PÊNDULO INVERTIDO ROTATIVO

Sistema de elevação e estabilização do pêndulo na posição vertical e controlo da posição do braço

Controlo por Computador no Espaço de Estados

PL1

Grupo 3

1 2  9 0

UNIVERSIDADE D
COIMBRA

FACULDADE DE CIÊNCIAS E TECNOLOGIA

MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES

Dylan Frédéric Denizon

Carlos Miguel Mendes Gaspar

Duarte de Sousa Cruz

Nuno Miguel Gomes Sénica

Conteúdo

Lista de Figuras

Listings

1 Introdução

1.1 Objetivos e Descrição

Este trabalho prático tem como objetivo implementar e analisar, recorrendo aos softwares Matlab e Simulink, um sistema de controlo de um Pêndulo Invertido Rotativo (PIR), utilizando técnicas estocásticas, devendo este convergir para uma posição vertical, mesmo quando sujeito a perturbações.

Este relatório está dividido em três partes, apresentação das métricas de desempenho, controlo de estabilização do pêndulo e do braço e o sistema de controlo completo, considerando comando por tensão.

A estrutura geométrica do PIR usado para a realização das simulações deste trabalho pratico está apresentado na figura seguinte:

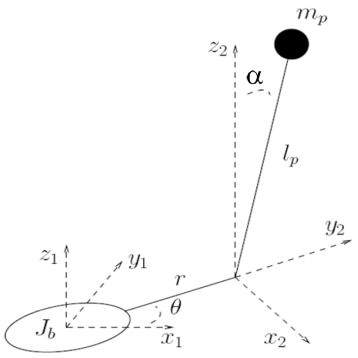

Figura 1.1: Estrutura geométrica do PIR

O pêndulo é um sistema não-linear com dois pontos de equilíbrio, um estável e um instável. O modelo em espaço de estados contínuo do Pêndulo Invertido Rotativo, linearizado em torno do ponto de equilíbrio instável, são representadas da seguinte forma matricial:

Para o modelo linear de corrente:

$$\begin{bmatrix} \dot{\theta} \\ \dot{\alpha} \\ \ddot{\theta} \\ \ddot{\alpha} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{m_p \cdot r \cdot g}{J_b} & 0 & 0 \\ 0 & \frac{g \cdot (J_b + mp \cdot r^2)}{lp \cdot J_b} & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \alpha \\ \dot{\theta} \\ \dot{\alpha} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{J_b} \\ -\frac{r}{l_p \cdot J_b} \end{bmatrix} T \tag{1.1}$$

Para o modelo linear de tensão:

$$\begin{bmatrix} \dot{\theta} \\ \dot{\alpha} \\ \ddot{\theta} \\ \ddot{\alpha} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{m_p \cdot r \cdot g}{J_b} & -\frac{K_m^2 \cdot K_g^2}{R_m \cdot J_b} & 0 \\ 0 & \frac{g \cdot (J_b + mp \cdot r^2)}{lp \cdot J_b} & -\frac{K_m^2 \cdot K_g^2 \cdot r}{R_m \cdot J_b \cdot l_p} & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \alpha \\ \dot{\theta} \\ \dot{\alpha} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -\frac{K_m \cdot K_g}{R_m \cdot J_b} \\ -\frac{K_m \cdot K_g \cdot r}{R_m \cdot J_b \cdot l_p} \end{bmatrix} V \tag{1.2}$$

Representámos os modelos lineares do PIR (modelos no espaço de estados com comando por tensão e comando por corrente) pelos seguintes diagramas de blocos em Simulink:

