



Pontifícia Universidade Católica do Rio de Janeiro

**Controle PID para Sistema Robótico Capaz de
Interagir com o Ambiente Externo Através de
Estímulos Gerados por Sensores IR**

Projeto de Graduação em Engenharia de Controle e Automação

Marcos Aurélio Pinto Marzano Júnior

2011.2

Orientador

Marco Antônio Meggiolaro

PUC-Rj - Departamento de Engenharia Mecânica

Coorientador

Alexandre Ormiga Galvão Barbosa

Dedicatória

Dedico este trabalho aos meus pais os quais eu amo muito, por todo apoio nos momentos mais difíceis da minha vida, pelo exemplo de vida e família, além de acreditarem em todo meu potencial. A estas duas pessoas o meu muito obrigado.

Agradecimentos

Aos professores Marco Antônio Meggiolaro e Mauro Schwanke da Pontifícia Universidade Católica do Rio de Janeiro, por despertar a paixão por robótica em seus alunos. Gostaria de agradecer também todas as pessoas que contribuíram de alguma forma para este projeto entre eles: Guilherme Mourão, Marcos Soares, Pedro Blois, João Luiz Souza, Paula Migueles Berwanger e Michel Feinstein.

“O sucesso é ir de fracasso em fracasso sem perder o entusiasmo”.

Winston Churchill

Sumário

<u>INTRODUÇÃO</u>	<u>7</u>
ROBÔS AUTÔNOMOS	8
OBJETIVO	9
<u>DUELO E REGRAS DA CATEGORIA DOS ROBÔS SUMÔ</u>	<u>11</u>
<u>PLATAFORMA ATUAL</u>	<u>12</u>
MECÂNICA E LOCOMOÇÃO	12
SENSORES DO ROBÔ	14
<i>SOFTWARE</i>	17
<u>PLATAFORMA DESENVOLVIDA</u>	<u>20</u>
SENSORES DO ROBÔ	20
MECÂNICA E LOCOMOÇÃO	28
ELETRÔNICA	34
<i>SOFTWARE</i>	48
<u>RESULTADOS</u>	<u>56</u>
<u>CONCLUSÃO</u>	<u>56</u>
<u>BIBLIOGRAFIA</u>	<u>57</u>

Lista de Figuras

Figura 1- Arena da competição de sumô	11
Figura 2 - Sumô Boi da Cara Preta.....	12
Figura 3 - Conjunto Caixa de Redução Banebots 16:1 e motor Mabuchi RS-550....	13
Figura 4 - Motor Integy	13
Figura 5 - Disposição dos Sensores (Barbosa)	14
Figura 6 - Sensor SRF10	14
Figura 7 - Início Protocolo I2C	15
Figura 8 - Término Protocolo I2C	16
Figura 9 - Funcionamento Global Protocolo I2C	16
Figura 10 - Sensor Infravermelho QRD1114	16
Figura 11 - Sistema <i>Fuzzy</i> (http://www.ica.ele.puc-rio.br)	18
Figura 12 - Técnicas de Defuzzificação	20
Figura 13 - Sensor Sharp Modelo GP2D12.....	21
Figura 14 - Base de Teste para os Sensores.....	22
Figura 15 - Gráfico Voltagem de Saída x Distância do Objeto do Datasheet.....	23
Figura 16- Gráfico Voltagem de saída x Distância do Objeto Feito no Excel.....	24
Figura 17 - Gráfico Valor Decimal Convertido x Distância do Objeto.....	26
Figura 18 - Equação Aproximada de 10 a 30 cm Valor Decimal Convertido x Distância Objeto	26
Figura 19 - Equação Aproximada de 35 a 80 cm Valor Decimal Convertido x Distância Objeto	26
Figura 20 - Trecho do Código Realizado para Calcular a Distância de um único Sensor	27
Figura 21 - Protótipo C3 para Validação de Conceito.....	28
Figura 22 - Vista Interna do Protótipo.....	29
Figura 23 - Vistas Laterais do Protótipo.....	29
Figura 24 - Ensaio de Tração do Imã na Instron 8502	30
Figura 25 - Bancada de Ensaio	30
Figura 26- Variação da Força do Imã em Relação ao Afastamento do Suporte de Aço	30
Figura 27 – Variação da Força do Imã pelo o Tempo de Ensaio	31
Figura 28 – Simulação de Tensão na Estrutura do Robô	31
Figura 29 – Simulação de Tensão na Estrutura do Robô (Visão Parcial).	31

Figura 30 – Simulação da Deformação Estrutural do Robô.....	32
Figura 31 - Motor Dewalt 18V New Style	32
Figura 32 - Características do Motor Dewalt 18V Modelo New Style (Meggiolaro, 2006).....	33
Figura 33 - Microcontrolador Pic16f877A.....	36
Figura 34 - Sinal PWM (Meggiolaro, 2006)	38
Figura 35 - Controlador de Velocidade da Banebot.....	38
Figura 36 - Esquemático Circuito 1.....	39
Figura 37 - Esquemático Recomendável do Circuito 1	40
Figura 38 - Esquemático Circuito de Comunicação Serial (Messias).....	41
Figura 39 - Esquemático para Cada Sensor de Linha.....	42
Figura 40 - Esquemático da Ligação no Hardware dos Sensores de Linha	43
Figura 41 - Diagrama de Simulação da Eletrônica no Software ISIS	45
Figura 42 - Layout do Desenvolvimento do Hardware no Software Ares	46
Figura 43 - Vista Frontal 3D do Hardware.....	46
Figura 44 - Vista Posterior 3D do Hardware.....	47
Figura 45 - Visto Superior 3D do Hardware.....	47
Figura 46 - Layout do Hardware	47
Figura 47 - Cabeçalho do Programa.....	50
Figura 48 - Função do PWM dos Controladores de Velocidade dos Motores.....	51
Figura 49 - Funcionamento do PWM no Controlador de Velocidade.....	51
Figura 50 - Representação do Período do Sinal do Controlador de Velocidade.....	51
Figura 51 - Representação da Variável de Processo.....	53

Lista de Tabelas

Tabela 1 - Tabela de Medalhas dos Robôs Autônomos da Equipe Riobotz.....	10
Tabela 2 - Tabela Estimada do Gráfico Voltagem de Saída x Distância do Objeto.....	24
Tabela 3 - Tabela Estimada do Valor Decimal Convertido x Distância do Objeto.....	25
Tabela 4 - Numeração Atribuída aos Sensores.....	52
Tabela 5 - Tabela de Erros PID	54

Lista de Equações

Equação 1 - Distância do Sensor SHARP entre 10 cm e 35 cm.....	27
Equação 2 – Distância do Sensor SHARP entre 35 cm e 80 cm.....	27
Equação 3 - Potência Dissipada no Caso de um Único Regulador	40
Equação 4 - Cálculo do Resistor do Led Infravermelho do Sensor de Linha	42
Equação 5 - Cálculo do Resistor para Todos os Emissores dos Sensores de Linha.....	43
Equação 6 - Cálculo da Potência do Resistor dos Emissores do Sensor de Linha.....	43
Equação 7 - Equação Contínua do PID	54
Equação 8 - Discretização da Variável Tempo	54
Equação 9 - Equação Discretizada do PID.....	55
Equação 10 - Equação Digital do PID.....	55
Equação 11 - Simplificação da Equação a ser Implementada no Microcontrolador.....	55

Introdução

É notório o avanço da tecnologia de um modo geral na área da ciência nas últimas décadas. Diversas tendências com o passar dos anos estão cada vez mais solidificadas em setores tais como: médico, industrial, aeroespacial, militar, dentre outros. Uma das áreas que receberam um maior enfoque é a robótica de um modo geral e tal motivo deve-se ao fato das diversas aplicações e benefícios que a tecnologia deste setor pode fornecer aos seus usuários.

No âmbito do ambiente de trabalho, as pessoas que dependem de um bom monitoramento e vigilância sofrem muito com as técnicas empregadas hoje em dia, pois além do difícil acesso humano em alguns lugares, existem tarefas bastante nocivas ao ser humano tais como: monitoramento e manutenção de usinas nucleares e termoelétricas, grandes extensões de terra como florestas e matas, ambientes extremos submersos e aéreos, dentre outros. Portanto a robótica é considerada atualmente uma excelente alternativa para este setor, auxiliando o desenvolvimento de projetos mecânicos e eletrônicos voltados para área de inspeção não destrutiva.

Além disso, a robótica juntamente com a automação influenciou o avanço industrial aumentando tanto a capacidade de produção das grandes indústrias como a flexibilidade da fabricação de diversos produtos. À medida que estes sistemas foram se sofisticando, o tamanho do conjunto destas aplicações também foi evoluindo. Antigamente um sistema que funcionava simplesmente com o intuito de transporte de peças, hoje participa efetivamente das operações de montagem dos produtos e do processo como um todo. Logo é possível concluir que esta área de estudo é de suma importância para o desenvolvimento humano, uma vez que este setor possui uma enorme quantidade de aplicações e benefícios para a sociedade.

Robôs Autônomos

O termo "Robô" vem da palavra checa "robota", que significa "trabalho forçado". Dentre as idéias mais antigas que se conhecem sobre dispositivos automáticos, ou autômatos, data de 350 A.C., criada pelo matemático grego Arquitas de Tarento, amigo de Platão. Ele criou um pássaro de madeira que batizou de "O Pombo". O pássaro era propulsionado por vapor e jatos de ar comprimido tendo, para muitos, mais méritos de ter sido a primeira máquina a vapor do que a inventada por James Watt.

Ignorando a definição oficial da RIA (Robotics Industries Association), um **robô** seria um dispositivo automático que possui conexões de realimentação (*feedback*) entre seus sensores, atuadores e o ambiente, dispensando a ação do controle humano direto para realizar determinadas tarefas, podendo também haver robôs parcialmente ou totalmente controlados por pessoas. O grau de automatização de um robô pode atingir o nível de aprendizado automático, dependendo dos algoritmos utilizados. Entretanto há ainda muitas limitações, devido às óbvias dificuldades de simular a realidade em nível computacional.

Os robôs executam tarefas através de atuadores (elétricos, pneumáticos, etc.), produzindo sons, acendendo elementos luminosos ou displays, movendo um braço, abrindo ou fechando uma garra robótica, realizando o seu próprio deslocamento, dentre outras. Este controle é provido por algoritmos que relacionam as entradas e saídas do robô, através de unidades de processamento e *softwares*, que podem ser desde um circuito eletrônico de controle até mesmo um computador pessoal.

Robôs autônomos são robôs que podem realizar os objetivos desejados em ambientes desestruturados sem a ajuda humana, através de uma autonomia para tomada de decisões. A principal diferença entre um simples robô e um robô autônomo é justamente o fato de que um simples robô executa determinadas tarefas que foram programadas através de um algoritmo pré-estabelecido e estes não podem interferir no processo. Entretanto os robôs autônomos possuem um objetivo que para este ser alcançado o robô deve tomar decisões que optem pelo o melhor caminho.

Há diferentes tipos de níveis e formas de autonomia para um robô. Um dos mais altos níveis de autonomia é particularmente desejado em campos como a exploração espacial, onde a comunicação possui atrasos e as interrupções são inevitáveis. Alguns robôs de fábrica modernos são "autônomos" com as limitações de seu ambiente normal.

Talvez não existam todos os níveis de liberdade no ambiente ao seu redor, mas o ambiente de trabalho de uma fábrica é complexo e pode ser imprevisível ou até mesmo caótico. A orientação e posição exata do próximo objeto a ser trabalhado e até mesmo o tipo do objeto e o trabalho requerido devem ser determinados, podendo variar de uma forma não previsível (ao menos no ponto de vista de um robô).

Uma área importante da pesquisa em robótica é permitir ao robô cooperar com o seu ambiente independente se este ambiente é terra, água, ar, cavernas ou até mesmo o espaço. Um robô totalmente autônomo no mundo real tem a habilidade de:

- Receber informações do seu ambiente.
- Trabalhar por um período sem nenhuma interferência humana.
- Se deslocar do ponto A ao ponto B, sem assistência de navegação humana.
- Evitar situações que são perigosas para as pessoas.
- Reparar-se sem ajuda externa. Entretanto, grande parte dos robôs autônomos ainda requer manutenção regular, assim como outras máquinas.

Um robô também pode ser capaz de aprender autonomamente. O aprendizado autônomo inclui a habilidade de:

- Aprender ou ganhar novas capacidades sem assistência externa.
- Ajustar suas estratégias baseados nos arredores.

Objetivo

Há uma crescente utilização de robôs em diversas aplicações como já mencionadas anteriormente, principalmente na exploração de locais onde há risco de vida. Tal motivo faz com que muitas pesquisas e competições surjam, visando o intuito de incentivar estudantes de engenharia e áreas afins a desenvolver tecnologia e conhecimento sobre o tema.

Portanto, o objetivo deste projeto é o desenvolvimento de uma nova plataforma robótica que seja capaz de interagir com o ambiente externo, identificando objetos através de estímulos gerados por sensores de distância e tomando decisões baseadas no controle pré-programado de um microcontrolador. Além disso, esta plataforma deverá representar a instituição de ensino PUC – Rio em diversas competições em que possa ser utilizada esta tecnologia.

Atualmente a equipe da PUC possui 5 robôs autônomos de competição que podem ser resumidos através da seguinte tabela:

Nome	Categoria	Total de Medalhas
Pé de Boi	Sumô 3kg	4
Boi da Cara Preta	Sumô 3 Kg	4
Mini Touro+	Combate 1,3 kG	2
Bezerro	Sumô 500grs	0
Papacuras	Seguidor de Linha	0

Tabela 1 - Tabela de Medalhas dos Robôs Autônomos da Equipe Riobotz

Os dois robôs da categoria sumô 3 kg podem ser considerados robôs de extremo sucesso, pois além da quantidade de medalhas conquistadas estas plataformas conseguiram obter resultados de grande importância, como por exemplo: o 1º e 2º lugar nas Olimpíadas Mundiais de Robótica – Robogames realizada em San Francisco no ano de 2009.

Entretanto com o passar dos anos a competição foi evoluindo, os adversários foram aperfeiçoando-se e nenhuma modificação significativa nestes robôs foi realizada. O resultado deste fato é que atualmente os dois robôs continuam conquistando medalhas, entretanto não conseguem mais alcançar o primeiro lugar absoluto em todas as competições que participam. Portanto a plataforma a ser desenvolvida deverá atender todas as regras desta categoria em competições, além disso, com o intuito de poupar recursos financeiros, as seguintes regras devem ser levadas em consideração neste projeto: a americana, a japonesa e a nacional.

É importante ressaltar que esta tecnologia pode ser aplicada em diversas atividades e não somente a competições de robótica. Uma simples mudança no código para ao invés do robô ir para cima de um objeto identificado pelo os sensores ele simplesmente se afastar dele, pode-se elaborar um sistema que transporta alguma determinada peça dentro de um ambiente fabril e consiga desviar de pessoas, paredes, obstáculos, dentre outros. Além disso, existem diversas aplicações desta tecnologia no ambiente doméstico como, por exemplo, um robô que aspire a sujeira do chão de uma casa desviando de paredes, móveis, animais domésticos e pessoas.

Duelo e Regras da Categoria dos Robôs Sumô

O sumô de robôs é uma das modalidades de competição de robôs mais difundidas no mundo. Da mesma forma que no tradicional combate humano, o objetivo é empurrar o robô oponente para fora de uma arena. Existem duas categorias nesta modalidade, uma autônoma e outra rádiocontrolada. A arena é constituída de uma chapa de aço de 5 milímetros e possui a forma de um círculo de 1,5 m de diâmetro. Além disso, toda arena é pintada da cor preta e possui uma faixa branca de 5 cm de largura ao longo da circunferência para que o robô reconheça o limite da mesma. O tamanho e o peso dos robôs são restritos por categoria, sendo a dimensão máxima permitida de 20 cm X 20 cm com altura ilimitada para a categoria sumô de 3 Kg.



