Centro de Tecnologia e Urbanismo
Departamento de Engenharia Elétrica

Jessica Fernanda Pereira Zamaia

Sistema de Navegação Autônoma para Plataforma Robótica Móvel com Restrições Não-Holonômicas.

Londrina
27 de Abril de 2018

Centro de Tecnologia e Urbanismo
Departamento de Engenharia Elétrica

Jessica Fernanda Pereira Zamaia

Sistema de Navegação Autônoma para Plataforma Robótica Móvel com Restrições Não-Holonômicas.

Dissertação apresentada ao Programa de Pós Graduação em Engenharia Elétrica da Universidade Estadual de Londrina como requisito parcial para obtenção do Título de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr. Leonimer Flávio de Melo

Londrina
27 de Abril de 2018

Ficha Catalográfica

Jessica Fernanda Pereira Zamaia
Sistema de Navegação Autônoma para Plataforma Robótica Móvel com Restrições Não-Holonômicas. - Londrina, 27 de Abril de 2018 - 130 p., 30 cm.
Orientador: Prof. Dr. Leonimer Flávio de Melo

I. Universidade Estadual de Londrina. Mestrado em Engenharia Elétirca. II. Sistema de Navegação Autônoma para Plataforma Robótica Móvel com Restrições Não-Holonômicas..

Jessica Fernanda Pereira Zamaia

Sistema de Navegação Autônoma para Plataforma Robótica Móvel com Restrições Não-Holonômicas.

Dissertação apresentada ao Programa de Pós Graduação em Engenharia Elétrica da Universidade Estadual de Londrina como requisito parcial para obtenção do Título de Mestre em Engenharia Elétrica.

Banca examinadora:

Prof. Dr. Leonimer Flávio de Melo
Departamento de Engenharia Elétrica
Universidade Estadual de Londrina

Profa. Dra. Maria Bernadete de Morais França
Departamento de Engenharia Elétrica
Universidade Estadual de Londrina

Prof. Dr. Francisco Granziera Júnior
Departamento de Engenharia Elétrica
Universidade Estadual de Londrina

Prof. Dr. Cristiano Marcos Agulhari
Departamento de Engenharia Elétrica
Universidade Tecnológica Federal do Paraná - CP

Londrina
27 de Abril de 2018

Dedico este trabalho à Deus, por ter me dado coragem quando era necessário e por ter estado ao meu lado em todos os segundos dessa jornada que é a vida.

Agradecimentos

Agradeço à Deus antes de qualquer coisa, por me dar coragem, pelo suporte emocional e por todas as oportunidades oferecidas pra que tudo ocorresse de uma maneira sempre melhor do que eu havia planejado.

Aos meus pais, Geraldo e Selma, por todo o apoio, investimento e amor fornecidos gratuitamente, por acreditarem em mim e serem meus exemplos de vida. Amo vocês.

Agradeço ao meu irmão, Victor, pelos conselhos e por me fazer parar quando era necessário. Você é chato, mas eu te amo infinitamente.

Ao professor Leonimer por me orientar nesse trabalho e me dar a liberdade necessária para traçar os rumos da pesquisa.

Por fim, agradeço a todos que, mesmo indiretamente, colaboraram para que esse trabalho chegasse até aqui: Carina, Ana Carolina, José Cláudio, Priscila e Jefferson. Vocês são incríveis.

"Não tenhais medo! Abri, melhor, escancarai as portas à Cristo!"
(São João Paulo II)

Jessica Fernanda Pereira Zamaia. 27 de Abril de 2018. 130 p. Dissertação do programa de Mestrado em Engenharia Elétrica - Universidade Estadual de Londrina, Londrina.

Resumo

Este trabalho descreve o processo de desenvolvimento de um protótipo robótico móvel, autônomo, com restrições não-holonômicas e sistema de tração diferencial, capaz de traçar rotas predefinidas e desviar de possíveis obstáculos presentes em seu trajeto através da aplicação de algoritmos de navegação autônoma. A modelagem cinemática e dinâmica é realizada e resulta em equações para o posicionamento do robô em um plano, bem como a sua representação em espaços de estados. Modelagens essas que são aplicadas a simulações do sistema em ambiente virtual e para sua verificação de estabilidade. Simulações do robô em navegação sobre linha reta, circunferência completa e dois algoritmos de desvio são executadas com a utilização do *software* Matlab e, por fim, são realizadas implementações em cinco diferentes ambientes reais de aplicação. Dados são colhidos durante a execução de trajetórias propostas em ambiente real e os mesmos são aplicados em um simulador de trajetória, que permite a visualização dos percursos traçados de forma gráfica. Conclui-se a pesquisa com a comparação entre rotas de navegação simuladas em ambiente virtual e rotas traçadas em ambiente real, obtenção de erro entre elas e as devidas conclusões são apresentadas validando os algoritmos de navegação autônoma propostos.

Palavras-Chave: Veículos Autônomos não Tripulados; Algoritmos de Navegação Autônoma; Otimização de Trajetórias; Modelagem Cinemática e Dinâmica.

Jessica Fernanda Pereira Zamaia. 27 de Abril de 2018. 130 p. Dissertation of the Master's Degree Program in Electrical Engineering - State University of Londrina , Londrina.

Abstract

This work describes the development process of a mobile robotic prototype, autonomous, with non-holonomic constraints and a differential traction system capable of tracing predefined routes and avoiding possible obstacles in its path through the application of autonomous navigation algorithms. Kinematic and dynamic modeling is performed and results in equations for positioning the robot in a plane, as well as its representation in state spaces, which are applied to simulations of the system in virtual environment and for its stability check. Simulations of the robot in straight-line navigation, full circumference, and two deviation algorithms are performed using Matlab software, and finally implementations are performed in five different real-world application environments. Data are collected during the execution of the proposed trajectories in real environment and they are applied in a trajectory simulator that allows the visualization of the routes traced in graphical form. We conclude the research with the comparison between simulated navigation routes in virtual environment and routes drawn in real environment, obtaining error between them and the appropriate conclusions are presented validating the proposed autonomous navigation algorithms.

Key-words: Unmanned Vehicles; Algorithms of Autonomous Navigation; Trajectory Optimization; Kinematic and Dynamic Modeling.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ADC	Conversor Analógico/Digital ou *Analogic to Digital Conversor*;
AGV	Veículo Guiado Automáticamente;
CC	Corrente Contínua;
CCI	Centro de Curvatura Instantânea;
CI	Circuito Integrado;
DAC	Conversor digital/analógico;
EEPROM	Memória Somente de Leitura Programável Apagável Eletricamente ou *Electrically-Erasebla Programmable Read-Only Memory*;
FGM	*Follow the Gap*;
GPS	Sistema de Posicionamento Global ou *Global Positioning System*;
HNA	Algorítmo de Navegação Híbrido ou *Hybrid Navigation Algorithm*;
IBA	Algorítmo Bug Inteligente ou *Bug Intelligent Algorithm*;
IBGE	Instituto Brasileiro de Geografia e Estatística;
ICSP	*In-Circuit Serial Programming*;
IDE	Ambiente de Desenvolvimento Integrado ou *Integrated Development Environment*;
MD	Motor Direito;
ME	Motor Esquerdo;
MISO	*Master IN Slave OUT*;
MOSI	*Master OUT Slave IN*;
NHNA	Novo Algorítmo de Navegação Híbrido ou *New Hybrid Na-vigation Algorithm*;
OMS	Organização Nacional da Saúde;
Pf	Ponto final;
Pi	Ponto inicial;

PPR	Pontos por rotação;
PWM	Modulação de Largura de Pulso ou *Pulse Width Modulation*;
RMA	Robô Móvel Autônomo;
ROS	*Robot Operating System*;
SCK	*Serial Clock*;
SPI	*Serial Peripheral Interface*;
SRAM	Memória Estática de Acesso Aleatório ou *Static Random Access Memory*;
SS	*Slave Select*;
ss	Espaço de Estado ou *state-space*;
UART	Transmissor/Receptor Universal Assíncrono ou *Universal Asynchronous Receiver/Transmitter*;
USB	*Universal Serial Bus*;
VFH	*Vector Field Histogram*;
V-REP	*Virtual Robot Experrimentation Plataform*.

Sumário

1 Introdução

O avanço tecnológico permitiu que, atualmente, a robótica seja vista como algo cotidiano e comum no dia a dia da população.

Robôs podem ser vistos em indústrias pintando, soldando ou fechando embalagens, em ambientes hospitalares auxiliando no processo cirúrgico, em zonas agrícolas plantando e colhendo alimentos, em atividades espaciais explorando outros planetas e até em zonas nucleares e de risco, realizando atividades que seriam de alto perigo ao ser humano (SECCHI, 2008).

A robótica móvel surge nesse âmbito, como área de grande interesse dos pesquisadores devido a, principalmente, o fato de a mesma poder auxiliar e melhorar a qualidade de vida humana de diversas maneiras.

Robôs autônomos que se movimentam livremente dentro de um ambiente permitem o acesso a áreas inóspitas ou com risco a vida humana, como é o caso de incêndios, catástrofes ambientais, desarmamento de explosivos (RUSSEL,1997) e até mesmo a exploração espacial.

Além da substituição humana, a autonomia em projetos móveis é de grande relevância a várias outras áreas da sociedade, podendo auxiliar na vida humana diretamente em contato com essa. Pessoas com dificuldade de locomoção teriam uma grande melhoria em sua qualidade de vida se um transporte seguro, autônomo e eficiente fizesse parte de sua rotina, e o tráfego de grandes cidades poderia ser mais eficiente, com menos acidentes e trânsito mais fluído.

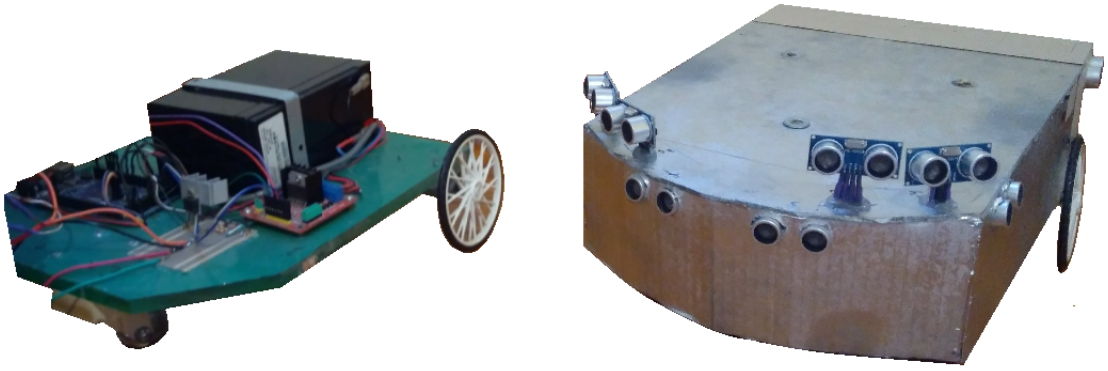
No dia 19 de Março de 2018 um veículo em modo semi-autônomo em testes atropelou uma pessoa nos Estados Unidos (EFE, 2018). No dia 23 de Março do mesmo ano um segundo veículo bateu em uma bifurcação de concreto e seu ocupante faleceu (SANTINO, 2018). Esses fatos evidenciam que a elaboração de um sistema autônomo deve ser feita com extremo cuidado e atenção, porém, também mostram as pesquisas cada vez maiores nessa área e indicam que esses veículos estarão presentes no dia a dia das populações, no que serão os veículos terrestres no futuro.

Tendo como incentivo às grandes melhorias que o avanço da robótica móvel autônoma traria a sociedade de forma geral, o trabalho apresenta o processo de desenvolvimento de um protótipo robótico real e, através do estudo de sua modelagem matemática e validação de estabilidade do sistema proposto, a implementação de algoritmos de navegação são executadas sobre o mesmo.

Baseando-se em Rodrigues (2014) e Melo (2017), rotas são definidas para o robô e, através do controle de sua velocidade angular, o mesmo tem como objetivo percorrer o percurso traçado desviando de possíveis obstáculos que se encontrem no caminho, tendo como algoritmos de desvio o algoritmo BUG 2 e um segundo baseado neste.

O robô desenvolvido é mostrado na Figura 1.1 em sua visão externa, com todos os seus sensores ultrassônicos, e interna, com seus principais componentes.

Figura 1.1 – Robô desenvolvido em sua vista externa e interna.



Fonte: Próprio Autor

1.1 Objetivos Gerais

Inicialmente um robô robusto com duas rodas de tração e uma roda castor foi desenvolvido tendo como base o modelo criado por Rodrigues (2014). Um controle de velocidade para locomoção do protótipo será criado a fim de fornecer autonomia ao robô e, após essa etapa um novo objetivo surge ao projeto: o desvio de obstáculos.

Para que o módulo seja capaz de desviar de obstáculos em sua rota e voltar à mesma após o obstáculo ser contornado, algoritmos (já elaborados pela literatura) foram desenvolvidos de forma que o robô seja capaz de, sozinho, detectar os problemas em sua rota e solucioná-los, sendo assim, um módulo autônomo.

Finalmente, o objetivo final desse trabalho foi o desenvolvimento de um robô móvel, autônomo, capaz de traçar rotas predeterminadas, de retas ou circunferências completas e, se for necessário, desviar de obstáculos presentes em seu percurso de origem, voltando ao mesmo quando o objeto estranho for contornado.

1.2 Objetivos Específicos

- Construção de protótipo robótico para execução de testes em ambiente real de todos os sistemas desenvolvidos durante o trabalho;
- Obtenção do sistema em espaço de estados do robô construído e validação do sistema obtido em ambiente de simulação;
- Execução de simulações de traçados de retas e circunferências para o sistema obtido;
- Desenvolvimento de algoritmo de desvio (BUG 2) sobre o sistema robótico criado;
- Execução de simulações de rotas com desvio de obstáculos para o robô desenvolvido;

- Inclusão de sensores ao módulo robótico físico para início de implementação de algoritmo de desvio em ambiente real;
- Aplicação do algoritmo de desvio estipulado sobre o robô físico em ambiente real.

1.3 Metodologia

A pesquisa aqui descrita teve como ponto de partida a remodelagem de um protótipo robótico já existente. Devido ao tempo que o mesmo estava parado, sua estrutura estava danificada e grande parte de seus componentes deteriorados. Dessa maneira, mantendo apenas os componentes principais do módulo, como motores, o robô foi totalmente refeito a fim de se obter uma estrutura física segura e confiável para execução de posteriores testes.

Um estudo matemático do módulo foi realizado a fim de se obter seu sistema cinemático e dinâmico, ambos de fundamental importância para realização de simulações virtuais. Com o sistema virtual obtido, testes de estabilidade foram realizados para validação do módulo.

Simulações do sistema em todas as rotas de navegação propostas foram realizadas com auxílio do *software* Matlab e posteriormente essas foram comparadas com as rotas traçadas em ambientes reais. Após a validação do sistema, o mesmo foi submetido a testes de trajetória em linha reta e circunferência.

Com as trajetórias executadas com sucesso, o desvio de obstáculos foi acoplado ao sistema. O algoritmo escolhido para o desvio foi o BUG 2 (em duas formas distintas) devido à sua usabilidade em meio acadêmico, que tem sido crescente em ambientes simulados, ou, seu custo de implementação inferior à grande maioria dos algoritmos existentes e facilidade de acoplamento do sensoriamento necessário para o mesmo.

O desvio de obstáculos também foi simulado com auxílio do simulador de trajetórias e, por fim, os algoritmos foram aplicados em diferentes ambientes com o objetivo de verificar se os mesmos realmente podem ser aplicados ao módulo.

1.4 Estruturação do Trabalho

O trabalho está dividido da seguinte maneira:

- Capítulo 1 - INTRODUÇÃO: Apresenta de forma geral o trabalho, situando o contexto em que o qual está inserido, as justificativas para a elaboração do projeto e os objetivos a serem atingidos em seu modelo final;
- Capítulo 2 - REVISÃO BIBLIOGRÁFICA: Apresenta um breve histórico da robótica relacionada à área móvel de autonomia, as principais aplicações de sua utilização no cotidiano humano e os principais algoritmos desenvolvidos pela literatura;

- Capítulo 3 - ESTUDO DO ESTADO DA ARTE: Apresenta alguns exemplos de aplicação de algoritmos de autonomia para robótica apresentada em ambiente acadêmico;
- Capítulo 4 - MODELAGEM MATEMÁTICA DO SISTEMA: Apresenta o tipo de sistema desenvolvido, suas restrições e descreve as modelagens de cinemática e de dinâmica implementadas para o mesmo;
- Capítulo 5 - DESCRIÇÃO DA PLATAFORMA ROBÓTICA: Apresenta os principais materiais utilizados para desenvolvimento da pesquisa, bem como sensores, atuadores, controladores e *softwares* e apresenta suas características mais relevantes para elaboração do trabalho;
- Capítulo 6 - IMPLEMENTAÇÃO DA MODELAGEM MATEMÁTICA E DINÂMICA: Apresenta o processo de desenvolvimento da pesquisa, sua validação de estabilidade e testes referentes à velocidade bem como simulações realizadas;
- Capítulo 7 - IMPLEMENTAÇÃO DE ALGORITMO DE NAVEGAÇÃO AUTÔNOMA EM AMBIENTES REAIS: Traz as formas de utilização do algoritmo BUG 2 aplicadas, e mostra o processo de implementação em um módulo real;
- Capítilo 8 - ANÁLISE DE RESULTADOS: Apresenta os resultados obtidos, comparações entre sistema simulado e implementado em ambiente real e os erros encontrados e realiza uma análise entre resultados esperados e obtidos.
- Capítulo 9 - CONCLUSÃO: Apresenta uma breve recordação de tudo que foi executado até o momento e evidencia a conclusão obtida ao fim da pesquisa.

2 Revisão Bibliográfica

A história da robótica móvel tem pontos de fundamental importância para o entendimento de seu estado atual. Desde o início da robótica em si, datada há milênios, diversos fatores foram cruciais para o surgimento da autonomia dentro dessa ciência.

Esse Capítulo apresenta um breve histórico da robótica até chegar, de fato, na robótica móvel e suas aplicações mais usuais e, por fim, apresenta os principais algoritmos já desenvolvidos para autonomia de sistemas robóticos, mesmo que grande parte destes seja existente apenas em modelos virtuais em simulações.

Para a pesquisa aqui apresentada, o algoritmo de navegação autônoma escolhido para aplicação em sistemas robóticos é o BUG 2. Dessa maneira, nesse Capítulo será dada ênfase À apresentação dos algoritmos dessa família (BUG) e dos que se originam dela.

2.1 Breve Histórico

A palavra robô, como a conhecemos atualmente, é muito nova se comparada à história que engloba seu significado. De origem tcheca, vem do termo *robata* que significa trabalho duro, ou trabalho escravo ("ROBO", 2008-2013).

Os significados atribuídos à palavra atualmente são resultados de uma história bem antiga, iniciada há mais de 4.000 anos, na Grécia, pela necessidade de referenciar mitos que envolviam mecanismos vivos, antes mesmo destes existirem (MAIA, 2003).

Os primeiros relatos de sistemas autômatos existentes de fato na história também são de origem grega e datam de 270 AC. Os robôs, para o povo grego, eram feitos para contemplação, ou seja, eram sistemas artísticos e muito simples. Essa falta de interesse no desenvolvimento de sistemas úteis à sociedade pode ser explicada pelo excesso de mão de obra escrava que se tinha na época e pela forma simples de se viver. Outro fator a ser levado em conta é a tecnologia muito primitiva que esse povo possuía, não sendo possível a execução de grandes trabalhos (PIRES, 2002).

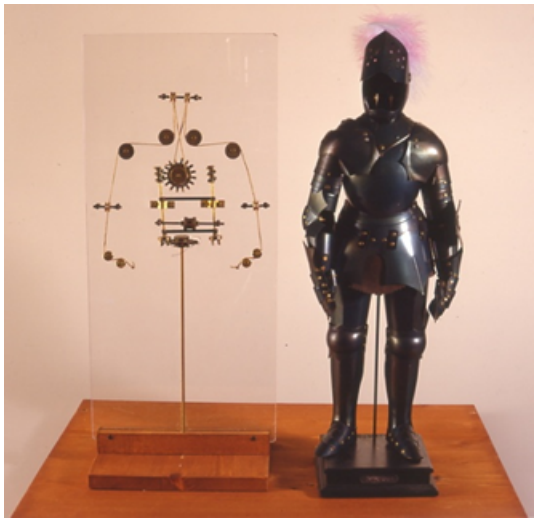
Vários foram os gênios que dedicaram tempo à robótica, entre eles está Philon de Bizâncio (280 AC), também grego. Cabe a Philon a primeira invenção de um robô que fosse útil de alguma forma ao homem. O robô era um sistema complexo composto por diversas bombas, molas, tubos e sistemas de peso que o controlavam. Sua função era basicamente servir um copo de vinho a quem quer que fosse. Porém, não coube a Philon a glória de mudar a história do que hoje conhecemos como robótica. Essa mudança foi provocada pelos árabes que, influenciados pelos estudos dos engenheiros gregos, desenvolveram grandes pesquisas a fim de incluir os robôs de vez nas atividades humanas (MAIA, 2003).

Com o passar dos anos, cada vez mais estudiosos se interessavam pela arte da robótica. O famoso artista Leonardo Da Vinci foi um deles.

Durante seu aprendizado como pintor, Leonardo escutou de Benedetto Aritmético sobre a necessidade de máquinas que agilizassem o trabalho manual dos artistas e, curioso como era, logo desenvolveu um protótipo que moía substâncias a fim de formar as cores que eram usadas em suas telas (ABIMAQ, 2006).

A máquina de moer foi apenas a primeira de muitas invenções de Da Vinci. O primeiro autômato humanóide documentado (1495), conhecido como "robô de Leonardo" (Figura 2.1), foi inspirado em suas pesquisas acerca do corpo humano (ABIMAQ, 2006) e é um dos robôs que serviu de inspiração para o desenvolvimento do primeiro robô móvel na década de 90.

Figura 2.1 – Robô de Leonardo, primeiro autômato humanóide documentado que serviu como inspiração para o desenvolvimento do primeiro robô móvel na década de 90.



Fonte: (GABRIELE, 2011)

Segundo as definições da robótica moderna, o primeiro robô eletrônico é atribuído a Nikola Tesla. O cientista, que é muito conhecido no campo do eletromagnetismo, foi responsável pela criação, em 1898, do "barco tele operado". O trabalho se refere a um submarino acionado eletricamente. O submarino recebe energia de um transmissor ou de uma bateria que permite que o mesmo possa ser acionado e controlado remotamente (CHILDRESS, 1993).

Após Tesla, em 1948, Grey Walter criou o primeiro robô eletrônico autômato, na Inglaterra (ABIMAQ, 2006). A partir desse momento, com o desenvolvimento tecnológico cada vez maior (SOUZA, 2005), surgiu um grande esforço para se automatizar, principalmente, as operações industriais (MAIA, 2003).

No final dos anos 50, uma nova etapa da evolução da robótica se deu com o desenvolvimento dos transistores. Este novo componente eletrônico veio para substituir peças muito grandes que eram usadas anteriormente nos projetos, porém, os transistores ainda não eram sufi-

cientemente pequenos já que os componentes tinham que ser ligados a vários fios para funcionar (SOUZA, 2005).

Em 1958, outra grande evolução surgia na área da eletrônica: os Circuitos Integrados (CIs) (FAVERO, 2011). Os CIs, como são popularmente conhecidos, são componentes eletrônicos muito difundidos atualmente e estão presentes em praticamente qualquer equipamento eletrônico desenvolvido. Seu desenvolvimento deixou clara a possibilidade de criação de circuitos mais complexos, o que de fato aconteceu em 1970 quando a INTEL Corporation lançou o microprocessador (FAVERO, 2011).

Com o passar do tempo os microprocessadores continuaram a evoluir, passando a ter tamanhos cada vez menores e preços também muito reduzidos. Dessa forma o componente passou a ser utilizado em sistemas embarcados, ou seja, sistemas onde o computador é completamente encapsulado e dedicado ao dispositivo que ele controla, tendo funções preestabelecidas e bem definidas. Dentre os sistemas embarcados que aderiram ao uso dos microprocessadores estão os robôs (SOUZA, 2005).

O avanço da robótica a partir do desenvolvimento dos novos componentes eletrônicos e do consequente avanço das linguagens de programação cresceu significativamente, principalmente na indústria. Embora ainda hoje a maior parte dos robôs esteja presente na área industrial, sendo estes, na grande maioria, braços industriais, nas últimas décadas, mais especificamente a partir dos anos 80, têm surgido robôs das mais diversas aplicações, muitas delas não muito convencionais (SOUZA, 2005).

2.2 Aplicações dos Robôs Móveis

A robótica tem estado presente em diversos campos de atuação (HAYES, 2002). Na agricultura, agilizando trabalhos que antes demandavam um tempo extremamente grande para serem executados, nas residências auxiliando em atividades domésticas, ou simplesmente fazendo companhia as pessoas que ali vivem, nos hospitais aprimorando intervenções cirúrgicas, em zonas de risco ao ser humano como limpezas de ambientes tóxicos e trabalhos com explosivos e em diversas outras áreas, inclusive no espaço, desbravando planetas e fazendo novas descobertas (SOUZA, 2005) (WAGNER et al., 1999).

A Figura 2.2 traz exemplos da aplicação de robôs móveis nas mais diversas áreas, mostrando que esses dispositivos estão muito presentes no dia a dia da sociedade, o que torna seu estudo muito importante.

Figura 2.2 – Robôs móveis, desenvolvidos em ambiente fechado de programação, em diversas áreas de atuação como: agricultura, residência, hospital, desarmamento de explosivo e exploração espacial.

Fonte: (IG, 2012)

2.3 Algorítmos para navegação autônoma

A autonomia é o grande objetivo da robótica móvel e, para que a movimentação do sistema se dê de maneira eficiente, duas ações são necessárias simultaneamente: o planejamento da rota e a manobra local, sendo que o primeira está relacionada ao posicionamento do sistema num ambiente e a segunda ao seu deslocamento.

O processo de movimentação de um robô se dá em quatro partes: navegação, mapeamento, cobertura e localização (CHOSET et al., 2004) (AMBROSE; ASKEY, 1995) e a Tabela 2.1 evidencia a importância e ao que se refere cada parte em relação ao sistema robótico e o algoritmo aplicado a este.

Tabela 2.1 – Processo de movimentação de um veículo autônomo e suas divisões segundo Choset et al. (2004)

Processo	Robô	Algoritmo
Navegação	Configuração espacial e graus de liberdade	Movimetação ótima/ não ótima
Mapeamento	Cinemática/Dinâmica	Complexidade computacional
Cobertura	Unidirecional ou não	Resolução
Localização		Sensoreamento

Fonte: (CHOSET et al., 2004)

Vários algoritmos vêm sendo propostos quando se trata de veículos autônomos e do desvio de obstáculos. Tendo em base os últimos 10 anos, os mais frequentes na literatura são o algoritmo BUG, o VFH (*Vector Field Histogram*), o FGM (*Follow the Gap*), o NHNA (*New Hybrid Navigation Algorithm*) e o HNA (*Hybrid Navigation Algorithm*), sendo os dois últimos algoritmos híbridos dos primeiros (ZOHAIB et al., 2014). Este trabalho trata justamente do

estudo e análise dos principais algoritmos e aplicação de um deles ao módulo desenvolvido para essa pesquisa.

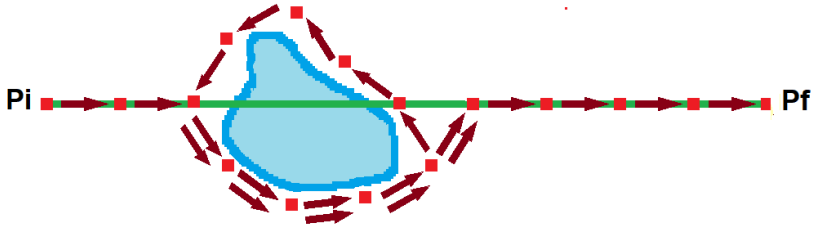
O algoritmo BUG, ou inseto, recebe esse nome devido ao fato de ter sido inspirado na movimentação de insetos, principalmente de formigas, diante um obstáculo em seu caminho (WEI, 2015). Foi um dos primeiros a serem criados para a aplicação de desvio de obstáculos. É um algoritmo bastante simples e teve três modelos desenvolvidos até o momento: BUG 1, BUG 2 e Tangent BUG e os diferentes modelos se devem ao fato do algoritmo ter passado por aperfeiçoamentos ao longo do tempo (ZOHAIB et al., 2014).

Tal algorítmo tem como principal vantagem o fato de sempre percorrer a menor distância possível entre o ponto inicial (Pi) e o ponto final (Pf) de um percurso e possui como desvantagem, o fato de não ter garantia de que o módulo chegará ao local de destino (ZOHAIB et al., 2014). Sua evolução será discutida nas seções a seguir.

2.3.1 BUG 1

O algoritmo BUG 1, o primeiro da série a ser desenvolvido, atua da seguinte forma: tendo um ponto inicial, Pi, e um ponto final, Pf, determinados, uma reta é traçada entre os pontos, calculando-se assim o menor percurso entre os mesmos. Ao detectar um obstáculo em seu caminho, para a trajetória que está percorrendo, anota o ponto em que está e circula todo o obstáculo anotando, a cada ponto, a distância entre o ponto que está e o destino final. Ao circundar todo o obstáculo, analisa todas as medições realizadas e escolhe o ponto onde a distância medida foi a menor. Assim, volta a circundar o obstáculo até atingir o ponto escolhido e a partir dai, segue em linha reta até o ponto de destino final (WEI, 2015). Esse comportamento é descrito pela Figura 2.3.

Figura 2.3 – Mapa de execução de desvio sobre obstáculo simples pelo algoritmo BUG 1, de acordo com a literatura teórica.



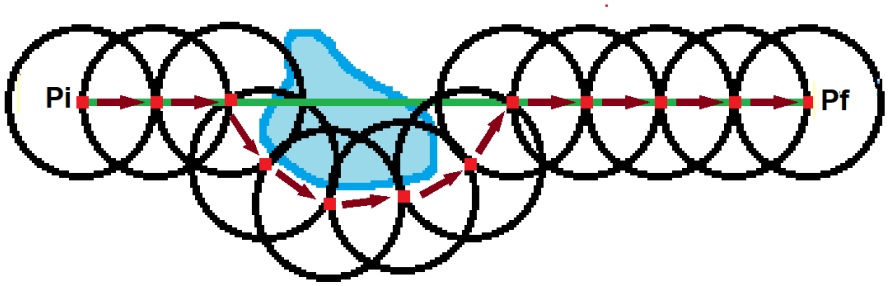
Fonte: Próprio Autor

Os pontos são definidos pelo projetista e podem ser determinados tendo em base a distância percorrida ou o tempo. Na pesquisa os pontos vão ser sempre definidos tomando como base um segundo de rota. Assim, a cada um segundo de rota percorrida, um ponto.

2.3.2 BUG 2

O algoritmo funciona da seguinte forma: tendo um ponto inicial, Pi, e um ponto final, Pf, determinados, uma reta é traçada entre os pontos, calculando-se assim o menor percurso entre os mesmos. Enquanto percorre a reta estabelecida, através de uma angulação e um raio predeterminado, um sensor faz uma busca por obstáculos no percurso a ser percorrido. Quando detecta um obstáculo faz com que o módulo circunde o mesmo até voltar à reta proposta como rota (ZOHAIB et al., 2014). Esse comportamento é descrito pela Figura 2.4.

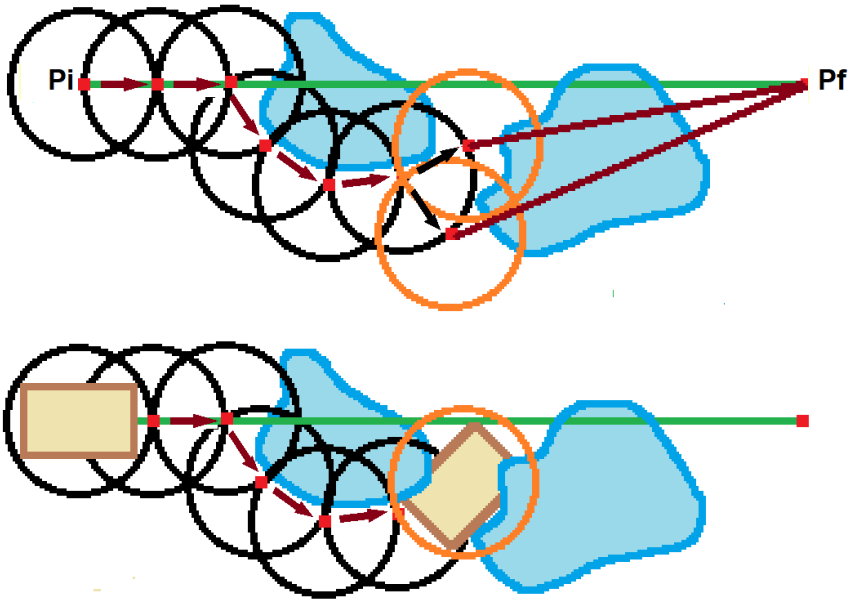
Figura 2.4 – Mapa de execução de desvio sobre obstáculo simples pelo algoritmo BUG 2, de acordo com a literatura teórica.



Fonte: Próprio Autor

O maior problema ocorre quando o algoritmo detecta mais de um obstáculo em seu caminho (SANKARANARAYANAR; VIDYASAGAR, 1990). No momento em que isso ocorre ele pode ficar preso entre dois obstáculos e não chegar a seu destino final. Quando encontra dois obstáculos, a fim de decidir de que maneira vai contorná-los, o algoritmo observa todos os caminhos possíveis e calcula retas dos mesmos até Pf e, por fim, escolhe o caminho onde a reta calculada for de menor comprimento. Caso o módulo escolha passar entre dois obstáculos, corre o risco de ficar preso devido a sua largura, já que o algoritmo não leva esse fator em questão. Esse fato é ilustrado pela Figura 2.5.

Figura 2.5 – Mapa de execução de desvio sobre obstáculo simples pelo algoritmo BUG 2 quando esse se depara com dois obstáculos paralelos, de acordo com a literatura teórica.



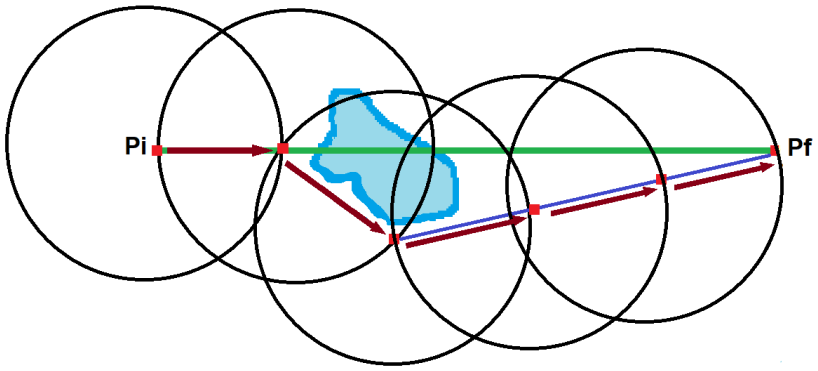
Fonte: Próprio Autor

2.3.3 Tangent BUG

Esse algoritmo é a última versão do algoritmo BUG presente na literatura e recebe esse nome devido ao fato de que, quando utilizado, o veículo autônomo tangencia o objeto que deseja desviar, obtendo um caminho ótimo (WEI, 2015).

Em teoria, o algoritmo funciona da seguinte forma: quando se depara com um obstáculo o sistema começa a contorná-lo ao mesmo tempo que calcula a distância até o destino a partir de cada ponto do contorno. O ponto com a distância mínima é conhecido como ponto de partida. Quando se encontra um ponto onde a distância até o destino é menor que a distância mínima conhecida, calcula uma nova reta até o destino e passa a seguí-la (ZOHAIB et al., 2014). Esse fato é demonstrado pela Figura 2.6.

Figura 2.6 – Mapa de execução de desvio sobre obstáculo simples pelo algoritmo Tangent BUG, de acordo com a literatura teórica.



Fonte: Próprio Autor

Sua principal desvantagem se resume ao fato de que, para ter uma boa eficiência, necessita de sensores com alta resolução e capacidade de medição em grandes distâncias (WEI, 2015).

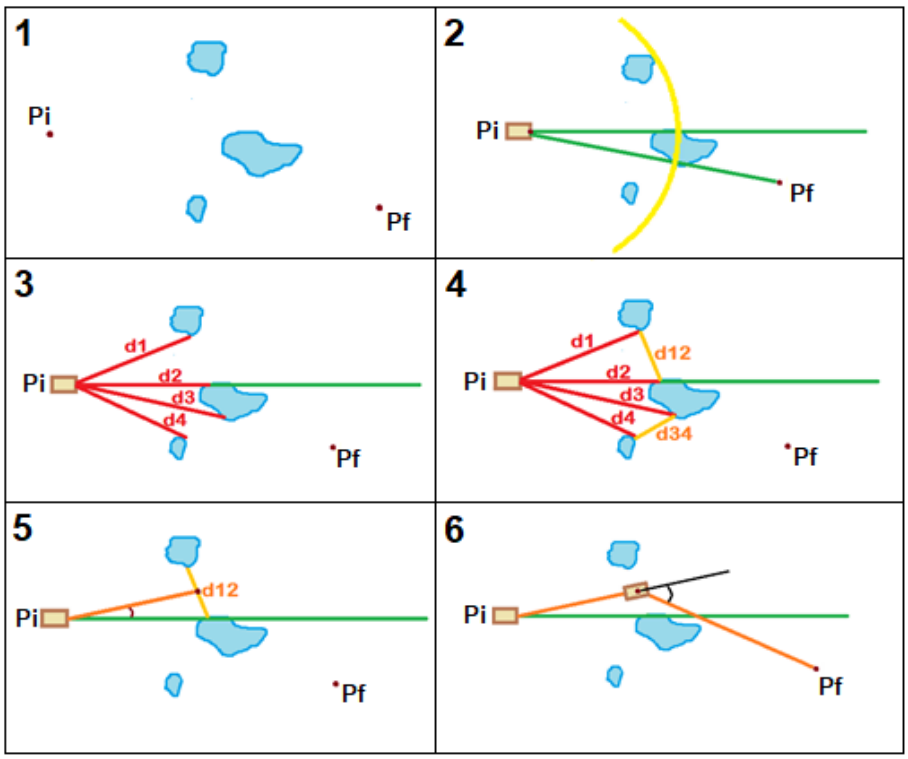
2.3.4 Algoritmo FGM

O algoritmo FGM trabalha em três etapas e evita obstáculos encontrando a distância entre eles. Uma de suas principais características é o fato de trabalhar com a largura do módulo, chamada de abertura limite L (ZOHAIB et al., 2014).

Tem como principal vantagem o fato de não correr o risco de manter o módulo preso em algum obstáculo, garantindo que o mesmo chegue a seu destino final, porém, não executa o percurso entre Pi e Pf necessariamente pelo menor percurso, tornando o tempo de deslocamento muito longo em algumas situações.

O algoritmo funciona da seguinte forma: inicialmente estabelece um eixo central e uma reta entre Pi e Pf. Em seguida passa a buscar por obstáculos em seu percurso de maneira semelhante à apresentada pelo algoritmo BUG. Quando detecta obstáculos, calcula a distância entre o módulo e os mesmos e guarda esses valores em uma matriz. Com os valores obtidos, calcula a distância entre os obstáculos e escolhe a maior delas. Compara a distância escolhida com o comprimento de L e, se o último for menor que o primeiro, calcula o valor do ângulo entre o eixo proposto inicialmente e o ponto central da distância escolhida. Quando o módulo chega ao ponto central entre os obstáculos, calcula o ângulo final até Pf e envia o módulo até ele (ZOHAIB et al., 2014). Todo esse processo é ilustrado pela Figura 2.7.

Figura 2.7 – Mapa de execução de desvio sobre obstáculo simples pelo algoritmo FGM, de acordo com a literatura teórica.



Fonte: Próprio Autor

O principal problema desse algoritmo se deve ao fato dele sempre considerar a maior distância entre obstáculos para enviar o módulo, desconsiderando todas as outras, mesmo que essas sejam adequadas para o envio do robô. Dessa forma, a distância total percorrida pode ser maior que a necessária.

2.4 Conclusão do Capítulo

Sendo a robótica móvel uma ciência que tem crescido a passos largos nos últimos anos, um breve estudo de sua história e a contextualização, principalmente se tratando de algoritmos de desvio, envolvendo autonomia, encontrados na literatura acadêmica, são fundamentais para compreensão de como estes se encontram na atualidade, tanto em questão de desenvolvimento como em áreas possíveis para sua aplicação.

Tendo uma base teórica fundamentada e bem compreendida, o próximo Capítulo desse documento apresenta um estudo do atual estado da arte da robótica móvel em pesquisas acadêmicas, bem como exemplo de atuação dos algoritmos discutidos, tanto virtualmente quanto em ambiente real, trazendo referências reais da aplicação da ciência.

3 Estudo do Estado da Arte

Existem vários tipos de robôs móveis atualmente, como AGV industriais, robôs *indoor* como veículos, holonômicos e humanoides, robôs *outdoor* como os terrestres, sub-marinos, aéreos, inter-planetários e diversos outros modelos.

Embora existam diversos veículos móveis e autônomos no nosso dia a dia, a grande maioria deles está desenvolvida em plataforma de arquitetura fechada, com patentes pertencentes a empresas privadas, o que dificulta o seu estudo.

Para essa pesquisa, serão levados em consideração trabalhos desenvolvidos com finalidade acadêmica e seus resultados, a fim de que se tenha um estado de pesquisa atual e um ponto de partida. É desse estado da arte, envolvendo robótica móvel autônoma, que esse capítulo trata.

3.1 Aplicação de Algoritmo BUG 2 em Módulo Robodeck

Wei (2015), em sua pesquisa, traz um estudo detalhado sobre os principais algoritmos de desvio utilizados na literatura para veículos autônomos móveis, além de discutir os principais sensores e suas tecnologias aplicadas. Após um estudo teórico detalhado o autor passa a aplicação de um dos algoritmos a uma plataforma robótica denominada Robodeck.

O Robodeck é uma plataforma robótica móvel desenvolvida pela empresa XBOT (WEI, 2015). Trata-se de um veículo terrestre com quatro rodas e diversos sensores como ultrassônicos, *web cam*, bússola, GPS (Sistema de Posicionamento Global ou *global positioning system*), acelerômetro, sensor infravermelho e sensor de temperatura. O robô pode ser visto na Figura 3.1.

Figura 3.1 – Plataforma Robodeck (plataforma robótica móvel desenvolvida pela empresa XBOT).

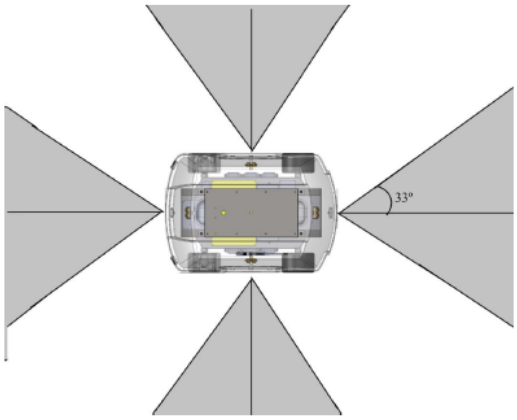
Fonte: (WEI, 2015)

Através do controle de tração presente nas duas rodas dianteiras e o controle de direção

independente para as quatro rodas do robô, o autor aplica o algoritmo BUG 2 ao veículo a fim de validar sua aplicabilidade.

