Modelagem de Sistemas Dinâmicos - Trabalho Nº3

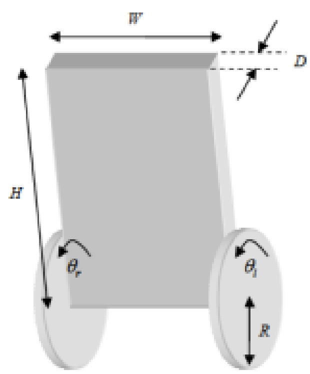
Leonardo Soares da Costa Tanaka - DRE: 121067652

Engenharia de Controle e Automação/UFRJ

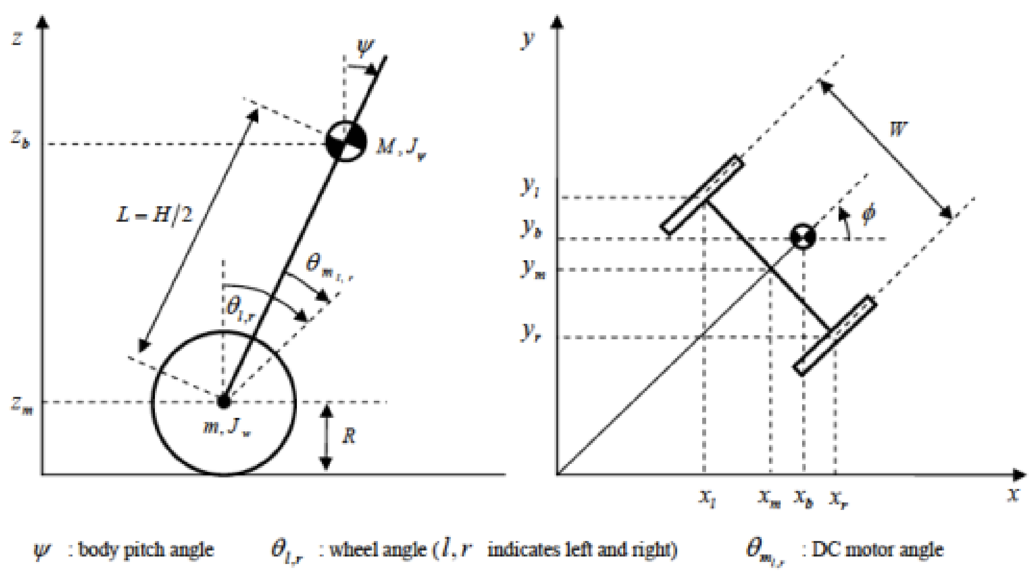
Rio de Janeiro, Brasil

Julho de 2023

Considerando um sistema de pêndulo invertido montado numa plataforma de 2 rodas:



A vista lateral e superior deste sistema com as variáveis associadas é mostrada na seguinte figura:



Os parâmetros físicos deste sistema são dados por:

- $g = 9.8\text{m/s}^2$: gravidade
- $m = 0.03\text{kg}$: peso da roda
- $R = 0.04\text{m}$: raio da roda
- $J_W = \frac{mR^2}{2}$: momento de inércia da roda
- $M = 0.6\text{kg}$: peso do corpo
- $W = 0.14\text{m}$: largura do corpo
- $D = 0.04\text{m}$: profundidade do corpo
- $H = 0.144\text{m}$: altura do corpo
- $L = \frac{H}{2}$: distância do centro de massa do corpo ao eixo da roda
- $J_\psi = \frac{ML^2}{3}\text{kgm}^2$: momento de inércia do corpo em pitch
- $J_\phi = \frac{M(W^2+D^2)}{12}\text{kgm}^2$: momento de inércia do corpo em rumo
- J_m: inércia do motor DC desprezível
- $R_m = 6.69\Omega$: resistência do motor DC
- L_m: indutância do motor desprezível
- $K_t = Ke = 0.4$: constante de torque (Nm/A) e constante de EMF (V s/rad)
- $f_m = 0.0022$: atrito entre o corpo e o motor
- $n = 1$: redução do motor

Considerando os graus de liberdade $\{\theta, \psi, \phi\}$ onde $\theta = (\theta_l + \theta_r)/2 \to \dot{\theta} = (\dot{\theta}_l + \dot{\theta}_r)/2$.

A posição cartesiana do carrinho (composto pelas 2 rodas), no sistema de coordenadas $\{x, y, x\}$, é dada por:

$$X_m = \begin{bmatrix} x_m \\ y_m \\ z_m \end{bmatrix} = \begin{bmatrix} x_m \\ y_m \\ R \end{bmatrix} \tag{1}$$

Considerando que as rodas do carrinho não escorregam, o carrinho é um sistema com restrições não-holonômicas, sendo que a velocidade do carrinho é dada por:

$$\dot{X_m} = R \begin{bmatrix} cos(\phi) \\ sin(\phi) \\ 0 \end{bmatrix} \dot{\theta} \tag{2}$$

A posição cartesiana das rodas (esquerda e direita) no sistema de coordenadas $\{x, y, x\}$, é dada por:

$$
\begin{aligned}
X_l = \begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} = X_m + \frac{W}{2} \begin{bmatrix} -sin(\phi) \\ cos(\phi) \\ 0 \end{bmatrix} \\
X_r = \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} = X_m + \frac{W}{2} \begin{bmatrix} sin(\phi) \\ -cos(\phi) \\ 0 \end{bmatrix}
\end{aligned} \tag{3}
$$

A velocidade cartesiana das rodas é dada por:

$$
\begin{aligned}
\dot{X}_l = \dot{X}_m + \frac{W}{2} \begin{bmatrix} -cos(\phi) \\ -sin(\phi) \\ 0 \end{bmatrix} \dot{\phi} \\
\dot{X}_r = \dot{X}_m + \frac{W}{2} \begin{bmatrix} cos(\phi) \\ sin(\phi) \\ 0 \end{bmatrix} \dot{\phi}
\end{aligned} \tag{4}
$$

A posição cartesiana do corpo no sistema de coordenadas $\{x, y, x\}$, é dada por:

$$
X_b = \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} = X_m + L \begin{bmatrix} cos(\phi)sin(\psi) \\ sin(\phi)sin(\psi) \\ cos(\psi) \end{bmatrix} \tag{5}
$$

A velocidade cartesiana do corpo é dada por:

$$
\dot{X}_b = \dot{X}_m + L \begin{bmatrix} cos(\phi)cos(\psi) \\ sin(\phi)cos(\psi) \\ -sin(\psi) \end{bmatrix} \dot{\psi} + L \begin{bmatrix} -sin(\phi)sin(\psi) \\ cos(\phi)sin(\psi) \\ 0 \end{bmatrix} \dot{\phi} \tag{6}
$$

Levando em consideração que a massa do motor DC é desprezível e utilizando a fórmula de energia cinética translacional de um único corpo, a energia cinética translacional é dada por:

