UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

DAVID LIMA MARINHO

APERFEIÇOAMENTO DE UM ROBÔ DE SUMÔ AUTÔNOMO

TRABALHO DE CONCLUSÃO DE CURSO 2

PATO BRANCO
2017

DAVID LIMA MARINHO

APERFEIÇOAMENTO DE UM ROBÔ DE SUMÔ AUTÔNOMO

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso Superior de Engenharia de Computação, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de bacharel.

Orientador: Prof. Dr. César Rafael Claure Torrico
Coorientadora: Profa. Dra. Kathya Silvia Collazos Linares

PATO BRANCO
2017

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Departamento Acadêmico de Informática
Curso de Engenharia de Computação

TERMO DE APROVAÇÃO

Às 8 horas e 20 minutos do dia 07 de abril de 2017, na sala V006, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, reuniu-se a banca examinadora composta pelos professores César Rafael Claure Torrico (orientador), Kathya Silvia Collazos Linares (coorientador), Marcelo Teixeira e Cleidimar Nardi para avaliar o trabalho de conclusão de curso com o título **Aperfeiçoamento de um robô de sumô autônomo**, do aluno **David Lima Marinho**, matrícula 01147064, do curso de Engenharia de Computação. Após a apresentação o candidato foi arguido pela banca examinadora. Em seguida foi realizada a deliberação pela banca examinadora que considerou o trabalho aprovado.

_____	_____
César Rafael Claure Torrico Orientador (UTFPR)	Kathya Silvia Collazos Linares Coorientador(UTFPR)
_____	_____
Marcelo Teixeira (UTFPR)	Cleidimar Nardi (UTFPR)
_____	_____
Beatriz Terezinha Borsoi Coordenador de TCC	Pablo Gauterio Cavalcanti Coordenador do Curso de Engenharia de Computação

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

RESUMO

MARINHO, David Lima. Aperfeiçoamento de um robô de sumô autônomo. 2017. 67f. Trabalho de Conclusão de Curso de Bacharelado em Engenharia de Computação - Universidade Tecnológica Federal do Paraná. Pato Branco, 2017.

A automação trouxe grandes avanços para a sociedade, causando grande impacto na vida do homem, facilitando processos, reduzindo fronteiras de tempo e espaço, acelerando produções e de uma maneira geral trazendo conforto e evolução nos tempos modernos. A robótica móvel consiste na atuação de robôs autônomos ou não, para resolver tarefas das quais o homem não seria capaz ou levaria um tempo maior para realizar, como voar e mapear um ambiente de maneira precisa e rápida, ou até mesmo facilitar transportes de material entre áreas em uma indústria. A autonomia do robô é importante pela sua atuação independente e a sua precisão de resposta a partir de sinais. Além disso, permite-se manter a produção constante sem a necessidade de pausas para descansos, acelerando assim os processos que antes eram feitos manualmente, alvo de atenção desde a sua primeira intensa utilização em 1914 nas linhas de produção na indústria. Assim, este projeto se insere na automação com a integração de sensores, controladores e acionadores visando o desenvolvimento e aprimoramento de um robô de sumô que incialmente possui apenas movimentos de ataque e proteção de faixa. Logo, este trabalho irá acrescentar decisões de defesa, onde o robô além da técnica de controle de Sistemas a eventos discretos, também irá possuir uma lógica *fuzzy* que será capaz de variar a potência de resposta do robô de acordo com as variações dos sinais de entrada dos novos sensores de ultrassom. O robô com a nova implementação possui a capacidade de desviar-se na direção contrária quando o adversário está atacando e também variar sua velocidade de acionamento com a atuação da lógica *fuzzy*. O sistema a eventos discretos também foi trabalhado para acrescentar movimentos de giro 360 graus e aprimorar os tempos de resposta do robô.

Palavras-chave: Sumô. Sistema de eventos discretos. Lógica *Fuzzy*. Automação. Robótica. Microcontrolador.

ABSTRACT

MARINHO, David Lima. Autonomous sumo robot improvement. 2017. 67f. Trabalho de Conclusão de Curso de Bacharelado em Engenharia de Computação - Universidade Tecnológica Federal do Paraná. Pato Branco, 2017.

The automation has brought to mankind a great advance, causing a meaningful impact to society and to the men's lives. It has provided new ways of processing information fast, including also ways of acting precisely and even ways to produce faster than usual. In general, it brought us comfort and evolution to the modern life style. Following this context, we propose a project, which works integrating hardware, sensors and software introducing ourselves into the automation area while working with a sumo robot. This introduction will help as a start point to understand some automation techniques. The autonomy of an autonomous robot is very important because it makes it independent e gives it a lot of precision using external signals, where it is possible to repeat movements continuously without the need of a break or even to accelerate manual processes, a pointing target within researchers since its first development. Following the path, this project will base itself in sensors integrated with controllers and actuators aiming the automation area development and introduction, while working with some robot improvements that initially has attack movements only, giving this work a mission to implement new techniques to complement its movements. To do that, it is implemented a Fuzzy logic, enhancing the robot to respond to new sensors. Using this technique, the robot is going to be able to operate with attack and defense movements, while it can choose the best velocity to each opponent's velocity and distance inputs.

Keywords: Sumo. Discrete event systems. Fuzzy logic. Automation. Robotics. Embedded System.

LISTA DE FIGURAS

FIGURA 1 - FUNÇÃO DE PERTINÊNCIA DO CONCEITO JOVEM EM DOIS CONTEXTOS. 20
FIGURA 2 - REPRESENTAÇÃO GRÁFICA DO PROCESSO DE FUZZIFICAÇÃO. 21
FIGURA 3 - REGRAS FUZZY 21
FIGURA 4: REPRESENTAÇÃO GRÁFICA DO PROCESSO DE DEFUZZIFICAÇÃO 22
FIGURA 5 - DOJO 24
FIGURA 6 - INTEGRAÇÃO DE HARDWARE 25
FIGURA 7 - FUNCIONAMENTO DO SENSOR ULTRASSÔNICO 26
FIGURA 8 - FUNCIONAMENTO DO SENSOR ULTRASSÔNICO 26
FIGURA 9 - MÓDULO COM SENSOR DE ULTRASSOM 27
FIGURA 10 - SENSOR ÓTICO 28
FIGURA 11 - MSP430G2553 29
FIGURA 12 - ESQUEMÁTICO DE UMA PONTE H 30
FIGURA 13 - IRS 2184 31
FIGURA 14 - CHASSI MECÂNICO 32
FIGURA 15 - DOJÔ DE SUMÔ 33
FIGURA 16 - ESQUEMÁTICO DO SENSOR DE FAIXA 35
FIGURA 17 - ESQUEMÁTICO DA CONEXÃO COM O SENSOR ULTRASSOM 36
FIGURA 18 - ESQUEMÁTICO DE CONEXÃO COM O SENSOR ÓTICO 36
FIGURA 19 - PLANTA DO SENSOR P1 (FRONTAL ESQUERDO) 39
FIGURA 20 - PLANTA DO SENSOR P2 (FRONTAL DIREITO) 39
FIGURA 21 - PLANTA DO SENSOR P3 (LATERAL ESQUERDO) 40
FIGURA 22 - PLANTA DO SENSOR P4 (LATERAL DIREITO) 40
FIGURA 23 - PLANTA DO SENSOR SF1 (FAIXA ESQUERDO) 41
FIGURA 24 - PLANTA DO SENSOR SF2 (FAIXA DIREITO) 41
FIGURA 25 - PLANTA DOS MOVIMENTOS 42
FIGURA 26 - ESPECIFICAÇÃO DE ATAQUE 43
FIGURA 27 - ESPECIFICAÇÃO DE DEFESA 43
FIGURA 28 - ESPECIFICAÇÃO DO SENSOR P3 44
FIGURA 29 - ESPECIFICAÇÃO DO SENSOR P4 44
FIGURA 30 - CONJUNTO DE PLANTAS DO SISTEMA 45
FIGURA 31 - CONJUNTO DE ESPECIFICAÇÕES DO SISTEMA 46

FIGURA 32 - FUNÇÃO DE PERTINÊNCIA (DISTÂNCIA) 47
FIGURA 33 - FUNÇÃO DE PERTINÊNCIA (VELOCIDADE) 47
FIGURA 34 - FUNÇÃO DE RESPOSTA DE SAÍDA 48
FIGURA 35 - ENTRADA DE 15CM 50
FIGURA 36 - ENTRADA DE 17CM/S 50
FIGURA 37 - COMBINAÇÕES DE REGRAS DE ENTRADA 51
FIGURA 38 - CIRCUITO DE CONTROLE 53

LISTA DE TABELAS

TABELA 1 - ESPECIFICAÇÕES DAS CATEGORIAS DO ROBÔ DE SUMÔ 23
TABELA 2 – ALFABETO DE EVENTOS NÃO CONTROLÁVEIS 38
TABELA 3 – ALFABETO DE EVENTOS CONTROLÁVEIS 38
TABELA 4 - CONJUNTO DE REGRAS 48

LISTA DE SIGLAS E ACRÔNIMOS

AGV	*Automated Guided Vehicle* (Veículos Guiados Automaticamente)
DARPA	*Defense Advanced Research Projects Agency* (Agência de Projetos de Pesquisa de Defesa Avançada)
DC	*Direct Current* (Corrente Contínua)
RAM	*Random access memory* (Memória de Acesso Randômico)
SED	Sistemas a Eventos Discretos
PWM	*Pulse Width Modulation* (Modulação de Largura de Pulso)

SUMÁRIO

INTRODUÇÃO **10**
1.1 CONSIDERAÇÕES INICIAIS 10
1.2 PROBLEMA 13
1.3 OBJETIVOS 13
1.3.1 Objetivo Geral 13
1.3.2 Objetivos Específicos 13
1.4 JUSTIFICATIVA 14
2 REFERENCIAL TEÓRICO **15**
2.1 Técnicas de controle para robôs móveis autônomos 15
2.1.1 Controle Supervisório de Sistemas a Eventos Discretos 15
2.1.2 Lógica Fuzzy 19
2.2 Competição de sumô de robôs 23
3 DESCRIÇÃO DOS MATERIAIS USADOS NO PROJETO **25**
3.1 Sensores de ultrassom HC-SR04 26
3.2 Sensores óticos 28
3.3 Microcontrolador MSP430G2553 28
3.5 Driver de acionamento IRS2184 29
3.4 Chassi mecânico 32
3.6 Dojô (Tatame) 33
4 PROJETO DE CONTROLE DE UM ROBÔ DE SUMÔ **34**
4.1 Projeto do sistema de sensoriamento e aquisição de dados 34
4.2 Projeto do sistema de acionamento 37
4.3 Projeto da estratégia de controle a eventos discretos 37
4.4 Projeto do sistema de controle de tempo contínuo 46
4.5 Integração dos dois sistemas anteriores e arquitetura de implementação 52
5 CONSIDERAÇÕES FINAIS **54**
6 REFERÊNCIAS **56**
APÊNDICE I – ESQUEMÁTICO DE CONTROLE **58**
APÊNDICE II – ESQUEMÁTICO DE POTÊNCIA **59**
APÊNDICE III – CÓDIGO DE CAPTURA DO SENSOR DE ULTRASSOM **60**
APÊNDICE IV – LÓGICA FUZZY **61**

1 INTRODUÇÃO

Este capítulo apresenta o porquê da necessidade de sistemas autônomos, as organizações que estimulam o desenvolvimento da robótica móvel e introduz as técnicas que serão utilizadas neste trabalho. Apresenta-se sobre a utilidade de veículos autônomos, além de citar algumas deficiências de um robô de sumô utilizado como objeto do trabalho e propostas de melhoria.

1.1 CONSIDERAÇÕES INICIAIS

O trabalho consiste em uma melhoria de um robô de sumô autônomo, o qual possui limitações de sensores e movimentos possuindo apenas uma técnica de controle para movimentação. As melhorias serão aplicadas para aumentar a inteligência do robô a partir de novos sensores e novas informações do meio externo, e também a partir de uma abordagem de controle híbrido, na qual a técnica de controle para sistemas a eventos discretos (SED) antes aplicada, irá se complementar com uma nova técnica controle de tempo contínuo, utilizando a lógica *fuzzy*. O SED se refere a um sistema cuja evolução dinâmica depende da ocorrência de eventos, podendo ter intervalos de tempo aleatórios (Montgomery, 2004).

Ao introduzir o tema de automação, pode ser visto que as tecnologias envolvendo sensores e atuadores para realizarem as mais diversas tarefas na indústria são utilizadas para otimizar processos e reduzir custos. Em um processo de fábrica existem muitas máquinas que utilizam sistemas embarcados. Estes ficam responsáveis pelo recebimento de informações do meio externo que são utilizadas para o controle e também para o tratamento de erros do processo, de acordo com requisitos bem definidos. Pode-se citar como exemplo prático os veículos guiados automaticamente (AGVs do inglês *Automated Guided Vehicle*), que são veículos autônomos no chão de fábrica ou até mesmo em hospitais. Também podemos citar como exemplos o carro do Uber e do Google que não precisam de motoristas. São eles responsáveis pelo transporte de peças, matérias primas ou medicamentos entre diferentes setores. Com o poder de compra e alta demanda de consumo da sociedade, as formas de produção foram evoluindo, necessitando de sistemas mais rápidos que interpretem o mundo externo e possam tomar decisões com uma margem de erro mínima.

A área da robótica móvel tem sido intensamente investida pelos setores mais importantes da economia. Nos EUA existe uma competição anual organizada pela DARPA (*Defense Advanced Research Projects Agency*), que incentiva pesquisadores e engenheiros para desenvolvimento de veículos autônomos. Entre as aplicações encontram-se veículos autônomos aéreos, com finalidades como o monitoramento para segurança e obtenção de estatísticas (CAZANGI, 2008).

O DARPA *Robotics Challenge* veio com o intuito de aprimorar veículos autônomos para ajudar os seres humanos no apoio a desastres naturais e desastres causados pelo próprio homem. O desafio distribui prêmios relevantes como incentivo ao desenvolvimento de tecnologia, e teve em 2007 como vencedor um veículo autônomo que circulou 96 km em pouco mais de 4 horas num circuito urbano, respeitando leis de trânsito e sem causar acidentes. O veículo não realizou nenhuma comunicação com o ser humano durante o percurso. Hoje, a DARPA investe em robôs humanoides e realizam competições na área todo ano oferecendo aos vencedores milhões de dólares em prêmios.

Os robôs móveis vão além, podendo ser citados no uso residencial, como por exemplo, o aspirador de pó autônomo que chegou a casa dos 2,5 milhões de vendas durante seis anos (CAZANGI, 2008). O uso também se dá para fins científicos como a exploração de novos planetas, recolhendo informações enquanto trafega por diferentes terrenos. Possibilita também o uso no já citado monitoramento aéreo e o controle veicular, que possibilita o controle de uma aeronave de grande porte que cada vez mais recebe melhorias para controle de estabilidade e de navegação.

Visando estimular o desenvolvimento da robótica móvel, no Brasil existe a competição organizada pela RoboCore, na qual entra a categoria de robô de sumô. A competição compreende um combate entre dois robôs, que tem por objetivo empurrar o adversário para fora do campo de batalha (Dojô). Além da nacional, outra competição bem conhecida é a RoboCup, que também tem como objetivo incentivos a pesquisa na área de robótica, sendo esta competição de nível internacional. No ano de 2014, o país sede foi o Brasil.

Para utilizar dos princípios da automação e da robótica este trabalho visa envolver essas áreas a fim de aperfeiçoar um protótipo de robô de sumô, introduzindo diferentes técnicas de inteligência, testando diferentes sensores e realizando uma análise da quantidade necessária de sinais de entrada para o controle do robô. O protótipo a ser utilizado foi parcialmente desenvolvido dentro de um projeto de extensão na UTFPR-PB,

porém ainda apresenta deficiências na lógica de locomoção, sensoriamento e sistema de acionamento.

O projeto aqui elaborado consistirá no conjunto de duas técnicas de controle. A primeira consiste no fato de que uma luta de sumô pode ser vista como um sistema a eventos discretos onde eventos assíncronos no tempo podem ocorrer, exigindo que uma ação seja tomada e ocorra uma migração de estado dentro do sistema. A segunda será um sistema que atuará em tempo contínuo com a lógica Fuzzy, sendo que este último atuará de maneira complementar ao primeiro, enriquecendo a lógica e dando mais flexibilidade ao robô.

Como as duas técnicas são sistemas, é importante entender o que é um sistema. Um sistema consiste na disposição das partes de um todo que são coordenadas entre si e que funcionam de maneira organizada com um objetivo geral a ser atingido (Montgomery, 2004).

Para a modelagem e o controle de sistemas a eventos discretos, uma alternativa é o uso de autômatos que possuem uma coleção de estados e transições que são responsáveis por definir um fluxo de ações. O conjunto dos autômatos será responsável por coordenar o comportamento do robô.

O robô de sumô precisa, a partir dos sinais de sensores, decidir ações de movimentos e acelerações dentro do dojô. A técnica de modelagem utilizando autômatos é uma das estratégias utilizadas, trabalhando com base em um sistema a eventos discretos, uma migração de estados no modelo da máquina de estados ocorre em resposta às bordas de subida e descida dos sinais dos sensores óticos e sensores de faixa, direcionando o robô para a ação de movimento a ser tomada. Após definida a primeira ação na transição do modelo a eventos discretos, será incluída uma dinâmica de tempo contínuo com a técnica de lógica Fuzzy, que é utilizada com um conjunto de equações e um sistema de fuzzificação/defuzificação, definindo se o robô deverá continuar com a ação inicial e definindo também a tensão a ser entregue aos motores, configurando o ciclo de trabalho da modulação de largura de pulso dos sinais (PWM). A entrada da lógica Fuzzy será a distância e a velocidade relativa do adversário, proveniente dos sinais dos sensores de ultrassom e a saída será o ciclo de trabalho para o PWM.

1.2 PROBLEMA

Na área de robótica o desafio é transformar muitas das ações do ser humano de forma autônoma e capaz de tomar decisões. Para ser autônomo, portanto, deve existir a capacidade de interpretar sinais de forma contínua e paralela, ou de forma discreta no tempo. A aquisição de sinais consiste em uma delicada ação onde nenhuma informação importante possa vir a ser desprezada ou perdida, além de que todas devem trabalhar de forma conjunta para responder precisamente aos requisitos. Desta forma, será dada autonomia ao robô de sumô aplicando a capacidade de interação com o campo de batalha e seu adversário, para que o robô saiba o que fazer em cada situação.

1.3 OBJETIVOS

A seguir estão o objetivo geral e os objetivos específicos deste trabalho.

