Controle de Sistemas Mecânicos por Lógica Fuzzy

Ben-Hur Gonçalves e Roseli Aparecida Francelin Romero

1. Introdução

Este relatório tem a finalidade de apresentar os diferentes tipos de controladores nebulosos bem como controladores P, PI e PID clássicos, desenvolvidos para controlar um time de robôs móveis da categoria 'very small size'. Estes robôs estão sendo projetados junto ao Laboratório de Aprendizado de Robôs – LAR do ICMC – USP para participação em competições robóticas.

Inicialmente, é apresentada, na seção 2, o sistema geral de controle destacando-se os módulos que constituem esse sistema. Em seguida, na seção 3, o funcionamento do sistema de controle do time de robôs é apresentado, descrevendo-se como os diferentes tipos de controladores foram implementados e destacando-se suas principais diferenças. Finalmente, na seção 4, uma conclusão sobre o desempenho dos diferentes controladores é apresentada.

2. Sistema de controle de um time de futebol de robôs

Nesta seção são apresentados os módulos que compõem o sistema de controle do time de futebol de robôs

2.1. A estrutura do sistema de controle

O futebol de robôs é composto pelos seguintes módulos: visão computacional, estratégia, controle, eletrônica e mecânica. Existem itens que fazem parte de mais de um módulo, como o rádio, que além de uma entidade eletrônica também precisa ser configurado no PC da estratégia e no PIC ou ainda os motores, que são elementos da mecânica e da eletrônica. Esses elementos são as pontes de conexão entra os vários módulos. A comunicação dos sistemas é feita como na Figura 1.

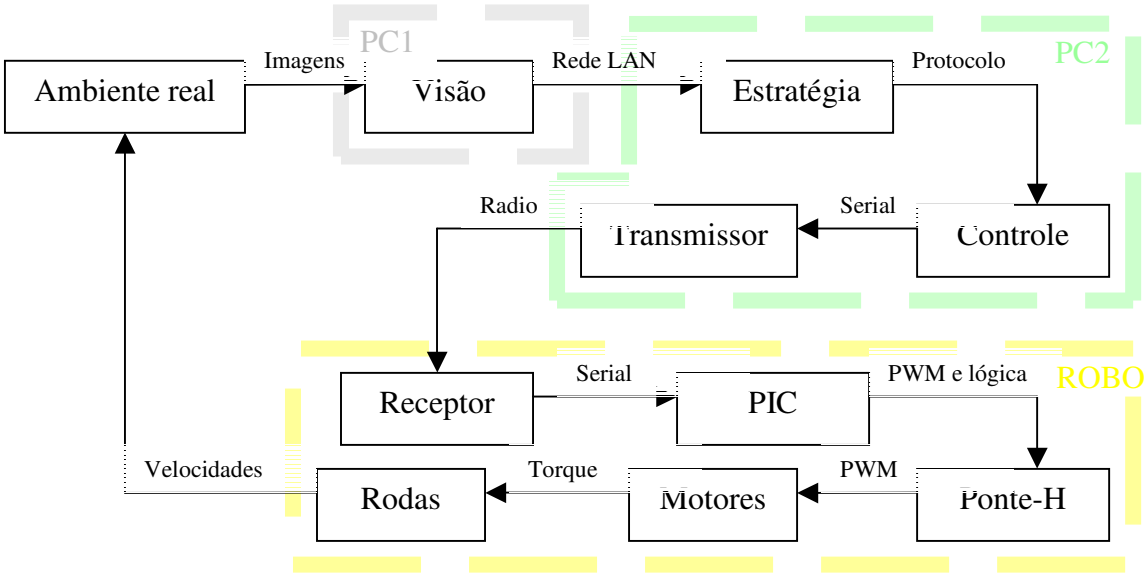

Figura 1. Modelo do processo e modelo da divisão de tarefas para cada computador.

A seguir, serão detalhados os módulos de visão, estratégia, eletrônica e mecânica que compõem o sistema. Mais detalhes podem ser encontrados na página da equipe http://uspds.sourceforge.net.

2.2. Visão Computacional

A visão computacional captura imagens através de uma câmera posicionada sobre o campo de jogo. A câmera passa as imagens para um computador com placa de captura de vídeo, que utiliza um filtro de cores HSV. Em seguida, esse mesmo computador calcula as posições dos robôs, assim como seus ângulos. Enfim, o computador responsável pela visão passa pela rede as coordenadas [x,y] e os ângulos dos robôs para o computador da estratégia.

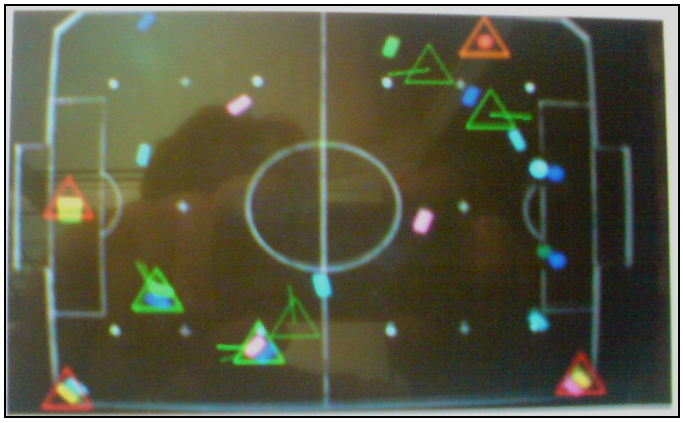

Figura 2. Imagem da visão e do tracking dos robôs.

É possível observar como a visão do time já está em funcionamento. O módulo de visão circula em verde os robôs amigos, em vermelho os inimigos e em laranja a bola, que nesse caso está no canto direito superior. As cores dispersas pelo campo servem para configuração da câmera, sendo que na hora do jogo devem ser retiradas. Maiores detalhes do sistema de visão podem ser encontrados em [Penharbel, 2008].

2.3. Estratégia

Após a estratégia receber via rede LAN as posições e os ângulos dos robôs, é realizado um processamento baseado na teoria de campos potenciais. Através dessa teoria, o robô calcula a repulsão proveniente dos robôs adversários e a atração proveniente da bola. Com esses cálculos, é possível montar um trajeto que melhor se adapta àquela situação instantânea, e com isso o robô sabe para onde deve ir. Após uma sucessiva seqüência de cálculos, a estratégia sabe para que ângulo o robô deve virar. Com esse ângulo, as funções de controle nebulosas descobrem a velocidade ideal de cada roda e consequentemente a proporção de fases que o PWM deve trabalhar.

