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

SIDNEY GASPARI

PROTÓTIPO DE UM ROBÔ MÓVEL AUTÔNOMO SEGUIDOR DE PAREDES INTERNAS

TRABALHO DE CONCLUSÃO DE CURSO 2

PATO BRANCO
2015

SIDNEY GASPARI

PROTÓTIPO DE UM ROBÔ MÓVEL AUTÔNOMO SEGUIDOR DE PAREDES INTERNAS

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

Orientador: Prof^{a}. Dr^{a}. Luciene de Oliveira Marin

Coorientador: Prof. Dr. Gustavo Weber Denardin

PATO BRANCO

2015

MINISTÉRIO DA EDUCAÇÃO
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Departamento Acadêmico de Informática
Curso de Engenharia de Computação

TERMO DE APROVAÇÃO

Às 10 horas e 20 minutos do dia 18 de dezembro 2014, na sala V009, UTFPR, Câmpus Pato Branco, reuniu-se a banca examinadora composta pelos professores Luciene Marin (orientadora), Gustavo Weber Denardin (coorientador), Marco Antonio de Castro Barbosa e Cesar Rafael Claure Torrico para avaliar o trabalho de conclusão de curso com o título **Protótipo de um robô móvel autônomo seguidor de paredes internas**, do aluno **Sidney Gaspari**, matrícula 1115081, 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.

Luciene Marin
Orientadora (UTFPR)

Gustavo Weber Denardin
Coorientador (UTFPR)

Marco Antonio de Castro Barbosa
(UTFPR)

César Rafael Claure Torrico
(UTFPR)

Beatriz Terezinha Borsoi
Coordenador de TCC

Marco Antonio de Castro Barbosa
Coordenador do Curso de
Engenharia de Computação

AGRADECIMENTOS

Agradeço a todos os professores por me proporcionarem o conhecimento não apenas racional, mas a manifestação do caráter e efetividade da educação no processo de formação profissional, por tanto que se dedicaram a mim, não somente por terem me ensinado, mas por terem me feito aprender. A palavra mestre, nunca fará justiça aos professores dedicados aos quais sem nominar terão os meus eternos agradecimentos.

A todos que direta ou indiretamente fizeram parte da minha formação, o meu muito obrigado.

RESUMO

GASPARI, Sidney. Protótipo de um robô móvel autônomo seguidor de paredes internas. 2015. 68f. Monografia (Trabalho de Conclusão de Curso 2) - Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

Este trabalho apresenta a construção do hardware e softwares de controle para um protótipo de robô móvel que seja hábil para locomover-se no ambiente, seguindo paredes internas. Para o projeto de hardware foi utilizado o chassi do robô Curumim e ainda, o robô contou com um processador de controle, sensores e atuadores que foram escolhidos no intuito de se construir um robô móvel autônomo. As soluções dadas aos desafios e problemas relacionados à montagem e configuração dos componentes foram apresentadas. O projeto dos softwares de controle de baixo nível partiu de abordagens clássicas que utilizam conceitos de controle PID, *on-off* e proporcional. O bom desempenho dos controles de baixo nível implementados permitiu a elaboração do controle de alto nível, o que proporcionou ao robô o comportamento de seguir as paredes internas de um ambiente aberto, com ou sem presença de obstáculos. Os mapeamentos das trajetórias do robô, mediante as diferentes combinações dos controladores de baixo nível e configurações de ambiente, foram apresentados e discutidos.

Palavras-chave: Robôs móveis autônomos, sensores, atuadores, navegação, mapeamento de trajetória.

ABSTRACT

GASPARI, Sidney. Protótipo de um robô móvel autônomo seguidor de paredes internas. 2015. 68f. Monografia (Trabalho de Conclusão de Curso 2) - Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco. Pato Branco, 2015.

This work presents the developed hardware and control software to a mobile robot prototype able to move around its environment and following internal walls. The Curumim robot chassis was used to the hardware project, in addition, the control processor, the sensors and actuators was chosen in order to build an autonomous mobile robot. Solutions related to the challenge of installing and configuring the components were presented. The low-level control software proposed includes classical concepts such as PID, on-off and proportional control. The good performance of the low-level controllers implemented has supported the development of the high-level control, which allows to the robot to follow internal walls in an open environment, with or without the presence of obstacles. The robot's trajectory mappings obtained by different combinations of low-level controllers and environments were presented and discussed.

Keywords: Autonomous mobile robots, sensors, actuators, navigation, trajectory mapping.

LISTA DE FIGURAS

LISTA DE SIGLAS

VHDL	Do acrônimo em inglês *Very High Speed Integrated Circuit*
RL	Do acrônimo em inglês *Reinforcement Learning*
RNA	Rede neural artificial
SLAM	Do acrônimo em inglês *Simultaneous Localization And Mapping*
CI	Circuito integrado
MO	Matriz de obstáculos
PD	Controlador proporcional derivativo
PID	Controlador proporcional integral derivativo
PWM	Do acrônimo em inglês *Pulse-Width Modulation*
SONAR	Do acrônimo em inglês *Sound Navigation and Ranging*
EHW	Hardware evolutivo, do acrônimo em inglês *Evolvable hardware*
CPU	Do acrônimo em inglês *Central Processing Unit*
GDL	Grau de liberdade
CC	Corrente contínua
I2C	Do acrônimo em inglês *Inter-Integrated Circuit*
TTL	Do acrônimo em inglês *Transistor-Transistor Logic*
IDE	Do acrônimo em inglês *Integrated Development Environment*

SUMÁRIO

1 INTRODUÇÃO E CONTEXTUALIZAÇÃO

Robôs móveis pertencem a uma classe de robôs que se distinguiu de seus antecessores, os robôs manipuladores, por serem aptos à locomoção. Segundo Arkin (1998), robôs podem ser diferenciados sob diversos aspectos, por exemplo, em termos de tamanho, materiais com que são feitos, como os mesmos estão integrados, atuadores e sensores que utilizam, a maneira como se movem e em especial como se constitui seu sistema computacional embarcado. Sendo assim, somente a estrutura física de um robô não é suficiente para diferenciá-lo. Robôs devem apresentar comportamentos providos por um sistema de controle interno, que lhes conferem a habilidade de se mover de maneira segura e coordenada.

É fato, que desde o surgimento dos primeiros mecanismos robotizados, é grande a expectativa por robôs que possam auxiliar ou até mesmo substituir pessoas em suas tarefas, em especial a tarefas domésticas. Isto tem inspirado, por exemplo, a área de pesquisa em robótica de serviço, que se encontra em expansão. Segundo Andersson *et al.* (1999), acredita-se fortemente que, em um futuro próximo, existirão pequenos robôs que darão assistência as tarefas domésticas. Um excelente exemplo é o robô aspirador de pó a vácuo, TriLobote da EletroLux, lançado ao público desde 1998 e ainda disponível no mercado. O potencial de aplicação desse tipo de robô é enorme, desde a tarefa maçante de aspirar pó, a tarefas mais complexas como arrumar toda a casa após um jantar para amigos. Progressos recentes têm proporcionado avanços na concepção dos robôs chamados “inteligentes”, entretanto é fato que esse tipo de robô ainda está longe de ser popularizado. Atualmente, são poucos os sistemas de robôs móveis produzidos para esta finalidade, devido à complexidade das tarefas envolvidas. Exemplos de sistemas móveis oferecidos no mercado incluem a plataforma robótica Help-Mate para distribuição de refeição e chapas de raio-x em hospitais, e o RoboKent, robô varredor de chão produzido pela Kent Corporation. Ambos produzidos em pequena escala, em ordem de centenas. Há também cenários como competições de robótica e missões de resgate de humanos, onde robôs móveis autônomos se encontram presentes e tornam-se o principal foco da pesquisa e desenvolvimento, em ambas as vertentes de hardware e software desse tipo de robô.

Sob esta motivação e devido à natureza multidisciplinar e desafiadora dessa

área de pesquisa, o trabalho de conclusão de curso apresentado focou-se em dois aspectos, o projeto físico de um robô móvel e a construção de um sistema de controle de baixo nível que o torne apto a se locomover de forma autônoma em um ambiente aberto, com presença de obstáculos. O controle de baixo nível servirá de base para que arquiteturas de controle de alto nível possam ser implementadas, visando a posteriori a construção de um protótipo de robô de serviço.

1.1 DEFINIÇÃO DO PROBLEMA

O principal problema que foi abordado neste trabalho consistiu na montagem e configuração do *hardware* e na implementação de *softwares* de controle de um robô móvel, que foram divididos em baixo e alto nível. O controle de baixo nível compreendeu implementar mecanismos que tratam dos problemas envolvidos na movimentação do robô, tais como, deslizamento das rodas e erros de posicionamento. Já a implementação do controle de alto-nível permitiu que o robô seguisse paredes internas, evitando colidir com obstáculos.

Isso consiste da tarefa do robô navegar o que diz respeito à tarefa do robô navegar de maneira autônoma e segura, mediante à percepção de seu ambiente. Também foi tratado o problema de mapeamento de trajetórias realizadas pelo robô, por meio dos diferentes sistemas de controle de baixo nível implementados.

1.2 JUSTIFICATIVA

Conceber a montagem, a configuração e o controle confiável de um robô móvel que seja autônomo e que navegue de maneira segura em seu ambiente, é uma tarefa desafiadora e requer o emprego de diversas áreas de conhecimento, que foram estudadas durante o curso de Engenharia de Computação. Além disso, a realização desse trabalho de conclusão de curso permitiu a alternativa de se construir um robô móvel econômica viável, ao invés de se dispor de opções de robôs móveis disponíveis no mercado. É fato que existem vários kits de robôs oferecidos para a venda, porém seu preço elevado, muitas vezes inviabiliza sua compra e também a sua utilização, devido a restrições físicas ou de *software*. Um exemplo é o robô Curumim[1] e o Lego Mindstorms[2], robôs pertencentes ao Departamento Acadêmico de Informática da Uni-

[1] **Para mais informações sobre o robô Curumim, consultar o sítio** `http://www.xbot.com.br/educacional/curumim`**. Acessado em 27/01/2014.**

[2] `http://www.lego.com/en-us/mindstorms/?domainredir=mindstorms.lego.com` **. Acessado em 27/01/2014.**

versidade Tecnológica Federal do Paraná, câmpus Pato Branco, os quais possuem limitações quanto aos sensores, alto preço e interface educacional, voltada para o público de estudantes do ensino médio. Todos esses aspectos restringiriam os objetivos a serem desenvolvidos nesse trabalho.

1.3 OBJETIVO GERAL

Este trabalho tem por objetivo geral a montagem e a configuração do *hardware*, bem como a implementação de funções de controle de um robô móvel que seja hábil a locomover-se de maneira autônoma e segura, com base na percepção de seu ambiente.

Para se alcançar o objetivo geral, os seguintes objetivos específicos foram definidos e são apresentados na próxima Seção.

1.3.1 OBJETIVOS ESPECÍFICOS

- Escolher componentes, tais como, microprocessador, sensores e atuadores que viabilizassem a concepção do *hardware* do robô.
- Montar o *hardware* a partir do chassis do robô Curumim, integrar e configurar os componentes obtidos no ítem anterior.
- Implementar o controle de baixo nível do robô visando resolver os problemas inerentes ao controle de movimento, tais como, deslizamento nas rodas e erros de posicionamento.
- Implementar a tarefa de navegação que consistiu do robô seguir paredes de maneira autônoma e evitando obstáculos, o qual define o sistema de o controle de alto nível do robô.
- Prover uma representação de como o robô executou sua tarefa de navegação (mapeamento de trajetória) mediante a combinação dos diferentes sistemas de controle de baixo nível.

1.4 ORGANIZAÇÃO DO TRABALHO

Este trabalho de conclusão de curso está dividido em seis capítulos, os quais essa introdução é o primeiro deles.

O Capítulo 2 apresenta uma revisão sobre o arcabouço teórico relacionado ao *hardware* e a sistemas de controle que implementam robôs móveis autônomos.

O Capítulo 3 descreve o desenvolvimento das etapas de construção de um robô móvel autônomo, que compreendem os seguintes processos: i) de montagem e configuração do *hardware* do robô, ii) a implementação das funções de controle de baixo nível, iii) a implementação do sistema de controle de alto nível e iv) um método para a construção de uma forma de representação do comportamento executado pelo robô, mediante sua percepção do meio e a combinação das funções de controle de baixo nível implementadas.

O Capítulo 4 apresenta os resultados obtidos com o desenvolvimentos das etapas descritas no capítulo anterior e também relata alguns problemas encontrados durante a execução do projeto.

Finalmente, no Capítulo 5 são apresentas as conclusões e considerações referentes ao desenvolvimento desse trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta uma introdução à robótica móvel, bem como uma revisão teórica dos conhecimentos envolvidos na construção do *hardware* e do *software* de robôs móveis.

2.1 INTRODUÇÃO A ROBÓTICA

O sucesso de robôs reais primeiramente depende da boa escolha de sensores, atuadores e outros componentes de *hardware*, apropriados à tarefa a ser realizada. Na fase de montagem do *hardware* de um robô, são considerados principalmente, aspectos de desempenho e utilização bem sucedida em pesquisas relacionadas.

Ainda com relação aos sensores, em muitas aplicações robóticas, a utilização de sensores ultra-sônicos é amplamente aplicada à percepção do ambiente e detecção de obstáculos. Na literatura, esses sensores são muito bem sucedidos em termos de eficiência, custo, tempo de processamento e precisão. Um exemplo de trabalho onde esses sensores são utilizados no sistema de localização pode ser encontrado em (KIM; KIM, 2013).

A área da inteligência artificial sempre incorpora à robótica possibilidades de inovações, o que volta a atenção dos pesquisadores mais para aspectos de *software* do que de *hardware* (NEHMZOW, 2000). Entretanto, segundo (NOGUEIRA, 2013), um fato muito importante é o uso de *hardware* flexível, isto é, capaz de se modificar e que também tem sido base para implementações de técnicas de inteligência artificial. Esta tecnologia alimenta a imaginação dos pesquisadores, que buscam aprimorar as técnicas sobre *Hardware* Evolutivo, por exemplo. Sendo assim, o autor propôs um controlador *fuzzy* projetado especificamente em uma linguagem de descrição de *hardware*, VHDL (do acrônimo em inglês *Very High Speed Integrated Circuit*), para um robô que tem três sensores que medem a distância a um objeto, e dois motores de passo cujo controle é implementado em *hardware* usando a lógica *fuzzy* e dispositivos lógicos programáveis.

Além disso, a literatura de inteligência artificial oferece diversas técnicas

para implementação de controladores de robôs móveis autônomos, tais como, EHW-*hardware* evolutivo, que utiliza algoritmos genéticos, redes neurais das mais variadas configurações, *Reinforcement Learning* (RL), ou seja aprendizagem baseada em recompensa, que também é uma técnica que está sendo muito explorada, sendo que vários trabalhos integrando RL com a lógica *fuzzy* já foram publicados (NOGUEIRA, 2013). A lógica *fuzzy* em especial é utilizada em diversos campos na robótica móvel, como controle por comportamento, navegação autônoma, criação de mapas em ambientes desconhecidos, controle de velocidade, controle de trajetória, desvio de obstáculos, e outros. O mesmo se aplica à redes neurais artificiais.

Outra área de pesquisa bastante desafiadora é a área de robótica de serviço. Segundo Lim *et al.* (2011), uma dificuldade significante para um robô de serviço é a execução de tarefas complexas em ambientes reais. Por exemplo, não é um problema trivial encontrar objetos que são parcialmente observáveis e estão localizados em um lugar o qual não é idêntico, mas próximo do lugar onde o robô o avistou previamente. Para tratar de maneira efetiva este problema, os autores propõem uma rede semântica como representação de conhecimento do robô. Outras pesquisas recentes nessa área podem ser vistas nos trabalhos de (LUO; LAI, 2012), (TAPUS *et al.*, 2008), entre outros. Como exemplo de robôs de serviço disponíveis no mercado, tem-se o robô Smarbo (do acrônimo em inglês “Smart Robot”), que é um robô de limpeza projetado exclusivamente para uso residencial, produto da empresa japonesa Toshiba. Ele é constituído de CPUs, uma câmera frontal e um conjunto de 38 sensores (detecção, aceleração, giroscópios, entre outros (TOSHIBA, 2013).

2.2 *HARDWARE* DE ROBÔS

O sucesso de robôs reais depende do projeto de sensores e atuadores adequados. A seguir é apresentada uma breve introdução sobre os componentes envolvidos na construção de um robô móvel.

2.2.1 SENSORES

Segundo Russell e Norvig (2004), os sensores são a interface perceptiva entre o robô e seu ambiente. Os sensores passivos como câmeras, são observadores do ambiente, ou seja, eles captam sinais gerados por outras fontes no ambiente. Os sensores ativos, como o sonar, enviam energia ao ambiente, e contam com o fato de que essa energia é refletida de volta para o sensor. Os sensores ativos tendem

a fornecer mais informações que os sensores passivos, mas ao custo de maior consumo de energia e com um perigo de interferência, quando vários sensores ativos são usados ao mesmo tempo. Quer sejam ativos ou passivos, os sensores podem ser divididos em três tipos, despendendo do fato de registrarem distâncias até objetos, imagens inteiras do ambiente ou propriedade do próprio robô.

Muitos robôs móveis fazem uso de telêmetros, que são sensores que medem a distância até objetos próximos. Um tipo comum telêmetro é o sensor de sonar, também conhecido como transdutor ultra-sônico. Os sensores de sonar emitem ondas sonoras direcionais, que são refletidas por objetos, com uma parte do som que volta ao sensor. Desse modo, o tempo e a intensidade desse sinal de retorno, transmitem informações sobre a distância até objetos próximos.

Em robôs móveis, os decodificadores de eixos que informam as revoluções das rodas podem ser usadas para hodometria - a medição da distância percorrida. Infelizmente, as rodas tendem a travar e patinar, e portando a hodometria só é exata quando se consideram distâncias curtas. Os sensores inerciais, como o giroscópio, podem ajudar, mas não são capazes de impedir sozinhos a acumulação inevitável de incertezas de posição.

2.2.2 ATUADORES

Segundo Russell e Norvig (2004), os atuadores são os meios pelos quais os robôs se movem e alteram a forma de seus corpos. Para entender o projeto de atuadores, utiliza-se o conceito de grau de liberdade (GDL), que conta um grau para cada direção independente em que o robô ou um de seus atuadores pode se mover.

Para robôs móveis, os GDLs não são necessariamente iguais ao número de elementos movidos. Por exemplo, considere um carro médio: ele pode se mover para frente e para trás, e também pode virar à esquerda ou à direita, o que lhe dá dois GDLs. Em contraste, uma configuração cinemática de um carro é tridimensional: em uma superfície plana aberta, é possível manobrar facilmente um carro para qualquer ponto (x, y), em qualquer orientação. Desse modo, o carro tem 3 graus efetivos de liberdade, mas 2 graus controláveis de liberdade. Diz-se que um robô é não-holonômico se ele tem mais GDLs efetivos que GDLs controláveis, e holonômico se os dois números são iguais. Os robôs holonômicos são mais fáceis de controlar - seria muito mais fácil estacionar um automóvel que pudesse se mover para os lados e também para frente e para trás - mas os robôs holonômicos também são mecanicamente mais complexos.

Em sua maioria, os braços de robôs são holonômicos, e a maioria dos robôs moveis é não-holonômica, o que é o caso do robô que será construído nesse trabalho.

