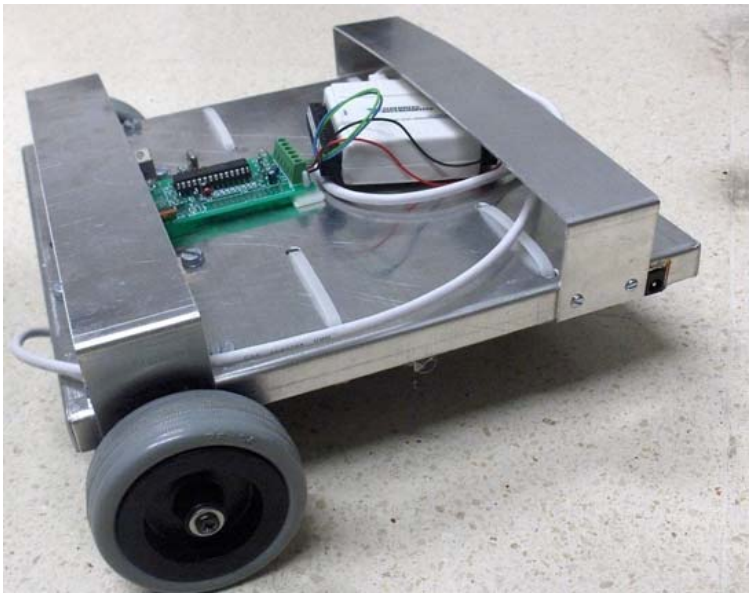
Trabalho Prático 2 – Controlo Óptimo

Licenciatura em Engenharia Mecânica
Ramo de Automação e Robótica
Prof. Responsável: Miguel Ayala Botto

Grupos 7 e 8

Controlo da Trajectória do robot *Rasteirinho*



Trabalho realizado por:
Tiago Rita Nº 53982
José Feio Nº 54005
Edwin Carvalho Nº54018
Pedro Morgado Nº54034

Ano lectivo 2006/2007
IST

Índice

1.Objectivo

O trabalho prático 2 da disciplina de Controlo Óptimo tem como principal objectivo, o Controlo da Trajectória do robot Rasteirinho através da aplicação de técnicas de Identificação de Sistemas e Controlo Óptimo, utilizando um software de programação (Matlab).

Para a elaboração do trabalho prático, recorreu-se ao conhecimento adquirido nas aulas de Controlo Óptimo sobre a *toolbox* de Identificação de Sistemas do Matlab, bem como ao modelo de grafos Simulink do Matlab, disponibilizado por colegas, pioneiros no projecto de controlabilidade dos robots *Rasteirinhos*.

2.Fundamentos Teóricos

De forma a elaborar correctamente o controlo óptimo do robot Rasteirinho, devem-se ter por bases algumas noções essenciais sobre Funções de Transferência discretas, Espaços de Estado e sua observação, Identificação de Sistemas e Controlo Óptimo e sub-Óptimo. Como tal, serão apresentadas as noções essenciais a reter para a elaboração do Trabalho Prático 2.

2.1.Noções sobre funções de transferência de sistemas discretos

Os sistemas físicos reais são por natureza sistemas dinâmicos contínuos, no entanto torna-se mais fácil estudar sistemas discretos devido à introdução do computador digital. O computador digital é actualmente um elemento chave para a simulação, identificação e controlo de sistemas físicos.

Para converter os sinais contínuos em sinais discretos, de modo ao computador poder proceder à identificação do sistema, usa-se um conversor A/D (analógico/digital). À saída do computador, deve usar-se um conversor D/A (digital/analógico) que reconstrói o sinal contínuo, a partir de um trem de impulsos.

Surge assim a **transformada Z,** com o objectivo de eliminar o termo irracional, $e^{-kT_0 s}$, que resulta sempre que se aplica a Transformada de Laplace a um sinal descrito por um trem de impulsos. A definição Geral de Transformada Z de um sinal é então a seguinte:

$$Y(z) = \sum_{k=0}^{+\infty} y(kT_0) z^{-k}$$

Nota: Consultar tabela das transformadas Z, (pág.101 do livro da cadeira de Controlo Óptimo)

Ao contrário da transformada de Laplace, a transformada Z inversa não é única, ou seja, não origina necessariamente a função original no domínio contínuo do tempo.

Apenas os valores múltiplos dos instantes de amostragem são coincidentes com os valores de f(t).

Transformada Z inversa: $f(kT_0) = Z^{-1}[F(z)]$

Para resolver este problema aplica-se a transformada usando o método da expansão em fracções parciais (pág. 103 do livro da cadeira de controlo Óptimo).

- *Função de transferência discreta G(z)*

Pode ser obtida a partir de:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{Z[y^*(t)]}{Z[u^*(t)]} \quad \text{ou} \quad G(z)=Z[G(s)]$$

A função de transferência discreta é dada na forma genérica por:

$$G(z) = \frac{Y(z)}{U(z)} = \frac{b_0 z^n + b_1 z^{n-1} + \ldots + b_{n-1} z + b_n}{z^n + a_1 z^{n-1} + \ldots + a_{n-1} z + a_n}$$

O ganho estático K de uma função de transferência G(z) é dado pela expressão que relaciona os termos dos polinómios numerador e denominador de G(z):

$$K = \lim_{z \to 1} G(z) \Leftrightarrow K = \frac{b_0 + b_1 + ... + b_n}{1 + a_1 + ... + a_n}$$

2.2.Representação em espaço de estados de sistemas discretos

Encontramos agora a necessidade de determinar o equivalente discreto de um sistema contínuo representado em espaço de estados. Representam-se de seguida, duas das aproximações possíveis:

- *Aproximação rectangular*

Considerando um sistema LIT: $\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned}$

Através de uma formulação matemática que envolve uma aproximação para o termo da derivada $\dot{x}(t)$, e para um período de amostragem T_0, (ver pagina 124 do livro da cadeira) chega-se à seguinte representação em espaço de estados discreta:

$$\begin{cases} x(k+1) = \Phi x(k) + \Gamma u(k) \\ y(k) = Cx(k) + Du(k) \end{cases} \quad \text{com} \quad \begin{aligned} \Phi &= (I + AT_0) \\ \Gamma &= BT_0 \end{aligned}$$

- *Considerando um retentor de ordem zero (ZOH) na entrada*

Para o mesmo sistema LIT anterior, a solução da sua equação de estado é a seguinte:

$$x(t) = e^{At} x(0) + \int_0^t e^{A(t-\tau)} Bu(\tau) d\tau$$

Onde $e^{At} = \Phi(t)$, que corresponde à matriz de transição de estados

Considerando um ZOH na entrada, ou seja, $u(t)=u(kT_0)$, e desenvolvendo matematicamente (página 125 do livro da cadeira), chega-se á representação em espaço de estados discreta:

$$\begin{cases} x(k+1) = \Phi x(k) + \Gamma u(k) \\ y(k) = Cx(k) + Du(k) \end{cases}$$

com $\begin{aligned} \Phi &= e^{AT_0} = L^{-1}\{[sI - A]^{-1}\}_{t=T_0} \\ \Gamma &= (\int_0^{T_0} e^{Aq} dq) B = \int_0^{T_0} \Phi(q) B dq \end{aligned}$

2.3.Identificação do Modelo dinâmico do Sistema

A identificação de um sistema é essencial para poder definir o seu comportamento.

A identificação de sistemas baseia-se na evolução dos sinais de entrada e saída do sistema desconhecido, de modo a determinar o seu modelo dinâmico.

2.3.1.Métodos de identificação de sistemas

Existem dois tipos de métodos:

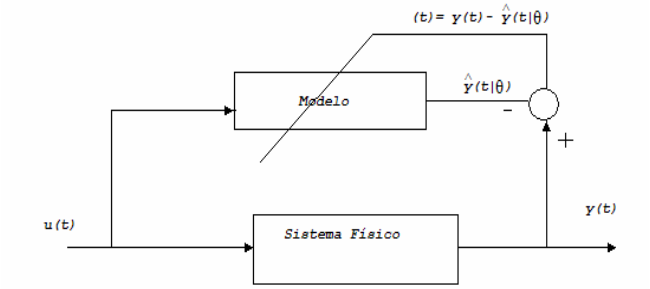
1.Via resposta em frequência: resultam num modelo em função de transferência contínua através do método e Bode ou pelo método de Levy.

2.Via resposta no tempo: resultam num modelo em função de transferência (continua ou discreta), ou numa representação em espaço de estados (continua ou discreta).

Primeiro excita-se o sistema com sinais de teste aperiódicos e retiram-se os dados experimentais resultantes, de seguida aplica-se o método de identificação para estimar os parâmetros da função de transferência ou do modelo de estado.

Neste trabalho estuda-se o método de identificação via resposta no tempo através da minimização do erro de predição

2.3.2.Identificação por minimização do erro de predição



O objectivo consiste em determinar os parâmetros θ de um modelo para que este possua as mesmas características dinâmicas do sistema a identificar, através da minimização do erro de predição medido em cada instante:

$$\varepsilon(t) = y(t) - \hat{y}(t \mid \theta)$$

Em que:

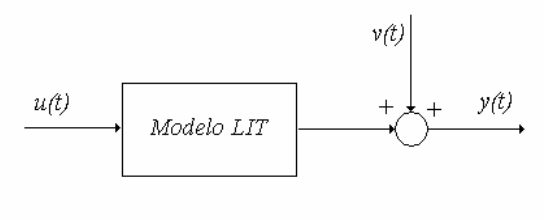
- $y(t)$, medida da saída do sistema em cada instante de amostragem
- $\hat{y}(t \mid \theta)$, estimação da saída do modelo, com base numa determinada parametrização θ (estimação a um passo)
- $\varepsilon(t)$, é o erro de estimação dos parâmetros do modelo em cada instante de amostragem
- θ, é o valor numérico dos parâmetros do modelo

Este método não necessita de armazenar grandes quantidades de dados em memória. É aplicável a sistemas variantes no tempo, sendo portanto utilizado em anéis de controlo adaptativo. Permite ao operador terminar o processo de identificação sempre que desejar, pois obtém sempre resultados.

- ***Modelos matemáticos e modelos de predição (estimação)***

De modo a identificar o modelo dinâmico torna-se necessário considerar algumas hipóteses simplificativas quanto ao sistema físico representado. Para além de se considerarem os sistemas físicos como sistemas estáveis, o modelo que representa a dinâmica do sistema pode ser representado por um

modelo linear e invariante no tempo (LIT). As perturbações do sistema são representadas por um sinal aditivo à saída do modelo do sistema (ruído) representado por v(t) no diagrama seguinte:

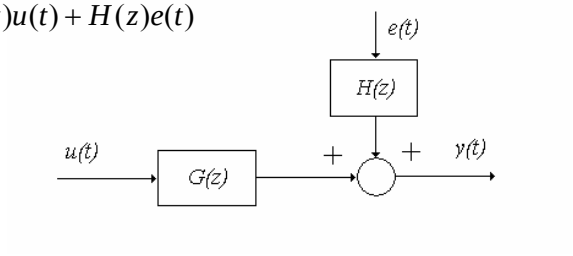


A componente v(t) representa a existência de perturbações externas que não podem ser reproduzidas pelo modelo do sistema, mas que afectam o próprio sistema. Assume-se, então, que o modelo das perturbações corresponde à saída de um sistema LIT, cuja entrada se considera ser a sequencia aleatória, {e(t)}, correspondente ao ruído branco, com media nula e variância fixa conhecida, onde {e(t)} é imprevisível:

$$v(t) = \sum_{k=0}^{+\infty} h(k)e(t-k)$$

Após um desenvolvimento matemático chega-se à equação que representa o modelo matemático genérico do sistema:

$$y(t) = G(z)u(t) + H(z)e(t)$$



Em que:

- G(z) representa as propriedades do sistema físico
- H(z) representa o modelo das perturbações, que descreve a forma como estas afectam a saída do sistema
- G(z) e H(z) correspondem a fracções racionais. Casos particulares resultam em modelos de predição do sistema

Existe uma variedade de modelos representativos da dinâmica de um sistema. Entre os modelos paramétricos de predição mais usados estão os ARX, FIR, ARMAX, ARMA, MA, BJ, OE e o Modelo genérico.

- *Modelo ARX*

Este modelo é designado por *equation model error*, e considera a seguinte equação ás diferenças para o modelo do sistema:

$$\underbrace{[1 + a_1 z^{-1} + \ldots + a_{n_a} z^{-n_a}]}_{A(z^{-1}),\ \text{parte AR}} y(t) = \underbrace{[b_1 z^{-1} + \ldots + b_{n_b} z^{-n_b}]}_{B(z^{-1}),\ \text{parte X}} u(t - n_k) + e(t)$$

Forma compacta: $A(z^{-1})y(t) = B(z^{-1})u(t-n_k) + e(t)$

Assim temos para este caso: $G(z) = \frac{B(z^{-1})}{A(z^{-1})}$ e $H(z) = \frac{1}{A(z^{-1})}$

O modelo ARX é representado por ARX(n_a, n_b, n_k), em que:

- n_a é a ordem do polinómio $A(z^{-1})$
- n_b é a ordem do polinómio $B(z^{-1})$
- n_k é o atraso

➢ *Modelo FIR (Finite Impulse Response)*

É uma forma particular do modelo ARX em que $A(z^{-1})=1$. A equação matemática é a seguinte:

$$y(t) = B(z^{-1})u(t-n_k) + e(t)$$

➢ *Modelo ARMAX (Moving Average)*

Descrito pela equação ás diferenças:

$$[1 + a_1 z^{-1} + ... + a_{n_a} z^{-n_a}]y(t) = [b_1 z^{-1} + ... + b_{n_b} z^{-n_b}]u(t-n_k) + [1 + c_1 z^{-1} + ... + c_{n_c} z^{-n_c}]e(t)$$

Forma compacta: $A(z^{-1})y(t) = B(z^{-1})u(t-n_k) + C(z^{-1})e(t)$

Assim temos para este caso: $G(z) = \frac{B(z^{-1})}{A(z^{-1})}$ e $H(z) = \frac{C(z^{-1})}{A(z^{-1})}$

O modelo ARMAX é representado por ARMAX(n_a, n_b, n_c, n_k), em que:

- n_a é a ordem do polinómio $A(z^{-1})$
- n_b é a ordem do polinómio $B(z^{-1})$
- n_c é a ordem do polinómio $C(z^{-1})$
- n_k é o atraso

O modelo ARMAX difere do modelo ARX na medida em que possui mais graus de liberdade, já que se considera neste modelo o ruído, e(t), ponderado pelos coeficientes do polinómio $C(z^{-1})$. Tem-se assim uma ponderação estendida como uma média pesada de e(t) ao longo do tempo, não nula (média móvel: *Moving Average*).

➢ *Outros modelos, na forma compacta:*

Modelo ARMA: $A(z^{-1})y(t) = C(z^{-1})e(t)$

Modelo MA: $y(t) = C(z^{-1})e(t)$

ModeloBJ: $y(t) = \frac{B(z^{-1})}{F(z^{-1})}u(t) + \frac{C(z^{-1})}{D(z^{-1})}e(t)$

ModeloOE: $y(t) = \frac{B(z^{-1})}{F(z^{-1})}u(t) + e(t)$

Modelo genérico: $A(z^{-1})y(t) = \frac{B(z^{-1})}{F(z^{-1})}u(t) + \frac{C(z^{-1})}{D(z^{-1})}e(t)$

2.3.3.Validação do Modelo

Após a identificação do modelo, é necessário verificar a qualidade deste, ou seja, verificar se o modelo obtido é ou não adequado à situação em que é aplicado.

O modelo do sistema considera-se bem identificado se conseguir captar as características dinâmicas e estacionarias do sistema físico, usando o conjunto de dados de estimação. Deve-se proceder a uma validação cruzada, ou seja, testar se o modelo tem capacidade para reproduzir um conjunto de dados diferente dos usados para a estimação (*conjunto de dados de teste*).

As técnicas mais usadas para a validação são:

- Validação por simulação do modelo
- Validação por analise dos resíduos

➢ <u>*Validação por simulação do modelo:*</u> Consiste em comparar as saídas, real e do modelo simulado, usando a mesma entrada em ambos os casos. Considera-se, sem perda de generalidade que $\hat{y}(t \mid \theta) = \theta^T \varphi(t)$

A qualidade do modelo pode ser quantificada através do somatório do erro quadrático:

$$SSE = \sum_{t=1}^{N} [y(t) - \hat{y}(t \mid \theta)]^2$$

Em que $\varphi(t)$ (vector regressor), pode ser actualizado usando um dos seguintes métodos:

- *Predição 1-passo à frente: a* actualização dos termos y(t-i) do vector regressor é baseada na saída amostrada do sistema real, em cada instante. Pretende-se verificar a capacidade do modelo em prever a saída do sistema no instante t, usando a informação real do sistema até ao instante anterior, t-1.
- *Predição k-passos à frente: a* actualização dos termos y(t-i) do vector regressor é baseada na saída do modelo de regressão, durante *k* instantes. Pretende-se verificar a capacidade do modelo em prever a saída do sistema no instante t, usando a informação real do sistema até ao instante t-k. É útil para verificar se modelo representa com eficácia o comportamento transitório do sistema.

- *Simulação autónoma do modelo:* a actualização dos termos y(t-i) do vector regressor é baseada nas estimativas da saída do modelo de regressão obtidas nos instantes anteriores. À semelhança dos casos anteriores, tem como objectivo avaliar a capacidade do modelo em prever a saída do sistema no instante t, usando informação, também estimada, dos instantes anteriores.

➢ <u>*Validação por análise de resíduos:*</u> pretende-se com este método verificar se o modelo tem a capacidade de distinguir o que na saída é influência das entradas do sistema, u(t), e dos resíduos do modelo, e(t).

A independência entre u(t) e e(t) é testada através do cálculo da função de correlação cruzada entre estas. Para um modelo perfeito os valores de correlação deverão ser baixos.

2.4. Controlo Óptimo

A formulação de um problema de controlo óptimo para sistemas dinâmicos baseia-se essencialmente numa *função de custo* e nas suas *restrições*:

Função de custo: hiper-superfície genérica no espaço de ordem n+m, variante no tempo, que caracteriza o desempenho desejado para o sistema:

$$J_i = \phi(N, x_N) + \sum_{k=i}^{N-1} L^k(x_k, u_k) \qquad \text{no intervalo [i, N]}$$

A primeira parcela corresponde ao peso da solução final, enquanto que a segunda parcela pesa a solução do problema durante os *n* passos.

Restrições: corresponde à dinâmica do sistema, que se move no espaço ao longo do tempo

$$f^k(x_k, u_k) - x_{k+1} = 0 \qquad \text{com } x \in \Re^n, u \in \Re^m$$

Incluindo as restrições na equação de custo, obtém-se:

$$J'_i = \phi(N, x_N) + \sum_{k=i}^{N-1} L^k(x_k, u_k) + \sum_{k=i}^{N-1} \lambda_{k+1}^T (f^k(x_k, u_k) - x_{k+1})$$

Onde λ_k representam os multiplicadores de Lagrange.

O objectivo do Controlo Óptimo resulta em encontrar a acção de controlo óptima, u_k* para o intervalo de interesse, k=[i, N-1], transportando o sistema ao longo da sua trajectória de estado óptima, x_k*, minimizando a função de custo, J_i .

2.4.1.Regulador Linear Quadrático (LQR) discreto

Usando o LQR discreto, é possível determinar os ganhos óptimos representados pela matriz K, obedecendo à lei:

$$u_k = -K_k x_k$$

e que minimizam a função de custo:

$$J_i = \frac{1}{2} x_N^T S_N x_N + \frac{1}{2} \sum_{k=i}^{N-1} (x_k^T Q x_k + u_k^T R_k u_k)$$

para o modelo em espaço de estados, no domínio discreto do tempo:

$$x_{k+1} = Ax_k + Bu_k$$

As matrizes S_N, Q_k e R_k são matrizes simétricas definidas positivas.