Como o controle realizado anteriormente não convergiu, foi necessário mudar um pouco a estrutura da estratégia. Para implementar o novo controle, foi necessário adicionar uma nova função a esse módulo. Essa função calcula a velocidade atual de cada roda do robô, baseando-se nas posições e ângulos de dois quadros diferentes. Como o ciclo de visão é realizado sempre a cada 33 milisegundos, podemos realizar esses cálculos com certa precisão.

Com a velocidade ideal calculada pela estratégia antiga e com a real calculada por essa nova função, podemos acionar a nova função de controle, sendo que esta passa como saída as proporções e fase para os PWMs das rodas. Note que com esse novo controle, se quisermos que o robô ande reto, deveremos passar intensidades diferentes para cada roda, já que as velocidades reais das rodas provavelmente seriam diferentes.

Após realizado esses cálculos, a estratégia aciona a porta serial, e essa está ligada diretamente ao rádio transmissor. Passa-se o vetor "$A,B,C,D,E*", onde o sinal "$" indica o início da transmissão, "A" o sentido da roda esqueda, "B" indica a velocidade desejada para a mesma, numa escala de 0 a 63, "C" o sentido da outra roda, "D" indica a velocidade desejada para a mesma, também numa escala de 0 a 63. e o caractere "*" indica o final da mensagem.

2.4. Eletrônica

No projeto da eletrônica estão presentes os seguintes componentes: rádio, microprocessador, duas baterias, motores, ponte-H, dentre outros. Acompanhe na Tabela 1 os elementos e pela Figura 3 abaixo o esquema da eletrônica.

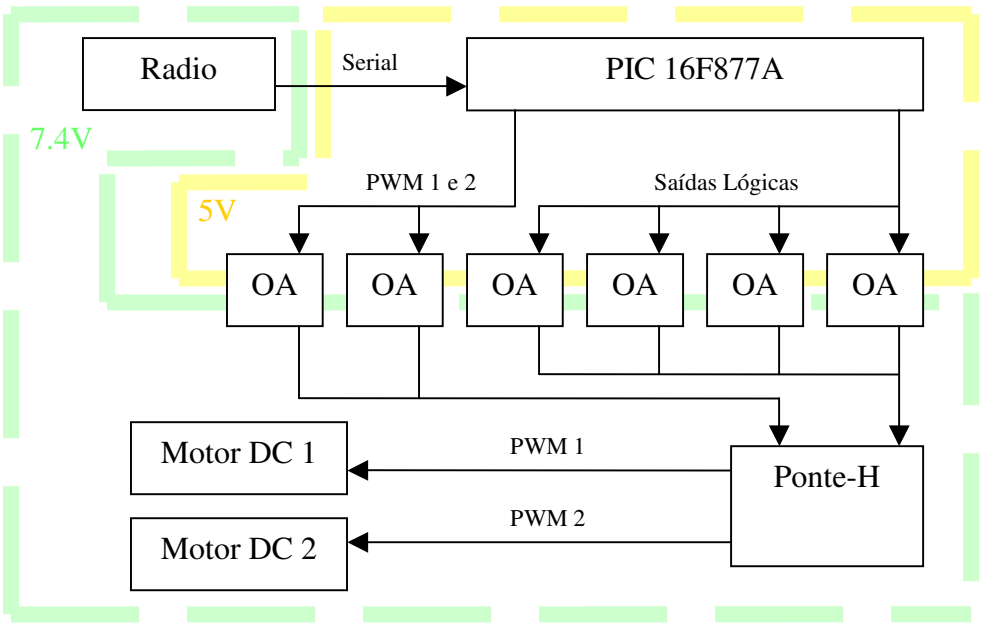

Figura 3. Módulo eletrônico do robô com opto acopladores e duas baterias.

Como podemos ver, foram utilizadas baterias diferentes para o PIC e para o resto do circuito. Isso porque se a alimentação fosse a mesma, o motor podia resetar o PIC toda vez que tentasse sair da inércia. Isso acontece porque o motor corrente contínua precisa de muita potência para sair do estado estático, e isso se resume a maiores correntes elétricas (Valentine, 1998). Os opto acopladores servem para separar as correntes e voltagens de um lado do circuito com as do outro. Ele funciona como um transistor: quando passa corrente no alimentador, ele deixa passar corrente na sua saída.

Quem for repetir o modelo eletrônico fique atento ao fato de cada opto acoplador trabalhar até uma freqüência máxima. A partir daí o elemento fica insensível à mudança

de estado e começa a não responder corretamente. Como o pulso do PWM trabalha em alta freqüência, é necessário calcular e ajustar certos parâmetros internos do PIC para que o sistema funcione bem (Souza, 2005).

Como estamos controlando dois motores de corrente contínua, utilizamos uma ponte-H, pois assim evitamos utilizar duas portas a mais do microprocessador, também evitamos usar mais quatro transistores, além de muitas trilhas desnecessárias. Destaque: foi utilizado um regulador de tensão para o PIC trabalhar com 5 Volts. Os componentes utilizados estão na tabela abaixo.

Tabela 1. Listagem dos componentes da eletrônica para cada robô.

Componente Utilizado	Especificação do Item	Quantidade
Rádio	Linx HP RXM-900-HP3-PPO	1
Bateria da eletrônica	9 V NiMh Recarregável	1
Bateria dos motores	7.4V LiPo 1300 mAh 20 C	1
Microprocessador	PIC 16F877A	1
Ponte-H	L298N	1
Capacitores	33 pf e 100 uF	2, 1
Resistores	560 Ohms , 1000 Ohms	7, 9
Cristal	4.000.000 Hz	1
Antena do rádio	Linx 916-SP2 Antena Planar	1
Acopladores ópticos	817C	6

O módulo da eletrônica se comunica com dois outros: o da mecânica e o do controle. O controle passa através do transmissor do rádio o vetor "$A,B,C,D,E*" para o rádio da eletrônica. O rádio passa o comando para o PIC através da interface RS-232 da comunicação serial. O microprocessador começa a ler o sinal "$" e percebe que está chegando uma nova mensagem. A partir daí associa "A" ao sentido da roda esquerda, "B" passa a ser a velocidade desejada para a mesma, numa escala de 0 a 63, "C" ao sentido da outra roda, "D" passa a ser a velocidade desejada para a mesma, também numa escala de 0 a 63. O caractere "*" indica o final da mensagem.

