UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

MARIANA GONÇALVES RODRIGUES

DESENVOLVIMENTO DE UM MODELO CINEMÁTICO PARA UM VEÍCULO TERRESTRE DE BAIXO CUSTO QUE UTILIZA DE SISTEMAS INTELIGENTES PARA NAVEGAÇÃO AUTÔNOMA

APUCARANA

2023

MARIANA GONÇALVES RODRIGUES

DESENVOLVIMENTO DE UM MODELO CINEMÁTICO PARA UM VEÍCULO TERRESTRE DE BAIXO CUSTO QUE UTILIZA DE SISTEMAS INTELIGENTES PARA NAVEGAÇÃO AUTÔNOMA

Development Of A Kinematic Model For a Low-cost Land Vehicle That Uses Intelligent Systems For Autonomous Navigation

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia da Computação do Curso Engenharia da Computação da Universidade Tecnológica Federal do Paraná.

Orientador: Dr. Maurício Eiji Nakai

Coorientador: Dr. Fábio Irigon Pereira

APUCARANA

2023

MARIANA GONÇALVES RODRIGUES

DESENVOLVIMENTO DE UM MODELO CINEMÁTICO PARA UM VEÍCULO TERRESTRE DE BAIXO CUSTO QUE UTILIZA DE SISTEMAS INTELIGENTES PARA NAVEGAÇÃO AUTÔNOMA

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia da Computação do Curso Engenharia da Computação da Universidade Tecnológica Federal do Paraná.

Data de aprovação: 13/novembro/2023

Maurício Eiji Nakai
Doutor
Universidade Tecnológica Federal do Paraná

Fábio Irigon Pereira
Doutor
Universidade Tecnológica Federal do Paraná

Amanda Sant'Anna Montecin Sciamarella
Especialização
Colégio Embraer Juarez Wanderley

APUCARANA

2023

Dedico este trabalho aos meus pais, Andrea e Marcelo, por me apoiarem, as minhas irmãs Ana e Mabila, aos amigos, e ao meu amor Anderson, por todo o carinho.

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus por me permitir chegar tão longe.
Agradeço em especial a minha família que me apoiou neste sonho,
Ao meu amor que esteve comigo em todos os momentos dessa jornada,
Aos meus colegas que se fizeram família e me ajudaram a passar pela graduação,
Aos meus orientadores, Maurício e Fábio, pelo suporte e ensinamentos que me fazem ser uma engenheira melhor.
Obrigada!

"Eu gosto de aprender. Isso é uma arte e uma ciência. - Katherine Johnson.

RESUMO

Veículos autônomos são robôs que possuem a capacidade de percorrerem um trajeto de forma a identificar obstáculos e não colidirem com estes, ou seja, não possuem interferências humanas para cumprir tarefas. Dessa forma, visto o interesse das montadoras automotivas em veículos autônomos desde a década de 1960, e o avanço nos estudos e desenvolvimento desses projetos, a seguinte proposta possuí o objetivo de entender com maior profundidade desde a montagem de um veículo autônomo até a sua modelagem dinâmica. Dessa forma, ao explorar essa área de conhecimento, faz-se necessário que hajam etapas no projeto como: estudar equipamentos de hardware a serem usados e como conectá-los entre si, modelar em 3D um chassi para o robô de forma a acomodar todos materiais, modelar cinematicamente o carrinho em plataformas de software e repassar as equações para o robô. Diante disso, espera-se que a modelagem do veículo acompanhe de forma fiel a simulação realizada.

Palavras-chave: modelagem 3d; robôs móveis ; robôs - sistemas de controle; veículos autônomos .

ABSTRACT

Autonomous vehicles are robots that have the ability to follow a path in order to identify obstacles and not collide with them, that is, they do not have human interference to fulfill tasks. In this way, given the interest of automakers in autonomous vehicles since the 1960s, and the progress in the studies and development of these projects, the following proposal had the objective of understanding in greater depth from the assembly of an autonomous vehicle to its . In this way, when exploring this area of knowledge, it is necessary to have steps in the project such as: studying hardware equipment to be used and how to connect them together, 3D modeling a chassis for the robot in order to accommodate all materials, cinematically model the cart on software platforms and pass wounds to the robot. In view of this, it is expected that the modeling of the vehicle faithfully follows the simulation performed.

Keywords: 3d modeling; autonomous vehicles; mobile robots; robots - control systems.

LISTA DE FIGURAS

Figura 1 – Fluxograma do projeto **11**

Figura 2 – Página da GTP automation **13**

Figura 3 – Gráfico de mortes por ano envolvendo acidentes com o carro da marca Tesla **14**