Para detecção de obstáculos, os sensores ultrassônicos foram os escolhidos pelo autor, que menciona a facilidade de medição pelos mesmos. Os quatro sensores disponíveis no módulo são utilizados e seus ângulos de abertura e a área de identificação de objetos em torno do módulo pode ser vistas na Figura 3.2.

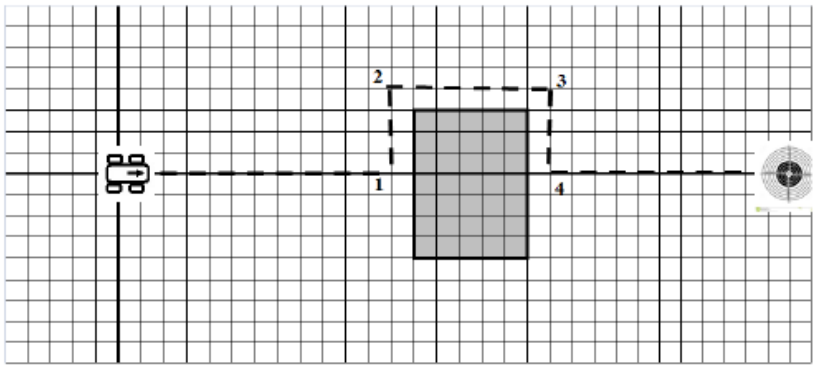
Figura 3.2 – Área de leitura de sensores ultrassônicos acoplados sobre plataforma Robodeck (plataforma robótica móvel desenvolvida pela empresa XBOT).



Fonte: (WEI, 2015)

O código de controle para a plataforma foi desenvolvido, na pesquisa, em linguagem Java na IDE NetBeans e apresenta a rota pretendida para o sistema diante de um obstáculo simples como a evidenciada pela Figura 3.3.

Figura 3.3 – Rota pretendida para Robodeck (plataforma robótica móvel desenvolvida pela empresa XBOT) diante de obstáculo simples.



Fonte: (WEI, 2015)

Após experimentos realizados em ambiente fechado e horizontal, o autor conclui que o sistema, embora realize o contorno do objeto, não chega ao ponto final pretendido e, quanto

mais curvas realiza, mais distante desse ponto fica, problema que atribui ao acúmulo de erros durante as execuções de curva do sistema.

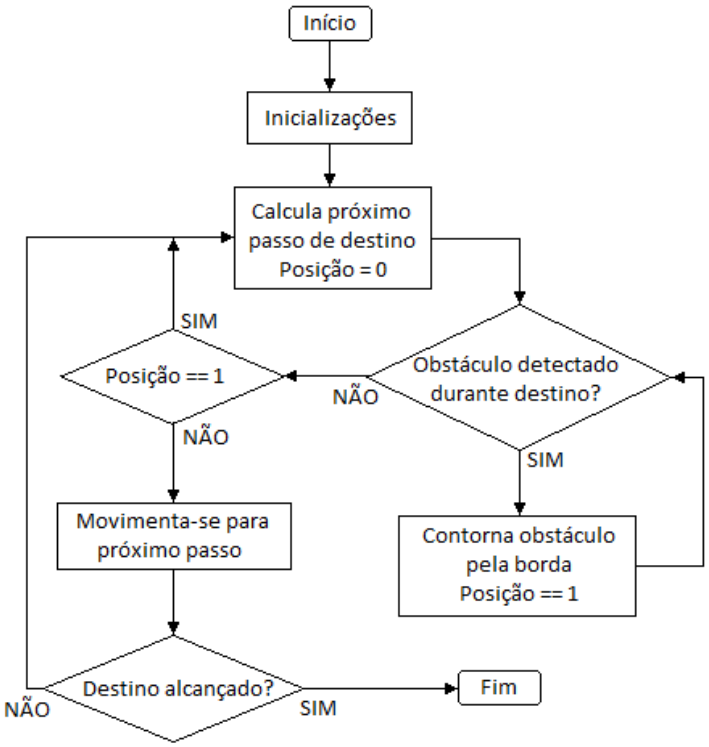
3.2 Aplicação de Algoritmo BUG em Ambiente Virtual

Na pesquisa, desenvolvida por Zohaib et al. (2014), novamente o algoritmo BUG é levado em conta, porém de maneira distinta do realizado por Wei (2015).

Com o objetivo de melhorar o algoritmo, o autor aborda seus problemas e limitações e propõe um novo algoritmo nomeado por ele de IBA (*Bug Intelligent Algorithm*), que não leva em conta o ponto de chegada, assim como os outros, mas sim a trajetória que irá percorrer até o ponto final.

O funcionamento inicial é igual ao do algoritmo BUG, evidenciado no Capítulo 2 desse documento. Uma reta entre ponto de partida e ponto de destino é traçada e, por ela, o sistema segue até se deparar com um obstáculo. É nessa etapa que o processo se torna diferente. Ao invés de contornar o obstáculo e escolher o ponto de menor distância até o destino, o sistema analisa também se há obstáculos entre os dois e , se houver, exclui a rota das possíveis a serem seguidas, como mostra o fluxograma do algoritmo, na Figura 3.4.

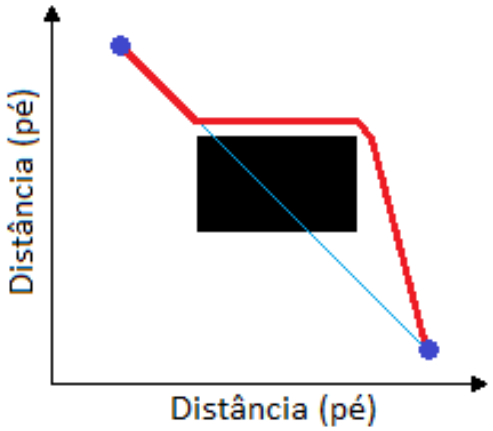
Figura 3.4 – Fluxograma de algoritmo de desvio denominado IBA, algoritmo que teve como inspiração o BUG.



Fonte: Traduzido de (ZOHAIB et al., 2014)

Segundo o autor, de acordo com simulações realizadas, como a evidenciada pela Figura 3.5, o algoritmo se mostra eficiente e segue um rota curta e suave, atingindo o objetivo num curto tempo, porém, o mesmo apresenta restrições a obstáculos que não sejam simples, como os em formato de U ou H. O sistema também não é aplicado em ambientes reais o que faz com que o mesmo não seja validado na prática.

Figura 3.5 – Simulação de desvio de obstáculo simples pelo algorítmo IBA, algorítmo que teve como inspiração o BUG. (2014).



Fonte: Traduzido de (ZOHAIB et al., 2014)

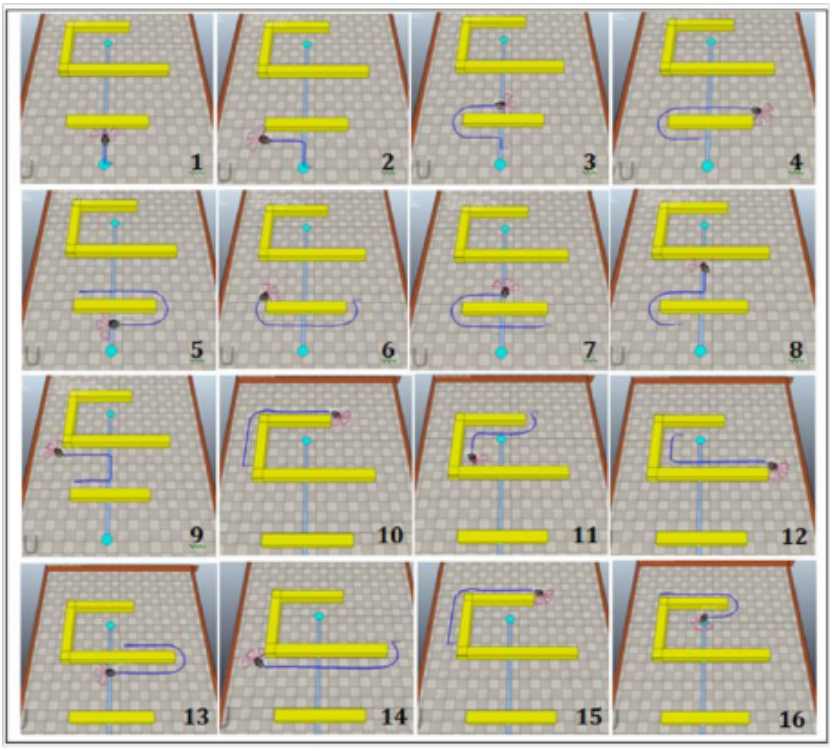
3.3 Comparação entre Algoritmo BUG 1 e BUG 2 por um Veículo Autônomo

Um terceiro exemplo de aplicação da autonomia em robôs móveis desenvolvida em pesquisa acadêmica é o trabalho proposto por Scanavini (2016). Tendo como ponto de partida o sistema *Robot Operating System* e utilizando a linguagem de programação C++ os algorítmos BUG 1 e BUG2 foram testados. Com auxílio do *software* V-REP (*Virtual Robot Experrimentation Plataform*) Scanavini (2016) realiza simulações com um módulo terrestre a fim de verificar as vantagens e desvantagens de cada algoritmo e por fim, realiza uma comparação entre eles.

O autor desenvolve diferentes mapas de simulação e executa os algoritmos sobre todos eles a fim de verificar suas diferenças e estabelecer seus pontos positivos e negativos.

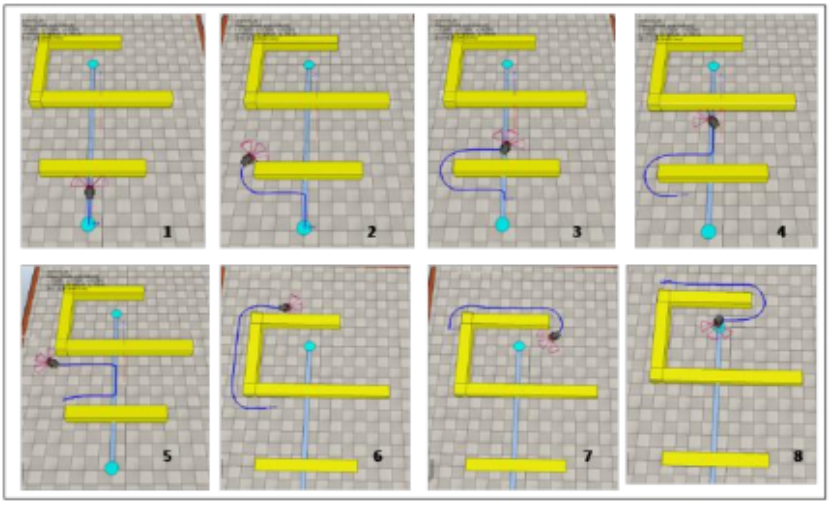
Levando em conta um mesmo mapa, para efeito de ilustração, as Figuras 3.6 e 3.7 mostram o resultado da simulação de trajetória desenvolvidas pelos algoritmos BUG 1 e BUG2, respectivamente. Nas Figuras, o estado (1) se refere ao ponto de partida do robô.

Figura 3.6 – Simulação de desvio de obstáculo pelo algorítmo BUG 1 com auxílio do software V-REP sobre o sistema ROS desenvolvido por Scanavini (2016).



Fonte: (SCANAVINI, 2016)

Figura 3.7 – Simulação de desvio de obstáculo pelo algoritmo BUG 2 com auxílio do software V-REP sobre o sistema ROS desenvolvido por Scanavini (2016).

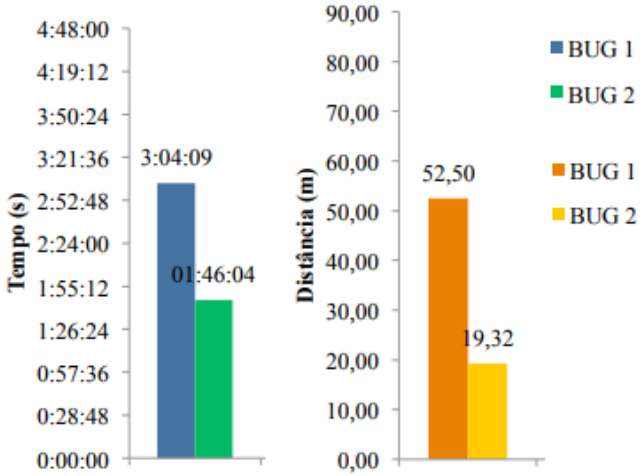


Fonte: (SCANAVINI, 2016)

Para os dois algoritmos, os tempos para execução de cada parte da trajetória foi registrado e esses foram diferentes entre si. Enquanto para o algoritmo BUG 1 o módulo leva 3,04 segundos, aproximadamente, para percorrer todo o trajeto do ponto inicial ao final, enquanto

que o algoritmo BUG 2 leva apenas 1,46 segundos. Já em relação à distância percorrida pelo módulo nos diferentes algoritmos, tem-se que para o BUG 1 o robô percorre 52,50 metros enquanto que para o BUG 2 a distância percorrida é de 19,32 metros. Um gráfico referente a esses resultados pode ser observado na Figura 3.8.

Figura 3.8 – Comparação entre simulações de desvio de obstáculo executadas pelos algoritmos BUG 1 e BUG 2 com auxílio do software V-REP sobre o sistema ROS.



Fonte: (SCANAVINI, 2016)

Pela pesquisa, Scanavini (2016) conclui que o algoritmo BUG 1 é mais lento e percorre uma distância maior até chegar em seu destino final devido ao fato de sempre contornar todo o obstáculo antes de tomar uma decisão e seguir para o objetivo final. Já o BUG 2 apresenta um desempenho mais evoluído e mantém a simplicidade de código e o baixo custo para execução, sendo assim, mais indicado para aplicações em ambientes reais.

Vale lembrar que na pesquisa o autor realiza todos os testes em ambiente virtual simulando um robô terrestre, porém, não aplica os algoritmos de fato em ambiente real.

3.4 Conclusão do capítulo

Algoritmos de controle para que robôs móveis se tornem autônomos podem ser vistos em aplicações diariamente, como mostra o Capítulo 2 desse documento. Entretanto, a maioria dos algoritmos desenvolvidos se encontra em ambiente fechado de desenvolvimento, permitindo apenas que seu desenvolvedor o conheça e estude.

Nesse capítulo, algoritmos desenvolvidos em ambiente aberto, tendo como propósito a pesquisa acadêmica, se fazem presentes com exemplos de desenvolvimento tanto em ambiente simulado quanto em ambiente real bem como a comparação entre alguns deles, proporcionando um estado atual da arte para que o trabalho aqui descrito possa ter prosseguimento de maneira eficiente.

Embora a aplicação de um algoritmo de controle e desvio seja essencial no desenvolvimento de um sistema autônomo, para que o mesmo possa ser desenvolvido de maneira correta, um estudo detalhado do veículo a ser utilizado precisa ser realizado e esse estudo é realizado no próximo Capítulo.

4 Modelagem Matemática do Sistema

Sendo robótica a ciência que estuda a conexão inteligente entre percepção e ação de uma estrutura, o profundo estudo dessa estrutura é de fundamental importância no processo de elaboração de um sistema e esse estudo será realizado no presente Capítulo.

Inicialmente, as principais características estruturais do módulo serão descritas assim como seu modelo mecânico. Posteriormente, a modelagem matemática cinemática e dinâmica do módulo será descrita passo a passo. É através dessa modelagem que se dará todo o processo de simulação do módulo dentro do simulador virtual utilizado, bem como validação de estabilidade realizada através de sua equação de transferência.

4.1 Composição Mecânica do Módulo Proposto

Certas partes são fundamentais em um robô para que a formação de sua estrutura seja concisa e robusta. A Figura 4.1 mostra estas partes de forma simples e as mesmas serão explicadas e detalhadas no decorrer desse trabalho.

Figura 4.1 – Componentes básicos de um sistema robótico para formação de uma estrutura concisa para o mesmo.

Fonte: (SICILIANO et al., 2009)

O sistema atuador corresponde aos componentes do robô capazes de gerar alguma ação ao módulo mecânico, no caso deste trabalho são motores de corrente contínua que geram movimento às rodas de tração do módulo.

O sistema sensorial é formado pelos componentes que comandam o comportamento do módulo através de coleta de dados. No robô desenvolvido na pesquisa aqui apresentada são representados por encoders incrementais que leem certo número de pulsos a cada rotação completa realizada pelo eixo do motor CC (Corrente Contínua).

O sistema de controle é o sistema que integra os dois primeiros (sensorial e de atuação). Ele é responsável por analisar os dados lidos pelos sensores e, através deles, tomar decisões sobre as ações futuras do módulo, que serão executadas pelos atuadores.

Todos esses sistemas são interligados pelo módulo físico. Para que as tomadas de decisão sejam feitas de maneira satisfatória, e para que o sistema de controle seja eficiente, a estrutura física do módulo em atuação deve, então, ser estudada de maneira completa.

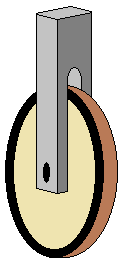
Robôs são classificados, basicamente, em fixos ou móveis, sendo os primeiros os manipuladores que tem base fixa e os últimos módulos que tem capacidade de se movimentar por um ambiente (WOLF et al., 2009). Dado que este trabalho trata de um robô móvel, será dada ênfase a esse modelo nessa seção.

Mecanicamente, um robô móvel é constituído de um (ou vários) corpo rígido equipado com um sistema de locomoção baseado em rodas ou juntas móveis, sendo que o primeiro tipo representa a grande maioria dos modelos móveis já desenvolvidos.

Os protótipos que tem seu movimento controlado por rodas podem possuir basicamente, três modelos diferentes:

- **I-Roda Fixa:** Roda que gira apenas sobre um eixo que é ortogonal ao eixo correspondente ao eixo da roda (Figura 4.2).

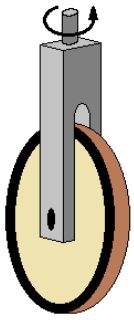
Figura 4.2 – Esquema gráfico representando o modelo de Roda Fixa



Fonte: (SICILIANO et al., 2009)

- **II-Roda Orientável:** Roda que gira sobre dois eixos. Um é o mesmo eixo de movimento que possui a roda fixa e o outro é o eixo vertical, que tem seu centro perpendicular ao eixo da roda (Figura 4.3).

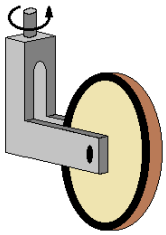
Figura 4.3 – Esquema gráfico representando o modelo de Roda Orientável



Fonte: (SICILIANO et al., 2009)

- **III-Roda Castor:** Roda que gira sobre dois eixos, assim como a roda orientável, com a diferença que o segundo eixo de rotação tem um *offset* em relação ao centro da roda (Figura 4.4).

Figura 4.4 – Esquema gráfico representando o modelo de Roda Castor

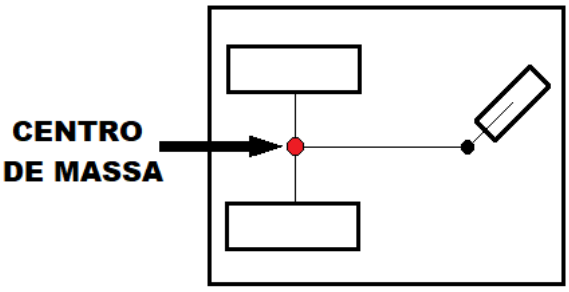


Fonte: (SICILIANO et al., 2009)

A quantidade de sistemas provenientes da aplicação dos modelos de roda apresentados é enorme, porém um tem grande predominância nas aplicações científicas e será também utilizado aqui, que é conhecido como sistema diferencial.

Um veículo com sistema diferencial possui duas rodas fixas com eixo de rotação em comum e uma (ou mais) roda castor para apoio. As rodas fixas possuem controles independentes entre si, podendo ter velocidades distintas, e são elas as responsáveis a dar movimento ao robô, sendo este movimento estipulado através do centro de massa do sistema, que está localizado entre as rodas de tração (SICILIANO et al., 2009). O sistema em questão, bem como a localização do ponto de centro de massa por onde se dá a localização do módulo, estão representados na Figura 4.5.

Figura 4.5 – Representação gráfica de um sistema diferencial de locomoção visto por baixo.

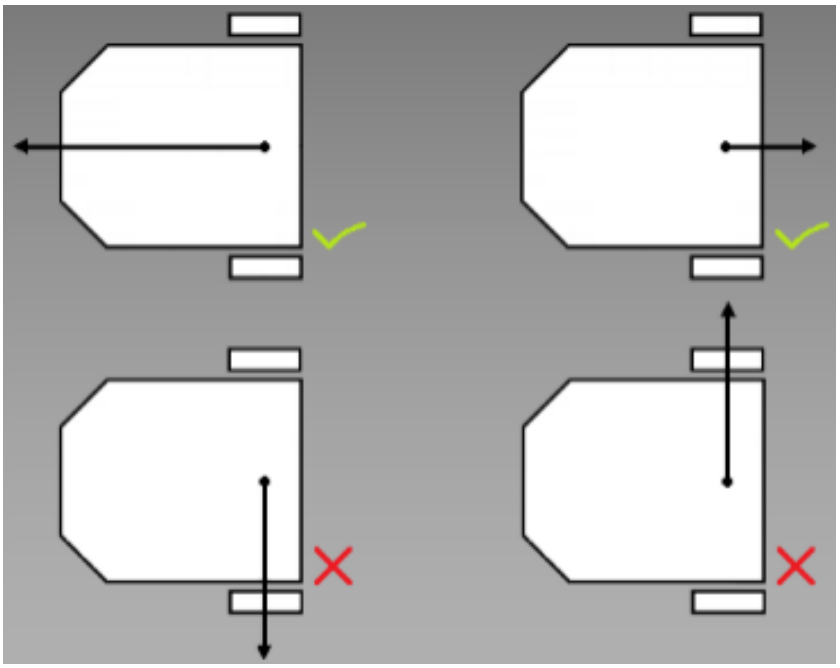


Fonte: (SICILIANO et al., 2009)

O sistema desenvolvido possui em sua orientação uma restrição não-holonômica, ou seja, reduzida dimensão do espaço de possíveis movimento diferenciais, ou o espaço das direções de velocidade (KUHNE, 2005).

O módulo, então, só pode se deslocar na direção normal ao eixo das rodas motoras, assim como mostrado na Figura 4.6 a seguir.

Figura 4.6 – Espaço de movimentação do sistema com restrição não-holonômica desenvolvido.



Fonte:Próprio Autor

4.2 Cinemática do Robô Móvel

Para se criar um *software* de controle para um robô móvel é de fundamental importância que se entenda a cinemática do mesmo, pois é ela que irá explicar o funcionamento mecânico do sistema em questão e, assim, possibilitará o desenvolvimento de um controle correto.

Para o robô desenvolvido o estudo da cinemática será aplicado em um simulador virtual de trajetória a fim de que se possa visualizar suas trajetórias pretendidas e, posteriormente, as executadas em ambiente real de simulação.

A robótica móvel tem sofrido um grande avanço nas últimas décadas e com isso, o estudo da cinemática também vem passando por ajustes e ficando cada vez mais complexo. Esse estudo pode ser visto em manipuladores industriais com diversos eixos móveis e em sistemas mais simples de tração diferencial. O que não se pode questionar é a fundamental importância para cada um dos sistemas, sendo ele mais simples ou mais complexo (SICILIANO et al., 2009).

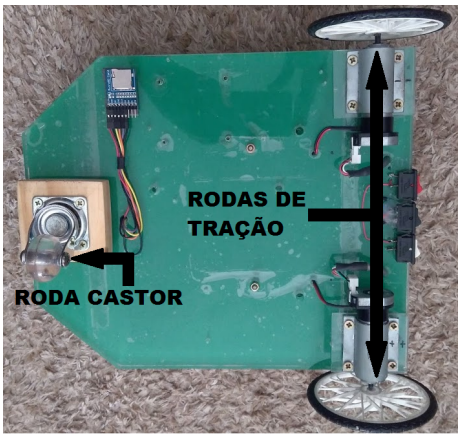
Partindo de um robô manipulador, tem-se que este está restrito a um espaço de trabalho e é a partir deste ambiente que sua cinemática deve ser estudada. Para um robô móvel a restrição ambiental também é de fundamental importância para se definir caminhos a se percorrer, porém seu dinamismo também interfere no comportamento de suas ações devido ao fato do mesmo não estar fixo sobre uma base sólida, podendo sofrer capotamentos, ficar preso, entre outros problemas. Além disso, a medição da posição de um robô móvel é muito complexa. Para se obter um posicionamento correto é necessária a aplicação de integrais de seu movimento através do tempo adicionado aos erros ocasionados por derrapagens, obstáculos e outros empecilhos que

tornam a medição do seu posicionamento uma tarefa complexa (SIEGWART; NOURBAKHSH, 2004a) (MELO, 2007).

Conhecendo as dificuldades apresentadas e a importância que um estudo correto da cinemática de um robô móvel apresenta, esse estudo será aqui realizado para o robô móvel projetado nesse trabalho.

Um dos primeiros pontos a ser levado em conta é que o robô aqui em estudo (Figura 4.7) possui duas rodas de tração com funcionamento independentes e uma roda castor, ou seja, duas rodas que contribuem individualmente para o movimento do sistema e de formas distintas entre si, unidas ao módulo pelo chassi e tendo seus movimentos combinados para a atuação final do modelo, tendo assim seus erros também combinados e uma roda atuando apenas como apoio ao sistema.

Figura 4.7 – Robô móvel desenvolvido e posicionamento de suas rodas, conforme vista inferior do módulo.

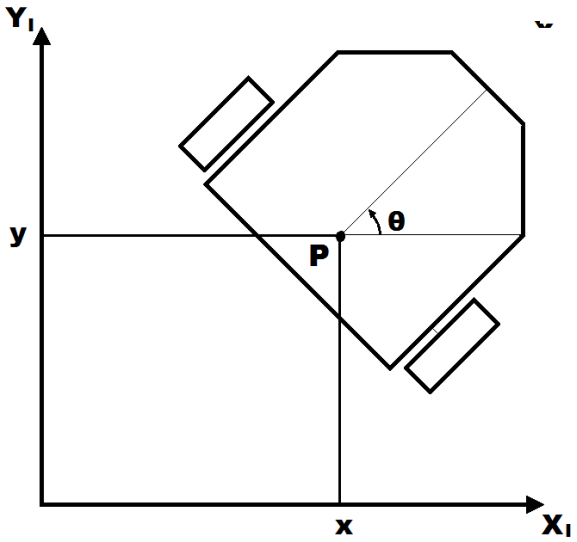


Fonte: Próprio Autor

4.2.1 Modelo cinemático direto

Colocando o robô sobre um plano horizontal (X_I,Y_I), o sistema apresenta três graus de liberdade (x,y,θ), dois referentes ao posicionamento linear, e um referente ao deslocamento angular do modelo. Esse sistema é conhecido como plano de referência global do sistema e pode ser visto na Figura 4.8.

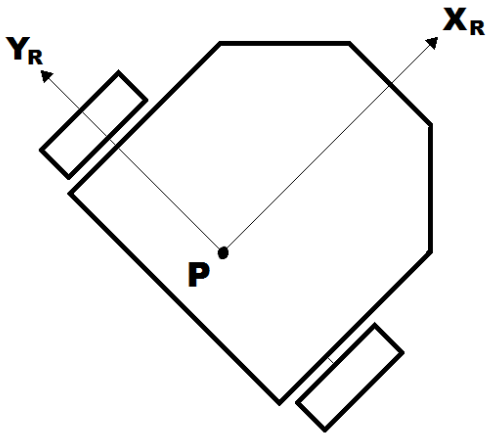
Figura 4.8 – Plano de referência global de um sistema, plano horizontal (X_I,Y_I) sobre o qual o módulo apresenta três graus de liberdade (x,y,θ).



Fonte: Próprio Autor

Aplicando um segundo plano com pontos de referência no espaço (X_R,Y_R) damos origem ao que é chamado de referência local do sistema que pode ser observado na Figura 4.9.

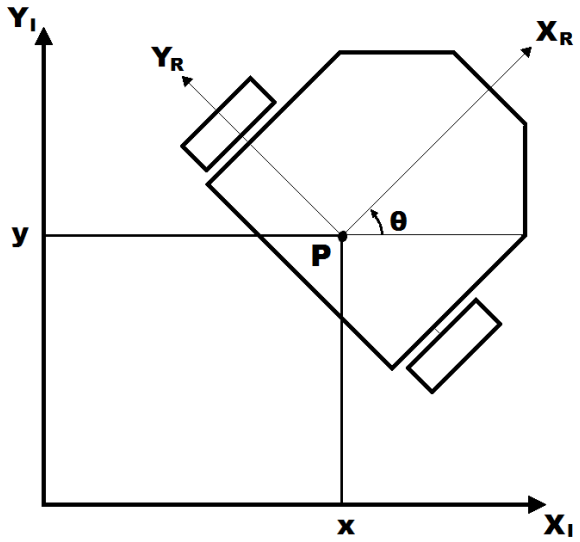
Figura 4.9 – Plano de referência local de um sistema, plano de referência espacial (X_R,Y_R) que fornece a posição de um objeto sobre o espaço definido para se locomover.



Fonte: Próprio Autor

Com o estudo dessas duas referências (global e local) é possivel especificar o posicionamento de um robô móvel sobre o plano. Para isso basta se obter a diferença angular (θ) entre os dois planos de referência, como é mostrado na Figura 4.10 (MELO, 2007).

Figura 4.10 – Diferença entre planos de referência global e local que é a responsável por especificar o posicionamento do módulo sobre o espaço definido para se locomover.



Fonte: (MELO, 2007)

O ponto P é o ponto de referência por onde será calculado todo o deslocamento do robô. Este foi alocado no eixo central entre as rodas de tração por serem essas que comandam a movimentação do módulo.

Descrevendo o posicionamento global do robô através de um vetor ξ_I, representado pela equação a seguir:

$$\xi_I = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \tag{4.1}$$

tem-se então três elementos: x, y e θ. Para se localizar o robô no espaço e assim conseguir descrever seu movimento em um plano, é necessário mapear sua movimentação ao longo do plano de referência global (X_R,Y_R) para o plano local. Para se obter esse mapeamento a matriz rotacional ortogonal ($\mathbf{R}(\theta)$), seguinte, é utilizada:

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.2}$$

Relacionando o vetor de posição local, ξ_I, com a matriz rotacional ortogonal, ***R****(θ)*, obtém-se o vetor de mapeamento da posição global para a local do módulo ((SIEGWART; NOURBAKHSH, 2004b)), ξ_R, dada por

$$\xi_R = \mathbf{R}(\theta)\xi_I, \tag{4.3}$$

que com as substituições dadas pela Eq. 4.1 e pela Eq. 4.2 torna-se:

$$\xi_R = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}. \tag{4.4}$$

De outra forma:

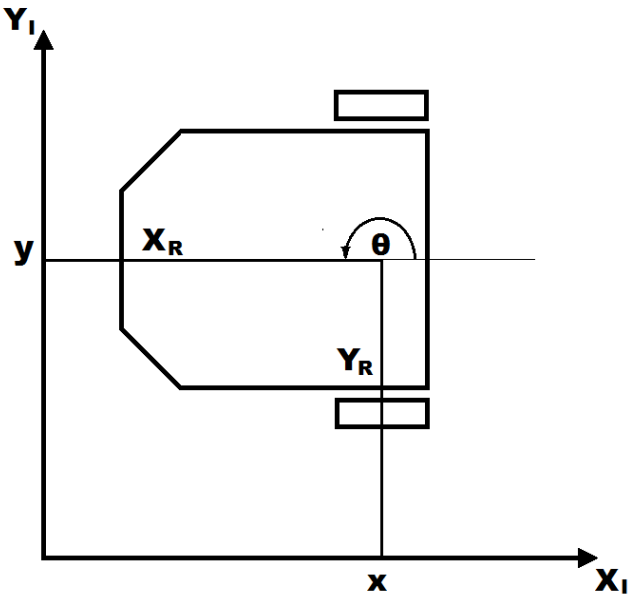
$$\xi_R = \begin{bmatrix} x\cos(\theta) + y\sin(\theta) \\ -x\sin(\theta) + y\cos(\theta) \\ \theta \end{bmatrix} = \begin{bmatrix} Y_R \\ X_R \\ \theta \end{bmatrix}. \tag{4.5}$$

Para efeito de exemplo considerando um robô com deslocamento angular de 180^o, tem-se que:

$$\xi_R = \begin{bmatrix} x\cos(180^o) + y\sin(180^o) \\ -x\sin(180^o) + y\cos(180^o) \\ (180^o) \end{bmatrix} = \begin{bmatrix} -x \\ -y \\ (180^o) \end{bmatrix}. \tag{4.6}$$

Dessa forma, o robô está alinhado com os eixos do plano de referência global como mostra a Figura 4.11.

Figura 4.11 – Robô com deslocamento angular de 180^o alinhado com os eixos do plano de referência global (plano horizontal (X_I,Y_I).



Fonte: Próprio Autor

As equações apresentadas até o momento calculam a posição de um robô móvel no espaço, porém um robô em movimento precisa levar em conta, também, a velocidade que descreve para assim obter sua cinemática direta 4.7 (MELO, 2007) como é evidenciado por

$$\dot{\xi}_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}. \tag{4.7}$$

Assim, segundo Rodrigues (2014), utilizando das deduções já apresentadas, é possível determinar a posição do robô no plano global como se segue:

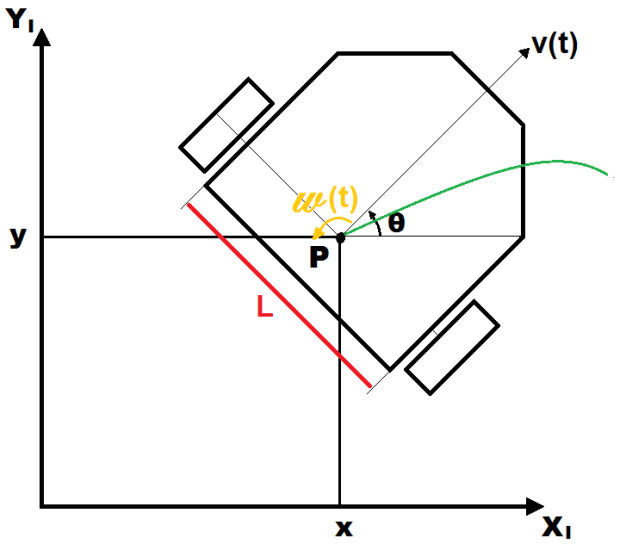
$$\dot{\xi_R} = \mathbf{R}(\theta)\dot{\xi_I}, \tag{4.8}$$

que rearranjando fica:

$$\dot{\xi_I} = \mathbf{R}(\theta)^{-1}\dot{\xi_R}. \tag{4.9}$$

O robô em questão para este trabalho, como já mencionado apresenta tração diferencial por duas rodas traseiras (Figura 4.12) e uma roda castor livre, dianteira, que serve como apoio. Para se calcular a velocidade do robô no plano global é necessário o conhecimento do diâmetro das rodas de tração, d, a distância entre as rodas de tração, L, o deslocamento angular, θ, do módulo e as velocidades de rotação de cada roda, $\dot{\varphi_1}$ e $\dot{\varphi_2}$, envolvendo esses valores consegue-se determinar a função que descreve a velocidade de cada roda e o deslocamento do módulo.

Figura 4.12 – Representação de sistema robótico que será utilizado nessa pesquisa, robô com tração diferencial por duas rodas traseiras e uma roda castor livre, dianteira, que serve como apoio, sobre o plano de referência global.



Fonte: Próprio Autor

Considerando o plano de referência local evidenciado na Figura 4.9, se o robô estiver se movendo em linha ao longo do eixo X_R, as velocidades das duas rodas, direita (Vd) e esquerda (Ve), são dadas por:

$$Vd = \dot{x}_{R1} = \frac{d}{2}\varphi_1 \tag{4.10}$$

e

$$Ve = \dot{x}_{R2} = \frac{d}{2}\varphi_2. \tag{4.11}$$

Dessa maneira, considerando que não há desvios laterais, basta somar o movimento de cada roda para se obter o posicionamento total do robô móvel, bem como sua velocidade.

Para se calcular o componente rotacional $\dot{\theta_R}$ de $\dot{\xi_R}$, as contribuições de cada roda podem ser armazenadas e depois somadas separadamente. Segundo Melo (2007), essa abordagem de cinemática pode fornecer informações sobre o movimento do robô, dadas as velocidades individuais das rodas, no caso de deslocamentos frontais.

4.2.2 Modelo cinemático através do CCI

Levando em conta um robô com rodas, este dependerá do atrito com o solo para se mover. Assim, se a roda está livre para se movimentar sobre um eixo, o movimento se dará sobre o eixo perpendicular a este podendo causar escorregamentos laterais, o que não acontece num sistema ideal.

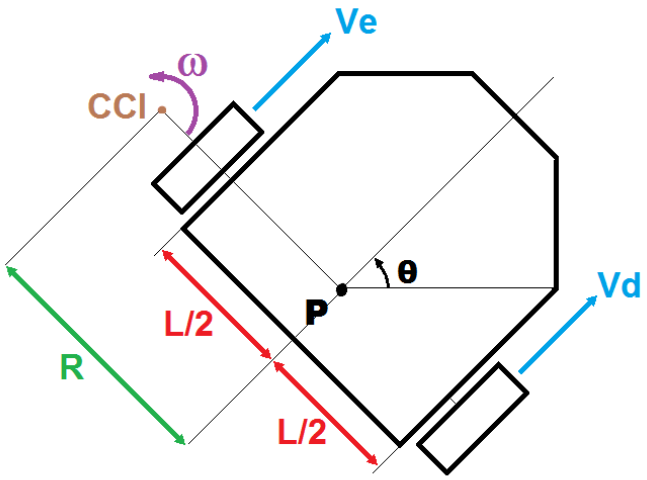
A forma mais comum de se obter o deslocamento linear de um sistema é a odometria, ou seja, a medição de quantidades de giros que uma roda executa durante determinado tempo. Atrelando esse valor a sua circunferência ($2.\pi$.r) o deslocamento é facilmente obtido, em teoria. Porém, em ambientes reais, o atrito da roda com o solo, a inclinação do percurso e outros obstáculos podem modificar a contagem de pulso e também o deslocamento calculado. Para resolver esse problema, muitas vezes utiliza-se uma roda livre sem carga ou força para estimativa de deslocamento executado.

Para que o robô execute uma curva precisa, esse deve rodar em torno de um ponto conhecido como centro de curvatura instantânea (CCI), ou seja, as duas rodas (direita e esquerda) devem girar de forma que a velocidade angular de cada uma, ω, ao redor do CCI sejam iguais durante a execução da curva em torno desse ponto.

Observando Figura 4.13 e sabendo que a velocidade linear v é dada por:

$$v = \omega R, \tag{4.12}$$

Figura 4.13 – Representação gráfica de execução de curva precisa por um robô em torno do ponto conhecido como centro de curvatura instantânea para módulo com trasmissão diferencial.



Fonte: Próprio Autor

tem-se que as velocidades lineares das duas rodas são, respectivamente:

$$V_d = \omega(R + L/2) \tag{4.13}$$

e

$$V_e = \omega(R - L/2). \tag{4.14}$$

Isolando ω e igualando as equações:

$$\omega = \frac{V_d}{R + L/2} = \frac{V_e}{R - L/2},$$

$$\frac{V_d}{\frac{2R+L}{2}} = \frac{V_e}{\frac{2R-L}{2}},$$

$$\frac{V_d}{2R + L} = \frac{V_e}{2R - L},$$

$$V_d(2R - L) = V_e(2R + L),$$

$$V_d 2R - V_d L = V_e 2R + V_e L,$$

$$2R(V_d - V_e) = L(V_e + V_d),$$

finalmente obtém-se o raio de curvatura do módulo como:

$$R = \frac{L(V_e + V_d)}{2(V_d - V_e)}. \tag{4.15}$$

Para determinar a velocidade angular do robô basta rearanjar a Eq. 4.13 e a Eq. 4.14, substituindo o valor de R dado pela Eq. 4.15, como segue:

$$\frac{V_d}{\omega} = R + \frac{L}{2},$$

$$\frac{V_e}{\omega} = R - \frac{L}{2},$$

$$R = \frac{V_d}{\omega} - \frac{L}{2} = \frac{V_e}{\omega} + \frac{L}{2},$$

$$\frac{2V_d - \omega L}{2\omega} = \frac{2V_e + \omega L}{2\omega},$$

$$2V_d - \omega L = 2V_e + \omega L,$$

e

$$2(V_d - V_e) = 2\omega L.$$

Finalmente a velocidade angular do robô móvel é dada por:

$$\omega = \frac{V_d - V_e}{L}. \tag{4.16}$$

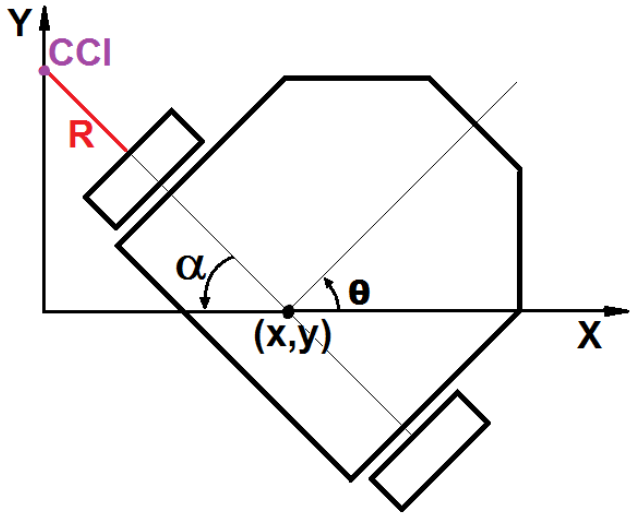
Analisando a Eq. 4.15 e a Eq. 4.16, tem-se que se as duas velocidades forem iguais, ou seja: $V_d = V_e$, o raio de curvatura do robô R será infinito, ou seja, o sistema andará em linha reta. Já se uma velocidade for oposta a outra, ou seja, $V_d = -V_e$ ou $V_e = -V_d$ o raio R será 0, o que significa que o sistema estará rodando sobre o ponto central, sem deslocamento linear. Qualquer outro caso destinará em deslocamento linear com rotação R.

4.2.3 Cinemática direta para transmissão diferencial

Segundo Melo (2007), a cinemática direta é conhecida como, a possibilidade de se estabelecer os posicionamentos possíveis de um robô, dadas as variáveis de controle do mesmo.

Para um módulo que se encontra em um ponto conhecido (x, y) e tem sua linha central direcionada a um ângulo θ do eixo x do plano, como mostra a Figura 4.14, basta a manipulação de V_e e V_d, que já foram equacionados, para se obter o posicionamento do robô.

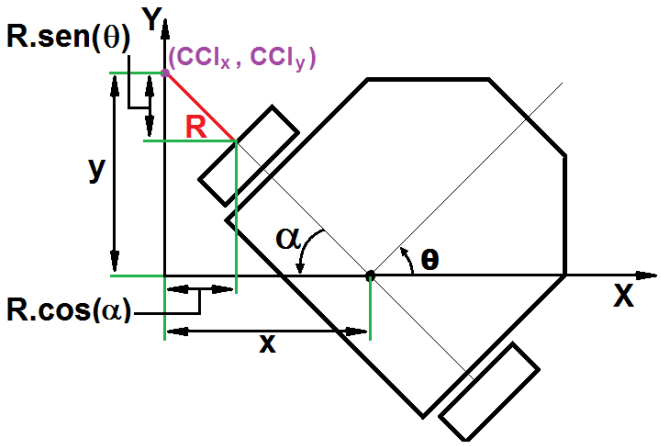
Figura 4.14 – Cinemática direta para o módulo que se encontra em um ponto conhecido (x, y) e tem sua linha central direcionada a um ângulo θ do eixo x do plano e sua geometria em relação ao CCI.



