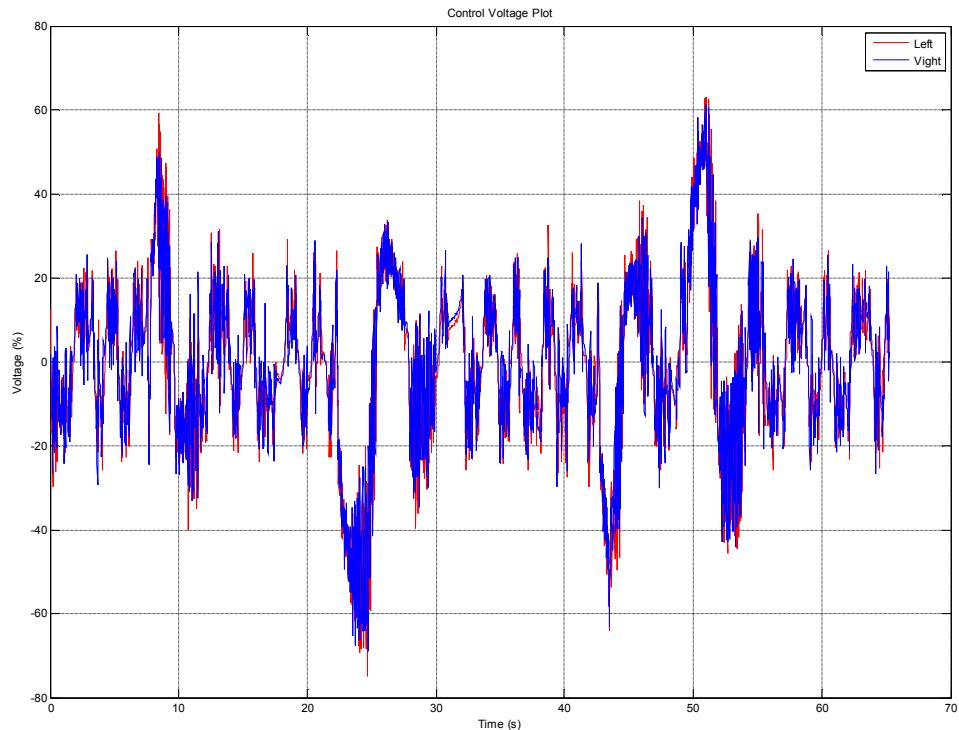
- Khi robot di chuyển trên địa hình phẳng:



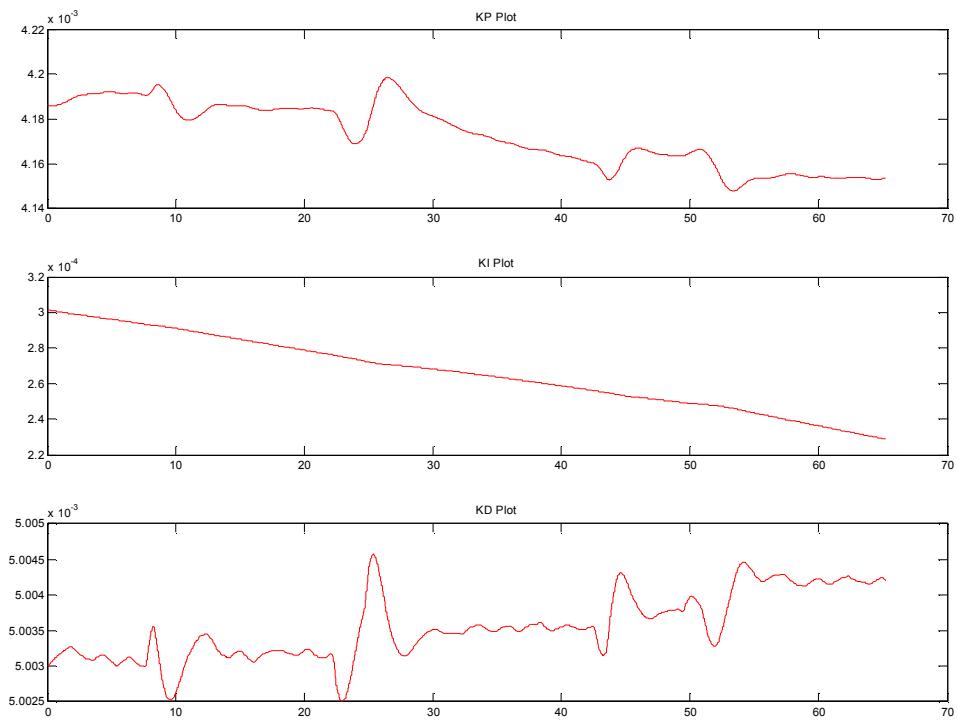
Hình 5.47 - Ngõ ra góc nghiêng khi robot di chuyển trên địa hình phẳng với tín hiệu đặt vị trí từ remote.