Figura 4 – Arquitetura de Von Neuman **17**

Figura 5 – Arquitetura de Harvard **17**

Figura 6 – Modelo de bicicleta **19**

Figura 7 – Robô móvel com rodas **22**

Figura 8 – Arduino Uno **24**

Figura 9 – Motor DC **25**

Figura 10 – Ponte H **25**

Figura 11 – xe, ye e alphae **26**

Figura 12 – Fluxograma - Modelagem Cinemática **35**

Figura 13 – Robô projetado **36**

Figura 14 – Robô projetado com câmera **37**

Figura 15 – Esquemático **38**

Figura 16 – Aplicativo **39**

Figura 17 – Rota projetada pelo algoritmo A* **39**

Figura 18 – Ponto (-0.5,-0.3) **40**

Figura 19 – Ponto (0.5,0.3) **41**

Figura 20 – Ponto (-0.9,0.6) **41**

Figura 21 – Circuito Planejado **42**

Figura 22 – Primeira visão **43**

Figura 23 – Segunda visão **43**

Figura 24 – Terceira visão **44**

Figura 25 – Quarta visão **44**

Figura 26 – Andar um do robô **63**

Figura 27 – Andar dois do robô **64**

SUMÁRIO

1 INTRODUÇÃO 10

1.1 Descrição do projeto 10

1.2 Motivação 11

1.3 Objetivos 13

1.3.1 Objetivo geral 13

1.3.2 Objetivos específicos 14

1.4 Estrutura do trabalho 14

2 REFERENCIAL TEÓRICO 16

2.1 Microcontroladores 16

2.2 Robôs móveis com rodas 18

2.2.1 Mobilidade 18

2.2.2 Robôs móveis semelhantes a carros 19

2.3 Modelo cinemático do robô 20

3 METODOLOGIA 24

3.1 Materiais 24

3.2 Métodos 25

3.2.1 Modelagem Cinemática no MATLAB 25

3.2.2 Modelagem Cinemática no Arduino IDE 28

3.2.3 Fluxograma de funcionamento do projeto 34

3.2.4 Projeto do Robô 35

4 RESULTADOS 36

4.1 Escopo do projeto 36

4.2 Apresentação dos resultados 40

5 CONCLUSÃO 46

REFERÊNCIAS 47

APÊNDICE A MODELAGEM CINEMÁTICA DESENVOLVIDA NO AMBIENTE DE SOFTWARE MATLAB VERSÃO 2023A 50

APÊNDICE B CÓDIGO DA MODELAGEM CINEMÁTICA DESENVOLVIDA PARA O HARDWARE DE ARDUINO NA PLATAFORMA ARDUINO IDE 54

APÊNDICE C CÓDIGO DA MODELAGEM CINEMÁTICA DESENVOLVIDA PARA O HARDWARE DE ARDUINO NA PLATAFORMA ARDUINO IDE RECEBENDO COMO ENTRADA ÂNGULO E DISTÂNCIA 60

ANEXO A MODELAGEM 3D 63

1 INTRODUÇÃO

Segundo (CORREA, 2013) um dos principais objetivos da área da robótica reside em aprimorar a qualidade de vida humana, promovendo o conforto e apresentando soluções para uma variedade de tarefas cotidianas. Essas tarefas englobam uma ampla gama de setores, abrangendo desde o âmbito doméstico, educacional e empresarial até o industrial e militar. Inicialmente, os robôs eram estáticos, consistindo principalmente em braços manipuladores, o que caracteriza a vertente da robótica industrial. Em uma segunda fase de desenvolvimento, surgiram os robôs móveis, que incorporam mecanismos que lhes permitem navegar e se deslocar no ambiente, seja em terra, ar ou água.

Ademais, os robôs de campo necessitam de estratégias de controle, pois não operam em condições ideais, como em locais insalubres ou inacessíveis, temos os exemplos de desastres naturais ou acidentes de usina nuclear. Além disso, algoritmos de navegação autônoma precisam de um robô modelado cinematicamente para diversos testes.

Dessa forma, o presente trabalho tem como proposta projetar um robô, desenvolvendo-o em uma versão de baixo custo para navegação autônoma de veículos terrestres, com materiais mais acessíveis. Além disso, será modelado cinematicamente para controlar o veículo em diferentes condições de ambiente e assim ajustar seu comportamento para maximizar a segurança e a eficiência.

1.1 Descrição do projeto

O projeto será composto de um robô feito de materiais de baixo custo, como motor DC, Arduino, ponte h, câmera, módulo bluetooth HC-05 e componentes eletrônicos simples como resistores e alguns jumpers. Além disso, o Arduino controlará as rodas juntamente com a ponte H, responsável pelo controle dos motores DC.

Ademais, vale ressaltar que a câmera irá tirar a foto e enviar para um computador que terá o algoritmo para processar imagens e traçar caminhos de forma autônoma, entretanto os algoritmos de processamento de imagens e predição de rotas não estão no escopo deste trabalho. Diante disso, o Arduino receberá um ângulo e uma distância do algoritmo de forma a proceder corretamente para suas funcionalidades autônomas.