$$
K_{tra} = \frac{1}{2} m \|\vec{v}\|^2 \tag{7}
$$

$$
\begin{gathered}
T_1 = \frac{1}{2}m\dot{X_l^T}\dot{X}_l + \frac{1}{2}m\dot{X_r^T}\dot{X}_r + \frac{1}{2}M\dot{X_b^T}\dot{X}_b = \\
= \frac{1}{2}m\begin{bmatrix} Rcos(\phi)\dot{\theta} - \frac{W}{2}cos(\phi)\dot{\phi} & Rsin(\phi)\dot{\theta} - \frac{W}{2}sin(\phi)\dot{\phi} & 0 \end{bmatrix} \begin{bmatrix} Rcos(\phi)\dot{\theta} - \frac{W}{2}cos(\phi)\dot{\phi} \\ Rsin(\phi)\dot{\theta} - \frac{W}{2}sin(\phi)\dot{\phi} \\ 0 \end{bmatrix} + \\
\frac{1}{2}m\begin{bmatrix} Rcos(\phi)\dot{\theta} + \frac{W}{2}cos(\phi)\dot{\phi} & Rsin(\phi)\dot{\theta} + \frac{W}{2}sin(\phi)\dot{\phi} & 0 \end{bmatrix} \begin{bmatrix} Rcos(\phi)\dot{\theta} + \frac{W}{2}cos(\phi)\dot{\phi} \\ Rsin(\phi)\dot{\theta} + \frac{W}{2}sin(\phi)\dot{\phi} \\ 0 \end{bmatrix} + \\
\tfrac{1}{2}M\begin{bmatrix} Rcos(\phi)\dot{\theta} + Lcos(\phi)cos(\psi)\dot{\psi} - Lsin(\phi)sin(\psi)\dot{\phi} & Rsin(\phi)\dot{\theta} + Lsin(\phi)cos(\psi)\dot{\psi} + Lcos(\phi)sin(\psi)\dot{\phi} & -Lsin(\psi)\dot{\psi} \end{bmatrix} \begin{bmatrix} Rcos(\phi)\dot{\theta} + Lcos(\phi)cos(\psi)\dot{\psi} - Lsin(\phi)sin(\psi)\dot{\phi} \\ Rsin(\phi)\dot{\theta} + Lsin(\phi)cos(\psi)\dot{\psi} + Lcos(\phi)sin(\psi)\dot{\phi} \\ -Lsin(\psi)\dot{\psi} \end{bmatrix} = \\
= \frac{1}{2}m\left[R^2cos(\phi)^2\dot{\theta}^2 - WRcos(\phi)^2\dot{\theta}\dot{\phi} + \frac{W^2}{4}cos(\phi)\dot{\phi}^2 + R^2sin(\phi)^2\dot{\theta}^2 - WRsin(\phi)^2\dot{\theta}\dot{\phi} + \frac{W^2}{4}sin(\phi)\dot{\phi}^2\right] + \\
\frac{1}{2}m\left[R^2cos(\phi)^2\dot{\theta}^2 + WRcos(\phi)^2\dot{\theta}\dot{\phi} + \frac{W^2}{4}cos(\phi)\dot{\phi}^2 + R^2sin(\phi)^2\dot{\theta}^2 + WRsin(\phi)^2\dot{\theta}\dot{\phi} + \frac{W^2}{4}sin(\phi)\dot{\phi}^2\right] + \\
\tfrac{1}{2}M\left[R^2cos(\phi)^2\dot{\theta}^2 + L^2cos(\phi)^2cos(\psi)^2\dot{\psi}^2 + L^2sin(\phi)^2sin(\psi)^2\dot{\phi}^2 + 2RLcos(\phi)^2cos(\psi)\dot{\theta}\dot{\psi} - 2L^2sin(\phi)cos(\phi)sin(\psi)\dot{\phi}\dot{\psi} - 2RLsin(\phi)cos(\phi)sin(\psi)\dot{\theta}\dot{\phi}\right] + \\
\tfrac{1}{2}M\left[R^2sin(\phi)^2\dot{\theta}^2 + L^2sin(\phi)^2cos(\psi)^2\dot{\psi}^2 + L^2cos(\phi)^2sin(\psi)^2\dot{\phi}^2 + 2RLsin(\phi)^2cos(\psi)\dot{\theta}\dot{\psi} + 2L^2sin(\phi)cos(\phi)sin(\psi)\dot{\phi}\dot{\psi} + 2RLsin(\phi)cos(\phi)sin(\psi)\dot{\theta}\dot{\phi}\right] + \\
\frac{1}{2}M = L^2sin(\psi)^2\dot{\psi}^2 = \\
= \frac{m\left[R^2\dot{\theta}^2 - WR\dot{\theta}\dot{\phi} + \frac{W^2}{4}\dot{\phi}^2 + R^2\dot{\theta}^2 + WR\dot{\theta}\dot{\phi} + \frac{W^2}{4}\dot{\phi}^2\right] + M\left[R^2\dot{\theta}^2 + L^2cos(\psi)^2\dot{\psi}^2 + L^2sin(\psi)^2\dot{\phi}^2 + 2RLcos(\psi)\dot{\theta}\dot{\psi} + L^2sin(\psi)^2\dot{\psi}^2\right]}{2} = \\
= \frac{m\left[2R^2\dot{\theta}^2 + \frac{W^2}{2}\dot{\phi}^2\right] + M\left[R^2\dot{\theta}^2 + L^2\dot{\psi}^2 + L^2sin(\psi)^2\dot{\phi}^2 + 2RLcos(\psi)\dot{\theta}\dot{\psi}\right]}{2}
\end{gathered} \tag{8}
$$

Utilizando a fórmula da energia cinética rotacional:

$$
\begin{gathered}
K_{rot} = \frac{1}{2}\vec{\omega} \cdot \mathbf{I} \cdot \vec{\omega} \\
T_2 = \frac{J_\phi \dot{\phi}^2}{2} + \frac{J_\psi \dot{\psi}^2}{2} + \frac{J_w(\dot{\theta}_r^2 + \dot{\theta}_l^2)}{2} + \frac{J_m n^2(\dot{\theta}_l - \dot{\psi})^2}{2} + \frac{J_m n^2(\dot{\theta}_r - \dot{\psi})^2}{2}
\end{gathered} \tag{9}
$$

Considerando que $\dot{\theta} = (\dot{\theta}_l + \dot{\theta}_r)/2$, temos que:

$$
\begin{gathered}
\dot{\theta}_r^2 + \dot{\theta}_l^2 = 2\dot{\theta}^2 + \frac{W^2}{2R^2}\dot{\phi}^2, \quad (\dot{\theta}_l - \dot{\psi})^2 + (\dot{\theta}_r - \dot{\psi})^2 = 2(\dot{\theta} - \dot{\psi})^2 + \frac{W^2}{2R^2}\dot{\phi}^2 \\
\dot{\phi} = \frac{R}{W}\left(\dot{\theta}_r - \dot{\theta}_l\right)
\end{gathered} \tag{10}
$$

Levando em consideração que J_m é desprezível e utilizando as fórmulas acima, então:

$$
\begin{gathered}
T_2 = J_w\dot{\theta}^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}\left(J_\phi + \frac{W^2}{2R^2}J_w + \frac{W^2}{2R^2}n^2J_m\right)\dot{\phi}^2 + n^2J_m(\dot{\theta} - \dot{\psi})^2 \\
T_2 = J_w\dot{\theta}^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}\left(J_\phi + \frac{W^2}{2R^2}J_w\right)\dot{\phi}^2
\end{gathered} \tag{11}
$$

A energia potencial do sistema pode ser descrito pela soma das energias potenciais dos centros de massa das rodas e do corpo, tal que:

$$U = mgz_l + mgz_r + mgz_b = 2mgR + Mg(R + Lcos(\psi)) \tag{12}$$

1 Derivação e obtenção das equações dinâmicas

Levando em consideração que o Lagrangiano do sistema pode ser descrito pela seguinte equação e os valores de T_1, T_2 e U:

$$\begin{gathered} L = T_1 + T_2 - U \\ L = \frac{m\left[2R^2\dot{\theta}^2 + \frac{W^2}{2}\dot{\phi}^2\right] + M\left[R^2\dot{\theta}^2 + L^2\dot{\psi}^2 + L^2 sin(\psi)^2\dot{\phi}^2 + 2RLcos(\psi)\dot{\theta}\dot{\psi}\right]}{2} + J_w\dot{\theta}^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}\left(J_\phi + \frac{W^2}{2R^2}J_w\right)\dot{\phi}^2 - 2mgR - Mg(R + Lcos(\psi)) \end{gathered} \tag{13}$$

Temos que:

$$\begin{gathered} \frac{\partial L}{\partial \theta} = 0,\ \frac{\partial L}{\partial \psi} = ML^2 sin(\psi)cos(\psi)\dot{\phi}^2 - MRLsin(\psi)\dot{\theta}\dot{\psi} + MgLsin(\psi),\ \frac{\partial L}{\partial \phi} = 0 \\ \frac{\partial L}{\partial \dot{\theta}} = 2mR^2\dot{\theta} + MR^2\dot{\theta} + 2J_w\dot{\theta} + MRLcos(\psi)\dot{\theta} \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) = 2mR^2\ddot{\theta} + MR^2\ddot{\theta} + 2J_w\ddot{\theta} - MRLsin(\psi)\dot{\psi}^2 + MRLcos(\psi)\ddot{\psi} \\ \frac{\partial L}{\partial \dot{\psi}} = ML^2\dot{\psi} + MRLcos(\psi)\dot{\theta} + j_\psi\dot{\psi} \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\psi}}\right) = ML^2\ddot{\psi} + j_\psi\ddot{\psi} - MRLsin(\psi)\dot{\theta}\dot{\psi} + MRLcos(\psi)\ddot{\theta} \\ \frac{\partial L}{\partial \dot{\phi}} = m\frac{W^2}{2}\dot{\phi} + ML^2 sin(\psi)^2\dot{\phi} + \left(J_\phi + \frac{W^2}{2R^2}J_w\right)\dot{\phi} \\ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\phi}}\right) = m\frac{W^2}{2}\ddot{\phi} + ML^2 sin(\psi)^2\ddot{\phi} + 2ML^2 sin(\psi)cos(\psi)\dot{\psi}\dot{\phi} + \left(J_\phi + \frac{W^2}{2R^2}J_w\right)\ddot{\phi} \end{gathered} \tag{14}$$

(a) Utilizando a formulação de Lagrange foi determinado a dinâmica entre as coordenadas generalizadas $\lambda = \begin{bmatrix}\theta & \psi & \phi\end{bmatrix}^T$ e as forças (torques) generalizadas $F = \begin{bmatrix}F_\theta & F_\psi & F_\phi\end{bmatrix}^T$.