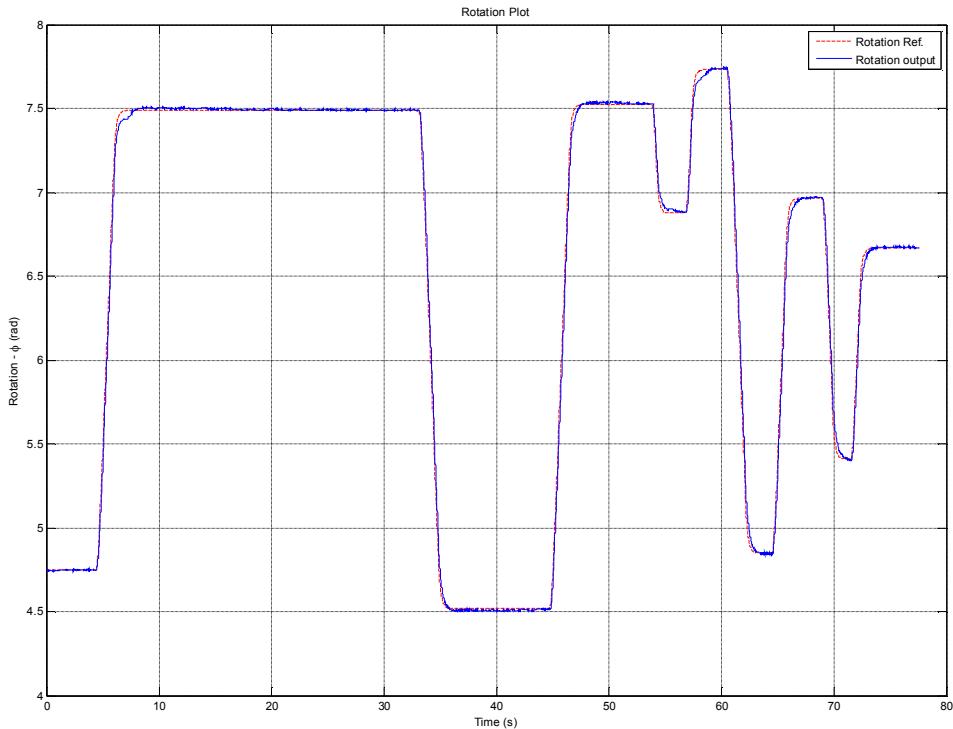
Hình 5.48 - Ngõ ra vị trí khi robot di chuyển trên địa hình phẳng với tín hiệu đặt vị trí từ remote, sai số khoảng 0.3 (rad)