2.2.2.1 PONTE H (DRIVER DE POTÊNCIA)

Ponte H é um circuito desenvolvido para controle de direção de motores. Este circuito é basicamente composto por 4 chaves, que com diferentes combinações, fazem com que a corrente possa fluir em sentidos opostos. Ele é muito utilizado em controle de motores de corrente contínua (CC), que permite o giro no sentido horário e anti-horário do motor. Uma ponte H pode ser montada de várias formas e com diferentes componentes tais como, MOSFET, transistores e até mesmo o circuito todo pode estar inseridos em um único CI. O circuito básico de uma ponte H é representado na figura 1. Quando as chaves S1 e S4 estiverem abertas e as chaves S2 e S3 fechadas, o motor gira em um sentido. Quando a situação oposta ocorrer, S1 e S4 estiverem fechadas e S2 e S3 abertas, o giro ocorrerá no sentido contrário. Deve-se ter cuidado principalmente com as chaves S1 e S2, assim como S3 e S4 que não podem ser fechadas simultaneamente, o que causaria um curto circuito na fonte (PESSOA *et al.*, 2011).

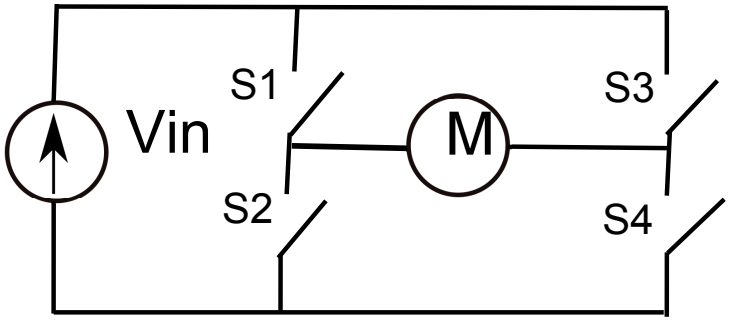


Figura 1: Esquema funcional de uma ponte H. Fonte: autoria própria.

O circuito de ponte H tem como principal função conectar circuitos com lógica TTL em circuitos de potência, como por exemplo, conectar o microcontrolador com motores, pois os microcontroladores não têm capacidade de suportar corrente suficiente para o acionamento de outros dispositivos elétricos como motores. Nesse caso a ponte H funciona como interface de potência, amplificando o sinal do microcontrolador e assim controlando qualquer que seja o dispositivo/sistema desejado.

2.2.3 MICROCONTROLADORES

Segundo Jones *et al.* (1998), um microcontrolador combina pouco espaço, baixo consumo de energia, habilidades computacionais de um microprocessador barato, com a proficiência do processamento de sinais de circuitos discretos. Em particular, microcontroladores geralmente incluem comodidades acopladas, tais como uma linha serial (para comunicação direta com um terminal ou computador), conversores analógico-digitais, timers (para capturar eventos ou ativação do *hardware*), e contadores de pulso. Essas características simplificam grandemente o projeto do sistema. Antes do advento do microcontrolador, para adquirir percepção e ação de um robô, por exemplo, era necessário construir um sistema que consistia de numerosas placas de circuito impresso conectadas. Um ou mais cartões ficavam reservados para o processador e a memória, cartões separados eram necessários para a função de sensores e atuadores. Hoje, o tamanho, complexidade, consumo de potência, e custo de um determinado sistema pode ser reduzido usando um microcontrolador para executar todas as tarefas de processamento em um chip.

2.3 PERCEPÇÃO ROBÓTICA

Segundo Russell e Norvig (2004), a percepção é o processo pelo qual os robôs mapeiam medições de sensores em representações internas do ambiente. A percepção é difícil porque em geral, os sensores são ruidosos e o ambiente é parcialmente observável, imprevisível e frequentemente dinâmico. Como uma regra prática, as boas representações internas têm propriedades: elas contêm informações suficientes para o robô tomar as decisões corretas, são estruturadas de tal modo que possam ser atualizadas com eficiência e são naturais, no sentido de que as variáveis internas correspondem a variáveis de estado naturais do mundo físico.

2.3.1 LOCALIZAÇÃO

A localização é um exemplo genérico de percepção dos robôs. Trata-se do problema de determinar onde estão os objetos. A localização é um dos problemas de percepção mais pervasivos em robótica, porque o conhecimento sobre os locais em que estão os objetos é o núcleo de qualquer interação física bem-sucedida. Por exemplo, os manipuladores de robôs devem conhecer a posição dos objetos que manipulam. Os robôs de navegação têm de saber onde estão a fim de encontrar seu caminho até as posições objetivo.

O problema de localização se apresenta em três variedades de dificuldade crescente. Se a pose inicial do objeto a ser localizado for conhecida, a localização será um problema de rastreamento. Os problemas de rastreamento se caracterizam pela incerteza limitada. Um problema de localização global se transforma em problema de rastreamento uma vez que o objetivo de interesse é localizado, mas também envolvem fases em que o robô tem de administrar incertezas muito amplas. Por fim, pode-se “sequestrar” o objeto que o robô está tentando localizar. A localização sob condições tão tortuosas é usado para testar a robustez de uma técnica de localização sob condições extremas.

2.3.1.1 LOCALIZAÇÃO ULTRA-SÔNICA

Os sensores ultra-sônicos são amplamente utilizados para a percepção do ambiente e detecção de obstáculos em muitas aplicações robóticas. Estes sensores são muito bem sucedidos em termos de eficiência de custos, tempo de processamento e precisão. Em (KIM; KIM, 2013) pode ser visto um sistema de localização por sensores ultra-sônicos que funcionam como faróis ativos.

Já o modelo do ultra-som descrito em (KONOLIGE, 1997) calcula as probabilidades condicionais de um modelo matemático do comportamento do sensor de sonar, tendo em vista que o sonar é uma boa opção para calcular a distância. Transdutores de sonar emitem um padrão de energia, como ilustrado na Fig. 2. Há um lóbulo principal, cuja largura depende de um número de fatores, incluindo o tamanho do elemento transdutor e a frequência do pulso ultra-sônico. Para transdutores polaroid eletrostáticos, a largura nominal do feixe a 50 kHz é de cerca de 25^0 graus. Embora exista uma calda insignificante de até $0,5$ metros, aproxima-se o padrão de feixe por um cone cilíndrico.

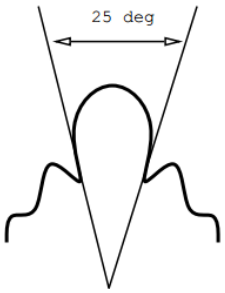


Figura 2: Padrão de energia gerado pelo sonar. Adaptada de (KONOLIGE, 1997)

Muito utilizado em robótica e aplicações industriais, o telêmetro ultra-sônico foi projetado para detectar a distância de um objeto, refletindo nele um pulso sonoro

ultra-sônico e verificando o tempo que demora até que o pulso retorne.

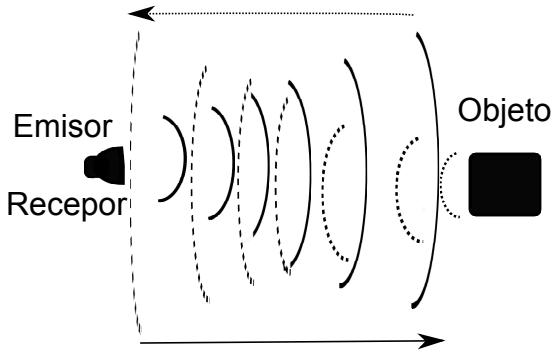


Figura 3: Princípio de funcionamento do sensor de distância ultra-sônico.

Como pode ser observado na Fig. 3, esse princípio também é conhecido como SONAR (*Sound Navigation and Ranging*, ou navegação e determinação de distância pelo som), e é utilizado em submarinos para detectar a distância de outras embarcações marítimas ou a proximidade de objetos. O mesmo princípio também é utilizado por morcegos para detectar a presença de obstáculos e presas. Sendo assim, o cálculo da distância está baseado na medida do eco produzido pelo som propagado até o obstáculo.

2.3.1.2 HODOMETRIA

Segundo Boas (2011), a hodometria é o método de localização mais utilizado em robótica móvel, dada a sua simplicidade de utilização e implementação. Sabe-se que a hodometria fornece uma boa precisão de curto prazo, é barata e permite taxas de amostragem muito altas. Um sistema hodométrico consegue, com o auxílio de *encoders*, determinar a rotação dos eixos dos atuadores responsáveis pela movimentação do robô, em um determinado intervalo de tempo. Integrando esses valores obtém-se o cálculo do deslocamento realizado nesse intervalo. No entanto, a idéia fundamental de hodometria é a integração das informações do movimento incremental ao longo do tempo, o que leva inevitavelmente à acumulação de erros. Particularmente, o acúmulo de erros de orientação irá causar erros de posição de grande porte, que aumentam proporcionalmente com a distância percorrida pelo robô.

Existem diversos tipos de robôs, cada um com a sua funcionalidade e com um sistema de locomoção adaptado. Por esse motivo, durante a navegação é necessário fazer a distinção entre dois tipos de robôs, os holonômicos e os não-holonômicos, sob risco de se incorrer em erro durante a estimativa da posição, ao

longo do movimento do robô.

É o sistema de atuação de um robô que o define como sendo ou não holonômico. Para um robô holonômico, o número de graus de liberdade controlados por ele tem de ser igual ao número de graus de liberdade total. Pelo contrário, o mesmo já não acontece com os robôs não-holonômicos, visto que estes possuem um número total de graus de liberdade superior àqueles que conseguem controlar. Dando como exemplo, um sistema holonômico tem a capacidade de parar em uma dada orientação e movimentar-se em qualquer direção sem constrangimento. Por outro lado, para robôs não holonômicos o movimento já não é independente da orientação, isto quer dizer que a direção tomada por um sistema qualquer está inerentemente associada à orientação com que este se encontra a cada instante, admitindo que não há escorregamento (MIGUEL, 2012).

2.3.2 CONTROLE PID

O controle PID combina as vantagens do controlador tipo P, PI, PD. A ação integral está diretamente ligada à precisão do sistema sendo responsável pelo erro nulo em regime permanente. O efeito desestabilizador do controle PI é contrabalanceado pela ação derivativa que tende a aumentar a estabilidade relativa do sistema ao mesmo tempo em que torna a resposta do sistema mais rápida devido ao seu efeito antecipatório, segundo Matas (2012).

O controlador PID é dado pela Equação 1. O termo derivativo K_d tem o papel de aumentar o amortecimento e em geral, melhorar a estabilidade de um sistema. Intuitivamente, a ação do termo derivativo pode ser entendida quando considera-se um controlador PID em um instante em que, o erro é momentaneamente nulo, mas sua taxa de variação, não. O termo proporcional K_P não terá contribuição alguma sobre a saída, mas o termo derivativo, sim. Este último tem assim o papel de fazer com que o controlador se antecipe à ocorrência do erro. Essa característica de tornar o controlador sensível à taxa de variação do erro, tem claramente o efeito de aumentar o amortecimento do sistema. O termo integral K_I é o somatório de todo erro acumulado do instante zero até o momento da ação de controle, se muito grande pode gerar instabilidade, segundo (OGATA, 1990).

$$u(t) = K_p e(t) + K_I \int_0^t e(t)dt + K_d \frac{d}{dt} e(t) \tag{1}$$

A ação derivativa, associada à ação proporcional, corresponde ao acréscimo

de um zero ao sistema, atuando beneficamente no regime transitório, tendendo a aumentar a estabilidade relativa do sistema e reduzindo o tempo de acomodação, contudo, contrapondo-se a estas vantagens, ele aumenta o tempo de subida e, por não atuar no regime permanente, não corrige o erro de estado estacionário. Este compensador, por introduzir um avanço de fase, é considerado na bibliografia como um caso particular de um compensador em avanço (SILVA, 2000).

A Fig.4 ilustra o diagrama de blocos de um sistema em malha fechada utilizando um controlador PID.

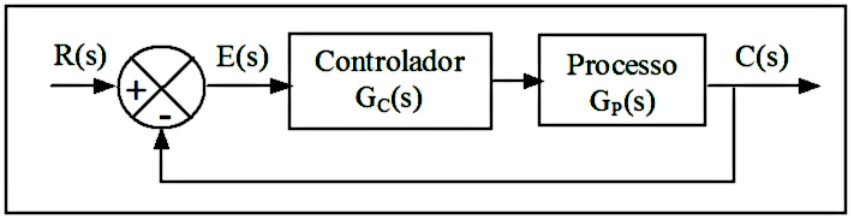


Figura 4: Diagrama de blocos do controle PID.

A função $R(s)$ é a entrada do sistema que corresponde a uma referência de posição ou de tensão do processo. $E(s)$ é o erro gerado com relação a referência de entrada e a saída. A função $Gc(s)$ é o controlador com os ganhos K_p, K_ie K_d. $Gp(s)$ é a planta do sistema que será aproximado, e $C(s)$ é a saída do sistema a cada instante.

2.3.3 MAPEAMENTO

Segundo Boas (2011), o problema clássico do mapeamento robótico pode ser dado como segue. Imagine um robô que não recebe um mapa de seu ambiente. Então, um problema natural em robótica é criar algoritmos que permitam aos robôs fazerem o mesmo. Conforme descrito em (FILLIAT; MEYER, 2003a), a navegação inteligente pode ser associada à navegação baseada em mapas, e a mesma compreende 3 processos básicos:

i. Aprendizado de mapa: processo de memorizar os dados adquiridos pelo robô durante exploração em uma representação adequada.

ii. Localização: processo de derivação da posição corrente do robô dentro do mapa.

iii. Planejamento de caminho: processo de escolher um curso de ação para alcançar o objetivo, dada uma posição corrente.

O terceiro processo é dependente do primeiro e segundo, ou seja, para planejar ações na direção do objetivo é necessário que haja o mapa do ambiente entre

a posição corrente e objetivo. Os dois primeiros processos são inter-dependentes. Esta interdependência torna complexo o problema da localização e aprendizado de mapa simultâneos (problema SLAM, do acrônimo em inglês Simultaneous Localization And Mapping). Uma ampla revisão de estratégias de localização, aprendizado de mapa e planejamento de caminho é encontrada nos trabalhos (FILLIAT; MEYER, 2003a) e (FILLIAT; MEYER, 2003b), neles são analisados vários métodos de navegação baseada em mapas, dentre eles muitos inspirados biologicamente.

Apesar de muitos avanços nessa área de pesquisa, o problema SLAM encontra-se ainda em aberto na literatura. Esse problema também está na base da área de pesquisa em robótica de serviço. Alguns exemplos de trabalhos relacionados ao problema da auto-localização, mapeamento e navegação são descritos em (KIM; KIM, 2013), (CHEN *et al.*, 2002), (IM *et al.*, 2002).

Portanto, o processo de construção de mapas por robôs móveis necessita receber informações tanto do meio explorado quanto do movimento e estado do robô. Tem-se então duas origens distintas de informações que podem ser usadas para navegação baseada em mapas. A primeira origem é a odométrica, envolvendo o giroscópio e o acelerômetro, que fornecem informações internas sobre os movimentos e estado atual do robô, como por exemplo, velocidade, aceleração, e direção das rodas. A segunda origem é sensorial, por meio do sensor de distância, o qual fornece informações externas sobre o ambiente.

2.3.3.1 MAPAS MÉTRICOS

Segundo Boas (2011), o objetivo do mapa métrico é representar o ambiente com um alto nível de detalhe, tratando as informações coletadas pelos sensores e relacionando-as com o tamanho e forma dos objetos e os limites das áreas livres para navegação. Para que isso seja possível, os mapas métricos dividem o espaço 2D ou 3D em células regulares, as quais representam características de interesse sobre o local, como por exemplo, ocupação (possui algum obstáculo que impede a navegação), a probabilidade da área estar vazia (pode ser atravessada pelo robô), dados como o conjunto de estados observados desde o início da navegação. Um exemplo clássico de mapa métrico é o mapa de grade de ocupação, que representa o ambiente como uma matriz de forma que cada célula contém a probabilidade de que esse espaço esteja ocupado por um obstáculo ou não. Em geral, os mapas métricos exigem um alto esforço computacional e um grande espaço na memória, pois a atualização do mapeamento é um processo que envolve somente algumas células

de cada vez e o mapa tende a crescer conforme o tamanho da área mapeada. No entanto, o mapa métrico salva as informações no próprio domínio do sinal do sensor, assim a tarefa de localização é executada de forma mais simples e devido a alta densidade de informação o resultado da localização é mais preciso e menos sujeito a ambiguidades.

2.4 ARQUITETURAS DE *SOFTWARE* DE ROBÓTICA

Segundo Russell e Norvig (2004), uma arquitetura de *software* é uma metodologia para estruturar algoritmos, o que inclui linguagens e ferramentas utilizadas. As arquiteturas de *software* modernas para robótica devem decidir como combinar controle reativo e controle deliberativo baseado em modelos. Em vários aspectos o controle reativo e o controle deliberativo têm pontos fortes e deficiências ortogonais. O controle reativo é orientado para sensores e é apropriado para a tomada de decisões de baixo nível em tempo real. Porém o controle reativo raramente gera uma solução plausível no nível global, porque as decisões de nível global dependem de informações que não podem ser percebidas no instante da tomada de decisões. Para tais problemas o controle deliberativo é mais apropriado.

Consequentemente, a maioria das arquiteturas de robôs utiliza técnicas reativas nos níveis mais baixos de controle, com técnicas deliberativas nos níveis mais altos. Arquiteturas que combinam técnicas reativas e deliberativas costumam ser chamadas arquiteturas híbridas.

3 DESENVOLVIMENTO

Este capítulo apresenta as etapas desenvolvidas durante o andamento do trabalho de conclusão curso. Os principais problemas abordados foram: 1) projetar um robô físico e 2) projetar um sistema de controle específico, que proporcione ao mesmo a habilidade de navegar de forma eficiente e segura em seu ambiente, desconhecido e com relação aos obstáculos presentes. Desta forma, o robô terá que evitar obstáculos e explorar seu ambiente desconhecido, com base na com base na informação dos sensores, para então realizar, com autonomia e precisão, comportamento de navegação autônoma.

O esquema da organização, montagem e comunicação dos componentes de *hardware* está descrito na Figura 5. Cada componente do robô se conecta diretamente ou indiretamente ao microcontrolador. O sensor de distância é o sensor principal de todo o sistema de controle. Na figura estão representadas todas as ligações físicas entre os componentes.

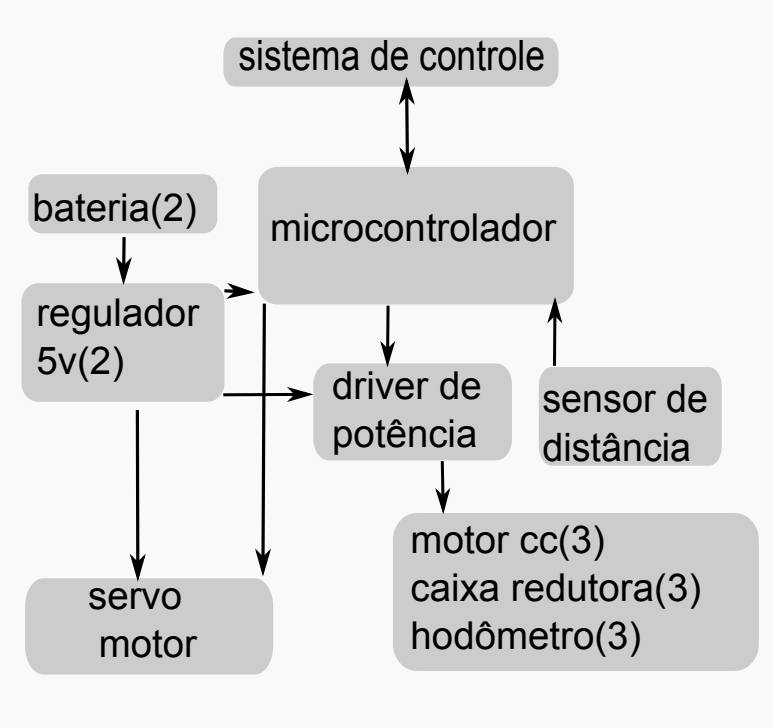


Figura 5: Projeto de *hardware* do robô: esquemático da disposição e comunicação entre dispositivos.