- *Controlador Óptimo por realimentação de estado:*

Equação de Riccati: $S_k = Q_k + A_k^T (S_{k+1}^{-1} + B_k R_k^{-1} B_k^T)^{-1} A_k$, com k<N

Ganhos de Kalman: $K_k = (R_k + B_k^T S_{k+1} B_k)^{-1} B_k^T S_{k+1} A_k$, com k<N

Dado qualquer estado, x_k, é possível saber qual o custo óptimo que resulta de se aplicar o controlo óptimo ao sistema desde o instante inicial ao final. Para isso calcula-se o valor de S_k em cada instante. Como este é calculado em *offline* , podemos saber o custo total da aplicação da lei de controlo óptimo sem nunca a ter aplicado.

- *Controlo sub-óptimo (anel fechado)*

A implementação do LQR tem os seus problemas associados, nomeadamente quanto ao facto de ser necessário armazenar uma elevada quantidade de dados (N matrizes S_k, N matrizes K_k). Outro factor é o problema da análise da estabilidade do anel fechado com dinâmica variante no tempo.

Uma possível solução para estes problemas é a implementação de uma solução sub-óptima, em que a matriz K é constante.

$$u_k = -Kx_k$$

As matrizes Q (que pesa os estados do sistema), e R (que determina a rapidez de resposta do sistema), são também constantes. As equações de Riccati e os ganhos de Kalman obtidos são os seguintes:

Equação de Riccati: $S = Q + A^T \left[S - SB\ (B^T SB + R)^{-1} B^T S \right] A$, $S = S_\infty$

Ganhos de Kalman: $K\ = (R + B^T S\ B\ )^{-1} B^T SA$, $S = S_\infty$

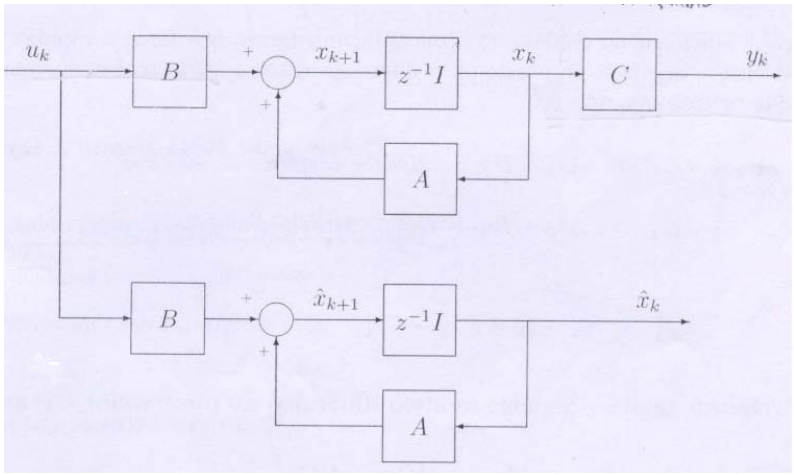
2.4.2.Noções sobre Observadores de Estado

Em casos onde não é possível medir os estados de um sistema torna-se necessário usar um *observador de estado*. O objectivo deste é reconstruir os estados não mensuráveis através da evolução dos sinais de entrada e de saída do sistema, u_k e y_k respectivamente, admitindo que se conhece o seu modelo dinâmico.

Formulação geral de um observador de estado

Para reconstruir o vector de estados, x_k, do sistema é necessário conhecer o modelo dinâmico do sistema, nomeadamente as matrizes A, B e C, e uma estimação do vector de estados, $\hat{x}_k$.

$$\begin{cases} x_{k+1} = Ax_k + Bu_k, & \text{Modelo dinâmico do sistema} \\ \hat{x}_{k+1} = A\hat{x}_k + Bu_k & \text{Modelo dinâmico do Observador} \end{cases}$$



No entanto, usar apenas $\hat{x}_{k+1} = A\hat{x}_k + Bu_k$ como modelo dinâmico do observador não é suficiente para reconstruir com exactidão os estados do sistema. Para além deste problema, há também o facto de que, sem realimentação, o sistema fica dependente do modelo exacto do sistema. Para minimizar esta dificuldade, usa-se o *erro de reconstrução do estado* como vector de estado no modelo dinâmico do observador do anel fechado:

$$e^x_{k+1} = x_{k+1} - \hat{x}_{k+1}$$

Substituindo x_{k+1} e $\hat{x}_{k+1}$ pelos respectivos modelos dinâmicos, obtêm-se o **modelo dinâmico do observador em anel fechado:**

$$e^x_{k+1} = Ax_k + Bu_k - A\hat{x}_k - Bu_k = A(x_k - \hat{x}_k) = Ae^x_k$$

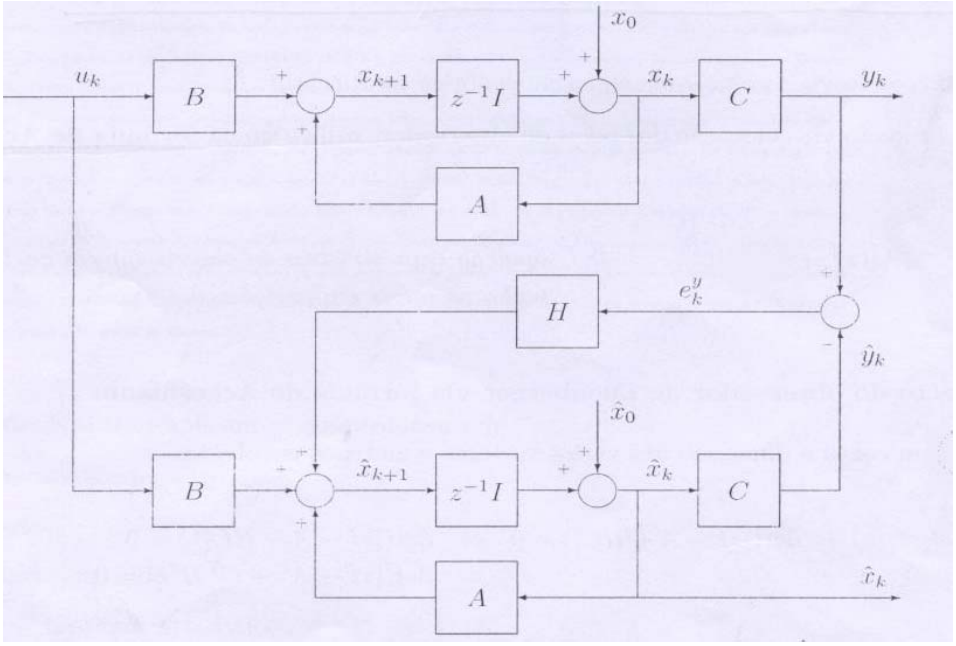
Este último tende para zero sempre que o sistema for assimptoticamente estável.

Pretende-se que, a reconstrução dos estados seja muito mais rápida que a própria saída do sistema. Para isso é necessário adicionar um novo termo ao modelo que tem o objectivo de forçar a convergência dos estados, e que seja proporcional à diferença entre a saída real e a saída estimada pelo observador (erro de reconstrução da saída). Fica-se, assim, com o seguinte modelo dinâmico do observador:

$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + He^y_k$ em que $e^y_k = y_k - \hat{y}_k = Cx_k - C\hat{x}_k$

Substituindo o termo correspondente ao erro, na equação do modelo dinâmico do observador e, desenvolvendo, obtém-se o *modelo dinâmico do observador em anel fechado:*

$e^y_{k+1} = (A - HC)e^x_k \longrightarrow 0$ Sempre que (A-HC) for assimptoticmente estável.



- *Determinação da matriz H do observador*

Projectar a matriz H de modo a tornar os estados observáveis não é de todo trivial. É possível determinar H utilizando o *Observador de Luenberger* (usado neste trabalho) ou recorrendo ao *Filtro de Kalman.*

Observador de Luenberger: a matriz H é determinada recorrendo à semelhança entre este termo e os ganhos de realimentação do problema do Regulador Linear Quadrático. Recorre-se então à **fórmula de Ackermann do observador**:

$$\det([zI - A + HC]) = 0 \Leftrightarrow \det([zI - A^T + C^T H^T]) = 0$$

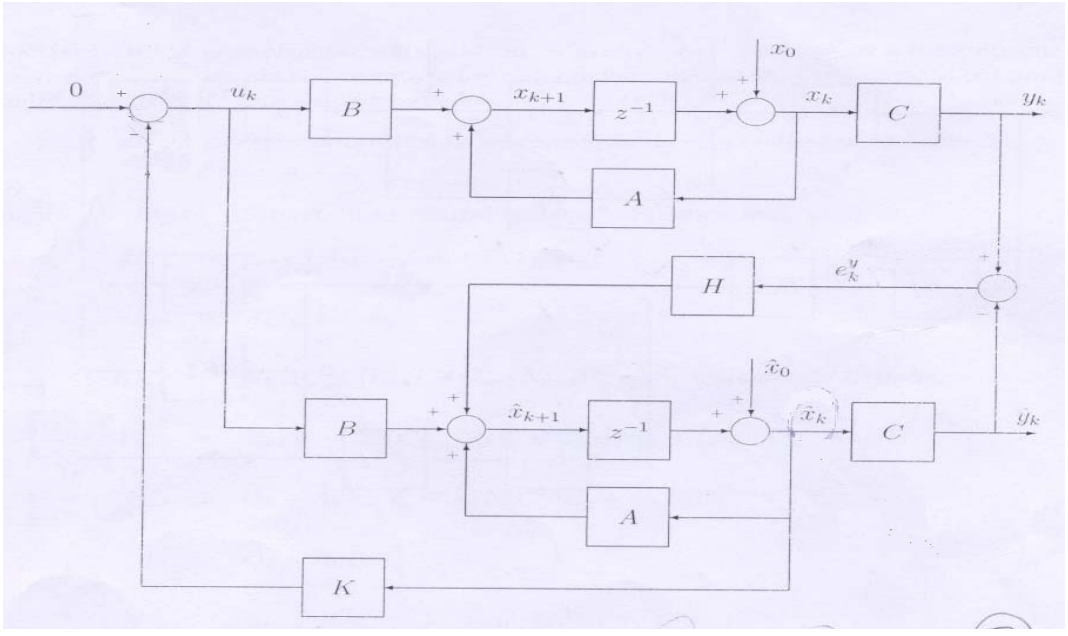
Onde:

$$H^T = [0 \quad 0 \quad \ldots \quad 1][C^T \quad A^T C^T \quad \ldots \quad (A^T)^{n-1} C^T]^{-1} \Phi(A^T)$$

Com:
$$\begin{cases} \Phi(A^T) = a'_n I + a'_{n-1} A^T + \ldots + a'_1 (A^T)^{n-1} + (A^T)^n \\ z^n + a'_1 z^{n-1} + \ldots + a'_{n-1} z + a'_n = 0 \end{cases}$$

Ou seja:
$$H = \Phi(A) \begin{bmatrix} C \\ CA \\ \ldots \\ CA^{n-1} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ \ldots \\ 1 \end{bmatrix}$$

Podemos agora escrever o diagrama de blocos do **controlador óptimo com o observador de estado:**



Sistema: Observador:

$$\begin{cases} x_{k+1} = Ax_k + Bu_k, \quad u_k = -K\hat{x}_k \\ y_k = Cx_k \end{cases} \qquad \begin{cases} \hat{x}_{k+1} = A\hat{x}_k + Bu_k + He_k^y \\ e_k^y = y_k - \hat{y}_k \end{cases}$$

Representação em espaço de estados do anel fechado:

$$\begin{bmatrix} x_{k+1} \\ e_{k+1}^x \end{bmatrix} =) \underbrace{\begin{bmatrix} A - BK & BK \\ 0 & A - HC \end{bmatrix}}_{A^*} \begin{bmatrix} x_k \\ e_k^x \end{bmatrix}$$

A equação característica do anel fechado é então:

$$\det([zI - A^*]) = 0 \Leftrightarrow \det([zI - A + BK)] \cdot \det[(zI - A + HC]) = 0$$

Desta equação, resultam as fórmulas de Akerman do controlador e do Observador (já referida anteriormente).

Para determinar a matriz K, recorre-se à fórmula de Akerman do Controlador: $\det([zI - A + BK)] = 0$

Em que:

$$K = [0 \quad 0 \quad \ldots \quad 1][B \quad AB \quad \ldots \quad A^{n-1}B]^{-1}\Phi(A)$$

Onde:

$$\begin{cases} \Phi(A) = a_n I + a_{n-1}A + \ldots + a_1 A^{n-1} + A^n \\ z^n + a_1 z^{n-1} + \ldots + a_{n-1} z + a_n = 0 \end{cases}$$

3.Descrição do robot Rasteirinho

Antes de iniciar a descrição sobre o corpo do trabalho em si, entendeu-se que seria necessário perceber como é constituído o robot com que se trabalhou. Como tal, uma breve descrição sobre o próprio robot Rasteirinho será apresentada.

O rasteirinho é um robot construído no laboratório de automação e robótica visível na imagem seguinte:

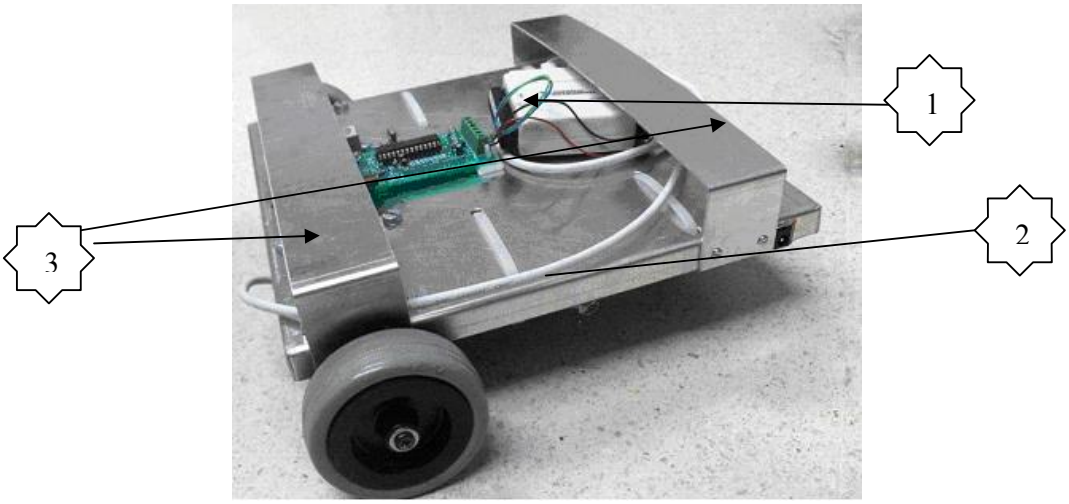


Figura 1 – Representação do Robot Rasteirinho

Como é possível observar, o rasteirinho é composto por uma base metálica sobre a qual se encontra uma placa controladora NI 6102 (1) que é ligado a um computador portátil através de um cabo USB (2). Aparafusado aos lados da base metálica encontram-se duas chapas que têm como função servirem de apoio para um computador portátil (3).

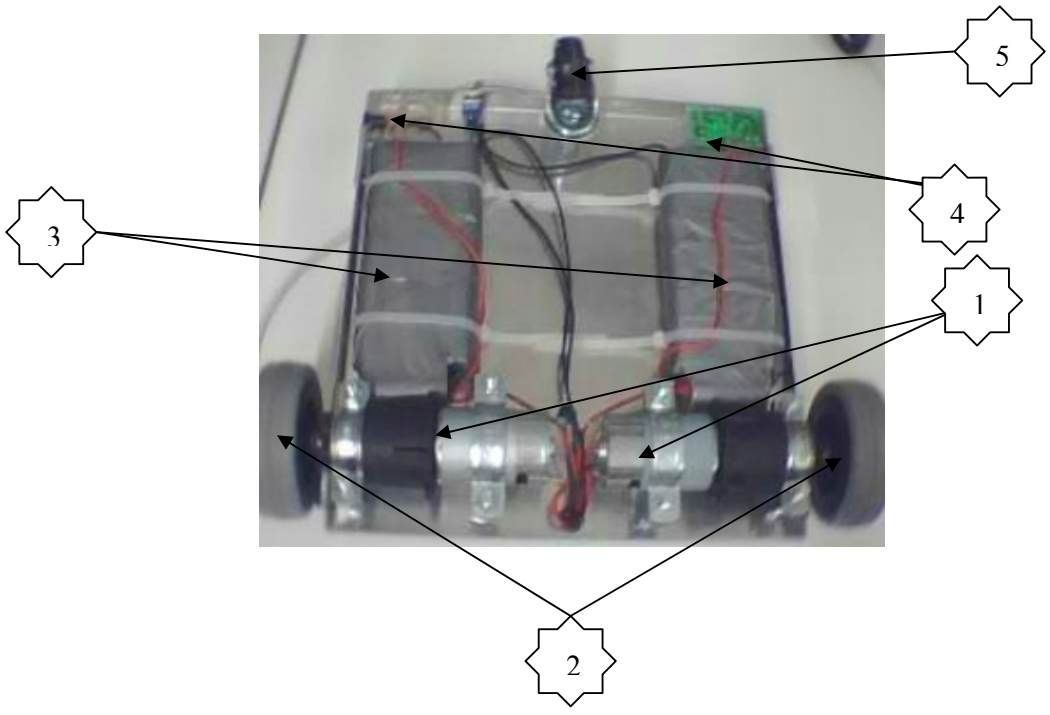


Figura 2 – Representação do Robot Rasteirinho (visto de baixo)

Observando o Rasteirinho visto de baixo, é possível verificar a existência de dois motores (1), idênticos que promovem o movimento a duas rodas (2); duas pilhas (3) que carregam o próprio motor através da alimentação à corrente eléctrica, recorrendo a circuitos eléctricos presentes lateralmente no corpo do robot (4); e uma roda piloto (5) que assegura o equilíbrio do robot quando este curva. Na parte traseira da base metálica, é possível encontrar um interruptor ON/OFF..

Teoricamente, o robot irá andar em frente, alimentando ambos os motores com uma tensão nominal constante. Da mesma forma, as curvas serão realizadas pelo mesmo principio, isto é, adicionando uma tensão "extra" a um motor, fazendo com que o robot rode sobre si e complete a curva. A grande dificuldade reside nas múltiplas variáveis que tornam o sistema não linear e invariante no tempo.

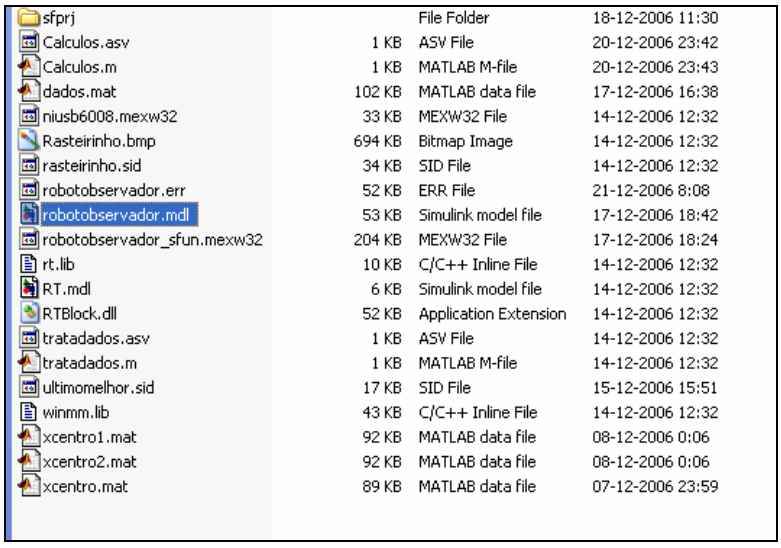
4.Funcionamento do software

4.1.Localização dos ficheiros que compõem o trabalho prático:

Os ficheiros relativos ao Trabalho Prático 2, encontram-se no ficheiro *zip* TP2G07_08.zip e os ficheiros devem ser extraídos para uma pasta no desktop com o mesmo nome (C:\Desktop\TP2G07_08). Ao abrir a pasta, a memória descritiva do trabalho encontra-se no ficheiro *doc* Trabalho Prático 2, e os ficheiros que compõem o programa em Matlab têm a extensão .mat, .mdl e .m..