Figura 1- Arena da competição de sumô

A partida é disputada por duas equipes, cada uma composta por um ou mais membros. Apenas um membro de cada equipe pode ficar na área do ringue nas regras nacionais e americana, enquanto que a japonesa permite dois membros de cada equipe, os demais membros deverão assistir à partida junto com o público. Cada equipe compete no dojô (arena do sumô) com um robô construído nas especificações já mencionadas. A partida deve ser iniciada pelo o juiz e deve continuar até que o primeiro competidor conquiste duas vitórias. Uma partida é constituída de 3 rounds com um total de tempo de 3 minutos, a menos quando estendidas pelo o juiz. É importante ressaltar que a regra da categoria autônoma é idêntica a da categoria rádio controlada, entretanto na autônoma o robô não poderá receber auxílio humano em suas tomadas de decisões. Todas as regras desta competição podem ser adquiridas nos seguintes *sites*: www.robocore.net, www.robogames.net, www.fsi.co.jp/Sumo.

Plataforma Atual

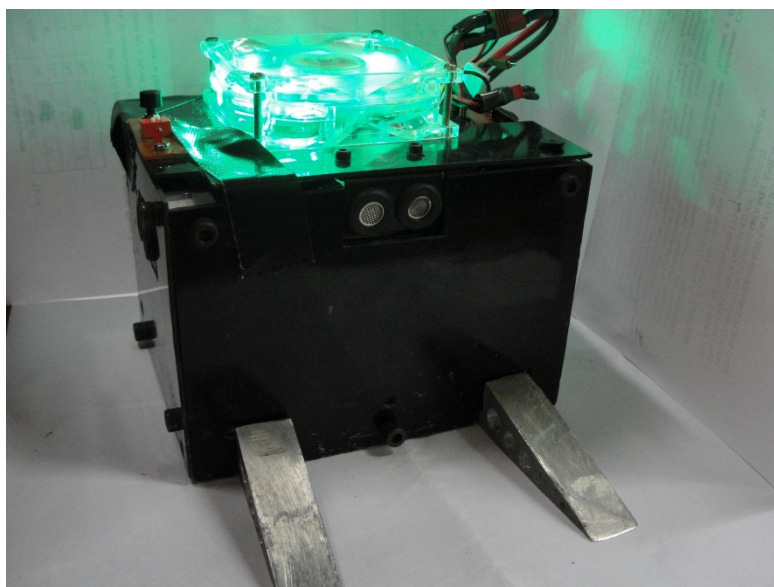


Figura 2 - Sumô Boi da Cara Preta

O sistema de controle do robô Pé-de-Boi é o mesmo sistema do robô Boi da Cara Preta, ou seja, todos os *hardwares* de inteligência, potência, sensores e *software* de controle são os mesmos. Entretanto o sistema de locomoção que também era idêntico no início de ambos os projetos, com o passar dos anos sofreu uma pequena alteração sendo esta uma das únicas modificações realizadas nesta plataforma desde sua concepção. Abaixo segue uma lista detalhada dos componentes já utilizados e os atuais por ambas as plataformas.

Mecânica e locomoção

O projeto de locomoção inicial ainda é o mesmo sendo este realizado por esteiras, entretanto ocorreram mudanças nos motores e caixas de redução. A escolha por esteiras deve-se ao fato de que como o objetivo principal do robô é empurrar o adversário para fora da arena e para isto este deve possuir a maior tração possível, acreditou-se que a esteira era a opção mais apropriada por possuir uma área de contato com o chão da

arena maior que as superfícies de rodas, aumentando assim o atrito e conseqüentemente a tração já que esta depende somente da normal do robô e do atrito com o chão da arena.

O sistema responsável por tracionar a esteira era o conjunto da caixa de redução Banebots modelo com redução 16:1 acoplada ao motor Mabuchi RS-550 que já vinha no próprio conjunto. Posteriormente descobriu-se a existência do motor Integy 65T, utilizado em um torno para usinar eixos de automodelos (carrinhos de controle remoto). É importante ressaltar que o número 65T determina o modelo do motor e a quantidade de espiras do mesmo. Um motor com um número maior de espiras possui maior torque, sendo o torque proporcional à corrente e a velocidade proporcional à tensão.

Este motor possui maior torque, melhor qualidade de fabricação e as mesmas dimensões do antigo Mabuchi RS-550. Logo visando aumentar o desempenho da locomoção de ambos os robôs, a caixa de redução Banebots 16:1 e o motor Integy 65T acabaram sendo adotados como o novo conjunto de locomoção de ambos os robôs.



Figura 3 - Conjunto Caixa de Redução Banebots 16:1 e motor Mabuchi RS-550



Figura 4 - Motor Integy

Além disso, a última modificação realizada na locomoção destes dois robôs foi à troca somente das caixas de redução do robô Boi de Cara Preta. Atualmente este robô utiliza uma caixa do mesmo fabricante (Banebots) modelo 25:1 no lugar das antigas caixas com relação 16:1. Esta ação foi devida à inserção de um novo adversário na competição, onde o conjunto antigo não tinha torque suficiente para empurrá-lo. É importante ressaltar que não foi realizada a troca em ambos os robôs por falta de caixas de redução disponíveis no momento.

Sensores do Robô

Abaixo segue uma ilustração com a disposição dos sensores:

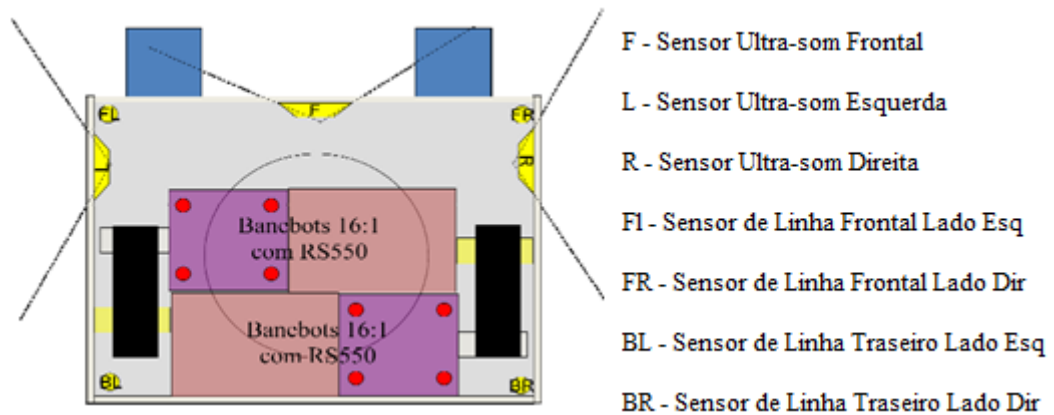


Figura 5 - Disposição dos Sensores (Barbosa)

As plataformas atuais utilizam dois tipos básicos de sensores:

1. Sensores de ultra-som → são sensores modelo SRF10 do fabricante “Devantech” utilizados para realizar a busca do oponente, ou seja, estes sensores emitem ondas sonoras com o objetivo de detectar objetos. A escolha inicial deste sensor foi graças há um estudo realizado sobre os outros robôs da categoria, onde foi visto que a maioria dos robôs é da cor preta, o que dificulta ou até mesmo impossibilita a detecção dos mesmos com sensores infravermelhos convencionais. Além disso, este sensor possui uma ampla abertura do seu ângulo de varredura, diminuindo assim a quantidade de sensores que devem ser utilizados.
- 2.

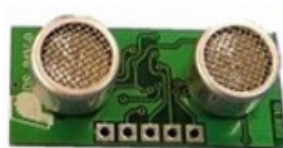


Figura 6 - Sensor SRF10

Os sensores ‘L’, ‘F’ e ‘R’ são sensores de distância ultra-som com ângulo de varredura de 120°, ou seja, este ângulo é suficiente para que com esta distribuição somente três sensores cubram toda a frente e a lateral do robô sem “pontos cegos”. Entretanto esta informação do ângulo de varredura fornecido pelo fabricante é questionável, pois os robôs atuais apresentam pontos de indecisão decorrentes há possíveis “pontos cegos” que não eram para existir. Logo um estudo da disposição

dos sensores da nova plataforma empregada deverá ser realizado antes da concepção mecânica do robô para que não haja problemas parecidos como os descritos. Além disso, estes sensores ultra-som permitem a utilização do protocolo I2C facilitando assim a elaboração da eletrônica uma vez que só é necessário apenas um barramento de sinal para todos os sensores. Este protocolo de comunicação foi originalmente desenvolvido pela Philips em meados de 1996. Atualmente este protocolo está amplamente difundido na indústria e interconecta uma ampla gama de dispositivos eletrônicos. O procedimento de comunicação do protocolo I2C é extremamente simples e pode ser analisado pelos seguintes itens:

1. O dispositivo MASTER envia para o barramento I2C um sinal de START. Com isso o dispositivo MASTER tem a atenção de todos os dispositivos conectados ao barramento.



Figura 7 - Início Protocolo I2C

2. O dispositivo MASTER envia um registro com o endereço que deseja acessar (7bits). Todos os dispositivos irão receber este registro, aqueles que não tiverem o endereço requisitado irão ignorar o registro e aguardar o sinal de STOP.
3. O dispositivo MASTER envia o 8º bit especificando se é para escrever ou ler.
4. Aquele que tiver o endereço enviado irá responder com um sinal de ACKNOWLEDGE.
5. Assim que o MASTER receber o sinal de ACKNOWLEDGE, poderá iniciar a transmissão ou requisição dos dados (pacotes de 8 bits) sempre seguidos de um sinal ACK enviado pelo o dispositivo confirmando a recepção.
6. Assim que a transferência terminar, o MASTER irá enviar um sinal de STOP, que liberará o barramento para que outros dispositivos possam então atuar como MASTER.



Figura 8 - Término Protocolo I2C

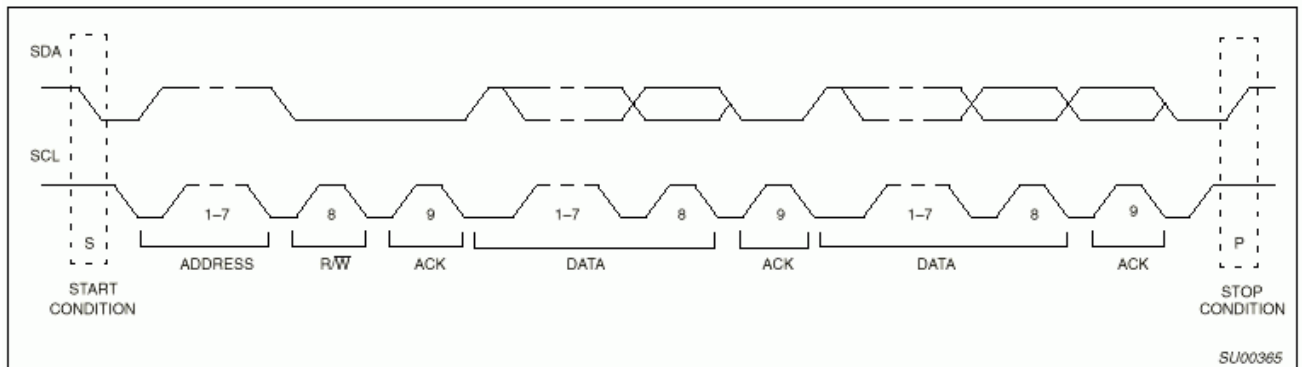


Figura 9 - Funcionamento Global Protocolo I2C

Sensores de infravermelho → são sensores do modelo QRD1114 que já possuem em um único encapsulamento o emissor e receptor infravermelho. Estes sensores são utilizados no fundo do robô para detectar a borda da arena, ou seja, a linha branca ao redor do dojô. Assim, quando a superfície abaixo destes sensores for branca, os raios do infravermelho irão refletir e serão absorvidos pelo receptor, ao contrário de uma superfície preta que absorverá a emissão dos raios e não retornará nada para o receptor. É importante ressaltar que a escolha deste sensor deve-se ao fato que além de apenas em um único encapsulamento já conter o emissor e o receptor, este modelo já possui um filtro infravermelho para o receptor.

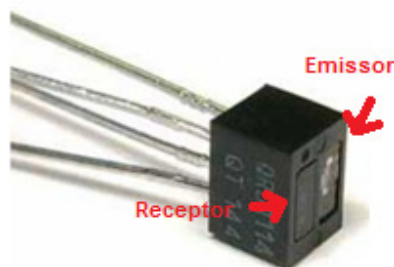


Figura 10 - Sensor Infravermelho QRD1114

Software

A lógica dos dois robôs é a mesma e foi escrita na linguagem C de programação com inspiração na técnica de controle *Fuzzy*. Esta técnica procura modelar o modo impreciso do raciocínio humano, proporcionando assim uma enorme habilidade de tomadas de decisões. Esta ferramenta serve de base para o raciocínio aproximado (*approximate reasoning*), além de dispor de todo o ferramental matemático para o tratamento de caráter vago ou impreciso.

Como exemplo, pode-se citar a seguinte proposição: Marcos é alto. Neste caso a proposição será verdadeira para uma altura de 1,68 m? Observe que ser alto é totalmente relativo, ou seja, comparando com uma altura de 1,60m Marcos é alto, mas comparando com uma altura maior que 1,68m Marcos se torna baixo. Portanto o termo lingüístico “alto” é vago.

Outro exemplo pode ser demonstrado da seguinte forma:

Proposição A → Toda pessoa deve começar a frear 10m antes do sinal vermelho.

Proposição B → Toda pessoa deve começar a frear perto da faixa de pedestres.

No caso A está explicitamente restrito o momento em que a pessoa deve frear o veículo, enquanto no caso B ocorre novamente uma informação vaga ou imprecisa, pois o termo lingüístico “perto” pode variar de opinião de uma pessoa para outra.

A lógica *Fuzzy* baseia-se na teoria dos conjuntos *Fuzzy*. Tradicionalmente uma proposição lógica possui 2 extremos: “completamente verdadeiro” ou “completamente falso”. Entretanto na lógica *Fuzzy* uma premissa varia com grau de verdade de 0 a 1, o que leva a ser parcialmente verdadeira ou parcialmente falsa. Para uma maior compreensão da lógica *Fuzzy* vamos analisar a seguinte situação abaixo:

A lógica convencional define a seguinte regra:

Pessoas jovens são aquelas cujas idades estão entre 0 e 20 anos.

Nesta lógica uma pessoa com 20 anos e 1 dia não é mais considerada jovem apesar de sabermos que isto não é verdade no mundo real. Daí a necessidade de ser

utilizada a lógica *Fuzzy* para descrever o grau de pertinência de uma pessoa no conjunto de jovens. Portanto utilizando a lógica *Fuzzy* este problema seria descrito da seguinte forma:

Bruno tem 20 anos → Bruno é 1 jovem e 0 adulto

Michel tem 20 anos e 1 semana → Michel é 0,9 jovem e 0,1 adulto, por exemplo,

Raquel tem 24 anos → Raquel é 0,7 jovem e 0,3 adulto, por exemplo,

E assim sucessivamente de acordo com o grau de pertinência escolhido.

O controle executado por *Fuzzy* imita um comportamento baseado em regras ao invés de um controle explicitamente restrito a modelos matemáticos como, por exemplo: as equações diferenciais. O objetivo da lógica *Fuzzy* é gerar uma saída lógica a partir de um conjunto de entradas não precisas, com ruídos ou até mesmo itens faltantes. A lógica *Fuzzy* apresenta as seguintes características em relação às outras formas de controle:

- Robusta, pois não requer entradas precisas.
- Modificada facilmente, pois é baseada em regras.
- Controle de sistemas não lineares sem modelos matemáticos.
- Solução mais rápida e barata em alguns casos.
- Implementável facilmente em microprocessadores.

O raciocínio com esta lógica consiste em realizar as seguintes etapas abaixo:



Figura 11 - Sistema *Fuzzy* (<http://www.ica.ele.puc-rio.br>)

Fuzzificação

Nesta etapa definem-se as variáveis lingüísticas de forma subjetiva bem como as funções de pertinência.