Fonte: Próprio Autor

A partir da Figura 4.14 e das equações trigonométricas básicas, obtém-se as distâncias de CCI_x e CCI_y, como mostra a Figura 4.15.

Figura 4.15 – Obtenção de CCI_x e CCI_y para o módulo robótico estabelecido nessa pesquisa com cinemática direta.



Fonte: Próprio Autor

Ainda tendo em vista a Figura 4.15 e as funções trigonométricas conhecidas, tem-se:

$$\alpha = \frac{\pi}{2} - \theta, \tag{4.17}$$

$$CCI_x = x - R\cos(\alpha) \tag{4.18}$$

e

$$CCI_y = y - R\sin(\alpha). \tag{4.19}$$

Substuindo o valor de α fornecido pela Eq. 4.17, tem-se que:

$$CCI_x = x - R\cos(\frac{\pi}{2} - \theta) \tag{4.20}$$

e

$$CCI_y = y - R\sin(\frac{\pi}{2} - \theta), \tag{4.21}$$

ou ainda:

$$CCI_x = x - R\sin(\theta) \tag{4.22}$$

e

$$CCI_y = y + R\cos(\theta). \tag{4.23}$$

Dessa forma, através de simples manipulações, num instante de tempo t onde o robô é referenciado pelo ponto P(x,y,θ), tem-se que o raio de curvatura R do módulo é dado por:

$$R = \frac{x - CCI_x}{\sin(\theta)} \tag{4.24}$$

e

$$R = -\frac{y - CCI_y}{\cos(\theta)}. \tag{4.25}$$

Analisando agora o posicionamento através do tempo de um robô traçando uma circunferência, supõe-se um instante de tempo $t = t + \delta t$. Para esse instante, tem-se que o ponto de referência do robô é dado por P(x', y', θ'). Sabe-se também que R e CCI permanecem constantes e que o deslocamento angular sofrido é de $\omega\delta t$. Dessa forma:

$$CCI_x = x' - R.\sin(\theta'), \tag{4.26}$$

$$CCI_y = y' + R.\cos(\theta'), \tag{4.27}$$

sendo

$$\theta' = \theta + \omega\delta t. \tag{4.28}$$

Substituindo (4.28) nas equações (4.26) e (4.27), tem-se:

$$CCI_x = x' - R.\sin(\theta + \omega\delta t) \tag{4.29}$$

e

$$CCI_y = y' + R.\cos(\theta + \omega\delta t). \tag{4.30}$$

Substituindo agora as equações (4.24) e (4.25) em (4.29) e (4.30), tem-se que:

$$CCI_x = x' - (\frac{x - CCI_x}{\sin(\theta)})\sin(\theta + \omega\delta t) \tag{4.31}$$

e

$$CCI_y = y' + (-\frac{y - CCI_y}{\cos(\theta)})\cos(\theta + \omega\delta t). \tag{4.32}$$

Isolando x'e y':

$$x' = CCI_x + (\frac{x - CCI_x}{\sin(\theta)})\sin(\theta + \omega\delta t) \tag{4.33}$$

e

$$y' = CCI_y - (-\frac{y - CCI_y}{\cos(\theta)})\cos(\theta + \omega\delta t). \tag{4.34}$$

Tendo conhecimento das regras de soma de seno e cosseno, tem-se que:

$$x' = (x - CCI_x)\cos(\omega\delta t) + (x - CCI_x)\frac{\cos(\theta)}{\sin(\theta)}\sin(\omega\delta t) + CCI_x \tag{4.35}$$

e

$$y' = (y - CCI_y)\cos(\omega\delta t) - (y - CCI_y)\frac{\sin(\theta)}{\cos(\theta)}\sin(\omega\delta t) + CCI_y. \tag{4.36}$$

A partir das relações fornecidas pelas equações (4.24) e (4.25), que dão:

$$R = \frac{x - CCI_x}{\sin(\theta)} = -\frac{y - CCI_y}{\cos(\theta)}, \tag{4.37}$$

tem-se que:

$$\frac{\sin(\theta)}{\cos(\theta)} = -\frac{x - CCI_x}{y - CCI_y}. \tag{4.38}$$

Substituindo a (4.38) em (4.35) e (4.36), tem-se que:

$$x' = (x - CCI_x)\cos(\omega\delta t) - (y - CCI_y)\sin(\omega\delta t) + CCI_x \tag{4.39}$$

e

$$y' = (x - CCI_x)\sin(\omega\delta t) + (y - CCI_y)\cos(\omega\delta t) + CCI_y, \tag{4.40}$$

demostrando de forma matricial que:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos(\omega\delta t) & -\sin(\omega\delta t) & 0 \\ \sin(\omega\delta t) & \cos(\omega\delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - CCI_x \\ y - CCI_y \\ \theta \end{bmatrix} + \begin{bmatrix} CCI_x \\ CCI_y \\ \omega\delta t \end{bmatrix}. \tag{4.41}$$

Finalmente, se o robô está na posição (x,y,θ) em um instante de tempo, t, em um instante posterior $t = \delta t$ ele estará na posição representada por (4.41) (DUDEK; JENKIN, 2000).

Integrando 4.41 a partir de suas condições iniciais é possível determinar onde o robô estará em qualquer intervalo de tempo levando em conta suas velocidades das rodas de tração. Para o sistema proposto:

$$x(t) = \frac{1}{2}\int_0^t [V_d(t) + V_e(t)]\cos[\theta(t)]dt, \tag{4.42}$$

$$y(t) = \frac{1}{2}\int_0^t [V_d(t) + V_e(t)]\sin[\theta(t)]dt \tag{4.43}$$

e

$$\theta = \frac{1}{L}\int_0^t [V_d(t) - V_e(t)]dt. \tag{4.44}$$

As equações 4.42, 4.43 e 4.44 apresentam restrições quanto a determinados posicionamentos, as restrições não-holonômicas.

Assumindo então que $v_e(t) = v_e$, $v_d(t) = v_d$ e $v_d \neq v_e$, tem-se que:

$$x(t) = \frac{L}{2}\frac{v_d + v_e}{v_d - v_e}\sin[\frac{t}{L}(v_d - v_e)], \tag{4.45}$$

$$y(t) = -\frac{L}{2}\frac{v_d + v_e}{v_d - v_e}\cos[\frac{t}{L}(v_d - v_e)]], \tag{4.46}$$

e

$$\theta(t) = \frac{t}{L}(v_d - v_e), \tag{4.47}$$

onde x, y e θ, com intervalo de tempo tendendo a zero, são iguais a zero.

Substituindo 4.15 e 4.16 em 4.45, 4.46 e 4.47, tem-se as seguintes equações em função do raio de curvatura:

$$x(t) = R\sin(\omega t), \tag{4.48}$$

$$y(t) = R[1 - \cos(\omega t)] \tag{4.49}$$

e

$$\theta(t) = \omega t. \tag{4.50}$$

Levando em conta dois casos específicos, o primeiro com $v_d = v_e = v$ e raio de curvatura tendendo ao infinito, ou seja, robô se movendo em linha reta 4.42, 4.43 e 4.44 simplifica-se para:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + v.\cos((\theta)\delta t) \\ y + v.\sin((\theta)\delta t) \\ \theta \end{bmatrix}, \tag{4.51}$$

já para um segundo caso onde $-v_e = v_d = v$, tem-se que:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta + 2v.\delta t/L \end{bmatrix}. \tag{4.52}$$

4.3 Modelagem Dinâmica para Robô Móvel

O modelo matemático de um sistema é capaz de representar com precisão a dinâmica do mesmo. Vale lembrar que um mesmo sistema pode ser representado de diversas maneiras, dependendo da forma que irá atuar (OGATA, 2003).

O comportamento dinâmico é descrito, na maioria dos casos, por equações diferenciais (DORF; BISHOP, 2001), o que não será diferente neste trabalho. Serão consideradas aproximações lineares e, a partir das leis da física, equações diferenciais capazes de descrever o protótipo robótico obtido para essa pesquisa, terá seu modelo dinâmico descrito.

O modelo citado não será utilizado na prática pelo módulo e por nenhum *software* envolvido com o mesmo, o que não diminui sua importância. O estudo a ser realizado é importante na verificação de estabilidade do sistema proposto e pode ser aplicado a outros robôs em trabalhos futuros que envolvam o mesmo princípio de funcionamento.

4.3.1 Modelagem no Espaço de Estados

Com o aumento da complexidade dos sistemas de engenharia, sistemas com múltiplas entradas e saídas variantes no tempo passaram a ser melhorados e, para estes, um novo modelo de representação matemática se fez necessário, aí que surge a modelagem no espaço de estados (OGATA, 2003).

Para se representar um sistema dinâmico em espaço de estados, é necessário que alguns conceitos como variáveis de estado, estado, vetor de estado, espaço de estados e equações no espaço de estados sejam conhecidos. OGATA (2003) os define da seguinte forma:

- **VARIÁVEIS DE ESTADO:** São capazes de determinar o estado do sistema dinâmico;
- **ESTADO:** Determina o comportamento do sistema em um instante de tempo t$\geq$0 desde que os valores da entrada nesse mesmo instante de tempo e os valores das variáveis de estado em $t = 0$ sejam conhecidas;
- **VETOR DE ESTADO:** É o vetor que determina o estado de um sistema dinâmico. Se o sistema possui n variáveis de estado, esse vetor terá n componentes;
- **ESPAÇO DE ESTADOS:** É o espaço que tem como eixos coordenados, os eixos das variáveis de estado. Assim, se o sistema possui n variáveis de estado, o espaço de estados será n-dimensional;
- **EQUAÇÕES NO ESPAÇO DE ESTADOS:** Envolvem três tipos de variáveis, as de entrada u(t), as de saída y(t) e as de estado x(t) e são conhecidas como equação de estado e equação de saída. As mesmas, respectivamente, podem ser vistas a seguir.

$$\dot{x}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \tag{4.53}$$

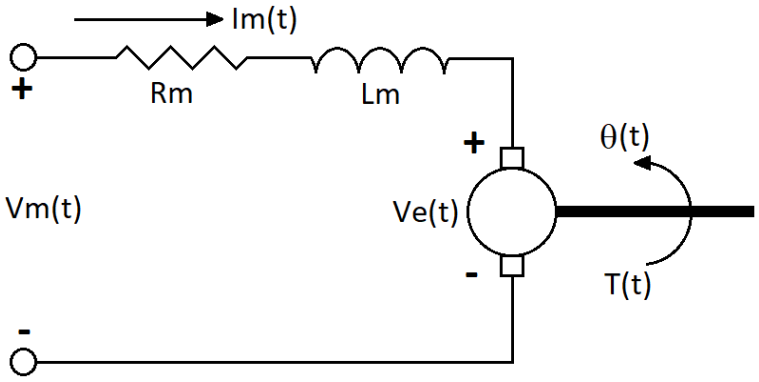
$$y(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t), \tag{4.54}$$

onde **A**(t) é chamada de matriz de estado, **B**(t) de matriz de entrada, **C**(t) de matriz de saída e **D**(t) de matriz de transmissão direta.

4.3.2 Representação em Espaço de Estados

O sistema dinâmico em estudo possui como atuadores motores de corrente contínua e o estudo desses é fundamental para a elaboração das equações do espaço de estados.

Figura 4.16 – Representação do sistema elétrico de um motor CC



Fonte: (SA,)

A Figura 4.16 exemplifica o esquema elétrico do motor CC utilizado nessa pesquisa. Seus parâmetros são descritos como:

- V_m(t): Tensão aplicada aos terminais do motor;
- R_m: Resistência de armadura do motor;
- L_m: Indutância de armadura do motor;
- I_m(t): Corrente de armadura do motor;
- V_e(t): Tensão contra eletromotriz aplicada;
- $\theta(t)$: Deslocamento angular sofrido pelo eixo do motor;
- T(t): Torque exercido sobre o eixo do motor.

Aplicando a lei de Kirchhoff sobre o circuito em questão, tem-se:

$$V_m(t) = I_m(t)R_m(t) + V_{Lm}(t) + V_e(t). \tag{4.55}$$

Sabendo que

$$V_{Lm} = L_m\dot{I}_m(t)$$

e

$$V_e(t) = K_m\dot{\theta}(t),$$

onde K_m é a constante da força eletromotriz aplicada ao motor. Dessa forma, tem-se:

$$V_m(t) = I_m(t)R_m(t) + L_m\dot{I}_m(t) + K_m\dot{\theta}(t). \tag{4.56}$$

A tensão aplicada aos terminais do motor $V_m(t)$ será determinada por modulação de largura de pulso PWM (*Pulse Width Modulation*), dessa forma essa tensão pode ser representada como:

$$V_m(t) = V_{\text{PWM}}(t),$$

onde $V_{\text{PWM}}(t)$ corresponde à tensão aplicada a cada PWM. Assim, a (4.56) pode ser reescrita da seguite forma:

$$V_{\text{PWM}}(t) = I_m(t)R_m(t) + L_m\dot{I}_m(t) + K_m\dot{\theta}(t). \tag{4.57}$$

Tendo equacionado o motor de atuação a partir de sua análise elétrica, é necessária uma análise que leve em consideração sua estrutura e comportamento mecânicos, o que será feito a partir daqui.

Segundo Dorf e Bishop (2001) o torque do motor, $T(m)$, está relacionado ao fluxo de campo no entreferro, (Φ), e a corrente de armadura, $I_m(t)$, da seguinte forma:

$$T(t) = K_L K_f I_m(t) I_f(t), \tag{4.58}$$

onde K_f corresponde à constante da corrente de campo no fluxo (Φ), $I_f(t)$ corresponde à corrente de campo e K_L corresponde à constante da corrente de armadura.

Como o sistema é linear uma das correntes permanece constante (DORF; BISHOP, 2001). Levando em conta que o motor é controlado pela armadura, ou seja, I_f(t) é constante, tem-se que (4.58) pode ser escrita da seguinte maneira:

$$T(t) = K_L K_f I_m(t) I_f = K_t I_m(t), \tag{4.59}$$

onde K_t corresponde a constante de torque do motor.

Sabe-se, também, que o torque do motor nada mais é que a somatória do torque devido a carga aplicada, $T_L(t)$, com o torque devido à perturbações, $T_d(t)$. Tendo que este último é quase sempre desprezível (DORF; BISHOP, 2001), tem-se:

$$T(t) = T_L(t)$$

e, para inércias rotativas

$$T(t) = J_L \ddot{\theta}(t) + b.\omega(t), \tag{4.60}$$

onde J_L corresponde ao momento de inércia do motor, ω(t) corresponde a velocidade angular do motor e b corresponde ao atrito.

Igualando (4.59) e (4.60), tem-se:

$$J_L \ddot{\theta}(t) + b\omega(t) = K_t I_m(t). \tag{4.61}$$

Tendo o sistema sido descrito elétrica e mecanicamente, seu equacionamento em espaço de estados pode ser finalmente obtido.

Inicialmente as variáveis de estado de interesse devem ser definidas e, para o robô, levando em conta que seu processo de atuação é dado por motores CC e que o objetivo é o estudo de seu posicionamento e deslocamento no espaço, as variáveis de estado foram definidas como sendo:

- Corrente de armadura: $I_m(t)$;
- Deslocamento angular do eixo: $\theta(t)$;
- Velocidade angular do eixo: $\omega(t)$.

Como já explicado no início dessa seção, para se trabalhar em espaço de estados é preciso conhecer as equações diferenciais das variáveis de estado, no caso desse estudo:

- $\dot{I}_m$(t);
- $\dot{\theta}$(t);
- $\dot{\omega}$(t).

Também é conhecido, a partir das leis da física, que o valor da velocidade angular é o mesmo que a função diferencial do deslocamento angular de um corpo, ou seja:

$$\dot{\theta}(t) = \omega(t)$$

e

$$\ddot{\theta}(t) = \dot{\omega}(t).$$

Dessa forma, levando as igualdades apresentadas em consideração e reorganizando as (4.57) e (4.58), chega-se às equações diferenciais das variáveis de estado definidas para o sistema como sendo:

$$\dot{I}_m(t) = \frac{K_{\text{ADC}} V_{\text{fonte}} - I_m(t) R_m - K_m \dot{\theta}(t)}{L_m}, \tag{4.62}$$

$$\dot{\theta}(t) = \omega(t) \tag{4.63}$$

e

$$\dot{\omega}(t) = \ddot{\theta}(t) = \frac{K_t I_m(t) - b\omega(t)}{J_L}. \tag{4.64}$$

As equações (4.51) e (4.52) mostram como são formadas as equações no espaço de estado. Partindo da equação de estado

$$\dot{x}(t) = \mathbf{A}x(t) + \mathbf{B}u(t)$$

e sabendo que de $x(t)$ representa o vetor das variáveis de estado e $u(t)$ as entradas aplicadas, que no caso do sistema aqui estudado é a tensão da fonte V_{fonte}, a partir da análise das equações (4.62), (4.63) e (4.64), os parâmetros são obtidos quando isoladas as variáveis.

A partir da análise das equações, tem-se que:

$$\dot{x}(t) = \begin{bmatrix} \dot{I}_m(t) \\ \dot{\theta}(t) \\ \dot{\omega}(t) \end{bmatrix}, \tag{4.65}$$

$$\mathbf{A} = \begin{bmatrix} -\frac{R_m}{L_m} & 0 & -\frac{K_m}{L_m} \\ 0 & 0 & 1 \\ \frac{K_m}{J_L} & 0 & -\frac{b}{J_L} \end{bmatrix}, \tag{4.66}$$

$$x(t) = \begin{bmatrix} I_m(t \\ \theta(t) \\ \omega(t) \end{bmatrix}, \tag{4.67}$$

$$\mathbf{B} = \begin{bmatrix} \frac{K_{\text{ADC}}}{L_m} \\ 0 \\ 0 \end{bmatrix} \tag{4.68}$$

e

$$u(t) = \begin{bmatrix} V_{\text{PWM}}(t) \end{bmatrix}. \tag{4.69}$$

Portanto, a equação de estado do sistema robótico é:

$$\begin{bmatrix} \dot{I}_m(t) \\ \dot{\theta}(t) \\ \dot{\omega}(t) \end{bmatrix} = \begin{bmatrix} -\frac{R_m}{L_m} & 0 & -\frac{K_m}{L_m} \\ 0 & 0 & 1 \\ \frac{K_t}{J_L} & 0 & -\frac{b}{J_L} \end{bmatrix} \begin{bmatrix} I_m(t) \\ \theta(t) \\ \omega(t) \end{bmatrix} + \begin{bmatrix} \frac{K_{\text{ADC}}}{L_m} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} V_{\text{PWM}}(t) \end{bmatrix}. \tag{4.70}$$

Passando agora para o estudo da equação de saída:

$$y(t) = \mathbf{C}x(t) + \mathbf{D}u(t) \tag{4.71}$$

tem-se que $y(t)$ representa a saída do sistema que são representadas pelos componentes responsáveis por calcular a velocidade angular $\omega(t)$ do motor (um encoder) e o deslocamento angular de seu eixo $\theta(t)$ (um redutor).

Observa-se que para a saída desse sistema a corrente de armadura é irrelevante, portanto, todos os parâmetros relacionados a ela serão nulos.

Para efeito de equacionamento, o redutor e o encoder, responsáveis por fornecerem a saída do sistema, serão identificados aqui por K_{RED} e K_{ENC} respectivamente, e tem seus valores dependentes da configuração de cada um. Esses valores serão evidenciados ainda nesse documento.

Finalmente, tem-se que as equações de saída do sistema são dadas por:

$$y_2(t) = K_{\text{ENC}}\theta(t) \tag{4.72}$$

e

$$y_3(t) = K_{\text{RED}}\omega(t). \tag{4.73}$$

Com uma análise simples das equações apresentadas obtém-se os elementos da equação de saída do sistema robótico, que são descritos por:

$$y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{bmatrix}, \tag{4.74}$$

$$\mathbf{C} = \begin{bmatrix} 0 & K_{\text{ENC}} & 0 \\ 0 & 0 & K_{\text{RED}} \end{bmatrix} \tag{4.75}$$

e

$$\mathbf{D} = \begin{bmatrix} 0 \end{bmatrix}. \tag{4.76}$$

Finalmente, através dos parâmetros descritos acima, tem-se que a equação de saída é descrita por:

$$\begin{bmatrix} y_2(t) \\ y_3(t) \end{bmatrix} = \begin{bmatrix} 0 & K_{\text{ENC}} & 0 \\ 0 & 0 & K_{\text{RED}} \end{bmatrix} \begin{bmatrix} I_m(t) \\ \omega(t) \\ \theta(t) \end{bmatrix}. \tag{4.77}$$

4.4 Conclusão do capítulo

O conhecimento do modelo matemático do módulo em utilização em uma pesquisa é de fundamental importância para realização de simulações, testes virtuais e validação de resultados e, tendo essa importância em mente, o estudo da modelagem do robô proposto na pesquisa foi executado no presente capítulo, fornecendo ao leitor suas equações de movimento e também seu equacionamento em espaço de estados.

Tendo os equacionamentos em evidência é importante, também, que os componentes elétricos presentes no módulo sejam estudados, a fim de se obter uma correta aplicação do equacionamento obtido.

Dessa maneira, o próximo Capítulo traz a descrição dos principais componentes que compõem o módulo aqui em estudo, bem como suas principais características.

5 Descrição da Plataforma Robótica

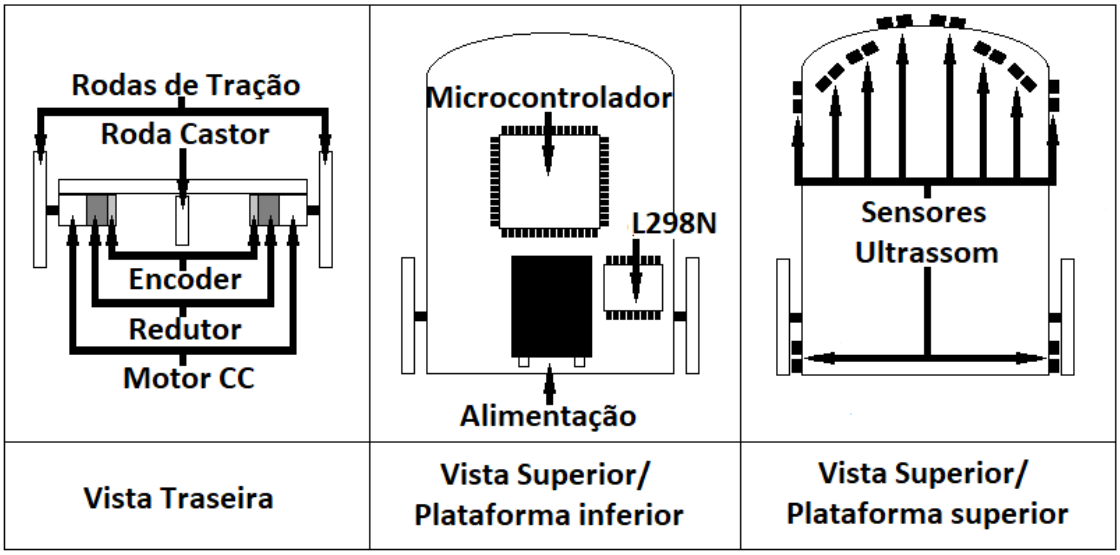
Um sistema de controle é um conjunto formado por um sistema a ser controlado e seu controlador (UFRN, 2003), como o robô desenvolvido.

O sistema robótico criado dispõe de um microcontrolado que determina as ações a serem tomadas, um sistema a ser controlado (robô), atuadores que executam as ações determinadas pelo microcontrolador e sensores que colhem resultados referentes as ações tomadas, a fim de auxiliar em correções de controle.

Todos esses componentes, bem como os *softwares* utilizados para controlá-los ou testá-los, terão suas características fundamentais evidenciadas no presente Capítulo.

Um esquema do robô construído e suas principais partes é mostrado na Figura 5.1.

Figura 5.1 – Esquema gráfico do módulo robótico desenvolvido em sua vista traseira, superior inferior e superior superior e alocação de seus principais componentes.



Fonte:Próprio Autor

O protótipo foi constituído com dois motores CC com redutores acoplados, dois encoders responsáveis por medir a quantidade de giros dados pelos eixos dos motores CC, um driver de potência L298N responsável por fornecer a potência estipulada pelo controle para cada motor CC, um circuito eletrônico com microcontrolador ATmega2560 que recebe informações dos sensores e toma decisões que são passadas aos atuadores do módulo, 6 sensores de ultrassom que medem a distância do módulo até um obstáculo.

Além das partes físicas, para que um sistema seja executado em ambiente real é de grande importância que o mesmo seja simulado em ambiente virtual. Dessa forma, possíveis erros podem ser encontrados antes da aplicação prática e corrigidos, evitando acidentes ou comportamentos inesperados.

Nesse Capítulo, tanto os materiais utilizados na elaboração do robô físico como controladores, sensores e atuadores quanto *softwares* aplicados em simulações, compilação e testes serão apresentados e suas características fundamentais evidenciadas.

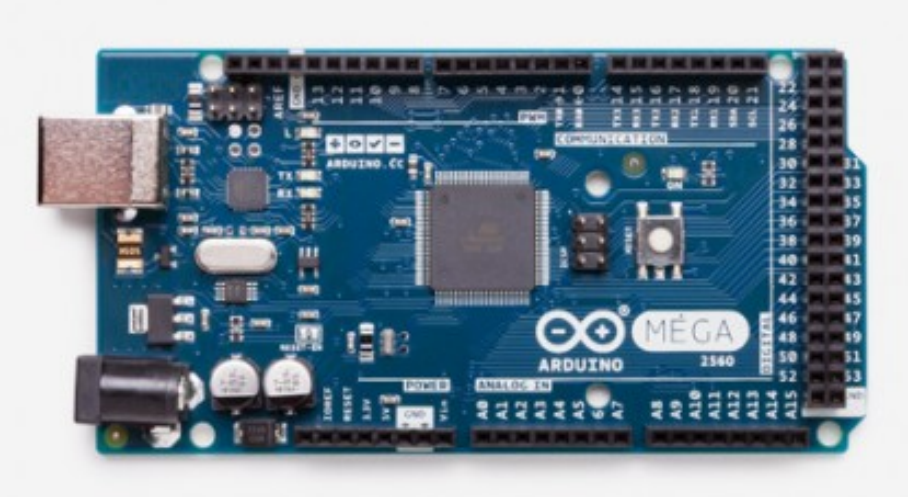
5.1 Controladores robóticos

Um controlador é um dispositivo que controla uma determinada ação. Nesse trabalho, o controlador utilizado é um microcontrolador que pode ser programado para funções específicas. Em geral os controladores são usados para controlar circuitos e, por isso, são comumente encontrados dentro de outros dispositivos, sendo conhecidos como "controladores embarcados". A estrutura interna de um microcontrolador apresenta um processador, bem como circuitos de memória, periféricos de entrada e saída, entre outros (ROBOLIVRE, 2003).

O micro controlador utilizado nesse trabalho é o ATmega2560, apresentado na próxima seção.

5.1.1 Microcontrolador Atmega2560

Figura 5.2 – Placa de prototipagem do micro controlador ATmega2560 utilizada no protótipo.



Fonte:(ARDUINO, 2017)

O controle do sistema robótico é executado pela placa de prototipagem Arduino MEGA, que possui um microcontrolador ATmega2560, que é visto na Figura 5.2.

A placa é destinada a projetos mais complexos como os que envolvem robótica devido a sua capacidade bem mais elevada comparada a outras placas da marca (ARDUINO, 2017).

O Arduino Mega 2560 é baseado no microcontrolador ATmega2560. Possui 54 pinos de entrada/saída digitais dos quais 15 podem ser utilizados como saídas PWM, 16 entradas analógicas, 4 UARTs (*Universal Asynchronous Receiver/Transmitter* ou Transmissor/Receptor Universal Assíncrono) (portas seriais de hardware), um oscilador de cristal de 16 MHz, uma

conexão USB (*Universal Serial Bus*), uma tomada de força, ICSP (*In-Circuit Serial Programming*), e um botão de reinicialização (ARDUINO, 2017). O diagrama referente à pinagem da placa, bem como a tabela mapeando cada entrada com seu pino encontram-se, respectivamente, nos Anexos A e B no final deste trabalho.

O microcontrolador ATmega2560 vem pré-programado com um carregador de inicialização que permite o carregamento de códigos sem uso de *hardware* externo. A comunicação desses se dá através do protocolo original STK500 (referência, arquivos em linguagem C). Possui 256 kB de memória flash para armazenamento de código (8 kB destinados ao carregador de inicialização), 8 kB de SRAM (*Static Random Access Memory* ou Memória Estática de Acesso Aleatório) e 4 kB de EEPROM (*Electrically-Erasebla Programmable Read-Only Memory* ou Memória Somente de Leitura Programável Apagável Eletricamente). (ARDUINO, 2017).

O Arduino Mega 2560 também possui proteção para portas USB de computadores. Embora a maioria dos computadores forneça sua própria proteção interna, um fusível fornece uma camada extra de proteção de forma que, se for aplicado mais de 500 mA à porta USB, automaticamente a conexão é interrompida até que o curto ou a sobrecarga seja removido (ARDUINO, 2017).

A placa pode ser alimentada através da conexão USB ou por uma fonte de alimentação externa. A fonte de energia é selecionada automaticamente. A faixa de alimentação é de 6 a 20 V, porém, se for fornecido menos de 7 V o pino de 5 V pode fornecer menos tensão que a nominal e a placa pode se comporta de maneira instável. Já se mais de 12V forem aplicados, o regulador de tensão pode superaquecer e danificar a placa. Dessa maneira, o intervalo de tensão a ser aplicada na alimentação da placa, pelo fabricante, é de 7 a 12 V.

As principais características da placa Arduino Mega 2560 podem ser vistas na Tabela 5.1 e seu diagrama esquemático pode ser encontrado no Anexo C ao fim deste trabalho.

Tabela 5.1 – Características gerais da Placa de prototipagem eletrônica Arduino Mega 2560, que possui como microcontrolador o Atmega2560.

CARACTERÍSTICA	DESCRIÇÃO
Microcontrolador	ATmega2560
Tensão de Operação	5 Volts
Tensão de Alimentação	6 a 20 Volts
Tensão de Alimentação (recomendada)	7 a 12 Volts
Pinos Digitais	54 (15 PWM)
Pinos Analógicos	16
Corrente por I/O	20 mA
Memória Flash	256 KB
SRAM	8 KB
EEPROM	4 KB
Clock	16MHz

Fonte:(ARDUINO, 2017)

5.2 Atuadores robóticos

Um atuador é que dispositivo capaz de transformar um tipo de grandeza física em outra (GROOVER, 2011), por exemplo, transformar energia elétrica em energia mecânica, como ocorre em um motor elétrico.

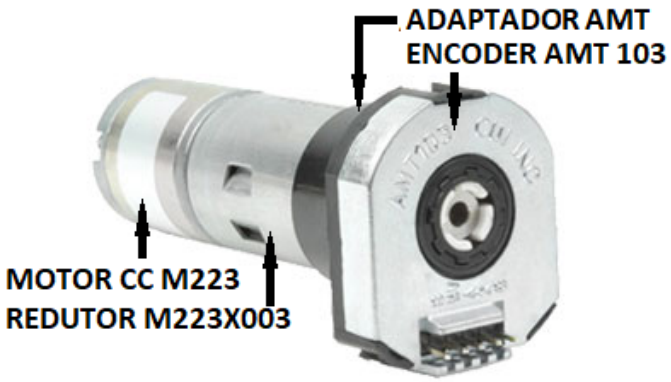
No módulo robótico construído nesse trabalho, dois motores CC são utilizados para dar movimento ao protótipo e são eles os atuadores do sistema.

5.2.1 Motor CC

O motor CC utilizado neste trabalho é o M223 do kit AMT103 da empresa CUI INC.

O kit, representado na Figura 5.3 é composto por um motor CC modelo M223 de 12 V de alimentação, um redutor de modelo M223X003 que gera uma redução na capacidade de giro do motor de 84 vezes e um encoder incremental que pode ter sua resolução ajustada.

Figura 5.3 – Kit AMT103 utilizado para locomoção e medição de deslocamento do módulo robótico desenvolvido.



Fonte: (INC, 2017)

O *datasheet* do kit se encontra no Anexo D ao fim desse trabalho e as características principais do kit, que serão relevantes para a pesquisa, estão apresentadas na Tabela 5.2.

Tabela 5.2 – Característica gerais do kit AMT103, composto por um motor CC, um encoder e um redutor.

PARÂMETRO	VARIÁVEL	VALOR	UNIDADE
Resistência de Armadura	Rm	7,2	Ω
Indutância de Armadura	Lm	3400	μH
Constante de Torque	Kt	13,15	mN(m/a)
Contante EMF	Km	1,38	mV/rpm
Momento de Inércia	Jr	4,5	g.cm^2
Velocidade Máxima	ω	9400	rpm
Constante de Redutor	K_{RED}	1:84	—
Atrito Viscoso	b	4,7989	μ Nm/(rad^2/2)
Momento de Inércia com Carga	Jl	0,009375	Kg.m^2
Atrito Viscoso com Carga	bl	0,0075	N

Fonte: (INC, 2017)

5.2.2 Redutor M223X003

Um redutor é um dispositivo que provoca a diminuição da velocidade de um motor e aumenta seu torque.

São também chamados de caixa de engrenagens pois é através de um conjunto delas que o processo de diminuição da velocidade é realizado e são, em sua maioria, mecanismos bem simples.

Um exemplo de caixa de redução acoplada a um motor CC é mostrado na Figura 5.4.

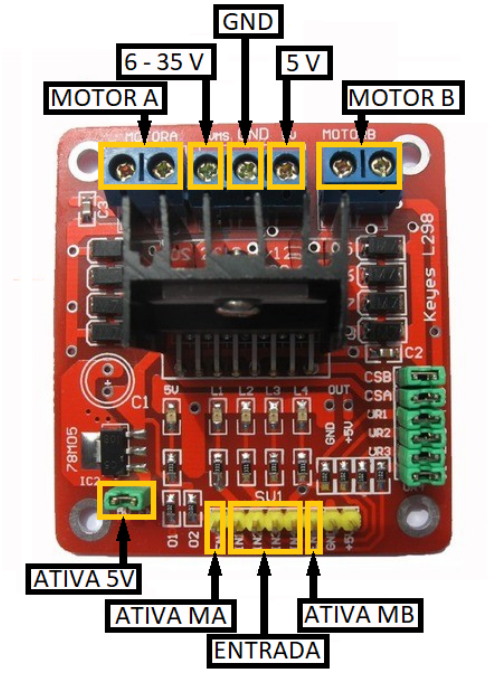
Figura 5.4 – Exemplo visual de motor CC com caixa de redução acoplada

Fonte: (ELETRÔNICA, 2009)

O redutor utilizado nesse trabalho é o do kit AMT103 da CUI INC, de modelo M223X003 e possui constante de redução de 1:84.

5.2.3 Driver Ponte H L298N

Figura 5.5 – Módulo ponte H L298N utilizado para controle de velocidade de motores CC de tração.



Fonte: (LINK, 2013)

O controle dos dois motores CC do módulo robótico são executados por um Driver Ponte H do modelo L298N, como o mostrado na Figura 5.5.

O driver possibilita o controle de até dois motores CC ou um motor de passo e é um dispositivo que permite o controle da velocidade do motor, bem como do seu sentido de rotação através de sinais de PWM (LINK, 2013), o que é o caso do projeto.

O driver tem seu funcionamento controlado por suas diversas entradas e saídas e essas são detalhadas a seguir:

- **MOTOR A:** Referem-se a dois conectores onde um motor CC deve ser ligado. Um conector para entrada de tensão e outra para terra;
- **MOTOR B:** Referem-se a dois conectores onde um segundo motor CC deve ser ligado. Um conector para entrada de tensão e outra para terra;
- **ATIVA MA:** Pino responsável pela Ativação do Motor A e por seu controle PWM;
- **ATIVA MB:** Pino responsável pela Ativação do Motor B e por seu controle PWM;
- **ATIVA 5 V:** Se o driver estiver operando com uma tensão entre 6 e 35 V, com os conectores dessa entrada em jumper, um regulador de tensão presente ao dispositivo disponi-

bilizará uma saída de 5 V no pino 5 V que pode ser utilizada para alimentação de outros componentes;

- **5 V:** Saída de tensão de 5 V quando o driver estiver operando entre 6 e 35 V e a entrada ATIVA 5 V estiver em curto. Se estiver controlando um motor de 4,5-5 V e a entrada ATIVA 5 V estiver sem jumper, pode se tornar uma entrada de alimentação, podendo assim, ter uma fonte qualquer de 5 V ligada a ele;
- **6-35V:** Entrada para conexão da fonte de alimentação externa para cntrole do motor. Nesse trabalho uma fonte de 12 V é utilizada;
- **GND:** GND para conexão da fonte de alimentação externa;
- **ENTRADA:** Barramento composto por quatro pinos (IN1, IN2, IN3 e IN4) que são resposáveis pela rotação dos motores, tanto o estado de rotação como a direção. Os pinos IN1 e IN2 referem-se ao motor A e os pinos IN3 e IN4 referem-se ao motor B. A Tabela 5.3 e a Tabela 5.4 mostram o comportamento dos motores A e B, respectivamente, para as possíveis estados desse barramento.

Tabela 5.3 – Comportamento do Motor A para possíveis entradas de seu barramento de controle

ENA	IN1	IN2	ROTAÇÃO
5V	5V	GND	HORÁRIA
5V	GND	5V	ANTI-HORÁRIA
GND	X	X	PARADA
X	5V	5V	FREIO
X	GND	GND	PONTO-MORTO

Fonte: Próprio Autor

Tabela 5.4 – Comportamento do Motor B para possíveis entradas de seu barramento de controle

ENB	IN3	IN4	ROTAÇÃO
5V	5V	GND	HORÁRIA
5V	GND	5V	ANTI-HORÁRIA
GND	X	X	PARADA
X	5V	5V	FREIO
X	GND	GND	PONTO-MORTO

Fonte: Próprio Autor

A Tabela 5.5 evidencia as principais e mais relevantes características do driver L298N.

Tabela 5.5 – Principais características do Driver Ponte H L298N utilizado para controle de velocidades dos motores CC de tração.

CARACTERÍSTICA	DESCRIÇÃO
Tensão de Operação	4-35V
Chip	ST L298N
Controle	2 motores CC ou 1 de passo
Corrente de Operação por Canal	2A
Corrente de Operação Máxima	4A
Tensão Lógica	5V
Corrente Lógica	0-36mA
Limites de Temperatura	-20 A +135^{o}C
Potência Máxima	25W

Fonte: (LINK, 2013)

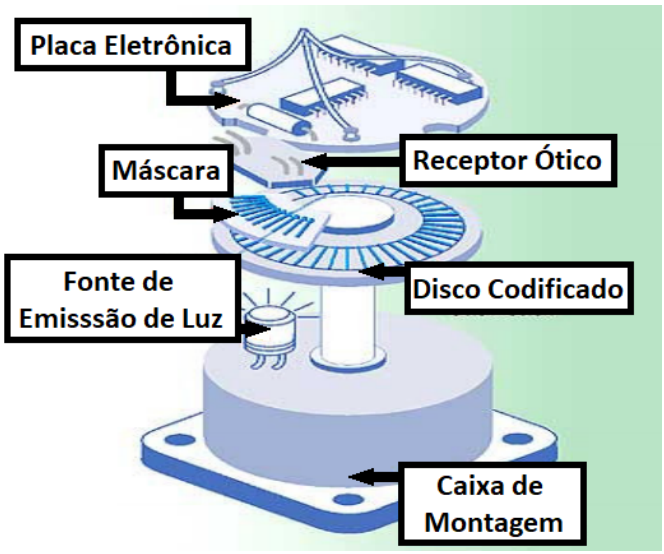
5.3 Sensores robóticos

Sensores são dispositivos capazes de estipular o valor de uma grandeza física (HWANG; KINTIGH, 1994). Eles podem ser internos ou externos com a diferença de que o primeiro fornece informação sobre algum parâmetro interno do sistema, como por exemplo nível de carga de bateria, e o segundo fornece dados externos como distância (RIBEIRO, 2004). Nessa pesquisa, os dois tipos de sensores são utilizados e os mesmos são descritos nessa seção.

5.3.1 Encoder AMT103

O sensor fundamental para execução dessa pesquisa é o encoder (Figura 5.6). Um encoder é um dispositivo eletrônico capaz de medir a quantidade de giros que o eixo de um motor dá através da transformação de energia luminosa em elétrica (MELO, 2008).

Figura 5.6 – Esquema estrutural de um encoder típico aplicado a sistemas robóticos.



Fonte: (MELO, 2008)

Dois encoders do modelo AMT103 são utilizados. Esse modelo é apresentado na Figura 5.7.

Figura 5.7 – Encoder AMT103 utilizado na obtenção do deslocamento do módulo robótico.

Fonte: AMT10 Datasheet

O funcionamento desse dispositivo consiste basicamente no envio de um feixe de luz emissor até um receptor e em um disco rígido fracionado que interrompe esse feixe. O disco é fixado a um eixo em movimento, o motor CC nesse trabalho, e conforme esse se movimenta o disco também se movimenta fazem com que o feixe de luz seja ou não interrompido. Dessa maneira, dependendo do número de partições contidas no disco, a cada volta completa do mesmo, um número de interrupções será causada no feixe de luz, tornando possível a determinação do número de voltas dadas pelo eixo e seu deslocamento linear e angular.

O sensor aqui descrito possui 16 resoluções possíveis de acordo com o ajuste de quatro chaves em sua estrutura. Essa configuração foi estabelecida como 48 PPR (Pontos por rotação), ou seja, 48 interrupções no feixe de luz a cada giro completo do disco.

5.3.2 Sensor de Distância Ultrassônico HC-SR04

Figura 5.8 – Sensor de distância HC-SR04 utilizado para detecção de obstáculos pelo módulo robótico desenvolvido.

Fonte: (SHOP, 2017)

O objetivo final dessa pesquisa é o desvio de obstáculos por parte de um robô. Para isso, alguns sensores de distância se fizeram necessário como instrumento de detecção de corpos em uma faixa fixa de distância do módulo em questão. Os sensores utilizados foram do modelo Ultrassônico HC-SR0 como o mostrado na Figura 5.8.

O HC-SR0 é composto por um circuito de controle, um transmissor e um receptor ultrassônico. Pode medir distâncias entre 2 cm e 400 cm e possui uma resolução de 3mm. Encontra objetos numa abertura de 15 graus a partir de seu transmissor, segundo o *datasheet* do dispositivo.

Seu funcionamento se dá de maneira bem simples: consiste basicamente no envio de um sinal que, ao atingir um corpo, é refletido e volta ao sensor tendo o tempo entre envio e recepção medido e utilizado para a detecção da distância entre objeto e sensor a partir da seguinte equação:

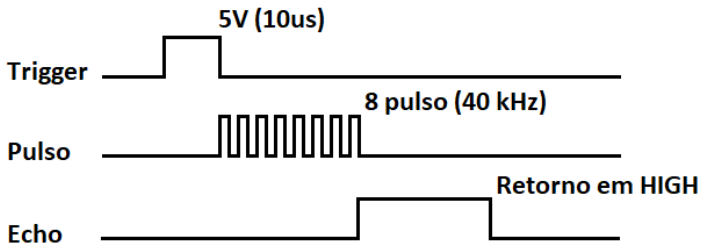
$$D = \frac{\Delta . v_{\text{som}}}{2} \tag{5.1}$$

onde:

- D: é a distância medida entre sensor e objeto de barreira;
- Δ: é o tempo transcorrido entre envio e recebimento dos pulsos ultrassônicos;
- v_{som}: velocidade do som (aproximadamente 340 m/s @ 20^{o}C / 25^{o}C).

