ROBÔ DE DUAS RODAS AUTOEQUILIBRADO SEGUIDOR DE LINHA

Samuel Jakobovitsch Goes*, Gabriel Pereira das Neves*, Bruno Augusto Angélico*

*Escola Politécnica da USP
Depto. de Engenharia de Telecomunicações e Controle
São Paulo, SP, Brasil

Emails: samuel.goes@usp.br, gabriel.pereira.neves@usp.br, angelico@lac.usp.br

Abstract— Vehicles guided by computational vision are ever closer of us. This article presents the study and implementation of computational vision in a two wheel self-balanced vehicle, as well as the steps to its stabilization. There are presented the construction steps of the prototype, the modeling using Euler-Lagrange's method, the implementation of the control system considering the linear quadratic regulator and the image processing of the camera's image to detect a band on the floor and send the analyzed data to the control of the robot. Practical results are presented.

Keywords— Robot Control, Computational Vision, Digital Control, LQR, Selfbalancing Robot

Resumo— Veículos guiados por visão computacional estão cada vez mais próximos de nós. Este artigo apresenta o estudo e a implementação de visão computacional em um veículo de duas rodas autoequilibrado, bem como as etapas para sua estabilização. São apresentadas as fases de construção do protótipo, a modelagem pelo método de Euler-Lagrange, a implementação do seu sistema controle considerando o regulador linear quadrático e o processamento das imagens da câmera para detectar uma linha no chão e enviar os dados analisados para o controle do robô. Os resultados práticos são apresentados.

Palavras-chave— Controle de Robôs, Visão Computacional, LQR, Robô Autoequilibrado

1 Introdução

O robô autoequilibrado com duas rodas é uma variação do pêndulo invertido, problema bastante estudado em controle ((Ogata, 2003)). O veículo *Segway* lançado pela DEKA mostra como este robô pode se tornar robusto e confiável com estratégias de controle.

Há vários trabalhos na literatura que consideram controle de robôs autoequilibrados com duas rodas, tais como em (Raffo et al., 2015; Kim and Kwon, 2017; Kralev et al., 2016). Em (Raffo et al., 2015) um controlador não linear $\mathcal{H}_\infty$ é projetado e aplicado ao sistema. O trabalho apresentado em (Kim and Kwon, 2017) considera um controle não linear ótimo aplicado a um robô similar ao considerado neste trabalho. Em (Kralev et al., 2016) dois controladores robusto projetados por síntese-μ e aplicados a um robô autoequilibrado de duas rodas.

Neste trabalho, um protótipo deste robô foi construído. Toda sua modelagem matemática foi realizada através do método de Euler-Lagrange, obtendo-se um modelo não linear. Em seguida, o modelo foi linearizado em torno do ponto de operação e discretizado para que se pudesse abordar uma técnica de controle digital. Foi então implementado o regulador linear quadrático (LQR) para estabilizar o robô e para que este seguisse as referências de ângulo de guinada e velocidade linear.

Para que a faixa no piso fosse identificada foram utilizadas técnicas de processamento de imagens, computadas em um *Raspbery pi*. Foram utilizados dois métodos para o processamento: pelo gradiente da imagem e pela limiarização de branco ou preto. Com isso, foi possível gerar referências para o algoritmo de controle através das imagens da câmera.

Os resultados práticos do controle e do processamento de imagens são apresentados ao final do trabalho.

2 Modelagem Matemática

O robô autoequilibrado de duas rodas pode ser modelado através de uma analogia com um manipulador (Craig, 2005). Para tanto, foi fixado o sistema de coordenadas {0} entre as rodas do robô e outro sistema de coordenadas {1} que acompanha o centro de massa do robô, como pode ser visto na Figura 1.