1.3.1 Objetivo Geral

Aprimorar o hardware e o software de um robô de sumô aplicando técnicas de automação de sistemas a eventos discretos para uma resposta precisa e autônoma aos sinais recebidos via sensores liga/desliga, acrescentando uma ação de tempo contínuo que se baseia nas informações de sensores de ultrassom capaz de alimentar o sistema com dados de distância e velocidade.

1.3.2 Objetivos Específicos

- Desenvolver o projeto de recondicionamento do sistema de sensoriamento do robô;
- Desenvolver o projeto do sistema de acionamento do robô;
- Implementar uma lógica de locomoção com a teoria de controle supervisório para sistemas a eventos discretos;
- Implementar a lógica de locomoção utilizando técnicas de IA (Lógica Fuzzy) para controle contínuo de decisão dentro das ações definidas pelos eventos discretos;
- Analisar capacidades e deficiências dos métodos implementados e aprimorá-las;
- Analisar possíveis aprimoramentos nos autômatos da planta e restrições;

- Verificar possíveis alterações nos tipos e nas quantidades dos sensores.

1.4 JUSTIFICATIVA

O robô apresenta uma quantidade reduzida de opções de movimentos, além de também possuir uma sequência inicial simples de movimentos, limitando o seu campo de visão onde em sua arquitetura inicial conta com quatro sensores ópticos direcionados, dois na parte da frente do robô e dois nas laterais, dificultando assim localizar o adversário. O robô também incialmente possui um sistema para implementar uma resposta de ataque, quando há uma alteração de borda no sinal dos sensores óticos, sendo eficiente apenas em casos específicos.

O trabalho vem aumentando o direcionamento dos sensores ópticos através de pequenas modificações nas especificações dos autômatos, além de aumentar também as opções de movimento, acrescentando 4 sensores de ultrassom e aumentando as informações do meio externo para a tomada de decisões. Assim, o robô ficará mais flexível e contará com um sistema encontra/ataca e um sistema encontra/defende adicional, além de uma velocidade apropriada para cada momento. Para esta velocidade apropriada será utilizado um acionamento através de comando PWM, o que fará variar a tensão média de acionamento e para isso o circuito de potência necessitará ser baseado no chaveamento com *mosfets*. Esse consegue chavear rapidamente, permitindo a configuração de um universo maior de ciclos de trabalho dentro do PWM. Ao contrário do antigo circuito de relés que somente permitia o controle liga e desliga, sendo limitado dentro de um universo bem restrito de chaveamento físico.

2 REFERENCIAL TEÓRICO

2.1 TÉCNICAS DE CONTROLE PARA ROBÔS MÓVEIS AUTÔNOMOS

As técnicas utilizadas podem ser de controle a eventos discretos, os quais são utilizados para definir os estados e os comportamentos do robô diante aos sinais dos sensores de eventos discretos (bordas de subida e descida). Adicionalmente podem ser utilizadas técnicas de tempo contínuo, por exemplo, a lógica fuzzy, desenvolvido de forma a complementar o controle a eventos discretos, definindo a ação mais apropriada dentro da transição de tempo discreto de acordo com a resposta do sensor de tempo contínuo.

2.1.1 Controle Supervisório de Sistemas a Eventos Discretos

Um Sistema a Eventos Discretos (SED) possui eventos que ocorrem de forma assíncrona sem um intervalo de tempo conhecido, ao contrário de um sistema contínuo onde existem eventos ocorrendo de maneira contínua no tempo (CASSANDRAS; LAFORTUNE, 2008). Um sistema de eventos discretos existe quando há, em qualquer situação, leituras abruptas no sistema, ou seja, ocorrências esparsas no tempo, como um robô de manufatura, redes de computadores e comunicações em geral (RAMADGE; WONHAM, 1989).

Tais sistemas podem ser modelados por autômatos, o qual é composto por alfabeto de eventos, conjunto de estados e transições. Os autômatos finitos podem ser formalizados como uma quíntupla $(\sum, Q, q^0, Q^w, \rightarrow)$ onde $\sum$ é o alfabeto finito de eventos, Q é o conjunto finito de estados, $q^0 \in Q$ é o estado inicial, $Q^w \subseteq Q$ é o

subconjunto de estados marcados e por último o símbolo $(\rightarrow) \subseteq Q \times \sum \times Q$ que representa a relação de transição dos estados (HOPCROFT; ULLMAN, 1979).

Eventos são a base da dinâmica deste tipo de sistemas. São eles que irão demonstrar o fluxo que ocorre durante diferentes estados do problema. A cadeia de eventos representa uma sequência finita de eventos que pertencem ao alfabeto, podendo ser esta cadeia vazia e denotada por ε que corresponde a uma sequência vazia. Ainda, se tem o conjunto de todas as sequências possíveis em $\sum$ que se denota por $\sum^*$, contendo inclusive a palavra vazia. A linguagem é responsável por filtrar as combinações em $\sum^*$ para apenas as cadeias reconhecidas nos SEDs em questão. Portanto, a linguagem limita as combinações de palavras possíveis em $\sum^*$ para as combinações que tem algum significado no sistema, definindo as cadeias em um dicionário particular.

Um importante conceito e muito utilizado ao modelar as plantas e especificações de um sistema a eventos discretos mais complexo em componentes menores, é o conceito de composição síncrona de autômatos. Ele se baseia na composição de dois ou mais autômatos, tal que $G1 \| G2 = (\sum_A U \sum_B, Q_A \times Q_B, (q_A^0, q_B^0), Q_A^w \times Q_B^w, \rightarrow)$, onde $\sum_A U \sum_B$ representa a união entre os alfabetos, podendo ser ou não compartilhados, $Q_A \times Q_B$ é o conjunto composto de estados de ambos os autômatos, (q_A^0, q_B^0) o estado inicial da

composição, $Q_A{}^w \times Q_B{}^w \subseteq Q$ são os conjuntos marcados e $\rightarrow \subseteq Q \times \sum \times Q$, representa a

relação de transição de estados. A função de transição de estados é dada por:

- $(qa,qb) \rightarrow (q'a,q'b)$ se $\sigma \in \sum_A \cap \sum_B$

- $(qa,qb) \rightarrow (q'a,qb)$ se $\sigma \in \sum_A / \sum_B$

- $(qa,qb) \rightarrow (qa,q'b)$ se $\sigma \in \sum_B / \sum_A$

A composição é responsável por compor um novo autômato, juntando os autômatos modulares e, sempre forçando a ocorrência simultânea dos eventos compartilhados para que todos os comportamentos individuais sejam representados em uma sequência única do sistema para todos os modelos. Durante a composição de autômatos, deve-se levar em consideração os eventos compartilhados, onde o sistema trabalha em todos os autômatos de forma simultânea, levando em consideração a maneira assíncrona, onde apenas alguns modelos que reconhecem o evento evoluem de estado (RAMADGE; WONHAM, 1989).

Para a construção dos autômatos dentro de um SED é necessário que se faça uma análise do objetivo geral do problema, criando estados e eventos para cada ação esperada. Pode-se modelar o SED utilizando-se autômatos que representam plantas e especificações. Ainda é necessário avaliar que sistemas complexos podem ter uma

explosão de estados, pois a quantidade de estados aumenta exponencialmente com o número de componentes do sistema, sendo os componentes possíveis subsistemas do processo (RAMADGE; WONHAM, 1989).

A composição das especificações e da planta rege o comportamento esperado do sistema onde as especificações atuam sobre as plantas. Este formato pode impedir que eventos na planta deixem de acontecer em determinado momento. Porém, alguns eventos podem não ser controlados, como, por exemplo, a quebra de um equipamento, ou sinal externo de um sensor. Frente a esta questão, a teoria de controle supervisório particiona o conjunto de eventos entre $\sum = \sum_c U \sum_u$, onde $\sum_c$ é o conjunto dos eventos controláveis e $\sum_u$ é o conjunto dos eventos não controláveis.

Assim, definem-se os eventos, no qual os controláveis são inícios de tarefas mecânicas, inícios de ações numa determinada transição, enquanto os não controláveis são sinais externos de sensores, términos de máquina, defeitos entre outros que não se tem como se evitar. (RAMADGE; WONHAM, 1989).

Para o controle de SEDs inicialmente destacam-se dois tipos de controle supervisório: o monolítico centralizado e o supervisório descentralizado. No primeiro caso tem-se apenas um supervisor, responsável globalmente pelo controle de todo o sistema para realizar o comportamento desejado. Na abordagem centralizada deve-se preocupar mais com a explosão de estados para sistemas complexos, pois esta solução acumula total responsabilidade e deve prever todas as possíveis transições do sistema. Para diminuir a ocorrência deste problema, a abordagem descentralizada aparece como solução. Uma das possíveis abordagens descentralizadas baseia-se em controles locais, onde projeta-se um supervisor local para cada especificação com as plantas nela envolvidas. O conjunto de supervisores é responsável pelo controle global, que anteriormente era realizado por apenas um supervisor (RAMADGE; WONHAM, 1989).

Neste trabalho será utilizado um controle monolítico, e para isto é necessário um agente de controle que atue de acordo com um conjunto de restrições. As restrições são as especificações do modelo, que atuam de modo permissivo diante da ação de controle. Esta ação de controle é determinada por um supervisor, o qual possui o objetivo de habilitar e desabilitar os eventos da planta, porém ele é apenas capaz de desabilitar os eventos controláveis. O supervisor interage com a planta observando os eventos ocorridos e definindo e habilitando as possíveis futuras ocorrências.

Ao possuir um supervisor, é possível realizar a implementação em um micro controlador, onde ao ocorrer um evento uma ação de controle poderá ser gerada, neste

caso o chamado modelo de estados finitos de Mealy. Existe também a ação de controle gerado no estado, ou seja, após ocorrer o evento e a migração de estados for concluída a ação é realizada, assim utilizando-se do conceito do modelo de estados finitos de Moore (AVNUR, 1990).

2.1.2 Lógica Fuzzy

Um sistema especialista é, um conjunto de elementos que formam o conhecimento e solucionam problemas onde especialistas seriam capazes de resolver. O sistema consiste num conjunto de regras, formadas por equações lógicas que simulam o raciocínio lógico do ser humano (FEIGENBAUN, 1977).

Dentro do contexto especialista, a lógica Fuzzy vem com o propósito de criar uma lógica multivalente de frente a lógica clássica bivalente. Onde a lógica multivalente define um grau de pertinência na resposta, enquanto a bivalente tem como saída uma resposta verdadeiro ou falso. Ainda a respeito do grau de pertinência, tem-se uma saída proporcional a um determinado conjunto de entradas e definições, resultando em escalas variáveis entre o grupo de verdadeiro e o grupo de falso (BOYER, 1996).

Enquanto na lógica clássica, um determinado elemento pertence ou não a um conjunto, na lógica fuzzy um elemento pode pertencer totalmente ou parcialmente a um conjunto, ou ainda pertencer a dois conjuntos ao mesmo tempo. Assim, em lógica fuzzy cria-se o conceito de “grau de pertinência”, conceito que engloba à lógica clássica, pois se um elemento pertence totalmente ao conjunto seria grau 1 em lógica fuzzy e $\in$ em lógica clássica; caso o elemento não pertence ao conjunto seria 0 em lógica fuzzy e $\notin$ em lógica clássica. Para representar a forma gradativa de pertinência a um conjunto são utilizadas as funções de pertinência. Essas funções podem ter diversos formatos: S, Z, seno, triangular, trapezoidal, impulso dentre outras. Na Figura 1 são apresentadas funções de formato trapezoidal.

No caso do robô de sumô tem-se entradas que definem se o adversário está perto, muito perto, não tão perto e isso passa a ter um grau de pertinência. Porém, essas chamadas funções de pertinências podem variar dependendo do contexto, sendo que o que antes era perto pode se tornar mais longe dentro de algum outro contexto (ZIMMERMANN, 1985).

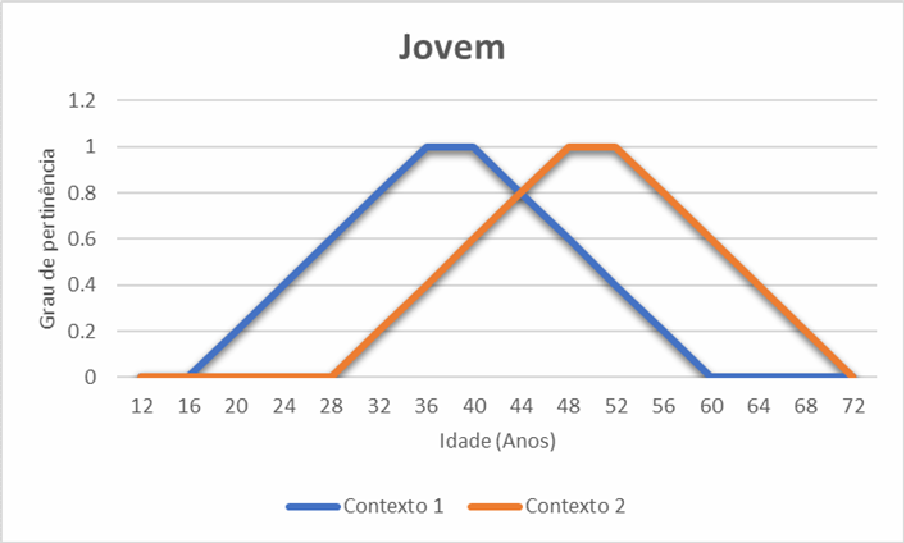


Figura 1 - Função de pertinência do conceito Jovem em dois contextos.

Como se pode observar na Figura 1: o contexto 1 refere-se a uma análise do conjunto "Jovem" no qual se pode pertencer de 16 até 60 anos, pertencendo totalmente de 36 á 40 anos; fora desse intervalo a pertinência diminui gradativamente, isto significa que em outras idades se é menos jovem. Tem-se um segundo contexto no qual ao conjunto "Jovem" pertencem os que têm de 28 até 72 anos, pertencendo totalmente de 48 á 52 anos. São apresentados dois contextos para exemplificar que se em um contexto 36 anos pertencia totalmente ao conceito "Jovem", em um outro contexto ele pertence menos ao conceito "Jovem".

O processo de raciocínio fuzzy como um todo consiste de funções de pertinência e um conjunto de regras, onde cada função representa um conjunto com grau de pertinência variável. As regras combinam todas as funções de entrada e as associam com as funções de saída. Comumente utilizam-se como entradas funções triangulares ou trapezoidais e como saída funções impulso. Isto deve-se tanto à facilidade de implementação quanto ao pouco gasto computacional quando se realiza o processamento de fuzzificação e defuzzificação.

O processo de fuzzificação refere-se à transformação de uma entrada do mundo real denominado valor "crisp" através da função de pertinência a um valor fuzzy denominado "grau de pertinência". Na Figura 2 apresenta-se o processo de fuzzificação do valor "crisp" 29,5 graus para o conceito Temperatura que possui três funções de pertinência e 38,5% para o conceito Umidade que também possui três funções de pertinência.

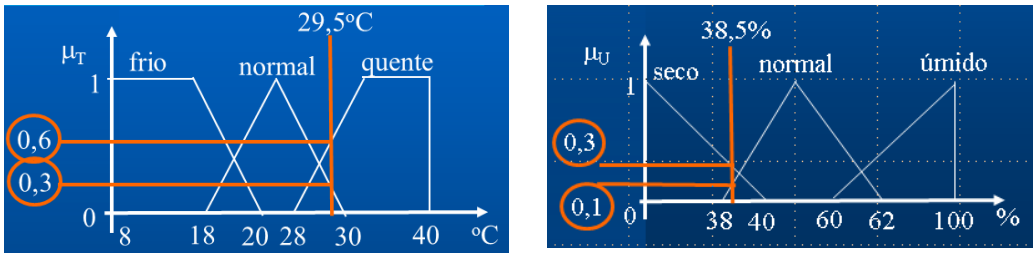


Figura 2 - Representação gráfica do processo de fuzzificação.

A inferência fuzzy pode ser feita através da aplicação dos operadores fuzzy mínimo (e) e máximo (ou), denominado de Min-Max. É aplicada aos antecedentes da regra o operador mínimo e seu resultado e dado como grau de pertinência ao consequente. Se há dois ou mais consequentes iguais, aplica-se o operador máximo, entregando para a saída do processo de inferência o consequente com o maior grau de pertinência. As regras fuzzy podem ser apresentadas em forma de tabela, na Figura 3 apresenta as regras fuzzy que tem como entradas Temperatura (Frio, Normal e Quente) e Umidade (Seco, Normal e Úmido), ambas são os antecedentes da regra. E a saída Ação (Curto, Médio, Longo) que é o consequente da regra. Exemplificando o processo de inferência, sejam as entradas: Temperatura 29,5 graus e Umidade 38,5%, as regras envolvidas são as que estão na área sombreada.

		Umidade		
		Seco	Normal	Úmido
Temperatura	Frio	Médio	Médio	Curto
	Normal	Médio	Médio	Curto
	Quente	Longo	Médio	Curto

Figura 3 - Regras fuzzy

As regras podem ser escritas como:

Se T Normal (μ_N=0,3) e U Seco (μ_S=0,3) então A Médio ($\mu_{M=}$0,3)

Se T Normal (μ_N =0,3) e U Normal (μ_N =0,1) então A Médio (μ_M=0,1)

Se T Quente (μ_Q=0,6) e U Seco (μ_S =0,3) então A Longo (μ_L=0,3)

Se T Quente (μ_Q=0,6) e U Normal (μ_N =0,1) então A Médio (μ_M=0,1)

As saídas do processo de inferência que é enviada à etapa de defuzzificação são a Médio (μ_M=0,3) e a Longo (μ_L=0,3).

Sendo que:

T = Temperatura,

U = Umidade,

μ_N = Grau de pertinência da função normal de temperatura,

μ_Q = Grau de pertinência da função quente de temperatura,

μ_S = Grau de pertinência da função seco de umidade,

μ_N = Grau de pertinência da função normal de umidade,

μ_M = Grau de pertinência da função médio de saída,

μ_L = Grau de pertinência da função longo de saída.