De maneira mais específica um sinal de 5 V com duração de, pelo menos, 10 μS é enviado ao pino de trigger indicando o início do processo de medição, o que faz com que o sensor transmita 8 pulsos a uma taxa de 40 kHz e inicie o processo de espera pelo retorno do sinal. Caso retorne, a distância entre sensor e obstáculo é então estimada. Todo esse processo é ilustrado pela Figura 5.9.

Figura 5.9 – Processo de medição do sensor HC-SR04 baseado nos sinais (inicialização, pulso enviado e detecção de pulso retornado) e suas respectivas durações.



Fonte: Próprio Autor

Todas as características do sensor HC-SR04 podem ser encontradas no Anexo G ao final desse trabalho e as fundamentais, relevantes a essa pesquisa, podem ser observadas na Tabela 5.6.

Tabela 5.6 – Principais características do Sensor HC-SR04 utilizado para detecção de obstáculos pelo módulo robótico desenvolvido.

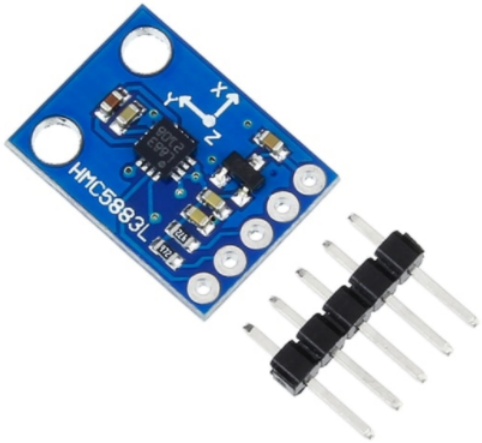
CARACTERÍSTICA	DESCRIÇÃO
Tensão de Operação	5V
Corrente quiescente	2mA
Corrente em funcionamento	15mA
Ângulo de medida	15 graus
Distância de detecção	2cm a 400cm
Resolução	3mm
Frequência ultrasônica	40kHz

Fonte: (SHOP, 2017)

5.3.3 Bússola Eletrônica HMC5883L

O módulo bússola eletrônica, ou magnetômetro, HMC5883L representado pela Figura 5.10 é um módulo que contém um sensor magnético de 3 eixos, fornecendo na saída informações sobre os eixos X, Y e Z.

Figura 5.10 – Módulo bússola eletrônica HMC5883L.



Fonte: (KURPIEL et al., 2010)

É comumente utilizado como bússola, detectando o norte magnético da Terra e possui como grande vantagem o baixo consumo de corrente em modo de medição (KURPIEL et al., 2010).

O módulo possui um chip QMC5883L, suporta calibração automática e possui conversor analógico digital de 12 bits integrado. As demais características relevantes do dispositivo podem ser encontradas na Tabela 5.7.

Tabela 5.7 – Principais características do módulo HMC5883L.

CARACTERÍSTICA	DESCRIÇÃO
Tensão de alimentação	3,3 à 5V
Precisão	1 à 2 graus
Corrente em funcionamento	0,1 mA
Ângulo de medida	15 graus
Interface de comunicação	I2C

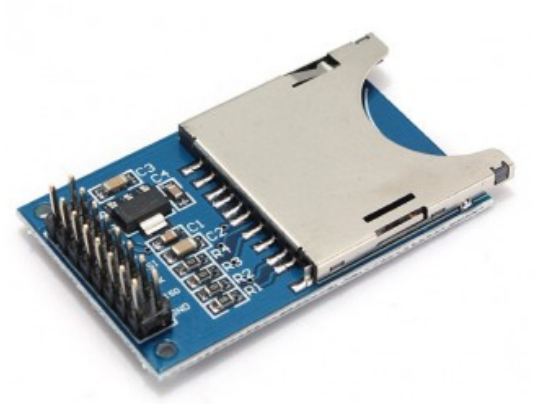
Fonte: (KURPIEL et al., 2010)

Embora não seja aplicada no robô desenvolvido é uma forte sugestão de implementação futura para o módulo.

5.3.4 Módulo SD Card

O módulo cartão SD, mostrado na Figura 5.11, é de fundamental importância. O mesmo é utilizado como armazenador de dados durante execução de atividades do robô em ambiente real, dados estes que são utilizados, posteriormente, para visualização da trajetória percorrida e comparações com simulações realizadas.

Figura 5.11 – Módulo SD Card utilizado para gravação de dados de deslocamento obtidos do módulo robótico desenvolvido, durante a execução de rotas pelo mesmo.



Fonte: (BORGES; NUNES; BORGES, 2014)

O Módulo *SD Card* possui capacidade de escrita e leitura e facilidade em conexão com o Arduino e outros microcontroladores (BORGES; NUNES; BORGES, 2014).

Sua comunicação com o microcontrolador se dá por SPI padrão, assim como o módulo de bússola, o que significa que os dispositivos podem transmitir dados usando a mesma linha de transmissão (BORGES; NUNES; BORGES, 2014).

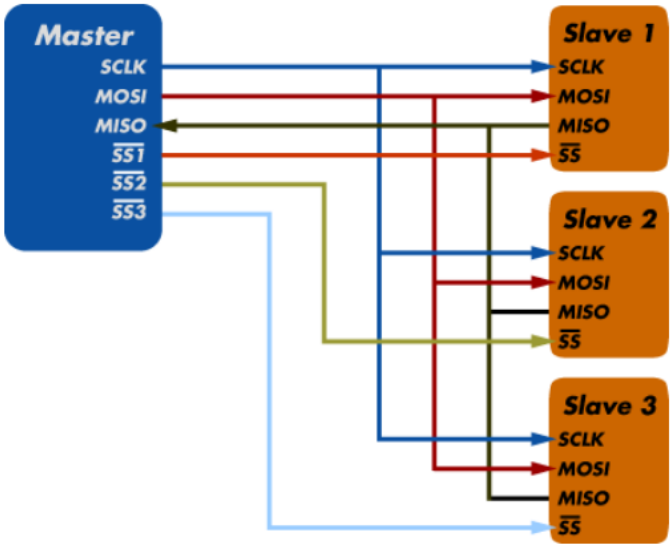
É compatível com cartões FAT16 ou FAT32 e possui tensão de entrada entre 3,3 e 5 V (BORGES; NUNES; BORGES, 2014).

Conforme evidenciado nas seções anteriores, tanto o módulo de bússola quanto o módulo do cartão SD possuem comunicação com o microcontrolador por SPI e seu entendimento, portanto, se torna necessário.

5.4 Comunicação SPI

A comunicação SPI (*Serial Peripheral Interface*) é um protocolo de dados seriais síncronos utilizado em microcontroladores para comunicação entre o microcontrolador e um ou mais periféricos que, também, pode ser utilizado entre dois microcontroladores (SILVA; ANJOS, 2013). É capaz de atingir velocidades maiores de comunicação com pouca deformação de sinal e tem como característica fundamental ser uma comunicação *full-duplex*, ou seja, é capaz de enviar e receber dados ao mesmo tempo. Esse dispositivo obedece a uma forma de ligação singular entre microcontrolador e módulo que deve ser respeitado como mostra a Figura 5.12.

Figura 5.12 – Esquema básico de ligação entre módulo SPI e microcontrolador.



Fonte: (SILVA; ANJOS, 2013)

A comunicação SPI sempre tem um *master*, ou seja, sempre um será o módulo principal e o restante será *slave* (SILVA; ANJOS, 2013). Nessa pesquisa o Arduino é o *master* e os outros periféricos (bússola e cartão SD) são *slave*.

Esta comunicação contém 4 conexões descritas a seguir:

- **MISO** (Master IN Slave OUT): Transmissão de dados do Slave para o Master;
- **MOSI** (Master OUT Slave IN): Transmissão de dados do Master para o Slave;
- **SCK** (Serial Clock): Clock de sincronização para transmissão de dados entre o Master e Slave;
- **SS** (Slave Select): Seleciona qual Slave receberá os dados.

É importante ressaltar que mais de um dispositivo SPI pode ser acoplado ao sistema. A porta SS é a responsável por ativar cada dispositivo quando se deseja estabelecer comunicação com este. A ativação do módulo de gravação para cartão SD pode ser vista na Figura 5.13.

Figura 5.13 – Ativação de comunicação SPI com módulo de cartão SD

```
// Inicia cartão SD
Serial.print("Iniciando cartão SD ...");
// Garanta que o pino padrão de envio SD (53) seja setado como saída, mesmo que vc não o tenha utilizado.
pinMode(53, OUTPUT);
// Verifique se cartão está presente e pode ser iniciado.
if (!SD.begin(53)) //se não iniciar SD no pino selecionado (53)
{
  Serial.println("SD com erro, ou não presente");// avisa que cartão está com erro, ou não está presente
  // não faça mais nada:
  while(1);
}
//se inicia no pino correto
Serial.println("Cartão inicado"); // avisa que cartão está iniciado
// Ativa timer1 e interrupções externas
```

Fonte: Próprio Autor

5.5 Softwares

Nessa seção serão descritos os *softwares* utilizados para execução da pesquisa.

Um *software* é o conjunto de programas, instruções e regras informáticas que permitem realizar atividades específicas em um sistema, no caso desse trabalho são responsáveis por gravar o controle criado no microcontrolador utilizado pelo robô e principalmente responsáveis pela realização de simulações e visualizações gráficas de resultados.

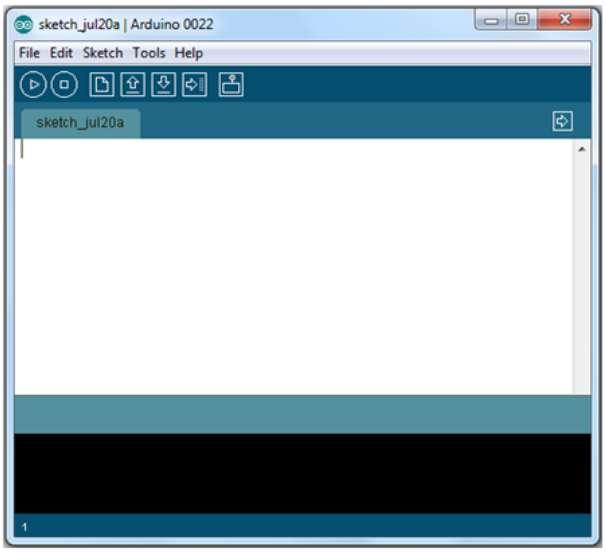
5.5.1 IDE Arduino

Para realizar a programação de um Arduino, é necessária a utilização de sua IDE(*Integrated Development Environment* ou Ambiente de Desenvolvimento Integrado) (Figura 5.14), que nada mais é que um *software* onde o código de programação é escrito e também é o meio pelo qual é realizado o *upload* (envio da programação para a placa) do mesmo código para a placa que, por fim, executa as instruções definidas.(RENNA et al., 2013)

O *software* de programação do Arduino pode ser executado no Windows, Mac OS X e Linux e possui ambiente escrito em Java e baseado em *Processsing*, tendo assim uma linguagem de programação própria.

A placa de prototipagem Mega2560, utilizada nessa pesquisa, pode ser programada através deste *software*, portanto essa IDE é de fundamental importância.

Figura 5.14 – IDE do Arduino utilizada para escrita e gravação de código desenvolvido para controle do módulo robótico desenvolvido.

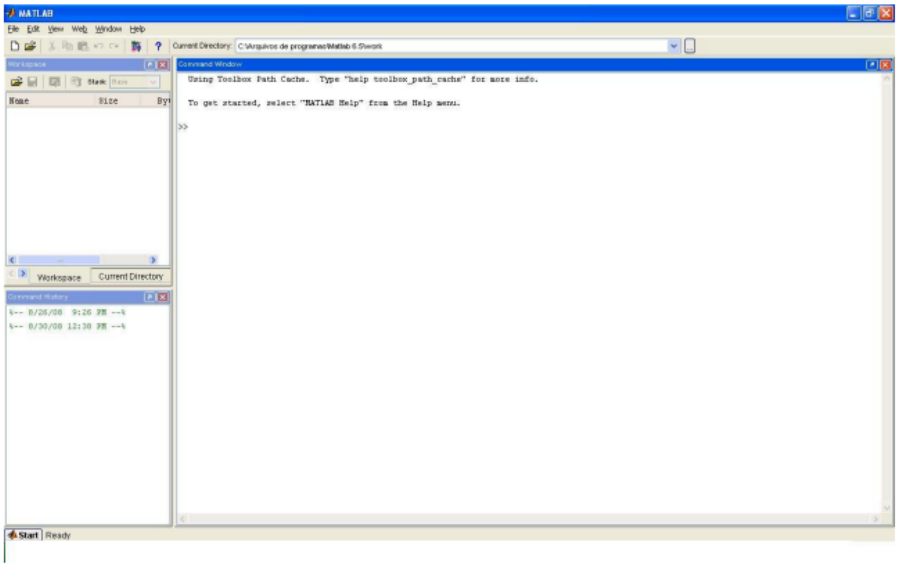


Fonte: Próprio Autor

5.5.2 MatLab

O *software* Matlab é um *software* de alto desempenho que surgiu com a finalidade de realizar cálculos matriciais. Porém, atualmente seu uso é bem mais amplo. (BECKER et al., 2010).

Figura 5.15 – Tela de trabalho do Matlab, *software* utilizado para realização de simulações e testes relevantes referentes ao modelo do módulo robótico desenvolvido.



Fonte: Próprio Autor

É uma ferramenta e uma linguagem de programação de alto nível com ambiente de

trabalho muito simples e intuitivo, mostrado na Figura 5.15, e tem como principais funções as citadas a seguir (BECKER et al., 2010):

- Cálculos matemáticos;
- Modelagem, simulação e confecção de protótipos;
- Desenvolvimento de interface gráfica;
- Gráficos científicos;
- Desenvolvimento de algoritmos;
- Análise, simulação e obtenção de dados.

Além disso, o MatLab possui uma grande quantidade de bibliotecas auxiliares que aperfeiçoam o tempo gasto para realizar tarefas, uma vez que, o usuário poderá utilizar muitas funções já definidas, poupando o tempo de criá-las. Porém, os algorítmos desenvolvidos no *software* raramente podem ser executados em outros ambientes.

Nesse Capítulo, apenas a execução de uma interface gráfica não é utilizada, todas as outras são aplicadas em algum momento. Todas as simulações e modelagens são executadas na linha de comando do *software*.

5.6 Conclusão do capítulo

Um sistema elétrico como é o robô desenvolvido nessa pesquisa, é composto de diversos componentes eletrônicos como controladores, sensores e atuadores e o estudo desses componentes é fundamental para o entendimento do módulo e para a correta aplicação de sua modelagem matemática, que depende das variáveis de cada componente de atuação.

Nessa seção, então, as principais características de cada dispositivo utilizado na composição do robô foram evidenciadas e, a partir do próximo capítulo, elas serão aplicadas nas simulações e nas definições de parâmetros de controle do sistema.

6 Implementação da Modelagem Cinemática e Dinâmica

O presente Capítulo traz o desenvolvimento inicial do projeto tendo em vista execução de trajetórias pré-estabelecidas.

Inicialmente, o sistema em Espaço de Estados para o módulo, que foi obtido implicitamente no Capítulo 4 desse documento, é evidenciado de forma numérica, dando origem ao sistema que será utilizado por todo o processo de pesquisa.

Através do equacionamento estabelecido, testes referentes à estabilidade do sistema proposto são realizados em ambiente virtual e posteriormente, simulações de trajetórias são executadas no mesmo ambiente.

São aplicadas trajetórias em linha reta e em circunferência completa. Por fim, as mesmas trajetórias são executadas em ambientes reais.

6.1 Modelagem em Espaço de Estados do Sistema Robótico

No capítulo anterior, o modelo dinâmico do robô construído para a pesquisa aqui apresentada foi descrito através de um sistema em espaço de estado, cujas equações que o representam são como:

$$\begin{bmatrix} \dot{I}_m(t) \\ \dot{\theta}(t) \\ \dot{\omega}(t) \end{bmatrix} = \begin{bmatrix} -\frac{R_m}{L_m} & 0 & -\frac{K_m}{L_m} \\ 0 & 0 & 1 \\ \frac{K_t}{J_L} & 0 & -\frac{b}{J_L} \end{bmatrix} \begin{bmatrix} I_m(t \\ \theta(t) \\ \omega(t) \end{bmatrix} + \begin{bmatrix} \frac{K_{ADC}}{L_m} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} V_{\text{PWM}}(t) \\ 0 \\ 0 \end{bmatrix} \tag{6.1}$$

$$\begin{bmatrix} y_2(t) \\ y_3(t) \end{bmatrix} = \begin{bmatrix} 0 & K_{\text{ENC}} & 0 \\ 0 & 0 & K_{\text{RED}} \end{bmatrix} \begin{bmatrix} I_m(t) \\ \omega(t) \\ \theta(t) \end{bmatrix}. \tag{6.2}$$

As variáveis descritas pelo espaço de estados formado, são referentes aos dispositivos eletrônicos que formam o módulo robótico e os mesmos foram discutidos no Capítulo 5 desse documento.

Dessa maneira, através da Tabela 5.2, encontrada na seção 5.3.1, tem-se as seguintes variáveis presentes em (6.2) e (6.2):

- **Resistência de Armadura** (R_m): $7,2\ \Omega$;
- **Indutância de Armadura** (L_m): 3.400 μH;

- **Constante de força eletro motriz** (Km): 13,178 $\frac{mV}{rad/s}$;
- **Constante de Torque** (K_t): 13,15 mW(m/a);
- **Momento de Inércia** (J_l): $4,5.10^{-7}/m^2$;
- **Atrito Viscoso** (b): 4,7989 $\mu Nm/(rad^2/2)$;
- **Constante de Redutor** (K_{RED}): 1/84.

Pela Tabela 5.1, presente na seção 5.2.1, tem-se que a tensão de operação do placa de prototipagem utilizada para controle do sistema é de 5 V. Como o conversor AD da mesma possui resolução de 10 bits, mais uma variável presente nas equações em espaço de estados do sistema pode ser estabelecida e é dada a seguir:

- **Constante de ADC** (K_{ADC}): $4,88.10^{-3}$.

A tensão da fonte CC utilizada como alimentação do robô é de 12 V e, segundo a seção 5.4.1, o encoder utilizado para realizar a leitura da velocidade que os motores de tração do sistema giram foram configurados para registrar 48 pulsos a cada volta completa de seu disco, ou a cada giro completo do eixo do motor. Dessa forma, as duas últimas variáveis do sistema podem ser obtidas e são as seguintes:

- **Tensão** (V_{PWM}): Tensão variável aplicada ao motor CC;
- **Constante de Encoder** (K_{ENC}): 7,6394 leituras/rad.

Tendo todos os termos e seus valores correspondentes descritos, os mesmos podem ser substituídos em (6.1) e (6.2) dando origem, assim, ao sistema que será utilizado daqui por diante.

Para a matriz **A** do sistema, tem-se:

$$\mathbf{A} = \begin{bmatrix} -\frac{R_m}{L_m} & 0 & -\frac{K_m}{L_m} \\ 0 & 0 & 1 \\ \frac{K_t}{J_L} & 0 & -\frac{b}{J_L} \end{bmatrix} = \begin{bmatrix} -\frac{7,2}{3400.10^{-6}} & 0 & -\frac{13,178.10^{-3}}{3400.10^{-6}} \\ 0 & 0 & 1 \\ \frac{13,15.10^{-3}}{4,5.10^{-7}} & 0 & -\frac{4,7989.10^{-6}}{4,5.10^{-7}} \end{bmatrix}, \tag{6.3}$$

para a matriz **B**:

$$\mathbf{B} = \begin{bmatrix} \frac{K_{\text{ADC}}}{L_m} \end{bmatrix} = \begin{bmatrix} \frac{4,88.10^{-3}}{3400.10^{-6}} \end{bmatrix}, \tag{6.4}$$

para a matriz **C**, tem-se:

$$\mathbf{C} = \begin{bmatrix} 0 & K_{\text{ENC}} & 0 \\ 0 & 0 & K_{\text{RED}} \end{bmatrix} = \begin{bmatrix} 0 & 7,64 & 0 \\ 0 & 0 & \frac{1}{84} \end{bmatrix}. \tag{6.5}$$

Como a matriz **D** é nula, temos finalmente que o sistema que representa o módulo robótico construído para essa pesquisa, em espaço de estado é dado por:

$$\begin{bmatrix} \dot{I}_m(t \\ \dot{\theta}(t) \\ \dot{\omega}(t) \end{bmatrix} = \begin{bmatrix} -2,1176.10^3 & 0 & -3,8758 \\ 0 & 0 & 1 \\ 29,2222.10^3 & 0 & -10,6642 \end{bmatrix} \begin{bmatrix} I_m(t \\ \theta(t) \\ \omega(t) \end{bmatrix} + \begin{bmatrix} 1,4352 \end{bmatrix} \begin{bmatrix} V_{pwm}(t) \end{bmatrix}, \quad (6.6)$$

$$\begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{bmatrix} = \begin{bmatrix} 0 & 7,54 & 0 \\ 0 & 0 & \frac{1}{84} \end{bmatrix} \begin{bmatrix} I_m(t) \\ \omega(t) \\ \theta(t) \end{bmatrix}. \quad (6.7)$$

6.2 Estabilidade do Sistema

Tendo o sistema em espaço de estados definido, o mesmo foi implementado no o *software* Matlab a fim de que simulações referente a estabilidade do mesmo pudessem ser realizadas.

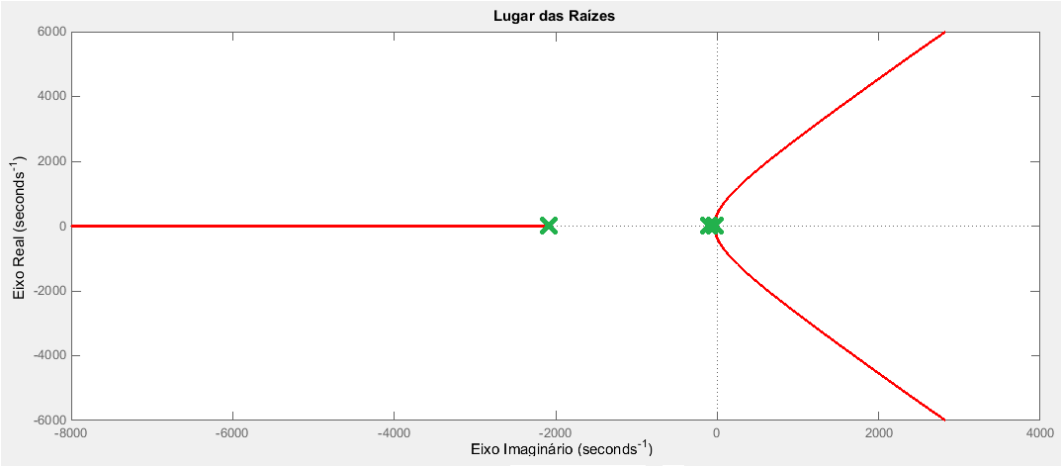
Considerando o sistema sem carga e com redutor, a estabilidade foi testada para as duas saídas controláveis: velocidade ângular de cada roda e deslocamento ângular do módulo sobre o plano de referência.

Inicialmente o sistema foi declarado na forma de *script* do Matlab, com o comando *state-space*, ou ss a fim de ser, posteriormente, executado na janela de comandos do *software*.

Com o espaço de estados pronto, a estabilidade é verificada através da alocação dos pólos para cada variável de controle e depois, confirmada através da aplicação de entradas degrau e impulso para o sistema, como sugere (DORF; BISHOP, 2001).

Levando em conta o sistema sem carga e com redutor, e desconsiderando a variável de corrente de motor, visto que o trabalho visa o controle do posicionamento do módulo e seu deslocamento, ainda com auxilio do script do Matlab, a primeira validação a ser realizada foi em relação ao lugar das raízes. Para isso, o comando *rootlocus* foi aplicado ao espaço de estados obtido. Levando em conta, inicialmente, a variável de controle referente ao posicionamento ângular do módulo, o resultado obtido pose ser observado na Figura 6.1.

Figura 6.1 – Lugar das raízes obtido para sistema robótico desenvolvido tendo em base sua variável de posicionamento angular.

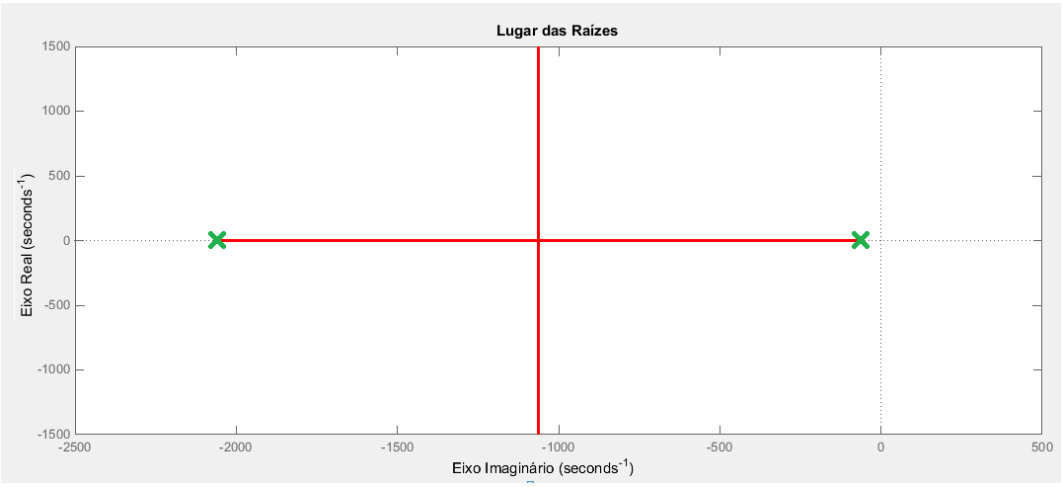


Fonte: Próprio Autor

De acordo com a Figura 6.1, existe um polo na origem, o que torna o sistema, para controle do posicionamento angular, instável considerando a teoria de estapilidade por lugar das raízes.

Realizando o mesmo processo para a variável referente à velocidade angular do módulo, o lugar das raízes para essa pode ser observada na Figura 6.2.

Figura 6.2 – Lugar das raízes obtido para sistema robótico desenvolvido tendo em base sua variável de velocidade angular.



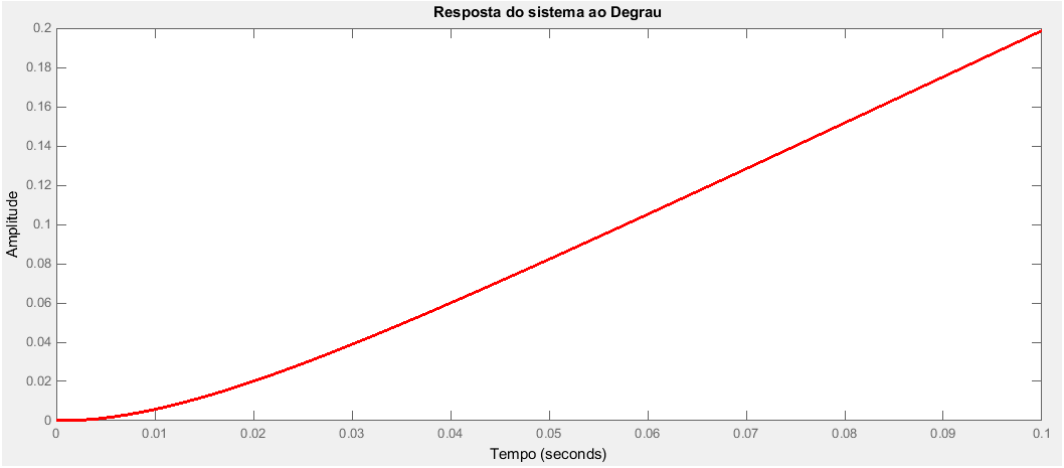
Fonte: Próprio Autor

De acordo com a Figura 6.2, não existem polos no eixo imaginário do plano, o que torna o sistema, para controle da velocidade angular, estável.

Finalmente, a fim de confirmação dos resultados obtidos na verificação de estabilidade do sistema, entradas degrau e impulso foram aplicadas ao mesmo para as duas variáveis em questão: posicionamento angular e velocidade angular.

Para o deslocamento angular, aplicando-se uma entrada degrau, a resposta obtida e mostrada na Figura 6.3.

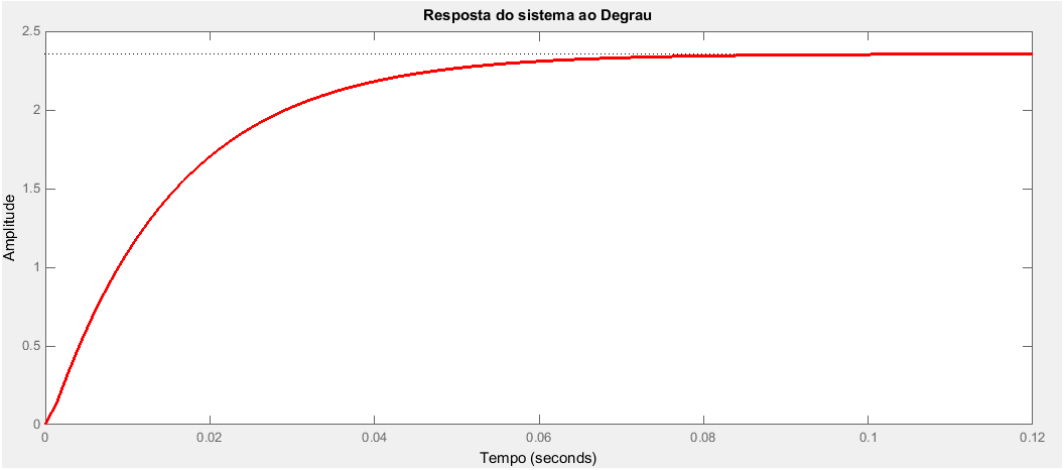
Figura 6.3 – Resposta à entrada degrau obtida para sistema robótico desenvolvido tendo em base sua variável de posicionamento angular.



Fonte: Próprio Autor

Aplicando-se uma entrada impulso para o mesmo sistema, a resposta obtida é mostrada pela Figura 6.4.

Figura 6.4 – Resposta à entrada impulso obtida para sistema robótico desenvolvido tendo em base sua variável de posicionamento angular.



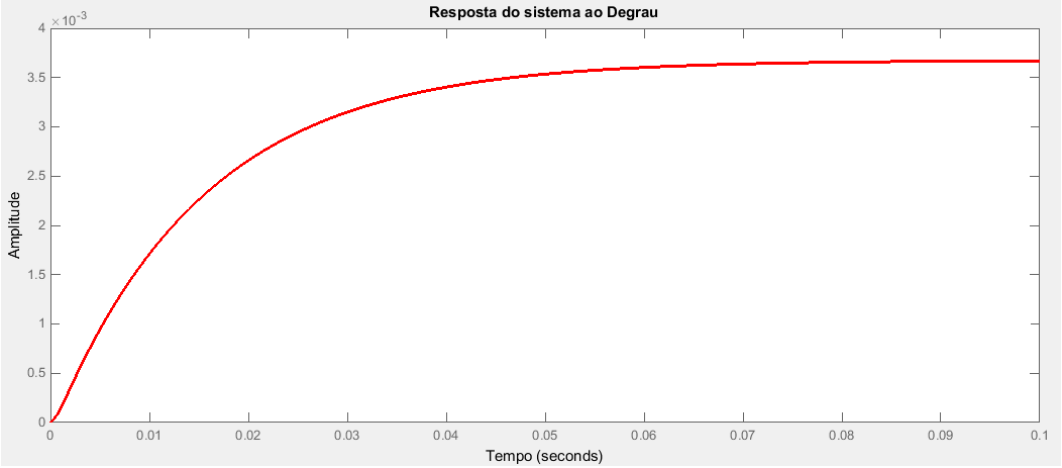
Fonte: Próprio Autor

Observando-se as respostas obtidas, o sistema se apresenta instável para a análise do posicionamento angular do módulo, se for considerado unicamente a análise literal, o que já havia sido verificado através do lugar das raízes para esse sistema. Se a análise se der em base no fato de uqe, ao receber uma tensão constante, o motor gira a velocidade constante e, portanto, seu deslocamento ângular aumenta, o sistema apresenta resultado esperado e estabilidade. Dessa

forma, a validação passa a se deter na variável de velocidade angular do sistema, que se mostrou estável.

Para a velocidade angular, aplicando-se uma entrada degrau, a resposta obtida e mostrada na Figura 6.5.

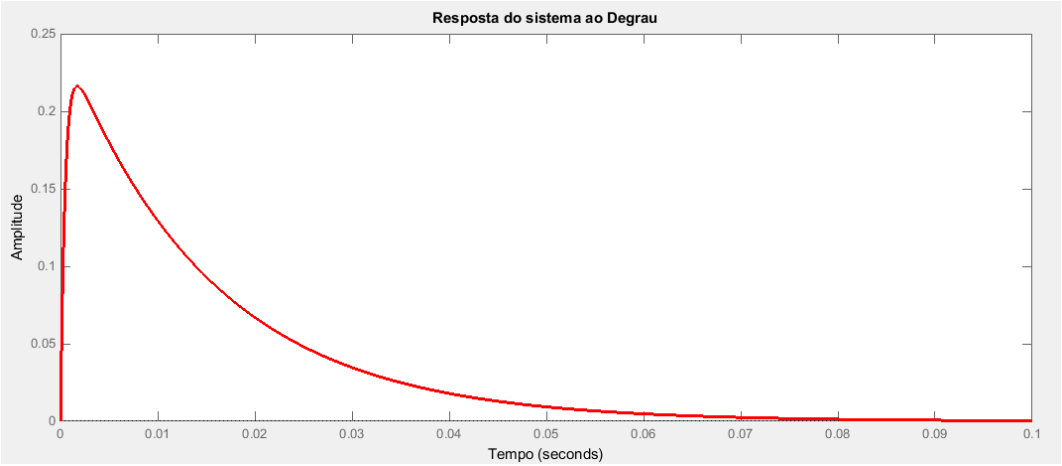
Figura 6.5 – Resposta à entrada degrau obtida para sistema robótico desenvolvido tendo em base sua variável de velocidade angular.



Fonte: Próprio Autor

Aplicando-se uma entrada impulso para o mesmo sistema, a resposta obtida é mostrada pela Figura 6.6.

Figura 6.6 – Resposta à entrada impulso obtida para sistema robótico desenvolvido tendo em base sua variável de velocidade angular.



Fonte: Próprio Autor

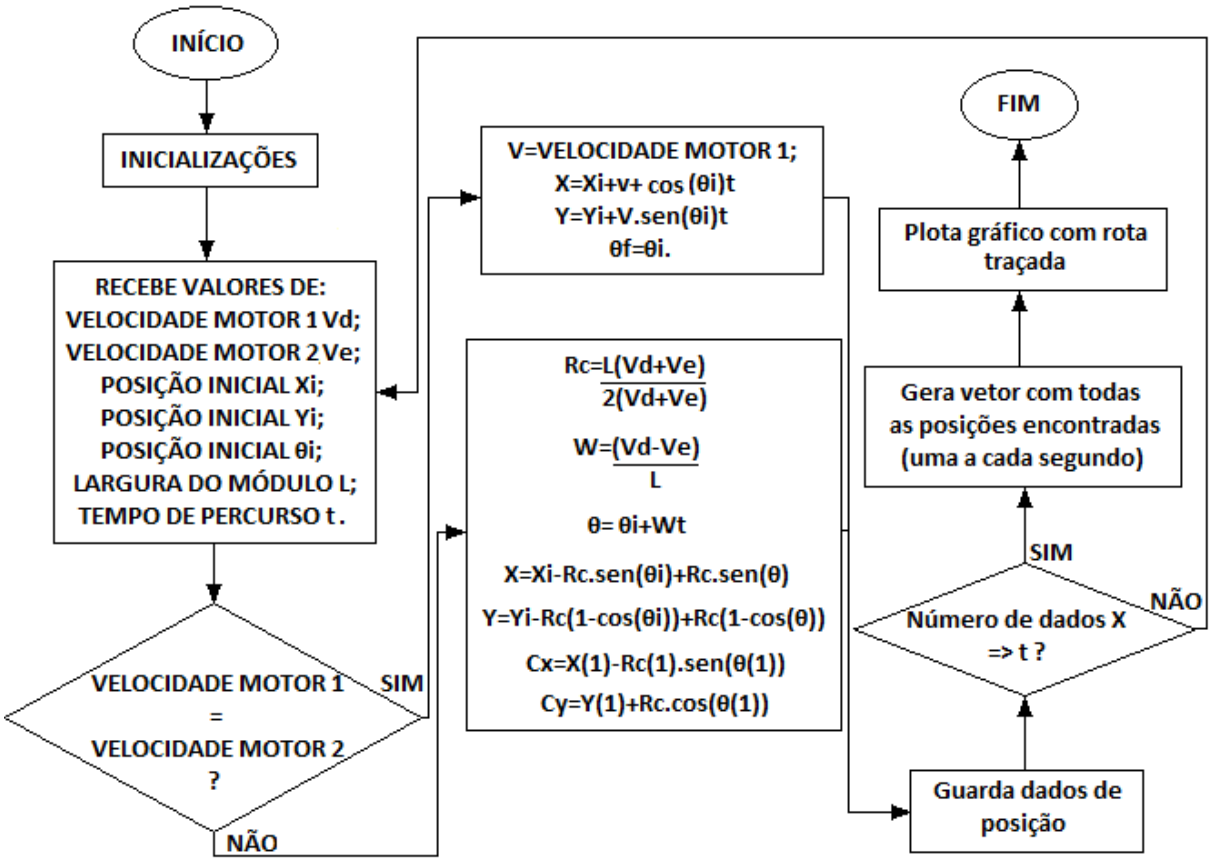
Observando-se as respostas obtidas, o sistema se apresenta estável para o controle da velocidade angular do módulo, o que já havia sido verificado através do lugar das raízes para esse sistema. Assim, a validação se confirma e o controle passa a se dar apenas sobre essa variável.

6.3 Implementação de Trajetórias

Com a variável de controle estabelecida e sua estabilidade validada, dá-se início a parte de simulações do protótipo executado.

A partir dos equacionamentos da cinemática e dinâmica estabelecidos para o robô (apresentados no Capítulo 4), e tendo como base o trabalho desenvolvido por Melo (2007) em que foi desenvolvido um simulador de trajetórias para módulos robóticos, um modelo similar foi elaborado para o protótipo em desenvolvimento e o mesmo pode ser visto pela Figura 6.7.

Figura 6.7 – Fluxograma representando simulador de trajetória desenvolvido a fim de visualizar as trajetórias traçadas pelo sistema através de entradas de velocidades de suas rodas de tração, tempo de percurso e largura do robô.



Fonte: Próprio Autor

A fim de que o simulador trabalhasse de acordo com o modelo de robô desenvolvido aqui, as equações do mesmo foram aplicadas ao simulador proposto por (MELO, 2007).

O simulador foi desenvolvido com auxílio do *software Matlab* e o mesmo funciona de maneira muito simples. Através da entrada de valores das velocidades de cada roda do módulo, de seu posicionamento em relação ao plano de referência global, sua largura e o tempo que o mesmo irá percorrer determinado trajeto, o simulador calcula a cada intervalo de tempo o

posicionamento do módulo em relação ao plano de referência local do mesmo e ao fim, terá vários pontos de posicionamentos encontrados pelos quais traça retas, formando uma trajetória.

O simulador apresentado é o mesmo utilizado por (RODRIGUES, 2014) em sua pesquisa referente a trajetórias num protótipo que utilizava de componentes idênticos ao utilizado nessa pesquisa.

Para que simulações pudessem ser realizadas com auxílio do simulador apresentado, velocidades para as duas rodas devem ser fornecidas. A fim de que as velocidades fossem coerentes com a prática a ser realizada posteriormente, um estudo cauteloso foi executado sobre cada motor do robô, independentemente, e o mesmo será evidenciado a partir de agora.

6.3.1 Estudo da velocidade do sistema

O robô tem seu movimento baseado em dois motores de tração alocados na parte traseira do módulo, nomeados de MD e ME. Cada motor é acoplado a um redutor com capacidade de redução de giro de 84 vezes e a um encoder que gera 48 pulsos elétricos a cada giro do eixo do motor.

O controle das velocidades de cada motor é dado através de controle PWM que fornece tensões diferentes, dependendo do valor de entrada fornecido ao sistema. Tendo em mente que o microcontrolador utilizado para controle do módulo é o MEGA2560, já estudado nesse documento, e que o mesmo possui 8 bits de processamento, conclui-se que o controlador é capaz de gerar 256 níveis de controle para o PWM a ser enviado a cada motor, sendo que o valor 0 equivale a uma tensão aplicada de 0 V e, portanto, não gera movimento ao sistema e 255 equivale à tensão máxima e, consequentemente, à velocidade máxima a ser executada pelo protótipo.

Pulsos entre 0 e 250 foram aplicados a cada motor, com intervalo de 25 entre cada teste realizado e, por 10 segundos, foram medidas as velocidades de cada motor em RPM. As medições obtidas para cada quantidade de pulsos PWM aplicado são evidenciada na Tabela 6.1.

Tabela 6.1 – Velocidades, em RPM, para motores A e B em relação à quantidade de pulsos aplicados a entrada PWM de controle para cada motor

PULSOS	RPM MA	RPM MB	PULSOS	RPM MD	RPM ME
0	0	0	25	10,86	3,5
	0	0		16	10,8
	0	0		16,18	10,46
	0	0		15,84	9,88
	0	0		15,42	8,99
	0	0		15,07	8,13
	0	0		15	7,02
	0	0		14,9	6,76
	0	0		14,76	7,22
	0	0		15,94	7,56

PULSOS	RPM MA	RPM MB	PULSOS	RPM MD	RPM ME
50	22,62	16,85	75	43,51	40,71
	36,86	34,43		51,8	49,69
	36,86	34,39		51,82	49,04
	36,86	33,72		51,62	49,08
	36,7	33,88		51,64	49,58
	36,59	34,33		51,5	49,06
	36,61	33,87		51,47	49,54
	36,44	33,57		51,4	49,51
	36,35	34,06		51,38	49,32
	36,37	34,17		51,35	49,54
PULSOS	**RPM MA**	**RPM MB**	**PULSOS**	**RPM MD**	**RPM ME**
100	59,46	56,86	125	72,51	70,8
	66,71	64,76		82,13	80,21
	66,4	64,52		82,14	80,45
	66,29	64,87		82,28	80,3
	66,38	64,64		82,26	80,09
	66,49	64,97		82,29	80,19
	66,47	65,07		82,14	79,9
	66,24	64,82		82,29	79,38
	66,38	64,81		82,23	79,39
	66,35	64,58		82,29	79,57
PULSOS	**RPM MA**	**RPM MB**	**PULSOS**	**RPM MD**	**RPM ME**
150	93,04	89,79	175	104,23	101
	97,72	94,49		114,03	110,52
	97,81	95,19		113,97	110,15
	97,99	95,48		114,24	110,13
	97,98	95,28		114,11	110,16
	97,93	95,24		114,11	110,42
	97,84	95,06		114,52	110,16
	97,89	95,34		114,72	110,43
	98,29	95,85		114,13	110,47
	98,08	95,42		114,51	110,15
PULSOS	**RPM MA**	**RPM MB**	**PULSOS**	**RPM MD**	**RPM ME**
200	125,64	120,19	225	132,59	126,01
	129,76	125,09		145,76	138,35
	130,1	124,45		148,88	138,35
	130,03	124,33		145,82	138,33
	130,12	124,32		145,8	138,56
	130,33	124,32		145,89	135,5
	130,55	124,52		146,03	138,17
	130,57	124,46		145,98	138,44
	130,57	124,55		145,74	138,6
	130,79	124,52		145,82	138,59

PULSOS	RPM MA	RPM MB
250	148,87	141,24
	157,69	149,81
	157,43	149,87
	157,32	149,99
	157,26	149,76
	157,2	149,96
	157,32	149,76
	157,74	149,94
	157,36	149,78
	157,47	149,84

Fonte: Próprio Autor

Com as medições realizadas, os valores das velocidades médias, em RPM para cada motor nas quantidades de pulsos estabelecidas foram calculadas e os resultados são mostrados na Tabela 6.2

Tabela 6.2 – Velocidades médias, em RPM para motores A e B em relação à quantidade de pulsos aplicados a entrada PWM de controle para cada motor.