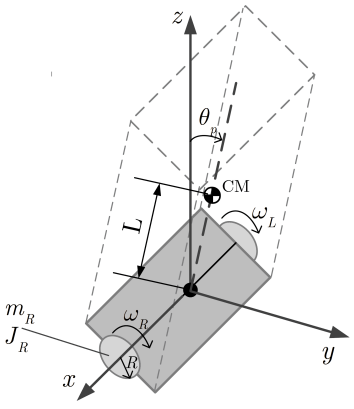

Figura 1: Modelo esquemático do sistema.

A modelagem foi realizada através da metodologia de Lagrange. Para tanto, é necessário cal-

cular as energias cinética e potencial do robô.

A energia potencial em relação à base depende da massa do corpo e da posição do centro de massa como em (1), em que m_p é a massa do corpo, L é a distância do centro de massa ao sistema de coordenadas da base, g é a aceleração da gravidade e θ_p é o ângulo de arfagem.

$$U = m_p g L cos(\theta_p) \tag{1}$$

Para o cálculo, a energia cinética será dividida na energia das rodas ((2) para a roda direita e (3) para a roda esquerda) e na energia do corpo (4).

$$K_R = \frac{1}{2}{\omega_R}^2(J_r + m_r R^2) \tag{2}$$

$$K_L = \frac{1}{2}{\omega_L}^2(J_r + m_r R^2) \tag{3}$$

$$K_p = \frac{1}{2}\overrightarrow{\Omega}_{CM}^T I_p \overrightarrow{\Omega}_{CM} + \frac{1}{2}m_p|V_{CM}|^2 \tag{4}$$

Onde ω_R e ω_L são, respectivamente, as velocidades angulares das rodas direita e esquerda; J_r é o momento de inercia da roda; m_r é a massa da roda; R é o raio da roda; $\overrightarrow{\Omega}_{CM}$ é a velocidade de rotação do corpo escrita em relação a o centro de massa; I_p é o tensor de inercia; m_p é a massa do corpo e V_{CM} é a velocidade do centro de massa do robô escrita em relação ao sistema de coordenadas $\{1\}$.

A velocidade de translação do centro de massa ($\vec{V}_{CM}$) é a soma da velocidade do sistema de coordenadas da base com a velocidade do centro de massa causada por sua rotação em torno da base (5). A velocidade de translação da base pode ser escrita em termos das velocidades angulares e do raio das rodas (6). Já a rotação do centro de massa pode ser escrita com relação à base (7) em que D é a distância entre as rodas, e então calcular a velocidade linear causada no centro de massa (8). Assim, obtém-se uma expressão da velocidade de translação do centro de massa.

$$\overrightarrow{V}_{CM} = \overrightarrow{V}_0 + \overrightarrow{\Omega}_0 \times (CM - 0) \tag{5}$$

$$\overrightarrow{V}_0 = \frac{1}{2}(\omega_R + \omega_L)R \tag{6}$$

$$\overrightarrow{\Omega}_0 = -\omega_p\hat{i} + 0\hat{j} + (\omega_R - \omega_L)\frac{R}{D}\hat{k} \tag{7}$$

$$(CM - 0) = 0\hat{i} + L\sin(\theta_p)\hat{j} + L\cos(\theta_p)\hat{k} \tag{8}$$

A velocidade de rotação do centro de massa ($\overrightarrow{\Omega}_{CM}$) pode ser escrita com referência no centro de massa através de uma rotação escrita em relação ao sistema de coordenadas $\{0\}$ ($\overrightarrow{\Omega}_0$) para o centro de massa (sistema de coordenadas $\{1\}$), conforme apresentado em (9).

$$\overrightarrow{\Omega}_{CM} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \sin(\theta_p) & \cos(\theta_p) \\ 0 & -\cos(\theta_p) & \sin(\theta_p) \end{bmatrix} \overrightarrow{\Omega}_0 \tag{9}$$

Pela simetria do problema, os termos do tensor de inercia I_p que estão fora da diagonal principal são desprezíveis, sendo considerados nulos na modelagem (10).