No processo de defuzzificação é realizada a transformação inversa, isto é, o número ou números fuzzy do processo de inferência devem ser transformados a um número "crisp" de saída. Como diferença ao processo de fuzzificação, a defuzzificação não utiliza apenas funções de pertinência, mas, métodos de defuzzificação. Dentre os métodos podem ser mencionados o Método do Centro de Gravidade (Center of Gravity – COG) e o Método do Máximo Valor. O mais usado é o COG, na Figura 4 apresenta-se o cálculo da saída utilizando esse método para funções de pertinência do tipo impulso seguindo a Equação 1:

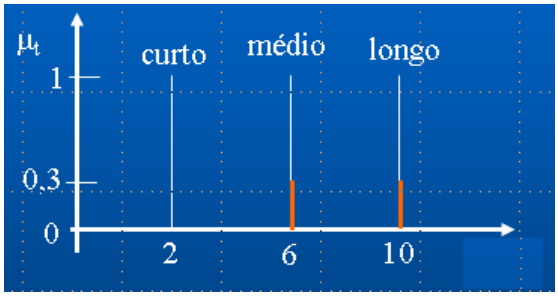


Figura 4: Representação gráfica do processo de defuzzificação

Equação 1 – Centro de gravidade

$$COG = \frac{\sum_i (\mu_i) * (I_i)}{\sum_i \mu_i}$$

$$COG = \frac{(0,3 * 6) + (0,3 * 10)}{0,3 + 0,3} = 8$$

Sendo que na Equação 1, μ_i é o grau de pertinência à função de saída (consequente da regra de inferência) e I_i é o valor de saída crisp da função impulso.

Assim, realizando a defuzzificação pode ser obtida a saída, a qual voltará a apresentar um valor dentro do universo de discurso pré-determinado de operação do sistema especialista fuzzy (SHAW, SIMÕES, 1999).

2.2 Competição de sumô de robôs

A competição brasileira organizada pela RoboCore e as internacionais como a RoboCup possuem uma diversidade de categorias de robôs. Robôs seguidores de linha, robôs de destruição, hockey, trekking, desafios de inteligência e o abordado neste trabalho, os robôs de sumô. As competições visam o crescimento profissional da área e o incentivo às pesquisas (ROBOCORE, 2015).

O desafio de sumô, possui seis categorias na Robocore, sendo o sumô para robôs de 3kg na categoria autônomo que será o abordado neste trabalho. As regras do combate são bem específicas, sendo algumas delas: os robôs não podem ultrapassar o peso da categoria, o combate é realizado por apenas dois robôs e o tempo da partida é de 3 minutos. Os detalhes para cada categoria podem ser observados na Tabela 1 (ROBOCORE, 2015).

Tabela 1 - Especificações das categorias do robô de sumô

Classe	Peso	Altura	Largura	Comprimento
3Kg Sumô	3000 g	Ilimitada	20 cm	20 cm
Mini Sumô	500g	Ilimitada	10 cm	10 cm
Micro Sumô	100g	5 cm	5 cm	5 cm
Nano Sumô	25 g	2.5 cm	2.5 cm	2.5 cm
Lego Sumô	1000 g	Ilimitada	15.2 cm	15.2 cm
Humanoide	3000 g	50 cm	20 cm	20 cm

Fonte: RoboCore

Seguindo as modalidades anteriormente estabelecidas, existem as opções: autônomo e radio-controlado. Para as categorias de menor porte todos deverão ser autônomos e para as categorias de 3000g poderão ser tanto autônomos como radio-controlados. Neste trabalho o foco será o robô autônomo de 3kg. Os robôs são colocados

dentro de um campo de batalha em formato de uma circunferência de diâmetro de 150 cm (Dojô) (Figura 5). Os robôs são inicialmente colocados em paralelo, em direções opostas e o início da partida se dá quando autorizado pelo juiz pelo acionamento dos robôs. Em seguida o robô deve permanecer imóvel durante 5 segundos para a saída dos membros das equipes do dojô e assim dão início ao combate. As pontuações acontecem quando um robô tocar o lado de fora do dojô por conta própria ou quando forçado legalmente pelo adversário (ROBOCORE, 2015).

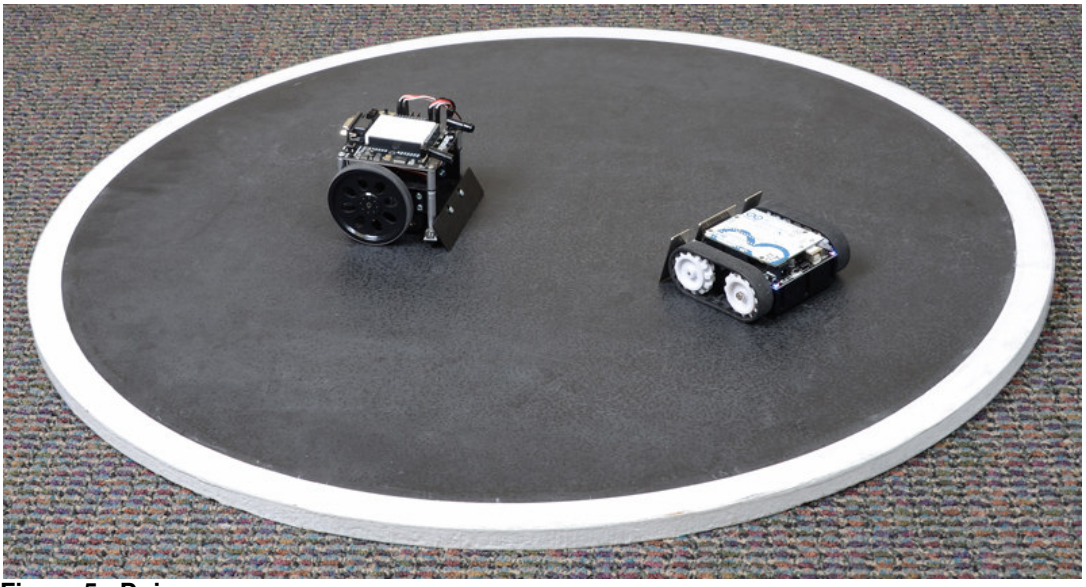


Figura 5 - Dojo
Fonte: Pololu (2016).

3 DESCRIÇÃO DOS MATERIAIS USADOS NO PROJETO

O sistema se baseia em uma central de controle sendo representada pelo microcontrolador MSP430G2553, fabricado pela *Texas Instruments*, que recebe todos os sinais do meio externo advindos dos sensores, enquanto este é responsável também por habilitar os sensores de ultrassom com o sinal PWM para o trigger do sensor. O microcontrolador, por meio de interrupções de porta e interrupções de captura, identifica as bordas de entrada, sendo capaz de trabalhar nas lógicas propostas de eventos discretos e lógica Fuzzy para captar os sinais necessários. Com o sistema de controle alimentado e as decisões tomadas, o microcontrolador emite os sinais para os *drivers* da ponte H, alimentando o circuito de potência e consequentemente acionando os motores. A Figura 6 mostra a integração do hardware.

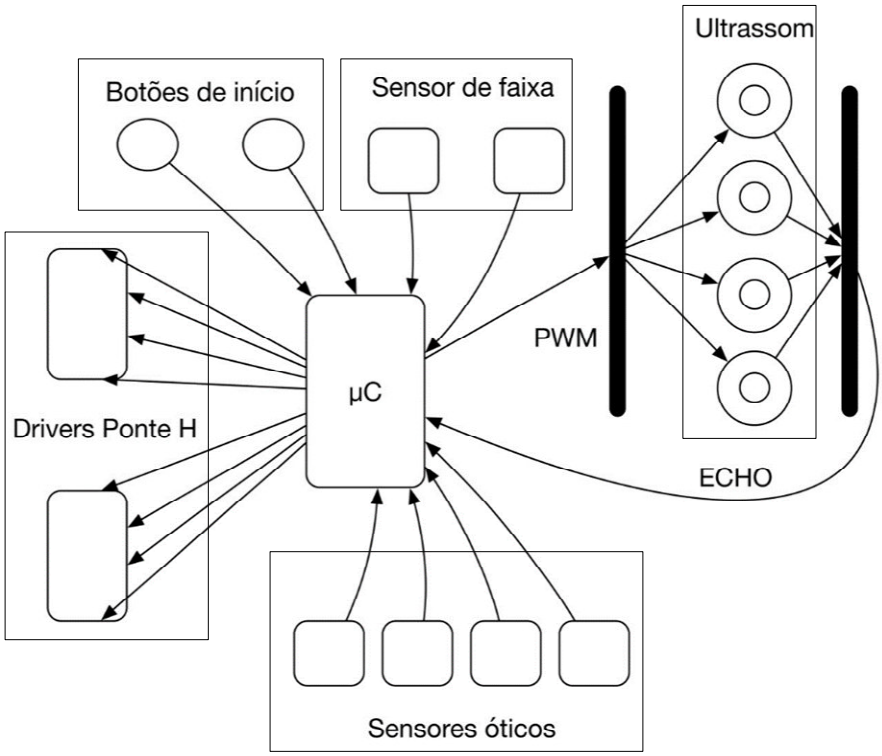


Figura 6 - Integração de hardware

3.1 Sensores de ultrassom HC-SR04

Sensores ultrassônicos de medição de distância fornecem informação da posição de um objeto em repouso ou em movimento (BAUMER, 2016, p.1). Os comprimentos das vibrações ultrassônicas emitidas devem ser de tal extensão que possam refletir no objeto, conforme a Figura 7, podendo ser captadas por um sensor colocado em posição apropriada.

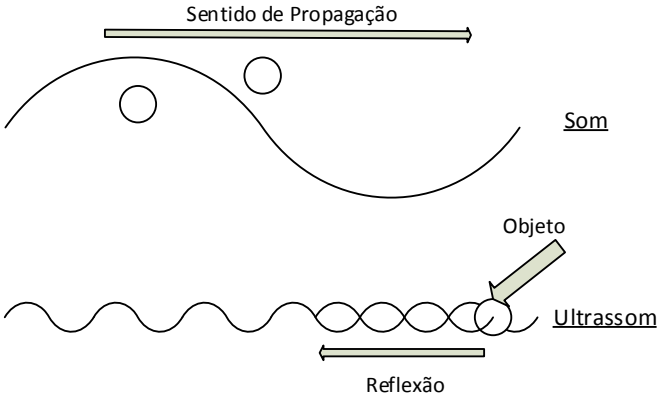


Figura 7 - Funcionamento do sensor ultrassônico
Fonte: Braga 2012.

O comprimento de onda usado e a frequência são muito importantes nesse tipo de sensor, pois ele determina as dimensões mínimas do objeto que pode ser detectado. Os módulos ultrassônicos são, usualmente, formados por um emissor e um receptor, conforme a Figura 8.

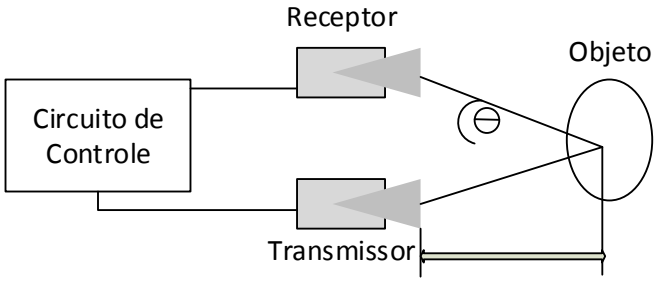


Figura 8 - Funcionamento do sensor ultrassônico
Fonte: Braga 2012.

O posicionamento correto e a observação de eventuais fontes capazes de interferir no funcionamento são fundamentais para se obter o bom desempenho de um sistema.

Por isso é necessário cuidado com reflexões indevidas por conta do tipo de material, região limite de detecção do sensor e o fenômeno da reverberação, que ocorre quando um pulso ultrassônico é aplicado a um objeto e este, ao absorver essa energia, vibra na mesma frequência, resultando em um pulso refletido de duração maior do que a do pulso transmitido (BRAGA, 2012, p.1).

Os módulos HC-SR04 com sensores de ultrassom são capazes de gerar sinais de saída conforme seu sinal de ultrassom emitido. Os módulos são acionados com um PWM com ciclo de trabalho de 0.016%, mantendo a onda 10µs em nível alto dentro de um período de onda de 60ms. A cada acionamento, o módulo emite uma onda de ultrassom e sua resposta sobe para nível lógico 1. Assim que o módulo recebe o som emitido seu nível lógico de saída se torna 0, sendo possível extrair o tempo do sinal em nível lógico 1 e consequentemente a distância do adversário. A Figura 9 representa o módulo e seus componentes.

Figura 9 - Módulo com sensor de ultrassom
Fonte: http://www.micropik.com/PDF/HCSR04.pdf (2016).

O módulo opera com 5 volts, uma corrente de trabalho de 15ma e frequência de 40Hz. Possui um máximo alcance de 4 metros e um mínimo de 2cm. Para seu acionamento como citado anteriormente, necessita de uma trigger de 10µs em nível alto.

3.2 Sensores óticos

Para identificar a presença do robô adversário foi necessário um sensor mais preciso de identificação em relação a sensores de ultrassom. Para esta necessidade o sensor ótico apresenta uma resposta com menos ruído. Eles funcionam a partir de uma emissão de uma onda infravermelha, elevando a sua saída para nível lógico 1 caso receba em seu receptor infravermelho a reflexão do sinal emitido. O sensor utilizado foi o modelo BA2M-DDT da Autonics e pode ser observado na Figura 10. O sensor tem alcance de até 2 metros de distância, opera com no mínimo 12 volts e no máximo 24 volts. Possui um tempo de resposta de aproximadamente 1ms e utiliza um Led infravermelho como fonte de luz.

Figura 10 - Sensor ótico
Fonte: http://www.autonicsonline.com/product/product&product_id=686 (2016).

3.3 Microcontrolador MSP430G2553

O microcontrolador MPS430G2553 (Figura 11) é fabricado pela Texas Instruments e possui a capacidade de memória de 512 bytes de RAM, um clock de 16MHz, 16Kbytes de memória não volátil e conta com um periférico temporizador muito utilizado no projeto. Os temporizadores são utilizados no formato da própria função temporizador, as quais realizam interrupções de acordo com a definição de sua frequência de clock e na configuração de captura, realizando interrupções de acordo as bordas de subida e descida no pino definido de entrada.

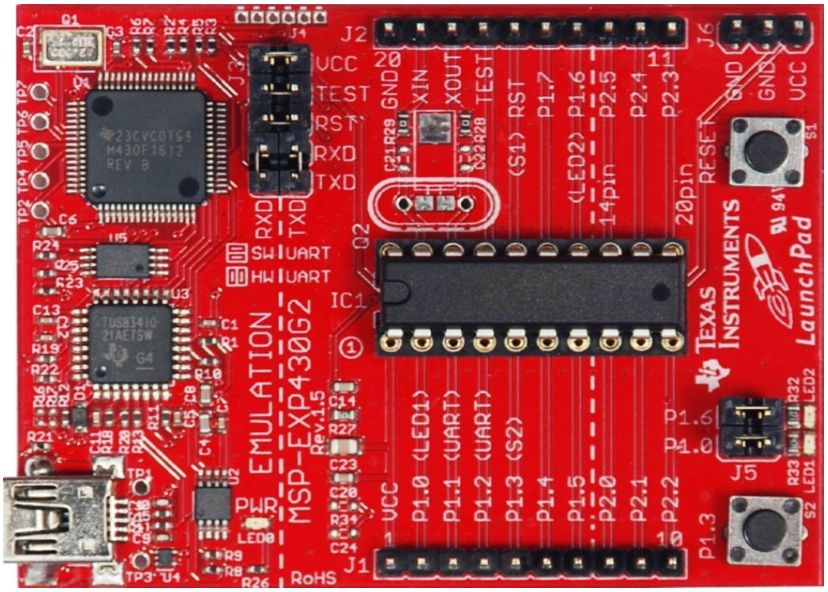


Figura 11 - MSP430G2553
Fonte: http://www.ti.com/tool/msp-exp430g2 (2016).

3.5 Driver de acionamento IRS2184

Pode-se definir uma Ponte H como um circuito eletrônico de potência que permite o controle de velocidade e direção, que serve para inverter a polaridade de uma carga sem a necessidade de utilizar uma fonte simétrica (SIEBEN, 2003, p.2).

Quando um microcontrolador fornece as instruções para uma carga, ele não possui a capacidade de energia necessária para alimentar esta carga, como, por exemplo, um motor. A ponte H faz o papel de intermediário entre o controle e a carga, sendo ela capaz de operar como um conjunto de chaves que permitem a circulação de corrente para fornecer a energia da fonte do circuito necessária de acordo com o comando do controlador.

Uma ponte H simplificada pode ser vista na Figura 12. É um circuito composto de 4 chaves S1, S2, S3 e S4, normalmente transistores, formando a letra H, que dá o nome ao arranjo eletrônico. Controlando-se essas chaves de forma individual, a corrente que passa pela carga pode ser variada, controlando-se as direções e intensidades de acionamento (MALVINO,2016,p.502).

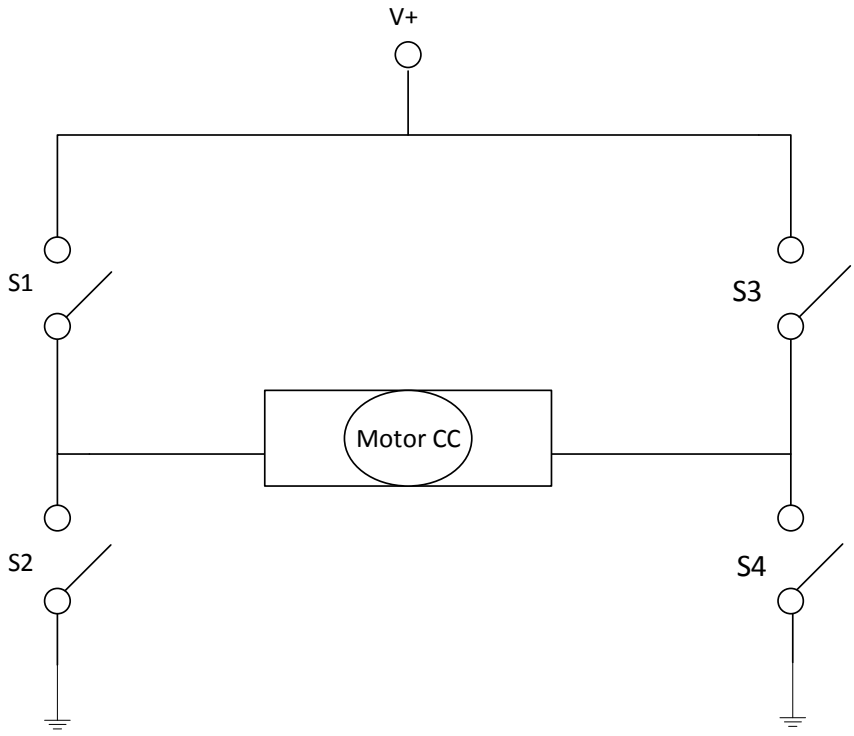


Figura 12 - Esquemático de uma ponte H
Fonte: Malvino (2016).