- Análise do problema
- Definições das variáveis *fuzzy*
- Definições das funções de pertinência
- Criação de regiões

Os conjuntos *fuzzy* constituem uma "ponte" no caminho de aproximar o raciocínio humano ao da lógica executada pela máquina. Matematicamente são funções que atribuem graus de pertinência dos valores de entrada aquele conjunto. Como no raciocínio humano, tais conceitos e definições dos conjuntos se sobrepõem, de modo que um valor não precisa deixar de pertencer a um conjunto para pertencer outro.

Assim, um conjunto definido como, por exemplo, “muito grande” receberá um grau de pertinência de acordo com a entrada, bem como o conjunto “grande” terá também o seu grau definido, podendo ambos os seres diferentes de zero. Este processo de atribuição de graus de pertinência é chamado fuzzificação.

Inferência

Nesta etapa definem-se as regras ou proposições.

- Definição das regras
- Criação da matriz de regras

A inferência é o processo pelo qual o controlador *Fuzzy* combina os sinais de entrada (já fuzzificados) com a base de regras definida para a obtenção dos conjuntos *Fuzzy* de saída.

Esta inferência pode ser realizada por diferentes métodos. Um dos métodos é o “Máximo dos Mínimos”. Nesse método, todas as regras recebem os graus de pertinência relevantes aquela regra e é calculado o mínimo das pertinências, atribuindo assim, um grau de pertinência desta regra. Uma vez feito isso, a regra com maior pertinência é selecionada a partir de uma função de máximo e será incluída no processo de defuzzificação.

Defuzzificação

Esta etapa visa converter as variáveis *Fuzzy* em valores numéricos ou aceitáveis pelo sistema. Nesta etapa diversas técnicas de defuzzificação podem ser usadas são elas:

- Centróide
- *First of Maxima*
- *Middle of Maxima*
- Critério Máximo

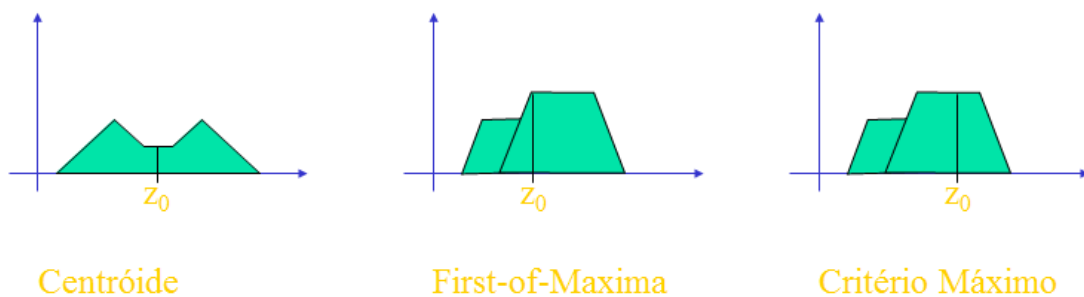


Figura 12 - Técnicas de Defuzzificação

Todo o código atual encontra-se em anexo no apêndice deste trabalho.

Plataforma Desenvolvida

Sensores do Robô

O primeiro passo é a determinação dos sensores que serão utilizados e o arranjo da disposição dos mesmos no robô. Optou-se primeiramente pela a escolha de somente a utilização de um tipo básico de tecnologia de sensores. Logo foi realizada a escolha pela a utilização de sensores infravermelhos pelo o rápido tempo de resposta deste tipo de sensor.

Infravermelho para o Oponente → O modelo escolhido foi o GP2D12 do fabricante SHARP, pois estes sensores levam em consideração para o cálculo da distância o ângulo de incidência dos raios retornados do objeto a ser observado, em vez da intensidade dos mesmos, possuindo assim uma sensibilidade e confiança muito superior aos sensores convencionais, ou seja, ele é menos sujeito a ruído externo como: a luz

ambiente. Além disso, o tempo de resposta deste sensor é menor em relação ao tempo de resposta do ultra-som, portanto este sensor possibilita um menor tempo de controle e conseqüentemente uma resposta mais rápida dos comandos na hora da partida. Outro aspecto importante é que devido à forma de calcular a distância do objeto, este sensor permite não haver preocupações antes existentes com as superfícies pretas dos oponentes. Um ponto negativo deste sensor é sua abertura de varredura que é bem inferior que a do ultra-som, sendo esta varredura quase que pontual. Contudo, estes sensores possuem um valor bem inferior aos sônicos, podendo ser utilizado em uma maior quantidade no projeto.

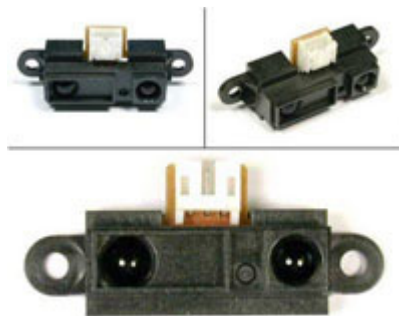


Figura 13 - Sensor Sharp Modelo GP2D12

Infravermelho para a Borda da Arena → Os sensores escolhidos para esta função foram os mesmos sensores da plataforma antiga (QRD114), devido suas qualidades já mencionadas e seu alto desempenho já comprovado.

Testes do Arranjo dos Sensores

Uma base para os sensores foi montada de tal forma que fosse fácil o ajuste e o reposicionamento dos mesmos até que a melhor configuração fosse encontrada. Logo foi realizada a escolha por uma caixa de papelão como base e uma fita dupla face para realizar a fixação dos sensores. Um programa na linguagem C de programação foi desenvolvido de tal forma que recebesse como variáveis de entrada os sinais de cada sensor, realizasse a conversão analógica/digital e armazenasse o valor da conversão em uma variável que é exibida para o usuário na tela do computador como a distância do sensor em relação ao objeto.

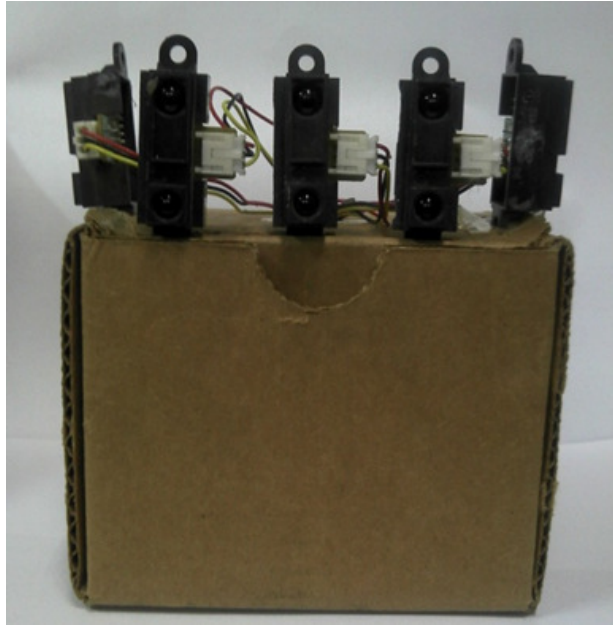


Figura 14 - Base de Teste para os Sensores

Para realizar a conversão do sinal analógico fornecido pelo o sensor infravermelho, deve-se primeiramente olhar o gráfico da voltagem de saída no pino do sensor pela a distância do objeto, sendo este gráfico fornecido pelo fabricante na folha de características do próprio sensor. Este gráfico representa a função que estabelece a relação de uma dada tensão que o sensor coloca no pino do sinal de saída, em relação à distância do objeto com o próprio sensor. Esta função pode variar de acordo com a superfície do objeto a ser medido, uma vez que a reflexão dos raios luminosos do emissor varia com a superfície de trabalho. Os pontos do gráfico foram estimados para a criação de uma tabela no *software* Excel, possibilitando assim a criação do gráfico do sensor e a determinação da equação do mesmo através deste programa. Abaixo segue o gráfico disponibilizado pelo o fabricante do sensor SHARP GP2D12:

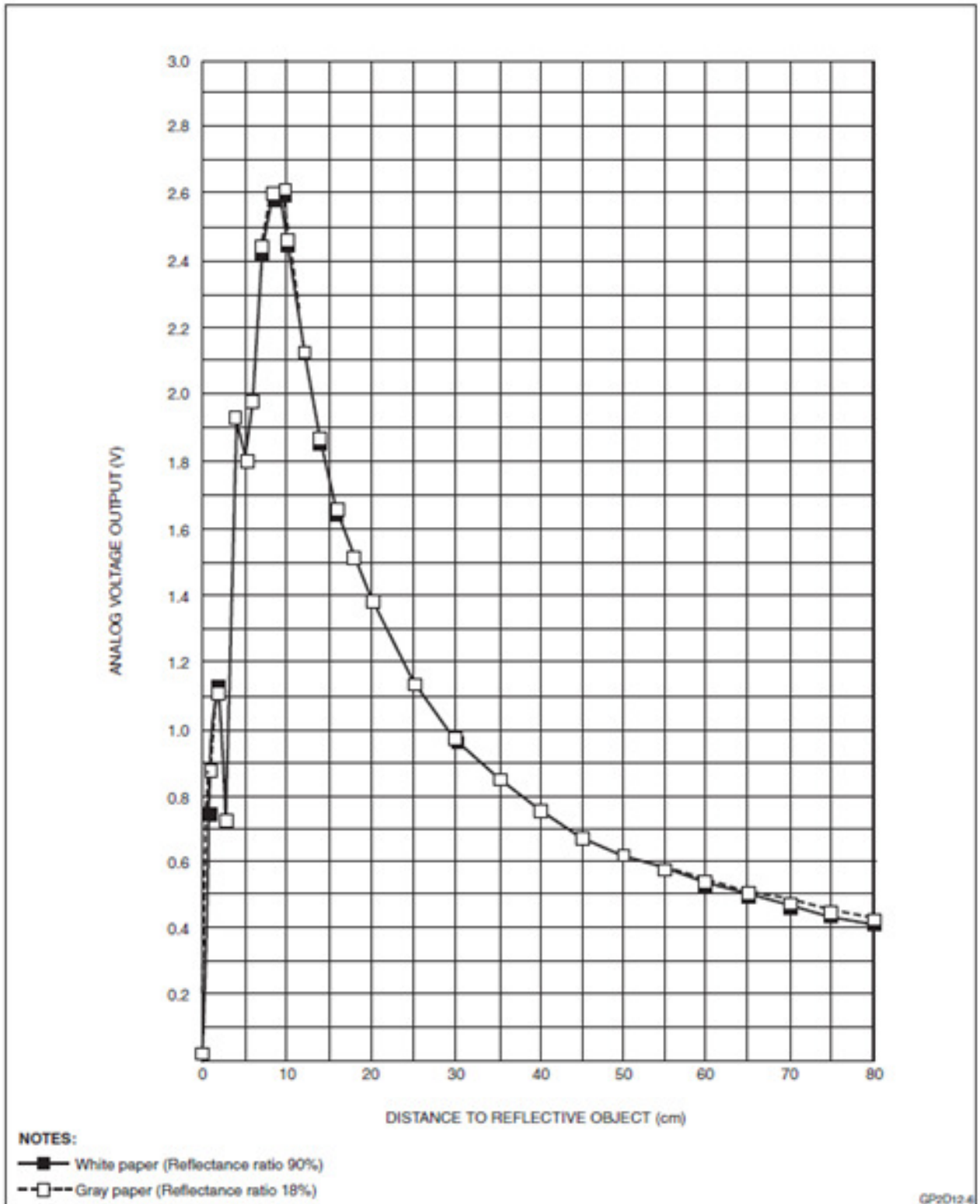


Figura 15 - Gráfico Voltagem de Saída x Distância do Objeto do Datasheet

Distancia (cm)	Voltagem (V)
0	0
10	2.6
10.5	2.45
12	2.11
14	1.84
16	1.66
17.5	1.51
20	1.39
25	1.13
30	0.98
35	0.85
40	0.75
45	0.68
50	0.61
55	0.59
60	0.56
65	0.5
70	0.47
75	0.45
80	0.4

Tabela 2 - Tabela Estimada do Gráfico Voltagem de Saída x Distância do Objeto

Abaixo o gráfico realizado pelo Excel através dos valores estimados:

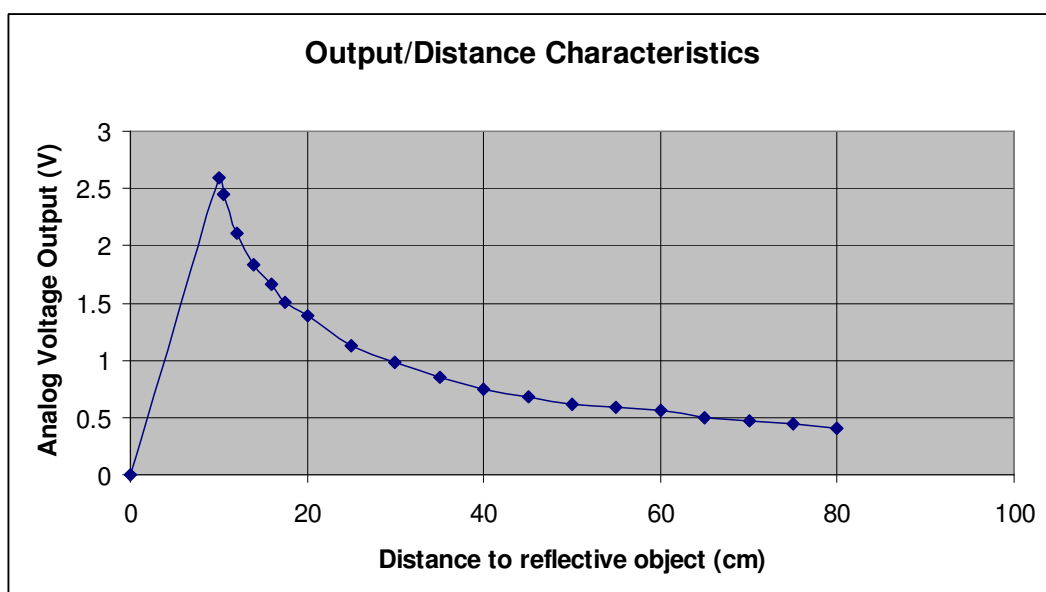


Figura 16- Gráfico Voltagem de saída x Distância do Objeto Feito no Excel

Após a realização deste procedimento, deve-se transformar a leitura analógica do sensor realizada pelo microcontrolador em um número inteiro. A tensão que o microcontrolador pode realizar como uma leitura analógica varia entre 0 V e 5 V, além disso, foram utilizados 8 de bits para realizar a conversão analógica. Logo o microcontrolador poderá realizar leituras de 0 V a 5 V que serão representadas por números entre 0 e 255 (2^8 bits) após a conversão analógica. Abaixo segue uma tabela com a conversão da tensão de saída do sensor por distância do objeto:

Distância (cm)	Valor correspondente
0	0
10	132,6
10.5	124,95
12	107,61
14	93,84
16	84,66
17.5	77,01
20	70,89
25	57,63
30	49,98
35	43,35
40	38,25
45	34,68
50	31,11
55	30,09
60	28,56
65	25,5
70	23,97
75	22,95
80	20,4

Tabela 3 - Tabela Estimada do Valor Decimal Convertido x Distância do Objeto

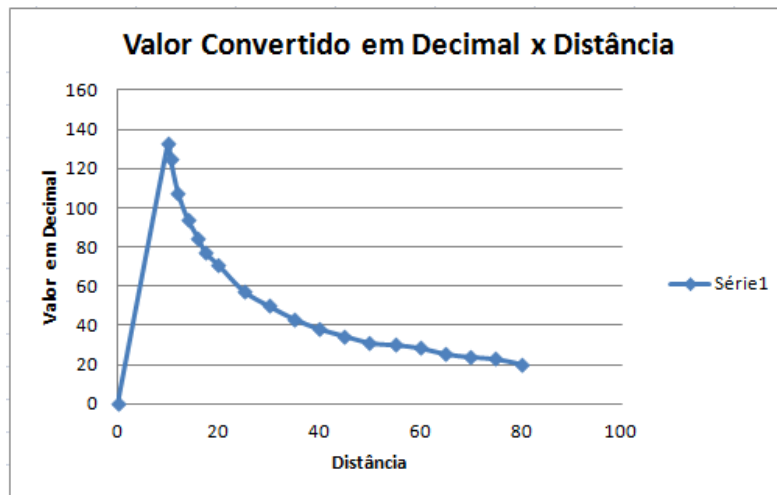


Figura 17 - Gráfico Valor Decimal Convertido x Distância do Objeto

Com o intuito de simplificar o modelo acima, realizou-se uma aproximação do gráfico obtido por duas equações lineares através do *software* Excel como demonstrado abaixo:

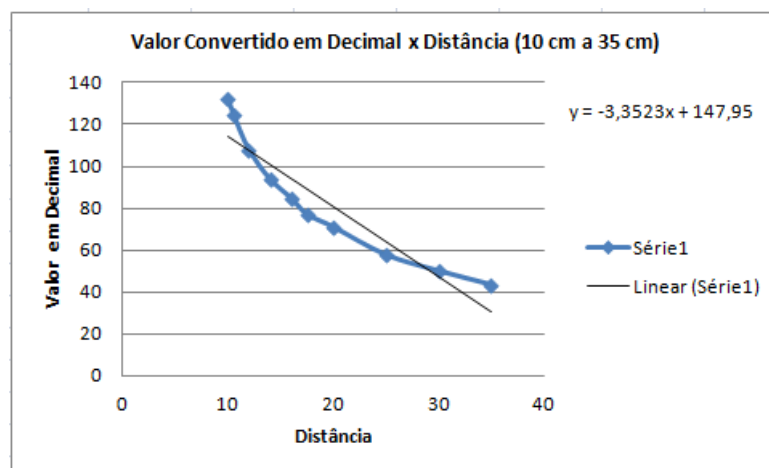


Figura 18 - Equação Aproximada de 10 a 30 cm Valor Decimal Convertido x Distância Objeto

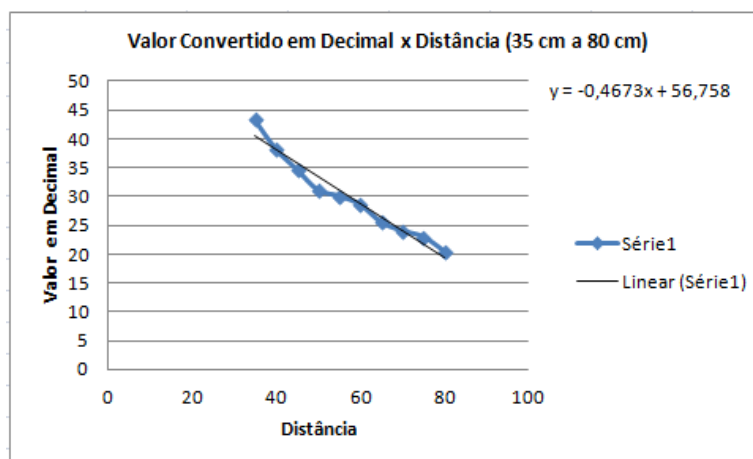


Figura 19 - Equação Aproximada de 35 a 80 cm Valor Decimal Convertido x Distância Objeto

O valor de entrada da função é o valor decimal convertido pelo o microcontrolador através da leitura analógica de cada sensor. Logo deve-se colocar a variável x em função da variável y para ser obtida a distância que o sensor está efetuando como leitura. Além disso, os valores foram arredondados para que não fosse utilizado ponto flutuante na programação, uma vez que esta ação requer um grande tempo de execução.

$$X = \frac{(148 - Y)}{3}$$

Equação 1 - Distância do Sensor SHARP entre 10 cm e 35 cm

$$X = \frac{(57 - Y)}{0.5} = (57 - Y) \times 2$$

Equação 2 – Distância do Sensor SHARP entre 35 cm e 80 cm

Abaixo segue um trecho detalhado do código implementado para realizar este procedimento no microcontrolador, além disso, é importante ressaltar que o código abaixo representa o procedimento para apenas um único sensor. Portanto para cada sensor deverá ser utilizado o mesmo código, havendo mudanças somente na variável armazenada e na porta analógica do microcontrolador em que é realizada a leitura do sensor. Todo o código desenvolvido para realizar este procedimento de testes dos sensores, pode ser encontrado em anexo a este trabalho.

```

set_adc_channel(0);      /*Especifica qual pino de leitura analógica será utilizado*/
delay_us(10);           /* Delay para esperar o multiplexamento do conversor analógico/digital*/
sensor_0 = read_adc();  /* sensor_0 recebe a leitura da porta analógica 0*/
if(sensor_0 >= 45)      /* Verifica em que caso a leitura se encontra (eq.1 ou eq.2)*/
    dist_0=(148-valor1)/3; /* Distância do sensor_0 se o caso for para a equação 1*/
else
    dist_0=(57-valor1)*2; /* Distância do sensor_0 se o caso for para a equação 2*/

```

Figura 20 - Trecho do Código Realizado para Calcular a Distância de um único Sensor

Mecânica e locomoção

Nesta nova plataforma optou-se por testar um novo conceito aprendido durante as pesquisas realizadas para a realização deste trabalho. Inicialmente toda plataforma que era desenvolvida para ser eficaz tinha como objetivo ser robusta, possuir uma grande normal através de um imã extremamente forte e ter uma grande redução na locomoção e conseqüentemente um torque alto que fosse capaz de vencer a força normal do imã e empurrar os adversários. Logo este tipo de projeto a princípio gera uma redução da velocidade do robô devido à grande relação de redução para obtenção de um alto torque.

O novo conceito a ser utilizado é oposto ao citado anteriormente, ou seja, o robô deverá possuir alta velocidade, agilidade e manobrabilidade. Este tipo de projeto visa aproveitar toda quantidade de movimento do robô para arremessar os adversários para fora da arena. Entretanto para que o robô consiga ser eficiente, este deverá ter uma rampa bem projetada e confeccionada, pois esta deverá conseguir penetrar por debaixo dos adversários descolando assim o imã do oponente do chão da arena. Uma vez que a capacidade magnética de um imã decai exponencialmente com a distância entre o imã e qualquer chapa de aço, podemos partir da premissa que depois que a rampa conseguir deslocar o imã adversário do chão da arena este poderá ser facilmente empurrado para fora do dojô.

Para auxiliar o projeto mecânico, a simulação estrutural e o desenvolvimento das plantas do protótipo, o *software* SolidWorks foi utilizado com o intuito de gerar um desenvolvimento virtual do robô, minimizando assim os custos e os desperdícios de material envolvidos no projeto. Abaixo segue algumas ilustrações do protótipo denominado C3 já com a melhor disposição dos sensores obtida através da bancada de testes:

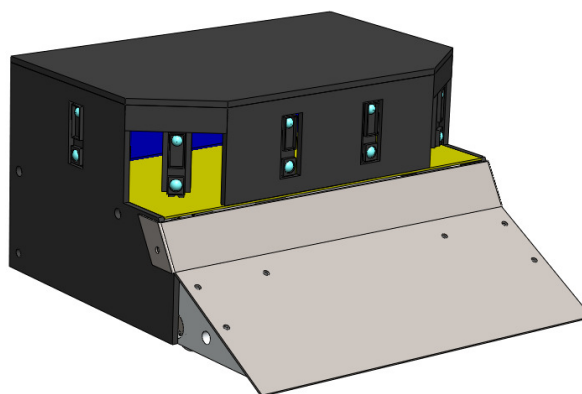


Figura 21 - Protótipo C3 para Validação de Conceito

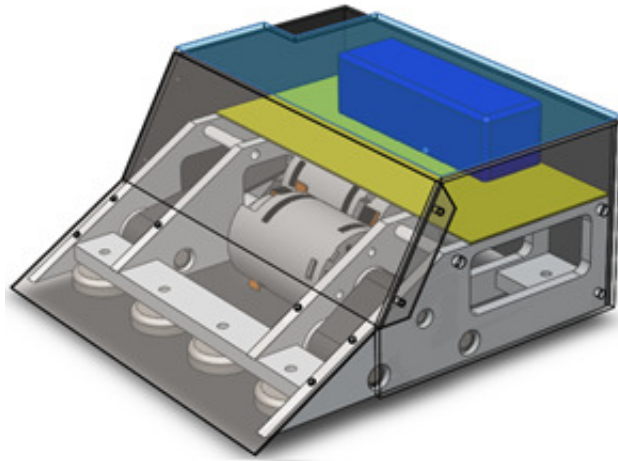


Figura 22 - Vista Interna do Protótipo

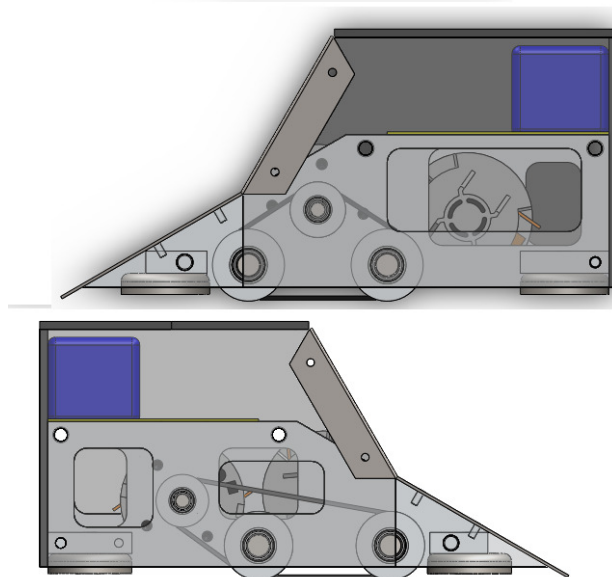


Figura 23 - Vistas Laterais do Protótipo

Com o intuito de desenvolver um projeto estruturalmente resistente e que fosse bem leve ao mesmo tempo, uma construção do tipo “caverna” normalmente realizada em aeronaves foi adotada. Este tipo de construção possibilitou que suas paredes de alumínio 6061 mesmo com uma espessura relativamente pequena, pudessem ser vazadas em determinados lugares para alívio de peso sem prejudicar a estrutura do robô. Algumas simulações estruturais realizadas pelo programa SolidWorks foram realizadas com o intuito de assegurar a estrutura do robô. É de suma importância ressaltar que para levar em consideração a força dos imãs nas simulações, alguns testes físicos tiveram que ser realizados, com objetivo de

aquisitar alguns dados do imã utilizado. Além disso, optou-se por neste protótipo utilizar imã de tamanho inferior ao antes adotado, entretanto em maior quantidade com o intuito de distribuir melhor a força exercida pelos imãs, uma vez que a plataforma anterior possuía um imã maior centralizado no fundo do robô.



Figura 24 - Ensaio de Tração do Imã na Instron 8502



Figura 25 - Bancada de Ensaio

Força (N)	Afastamento (mm)
324	0
220	0,1
249	0,2
170	0,3
150	0,4
140	0,5
122	0,6
100	0,7
92	0,8
87	0,9
76	1
68	1,1
64	1,2
62	1,3
57	1,4

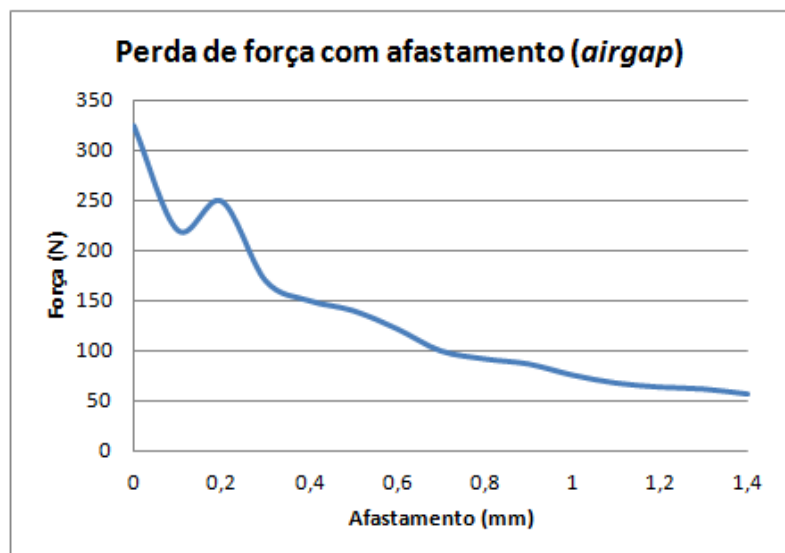


Figura 26- Variação da Força do Imã em Relação ao Afastamento do Suporte de Aço

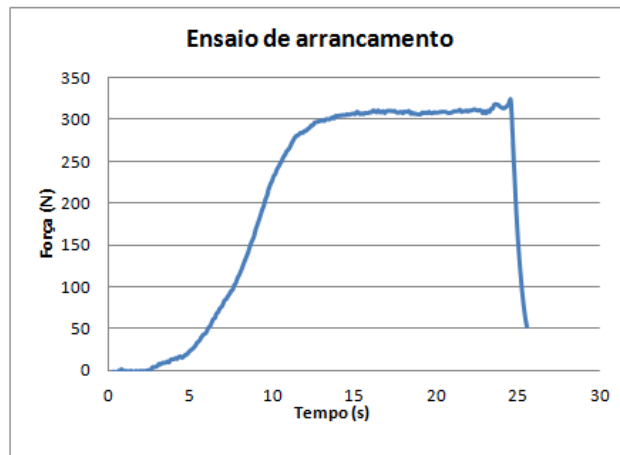


Figura 27 – Variação da Força do Ímã pelo o Tempo de Ensaio

As simulações demonstradas abaixo foram realizadas utilizando uma força equivalente a 76 N para cada ímã, ou seja, considerando uma distância da superfície da arena equivalente a 1 mm de acordo com os ensaios mecânicos realizados e o projeto virtual do robô.

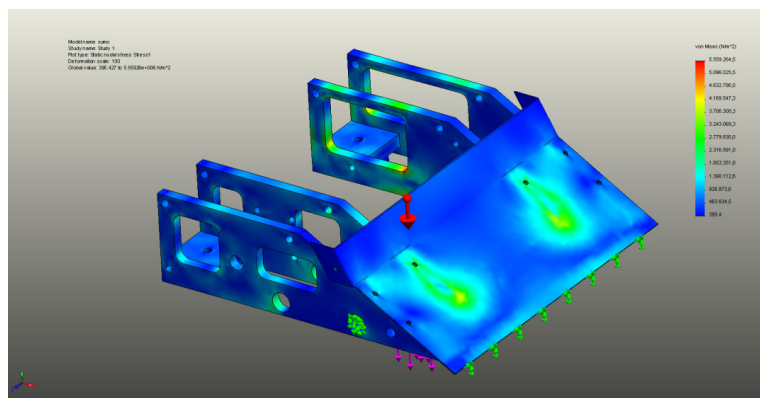


Figura 28 – Simulação de Tensão na Estrutura do Robô

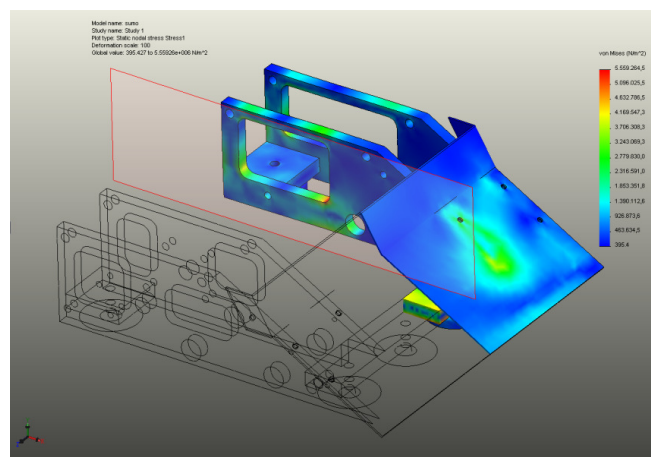


Figura 29 – Simulação de Tensão na Estrutura do Robô (Visão Parcial).