$$I_p = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{10}$$

Finalmente, a energia cinética pode ser escrita em termos dos parâmetros do sistema e dos estados do sistema (11)

$$\begin{aligned} &K_p = \tfrac{1}{2}(I_{xx}{\omega_p}^2) + \\ &\tfrac{1}{2}m_p[\tfrac{1}{2}R(\omega_L + \omega_R) + L\cos(\theta_p)\omega_p^2] + \\ &\tfrac{1}{2}(\omega_L - \omega_R)^2\frac{R^2}{D^2}(m_p L^2 + I_{yy}\cos(\theta_P)^2 + I_{zz}\sin(\theta_p)^2) \end{aligned} \tag{11}$$

Tendo as energias cinética e potencial definidas, podemos calcular o Lagrangiano (12) (13).

$$L = (K_R + K_L + K_p) - U \tag{12}$$

$$\begin{aligned} &L = \tfrac{1}{2}(J_r + m_r R^2)(\omega_R^2 + \omega_L^2) + \tfrac{1}{2}(\omega_p^2 I_{xx}) + \\ &\tfrac{1}{2}(\omega_R - \omega_L)^2\frac{R^2}{D^2}(I_{yy}\cos(\theta_p)^2 + I_{zz}\sin(\theta_p)^2) + \\ &\tfrac{1}{2}m_p\left[\left(\frac{LR}{D}\right)^2(\omega_R - \omega_L)^2\sin(\theta_p)^2 + L^2\omega_p^2\right] + \\ &\tfrac{1}{2}m_p\left[\tfrac{1}{4}(\omega_R + \omega_L)^2R^2 + LR\omega_p(\omega_R + \omega_L)\cos(\theta_p)\right] \\ &-m_p g L\cos(\theta_p) \end{aligned} \tag{13}$$

Para obter as equações da dinâmica basta resolver as equações de Euler-Lagrange (14).

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{q}_k}\right) - \frac{\delta L}{\delta q_k} = \tau_k \tag{14}$$

Em que q_k são as k variáveis de junta do sistema e τ_k são os torques externos aplicados. Neste caso será aplicada a equação de Euler-Lagrange para as três variáveis: o ângulo da roda direita (θ_R), o ângulo da roda esquerda (θ_L) e o ângulo de arfagem do corpo (θ_p).

Os torques externos das rodas (τ_R para a roda direita e τ_L para a roda esquerda) podem ser obtidos pelo modelo do motor apresentado em (15), onde $\dot{\theta}$ é a velocidade de rotação do eixo do motor. O ângulo θ pode ser escrito como em (16) considerando os ângulos da roda (no caso, a roda direita) e do pêndulo.

$$PWM \cdot V_{max} - R_m i - L_m\frac{di}{dt} - K_e\dot{\theta} = 0 \tag{15}$$

$$\theta = \theta_R - \theta_p \tag{16}$$

sendo V_{max} a tensão de alimentação do driver, PWM o valor (de 0 a 1) do valor do *duty cycle* do PWM, i a corrente da armadura, L_M e R_m são, respectivamente, a indutância e a resistência da armadura e K_e é a constante de força contra eletromotriz.

Além disso, o torque no motor é dado por $\tau_m = K_t \cdot i$, onde K_t é uma constante de torque do motor.

Com tais considerações, obtém-se o torque aplicado a cada roda (17).

$$\tau_r = \frac{K_t}{R_m}\left(PWM \cdot V_{max} - K_e(\dot{\theta}_r - \dot{\theta}_p)\right). \quad (17)$$

Já o torque externo aplicado no centro de massa do corpo é, por reação, menos a soma dos torques das rodas (18).

$$\tau_p = -(\tau_R + \tau_L) \quad (18)$$

Substituindo o lagrangiano (13) e os torques das rodas (17) e do corpo (18) na equação do método de Euler-Lagrange, obtém-se como solução as equações não lineares da dinâmica do robô.