PULSOS	RPM MD	RPM ME
0	0	0
25	14,992	8,007
50	35,226	32,327
75	50,749	48,54
100	65,717	63,99
125	81,256	79,028
150	97,457	94,714
175	113,26	109,36
200	129,85	124,08
225	144,83	137,22
250	156,57	149

Fonte: Próprio Autor

Observou-se também, através das medições realizadas, que os motores apresentam maior estabilidade de funcionamento entre as entradas de pulso entre 50 e 125, tendo entre essa o desvio máximo entre medidas subsequentes todas menores que 1 RPM e, por esse fato, os valores aplicados para o movimento do módulo foram escolhidos entre esses dois e serão evidenciados mais à frente nesse trabalho.

6.3.2 Conversão de velocidades

No simulador de trajetórias, as entradas de velocidade são lineares, em milímetros por segundo. Porém, como já discutido nesse documento, o controle do sistema se dá pelas suas velocidades angulares. Dessa forma, o fator de conversão de um valor para outro é de fundamental importância para o prosseguimento da pesquisa.

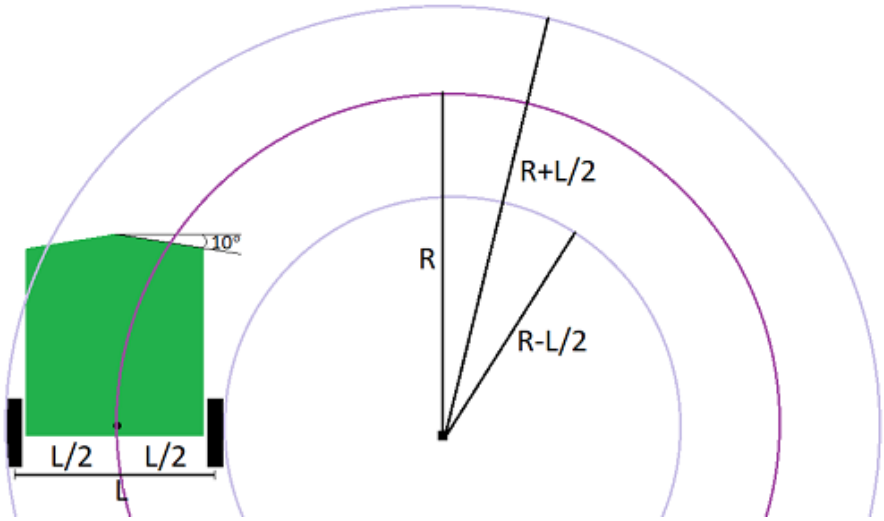
É de conhecimento que o comprimento de uma circunferência, c, é igual à: $c = 2\pi r$ onde r representa o raio da circunferência. Sabe-se, também, que a unidade do sistema internacional de unidades para medição de velocidade angular é o rpm, ou seja, rotação por minuto. Como a entrada do sistema simulado se dá em mm/s, a conversão se da conforme e Eq. 6.8.

$$V(mm/s) = (\omega/60)2\pi r, \tag{6.8}$$

sendo que o raio r é fornecido em milímetros.

Outro cálculo relevante se dá sobre a aplicação de velocidades para que o sistema realize uma curva. Pra auxiliar esse estudo, analisa-se Figura 6.8.

Figura 6.8 – Esquema de representação de deslocamento executado por ambas as rodas de tração do módulo robótico desenvolvido sobre trajetória de circunferêcia.



Fonte: Próprio Autor

Estabelecendo R como o raio de curvatura desejado para o módulo, tendo o ponto central entre as duas rodas de tração como referência, pode-se observar que a roda externa ao ponto de curvatura deverá percorrer um raio de $R + L/2$ enquanto que a roda que se encontra do lado interno ao ponto de curvatura deverá percorrer um raio de $R - L/2$. Com base nesses dados e nas equações de deslocamento já obtidas no estudo cinemático desse trabalho, apresentado no Capítulo 4, tem-se que as velocidades angulares para roda externa e interna, respectivamente, do módulo, quando esse está traçando uma curva, são as evidenciadas por

$$\omega = \frac{2\pi(R + L/2)2}{t}\frac{1}{2r.0,10472} \tag{6.9}$$

e

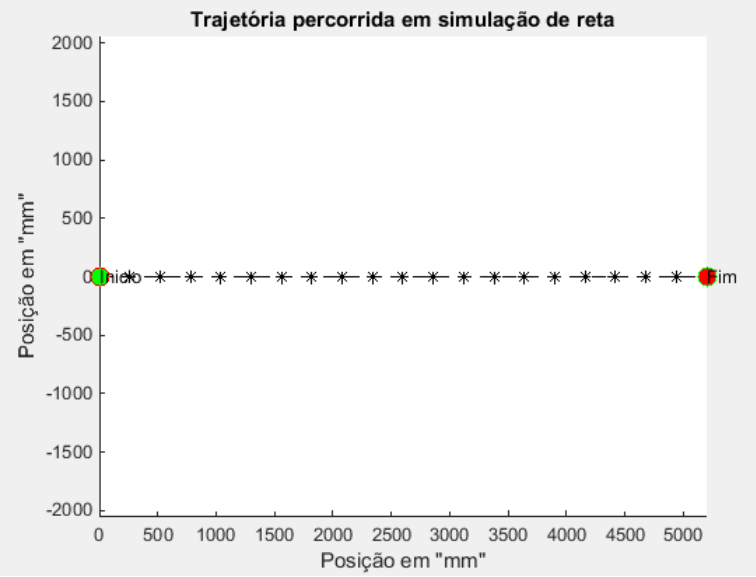
$$\omega = \frac{2\pi(R - L/2)2}{t}\frac{1}{2r.0,10472}. \tag{6.10}$$

6.3.3 Simulação de trajetórias

Tendo conhecimento do funcionamento dos motores, simulações de suas trajetórias foram executadas, sendo a primeira em linha reta e a segunda uma circunferência completa de raio igual a 1 metro.

Tendo em base o trabalho desenvolvido por (RODRIGUES, 2014), velocidades iguais para as duas rodas foram aplicadas ao simulador. Aplicando-se a velocidade de 50 rpm, que equivale a aproximadamente 260 mm/s a cada roda e estabelecendo um tempo de simulação de 20 s, iniciando-se a contagem em 0, tem-se que o módulo deverá percorrer idealmente 4.974,18 mm. O resultado da simulação realizada para trajetória em linha reta realizada pode ser observada pela Figura 6.9.

Figura 6.9 – Resultado de trajetória executada pelo sistema robótico desenvolvido sobre simulação de rota em linha reta num plano bidimensionsal.

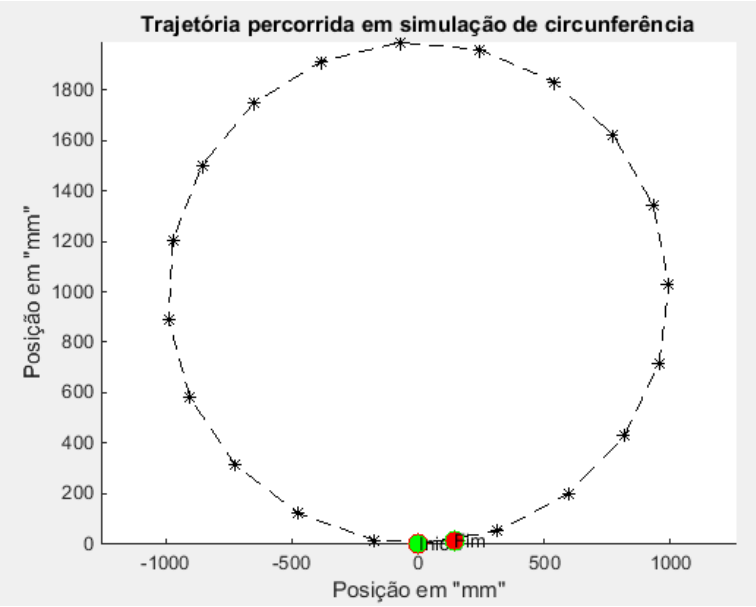


Fonte: Próprio Autor

Como o esperado, o módulo percorre os 4974,18 mm, o que valida a simulação.

Para um segundo teste, pretendendo uma trajetória circular de raio igual a um metro, velocidades de 50 rpm e 52 rpm foram aplicadas as rodas externa e interna, respectivamente do módulo, obedecendo às equações de cinemática do sistema. Para essa simulação o tempo de execução foi fixado em 20 segundos, o que geraria um percurso idealmente circular. O resultado da simulação para esses valores de entrada pode ser observado pela Figura 6.10.

Figura 6.10 – Resultado de trajetória executada pelo sistema robótico desenvolvido sobre simulação de rota em circunferência num plano bidimensionsal.



Fonte: Próprio Autor

Novamente, o resultado da simulação é igual ao esperado e o sistema pode ser aplicado no módulo real.

6.4 Desvio de obstáculos

Com o sistema de trajetórias em funcionamento o desvio de obstáculos passou a ser analisado para o robô.

Levando em conta a estrutura física do módulo e a usabilidade mais frequente em meio acadêmico de pesquisa, o algoritmo BUG 2 foi o escolhido para o sistema de duas formas distintas, a primeira com desvio realizado exclusivamente por manobras com ângulos de 90 graus e a segunda com desvio realizado a partir de uma semi circunferência ao redor do obstáculo.

Para que um corpo robótico possua a capacidade de desviar de obstáculos, esse precisa ter a capacidade de detectar os mesmos e essa detecção se dá por sensores que podem ser os mais diversos possíveis.

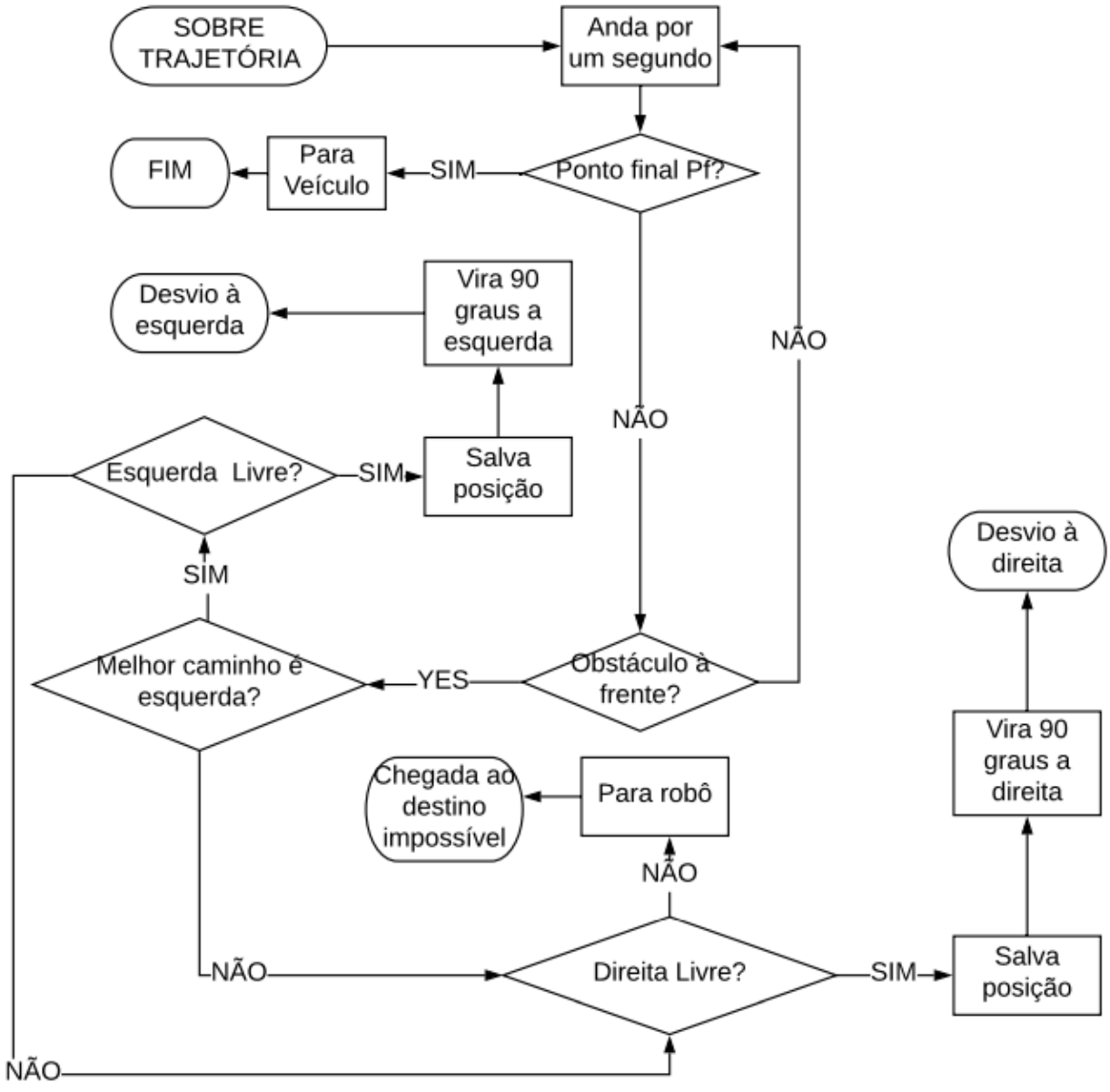
Entre os sensores de aplicação mais usuais estão o sensor de distância ultrassônico e o sensor de distância óptico, sendo que o primeiro foi o escolhido para ser utilizado.

6.4.1 Algoritmos de desvio

Como já mencionado nesse Capítulo 2, o algoritmo utilizado nessa pesquisa será o BUG 2, famoso algoritmo da literatura científica.

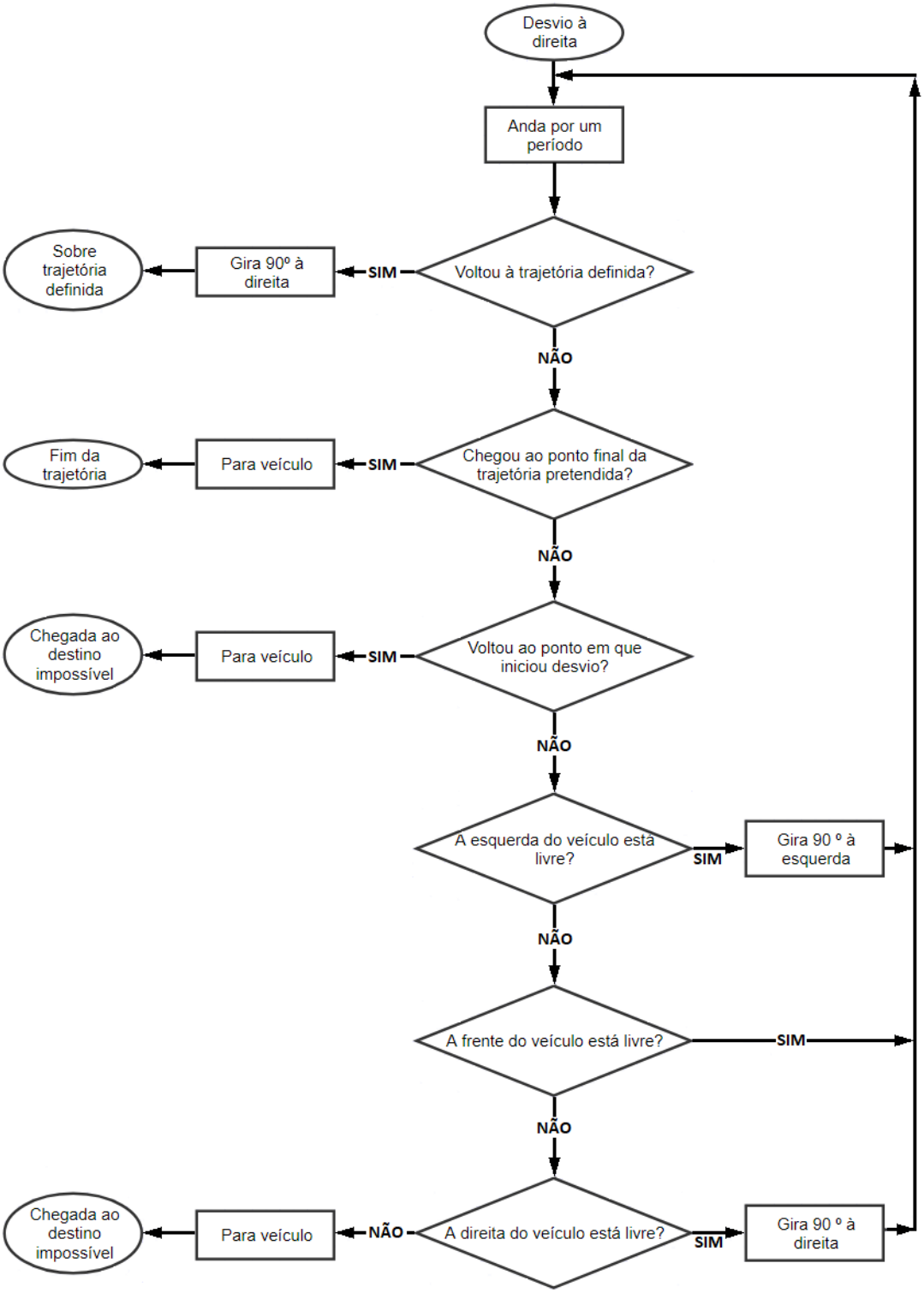
Duas versões do algorítmo serão executadas, a primeira com manobras de desvio exclusivamete com curvas de 90 graus sobre o eixo, tem o fluxograma representado pelas Figuras 6.11 6.12 6.13.

Figura 6.11 – Fluxograma de trajetória para navegação autônoma de robô móvel segundo algorítmo BUG 2 proposto na pesquisa.



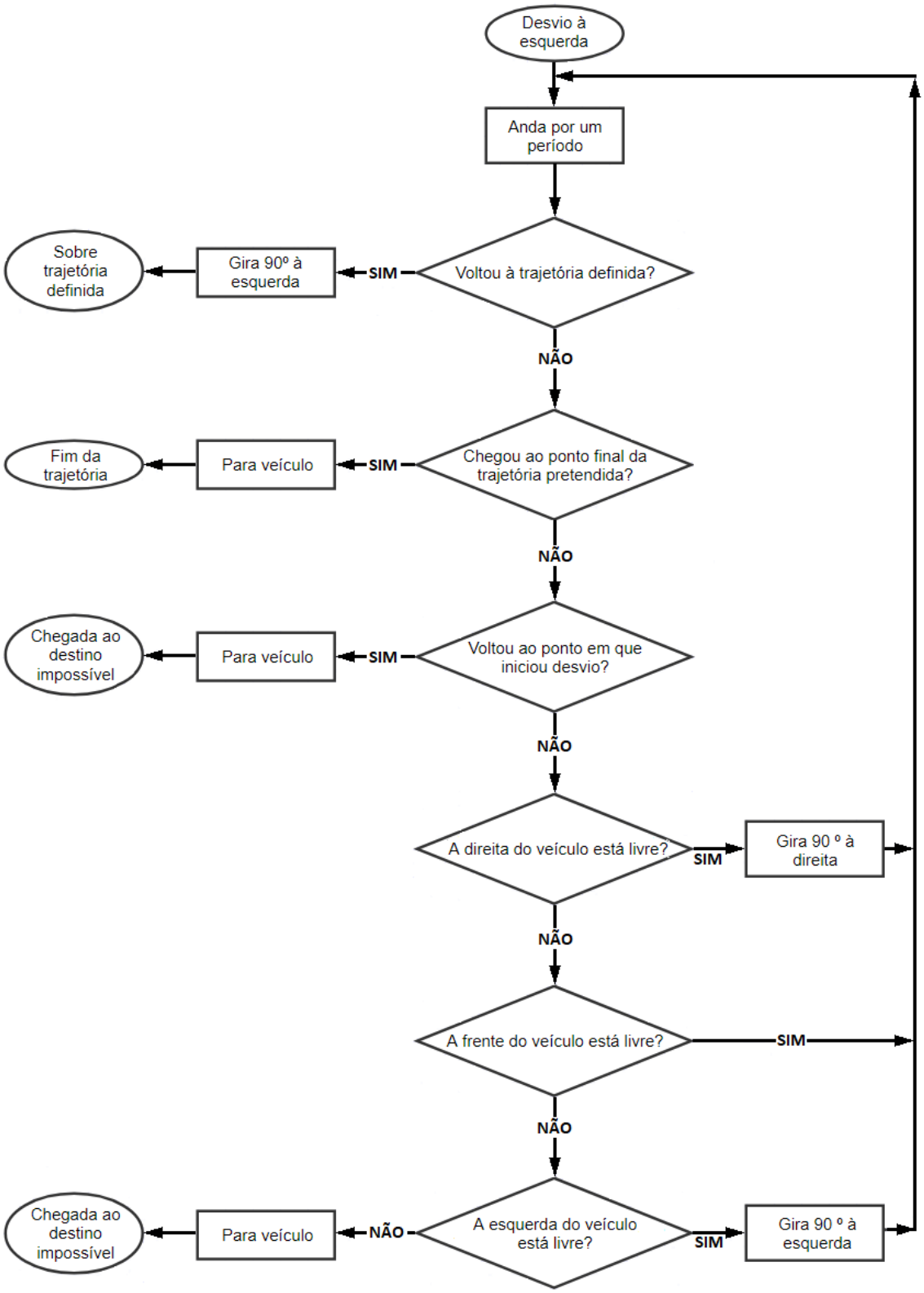
Fonte: Próprio Autor

Figura 6.12 – Fluxograma de processo desvio de obstáculo, pela direita, para navegação autônoma de robô móvel segundo algoritmo BUG 2.



Fonte: (WEI, 2015)

Figura 6.13 – Fluxograma de processo desvio de obstáculo, pela esquerda, para navegação autônoma de robô móvel segundo algoritmo BUG 2.

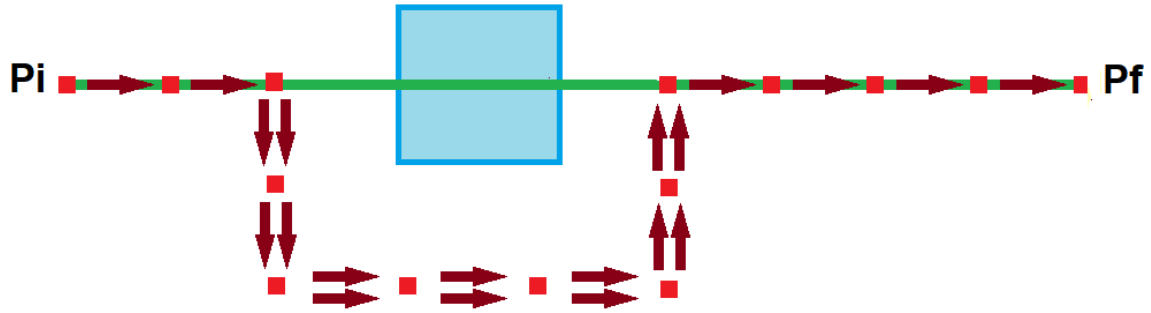


Fonte: (WEI, 2015)

O algoritmo funciona da seguinte maneira: uma trajetória para o sistema, sobre um plano que aqui será bidimensional, é definida para o módulo e esse passa a traçá-la em partes. Vários pontos sobre a trajetória são definidos e o módulo realiza sua movimentação de ponto em ponto, parando em cada um para realizar análises referentes a chegada ao destino e presença de obstáculos. Para o primeiro caso o algoritmo estabelece a parada do robô e, consequentemente, o fim do experimento, para o segundo caso o sistema escolhe um lado (direito ou esquerdo) para dar início a uma segunda análise referente a obstáculos e, assim, decidir por que lado seguir. Escolhendo uma direção, o módulo realiza um giro de 90 graus sobre o próprio eixo para o lado escolhido e passa a manobra de desvio, também realizada ponto a ponto.

Para a manobra de desvio, a cada ponto análises como chegada ao destino, retorno à trajetória original ou retorno ao ponto de partida de desvio são realizadas. Para a primeira, o algoritmo estabelece a parada do robô e, consequentemente, o fim do experimento, para o segundo caso o módulo volta a traçar a trajetória inicial ponto a ponto e, para o terceiro caso, o algoritmo estabelece a parada do módulo e o fim do experimento se dá sem sucesso. Um diagrama do trajeto a ser executado pelo sistema diante de um obstáculo simples, segundo esse algoritmo pode ser visto na Figura 6.14 a seguir.

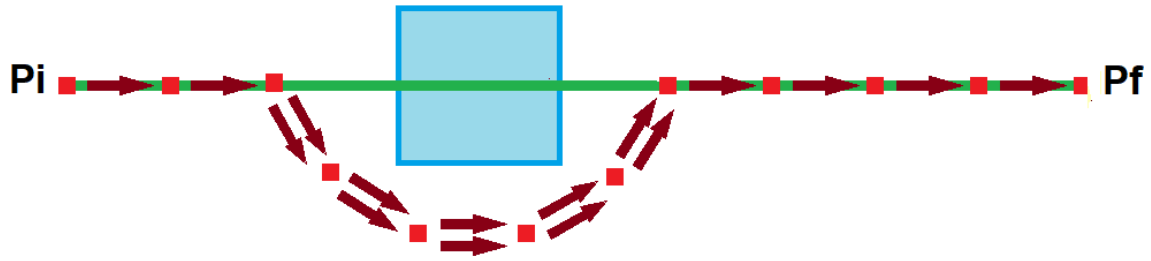
Figura 6.14 – Trajeto de desvio proposto por algorítmo BUG 2 com desvio manobras de 90 graus



Fonte: Próprio Autor

Para a segunda proposta de algoritmo BUG 2, o desvio do obstáculo se dá de forma que o módulo trace uma semi circunferência em torno do objeto detectado, obedecendo limites de distâncias estipulados pelo desenvolvedor. Aqui o limite foi estipulado em 20 centímetros, ou seja, o módulo ao detectar um obstáculo inicia um desvio de semi circunferência, com raio estipulado em 1 metro e realiza monitoramento da distância entre robô e obstáculo. Se essa distância ficar inferior ao limite de 20 centímetros, o módulo inicia uma nova circunferência para o lado oposto do obstáculo. Um diagrama do trajeto a ser executado pelo sistema diante de um obstáculo simples, segundo esse algoritmo pode ser visto na Figura 6.15 a seguir.

Figura 6.15 – Trajeto de desvio proposto por algorítmo BUG 2 com desvio em semi circunferência



Fonte: Próprio Autor

6.4.2 Simulação de Desvio com Obstáculo Simples

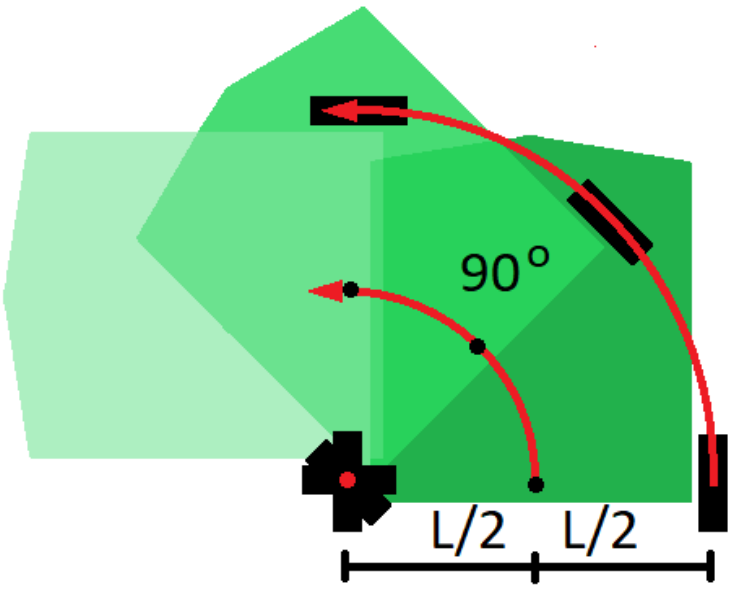
Utilizando o mesmo simulador de trajetórias que foi utilizado para executar retas e circunferências na seção 6.3.3 e, propondo uma trajetória em linha reta para o módulo, um obstáculo simples foi suposto após 2,1m de trajetória.

Sem supostos obstáculos à esquerda do módulo e supondo que esse caminho seja o mais curto, pelo fluxograma da Figura 6.11 o sistema deve realizar uma curva de 90 graus para essa direção para daí dar início à manobra de desvio.

O obstáculo suposto na trajetória foi pontual, ou seja, está alocado cerca de 2,1 m do ponto de partida do módulo e não possui extremos laterais. Dessa forma, o módulo apenas o detecta e traça o desvio como é descrito pelo fluxograma da Figura 6.12.

Supondo, em linha reta, que o módulo está a uma velocidade de 50 RPM aproximadamente, por (6.8) explicitada no Capítulo anterior desse documento, tem-se que essa velocidade em escala linear é equivalente a 260 mm/s. Se o objeto está alocado a 2,1 m do ponto de partida, levando em consideração que a detecção se dá aproximadamente 0,6 m antes do ponto de alocação do obstáculo, como explicado na seção anterior desse Capítulo, o módulo inicia seu processo de desvio após percorrer 2,09 m, aproximadamente, ou 4 segundos após iniciar sua trajetória.

Figura 6.16 – Esquema visual de processo de curva a ser realizada em pelo Módulo quando esse sen encontra sobre desvio.



Fonte: Próprio Autor

O desvio se inicia com o processo de giro, sobre o próprio eixo, do módulo para a esquerda, como mostra a Figura 6.16. O módulo possui uma largura de 280 mm, que também será o valor do diâmetro da circunferência que irá executar. Sabe-se, também, que o robô deve percorrer 1/4 de uma circunferência para percorrer o ângulo de 90 graus pretendido e, dessa forma, tem-se que a distância a ser percorrida pelo módulo em escala linear é a representada por:

$$d = \frac{1}{4}2\pi r = \frac{1}{4}2\pi 140 = 219,91mm. \tag{6.11}$$

Supondo um tempo para execução da curva de 2 segundos e levando em conta o valor da trajetória a ser percorrida pelo módulo fornecida por (6.11), tem-se que o módulo deve possui uma velocidade linear de aproximadamente 110 mm/s.

Como o robô deve girar sobre o próprio eixo, uma das rodas deve sempre permanecer parada enquanto a outra se movimenta. Assim, através de (6.9) e (6.10) tem-se que, para o módulo executar uma curva à esquerda com velocidade linear de 110 mm/s, a roda esquerda deve ter velocidade de 0 rpm, aproximadamente, enquanto que a roda direita deve possuir velocidade de 10 rpm.

Finalmente, tendo todas as velocidades de trajetória em linha reta e desvio definidas e supondo um obstáculo pontual há uma distância de aproximadamente 2,1 m do ponto de partida do robô, as velocidades lineares para cada instante de tempo foram alocadas dentro do simulador virtual, como é mostrado na Tabela 6.3 e seus valores iniciais estipulados em: $X_i = 0$, $Y_i = 0$, $\theta_i = 0$, e $L = 280\ mm$ a fim de se visualizar se o desvio se dará de maneira correta com as velocidades calculadas.

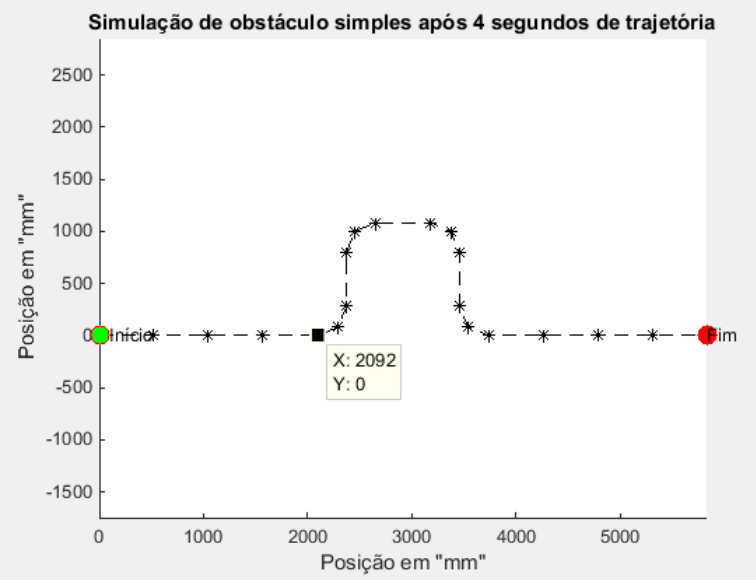
Tabela 6.3 – Valores de velocidades determinadas para simulação de desvio de obstáculo simples pela esquerda, com auxílio de simulador de trajetória, sobre o módulo robótico desenvolvido nessa pesquisa.

Vd (mm/s)	Ve (mm/s)	t (s)
523	523	4
110	0	2
523	523	1
0	110	2
523	523	1
0	110	2
523	523	1
110	0	2
523	523	4

Fonte:Próprio Autor

Com as velocidades e os intervalos de tempo estabelecidos por cálculos alocados no simulador, o traçado executado virtualmente pelo sistema robótico em quartão nessa pesquisa pode ser visto através da Figura 6.17.

Figura 6.17 – Simulação Desvio à Esquerda para algorítmo BUG 2 sobre robô desenvolvido nesse trabalho.

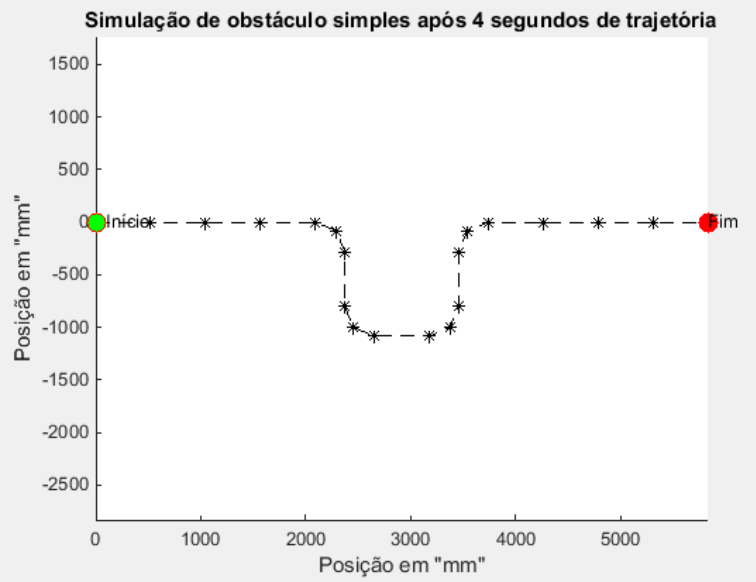


Fonte: Próprio Autor

É possível observar que o desvio se inicia realmente após 2,09 m de trajetória, como era o previsto e que o mesmo realiza com sucesso um desvio simples, girando inicialmente 90 graus a esquerda, avançando uma célula para tomada de nova decisão, girando à direita devido ao fato de que foi suposto um obstáculo pontual. Portanto o mesmo já não é mais detectado após o avanço de célula, andando mais uma célula para terminar o processo de desvio, girando novamente à direita a fim de voltar a trajetória proposta e, por fim, girando a esquerda reestabelecendo a direção para terminar seu percurso original.

O desvio à direita se dá de maneira semelhante ao executado para a esquerda. Este, em simulação aplicada, pode ser observado pela Figura 6.18. Levam-se em conta as mesmas análises já realizadas para o desvio à direita discutidas nessa seção.

Figura 6.18 – Simulação de Desvio à Direita para algorítmo BUG 2 sobre robô desenvolvido nesse trabalho.



Fonte: Próprio Autor

Para o caso do algoritmo BUG 2 com desvio em circunferência simples, propondo a mesma trajetória em linha reta e obstáculo simples suposto após 2,1m de trajetória, supondo desvio ideal pela esquerda, inicialmente o mesmo realiza uma curva de 90 graus para essa direção, inicia o processo de semi circunferência com raio de 1 metro em torno do obstáculo até que retorne a rota pretendida. Mantendo as velocidades propostas para o primeiro sistema quando em linha reta e admitindo o mesmo estudo sobre giro de 90 graus em seu próprio eixo, a Tabela 6.4 traz os valores de velocidades aplicada à simulação a cada intervalo de tempo.

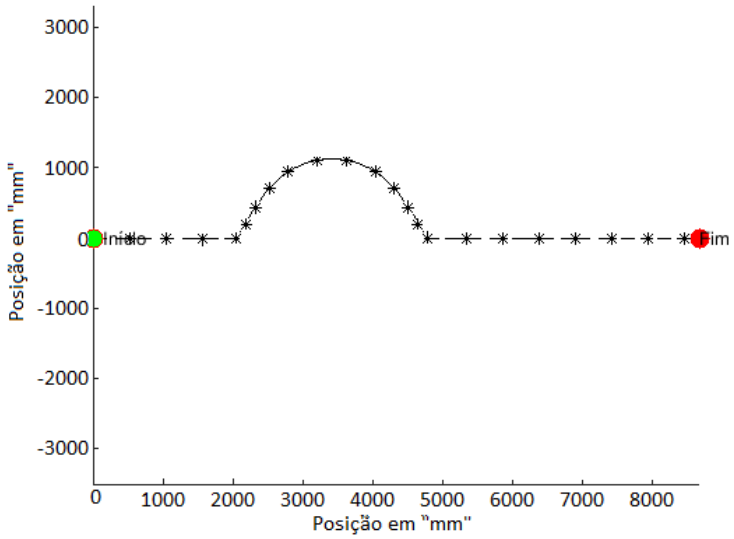
Tabela 6.4 – Valores de velocidades determinadas para simulação de desvio de obstáculo simples pela esquerda, com auxílio de simulador de trajetória, sobre o módulo robótico.

Vd (mm/s)	Ve (mm/s)	t (s)
523	523	4
110	0	2
544	523	10
110	0	2
523	523	7

Fonte:Próprio Autor

A partir dos valores da Tabela 6.4, o resultado do sistema proposto pelo algoritmo BUG 2 com desvio em semi circunferência, pode ser observado pela Figura 6.19.

Figura 6.19 – Simulação de Desvio em semi circunferência à esquerda para algorítmo BUG 2 sobre robô desenvolvido .



Fonte: Próprio Autor

Também para esse sistema, o desvio se inicia após 2,09 m de trajetória e obtém sucesso em sua execução.

6.5 Conclusão do Capítulo

O Capítulo que aqui se encerra trouxe as implementações realizadas sobre o sistema e as simulações realizadas para trajetórias de navegações propostas.

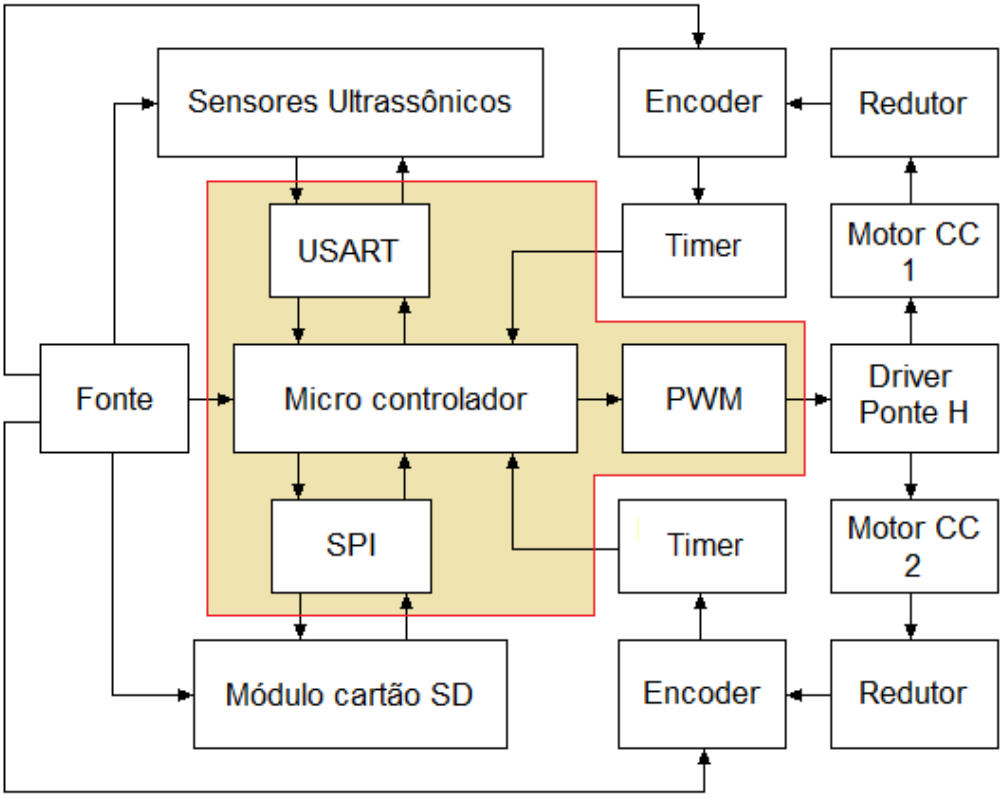
Validações de estabilidade foram realizadas e, através delas, a velocidade angular foi estabelecida como sendo o parâmetro de controle para o robô.

Com o sistema em simulação validado, o mesmo passa a ser desenvolvido em ambientes reais de navegação no próximo capítulo.

7 Implementação de Algoritmo de Navegação Autônoma em Ambientes Reais

Para que os resultados observados em ambiente de simulação pudessem ser vistos também na prática, o que é o grande objetivo desse trabalho é importante entender como se dá o funcionamento do módulo robótico construído para esse fim. Com o objetivo de facilitar a compreensão, a Figura 7.1 traz um esquemático do funcionamento do módulo que será descrito a seguir.