4.2.Manual de utilizador:

Para iniciar o programa, o utilizador deverá carregar no ficheiro robotobservador.mdl, que se encontra na directoria anteriormente referida, que irá de forma automática abrir o Matlab



sfprj		File Folder	18-12-2006 11:30
Calculos.asv	1 KB	ASV File	20-12-2006 23:42
Calculos.m	1 KB	MATLAB M-file	20-12-2006 23:43
dados.mat	102 KB	MATLAB data file	17-12-2006 16:38
niusb6008.mexw32	33 KB	MEXW32 File	14-12-2006 12:32
Rasteirinho.bmp	694 KB	Bitmap Image	14-12-2006 12:32
rasteirinho.sid	34 KB	SID File	14-12-2006 12:32
robotobservador.err	52 KB	ERR File	21-12-2006 8:08
robotobservador.mdl	53 KB	Simulink model file	17-12-2006 18:42
robotobservador_sfun.mexw32	204 KB	MEXW32 File	17-12-2006 18:24
rt.lib	10 KB	C/C++ Inline File	14-12-2006 12:32
RT.mdl	6 KB	Simulink model file	14-12-2006 12:32
RTBlock.dll	52 KB	Application Extension	14-12-2006 12:32
tratadados.asv	1 KB	ASV File	14-12-2006 12:32
tratadados.m	1 KB	MATLAB M-file	14-12-2006 12:32
ultimomelhor.sid	17 KB	SID File	15-12-2006 15:51
winmm.lib	43 KB	C/C++ Inline File	14-12-2006 12:32
xcentro1.mat	92 KB	MATLAB data file	08-12-2006 0:06
xcentro2.mat	92 KB	MATLAB data file	08-12-2006 0:06
xcentro.mat	89 KB	MATLAB data file	07-12-2006 23:59

Figura 3 – Apresentação dos ficheiros presentes na directoria C:\Desktop\TP2G07_08

Após o Matlab se inicializar deverão ser visíveis as seguintes janelas

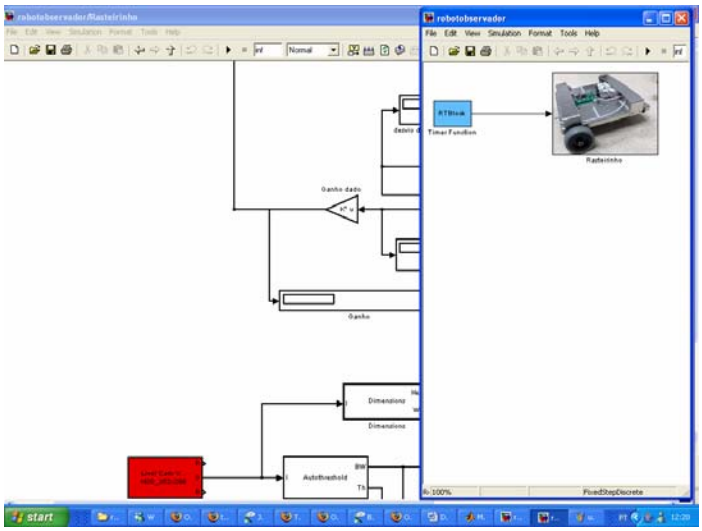


Figura 4 – Execução do Ficheiro robotobservador.mdl

Para fazer correr o programa, o Rasteirinho deverá ser colocado sobre um trajecto definido, com uma webcam apontada para as linhas desse trajecto, de forma a não ser afectado pelos reflexos e pelas sombras no chão. De seguida, o interruptor na parte posterior deverá ser ligado e, por fim, deverá ser carregada a opção play na barra de ferramentas do Matlab como visível na imagem abaixo.

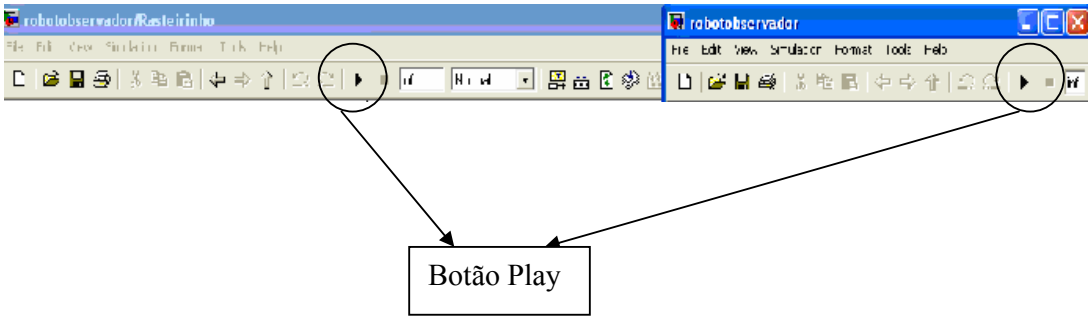


Figura 5 – Interruptor que actua o Rasteirinho e botão que corre o programa.

O Rasteirinho encontra-se agora em condições de começar a funcionar. Como tal, começou-se, pois, por identificar o Sistema que caracteriza o próprio robot.

5. Identificação e Controlo do Sistema

5.1.Passos a dar para a Identificação do Sistema que rege os Robots:

Como objectivo de Controlar o robot Rasteirinho, através de Controlo Óptimo, isto é, fazer com que o robot siga um objecto móvel, deve-se *à priori*, ter conhecimento de como é que este reagirá a uma solicitação externa. Pretende-se assim determinar a Função de Transferência que rege o movimento do Rasteirinho. Numa primeira abordagem, idealizou-se que o Rasteirinho era um sistema do tipo MIMO, multi-input/multi-output, pois existem duas solicitações externas que podem ser actuadas (motor esquerdo e motor direito) e duas saídas, para as quais o sistema responde (ângulo que o rasteirinho faz com o objecto a seguir e a distância a que o robot faz com o centróide do objecto a seguir). À medida que se progrediu no conhecimento adquirido acerca do desempenho do próprio robot, chegou-se à conclusão que o sistema, de facto, era MISO, multi-input/single-output, uma vez que o ângulo que o rasteirinho fazia com o objecto a seguir não passava de uma redundância, pois a distância relativa do rasteirinho ao centróide do objecto a seguir, identificava correctamente a posição do próprio robot.

Pretendia-se, deste modo, que, para a identificação do sistema, o robot seguisse um trajecto pré-definido pelos projectistas (um trajecto desenhado no chão, por exemplo), para que as respostas (saídas) do sistema fossem mensuráveis e facilmente identificáveis. Contudo, não é de todo uma tarefa simples "criar" um sinal de excitação para os dois motores, de forma a que todos os modos do sistema sejam activados, sem nunca perder o objectivo da medição das saídas.

Uma vez que os motores que alimentam as rodas do Rasteirinho são altamente não-lineares e não respondem os dois (motor esquerdo e direito) exactamente da mesma maneira, não era, de certa forma, expectável, que fosse possível a nós, operadores, controlar manualmente o robot, de forma a manter as linhas do caminho a seguir visíveis.

Deste modo, entendeu-se ser necessário aplicar tensões ao robot de forma a que os motores fossem excitados sem que a webcam perdesse de vista o trajecto a realizar. Pensou-se então em recorrer a projectos anteriores, nomeadamente num controlador onde as actuações nos motores foram calculadas baseadas em *try-and-fail*. Este controlador seria uma ferramenta poderosa e uma ajuda preciosa para a identificação do sistema do Rasteirinho, não fossem os ganhos fornecidos ao sistema irrepresentáveis na forma de uma função. O grande problema deste primeiro controlador testado era o facto de não ser possível quantificar os ganhos do sistema sob a forma duma função, sendo no máximo, uma função definida por ramos (se o centróide se encontra à esquerda do centro da imagem, então fornece-se mais tensão ao motor direito, e vice-versa).

Surge então a possibilidade de utilizar um segundo controlador, um sistema em anel fechado, representado pelo seguinte diagrama de blocos:

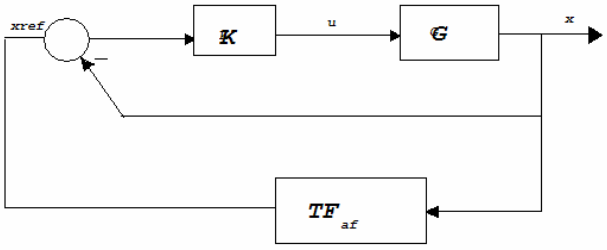


Figura 6 – Controlador em Anel Fechado

E através deste sistema, e com base em conhecimento adquirido noutras cadeiras, como Identificação de Sistemas e Controlo de Sistemas, tem-se que, a Função de Transferência que rege este

diagrama de blocos é tal que: $FT_{af} = K.G / (1 + K.G)$ e xref representa o centro da imagem, e x a saída. Neste controlador, os ganhos K são constantes e calculados de uma forma empírica, sendo um parâmetro a ajustar antes de efectuar qualquer medição, bem como o treshold (contraste entre a imagem – maior ou menor evidencia do preto sobre o branco na imagem a captar pela webcam) e a tensão nominal (deve ser superior a uma zona morta para a qual o robot não responde e inferior a um valor de tensão que permita a saturação do robot – o robot deve ser suficientemente lento de modo a que o maior numero de pontos sejam possíveis "ler" e, em caso de desvio completo do robot relativamente à linha, ele não "choque" com as paredes ou outros objectos do laboratório).

Experimentalmente, o grupo de trabalho verificou que os parâmetros que melhor se ajustavam seriam um K= 0.00017 e uma tensão nominal= 1.3 V. Contudo, estes valores não são dados como os únicos valores possíveis, podendo haver outras combinações, influenciadas por os motores terem ou não a mesma carga, trabalharem ou não à carga máxima, à luminosidade do pavimento, etc.

Após o ajuste correcto dos parâmetros deste controlador de anel fechado, começou-se por fazer o robot percorrer uma linha recta com um certo comprimento, de modo a que a resposta dos motores estabilize e um elevado numero de dados sejam utilizados para identificação e validação. A experiência foi repetida sucessivamente até adquirir resultados quase perfeitos, com poucas perturbações. Verificou-se então, claramente que, a resposta do robot é um sistema de segunda ordem, uma vez que tem uma oscilação inicial – um máximo de sobreimpulso – e posteriormente uma estabilização. Reconstruindo, então, agora os valores recolhidos e, fazendo o tratamento dos dados de identificação e validação do sistema em anel fechado, obteve-se, recorrendo à *toolbox* de identificação do Matlab, *ident*, um modelo armax2221 com 87.64% grau de semelhança com os resultados. Mais uma vez, estes valores podem ser melhorados, consoante a utilização de um melhor ou pior aproximador (BJ, arx, ou outro), e com a maior ou menor semelhança do modelo aos dados reais.

Uma vez representada a Função de Transferência em Anel Fechado, tem-se então o Sistema que define o Rasteirinho, do seguinte modo: $G = Ft_{af} / K.(1-Tf_{af})$; sendo este o modelo identificado para o robot. Este modelo não é único, conforme já se referiu, contudo, para o caso em estudo, foi o melhor conseguido.

5.2. Do Sistema ao Controlo do Rasteirinho:

Identificado o sistema, pretendia-se recorrer ao Regulador Linear Quadrático (LQR) de modo proceder ao controlo óptimo do robot. Um dos aspectos verificados anteriormente é que, este tipo de problemas de Controlo Óptimo apenas se socorre de Sistemas onde a identificação dos seus espaços de estado são conhecidos. Entende-se como uma boa identificação, o conhecimento exacto, isto é, o significado físico dos elementos de estado. Contudo, sabe-se que, um sistema é apenas representado por uma única Função de Transferência, mas pode ter inúmeros espaços de estado que o representam (muitos sem um significado físico atribuído). Desta forma, a reconstrução directa dos estados é quase impossível, e através do Matlab, a aplicação do comando *ss* ("space stade") a uma Função de Transferência, só por mero acaso é que os estados devolvidos pelo Matlab seriam algo conhecido (como uma velocidade, ou uma posição). Torna-se então necessário recorrer à Observação dos espaços de estado, onde a reconstrução dos estados é feita de uma forma correcta.

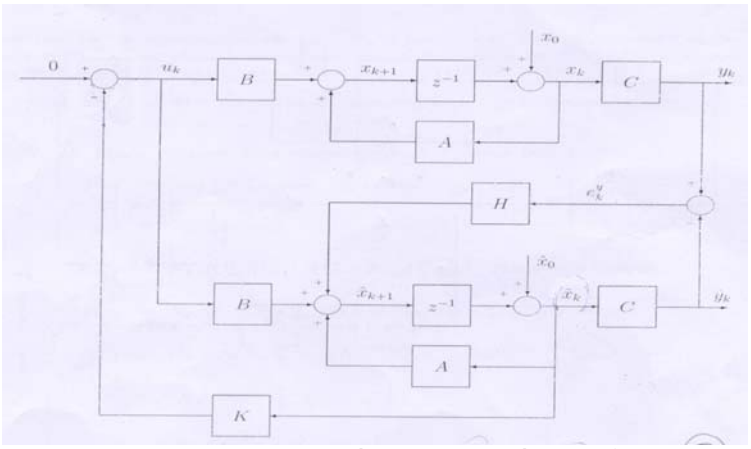


Figura 7 – Diagrama de blocos de um controlador óptimo recorrendo à observação dos estados do sistema

O conhecimento acerca da observação de um estado é matéria que transcende o âmbito da cadeira de Controlo Óptimo, pelo que se tornou necessário ter uns breves conhecimentos acerca deste conceito. Assim, em traços gerais, um Sistema é observável através da reconstrução dos seus estados, recorrendo unicamente às matrizes de estado A, B, C e D (podendo ter significado físico ou não) fornecidas pelo comando *ss* do Matlab.

A partir destas matrizes, torna-se necessário o cálculo da matriz H para que os estados sejam observáveis. Como tal, tem-se que, se o sistema é de segunda ordem (do tipo z^2), a função φ é do tipo $\Phi(A^T) = a'_n\, I + a'_{n-1}\, A^T + ... + a'_1\, (A^T)^{n-1} + (A^T)^n$ Onde os zeros da função podem ser determinados $z^n + a'_1\, z^{n-1} + ... + a'_{n-1}\, z + a'_n = 0$ empiricamente, de modo a ajustar melhor ou pior, os resultados. Para o caso em estudo, utilizou-se a função $\varphi(z) = z^2$ (os zeros na origem). Os valores de H são [0.9253;0.5778]

Finalmente, para controlar o Sistema, isto é, o Rasteirinho, é necessário realimentar a entrada com os ganhos de Kalman, K, calculados através de um programa criado utilizando o Matlab, como referido anteriormente ("calculos.m"). Utilizou-se a formula LQR, sendo que as matrizes Q e R podem ser ajustadas consoante a resposta que se pretende. No caso em estudo, a matriz Q foi sempre a identidade e o valor de R variou consoante a tensão nominal a aplicar no Rasteirinho controlado. A variação de Q faz com que haja um maior ou menor destaque de determinados estados do sistema; a variação de R determina a rapidez de resposta, uma atenuação às tensões bruscas, suavizando ou excitando mais fortemente os motores. Posteriormente, será apresentada uma tabela de valores com os parâmetros alterados e calculados experimentalmente pelo grupo, ou seja, uma tabela de variação de tensão nominal com Q, servindo como ajuste a posteriores experiências.

5.3.Passos Principais para o Controlo do Rasteirinho:

Resumidamente, pôde-se constatar que a aplicação de Controlo Óptimo ao robot Rasteirinho se resume nos seguintes itens:

- Inicialmente, devem-se instalar as *drives* de aquisição de sinal da National Instruments (placa que faz a interface hardware/software das instruções dadas pelo computador com os motores que alimentam o robot), o software de programação Matlab 2006 (versão a ou b) e uma webcam;
- Seguidamente, o principal objectivo passa por fazer com que o robot siga um trajecto pré-definido de modo a proceder à identificação do Sistema que rege o robot. Recorrendo ao Simulink do Matlab, é facilmente construído um sistema em anel fechado, onde o robot é realimentado por ganhos que controlam a trajectória do próprio Rasteirinho. Os valores

desses ganhos são calculados por experimentação, bem como o treshold a definir pela imagem;

- Depois de garantido que o robot é "guiado" ao longo do trajecto definido, deve-se proceder à recolha de dados que identificam o sistema em anel fechado. Quer-se com isto dizer que, as entradas (uma posição de referência para a qual o robot deve atingir – tipicamente o centro da imagem captada pela câmara) e as saídas (posição efectiva do centróide do trajecto), devem ser os dados a recolher e a tratar.
- Os dados recolhidos serão tratados, então, com recurso à toolbox de Identificação de Sistemas do Matlab (Ident) e, desta forma, deve-se procurar obter o melhor modelo para este sistema em anel fechado (arx, armax ou outro – verifica-se que o sistema é oscilatório até estabilizar, o que induz a existência de pólos complexos conjugados; tipicamente um sistema de segunda ordem);
- Obtido um modelo que aproxima razoavelmente bem o sistema em anel fechado (a obtenção de um modelo com um grau de semelhança de 100% é praticamente impossível, quer devido à existência de ruído, mas em especial, devido à forte não-linearidade dos motores do robot), pretende-se agora encontrar o "verdadeiro" sistema que caracteriza o próprio robot, isto é, a função de transferência em anel aberto. Para tal, recorrendo à álgebra dos diagramas de blocos, sabe-se que o sistema que caracteriza o robot é tal que $G(z) = FT / K (1-FT)$;
- Tem-se então em nossas mãos o sistema que caracteriza o robot. Pode-se agora começar a pensar em construir os ganhos de Kalman que vão realimentar os motores, para que o Rasteirinho seja controlado de uma forma óptima. Contudo, como referido anteriormente, a passagem "automática" para espaços de estado não é de todo segura, pois não existe a garantia de que os estados encontrados sejam os que se procuram para o cálculo dos ganhos K. Torna-se então necessário observar os estados do Sistema (através de um Observador de Estado) e realizar o controlo, baseado nesse Observador. É, pois, necessário a criação de um novo bloco em Simulink que faça a leitura correcta dos estados do Sistema e da sua evolução, introduzindo então, as matrizes A, B e C, provenientes da construção dos espaços de estado do Sistema. É igualmente necessário, aquando a reconstrução dos estados, fornecer ao sistema a forma da função $\varphi(z)$ – que representa a ordem do sistema e a localização pretendida para os pólos do sistema -, para o calculo do bloco H.
- Com os principais parâmetros de controlo obtidos, é então deste modo, fácil fazer uma alimentação aos motores, através do K, calculado através do LQR. Uma nota de cálculo, em relação às matrizes Q e R a introduzir no cálculo dos ganhos: as matrizes não têm um valor melhor ou pior, e, à semelhança do que acontece relativamente aos ganhos fornecidos na realimentação, aquando a identificação do Sistema, e o treshold, os valores de Q e R devem ser obtidos novamente por "try-and-fail", não havendo um valor ideal para cada robot.

5.4.Dificuldades na elaboração do Controlo Óptimo do Rasteirinho:

Uma vez sugerido o projecto para o Controlo Óptimo do robot, o primeiro grande problema que surge remota para a forma em como o problema é, de facto, abordado. Inicialmente, pensou-se que o sistema que caracteriza o robot seria MIMO (multi-input/multi-output) onde as entradas seriam as tensões aplicadas a cada um dos motores e as saídas, o desvio do próprio robot relativamente ao centróide da imagem captada e a inclinação do robot, relativamente ao trajecto a efectuar. Numa segunda abordagem, como já foi referido anteriormente, verificou-se que o calculo da inclinação do robot relativamente à linha que compõe o trajecto não passa de uma redundância, aquando do calculo do

desvio do robot relativamente ao centróide da imagem. Posteriormente, concluiu-se que, efectivamente, a entrada do Sistema corresponde ao centróide da imagem, sendo a sua saída a posição do robot relativamente a esse mesmo centróide. O problema é, pois, simplificado a um Sistema SISO (single-input/single-output).