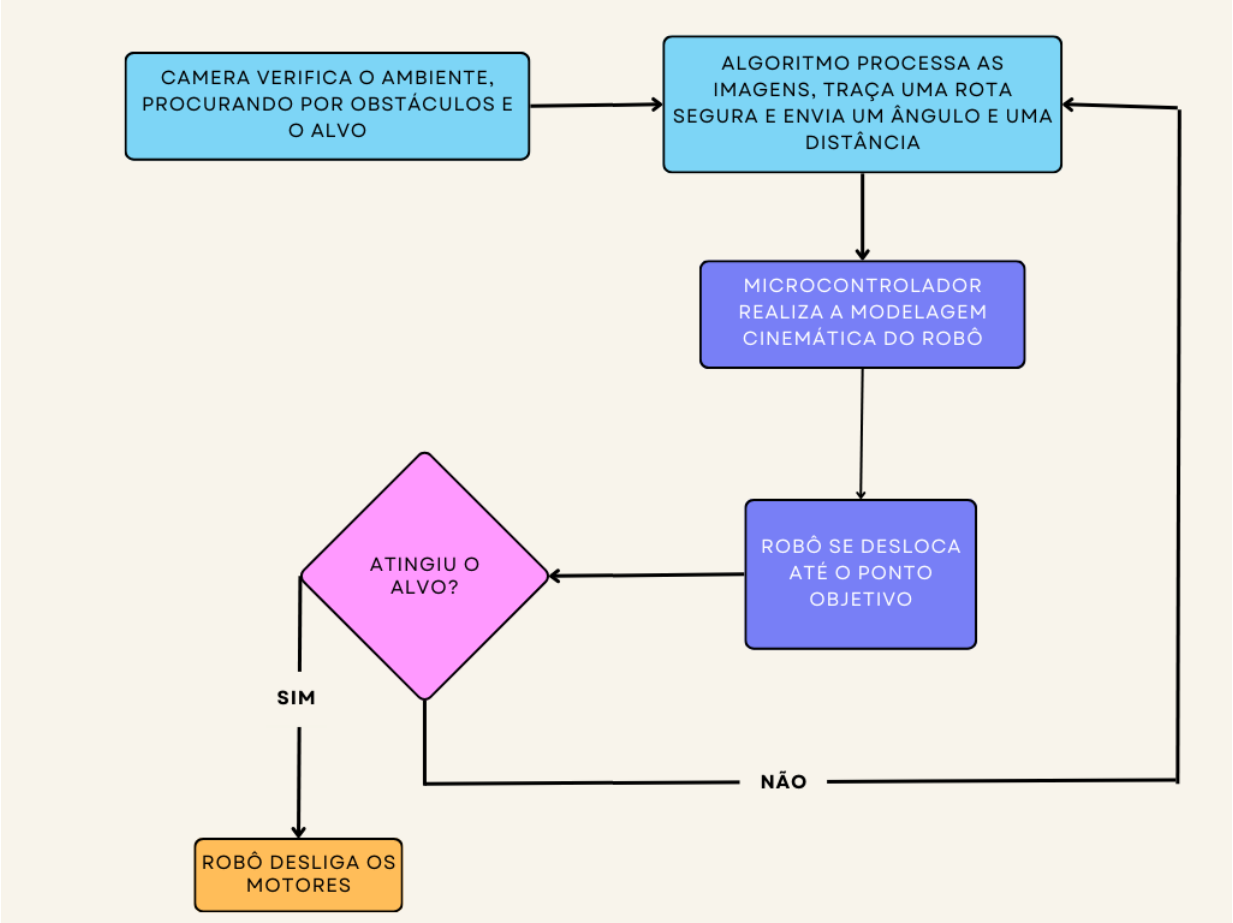
Dessa forma, o robô receberá um ângulo e uma distância do algoritmo, que por sua vez processa as imagens, detecta os obstáculos e traça a melhor rota para alcançar o objetivo, que é um objeto da cor vermelha. Ao receber o ângulo e distância, o Arduino calcula a velocidade linear, angular e as velocidades angulares de cada roda com o intuito de se aproximar o melhor possível do primeiro ponto objetivo. Deste modo, a câmera tira outra foto e o algoritmo processa novas entrada para próximo ponto objetivo da rota, até alcançar o alvo.

Vale ressaltar que, o algoritmo de processamento de imagens e predição de rotas que é utilizado como complemento neste projeto, considerou o trabalho de conclusão de curso de um

outro membro do curso de engenharia de computação da UTFPR Apucarana, Anderson Soares da Silva (SILVA, 2023).

O fluxograma de funcionamento do projeto pode ser melhor visualizado na figuras 1 a seguir.

Figura 1 – Fluxograma do projeto



Fonte: Autoria própria, 2023 .

1.2 Motivação

Este projeto visa a exploração da funcionalidade dos veículos autônomos, que são aqueles que possuem a capacidade de se locomoverem de uma origem a um destino de forma a perceberem obstáculos à sua volta, sem que haja colisão nestes, e como aplicá-los com baixo custo em robôs de campo . Além disso, segundo (PISSARDINI DANIEL CHIN MIN WEI, 2013) "a definição de carro robótico está associada ao processo de autonomia de controle da condução, independentemente do tipo de veículo e da tecnologia empregada neste processo".

Em 1964, a General Motors patrocinou uma feira mundial em Nova Iorque, na qual apresentou uma proposta, em que uma torre de controle operaria a direção, freios e velocidade de

cada veículo em uma pista automática. Entretanto, entre 1966 e 1972 foi desenvolvido o robô Shakey pelo Centro de Inteligência Artificial do Instituto de Pesquisa de Stanford (SRI), o qual recebia instruções necessárias para navegação, enviadas de um computador que recebia os dados captados pelos sensores e realizava o planejamento de rota.

Dessa forma, a ideia de um veículo autônomo foi desenvolvida por outras empresas e instituições, como em 1977 pelo Laboratório de Engenharia Mecânica da Universidade de Tsukuba, no Japão, construiu um carro com um sistema de visão computacional baseada em câmeras de televisão e uma unidade de processamento que permitia detecção de obstáculos e seguimento de linhas brancas no solo.

Além disso, em 1980 o pioneiro do carro autônomo, o engenheiro aeroespacial alemão Ernst Dickmanns e sua equipe da Universität der Bundeswehr München, na Alemanha, projetaram o veículo VaMoRs de 1985, uma van Mercedes-Benz, equipada com câmeras e outros sensores, onde a direção e outros componentes eram controlados por comandos computacionais. O veículo podia, de forma autônoma, atingir até 100 km/h em vias sem tráfego (PISSARDINI DANIEL CHIN MIN WEI, 2013) . Também na década de 1980 as empresas japonesas Fujitsu e Nissan projetaram um sistema baseado em visão computacional e aplicada em um micro-ônibus, que utilizava de um conjunto de câmeras estéreos, processador de imagens e computadores de controle que permitiam detectar marcações na pista e detecção de obstáculos (PISSARDINI DANIEL CHIN MIN WEI, 2013).

Contudo, na contemporaneidade há robôs autônomos para outros ambientes, como campo, industriais e de serviço. Para os industriais essa tecnologia é utilizada para tarefas pesadas de modo que transporta materiais e produtos finais em sistemas de manufatura, como a empilhadeira autônoma da empresa GTP automation que é possível observar na Figura 2 .