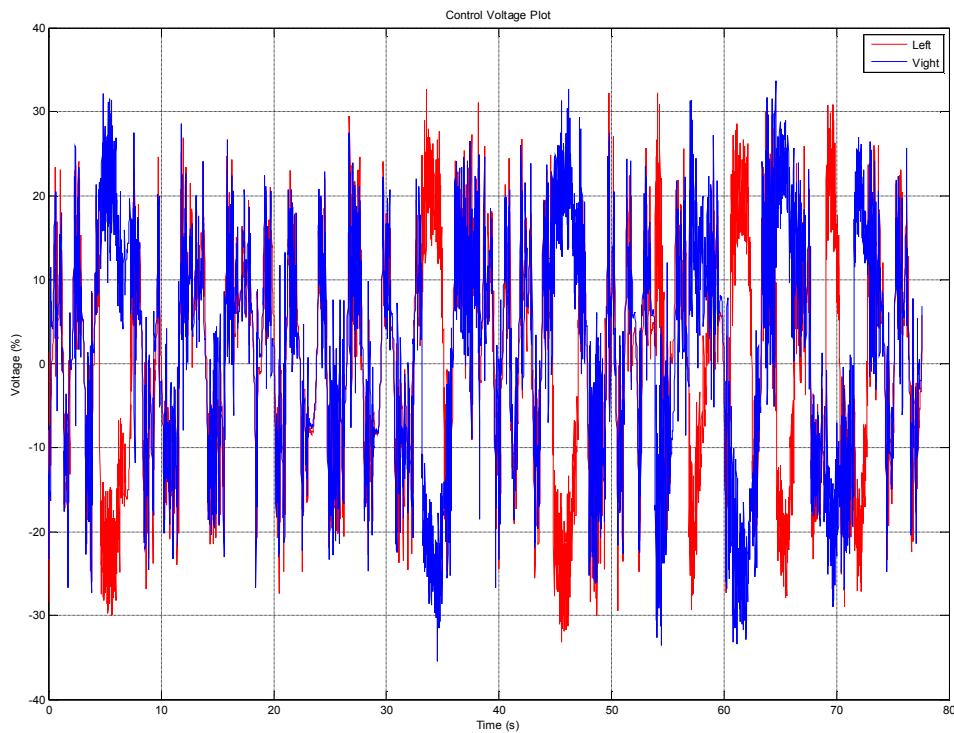
Hình 5.49 - Phần trăm điện áp điều khiển (24V) cho động cơ trái/phải khi robot di chuyển trên mặt phẳng