Os parâmetros do sentido das rodas passam por uma lógica e se dividem em 4 portas do PIC, acionando os pinos de sentido da ponte-H. Já as velocidades das rodas acionam o mecanismo PWM do microprocessador e partem dois pulsos de portas diferentes, também em direção à ponte-H. São esses dois pulsos, associados aos quatro

pinos lógicos de sentido que realizam a comunicação com o módulo da mecânica. O código em Assembly da programação do 16F877A pode ser encontrado na página do grupo. Esse código inclui a plataforma de comunicação RS-232 e o mecanismo de controle dos pulsos.

2.5. Mecânica

Um protótipo mecânico do robô foi desenhado na ferramenta Solid Edge (Figuras 4 e 5), licenciada para USP – São Carlos e o modelo foi usinado em alumínio (carcaça) e náilon preto (rodas). As rodas foram usinadas no torno mecânico e o alumínio foi serrado e acabado na fresa de precisão. Os furos foram realizados através da furadeira de precisão. Os furos também foram escareados e realizados para parafusos M3. Já as placas de alumínio possuíam profundidade de 5mm. A próxima tabela ilustra os processos. A seguir, encontram-se as características dos motores, assim como as da redução utilizadas e o rendimento final do sistema.

Tabela 2. Listagem de processos para confecção da redução do robô.

Elemento	Material	1º Processo	2º Processo	3º Processo
Rodas	Náilon	Tornear exterior	Furar com precisão	Nenhum
Carcaça	Alumínio 5 mm	Serrar com folga	Fresar com precisão	Furar em precisão

Tabela 3. Características do motor em máximo rendimento.

Modelo de Motor	Voltagem	Torque	RPM	Potencia	Rendimento	Distribuidora
PM-080 SS05	5 V	14 gcm	5450	0,77 W	55,4 %	Action Motors

Tabela 4. Características da redução.

Redução	Roda	Torque Final	RPM Final
1 para 4 vezes	50 mm	204 g.cm por motor	1362

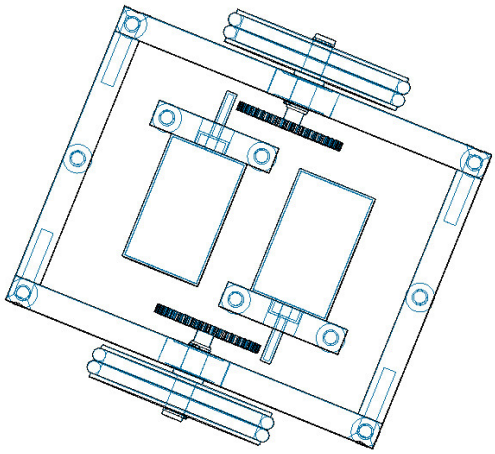

Figura 4. Visão geral da mecânica.

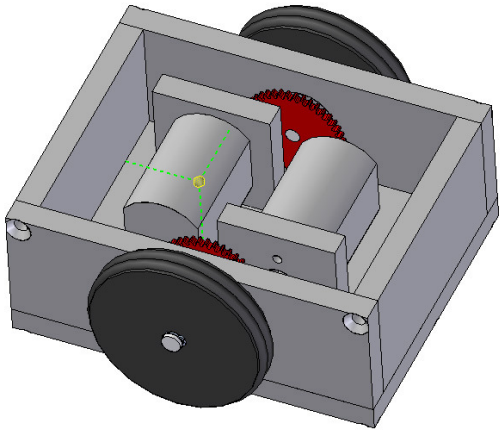

Figura 5. Caixa de redução completa.

O módulo mecânico se comunica com o restante do robô pelos motores e se comunica com o ambiente ao redor por suas rodas. É bom lembrar que o motor PM-080 foi projetado para ser utilizado com uma voltagem nominal de 5V, enquanto que para obtermos maior torque estamos fornecendo 7.4V de pico. Isso corresponde a uma maior potencia de 112900 gcm.RPM contra 76300 gcm.RP. É uma sobrecarga de 48%.

3. Controle

Nesta seção será descrito como o sistema de controle do time de robôs foi desenvolvido, especificando o seu funcionamento, descrevendo como os diferentes tipos de controladores foram implementados e destacando-se suas principais diferenças.

3.1. Implementação

O intuito do controle desenvolvido até aqui era ter os robôs constantemente monitorados pela visão, de forma que se o robô saísse da trajetória, a visão avisaria a estratégia e esta mandaria pulsos de PWM com maior duty-cycle, fazendo uma roda andar mais que a outra, corrigindo a trajetória. Como o processamento da visão aconteceria em menos de 33 milisegundos, essas correções seriam imperceptíveis. A figura abaixo ilustra o processo.

Assim que o controle foi implementado, a solução não convergiu, como era de se esperar. Seria necessário então ajustar os parâmetros do controle, mas isso se mostrou muito difícil, pois esse tipo de controle não é estável. O projeto mecânico ficou muito forte, ou seja, o robô é rápido demais e possui muita força inercial – o que faz com que seja um mecanismo de difícil controle. Seria necessário um sistema de controle mais fino.

Por exemplo, no caso de uma curva fechada seguida de uma reta. A diferença de duty-cycles do pulso de um motor para o outro faz o robô girar muito rápido. Quando é requerido a este que vá em linha reta, ele não consegue em função de sua inércia. Continua então a girar, mas com um raio de curva maior. Isso porque mandar pulsos iguais para as rodas não é suficiente.

É conhecido da teoria de controle PID que se o erro de uma variável é zero, ou seja, já atingiu o valor desejado, mas a derivada do erro ainda é grande, ou seja, o controle está avançando muito rapidamente, então é necessária uma força contrária de módulo alto, senão o erro passará a ter sinal invertido e módulo alto.