Pensando-se na carga como um motor, ao se fechar S1 e S4 tem-se o motor rodando em um sentido. Já com S3 e S2 acionadas, o sentido do fluxo da corrente sobre o motor é invertido, fazendo com que a rotação do motor também se inverta. Também pode-se acionar S1 e S3 ou S2 e S4 para o freio do motor.

Todavia, deve-se atentar para alguns acionamentos perigosos para o circuito, como S1 e S2 ou S3 e S4 ou até as quatro chaves ao mesmo tempo. Nestes acionamentos ocorre o efeito de *shoot-trough*. Este efeito representa um curto quando ambas as chaves de um mesmo lado são fechadas de forma simultânea. Para isso o chaveamento deve ser feito com uma temporização chamada tempo morto, dando tempo suficiente para uma das chaves se fechar e a outra se abrir, eliminando o curto.

A Ponte H foi projetada para ser capaz de receber os sinais de controle e ser capaz de variar a velocidade e direção do robô. Para isso ela conta com quatro drivers de acionamento IRS2184 e 8 MOSFETS em sua essência. Para outros fins de isolamento entre o circuito de controle e o circuito de potência são utilizados 6 optoacopladores, sendo estes responsáveis por proteger o circuito de controle de algum eventual pico de corrente.

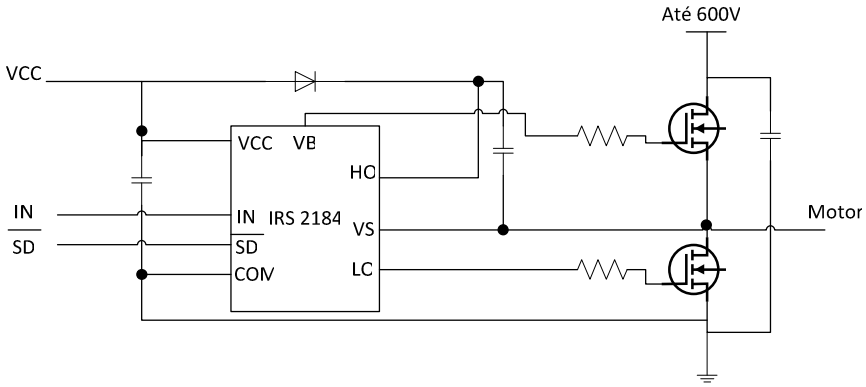


Figura 13 - IRS 2184
Fonte:
http://www.infineon.com/dgdl/ir2184.pdf?fileId=5546d462533600a4015355c955e616d4 (2016).

A Figura 13 representa a conexão de um CI do driver a dois mosfets do circuito. O sistema é controlado pelas entradas IN e SD, as quais coordenarão o sentido e movimento do motor, acionados pelo sistema de controle no microcontrolador. Para o sistema completo foram utilizados 4 drivers para acionar os 8 mosfets ao mesmo instante, sendo o movimento sincronizado e trabalhado com 4 entradas IN e quatro entradas SD. Ainda é possível analisar o comportamento do conjunto na Tabela 2, na qual o X representa qualquer entrada 0 ou 1.

Entradas				Saída do motor
Driver 1		Driver 2		
IN	$\overline{SD}$	IN	$\overline{SD}$	
X	0	X	0	Rotor Livre
1	PWM	0	1	Sentido 1
0	1	1	PWM	Sentido 2
1	1	1	1	Freio Motor
0	1	0	1	

Tabela 2 - Tabela verdade IRS2184

3.4 Chassi mecânico

O chassi previamente desenvolvido na versão inicial do robô foi projetado para receber 6 motores de 5 watts de potência, além de contar com a conexão para 6 rodas e com a parte superior do robô, onde encontra-se os circuitos de controle e os sensores. Também conta com uma chapa frontal, utilizada para retirar a aderência do adversário ao atacá-lo. Pode-se observar o chassi projetado no *Solid Works* na Figura 14.

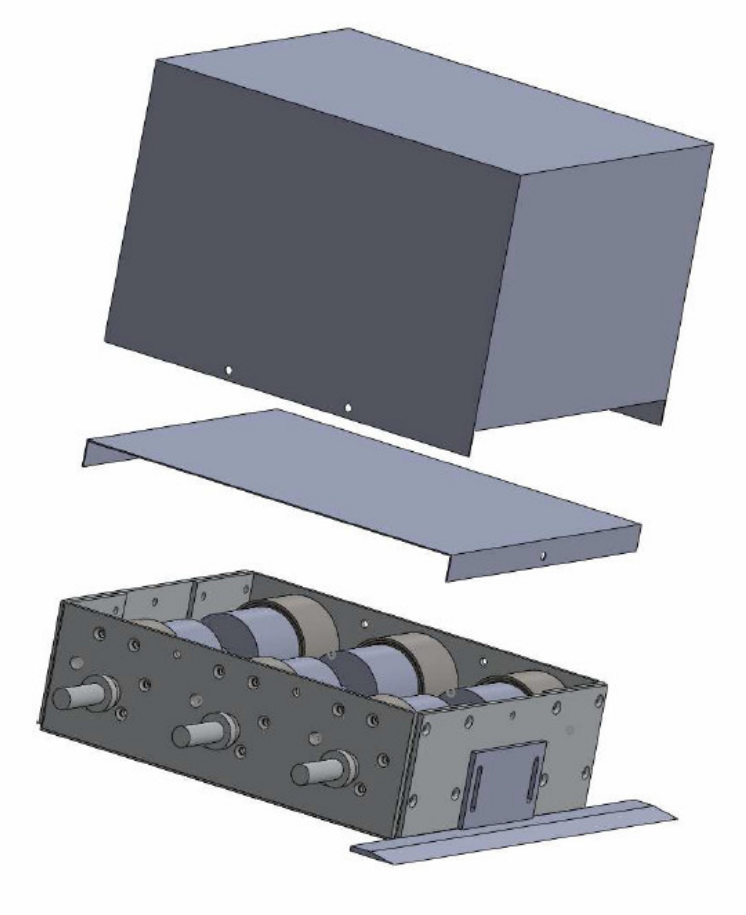


Figura 14 - Chassi mecânico

3.6 Dojô (Tatame)

O dojô é o espaço onde a luta entre os robôs acontece. Este tatame na competição da Robocore é constituído de material metálico, já que o regulamento permite a utilização de imãs na parte inferior do robô, dando a eles maior fixação ao solo. Pode-se observar as características do dojô na Figura 15.

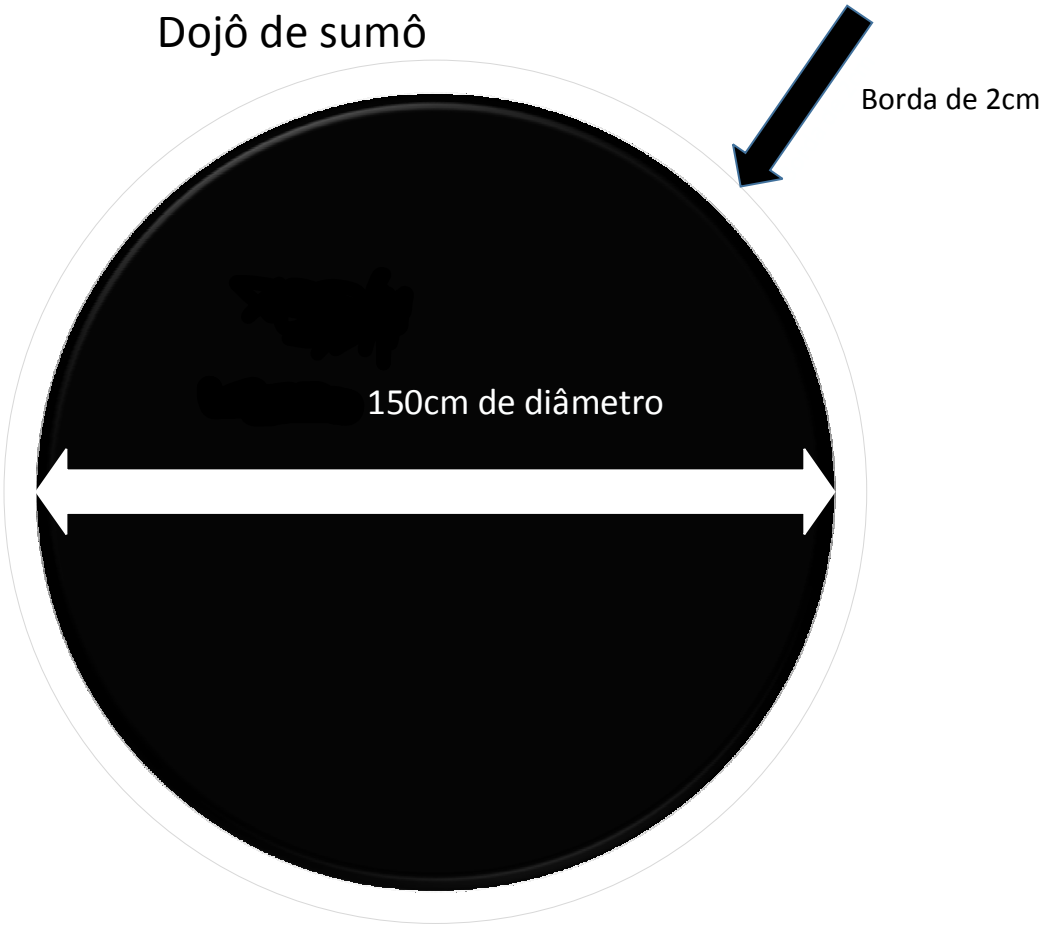


Figura 15 - Dojô de sumô

4 PROJETO DE CONTROLE DE UM ROBÔ DE SUMÔ

Realizado o levantamento do referencial teórico, foi iniciada a implantação dos sensores de ultrassom e dos sensores óticos, de forma a habilitá-los da maneira correta para gerar o sinal de presença, a distância e velocidade do oponente e a detecção de faixa, de forma que ambos os sistemas pudessem trabalhar em conjunto.

Assim, inicialmente foi realizado um projeto do sistema de sensoriamento, para a aquisição das informações do mundo externo, para depois entrar no projeto de controle e por fim no sistema de acionamento.

Para o desenvolvimento da primeira estratégia de locomoção, no projeto de controle, foram retrabalhados os modelos com autômatos referentes a cada sensor e especificações do sistema, de forma que fosse possível uma limpeza de redundâncias e desenvolvido um novo movimento para o robô. Com os novos autômatos da planta e das especificações foi gerado um supervisor responsável pelo controle do sistema e a partir dele gerado um código equivalente para uso no microcontrolador MSP430G2553. Após o aprimoramento dos autômatos, foi adicionada a técnica de lógica Fuzzy, capaz de acrescentar decisões de tempo contínuo dentro das ações definidas pelo sistema de tempo discreto. Esta nova lógica recebe os sinais dos sensores de tempo contínuo e utiliza informações como velocidade e distância para promover uma resposta de ataque ou defesa e definir a velocidade necessária do movimento. Deficiências e melhorias para cada sistema foram analisadas e, finalmente, realizada uma comparação de eficiência entre as técnicas atribuídas.

4.1 Projeto do sistema de sensoriamento e aquisição de dados

O robô possui dois sensores de faixa, os quais foram desenvolvidos utilizando dois LDR (do inglês *Light Dependent Resistor*), para cada lado na parte frontal do robô. Cada LDR possui uma alimentação de 7,5 volts e um resistor para regular a corrente de entrada e não danificar o LED e o LDR. Na saída do LDR está conectado o CI LM358N que atua como um comparador, que é alimentado com 7,5 volts regulados por uma resistência variável capaz de configurar a tensão de referência a ser utilizada para diferenciar a faixa de fronteira branca do chão preto do dojô. A Figura 16 mostra o esquemático do sensor de faixa.

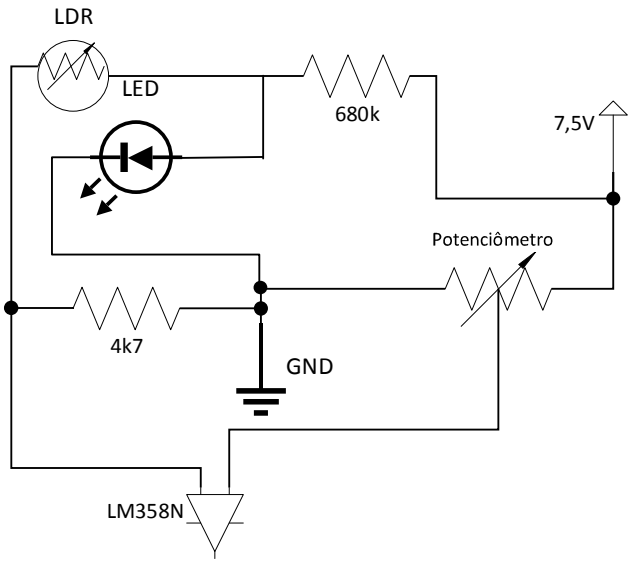


Figura 16 - Esquemático do sensor de faixa

Além do sensor de faixa, foi utilizado o sensor de ultrassom. Para utilizá-lo foi preciso amplificar o sinal de acionamento que vem do microcontrolador, pois este é capaz de ter em sua saída apenas 3,6 volts, ao invés dos 5 volts necessários. Também, para regular a saída *Echo* do sensor de 5 volts para 3,6 volts em processo contrário, foi utilizado um divisor de tensão, o que pode observar na Figura 17.

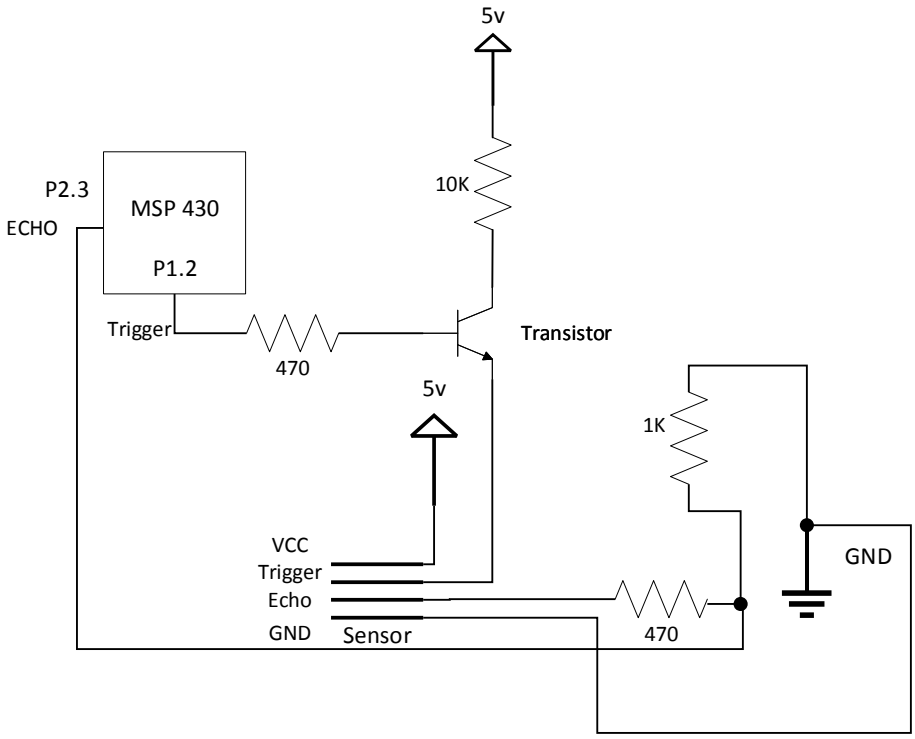


Figura 17 - Esquemático da conexão com o sensor ultrassom

Para o sistema a eventos discretos, além dos sensores de faixa, também foram utilizados os sensores óticos, responsáveis também por acionar os eventos da máquina de estados. O sensor é alimentado com 15 volts e seu sinal de saída, portanto, tem 15 volts também. Para adaptar a tensão ao microcontrolador para 3.6 volts foi utilizado um divisor de tensão apenas. Na Figura 18 está o circuito de conexão com o sensor ótico.

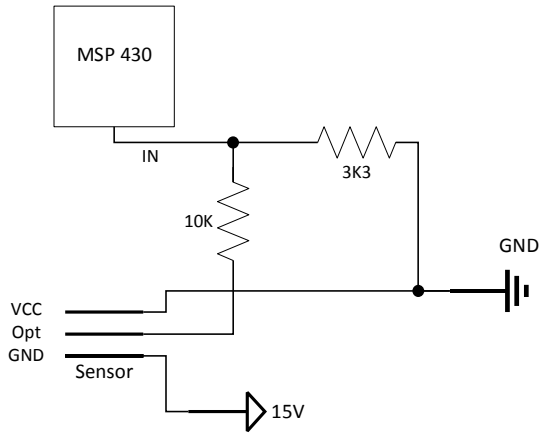


Figura 18 - Esquemático de conexão com o sensor ótico

4.2 Projeto do sistema de acionamento

Para o circuito de acionamento da abordagem discreta foi construída uma ponte H que é responsável por chavear a direção da corrente que produz o sentido dos motores. Na versão inicial do projeto, o chaveamento era realizado com relés de 12v e por segurança o circuito de controle, onde se encontra o microprocessador, foi separado eletricamente caso alguma alta corrente do circuito de acionamento fosse gerada podendo danificar o circuito de controle. Para esta proteção foram utilizados optoacopladores, que são fototransistores capazes de chavear uma fonte isolada quando gerado um sinal em sua base, eliminando o contato elétrico entre os sistemas de controle e de acionamento.

Visto que os relés utilizam contatos físicos, eles não são ideais para trabalhar com alta frequência de chaveamento. Como a abordagem de sistema de tempo contínuo utiliza sinais PWM para o chaveamento, foi desenvolvida uma nova ponte H com Mosfets, capaz de gerenciar o nível de potência dos motores, sendo ela capaz de receber um sinal PWM e trabalhar com potências variáveis de acionamento. Assim, a cada decisão de saída da lógica Fuzzy, os motores atuam em diferentes velocidades. Manteve-se o uso dos optoacopladores para proteger e criar o isolamento entre os sistemas de controle e potência.

Com a segunda abordagem, foi utilizado o circuito integrado IRS2184 para não permitir que o circuito fechasse inadequadamente, eliminando assim as chances de curto. Também sendo o driver responsável por guiar a ponte H de forma complementar, direcionando a corrente para cada movimento do motor. Pode-se observar o esquemático da nova ponte H no Apêndice II.