Por fim, foi feita uma mudança de coordenadas para que o sistema de controle pudesse atuar diretamente sobre o ângulo de guinada (α) e sobre a velocidade na direção y, conforme é visto na Figura 1. As mudanças de coordenadas são vistas em (19).

$$\begin{aligned} y &= \tfrac{R}{2}(\theta_R + \theta_L) \\ \alpha &= \tfrac{R}{D}(\theta_R - \theta_L) \end{aligned} \quad (19)$$

O sistema não linear obtido foi linearizado para o ponto de operação de $\theta_p = 0, \dot{\theta}_p = 0, \dot{y} = 0, \alpha = 0, \dot{\alpha} = 0$, obtendo um modelo do formato $\dot{x} = Ax + Bu$, em que x é o vetor de estados (20). Note que o estado de posição y não foi levado em conta no modelo final, pois não se deseja controlar esse estado, mas sim sua velocidade $\dot{y}$.

$$x = \begin{bmatrix} \theta_p & \dot{\theta}_p & \alpha & \dot{\alpha} & \dot{y} \end{bmatrix}^T \quad (20)$$

Para a obtenção dos valores numéricos dos parâmetros da matriz de inercia do robô, do momento de inercia da roda e a posição do centro de massa, foi feito um modelo 3D do robô, visto na Figura 2. Foram realizados experimentos para estimar as constantes de torque e de velocidade dos motores.

Os parâmetros do protótipo são apresentados na Tabela 1.

Ao substituir os valores numéricos dos parâmetros nas matrizes A e B, obteve-se (21).

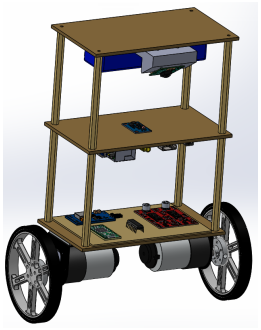

Figura 2: Modelo 3D do robô.

Tabela 1: Parâmetros do modelo.

Parâmetro	Valor
D	0,19 m
L	0,098 m
R	0,045 m
m_r	0,022 kg
m_p	0,7085 kg
I_{xx}	$2{,}87\times10^{-3}$ kg/m^2
I_{yy}	$7{,}95\times10^{-3}$ kg/m^2
I_{zz}	$5{,}91\times10^{-3}$ kg/m^2
J_r	$31{,}39\times10^{-6}$ kg/m^2
K_t	0,118 Nm/A
K_e	0,229 V s/rad
R_m	3,6 Ω

$$A = \begin{bmatrix} 0,00 & 1,00 & 0,00 & 0,00 & 0,00 \\ 192,79 & -4,64 & 0,00 & 0,00 & 103,19 \\ 0,00 & 0,00 & 0,00 & 1,00 & 0,00 \\ 0,00 & 0,00 & 0,00 & 3,68 & 0,00 \\ -17,03 & 0,43 & 0,00 & 0,00 & -9,54 \end{bmatrix}$$

$$B = \begin{bmatrix} 0,00 & 0,00 \\ -121,67 & -121,67 \\ 0,00 & 0,00 \\ -45,61 & 45,61 \\ 11,25 & 11,25 \end{bmatrix} \quad (21)$$

3 Construção do Robô

O robô foi construído a partir de três chapas retangulares de MDF afastadas por hastes de latão.

Os motores DC utilizados têm redução com fator de 18,75:1 e são acoplados aos sensores, *Encoders* com resolução de 64 pulsos por revolução. Efetivamente, devido à redução, a resolução é de $18,75 \times 64 = 1200$.

O algoritmo de controle é executado no microcontrolador *Teensy 3.2*, que foi programado em C++ através da plataforma MBED (MBED, n.d.).

Duas pontes-H VNH2SP30 foram usadas como drivers de potência que alimentam os motores.

Para alimentação foi utilizada uma bateria LiPo de 3 células com carga de 2200 mAh. A conversão de tensão de 12V para 5V foi feita por uma UBEC (*Universal Battery Eliminator Circuit*) de 5A.

Para determinação do ângulo de arfagem foi utilizada uma IMU (*Inertial Measurement Unit*) do modelo Mpu6050. Para a fixação deste sensor foi utilizado um suporte feito em impressora 3D.