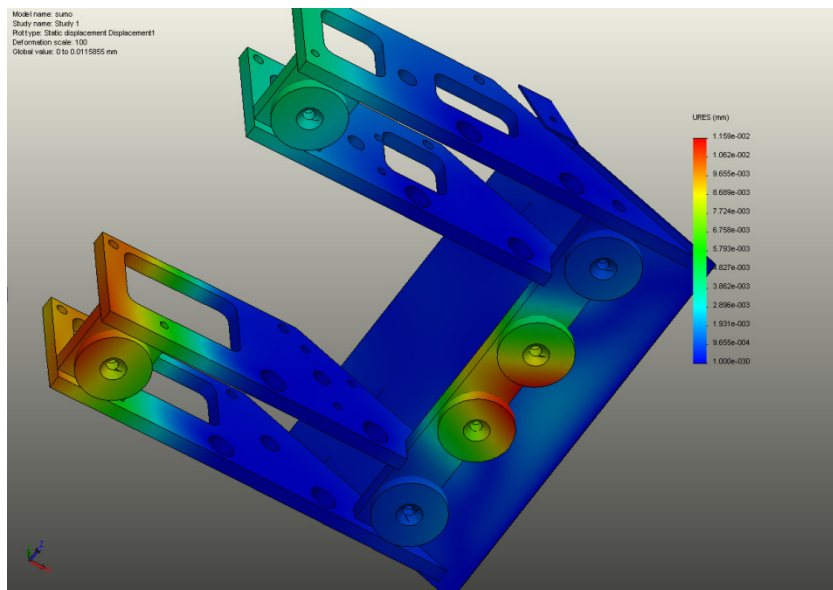


Figura 30 – Simulação da Deformação Estrutural do Robô.

Através da simulação da deformação estrutural do robô, pode-se concluir que mesmo com a influência do imã e tendo todas as suas paredes vazadas com o intuito de alívio de peso, a base do robô praticamente não sofre flexão de acordo com as simulações realizadas.

A locomoção adotada para este protótipo constitui de dois motores modelo 18V New Style do fabricante Dewalt, normalmente utilizados em furadeiras elétricas do próprio fabricante como mostrado na figura abaixo:

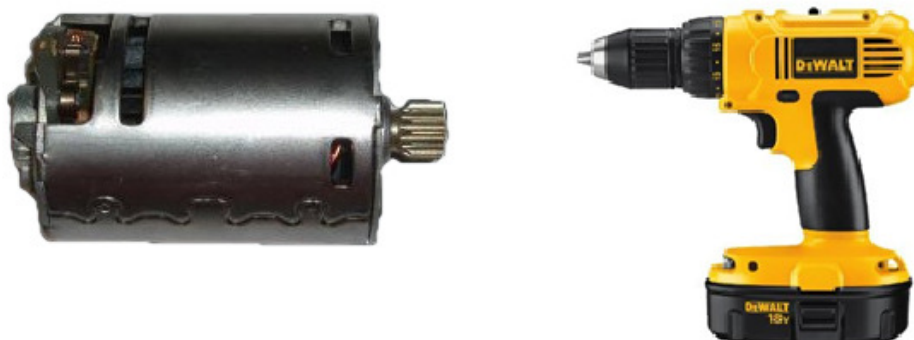


Figura 31 - Motor Dewalt 18V New Style

Este motor é bastante utilizado por diversas equipes que participam em competições de robôs de combate, portanto ele pode ser considerado de grande robustez para esta aplicação, uma vez que os motores juntamente com a eletrônica de potência são normalmente os primeiros componentes a se danificarem em uma partida de robôs

sumô. Este fato ocorre devido ao extremo trabalho de seus componentes. Abaixo segue algumas características deste motor:





				
Nome	Bosch GPA	Bosch GPB	D-Pack	DeWalt 18V
Robô que usa	Orion (Triton)	Lacraia (RioBotz)	Wedgie (Soh Toskeira)	Ciclone (RioBotz)
Tensão (V)	24	12	12	24
Pot.Máx. (W)	1175	282	3561	946
Peso (kg)	3,8	1,5	3,5	0,5
Potência/Peso	309	188	1017	1892
I_{stall} / I_{no_load}	23	25	63	128
K_t (N·m/A)	0,061	0,042	0,020	0,0085
K_v (RPM/V)	167	229	485	1100
R_{motor} (Ω)	0,13	0,121	0,00969	0,072
I_{no_load} (A)	8,0	3,9	19,6	2,6

Figura 32 - Características do Motor Dewalt 18V Modelo New Style (Meggiolaro, 2006)

A locomoção do robô além de possuir os dois motores descritos, conta com uma redução baixa de aproximadamente 1.4:1 para poder ser extremamente veloz e testar a proposta já descrita anteriormente de um robô ágil ao invés de um robô lento com grande torque. Além disso, esteiras foram utilizadas para tracionar o robô devido aos pontos positivos deste sistema também já descritos na plataforma atual.

A nova proposta foi testada no recente campeonato nacional Winter Challenge 2011 e demonstrou-se ser de grande potencial, entretanto o robô protótipo C3 não conseguiu alcançar o pódio devido alguns problemas mecânicos. Além disso, este sistema revelou-se ser praticamente impossível de ser controlado remotamente por um ser humano devido a toda sua agilidade, havendo necessidade de um controle auxiliar. É importante ressaltar que para um projeto definitivo alguns aspectos devem ser modificados tais como:

Esteiras → Este tipo de locomoção apresenta uma enorme dificuldade na hora de realizar a manutenção e montagem do robô, além disso, os pontos positivos deste método não realizam tanta diferença como imaginado. Portanto para o projeto

definitivo, a utilização de rodas com diâmetros não tão grandes (para não diminuir tanto o torque) deverá ser adotada.

Motores → Os motores 18V New Style da Dewalt não serão mais utilizados e serão substituídos por motores Maxon modelo RE40 que são mais eficientes. É importante ressaltar que estes motores não foram utilizados anteriormente devido ao seu alto custo para apenas um protótipo que serviu como prova de conceito.

Eletrônica

Microcontroladores

Os microcontroladores (também denominados MCU) são chips inteligentes, que tem um processador, pinos de entradas/saídas e memória, ou seja, são verdadeiros computadores em um chip. Através da programação dos microcontroladores podemos controlar suas saídas, tendo como referência as entradas ou um programa interno. Estes dispositivos são embarcados no interior de algum outro dispositivo (geralmente um produto comercializado) para que possam controlar as funções ou ações deste determinado produto.

Os microcontroladores se diferenciam dos processadores, pois além dos componentes lógicos e aritméticos usuais de um microprocessador de uso geral, o microcontrolador integra elementos adicionais em sua estrutura interna, como memória de leitura e escrita para armazenamento de dados, memória somente de leitura para armazenamento de programas, EEPROM para armazenamento permanente de dados, dispositivos periféricos como conversores analógico/digitais (ADC), conversores digitais/analógicos (DAC) em alguns casos e interfaces de entrada e saída de dados.

Com frequências de *clock* de poucos MHz (Megahertz) ou talvez menos, os microcontroladores operam a uma frequência muito baixa se comparados com os microprocessadores atuais, no entanto são adequados para a maioria das aplicações usuais como: controlar uma máquina de lavar roupas, uma esteira de chão de fábrica ou até mesmo no desenvolvimento de um sistema robótico como é neste caso. O seu consumo em geral é relativamente pequeno e possuem geralmente habilidade para

entrar em modo de espera (*Sleep* ou *Wait*) aguardando por uma interrupção ou evento externo, como por exemplo, o acionamento de uma tecla, ou um sinal que chega via uma interface de dados.

O que diferencia os diversos tipos de microcontroladores são as quantidades de memória interna (programa e dados), velocidade de processamento, quantidade de pinos de entrada/saída (I/O), alimentação, periféricos, arquitetura e instruções.

Após uma vasta pesquisa e avaliação da relação custo/benefício que os diversos microcontroladores disponíveis no mercado oferecem, além da facilidade de encontrar os modelos existentes no mercado, optou-se pelo microcontrolador PIC16F877A. Os fatores decisivos para esta escolha estão listados logo abaixo:

Um conversor A/D de no máximo 10 bits de resolução e multiplexado com 8 canais → Uma vez utilizado 5 sensores no projeto e conseqüentemente 5 leituras analógicas, esta característica torna-se essencial na escolha deste microcontrolador não havendo a necessidade de um super dimensionamento com mais portas analógicas.

Memória Flash de 8 Kb → Como o microcontrolador é dotado de memória *Flash* é possível a utilização do sistema *bootloader*, além disso, esta quantidade é suficiente para o projeto.

Memória RAM de 368 bytes → Totalmente suficientes para rodar o código proposto.

Frequência de *clock* de até 20 MHz → Suficiente para processar todo o programa de uma forma rápida e eficaz sem prejudicar a resposta de controle do robô.

Bootloader → Este microcontrolador possui *bootloader* que será detalhado mais adiante, entretanto esta característica permite uma facilidade na hora de realizar a gravação do código no *chip* sem ter que retirá-lo do *hardware*.

Quantidade de 40 pinos → Há uma quantidade suficiente de pinos digitais e analógicos para realizar a leitura de todos os sensores, gerar um protocolo de comunicação serial (RS232) com o computador, acionar todos os *leds* de depuração do circuito e dos sensores.

Interrupção externa e interrupção na porta B → Este microcontrolador possui interrupção externa e interrupção da porta B que serão detalhadas mais adiante, entretanto tais características permitirão o desenvolvimento do código que servirá para reconhecimento da borda da arena através da leitura dos sensores de linha.

Para utilização do PIC 16f877a é necessário o desenvolvimento de um hardware específico. É importante ressaltar que todo o desenvolvimento do hardware foi realizado através do pacote de programas Proteus com os *softwares* ISIS e ARES. Estes programas foram utilizados com o intuito de simular toda a eletrônica desenvolvida e a confecção dos desenhos da planta do *hardware* para fabricação respectivamente.

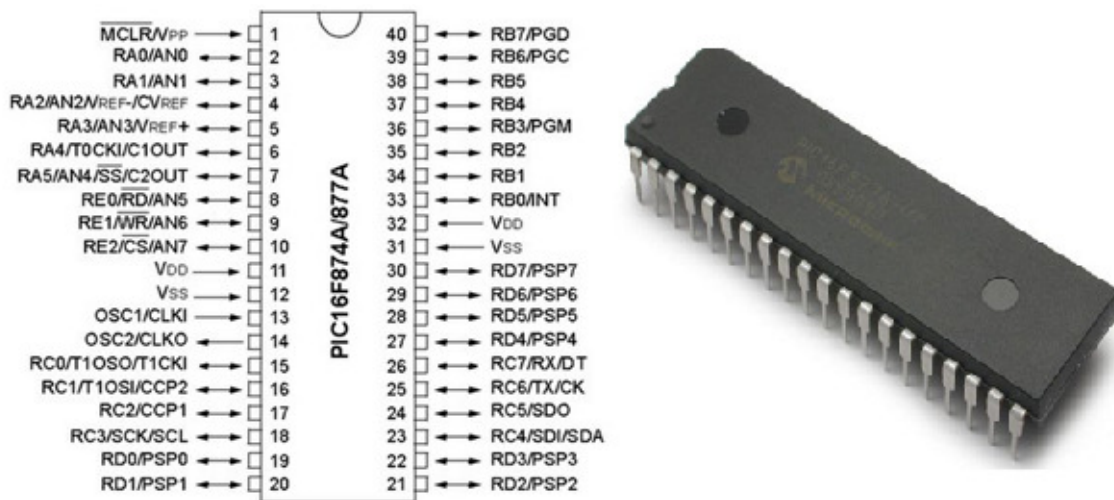


Figura 33 - Microcontrolador Pic16f877A

Bootloader

O *bootloader* é um programa muito pequeno que é programado na parte baixa ou alta da memória do microcontrolador que será utilizado no circuito em desenvolvimento. Este programa consegue comunicar com as ferramentas de desenvolvimento (utilizadas para escrever o código fonte) através de uma porta série ou outro tipo de ligação série, como por exemplo, um barramento USB, I2C, ou CAN. Logo o *bootloader* é um programa que carrega o código do programa a ser executado na memória do microcontrolador, evitando assim a necessidade de um programador externo e a retirada do microcontrolador do *hardware* toda vez que se quiser fazer uma nova gravação ou mudança no programa de controle.

Depois de completar esta operação, o *bootloader* transfere o controle para o programa carregado, e o utilizador pode então executar e testar o seu programa. Um novo programa pode ser carregado com o *bootloader* sempre que for necessário (após o programa anterior ter sido apagado). O microcontrolador tem que obedecer a um determinado número de requisitos para que se possa utilizar um *bootloader*. Em particular, o microcontrolador deve possuir:

1° → Memória de programa suficiente para armazenar o *bootloader* e o programa desenvolvido.

2° → Ser capaz de apagar e programar internamente a memória de programa.

3° → Uma porta RS232 ou outro tipo de ligação série, como por exemplo, USB ou CAN.

Interrupção Externa e Interrupção da Porta B

Uma interrupção é equivalente a uma chamada de procedimento, ou seja, toda vez que o microcontrolador detectar uma mudança de estado em determinados pinos o programa para de executar o que está fazendo e é direcionado para a execução de um determinado trecho de código. Um exemplo típico é na varredura de teclados. Sempre que pressionamos uma tecla, uma interrupção é gerada para o microcontrolador, solicitando o tratamento desta e conseqüentemente a busca de qual tecla foi apertada. Assim, o microcontrolador não é obrigado a varrer constantemente o teclado em busca se uma tecla foi apertada, otimizando assim o código e tornando o programa mais eficaz.

Esta característica será importante para a programação dos sensores de linha, uma vez que a interrupção irá gerar uma precedência sobre o restante do código, evitando assim que o robô saia da arena de forma mais eficiente. Toda vez que algum sensor de linha for acionado a interrupção irá ocorrer e independentemente de que linha de código o programa estiver executando, este será direcionado para executar o que estiver programado na interrupção (estratégia de recuo da borda da arena).

PWM (*Pulse Width Modulation*)

A modulação por largura de pulso (MLP) também conhecida pela sigla em inglês "PWM" (*Pulse Width Modulation*), envolve em manter uma frequência de sinal e variar a largura de um pulso (o tempo de alta do sinal) com o objetivo de transportar qualquer informação sobre um canal de comunicação ou controlar o valor da alimentação entregue a um determinado componente. No caso deste projeto, o PWM é um sinal que deverá ter como função ligar e desligar os motores do robô através de uma chave (transistor bipolar ou MOSFET), fazendo com que o motor gire em uma velocidade proporcional à relação entre o intervalo de tempo ligado e o período do pulso.



Figura 34 - Sinal PWM (Meggiolaro, 2006)

Este tipo de técnica é utilizado pelo controlador de velocidade do fabricante Banebot que será utilizado no projeto para controlar os motores.

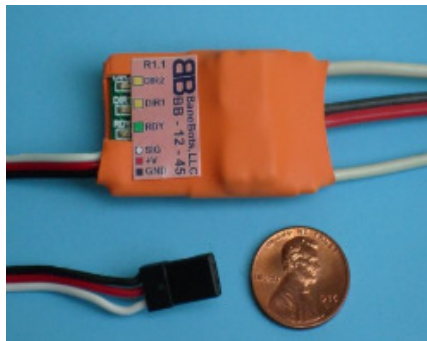


Figura 35 - Controlador de Velocidade da Banebot

Hardware

Como mencionado anteriormente, todo o desenvolvimento do hardware foi realizado através do pacote Proteus que agrega o ambiente de simulação de circuitos eletrônicos através do programa ISIS e o programa ARES para desenho de circuito impresso. Logo este tópico será explicado através de duas etapas: o desenvolvimento do layout do circuito para fabricação da placa e a simulação computacional do circuito.