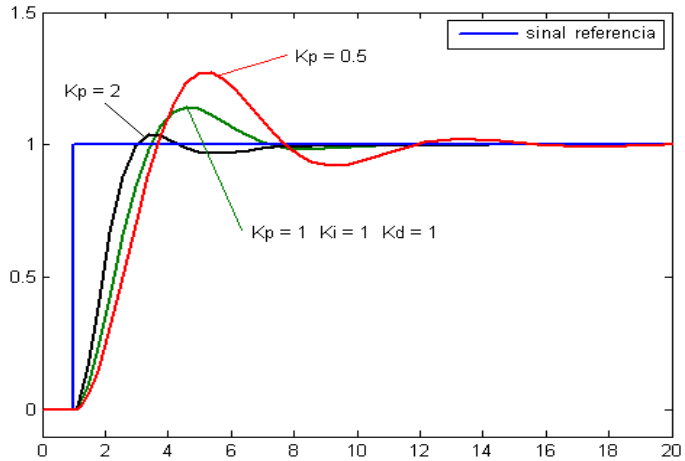

Figura 6. Várias respostas do PID para um sinal de referência do tipo degrau.

Seria muito mais custoso ajustar um controle instável do que desenvolver um novo controle robusto, do tipo PID ou ainda PID-fuzzy. Isso porque o controle antigo prediz a resposta apenas em função da velocidade desejada, portanto não possui realimentação. Já os novos controles propostos predizem a saída em função do erro (diferença da velocidade desejada e da real), da derivada do erro e da integral do mesmo. A própria lógica nebulosa pode suportar esse sistema PID, é só possuir as três entradas mencionadas.

3.2. Modelagem do robô

Antes de projetarmos o controle PID, devemos modelar o robô. Dessa maneira será muito mais fácil ajustar os parâmetros do controle, pois esse processo será feito off-line. Na maioria dos processos industriais os ajustes são feitos na prática, mas esse procedimento é ruim, uma vez que impede que o ser humano tenha uma visão clara do sistema. Através da modelagem será possível também diminuir o tempo de ajuste, uma vez que o processo real é muito dinâmico para os operadores humanos avaliarem.

A modelagem começa com a estrutura interna elétrica do motor. Sabemos, pelas teorias elétricas que a voltagem equivalente V aplicada ao motor gera uma corrente I que transmite potência para a resistência interna Rm do motor, para a indutância L do motor e para a força eletromotriz E, que é a responsável por gerar movimento. Acompanhe:

$$V = R_m I + L\dot{I} + E$$

$$I = \frac{(V - E)}{R_m} \qquad (1)$$

Também podemos relacionar E com a velocidade angular do eixo do motor, ωm. Sabe-se (Cha, Rosenberg, e Dym, 2000) que ambos são dependentes por um termo constante Kg. A fórmula (1) será modificada:

$$E = K_g \, \omega_m$$

$$I = \frac{(V - K_g \, \omega_m)}{R_m} \tag{2}$$

A corrente que passa pelo motor é diretamente proporcional ao torque gerado no mesmo. Tm indica esse torque e Km é a constante da fórmula, na unidade de Nm/Ampere. O símbolo η é o rendimento do motor. Os raios do pinhão, da engrenagem e da roda são respectivamente, Rpin, Ren, Rr. Acompanhe o resultado:

$$T_m = \eta K_m I$$

$$\frac{T_m}{\eta K_m} = \frac{(V - K_g \omega_m)}{R_m}$$

$$\omega_m = \omega_{pin} = \frac{\omega_r R_{en}}{R_{pin}} \tag{3}$$

$$T_2 = \frac{(\eta . K_m)}{R_m} \frac{(V - K_g \omega_r R_{en})}{R_{pin}} - \frac{J_m \dot{\omega}_r R_{en}}{R_{pin}} \tag{4}$$

$$T_1 = \frac{T_2 R_{en}}{R_{pin}} \tag{5}$$

$$T_1 = (J_{en} + J_{eixo} + J_r) \, \dot{\omega}_r + f_{at} R_r \tag{6}$$

$$f_{at} = \frac{\dot{\omega}_r J_{robo} R_r}{4 R_{robo}^2} \tag{7}$$

Essa equação (5) vem do fato de que as forças nas engrenagens são iguais (ação e reação), portanto seus momentos são proporcionais a seus próprios raios. Já a equação (6) vem da terceira lei de Newton para o momento do eixo do motor. Nela, Jen, Jeixo, Jr são respectivamente os momentos de inércia da engrenagem, do eixo e da roda. Note que o momento do pinhão faz parte do momento do motor, pois estes dois são componentes ligados uns aos outros. Posteriormente foram desconsiderados os momentos de inércia, exceto o do robô, pois Jrobo >>> Jeixo + Jen + Jr. Unindo as fórmulas (3), (4), (5), (6) e (7) em função de V e de ωr, teremos:

$$\frac{\omega_r}{V}(s) = \frac{A}{Bs + 1} \tag{8}$$

$$A = \frac{R_{pin}}{R_{en} K_g} \qquad B = \frac{J_{robo} R_r^2 R_{pin}^2 R_m}{4 R_{robo}^2 R_{en}^2 K_m K_g \eta}$$

3.3. Adquirindo dados do sistema

Precisamos agora determinar os valores das constantes do motor, da roda, da caixa mecânica. O data-sheet do motor PM080-SS05, documento que foi encomendado por nós na própria empresa fabricante, fornece a seguinte tabela para consulta.

Tabela 5. Características do motor.

Modelo	Voltagem	Sem carga		Stall	
		Rotação (Rpm)	Corrente (A)	Torque (gcm)	Corrente (A)
PM080	5V	7450	0,11	51	0,78

No início, a velocidade angular do motor é nula, portanto, com V = 5 Volts, podemos calcular pela fórmula (1) a resistência elétrica do motor. Com o motor sem carga podemos calcular a constante Kg do motor. Com a proporção da reta criada do torue pela corrente, podemos determinar Km. Também podemos calcular as outras constantes do sistema com base na geometria das peças do robô. A Tabela 6 fornece os valores encontrados.

Tabela 6. Listagem das constantes do robô.