Hình 5.50 - Thông số PID được tự động cập nhật trong quá trình điều khiển

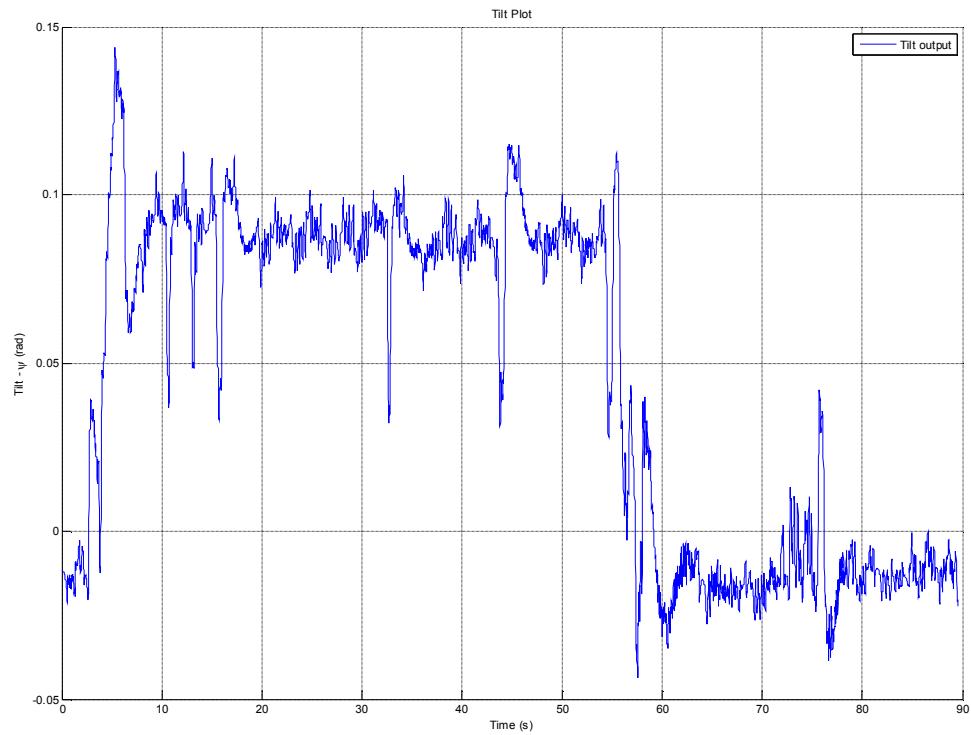


Hình 5.51 - Ngõ ra góc xoay khi robot di chuyển trên địa hình phẳng với tín hiệu đặt góc xoay từ remote.

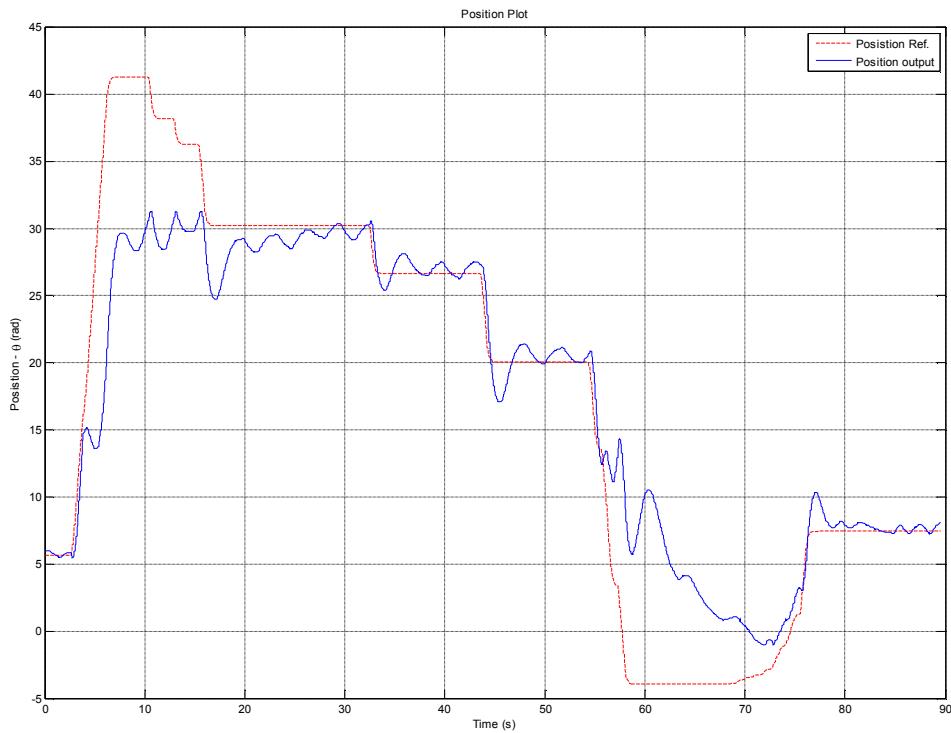


Hình 5.52 - Phần trăm điện áp điều khiển (24V) cho động cơ trái/phải khi robot di chuyển trên mặt phẳng với tín hiệu đặt là góc xoay