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\lambda}}\right) - \frac{\partial L}{\partial \lambda} = F \tag{15}$$

Utilizando a fórmula acima e outras equações (14) para começar a montar a forma matricial:

$$\begin{gathered} \left[(2m + M)R^2 + J_w\right]\ddot{\theta} + MRLcos(\psi)\ddot{\psi} - MRLsin(\psi)\dot{\psi}^2 = F_\theta \\ MRLcos(\psi)\ddot{\theta} + (ML^2 + J_\psi)\ddot{\psi} - ML^2 sin(\psi)cos(\psi)\dot{\phi}^2 - MgLsin(\psi) = F_\psi \\ \left[m\frac{W^2}{2} + ML^2 sin(\psi)^2 + J_\phi + \frac{W^2}{2R^2}J_w\right]\ddot{\phi} + 2ML^2 sin(\psi)cos(\psi)\dot{\psi}\dot{\phi} = F_\phi \end{gathered} \tag{16}$$

Esta dinâmica pode ser escrita em forma matricial como:

$$M(\lambda)\ddot{\lambda} + C(\lambda, \dot{\lambda}) + G(\lambda) = F \tag{17}$$

onde $M(\cdot) \in \mathbb{R}^{3\times3},\ C(\cdot,\cdot) \in \mathbb{R}^3,\ G(\cdot) \in \mathbb{R}^3$.

Utilizando as equações (16) em forma matricial:

$$\begin{bmatrix} (2m+M)R^2 + 2J_w & MRLcos(\psi) & 0 \\ MRLcos(\psi) & ML^2 + J_\psi & 0 \\ 0 & 0 & m\frac{W^2}{2} + ML^2 sin(\psi)^2 + J_\phi + \frac{W^2}{2R^2}J_w \end{bmatrix}\ddot{\lambda} + \begin{bmatrix} -MRLsin(\psi)\dot{\psi}^2 \\ -ML^2 sin(\psi)cos(\psi)\dot{\psi}^2 \\ 2ML^2 sin(\psi)cos(\psi)\dot{\psi}\dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ -MgLsin(\psi) \\ 0 \end{bmatrix} = F \tag{18}$$

(b) Determinando a relação entre as forças generalizadas e os torques de cada um das rodas $\{Fl, Fr\}$:

$$F = \begin{bmatrix} F_\theta \\ F_\psi \\ F_\phi \end{bmatrix} = E \begin{bmatrix} F_l \\ F_r \end{bmatrix} \tag{19}$$

onde $E \in \mathbb{R}^{3\times2}$

Observando o sistema de pêndulo invertido e os graus de liberdade, é possível determinar as seguintes relações:

$$\begin{gathered} F_\theta = F_l + F_r \\ F_\psi = -F_l - F_r \\ F_\phi = -\frac{W}{2R}F_l + \frac{W}{2R}F_r \\ F = \begin{bmatrix} F_\theta \\ F_\psi \\ F_\phi \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ -\frac{W}{2R} & \frac{W}{2R} \end{bmatrix} \begin{bmatrix} F_l \\ F_r \end{bmatrix} \end{gathered} \tag{20}$$

(c) Determinando a relação entre as forças (torques) generalizadas $\{F_\theta, F_\psi, F_\phi\}$ e a tensão elétrica aplicada em cada motor $\{Vl, Vr\}$ utilizando o modelo de um motor DC:

$$F = Q\dot{\lambda} + H \begin{bmatrix} V_l \\ V_r \end{bmatrix} \tag{21}$$

onde $Q \in \mathbb{R}^{3\times3}$ e $H \in \mathbb{R}^{3\times2}$.

Levando em consideração a Lei de Kirchhoff das Tensões, a Equação do Torque, a Lei do Motor e a Lei do Gerador. Além disso, o atrito entre o corpo e o motor e o indutância do motor DC L_m é desprezível:

$$
\begin{aligned}
F_{l,r} &= nK_t i_{l,r} + f_m(\dot{\psi} - \dot{\theta}_{l,r}) \\
V_{l,r} + K_e(\dot{\psi} - \dot{\theta}_{l,r}) &= L_m \dot{i}_{l_r} + R_m i_{l,r} \\
i_{l,r} &= \frac{V_{l,r} + K_e(\dot{\psi} - \dot{\theta}_{l,r})}{R_m} \\
F_{l,r} &= \left(\frac{nK_tK_e}{R_m} + f_m\right)(\dot{\psi} - \dot{\theta}_{l,r}) + \frac{nK_tV_{l,r}}{R_m}
\end{aligned} \tag{22}
$$

Pegando a relação (20) entre as forças generalizadas e os torques de cada um das rodas $\{Fl, Fr\}$ e substituindo $\{Fl, Fr\}$ pelos valores obtidos (22):

$$
\begin{aligned}
F &= \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ -\frac{W}{2R} & \frac{W}{2R} \end{bmatrix} \begin{bmatrix} F_l \\ F_r \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ -\frac{W}{2R} & \frac{W}{2R} \end{bmatrix} \begin{bmatrix} \left(\frac{nK_tK_e}{R_m} + f_m\right)(\dot{\psi} - \dot{\theta}_l) + \frac{nK_tV_r}{R_m} \\ \left(\frac{nK_tK_e}{R_m} + f_m\right)(\dot{\psi} - \dot{\theta}_r) + \frac{nK_tV_r}{R_m} \end{bmatrix} \\
F &= \begin{bmatrix} -2\left(\frac{nK_tK_e}{R_m} + f_m\right) & 2\left(\frac{nK_tK_e}{R_m} + f_m\right) & 0 \\ 2\left(\frac{nK_tK_e}{R_m} + f_m\right) & -2\left(\frac{nK_tK_e}{R_m} + f_m\right) & 0 \\ 0 & 0 & -\frac{W^2}{2R^2}\left(\frac{nK_tK_e}{R_m} + f_m\right) \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\psi} \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} \frac{nK_t}{R_m} & \frac{nK_t}{R_m} \\ -\frac{nK_t}{R_m} & -\frac{nK_t}{R_m} \\ -\frac{W}{2R}\frac{nK_t}{R_m} & \frac{W}{2R}\frac{nK_t}{R_m} \end{bmatrix} \begin{bmatrix} V_l \\ V_r \end{bmatrix}
\end{aligned} \tag{23}
$$

2 Derivação e obtenção das equações de estado

Definindo as variáveis de estado do sistema:

$$
x_1 = \lambda = \begin{bmatrix} \theta \\ \psi \\ \phi \end{bmatrix}, \quad x_2 = \dot{\lambda} = \begin{bmatrix} \dot{\theta} \\ \dot{\psi} \\ \dot{\phi} \end{bmatrix} \tag{24}
$$

Derivando as variáveis de estado do sistema e utilizando as representação (24) como base:

$$
\dot{x_1} = \dot{\lambda} = x_2 = \begin{bmatrix} \dot{\theta} \\ \dot{\psi} \\ \dot{\phi} \end{bmatrix} \tag{25}
$$

Para calcular o $M(\lambda)^{-1}$, será necessário calcular o determinante:

$$
\begin{aligned}
det(M) &= \begin{vmatrix} (2m+M)R^2 + 2J_w & MRLcos(\psi) & 0 \\ MRLcos(\psi) & ML^2 + J_\psi & 0 \\ 0 & 0 & m\frac{W^2}{2} + MLsin(\psi)^2 + J_\phi + \frac{W^2}{2R^2}J_w \end{vmatrix} = \\
&= [(2m+M)R^2 + 2J_w](ML^2 + J_\psi)(m\tfrac{W^2}{2} + MLsin(\psi)^2 + J_\phi + \tfrac{W^2}{2R^2}J_w) - M^2R^2L^2cos(\psi)^2(m\tfrac{W^2}{2} + MLsin(\psi)^2 + J_\phi + \tfrac{W^2}{2R^2}J_w)
\end{aligned} \tag{26}
$$

Além disso, é necessário calcular a matriz adjunta:

$$M_{adj} = \begin{bmatrix} (ML^2+J_\psi)(m\frac{W^2}{2}+ML^2sin(\psi)^2+J_\phi+\frac{W^2}{2R^2}J_w) & -MRLcos(\psi)(m\frac{W^2}{2}+ML^2sin(\psi)^2+J_\phi+\frac{W^2}{2R^2}J_w) & 0 \\ -MRLcos(\psi)(m\frac{W^2}{2}+ML^2sin(\psi)^2+J_\phi+\frac{W^2}{2R^2}J_w) & [(2m+M)R^2+2J_w](m\frac{W^2}{2}+ML^2sin(\psi)^2+J_\phi+\frac{W^2}{2R^2}J_w) & 0 \\ 0 & 0 & [(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2cos(\psi)^2 \end{bmatrix} \tag{27}$$