O *hardware* deve dar suporte à realização das tarefas que serão implemen-

tadas por meio do sistema de controle. Os elementos de *hardware* devem ser acoplados de forma precisa e funcional, de maneira que o microcontrolador seja o componente principal. O microcontrolador terá o sistema de controle embarcado e será responsável pela comunicação entre *hardware* e *software* em diferentes níveis. Neste projeto o microcontrolador receberá dados dos sensores de distância e hodômetros, que são as entradas do sistema de controle que fornece como saída a potência a ser transmitida para os motores, por meio do driver de potência.

3.1 PROJETO DE *HARDWARE*

Nesta seção serão descritos os detalhes envolvidos na construção do *hardware* do robô. Todos os componentes utilizados foram escolhidos com base em uma pesquisa, considerando-se quesitos de qualidade, preço, disponibilidade e facilidade de configuração.

3.1.1 DESCRIÇÃO DO ROBÔ

O projeto físico do robô iniciou-se a partir da utilização da estrutura mecânica, motores e hodômetros do robô Curumim (Figura 6), pertencente ao departamento acadêmico de informática da UTFPR, câmpus Pato Branco. O emprego da estrutura mecânica (chassis) se deve à qualidade de seu material e à disposição com que os hodômetros e os motores, embutidos com suas respectivas caixas redutoras, facilita o desenvolvimento final do robô. No *site*XBot (2014)são encontradas mais informações sobre como obter e utilizar o robô Curumim.

O robô desenvolvido neste trabalho contou com apenas um microcontrolador, um sensor de distância, um driver de potência, três caixas redutoras, três hodômetros, três motores, um servo-motor e duas baterias. A estrutura física do robô final é composto por dois blocos, separados por uma placa de fenolite. Nesta placa foram feitas várias trilhas com solda para prover a alimentação dos componentes e a transmissão de dados de controle, deixando a estrutura mais limpa e organizada, eliminando parcialmente problemas corriqueiros como mau contato entre os componentes.

Na placa de fenolite foram inseridos: resistores que têm o papel de limitar a corrente de entrada/saída do microcontrolador na aquisição de dados dos sensores e acionamentos dos atuadores, um diodo de roda livre, para proteção do circuito, dois circuitos com regulador de tensão 5 volts para alimentação do microcontrolador e servo motor, leds e conectores para sensores e ponte H, ocupando um espaço de

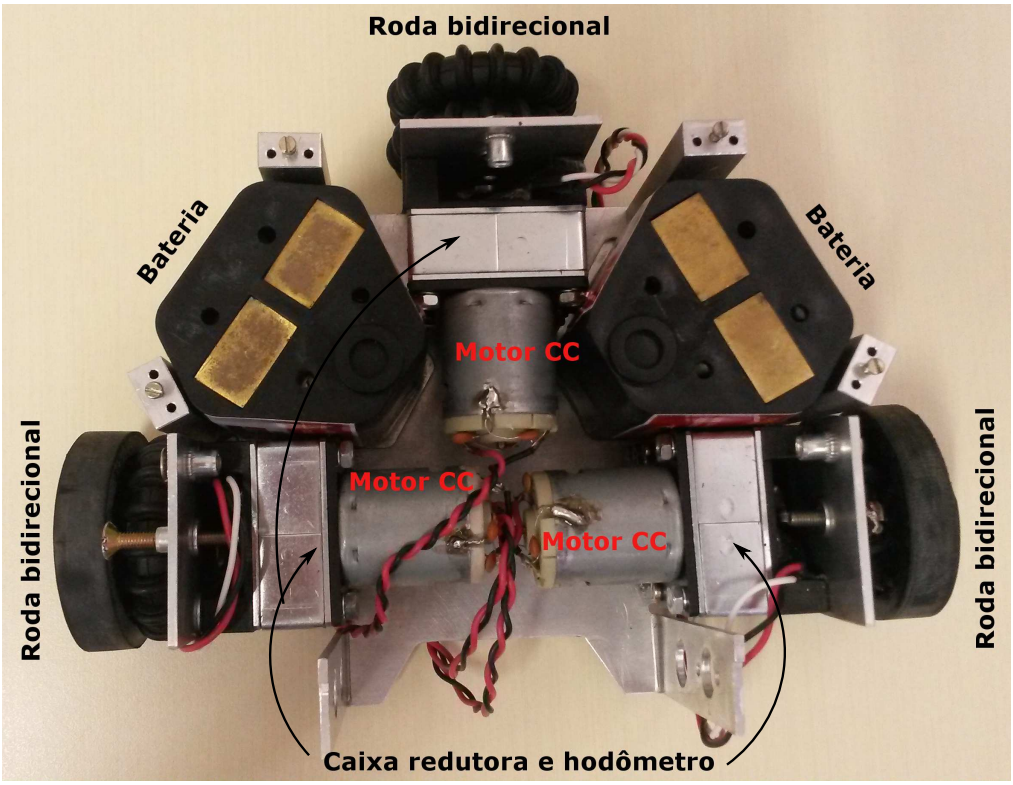


Figura 6: Chassi do robô Curumim.

cerca de $18 \times 18\ cm^2$, com $15\ cm$ de altura.

3.1.2 DESCRIÇÃO DO *HARDWARE*

O *hardware* foi projetado considerando-se principalmente aspectos de desempenho, preço acessível e funcionamento adequado do mesmo. Por isso, alguns componentes que foram considerados na primeira etapa do trabalho de conclusão de curso (TCC1) foram substituídos por outras opções, similares ou iguais, devido a problemas encontrados durante o desenvolvimento do projeto. Essas considerações são apresentadas no Capítulo Resultados e Discussões 4.8.

A Figura 7 mostra alguns elementos essenciais que foram utilizados para a construção do robô final e são eles: sensor ultra-sônico, servo motor, drives de potência (pontes H) e microcontrolador. Além desses componentes também foram utilizados hodômetros, baterias, motores cc, caixas redutoras que podem ser observados na Figura 6.

Como pode ser visto na Figura 9, na estrutura superior do robô final foram acoplados o microcontrolador, o sensor de distância, a ponte H e o servo motor. Essa camada superior está separada da camada inferior por uma placa de fenolite. A estrutura inferior é composta do chassis, onde são inseridas as baterias, caixas redutoras

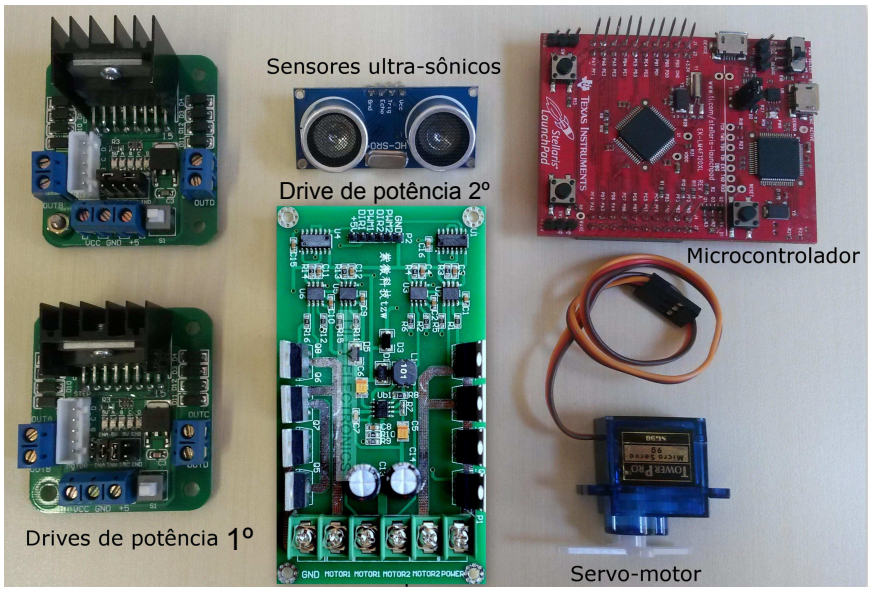


Figura 7: Componentes de *hardware* obtidos: driver de potência (pontes H), sensores ultra-sônicos, micro-controlador e servo-motor.

e os hodômetros.

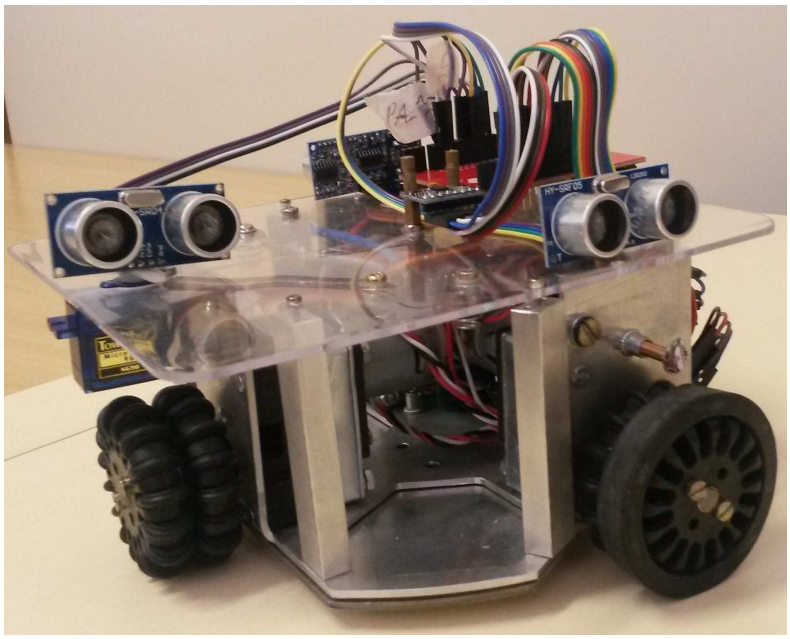


Figura 8: Montagem do protótipo apresentado no TCC1.

Já a Figura 8, mostra o primeiro protótipo do robô construído no desenvolvimento do TCC1. Foram realizadas alterações neste protótipo, tais como a substituição da placa de acrílico por uma de fenolite. Essa substituição permitiu a inclusão de circuitos que não foram planejados, tais como: resistências, diodo de roda livre, circuito condicionador de sinais do hodômetro e reguladores de tensão com saída $5V$. Outra alteração bastante visível foi a troca das duas ponte H $L298$ por outra baseada em MOSFET. O projeto inicial também contava com três sensores de distância, que foram

substituídos por apenas um. Este sensor foi acoplado sobre um servo motor, para executar um giro de 180^o na frente do robô, fazendo todas as leituras necessárias para localização. O mesmo deve ficar a 3 cm dos extremos do robô, dado que os sensores não conseguem realizar a leitura a menos de 2 cm de distância.

Além disso, o protótipo inicial apresentava muitos fios para interligar sensores e atuadores. Por esta razão, durante a implementação, foram encontrados muitos problemas com relação a descontinuidade dos circuitos, provenientes da trepidação da estrutura. Esse problema com mau contato foi solucionado ao utilizar a placa de fenolite, contendo uma parte do circuito e soldando os demais fios nos respectivos terminais (Figura 9).

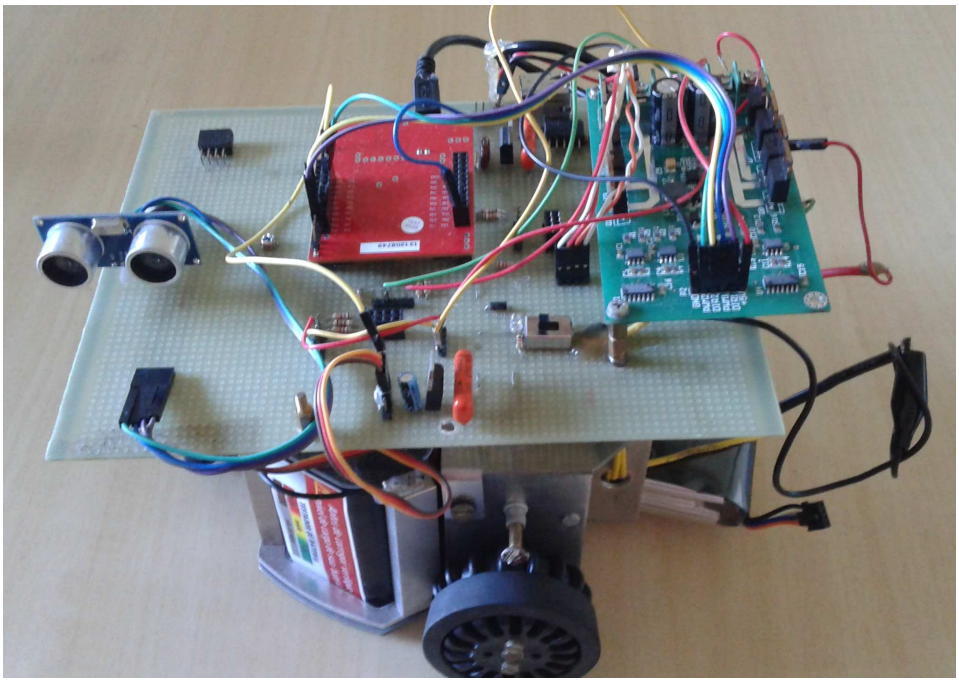


Figura 9: Robô final.

Por apresentarem resultados satisfatórios mediante os testes preliminares, foram utilizados medidores de distância ultra-sônico `HC-SR04`. Como dito anteriormente, o hodômetro, caixa redutora, bateria e motores CC presentes no chassis do robô Curumim foram utilizados devido à qualidade e ótimo desempenho. O driver de potência escolhido como segunda alternativa foi o baseado em MOSFET. Ele é composto por uma placa onde são fornecidas duas saídas para o controle dos motores CC através de PWM, que por questões de eficiência tem menor dissipação de potência em relação ao controle variado, sem PWM. O servo-motor a ser utilizado é o `SG90`, com um torque de $2kg/cm$, tensão de alimentação de $5V$ e o controle feito por PWM. Já o microcontrolador utilizado corresponde a um `Tiva TM C Serie Launchpad TM4C123G`, que foi escolhido devido a atender os requisitos de desenvolvimento deste trabalho.

Nas seções que se seguem, todos esses componentes de *hardware* acima relacionados serão descritos com maiores detalhes.

3.1.2.1 MODELAGEM DO CIRCUITO ELÉTRICO

Como pode ser visto na Figura 10, o circuito do robô é alimentado por duas baterias, uma fornece $12V$ de tensão para alimentação dos motores CC e o regulador de tensão para alimentar o servo motor. Já a bateria de $7,5V$ é conectada a entrada do regulador 7805 que tem como saída $5V$ de tensão para alimentar o microcontrolador, sensores de distância, hodômetro e circuito lógico do driver de potência. Essa configuração do circuito proporciona maior confiabilidade por não sobrecarregar o regulador, com todo o circuito lógico e com o servo-motor. Duas baterias foram utilizadas para dividir o circuito em duas partes, o circuito de potência e o circuito lógico. Esta configuração faz com que as oscilações geradas pelos motores, deixam de afetar o circuito lógico, que é sensível a variações.

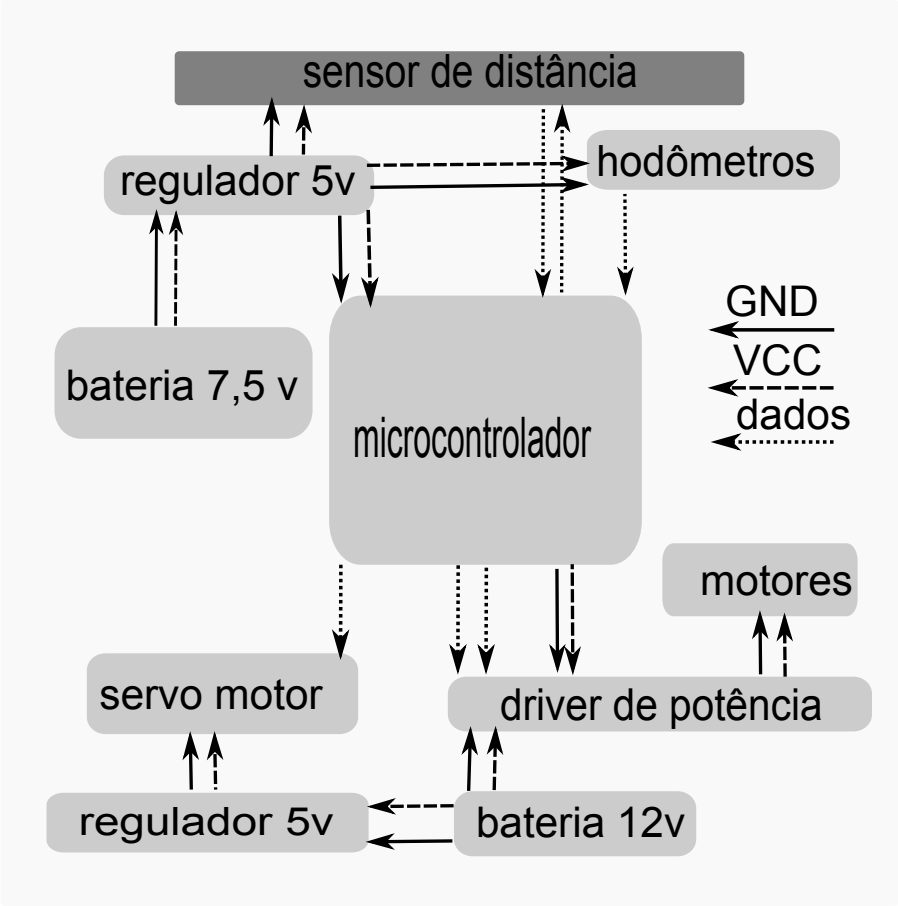


Figura 10: Esquema elétrico do robô.

Embora os sensores de distância e hodômetro tenham alimentação de 5V, o microcontrolador opera com 3.3V. Portanto, um segundo regulador acoplado na saída

do regulador de 5V é necessário para adequar a tensão de operação do microcontrolador, este regulador é encontrado na placa do microcontrolador. O microcontrolador é o centro de aquisição e de controle do robô, que tem como função gerar o PWM que controla o servo-motor e os três motores CC, através do driver de potência e o sinal de direção de giro dos motores CC. Ele também é responsável pela geração dos pulsos do sonar, do tratamento das interrupções do sonar e do hodômetro, do cálculo do tempo do sonar e é responsável por executar o sistema de controle. O microcontrolador juntamente com o driver de potência e as duas baterias possuem a mesma referência negativa.