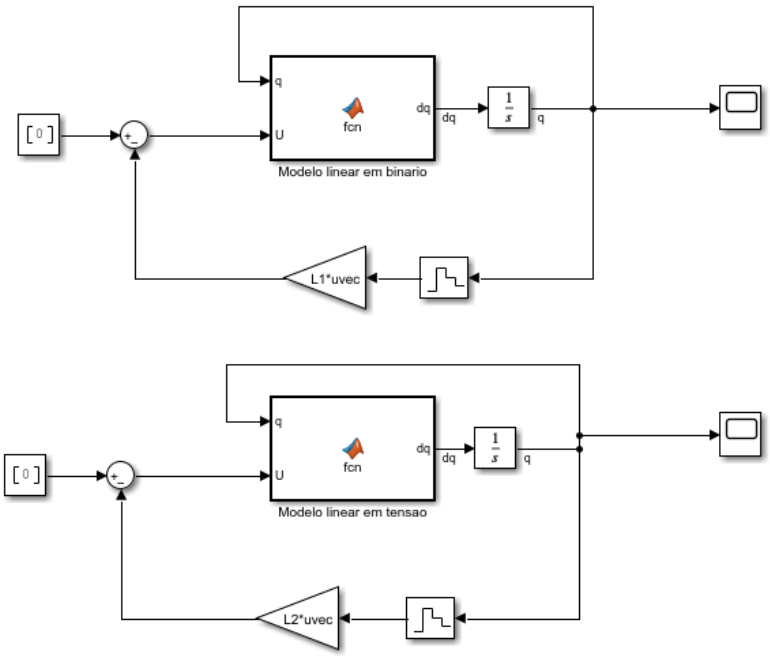

Figura 1.2: Modelos lineares

1.2 Parâmetros e métricas de desempenho

Considerando um PIR que é definido pelos os seguintes parâmetros:

- $Rm = 2.6\Omega$
- $Km = 0.00767 Nm/A$
- $K_g = 70:1 (K_{gi} = 14:1; K_{ge} = 5:1)$
- massa do pêndulo, $mp = 0.126 Kg$
- $Lp = 0.33m$
- raio do braço, $r = 0.22m$

- $Jb = 0.0044Kgm^2$
- Tensão máxima do motor ($\pm$ 6V)
- Gama de funcionamento do braço ($\theta : -\pi$ a π rad)
- Gama de funcionamento do pêndulo (α: -0.52 a 0.52 rad)
- h = 5ms

1.3 Simulação gráfica do PIR

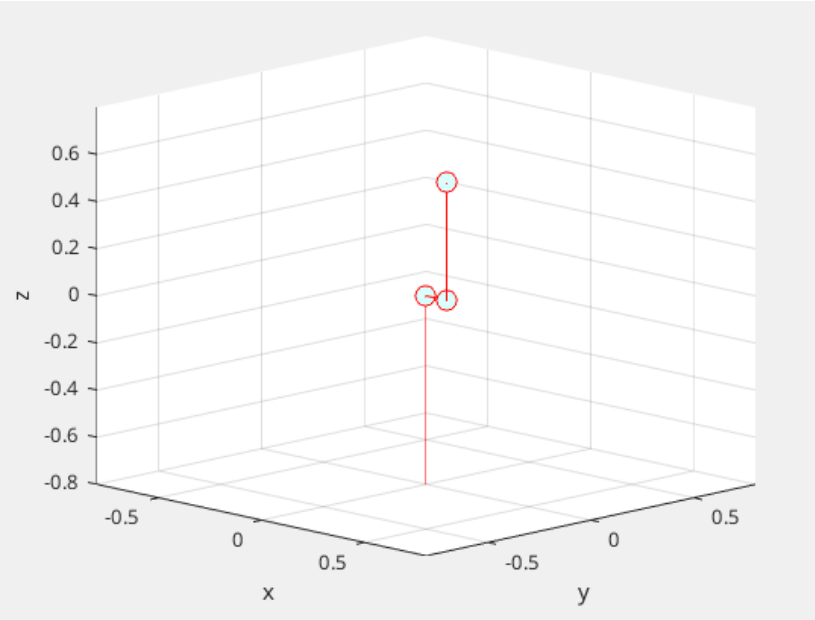

Figura 1.3: Simulação do PIR

Ao longo deste relatório estão presentes anexos contendo os modelos simulinks e os scripts Matlabs usados para a realização deste trabalho prático. Alguns resultados de simulações realizados para justificar as conclusões chegadas serão fornecidos junto com o pasta com todo o código desenvolvido.

2 Controlo de estabilização do pêndulo e do braço

2.1 Modelo não linear do PIR – comando por tensão

De forma a obter a simulação não-linear do PIR, desenhámos o seguinte diagrama de blocos:

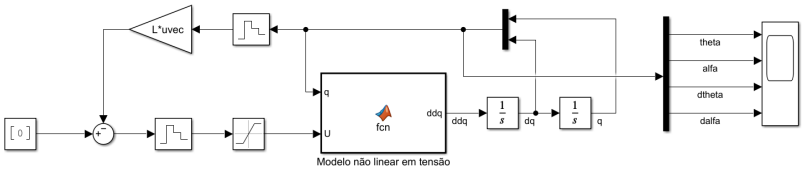

Figura 2.1: Modelo PIR não linear

```
function ddq  = fcn(q,U)
    %parametros motor
    Rm = 2.6;
    Km = 0.00767;
    Kg = 70;
    mp = 0.126;
    Lp = 0.33;
    lp = Lp/2;
    r = 0.22;
    Jb = 0.0044;
    g = 9.8;

    theta=q(1);
    alfa=q(2);
    dtheta=q(3);
    dalfa=q(4);

    a = Jb + mp*r^2;
    b = mp*r*lp*cos(alfa);
    c = mp*lp*r*cos(alfa);
    d = mp*lp^2;
    M = [a b;c d];

    Vb = U;
    Tb = (Kg*Km)*(Vb-Kg*Km*dtheta)/Rm;

    a = -mp*r*dalfa^2*lp*sin(alfa);
    b = -mp*g*lp*sin(alfa);
    H = [a;b];

    ddq = inv(M)*([Tb;0]-H);
end
```

Listing 2.1: Código Matlab do modelo não linear em espaço de estados

Na função *dqq* vamos calcular $\ddot{q} = [\ddot{\theta}; \ddot{\alpha}]$ em função da tensão a entrada do motor, U e dos estados atuais q. Para isto vamos usar as equações (14) e (15) do enunciado sobre o sistema não linear controlado em corrente:

$$(J_b + m_p r^2)\ddot{\theta} + m_p l_p r \ddot{\alpha} cos(\alpha) - m_p l_p r \dot{\alpha}^2 sin(\alpha) = T_b \tag{2.1}$$

$$m_p l_p r \ddot{\theta} cos(\alpha) + m_p l_p^2 \ddot{\alpha} - m_p l_p g sin(\alpha) = 0 \tag{2.2}$$

Rearranjamos estas equações na forma matricial para obtermos as seguintes matrizes:

$$M = \begin{bmatrix} J_b + m_p r^2 & m_p r l_p cos(v) \\ m_p l_p r cos(v) & m_p l_p^2 \end{bmatrix} \tag{2.3}$$

$$H = \begin{bmatrix} -m_p r \dot{v}^2 l_p sin(v) \\ -m_p g l_p sin(v) \end{bmatrix} \tag{2.4}$$

De modo a obter o modelo do sistema não linear controlado em tensão fizemos a seguinte substituição:

$$T_b = \frac{K_g * K_m}{R_m} * (U - K_g * K_m * \dot{\theta}) \tag{2.5}$$

Por último, obtemos a seguinte equação não linear que descreve o movimento do pêndulo:

$$\ddot{q} = M^{-1}(T_b - H) \tag{2.6}$$

2.2 Controlo de estabilização do pêndulo na posição vertical (comando por tensão)

A representação genérica em espaço de estados na forma matricial é:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + D \end{aligned} \tag{2.7}$$

Onde A é a mtariz dinâmica do sistema, B o vetor de entrada, C o vetor de saída e D a matriz de feedfoward.
Considera-se uma aproximação ao ponto de equilíbrio ,$v = 0$, a representação em espaço de estados.

Dinâmica escolhida contínua em malha fechada

```
wn = 5;
zeta = 1;
wn*h
polos_n_domi = [0.5; 0.5];

% obtencao dos polos equivalentes discretos de modelo malha fechada
    em tensao
wnh=wn*h;
p1=-2*exp(-zeta*wnh)*cos(wnh*sqrt(1-zeta^2));
p2= exp(-2*zeta*wnh);
pmfd=roots([1 p1 p2])

pmf = [pmfd; polos_n_domi];
L = acker(phi,gama,pmf)
```

Listing 2.2: Código Matlab para dinâmica contínua escolhida em malha fechada controlado em tensão

O modelo equivalente discreto, com período de amostragem h, é:

$$\begin{aligned} \dot{x}(k+1) &= \phi x(t) + \Gamma u(t) \\ y(t) &= Cx(t) \end{aligned} \tag{2.8}$$

Os valores de ϕ e Γ, são obtidos pela função:

$$[\phi, \Gamma] = c2d(A, B, h);$$

$$\phi = \begin{bmatrix} 1.0000 & -0.0007 & 0.0047 & 0 \\ 0 & 1.0017 & 0.0004 & 0.0050 \\ 0 & -0.2902 & 0.8816 & -0.0007 \\ 0 & 0.6841 & 0.1579 & 1.0017 \end{bmatrix} \quad \Gamma = \begin{bmatrix} 0.0006 \\ -0.0008 \\ 0.2206 \\ -0.2941 \end{bmatrix}$$