Os drivers de potência, o microcontrolador, a UBEC e todas as ligações com sensores foram agrupados em uma placa de circuito de modo a minimizar ruídos e problemas de mau contato.

O processamento de imagem é realizado por um *Raspberry Pi 2011.12* e as imagens são obtidas pela câmera *Raspeberry Pi Camera Module v2*. Foi projetado e impresso em 3D um suporte para a câmera. A medida do ângulo de guinada bem como da velocidade linear foram transmitidas do Raspberry para a Teensy por meio da comunicação serial SPI (*Serial Peripheral Interface*). A foto do protótipo é apresentada na Figura 3.

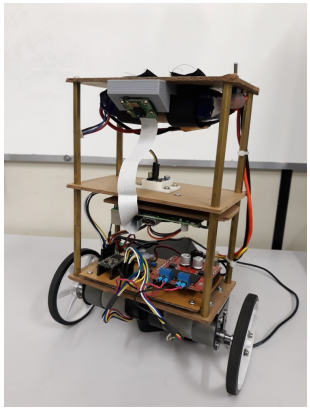

Figura 3: Protótipo do robô autoequilibrado.

4 Processamento Das Imagens

As imagens obtidas através da câmera foram analisadas de modo a reconhecer uma faixa branca em um chão preto e enviar a referência para a velocidade linear ($\dot{y}$) e a medida do ângulo de guinada do robô (α).

A programação foi desenvolvida em C++, com o auxílio de (Kim, n.d.).

Para detectar a linha foram implementados dois métodos diferentes: o método do gradiente e o método de limiarização. Para ambos foi feito o cálculo sobre a imagem em tons de cinza.

4.1 Método do Gradiente

O primeiro método procura as fronteiras da faixa branca calculando os máximos do gradiente de cada linha da imagem e acha o centro da faixa através do ponto médio entre as posições das fronteiras.

Para cada linha é calculado o módulo do gradiente horizontal tal como apresentado em (22). Em que $Grad$ é a imagem que contém o resultados do gradiente, Cam_{gray} é a imagem da câmera já em tons de cinza e l e c são, respectivamente, os números linha e da coluna da imagem.

$$Grad(l,c) = |Cam_{gray}(l,c+1) - Cam_{gray}(l,c-1)| \tag{22}$$

A borda da faixa é a região em que há maior diferença de cor de um pixel para o outro, considerando que não haja componentes espúrias na imagem. Desta forma, a borda da faixa é também a região em que o gradiente horizontal tem o seu valor máximo.

Contudo, caso haja alguma falha na faixa ou esta tenha suas bordas pouco definidas, é possível que os dois pontos de máximo do módulo do gradiente estejam do mesmo lado da faixa. A fim de evitar esta falha, cada linha da imagem foi dividida em vinte trechos consecutivos. Foi detectado o máximo para cada um desses trechos. Os dois trechos com o maior valor de máximo são os que contêm a borda da faixa, evitando assim, máximos consecutivos.

4.2 Método de Limiarização

Este método consiste em estabelecer um limiar que divide os tons de cinza em branco ou preto. Ou seja, para valores maiores do que o limiar estabelecido, o pixel é declarado como branco, e para valores menores do que o limiar, o pixel é dito preto.

Este limiar foi obtido através de experimentos com a câmera no trajeto estabelecido para o robô.

Este método diz que a faixa branca está onde o maior número de pixels brancos consecutivos estiver. Marcando o ponto inicial e final deste trecho, pode-se calcular o ponto médio da faixa.