Eis, então, que surgem dificuldades na forma em como o Sistema deve ser identificado. Partiu-se dum controlador disponibilizado por colegas que, anteriormente, já tinham controlado o robot, de modo a que fizesse os trajectos presentes no laboratório. Contudo, o principal problema aparece na forma em como é calculado o ganho a "injectar" nos motores, isto é, os ganhos eram calculados por ramos, numa estrutura não-representável sob a forma de uma função; quando o robot se encontrava a uma certa distância do centro da imagem, seria fornecido o ganho X, caso contrário, seria Y. Através da aplicação dum controlador em anel fechado, esse problema era suprimido e a principal dificuldade passaria, então, pelo cálculo do ganho K, constante, a fornecer aos motores e o treshold que definia a qualidade da imagem a capturar pela webcam.

Posta esta situação, o cálculo do Sistema que caracteriza o Rasteirinho torna-se então, simples, através da identificação dos parâmetros de entrada e saída. À partida, as principais dificuldades já haviam sido superadas, não obstante o facto de os dados de identificação/validação serem os mais desejados, devido à existência de ruído, à não-linearidade dos motores e a outros problemas característicos do próprio sistema. Por este facto, também a Função Transferência que caracteriza o Rasteirinho não tem uma semelhança 100% relativamente aos dados introduzidos. Contudo, uma boa aproximação foi conseguida, recorrendo a dados de identificação e validação.

Através da Função Transferência do Controlador em anel fechado, foi, então, possível obter a Função Transferência do sistema que caracteriza o Rasteirinho. No entanto, nova adversidade surge aquando o cálculo dos espaços de estado do robot. Recorre-se então a um modelo de observação dos espaços de estado do sistema, socorrendo-nos de matéria que não foi leccionada na Cadeira de Controlo Óptimo, o que induziu uma certa dificuldade na aprendizagem do modelo adoptado.

Passada esta fase, o último problema que desponta, passa pelos valores das matrizes Q e R a adoptar no cálculo dos ganhos de Kalman (através do método LQR) a utilizar na realimentação do sistema. Os ganhos de Kalman também poderiam ser calculados pelo método de Ackerman.

6.Descrição dos Principais Blocos: Por tudo o que foi anteriormente descrito, recorreu-se à linguagem de programação Matlab, em especial às toolboxs de Identificação de Sistemas e a uma interface que opera em grafos, Simulink, para concretizar os objectivos referidos. Seguidamente será apresentado o programa desenvolvido, os seus grafos e os aspectos mais importantes.

6.1. Descrição do Controlador

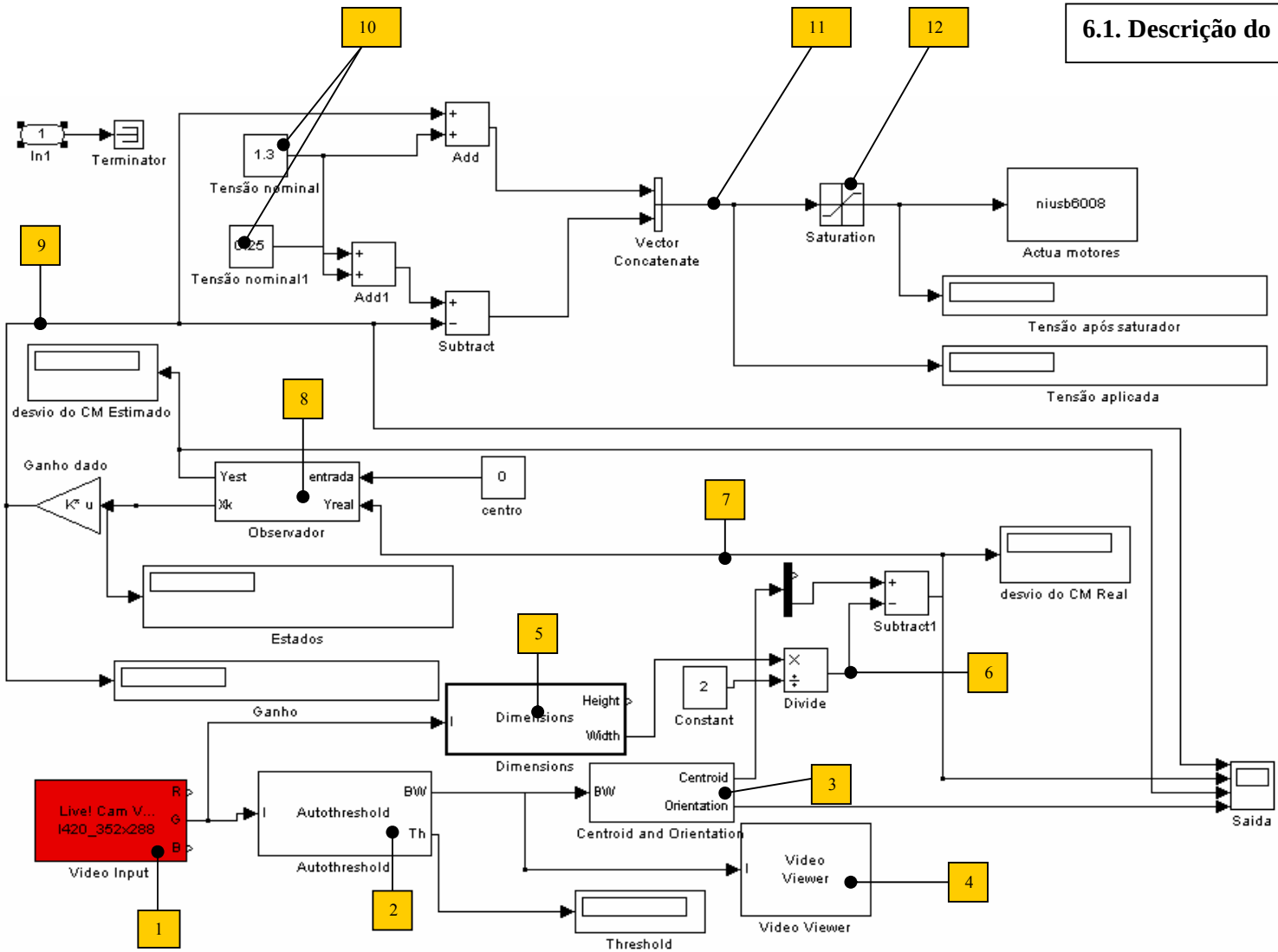


Figura 8 – Apresentação do corpo do programa

1 – Entrada Vídeo
2 – Bloco autothreshhold
3 – Subsistema “calcula centróide”
4 – display vídeo
5 – Bloco para calcular comprimento e largura da imagem
6 – X de referencia
7 – Desvio do centro de massa
8 – Subsistema do observador
9 – Valor de tensão a somar e subtrair aos motores
10 – Tensão nominal para cada motor
11 – Tensões de cada motor
12 – Saturador de tensões

O modelo *Simulink* do controlador criado, é composto por três secções ou subsistemas, designados, nomeadamente por, subsistema do controlador (figura 7), subsistema do observador de estados (figura 8) e por fim o subsistema de cálculo do centróide (figura 9).

Como se pode verificar pela figura abaixo, o sistema tem entrada num bloco de aquisição de imagem (1) que irá ligar à webcam em uso para obter imagens a partir dela. Por defeito, para obter um controlo adequado do robot, definiu-se como parâmetros da webcam, a resolução de 280 x 320 e uma taxa de amostragem de 20 imagens por segundo. Com estes parâmetros sabe-se que, vinte vezes por segundo, o controlador irá corrigir o trajecto que está a ser percorrido pelo robot.

Do mesmo modo, a saída do controlo de aquisição de imagem está ligado a um bloco autothreshold (2). Este bloco tem como função definir o que deve ser visto como preto e o que deve ser interpretado pelo controlador como branco na imagem. As opções escolhidas no autothreshold foram a opção “specify data range” em que o valor máximo definido foi de setenta e o mínimo de zero. A outra opção definida foi a de “scale threshold” onde se inseriu o valor de 0.7.

O sinal que irá ser obtido à saída do autothreshold é aquela que irá ser processado pelo controlador. Paralelamente ao processamento feito pelo autothreshold, existe um bloco que calcula o número de pixeis obtidos pela imagem (ou seja, o número de pontos da imagem) (5), dividindo o número de pixeis da largura do ecrã por dois, é possível obter a localização exacta do centro da imagem, que é a entrada do sistema (6).

A saída do bloco autothreshold irá ligar-se a um visualizador de vídeo (4) e ao subsistema de cálculo do centróide (3).

Após o cálculo da posição do centróide realizada pelo subsistema criado para esse fim (3), será calculada a diferença entre o centro da imagem e o centróide da região a preto, calculado pelo subsistema “cálculo do centróide”. Ao valor obtido desta diferença, é denominado desvio ou erro do sistema (7). O valor do desvio e uma constante de valor zero serão as entradas do subsistema do observador (8) e o valor de saída é, então, o vector u que quando multiplicado pela matriz K, produz um valor de tensão ideal (9) que será somado e subtraído à tensão nominal de cada motor, respectivamente (10) de modo a controlar o Rasteirinho. Para não ultrapassar o valor máximo de tensão aceite pelo rasteirinho, torna-se necessário utilizar um saturador (12) que pode ser encontrado na implementação, imediatamente antes da saída para a placa de controlo.

6.2. Subsistema de cálculo do centróide

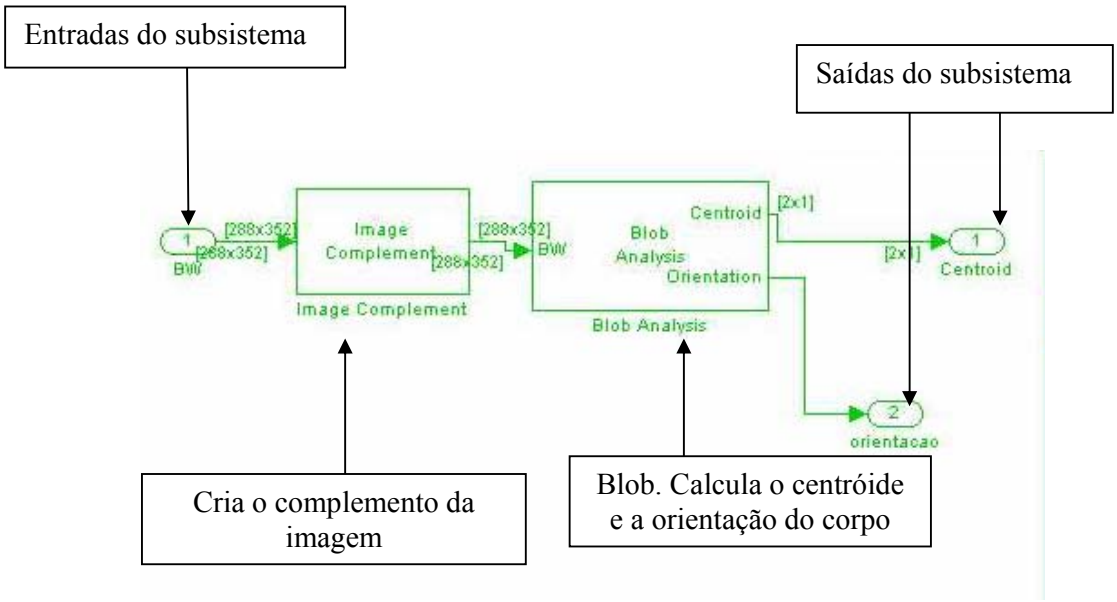


Figura 9 – Subsistema do cálculo do Centróide

O subsistema de cálculo do centróide tem como entrada a imagem a preto e branco. Posteriormente, essa imagem é processada num bloco que devolve o complemento da imagem. Seguidamente, recorreu-se a um bloco denominado *blob,* que calcula o centróide da região branca. O *blob* tem como principal característica o cálculo exclusivo do centróide do primeiro corpo preto que encontrar, sendo este, o corpo negro mais à esquerda na imagem (mais próximo da origem). O *blob* é, portanto, um modo pouco eficiente de calcular o centróide da imagem pois necessita de elevadas quantidades de recursos computacionais e, em certo ponto, torna-se ineficaz no cálculo do centróide da imagem. As saídas do *blob* representam, também, as saídas do subsistema "calcula centróide" e, correspondem à orientação do corpo negro e ao seu centróide.

6.3. Subsistema do Observador

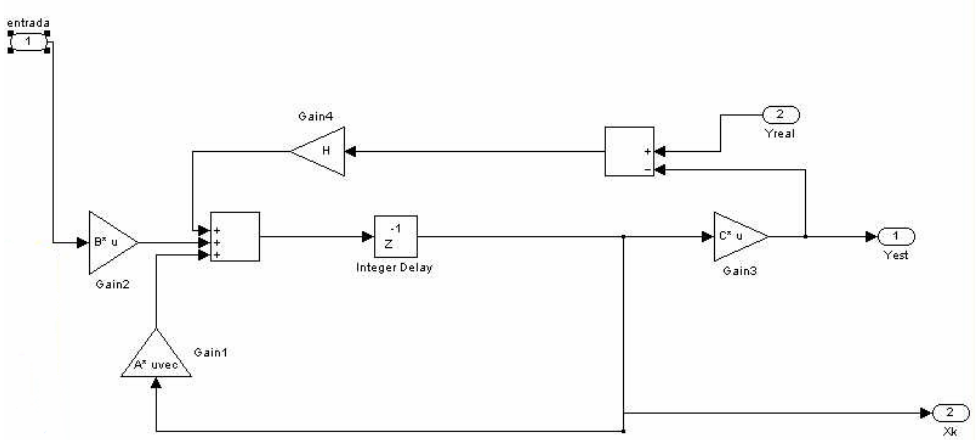


Figura 10 – Subsistema do Observador

As entradas do subsistema são transformadas no observador de estados e convertidas num conjunto de valores: estados do sistema e desvio estimado. O primeiro conjunto vai ser, posteriormente, multiplicado pela matriz K e o segundo é utilizado apenas para observação e comparação com o desvio real.

7.Estudos Realizados e Observações

Usando o programa inicial (controlador proporcional) feito em blocos de *Simulink* e em conjunto com funções de *MatLab,* colocou-se o robot a seguir a linha negra, cujo método já foi referido anteriormente, de modo a retirar conjuntos de dados de resposta do sistema no tempo.

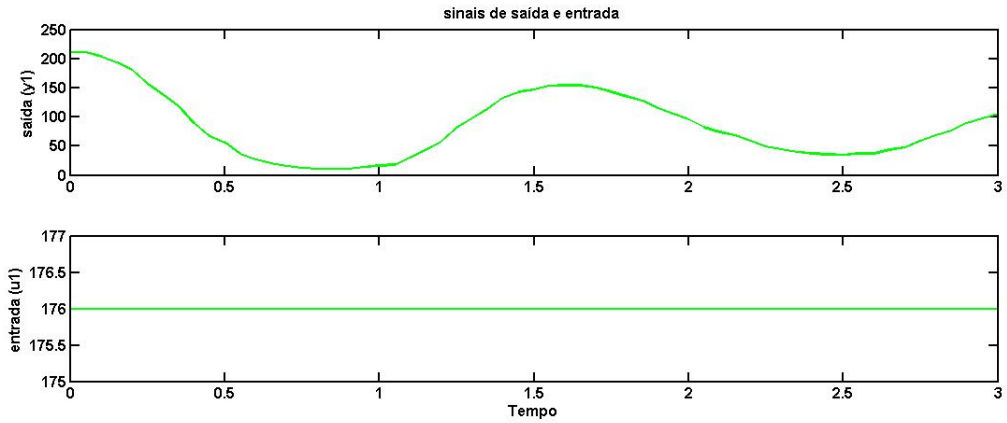


Figura 11 -Conjunto de dados seccionados, de entrada e de saída do sistema, em função do tempo

A partir destes dados e através do *Ident*, foi possível criar um conjunto de estruturas *ARMAX* das quais, a seguinte, é a que melhor simula os dados (melhor best fit).

$$FT(arma\,x2221) = \frac{0{,}008736z + 0{,}008736}{z^2 - 1{,}922 + 0{,}9584}$$ (função de transferência do anel fechado)

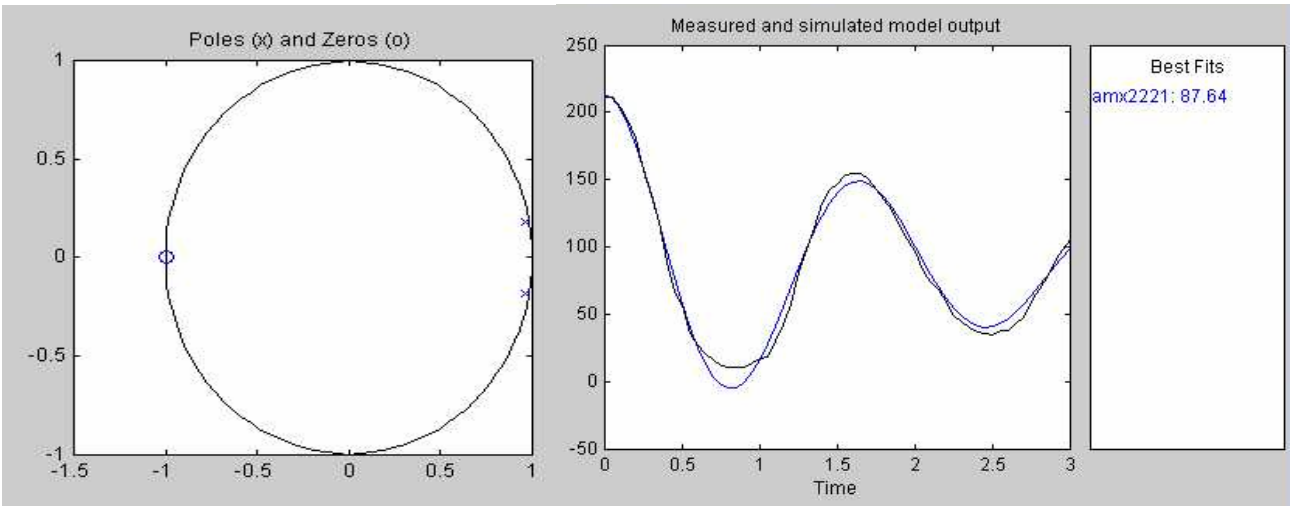


Figura 12 – Modelo armax2221 que simula os dados de saída

De modo a testar o modelo, foram usados dados de estimação diferentes dos de validação, dos quais se obteve o seguinte resultado:

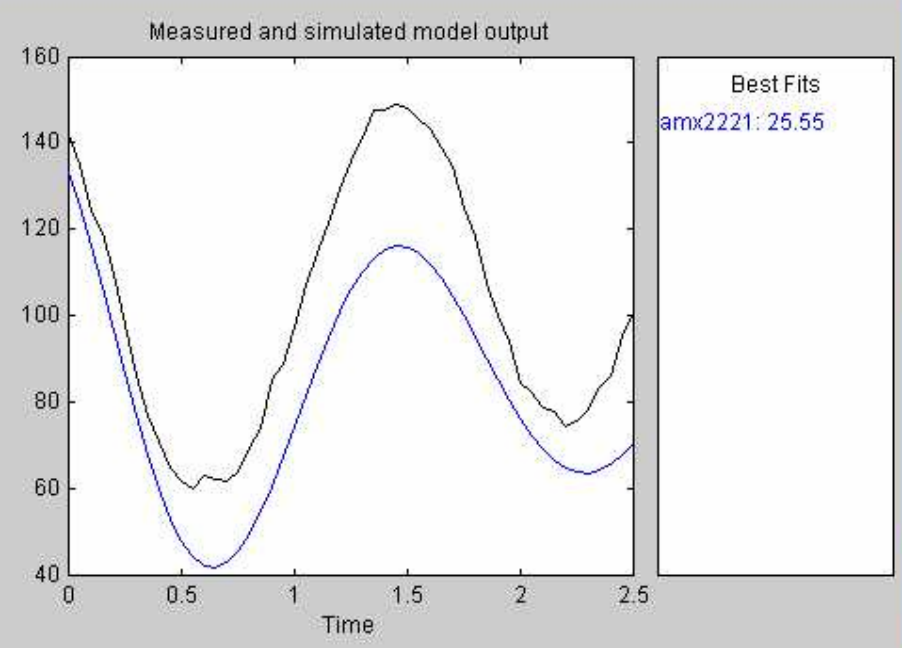


Figura 13 – Modelo armax2221 que simula os dados de saída

Observa-se, então, que, as duas linhas se encontram afastadas na vertical e por essa razão o best fit é inferior. Apesar dessa diferença, as linhas têm aproximadamente a mesma forma o que mostra que o modelo escolhido representa um bom estimador do sistema.