4.3 Projeto da estratégia de controle a eventos discretos

Inicialmente a partir dos sensores externos e das necessidades de movimentos, foram estabelecidos eventos, os quais fariam parte do alfabeto do autômato. As ações foram definidas para cada evento, visto que foi utilizada a máquina de estados de Mealy, portanto, cada transição seria responsável por uma ação no sistema. Neste trabalho foram agrupadas as transições com a mesma característica de controlabilidade. De acordo com a teoria de controle supervisório, os eventos foram divididos em dois grupos: o grupo dos eventos controláveis, que são transições geradas e controladas no software de controle e o

grupo de eventos não controláveis, que são gerados a partir de borda dos sensores. Os eventos obtidos podem ser observados na Tabela 2 e Tabela 3.

Tabela 2 – Alfabeto de eventos não controláveis

Evento	Descrição
Bd	Borda de descida do botão para dar início ao confronto pelo lado direito
Be	Borda de descida do botão para dar início ao confronto pelo lado esquerdo
S1	Borda de subida do sensor direito de faixa indicando que encontrou o limite do dojô
S2	Borda de subida do sensor esquerdo de faixa indicando que encontrou o limite do dojô
P1a	Borda de subida do sensor frontal esquerdo indicando que encontrou o adversário
P1p	Borda de descida do sensor frontal esquerdo indicando que perdeu o adversário
P2a	Borda de subida do sensor frontal direito indicando que encontrou o adversário
P2p	Borda de descida do sensor frontal direito indicando que perdeu o adversário
P3a	Borda de subida do sensor lateral esquerdo indicando que encontrou o adversário
P4a	Borda de subida do sensor lateral direito indicando que encontrou o adversário

Tabela 3 – Alfabeto de eventos controláveis

Evento	Descrição
F	Robô anda para frente com restrição de tempo t1
Ft	Robô anda para frente com velocidade máxima sem restrição de tempo
Ge1	Robô gira para esquerda travando o motor da esquerda sem restrição de tempo
Gd1	Robô gira para direita travando o motor da direita sem restrição de tempo
Ge2	Robô gira para esquerda em seu próprio eixo com restrição de tempo t2 necessária para girar 120^0
Gd2	Robô gira para direita em seu próprio eixo com restrição de tempo t2 necessária para girar 120^0
Ge3	Robô gira para esquerda em seu próprio eixo com restrição de tempo t3 necessária para girar 90^0
Gd3	Robô gira para direita em seu próprio eixo com restrição de tempo t3 necessária para girar 90^0
R	Robô ande de ré com restrição de tempo t4
G360	Robô gira para direita em seu próprio eixo com restrição de tempo t5 necessária para girar 360^0

Com os eventos definidos, os autômatos das plantas e especificações foram gerados com o objetivo de dar ao sistema final a dinâmica proposta de ataque e defesa do robô de acordo com os sinais dos sensores e os eventos controláveis que serão ministrados pelo microcontrolador. As plantas correspondem a cada sensor do robô, portanto, o sistema foi inicialmente composto de seis plantas que representam os sensores e uma planta de movimento. A ferramenta utilizada para a modelagem dos autômatos das plantas e especificações foi o software Supremica, que permite a criação das plantas e a simulação da composição síncrona e do supervisor. As plantas e as especificações estão apresentadas nas figuras a seguir.

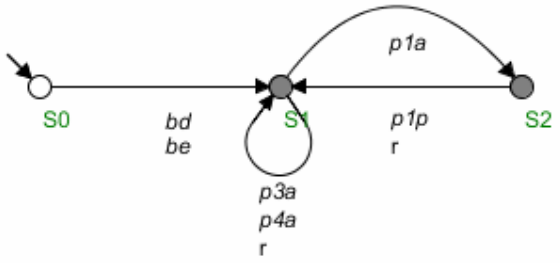


Figura 19 - Planta do sensor P1 (frontal esquerdo)

O autômato do sensor P1 apresentado na Figura 19 é responsável por tratar o sinal frontal esquerdo onde sempre que ativado por um objeto externo ele deve migrar de estado com a transição “p1a” indicando que achou o adversário. Se o sensor perdeu o adversário o estado retornara pela transição “p1p” indicando que o perdeu. Caso o sensor não tenha encontrado nada ou perdido e a transição de início “bd” ou “be” tenha sido realizada ele permanecera no estado S1. Neste estado são colocadas as transições “p3a” e “p4a” e “r” em forma de auto laço para que não haja nenhum bloqueio indevido. Nos estados S1 e S2 da planta P1, a transição “r” deve estar presente, pois quando o robô encontrar uma faixa, deve existir a possibilidade de priorizar a transição “r” para evitar o robô sair fora do Dojô. A ausência de “p3a” e “p4a” no estado S2 se deve ao fato do sensor P1 estar impossibilitado de encontrar o objeto, pois neste estado o objeto encontra-se em frente ao robô, portanto os sensores laterais serão bloqueados ou ignorados por não existir a possibilidade de o oponente estar nas laterais. A Figura 20 representa o sensor P2, e sua construção é análoga ao descrito acima.

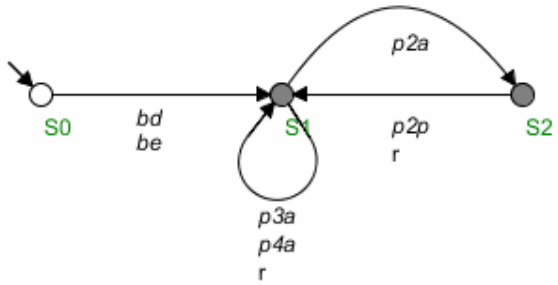


Figura 20 - Planta do sensor P2 (frontal direito)

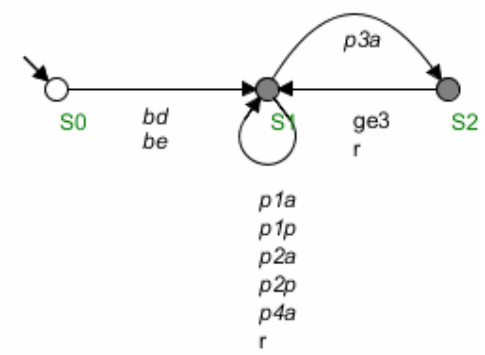


Figura 21 - Planta do sensor P3 (lateral esquerdo)

O autômato do sensor P3 representado na Figura 21 é responsável por tratar o sinal do sensor lateral esquerdo e segue o mesmo conceito inicialmente descrito para os sensores frontais. Para este sensor não foi necessário modelar o evento "p3p" de perda do oponente, pois sempre que encontrado o oponente pela lateral a transição de resposta "ge3" que corresponde a um giro de 90^0 será responsável pelo retorno ao estado de partida. A transição "r" ainda continua sempre presente por sua prioridade caso ocorra um encontro de faixa de limite do Dojô. O auto laço se define pela mesma lógica anterior, ou seja, sempre que a transição "p3a" ocorrer não é necessário se preocupar com as transições dos sensores do auto laço no estado S2, pois com certeza o oponente não estará localizado naquelas posições naquele instante. Para o autômato do sensor lateral direito P4 na Figura 22, segue a mesma lógica de implementação.

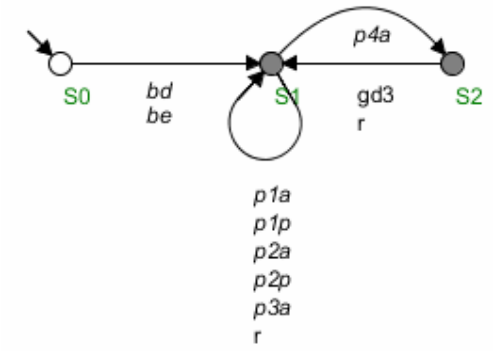


Figura 22 - Planta do sensor P4 (lateral direito)

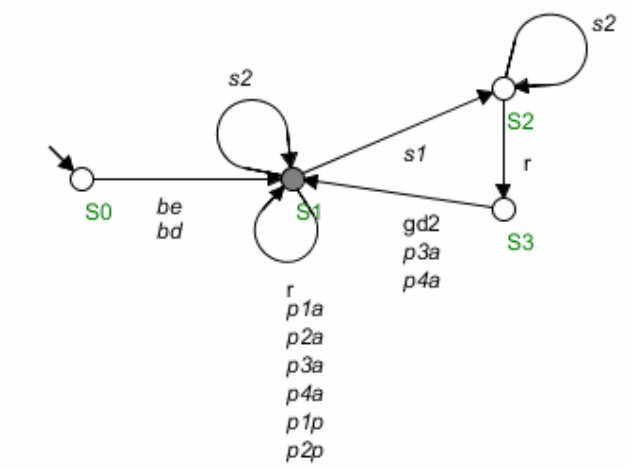


Figura 23 - Planta do sensor SF1 (faixa esquerdo)

A planta do autômato do sensor de faixa SF1 apresentado na Figura 23 é responsável por tratar quando a faixa é encontrada pelo lado esquerdo. Assim que encontrado a faixa, o evento "s1" irá ocorrer. Logo após, o evento controlável em sequência será o "r" para o robô voltar ao dojô e não atravessar o limite. Após ocorrer o evento "r", apenas os sensores laterais voltam a estar habilitados e caso nenhum gere um evento, o evento controlável "gd2" irá ocorrer, fazendo que o robô se direcione ao centro do dojô. Para o autômato do sensor de faixa SF2 apresentado na Figura 24 segue o mesmo raciocínio, invertendo a posição dos eventos "s1" e "s2" e após ocorrer "r" o próximo evento controlável habilitado será o "ge2" e não mais o "gd2".

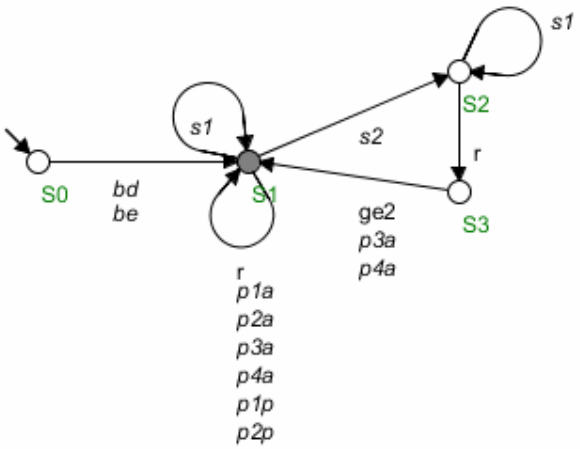


Figura 24 - Planta do sensor SF2 (faixa direito)

A planta da Figura 25 representa os movimentos que o robô pode fazer e que não estão representados nas plantas especificas dos sensores. Inicialmente, apenas o evento

“f” estará habilitado segundo a especificação de ataque, que especifica a dinâmica entre todos os eventos controláveis desta planta, como pode ser observado na figura 26.

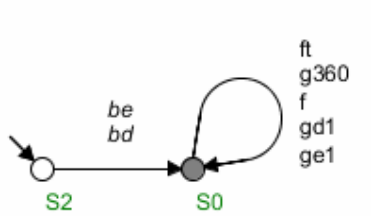


Figura 25 - Planta dos movimentos

Para a especificação de ataque apresentada na Figura 26, encontra-se situações que serão seguidas após encontrar o oponente em cada sensor frontal e a movimentação padrão para quando não houver nenhuma excitação externa ao sistema. Inicialmente após os eventos de início “bd” e “be”, a planta permanece num ciclo controlável dos eventos “f” e “g360”, os quais farão o robô ir para frente e girar 360^0 ambos temporizado, esta estratégia de movimento fará com que o robô ache o adversário de uma forma mais rápida.

Quando ocorrer um estimulo num dos sensores frontais a especificação habilita na planta de movimento da Figura 25 os eventos controláveis “gd1” e “ge1” dependendo de qual sensor encontrou o oponente e bloqueia os eventos “f” e giro 360^0 na própria especificação. Se ao ocorrer um desses eventos ambos os sensores P1 e P2 detectaram o oponente simultaneamente o evento “ft” será o evento controlável habilitado no estado S4 da especificação, dando ao robô o máximo de potência para atacar o oponente.

Como explicado anteriormente, o evento de ré deve estar modelado para que exista uma prioridade do robô nunca sair do Dojô, portanto, independente do estado em que o sistema se encontra o evento “r” deve estar disponível e possuir prioridade. Esta especificação também delimita que sempre ao retornar ao estado de início que ocorre após o evento “bd” e “be”, o robô sempre começará andando para a frente e não girando 360^0, independente do estado em que partiu o evento “p1a” e “p2a”.

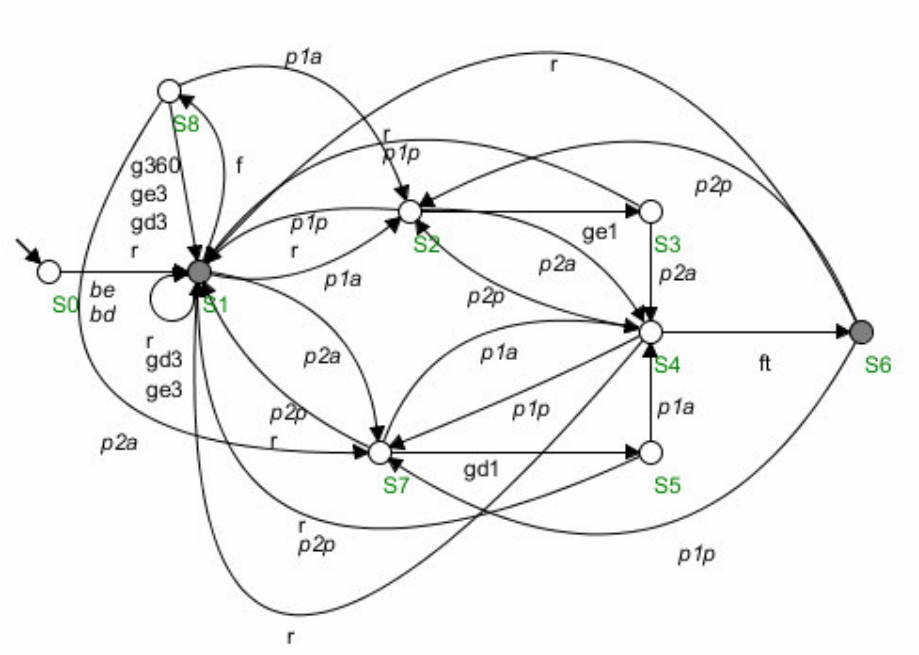


Figura 26 - Especificação de ataque

A especificação representada pelo autômato na Figura 27 é responsável por restringir a situação de defesa quando o robô se encontra no limite de onde ele não pode sair. Ao encontrar a faixa por baixo dele o evento “s1” ou “s2” será gerado, dependendo de qual sensor de faixa está no limite. Após este evento todos os eventos controláveis menos o evento “r” serão bloqueados para que o robô só possa voltar à região de circulação após encontrar a faixa. O evento “r” é um evento temporizado, e se no meio do movimento de ré os sensores laterais encontrarem o adversário, por meio dos eventos p3a ep4a, a especificação retorna ao estado marcado e o robô volta a se movimentar segundo a regra de ataque.

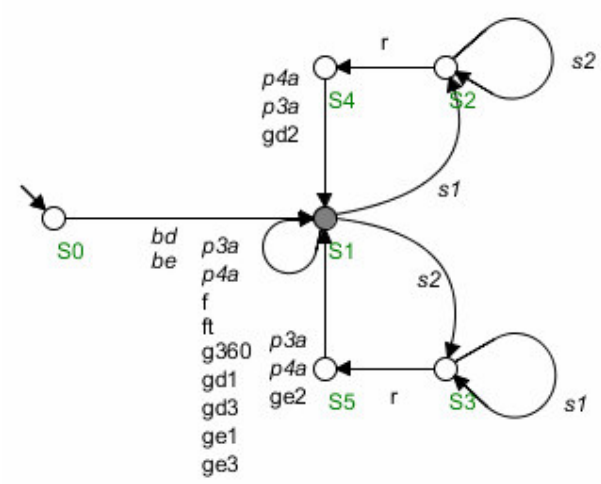


Figura 27 - Especificação de defesa

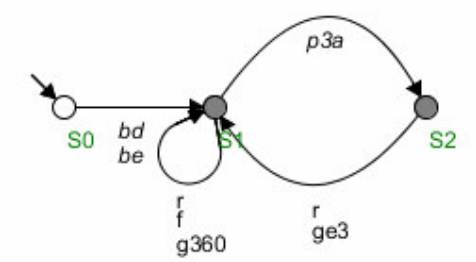


Figura 28 - Especificação do sensor P3

A especificação acima na Figura 28 restringe os eventos "f" e "g360^{0}" após a ocorrência da transição "p3a", ou seja, quando o robô encontrar pelas laterais o robô não pode ir para frente nem girar 360^{0}. A especificação restringe também o evento "ge3" para ocorrer apenas após um "p3a", exatamente como na planta. Para a especificação da Figura 29, segue a mesma lógica, só que agora para o outro sensor lateral P4.

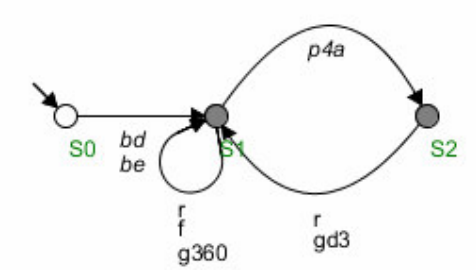


Figura 29 - Especificação do sensor P4

A composição das plantas resultou num autômato com 28 estados e 213 transições. Já a composição do comportamento desejado resultou em 69 estados e 151 transições. Finalmente, o supervisor para o SED também ficou com 69 estados e 151 transições e ao minimizá-lo resultou em 23 estados e 79 transições.

De posse das plantas e das especificações a primeira tarefa foi gerar o código para o msp430G utilizando a ferramenta Deslab, que permite após a definição das plantas e das especificações a geração do autômato supervisor mínimo que resultou no código base do microcontrolador MSP430G para as saídas de Mealy, contando ainda com a opção de Moore. Neste caso foi escolhido Mealy, onde cada transição será responsável pelas ações do robô, sendo que é possível alcançar mesmo estados com transições diferentes, definindo ações diferentes naquele momento pela transição.

No código foi implementado todas as ações para cada evento controlável como apresentado na tabela 3, sendo considerado os sinais de direcionamento dos motores para

o circuito de acionamento e os temporizadores gerados no timer TA0 do MSP430G quando no evento se fazia necessário.