Os dois métodos foram aplicados no processamento de imagens. Assim que a imagem é obtida, é utilizado, para cada linha, o método do gradiente. Em seguida os pontos da borda e do centro passam por uma verificação de coerência do resultado. Isso acontece analisando se o centro da linha é branco (tem valor maior do que o limiar), se a borda esquerda tem os pontos à sua esquerda pretos e à sua direita brancos e se acontece o contrário para a borda direita da faixa. Caso o resultado passe por esses testes, ele é aprovado. Caso o teste acuse falha em alguma das bordas, é repetido o método do gradiente, mas desta vez ignorando o trecho que contém o ponto que falhou no teste. Mais uma vez o teste é aplicado, em caso de falha, é alterado o método, usando desta vez o método da limiarização. Aplicando o teste verifica-se se este resultado é válido. Caso não seja, o algoritmo retorna o ponto como valor inválido.

Tendo o ponto médio da faixa, foi calculado o ângulo de guinada. Com algumas medidas expe-

rimentais, pode-se determinar uma aproximação para a relação *pixel/metro* em uma dada região da imagem. Com a medida em metros do desvio entre o centro da faixa e o centro da imagem e a distância do ponto em que se toma a medida até o robô, pode-se calcular o ângulo (23).

$$\alpha = \text{atan2}\left(\frac{\Delta x}{\Delta y}\right) \tag{23}$$

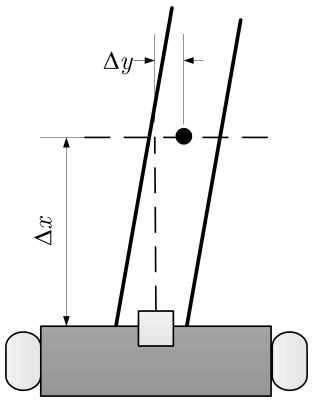

Figura 4: Cálculo do ângulo de guinada.

A Figura 5 apresenta uma foto da linha no chão com o cálculo do ângulo de guinada. Neste caso, o ângulo medido foi de $-1,686606°$.

Figura 5: Exemplo de cálculo da referência do ângulo de guinada.

Também foi realizado um cálculo para determinar a velocidade linear do robô. A velocidade é determinada de modo que se o ângulo de guinada for grande, o robô deve ficar mais lento. De outra forma, é possível que este perca a faixa durante seu trajeto, caso o controle não seja rápido o suficiente para levar o ângulo à referência. Para isso foi utilizada a equação inspirada no filtro de cosseno levantado (24), de modo a descrever referência da velocidade no eixo y em função do ângulo de guinada α.

$$\dot{y}_r(\alpha) = \begin{cases} 1 & , |\alpha| \leq \frac{1-\beta}{2T} \\ \frac{1}{2}\left[1+\cos\left(\frac{\pi T}{\beta}\left[|\alpha|-\frac{1-\beta}{2T}\right]\right)\right] & , \frac{1-\beta}{2T} < |\alpha| \leq \frac{1+\beta}{2T} \\ 0 & , \text{caso contrário} \end{cases} \tag{24}$$

Os parâmetros β, T utilizados foram os seguintes: $\beta = 0,7$ e $T = 2,8648$, de modo a obter a curva apresentada na Figura 6.

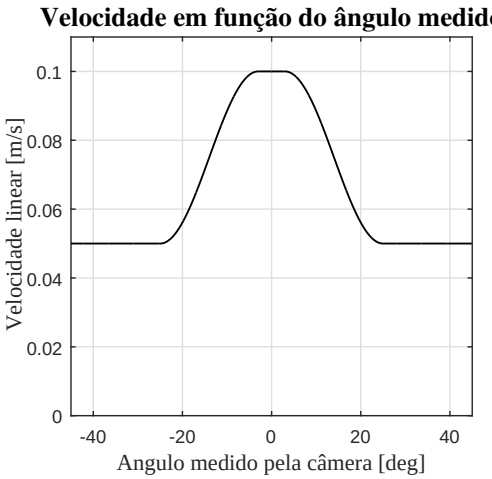

Figura 6: Velocidade linear em função do ângulo de guinada

5 Controle Digital LQR

O controle implementado para o equilíbrio e rastreamento de trajetória foi o controle ótimo LQR Discreto (*Discrete Linear Quadratic Regulator*) (Franklin et al., 2006)