Outrossim, os robôs de serviço podem se estender de aspirador autônomo até os que transportam exames laboratoriais, aqueles que podem melhorar o tempo das pessoas que buscam otimizar suas rotinas. Já estes auxiliam na diminuição de contágio dos profissionais que trabalham em laboratório.

Outro fator que motiva o estudo da navegação autônoma é a quantidade de acidentes envolvendo carros autônomos, na Figura 3 é possível observar que o número de mortes envolvendo acidentes com os carros da marca Tesla vem aumentando ao longo dos anos. Vários desses acidentes evidenciados no gráfico são relacionados a aceleração repentina entre outras funcionalidades autônomas.

Diante desses cenários, a modelagem cinemática dos veículos autônomos torna-se intrinsecamente importante, pois diferentemente das condições ideais de locomoção, que nem todos os veículos acompanham, os robôs de campo necessitam de estratégias de controle.

Dessa forma, faz-se fundamental realizar a modelagem cinemática e incorporá-los no microcontrolador. Além disso, a modelagem cinemática é essencial para o desenvolvimento e teste de algoritmos de direção autônoma.

Figura 2 – Página da GTP automation

Fonte: (GTP, 2020) .

Portanto, esse projeto motiva-se nos robôs autônomos sobre condições genéricas, seja industriais, de serviço ou de campo, nas quais os efeitos da cinemática trazem interessantes desafios e perspectivas de pesquisa.

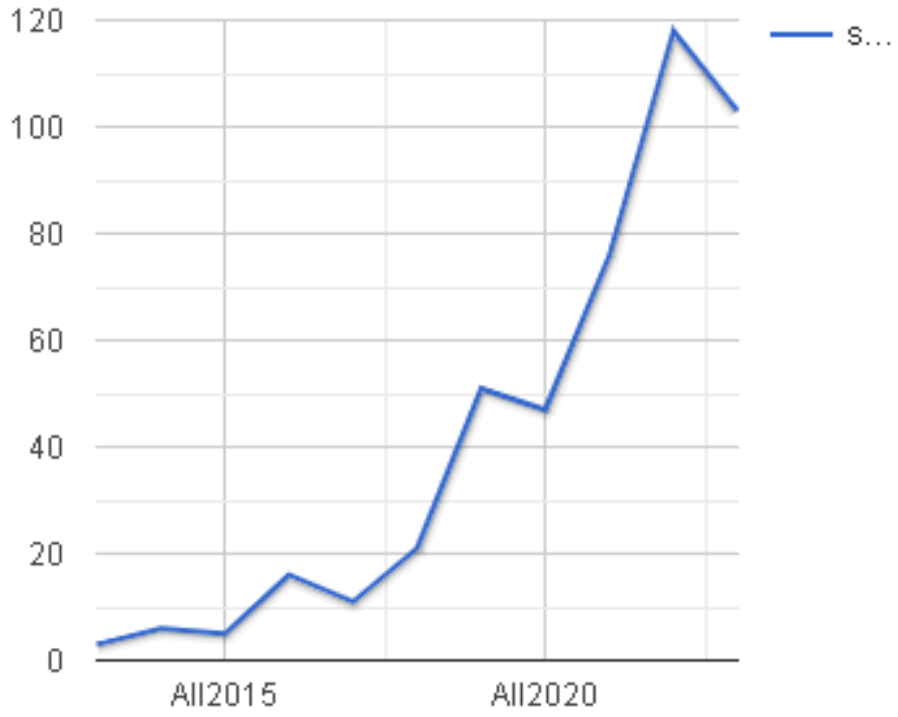
1.3 Objetivos

O objetivo deste trabalho é um robô que tenha navegação autônoma, e para esta meta dispões que a modelagem cinemática faça diferença no comportamento do projeto, de forma que se otimize os processos de navegação.

1.3.1 Objetivo geral

Desenvolver um robô de baixo custo modelado cinematicamente para obter maior eficiência que utiliza de processamento de imagens e predição de rotas para navegação autônoma com o objetivo de ser mais acessível a instituições científicas e a empresas que se proponham a melhor esta tecnologia.

Figura 3 – Gráfico de mortes por ano envolvendo acidentes com o carro da marca Tesla



Fonte: (TESLA, 2023) .

1.3.2 Objetivos específicos

Com o presente trabalho, objetiva-se obter um veículo terrestre capaz de navegar de forma autônoma e segura. Dessa forma, a modelagem cinemática proporcionará uma rota eficaz para o robô e um controle de velocidade a medida que se aproxima do objetivo.

Diante disso, espera-se que o projeto se adapte a diferentes ambientes em que estiver presente, através da modelagem cinemática.

1.4 Estrutura do trabalho

A estrutura deste trabalho é composta de Introdução, Referencial Teórico, Metodologia, Resultados e conclusão. Dessa forma, a introdução apresenta a ideia principal e sucinta do projeto , bem como a descrição como será o escopo deste, a motivação descreve fatos históricos importantes sobre a navegação autônoma, e o referencial teórico apresenta resumidamente as ideias de metodologias a serem usadas para a modelagem dinâmica.

A metodologia desenvolve de forma teórica e detalhada os passos que serão seguidos para a construção do trabalho, como a modelagem cinemática, a montagem do robô, funções

importantes dos códigos e materiais utilizados. Dessa forma, a seção possui o objetivo de auxiliar aqueles que tenham o interesse de replicar o projeto.

Os resultados é uma seção que expõe de forma concreta os resultados alcançados dos objetivos específicos, como o carrinho se comportou com a modelagem cinemática, e como esta funcionou bem para a navegação autônoma do robô.

A conclusão apontará uma discussão sobre os resultados, e se o trabalho alcançou os objetivos esperados.