A Função de Transferência em anel fechado do modelo foi utilizada para calcular os espaços de estados da função de transferência em anel aberto do Sistema que caracteriza o robot. Desta forma, calculando a função de transferência em anel aberto obteve-se:

$$G = FT/K.(1-FT) \Leftrightarrow G = (5.139z + 5.139)/(z^2-1.931z+0.9496)$$

Procurando *ss* (espaços de estado) da função de transferência em anel aberto (que devolve os espaços de estado de uma função de transferência), obtiveram-se os seguintes valores matriciais

$$A = \begin{bmatrix} 1.9310 & -0.9496 \\ 1.0000 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 4 \\ 0 \end{bmatrix} \qquad \text{e} \qquad C = \begin{bmatrix} 1.2846 & 1.2846 \end{bmatrix}$$

que foram inseridas no observador de estados, assim como no cálculo da matriz H e da matriz K. Para o cálculo da última matriz foi usado o método DLQR (função da toolbox do MatLab de LQR de dados discretos).Obteve-se, então,

$$H = \begin{bmatrix} 0.9253 \\ 0.5778 \end{bmatrix} \qquad \text{e} \qquad K = \begin{bmatrix} 0.0377 & -0.0337 \end{bmatrix}$$

Com este novo controlador, para uma tensão nominal de 1,3V[1] e, após algumas tentativas, encontrou-se o valor de R que tornava as reacções do robot controladas. Utilizou-se, inicialmente, uma matriz $Q=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$.

Nota: Depois de algumas experiências em linha recta observou-se que um dos motores reagia significativamente menos que o outro, o que provocava o erro estacionário (o robot segue em linha recta mas não "caminha" exactamente sobre esta, encontrando-se ligeiramente desviado para um dos lados). De forma a minimizar este problema, adicionou-se um ganho extra (necessário variar quando se aumenta ou diminui a tensão nominal) ao motor menos sensível, de modo a que, o Rasteirinho ande em linha recta e em cima da linha.

Obtiveram-se, então, os seguintes resultados:

Em linha recta: 1,3V R=3E4 Q=[1 0; 0 1]

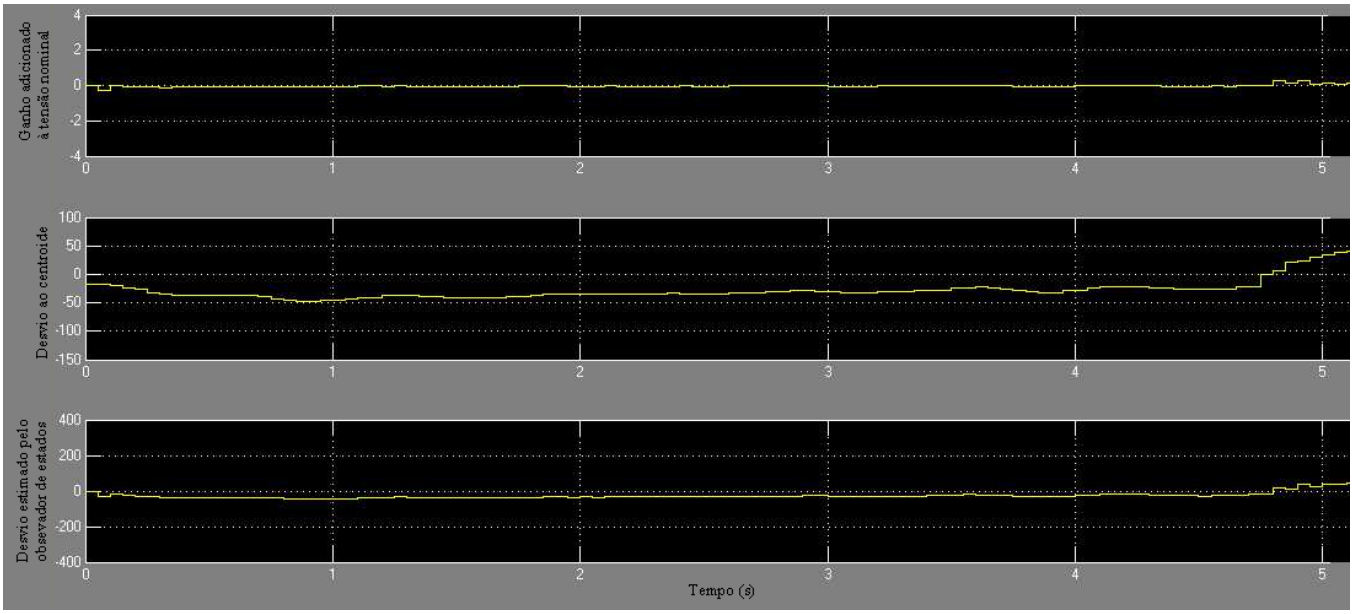


Gráfico 1- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha recta, a 1,3V com $R = 3\times10^4$ e $Q = \begin{bmatrix}1 & 0;\ 0 & 1\end{bmatrix}$

Pode-se verificar que, o ganho adicionado às rodas não é muito elevado e o desvio em relação ao centro de massa é relativamente pequeno. Apesar de existir, neste caso, uma diferença de escala, é de notar que, os desvios estimados estão próximos dos desvios do centróide.

Foram feitos, também, alguns testes com o robot afastado uma certa distância da linha, de modo a verificar a reacção do mesmo. Os resultados desta experiência encontram-se ilustrados nos gráficos seguintes:

[1] Esta tensão não é a tensão aplicada às rodas. mas sim uma tensão de saída do computador, em que para o máximo de 5V é aplicada a tensão máxima das baterias às rodas

Afastado à esquerda da linha recta: 1,3V R=3E4 Q=[1 0; 0 1]

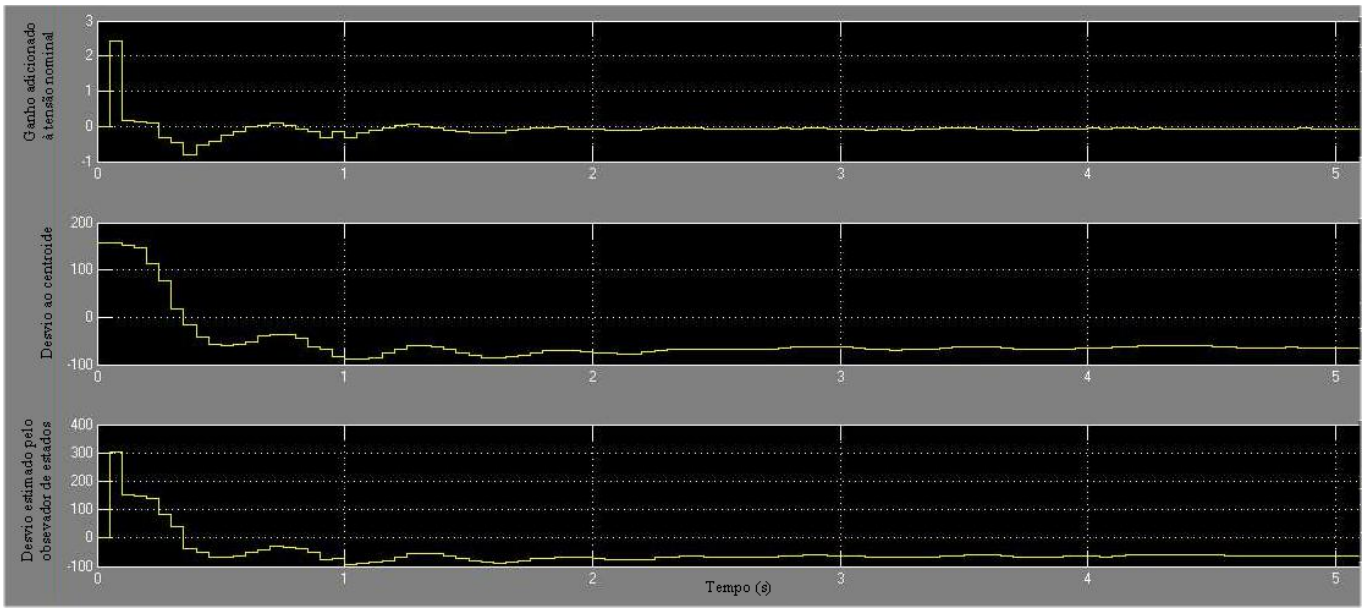


Gráfico 2- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, afastado à esquerda da linha recta, a 1,3V com $R = 3\times10^4$ e $Q = \begin{bmatrix}1 & 0;\ 0 & 1\end{bmatrix}$

Afastado à direita da linha recta: 1,3V R=3E4 Q=[1 0; 0 1]

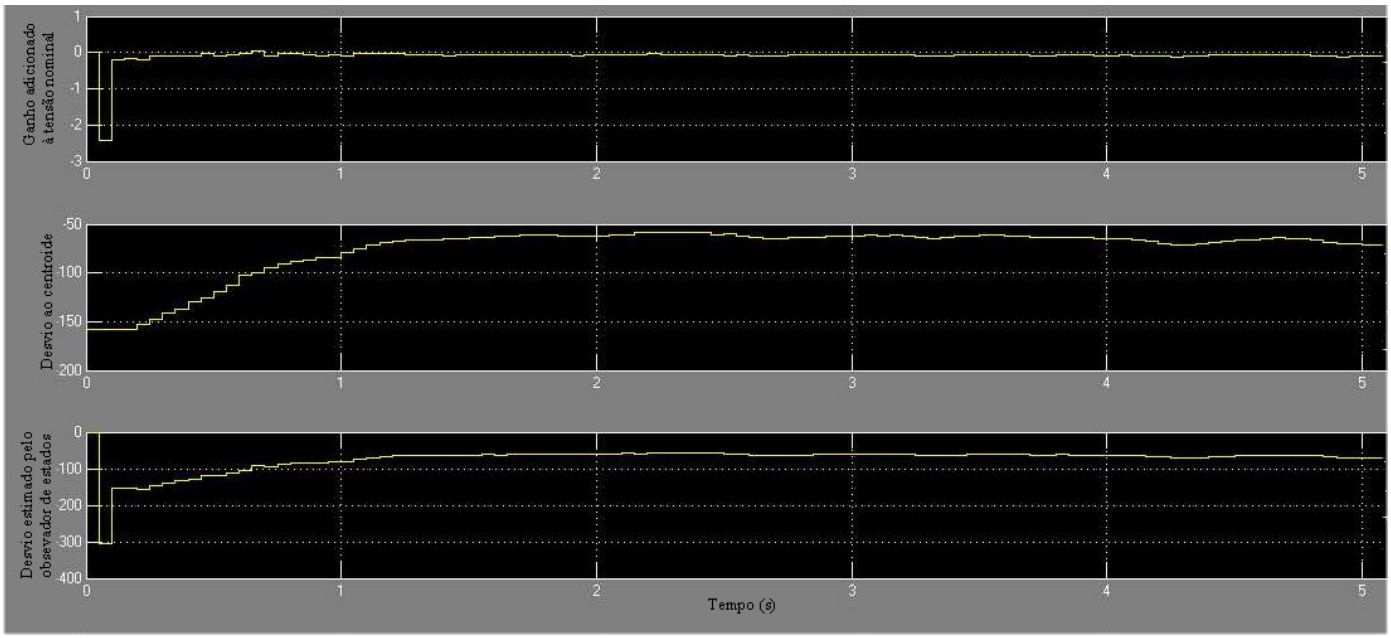


Gráfico 3- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, afastado à direita da linha recta, a 1,3V com $R = 3\times10^4$ e $Q = \begin{bmatrix}1 & 0;\ 0 & 1\end{bmatrix}$

Observou-se, como era esperado, que o ganho é muito elevado nos instantes iniciais devido ao elevado afastamento do trajecto a percorrer. Pode-se reparar, também, que depois desse estímulo inicial, o desvio em relação ao centróide tende a estabilizar e a aproximar-se de zero. Este fenómeno ocorre, pois o robot tende em alinhar-se

perfeitamente com o centróide do trajecto, sendo necessária uma "sobretensão" inicial; tendendo as respostas posteriores, para valores de tensão constantes mais baixos.

Após estas verificações experimentou-se colocar o robot a andar numa linha curva de modo a perceber quais as diferenças de comportamento em relação à linha recta.

Em linha curva: 1,3V R=3E4 Q=[1 0; 0 1]

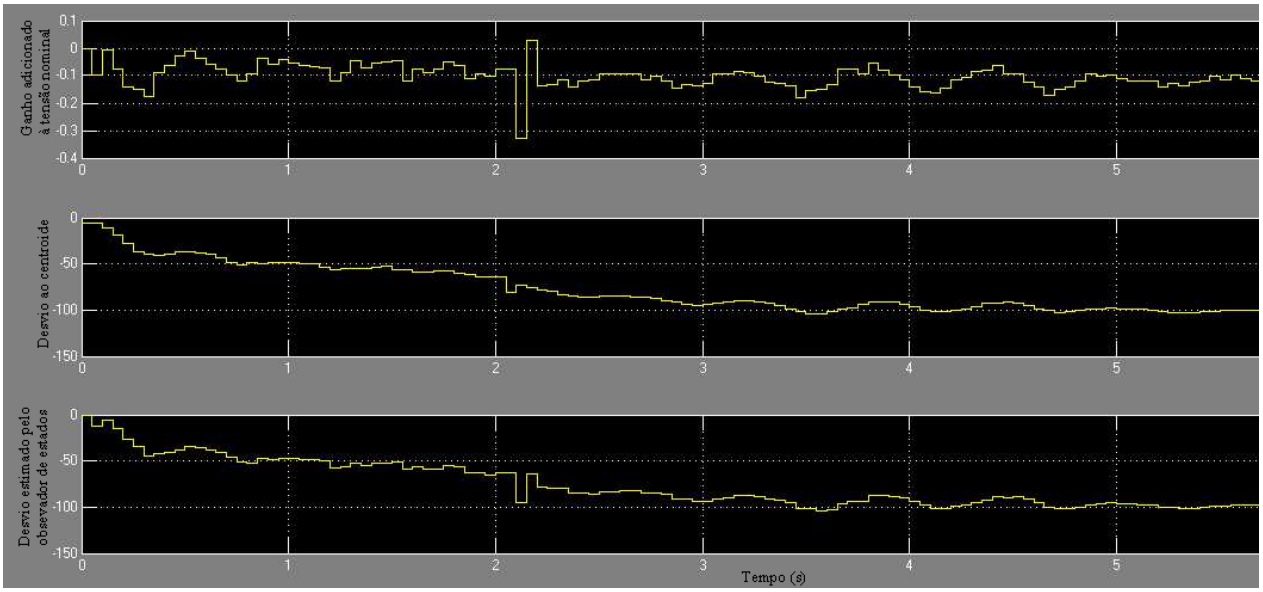


Gráfico 4- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,3V com $R = 3\times10^4$ e $Q = \begin{bmatrix}1\ 0;\ 0\ 1\end{bmatrix}$

Pode-se reparar que os ganhos são muito mais oscilatórios do que em linha recta, o que era de prever; visto que o robot tende a seguir em frente (e por isso a perder linha) caso estes picos de tensão não fossem fornecidos. Assim sendo, o ganho vai aumentando e diminuído conforme o robot se aproxima ou se afasta da linha.

Existe um pico a aproximadamente 2s, devido a reflexos/sombras captados pela câmara, o que provocou uma variação muito brusca do centróide da linha preta, e consequentemente, um pico de ganho.

Verificou-se, também, que, os ganhos permitem ao robot um razoável seguimento da linha; contudo, não é garantido que o robot passe exactamente sobre o centróide do trajecto. De modo a diminuir este afastamento, diminuiu-se o valor do escalar R. Foi necessário, então, ter em atenção que, diminuir em demasia esse valor torna o sistema muito "reactivo" e oscilatório. Foi realizada então a seguinte experiência:

Em linha curva: 1,3V R=1E4 Q=[1 0; 0 1]

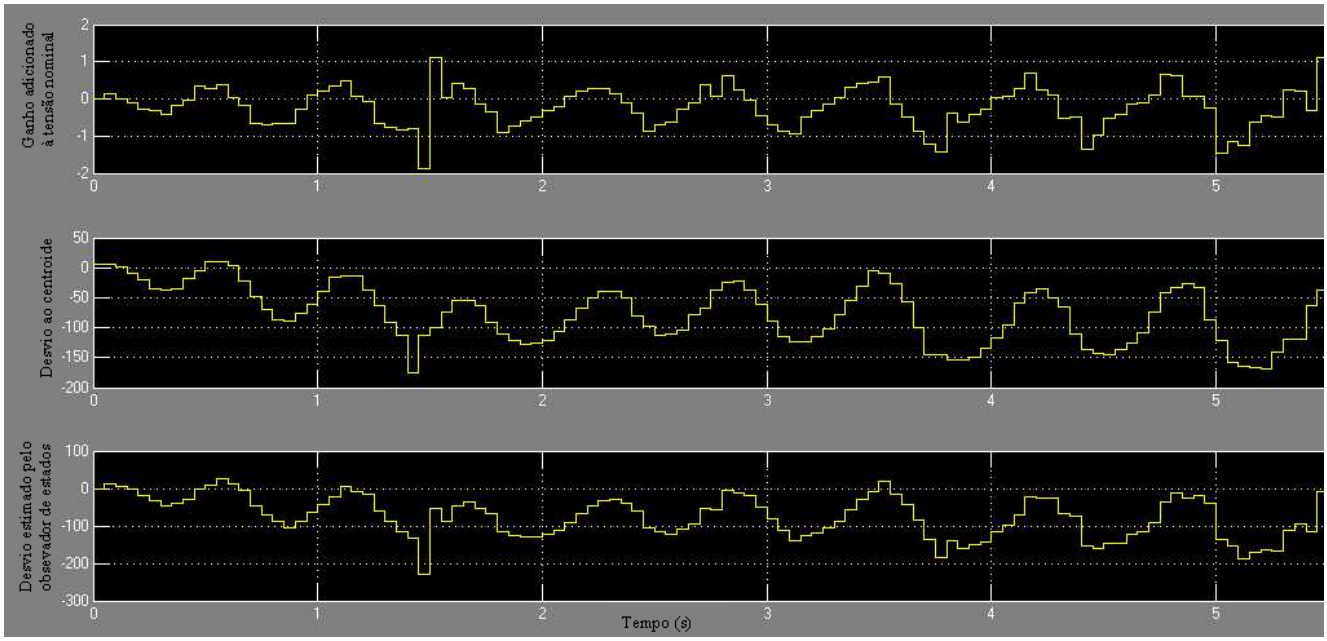


Gráfico 5- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,3V com $R = 1\times10^4$ e $Q = \begin{bmatrix}1 & 0;\ 0 & 1\end{bmatrix}$

Verifica-se, então, o que se esperava, ou seja, o comportamento do robot torna-se demasiadamente oscilatório para o $R = 1\times10^4$.