Constante	Valor	Descrição do item
R_m	6,41 Ohms	Resistência elétrica do motor
K_g	0,00505 Vs/rad	Resistência de rotação do motor
K_m	0,006414 Nm/A	Resistência de torque do motor
J_{robo}	0,0006125 Kg.m^2	Momento de inércia do robô todo
R_r	0,030 m	Raio da roda
R_{en}/R_{pin}	3	Proporção da redução
η	12 – 100 %	Eficiência do motor
R_{robo}	0,035	Distância entre o centro de gravidade e a roda

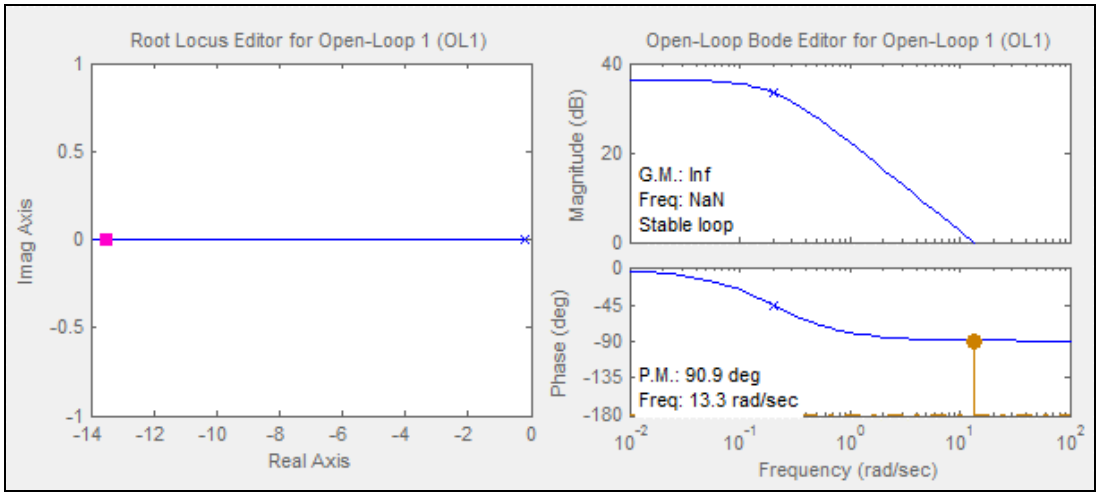

Figura 7. Lugar das raízes para o modelo matemático do robô.

3.4. Desenvolvendo o controlador PID

O controlador PID possui três constantes a serem ajustadas. A constante proporcional K acelera a reação do sistema, enquanto a constante integral Ti diminui o erro do sistema em estado permanente e a constante derivativa Td amortece as curvas de reação, evitando que entre em instabilidade. A parte derivativa do controle aumenta a estabilidade do modelo matemático por adicionar um zero à função de transferência em malha fechada, deslocando o gráfico do lugar das raízes para esquerda [Franklin, 1994]. A fórmula padrão de um controlador proporcional-integral-derivativo é dada:

$$D(s) = K(1 + \frac{1}{T_i s} + T_d s)$$

Por diversas décadas os engenheiros tentaram analisar sistemas dinâmicos e definir valores para os ganhos de um controlador PID, puramente baseados nas plantas dos processos, mas foi com J. G. Ziegler e N. B.Nichols que foi definida uma lógica para K, Ti e Td. O método Z-N de malha fechada é um dos métodos mais comuns utilizados para sintonizar controladores. O método de laço aberto também é útil para a maioria dos processos de controle. O algoritmo de laço fechado determina o ganho proporcional Kcr no qual um sistema ficará submetido a oscilações sustentadas. Em seguida, o algoritmo recalcula novos ganhos proporcional, integral e derivativo em função desse valor encontrado. Esta é considerada boa afinação, mas não é necessariamente um ajuste ótimo. Com esse segundo método, encontramos Kcr e T, que é o tempo do ciclo de oscilação sustentada, e calculamos os outros ganhos:

Tabela 7. Listagem das constantes do controlador.

	K	Ti	Td
Fórmula	0,6.Kcr	0,5.T	0,125.T
Valor Encontrado	0,24	0,825	0,206

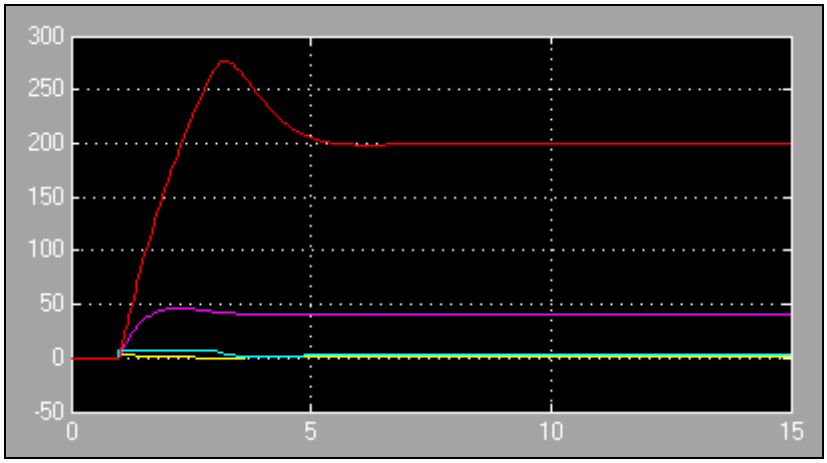

Figura 8. Resposta a entradas degrau de 40 e 200 rad/s para PID com Ziegler-Nichols.

O controlador da Figura 8 acima está submetido à função de saturação, que limita o fornecimento até 7,4 Volts ao motor. Vemos que o chute inicial de Ziegler-Nichols funciona relativamente bem para o controlador, porém seria muito útil retirar esse sobressinal da resposta. Para isso, foram feitos ajustes manuais das constantes.

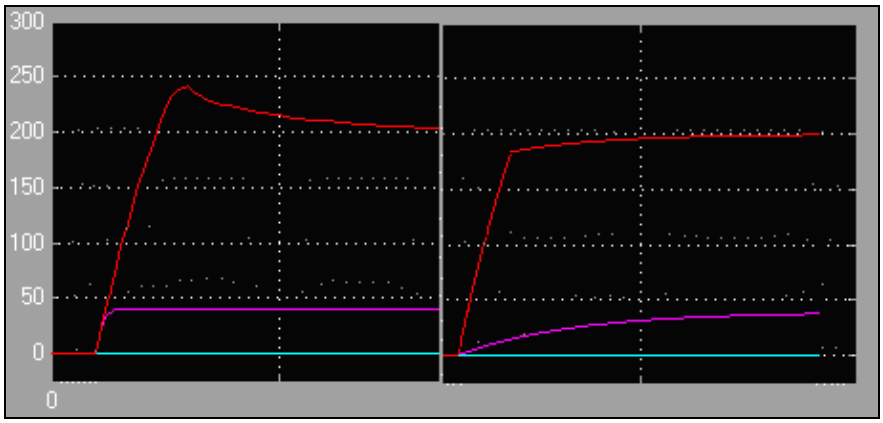

Figura 9. Comparação entre Td altos e baixos para entradas altas e baixas.