Este controle é uma realimentação de estados que calcula a matriz de ganho K visando minimizar um funcional (25).

$$J = \frac{1}{2}\sum_{n=0}^{\infty}(x^\top[n]Qx[n] + u^\top[n]Ru[n]), \tag{25}$$

sendo $x[n]$ e $u[n]$ os vetores de estado e de entrada, respectivamente. Os termos Q e R são matrizes de ponderação simétricas, sendo Q semi-definida positiva e R definida positiva. Ao minimizar J, chega-se em

$$K = R^{-1}\Gamma^\top X, \tag{26}$$

onde Γ é uma matriz do modelo discreto e X é a solução da equação algébrica de Riccati discreta (DARE)

$$\Phi^\top(X - X\Gamma(R+\Gamma^\top X\Gamma)^{-1}\Gamma^\top X)A + Q = X. \tag{27}$$

O modelo dinâmico foi discretizado para um tempo de amostragem de $T = 0.01s$ (frequência de $100Hz$) (28).

$$\begin{aligned} x[n+1] &= \Phi x[n] + \Gamma u[n] \\ y[n] &= Cx[n] + Du[n], \end{aligned} \tag{28}$$

em que $C = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$. Como o controle visa seguir trajetórias tanto para o estado velocidade linear ($\dot{y}$) quanto para o ângulo de guinada

(α), foram adicionados dois integradores no erro destes estados, criando dois novos estados. Para tanto, foi escrito o sistema aumentado, alterando as matrizes Φ e Γ do sistema (Fadali et al., 2013).

Dessa forma, as matrizes Φ^{Aug} e Γ^{Aug} obtidas do sistema aumentado são:

$$\Phi_{Aug} = \begin{bmatrix} \Phi & 0_{5,2} \\ -C & I_{2,2} \end{bmatrix} \quad \Gamma_{Aug} = \begin{bmatrix} \Gamma \\ 0_{2,2} \end{bmatrix} \tag{29}$$

As matrizes Q e R foram escolhidas segundo a Regra de Bryson:

$$\begin{aligned} Q &= diag(1459,0;\ 8,2;\ 1,2;\ 0,5;\ 0,0;\ 0,3;\ 1,0) \\ R &= \begin{bmatrix} 100,0 & 0,0 \\ 0,0 & 100,0 \end{bmatrix} \end{aligned} \tag{30}$$

Para estas matrizes Q e R foram obtidos os seguintes ganhos:

$$K = \begin{bmatrix} -4,0 & -0,82 & -1,2 & -0,20 & -7,0 & \dots \\ -4,0 & -0,82 & 1,2 & 0,20 & -7,0 & \dots \\ \dots & 0,037 & 0,054 \\ \dots & 0,037 & 0,054 \end{bmatrix} \tag{31}$$

6 Resultados Práticos

Para estimar a velocidade das rodas foi usada a derivada por *backward*. Para atenuar o efeito do ruído na estimação, essas velocidades angulares passaram por um filtro discreto de primeira ordem com constante de tempo $\tau_F = 5 \times 10^{-2} s$.

Para a obtenção do ângulo de arfagem do robô foi feita uma fusão dos sensores giroscópio e acelerômetro por meio de um filtro de Kalman.

A Figura 7 apresenta os resultados para quatro dos estados do sistema: ângulo de arfagem, velocidade do ângulo de arfagem, ângulo de guinada e velocidade linear.

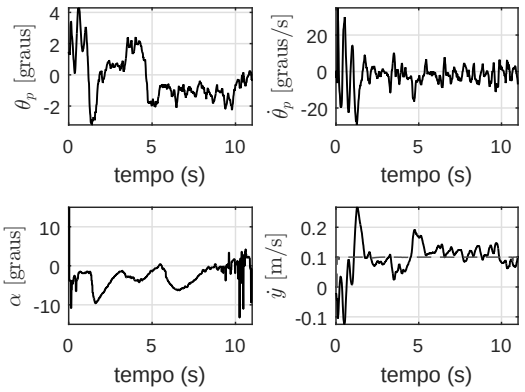

Figura 7: Resposta real do sistema.