A comunicação do microcontrolador com os demais periféricos é feita através dos pinos de entrada e saída. O robô possui três hodômetros, cada um contém um único pino de entrada para o microcontrolador, o qual recebe os dados, além de possuir dois pinos de alimentação cada hodômetro. Para que o sensor óptico funcione, é necessário um circuito de condicionamento de sinais para seu bom funcionamento. Outros detalhes estão contidos nos testes realizados no capitulo 4(Resultados e Discussões). Já o sensor de distância necessita de dois pinos de alimentação e dois pinos de dados, um para o envio e outro para o recebimento dos mesmos. Todas estas conexões podem ser vistas na Figura 10.

3.1.2.2 HODÔMETRO

Para a medição de distâncias percorridas pelo robô através da movimentação das rodas utilizou-se a chave óptica interruptiva `C860NP`, que se encontra integrada à roda e à caixa redutora. Esse sensor consiste de um conjunto composto por um fototransistor e um diodo emissor de luz. O princípio de funcionamento do sensor se baseia na disposição do diodo emissor de luz, de tal forma que o transistor sensível à luz satura mediante a incidência de luz no transistor. A engrenagem da roda é composta por 32 dentes, ou seja, existem 32 ranhuras para a passagem de luz em uma volta completa da roda, onde cada dente corresponde a um pulso de tensão na saída em forma de PWM, que varia sua frequência, dependendo da velocidade da roda.

Com base na documentação do sensor foi desenvolvido um circuito que define uma saída com dois estados possíveis. No primeiro estado, o nível alto ou $3,3V$ de tensão é causado pela ausência de obstáculos (dentes da engrenagem) entre o diodo emissor de luz e o foto transistor. Já no segundo estado o nível baixo de $0V$ de tensão é causado pela presença de obstáculos. Os obstáculos que geram a mudança de nível no sensor é uma engrenagem dentada presa ao eixo dos motores.

Esta informação é interpretada pelo microcontrolador da seguinte maneira, conta-se o número de pulsos por intervalo de tempo e obtém-se a velocidade do robô.

A configuração do microcontrolador para fazer a leitura do hodômetro consiste em definir os pinos $PD2$, $PD4$ e $PD6$ como entrada, cada pino de entrada para um hodômetro. Para que esta leitura seja possível é preciso configurar o $WTIMER$ no modo de captura de bordas de subida do sinal, o qual gera uma interrupção. Assim, toda vez que o microcontrolador verifica uma interrupção conta-se um pulso. Para determinar o sentido de deslocamento do motor implementou-se uma lógica que quando o motor gira no sentido horário incrementa a variável do hodômetro e quando o motor gira no sentido anti-horário decrementa a mesma variável.

3.1.2.3 SENSOR DE DISTÂNCIA ULTRA-SÔNICO

A precisão de leitura de um sensor de distância é fundamental para o bom desempenho do sistema de controle. Através de estudos e testes realizados notou-se que algumas leituras erradas eram feitas. Através de estudos e testes realizados, notou-se que algumas leituras erradas eram feitas pelo sensor, para contornar esse problema, aplicou-se o filtro da mediana, que eliminou a maioria das leituras fora do padrão.

O sensor ultra-sônico escolhido para a medição de distância foi o HC-SR04. Esse sensor contém filtros e comparação da resposta sonora. Nas configurações do microcontrolador para gerar os pulsos do sonar, o sensor deve receber uma sequência de pulsos no pino “trigger”, com isso o sensor transmite uma rajada de pulsos sonoros. Após um determinado período de tempo que varia com a distância, o sensor gera uma interrupção no pino “eco”e com o cálculo deste tempo baseado na velocidade do som no ar, pode-se calcular a distância do objeto que está na frente do sensor, que depende da resposta da comparação do sensor e do número de bits Wtimer do microcontrolador.

A configuração do microcontrolador com base nas informações do sensor de distância foi estabelecida da seguinte maneira. O pino $PB3$ como gerador da sequência de pulsos no pino “trigger”e o pino $PC6$ conectando-se ao pino “eco”do sensor. No pino conectado ao “eco”o microcontrolador espera a interrupção de borda de subida, a qual foi configurado inicialmente com Timer de 16 bits no modo captura, o qual funcionava com a seguinte restrição caso o sensor demorasse mais que o tempo da contagem que é de 65535 ($16bits$) com isso reiniciasse a contagem, fazendo com

que o microcontrolador realizasse as próximas leituras de maneira incorreta. Segundo a documentação do sensor, o mesmo sempre que envia uma sequência de pulsos espera o retorno do eco no ambiente se o eco não retornar em $35ms$ o sensor gera uma interrupção de estouro de tempo(timeout). Se utilizando desta informação, foi configurado o WTimer de 64 bits para que seu tempo máximo de contagem ultrapasse este limite.

3.1.2.4 MICROCONTROLADOR

O microcontrolador é o componente chave tanto para o projeto de *hardware* quanto para o projeto do sistema de controle. O componente escolhido para a montagem do robô foi o microcontrolador `Tiva TM C Serie Launchpad TM4C123G`, que está ilustrado na Figura 11. Ele é composto por alguns periféricos embutidos em

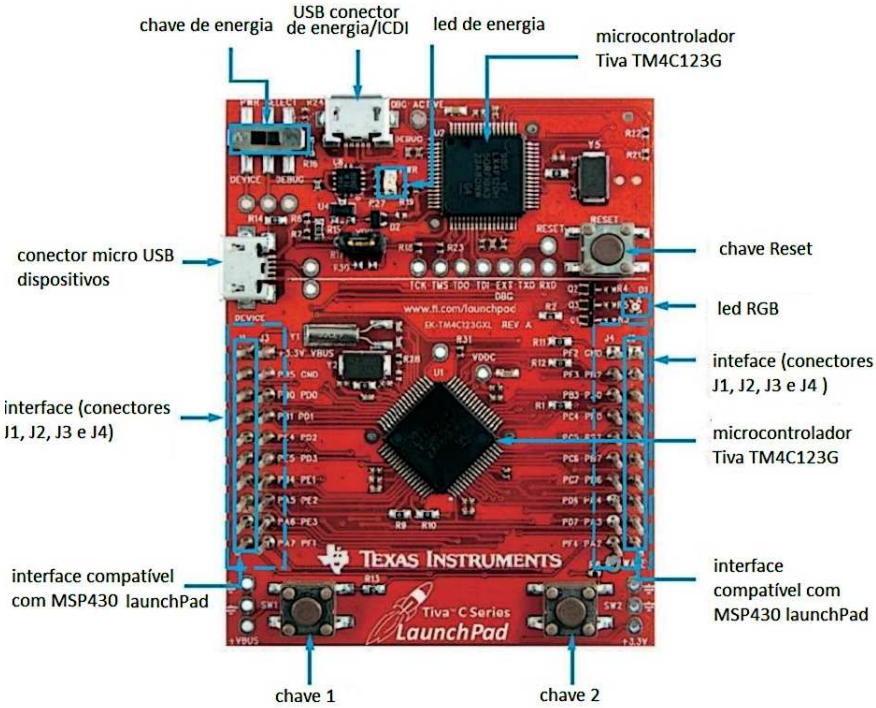


Figura 11: Microprocessador `Tiva C series TM4C123G`**.**

sua pastilha de silício, tais como, o `WTimer` que foi utilizado para cronometrar o tempo que o som leva para ecoar nos objetos após ser transmitido pelo sensor de distância. Ele apresenta quarenta pinos de entrada e saída, onde alguns deles serão empregados para o controle dos drivers de potência e outros para o recebimento de dados dos hodômetros, timers (temporizadores para implementação do controle) e no PWM (*Pulse-Width Modulation*), para controle dos motores CC e servo motor. Seu clock máximo é de $80\ Mhz$, com memória Flash de $256\ kB$ e memória RAM de $32\ kB$, tais

configurações foram suficientes para atender os requisitos de construção do trabalho apresentado.

3.1.2.5 DRIVER DE POTÊNCIA (PONTE H)

A primeira ponte H utilizada é um circuito que controla a rotação dos motores e que está contido em um circuito integrado (CI). Ela permite o controle de velocidade dos motores de corrente contínua e a direção de giro (horário ou anti-horário). A ponte H, ilustrada na Figura 12, recebe este nome devido a sua configuração de circuito, e também pode ser chamada de circuito integrado monolítico L298. O circuito principal é controlado por 4 chaves que são comutadas para o controle de rotação e velocidade. Ele possui um excitador duplo de alta tensão, que permite aceitar níveis padrão da lógica TTL e comandar cargas indutivas, tais como relés, motores C.C. e motores de passo. Este dispositivo é fornecido com duas entradas independentes, o que permite habilitar ou desabilitar o dispositivo dos respectivos sinais de entrada.

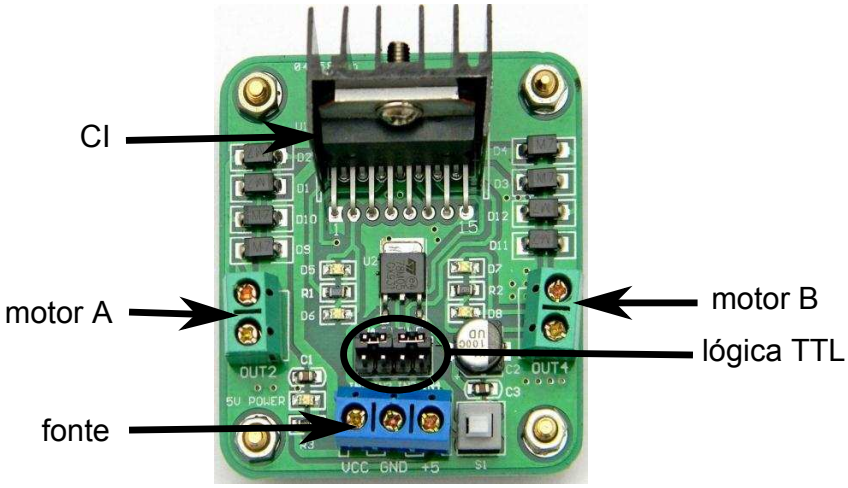


Figura 12: Esquema interno do driver de potência (ponte H) L298.

A segunda ponte H utilizada que pode ser observada na Figura 13,tem as mesmas funções que a primeira ponte H, porém com uma margem maior de segurança nas especificações técnicas. Esta margem maior de segurança é alcançada devido a forma que é feita, a qual foi baseada em transistor MOSFET (acrônimo de *Metal Oxide Semiconductor Field Effect Transistor*) e não com o CI L298. Desta forma os limites de segurança ficam: Tensão de 3 a $39V$, corrente de $10A$, aceitando picos de $30A$. O sistema de controle é composto por apenas um pino de direção, este em nível lógico alto conduz em uma direção e em nível lógico baixo conduz na outra direção. Além deste, possui um pino para o controle de potência do motor, o qual é controlado com PWM. A ponte H é composta por dois conjuntos de controle, podendo controlar

dois motores.

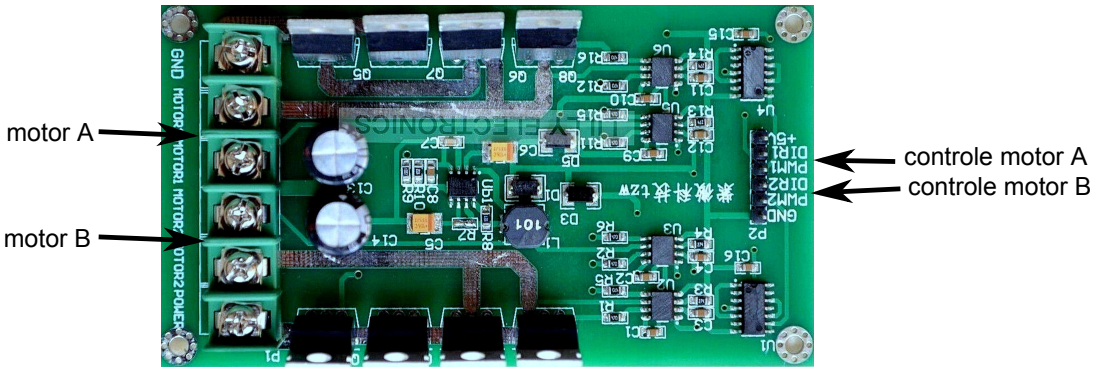


Figura 13: Driver de potência.

No início do projeto de TCC1 definiu-se a primeira ponte H (Figura 12) como sendo a que seria utilizada para montagem do robô. Porém, durante o desenvolvimento do projeto, na etapa de testes dos controladores, notou-se a instabilidade do circuito do robô. Esta instabilidade era gerada quando o controlador acionava os motores consecutivamente em direções opostas. Esta ação de controle demanda maior corrente do circuito, o qual ultrapassava os limites que a ponte H L298 suportava e por consequência causou a queima do microcontrolador, sendo assim, o mesmo teve de ser substituído por outro igual. Maiores detalhes dos problemas causados pela primeira ponte H são descritos na Seção 4.8 (Discussões).

Devido ao problema que causou a queima do microcontrolador foi necessário realizar a troca da primeira ponte H (Figura 12) pela segunda (Figura 13), a qual foi configurada para alimentar os motores, seguindo os parâmetros do fabricante. Portanto, o funcionamento do driver de potência se baseou na seguinte configuração. Cada ramo da ponte é composto por dois pinos, um que indica a direção e o outro controla a potência dos motores, os quais são acionados por PWM. Como o robô possui três motores, foi necessário manter os dois motores traseiros no mesmo ramo da ponte H e o dianteiro em outro ramo.

3.2 *SOFTWARE*

Nesta seção são detalhadas as implementações de *software* necessárias nos diferentes níveis de construção do robô. Na Sessão 3.2.1 é exposto o *software* de interface de configuração do microcontrolador. Esta interface de configuração é uma plataforma de desenvolvimento que trata e transfere as implementações realizadas no computador para o microcontrolador. As implementações realizadas no computador são referentes ao sistema de baixo nível (Sessão 3.2.2) e alto nível do robô

(Sessão 3.2.3).

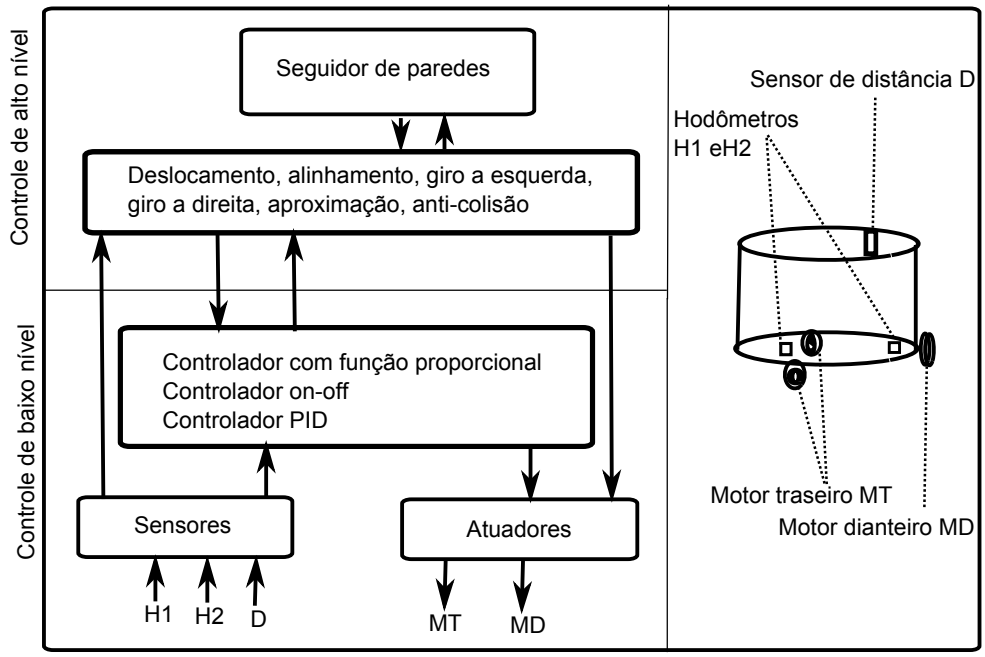


Figura 14: Divisão do sistema de *software*

A Figura 14 representa como o sistema de *software* que foi desenvolvido e pode ser dividido em controle de alto nível e controle de baixo nível. O controle de baixo nível é responsável pela captura dos dados dos sensores e também é responsável pelos controladores dos motores e do driver de potência que são os atuadores. O controle de alto nível implementa as funções básicas de deslocamento, alinhamento, giro à esquerda, giro à direta, anti-colisão e aproximação as quais se utilizam dos dados tratados ou do controle de baixo nível. O alto nível também é o responsável pelo comportamento e decisões que o robô toma para seguir paredes do ambiente.

3.2.1 INTERFACE DE CONFIGURAÇÃO

Após o término da montagem da estrutura física do robô, o próximo passo realizado foi a configuração dos componentes. Esta é uma tarefa bastante complexa o que demanda quantia significativa de tempo. De início, tentou-se utilizar neste projeto a ferramenta de desenvolvimento © Energia (ENERGIA, 2014) a qual foi substituída pela ferramenta `Code Composer Studio(CCS)`.

Quanto à ferramenta CCS, foi utilizada apenas a biblioteca padrão do microcontrolador `TivaWare`, a qual conta com poucas funções prontas. Devido a esta limitação, a complexidade do projeto aumentou significativamente, demandando um tempo maior para estudo e configuração dos periféricos.

3.2.2 CONTROLE DE BAIXO NÍVEL

Para que o robô possa realizar com segurança e precisão sua tarefa de navegação, uma série de problemas envolvidos na movimentação do robô necessitam ser tratados, tais como deslizamento nas rodas e erros de posicionamento, por exemplo.

3.2.2.1 ATUADORES

Durante a execução de sua tarefa de navegação, o robô poderá se movimentar de diferentes maneiras. A Figura 15 apresenta uma modelagem dos possíveis movimentos do robô mediante as características de suas rodas, o que proporcionam movimentos controlados e não controlados.

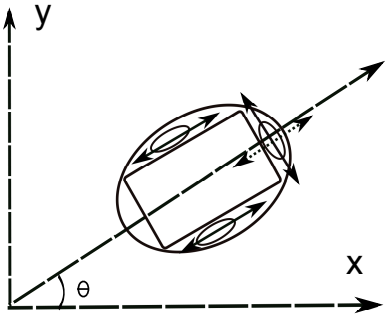


Figura 15: Mapeamento dos possíveis movimentos executados pelo robô

As setas contínuas indicam os movimentos controlados, pois é a direção do torque dos motores, os quais podem ser lidos a partir dos hodômetros. Já a seta pontilhada indica o movimento não controlado, devido ao formato bidirecional das rodas do robô. Entretanto, essas características das rodas podem facilitar a movimentação no plano, por exemplo, para se aplicar o ângulo θ, basta acionar o motor dianteiro. Já o movimento linear na direção da reta com ângulo θ se dá acionando os dois motores laterais com mesma potência.

Para o movimento linear do robô a velocidade dos motores é controlada para evitar colisões, evitar deslizamento, evitar leituras erradas do sonar e evitar que o mesmo se localize fora da referência desejada. Para tal, foi aplicado controladores PID (proporcional, integral e derivativo), on-off e controle proporcional, os quais serão apresentados nas sessões seguintes.

3.2.2.2 SENSORES

O sensor de distância faz a medição da distância entre o robô e obstáculos, para evitar colisões e ao mesmo tempo, gera informações para os sistemas de controle de baixo nível e alto nível. A cada leitura do sensor faz-se uma transmissão pela UART (comunicação serial) a fim de plotar o mapa da trajetória executada pelo robô no computador. A Figura16 ilustra o campo de varredura do sensor.