Para o controlo do sistema escolhemos os seguintes parâmetros continuo, uma frequência natural $w_n = 5$ e um coeficiente de amortecimento $\zeta = 1$. O pólo duplo discreto não dominante foi colocado em 0.5 para não ter impacto na resposta. Com estes valores, é possível construir o polinómio característico. Sabendo os polos de malha fechada do sistema contínuo, obtivemos os seus equivalentes discretos, pela seguinte expressão:

$$z = e^{sh}$$

Assim, os polos em malha fechada do polinómio são:

$$\begin{aligned} z_1 &= 0.9753 + 0.0000i \\ z_2 &= 0.9753 - 0.0000i \\ z_3 &= 0.5000 + 0.0000i \\ z_4 &= 0.5000 + 0.0000i \end{aligned}$$

Para o estudo do projeto por colocação de polos, no espaço de estados, a matriz de ganho, L, foi calculada pela funcão `acker`:

$$\begin{gathered} L = \texttt{acker}(\phi, \Gamma, z) \\ L = \begin{bmatrix} -93.0762 & -267.3507 & -39.3932 & -32.2100 \end{bmatrix} \end{gathered} \tag{2.9}$$

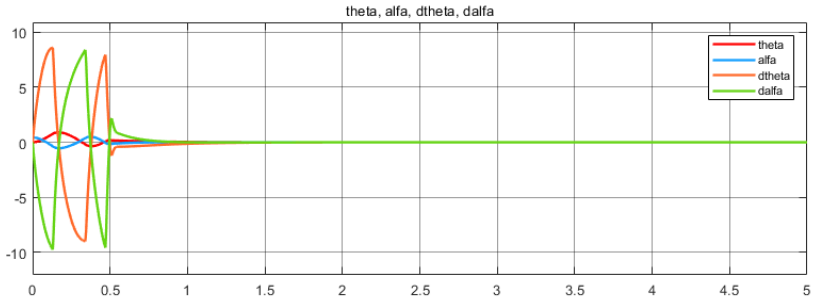

Figura 2.2: Simulação por colocação de pólos

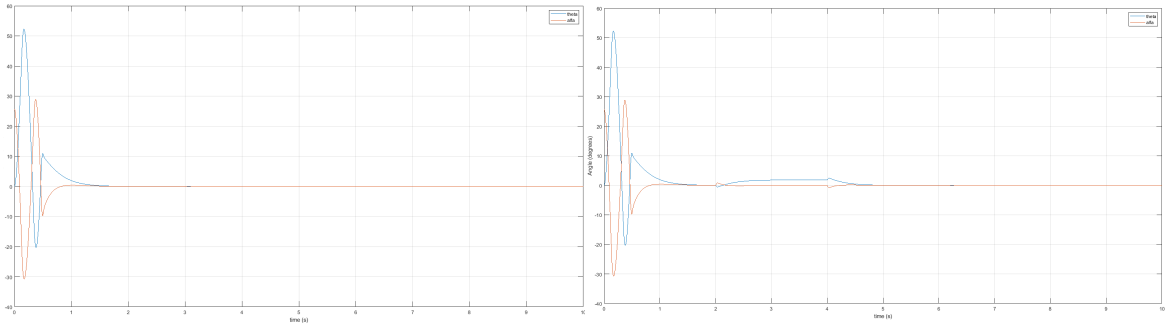

Figura 2.3: Simulação por colocação de pólos sem e com perturbação

Controlo LQR

```
Q1 = [100 1000 0.1 1].*eye(4);
Q2=5;
L=dlqr(phi,gama,Q1,Q2)
```

Listing 2.3: Código Matlab para controlo LQR

$$Q_1 = \begin{bmatrix} q_{11} & 0 & 0 & 0 \\ 0 & q_{22} & 0 & 0 \\ 0 & 0 & q_{33} & 0 \\ 0 & 0 & 0 & q_{44} \end{bmatrix} \tag{2.10}$$

A matriz Q_1 representa os pesos atribuídos à posição do braço, à posição do pêndulo, à velocidade do braço e à velocidade do pêndulo. Os fatores de peso foram determinados experimentalmente através do método tentativa erro. O resultado do ganho do controlador é para uma relação de $q_{22}/q_{11} = 10$. Cada variação de parâmetro irá estar diretamente relacionado com os parâmetros da matriz:

$$\begin{bmatrix} \theta \\ \alpha \\ \dot{\theta} \\ \dot{\alpha} \end{bmatrix}$$

Com o parâmetro Q_2 dá-se maior ou menor peso ao esforço de comando. Para um valor mais baixo aumenta o esforço de comando.