2 REFERENCIAL TEÓRICO

O referencial teórico da presente pesquisa foi estruturado em três tópicos, a saber: Microcontroladores, Robôs móveis com rodas, Modelo cinemático do robô.

2.1 Microcontroladores

Um microcontrolador é como ter um computador inteiro compactado em um único chip. Esse chip contém uma unidade de processamento (ULA - Unidade Lógica e Aritmética), memória, dispositivos de entrada e saída, temporizadores, e outros periféricos para comunicação. Os microcontroladores surgiram como uma evolução natural dos circuitos digitais, acompanhando o aumento de complexidade desses sistemas. Chega um ponto em que é mais prático, econômico e eficiente substituir a lógica das portas digitais por um conjunto integrado de processador e software (PENIDO, 2013).

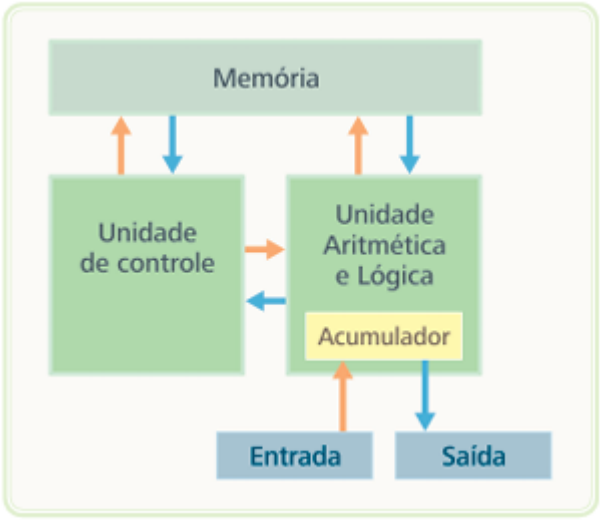
A empresa Intel lançou o primeiro microcontrolador, chamado "8048", em 1977. Esse lançamento foi seguido pela evolução para a família "8051". Esse chip é programado em linguagem Assembly e oferece um conjunto robusto de instruções. Por ser um dos pioneiros, é amplamente utilizado em uma variedade de aplicações de automação em diferentes partes do mundo. O microcontrolador contém internamente: a) Uma unidade central de processamento (CPU) responsável por interpretar as instruções do programa. b) Uma memória PROM (Programmable Read Only Memory ou Memória Programável Somente de Leitura) onde as instruções do programa são gravadas. c) Uma memória RAM (Random Access Memory ou Memória de Acesso Aleatório) usada para armazenar as variáveis que o programa utiliza. d) Um conjunto de portas de entrada/saída para controlar dispositivos externos ou receber sinais de sensores, interruptores, etc. e) Vários dispositivos auxiliares ao funcionamento, como um gerador de clock, contadores, USART para comunicação, entre outros (PENIDO, 2013).

Quando um sistema de processamento de dados (incluindo processadores e microcontroladores) tem uma única área de memória para armazenar dados (variáveis) e o programa a ser executado (software), isso é conhecido como arquitetura de Von Neuman como exposto na Figura 4. Por outro lado, se os dados (variáveis) são armazenados em uma área de memória separada do programa a ser executado (software), temos a arquitetura Harvard como exibido na Figura 5 (PENIDO, 2013).

É importante ressaltar que, o surgimento da indústria, levou a uma demanda crescente por processos de produção mais estáveis. Para atender a essa necessidade, surgiu o controle automático de processos, que ajuda a indústria a atender a requisitos de qualidade cada vez mais restritos e, cumprir normas de segurança e ambientais, culminando em uma redução dos seus custos operacionais (SEBORG D.E.; MELLICHAMP, 2004).

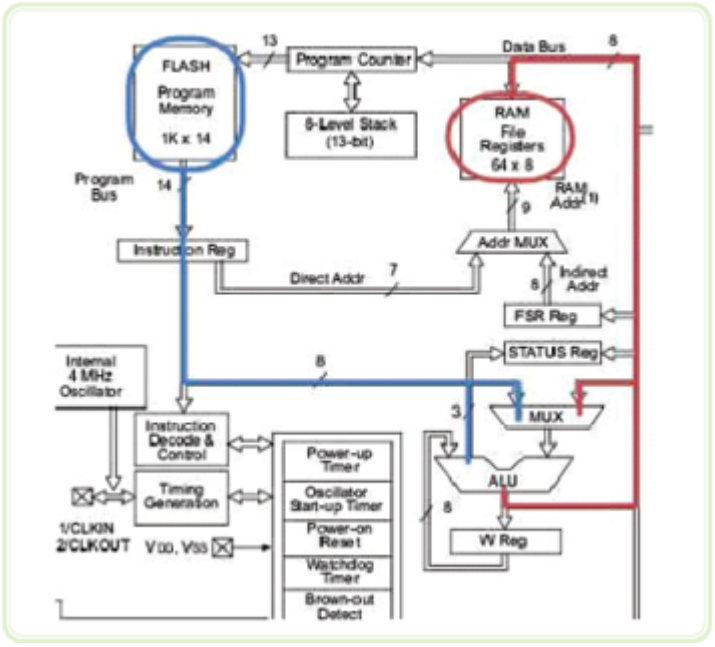
Diante disso, o objetivo principal de um sistema de controle é manter as variáveis controladas nos valores desejados pelo projeto (OGATA, 2010). Dessa forma, para controlar um robô,

Figura 4 – Arquitetura de Von Neuman



Fonte: (PENIDO, 2013) .

Figura 5 – Arquitetura de Harvard



Fonte: (PENIDO, 2013) .

é necessário primeiro definir quais variáveis serão controladas e medi-las, como pontos iniciais e finais, e então utilizar da modelagem cinemática paragarantir o movimento dos atuadores.