Veja que o controlador PID é sujeito ao tipo de entrada que lhe é fornecida. Para valores altos de Td, valores altos de entrada convergem corretamente, mas o mesmo controlador, para valores baixos de entrada, demora a convergir. Para valores baixos de Td, a resposta a entrada alta demonstra sobressinal, enquanto que a entrada baixa converge corretamente.

3.4. Desenvolvendo o controlador PI-fuzzy

O controlador PI-fuzzy trabalha com o erro e com a integral do erro. Isso quer dizer que tem as mesmas entradas de um controlador PI clássico, mas ao invés dos ganhos Kp e Ti, possui essas entradas tratadas por um conjunto de funções nebulosas. Essas funções estão descritas abaixo em termos de suas entradas, onde cada N dá uma idéia de como a entrada é negativa e cada P indica o grau que a entrada é positiva. Esse sistema de nomeação também funciona para a saída, que é a voltagem aplicada ao motor.

Como é possível ver na Tabela 8, a variável erro tem um fator de 30% na saída, em comparação com a variável da integral do erro. Vemos também que o que está aplicado na Tabela 8 abaixo é a mesma idéia que um operador humano utilizaria para controlar o sistema: se o erro é positivo, é porque a velocidade de referência é maior que a real, portanto é necessária uma potencia positiva na saída, enquanto que a integral do erro detecta quando o erro de regime é grande.Nesta tabela, NNN, NN, N representam as funções de pertinência para quantizar valores negativos e P, PP e PPP representam as funcões de pertinência para quantizar valores positivos.

Tabela 8. Tabela de relações das entradas com a saída.

Variável	Relações entre as funções						
Int. Erro	NNN	NN	N	Z	P	PP	PPP
Voltagem	NNN	NN	N	Z	P	PP	PPP
Erro	NNN	NN	N	Z	P	PP	PPP
Voltagem	NNN.0,3	NN.0,3	N.0,3	Z.0,3	P.0,3	PP.0,3	PPP.0,3

O fator de 30% do erro foi obtido empiricamente através das simulações no Matlab. Foram testados vários tipos de fuzzyficação e defuzzyficação. Pela Tabela 8 também vemos que as variáveis de entrada são independentes entre si. A superfície de controle do controlador PI-fuzzy está na Figura 10, abaixo. Vemos pela Figura 10 que a superfície de controle é complexa, mas ainda assim suave. A variável do erro está no espaço [-400, 400], enquanto que a integral do erro está no espaço [-50, 50]. Note que quando se usa controle nebuloso não é necessário utilizar uma função de saturação da saída, pois é possível configurar esse valor máximo e mínimo na própria lógica nebulosa. No gráfico da Figura 11 fica claro que o sistema nebuloso respondeu mais rápido que o PID clássico, porém sobrou um erro de regime no valor de 5% do degrau de entrada.

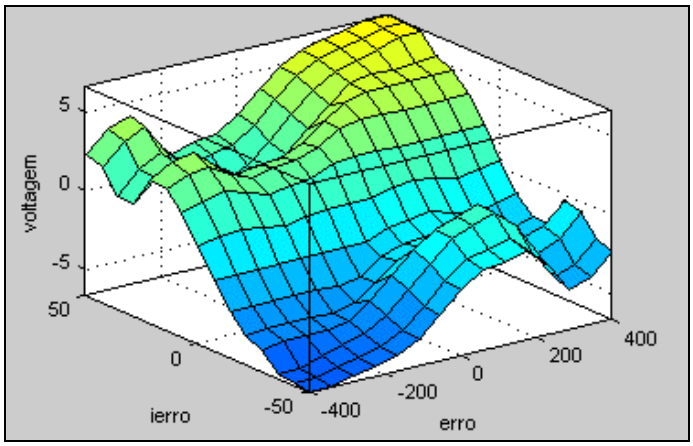

Figura 10. Gráfico do sistema de controle nebuloso.

No gráfico 11 em seqüência, vermelho indica a resposta do sistema para uma entrada degrau no valor de 200 rad/s. O roxo indica a resposta para uma entrada no valor de 40 rad/s. Como podemos perceber, o controlador nebuloso permite convergência tanto para entradas altas como para baixas, ao contrário do PID. O problema do controlador nebuloso está no fato de ser mais difícil de configurar, o que provoca o erro de regime.

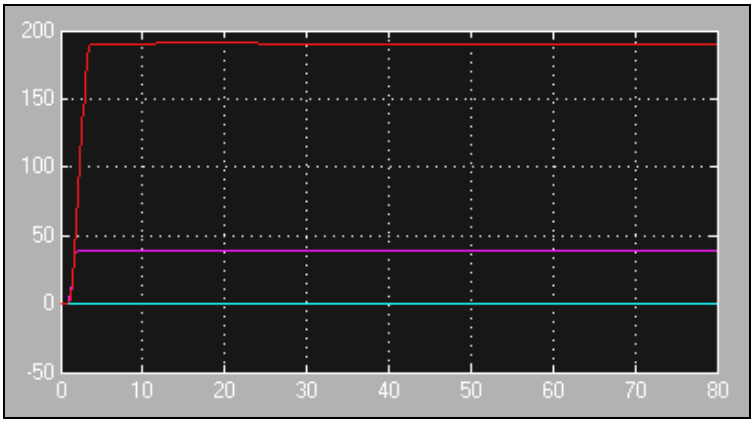

Figura 11. Gráfico do sistema em malha fechada submetido à entrada degrau.

3.4. Desenvolvendo o controlador PID-fuzzy

O controlador PID-fuzzy trabalha com o erro, com a integral do erro e com a derivada do erro. Isso quer dizer que tem as mesmas entradas de um controlador PID clássico, mas ao invés dos ganhos Kp, Ti e Td, possui essas entradas tratadas por um conjunto de funções nebulosas. As mesmas funções do controlador PI-fuzzy foram mantidas, mas também foram adicionadas novas, relacionadas à derivada do erro. As tabelas abaixo mostram as inferências adicionadas ao controlador nebuloso.