Figura 7.1 – Diagrama de blocos referente a montagem do módulo robótico



Fonte: Próprio Autor

Como já mencionado, níveis entre 0 e 255 são aplicados ao driver de controle PWM que é o responsável pelo controle dos motores de tração CC do robô. Os níveis são estipulados pelo programador e, através do DAC (Conversor digital/analógico) do microcontrolador é retransmitido ao driver de potência, que aplica as tensões de controle equivalentes a cada PWM enviado ao motores respectivos. Na saída de cada motor existe um sensor (encoder) que gera 48 pulsos a cada rotação do eixo. Para o robô desenvolvido aqui, a quantidade de pulsos medidos pelo encoder é contada e armazenada em intervalos de 1 s. Levando em conta, também, que o motor possui um redutor de fator redutivo igual à 84 vezes acoplado à seu eixo, a velocidade

angular do motor em questão é facilmente obtida a partir da aplicação de um fator simples sobre a leitura obtida do encoder, que é transmitida de volta ao micro controlador pela entrada ADC (Conversor Analógico/Digital). Esse fator de conversão pode ser observado na Equação a seguir:

$$\omega(\text{rpm}) = \text{leitura(encoder)}\frac{60s}{48 \times 84}. \tag{7.1}$$

A fim de conseguir captar os dados de posicionamento do módulo e visualizar a trajetória real percorrida pelo mesmo, os dados de sua velocidade angular a cada segundo foram medidos e alocados em um cartão micro SD através de uma comunicação SPI (*Serial Peripheral Interface*).

Tendo estudado o comportamento do sistema e estabelecido a forma de captura de dados, aplicou-se ao código de controle entradas equivalente as velocidades simuladas, tanto para reta quanto para circunferência para um tempo de percurso também igual, ou seja, 20 segundos e os dados captados foram depois enviados ao Matlab e aplicados ao simulador virtual a fim de que se pudesse observar, de forma gráfica, a trajetória percorrida pelo módulo. Os resultados obtidos serão evidenciados e discutidos no Capítulo 8.

7.1 Confiabilidade de Sensor Ultrassônico

O sensor utilizado no módulo robótico em discussão nessa pesquisa é o HC-SR04 e as especificações gerais desse já foram discutidas no Capítulo 5.

O HC-SR04 possui como resolução 0,003 metro em sua capacidade de medida, segundo seu *datasheet*. A fim de confirmar essa informação e estabelecer a confiabilidade das medidas estabelecidas pelo módulo, distâncias entre 0,1 e 1 metro foram estabelecidas e aplicadas nas medições de 8 sensores diferentes. Os resultados desse teste podem ser observados na Tabela 7.1.

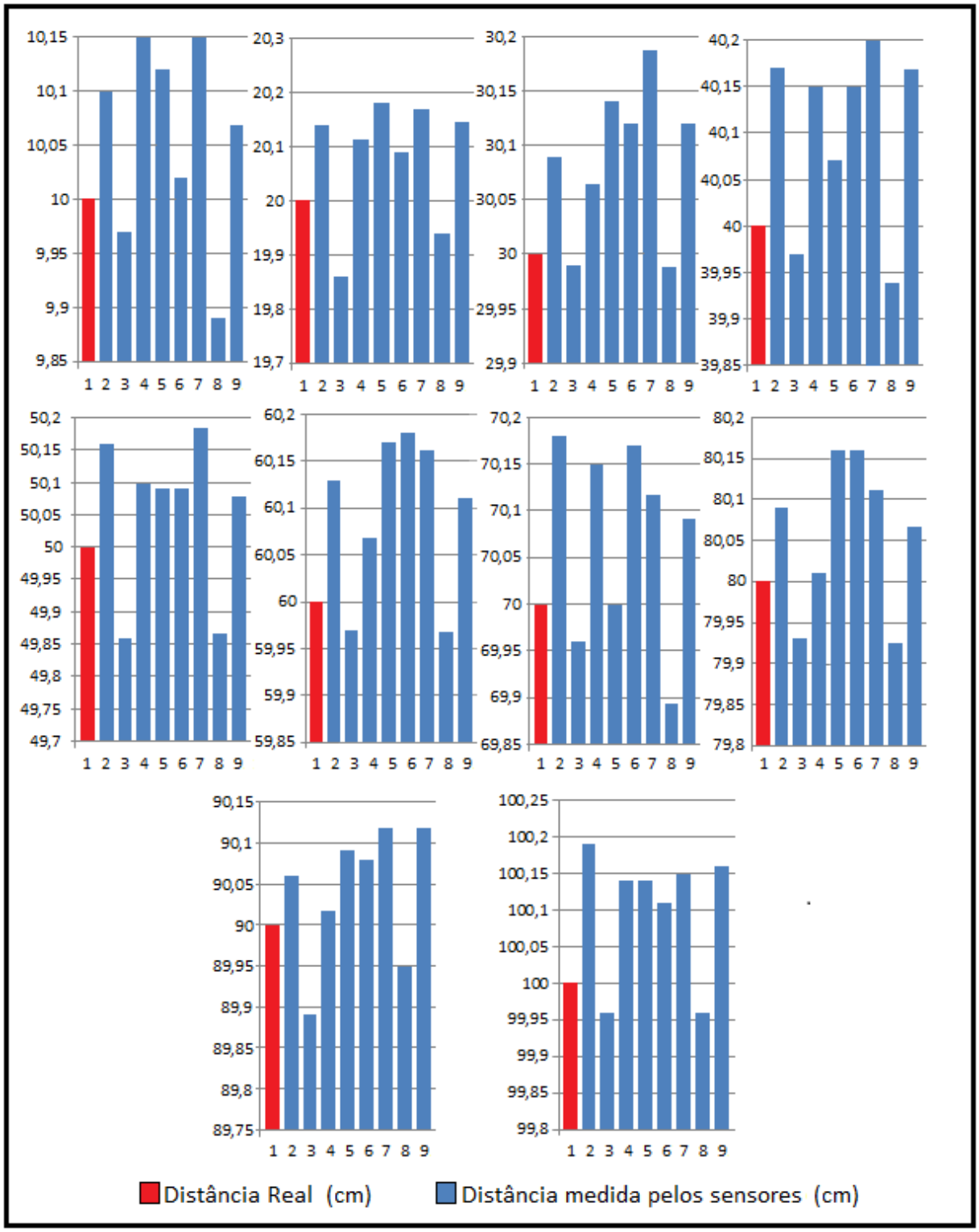
Tabela 7.1 – Medições de diferentes distâcias para 8 sensores ultrassônicos para validação de suas especificações.

	SENSOR 1	**SENSOR 2**	**SENSOR 3**
Distância Real	**Distância Medida**	**Distância Medida**	**Distância Medida**
10 cm	10,1 cm	9,97 cm	10,15 cm
20 cm	20,14 cm	19,86 cm	20,11 cm
30 cm	30,09 cm	29,99 cm	30,07 cm
40 cm	40,17 cm	39,97 cm	40,15 cm
50 cm	50,16 cm	49,86 cm	50,10 cm
60 cm	60,13 cm	59,97 cm	60,07 cm
70 cm	70,18 cm	69,96 cm	70,15 cm
80 cm	80,09 cm	79,93 cm	80,01 cm
90 cm	90,06 cm	89,89 cm	90,02 cm
100 cm	100,19 cm	99,96 cm	100,14 cm
	SENSOR 4	**SENSOR 5**	**SENSOR 6**
Distância Real	**Distância Medida**	**Distância Medida**	**Distância Medida**
10 cm	10,12 cm	10,02 cm	10,15 cm
20 cm	20,18 cm	20,09 cm	20,17 cm
30 cm	30,14 cm	30,12 cm	30,18 cm
40 cm	40,07 cm	40,15 cm	40,20 cm
50 cm	50,09 cm	50,09 cm	50,18 cm
60 cm	60,17 cm	60,18 cm	60,16 cm
70 cm	70 cm	70,17 cm	70,12 cm
80 cm	80,16 cm	80,16 cm	80,11 cm
90 cm	90,09 cm	90,08 cm	90,12 cm
100 cm	100,14 cm	100,11 cm	100,15 cm
	SENSOR 7	**SENSOR 8**	
Distância Real	**Distância Medida**	**Distância Medida**	
10 cm	9,88 cm	10,07 cm	
20 cm	19,94 cm	20,14 cm	
30 cm	29,98 cm	30,13 cm	
40 cm	39,94 cm	40,17 cm	
50 cm	49,87 cm	50,08 cm	
60 cm	59,97 cm	60,11 cm	
70 cm	69,89 cm	70,09 cm	
80 cm	79,93 cm	80,07 cm	
90 cm	89,95 cm	90,12 cm	
100 cm	99,96 cm	100,16 cm	

Fonte: Próprio Autor

Para melhor visualização, os dados fornecidos pela Tabela 7.1 foram alocados em gráficos que estão evidenciados pela Figura 7.2. Nesses gráficos, em cor vermelha está a medida real estabelecida para teste, representada pelo número 1 e, em azul, as medidas obtidas por cada sensor ultrassônico em utilização.

Figura 7.2 – Gráficos referentes às distâncias reais em comparação com distâncias medidas por 8 sensores ultrassônicos que são utilizados no robô desenvolvido.



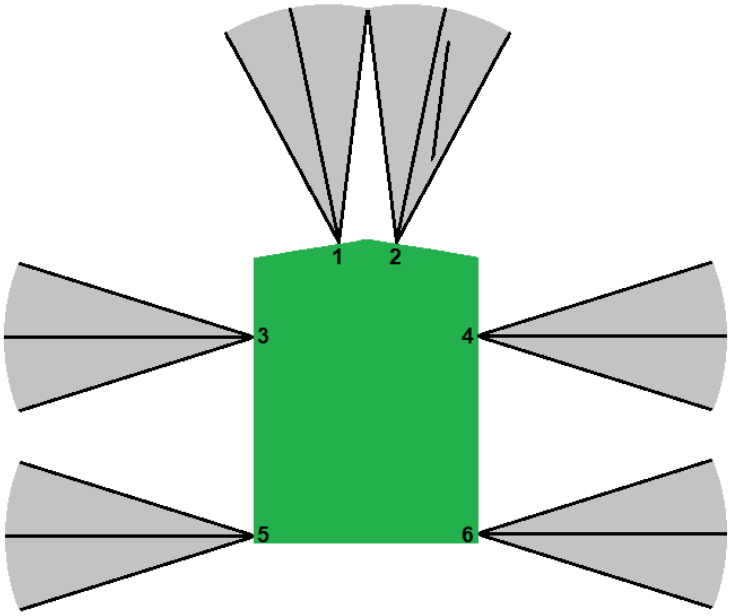
Fonte: Próprio Autor

O erro máximo encontrado para todas as medidas realizadas, considerando todos os sensores testados foi de 0,19 cm, ou 0,0019 m, o que é um valor muito pequeno ao que se refere

ao sistema como um todo. Dessa maneira, os sensores são validados para utilização no módulo robótico.

Tendo a confiabilidade de medição validada, os sensores foram alocados ao módulo robótico. A alocação, inicialmente, se deu de forma que sua leitura cobrisse a área frontal do módulo por completo e partes das laterais, a fim de se detectar obstáculos em sua rota. Dessa maneira, seis sensores foram alocados ao módulo e estes são representados na Figura 7.3 pelos números 1, 2, 3, 4, 5 e 6.

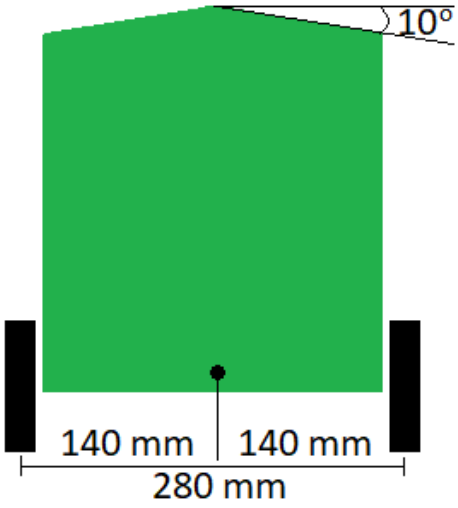
Figura 7.3 – Visulização gráfica do posicionamento e área de cobertura dos seis sensores ultrassônicos acoplados ao módulo robótico com objetivo de detecção de obstáculos.



Fonte: Próprio Autor

Sabendo que a abertura de medição do HC-SR04 é de 15 graus a partir de seu centro, ou seja, 30 graus no total, os sensores da parte frontal do módulo foram alocado com uma angulação de 10 graus em relação ao eixo do módulo, como mostra a Figura 7.4. Considerando essa angulação, formou-se um espaço na parte da frente do módulo em que não há detecção de obstáculos. Para evitar que o módulo se depare com um obstáculo e não consiga detecta-lo, um raio superior a esse espaço foi estabelecido como distância inicial para a realização das medições e detecções de corpos.

Figura 7.4 – Visualização gráfica superior do módulo robótico desenvolvido e indicação de sua angulação frontal devido ao acoplamento e área de cobertura dos sensores ultrassônicos utilizados.

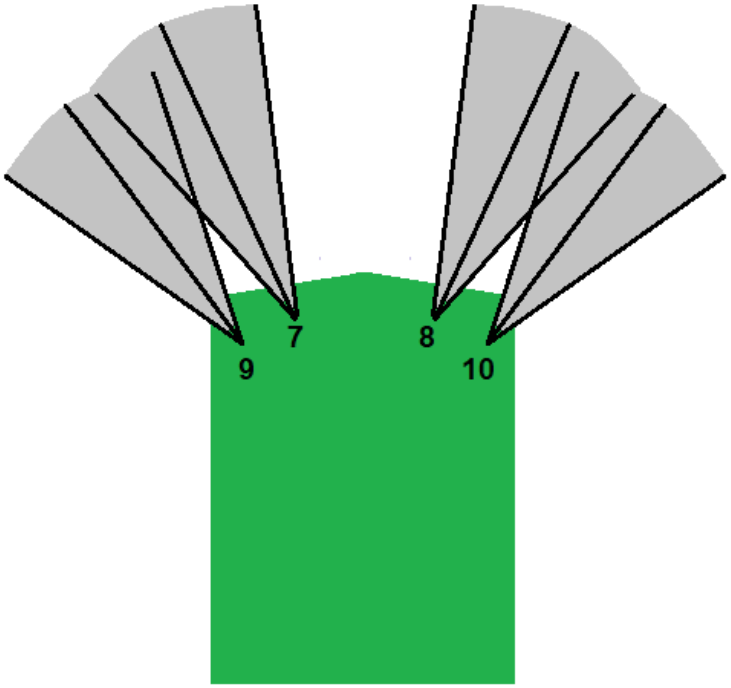


Fonte: Próprio Autor

Foi determinado que o módulo, quando em linha reta, deverá operar a uma velocidade de 50 rpm, ou seja, aproximadamente 260 mm/s, terá as velocidades medidas a cada segundo. Alocando dois sensores ultrassônicos na parte frontal do módulo, cada um posicionado a 5 cm do centro do mesmo e com angulação de 10 graus em relação ao eixo do robô, tem-se que, a partir das leis básicas da geometria, o ponto de encontro das áreas de detecção dos dois sensores se dá ha uma distância de aproximadamente 40 cm do módulo. Levando em conta as duas distâncias limites mencionadas (52 e 40 cm) a detecção de distância de objetos escolhida se deu há 60 cm do módulo, tendo assim uma margem de erro de 8 cm.

Por último, mais quatro sensores foram alocados na parte superior do módulo, com angulação de 20, 45, 135 e 160 graus em relação ao eixo do robô, com finalidade de detecção de melhor direção a seguir quando um obstáculo for detectado. Esses sensores e sua alocação em relação ao robô são vistos pela Figura 7.5 a seguir e estão representados pelos números 7, 8, 9 e 10, respectivamente.

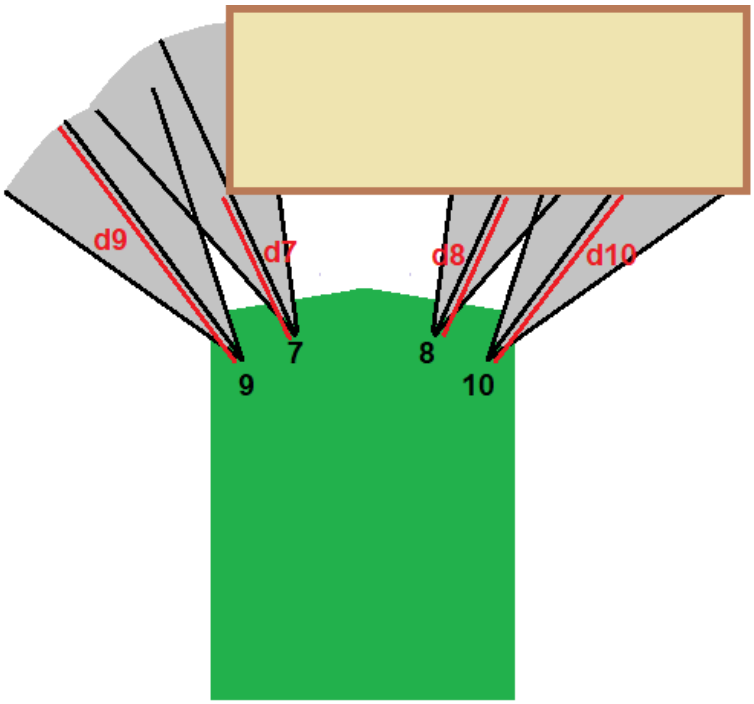
Figura 7.5 – Visualização gráfica do posicionamento e área de cobertura dos quatro sensores ultrassônicos acoplados ao módulo robótico com objetivo de tomada de decisão sobre rota a seguir.



Fonte: Próprio Autor

Dessa maneira, quando um objeto é detectado no caminho do módulo e o mesmo passa a iniciar uma manobra de desvio, a direção a seguir (direita ou esquerda) é tomada a partir das medições realizadas por esses sensores. Os valores medidos pelos sensores 7 e 9 (d7 e d9) são somados e comparados com a soma dos valores medidos por 8 e 10 (d8 e d10). Se o valor obtido pela primeira soma for superior ao valor obtido pela segunda, o robô realiza o desvio pela esquerda e vice-versa. Um esquema dessa situação pode ser observado pela Figura 7.6.

Figura 7.6 – Tomada de decisão sobre direção a seguir em caso de detecção de obstáculo.



Fonte: Próprio Autor

7.2 Implementação de Desvio em Ambiente Real

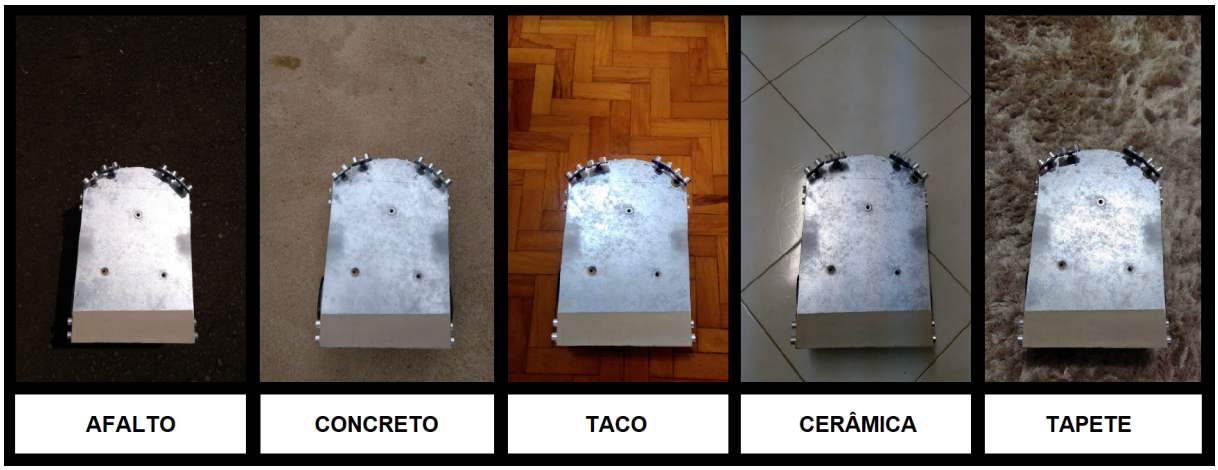
Com o sucesso obtido em simulação, os algoritmos de desvio BUG 2, com desvio em manobras de giro de 90 graus e o em semi circunferência, passaram a ser implementados na prática para o robô físico, em ambiente real de execução de trajetórias, e é esse processo que será mostrado nessa seção.

O processo de controle do módulo se deu da mesma maneira que o executado em percursos de trajetória, como é evidenciado pela seção 6.3.4 e pela Figura 7.1, referente ao Esquemático de funcionamento do robô.

O robô foi alocado em cinco superfícies diferentes para a obtenção de possíveis variações em sua trajetória, referentes as mudanças nas superfícies.

Os ambientes de teste podem ser vistos na Figura 7.7 e consistem de piso de asfalto novo, concreto, taco, cerâmica molhada e tapete.

Figura 7.7 – Ambientes de aplicação do módulo robótico.



Fonte: Próprio Autor

Todos os sistemas propostos foram aplicados em todas as diferentes superfícies e os dados foram colhidos para dez aplicações em cada situação, totalizando 200 aplicações (10 para cada superfície em trajetória de linha reta, 10 para cada superfície em trajetória de circunferência, 10 para cada superfície em trajetória de desvio com deslocamento de rotação em 90 graus e 10 para cada trajetória de desvio com deslocamento em semi circunferência).

Ao final dos testes, os dados foram aplicados em gráficos para visualização do caminho percorrido e a média para cada situação foi obtida a fim de se obter uma melhor análise de desempenho entre todas as situações estabelecidas.

7.3 Navegação em Ambiente Real

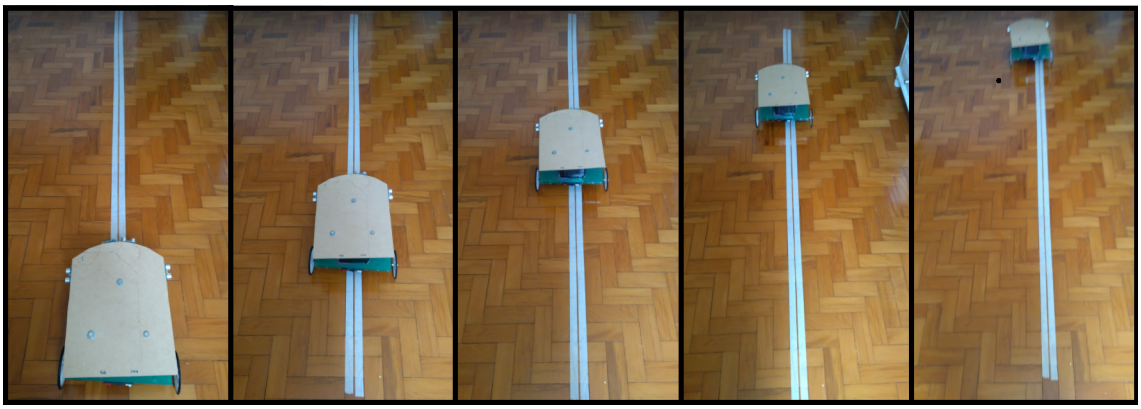
Com a implementação do código de controle para execução de rotas em linha reta e circunferência, mostrado no Capítulo 6 executado, os traçados realizados pelo módulo em um ambiente real pode ser observado.

O ambiente em questão é composto por uma área de 20 metros quadrados de terreno plano (para todas as superfícies) e, num primeiro momento, sem obstáculos em seu interior.

Inicialmente, uma trajetória em linha reta foi implementada sobre o robô, depois uma trajetória em circunferência de raio 1 m e, por fim, trajetórias com desvio de obtáculos.

Tomando como exemplo a superfície referente ao taco, os traçados executados pelo módulo com as mesmas velocidades aplicadas em simulação, podem ser observados pelas Figuras 7.8, Figura 7.9 e Figura 7.10 a seguir.

Figura 7.8 – Trajetória executada pelo módulo robótico desenvolvido, em linha reta, sobre solo de taco.

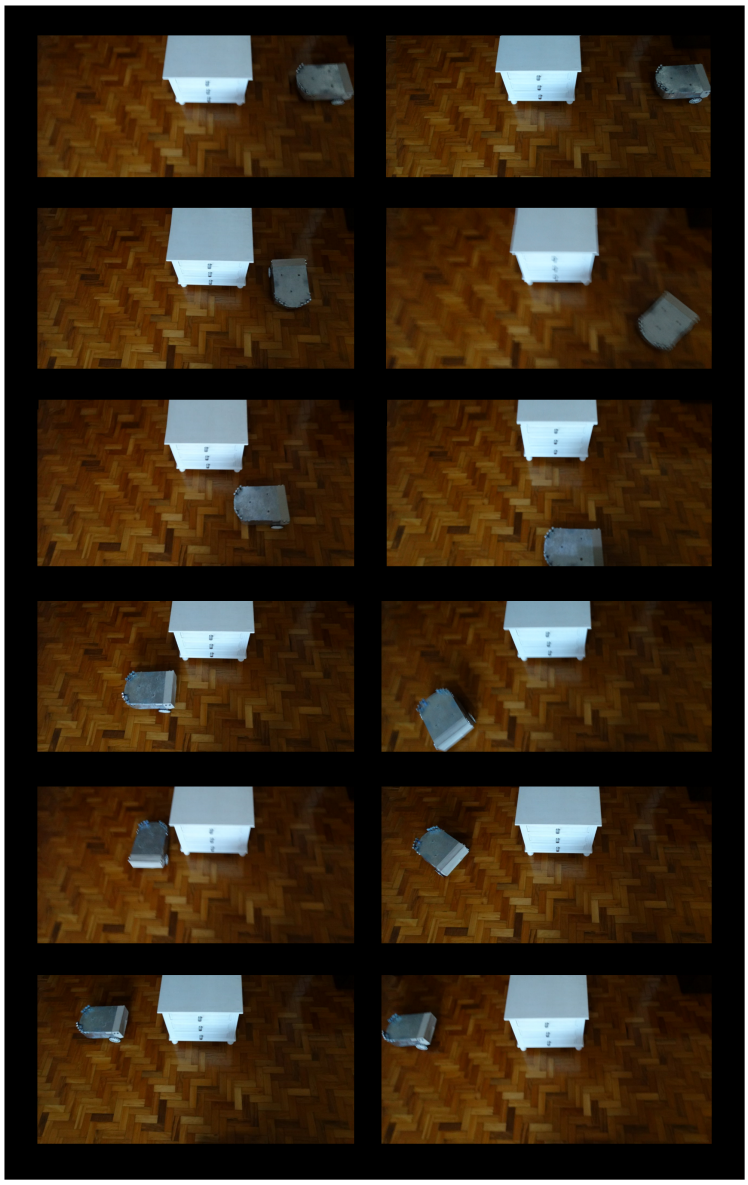


Fonte: Próprio Autor

Figura 7.9 – Trajetória executada pelo módulo robótico desenvolvido, em circunferência, sobre solo de taco.

Fonte: Próprio Autor

Figura 7.10 – Trajetórias executadas pelo módulo robótico desenvolvido, em desvio, sobre solo de taco.



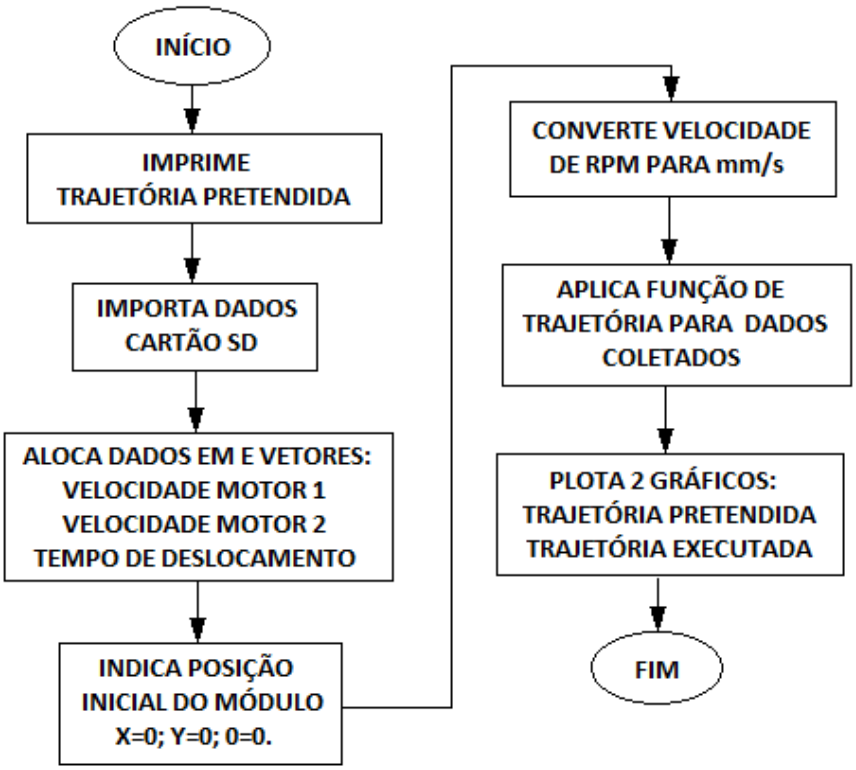
Fonte: Próprio Autor

Com o intuito de comparação entre as trajetórias pretendidas, simuladas com auxílio de um simulador virtual, conforme mostrado na seção 6.3.3 desse documento, um comparador foi criado no *software Matlab* e um fluxograma deste é mostrado na Figura 7.11.

Para se estabelecer a comparação, encoders acoplados aos motores de tração do módulo contabilizaram pulsos conforme sua resolução a cada 1 s (valor escolhido pelo projetista) e o valor medido foi transformado, dentro do próprio código, em RPM. Esses valores foram alocados dentro de um cartão micro SD acoplado ao módulo, também em intervalos de 1 s. Finalmente, os valores das velocidades de cada motor foram alocados em vetores referentes à velocidade do motor direito, MD, velocidade do motor esquerdo, ME, e tempo percorrido e esses foram importados pelo comparador que, aplicando os valores medidos no simulador

virtual, obteve o percurso real executado de forma gráfica.

Figura 7.11 – Fluxograma representando código para sistema de comparação de trajetórias.



Fonte: Próprio Autor

7.4 Conclusão do capítulo

O Capítulo que aqui se encerra trouxe o processo de simulação do algoritmo de desvio BUG 2 para o sistema robótico em estudo e a forma da implementação do mesmo sobre o módulo robótico real desenvolvido para a pesquisa.

Após a realização de simulações em ambientes virtuais e a obtenção de resultados satisfatórios para essas, o processo de implementação dos mesmos algoritmos foram evidenciados a fim de se compreender como se dá o controle em linguagem de programação para o sistema.

Com todas as simulações referentes à pesquisa realizada, o próximo Capítulo apresenta os resultados obtidos na aplicação dos sistemas em ambiente real.

8 Análise de Resultados

O presente Capítulo visa a discussão dos resultados obtidos pela pesquisa.

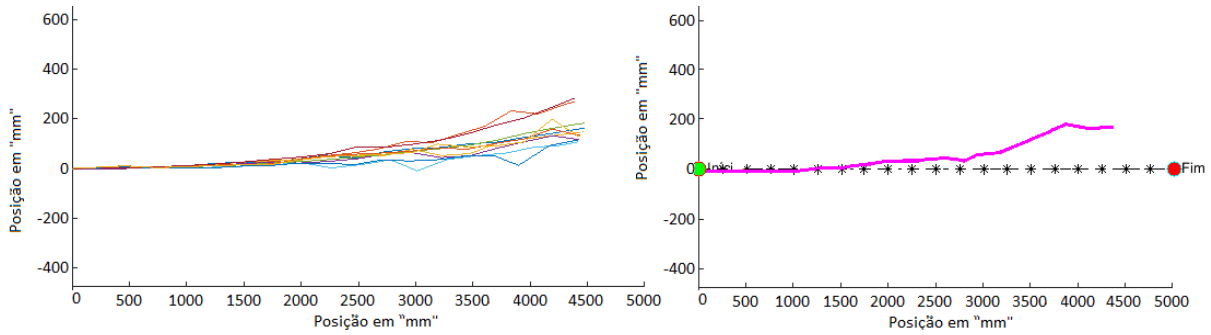
Após a obtenção do modelo matemático, validação do sistema por testes de estabilidade e simulações realizadas, algoritmos destinados à execução de rotas predefinidas e navegação com desvio de obstáculos foram implementados sobre um módulo robótico físico a fim de que os resultados obtidos em simulação pudessem ser comprovados em ambiente real de navegação. E são esses resultados que serão aqui apresentados.

Comparações entre rotas ideais de simualção e rotas traçadas em diversos ambientes reais são realizadas e os erros, em mm, entre ambas são obtidos e analisados.

8.1 Aplicação em trajetória de linha reta

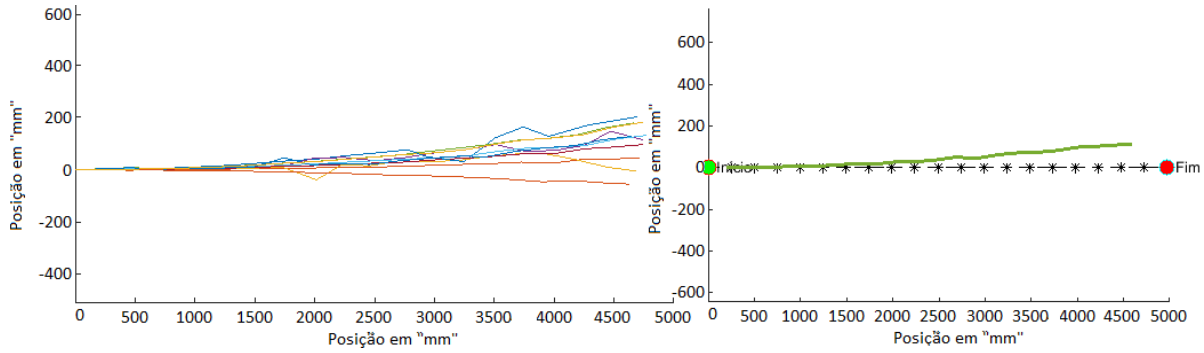
Os gráficos referentes aos resultados obtidos pelo comparador para a trajetória em linha reta nas diferentes superfícies, e suas respectivas médias são mostrados a seguir pelas Figuras 8.1, 8.2,8.3, 8.4 e 8.5.

Figura 8.1 – Visualização gráfica de trajetória executada pelo módulo, em linha reta, sobre superfície de asfalto e sua média em relação ao traçado ideal, respectivamente.



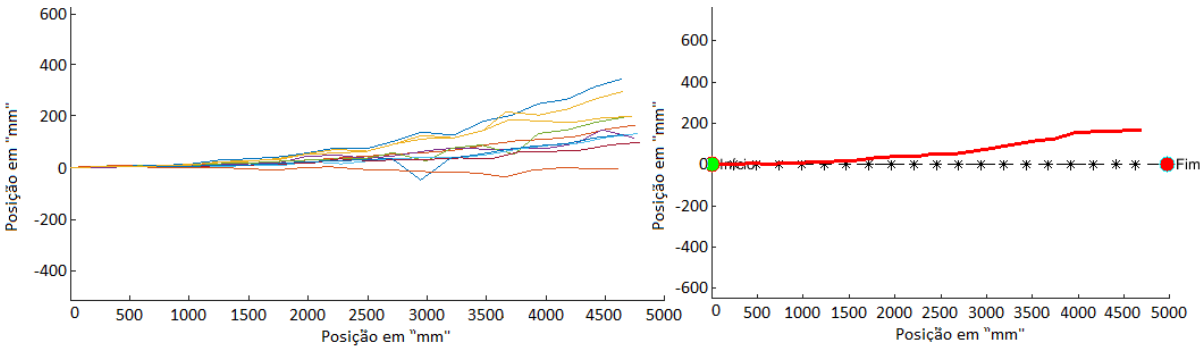
Fonte: Próprio Autor

Figura 8.2 – Visualização gráfica de trajetória executada pelo módulo, em linha reta, sobre superfície de concreto e sua média em relação ao traçado ideal, respectivamente.



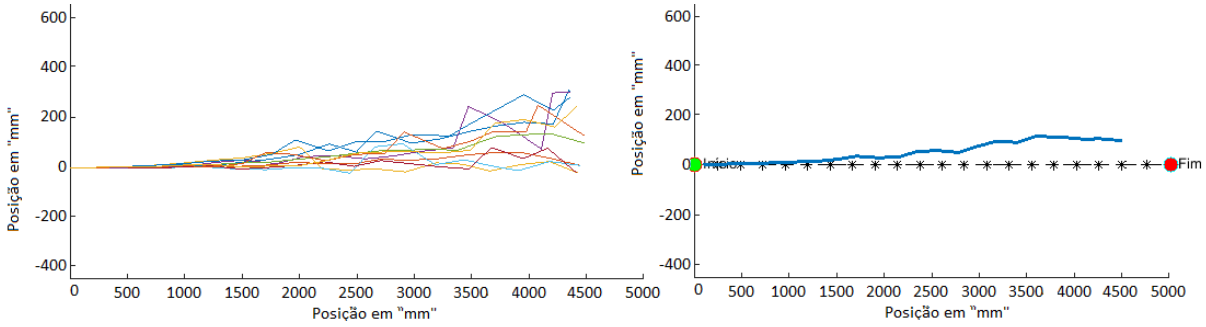
Fonte: Próprio Autor

Figura 8.3 – Visualização gráfica de trajetória executada pelo módulo, em linha reta, sobre superfície de taco e sua média em relação ao traçado ideal, respectivamente.



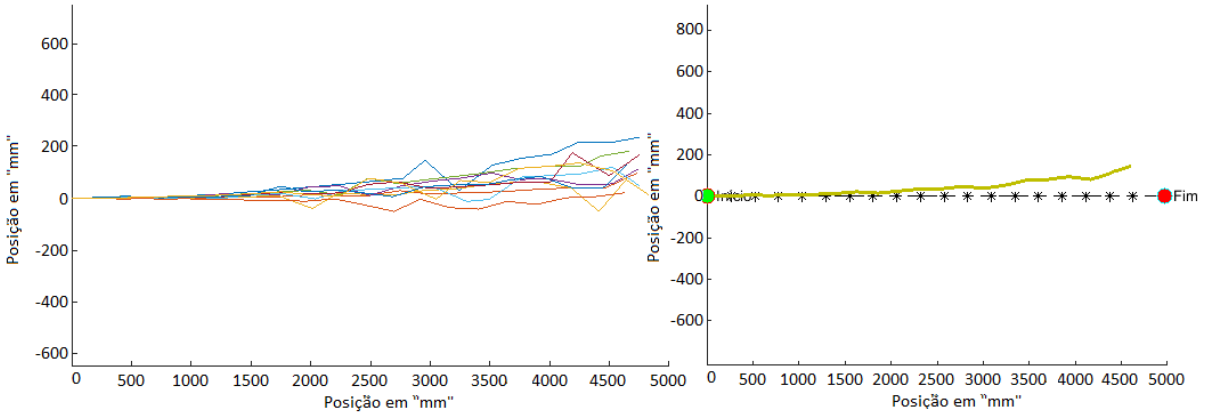
Fonte: Próprio Autor

Figura 8.4 – Visualização gráfica de trajetória executada pelo módulo, em linha reta, sobre superfície de cerâmica molhada e sua média em relação ao traçado ideal, respectivamente.



Fonte: Próprio Autor

Figura 8.5 – Visualização gráfica de trajetória executada pelo módulo, em linha reta, sobre superfície de tapete e sua média em relação ao traçado ideal, respectivamente.

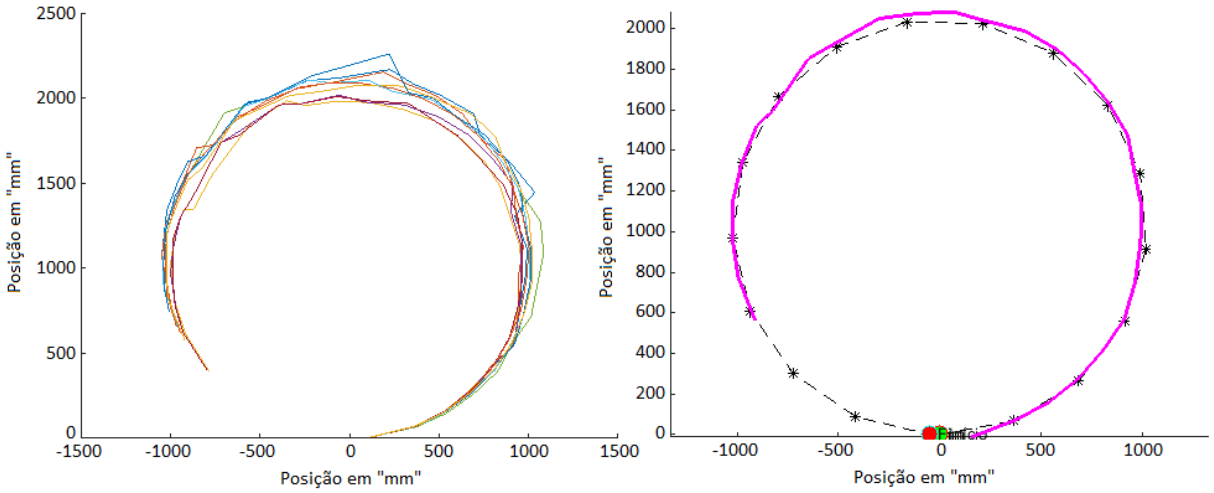


Fonte: Próprio Autor

8.2 Aplicação em trajetória de circunferência

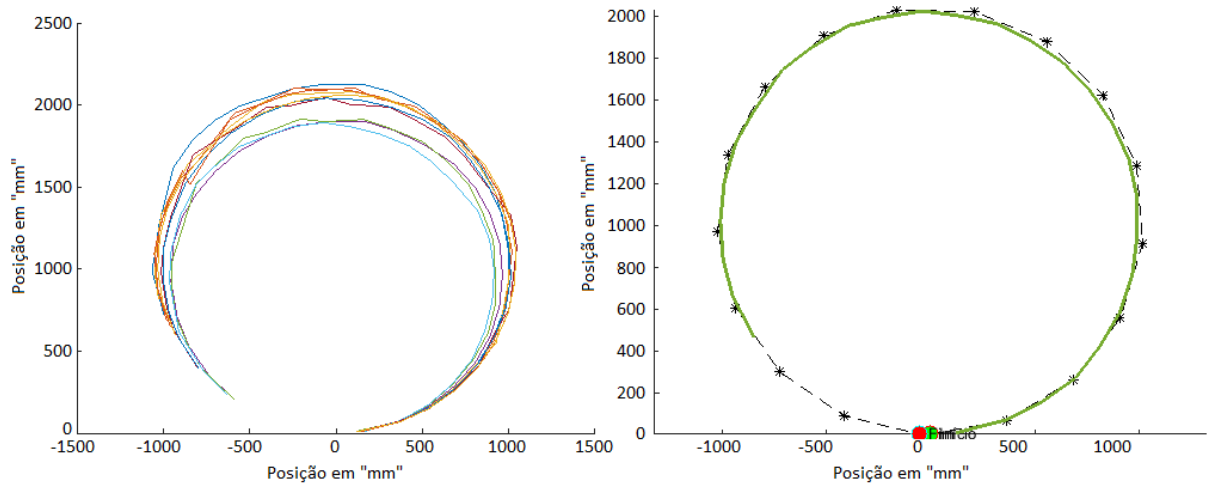
Os gráficos referentes aos resultados obtidos pelo comparador para a trajetória em circunferência com raio igual a um metro definido nas diferentes superfícies, e suas respectivas médias são mostrados a seguir pelas Figuras 8.6, 8.7,8.8, 8.9 e 8.10.

Figura 8.6 – Visualização gráfica de trajetória executada pelo módulo, em circunferência, sobre superfície de asfalto e sua média em relação ao traçado ideal, respectivamente.