Simulação do Circuito → o primeiro passo para realizar a simulação de toda eletrônica foi elaborar a divisão da mesma em quatro módulos menores, com o objetivo de facilitar o desenvolvimento de todo o circuito e os testes envolvidos. Logo os seguintes subcircuítos foram considerados:

- Circuito 1 – regula a tensão de 8.4 V (bateria de Lipo 2S) para 5V com o objetivo de alimentar o pic, sensores de linha e *leds* periféricos do microcontrolador.
- Circuito 2 – regula a tensão de 8.4 V (bateria de Lipo 2S) para 5V com o objetivo de alimentar os sensores infravermelhos de detecção do oponente.
- Circuito 3 – realiza a comunicação serial entre o microcontrolador e o computador com o intuito de realizar a gravação do pic por *bootloader* e realizar a depuração de possíveis problemas no robô.
- Circuito 4 – realiza a identificação da borda da arena através dos sensores infravermelhos de linha.

Circuito 1

O microcontrolador requer uma tensão de alimentação de 5V e a bateria utilizada no projeto possui 2 células de lítio polímero (3.3V nominais e 4.2V para cada célula carregada), havendo portanto uma necessidade de relugar a tensão da bateria para o pic16f877a e conseqüentemente seus periféricos. Logo a escolha do regulador de tensão linear LM7805 de fácil acesso no mercado e capacidade de fornecer até 1A foi realizada para esta função. Além disso, utilizou-se um capacitor de 0.1 μ F entre a saída e o terra do regulador com o objetivo de realizar um filtro passa baixa diminuindo assim possíveis ruídos. Abaixo segue o esquemático utilizado:

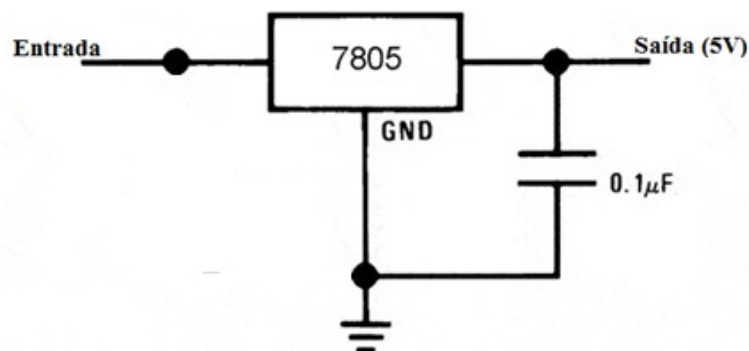


Figura 36 - Esquemático Circuito 1

É importante ressaltar que para projetos futuros é recomendável a utilização de outro capacitor entre a entrada e o terra do regulador. Esta medida possui a mesma função de filtro do capacitor de saída.

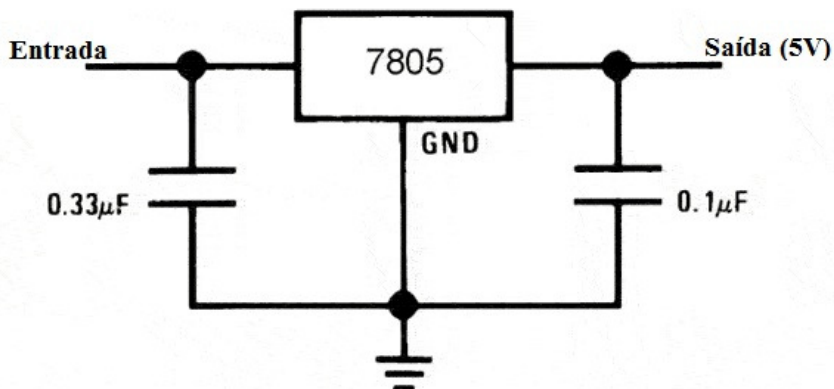


Figura 37 - Esquemático Recomendável do Circuito 1

Circuito 2

Além do microcontrolador, todos os sensores utilizados necessitam ser alimentados também com 5 V. Portanto como o regulador pode fornecer no máximo 1A e o pic16f877a de acordo com o fabricante consome até 200 mA que somados com 200mA dos quatro sensores de linha (50 mA cada) totalizam 400 mA, adicionados com 20mA do *led* indicador de circuito ligado, o integrado MAX232 atingem aproximadamente 50% do valor permitido.

O sensor infravermelho da SHARP GP2D12 para detectar o oponente consome aproximadamente 33 mA, como são utilizados 5 sensores para realizar a função de busca do oponente, podendo ser expandido futuramente para no máximo até 7 sensores temos no total 231 mA que somados com 500mA já descritos totalizam 731mA. Logo como o regulador consegue fornecer até no máximo 1A, optou-se pela a utilização de outro regulador para realizar a alimentação somente dos sensores SHARP GP2D12, evitando assim um possível aquecimento do regulador caso fosse utilizado um único componente para regular a tensão para todo o circuito como mostrado na equação abaixo:

$$P = v.i \therefore P = (8.4 - 5) \times 0.731 \therefore P \cong 2.48 W \text{ (dissipados em forma de calor)}$$

Equação 3 - Potência Dissipada no Caso de um Único Regulador

Circuito 3

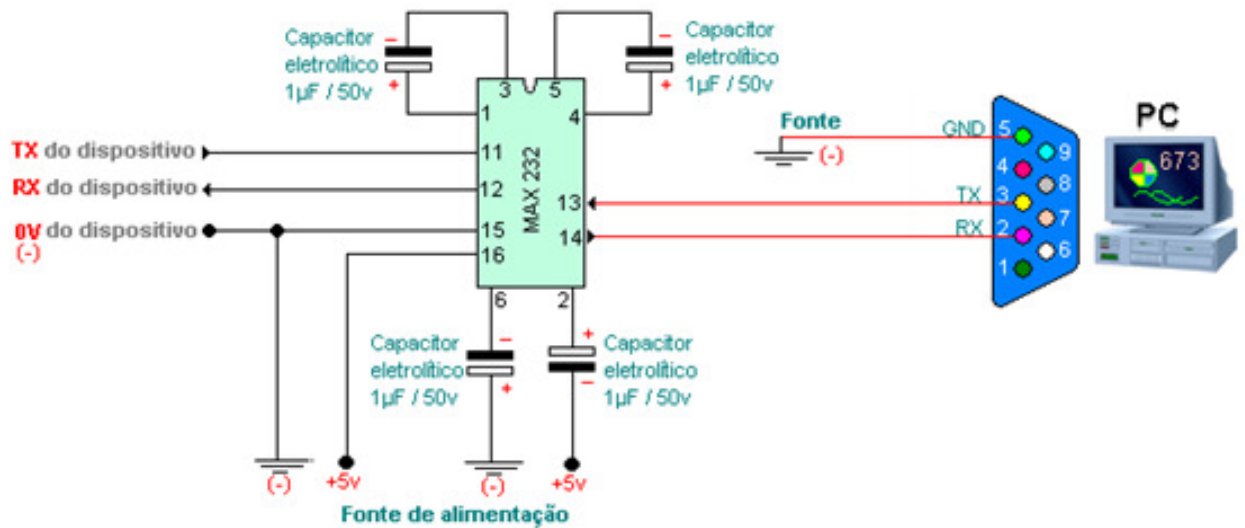


Figura 38 - Esquemático Circuito de Comunicação Serial (Messias)

O circuito acima é um conversor de sinais RS232/TTL utilizado para fazer a conexão entre o microcontrolador e o computador através da porta serial. Ele fornece uma ótima rejeição a ruído, além de ser mais robusto às descargas e curtos. Observe que os pinos 11 e 12 do Max232 são ligados respectivamente aos pinos TX (transmissor) e RX (receptor) do PIC. O cabo serial ligado ao computador é composto de 3 fios (RX, TX e GND). Os pinos 2 e 3 do conector DB9 são conectados através do cabo serial respectivamente aos pinos 14 e 13 do Max232. O pino 5 (GND) do conector é ligado à fonte de alimentação da placa controladora de acessos.

Os capacitores eletrolíticos são utilizados para configurar o funcionamento correto do Max232, sendo importante ressaltar que alguns deles trabalham com sua polaridade invertida. A comunicação serial é realizada com o intuito de realizar a gravação do código do microcontrolador por *bootloader*, além de realizar a depuração de possíveis problemas no robô.

Circuito 4

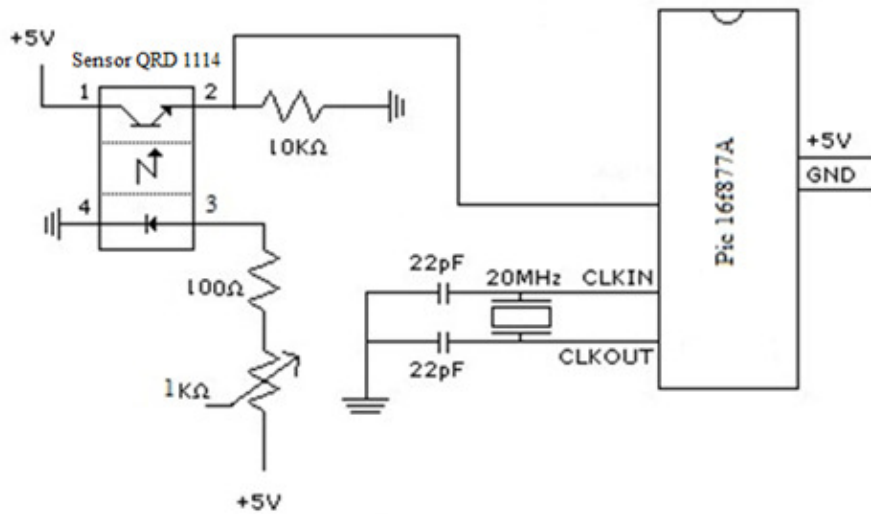


Figura 39 - Esquemático para Cada Sensor de Linha

O sensor escolhido para detectar a borda da arena foi QRD1114 já utilizado na plataforma anterior, pois como mencionado anteriormente este sensor já possui em um único encapsulamento o emissor e receptor infravermelho. Além disso, este modelo já possui um filtro infravermelho para o receptor diminuindo assim possíveis ruídos.

Portanto este sensor possui duas partes: um *led* emissor de luz infravermelha e um fototransistor sensível à luz infravermelha. O *led* emissor pode suportar uma corrente de até 50 mA de acordo com seu *datasheet*, logo temos:

$$V = R \times i \therefore 5 = R \times 0.05 \therefore R = 100 \Omega$$

Equação 4 - Cálculo do Resistor do Led Infravermelho do Sensor de Linha

Com o intuito de poder regular a sensibilidade do sensor de linha, ou seja, a intensidade de luz emitida pelo *led* infravermelho uma vez que a cor branca da borda da arena pode variar de tonalidade de acordo com a tinta que foi utilizada para pintar, optou-se por colocar um potenciômetro de **1KΩ** em série com o resistor afim de regular a corrente no emissor e conseqüentemente o seu brilho.

No *hardware* desenvolvido todos os emissores foram ligados juntos com o intuito de reduzir o tamanho do mesmo e o número de componentes, logo temos:

$$V = R \times i \therefore 5 = R \times (0.05 \times 4) \therefore R = 25 \Omega$$

Equação 5 - Cálculo do Resistor para Todos os Emissores dos Sensores de Linha

É importante ressaltar que este resistor não pode ser um resistor comum de $\frac{1}{8}$ watts, pois temos:

$$P = R \times I^2 \therefore P = 25 \times (4 \times 0.05)^2 \therefore P = 1 \text{ Watts (dissipados em forma de calor)}$$

Equação 6 - Cálculo da Potência do Resistor dos Emissores do Sensor de Linha

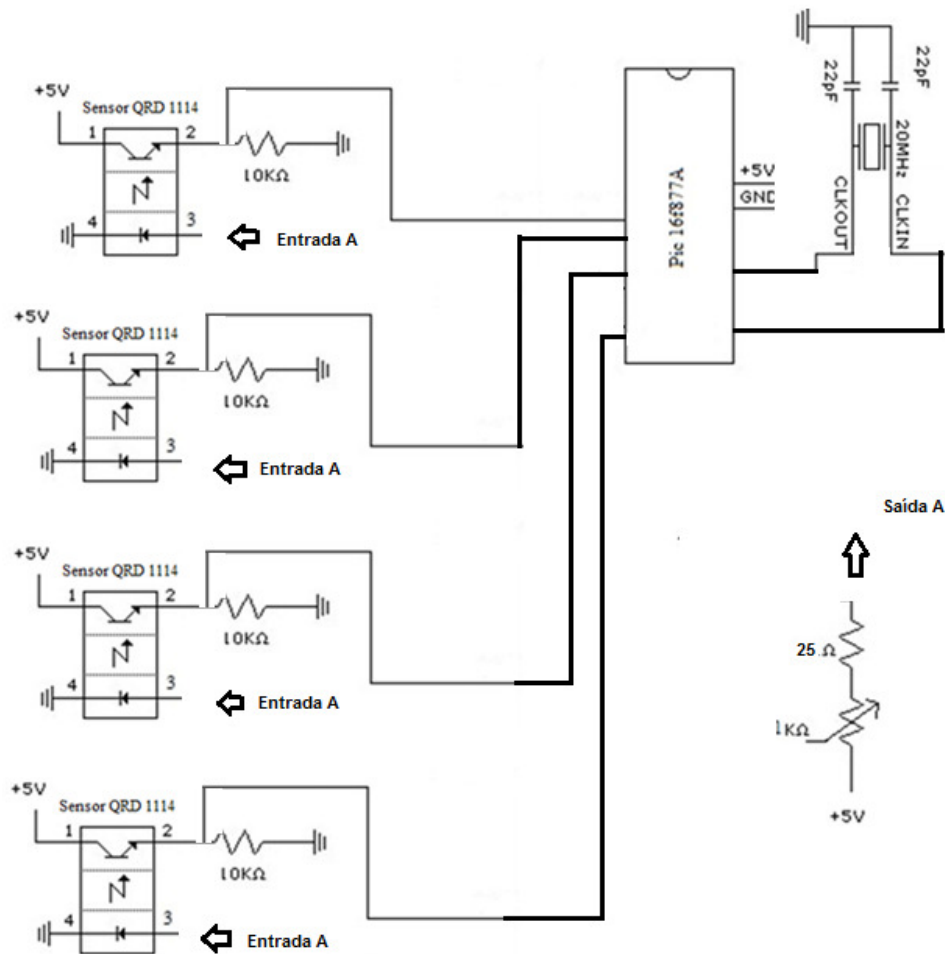


Figura 40 - Esquemático da Ligação no Hardware dos Sensores de Linha

Um resistor de $10K\Omega$ coloca a saída do receptor em nível alto quando ocorre o acionamento do fototransistor, logo quando isto ocorre à interrupção da porta b do microcontrolador é acionada e a rotina de fuga da borda da arena é executada de acordo com o sensor acionado.

Como mencionado anteriormente, a fim de testar toda eletrônica e o *software* desenvolvido foi utilizado o programa ISIS do pacote Proteus visando os seguintes aspectos:

1. Economia na compra de componentes eletrônicos para o protótipo → Evitar a compra desnecessária de diversos componentes até chegar a algum projeto funcional.
2. Economia de tempo → Perda de tempo na busca de componentes em lojas especializadas em comércio eletrônico. Além do fator prazo de entrega poder prejudicar o planejamento do produto final.

Os sensores SHARP GP2D12 por fornecerem uma tensão de saída proporcional à distância foram simulados através de potenciômetros como demonstrado no diagrama de simulação que será apresentado mais adiante. Assim ao mudar o valor da resistência do potenciômetro, o valor da tensão sobre o mesmo varia simulando o comportamento do sensor SHARP (como se a distância em relação a algum objeto estivesse sendo variada). Já os sensores de linha por fornecerem uma saída digital, foram simulados somente com uma simples chave de liga/desliga. Além disso, a simulação possui uma série de *leds* a fim de demonstrar quais sensores estão sendo ativados e para realizar a depuração de possíveis problemas com os sensores no futuro, sem haver a necessidade de desconectá-los do robô para verificar a queima de algum deles por exemplo.