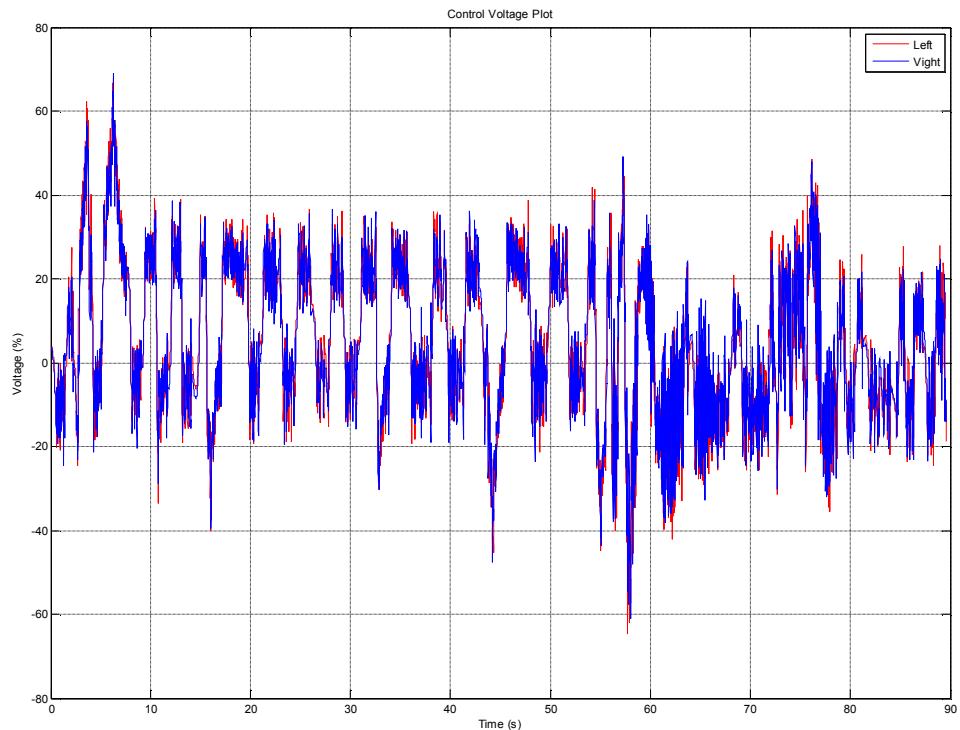
- Khi robot di chuyển trên địa hình dốc nghiêng 12.5° :



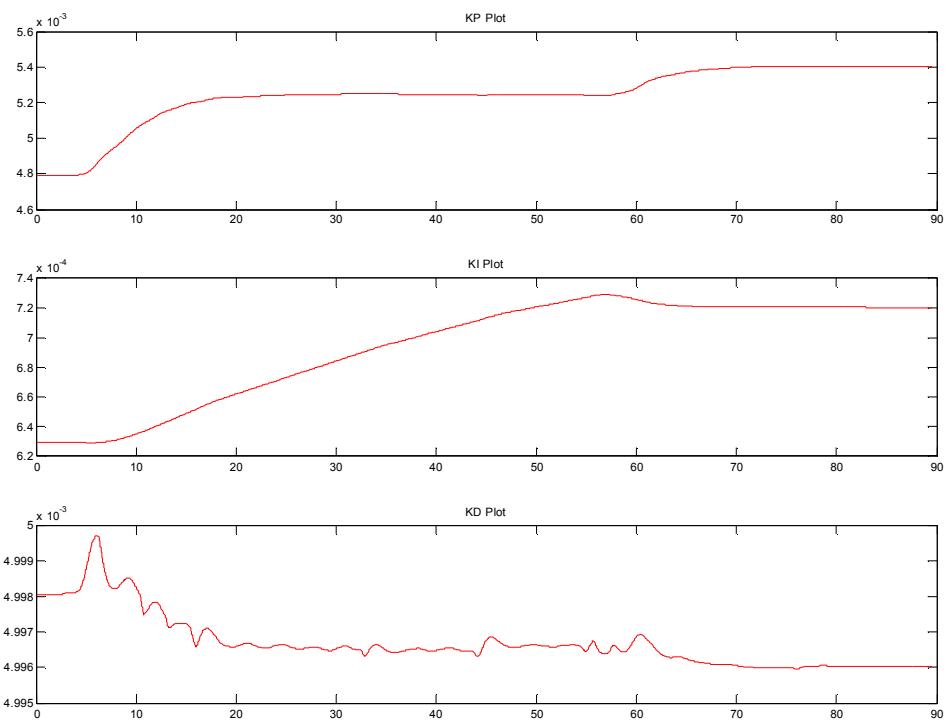
Hình 5.53 - Ngõ ra góc nghiêng khi robot di chuyển trên địa hình dốc nghiêng 12.5° , sai số khoảng 0.09 (rad)



Hình 5.54 - Ngõ ra vị trí khi robot di chuyển trên địa hình dốc nghiêng 12.5^0 , sai số khoảng 2 (rad)



Hình 5.55 - Phần trăm điện áp điều khiển (24V) cho động cơ trái/phải khi robot di chuyển trên dốc nghiêng 12.5^0 .