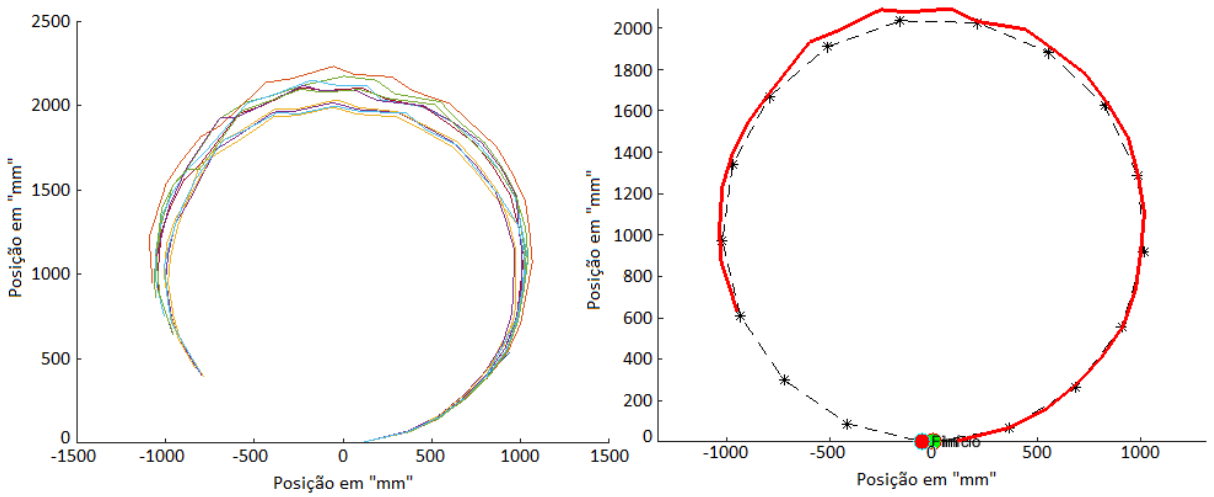
Fonte: Próprio Autor

Figura 8.7 – Visualização gráfica de trajetória executada pelo módulo, em circunferência, sobre superfície de concreto e sua média em relação ao traçado ideal, respectivamente.



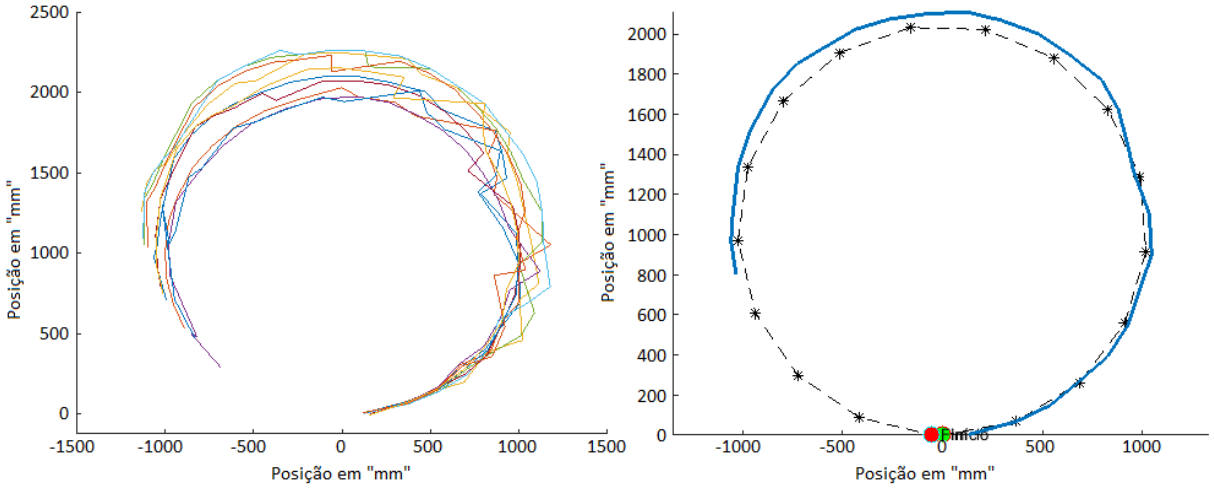
Fonte: Próprio Autor

Figura 8.8 – Visualização gráfica de trajetória executada pelo módulo, em circunferência, sobre superfície de taco e sua média em relação ao traçado ideal, respectivamente.



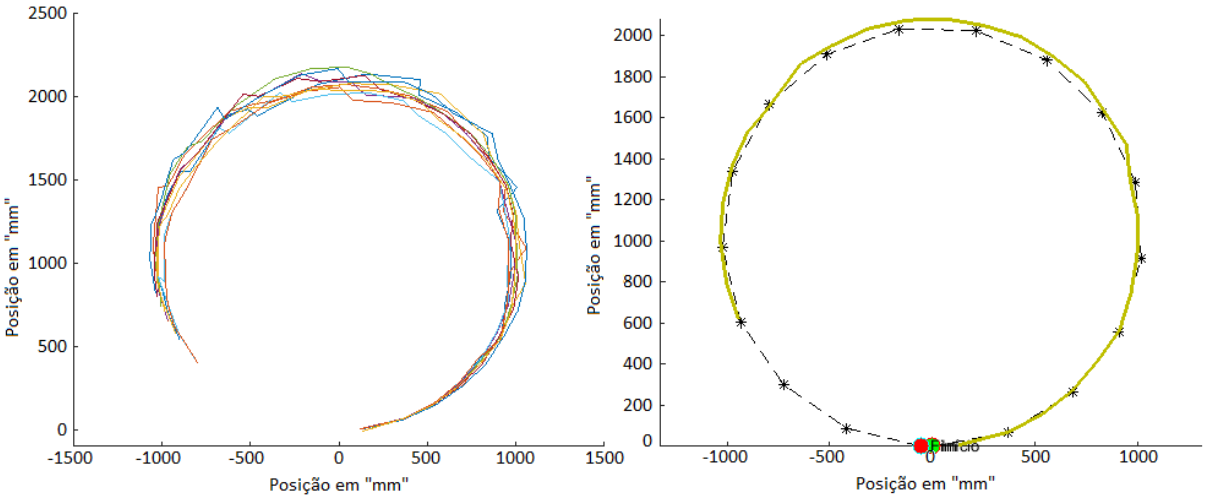
Fonte: Próprio Autor

Figura 8.9 – Visualização gráfica de trajetória executada pelo módulo, em circunferência, sobre superfície de cerâmica molhada e sua média em relação ao traçado ideal, respectivamente.



Fonte: Próprio Autor

Figura 8.10 – Visualização gráfica de trajetória executada pelo módulo, em circunferência, sobre superfície de tapete e sua média em relação ao traçado ideal, respectivamente.

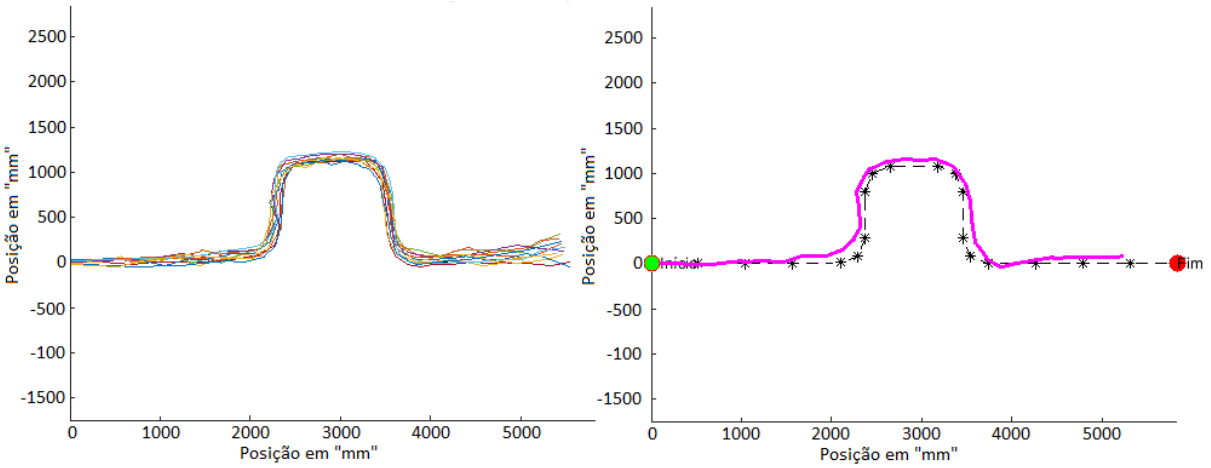


Fonte: Próprio Autor

8.3 Aplicação em trajetória de desvio BUG 2 com giro de 90 graus

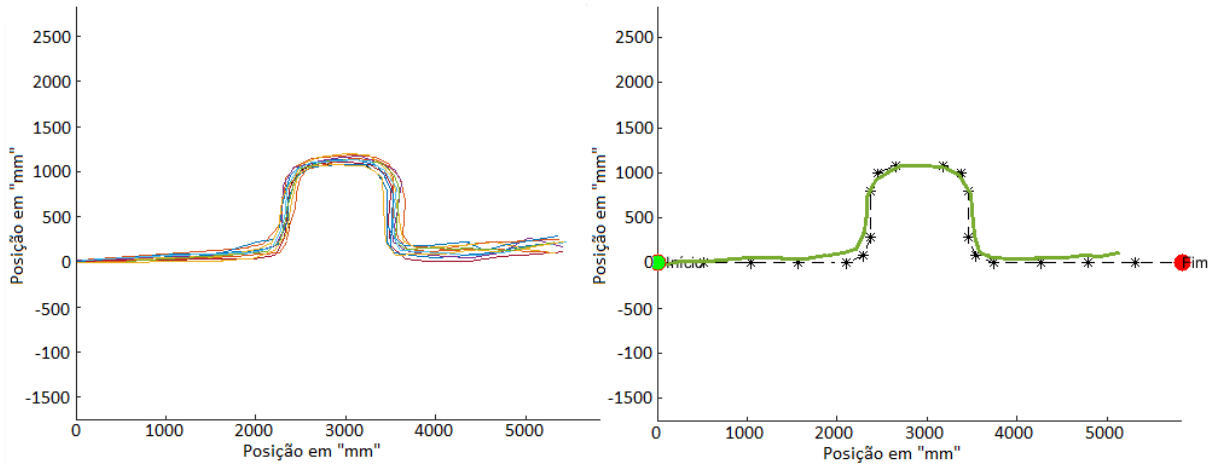
Os gráficos referentes aos resultados obtidos pelo comparador para a trajetória com detecção de obstáculo simples e manobra de desvio definida pelo algorítmo BUG 2 com desvio sobre manobras de 90 graus sobre o próprio eixo robótico, nas diferentes superfícies, e suas respectivas médias são mostrados a seguir pelas Figuras 8.11, 8.12,8.13, 8.14 e 8.15.

Figura 8.11 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com manobras de rotação em 90 graus, sobre superfície de asfalto e sua média em relação ao traçado ideal, respectivamente.



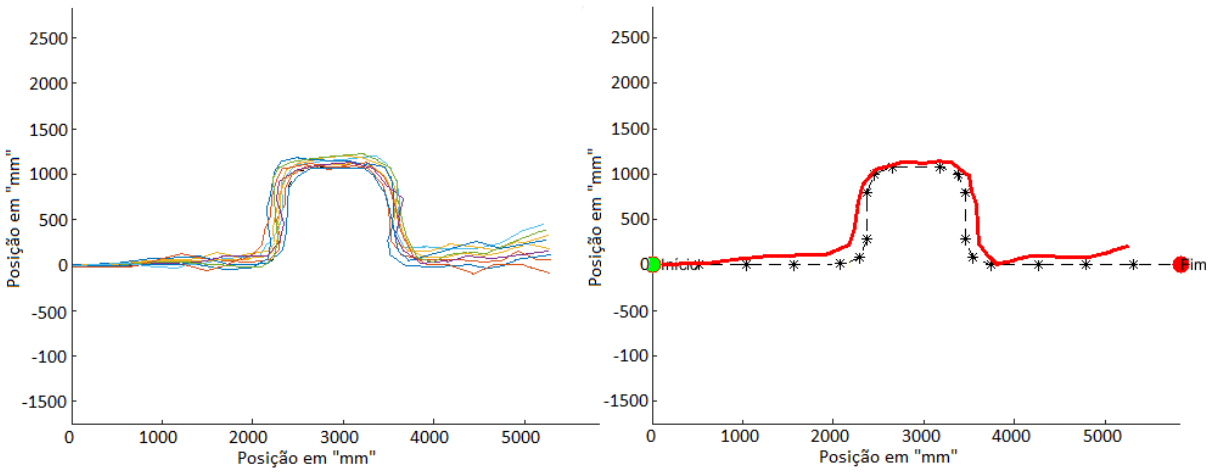
Fonte: Próprio Autor

Figura 8.12 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com manobras de rotação em 90 graus, sobre superfície de concreto e sua média em relação ao traçado ideal, respectivamente.



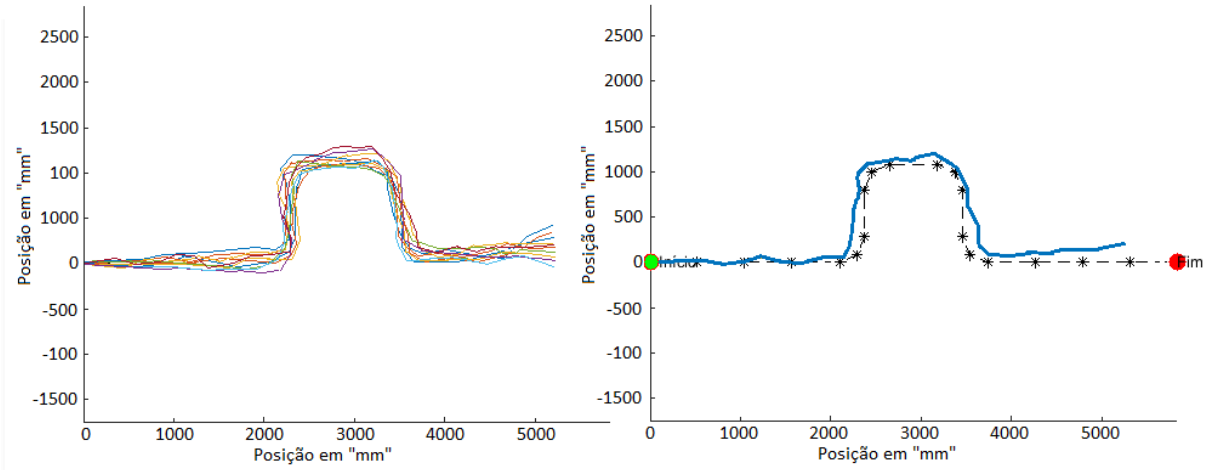
Fonte: Próprio Autor

Figura 8.13 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com manobras de rotação em 90 graus, sobre superfície de taco e sua média em relação ao traçado ideal, respectivamente.



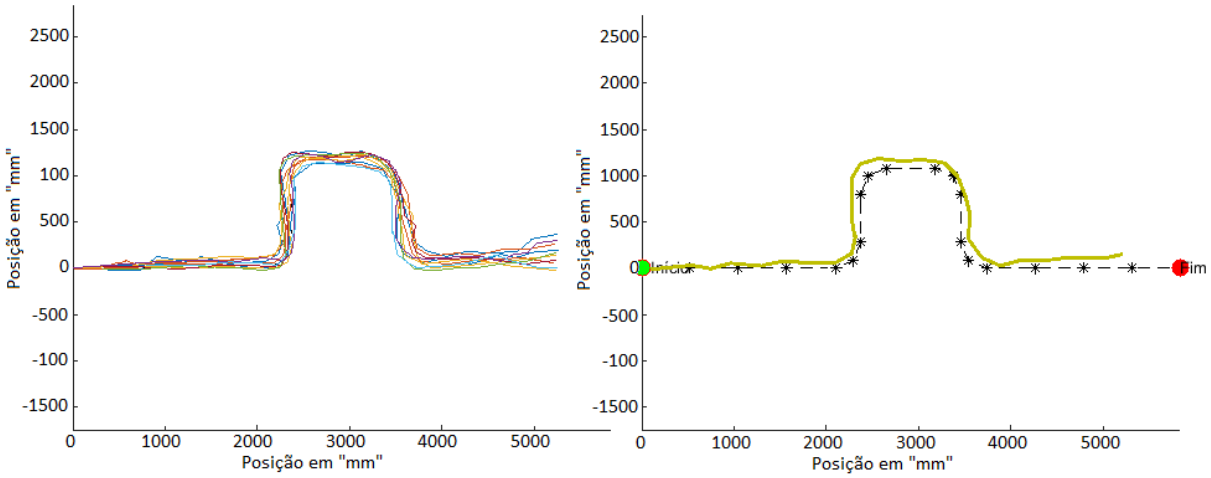
Fonte: Próprio Autor

Figura 8.14 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com manobras de rotação em 90 graus, sobre superfície de cerâmica molhada e sua média em relação ao traçado ideal, respectivamente.



Fonte: Próprio Autor

Figura 8.15 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com manobras de rotação em 90 graus, sobre superfície de tapete e sua média em relação ao traçado ideal, respectivamente.

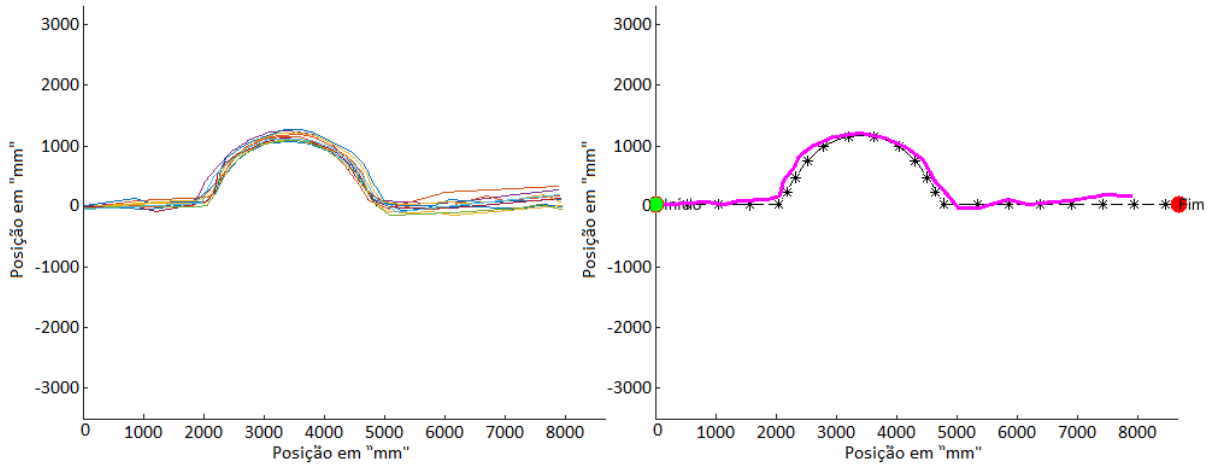


Fonte: Próprio Autor

8.4 Aplicação em trajetória de desvio BUG 2 em semi circunferência

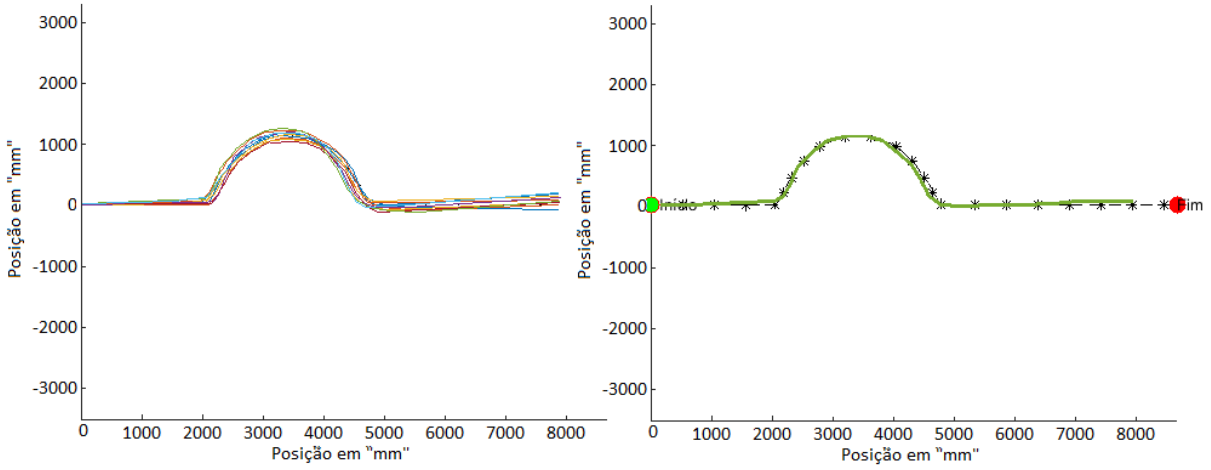
Os gráficos referentes aos resultados obtidos pelo comparador para a trajetória com detecção de obstáculo simples e manobra de desvio definida pelo algorítmo BUG 2 com desvio sobre manobras em semi circunferência, nas diferentes superfícies, e suas respectivas médias são mostrados a seguir pelas Figuras 8.16, 8.17,8.18, 8.19 e 8.20.

Figura 8.16 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com desvio sobre manobras em semi circunferência, sobre superfície de asfalto e sua média em relação ao traçado ideal, respectivamente.



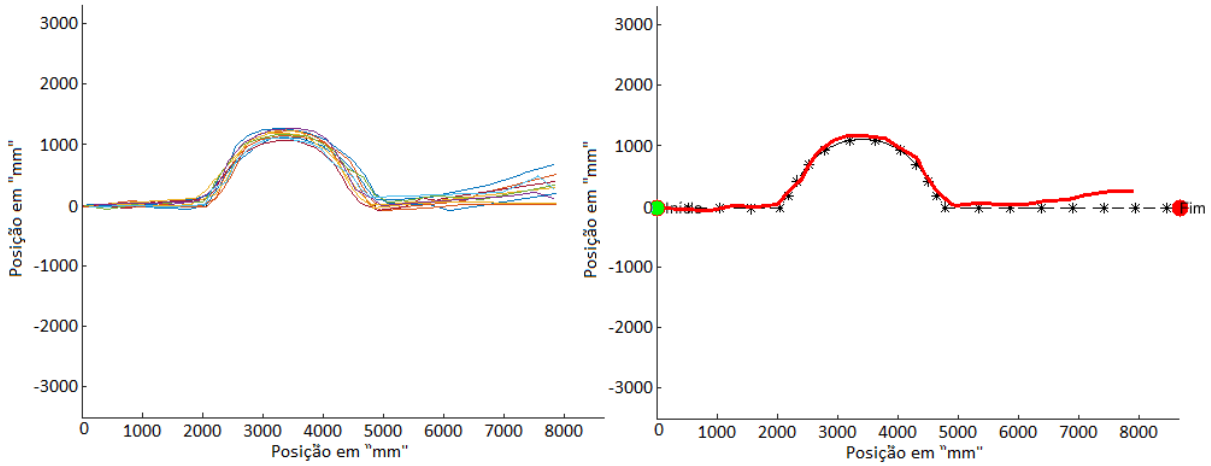
Fonte: Próprio Autor

Figura 8.17 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com desvio sobre manobras em semi circunferência, sobre superfície de concreto e sua média em relação ao traçado ideal, respectivamente.



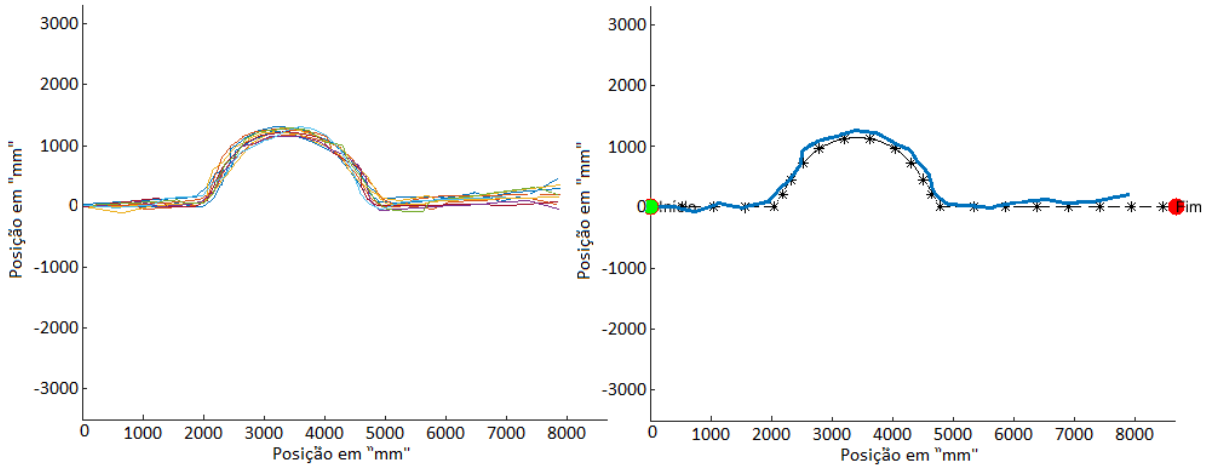
Fonte: Próprio Autor

Figura 8.18 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com desvio sobre manobras em semi circunferência, sobre superfície de taco e sua média em relação ao traçado ideal, respectivamente.



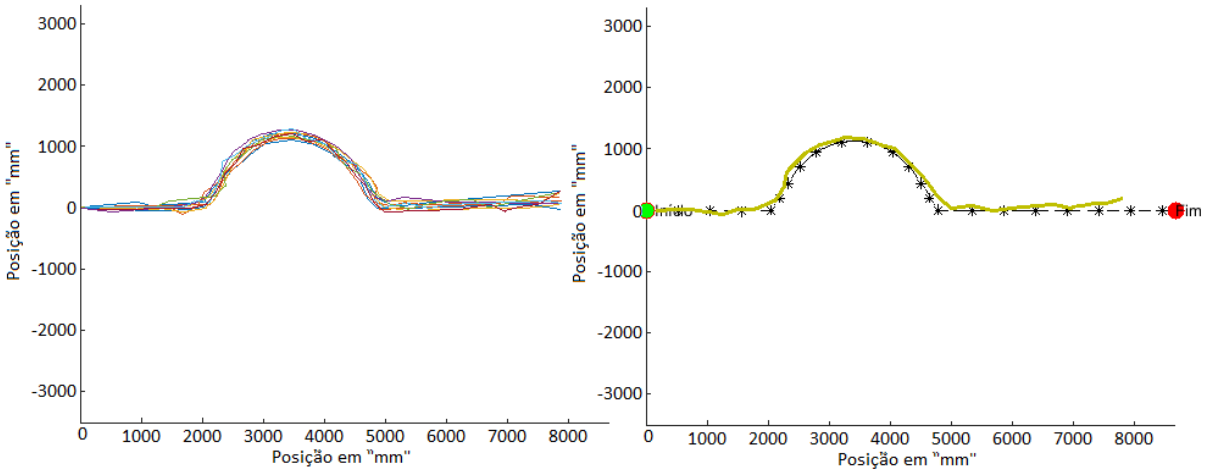
Fonte: Próprio Autor

Figura 8.19 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com desvio sobre manobras em semi circunferência, sobre superfície de cerâmica molhada e sua média em relação ao traçado ideal, respectivamente.



Fonte: Próprio Autor

Figura 8.20 – Visualização gráfica de trajetória executada pelo módulo, em desvio BUG 2 com desvio sobre manobras em semi circunferência, sobre superfície de tapete e sua média em relação ao traçado ideal, respectivamente.

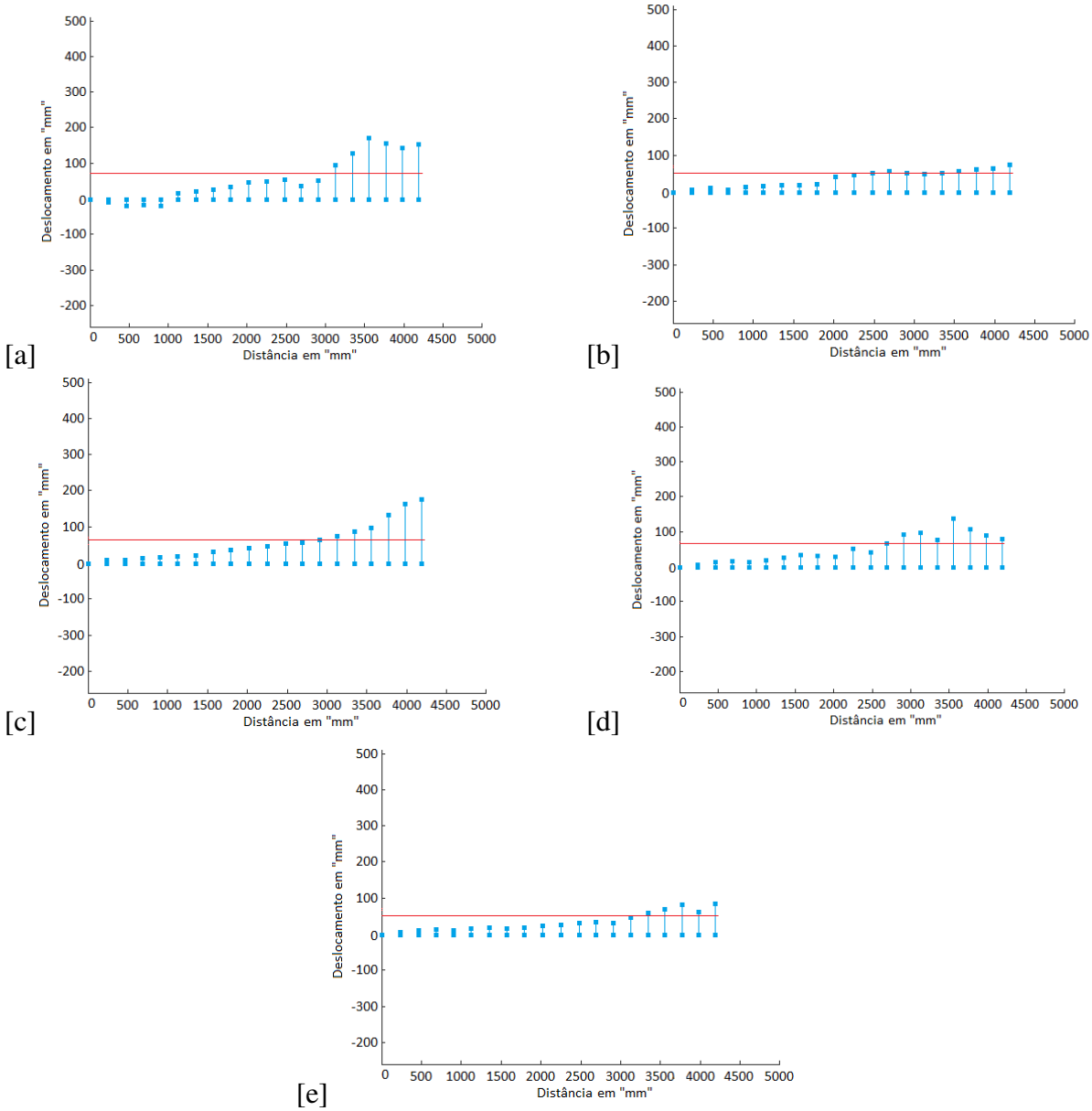


Fonte: Próprio Autor

Pode-se observar, para todas as rotas propostas, que na prática o módulo sofre certo desvio em sua rota ideal, dada pela simulação e, também, não completa o percurso total.

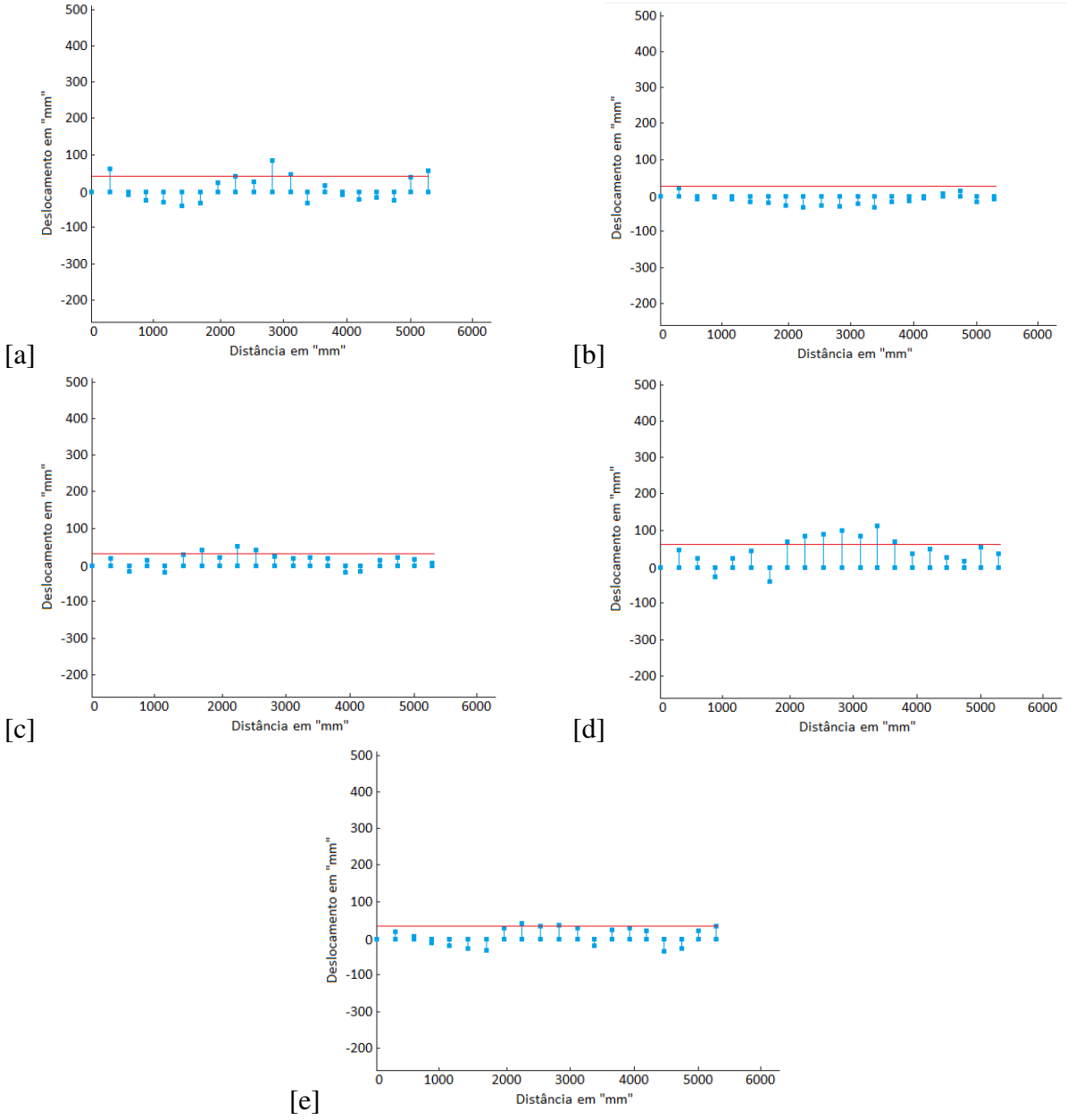
A fim de visualizar essas variações com maior eficácia os erros de desvio, em milímetros, entre rota ideal e rota traçada na realidade foram calculados e posteriormente alocados em gráficos que podem ser vistos nas Figuras8.21, 8.22, 8.23 e 8.24 respectivamente para as trajetórias em linha reta, circunferência, desvio com algorítmo BUG 2 em manobras de 90 graus e desvio em manobras de semi circunferências.

Figura 8.21 – Erro de trajetória executada em linha reta sobre ambiente real em comparação à rota pretendida idealmente, referente às superfícies de [a] asfalto, [b] concreto, [d] taco, [e] cerâmica molhada e [e] tapete.



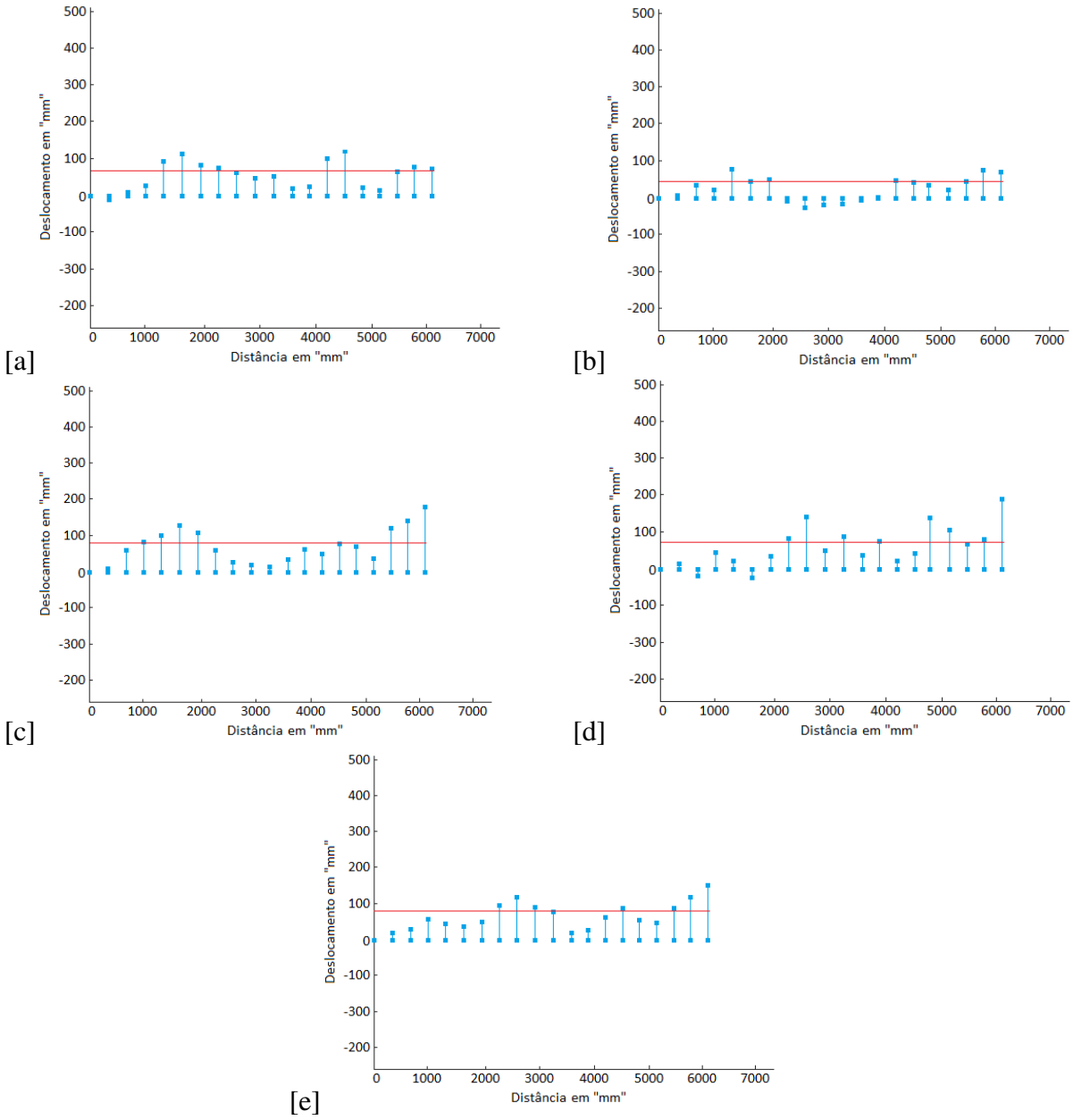
Fonte: Próprio Autor

Figura 8.22 – Erro de trajetória executada em circunferência sobre ambiente real em comparação à rota pretendida idealmente, referente às superfícies de [a] asfalto, [b] concreto, [d] taco, [e] cerâmica molhada e [e] tapete.



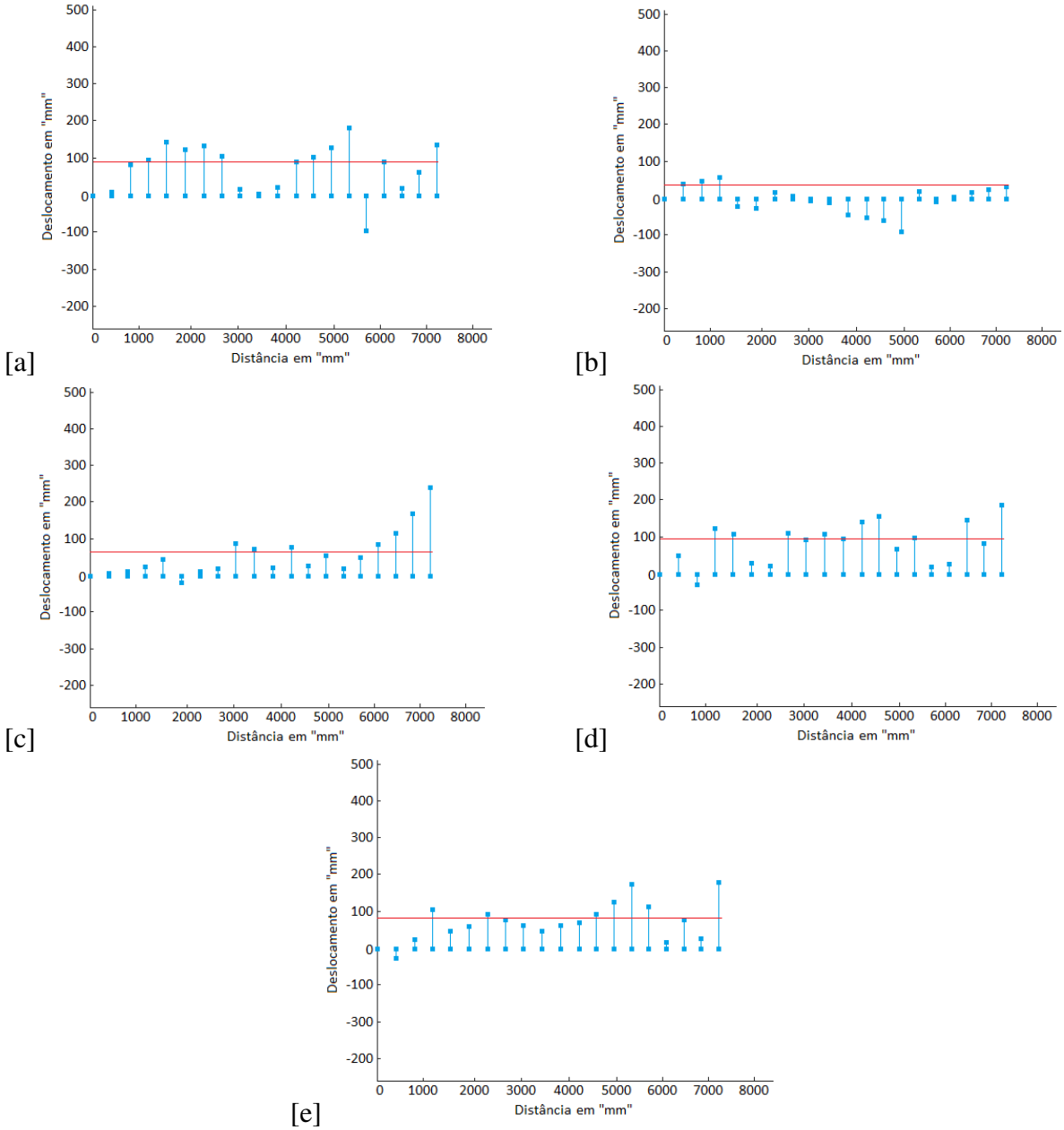
Fonte: Próprio Autor

Figura 8.23 – Erro de trajetória executada em desvio BUG 2 com manobras de 90 graus sobre ambiente real em comparação à rota pretendida idealmente, referente às superfícies de [a] asfalto, [b] concreto, [d] taco, [e] cerâmica molhada e [e] tapete.