Foi utilizado o software Supremica para realizar os estudos de caso e simulações do conjunto de plantas e especificações criadas no projeto. As repostas foram dentro da restritividade esperada no planejamento inicial e suas liberdades de movimentos e acionamentos de sensores também corresponderam à rotina esperada. Nas Figuras 30 e 31 pode-se observar os autômatos explicados anteriormente na modelagem das plantas e especificações dentro do ambiente de simulação do Supremica. O último evento a ser simulado está em cor roxa e os possíveis próximos eventos estão em verde.

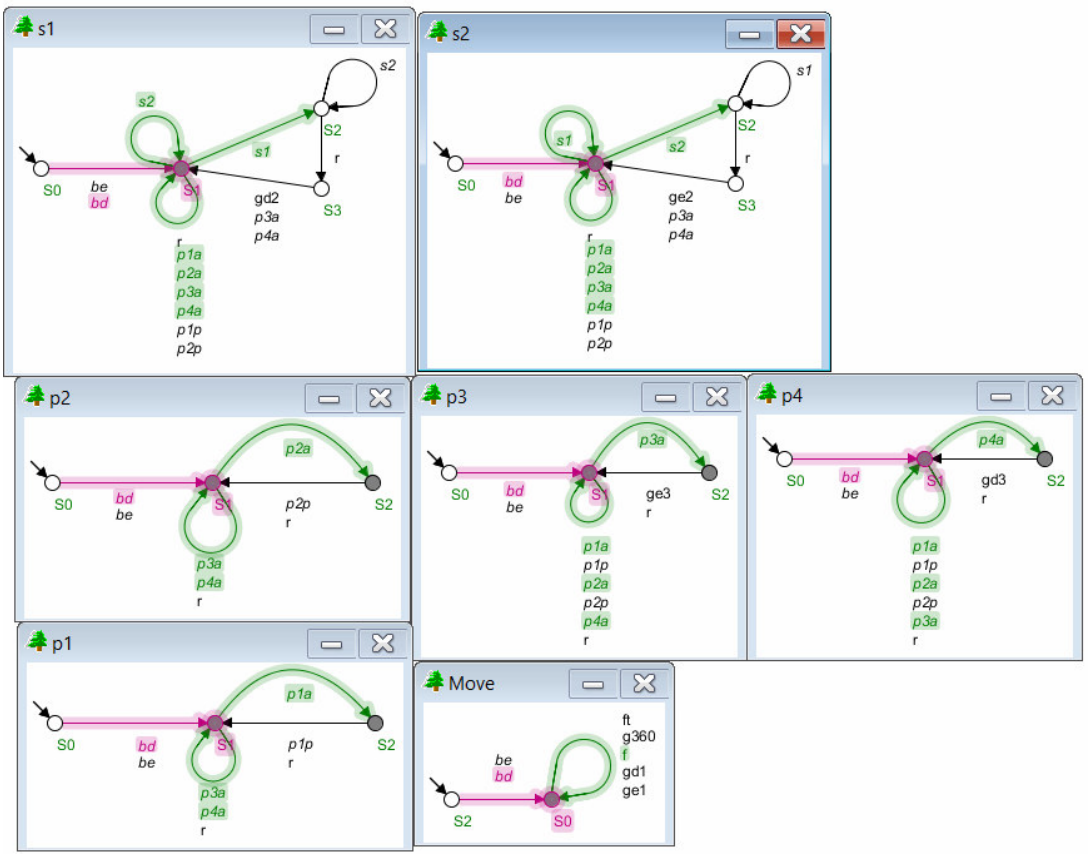


Figura 30 - Conjunto de plantas do sistema

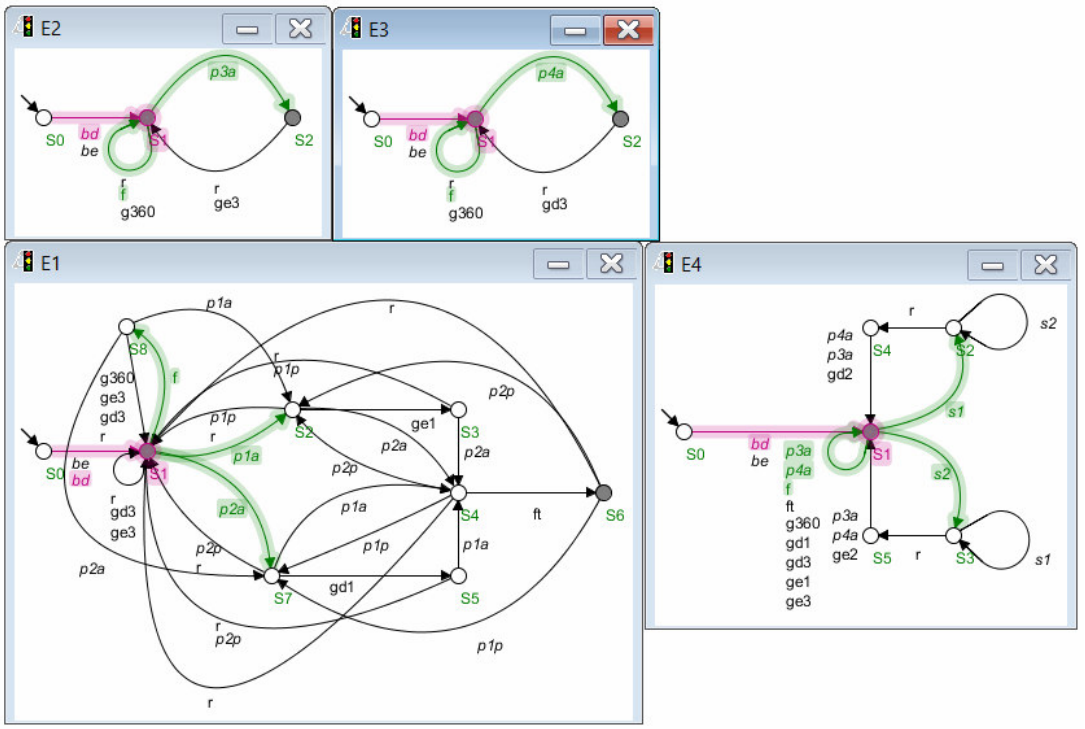


Figura 31 - Conjunto de especificações do sistema

Assim como nas simulações, o robô obteve os movimentos esperados e encontrou-se dentro do mesmo padrão de movimento. O mesmo foi observado com a opção de Debug do software Code Composer, utilizado para desenvolver e gravar o código no microcontrolador.

4.4 Projeto do sistema de controle de tempo contínuo

Foi inserido nas transições dos autômatos a lógica Fuzzy, que foi capaz de introduzir a abordagem contínua, em que a entrada do sistema é constantemente alimentada com um sinal que representa a distância do adversário e a sua velocidade relativa. De posse dessa lógica, definiram-se quatro novas decisões nas ações dos eventos de P1a, P2a, P3a, P4a.

As funções de pertinência da lógica Fuzzy estão representadas na Figuras 32, a qual representa as distâncias, e a Figura 33 que representa as velocidades do adversário. Com as funções de pertinências define-se os vetores do grau e do conjunto de regras. Paras as distâncias foi criado o conjunto {Perto, Médio, Longe} e para as velocidades o conjunto {Lento Af, Médio Af, Rápido Af, Lento Ap, Medio Ap, Rápido Ap}. No caso

das velocidades Af. quando o adversário está se afastando e Ap. quando está se aproximando.

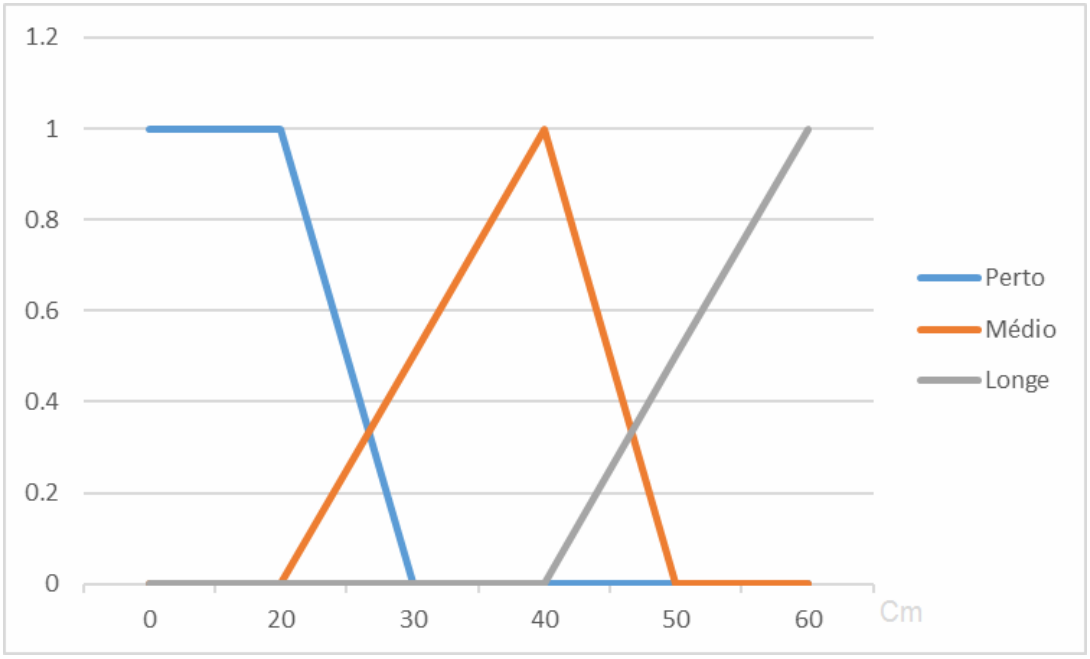


Figura 32 - Função de pertinência (Distância)

`

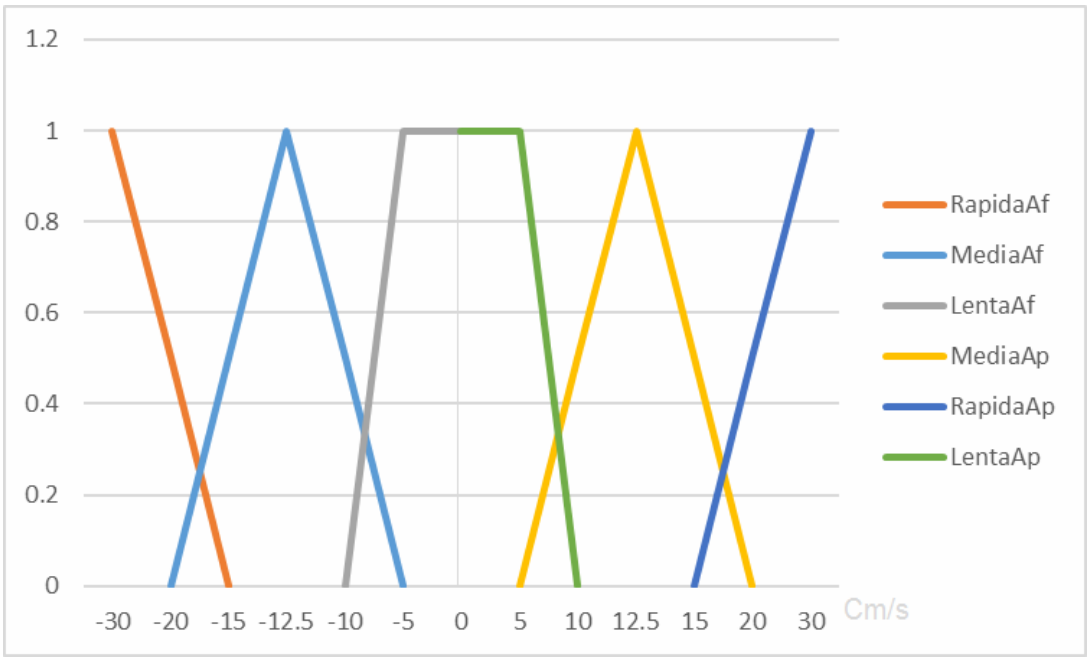


Figura 33 - Função de pertinência (Velocidade)

A partir das equações de pertinência, é possível identificar os grupos de regras para levantar os graus finais a serem considerados para o processamento e também para realizar a fuzzificação. Na Tabela 4 tem-se os conjuntos de regras definidos pelas combinações das entradas de distância e velocidade.

Tabela 4 - Conjunto de regras

Velocidade/Distância	Perto	Médio	Longe
Lento Af	Alta	Alta	Média
Médio Af	Alta	Média	Média
Rápido Af	Alta	Média	Baixa
Lento Ap	Média	Média	Baixa
Médio Ap	Alta	Média	Média
Rápido Ap	Alta	Alta	Média

Dentro do conjunto de regras, foram definidas 3 saídas para o sinal que alimentará a ponte H. A saída Alta representa 99% do ciclo de trabalho, Média representa a saída em 66% e Baixa 33%. Como pode ser observado na Figura 34.

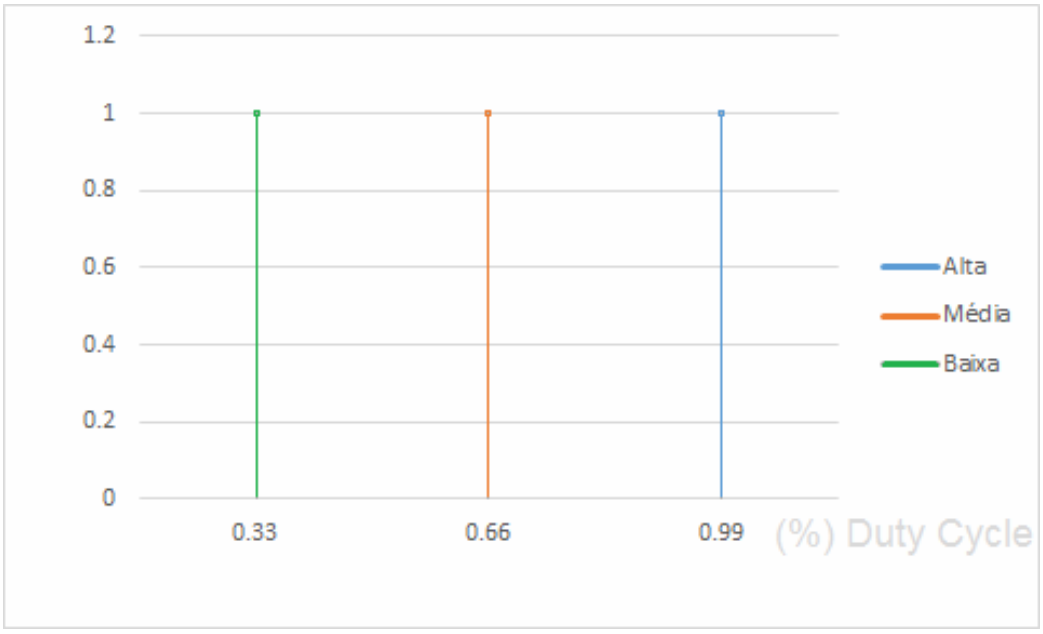


Figura 34 - Função de resposta de saída

Para trabalhar com os novos sinais de entradas foi desenvolvido um novo circuito de controle capaz de receber os 8 sensores, nos quais 4 são óticos e 4 são de ultrassom. Para o MSP430G2553 ser capaz de receber os 8 sinais, foi utilizada uma técnica onde os

sinais dos quatro sensores de ultrassom passam por uma por lógica AND e apenas a saída desta porta vai para o micro controlador, sendo este capaz de suportar a quantidade de entradas necessárias para o objetivo proposto após esta modificação.

Com o novo sinal de entrada dos sensores de ultrassom, foi utilizado o timer TA0, configurado como captura de sinal, onde por meio de interrupções de borda, foi-se capaz de interpretar e calcular o tempo de reflexão do som emitido pelo sensor, identificando a distância entre o robô e o adversário.

A lógica Fuzzy, assim como o sistema a eventos discretos foi testada na forma teórica com o uso de exemplos práticos de entradas. A lógica apresentou a partir das triangulações das funções de pertinência algumas situações onde o robô tinha 0% como resposta de saída para o ciclo de trabalho dos motores, o que representava o robô parado. Os casos nulos representavam todo fim de função de pertinência onde havia apenas uma reta extrema levando ao grau a 0. Foram realizados ajustes e calibrações nas ondas triangulares para eliminar os pontos onde essa saída do conjunto de fuzzificação fosse 0%, não sendo esperado na estratégia este comportamento. Assim, todos os extremos foram fixados em 1 e não mais em 0, eliminando a parada inesperada do robô.

Para exemplificar um caso natural, foi montado uma sequência de entrada e calculado a saída Fuzzy para os casos propostos. Primeiro caso para uma entrada natural de distância de 15cm e velocidade 17cm/s de aproximação do adversário pelo lado direito.