A simulação ainda disponibiliza um osciloscópio virtual, um *hyper* terminal virtual e voltímetros com o intuito de ajudar a solucionar possíveis erros de *software* e eletrônica ao longo de seus respectivos desenvolvimentos. Paralelamente a todas estas ferramentas, a simulação ainda conta com o subcircuito para regular a tensão para os componentes (apenas foi simulado com um único regulador) e o cristal do microcontrolador. É importante ressaltar que todo o subcircuito do Max232 é simulado automaticamente pelo o programa uma vez que o *hyper* terminal virtual foi utilizado.

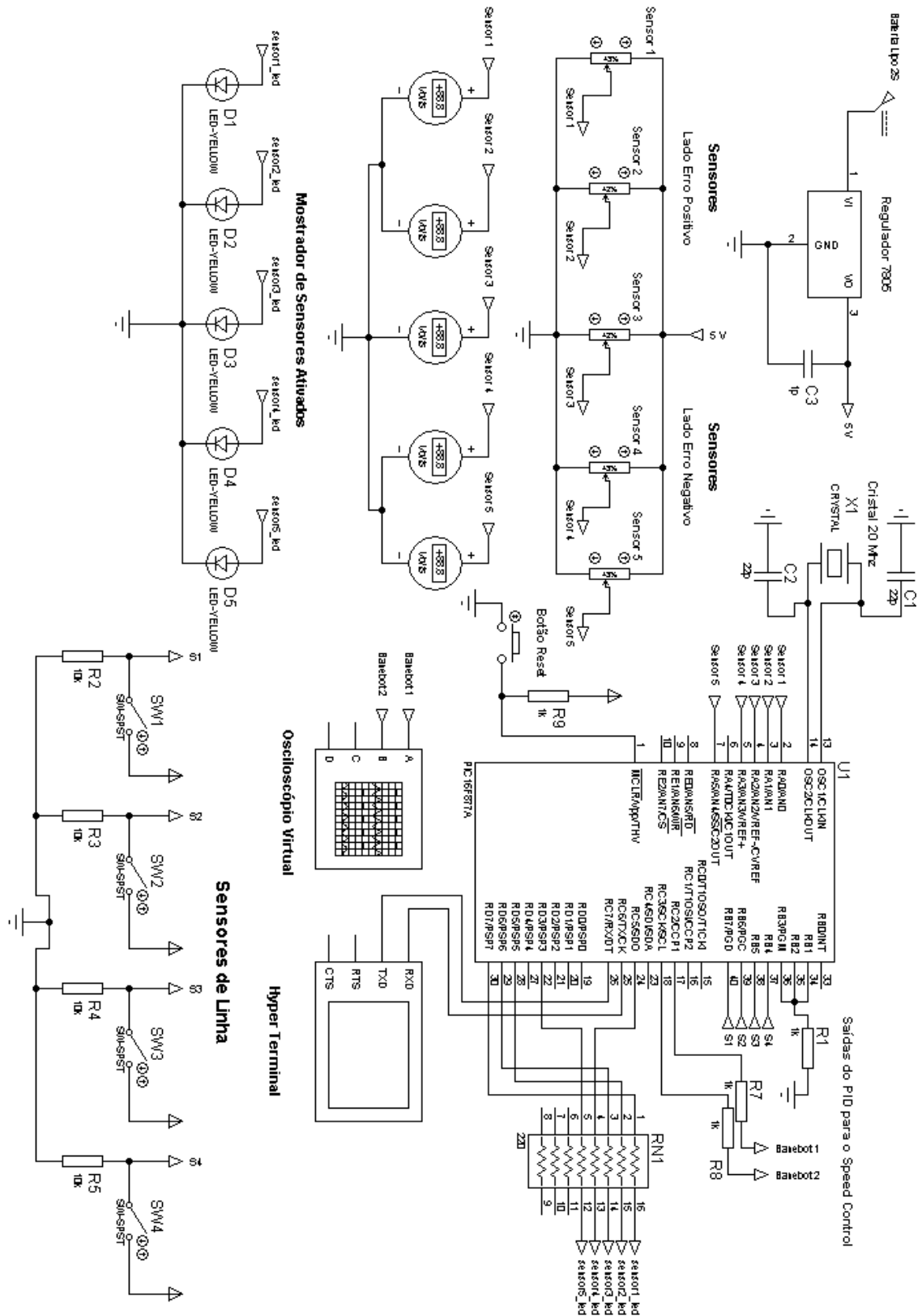


Figura 41 - Diagrama de Simulação da Eletrônica no Software ISIS

Desenvolvimento do Layout do Circuito para Fabricação da Placa → Uma vez realizada toda a etapa do desenvolvimento e testes da eletrônica no ISIS, deve-se projetar o layout da placa definitiva a ser fabricada e que deverá ser utilizada no robô. Como já foi mencionada anteriormente, esta etapa também foi realizada com o suporte do pacote Proteus através do software ARES.

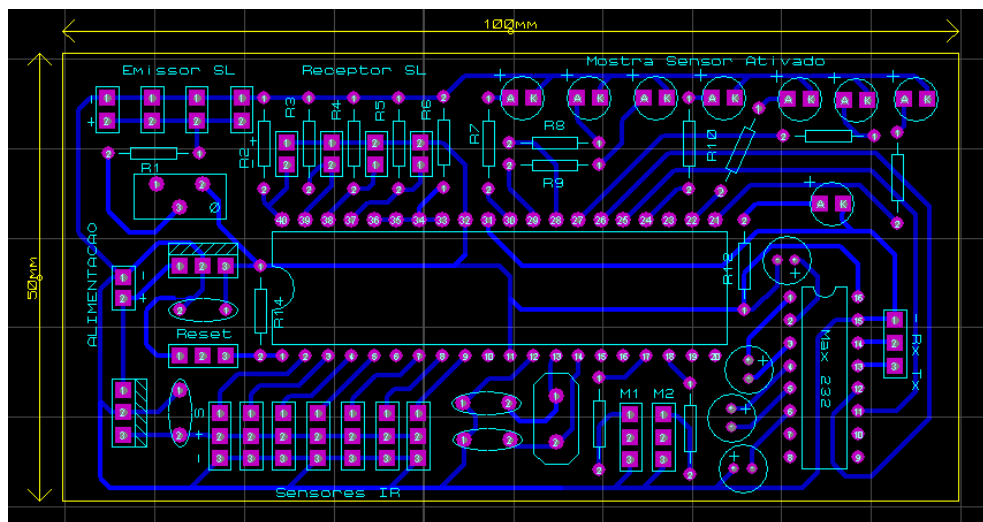


Figura 42 - Layout do Desenvolvimento do Hardware no Software Ares

Após a confecção de todo o layout da placa de fabricação, desenvolveu-se o desenho da placa em 3D com o mesmo programa, com o intuito de gerar todas suas vistas para verificação de possíveis erros e otimizações, além de servir como guia na hora de realizar a soldagem de todos os componentes.

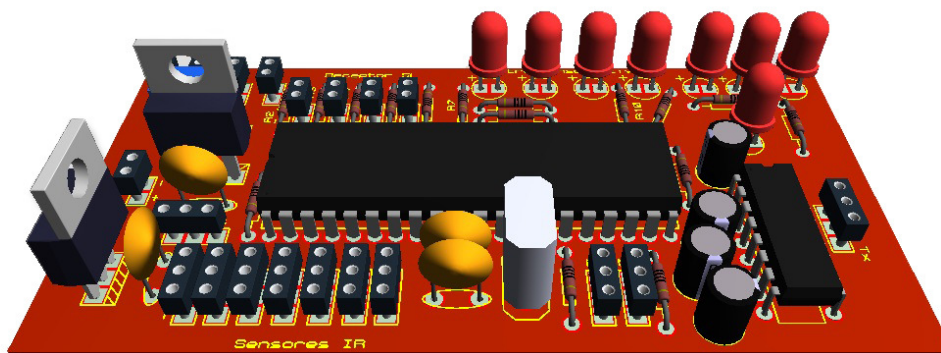


Figura 43 - Vista Frontal 3D do Hardware

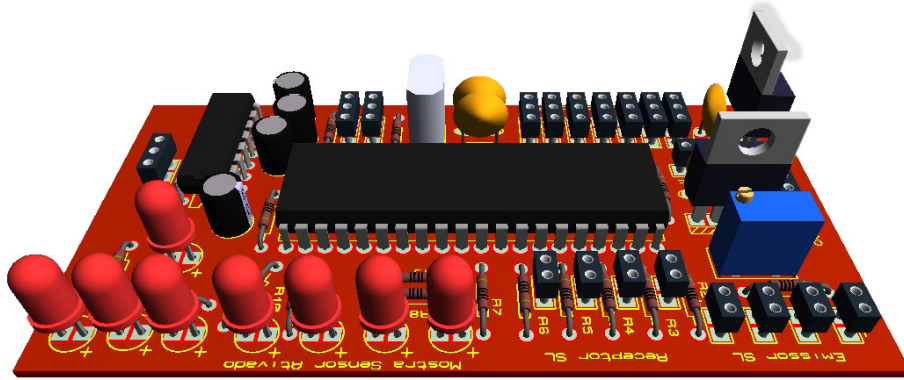


Figura 44 - Vista Posterior 3D do Hardware

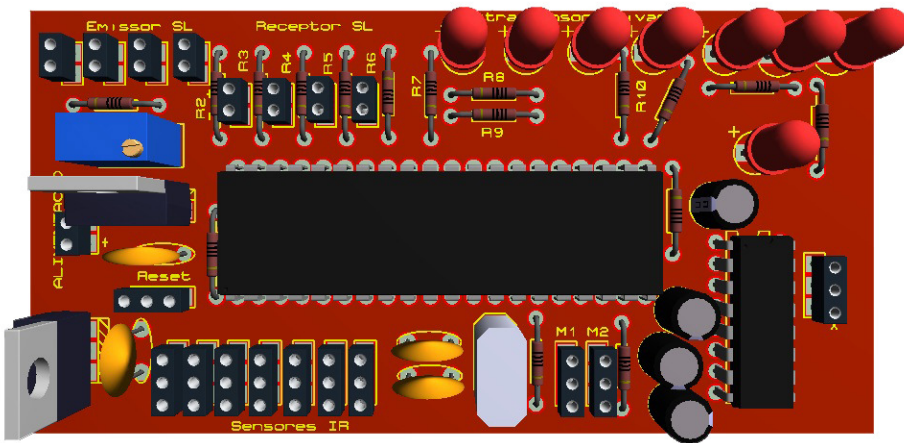


Figura 45 - Visto Superior 3D do Hardware

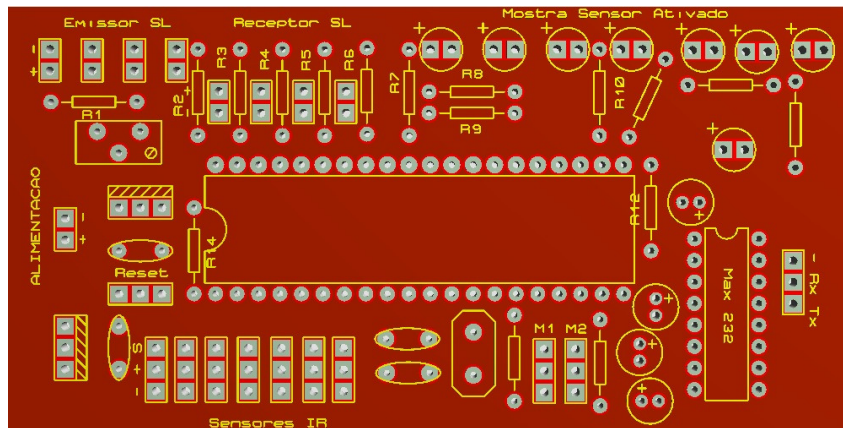


Figura 46 - Layout do Hardware

Software

O *software* foi desenvolvido com base na linguagem de programação C através do programa *PIC C Compiler*, além disso, todo o controle foi baseado nos conceitos da técnica do controle proporcional (P) integral (I) derivativo (D) que já é difundido na literatura.

O controle PID é uma técnica utilizada na maior parte da indústria que compara um valor medido de um processo (VP, variável de processo) com um valor de referência (SP, *set point*). A diferença destes valores (erro) é usada para calcular um novo valor, desta vez para a variável manipulada, que levará o processo ao valor desejado, ou seja, para o *set point*. O algoritmo do PID ajusta as saídas do processo baseado no histórico e a taxa de variação do erro do sinal, o que confere ao controlador mais precisão e estabilidade.

Modo Proporcional

Como o nome já indica, o modo de controle proporcional faz a correção de modo proporcional ao desvio. Levando em consideração os parâmetros ajustáveis e os parâmetros do processo ao qual o controle P irá atuar, a utilização isolada deste controle pode deixar um erro de *offset*. Logo apesar de ocorrer o controle da variável, esta pode não retornar ao seu valor desejado.

Modo Integral

Neste modo de ação o controlador gera um sinal de correção que depende da integração do desvio ao longo do tempo, o sinal de controle no modo integral aumenta proporcionalmente ao tempo no caso de o erro ser constante.

Para o controle integral a velocidade de correção é proporcional à amplitude do desvio. O controlador integral é bastante eficiente em conjunto com outros modos de controle como o proporcional, pois elimina o erro de *offset*.

Modo Derivativo

Esta ação de controle gera um sinal de correção que depende da velocidade de variação do desvio (erro). No controlador derivativo deve-se levar em consideração o ajuste do tempo derivativo. Este modo não pode ser utilizado isoladamente, pois a ação de controle para uma variação em degrau não faz sentido, visto que teríamos a derivada de/dt infinita no ponto de descontinuidade em que ocorre o desvio.

Modo P+I

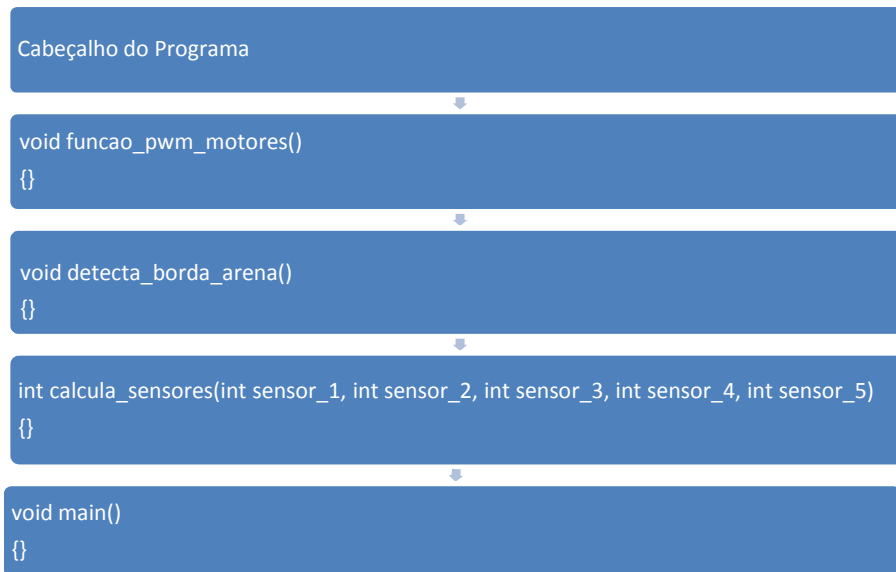
A ação proporcional utilizada sozinha não satisfaz todos os tipos de controle, só os com poucas exigências, pois geram desvios permanentes de *offset* da variável controlada como já mencionado anteriormente. Este desvio permanente pode ser anulado pelo uso das ações do modo proporcional e o modo integral combinados.

Modo P+I+D

Outro modo de controle é feito através da associação dos 3 modos já citados. Além de eliminar o *offset*, consegue com regulações apropriadas uma estabilização do processo e um retorno mais rápido ao equilíbrio do que com o modo PI (este modo não é usado para variáveis muito rápidas, pois tende ao seu descontrole).