Podemos observar que, para diferentes valor de **Q** e **R**, estes valores são escolhidos consoante o fator que pretendemos otimizar no projeto. Para as seguintes simulações foram dados os seguintes valores,

$$Q_1 = [100, 1000, 0.1, 1]$$
$$R_1 = 5$$
$$Q_2 = [177.8062, 1257.3748, 1.7537, 0.5337]$$
$$R_2 = 0.2574;$$

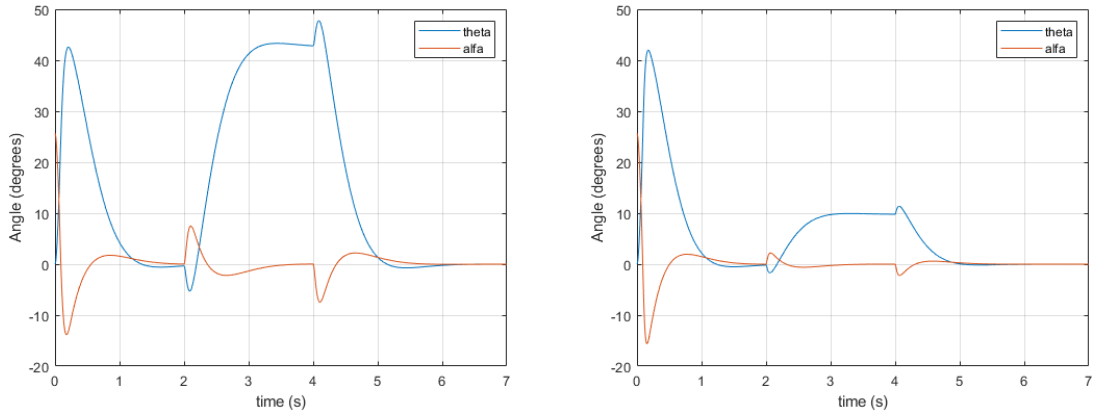

Figura 2.4: **(1)** Esquerda ; **(2)** Direita

A matriz de ganho resultante do controlador é:

$$L = \begin{bmatrix} -4.0364 & -24.6530 & -2.6101 & -2.5656 \end{bmatrix} \tag{2.11}$$

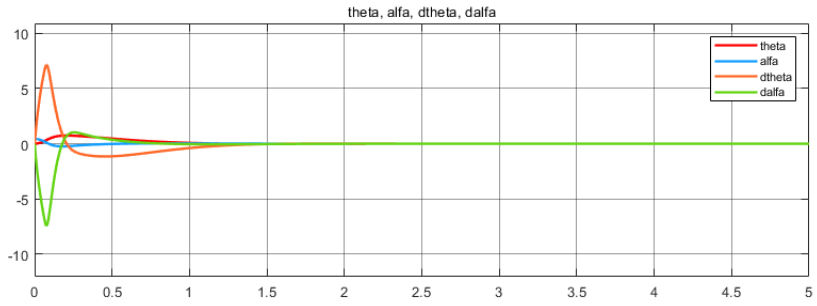

Figura 2.5: Simulação por controlo LQR

Por observação dos gráficos obtidos pelas simulações, e o vetor de ganhos **L** obtido, podemos afirmar que o controlador LQR apresenta maior velocidade e menos oscilação na resposta, originando assim um controlo mais estável.

2.3 Controlo simultâneo de estabilização do pêndulo na posição vertical e do posicionamento angular do braço (comando por tensão)

Quando simulamos o sistema com uma perturbação de valores 3 e -3 no tempo 2 e 4 respetivamente, obtemos a seguinte resposta:

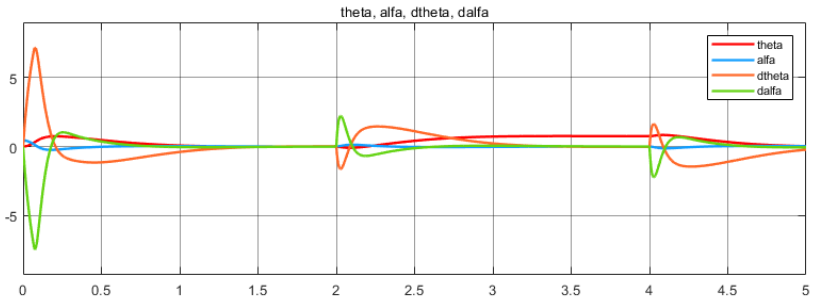

Figura 2.6: Perturbação no controlo LQR

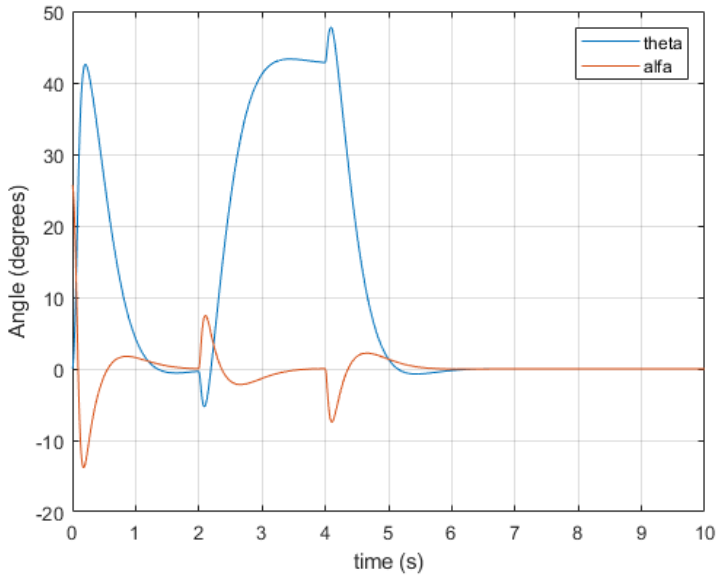

Figura 2.7: Perturbação no controlo LQR

Como explicado na alínea anterior, o controlador LQR apresenta uma reduzida oscilação na resposta e maior velocidade, mesmo com o controlo em simultâneo da posição vertical e da posição ângular. Este controlador foi desenhado para usar a posição vertical/*upright* como referência e consegue estabilizar com uma perturbação angular do braço em 3 segundos. O que justifica e prova a rapidez e estabilidade do controlador LQR.

Nota: No código matlab foi implementada uma section `visualizacao do pendulo` que nos permite ver a ação e o movimento do pêndulo.

3 Sistema de controlo completo, considerando comando por tensão

3.1 Componente para elevação do pêndulo (swing-up) e controlo de estabilização do pêndulo e da posição do braço

Para a implementação do movimento swing-up do pêndulo, desenvolvemos o diagrama de blocos representado na Figura 3.1

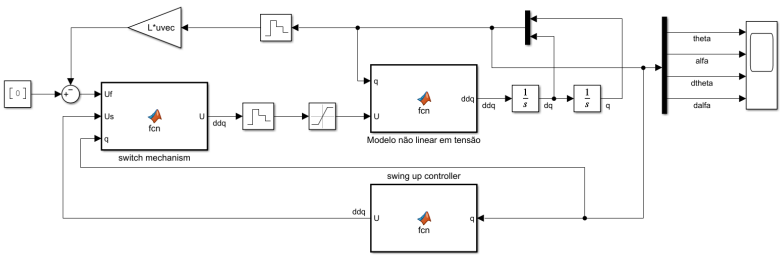

Figura 3.1: Modelo PIR não linear para o swing-up

Para esta implementação, não fizemos alterações ao bloco do modelo não linear, mantendo-se como visto em **(2)**.

Contudo, implementámos dois blocos funções MatLab, `switch mecanism` e `swing up controller`.

O bloco `switch mecanism` é o responsável por computar e decidir, em função do ângulo, se o sistema deve ser controlado pelo controlador swing-up ou pelo LQR, sendo que definimos esse treshold para alfa igual a 30 graus. O código matlab da implementão do bloco é apresentado a seguir:

```
function U  = fcn(Uf,Us,q)
    theta=q(1);
    alfa=q(2);
    dtheta=q(3);
    dalfa=q(4);

    if abs(alfa)<deg2rad(30)
        U = Uf;
    else
        U = Us;
    end
end
```