Para o presente projeto será utilizado o Arduino, que foi criado por Massimo Banzi em 2005 para facilitar a integração de componentes eletrônicos em projetos acadêmicos e de hobby (BANZI, 2011). O Arduino é uma família de microcontroladores com projeto elétrico e ambiente de desenvolvimento integrado livres.

2.2 Robôs móveis com rodas

Nesta seção será exposto como uma robô pode se locomover, e como sua pose, ou seja, sua posição em relação ao referencial do mundo, muda ao longo do tempo em função das suas entradas de controle.

2.2.1 Mobilidade

Segundo Corke (CORKE, 2011) um trem explicita um exemplo simples e de fácil entendimento para algumas das variáveis que serão utilizadas. Um trem pode se locomover em um trilho, e sua posição é definida pela distância de algum ponto referencial. Diante disso, uma coordenada generalizada que representa a posição do trem é $q = (x, y, \theta)$, na qual θ representa o ângulo da orientação do trem com relação as coordenadas do mundo, e *q* pertence a um conjunto de poses que podem ser alcançadas pelo veículo ferroviário. Além disso, como o comboio pode mover-se em duas direções, mas em apenas um eixo, diz-se que este possui um grau de liberdade.

Ainda segundo Corke (CORKE, 2011) a invenção da roda remonta a cerca de 3.000 a.C., enquanto o carrinho de duas rodas foi criado por volta de 2.000 a.C. Atualmente, os veículos de quatro rodas são amplamente presentes, e a população de carros no planeta está se aproximando de um bilhão. Devido à eficácia e familiaridade dos humanos com os carros, eles se tornaram uma escolha óbvia para plataformas robóticas de locomoção terrestre.

A configuração de um carro em movimento em um plano é definida por suas coordenadas generalizadas $q = (x, y, \theta) \in$ C, onde C $\subset$ R2× S e possui 3 dimensões. Um carro é controlado por apenas dois atuadores: um para avançar ou retroceder e outro para mudar a direção. Portanto, o carro é sub-atuado. Como já percebe-se, um veículo sub-atuado não pode se mover diretamente para qualquer ponto em seu espaço de configuração; ele deve seguir um caminho geralmente não linear. A partir de experiências diária com carros, sabe-se que não é possível andar lateralmente. No entanto, com prática, pode-se aprender a seguir um caminho que faça o veículo se mover para um lado a partir de sua posição inicial, o que é conhecido como estacionamento paralelo. Da mesma forma, um carro não pode girar em seu próprio eixo, mas pode seguir um caminho que resulta no veículo permanecendo na mesma posição, mas girado em 180° - um movimento de três pontos (CORKE, 2011).

Na robótica, um carro é frequentemente descrito como um veículo não holonômico, um termo derivado da matemática que indica que o movimento do carro está sujeito a uma ou mais restrições não holonômicas. Restrições holonômicas são equações expressáveis em termos das variáveis de configuração x, y e θ. Por outro lado, uma restrição não holonômica só pode ser expressa em termos das derivadas das variáveis de configuração e não pode ser integrada para se tornar uma restrição em termos das variáveis de configuração. Tais sistemas são conhecidos como sistemas não integráveis. Uma característica crucial desses sistemas é que eles não

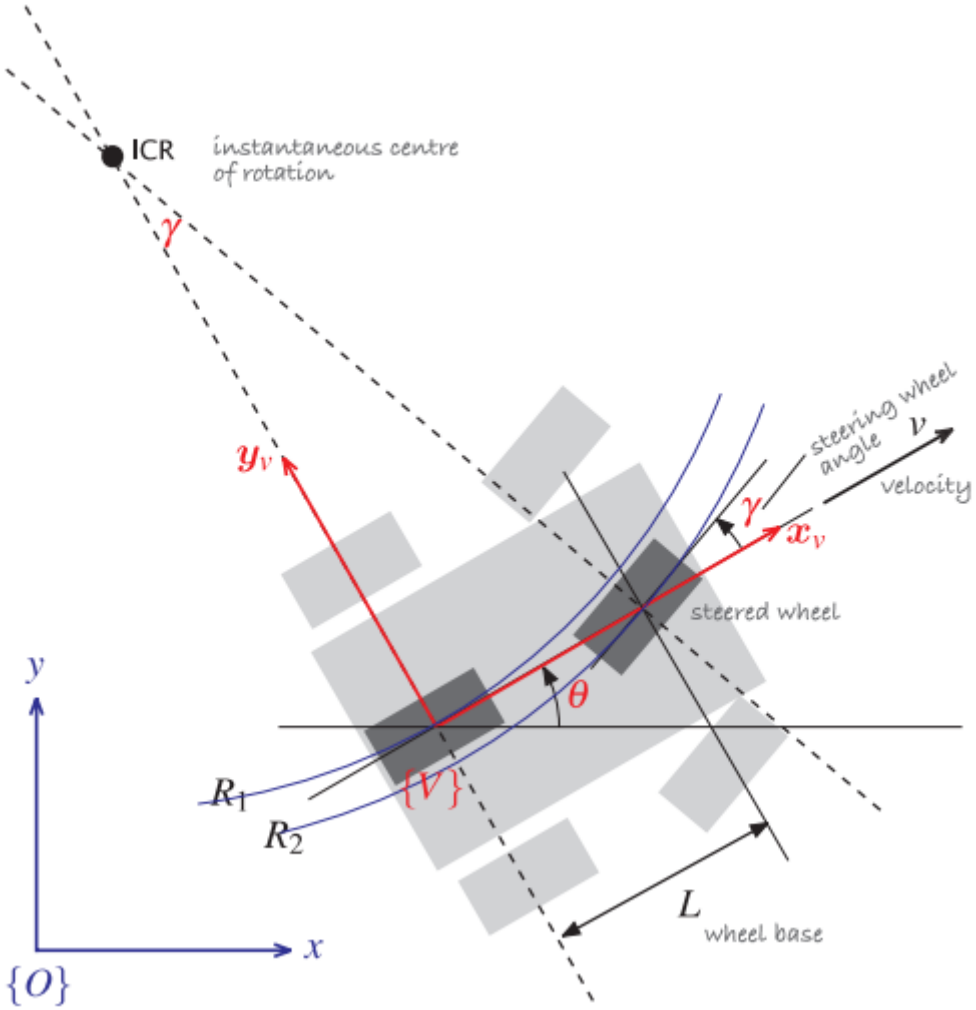
podem se mover diretamente de uma configuração para outra - precisam executar uma manobra ou uma sequência de movimentos (CORKE, 2011).