Desenvolvimento do Código

Todo código será explicado detalhadamente através da explicação separada de cada função, entretanto podemos criar uma estrutura básica com suas principais funções como relacionado abaixo:



Cabeçalho → Nesta parte do código, todas as configurações do microcontrolador são definidas tais como: a utilização da porta serial e seus respectivos pinos, a utilização de um cristal externo, a frequência de *clock*, o número de bits para realização da conversão analógica/digital, todas as variáveis do programa, a não utilização do recurso *watch dog time* do microcontrolador e a respectiva biblioteca do mesmo.

```

#include <16f877a.h>
#DEVICE ADC=10
#fuses HS,NOWDT,NOPROTECT,PUT
#use delay(clock=20000000)
#use rs232(baud=9600,parity=N,xmit=PIN_c6,rcv=PIN_c7)

signed int16 kp=0, ki=0, kd=0, erro_integral=0, erro_derivativo=0, contador_derivativo=0, contador_integral;
signed int16 erro = 0, erro_anterior = 0;
int16 sensor_1, sensor_2, sensor_3, sensor_4, sensor_5;
int16 motor1 = 1500;
int16 motor2 = 1500;
int16 leitura_borda = 0;
int define_sensor_1 = 0, define_sensor_2 = 0, define_sensor_3 = 0, define_sensor_4 = 0, define_sensor_5 = 0;
int32 resultado_sensores = 0;

```

Figura 47 - Cabeçalho do Programa

Funcao_pwm_motores → Esta função é responsável por realizar o sinal PWM que é enviado para os controladores de velocidade da *Banebot*. O sinal descrito é gerado através de uma interrupção por tempo, ou seja, uma vez que é definido um valor de tempo (período), o microcontrolador dispara um contador que quando alcançado o valor definido aciona a interrupção e independentemente da linha de código que está sendo executado, o programa é direcionado para executar o que está planejado na função de interrupção. É importante ressaltar que uma vez que os comandos da função de

interrupção foram executados o programa é novamente direcionado para o lugar onde se encontrava. Abaixo segue uma ilustração da função implementada:

```
#int_timer1 /*PIN_C2 = Sinal Banebots 1 e PIN_C3 = Sinal Banebots 2*/
void funcao_pwm_motores()
{
    output_high(PIN_C2);
    delay_us(motor1);
    output_low(PIN_C2);
    output_high(PIN_C3);
    delay_us(motor2);
    output_low(PIN_C3);
    set_timer1(53036 + get_timer1());
}
}
```

Figura 48 - Função do PWM dos Controladores de Velocidade dos Motores

As variáveis motor1 e motor2 representam o tempo de alta do sinal. Os controladores de velocidade utilizados devem receber um sinal com período de 20 ms, sendo que o período de alta pode variar entre 1 ms e 2 ms. Quando o sinal possuir um período de alta entre 1ms e 1.5 ms o motor estará girando para um sentido, quando o sinal estiver com um período de alta de 1.5 ms o motor deverá estar parado e por último quando o sinal tiver com um período de alta entre 1.5 ms e 2 ms o motor deverá girar no sentido oposto ao inicial.

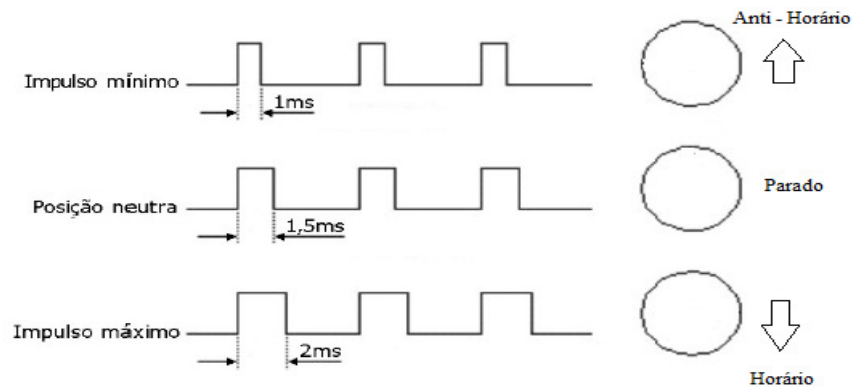


Figura 49 - Funcionamento do PWM no Controlador de Velocidade

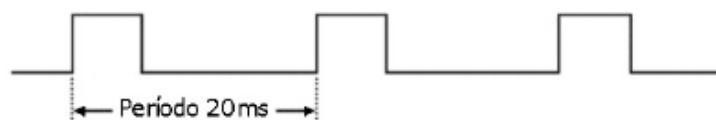


Figura 50 - Representação do Período do Sinal do Controlador de Velocidade

O valor 53036 do comando *set_timer1* representa uma frequência de 50 Hz, ou seja, o período de 20 ms que o microcontrolador deve utilizar de sinal, além disso, o comando *get_timer1* serve para que o mesmo contabilize o tempo de execução das instruções neste período.

Detecta_borda_arena → Esta função ocorre quando a interrupção da porta b do microcontrolador ocorre, ou seja, um sensor de linha é ativado. Assim que este determinado evento acontece independentemente da instrução que está sendo executada, o programa é direcionado para realizar a leitura da porta b. Uma vez que a leitura da porta é realizada, este valor é inserido em uma função *switch* que determina qual o caso que os sensores de linha estão ativados. Logo em seguida os valores de motor1 e motor2 são alterados, mudando os sinais do PWM dos controladores de velocidade, ou seja, realizando uma manobra de recuo da borda de acordo com os sensores ativados.

Calcula_sensores → Esta função recebe o número 1 para cada sensor ativado e o número 0 para cada sensor desativado, fazendo ascender um respectivo *led* para cada sensor ativado. Logo podemos saber a leitura que está sendo realizada pelo o microcontrolador sem auxílio de um computador e realizar uma rápida depuração dos sensores identificando, por exemplo, se há algum sensor queimado. Além disso, esta função realiza a soma dos respectivos sensores ativados com o intuito de gerar um número inteiro que será utilizado mais adiante no código para determinar o erro de controle.

	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5
Número Correspondente Ativado:	1	2	4	8	16
Número Correspondente Desativado:	0	0	0	0	0

Tabela 4 - Numeração Atribuída aos Sensores

Portanto caso os sensores 4 e 5 estejam ativados, esta função deve retornar o valor inteiro 24 para ser analisado em um comando de *switch* que irá definir o erro do controle de acordo com o número recebido da função *calcula_sensores*.

Main → Esta como a própria tradução da palavra para o português é a função principal de todo o programa, sendo responsável por executar o controle PID, a leitura analógica

dos sensores infravermelhos, a conversão dos sensores para funcionar como sensores digitais e a configuração de todas as interrupções.

Os dois primeiros comandos que são executados por esta função definem a frequência de 50 Hz da interrupção do *timer1* responsável por gerar o sinal PWM para os controladores de velocidade como já foi mencionado (Pereira, 2003).

Logo após a definição da frequência do *timer1* e da habilitação de todas as interrupções o programa espera o equivalente a 1 segundo fazendo o robô ficar imóvel por este período, ou seja, aguardando o início da partida. Em seguida, o programa entra em um *looping* infinito que contém todo o controle do robô até que este seja resetado ou desligado. Portanto é neste momento que é realizada a leitura dos sensores infravermelhos e o controle PID que conta com os valores do PWM enviados aos controladores de velocidade dos motores como as variáveis a serem manipuladas, além do ângulo da frente do robô em relação ao adversário como variável de processo que por sua vez deve ser zero como *set point*.

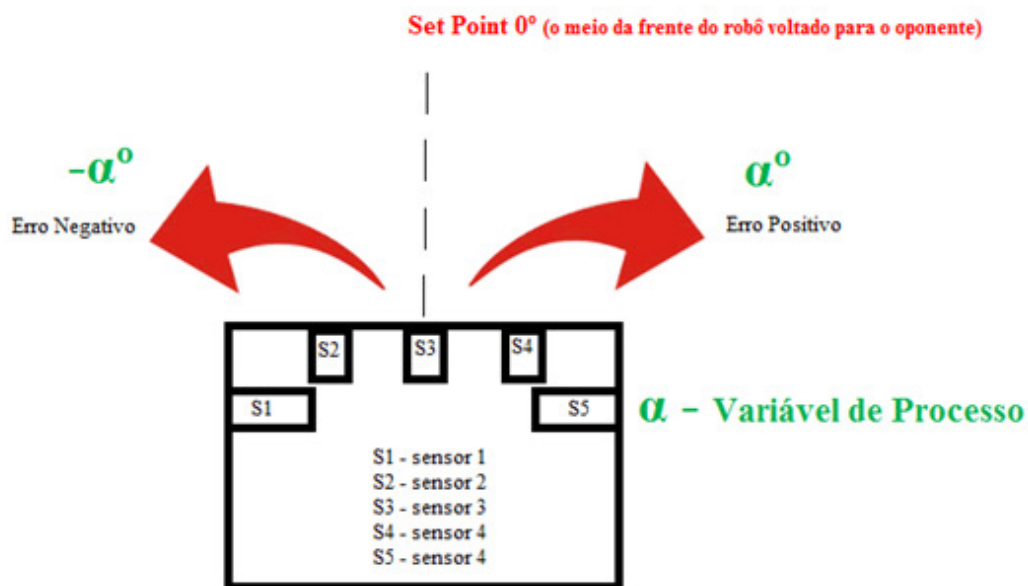


Figura 51 - Representação da Variável de Processo

Todo o processo de leitura dos sensores é realizado da mesma forma que já foi descrito anteriormente, ou seja, na bancada de teste que visava obter a melhor geometria dos sensores. Entretanto, nesta parte do projeto são utilizados 10 *bits* para a resolução do conversor analógico/digital ao invés de apenas 8 *bits*. Após a leitura de todos os sensores, a função *calcula_sensores* já detalhada anteriormente é chamada para calcular o valor que deverá representar uma configuração de sensores ativados. Este valor é

inserido em uma função *switch* que tem como dever selecionar o erro do controle de acordo com a combinação de sensores ativados. É importante ressaltar que quanto maior a quantidade de sensores no projeto, maior é o número de combinações de sensores ativados e conseqüentemente melhor é a discretização do erro. Abaixo segue uma tabela com as possíveis combinações de sensores que foram consideradas e os respectivos erros atribuídos:

	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor5	Erro
Ativado		x	x			-50
Ativado		x				-100
Ativado	x	x				-150
Ativado	x					-200
Ativado			x	x		50
Ativado				x		100
Ativado				x	x	150
Ativado					x	200
Ativado			x			0
Ativado		x	x	x		0

Tabela 5 - Tabela de Erros PID

Uma vez que o valor do erro foi calculado, ele é inserido no código do controle PID. Este código pode ser resumido em quatro partes, são elas: o cálculo do erro derivativo, o cálculo de erro integral, a saturação do erro integral para quando este ultrapassar um determinado valor e por último a equação em si do controle PID, ou seja, a lei de controle. A equação do PID pode ser definida como:

$$y(t) = K_p \cdot e(t) + K_i \int_0^t e(t) dt + K_d \cdot \frac{d e(t)}{dt}$$

Equação 7 - Equação Contínua do PID

onde $y(t)$ é o sinal de saída (atuação), $e(t)$ o erro entre a saída e a referência (*set point*) e K_p , K_i e K_d as constantes proporcional, integral e derivativa do PID. Entretanto não é possível realizar esta equação no microcontrolador, uma vez que não há um valor de erro contínuo, ou seja, o valor do erro é discretizado através da tabela mostrada anteriormente. Logo devemos realizar a discretização da equação substituindo a variável tempo (t) por:

$$t = n \cdot T$$

Equação 8 - Discretização da Variável Tempo

A variável T é o período de amostragem e n o número da amostra, logo a equação do PID discretizada pode ser representada da seguinte forma:

$$y(nT) = K_P \cdot e(nT) + K_I \cdot T \sum_{j=0}^n e(jT) + K_D \cdot \frac{e(nT) - e((n-1)T)}{T}$$

Equação 9 - Equação Discretizada do PID

Como nT representa a amostra número do sistema, pode-se substituir este termo simplesmente por n , dando origem a equação do PID digital:

$$y(n) = K_P \cdot e(n) + K_I \cdot T \cdot \sum_{j=0}^n e(j) + K_D \cdot \frac{e(n) - e((n-1))}{T}$$

Equação 10 - Equação Digital do PID

Considerando que o período de amostragem é sempre igual, pode-se realizar a seguinte simplificação:

$$y(n) = A \cdot e(n) + B \cdot \sum_{j=0}^n e(j) + C \cdot (e(n) - e(n-1))$$

Equação 11 - Simplificação da Equação a ser Implementada no Microcontrolador

Logo a equação acima descrita deve ser implementada no microcontrolador como a lei de controle da plataforma. É importante ressaltar que todo o código desenvolvido para implementação desta lei e consequentemente o controle PID está anexado a este trabalho, além disso, foram realizados todos os comentários necessários para que se faça o entendimento do mesmo e de como o processo foi desenvolvido.

Resultados

Alguns resultados em relação ao tempo de execução do código foram obtidos através do programa ISIS já mencionado. O código desenvolvido demora aproximadamente 0.5 ms para ser executado, enquanto que o código antigo para realizar um *loop* de controle demora o equivalente a 3 ms. Logo o código utilizando a técnica de controle PID, demonstra ter um tempo de resposta mais veloz do que a atual técnica *Fuzzy* implementada no microcontrolador.

Além disso, toda a eletrônica desenvolvida demonstrou-se ser bastante eficiente e robusta, sendo capaz de realizar todo o controle de forma rápida e eficaz. A utilização do circuito integrado Max232 para realizar a gravação do microcontrolador através do sistema *bootloader* demonstrou-se ser bastante útil, devido a toda praticidade durante o desenvolvimento e testes do código. Logo todo este sistema possibilitou uma economia enorme de tempo ao longo do processo de criação. Além disso, a depuração dos sensores através de *leds* indicadores, demonstrou-se ser bastante eficiente e prática na hora de realizar a manutenção do robô.

Conclusão

A plataforma desenvolvida confirma a eficiência do controle utilizando a técnica PID. Este sistema demonstra ter um grande potencial em competições deste gênero, entretanto como o sistema mecânico definitivo ainda não foi desenvolvido não é possível afirmar com toda certeza que este sistema irá superar o atual em competições.

Uma proposta para projetos futuros é utilizar uma das duas plataformas atuais e realizar modificações para que toda a eletrônica desenvolvida possa ser utilizada com a mecânica da plataforma atual. Assim com uma mecânica quase idêntica ficaria mais fácil de comparar de forma experimental os resultados de desempenho da técnica de controle PID em relação à técnica de controle *Fuzzy*.

Bibliografia

Barbosa, A. O. *Controle de Robô Usando Técnicas Inteligentes*.

Hamdan, N. B. (2007). *The Development of Mini-Sumo Robot*.

<http://itp.nyu.edu/physcomp/sensors/Main/HomePage>. (s.d.). Obtido de ITP Sensor Workshop Wiki.

<http://www.eletrica.ufpr.br/artuzi/apostila/cap4/pg07.html>. (s.d.). Obtido de Elétrica UFRR.

<http://www.ica.ele.puc-rio.br>. (s.d.). Obtido de ICA Laboratório Puc-Rio.

James Wolfer, H. R. (2005). An Integrated Khepera and Sumo-Bot Development Environment for Assembly Language Programming.

Meggiolaro, M. A. (2006). *Tutorial em Robôs de Combate* (Versão 1 ed.). Rio de Janeiro.

Messias, A. R. (s.d.). <http://www.rogercom.com/>. Obtido de rogercom.

Nawid Yakuby, C. L. (2003). *Knight Rider : Autonomous Sumo Robot*.

Ogata, K. (2011). *Engenharia de Controle Moderno* (5 ed.). Pearson Education.

Pereira, F. (2003). *Microcontroladores PIC: Programação em C*. Érica.

S. Chow, M. H. (2008). The Design of an Autonomous Sumo Robot.

Smith, S. *Microeletrônica* (5 ed.). Pearson.

Anexos