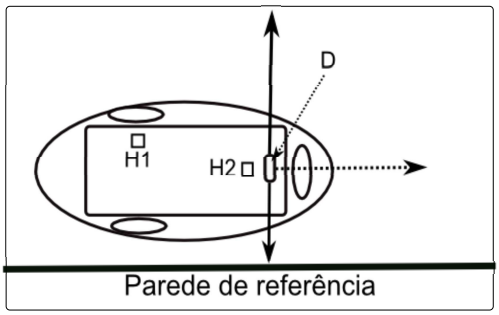


Figura 16: Varredura do sensor de distância ultra-sônico.

Através do giro do servo motor posicionado à frente do robô, o sensor de distância obterá todas as leituras que o robô pode fazer. O sensor girará em 180^o, varrendo toda a parte frontal do robô. Sendo assim, poderão ser feitas leituras na frente ou nas laterais, devido as características do sensor de distância. À medida que o robô se movimenta para frente, este sensor ficará periodicamente executando leituras na lateral de referência. Este sensor também é responsável pelas leituras feitas a cada parada do robô que verifica a distância frontal, e analisa se pode ou não executar um próximo deslocamento ou se deve executar um giro.

As leituras do sensor de distância são os dados de entrada para o controle da roda dianteira, onde foram aplicadas técnicas de controle on-off e controle proporcional. Desta forma o robô pode se locomover mantendo o alinhamento com as paredes e evitando possíveis colisões.

A percepção do ambiente conta também com os dados do sensor de hodometria que capta os movimentos das rodas. Esta percepção é importante, pois é a entrada para o controlador controlador PID ou on-off os quais controlam o movimento retilíneo do robô. Esta percepção indica a direção e velocidade de deslocamento para o controle.

3.2.2.3 CONTROLE PID

No início do desenvolvimento do projeto foram implementadas as funções de baixo nível, entre elas o deslocamento que contou com um controlador PID. Devido à complexidade em se obter a função de transferência, optou-se por ajustar o controlador de forma empírica, pois, é conhecido o efeito das ações básica de controle, além disso o ajuste inicial (provável mal ajuste), não traz prejuízos ao sistema como um todo. Sendo assim, foi desenvolvido um controlador com base na teoria de controle, o qual cada termo do controlador tem sua função específica na resposta final, como descrito abaixo.

No controle PID, o termo P é a componente proporcional, a mesma tem grande influência no controle quando o erro é grande. O erro é o cálculo da diferença entre a referência e a posição atual. No caso do robô, a entrada do controlador são pulsos do hodômetro das rodas traseiras, e sua referência é o número de posições que o robô se deslocou a partir do início do deslocamento até a posição atual. Não é difícil notar que, quanto mais próximo da referência, este termo terá menor influência na saída. No início do deslocamento este termo terá valor máximo.

O termo derivativo D é a componente que faz com que a saída diminua se a variável do processo está aumentando rapidamente. Assim, se o motor acelera rapidamente, a ação derivativa ajuda a evitar o “sobre sinal” e melhora o tempo de acomodação do controlador. A derivada de resposta é proporcional à taxa de variação da variável de processo. Aumentar o parâmetro do ganho derivativo Kd faz com que o sistema de controle reaja mais fortemente a mudanças no parâmetro de erro, aumentando a velocidade da resposta global de controle do sistema. Na prática, a maioria dos sistemas de controle utilizam o tempo derivativo Kd muito pequeno, pois a derivada de resposta é muito sensível ao ruído do sinal da variável de processo, com base em (GOMES, 2008). No caso do controle neste trabalho, o sensor de referência para o controle é o hodômetro, foi utilizado um Kd pequeno devido à resposta do controle. A ação do derivativo é facilmente entendida quando num instante em que o erro é momentaneamente nulo, mas sua taxa de variação não. Analisando esta teoria em relação a variável do robô (hodômetro) o erro nulo indica que o robô encontra-se na referência de posição, porém sua velocidade é diferente de zero.

Já o termo integral I tem como função eliminar o erro em regime permanente, ou seja, depois que o motor chega próximo à referência, o integrador continua acumulando o erro. Se o robô esta próximo à referência mas existe um erro, o in-

tegrador aumenta lentamente até que o erro se torne zero. Teoricamente, não seria necessário a implementação do integrador no controle de posição para motores CC, devido à característica de ter um integrador intrínseco nos sistemas de posicionamento. Devido à zona morta de controle inerente aos motores CC foi necessário a ação integral. No caso dos sistemas reais com grande inércia, como é o caso do robô, existe uma tensão mínima de arrancada, que na prática ficou aproximadamente em torno de 20% da largura do PWM. Uma forma de visualizar o uso do termo integral neste projeto, foi observar o deslocamento do robô utilizando um controlador PD, onde o mesmo parava em posição diferente da desejada, devido ao à zona morta. O termo integrador é usado apenas para eliminar o erro inserido pela zona morta.

No intuito de observar a aceleração e desaceleração do motor, o controlador desenvolvido foi ajustado fora do ambiente em que o robô será inserido, com maior dimensão, para se observar as características do PID atuando. Desta maneira, o controlador pode ser sintonizado com diferentes ganhos até que fossem obtidos bons resultados, ou seja, aceleração e desaceleração adequadas. Porém, este controle não gerou as características de um PID na aceleração do robô, porque na etapa de integração do controle lateral e do deslocamento, notou-se que a velocidade do robô deve ser limitada em 30% do PWM. Isto se deve a falha de leitura do sensor de distância que sofre interferências com o robô em movimento. A leitura do sensor de distância é usado para entrada do controle lateral que deve ser executado durante todo o deslocamento. Desta forma, os limites máximo e mínimo da velocidade são muito próximos devido a este problema, isso faz com que o robô possa ser facilmente controlado com a técnica on-off, a qual é descrito a seguir.

3.2.2.4 CONTROLE ON-OFF

O controle on-off foi obtido a partir da simplificação do controle PID aplicado anteriormente. Sendo assim, foram feitos alguns testes para delimitar seus limites de atuação. Como pode ser observado na Figura 17, os limites do controle on-off foram definidos a 25% do PWM, como potência máxima, e o erro aceitável de $2cm$.

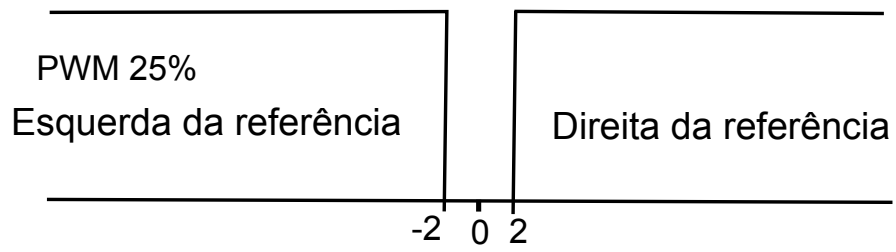


Figura 17: Controle On-off.

O já nas rodas dianteiras o controle on-off foi o primeiro a ser aplicado, devido a sua facilidade e simplicidade. Apesar do robô não desenvolver uma trajetória retilínea, ele se comporta de maneira satisfatória, ou seja, com poucas oscilações em seus movimentos. A velocidade foi mantida em 25% do PWM devido a restrições impostas pelo sensor de distância que é entrada para este controle.

O controle on-off opera basicamente a partir de uma verificação. Caso o sinal de entrada seja maior que a referência mais um limiar de dois centímetro, então o motor é acionado no sentido horário. Caso seja menor que a referência menos um limiar de dois centímetro, então o motor é acionado no sentido anti-horário. Em ambos os casos os motores são acionados com 25% de PWM, caso não ocorram as opções anteriores, então os motores são desligados, conforme é apresentado na Figura17.

Portanto, o controle foi aplicado nas rodas traseiras e dianteiras, ambos com a mesma lógica, porém adaptados de forma que o erro seja de no máximo dois centímetros. Para o controle traseiro o robô arranca e vai até a posição de referência, tendo como entrada os dados do hodômetro das rodas traseiras. Já o controle on-off frontal se posiciona com base nos dados do sensor de distância, e sua referência é uma distância da parede. Sendo assim o controle dianteiro alterna a direção da roda dianteira, forçando o robô a ficar próximo à referência, este controle segue a máquina de estados da Figura 18 .

3.2.2.5 CONTROLE PROPORCIONAL

O controle proporcional desenvolvido teve como objetivo diminuir as oscilações apresentadas pelo controle on-off, o qual foi aplicado à roda frontal do robô. Este controle proporcional variável só pode ser aplicado após alguns testes e uma análise do comportamento do robô. Esta análise permitiu um ajuste de maneira adequada dos limites de velocidade e tempo de resposta do sistema, respeitando as limitações do robô. O controlador tem diferentes valores de proporcional que são distribuídos em relação à distância da referência da parede (entrada) e a potência do motor (saída). Este controle entra em ação sempre que o robô executa um alinhamento ou se desloca no ambiente seguindo paredes.

O controlador proporcional projetado é descrito na Figura 19, o qual tem seus valores definidos através da função que é definida por segmentos de retas. Esta função é definida no lado positivo da referência e no lado negativo com mesmo valores de ganho proporcional. Os segmentos de retas interligadas que definem a potência

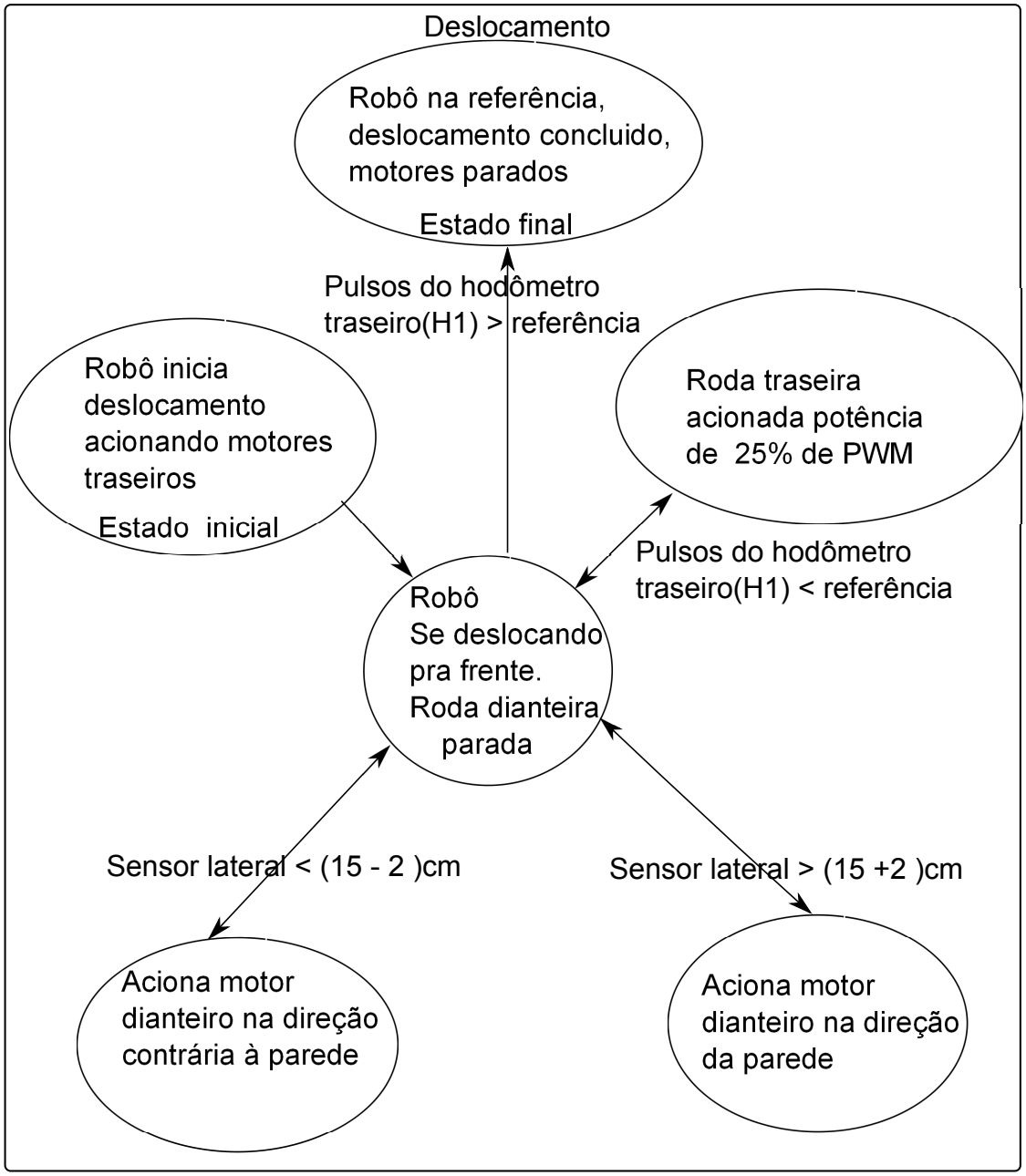


Figura 18: Máquina de estados on off.

em relação à distancia de referência tem como potência máxima 40% do PWM.

A função proporcional é definida da seguinte maneira: o controle tem sua potência mínima quando está na referência, ou seja, com o sinal do PWM desabilitado. A primeira rampa inclinada que parte da origem vai até 3 centímetros onde sua potência é definida como máxima de 25% da largura do PWM. A segunda rampa inclinada parte deste ponto e vai até 7 centímetros, onde sua potência é de 35% da largura de PWM. A terceira reta parte do ponto anterior e vai até 14 centímetros e sua potência vai até 40% da largura do PWM. Após este ponto, a função é definida constante para todos os valores com potência de 40% de largura do PWM. A função

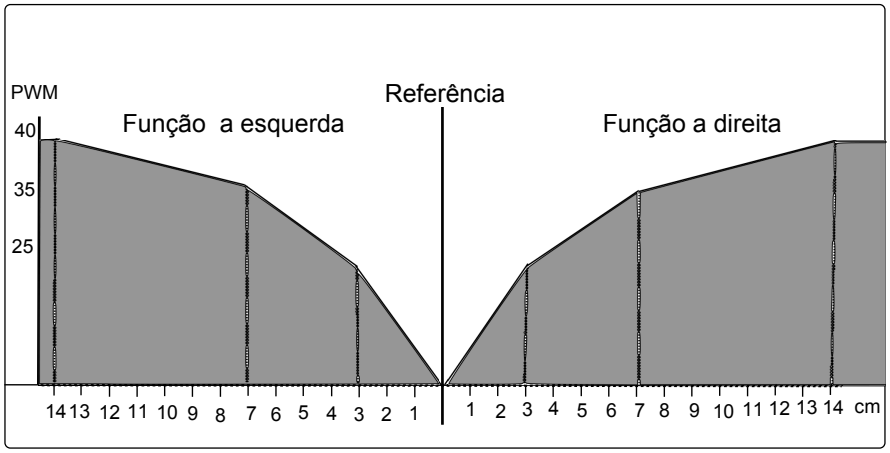


Figura 19: Representação do controle especifico. Fonte: autoria própria.

proporcional pode ser observado na Figura 19.

Conforme foi descrito acima o controle dianteiro poderá ser realizado tanto pela técnica on-off quanto pelo controlador proporcional. O robô tem como objetivo se manter a uma distância de referência da parede em ambas as técnicas. A roda dianteira executa movimentos para a esquerda e para a direita durante o deslocamento, buscando manter-se na referência. A Figura 20 descreve o movimento do robô seguindo paredes, onde o traço pontilhado representa a trajetória resultante do controle.

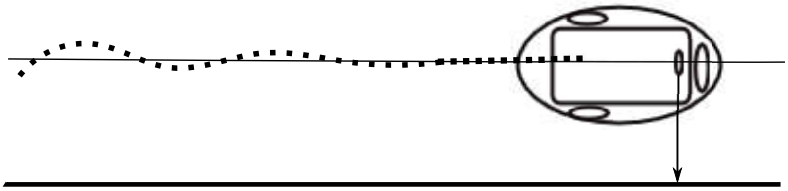


Figura 20: Movimentos gerado pela ação de controle. Fonte: autoria própria.

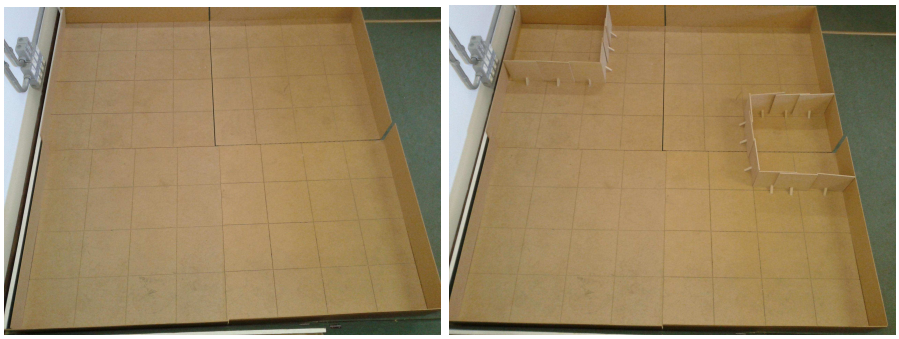
3.2.3 CONTROLE DE ALTO NÍVEL

O controle de alto nível, tem como objetivo implementar as funções básicas utilizando-se dos controle de baixo nível e dos dados dos sensores. Com base nas funções desenvolvidas foi construído um algoritmo que tem como objetivo seguir paredes do ambiente e com isso plotar o gráfico do mesmo no computador.

3.2.3.1 AMBIENTE

O robô foi testado em um ambiente projetado segundo o tamanho de sua estrutura física e de acordo com o alcance máximo para as medidas de seu sensor de distância. O ambiente possui aproximadamente $4\ m^2$ e está dividido em células de $25 \times 25\ cm$, como pode ser visto nas Figuras 21(a) que representa o ambiente sem

obstáculo e 21(b) que é o ambiente com obstáculos respectivamente.



(a) (b)

Figura 21: (a) Sem obstáculos. (b)Com obstáculos.

O ambiente, projetado para o robô, se encontra nas dependências do Laboratório de Informática (sala V009), do Departamento Acadêmico de Informática da UTFPR-PB. Ele foi construído especialmente para experimentos com o robô Curumim, e pode ser facilmente montado e desmontado.

3.2.3.2 ALGORITMO DO SEGUIDOR DE PAREDES

O algoritmo de seguir paredes implementado é exposto na sequência. Ele tem como objetivo se utilizar das funções do controle de baixo nível para seguir as paredes do ambiente e enviar os dados para a construção do mapeamento da trajetória desenvolvida pelo robô, no ambiente. Este algoritmo se baseia em algumas informações fornecidas a priori, tais como, localização inicial e o formato retangular do ambiente. À medida que o robô segue as paredes, o mesmo envia as dimensões do ambiente através da comunicação serial, para uma aplicação implementada no computador com intuito de gerar o mapeamento da trajetória executada pelo robô.

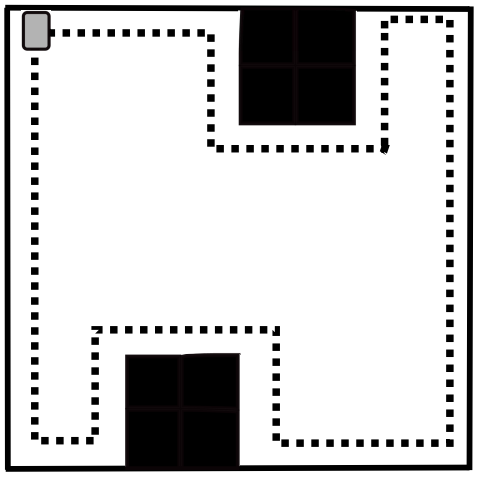


Figura 22: Esquema da estratégia de exploração do ambiente.