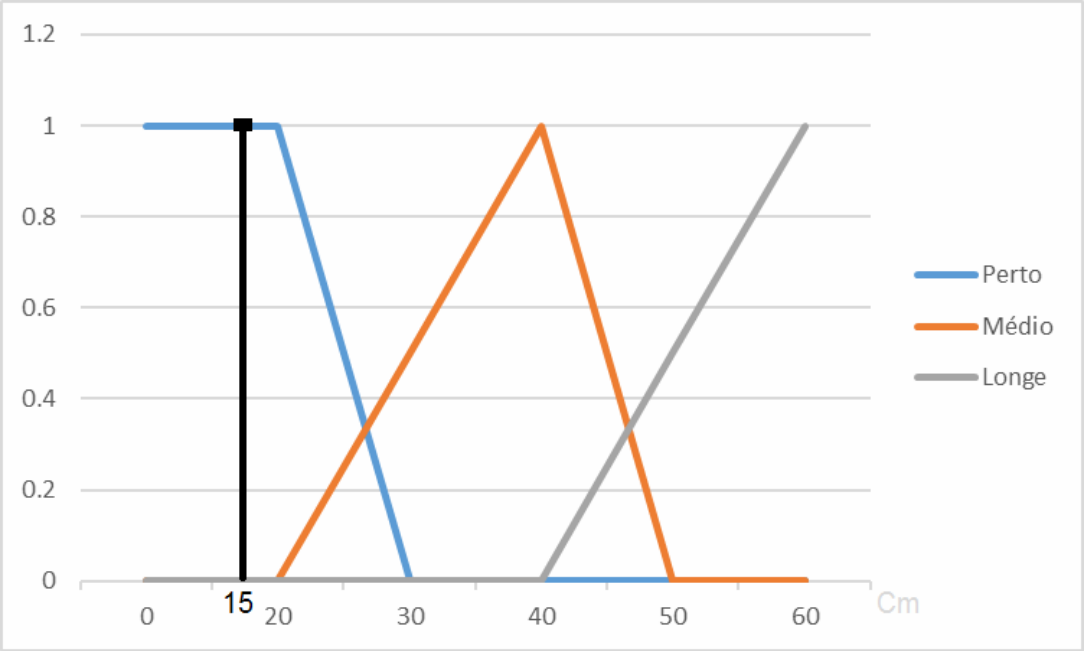


Figura 35 - Entrada de 15cm

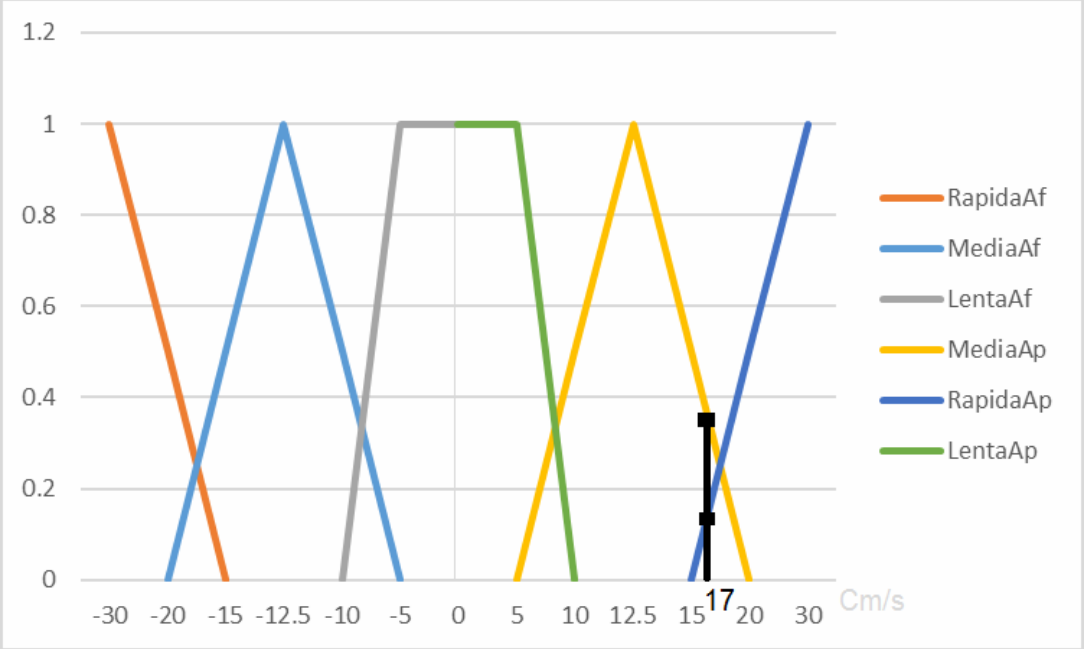


Figura 36 - Entrada de 17cm/s

Como representado na Figura 35, tem-se a intersecção da entrada de 15cm com a função que representa a distância perto com um grau 1 de pertinência. Também se tem as intersecções da entrada de 17cm/s com a função “MédiaAp” e “RápidaAp” com graus 0,37 e 0,16 respectivamente na Figura 36. Feita a combinação entre as regras tem-se as saídas Perto(1) e MédiaAp(0,37), a saída “Alta” com 0,37 de grau de pertinência. Para a

outra combinação Perto(1) e RápidaAp(0,16), tem-se Alta com 0,16 de grau de pertinência. Com as duas saídas em Alta, retira-se o maior grau entre elas, resultando na Saída Alta(0,4).

Velocidade/Distância	Perto	Médio	Longe
Lento Af	Alta	Alta	Média
Médio Af	Alta	Média	Média
Rápido Af	Alta	Média	Baixa
Lento Ap	Média	Média	Baixa
Médio Ap	Alta	Média	Média
Rápido Ap	Alta	Alta	Média

Figura 37 - Combinações de regras de entrada

Para o processamento da saída é feito o seguinte cálculo:

$$\frac{0,99 \times 0,4}{0,4}$$

O cálculo resultou num PWM de 99% do valor de acionamento definido pela lógica de eventos discretos. Além do PWM definido, a ação será de fuga pois o adversário está se aproximando do lado direito. O robô terá uma ação de fuga em movimento contrário seguindo a proporção de 0,99 do PWM configurado para tal ação.

Foi realizada uma segunda simulação de caso real onde ocorrem intersecções em ambas funções de pertinência. Neste caso as entradas de distância e velocidade foram respectivamente 22cm e -9cm/s. Para esta nova entrada é retirada as quatros seguintes combinações: Medio(0.08) e MediaAf(0.37), Medio(0.08) e LentaAf(0.35), Perto(0.8) e MediaAf(0.37), Perto(0.8) e LentaAf(0.35). A partir destas combinações, são definidos pelas regras as seguintes saídas parciais: Media(0.08), Alta(0.08), Alta(0.37), Alta(0.35), portando, Media(0,08) e Alta(0.37), resultando na seguinte expressão:

$$\frac{0,66x0,08 + 0,99x0,37}{0,08 + 0,37}$$

Para o segundo, resultou-se num PWM de 93% onde existe uma tendência do robô adversário se distanciar, porém, por ele ainda permanecer com uma distância próxima, o robô atuará com um ciclo de trabalho elevado e com perfil de ataque.

Com a aplicação prática das equações no microcontrolador e com as entradas provenientes do sensor de ultrassom, permite-se calcular para cada situação um valor de reposta Fuzzy onde o robô tende a ficar mais lento em situações mais confortáveis para decisões e mais rápido em situações de defesa e ataque imediato. O robô responde assim com dinamismo frente a diversidade de informação.

4.5 Integração dos dois sistemas anteriores e arquitetura de implementação

O sistema de controle compreende o todo desenvolvido para suportar o esquema apresentado na Figura 6. Este sistema suporta os sensores de ultrassom que definem a distância e velocidade do adversário, sensores óticos responsáveis por encontrar o adversário, sensores de faixa para não permitir o robô de sair do Dojô e os botões de início para controlar a direção inicial que o robô irá seguir, ainda como alimentar a ponte H com os sinais de acionamento para o motor.

Foi aprimorado o protótipo existente, para ser possível o uso em paralelo de todos os sensores, caso que não acontecia na versão anterior. Pode-se observar o novo esquemático de controle desenvolvido no apêndice I.

O novo sistema de controle compreendendo todo o sistema entre receber os sinais, processá-los e enviá-los para a ponte H, apresentou resultados esperados com os Timers utilizados em todos os processos. O Timer 1 módulo *B* do MSP realizou a função de captura corretamente. Os testes foram realizados utilizando-se o *Debug* do CodeComposer e apresentaram as distâncias corretas aos objetos com relação aos sensores de ultrassom. O código de captura está apresentado no apêndice 3, onde todas as bordas de subidas e descidas dos sensores de ultrassom são contabilizadas para o cálculo da distância e velocidade. O Timer 0 do MSP realizou a função de temporização adequadamente, permitindo que os movimentos tivessem um tempo em ms determinados de acordo a estratégia. A função de interrupção de porta foi capaz de interpretar as bordas dos sinais de entrada dos sensores óticos e os botões de início. Pela limitação de Timers do microcontrolador, o *WatchDogTimer* foi responsável e capaz de a cada interrupção configurar os ciclos de trabalho dos sinais PWM para a ponte H e também para o

acionamento dos sensores de ultrassom, utilizando como base de interrupção um período de aproximadamente 60ms.

O novo circuito desenvolvido para o controle funcionou corretamente, sendo capaz de comportar todos os sinais propostos. Pode-se verificar a foto do circuito na Figura 38 e o esquemático no Apêndice I.

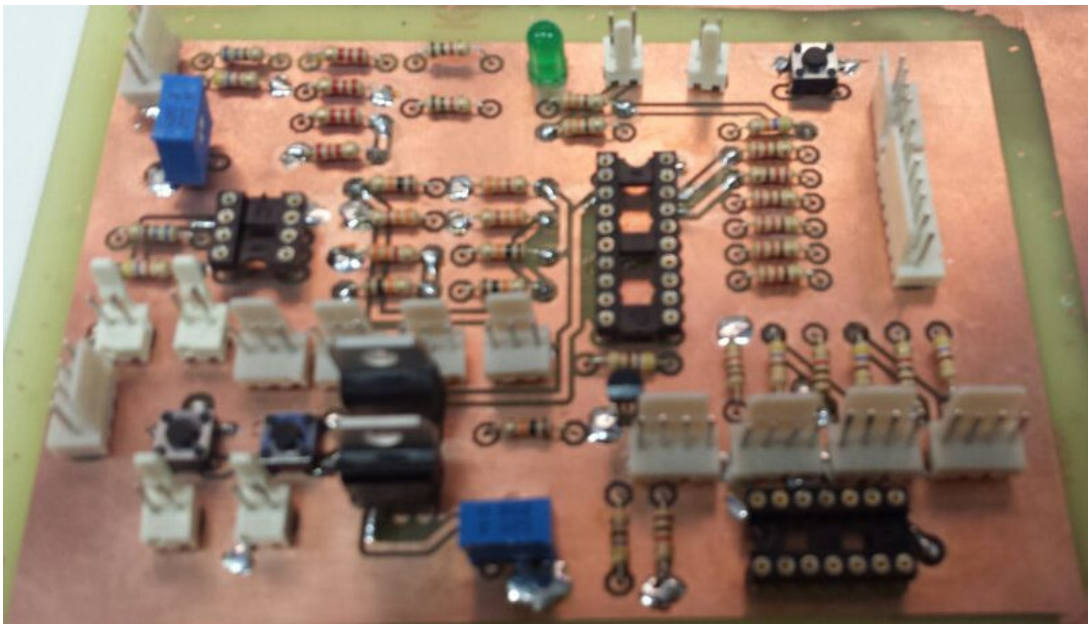


Figura 38 - Circuito de controle

5 CONSIDERAÇÕES FINAIS

Utilizando sistemas a eventos discretos, um simples circuito de acionamento acionado por relés foi suficiente e capaz de controlar o movimento do robô. Porém essa técnica limita as movimentações flexíveis do robô por não possuir um acionamento com sinal PWM capaz de variar a intensidade do movimento para diferentes situações. Se implementado desta forma a alta frequência de chaveamento e a velocidade adequada não corresponderia à resposta do relé. Portanto, diferente da primeira versão do robô o projeto visou modificar o acionamento que antes não era PWM para um que se utilizasse desta técnica de modulação.

O robô quando possuía apenas o controle de um sistema a eventos discretos também se mostrou pouco dinâmico, mas sendo muito preciso. Com o movimento do adversário e a faixa do Dojô, apresenta-se capaz de movimentar-se e de atacar o adversário, mas limita-se apenas às estratégias de ataques simplificadas.

Para torná-lo capaz de tomar decisões diferenciadas, o sistema contínuo foi a estratégia utilizada. O robô foi capaz de decidir ao encontrar adversário, se atacaria ou se defenderia aumentando assim a sua capacidade de ações e decisões. Com a nova ponte H desenvolvida com mosfets, possibilitou-se ao sistema controlar a velocidade do robô para cada caso, diferentemente da abordagem discreta com relés. Com a lógica Fuzzy integrada as entradas de distância e velocidade o foi capaz de variar a saída do robô, entretanto, a variação foi pouca perceptível pela baixa potência dos motores e pela calibração da lógica fuzzy.

Para a integração dos sensores de ultrassom, inicialmente foi utilizado um CI de portas lógicas OR para os quatro sensores de ultrassom, causando uma falha por erro de projeto na resposta de ECHO dos sensores, pois quando um sensor encontrava o adversário ele ia para nível lógico baixo, enquanto que os outros três não encontravam e permaneciam em nível lógico 1. Com esta abordagem a saída do CI OR se mantinha em nível lógico 1 fazendo com que o microcontrolador ignorasse o sensor que encontrou, não calculando a distância corretamente. O projeto foi alterado para a utilização de um CI de portas lógicas AND, respondendo aos requisitos do projeto. Dessa forma, assim que os quatros sensores são acionados, o nível lógico de saída do CI vai para 1 e quando um dos quatro sensores encontra o adversário leva o nível lógico de saída do CI para 0, sendo possível o reconhecimento da borda de descida por parte do microcontrolador para o cálculo da distância corretamente.

É possível observar que com uma quantidade maior de sensores o robô poderia ter uma dinâmica ainda melhor. Se adicionados, na parte de trás do robô, mais dois sensores óticos, dois sensores de ultrassom e mais dois sensores de faixa, ele passaria a ter duas frentes e não mais uma parte de trás, tornando-o muito mais flexível. O gargalo encontrado com essa abordagem é que o MSPG2553 possui pinos limitados, não sendo possível estender essa abordagem com este microcontrolador. Também um outro passo de melhoria seria a migração do projeto para um microcontrolador mais robusto, capaz de realizar processamentos de ponto flutuante, sendo, assim, capaz de atuar com a lógica fuzzy sem atrasos de processamento.

REFERÊNCIAS

AVNUR, Arie. Finite state machines for real-time software engineering. **Computing & Control Engineering Journal,** Santa Clara, nov. 1990. Engenharia, p.275-278.

BARELLA, Wagner D. **Sistemas especialistas modulados e abrangentes para a gestão de operações.** 2000. Tese (Doutorado em Engenharia de Produção) - Programa de Pós-Graduação em Engenharia de Produção, Universidade de São Paulo, São Paulo.

BAUMER, 2016. Disponível em <http://www.baumer.com/us-en/products/distance-measurement/ultrasonic-sensors/>. Acesso em: 02 dez. 2016.

BOYER, Carl B., 1996, **História da Matemática**, São Paulo, Editora Edgar Blücher.

BRAGA, 2012. Disponível em <http://www.newtoncbraga.com.br/index.php/como-funciona/5273-art691>. Acesso em: 02 dez. 2016.

CASSANDRAS, Christos G.; LAFORTUNE, Stéphane; **Introduction to discrete event systems**, 2nd ed., LLC: EUA, 2008.

CAZANGI, Renato R. **Síntese de controladores autônomos em robótica móvel por meio de comunicação bio-inspirada**. 2008. Tese (Doutorado em Engenharia Elétrica) - Programa de Pós-Graduação em Engenharia de Computação, UNICAMP, Campinas.

DARPA, 2015. Disponível em: <http://www.theroboticschallenge.org/>. Acesso em: 27 abr. 2015.

FEIGENBAUM, Edward A. The art of Artificial Intelligence: I. Themes and Case Studies of Knowledge Engineering. In: International Joint Conference on Artificial Intelligence. **Proceedings….** p.1014-1029, 1977.

JAYASIRI, Awantha; MANN, George K. I.; GOSINE, Raymond G. Modular Supervisory Control and Hierarchical Supervisory Control of Fuzzy Discrete-Event Systems. **IEEE Transactions on Automation Science and Engineering,** v.9, n.2, p.353-364, abr. 2012.

MALVINO, Albert P.; BATES, David J. **Electronic principles**, 8 ed. Nova Yorque: Editora Mcgraw-Hill Education, 2016.

MONTGOMERY, Eduard. **Introdução aos Sistemas a Eventos Discretos e à Teoria de Controle Supervisório,** 1st ed, Alta books, 2004.

RAMADGE, Peter J. G.; WONHAM, Murray. The control of discrete event systems. **Proceedings of the IEEE,** v.77, n.1, p.81-98, jan. 1989.

ROBOCORE, 2015. Disponível em: < https://www.robocore.net/>. Acesso em: 26 abr. 2015.

ROBOCUP, 2015. Disponível em: <http://www.robocup2015.org/>. Acesso em: 26 abr. 2015.

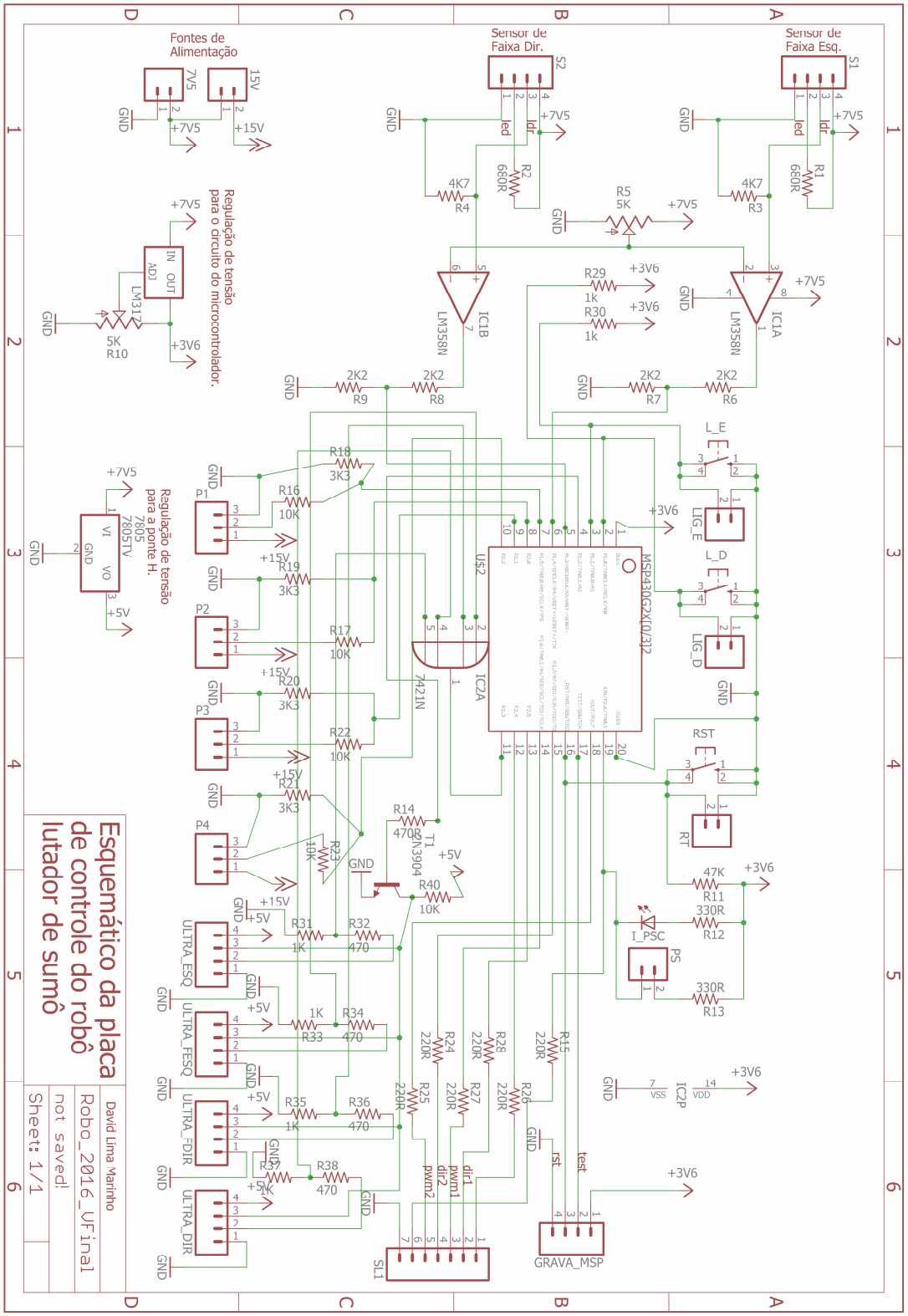
SHAW, Ian S., SIMÕES, Marcelo Godoy. **Controle e modelagem fuzzy**, São Paulo, Editora Edgar Blücher, 1999.