2.2.2 Robôs móveis semelhantes a carros

Nesta subseção será exposto como projetar um modelo para um veículo, e seus controladores que possam conduzir um carro até o ponto objetivo.

A Figura 6 exibe um "Modelo de bicicleta de um carro. O carro é mostrado em cinza claro e a aproximação da bicicleta está escura. A coordenada do veículo esta em vermelho e as coordenadas mundiais em azul. O ângulo do volante é γ e a velocidade das costas da roda, na direção x, é v."(CORKE, 2011).

Figura 6 – Modelo de bicicleta



Fonte: (CORKE, 2011) .

A velocidade do veículo é, por definição, v no direção x e zero na direção y, pois as rodas não podem deslizar lateralmente. Na estrutura do veículo isto é exposto nas equações a seguir.

$$\dot{v_x} = v \tag{1}$$

$$\dot{v_y} = 0 \tag{2}$$

Para um determinado ângulo fixo do volante, o carro percorre uma trajetória em forma de arco circular. Isso explica por que as curvas nas estradas são geralmente arcos circulares ou clotóides, o que facilita a condução para o motorista. Manter um ângulo constante ou suavemente variável no volante permite que o carro siga a curva da estrada de maneira mais natural. É importante notar que o raio R_2 é maior que o raio R_1, indicando que a roda externa percorre uma distância maior e, portanto, precisa girar mais rapidamente que a roda interna. Quando um veículo de quatro rodas faz uma curva, as duas rodas direcionais seguem trajetórias circulares com raios diferentes. Além disso, as equações de movimento expressas abaixo exibem o modelo cinemático (CORKE, 2011).

$$\dot{x} = v \cdot cos\theta \tag{3}$$

$$\dot{y} = v \cdot sin\theta \tag{4}$$

$$\dot{\theta} = \frac{v}{L} \cdot tan\gamma \tag{5}$$

2.3 Modelo cinemático do robô

De acordo com a estrutura do robô, as conexões cinemáticas entre as juntas limitam o movimento do seu elemento final. Portanto, diferentes conjuntos de coordenadas podem ser usados para descrever a posição e orientação do elemento final, como as coordenadas dos atuadores ou do próprio elemento final em seu espaço de trabalho. As coordenadas retangulares, cilíndricas ou esféricas podem ser usadas, dependendo da aplicação. Dessa forma, através das equações que descrevem essas conexões cinemáticas, as coordenadas dos atuadores podem ser determinadas a partir das coordenadas do elemento final (ALMEIDA, 2014) .

A coordenada de um automóvel pode ser descrita como a coordenada generalizada q, e o espaço de configurações, ou seja, o conjunto de todas as configurações de coordenadas possíveis, descreve-se como C, denota-se q $\in$ C. Além disso, diz-se que o automóvel possuí três graus de liberdade, pois pode mover-se no eixo x, no eixo y e rotacionar no eixo z. Dessa forma, C $\in \mathbb{R}^2$ (CORKE, 2011).

As equações cinemáticas são úteis não apenas para modelar dinamicamente mecanismos e desenvolver leis de controle, mas também para avaliar o espaço de trabalho do robô e identificar posições singulares. A análise de matrizes jacobianas que relacionam a derivada primeira das coordenadas dos atuadores com a derivada primeira das coordenadas do efetuador pode ser utilizada para este fim. (TSAI, 1999).

Dessa forma, para mover um robô a um ponto objetivo necessita-se de uma equação de velocidade, ângulo do robô em relação ao objetivo, o ponto cartesiano em que o robô está, o ponto objetivo, e velocidades angulares de cada roda, visto que cada uma dela deve estar em uma velocidade uma vez que traçam distâncias diferentes (CORKE, 2011).

As equações a seguir foram descritas por Corke (CORKE, 2011), e representam respectivamente a velocidade do robô (6), ângulo do robô em relação ao objetivo (7), raio de giro do centro do eixo traseiro (8),velocidade angular da roda de dentro (9), e velocidade angular da roda mais externa (10).