Como pode ser visualizado na Figura 22, o ambiente foi simplificado, de forma a estar ao alcance do sensor de distância que é de 5 metros. O robô percebe o ambiente fazendo leituras na sua lateral de referência e a sua frente. A trajetória pontilhada indica o caminho esperado a ser executado pelo robô, através da execução do algoritmo.

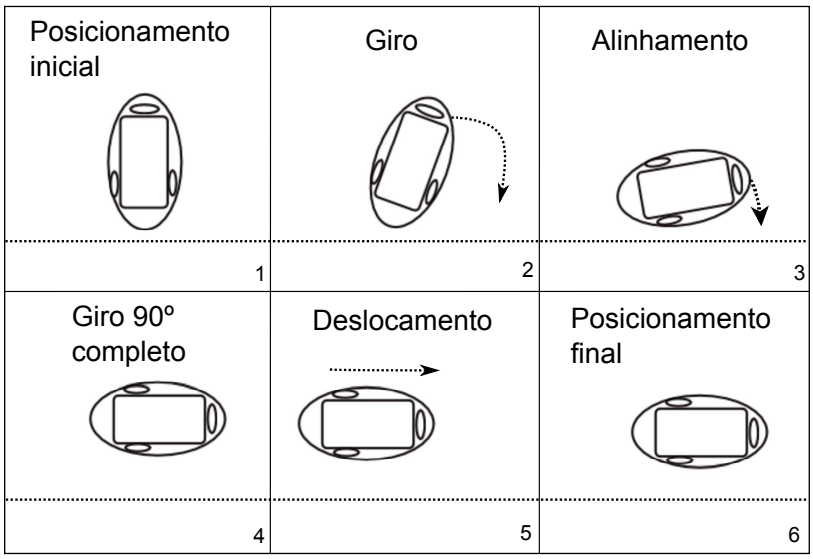


Figura 23: Exemplo de como o robô irá se locomover até a próxima posição, mediante a necessidade de um giro à direita.

O algoritmo proposto executa uma série de passos elementares para cumprir seu objetivo que é o de fazer o robô girar, mediante um canto do objeto, e em seguida se deslocar. Este seria um caso onde alguns erros e perda de referência poderiam ocorrer. Entretanto, com as funções executadas na sequência correta, esses erros seriam reduzidos, como pode ser analisado no esquemático da Figura 23, a qual robô se encontra em uma posição inicial, executa um giro de 90°, e em seguida executa a função alinhamento, para corrigir possíveis erros de posição. Na sequência, ele então executa o deslocamento, o qual mantém uma referência com a parede (linha pontilhada).

Esse algoritmo seguidor de paredes é o ponto de partida para que se possa detectar todos os problemas envolvidos com a locomoção do robô proposto. E então, a partir da resolução desses problemas, aperfeiçoar o controle de alto nível do robô, o que permitirá que ele realize tarefas de navegação das mais simples as mais elaboradas.

Listagem 1: Algoritmo do seguidor de paredes.

```
1 Algoritmo para seguir paredes internas.
2 Void main (){
3 inicializa periférico
```

```
4
5  While {
6      if(distancia frontal maior que 2 vezes o deslocamento){
7          Deslocamento();
8      }else if (verifica se distancia frontal menor 20cm){
9          Giro(90º);
10     }else if(distância da parede frontal menor que
11      deslocamento + limiar e maior que 20cm ){
12         Aproximação();
13     }
14 }
15 Void Deslocamento(){
16     Sensor_anti_colisão();
17     controlador_roda_traseira(distancia);
18     controlador_lateral(distância da parede);
19 }
20 Void Giro(){
21     Verifica_distância_parede();
22     Liga_motor_frente(25% de PWM, (direita ou esquerda));
23 if(pulsos do hodômetro >40){
24     Liga_motor_frente(0% de PWM, (direita ou esquerda));
25 }
26     Alinhamento(20cm da parede);
27 }
28
29 Void aproxima(){
30     Sensor_anti_colisão();
31     Liga_motor_traseiro(25% de PWM, frente);
32 If(distancia frontal menor que 20){
33     Liga_motor_traseiro(0% de PWM, frente);
34 }
```

O algoritmo seguidor de paredes implementado para o robô está descrito na Listagem 1.

Na sequência, são descritas as funções que compõem o algoritmo seguidor de paredes. São elas `Giro`, `Alinhamento`, `Deslocamento`, `Aproximação` e `Sensor_anti_colisão`, as quais se utilizam do controle de baixo nível e de dados dos sensores. As mesmas estão descritas na sequência.

A função `Giro` terá duas sub-funções, giro 90^o à esquerda e giro 90^o à direita. Essas sub-funções poderão ser realizadas mediante o acionamento do motor dianteiro por n pulsos na direção do giro. Na execução do algoritmo proposto, esta

função tem preferência de giro à esquerda. Esta função se comporta bem com baixa potência, mas se for aumentar a potência de giro, teria que contar com o auxílio de um sensor de giro, como uma bússola ou um giroscópio. O giro teve como entrada os pulsos do hodômetro da roda dianteira e para executar o giro, a mesma conta n igual a 40 pulsos, desta forma aproxima-se de 90^o e com a função `Alinhamento` este giro é completado.

A função `Alinhamento` é utilizada normalmente depois de um deslocamento, a mesma utiliza-se do controlador da roda dianteira, o qual é acionado com o robô parado e desta forma o robô alinha-se com a parede do ambiente.

A função `Deslocamento` toma como entrada a distância deslocada anteriormente. Esta função permite que o robô se locomova em linha reta e para frente, seguindo paredes.. O movimento para frente é obtido pelo acionamento dos dois motores laterais com mesma potência, este deslocamento conta com o controlador PID ou on-off para evitar deslizamento das rodas. Na roda dianteira o controlador on-off ou controlador proporcional são executados durante o deslocamento para evitar colisões com as paredes, os controladores foram descritos na Seção 3.2.2.

A função `Aproximação` tem como objetivo fazer com que o robô se aproxime da parede à sua frente. Esta função será acionada quando o robô detectar uma parede a sua frente, sem que o mesmo tenha espaço para executar um deslocamento completo e que ao mesmo tempo ainda esteja longe da parede para executar o giro. A aproximação não tem referência lateral, porém tem referência frontal, onde são feitas leituras do sensor de distância periodicamente. Os motores traseiros são acionados com potência constante, até que a distância frontal seja menor que uma distancia de 20 cm.

A função `Sensor_anti_colisão` tem como objetivo fazer a leitura da distância dos objetos na frente do robô. A função deverá manter uma distância mínima de um obstáculo. Caso a função detecte um objeto com distância menor que 20 cm na frente do robô, é executado um giro. Já se a distância frontal for maior que 20 cm e menor que um deslocamento (25 cm) mais 20 cm (distância de referência com a parede) , então o robô executa uma aproximação. No caso em que o robô tem espaço livre a sua frente então o mesmo pode se deslocar. Essa função deve ser executada com o robô parado, para realizar leituras frontais do sensor de distância.Também pode ser realizada a leitura em movimento do sensor na lateral do robô, o qual é entrada para os controladores on-off ou proporcional, os quais impedem a colisão do robô nas laterais. Esta função encontrou problemas com relação as leituras frontais do robô, devido

as falhas de leituras inclinadas que o sensor de distância apresentou. Com isso foram feitas apenas leituras perpendiculares a paredes frontal, e não uma leitura a cada 20º na frente do robô.

Giro em ambiente sem obstáculos

Apesar da pouca percepção do meio, o robô pode realizar sua tarefa de navegação no ambiente. Sua tarefa mais básica foi contornar o ambiente sem obstáculos. Para executar o contorno seguindo as paredes internas do ambiente, o robô basicamente executa todas as funções básicas, as quais são,`Giro`, `Alinhamento`, `Deslocamento`, `Aproximação` e `Sensor_anti_colisão`.

O algoritmo para executar o contorno do ambiente, é descrito através dos seguintes passos:

1. Verifica se pode deslocar-se à frente, para isso mede a distância frontal, que deve ser maior que um deslocamento (25 pulsos do hodômetro) mais um limiar. Se a distância for maior que a referência, então vai para o passo 2, caso não possa deslocar-se então executa o passo 3.

2. Executa um deslocamento à frente do robô.Voltar para o passo 1.

3. Aproxima-se da parede lentamente fazendo leituras frontais até ficar a 20cm da mesma, após executar esta aproximação vai para o passo 4.

4. Executa um giro à esquerda de 90º e volta para o passo 1. Os passos acima descritos podem ser observados na Figura 24.

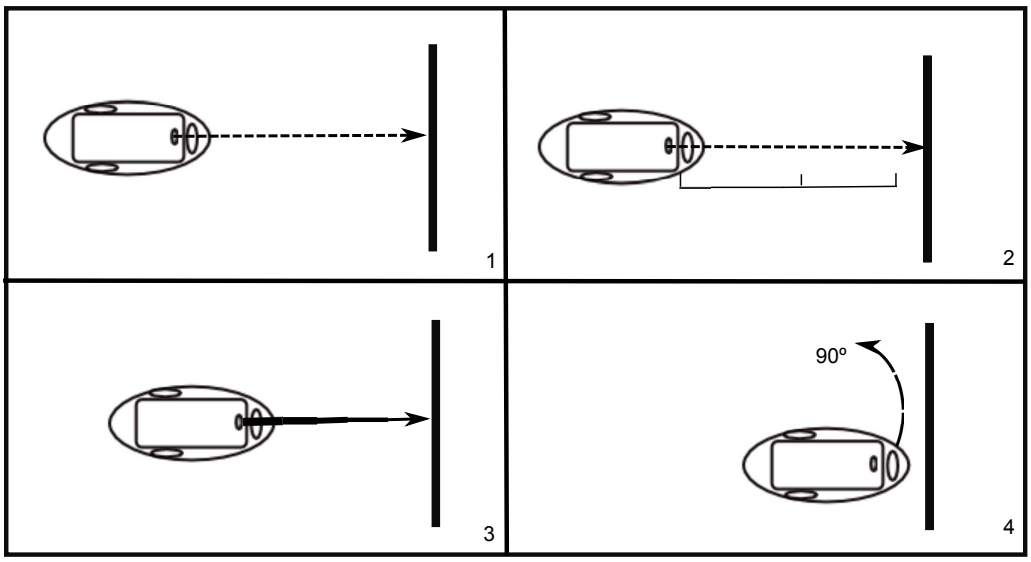


Figura 24: Estratégia de aproximação de obstáculos e giro.

Giro em ambiente com obstáculos

Conforme descrito na seção anterior, o robô pode deslocar-se da maneira correta em ambiente sem obstáculos. Já no caso da presença de obstáculos conjugados com as paredes, o robô não conseguiu executar o deslocamento de contornar o ambiente utilizando-se das funções básicas. A causa dessa falha se deve ao problema de perda de referência, que ocorre quando o robô está em um ponto em que a parede de referência tem uma quina.Esse problema é descrito na Figura 25. Quando o robô faz a primeira leitura do sensor de distância após ter passado da quina, ou seja, a leitura da distância vai apresentar uma mudança abrupta da leitura anterior o que implica em uma saída de potência de 40% de PWM que é a saída máxima limitada e com esta potência o robô colide com a quina. Desta forma, desenvolveu-se uma função para buscar eliminar esta falha.

Devido a restrição de sensores, a função desenvolvida não obteve sucesso em tratar os problemas encontrados neste caso de navegação. Com apenas os sensores de hodometria e de distância não se pode ter a percepção suficiente para que o robô siga as paredes em uma quina. Nesta configuração de quina, por exemplo, ocorre falha de leitura (leitura maior que duas vezes a referência) ou seja, devido ao posicionamento (distância x, y em relação à quina) do robô no momento da leitura abrupta e da posição de giro (ângulo do robô em relação à direção anterior). A Figura 25 (a) e (b) mostram os problemas relacionados a seguir o contorno de um objeto.

Além disso, ao se considerar os dados dos sensores de hodometria, os mesmos não podem garantir de que o robô se encontra em paralelo com a parede. A única informação advinda dos hodômetros é o deslocamento de x posições do hodômetro traseiro e y no dianteiro.

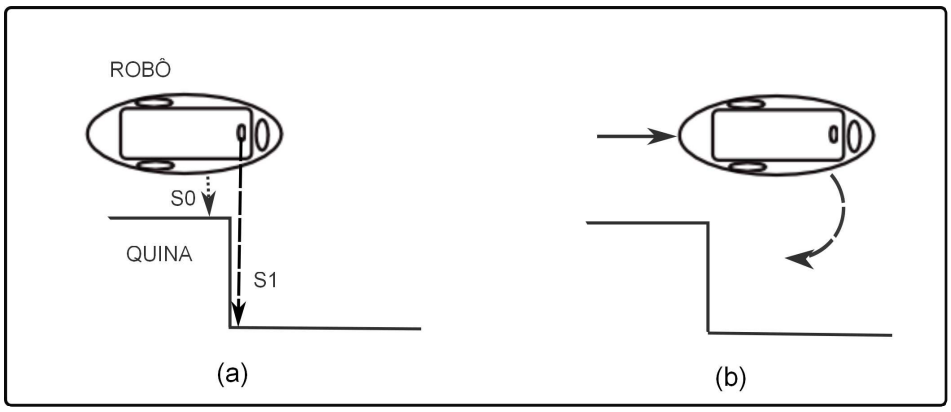


Figura 25: Estratégia para contornar as quinas.

A fim de resolver esses problemas, foi implementado um algoritmo que tratou primeiro a falha gerada pelo controle dianteiro. Nesse caso, ao se deparar com uma quina, o robô batia na parede, devido ao controle gerar uma saída de 40% de

PWM. A solução dada a esse problema foi aplicar um deslocamento de 30 pulsos na roda traseira e manter a roda dianteira parada. Porém esta solução não é eficiente em todos os casos, por exemplo, sempre que o robô faz leituras duas vezes maior que sua referência lateral, o mesmo irá executar um deslocamento não esperado para frente. Após o deslocamento o robô executa um giro de 90º, como pode ser visto na Figura 25(b). Porém nem sempre o mesmo se localiza na posição adequada, que seria em paralelo com a parede , e uma distância de aproximadamente o tamanho de sua referência lateral.

3.2.3.3 CONSTRUÇÃO DO MAPEAMENTO DA TRAJETÓRIA NO COMPUTADOR

No contexto do desenvolvimento desse trabalho, implementar uma forma de transcrever o comportamento de navegação do robô para um gráfico contento sua trajetória no plano tornou-se uma tarefa complexa devido à percepção restrita imposta pelos sensores de distância e hodômetros.

Entretanto, com o objetivo de plotar a trajetória do robô no ambiente, foram enviados os dados dos sensores e movimentos realizados, pela comunicação serial do robô. Para plotar a trajetória do robô foi desenvolvido um programa em *(*LabView) para extração de dados, que tem como objetivo receber os dados enviados pelo robô através da comunicação serial. Os dados recebidos são armazenados em um arquivo, com isso, tem-se todos os movimentos que o robô desenvolveu. A partir destes dados pode-se extrair as coordenadas das posições do robô durante seu deslocamento. Os dados recebidos pela comunicação serial são no formato "$s2\ 17\ E\ s2\ 17\ T\ 52\ D$". O "$s2$"indica que o próximo número é uma leitura lateral, o "E"indica que o robô executou um giro a esquerda, o "D"indica que o robô executou um giro a direita e o "T"indica que a próxima leitura de distância será realizada na frente do robô.

O segundo programa extrai as coordenadas de pontos a partir dos dados fornecidos pelo programa anterior. Este programa produz coordenadas aproximadas referentes a trajetória executada pelo robô. Para a construção deste programa, sub-programas (sub VIS) foram responsáveis por tratar determinados pontos dependendo do tipo de deslocamento. Isto se deve ao fato de que os deslocamentos contêm variações no tempo e na distância percorrida em linha reta. Como saída, do programa exibe "n"pontos em coordenadas cartesianas que são distribuídos uniformemente na distância percorrida pelo robô.

A Figura 26 representa o processo para a construção do mapeamento da

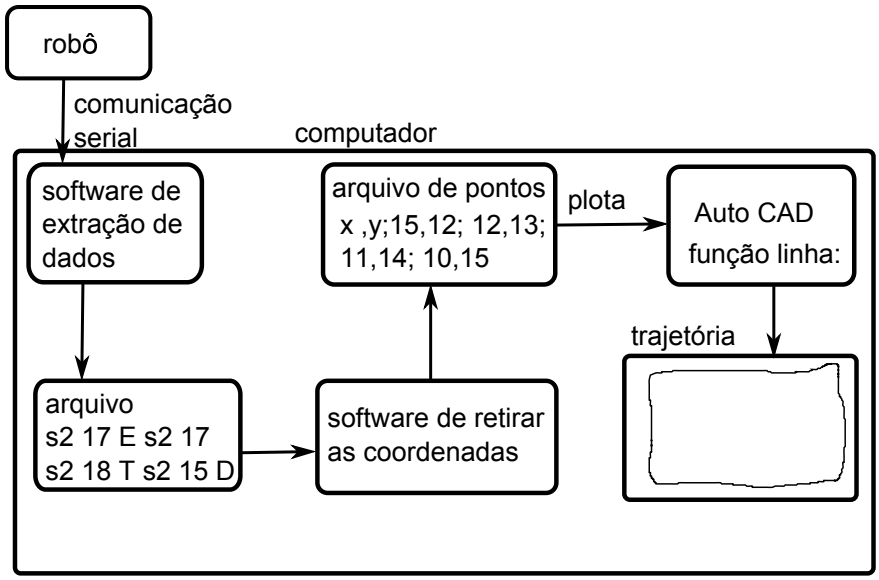


Figura 26: Construção da trajetória.

trajetória executada pelo robô. Portanto, a primeira etapa deste processo consiste em obter os dados dos sensores e atuadores via comunicação serial. A partir dos dados obtidos, um segundo programa gera dados de pontos cartesianos aproximados. Estes pontos são então inseridos na ferramenta AutoCAD que produz como saída a plotagem de uma aproximação da trajetória executada pelo robô em seu ambiente.

O mecanismo de mapeamento de trajetória apresentado anteriormente teve de lidar com o problema da leitura incorreta advinda do sensor de distância em determinado caso. Ou seja, quando o robô transmite a distância com a parede ao executar um giro, o programa executa uma interpolação, interligando todos os pontos. Sendo assim, o próximo ponto vai depender do ponto anterior e esta dependência resulta em um efeito cascata. Ou seja, se um ponto tiver uma variação de leitura em x, todos os pontos posteriores apresentarão erro acumulado às suas posições verdadeiras.

4 RESULTADOS E DISCUSSÕES

Neste capítulo são apresentados os resultados obtidos a partir de todo o processo de construção do robô apresentado no Capítulo 3. Os resultados se referem à montagem e configuração do hardware, as implementações das funções de controle de baixo e alto nível e ao procedimento para reproduzir as trajetórias executadas pelo robô.