Utilizando o determinante e a matriz adjunta para encontrar $M(\lambda)^{-1}$:

$$M(\lambda)^{-1} = \frac{M_{adj}}{det(M)} = \begin{bmatrix} \frac{ML^2+J_\psi}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2cos(\psi)^2} & -\frac{MRLcos(\psi)}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2cos(\psi)^2} & 0 \\ -\frac{MRLcos(\psi)}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2cos(\psi)^2} & \frac{(2m+M)R^2+2J_w}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2cos(\psi)^2} & 0 \\ 0 & 0 & \frac{1}{m\frac{W^2}{2}+MLsin(\psi)^2+J_\phi+\frac{W^2}{2R^2}J_w} \end{bmatrix} \tag{28}$$

Com isso, tenho tudo que é preciso para preencher a seguinte equação:

$$\dot{x_2} = \ddot{\lambda} = M(\lambda)^{-1}\left[-C(\lambda,\dot{\lambda}) - G(\lambda) + F(\dot{\lambda},V)\right] \tag{29}$$

3 Determinação dos pontos de equilíbrio do sistema

Considerando $V_l = V_r = 0$, é possível determinar os pontos de equilíbrio de forma trivial. Já que um pêndulo invertido montado numa plataforma de 2 rodas só estará em equilíbro sem a diferença de potencial nos motores, quando estiver em pé ou de cabeça para baixo verticalmente e sem nenhuma velocidade angular em nenhuma das direções. Logo:

$$\begin{gathered} \bar{\psi} = \{0, \pm\pi, \pm 2\pi, ..., \pm n\pi\}, n \in \mathbb{N} \\ \bar{x}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \end{gathered} \tag{30}$$

Então, o $\bar{\theta}$ e o $\bar{\phi}$ podem ser quaisquer valores porque não afetam no equilíbrio do sistema.

4 Simulação do sistema não linear com $V_l = V_r = 0$

Podemos simular numericamente o sistema através da função ode45 no matlab ou de uma S-functions com um bloco do mesmo nome no Simulink

(a) Função ode45 (Matlab):

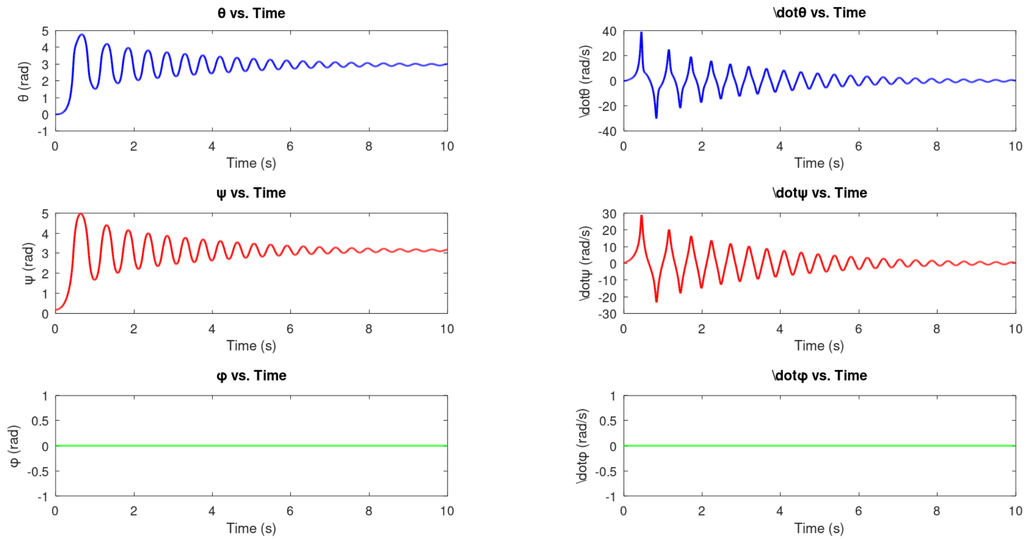


Figura 1: Simulação com condições iniciais x0 = [0; 0.18; 0; 0; 0; 0]

Observando o gráfico é possível observar o comportamento de pêndulo invertido do sistema, que não tem variação em ϕ e $\dot{\phi}$ ao ser colocado zero nas condições inicias. Porém, as outras variáveis de estado ligadas a posição tendem a π e as variáveis de estado ligadas a velocidade tendem a 0.

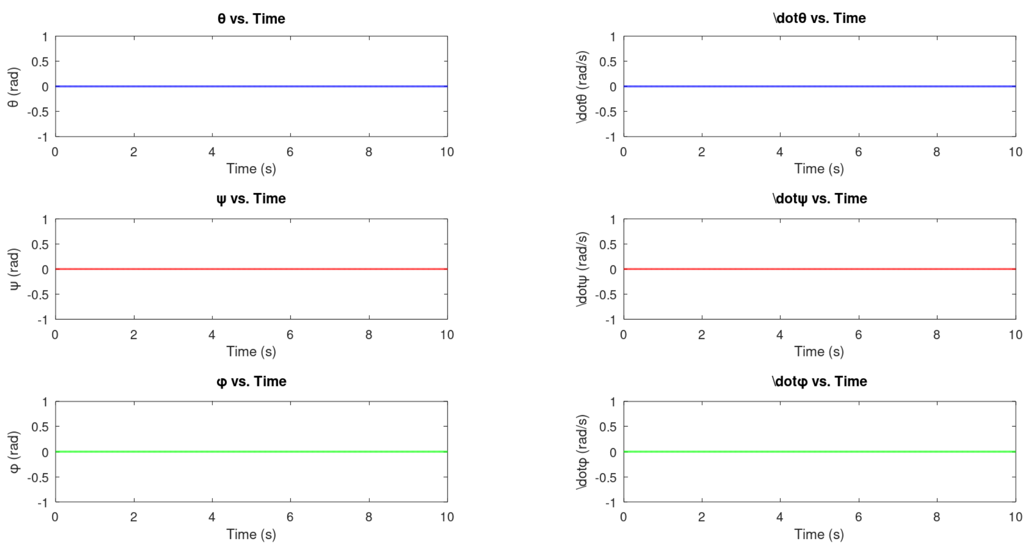


Figura 2: Simulação com condições iniciais x0 = [0; 0; 0; 0; 0; 0]

Ao colocar as condições iniciais em um dos pontos de equilíbrio, o sistema não tem variação das suas variáveis de estado o que acaba comprovando um dos casos propostos na parte 3 do relatório.

(b) S-functions (Simulink)

Não foi possível implementar a S-function, porque o Simulink que está disponível atualmente r2023a não consegue rodar o template que foi enviado. Então, vou mostrar o máximo que consegui chegar de desenvolvimento. Começando pelo penduloinvertido.m :

Listing 1: penduloinvertido.m

```
function [sys,x0,str,ts] = penduloinvertido(t,x,flag)
    switch flag
        case 0
            str = [] ;
            ts = [0 0] ;
            s = simsizes ;
            s.NumContStates = 6;
            s.NumDiscStates = 0 ;
            s.NumOutputs = 6 ;
            s.NumInputs = 2 ;
            s.DirFeedthrough = 0 ;
            s.NumSampleTimes = 1 ;
            sys = simsizes(s) ;
            x0 = [0 0 0 0 0 0] ;
        case 1
            sys = penduloinvertido_f(t,x) ;
        case 3
            sys = x;
        case {2 4 9}
            sys =[];
        otherwise
            error(['unhandled flag =',num2str(flag)]) ;
    end
end
```

Além disso a função que estaria na S-function:

Listing 2: penduloinvertido_f.m

```
function dx = penduloinvertido_f(t, x)
    % Variaveis de estado
    theta = x(1);
    psi = x(2);
    phi = x(3);
    % Derivadas das variaveis de estado
    theta_dot = x(4);
    psi_dot = x(5);
    phi_dot = x(6);
    % Parametros do sistema
    g = 9.8;  % gravidade
```

```
        m = 0.03; % peso da roda
        R = 0.04; % raio da roda
        Jw = m * R^2 / 2; % momento de inercia da roda
        M = 0.6;  % peso do corpo
        W = 0.14; % largura do corpo
        D = 0.04; % profundidade do corpo
        H = 0.144; % altura do corpo
        L = H / 2; % distancia do centro de massa do corpo ao eixo da roda
        Jphi = M * L^2 / 3; % momento de inercia do corpo em pitch
        Jpsi = M * (W^2 + D^2) / 12; % momento de inercia do corpo em rumo
        Rm = 6.69; % resistencia do motor DC
        Kt = 0.4; % constante de torque (Nm/A)
        Ke = 0.4; % e constante de EMF (V s/rad)
        fm = 0.0022; % atrito entre o corpo e o motor
        n = 1; % reducao do motor
        % Matriz de inercia inversa
        M_inv = inv([
            R^2 * (2 * m + M) + 2 * Jw, M * R * L * cos(psi), 0;
            M * R * L * cos(psi), M * L^2 + Jpsi, 0;
            0, 0, m * (W^2 / 2) + M * L * sin(psi)^2 + Jphi + (W^2 / (2 * R^2)) * Jw
        ]);
        % Matriz G
        G = [
            0;
            -M * g * L * sin(psi);
            0];
        % Matriz C
        C = [
            -M * R * L * sin(psi) * psi_dot^2;
            -M * L^2 * sin(psi) * cos(psi) * phi_dot^2;
            M * L^2 * sin(2 * psi) * psi_dot * phi_dot
        ];
        % Matriz Q
        Q = [
            -2 * ((n * Kt * Ke / Rm) + fm), 2 * ((n * Kt * Ke / Rm) + fm), 0;
            2 * ((n * Kt * Ke / Rm) + fm), -2 * ((n * Kt * Ke / Rm) + fm), 0;
            0, 0, -W^2 / (2 * R^2) * ((n * Kt * Ke / Rm) + fm)
        ];
        F = Q*[theta_dot; psi_dot; phi_dot]
        % Vetor de entrada
        V = [0; 0]; % [V_l; V_r]
        % Sistema de equacoes diferenciais
        dx = zeros(6, 1);
        dx(1:3) = [theta_dot; psi_dot; phi_dot];
        dx(4:6) = M_inv * (F - G - C);
    end
```

Ademais, montei no simulink como funcionaria o sistema não-linear. Tendo uma entrada com amplitude 0 e seis saídas que seriam os estados do sistema que poderiam ser visualizadas pelos scopes.

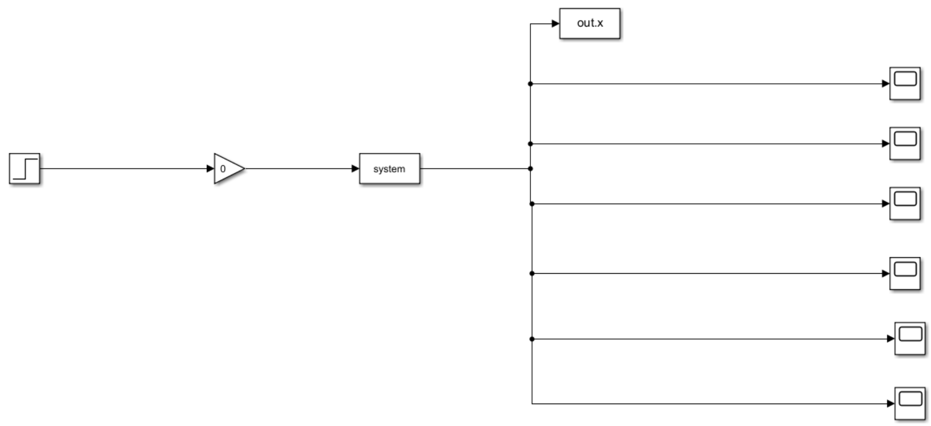


Figura 3: Simulink

(c) scipy.integrate.solve_ivp (Python):

Foi obtido um resultado muito similar ao da função ode45 do Matlab:

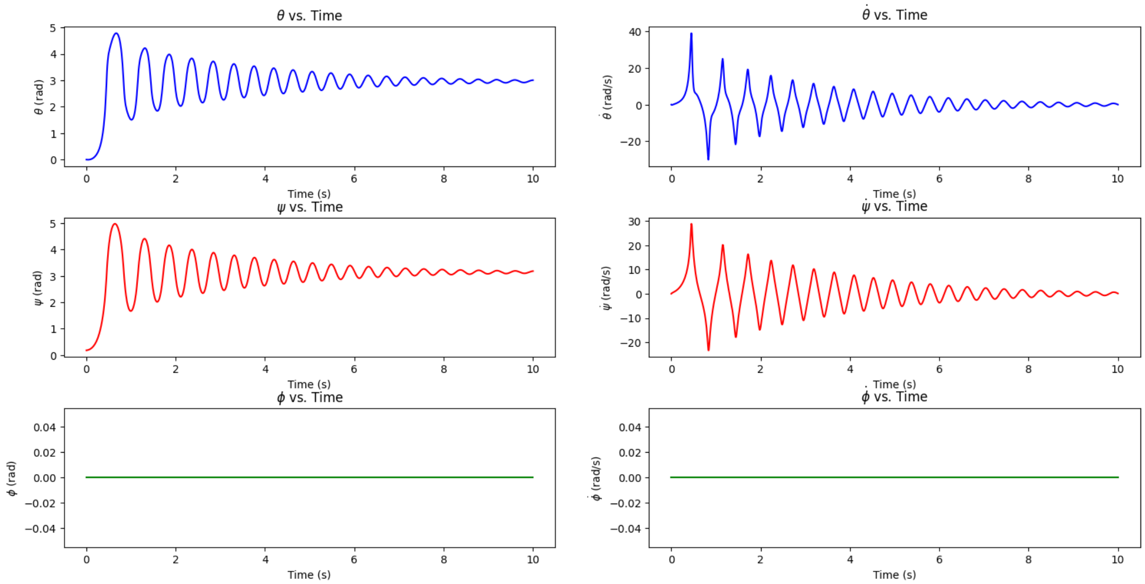


Figura 4: Simulação com condições iniciais x0 = [0; 0.18; 0; 0; 0; 0]

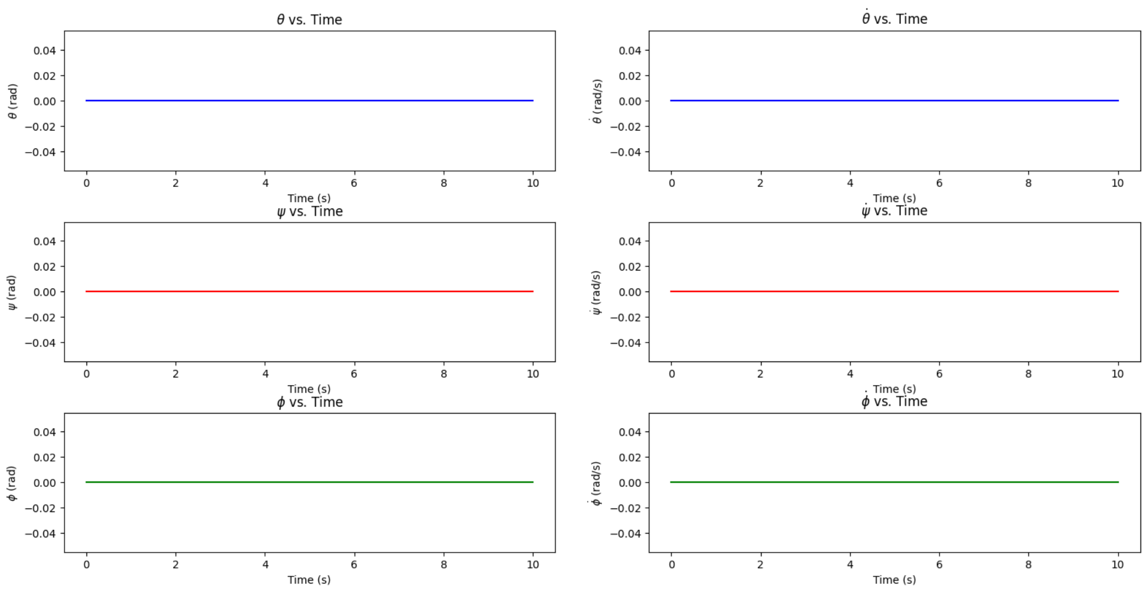


Figura 5: Simulação com condições iniciais x0 = [0; 0; 0; 0; 0; 0]

5 Linearização do sistema com $\psi = 0$

O sistema será linearizado em relação ao ponto de equilíbrio quando $\psi = 0$. Logo, teremos as seguintes aproximações:

$$\begin{gathered}
\widetilde{\theta} = \theta - \bar{\theta}, \widetilde{\psi} = \psi - \bar{\psi}, \widetilde{\phi} = \phi - \bar{\phi} \\
\dot{\widetilde{\theta}} = \dot{\theta} - \dot{\bar{\theta}}, \dot{\widetilde{\psi}} = \dot{\psi} - \dot{\bar{\psi}}, \dot{\widetilde{\phi}} = \dot{\phi} - \dot{\bar{\phi}} \\
sin\psi = \widetilde{\psi} \\
cos\psi = 1
\end{gathered} \tag{31}$$