Após esta experiência, foi realizada uma segunda, onde se aumentaria o R em vez de diminuir. Os resultados encontram-se no seguinte gráfico

Em linha curva: 1,3V R=5E4 Q=[1 0; 0 1]

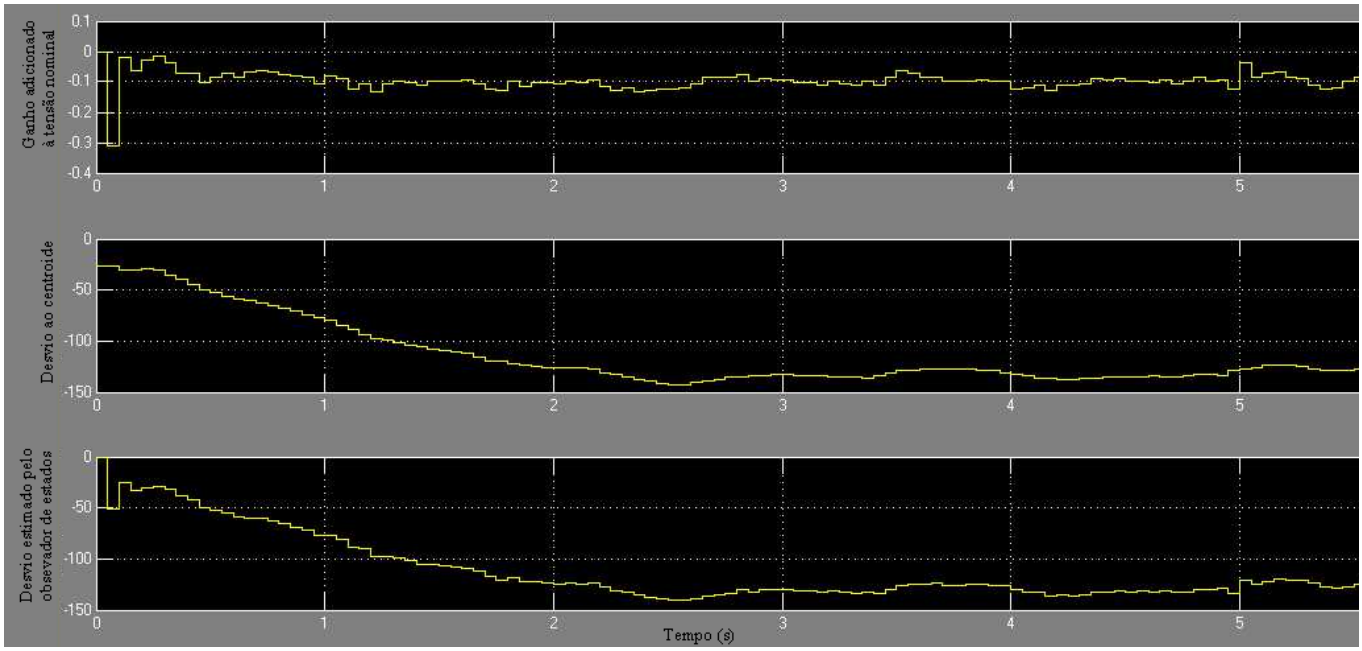


Gráfico 6- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,3V com $R = 5\times10^4$ e $Q = \begin{bmatrix}1 & 0;\ 0 & 1\end{bmatrix}$

Comparando o gráfico 4 com o gráfico 6 verifica-se, então, o expectável, isto é, os motores têm uma reacção muito mais lenta (o ganho é menor), o que leva a que o desvio em relação à curva aumente.

Findos estes testes, foi executada uma nova avaliação de resultados que tinha como objectivo verificar a reacção do robot a perturbações externas, à medida que percorre uma trajectória curva.

Perturbado em linha curva: 1,3V R=3E4 Q=[1 0; 0 1]

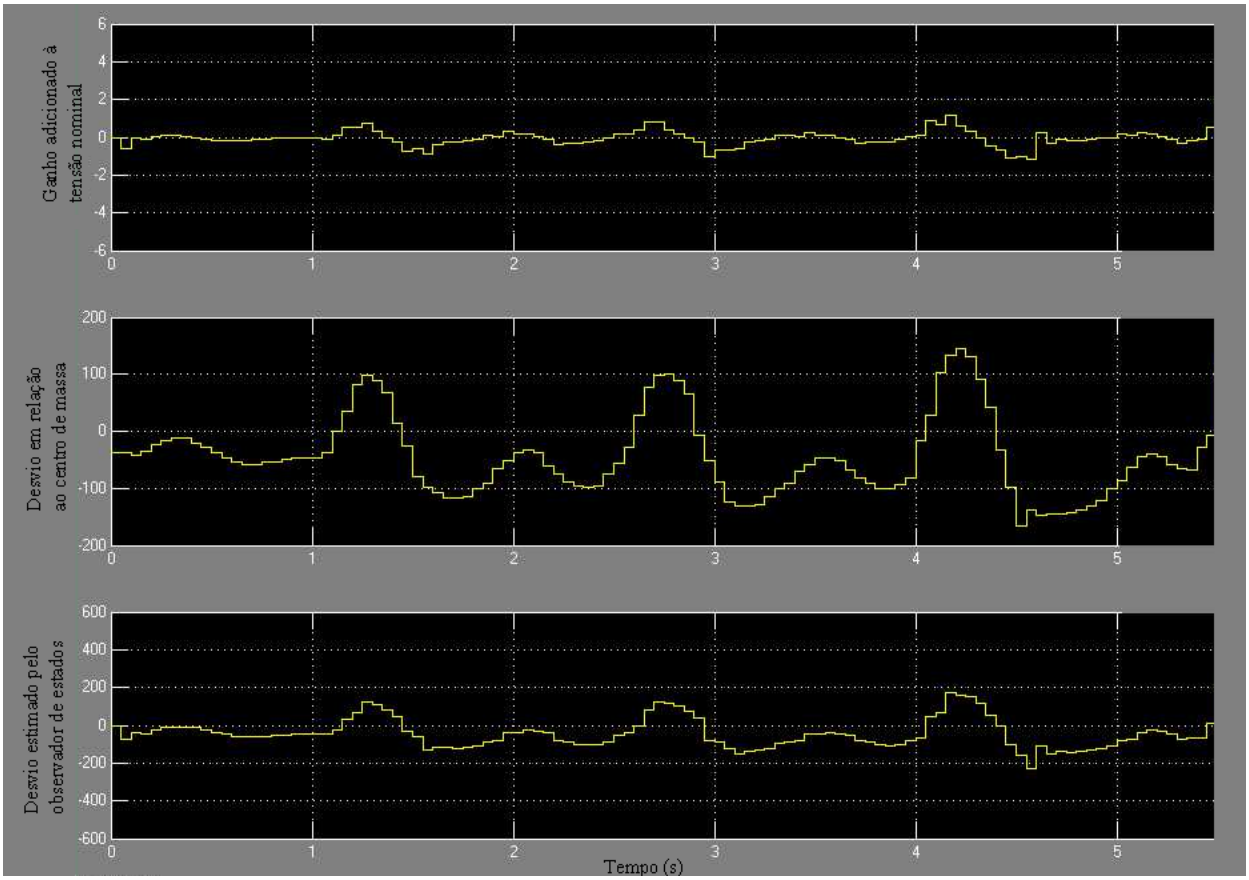


Gráfico 7- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,3V com $R = 3\times10^4$ e $Q = \begin{bmatrix}1\ 0;\ 0\ 1\end{bmatrix}$

Apesar do sistema ser perturbado e de alguma oscilação até atingir novamente um ponto de equilíbrio, verificou-se que, raramente, o robot perdeu a linha, estabilizando pouco depois da perturbação.

Como curiosidade, correu-se o controlador proporcional que inicialmente foi usado para fazer a identificação do Sistema e, compararam-se as qualidades do seguimento da linha entre os dois controladores, como se poderá confirmar pelos gráficos seguintes:

Em linha curva: 1,3V K=0.00017

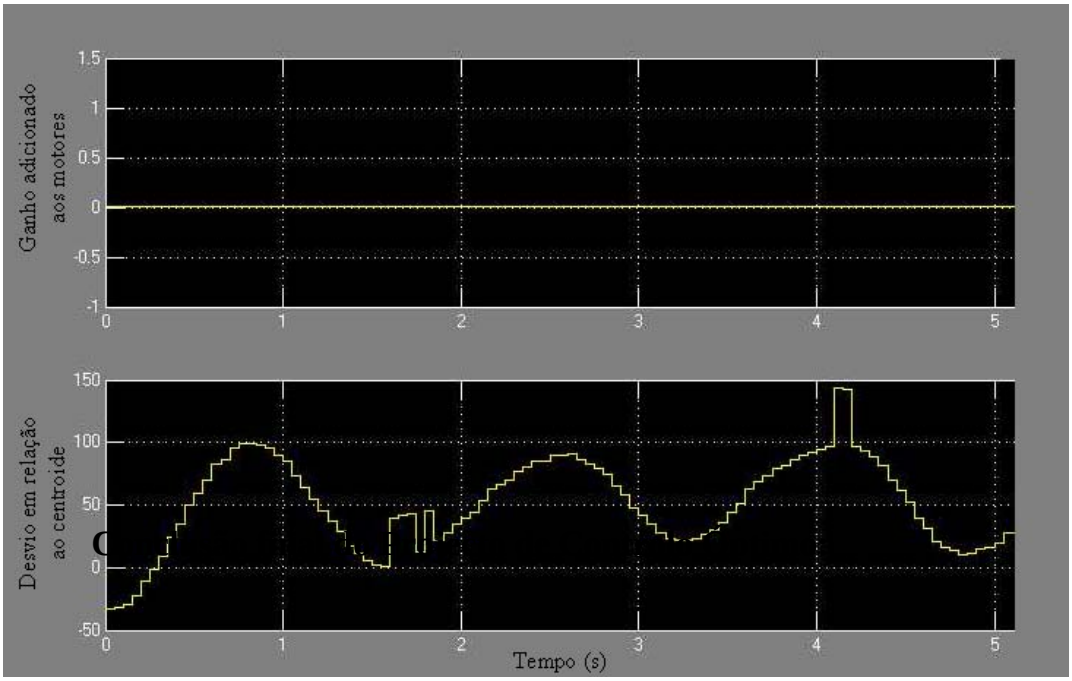


Gráfico 8- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide, aplicado ao controlador proporcional, em função do tempo, em linha curva, a 1,3V com K=0.00017

Perturbado em linha curva: 1,3V K=0.00017

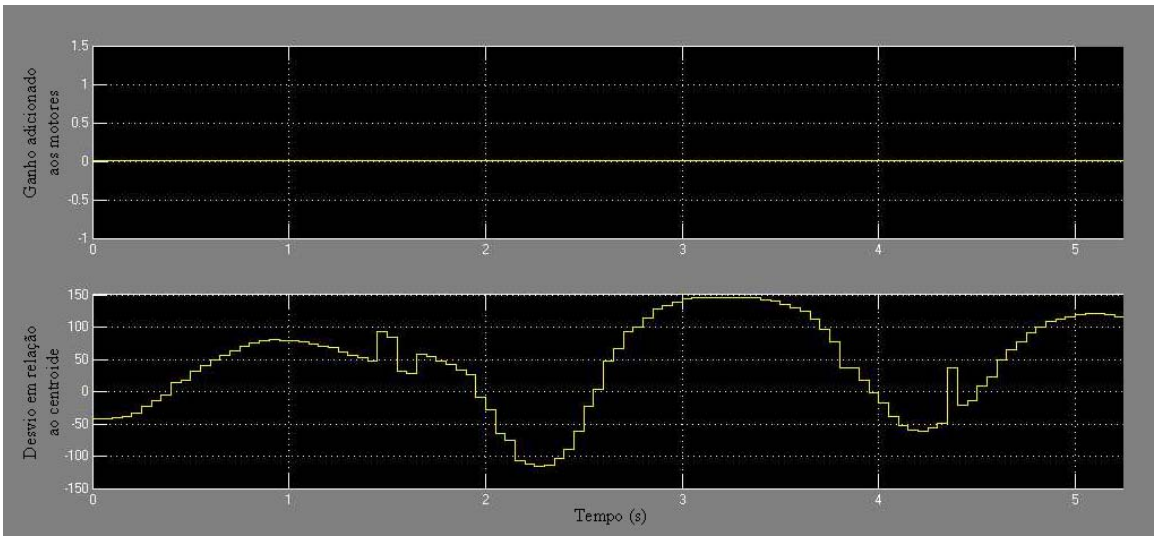


Gráfico 9- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide aplicado ao controlador proporcional, em função do tempo, em linha curva, a 1,3V com K=0.00017

Usando novamente o "controlo óptimo" aumentou-se a tensão e, usando o valor de R (anterior) em que se obteve melhores resultados, colocou-se o robot a andar na linha curva.

Em linha curva: 1,8V R=3E4 Q=[1 0; 0 1]

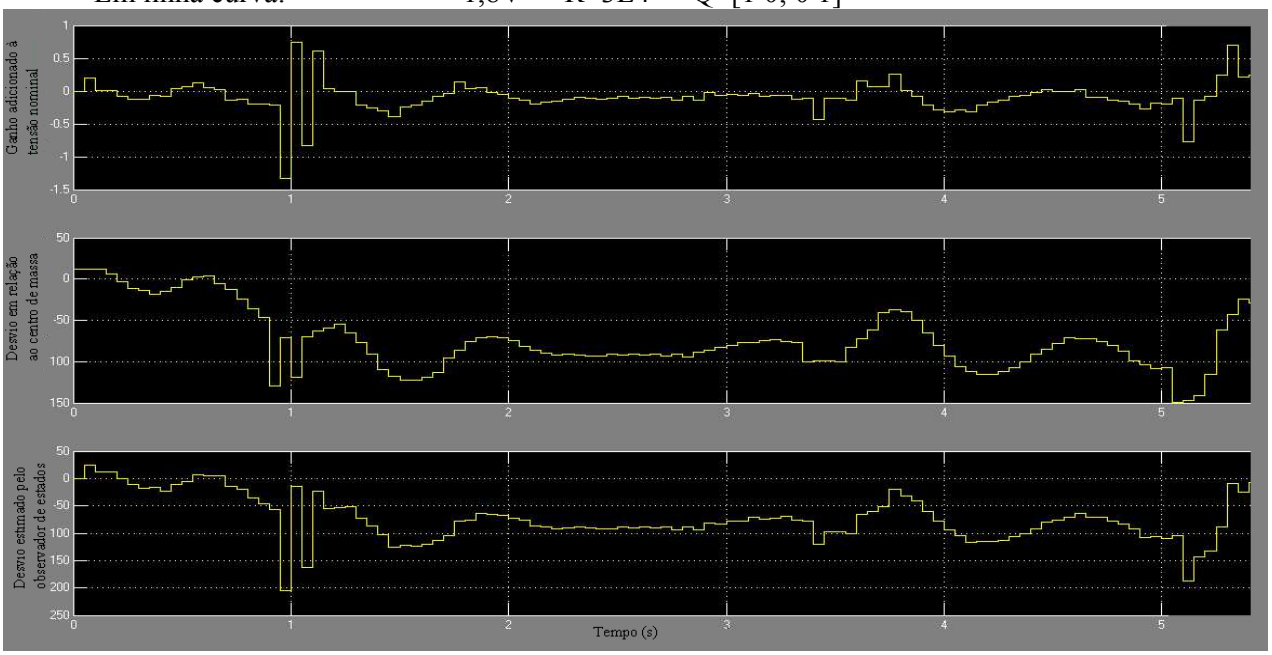


Gráfico 10 - Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 3\times10^4$ e $Q = \begin{bmatrix}1 & 0;\ 0 & 1\end{bmatrix}$

Observa-se que, no início, existem alguns picos, devido a sombras. Verifica-se também que, o Rasteirinho é razoavelmente "suave" a fazer a curva. Mostra-se no gráfico seguinte que, diminuindo o R, as oscilações são mais acentuadas.

Perturbado em linha curva: 1,8V R=2E4 Q=[1 0; 0 1]

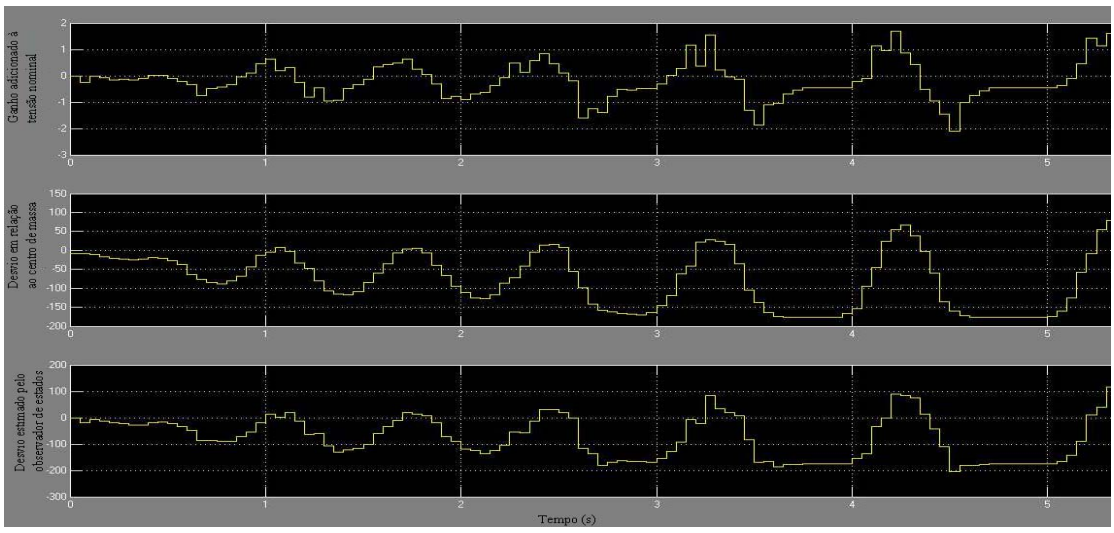


Gráfico 11- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 1\times10^4$ e $Q = \begin{bmatrix}1 & 0;\ 0 & 1\end{bmatrix}$

O passo seguinte consiste na alteração da matriz Q, de modo a tentar tornar o controlo o mais estável possível. Antes dessa alteração, o robot foi posto a seguir a linha e foram efectuados os gráficos dos estados do sistema em função do tempo. Verificou-se que, em módulo, o primeiro estado tomava valores um pouco mais elevados. Assim sendo, foram elaborados dois conjuntos de experiências: uma primeira, em que se minimizou o primeiro estado (diminui-se o valor da primeira coluna da matriz Q); e uma segunda, em que se aumentou o valor da segunda coluna da matriz em questão. Foi necessário variar o valor de R para se obterem melhores resultados.

Obtiveram-se, então, os seguintes resultados:

Em linha curva: 1,8V R=2,5E4 Q=[0.8 0; 0 1]

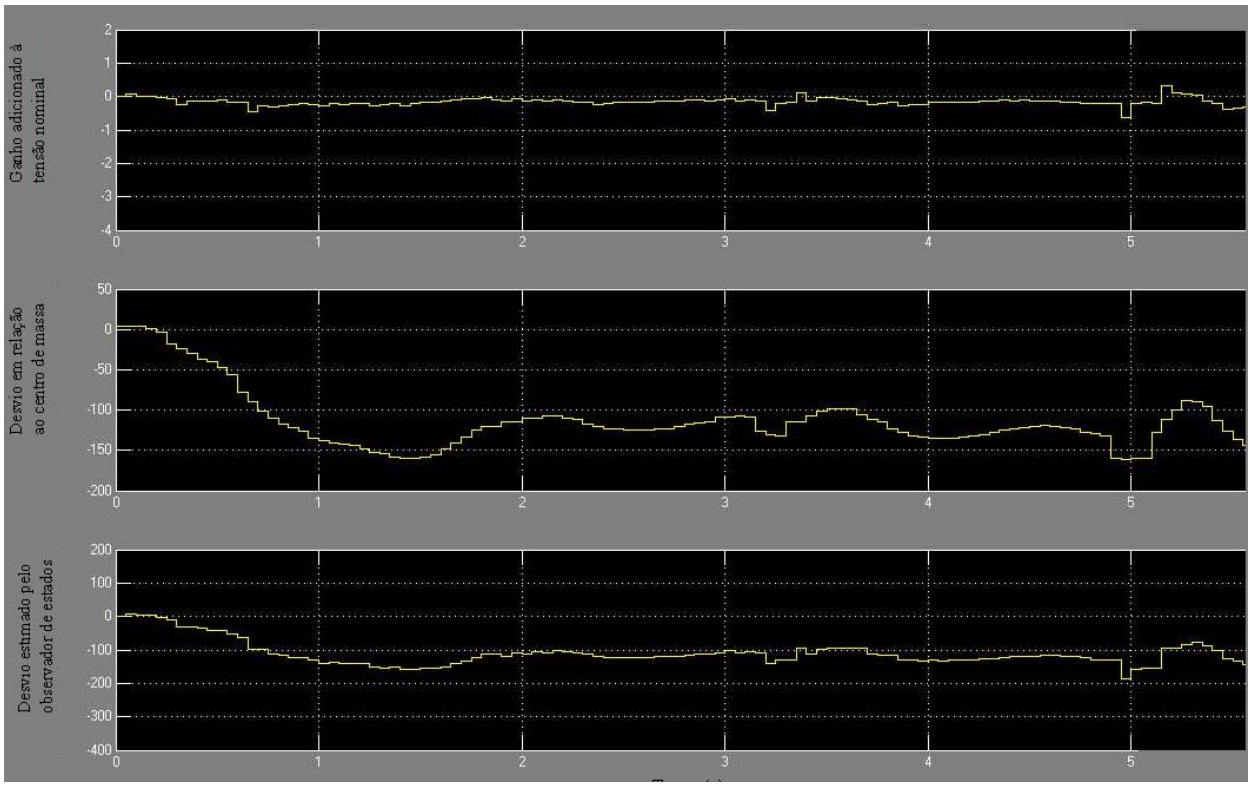


Gráfico 12- Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 2.5 \times 10^4$ e $Q = \begin{bmatrix} 0.8 & 0; & 0 & 1 \end{bmatrix}$

O valor de R foi reduzido e foram obtidos relativamente "bons" valores, isto é, o Rasteirinho apesar de se afastar um pouco da curva, verificou-se que não a perdia e que também reagia suavemente.