4.1 HODÔMETRO

Para desenvolver o circuito condicionador de sinais foi utilizada uma resistência de 240Ω em série com o diodo emissor de luz do hodômetro. Já o fototransistor para funcionar como chave deve operar nas regiões de corte e saturação, quando o transistor entra em saturação é necessário um resistor em série para evitar curto circuito no sistema, sendo que o coletor recebe $3.3V$ e o emissor é conectado na referência $0V$ [1]. Esta configuração de circuito é chamada de amplificador baseado na corrente de base (fonte de luz). Desta forma, optou-se por um resistor com resistência elevada para aumentar a eficiência do sistema (diminuir a potencia consumida, foram testados $10K$, porém a resposta não foi boa. Sendo assim, foram testados outros até chegar a $4.7K$ que foi satisfatória. O microcontrolador fará a leitura do sinal de tensão sobre este resistor. Este circuito foi simulado no Multisim como pode ser visto na Figura 27.

Portanto, a resposta na saída do osciloscópio pode ser observada na Figura 28, a partir do circuito da Figura 27. Essa resposta foi adequada, pois tem-se no canal 2 a tensão de alimentação do circuito de condicionamento do hodômetro e a tensão de alimentação vindo do microcontrolador (cerca de $3,3V$), enquanto no canal 1 tem-se a saída gerada pelo chaveamento do fototransistor com os motores ligados com a tensão de $12V$ de alimentação.

[1] **Fonte: Datasheet, sítio:** `http://ebookbrowsee.net/c860np-pdf-d200118438` **acessado em 05/12/2013**

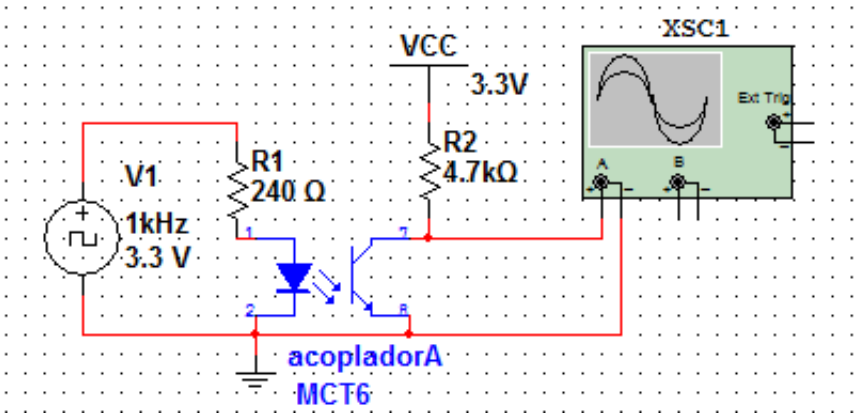


Figura 27: Circuito simulado no Multisim.

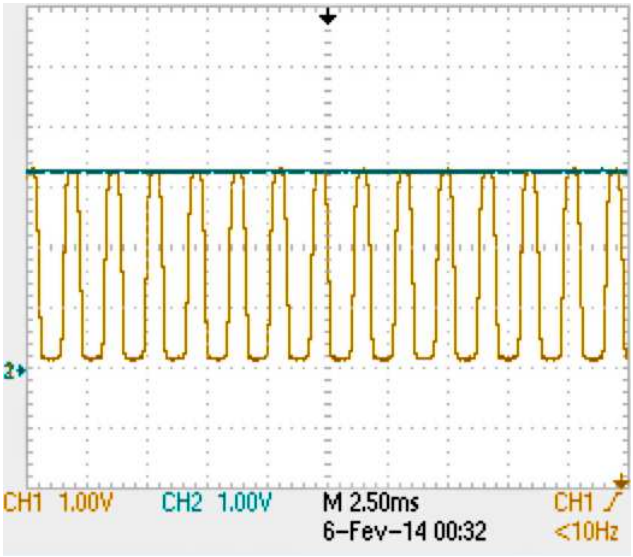


Figura 28: Sinal gerado pelo hodômetro no osciloscópio.

4.2 SENSOR DE DISTÂNCIA

O sensor de distância utilizado foi configurado seguindo sua documentação. Alguns de seus limitantes eram conhecidos a priori, como largura do espectro do feixe de som emitido a 25^{o}. Seu erro máximo em medições é cerca de $2cm$ e suas leituras é de no máximo $5m$ e no mínimo $2cm$. O mesmo foi configurado da seguinte maneira, o pino trigger recebe uma sequencia de pulsos do microcontrolador. E o pino “eco” gera uma borda de subida quando o sensor detecta o eco do ambiente.

Com o intuito de se configurar o sensor de distância, sua resposta foi observada em um osciloscópio, conforme apresentado na Figura 29. Como pode ser visto, o canal 2 apresenta os sinais do pino “trigger”e o canal 1 os sinais do pino “eco”. Ou seja, o canal 2 mostra o sinal emitido pelo microcontrolador para o sensor e o canal 1 apresenta o sinal que indica que o eco do ambiente foi recebido pelo sensor de distância. O cálculo da distância é feito calculando o tempo entre o sinal enviado do

trigger e a borda de decida do sinal eco do sensor.

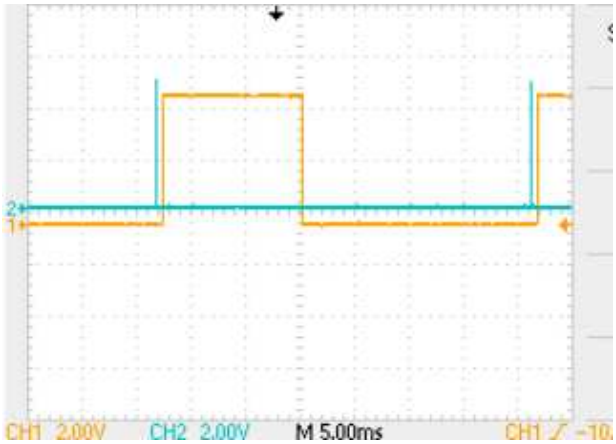


Figura 29: Sinais do sensor de distância.

Este sensor na prática apresentou erros nas leituras com planos inclinados conforme descrito na sessão Discussões/Sensor de Distância.

4.3 PONTE H

A primeira ponte H $L298$ devido as suas características elétricas não suportava a corrente que os motores necessitavam. Devido essa extrapolação do limite de corrente ocasionava a queima do microcontrolador. O problema da queima do microcontrolador foi sanado utilizando-se a segunda ponte H baseada em MOSFET que tem seu limite de corrente maior. Esta segunda ponte H se comportou de maneira adequada e não apresentou mais problemas relacionados a mesma, maiores detalhes do problema ocasionado pela primeira ponte H estão expostos na Discussões.

4.4 INTERFACE DE CONFIGURAÇÃO

No início do projeto quando foi configurado os periféricos com a plataforma Energia, a mesma apresentou problemas que são apresentados na Seção Discussões. Devido aos mesmos, as configurações foram refeitas na ferramenta CCS, a qual utiliza-se da biblioteca TivaWare. Durante a configuração dos periféricos foram desenvolvidas quatro bibliotecas bibliotecas, com todas as configurações dos periféricos do robô que são: encoder.h, PWM.h, sensor_de_distancia.h e controladores.h.

A encoder.h contém as configurações dos hodômetros com todas as configurações mínimas necessárias para o bom funcionamento do mesmo. A PWM.h contém todas as configurações de PWM, para os três motores CC das rodas, que são feitos com timer e a configuração de um PWM de *hardware*, para o servo motor. A sen-

sor_de_distancia.h contém as configurações de dois sensores de distância, o objetivo era inserir três sensores de distância no robô, porém devido a problemas de interferência não foram utilizados, maiores detalhes dos mesmos referenciar corretamente a seção. A controlador.h contém todas as configurações relacionadas com a ponte H, todas as configurações dos controladores on-off, controle proporcional e PID e a interrupção do encoder traseiro. As demais interrupções e funções podem ser encontradas na função principal.

4.5 CONTROLE DE BAIXO NÍVEL

O Controle de baixo nível contou com o controladores on-off e proporcional na roda dianteira e PID e o on-off nas rodas traseiras. Todos os sistemas de controle citados foram desenvolvidos e aplicados nas funções básicas que são, `Giro`, `Alinhamento`, `Deslocamento`, `Aproximação` e `Sensor_anti_colisão`.

O deslocamento pode se utilizar de todos os controles citados, um nas rodas traseiras e um na roda dianteira, podendo trocar a cada execução. O PID não foi aplicado, porque o mesmo não é mais necessário, pois a velocidade máxima de deslocamento do robô ficou limitada devido ao sensor de distância necessitar fazer leituras laterais durante deslocamento. O PID iria ser aplicado para evitar o deslizamento das rodas do robô. A velocidade limitada muito baixa que é suficiente apenas para o mesmo arrancar, assim notou-se que um on-off é suficiente. A função deslocamento se utiliza do on-off nos motores traseiros e on-off ou controle proporcional nas rodas dianteiras.

As Figuras 30 e 31 permitem analisar os resultados obtidos a partir dos controladores on-off e proporcional respectivamente. Enquanto o robô percorre sua trajetória em linha reta, os mapeamentos apresentados representam a leitura dos sensores. Como pode ser visto na Figura 30, o controlador on-off gera muito mais leituras

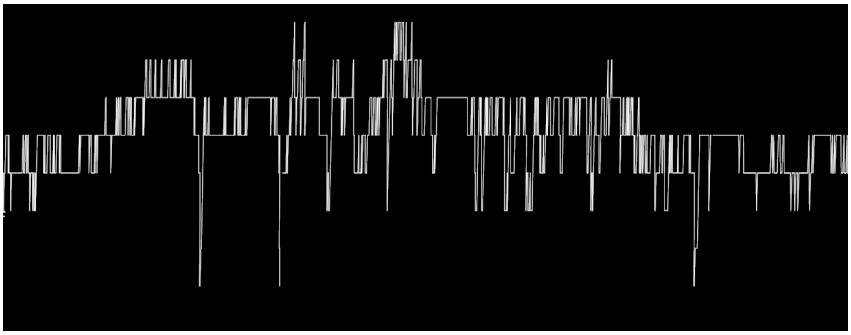


Figura 30: Mapeamento de trajetória e percepção a partir do controle on-off.

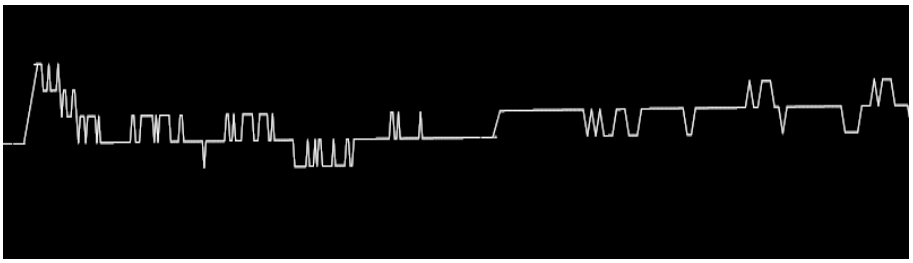


Figura 31: Mapeamento de trajetória e percepção a partir do controle Proporcional.

fora da referência que o controlador proporcional. As leituras fora da referência mostram o comportamento do robô de se manter dentro da faixa de erro de 2cm onde o controlador on-off tem os motores desligados.

Já o controlador proporcional (Figura 31) não gera tantas variações de leituras devido a sua característica de que, se o robô fizer leituras fora da referência, o mesmo terá uma ação de controle diferente de zero. Isto faz com que o controlador mantenha o robô na referência mais facilmente. O caso em que o robô encontra-se fora da referência a 1 cm, a saída do controlador proporcional gerada não é suficiente para deslocar o robô, pois sua saída é de 15% do PWM e o zona morta dos motores para o sistema é de 20% do PWM. Porém, isso irá gerar uma força que impede o robô de se deslocar para longe da referência.

O controle on-off foi calibrado com 2 cm de erro devido as suas características no meio, durante o experimento. Quando a faixa de erro do on-off é menor que 4 cm, o mesmo oscila, acionando os motores para um lado e para o outro. O controle proporcional foi calibrado seguindo esta observação, o mesmo tem saída suficiente para acionar os motores, quando o robô se encontra a 2 cm fora da referência.

4.6 CONTROLE DE ALTO NÍVEL

Durante a fase de desenvolvimento foram encontrados diversos problemas relacionados a como se construir uma percepção do meio que pudesse prover informação para a construção de um mapa métrico do ambiente. Devido às características do arranjo final dos sensores, que não moldaram uma percepção confiável do ambiente, o resultado final do controle de alto nível consistiu apenas do comportamento de seguir de paredes internas.

Apesar da pouca percepção dos sensores, o robô conseguiu realizar movimentos básicos como o giro de 90^o à esquerda, giro de 90^o à direita, deslocamento retilíneo seguindo paredes internas e alinhamento. Estes controles de movimento foram detalhados na Seção 3.2.3. As funções implementadas para o controle de alto

nível provê que o robô de siga paredes, contornando internamente o ambiente. Durante a execução desse controle, dados dos sensores e saída dos atuadores coletados pra que o mapeamento da trajetória do robô pudesse ser representado.

O deslocamento para frente, seguindo uma parede lateral, consiste em se locomover a uma distância variável, que depende do controle de alto nível. Essa característica do controle não gerou problemas na construção do mapeamento de trajetória.

A função de alinhamento é acionada após o robô realizar uma rotação. Essa função conta com o sensor de distância como entrada para o controle da roda dianteira. Basicamente o controle atua quando o robô está fora da sua referência com as paredes. Este movimento é limitado devido às restrições do sensor de distância, que são: espectro de leitura de 25° frontal e inclinação com relação ao plano de impacto do som de no máximo 20°, o que limita muito os movimento do robô e faz com que ele possa se perder durante a navegação.

A função de giro 90^o à esquerda e à direita tem o mesmo princípio de funcionamento, sendo a única diferença a direção dos motores no momento da execução. Esta função conta com apenas o sensor de hodometria. Seu funcionamento é satisfatório quando os motores estão em baixa tensão, de no máximo 35% de PWM. Nos experimentos realizado nesse trabalho, foi aplicado aproximadamente 25% de PWM na maioria dos casos.

A função aproximação funcionou de maneira satisfatória pois seu princípio de funcionamento é elementar, ou seja, a posição final a ser alcançada não requer grande precisão. Esta função antecede o giro e o posterior alinhamento, o qual elimina o erro.

4.7 MAPEAMENTO DA TRAJETÓRIA DO ROBÔ

Apesar das limitações acima citadas, a trajetória do robô e sua percepção do ambiente puderam ser representados na forma de gráficos. A plotagem dos movimentos em linha reta representam os pontos onde o robô executa as ações de maneira mais precisa, ao mesmo tempo em que tem uma boa percepção do ambiente.

Quanto ao gráfico da trajetória completa no ambiente, a função de plotagem não foi bem sucedida. O problema ocorreu quando o robô terminava uma reta (cantos). A função de construção do gráfico armazenava este ponto como sendo a posição do robô, porém isso não é correto. Os pontos plotados correspondem aos dados do

sensor de distância, e não ao ponto em que o robô se localiza. E ainda, como foi visto, o sensor de distância gera erros em sua leitura, o que implica uma construção da trajetória não muito precisa. Uma trajetória que foi construída sem apresentar estas falhas pode ser observada na Figura 32. O gráfico foi gerado a partir da execução do robô em ambiente sem obstáculos. Já em ambientes com obstáculos não foi possível construir gráficos com mapeamento da trajetória do robô de boa qualidade, devido ao problema da quina, o qual foi apresentado.

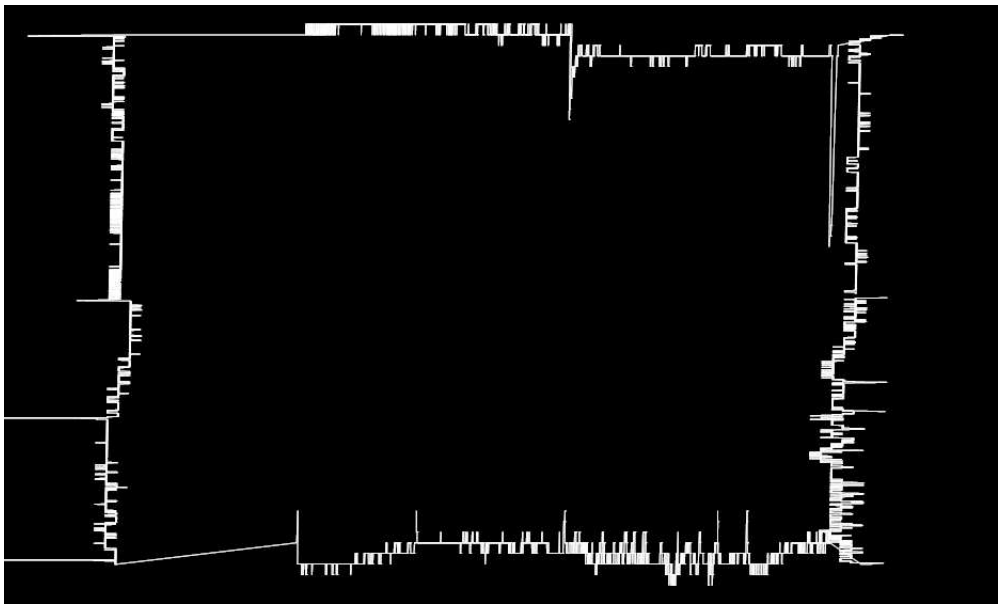


Figura 32: Mapeamento da trajetória em ambiente sem obstáculo .

Como pode ser observado na Figura 32, a trajetória do robô apresenta falhas nos cantos do ambiente. O último ponto de uma reta não coincide com o ponto de início da outra reta perpendicular. As falhas são geradas devido a forma como é construído o mapeamento da trajetória, o mesmo tem como tamanho sem objeto de 2x2 metros, porém como o robô se localiza com base no sensor de distância, se o sensor fazer uma medição errada na distância frontal isto gerará erro no mapeamento de trajetória. Uma solução para este problema é trocar o sensor de distância sonar por outro sensor com maior precisão.

4.8 DISCUSSÕES

Neste capítulo são relatados os problemas encontrados durante as etapas de montagem e configuração dos sensores e atuadores do robô. É descrito também como a ocorrência destes problemas interferiu em sua configuração final e na implementação de seus controles. Além disso, são relatados os problemas inerentes à implementação da tarefa de navegação de alto nível, que consiste de seguir

paredes, desviando de obstáculos. Alguns problemas não são apresentados neste capítulo pois já foram contextualizados no Capítulo 3.

4.8.1 PONTE H

Durante o desenvolvimento do projeto, na etapa de testes dos controladores notou-se a instabilidade do circuito do robô. Esta instabilidade era gerada quando o controlador acionava os motores consecutivamente em direções opostas, esta ação de controle que exige mais do circuito, ultrapassando o limite que a ponte H L298 suportava de 2 Ampère. Com isso, o microcontrolador queimou, devido a sobre corrente nas portas.

Analisando o circuito do robô notou-se que a causa era devido aos motores na inversão de polaridade. O motor gerava oscilações na fonte e a ponte H ultrapassava os limites de corrente, o qual gerava uma corrente de fuga onde a rota era o regulador do microcontrolador. Uma das opções foi inserir resistências em todos os pinos do microcontrolador, reduzindo a corrente dentro dos limites que o microcontrolador suporta. Como os problemas persistiram e o driver de potência continha um diodo de roda livre sem especificação técnica conhecida, optou-se por inserir diodo de roda livre externo, para solucionar os problemas, os quais não foram solucionados.

A única forma encontrada para solucionar estes problemas foi isolar o circuito lógico do circuito de potência, utilizando duas baterias. Também foi trocado o driver de potência por outro com maior margem de segurança, o qual seu limite de tensão é de 3 a $36V$ e corrente é de até $10A$. O driver de potência utilizado é baseado em MOSFET e possui também um sistema de controle, o qual cada motor recebe um pino de direção e outro com o PWM para controle de potência.