Observa-se que o robô manteve-se estável, com o ângulo de arfagem (θ_p) variando entre -4 e +4 graus. A velocidade deste ângulo ($\dot{\theta}_p$) também ficou limitada. Verifica-se que o ângulo de guinada (α) varia em alguns momentos, sendo estes os momentos em que o robô passa por curvas do circuito. A velocidade linear ($\dot{y}$) (linha sólida) manteve-se em torno de sua referência (linha tracejada).

O sinal de controle, é apresentado na Figura 8. Observa-se que, para ambos os motores, os *duty cycles* dos sinais PWM ficaram limitados entre ± 1, bem distantes do limite de saturação.

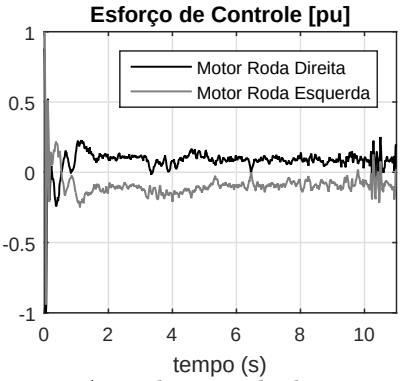

Figura 8: Ação de controle do sistema real.

Um vídeo do sistema pode ser encontrado no canal Youtube do Laboratório de Controle Aplicado da Escola Politécnica da USP (LCA), pelo link `www.youtube.com/c/laboratoriodecontroleaplicadopoliusp`.

7 Conclusões

O robô foi construído, modelado com base nas equações de Euler-Lagrange e estabilizado com sucesso com o controlador digital LQR. A estabilização mostrou bons resultados, bem como o controle para seguir trajetórias através de integradores no erro entre a referência dos estados e os estados em si. O processamento de imagens se mostrou bastante robusto para as condições estabelecidas, conseguindo calcular as referências para o ângulo de guinada e para a velocidade linear. De forma geral, os resultados foram satisfatórios. Para trabalhos futuros, pretende-se melhorar o controle para seguimento de referência, de modo que se possa aumentar a velocidade do robô durante o percurso. Outra melhoria a ser feita é deixar o processamento de imagens habilitado para casos mais gerais, para outras combinações de cores de faixa e piso, e utilizar outro método para implementar a limiarização.

Referências

Craig, J. J. (2005). *Introduction to Robotics - Mechanics and Control*, 3a. ed. edn, Prentice-Hall, Upper Saddle River, NJ, USA.

Fadali, M. S., and Visioli, A. (2013). *Digital Control Engineering*, 2nd edn, Academic Press, Boston.

Franklin, G., Powell, J. and Workman, M. (2006). *Digital Control of Dynamic Systems*, 3rd ed. (reprint) edn, Ellis-Kagle Press.

Kim, H. Y. (n.d.). Cekeikon5. rotinas e programas em c++ para processamento de imagens e visão computacional, `http://www.lps.usp.br/hae/software/cekeikon5.html`. Acessado em 22/03/2018.

Kim, S. and Kwon, S. (2017). Nonlinear optimal control design for underactuated two-wheeled inverted pendulum mobile platform, *IEEE/ASME Transactions on Mechatronics* **22**(6): 2803–2808.

Kralev, J., Slavov, T. and Petkov, P. (2016). Design and experimental evaluation of robust controllers for a two-wheeled robot, *International Journal of Control* **89**(11): 2201–2226.

MBED (n.d.). Arm mbed, `https://www.mbed.com`.

Ogata, K. (2003). *Engenharia de Controle Moderno*, 4 ed. edn, Prentice-Hall, São Paulo, SP.

Raffo, G. V., Ortega, M. G., Madero, V. and Rubio, F. R. (2015). Two-wheeled self-balanced pendulum workspace improvement via underactuated robust nonlinear control, *Control Engineering Practice* **44**(Supplement C): 231 – 242.