Listing 3.1: Código Matlab do bloco switch mecanism

O bloco `swing up controller` é o responsável por controlar o movimento do pendulo de modo a ele atingir o angulo minimo desejado com certa velocidade para que o controlador LQR consiga funcionar e manter o pendulo em pé. Isto é conseguido da seguinte forma:
Considerando que a componente de fricção pode ser desprezada, podemos assumir que o pêndulo é um corpo rígido. Com isto, realizámos esta tarefa controlando a energia que o pêndulo necessita para executar o movimento de *swing-up* para a posição *upright*. Também implementamos dentro do controlador um saturador de modo a controlar as acelerações do pêndulo para que ele não chega ao topo com demasiada velocidade e que o controlador LQR não consiga trabalhar. Temos que,

$$w_0 = \sqrt{\frac{mp \cdot g \cdot Lp}{Jb}}$$

A trajetória será descrita por:

$$E = \frac{1}{2} \cdot J \cdot (\frac{\dot{\alpha}}{w_0})^2 + m \cdot g \cdot l \cdot (cos(\alpha) - 1) = 0$$

Como $J = m \cdot g \cdot l_p$, então:

$$E = m \cdot g \cdot l_p \cdot (\frac{1}{2} \cdot (\frac{\dot{\alpha}}{w_0})^2 + (cos(\alpha) - 1) \tag{3.1}$$

Depois, com a energia associada ao movimento, usando um controlador proporcional com valor de referência = 0.1, computámos para que lado será necessário que o motor rode para ajudar a elevar o pêndulo para o levar à posição de *upright*.

```
function U  = fcn(q)
    %parametros motor
    Rm = 2.6;
    Km = 0.00767;
    Kg = 70;
    mp = 0.126;
    Lp = 0.33;
    lp = Lp/2;
    r = 0.22;
    Jb = 0.0044;
    g = 9.8;
    n= 0.4;
    k =15;

    theta=q(1);
    alfa=q(2);
    dtheta=q(3);
    dalfa=q(4);

    w0 = sqrt(mp*g*Lp/Jb);
    E=mp*g*lp*(0.5*(dalfa/w0)^2+cos(alfa)-1);

    u =  k*(E-0.1);
    if u> n*g
        u = n*g;
    elseif u<-n*g
        u = -n*g;
```

```
    end

U = u*sign(dalfa*cos(alfa));
end
```

Listing 3.2: Código Matlab do bloco switch mecanism

O estudo do projeto do movimento **swing-up**, por colocação de pólos, no espaço de estados, é conseguido usando os mesmos valores para ζ e para w_n, o que vai assim gerar o mesmo vetor de ganhos **L** apresentado na secção **(2)**. Simulando obtivémos as seguintes respostas:

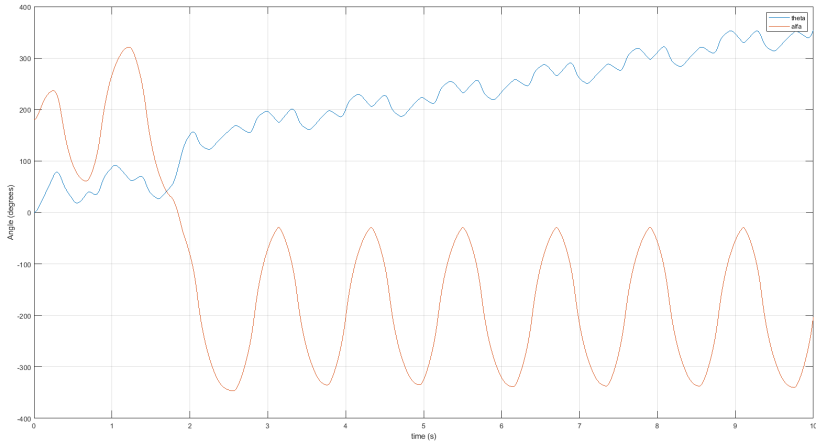

Figura 3.2: Simulação por colocação de pólos para o movimento o swing-up (em graus)

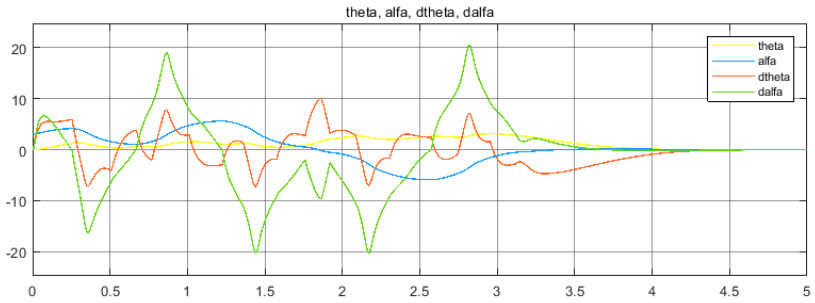

Figura 3.3: Simulação por colocação de pólos para o movimento o swing-up

O estudo do projeto do movimento **swing-up**, por controlo LQR, à semelhança dos passos anteriores, mantivémos os mesmos valores para Q_1 e Q_2, o que, como visto anteriormente, vai dar origem ao mesmo vetor **L**. Simulando obtivémos as seguintes respostas:

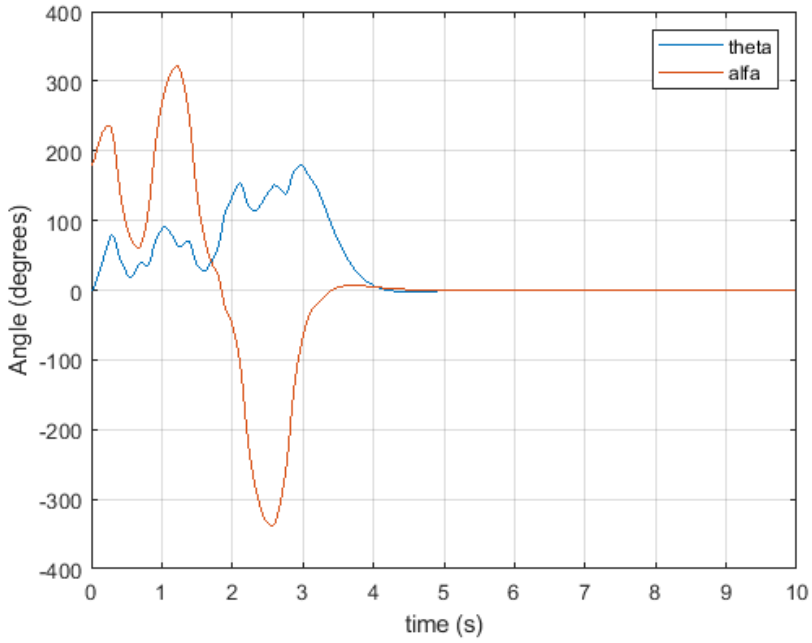

Figura 3.4: Simulação por controlo LQR para o movimento o swing-up (em graus)

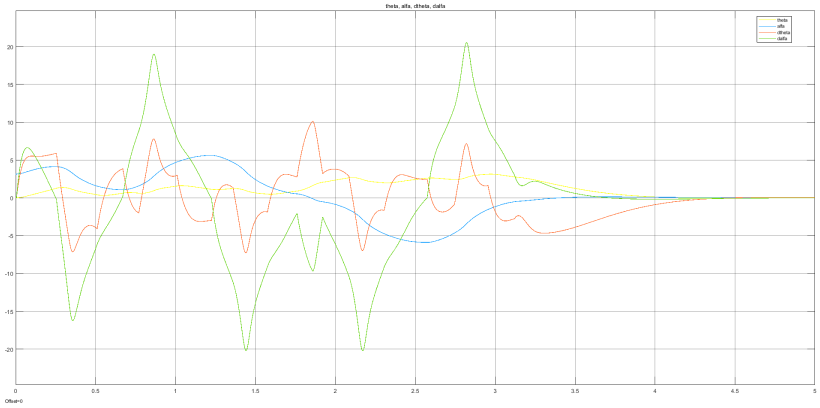

Figura 3.5: Simulação por controlo LQR para o movimento o swing-up

Podemos verificar pelas simulações e pelo movimento descrito pelo pêndulo, no ambiente MatLab, que o controlo e estabilização do pêndulo é conseguida com maior rapidez e menos oscilação para quando este é feito com recurso ao controlo LQR.

Nota: No código matlab foi implementada uma section `visualizacao do pendulo` que nos permite ver a ação e o movimento do pêndulo.

4 Conclusões

Em suma, acreditamos que os resultados obtidos foram bastante satisfatórios e que foi possível simular um sistema de pêndulo bastante estável. A partir das simulações realizadas concluímos ainda que o controlador LQR apresenta melhores resultados pois permite controlar o peso atribuído a cada um dos estado ou até eliminá-los o que não é possível com o controlo por colocação de polos onde todos os estados serão usados, onde verificamos que podemos controlar melhor a resposta do sistema e obtivemos resultados melhores usando este método mesmo quando submetido a perturbações.