4.8.2 SENSOR DE DISTÂNCIA

No início do desenvolvimento do trabalho de conclusão pretendia-se usar 3 sensores de distância, porém devido a problemas com interferência na leitura dos sensores, foi mantido apenas um sensor sobre o servo motor, o que realiza todas as leituras fornecidas ao controle. A interferência era gerada devido a reflexão do som nas paredes, e um sensor fazia a leitura do som que o outro estava transmitido. Sendo assim, optou-se por utilizar apenas um sensor para fazer a leitura do ambiente necessária. Além disso, os dois leitores que estavam sobre o eixo traseiro apenas seriam necessários caso o robô precisasse andar para trás.

O sensor de distância teve seu desempenho afetado quando o robô realizava uma leitura com inclinação do sensor em relação as paredes. Este problema é gerado devido a reflexão do som transmitido com as paredes do ambiente. Como pode ser visto na Figura 33, a qual representa a reflexão do som em uma parede inclinada, S0 é o emissor de som e o S1 é o receptor. Assim, em determinadas situações em que o robô se movimenta e se depara com uma parede inclinada, o mesmo perder sua referência e com isso o algoritmo de controle não retorna uma resposta precisa.

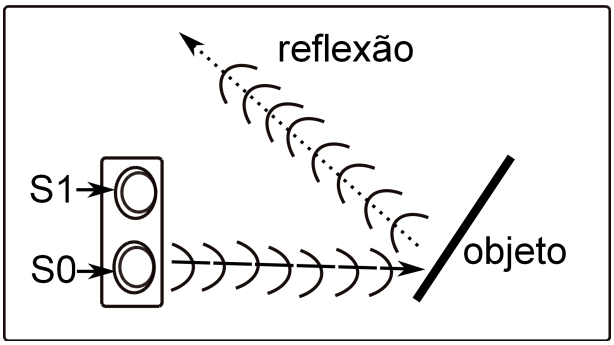


Figura 33: Descrição da trajetória sonora.

Outro problema encontrado foi o erro do sensor de distância ultra-sônico causado quando o robô executa um movimento com velocidade maior que 30% de PWM. Este erro é causado devido as interferências da vibração do robô no deslocamento. Também é causado devido o atraso sonoro causado pelo deslocamento entre a emissão do som e a recepção do mesmo.

4.8.3 INTERFACE DE CONFIGURAÇÃO

Na proposta de desenvolvimento inicial apresentada no TCC1 pretendia-se utilizar a plataforma Energia devido as facilidades aprensentadas por seu fabricante. Segundo Lima (2013), o Energia é um *software open-source* feito para protótipos rápidos com as *Launchpads* da `Texas Instruments`. As Launchpads são placas de avaliação de baixo custo dos microcontroladores da Texas. A interface de trabalho do Energia é mostrada na Figura 34. Outra vantagem da ferramenta é que ela viabiliza a utilização de bibliotecas do microcontrolador `Arduino` (ARDUINO, 2014). Ela busca reproduzir a mesma forma de programação do Arduino em outros microcontroladores, entre eles o `MSP`, `Stellaris` e `Tiva`. Dessa forma, pode-se aplicar códigos ao microprocessador `Tiva`, que são primeiramente destinados à plataforma Arduino.

Sob esta motivação, o objetivo inicial foi utilizar esta ferramenta de desenvolvimento rápido no intuito de construir um robô com um *software* embarcado mais elaborado. Durante três meses foram configurados todos os periféricos envolvidos

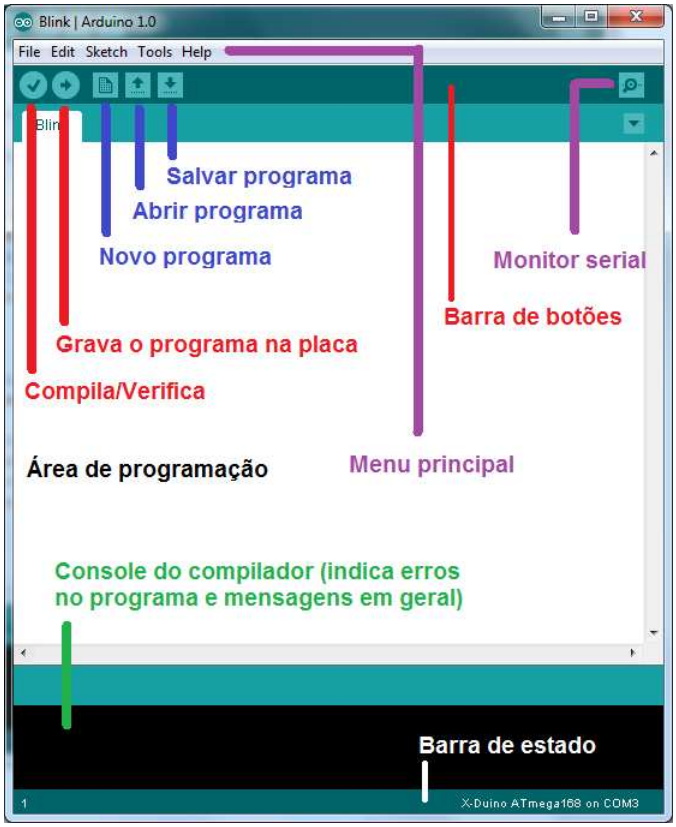


Figura 34: Interface de trabalho da ferramenta © Energia

no projeto, cada periférico foi configurado individualmente com as bibliotecas do Arduino, todos funcionaram bem. Em um segundo momento buscou-se integrar todas as configurações em um único projeto, neste momento foram detectados alguns problemas de conflitos de bibliotecas. Estes conflitos eram gerados devido a utilização de um mesmo periférico, como timers, portas e entre outros com diferentes bibliotecas. Os problemas encontrados tornaram-se difíceis de serem resolvidos pois o Energia não conta com alguns acessórios como depurador de código e devido a falta de *real-time debugging*. Sem estas características tornou-se inviável fazer o passo a passo dos códigos, tornando difícil a tarefa de encontrar os conflitos nas bibliotecas. Após várias tentativas para solucionar os problemas de conflito das bibliotecas, a escolha por esta interface foi deixada de lado.

Buscando se utilizar das configurações que foram feitas no Energia, seguindo as indicações do site (INTRUMENTS, 2014) do fabricante do microcontrolador, o qual informava que as configurações feitas no Energia poderiam ser importadas para a plataforma CCS, porém isso so pode ser feito para os código que se utilizavam de bibliotecas padrão da ferramenta, que não era o caso. Portanto, a solução encontrada foi utilizar o CCS como ferramenta para implementarmos a configuração dos periféricos, o qual dificultou mais o projeto devido a falta de bibliotecas com funções

prontas. O Energia com as bibliotecas do Arduino gerava abstração de código, pois não precisava saber as especificações técnicas dos periféricos. Já com o CCS foi necessário estudar todas as especificações técnicas dos periféricos para implementar as funções de abstração de hardware.

4.8.4 ALGORITMO SEGUIDOR DE PAREDES

Ao desenvolver os primeiros movimentos controlados pelo robô, notou-se uma grande variedade de problemas surgidos durante a implementação do primeiro algoritmo de navegação. Estes problemas surgem quando se analisa o ambiente em contraste com a percepção do robô. Existem vários possíveis movimentos que podem ser executados e em diversos pontos do ambiente, os quais não podem ser previstos.

O primeiro problema é qual movimento executar? Quando se pergunta sobre a resposta nos sensores de qual ação tomar, percebe-se que essa pergunta tem várias saídas para uma mesma entrada. Um ponto em que o robô se encontrava em um impasse lógico está representado na Figura 35, enfatizado com o círculo pontilhado. Quando o robô se posiciona em um ponto em que a parede de referência tem uma quina, qual seria o movimento correto a ser aplicado? Durante esta etapa foram aplicadas várias possíveis lógicas, uma das soluções teve maior qualidade, porém não foi suficiente para evitar falhas nestes pontos.

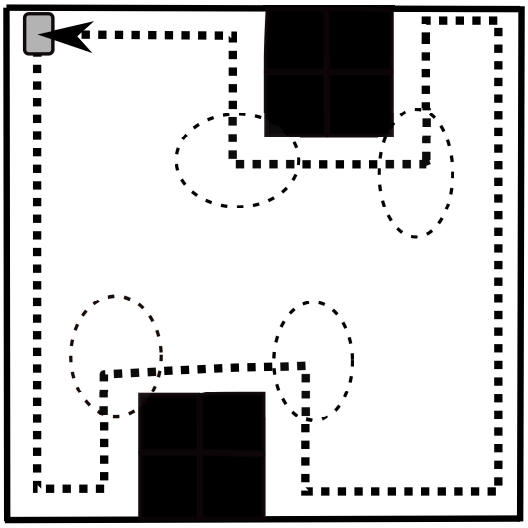


Figura 35: Problema da decisão do próximo movimento.

A lógica aplicada foi deslocar o robô sem manter uma referência de direção e após o deslocamento executar um giro de 90^o. Esta lógica é aplicada quando o sensor faz uma leitura maior que duas vezes a sua referência. E como não pode-se confiar no sensor de distância, quando realiza leituras com uma inclinação com a pa-

rede de referência, o robô pode perder-se após uma leitura errada do sensor. Uma solução para este tipo de problema é usar um sensor que possa passar coordenadas das posições do robô. Como por exemplo, utilizar a técnica de triangulação de antenas, esta técnica utilizando-se de três ou mais transmissores de rádio frequência com a leitura da potência de cada rádio pode se calcular as coordenadas em que o robô se encontra no ambiente.

4.8.5 MAPAS MÉTRICOS

O projeto tinha como objetivo a construção de um mapa métrico do ambiente. Este é uma forma de se representar o ambiente de tal maneira que o mapa seja a representação todas as dimensões do ambiente, bem como o posicionamento de todos os obstáculos. Portanto, o robô final não teve sensores suficientes para calcular as dimensões e as coordenadas dos objetos no ambiente. Por esta razão, apenas se obteve um comportamento de navegação onde o robô segue paredes internas de seu ambiente e um método para representar a trajetória executada pelo mesmo.

5 CONCLUSÃO

Neste projeto de conclusão de curso o objetivo inicial foi desenvolver um robô autônomo, o qual teria como função mapear ambientes, com algumas restrições nos mesmos. Para tal função, foram pesquisados alguns sensores e atuadores para a construção do *hardware* do robô. O sistema de controle implementado é um *software* embarcado, o qual é responsável pelas ações realizadas pelo robô.

Os sensores escolhidos foram sonar, hodômetro e o acelerômetro. O sonar e o hodômetro foram configurados e utilizados. Já o sensor inercial não foi configurado devido as limitações de velocidade que o sonar causou. Os atuadores são os motores CC e o servo motor, os quais foram utilizados. Os circuitos utilizados foram o regulador de tensão $5\ V$ e Ponte H. O microcontrolador utilizado atendeu todos os quesitos relacionados à memória, processamento e a periféricos internos suficientes.

O robô foi desenvolvido a partir da divisão em dois sistemas de controle, o de baixo e o de alto nível. O sistema de baixo nível tem como função tratar de sensores e atuadores, dando suporte para o sistema de alto nível. Como controladores de baixo nível foram implementados o *on-off* e o proporcional. Ambos foram bem sucedidos em controlar a velocidade do robô. Entretanto, a velocidade ficou limitada devido ao sensor de distância (sonar) ter restrições quanto as leituras em movimento, o que repercutiu na não utilização de um controlador PID. Portanto, os sensores e os atuadores foram configurados através do desenvolvimento de funções e controladores que deram abstração ao *hardware* para o sistema de controle de alto nível.

O sistema de alto nível desenvolvido consistiu de um seguidor de paredes que contorna o ambiente, desviando de obstáculos, tendo como referência uma parede lateral. O robô foi testado em dois ambientes, primeiramente sem obstáculos, e em seguida, acrescentando-se obstáculos. No ambiente sem obstáculos o robô realizou sua tarefa, executando uma trajetória sem bater nas paredes. Já no ambiente com obstáculos, o robô não desempenhou sua tarefa de maneira adequada devido a sua percepção restrita, proveniente de seu arranjo de sensores.

A instrumentação foi um dos pontos mais complexos do desse trabalho. De fato, o arranjo final dos sensores utilizados não forneceram dados suficientes para que o robô pudesse construir uma percepção mais abrangente do ambiente. Isto

inviabilizou o objetivo de construir um mapa métrico do mesmo. Sendo assim, a implementação de tarefas de navegação planejadas em camadas mais altas do controle não se tornou possível. Portanto, o sistema de controle de alto nível implementado nesse trabalho consistiu apenas do seguidor de paredes.

A mapeamento das trajetórias realizas pelo robô foi realizado por meio da coleta dos dados dos sensores, durante a navegação. Os dados dos sensores foram transmitidos via comunicação serial para um computador. O programa que parametriza os dados recebidos foi desenvolvido no *software* LabView. Ele faz o tratamento dos dados recebidos gerando um arquivo de pontos, o qual é processado e em seguida gerado o gráfico da trajetória pelo *software* AutoCAD. Esta transformação dos dados dos sensores depende da qualidade dos mesmos. Sendo assim, se os dados sensórios não forem de boa qualidade (ruidosos), serão geradas falhas no mapeamento da trajetória executada.

Portanto, conclui-se que a construção de um robô móvel autônomo envolve diversos problemas que requerem estudo aprofundado de tópicos relacionados a sensores e atuadores. A construção de um sistema de percepção que compreenda informações precisas do meio conta fundamentalmente com a qualidade e amplitude da informação sensória. A qualidade da percepção ser torna um fator crucial na construção de sistemas de controle que implementam tarefas de navegação mais elaboradas, as quais necessitam de um sistema de mapeamento do ambiente eficiente e livre de erros.

REFERÊNCIAS

ANDERSSON, M.; OREBäCK, A.; LINDSTRöM, M.; CHRISTENSEN, H. I. Isr: An intelligent service robot. In: **In IEEE/RSJ International Conference on Intelligent Robots and Systems**. [S.l.: s.n.], 1999.

ARDUINO ©. Ferramenta arduino. `http://arduino.cc/`, acessado em 13/jan. 2014.

ARKIN, Ronald C. **Behavior-Based Robotics**. [S.l.]: MIT Press, 1998. ISBN 0262011654.

BOAS, Elias Ramos Vilas. **Mapeamento e Localização Simultânea de Ambientes Dinâmicos Aplicados na Navegação de Veículo Autônomo Inteligente**. Dissertação — Universidade Federal de Itajubá, 2011.

CHEN, Xuedong; WATANABE, Keigo; KIGUCHI, Kazuo; IZUMI., Kiyotaka. An art-based fuzzy controller for the adaptive navigation of a quadruped robot. **IEE/ASME Transactions On Mechatronics**, v. 7, n. 3, September 2002.

ENERGIA ©. Plataforma open-source da eletrônica de prototipagem. `http://energia.nu/`, acessado em 13/jan. 2014.

FILLIAT, David; MEYER, Jean-Arcady. Map-based navigation in mobilie robots: I. A review of localization strategies. **Cognitive Systems**, v. 4, p. 243–282, 2003.

FILLIAT, David; MEYER, Jean-Arcady. Map-based navigation in mobilie robots: II. A review of map-learning and path-planning strategies. **Cognitive Systems**, v. 4, p. 283–317, 2003.

GOMES, Sérgio Augusto Pereira. Comparação entre métodos de identificação de plantas com resposta ao degrau monotonicamente crescentes e sintonia de controladores pid. `http://monografias.poli.ufrj.br/monografias/monopoli10001117.pdf`, acessado em 21/11/2014. 2008.

IM, Kwang-Young; OH, Se-Young; HAN., Seong-Joo. Evolving a modular neural network-based behavioral fusion using extended vff and environment classification for mobile robot navigation. **IEEE Transactions On Evolutionary Computation**, v. 6, n. 4, August 2002.

INTRUMENTS, TeXas. Site. 2014.

JONES, Joseph L.; FLYNN, Anita M.; SEIGER, Bruce A. **Mobile Robots: Inspiration to Implementation**. Second edition. [S.l.]: A K Peters/CRC Press, 1998.

KIM, S. J.; KIM, B. K. Dynamic ultrasonic hybrid localization system for indoor mobile robots. **IEE Transactions On Industrial Electronics**, v. 60, n. 10, October 2013.

KONOLIGE, Kurt. Improved occupancy grids for map building. **Artificial Intelligence Center. Center for the Study of Language and Information SRI International**, 1997.

LIM, G. H.; SUH, I. H.; SUH, I. Ontology-based unified robot knowledge for service robots in indoor environments. **IEEE Transactions on Systems, Man, And Cybernetics - Part A: Systems And Humans**, v. 41, n. 3, May 2011.

LIMA, Thiago. Ferramenta arduino. `http://www.embarcados.com.br/software-energia/`, acessado em 28/dez. 2013.

LUO, R. C.; LAI, C. C. Enriched indoor map construction based on multisensor fusion approach for intelligent service robot. **IEEE Transactions On Industrial Electronics**, v. 59, n. 8, August 2012.

MATAS, Alexandre Luiz. **sintonia de conroladores PID com controle adaptativo por modelo de referência(MRAC)aplicado aum motor de corrente contínua**. Tese — Engenharia de São Carlos, 2012.

MIGUEL, Godinho Salvado Pedro. **Reconstrução Dinâmica de Mapa Local para o AtlasCar**. Dissertação — Universidade de Aveiro. Departamento de Engenharia Mecânica, 2012.

NEHMZOW, Ulrich. **Mobile Robotics: A Practical Introduction**. [S.l.]: Springer-Verlag London Limited, 2000. ISBN 1852331739.

NOGUEIRA, Maycon Mariano. **Aplicando Lógica Fuzzy no Controle de Robôs Móveis usando Dispositivos Lógicos Programáveis e a Linguagem VHDL.** Dissertação — Universidade Estadual Paulista “Júlio de Mesquita Filho”. Campus de Ilha Solteira, 2013.

OGATA, K. **Engenharia de Controle Moderno**. [S.l.]: Prentice-Hall do Brasil, 1990.

PESSOA, Igor Rodrigues; ARAúJO, Fabiano Guilherme Prado; TURCHENSKI, Rafael Góes. **Desenvolvimento de um robô perseguidor de objetos**. Tese — Universidade Tecnológica Federal do Paraná, http://paginapessoal.utfpr.edu.br/msergio/portuguese/ensino-de-fisica/oficina-de-integracao-ii/oficina-de-integracao-ii/Monog-11-1-Robo-Perseguidor.pdf, 2011.

RUSSELL, Stuart J.; NORVIG, Peter. **Inteligência Artificial**. [S.l.]: Campus, 2004. ISBN 0-13-103805-2.

SILVA, Joao Manoel Gomes da. O controlador pid. Acessado em 24/01. 2000.

TAPUS, Adriana; TAPUS, Cristian; MATARIC, Maja J. User-robot personality matching and assistive robot behavior adaptation for post-stroke rehabilitation therapy. **In Intelligent Service Robotics: Multidisciplinary Collaboration for Socially Assistive Robotics**, n. 1, p. 169–183, April 2008.

TOSHIBA. Smarbo, um robô de limpeza doméstica. Http://worpmidia.com.br/blog/2011/09/toshiba-acaba-de-criar-o-smarbo-um-robo-de-limpeza-domestica/, acessado em 25/out. 2013.

XBOT. Site. 2014.