SIEBEN, Vincent. A High Power H-Bridge, 2003.

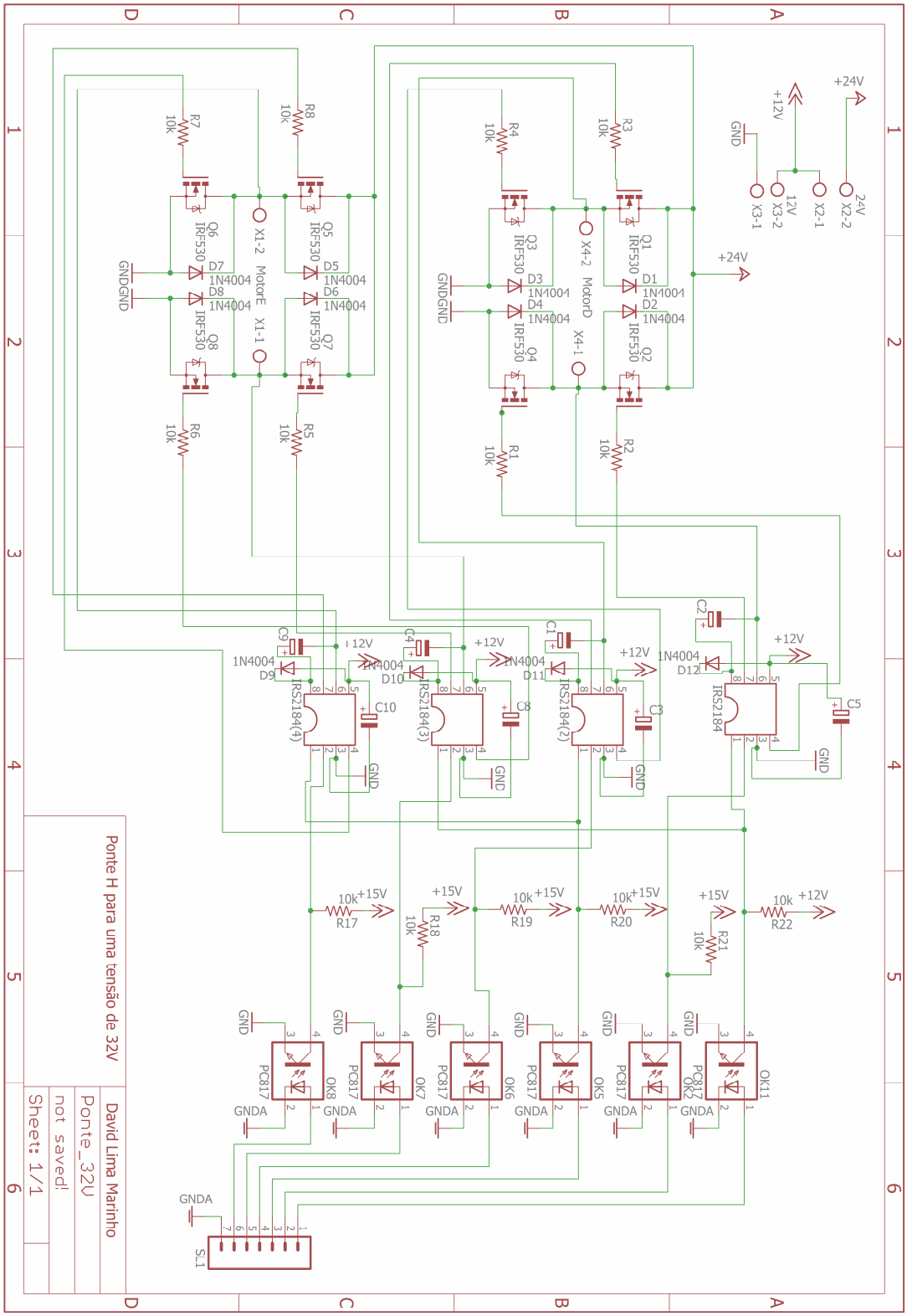
TEXAS INSTRUMENTS. MSP430x2xx family user's guide. Disponível em: <http:// www.ti.com/lit/ug/slau144j/slau144j.pdf>. Acesso em: 30 nov. 2016.

ZIMMERMANN, Hans-Jiirgen **Fuzzy set theory and its applications**, U.S.A, Kluwer Nijhoff Publishing, 1985.

APÊNDICE I – Esquemático de controle



APÊNDICE II – Esquemático de potência



APÊNDICE III – Código de captura do sensor de ultrassom

```
//Configuração do Timer1 para captura
//SMCLK | Modo Contínuo | Clock SMCLK/8
TA1CTL = TASSEL_2 + MC_2 + ID_3;
//Modo Captura | Captura Borda Desc. e Sub.| Captura Síncrona | CCIxB |
Interrupcão de Captura Habil.
TA1CCTL0 = CAP + CM_3 + SCS + CCIS_1 + CCIE;

#pragma vector = TIMER1_A0_VECTOR  //Realiza somente os cálculos de distância
__interrupt void Timer_A0(void)
{
    if(((P2IN & P2_ECHO)>0)) //Se o Echo do sensor estiver em nível alto,
então TACCR0 recebe TAR de início
    {
        tempo_bsubida = TA1CCR0;
    }

    else   //Echo do sensor em nível baixo, TACCR0 recebe TAR de fim
    {

        medida = TA1CCR0 - tempo_bsubida; //Número de contagens do ECHO em
nível alto
        TA1CCR0 = 0;
        if(medida<6690){
            media[j]=medida;
            if(j>0){
                  if(media[j]<(media[j-1]+116) && media[j]>(media[j-1]-116)){
                        j++;
                  }
                  else{
                        j=0;
                  }
            }else{
                  j++;
            }

            if(j==2){
                  j=0;
                  realizamedida=1;
            }
        }
    }

    TA1CTL &= ~TAIFG;
}

void main(void) //...Código de configuração também dentro da main (Está
oculto)
 {
      while(1)
      {
            if(realizamedida==1){
                 medida = (int)((media[0]+media[1])/2);
                 distancia = (medida/2)/58;      //Distância em cm
                 realizamedida=0;

            //...Código de SED e Fuzzy dentro while(1) (Está oculto)
      }
```

APÊNDICE IV – Lógica Fuzzy

```
void Fuzzy(unsigned int movimento){

	GrauD1 = 0;
	GrauD2 = 0;
	GrauD3 = 0;
	GrauV1 = 0;
	GrauV2 = 0;
	GrauV3 = 0;
	GrauV4 = 0;
	GrauV5 = 0;
	GrauV6 = 0;
	Saida1=0;
	Saida2=0;
	Saida3=0;
	Saida4=0;
	Saida5=0;
	Saida6=0;
	Saida7=0;
	Saida8=0;
	Saida9=0;
	Saida10=0;
	Saida11=0;
	Saida12=0;
	Saida13=0;
	Saida14=0;
	Saida15=0;
	Saida16=0;
	Saida17=0;
	Saida18=0;
	saidaFuzzy=0;

	if(distanciaFuzzy >= 0 && distanciaFuzzy <=30){
		if(distanciaFuzzy<=20){
			GrauD1 = 1;
		}else{
			GrauD1 = ((-0.1*distanciaFuzzy)+3);
		}
	}

	if(distanciaFuzzy >= 20 && distanciaFuzzy <=50){
		if(distanciaFuzzy<=40){
			GrauD2 = ((0.05*distanciaFuzzy)-1);
		}else{
			GrauD2 = ((-0.1*distanciaFuzzy)+5);
		}
	}

	if(distanciaFuzzy >= 40 && distanciaFuzzy <=60){

			GrauD3 = ((0.05*distanciaFuzzy)-2);
	}

	if(velocidade[2] >= -30 && velocidade[2] <=-15){

		GrauV1 = ((-0.07*velocidade[2])-1);
```

```
  }

  if(velocidade[2] >= -20 && velocidade[2] <=-5){
        if(velocidade[2]<=-12.5){
              GrauV2 = ((0.13*velocidade[2])+2.67);
        }else{
              GrauV2 = ((-0.13*velocidade[2])-0.67);
        }
  }

  if(velocidade[2] >= -10 && velocidade[2] <=0){
              if(velocidade[2]<=-5){
                    GrauV3 = ((0.2*velocidade[2])+2);
              }else{
                    GrauV3 = 1;
              }
  }

  if(velocidade[2] >= 0 && velocidade[2] <=10){
        if(velocidade[2]<=5){
              GrauV4 = 1;
        }else{
              GrauV4 = ((-0.2*velocidade[2])+2);
        }
  }

  if(velocidade[2] >= 5 && velocidade[2] <=20){
        if(velocidade[2]<=12.5){
              GrauV5 = ((0.13*velocidade[2])-0.67);
        }else{
              GrauV5 = ((-0.13*velocidade[2])+2.67);
        }
  }

  if(velocidade[2] >= 15 && velocidade[2] <=30){

        GrauV6 = ((0.07*velocidade[2])-1);

  }

  //Escolha dos menores graus entre as regras

  //Distância perto com todas as combinações de velocidade

if (GrauD1 != 0 && GrauV1 != 0) {
    if (GrauD1 < GrauV1) {
        if (Saida1 == 0 || Saida1 < GrauD1) {
            Saida1 = GrauD1;
        }
    } else {
        if (Saida1 == 0 || Saida1 < GrauV1) {
            Saida1 = GrauV1;
        }
    }
}

if (GrauD1 != 0 && GrauV2 != 0) {
    if (GrauD1 < GrauV2) {
        if (Saida2 == 0 || Saida2 < GrauD1) {
```

```
            Saida2 = GrauD1;
        }
    } else {
        if (Saida2 == 0 || Saida2 < GrauV2) {
            Saida2 = GrauV2;
        }
    }
}

if (GrauD1 != 0 && GrauV3 != 0) {
    if (GrauD1 < GrauV3) {
        if (Saida3 == 0 || Saida3 < GrauD1) {
            Saida3 = GrauD1;
        }
    } else {
        if (Saida3 == 0 || Saida3 < GrauV3) {
            Saida3 = GrauV3;
        }
    }
}

if (GrauD1 != 0 && GrauV4 != 0) {
    if (GrauD1 < GrauV4) {
        if (Saida4 == 0 || Saida4 < GrauD1) {
            Saida4 = GrauD1;
        }
    } else {
        if (Saida4 == 0 || Saida4 < GrauV4) {
            Saida4 = GrauV4;
        }
    }
}

if (GrauD1 != 0 && GrauV5 != 0) {
    if (GrauD1 < GrauV5) {
        if (Saida5 == 0 || Saida5 < GrauD1) {
            Saida5 = GrauD1;
        }
    } else {
        if (Saida5 == 0 || Saida5 < GrauV5) {
            Saida5 = GrauV5;
        }
    }
}

if (GrauD1 != 0 && GrauV6 != 0) {
    if (GrauD1 < GrauV6) {
        if (Saida6 == 0 || Saida6 < GrauD1) {
            Saida6 = GrauD1;
        }
    } else {
        if (Saida6 == 0 || Saida6 < GrauV6) {
            Saida6 = GrauV6;
        }
    }
}

//Distância média com todas as combinações de velocidade
```

```
if (GrauD2 != 0 && GrauV1 != 0) {
    if (GrauD2 < GrauV1) {
        if (Saida7 == 0 || Saida7 < GrauD2) {
            Saida7 = GrauD2;
        }
    } else {
        if (Saida7 == 0 || Saida7 < GrauV1) {
            Saida7 = GrauV1;
        }
    }
}

if (GrauD2 != 0 && GrauV2 != 0) {
    if (GrauD2 < GrauV2) {
        if (Saida8 == 0 || Saida8 < GrauD2) {
            Saida8 = GrauD2;
        }
    } else {
        if (Saida8 == 0 || Saida8 < GrauV2) {
            Saida8 = GrauV2;
        }
    }
}

if (GrauD2 != 0 && GrauV3 != 0) {
    if (GrauD2 < GrauV3) {
        if (Saida9 == 0 || Saida9 < GrauD2) {
            Saida9 = GrauD2;
        }
    } else {
        if (Saida9 == 0 || Saida9 < GrauV3) {
            Saida9 = GrauV3;
        }
    }
}

if (GrauD2 != 0 && GrauV4 != 0) {
    if (GrauD2 < GrauV4) {
        if (Saida10 == 0 || Saida10 < GrauD2) {
            Saida10 = GrauD2;
        }
    } else {
        if (Saida10 == 0 || Saida10 < GrauV4) {
            Saida10 = GrauV4;
        }
    }
}

if (GrauD2 != 0 && GrauV5 != 0) {
    if (GrauD2 < GrauV5) {
        if (Saida11 == 0 || Saida11 < GrauD2) {
            Saida11 = GrauD2;
        }
    } else {
        if (Saida11 == 0 || Saida11 < GrauV5) {
            Saida11 = GrauV5;
        }
    }
}
```

```
if (GrauD2 != 0 && GrauV6 != 0) {
    if (GrauD2 < GrauV6) {
        if (Saida12 == 0 || Saida12 < GrauD2) {
            Saida12 = GrauD2;
        }
    } else {
        if (Saida12 == 0 || Saida12 < GrauV6) {
            Saida12 = GrauV6;
        }
    }
}

//Distância longe com todas as combinações de velocidade

if (GrauD3 != 0 && GrauV1 != 0) {
    if (GrauD3 < GrauV1) {
            if (Saida13 == 0 || Saida13 < GrauD3) {
                Saida13 = GrauD3;
            }
        } else {
            if (Saida13 == 0 || Saida13 < GrauV1) {
                Saida13 = GrauV1;
            }
        }
    }

    if (GrauD3 != 0 && GrauV2 != 0) {
        if (GrauD3 < GrauV2) {
            if (Saida14 == 0 || Saida14 < GrauD3) {
                Saida14 = GrauD3;
            }
        } else {
            if (Saida14 == 0 || Saida14 < GrauV2) {
                Saida14 = GrauV2;
            }
        }
    }

    if (GrauD3 != 0 && GrauV3 != 0) {
        if (GrauD3 < GrauV3) {
            if (Saida15 == 0 || Saida15 < GrauD3) {
                Saida15 = GrauD3;
            }
        } else {
            if (Saida15 == 0 || Saida15 < GrauV3) {
                Saida15 = GrauV3;
            }
        }
    }

    if (GrauD3 != 0 && GrauV4 != 0) {
        if (GrauD3 < GrauV4) {
            if (Saida16 == 0 || Saida16 < GrauD3) {
                Saida16 = GrauD3;
            }
        } else {
            if (Saida16 == 0 || Saida16 < GrauV4) {
                Saida16 = GrauV4;
```

```
                }
            }
        }

        if (GrauD3 != 0 && GrauV5 != 0) {
            if (GrauD3 < GrauV5) {
                if (Saida17 == 0 || Saida17 < GrauD3) {
                    Saida17 = GrauD3;
                }
            } else {
                if (Saida17 == 0 || Saida17 < GrauV5) {
                    Saida17 = GrauV5;
                }
            }
        }

        if (GrauD3 != 0 && GrauV6 != 0) {
            if (GrauD3 < GrauV6) {
                if (Saida18 == 0 || Saida18 < GrauD3) {
                    Saida18 = GrauD3;
                }
            } else {
                if (Saida18 == 0 || Saida18 < GrauV6) {
                    Saida18 = GrauV6;
                }
            }
        }

        saidaFuzzy =
(Saida1*(0.99)+Saida2*(0.99)+Saida3*(0.99)+Saida4*(0.66)+Saida5*(0.99)+Saida6*
(0.99)+Saida7*(0.66)+Saida8*(0.66)+Saida9*(0.99)+Saida10*(0.66)+Saida11*(0.66)
+Saida12*(0.99)+Saida13*(0.33)+Saida14*(0.66)+Saida15*(0.66)+Saida16*(0.33)+Sa
ida17*(0.66)+Saida18*(0.66))/(Saida1+Saida2+Saida3+Saida4+Saida5+Saida6+Saida7
+Saida8+Saida9+Saida10+Saida11+Saida12+Saida13+Saida14+Saida15+Saida16+Saida17
+Saida18);

        if(movimento==1){//Ge1
             if(velocidade[2]>0){
                   D_F;         //Direita Frente
                   RC1=(int)saidaFuzzy*15;          //Duty cycle de 75%
                   E_F;         //Esquerda Frente
                   RC2=(int)saidaFuzzy*95;          //Duty cycle de 10%
             }else{//Giro direita
                   GiroDireita();
             }

        }else if(movimento==2){//Ge90
             if(velocidade[2]>0){
                   D_F;         //Direita Frente
                   RC1=(int)saidaFuzzy*25;          //Duty cycle de 75%
                   E_R;         //Esquerda Re
                   RC2=(int)saidaFuzzy*75;          //Duty cycle de 75%
             }else{//Giro direita
                   GiroDireita();
             }

        }else if(movimento==4){//Gd90
             if(velocidade[2]>0){
             D_R;         //Direita Re
```

```
            RC1=(int)saidaFuzzy*75;          //Duty cycle de 75%
            E_F;          //Esquerda Frente
            RC2=(int)saidaFuzzy*25;          //Duty cycle de 75%
            }else{//Giro esquerda
                  GiroEsquerda();
            }
        }else if(movimento==3){//Gd1
            if(velocidade[2]>0){
            D_F;          //Direita Frente
            RC1=(int)saidaFuzzy*95;          //Duty cycle de 10%
            E_F;          //Esquerda Frente
            RC2=(int)saidaFuzzy*15;          //Duty cycle de 75%
            }else{//Giro esquerda
                  GiroEsquerda();
            }
}


void GiroDireita(){
      D_F;
      RC1=(int)saidaFuzzy*95;
      E_F;
      RC2=(int)saidaFuzzy*15;
}

void GiroEsquerda(){
      D_F;
      RC1=(int)saidaFuzzy*15;
      E_F;
      RC2=(int)saidaFuzzy*95;
}
```