Baixando ainda mais o valor da primeira coluna da matriz Q, obteve-se o seguinte gráfico

Em linha curva: 1,8V R=2,5E4 Q=[0.6 0; 0 1]

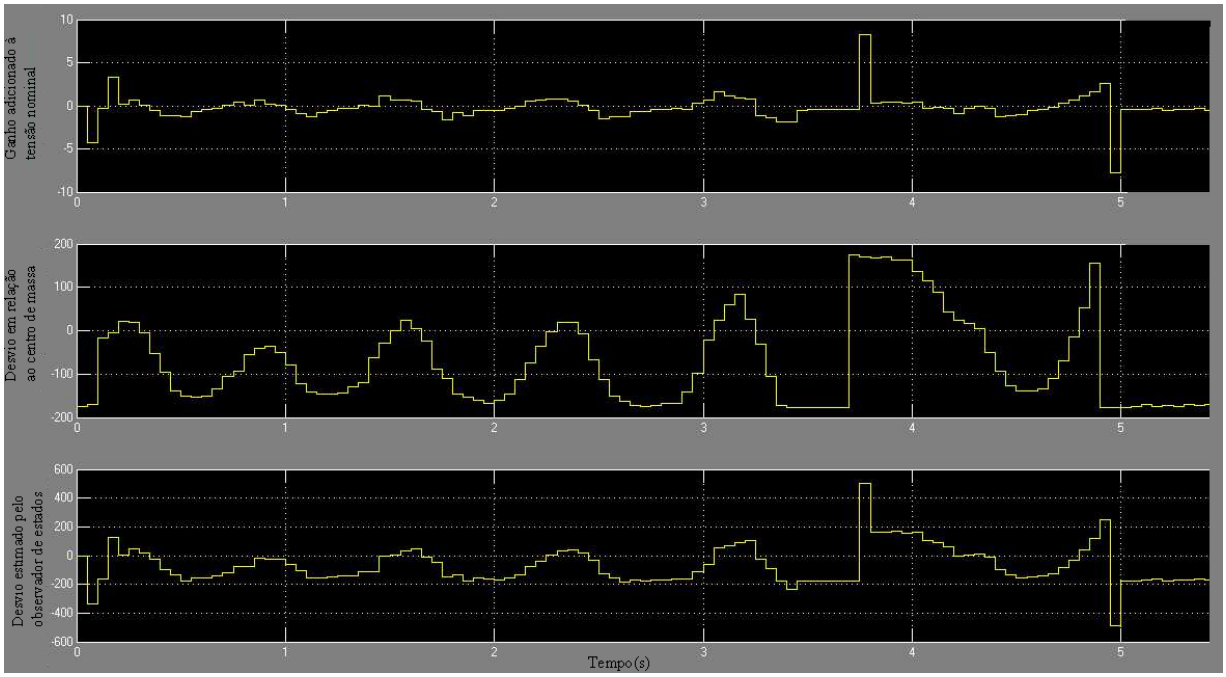


Gráfico 13 – Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 2.5 \times 10^4$ e $Q = \begin{bmatrix} 0.6 & 0; & 0 & 1 \end{bmatrix}$

Pode-se observar que o robot oscila demasiado. Para controlar essas oscilações, aumentou-se o valor de R

Em linha curva: 1,8V R=3E4 Q= [0.6 0; 0 1]

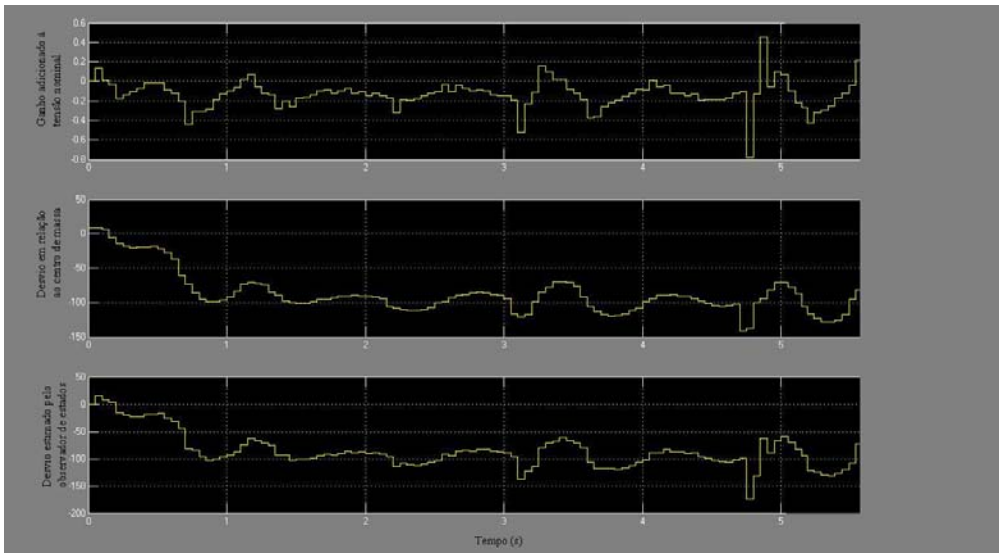


Gráfico 14 – Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 3 \times 10^4$ e $Q = \begin{bmatrix} 0.6 & 0; & 0 & 1 \end{bmatrix}$

Verificou-se que o Rasteirinho não oscila tanto e que mantém a linha menos afastada. Observando o gráfico, verifica-se a existência de alguns picos de ganho que se devem a reflexos (sombras) captados pela imagem.

Aumentando o valor da segunda coluna de Q, sem diminuir os valores da primeira, obtiveram-se alguns bons resultados que são mostrados a seguir

Em linha curva: 1,8V R=2,5E4 Q= [1 0; 0 1,1]

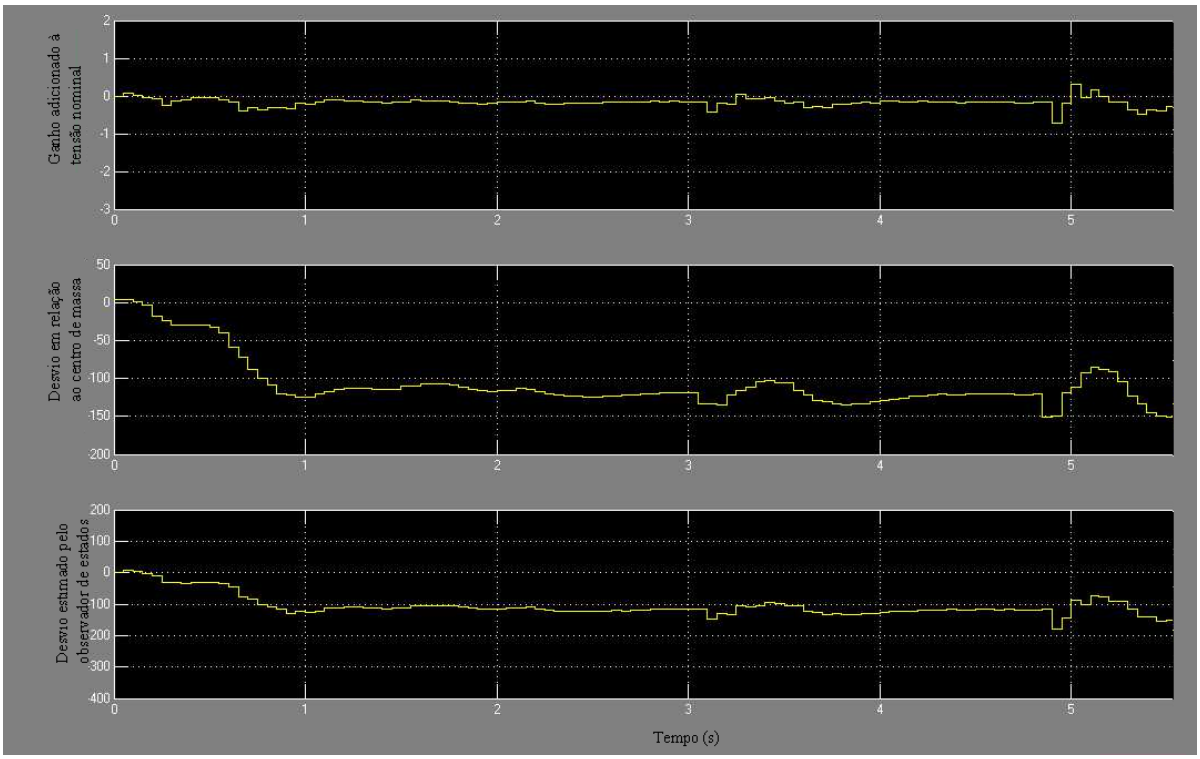


Gráfico 15 – Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 2.5 \times 10^4$ e $Q = \begin{bmatrix} 1 & 0; & 1 & 1 \end{bmatrix}$

Verifica-se que apesar do robot se afastar um pouco da linha quando é feita a curva, observa-se que a mesma é feita muito suavemente.

De modo a tornar ainda mais suave a curva aumentou-se o valor do escalar R e obtiveram-se os seguintes resultados

Em linha curva: 1,8V R=3E4 Q= [1 0; 0 1,1]

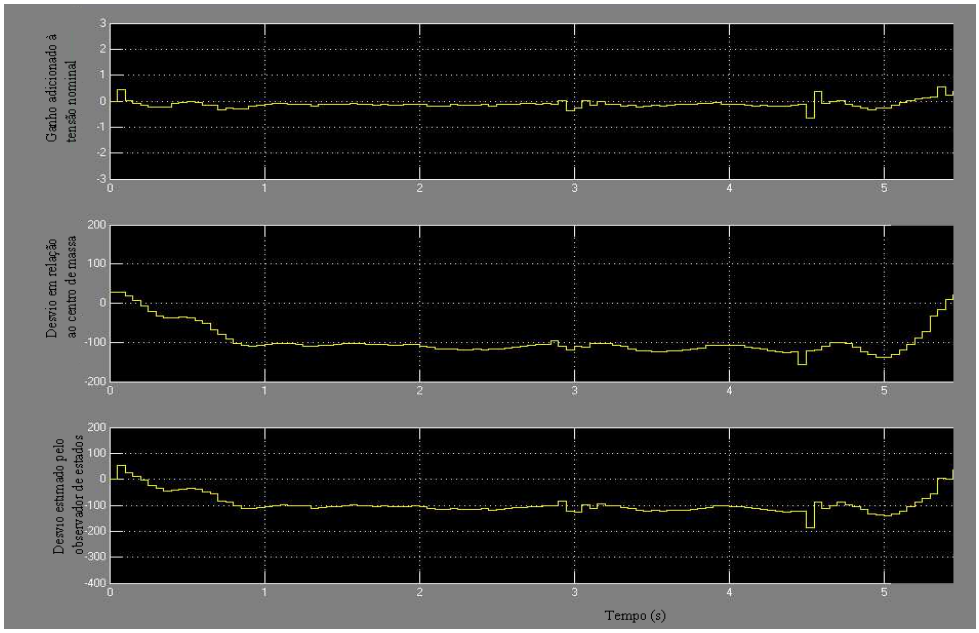


Gráfico 16 – Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 3 \times 10^4$ e $Q = \begin{bmatrix} 1 & 0; & 0 & 1.1 \end{bmatrix}$

Observa-se, então, uma menor oscilação e um menor afastamento em relação à curva.

Aumentou-se ainda mais o R

Em linha curva: 1,8V R=3,2E4 Q= [1 0; 0 1,1]

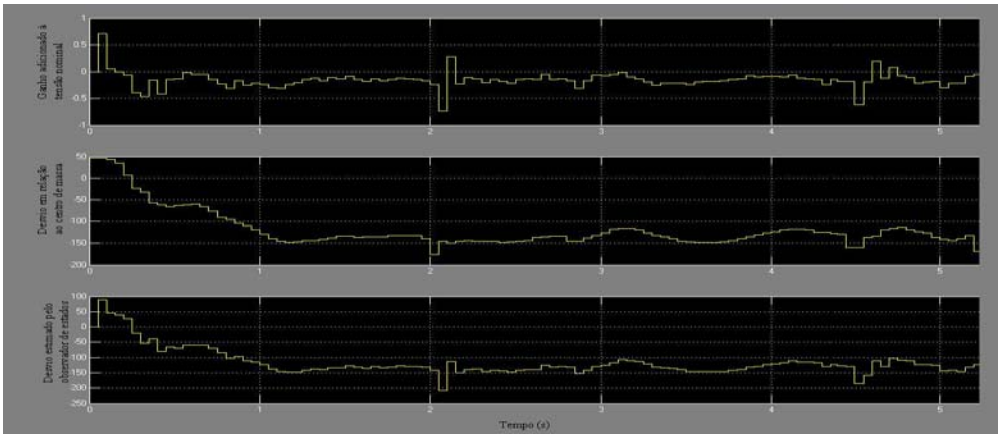


Gráfico 17 – Evolução do ganho de tensão aplicado às rodas, do desvio do centróide e do desvio estimado pelo observador em função do tempo, em linha curva, a 1,8V com $R = 3.2 \times 10^4$ e $Q = \begin{bmatrix} 1 & 0; & 0 & 1.1 \end{bmatrix}$

E observou-se que o rasteirinho se afastava em demasia devido a um excesso de suavidade de reacção.

8.Conclusão final do Projecto de Controlo Óptimo

Como conclusões finais deste trabalho prático, pode-se inferir que o objectivo do projecto foi atingido com sucesso, pois conseguiu-se fazer com que o robot descrevesse uma trajectória definida, aplicando técnicas de controlo óptimo.

Pode-se, igualmente, constatar que, não existe um modelo ideal que define o sistema que rege o robot, isto é, pode-se concluir que existem melhores ou piores aproximações. No caso dos modelos testados, verificou-se que os modelos armax e BJ eram os que melhor caracterizavam o Rasteirinho.

À medida que se efectuaram cálculos, eram perdidos valores por simplificações, ruído, etc, o que torna impossível, na prática, obter um modelo identificado totalmente equivalente ao sistema.

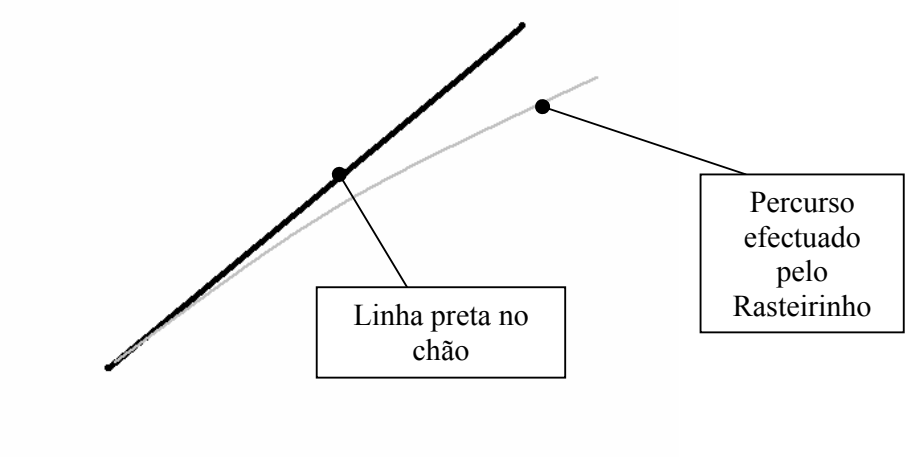
Com a obtenção de um modelo que aproxima relativamente bem os dados de origem, foi possível realizar o controlo óptimo, apenas conjugando as matrizes Q e R, consoante o percurso a efectuar e a tensão nominal que alimenta as rodas dos motores. Desta forma, obtiveram-se excelentes resultados aquando, no percurso do circuito presente no laboratório, se aplicou uma tensão nominal de 1.8V, conjugado por uma matriz Q = [0.6 0; 0 1] e do escalar $R=2.5x10^4$. Igualmente, o robot seguiu na perfeição o circuito para uma tensão de 1.8V, Q= [1 0; 0 1.1] e $R=3x10^4$. No caso em que se utiliza Q= [1 0;0 1] e $R=3x10^4$ e, igualmente, uma tensão nominal de 1.8V, também o circuito é feito sem quaisquer distúrbios.

Conclui-se assim que, igualmente, não existe um par de valores Q e R óptimos para um determinado circuito e tensões aplicadas. Todos estes "best values" foram obtidos por experimentação, não havendo uma forma matemática para prever quais os melhores valores a testar.

Quanto a aplicações, o Rasteirinho pode representar uma simulação industrial de como funcionam certos robots, sem fios, sem circuitos eléctricos ou qualquer tipo de ligações, sendo apenas accionado por um "cérebro" (computador) que antecipa o comportamento do próprio robot, evitando que este descarrile.

9.Resolução de Problemas

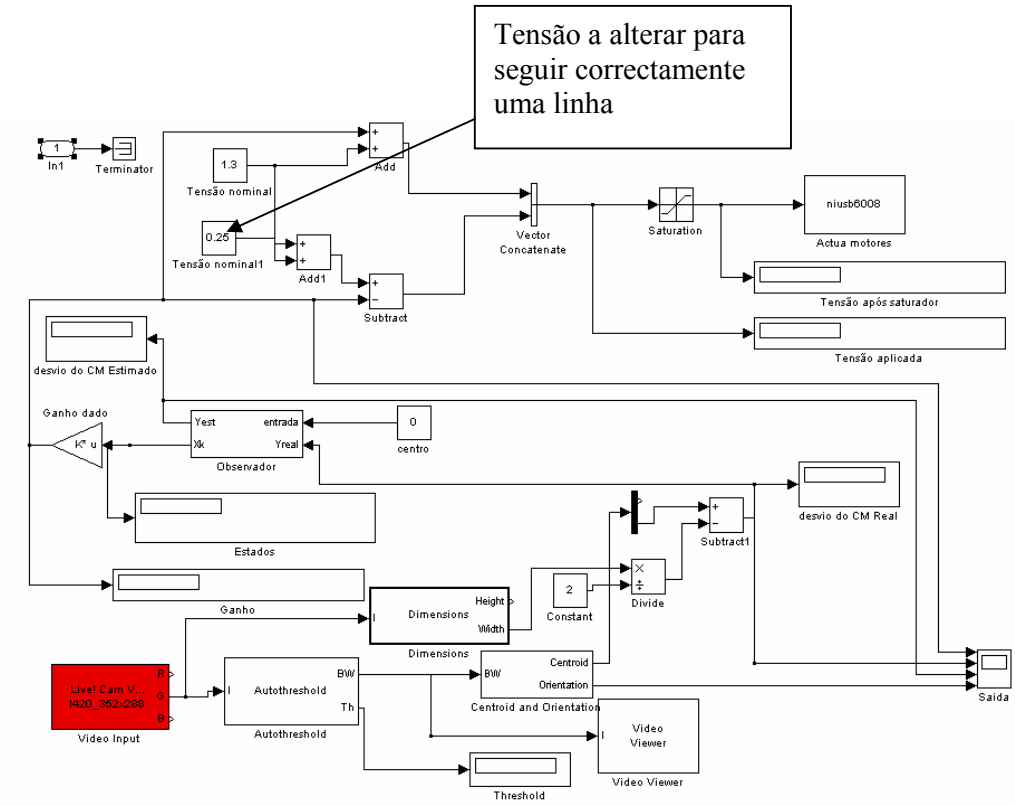
Problema 1: Dificuldades a conseguir seguir uma linha recta (robot desvia-se para um dos lados).