Assim, obtemos a linearização do sistema:

$$\begin{gathered}
M^{-1} = \begin{bmatrix} \frac{ML^2+J_\psi}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2} & -\frac{MRL}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2} & 0 \\ -\frac{MRL}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2} & \frac{(2m+M)R^2+2J_w}{[(2m+M)R^2+2J_w](ML^2+J_\psi)-M^2R^2L^2} & 0 \\ 0 & 0 & \frac{1}{m\frac{W^2}{2}+\frac{W^2}{2R^2}J_w} \end{bmatrix} \\
C = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \\
G(\lambda) = \begin{bmatrix} 0 \\ -MgL\widetilde{\psi} \\ 0 \end{bmatrix} \\
\dot{\widetilde{x}}_1 = \begin{bmatrix} \dot{\widetilde{\theta}} \\ \dot{\widetilde{\psi}} \\ \dot{\widetilde{\phi}} \end{bmatrix} \\
\dot{\widetilde{x}}_2 = M^{-1} \left[-G(\widetilde{\lambda}) + F(\dot{\widetilde{\lambda}}, \widetilde{V}) \right]
\end{gathered} \tag{32}$$

6 Simulação do sistema linearizado com $V_l = V_r = 0$

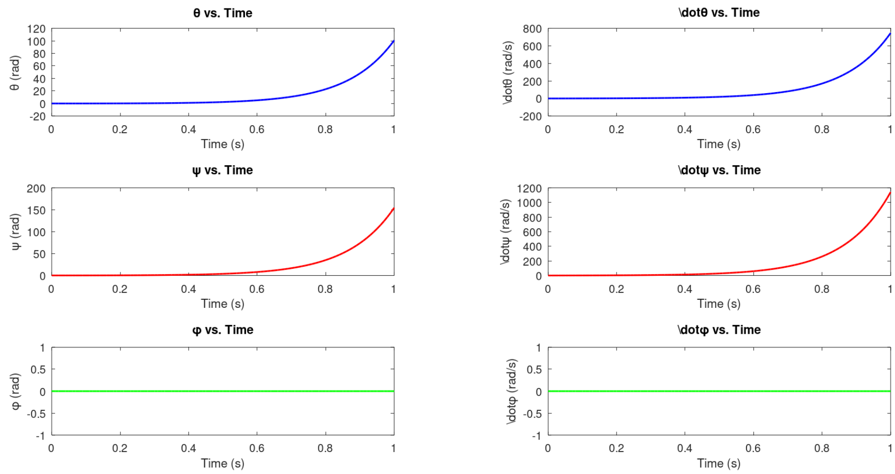


Figura 6: Simulação com condições iniciais x0 = [0; 0.18; 0; 0; 0; 0]

Observando o gráfico fica bem claro que a linearização não é muito conveniente após passar de $\psi > \pi$. É possível ver a gigantesca de diferença entre o sistema não-linear e o sistema linearizado.

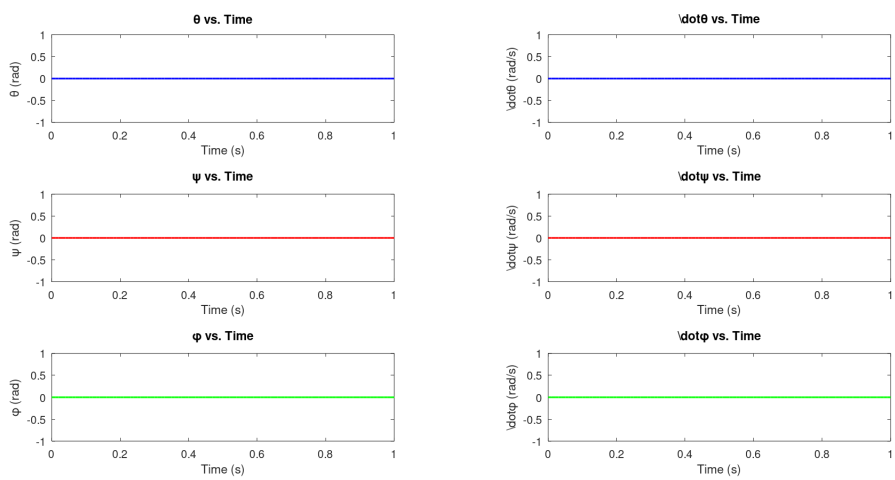


Figura 7: Simulação com condições iniciais x0 = [0; 0; 0; 0; 0; 0]

Já no caso de colocar as condições iniciais $x0 = [0; 0; 0; 0; 0; 0]$, o comportamento do sistema se mantém no ponto de equilíbrio.

7 Controle realimentado

O sistema considerado é instável em $\psi = 0$. Desta forma é projetado o seguinte controle realimentado:

$$u = -Kx + Nr \tag{33}$$

onde $u^T = [V_l \quad V_r]$, $x^T = [\theta \quad \psi \quad \phi \quad \dot{\theta} \quad \dot{\psi} \quad \dot{\phi}]$, r é uma entrada de referências para comandar sistema para frente, $N^T = [-0.7071 \quad 0.7071]$ e

$$K = \begin{bmatrix} -0.0224 & -25.4867 & -0.7071 & -1.0362 & -2.2530 & -0.0076 \\ -0.0224 & -25.4867 & 0.7071 & -1.0362 & -2.2530 & 0.0076 \end{bmatrix} \tag{34}$$

Foi simulado a resposta do sistema não linear com este controle realimentado, considerando diferentes condições iniciais.

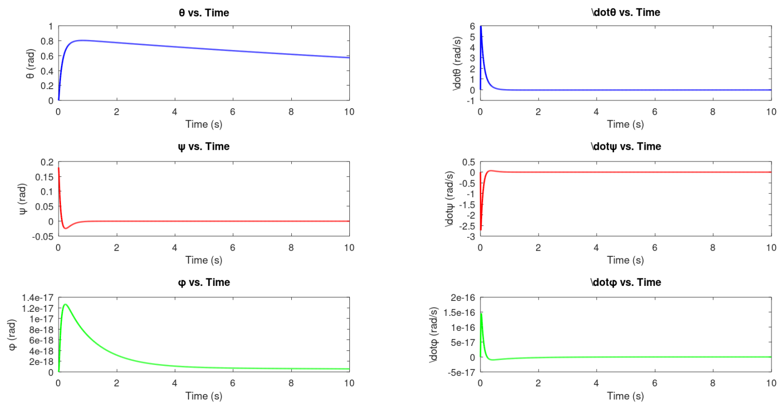


Figura 8: Simulação com condições iniciais x0 = [0; 0.18; 0; 0; 0; 0]

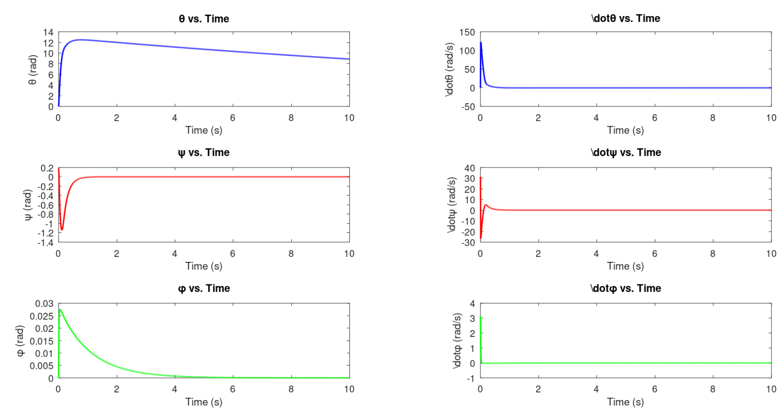


Figura 9: Simulação com condições iniciais x0 = [0; 0.18; 0; 0; 31.4; 3.14]

É possível observar que o controle realimentado tem como objetivo equilibrar o sistema de uma forma que se torne próximo da estabilidade.