Fonte: Próprio Autor

Figura 8.24 – Erro de trajetória executada em desvio BUG 2 com manobras se semi circunferência sobre ambiente real em comparação à rota pretendida idealmente, referente às superfícies de [a] asfalto, [b] concreto, [d] taco, [e] cerâmica molhada e [e] tapete.



Fonte: Próprio Autor

Os erros encontrados tanto na execução pura das trajetórias como no desvio podem ser devido a diversos fatores como irregularidades no terreno em que o módulo opera, bateria de alimentação do módulo com carga insuficiente para manter a potência necessária nos motores, ou discrepância entre atuação dos dois motores para uma mesma entrada aplicada.

A Tabela 8.1 evidencia os valores médios de erro para cada rota e piso de aplicação. As médias foram obtidas a partir dos valores absolutos de erro, sem levar em conta a direção em relação a rota original proposta.

É importante ressaltar que, se o módulo passar por um terreno irregular que provoque

a elavação de uma das rodas, deixando essa suspensa, o motor que gera velocidade a mesma continua atuando e, também o encoder. Assim, os gráficos podem fornecer valores que não correspondem idealmente ao movimento do robô na realidade, mas sim das velocidades de suas rodas de tração.

Tabela 8.1 – Valores médios de erro para cada rota e superfície de aplicação, onde I: Reta, II: Circunferência, III: Desvio com manobra de 90 graus e IV: Desvio com manobra em circunferência.

	ASFALTO	**CONCRETO**	**TACO**	**CERÂMICA**	**TAPETE**
I	74 mm	52 mm	66 mm	69 mm	55 mm
II	42 mm	27 mm	37 mm	64 mm	38 mm
III	67 mm	45 mm	84 mm	72 mm	83 mm
IV	90 mm	39 mm	70 mm	94 mm	82 mm

Fonte:Próprio Autor

Observa-se ainda um erro menor, em quase todas as rotas analisadas, para a superfície em concreto. Esse fato se deve ao fato de o atrito com o solo ser maior e o módulo derrapar com menos frequência enquanto executa seu percurso.

Nos dois percursos que tem circunferência em seu trajeto, o piso de cerâmica se mostra mais instável. Isso acontecece devido ao fato do mesmo estar molhado e o escorregamento, principalmente quando as velocidades das rodas aplicadas são distintas, ser frequente.

Comparando os dois algoritmos de desvio propostos, para o desvio exclusivamente em manobras de 90 graus, e uma rota em linha reta de aproximadamente 6 metros de comprimento com um obstáculo pontual em seu caminho, o módulo leva aproximadamente 20 segundos para traçar o trajeto proposto. Levando em conta o algoritmo com desvio em semi circunferência, esse tempo é reduzido para 18 segundos.

Para um trajeto pequeno, com apenas um desvio, o tempo de 2 s de diferença não é significativo, porém, ao ser aplicado em trajetos com mais obstáculos, o acúmulo de tempo ao final pode ser um fator importante na escolha do algoritmo a ser utilizado.

9 Conclusão

A pesquisa teve como objetivo principal a navegação autônoma por um protótipo robótico com restrições não-holonômicas em ambientes passíveis de obstáculos, desviando desses de maneira segura através da implementação de um algoritmo fundamental de desvio para a literatura, o BUG 2 (em duas formas de desvio distintas), tendo sua rota inicial predefinida só alterada em caso de obstáculos em seu caminho.

Tendo sido um protótipo robótico estabelecido, seu equacionamento matemático, dinâmico e cinemático, e a validação do sistema encontrado, a partir desse, desenvolvidos, duas trajetórias foram aplicadas ao módulo, tanto em ambiente virtual de simulação quanto em ambiente real de implementação.

As trajetórias de teste foram especificamente uma em linha reta e uma em circunferência completa e após a execução dessas, os algoritmos de desvio passaram a ser desenvolvidos, sendo que os mesmos foram simulados em ambiente virtual e, depois, foram implementados em ambientes reais.

Com o objetivo inicial de desenvolvimento de um módulo robótico e validação de algoritmo de desvio sobre o mesmo, o projeto se conclui tendo este sido alcançado.

Como se trata de um protótipo algumas limitações puderam ser observadas, como a capacidade limitada do robô em concluir rotas em sua totalidade ou os frequentes desvios sofridos ao longo de seu percurso, problemas que apresentam possíveis soluções sugeridas no capítulo anterior, principalmente pela aplicação de uma bússola eletrônica no sistema.

De maneira geral o projeto atendeu as especificações pretendidas, principalmente a autonomia robótica, porém pode ainda ser melhorado em diversos aspectos, o que fica como sugestão para um desenvolvimento futuro.

9.1 Sugestões para Trabalhos Futuros

Com os resultados de diferentes aplicações realizadas pelo módulo em diferentes situações, fica clara a necessidade de implementação de um controle mais preciso para o sistema, visto que o mesmo se deu em malha aberta até o momento.

Como o sistema se trata de um robô pequeno, com correntes aplicadas muito pequenas, o controle através de sua corrente se torna inviável devido aos custos e incertezas que isso acarretaria, já que amplificadores seriam necessários.

Um controle diferencial sobre as velocidades dos motores de tração também não seriam eficientes uma vez que se uma roda ficar sem contato com o solo devido a alguma situação imprevista, essa continuará se movendo da mesma forma que a roda que permanece em solo, acarretando um movimento de giro ao módulo real, mas uma interpretação de movimento em linha reta para o sistema.

Uma solução para os desvios encontrados sobre as rotas propostas e para a falha do sistema em chegar ao destino pretendido é proposta por uma aplicação de bússola sobre o módulo.

Uma bússola permitiria o controle de navegação do sistema real sobre o ambiente proposto e, através de análises em seus dados, correções seriam possíveis com maior facilidade.

Propõe-se aqui, também, a alteração das rodas do robô. As rodas aplicadas, embora eficientes, são muito lisas e finas, o que aumenta a derrapagem do robô em ambientes com pouca tração, como o caso da cerâmica molhada. Uma roda mais espessa e de material mais aderente diminuiria esse problema.

Uma forte sugestão ao avanço da pesquisa se dá para aplicação de diferentes obstáculos em sua rota de navegação para a correta analise do sistema em diversas situações, além de diferentes ambientes.

Referências

ABIMAQ. *A História das Máquinas – 70 Anos Abimaq.* 2006.

AMBROSE, T.; ASKEY, R. S., *An Experimental Investigation of Actuators for Space Robots*, IEEE International Conference on Robotics and Automation, p. 2625 – 2630, 1995.

ANJOS, H. R.; SANTOS, R. R. *Protocolo Serial Peripheral Interface - Descrição do processo e aplicações para Arduino.* Campo Grande, Brasil: Universidade Federal do Rio Grande do Sul, 2013.

ARDUINO. *Arduino Mega 2560*. 2017. Acessado em 15/07/2017. Disponível em: <http://www.arduino.cc>.

BECKER, A. J. et al. *Noções Básicas de Programação em MATLAB.* 2010.

BORGES, F. F. et al. *Coletor de Dados de Baixo Custo, baseado na Plataforna Arduino, para Aplicações em Pesquisas sobre Meio Ambiente.* 2014. Acessado em 04/12/2017. Disponível em: <www.meioambientepocos.com.br>.

CHILDRESS, D. H. *The fantastic inventions of Nikola Tesla.* Adventures Unlimited Press, 1993.

CHOSET, H. et al. *Principles of Robot Motion [S.l.]:* MIT Press, 2004.

DORF, R. C.; BISHOP, R. H. *Sistemas de controle moderno.* 8ª ed. Rio de Janeiro, RJ, Brasil: LTC Editora, 2001.

DUDEK, G.; JENKIN, M. *Computational Principles of Mobile Robotics.* 2000.

EFE,A. *Polícia divulga vídeo do acidente fatal com carro autônomo da Uber*. 2018. Acessado em 03/04/2018. Disponível em <https://g1.globo.com>.

FAVERO, E. M. de B. *Organização e arquitetura de computadores*. Universidade Tecnológica Federal do Paraná. 2011.

GABRIELE, N. *Insetto, uccello, libellula, robot Leonardo da Vinci(drone).* 2011. Acessado em 16/01/2015. Disponível em: <http://robotleonardodavinci.blogspot.com.br/>.

GROOVER, P. M. *Automação Industrial e Sistema de Manufatura*. 2011.
HAYES, A. et al. *Distributed Odor Source Localization*, IEEE Sensors Journal, Vol. 2, No. 3, Junho, 2002.

HWANG, S.; KINTIGH, B. P., *Implementation of An Intelligent Roving Robot Using Multiple Sensors*, Proceedings of the 1994 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFT 94), Las Vegas, NV, Outubro, 1994.

IG. *Nasa vai enviar outro robô a Marte em 2020*. 2012. Acessado em 16/01/2015. Disponível em: <http://ultimosegundo.ig.com.br/>.

INC, C. *M223X000X*. 2017. Acessado em 15/07/2017. Disponível em: <http://www.cui.com>.

KUHNE, F. *Controle Preditivo de Robôs Móveis Não Holonômicos.* 2005. Universidade Federal do Rio Grande do Norte. Dissertação de Mestrado.

KURPIEL, F. D. et al. *Robô OMNI2 - Sistema de Navegação do Robô Omnidirecional*. 2010. Universidade Tecnológica Federal do Paraná. Monografia.

LINK, S. *Módulo Ponte H Dupla – Manual Técnico.* 2013. Acessado em 15/07/2017. Disponível em: <http://www.seriallink.com.br>.

MAIA, D. V. de A. *Automação Industrial e Robótica*. Programa de Pós Graduação em Engenharia Elétrica, UFRN. Natal, RN, Brasil: [s.n.], 2003.

MELO, F. F. S. de. *ENCODER Instrumentação Eletrônica*. 2008.

MELO, L. F. de. *Proposta de Simulador Virtual para Sistema de Navegação de Robôs Móveis Utilizando conceitos de Prototipagem Rápida*. 2007. Tese apresentada ao programa de doutorado em Engenharia Mecânica da Universidade Estadual de Campinas.

OGATA, K. *Engenharia de Controle Moderno*. 2003.

PIRES, J. N. *Robótica: das máquinas gregas à moderna robótica industrial*. Jornal Público, 2002.

RENNA, R. B. D. et al. *Introdução ao kit de desenvolvimento Arduino.* 2013.

RIBEIRO, M. I. *Sensores em Robótica*. 2004. Pags. 228-229.

"ROBO". IN *Dicionário Priberam da Língua Portuguesa*. 2008–2013. Acessado em 23/12/2014. Disponível em: <http://www.priberam.pt/dlpo/robo>.

ROBOLIVRE. *Microcontroladores*. 2003. Acessado em 02/12/1017. Disponível em: <http://www.roboliv.re>.

RODRIGUES, E. J. *Sistema de Navegação Autônoma para Plataformas Robóticas Móveis Utilizando Técnicas de Controle Embarcado*. 2014. Tese apresentada ao programa de mestrado em Engenharia Elétrica da Universidade Estadual de Londrina.

RUSSEL, R. A., *Heat Trails as Short-Lived Navigational Markers for Mobile Robots*, Proceedings of the 1997 IEEE International Conference on Robotics and Automation, Albuquerque, New Mexico, Abril, 1997.

SA, W. I. *DT-3 Características e Especificações de Motores de Corrente Contínua e Conversores CA/CC*. Acessado em 25/09/2016. Disponível em: <http://www.weg.net>.

SANKARANARAYANAR A., VIDYASAGAR M., *Path planning for moving a point object amidst unknown obstacles in a plane: a new algorithm and a general theory for algorithm development.* Proceedings of the 29th IEEE Conference on Decision and Control, Honolulu, HI, USA, 1990

SANTINO, R. *Veículo da Tesla em modo autônomo bate e mata ocupante*. 2018. Acessado em 03/04/2018. Disponível em <https://olhardigital.com.br>.

SCANAVINI, L. C. *Implementação e Comparação de Algoritmos de Navegação com ROS e Simulação com VERP*. 2016. Monografia apresentada ao programa de graduação em Engenharia Elétrica da Universidade de São Paulo.

SECCHI, H. A. *Una Introducción a los Robots Móviles*. 2008. Textos e Material Didático.

SHOP, M. *Sensor de Distância Ultrassônico HC-SR04*. 2017. Acessado em 25/09/2017. Disponível em: <https://multilogica-shop.com>.

SICILIANO, B. et al. *Robotics: Modelling, Planning and Control*. Springuer, 2009.

SIEGWART, R.; NOURBAKHSH, I. R. *Introduction to Autonomous Mobile Robots*. A Bradford Book, The MIT Press, 2004.

SOUZA, J. A. M. F. de. *Robótica*. 2005. Textos e Material Didático. Acessado em 04/01/2015. Disponível em: <http://webx.ubi.pt/>.

UFRN. *Sistemas de controle moderno*. Universidade Federal do Rio Grande do Norte, 2003.

WAGNER, I. A. et al. *Distributed Covering By Ant-Robots Using Evaporating Traces*, IEEE Transactions on Robotics and Automation, Vol. 15, No. 5, Outubro, 1999.

WEI, D. C. M. *Método de Desvio de Obstáculos aplicado em Veículo Autônomo*. 2015. Tese apresentada ao programa de mestrado da Escola Politécnica da Universidade de São Paulo.

ZOHAIB, M. et al. *Control Strategies for Mobile Robot With Obstacle Avoidance.* COMSATS Institute of Information Technology 2014.

APÊNDICE A – Produção Científica

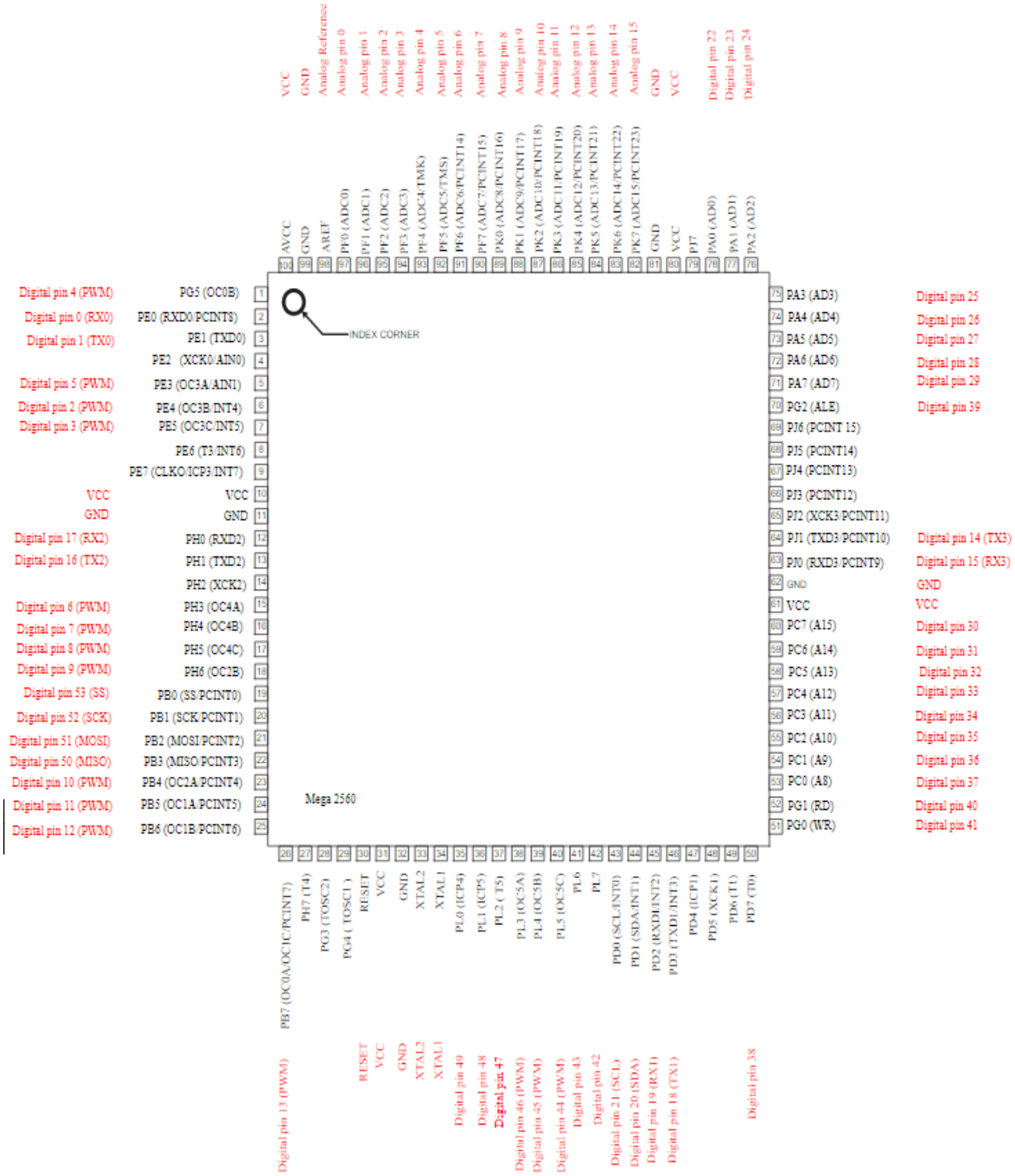
A.1 Artigo Publicado

ZAMAIA, J. F. P. ; MELO, L. F. . Nonholonomic Mobile Robot Prototype for Standalone Navigation with Embedded System. In: IEEE CHILECON2017, 2017, Pucón, Chile. Proceedings of IEEE CHILECON2017, 2017. v. 1. p. 67-73.

A.2 Artigo submetido à revista A1

ZAMAIA, J. F. P. ; MELO, L. F. . Multi-Trajectory testing and Navigation Optimizing for Non-holonomic Mobile Robot Prototype with embedded Control System. To: Jornal of Field Robotics, 2018, 12 pag.

ANEXO A – Diagrama de pinos do Arduino Mega 2560



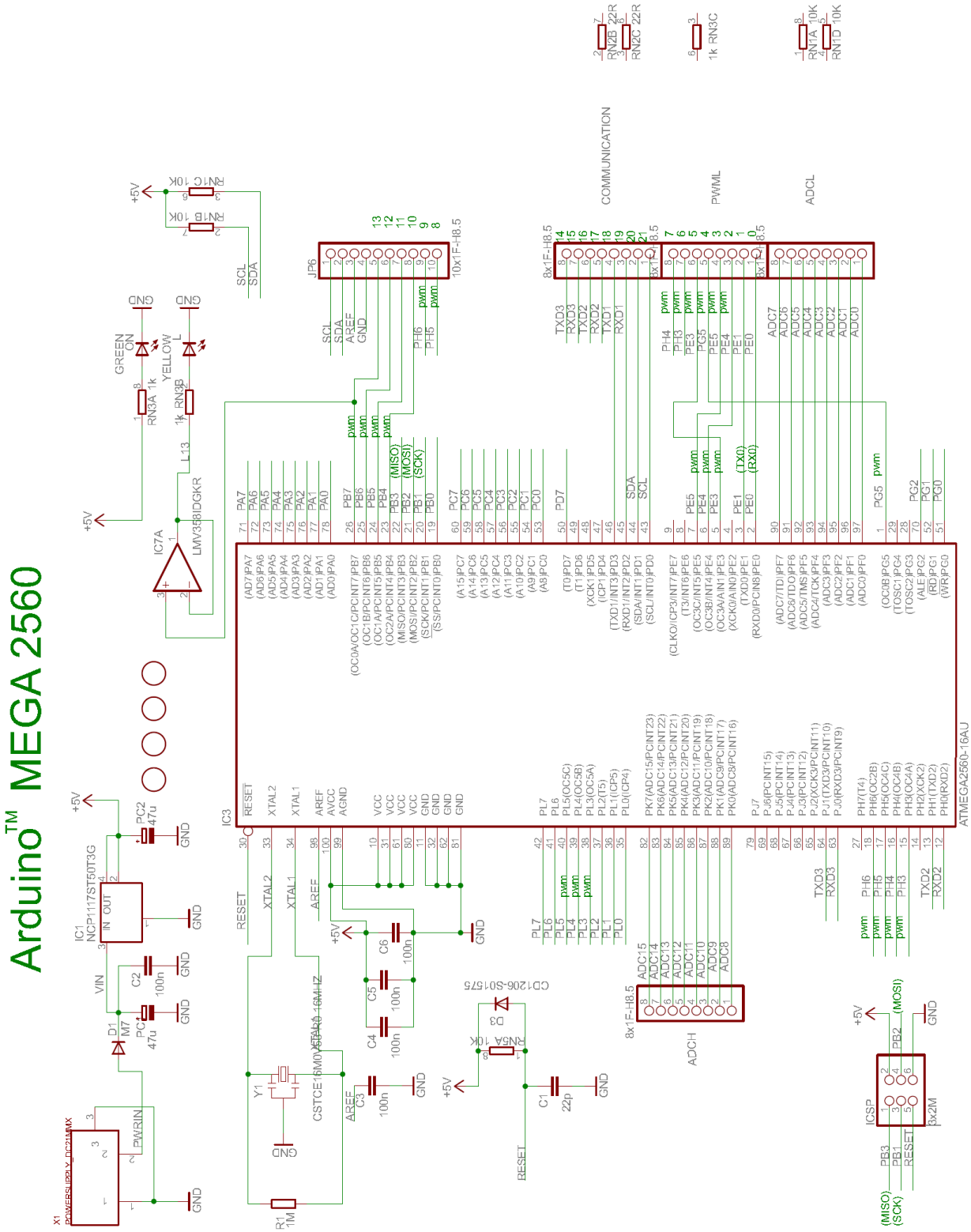
ANEXO B – Mapeamento de pinos do Arduino Mega 2560

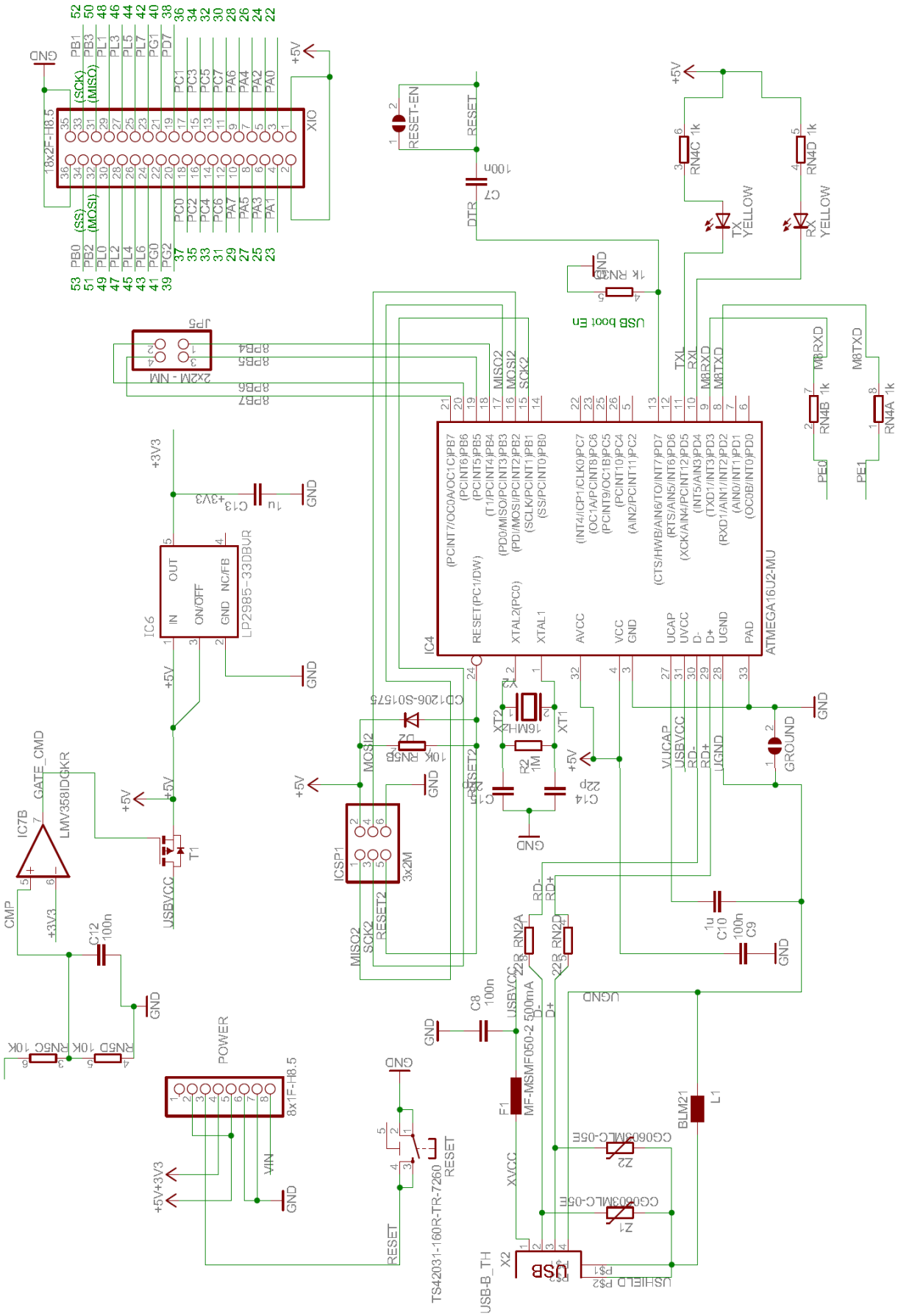
Pin Number	Pin Name	Mapped Pin Name
1	PG5 (OC0B)	Digital pin 4 (PWM)
2	PE0 (RXD0/PCINT8)	Digital pin 0 (RX0)
3	PE1 (TXD0)	Digital pin 1 (TX0)
4	PE2 (XCK0/AIN0)	
5	PE3 (OC3A/AIN1)	Digital pin 5 (PWM)
6	PE4 (OC3B/INT4)	Digital pin 2 (PWM)
7	PE5 (OC3C/INT5)	Digital pin 3 (PWM)
8	PE6 (T3/INT6)	
9	PE7 (CLKO/ICP3/INT7)	
10	VCC	VCC
11	GND	GND
12	PH0 (RXD2)	Digital pin 17 (RX2)
13	PH1 (TXD2)	Digital pin 16 (TX2)
14	PH2 (XCK2)	
15	PH3 (OC4A)	Digital pin 6 (PWM)
16	PH4 (OC4B)	Digital pin 7 (PWM)
17	PH5 (OC4C)	Digital pin 8 (PWM)
18	PH6 (OC2B)	Digital pin 9 (PWM)
19	PB0 (SS/PCINT0)	Digital pin 53 (SS)
20	PB1 (SCK/PCINT1)	Digital pin 52 (SCK)
21	PB2 (MOSI/PCINT2)	Digital pin 51 (MOSI)
22	PB3 (MISO/PCINT3)	Digital pin 50 (MISO)
23	PB4 (OC2A/PCINT4)	Digital pin 10 (PWM)
24	PB5 (OC1A/PCINT5)	Digital pin 11 (PWM)
25	PB6 (OC1B/PCINT6)	Digital pin 12 (PWM)
26	PB7 (OC0A/OC1C/PCINT7)	Digital pin 13 (PWM)
27	PH7 (T4)	
28	PG3 (TOSC2)	
29	PG4 (TOSC1)	
30	RESET	RESET
31	VCC	VCC
32	GND	GND
33	XTAL2	XTAL2
34	XTAL1	XTAL1
35	PL0 (ICP4)	Digital pin 49
36	PL1 (ICP5)	Digital pin 48
37	PL2 (T5)	Digital pin 47
38	PL3 (OC5A)	Digital pin 46 (PWM)

39	PL4 (OC5B)	Digital pin 45 (PWM)
40	PL5 (OC5C)	Digital pin 44 (PWM)
41	PL6	Digital pin 43
42	PL7	Digital pin 42
43	PD0 (SCL/INT0)	Digital pin 21 (SCL)
44	PD1 (SDA/INT1)	Digital pin 20 (SDA)
45	PD2 (RXDI/INT2)	Digital pin 19 (RX1)
46	PD3 (TXD1/INT3)	Digital pin 18 (TX1)
47	PD4 (ICP1)	
48	PD5 (XCK1)	
49	PD6 (T1)	
50	PD7 (T0)	Digital pin 38
51	PG0 (WR)	Digital pin 41
52	PG1 (RD)	Digital pin 40
53	PC0 (A8)	Digital pin 37
54	PC1 (A9)	Digital pin 36
55	PC2 (A10)	Digital pin 35
56	PC3 (A11)	Digital pin 34
57	PC4 (A12)	Digital pin 33
58	PC5 (A13)	Digital pin 32
59	PC6 (A14)	Digital pin 31
60	PC7 (A15)	Digital pin 30
61	VCC	VCC
62	GND	GND
63	PJ0 (RXD3/PCINT9)	Digital pin 15 (RX3)
64	PJ1 (TXD3/PCINT10)	Digital pin 14 (TX3)
65	PJ2 (XCK3/PCINT11)	
66	PJ3 (PCINT12)	
67	PJ4 (PCINT13)	
68	PJ5 (PCINT14)	
69	PJ6 (PCINT 15)	
70	PG2 (ALE)	Digital pin 39
71	PA7 (AD7)	Digital pin 29
72	PA6 (AD6)	Digital pin 28
73	PA5 (AD5)	Digital pin 27
74	PA4 (AD4)	Digital pin 26
75	PA3 (AD3)	Digital pin 25
76	PA2 (AD2)	Digital pin 24
77	PA1 (AD1)	Digital pin 23
78	PA0 (AD0)	Digital pin 22
79	PJ7	
80	VCC	VCC
81	GND	GND
82	PK7 (ADC15/PCINT23)	Analog pin 15
83	PK6 (ADC14/PCINT22)	Analog pin 14
84	PK5 (ADC13/PCINT21)	Analog pin 13
85	PK4 (ADC12/PCINT20)	Analog pin 12
86	PK3 (ADC11/PCINT19)	Analog pin 11

87	PK2 (ADC10/PCINT18)	Analog pin 10
88	PK1 (ADC9/PCINT17)	Analog pin 9
89	PK0 (ADC8/PCINT16)	Analog pin 8
90	PF7 (ADC7)	Analog pin 7
91	PF6 (ADC6)	Analog pin 6
92	PF5 (ADC5/TMS)	Analog pin 5
93	PF4 (ADC4/TMK)	Analog pin 4
94	PF3 (ADC3)	Analog pin 3
95	PF2 (ADC2)	Analog pin 2
96	PF1 (ADC1)	Analog pin 1
97	PF0 (ADC0)	Analog pin 0
98	AREF	Analog Reference
99	GND	GND
100	AVCC	VCC

ANEXO C – Esquemático da Placa Arduino MEGA 2560




ATMEGA16U2-MU
IC4
LP2985-33DBVR
IC6
LMV358IDGKR
IC7B
TS42031-160R-TR-7260
USB-B_TH
MF-MSMF050-2
ICSP1
ICSP2
POWER
XIO
RESET-EN
USB boot en
TX YELLOW
RX YELLOW
GROUND
USHIELD
M8RXD
M8TXD
TXL
RXL
MISO2
MOSI2
SCK2
RESET2
VUCAP
USBVCC
UGND
XVCC
+5V
+3V3
GND
VIN
RESET
DTR
CMP
GATE_CMD
USB

ANEXO D – Datasheet KIT AMT103

CUI INC

date 09/2009
page 1 of 5

SERIES: M223X000X **DESCRIPTION: M223 with AMT103 kit**

AMT SPECIFICATIONS

output phase difference	90° ±45°
frequency response	0 ~ 250 kHz
output current	0 ~ 5 mA max.
output waveform	square wave
output signals	A, B, Z phase
current consumption	6 mA typ., 10 mA max.
supply voltage	3.6 ~ 5.5 V dc
output resolution (ppr)[1]	48, 96, 100, 125, 192, 200, 250, 256, 384, 400, 500, 512, 800, 1000, 1024, 2048
time constant	0.2 ms (at 48, 96, 100, 125, 200, 250, 256, 512 ppr) 0.4 ms (at 192, 384, 400, 500, 800, 1000, 1024, 2048 ppr)
accuracy	±15 arcmin (at 192, 384, 400, 500, 800, 1000, 1024, 2048 ppr) ±30 arcmin (at 96, 200, 250, 512 ppr) ±60 arcmin (at 48, 100, 125, 256 ppr)
max. rotational speed	7500 rpm (at 384, 800, 1000, 2048 ppr) 15000 rpm (at 192, 400, 500, 1024 ppr) 30000 rpm (at 48, 96, 100, 125, 200, 250, 256, 512 ppr)
operating temp.	-40° ~ 100°C
mounting hole options	A) 2 each M1.6 holes on 16 mm (0.63") bolt circle B) 2 each #4 holes on 19.05 mm (0.75") bolt circle C) 2 each M1.6 or M2 holes on 20 mm (0.787") bolt circle D) 3 each M1.6 or M2 holes on 20.9 mm (0.823") bolt circle E) 3 each M1.6 or M2 holes on 22 mm (0.866") bolt circle F) 4 each M1.6 or M2 holes on 25.4 mm (1") bolt circle

note: 1. All resolutions stated are before quadrature decoding (example: 1000 ppr x 4 = 4000 counts)
2. Some stepper motors may leak a magnetic field causing the AMT index pulse to not function properly.

SERIES: M223X000X **DESCRIPTION: M223 with AMT103 kit**

M223X DC MOTOR SPECIFICATIONS

Micro-Drives

parameter	conditions/description	min	nom	max	units
gear reduction	M223X0002 M223X0003 M223X0004		none 1:84 1:231		
voltage			12		V dc
winding resistance		6.34	7.2	8.06	Ω
output power				4.65	W
efficiency				66	%
no load speed		7,400	8,400	9,400	rpm
no load current	shaft ø2 mm	0.03	0.06	0.09	A
stall torque			21.92		mNm
friction torque			0.79		mNm
speed constant			726		rpm/V
back-EMF constant			1.38		mV/rpm
torque constant			13.15		mNm/A
current constant			0.076		A/mNm
rotor inductance			3,400		µH
mechanical time constant			19		ms
rotor inertia			4.5		gcm^2
angular acceleration				49	10^3 rad/s^2
thermal resistance			17/22		k/W
thermal time constant			450		s
operating temp.		-10		50	°C
shaft bearing	sintered bronze sleeves				
shaft load	shaft ø2 mm: radial at 3,000 rpm (3 mm from bearing) axial at 3,000 rpm axial at standstill (shaft supported)			 2 1 294	 N N N
shaft play	radial ≤ axial ≤	 0.5		0.025 0.45	mm mm
direction of rotation	clockwise, viewed from the front face				
speed up to				10,000	rpm
torque up to				5,307	mNm
current up to	thermal limits			0.502	A

ANEXO E – Datasheet do ENCODER AMT103

CUI INC®

date 07/18/2014
page 1 of 8

SERIES: AMT10 | **DESCRIPTION:** MODULAR INCREMENTAL ENCODER

FEATURES

- patented capacitive ASIC technology
- low power consumption
- CMOS outputs
- 16 DIP switch selectable resolutions
- index pulse
- modular package design
- straight (radial) and right-angle (axial) versions
- 9 mounting hole options for radial version
- 8 mounting hole options for axial version
- -40~100°C operating temperature

RoHS

ELECTRICAL

parameter	conditions/description	min	typ	max	units
power supply	VDD	3.6	5	5.5	V
current consumption	with unloaded output		6		mA
output high level		VDD-0.8			V
output low level				0.4	V
output current	CMOS sink/source per channel			2	mA
rise/fall time			30		ns

INCREMENTAL CHARACTERISTICS

parameter	conditions/description	min	typ	max	units
channels	quadrature A, B, and X index				
waveform	CMOS voltage square wave				
phase difference	A leads B for CCW rotation (viewed from front)		90		degrees
quadrature resolutions[1]	48, 96, 100, 125, 192, 200, 250, 256, 385, 400, 500, 512, 800, 1000, 1024, 2048				PPR
index[2]	one pulse per 360 degree rotation				
accuracy			0.25		degrees
quadrature duty cycle (at each resolution)	256, 512, 1024, 2048 48, 96, 100, 125, 192, 200, 250, 384, 400, 500 800, 1000	49 47 43	50 50 50	51 53 56	% % %

Notes: 1. Resolution selected via adjustable DIP switch
2. Some stepper motors may leak a magnetic field causing the AMT index pulse to not function properly (non-magnetic version available with 8 pulses per revolution).

MECHANICAL

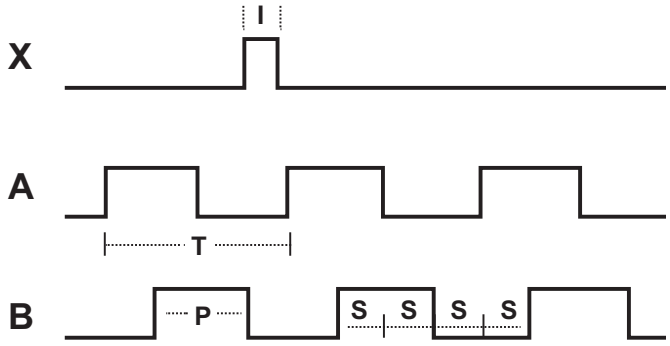
parameter	conditions/description	min	typ	max	units
motor shaft length		9			mm
weight	AMT102 AMT103		20.5 14.0		g g
axial play				±0.3	mm
rotational speed (at each resolution)	192, 384, 400, 500, 800, 1000, 1024, 2048			7500	RPM
	48, 96, 100, 125, 200, 250, 256, 512			15000	RPM

ENVIRONMENTAL

parameter	conditions/description	min	typ	max	units
operating temperature		-40		100	°C
humidity	non-condensing			95	%
vibration	20~500 Hz, 1 hour on each XYZ			10	G
shock	11 ms, ±XYZ direction			50	G
RoHS	2011/65/EU				

WAVEFORMS

Figure 1
Quadrature signals with index showing counter-clockwise rotation



The following parameters are defined by the resolution selected for each encoder, where R = resolution.

Parameter	Description	Expression	Units
T	period	360/R	mechanical degrees
P	pulse width	T/2	mechanical degrees
I	index width	P/2	mechanical degrees
S	A/B state width	P/2	mechanical degrees

PART NUMBER KEY

For customers that prefer a specific AMT10 configuration, please reference the custom configuration key below.

AMT10X - XXXXX - X XXXX - X

Base Number

Orientation:
2 = Radial
3 = Axial
*See Mechanical Drawings

Resolution Option:
"blank" = fixed resolution
D = adjustable resolution

Resolution[1] (ppr):

0048	0384
0096	0400
0100	0500
0125	0512
0192	0800
0200	1000
0250	1024
0256	2048

Index Pulse:
I = single index pulse per revolution
N = 8 index pulses per revolution

Sleeve Bore Diameter:
2000 = 2 mm
3000 = 3 mm
3175 = 3.175 mm (1/8")
4000 = 4 mm
4760 = 4.76 mm (3/16")
5000 = 5 mm
6000 = 6 mm
6350 = 6.35 mm (1/4")
8000 = 8 mm

Mounting Base:
S = Standard
W = Wide (AMT102 only)

Note: 1. Fixed resolutions are permanently set at this value; adjustable resolutions are preset via DIP switch to this value upon shipment.

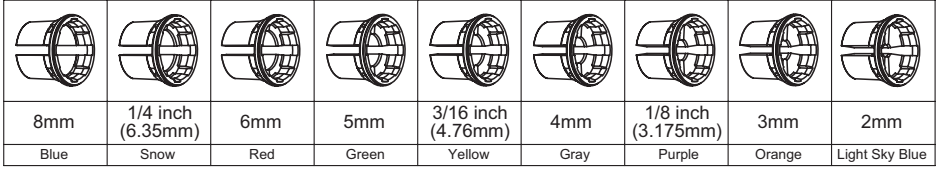
AMT10-V KITS

In order to provide maximum flexibility for our customers, the AMT10 series is provided in kit form standard. This allows the user to implement the encoder into a range of applications using one sku#, reducing engineering and inventory costs.

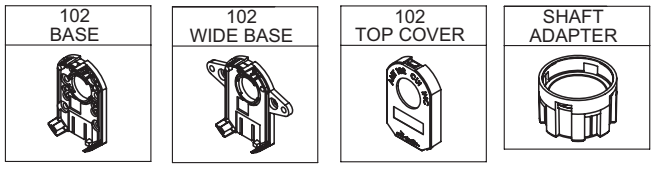
ORDERING GUIDE
AMT10X-V

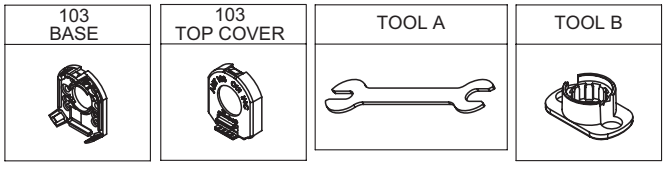
Orientation:
2 = Radial
3 = Axial
*See Mechanical Drawings

SLEEVES



8mm	1/4 inch (6.35mm)	6mm	5mm	3/16 inch (4.76mm)	4mm	1/8 inch (3.175mm)	3mm	2mm
Blue	Snow	Red	Green	Yellow	Gray	Purple	Orange	Light Sky Blue



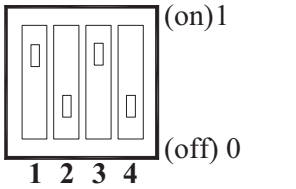


RESOLUTION SETTINGS

1 = On, 0 = Off

Resolution (PPR)	Maximum RPM	1	2	3	4
2048	7500	0	0	0	0
1024	7500	0	0	1	0
1000	7500	1	0	0	0
800	7500	0	1	0	0
512	15000	0	0	0	1
500	7500	1	0	1	0
400	7500	0	1	1	0
384	7500	1	1	0	0
256	15000	0	0	1	1
250	15000	1	0	0	1
200	15000	0	1	0	1
192	7500	1	1	1	0
125	15000	1	0	1	1
100	15000	0	1	1	1
96	15000	1	1	0	1
48	15000	1	1	1	1

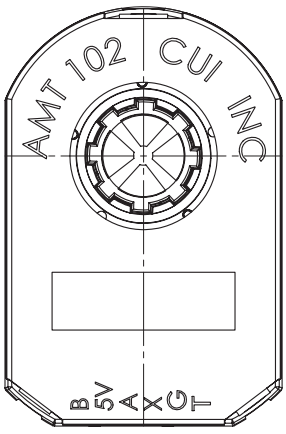
DIP switch:
Example setting: 500 PPR



ENCODER INTERFACE

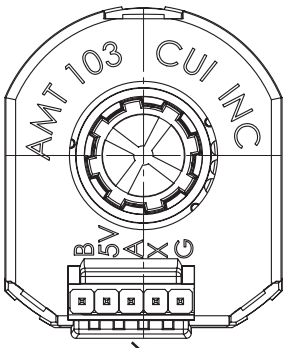
PINOUT CONNECTOR		
Function		
#	AMT102	AMT103
B	B CHANNEL	B CHANNEL
5V	+5 V	+5 V
A	A CHANNEL	A CHANNEL
X	INDEX CHANNEL	INDEX CHANNEL
G	GND	GND
T	UNUSED	N/A

AMT102



Mating Connector:
Molex 50-57-9405 Housing
Molex 16-02-0086 Terminals

AMT103



Mating Connector:
AMP 3-640440-5 (tin)
AMP 3-641237-5 (gold)