Causa provável: Motores do Rasteirinho são diferentes e, portanto, para a mesma tensão nominal uma roda gira mais rapidamente do que a outra.

Solução:

- Determinar qual o motor menos sensível e aplicar uma tensão adicional a este até que o robot consiga seguir a linha de forma perfeita.



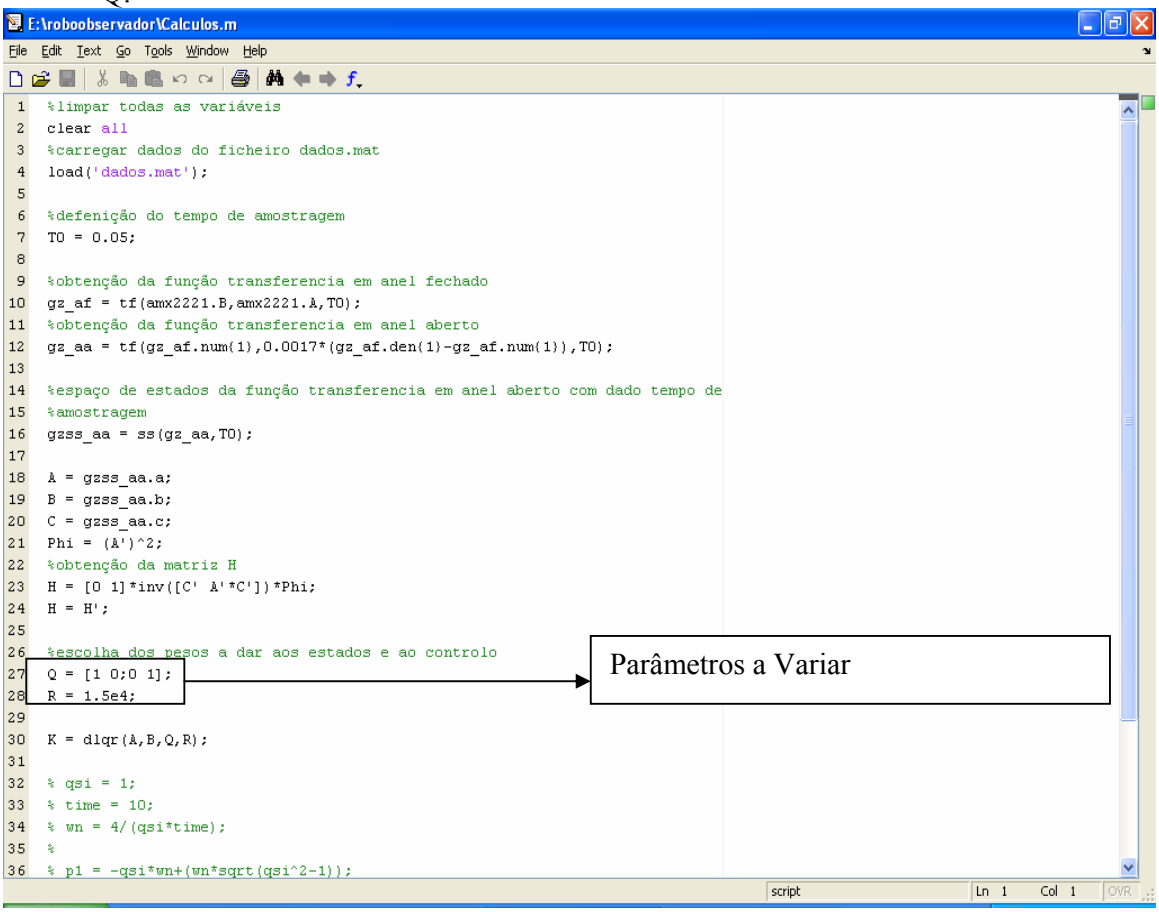
- Se o problema persistir, recarregar ambas as pilhas uma vez que uma delas poderá estar mais carregada que a outra.

<u>Problema 2</u>: Falta de estabilidade do rasteirinho

Causa provável: Valores de R e Q pouco adequados

Solução:

- O utilizador deverá abrir o ficheiro calculos.m que poderá ser encontrado na pasta do sistema. No ficheiro calculos.m poderá ser alterado os parâmetros R e Q.



Outras causas prováveis:

- Iluminação muito variável ao longo do percurso a efectuar ou outros objectos ou linhas no chão

Solução:

- Variar os parâmetros do bloco autothreshold de modo a que na imagem obtida após o threshold só a linha a seguir seja visível.

<u>Problema 3</u>: Rasteirinho pára a meio do percurso

Causa provável:

- Mau contacto entre a placa de aquisição e o computador

Solução:

- Retirar o cabo USB que liga o Computador portátil e a placa de aquisição e voltar a ligá-lo.

<u>Problema</u> 4:

- Não reconhecimento da câmara por parte do Matlab

Causa provável:

- Má instalação dos drivers da webcam.

Solução:

- Reinstalar os drivers da webcam.

<u>Problema 5</u>:

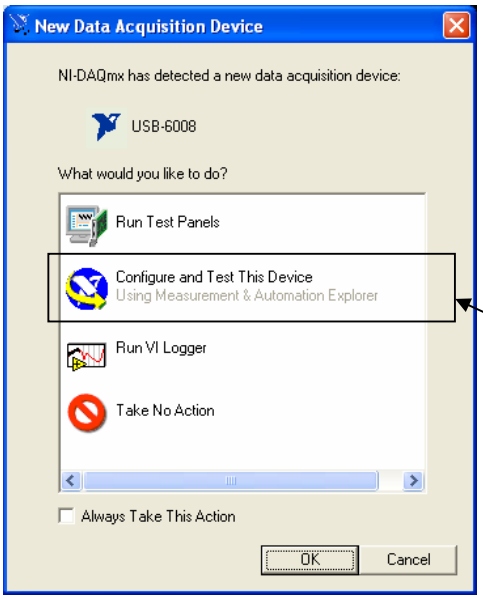
- Matlab não reconhece a placa de aquisição de dados da National Instruments.

Causa provável:

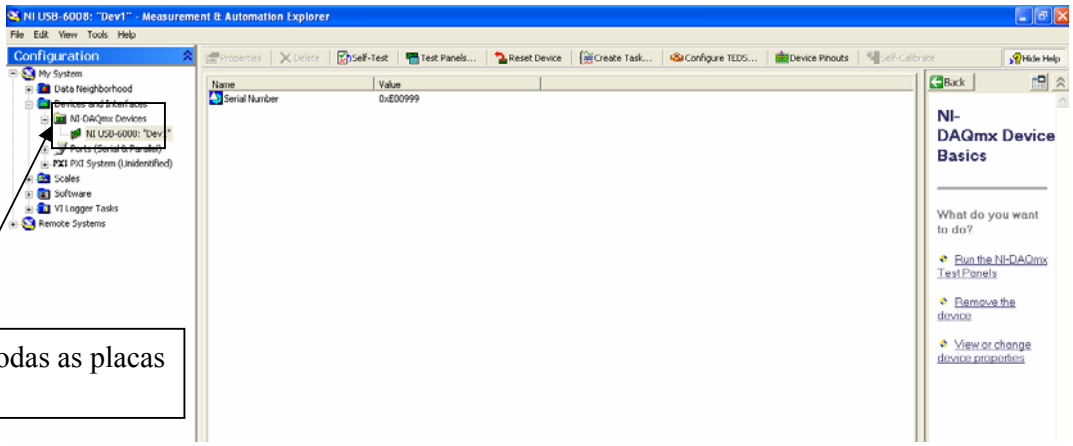
- Mais que uma placa de aquisição de dados instalada

Solução:

- Correr o programa "Measurments&Automation" e apagar todas as placas lá instaladas, de seguida deverá voltar a ligar a placa que quer utilizar.



Measurements & Automation



Apagar todas as placas

Outras causas prováveis:

- Drivers da National Instruments mal instalada

Solução:

- Desinstalar os drivers da placa de aquisição de dados e voltar a instalar os drivers[2].

[2] poderá ser necessário instalar novamente o Matlab

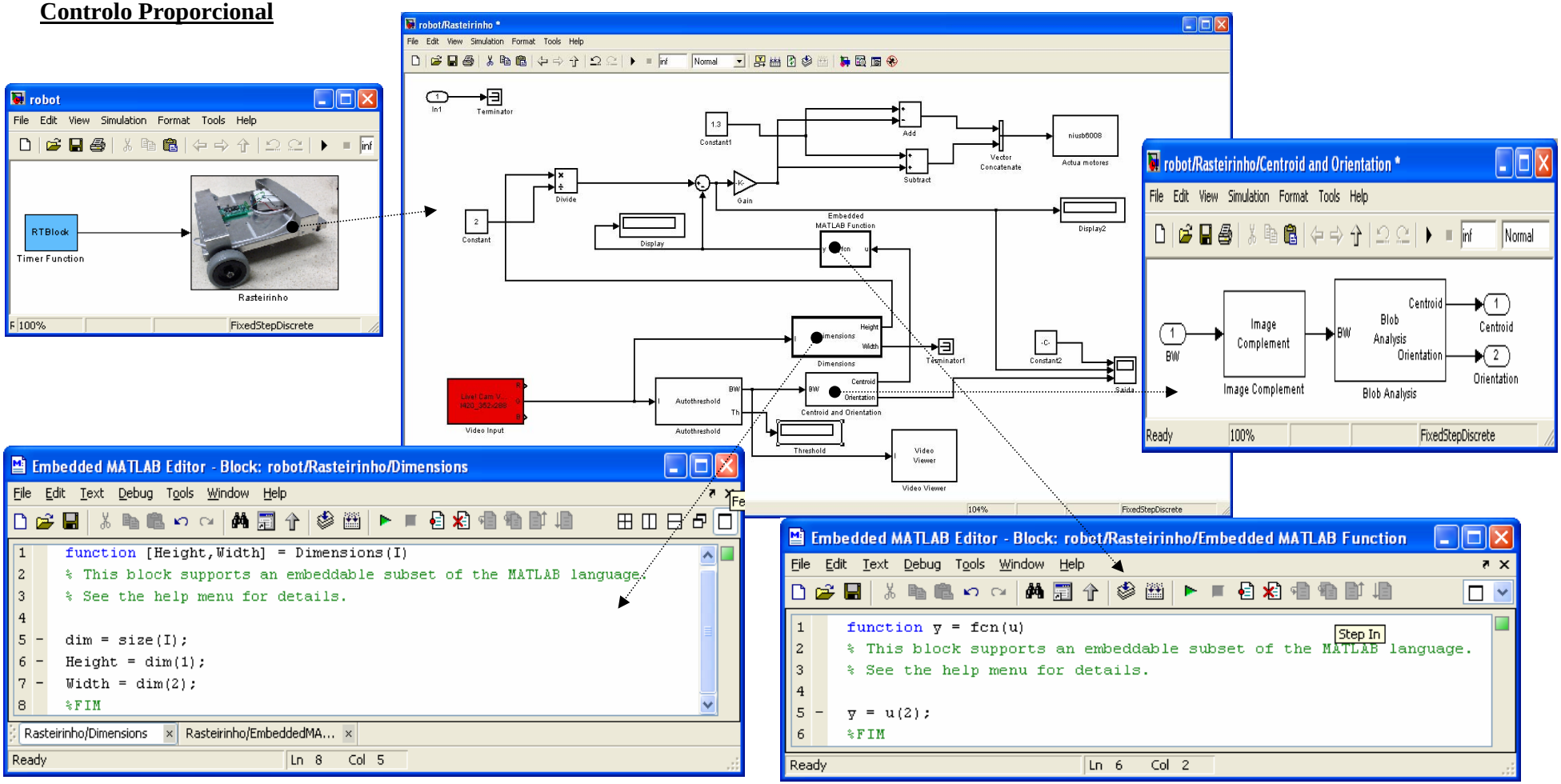
10.Bibliografia

- Botto, M.A. "*Controlo Óptimo*", AEIST, 2006

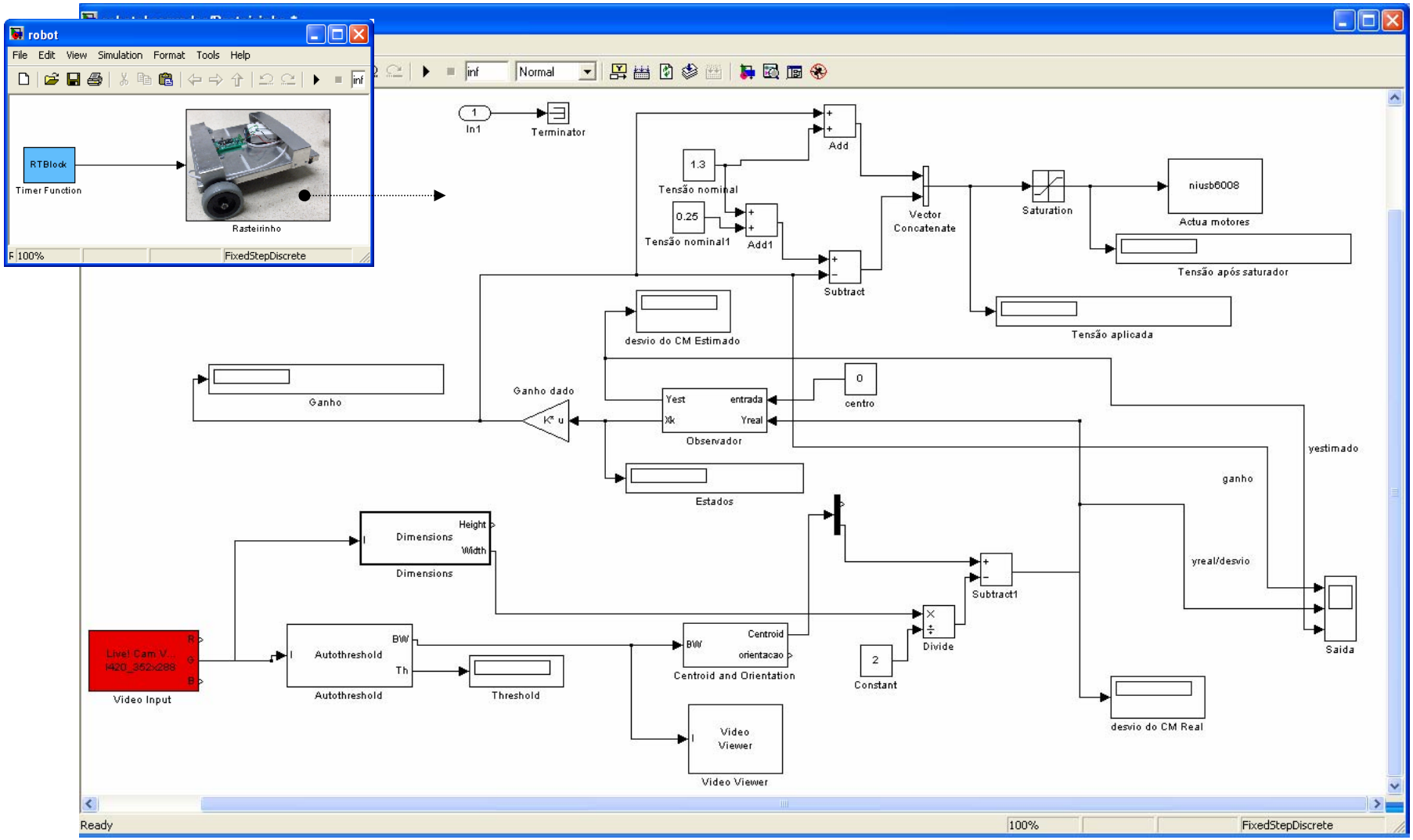
11 – Anexos

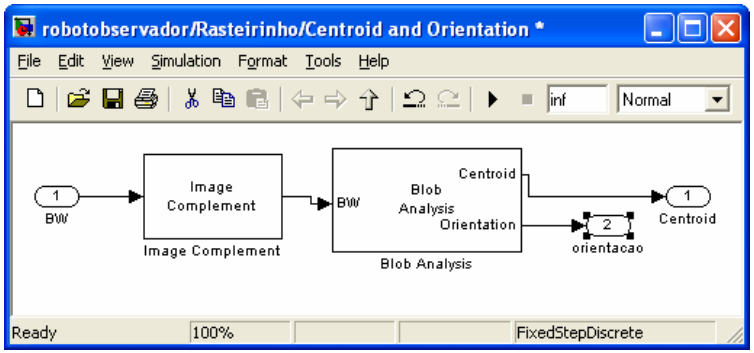
Seguidamente, será apresentado o código em Matlab e os blocos de Simulink utilizados ao longo do trabalho.

Controlo Proporcional

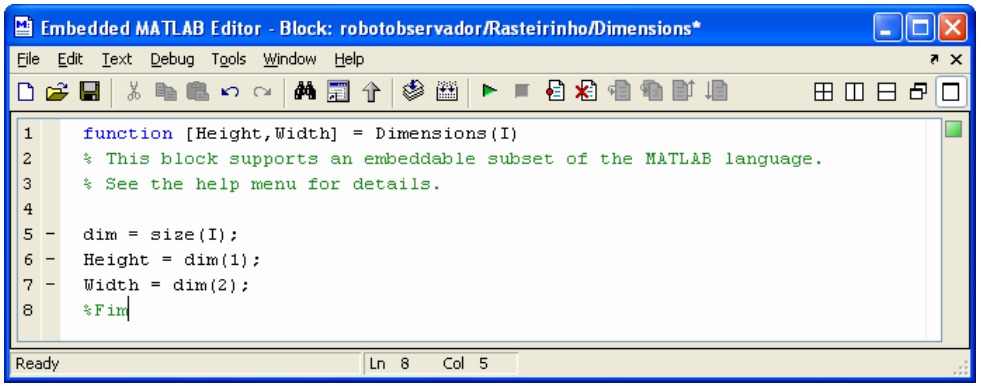


Controlo com observador de estados

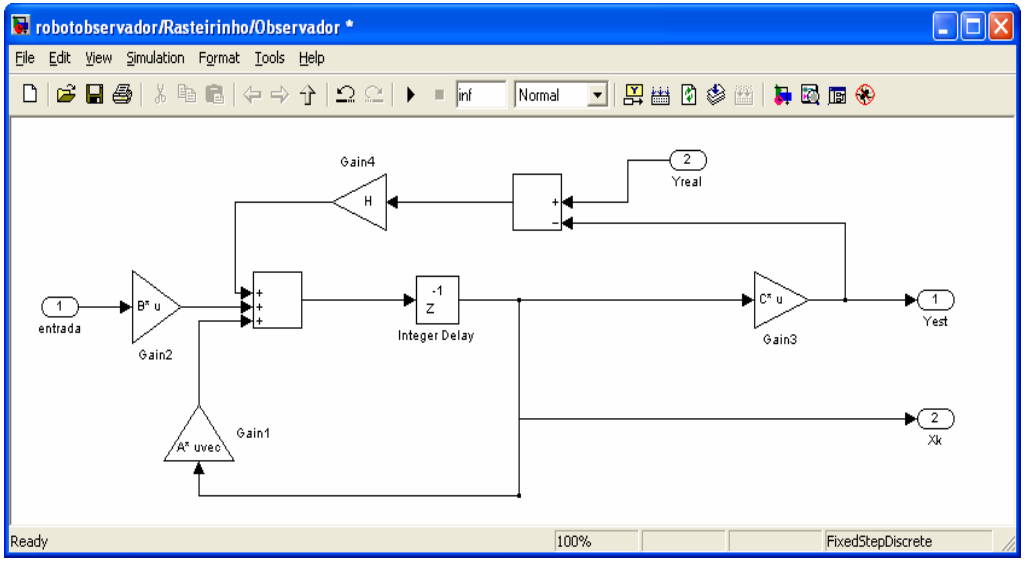




Conteúdo do bloco *Centroide and Orientation*



Conteúdo do bloco *Dimension*



Conteúdo do bloco Observador