A Código: Simulação do sistema não-linear

Listing 3: Código da simulação do sistema não-linear com ode45

```
function pendulum_simulation()
    % Initial conditions
    x0 = [0; 0.18; 0; 0; 0; 0]; % [theta; psi; phi; dot{theta}; dot{psi}; dot{phi}]
    % Time span for simulation
    tspan = 0:0.01:10; % From 0 to 10 seconds with a time step of 0.01 seconds
    % Solve the differential equations using ode45 solver
    [t, x] = ode45(@penduloinvertido_f, tspan, x0);
    % Extract the states for plotting
    theta = x(:, 1);
    psi = x(:, 2);
    phi = x(:, 3);
    dot_theta = x(:, 4);
    dot_psi = x(:, 5);
    dot_phi = x(:, 6);
    % Plot the results
    figure;
    subplot(3, 2, 1);
    plot(t, theta, 'b', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\theta (rad)');
    title('Pendulum Motion: \theta vs. Time');
    subplot(3, 2, 3);
    plot(t, psi, 'r', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\psi (rad)');
    title('Pendulum Motion: \psi vs. Time');
    subplot(3, 2, 5);
    plot(t, phi, 'g', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\phi (rad)');
    title('Pendulum Motion: \phi vs. Time');
    subplot(3, 2, 2);
    plot(t, dot_theta, 'b', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\dot{\theta} (rad/s)');
    title('Pendulum Motion: \dot{\theta} vs. Time');
    subplot(3, 2, 4);
    plot(t, dot_psi, 'r', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\dot{\psi} (rad/s)');
    title('Pendulum Motion: \dot{\psi} vs. Time');
    subplot(3, 2, 6);
```

```
        plot(t, dot_phi, 'g', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\dot{\phi} (rad/s)');
        title('Pendulum Motion: \dot{\phi} vs. Time');
    end

    function dx = penduloinvertido_f(t, x)
        % Variaveis de estado
        theta = x(1);
        psi = x(2);
        phi = x(3);
        % Derivadas das variaveis de estado
        theta_dot = x(4);
        psi_dot = x(5);
        phi_dot = x(6);
        % Parametros do sistema
        g = 9.8;  % gravidade
        m = 0.03; % peso da roda
        R = 0.04; % raio da roda
        Jw = m * R^2 / 2; % momento de inercia da roda
        M = 0.6;  % peso do corpo
        W = 0.14; % largura do corpo
        D = 0.04; % profundidade do corpo
        H = 0.144; % altura do corpo
        L = H / 2; % distancia do centro de massa do corpo ao eixo da roda
        Jphi = M * L^2 / 3; % momento de inercia do corpo em pitch
        Jpsi = M * (W^2 + D^2) / 12; % momento de inercia do corpo em rumo
        Rm = 6.69; % resistencia do motor DC
        Kt = 0.4; % constante de torque (Nm/A)
        Ke = 0.4; % e constante de EMF (V s/rad)
        fm = 0.0022; % atrito entre o corpo e o motor
        n = 1; % reducao do motor
        % Matriz de inercia inversa
        M_inv = inv([
            R^2 * (2 * m + M) + 2 * Jw, M * R * L * cos(psi), 0;
            M * R * L * cos(psi), M * L^2 + Jpsi, 0;
            0, 0, m * (W^2 / 2) + Jphi + (W^2 / (2 * R^2)) * Jw
        ]);
        % Matriz G
        G = [
            0;
            -M * g * L * sin(psi);
            0
        ];
        % Matriz C
        C = [
```

```
            -M * R * L * sin(psi) * psi_dot^2;
            -M * L^2 * sin(psi) * cos(psi) * phi_dot^2;
            M * L^2 * sin(2 * psi) * psi_dot * phi_dot
        ];
        % Matriz Q
        Q = [
            -2 * ((n * Kt * Ke / Rm) + fm), 2 * ((n * Kt * Ke / Rm) + fm), 0;
            2 * ((n * Kt * Ke / Rm) + fm), -2 * ((n * Kt * Ke / Rm) + fm), 0;
            0, 0, -W^2 / (2 * R^2) * ((n * Kt * Ke / Rm) + fm)
        ];
        F = Q*[theta_dot; psi_dot; phi_dot]
        % Vetor de entrada
        V = [0; 0]; % [V_l; V_r]
        % Sistema de equacoes diferenciais
        dx = zeros(6, 1);
        dx(1:3) = [theta_dot; psi_dot; phi_dot];
        dx(4:6) = M_inv * (F - G - C);
    end
    pendulum_simulation();
```

B Código: Simulação do sistema não-linear em Python

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp
def pendulum_simulation():
    # Initial conditions
    x0 = [0, 0, 0, 0, 0, 0]  # [theta; psi; phi; dot{theta}; dot{psi}; dot{phi}]
    # Time span for simulation
    tspan = np.arange(0, 10.01, 0.01)  # From 0 to 10 seconds with a time step of 0.01 seconds
    # Solve the differential equations using solve_ivp
    sol = solve_ivp(penduloinvertido_f, [tspan[0], tspan[-1]], x0, t_eval=tspan)
    # Extract the states for plotting
    t = sol.t
    theta, psi, phi, dot_theta, dot_psi, dot_phi = sol.y
    # Plot the results
    fig, axs = plt.subplots(3, 2, figsize=(12, 10))
    axs[0, 0].plot(t, theta, 'b', linewidth=1.5)
    axs[0, 0].set_xlabel('Time (s)')
    axs[0, 0].set_ylabel(r'$\theta$ (rad)')
```

```
    axs[0, 0].set_title(r'$\theta$ vs. Time')
    axs[1, 0].plot(t, psi, 'r', linewidth=1.5)
    axs[1, 0].set_xlabel('Time (s)')
    axs[1, 0].set_ylabel(r'$\psi$ (rad)')
    axs[1, 0].set_title(r'$\psi$ vs. Time')
    axs[2, 0].plot(t, phi, 'g', linewidth=1.5)
    axs[2, 0].set_xlabel('Time (s)')
    axs[2, 0].set_ylabel(r'$\phi$ (rad)')
    axs[2, 0].set_title(r'$\phi$ vs. Time')
    axs[0, 1].plot(t, dot_theta, 'b', linewidth=1.5)
    axs[0, 1].set_xlabel('Time (s)')
    axs[0, 1].set_ylabel(r'$\dot{\theta}$ (rad/s)')
    axs[0, 1].set_title(r'$\dot{\theta}$ vs. Time')
    axs[1, 1].plot(t, dot_psi, 'r', linewidth=1.5)
    axs[1, 1].set_xlabel('Time (s)')
    axs[1, 1].set_ylabel(r'$\dot{\psi}$ (rad/s)')
    axs[1, 1].set_title(r'$\dot{\psi}$ vs. Time')
    axs[2, 1].plot(t, dot_phi, 'g', linewidth=1.5)
    axs[2, 1].set_xlabel('Time (s)')
    axs[2, 1].set_ylabel(r'$\dot{\phi}$ (rad/s)')
    axs[2, 1].set_title(r'$\dot{\phi}$ vs. Time')
    plt.tight_layout()
    plt.show()
def penduloinvertido_f(t, x):
    # Unpack the state variables
    theta, psi, phi, theta_dot, psi_dot, phi_dot = x
    # Parameters of the system
    g = 9.8
    m = 0.03
    R = 0.04
    Jw = m * R ** 2 / 2
    M = 0.6
    W = 0.14
    D = 0.04
    H = 0.144
    L = H / 2
```

```
    Jphi = M * L ** 2 / 3
    Jpsi = M * (W ** 2 + D ** 2) / 12
    Rm = 6.69
    Kt = 0.4
    Ke = 0.4
    fm = 0.0022
    n = 1
    # Matrices
    M_inv = np.linalg.inv(np.array([
        [R ** 2 * (2 * m + M) + 2 * Jw, M * R * L * np.cos(psi), 0],
        [M * R * L * np.cos(psi), M * L ** 2 + Jpsi, 0],
        [0, 0, m * (W ** 2 / 2) + Jphi + (W ** 2 / (2 * R ** 2)) * Jw]
    ]))
    G = np.array([0, -M * g * L * np.sin(psi), 0])
    C = np.array([
        -M * R * L * np.sin(psi) * psi_dot ** 2,
        -M * L ** 2 * np.sin(psi) * np.cos(psi) * phi_dot ** 2,
        M * L ** 2 * np.sin(2 * psi) * psi_dot * phi_dot
    ])
    Q = np.array([
        [-2 * ((n * Kt * Ke / Rm) + fm), 2 * ((n * Kt * Ke / Rm) + fm), 0],
        [2 * ((n * Kt * Ke / Rm) + fm), -2 * ((n * Kt * Ke / Rm) + fm), 0],
        [0, 0, -W ** 2 / (2 * R ** 2) * ((n * Kt * Ke / Rm) + fm)]
    ])
    F = Q @ np.array([theta_dot, psi_dot, phi_dot])
    # Input vector
    V = np.array([0, 0])  # [V_l, V_r]
    # System of differential equations
    dxdt = np.zeros_like(x)
    dxdt[0:3] = [theta_dot, psi_dot, phi_dot]
    dxdt[3:6] = M_inv @ (F - G - C)
    return dxdt
# Call the simulation function to run the simulation and plot the results
pendulum_simulation()
```

C Código: Simulação do sistema linearizado

Listing 4: Código da simulação do sistema com ode45