Hình 5.56 - Thông số PID được tự động cập nhật trong quá trình điều khiển

c. Nhận xét

Như vậy, với bộ điều khiển PID thích nghi mô hình tham chiếu trên cơ sở cấu trúc mạng neuron, đáp ứng vị trí được cải thiện một cách rõ rệt khi điều khiển robot trên địa hình phẳng và trên dốc nghiêng nhờ vào quá trình tự động cập nhật giá trị PID trong quá trình điều khiển. Tuy nhiên, việc chọn lựa các giá trị khởi tạo PID cũng như tốc độ cập nhật là rất quan trọng, ảnh hưởng đến chất lượng điều khiển cũng như tính ổn định của hệ thống.

CHƯƠNG VI

KẾT LUẬN

6.1 Kết quả đạt được

Với đề tài nghiên cứu trên, tác giả đã đạt được một số kết quả nhất định như sau:

Trong hai chương đầu, tác giả đã đưa ra mô hình toán học dựa trên nguyên lý động lực học của robot hai bánh tự cân bằng trên địa hình phẳng, nắm được đặc tính phi tuyến của mô hình đối tượng cần điều khiển.

Khi mô phỏng trong matlab simulink với nhiều giải thuật điều khiển khác nhau lên mô hình đối tượng: LQR, LQR PI, PID thông số cố định và PID thích nghi mô hình tham chiếu trên cấu trúc mạng neuron... cho kết quả tốt đã chứng tỏ khả năng điều khiển được của đối tượng. Thông qua quá trình thiết kế và mô phỏng bộ điều khiển, cho thấy: bộ điều khiển LQR thiết kế trên nguyên tắc tuyến tính hoá mô hình đối tượng tại điểm cân bằng, quá trình điều khiển nhằm đưa quỹ đạo của tất cả các biến trạng thái ngõ ra tiến về 0, khâu PI được đưa vào để giảm sai số xác lập khi điều khiển vị trí. Trong khi đó, bộ điều khiển PID với thông số cố định cũng được đưa vào khảo sát.

Cả hai bộ điều khiển trên đều đáp ứng tốt với các tín hiệu đặt khác nhau. Tuy nhiên, khi thông số mô hình thay đổi thì chất lượng điều khiển của hệ thống bị giảm đáng kể vì trên nguyên tắc, cả 2 bộ điều khiển tuyến tính chỉ làm việc tốt trong miền làm việc giới hạn vì đối tượng là phi tuyến.

Để khắc phục phần nào nhược điểm trên, bộ PID thích nghi mô hình tham chiếu cấu trúc mạng neuron được đưa ra nhằm chỉnh định thông số điều khiển nếu có thay đổi thông số mô hình đối tượng trong quá trình điều khiển. Kết quả mô phỏng với sự thay đổi hệ số ma sát đã cho kết quả khả quan, chất lượng điều khiển hệ thống được cải thiện đáng kể. Tuy nhiên, giá trị khởi tạo thông số bộ điều khiển cũng như việc chọn lựa tốc độ cập nhật các thông số trong quá trình điều khiển là rất quan trọng.

Bộ lọc Kalman đã được khảo sát và thực hiện hóa với mô hình 3 biến trạng thái, ước lượng giá trị góc nghiêng và vận tốc góc nghiêng thân robot từ 2 ngõ vào là góc nghiêng tính toán từ cảm biến gia tốc, và vận tốc góc nghiêng từ con quay hồi chuyển cho kết quả mô phỏng đạt kết quả tốt.