Tabela 9. Tabela de relações das entradas com a saída.

Variável	Derivada do erro						
Erro	NNN	NN	N	Z	P	PP	PPP
NNN							
NN							
N	PPP	PP	P		N	NN	NNN
Z	PPP	PP	P		N	NN	NNN
P	PPP	PP	P		N	NN	NNN
PP							
PPP							

Tabela 10. Tabela de relações das entradas com a saída.

Variável	Derivada do erro						
Int. Erro	NNN	NN	N	Z	P	PP	PPP
NNN							
NN							
N	PPP	PP	P		N	NN	NNN
Z	PPP	PP	P		N	NN	NNN
P	PPP	PP	P		N	NN	NNN
PP							
PPP							

O fator de 30% do erro foi mantido nas funções antigas, mas essas novas funções possuem todas fator de valor unitário. Vemos pelas Figuras 12 e 13 que a superfície de

controle também é complexa, mas ainda assim suave. A variável do erro do controlador está no espaço [-400, 400], enquanto q ue a integral do erro está no espaço [-50, 50] e a derivada do mesmo está em [-1000, 1000]. Note que pelo gráfico da Figura 14 abaixo, o sistema respondeu rapidamente e desta vez praticamente não apresentou erro de regime. Isso mostra que o controlador PID-fuzzy ficou melhor que o PI-fuzzy, que ficou melhor que o PID. O interessante é que realmente o controlador PID tem suas limitações, principalmente quando a entrada tem valores muito diferentes para cada situação.

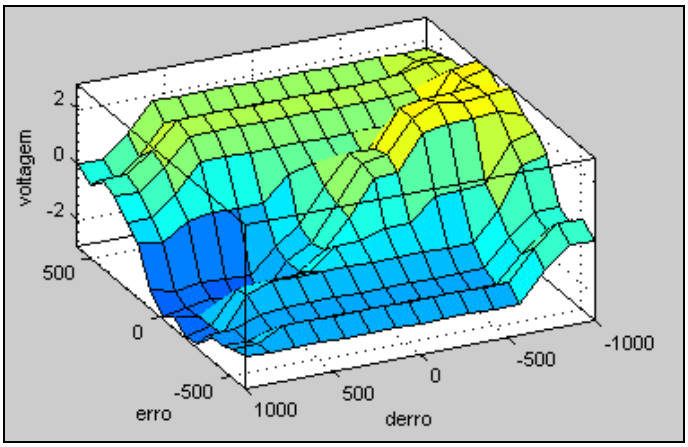

Figura 12. Gráfico da relação do erro e de sua derivada.

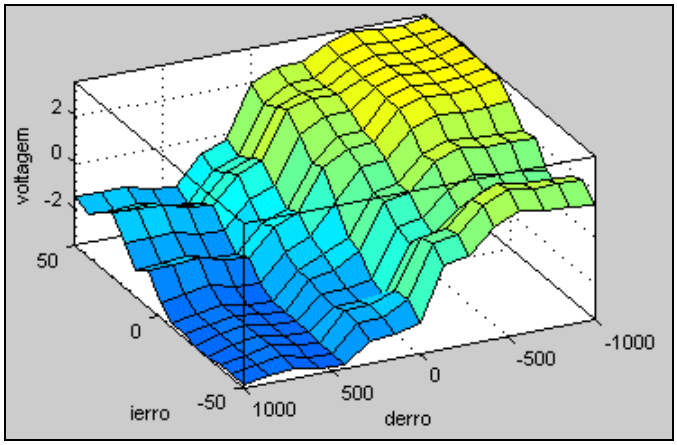

Figura 13. Gráfico da integral do erro e da derivada do mesmo.

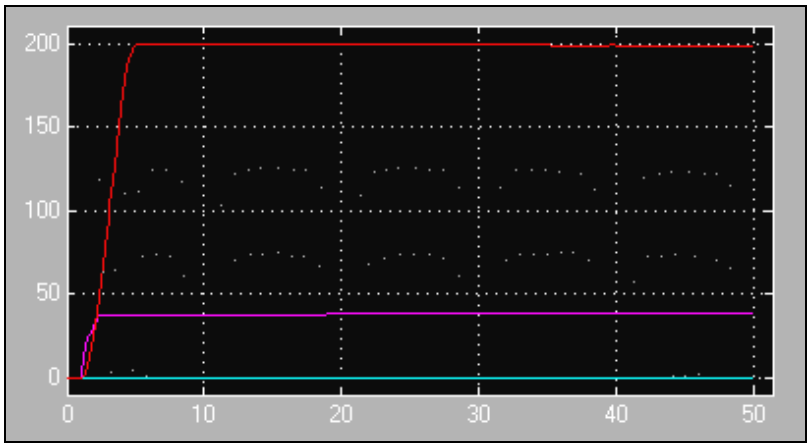

Figura 14. Gráfico do PID-fuzzy com diferentes entradas em degrau.

3.4. Controle PID ajustado pela lógica nebulosa

Uma solução inteligente para o problema do controlador PID seria ajustar seus ganhos K, Ti e Td de acordo com a entrada fornecida (Saito e Massa, 2004). Uma forma de fazer isso é utilizando um controlador nebuloso específico. A entrada seria o valor da rotação (em rad/s) e a saída seria o ganho a ser utilizado no controlador clássico. As simulações anteriores mostram que podemos melhorar muito o rendimento do controlador proporcional-integral-derivativo. Quanto maior o valor da entrada, maior deve ser o ganho derivativo ao qual a planta do sistema é submetida. Vale o mesmo para o ganho proporcional e o inverso para o ganho integral. Para esse controle nebuloso foram utilizadas três funções nebulosas de entrada e três de saída para cada saída (Alta, Média, Baixa). A entrada é a rotação da roda em módulo e varia em [0, 600], enquanto que as saídas são a constante Td que varia em [0,02 , 0,15], a constante Ti que varia em [0,02 , 0,2] e a constante K que varia em [0,5 , 0,9]. O esquema do projeto no Matlab está na Figura 15 e o gráfico da transferência nebulosa é dado na Figura 16.