```
function pendulum_simulation()
    % Initial conditions
    x0 = [0; 0.18; 0; 0; 0; 0]; % [theta; psi; phi; dot{theta}; dot{psi}; dot{phi}]
    % Time span for simulation
    tspan = 0:0.01:0.4; % From 0 to 10 seconds with a time step of 0.01 seconds
    % Solve the differential equations using ode45 solver
    [t, x] = ode45(@penduloinvertido_f, tspan, x0);
    % Extract the states for plotting
    theta = x(:, 1);
    psi = x(:, 2);
    phi = x(:, 3);
    dot_theta = x(:, 4);
    dot_psi = x(:, 5);
    dot_phi = x(:, 6);
    % Plot the results
    figure;
    subplot(3, 2, 1);
    plot(t, theta, 'b', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\theta (rad)');
    title('Pendulum Motion: \theta vs. Time');
    subplot(3, 2, 3);
    plot(t, psi, 'r', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\psi (rad)');
    title('Pendulum Motion: \psi vs. Time');
    subplot(3, 2, 5);
    plot(t, phi, 'g', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\phi (rad)');
    title('Pendulum Motion: \phi vs. Time');
    subplot(3, 2, 2);
    plot(t, dot_theta, 'b', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\dot{\theta} (rad/s)');
    title('Pendulum Motion: \dot{\theta} vs. Time');
    subplot(3, 2, 4);
    plot(t, dot_psi, 'r', 'LineWidth', 1.5);
    xlabel('Time (s)');
    ylabel('\dot{\psi} (rad/s)');
    title('Pendulum Motion: \dot{\psi} vs. Time');
    subplot(3, 2, 6);
```

```
        plot(t, dot_phi, 'g', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\dot{\phi} (rad/s)');
        title('Pendulum Motion: \dot{\phi} vs. Time');
    end
    function dx = penduloinvertido_f(t, x)
        % Variaveis de estado
        theta = x(1);
        psi = x(2);
        phi = x(3);
        % Derivadas das variaveis de estado
        theta_dot = x(4);
        psi_dot = x(5);
        phi_dot = x(6);
        % Parametros do sistema
        g = 9.8;  % gravidade
        m = 0.03; % peso da roda
        R = 0.04; % raio da roda
        Jw = m * R^2 / 2; % momento de inercia da roda
        M = 0.6;  % peso do corpo
        W = 0.14; % largura do corpo
        D = 0.04; % profundidade do corpo
        H = 0.144; % altura do corpo
        L = H / 2; % distancia do centro de massa do corpo ao eixo da roda
        Jphi = M * L^2 / 3; % momento de inercia do corpo em pitch
        Jpsi = M * (W^2 + D^2) / 12; % momento de inercia do corpo em rumo
        Rm = 6.69; % resistencia do motor DC
        Kt = 0.4; % constante de torque (Nm/A)
        Ke = 0.4; % e constante de EMF (V s/rad)
        fm = 0.0022; % atrito entre o corpo e o motor
        n = 1; % reducao do motor
        % Matriz de inercia inversa
        M_inv = inv([
            R^2 * (2 * m + M) + 2 * Jw, M * R * L , 0;
            M * R * L , M * L^2 + Jpsi, 0;
            0, 0, m * (W^2 / 2) + Jphi + (W^2 / (2 * R^2)) * Jw
        ]);
        % Matriz G
        G = [
            0;
            -M * g * L * psi;
            0
        ];
        % Matriz C
        C = [
            0;
```

```
            0;
            0
        ];
        % Matriz Q
        Q = [
            -2 * ((n * Kt * Ke / Rm) + fm), 2 * ((n * Kt * Ke / Rm) + fm), 0;
            2 * ((n * Kt * Ke / Rm) + fm), -2 * ((n * Kt * Ke / Rm) + fm), 0;
            0, 0, -W^2 / (2 * R^2) * ((n * Kt * Ke / Rm) + fm)
        ];
        F = Q*[theta_dot; psi_dot; phi_dot]
        % Vetor de entrada
        V = [0; 0]; % [V_l; V_r]
        % Sistema de equacoes diferenciais
        dx = zeros(6, 1);
        dx(1:3) = [theta_dot; psi_dot; phi_dot];
        dx(4:6) = M_inv * (F - G - C);
    end
    pendulum_simulation();
```

D Código: Simulação do sistema com controle realimentado

Listing 5: Código da simulação do sistema com ode45

```
    function pendulum_simulation()
        % Initial conditions
        x0 = [0; 0.18; 0; 0; 0; 0]; % [theta; psi; phi; dot{theta}; dot{psi}; dot{phi}]
        % Time span for simulation
        tspan = 0:0.01:10; % From 0 to 10 seconds with a time step of 0.01 seconds
        % Solve the differential equations using ode45 solver
        [t, x] = ode45(@penduloinvertido_f, tspan, x0);
        % Extract the states for plotting
        theta = x(:, 1);
        psi = x(:, 2);
        phi = x(:, 3);
        dot_theta = x(:, 4);
        dot_psi = x(:, 5);
        dot_phi = x(:, 6);
        % Plot the results
        figure;
        subplot(3, 2, 1);
        plot(t, theta, 'b', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\theta (rad)');
        title('Pendulum Motion: \theta vs. Time');
        subplot(3, 2, 3);
```

```
        plot(t, psi, 'r', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\psi (rad)');
        title('Pendulum Motion: \psi vs. Time');
        subplot(3, 2, 5);
        plot(t, phi, 'g', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\phi (rad)');
        title('Pendulum Motion: \phi vs. Time');
        subplot(3, 2, 2);
        plot(t, dot_theta, 'b', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\dot{\theta} (rad/s)');
        title('Pendulum Motion: \dot{\theta} vs. Time');
        subplot(3, 2, 4);
        plot(t, dot_psi, 'r', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\dot{\psi} (rad/s)');
        title('Pendulum Motion: \dot{\psi} vs. Time');
        subplot(3, 2, 6);
        plot(t, dot_phi, 'g', 'LineWidth', 1.5);
        xlabel('Time (s)');
        ylabel('\dot{\phi} (rad/s)');
        title('Pendulum Motion: \dot{\phi} vs. Time');
    end
    function dx = penduloinvertido_f(t, x)
        % Variaveis de estado
        theta = x(1);
        psi = x(2);
        phi = x(3);
        % Derivadas das variaveis de estado
        theta_dot = x(4);
        psi_dot = x(5);
        phi_dot = x(6);
        % Parametros do sistema
        g = 9.8;  % gravidade
        m = 0.03; % peso da roda
        R = 0.04; % raio da roda
        Jw = m * R^2 / 2; % momento de inercia da roda
        M = 0.6;  % peso do corpo
        W = 0.14; % largura do corpo
        D = 0.04; % profundidade do corpo
        H = 0.144; % altura do corpo
        L = H / 2; % distancia do centro de massa do corpo ao eixo da roda
        Jphi = M * L^2 / 3; % momento de inercia do corpo em pitch
        Jpsi = M * (W^2 + D^2) / 12; % momento de inercia do corpo em rumo
```

```
        Rm = 6.69; % resistencia do motor DC
        Kt = 0.4; % constante de torque (Nm/A)
        Ke = 0.4; % e constante de EMF (V s/rad)
        fm = 0.0022; % atrito entre o corpo e o motor
        n = 1; % reducao do motor
        % Matriz de inercia inversa
        M_inv = inv([
            R^2 * (2 * m + M) + 2 * Jw, M * R * L * cos(psi), 0;
            M * R * L * cos(psi), M * L^2 + Jpsi, 0;
            0, 0, m * (W^2 / 2) + Jphi + (W^2 / (2 * R^2)) * Jw
        ]);
        % Matriz G
        G = [
            0;
            -M * g * L * sin(psi);
            0
        ];
        % Matriz C
        C = [
            -M * R * L * sin(psi) * psi_dot^2;
            -M * L^2 * sin(psi) * cos(psi) * phi_dot^2;
            M * L^2 * sin(2 * psi) * psi_dot * phi_dot
        ];
        % Matriz Q
        Q = [
            -2 * ((n * Kt * Ke / Rm) + fm), 2 * ((n * Kt * Ke / Rm) + fm), 0;
            2 * ((n * Kt * Ke / Rm) + fm), -2 * ((n * Kt * Ke / Rm) + fm), 0;
            0, 0, -W^2 / (2 * R^2) * ((n * Kt * Ke / Rm) + fm)
        ];
        % Matriz H
        H = [
            n * Kt / Rm, n * Kt / Rm;
            -n * Kt / Rm, -n * Kt / Rm;
            -W / (2 * R) * n * Kt / Rm, W / (2 * R) * n * Kt / Rm
        ];
        % Matriz K
        K = [-0.0224, -25.4867, -0.7071, -1.0362, -2.2530, -0.0076;
            -0.0224, -25.4867, 0.7071, -1.0362, -2.2530, 0.0076];
        F = Q*[theta_dot; psi_dot; phi_dot] - H * K * [theta;psi;phi;theta_dot;psi_dot;phi_dot];
        % Sistema de equacoes diferenciais
        dx = zeros(6, 1);
        dx(1:3) = [theta_dot; psi_dot; phi_dot];
        dx(4:6) = M_inv * (F - G - C);
    end
    pendulum_simulation();
```