Vì điều khiển chuyên xử lý số tín hiệu mạnh mẽ được chọn lựa là TMS 320F28335 để hiện thực hóa bộ điều khiển cho mô hình robot thực. Với những đặc điểm mạnh về tốc độ tính toán, ngoại vi và được Matlab hỗ trợ trong Target Support Package là một ưu điểm để hiện thực hóa thiết kế trong Simulink thành bộ điều khiển nhúng trên thực tế một cách nhanh nhất và chính xác nhất.

Mô hình robot đã được thiết kế và thi công hoàn chỉnh, nhỏ gọn (cao 185mm, nặng 0.92kg) chất lượng và có tính thẩm mĩ, với một số tùy chọn để quá trình điều khiển robot được linh hoạt (RF hoặc bluetooth).

Tất cả các giải thuật điều khiển trong quá trình thiết kế mô phỏng được hiện thực hóa thành bộ điều khiển nhúng để điều khiển robot. Tất cả các bộ điều khiển trong quá trình thực nghiệm đều cho thấy khả năng giữ thăng bằng robot tốt trên địa hình phẳng hoặc cả dốc nghiêng (đến 12.5 độ), tuy nhiên, đáp ứng vị trí của robot không còn tốt trên địa hình dốc nghiêng khi áp dụng LQR hoặc PID thông số cố định. Bộ điều khiển sử dụng PID thích nghi mô hình tham chiếu cấu trúc mạng neuron đã khắc phục đáng kể nhược điểm đó.

Tuy vậy, do thời gian thực hiện có hạn nên trong luận văn còn nhiều hạn chế.

6.2 Một số hạn chế

Đối tượng robot hai bánh tự cân bằng trên địa hình không phẳng chưa được mô hình hóa một cách cụ thể bằng toán học (hiện tại vẫn xem thông số sai biệt này là chưa biết trong quá trình điều khiển)

Nhiều địa hình khác nhau như: gồ ghề, nhấp nhô, lượn sóng, bề mặt nhám hay bám dính cao... chưa được khảo sát cụ thể và mô hình hóa vì tính phức tạp của chúng.

Mô hình thực tế so với mô phỏng vẫn còn nhiều sai số (như moment quán tính không đều, độ rơ hộp số trên motor bánh xe...) chưa được mô hình hóa cụ thể, dẫn đến kết quả điều khiển trên thực tế so với mô phỏng có khoảng cách khá lớn.

Nhiều phương pháp điều khiển phi tuyến, thông minh chưa được đưa ra khảo sát và thiết kế để điều khiển robot (tính đến thời điểm này, chỉ có một phương pháp phi tuyến đang được khảo sát là trượt PI – tuy nhiên chưa hiện thực hóa trên mô hình thực nên chưa đưa ra).

Robot chưa có khả năng tự kiểm soát hay đưa ra quyết định, như: khả năng tự di chuyển, vượt chướng ngại vật, tự đứng dậy (swing-up), khả năng quản lý năng lượng để ổn định quá trình điều khiển trong thời gian dài...

Như vậy, tác giả xin đưa ra một số hướng phát triển để hoàn thiện đề tài trong tương lai như sau.

6.3 Hướng phát triển

Khảo sát và xây dựng hoàn chỉnh đặc tính động lực học của mô hình đối tượng trong những điều kiện địa hình khác nhau.

Xây dựng giải thuật thông minh để nhận dạng những thông số không mô hình hoá được như đã đề cập trên.

Khảo sát và xây dựng nhiều giải thuật điều khiển dựa trên cơ sở phi tuyến hơn nữa, kết hợp với bộ nhận dạng, bộ bù trên cơ sở điều khiển thông minh, thích nghi để giúp cho robot có thể di chuyển được và đáp ứng ngõ ra chính xác hơn trong nhiều điều kiện khác nhau.

Khảo sát và xây dựng khả năng swing-up , khả năng tự hành của robot để giúp cho robot có thể hoạt động độc lập hơn, khả năng ứng dụng sẽ cao hơn nữa.