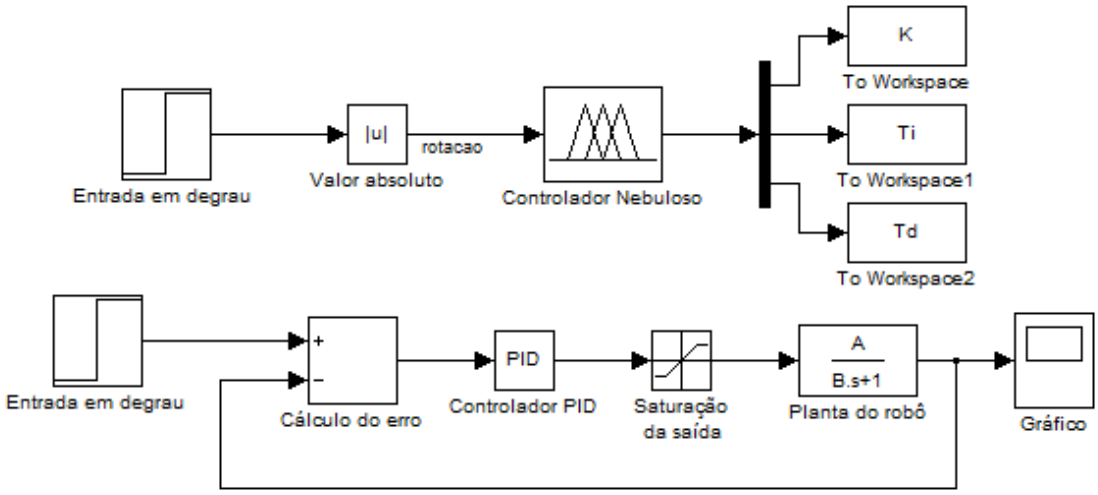

Figura 15. Projeto do controlador PID configurado por lógica nebulosa.

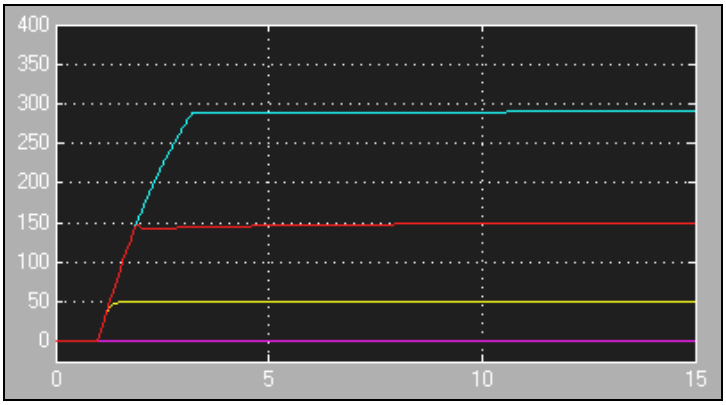

Figura 16. Gráfico do PID configurado por fuzzy para diversas entradas.

Como podemos ver no gráfico acima, o mesmo controlador PID atingiu com rapidez todos os valores requisitados, o que mostra que a lógica nebulosa foi a responsável por ajustar o mesmo, aumentando sua eficiência. Esse tipo de projeto é recomendado para situações que o controlador deva ser mais robusto que um

controlador clássico comum, mas mesmo assim relativamente simples. Este trabalho também pode ser encontrado em [Gonçalves, 2007].

4. Conclusão

Neste relatório foram apresentados inúmeros tipos de controladores desenvolvidos para o controle de vários robôs pertencentes a um time de futebol de robôs: PID, PI-fuzzy, PID-fuzzy e PID com ganhos ajustados pela lógica nebulosa. A simulação mostrou que tudo o que o controlador clássico faz é possível ser transferido para uma abordagem que utilize lógica nebulosa, que é mais robusta. Foi relatado que todo o sistema pode ser ajustado com a ferramenta de lógica nebulosa do software Matlab.

Com base nos controladores desenvolvidos, as seguintes conclusões foram estabelecidas. Em primeiro lugar, o controle PID, quando submetido a grandes variações de entrada pode se mostrar impreciso e insatisfatório, problema esse que pode ser corrigido implementando-se um controlador nebuloso para administrar os ganhos do controlador clássico. Em segundo lugar, embora a lógica nebulosa seja mais trabalhosa de se desenvolver, ela é mais intuitiva, pois trabalha com parâmetros intuitivos como o erro, a integral do erro e a derivada do erro. Este ponto só não é válido para o controle nebuloso dos ganhos do PID. Em terceiro lugar, observou-se que a lógica nebulosa precisa ser testada e otimizada antes de ser programada, pois é muito complexa e tem muitas variáveis, mas isso pode ser feito calculando-se a planta do sistema e simulando-a através do Matlab. Por fim, os controladores nebulosos tendem a ser bem mais precisos e bem mais robustos, como foi mostrado nos gráficos apresentados nesse relatório.

Agradecimentos

Os autores agradecem a FAPESP e o CNPq pelo apoio recebido durante a realização do presente trabalho.

5. Referências

Cha, P.D., Rosenberg, J. J., Dym, C. L., *"Fundamentals of Modeling and Analysing Engineering Systems"*, Cambridge University Press, 2000.

Fitzgerald, A. E. *"Máquinas elétricas"*, Ed. McGraw-Hill, 1975.

Gonçalves, Ben Hur e Romero, Roseli A. F , "*Controle de Sistemas Mecânicos por Lógica Fuzzy*", trabalho apresentado do 15º.SIICUSP, realizado em novembro de 2007, em São Carlos-SP e publicado nos Anais do 15º.SIICUSP.

Kaehler, S., D., *"Fuzzy Logic Tutorial – An Introduction"*. Publicação On-line, URL: http://www.seattlerobotics.org/encoder/mar98/fuz, 1998.

Pedrycz, W., Gomide, F., "*An Introduction to Fuzzy Sets*". The MIT Press, 1998.

Penharbel, Eder A., Gonçalves, Ben Hur e Romero, Roseli A. F. 2008**.** A Real-time Color Classifier. *10th Brazilian Workshop on Real-Time and Embedded Systems, Anais do WTR´2008, CR-ROM, realizado no período de 26 a 30 de maio de* 2008.

Saito, K. e Massa, M. C., *"Sistemas inteligentes em controle e automação de processos"*, Ed. Ciência Moderna, 2004.

Souza, D. J., "*Desbravando o PIC: ampliado e atualizado para PIC16F628A"*, Ed. Érica, 2005

Valentine, R., *"Motor Control Electronics Handbook*", Mc-Graw Hill. 1998.