$$v = K_v \times \sqrt{(xo - xc)^2 + (yo - yc)^2} \tag{6}$$

$$\theta = tan^{-1} \frac{yo - yc}{xo - xc} \tag{7}$$

$$R_1 = \frac{eixo}{tan\theta} \tag{8}$$

$$vai = \frac{v}{(R_1 + eixo/2)} \tag{9}$$

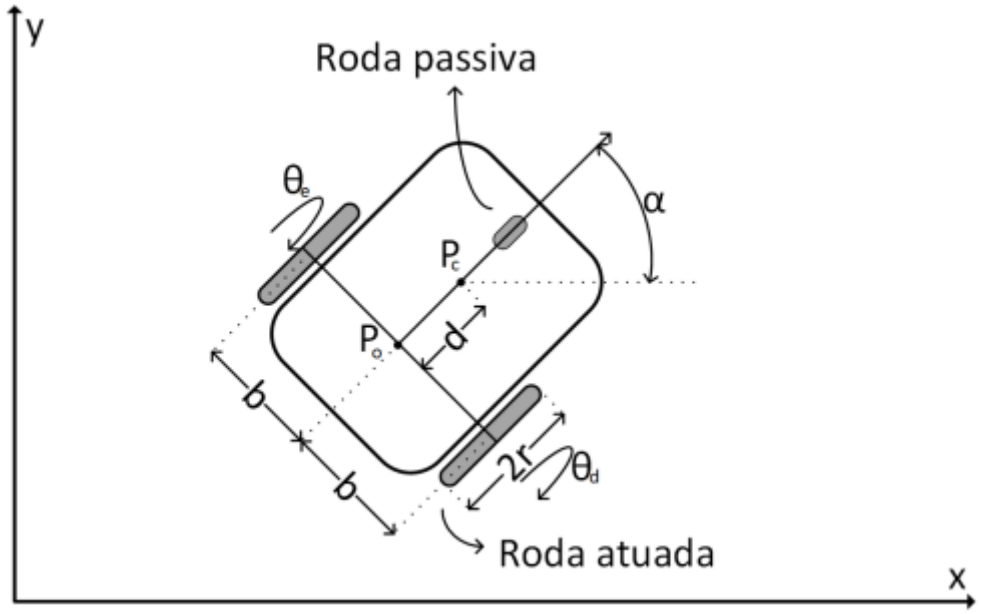
$$vae = \frac{v}{(R_1 - eixo/2)} \tag{10}$$

Para estabelecer o modelo do robô de maneira precisa, é importante ter um entendimento completo de cada componente atuador. Isso se deve à influência significativa de cada roda no movimento do robô, ao mesmo tempo que impõe restrições ao deslocamento. As rodas estão firmemente integradas à geometria do chassi, exercendo influência nas restrições do movimento do robô. É imperativo referenciar as forças e restrições de cada roda a um ponto consistente no chassi, comumente utilizando o centro de massa para essa referência. Esse ponto assume um papel fundamental em robôs móveis, pois é a partir dele que a referência de posição é estabelecida em um sistema de coordenadas globais (SIEGWART ILLAH R. NOURBAKHSH, 2011).

A Figura 7 apresenta um robô com rodas independentes e uma roda deslizante, no qual: P_c é o centro de massa do carrinho, P_o o centro de eixo das rodas, b a distância entre o centro do eixo e as rodas, d a distância entre P_o e P_c, θ_d e θ_e são, respectivamente, os ângulos de rotação das rodas direita e esquerda, (x, y) o sistema de coordenada global, $\dot{x}_o$ e $\dot{y}_o$ as velocidades no ponto P_o, $\dot{\alpha}$ a velocidade angular no ponto P_c, r é o raio da roda e v_o a velocidade linear no ponto P_o. Diante disso, deduzirá as equações (11), (12) e (13) abaixo (NAKAI, 2018).

$$\tan(\alpha) = \frac{\dot{y}_o}{\dot{x}_o} = \frac{\sin\alpha}{\cos\alpha} \Rightarrow \dot{y}_o \cos\alpha - \dot{x}_o \sin\alpha = 0 \tag{11}$$

Figura 7 – Robô móvel com rodas



Fonte: (NAKAI, 2018) .

$$r\dot{\theta}_d = v_o + b\dot{\alpha}$$
$$r\dot{\theta}_e = v_o - b\dot{\alpha} \tag{12}$$

$$v_o = \dot{x}_o \cos\alpha + \dot{y}_o \sin\alpha \tag{13}$$

A partir destas equações é possível deduzir a matriz de espaços (14), que permite transformar as velocidades angulares das rodas em relação à P_c como na equação 15 (NAKAI, 2018).

$$S_c = \begin{bmatrix} \frac{r}{2b}b\cos\alpha - d\sin\alpha & \frac{r}{2b}b\cos\alpha + d\sin\alpha \\ \frac{r}{2b}b\sin\alpha + d\cos\alpha & \frac{r}{2b}b\sin\alpha - d\cos\alpha \\ \frac{r}{2b} & -\frac{r}{2b} \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{14}$$

$$\dot{q}_1 = S_c(q_1)\dot{q}_2(t) \tag{15}$$

Além disso, as equações da lei de controle inspirada na cinemática de Kanayama (UM. . . ,) podem ser observadas nas equações (16) e (17).

$$
\begin{aligned}
v^d &= v_o r \cos(\alpha_e) + k_x x_e, \\
w^d &= w_r + v_o r (k_y y_e + k_\alpha \sin(\alpha_e))
\end{aligned} \tag{16}
$$

Na qual, v^d e w^d são respectivamente velocidade linear desejada e velocidade angular desejada, enquanto k_x, k_y, k_α são os ganhos dos respectivos erros.

$$
\begin{aligned}
x_e &= (x_r - x_o)\cos(\alpha) + (y_r - y_o)\sin(\alpha), \\
y_e &= -(x_r - x_o)\cos(\alpha) + (y_r - y_o)\sin(\alpha), \\
\alpha_e &= \alpha_r - \alpha
\end{aligned} \tag{17}
$$

Além disso, x_e e y_e são os erros nas coordenadas x e y, e α_e o erro do ângulo de orientação do robô.

Ademais, x_r é a próxima posição no eixo x , y_r a próxima posição no eixo y e α_r o futuro do ângulo de orientação. Dessa forma, V_{or} é descrito como na equação 18

$$
V_{or} = \sqrt{\dot{x}_r^2 + \dot{y}_r^2} \tag{18}
$$

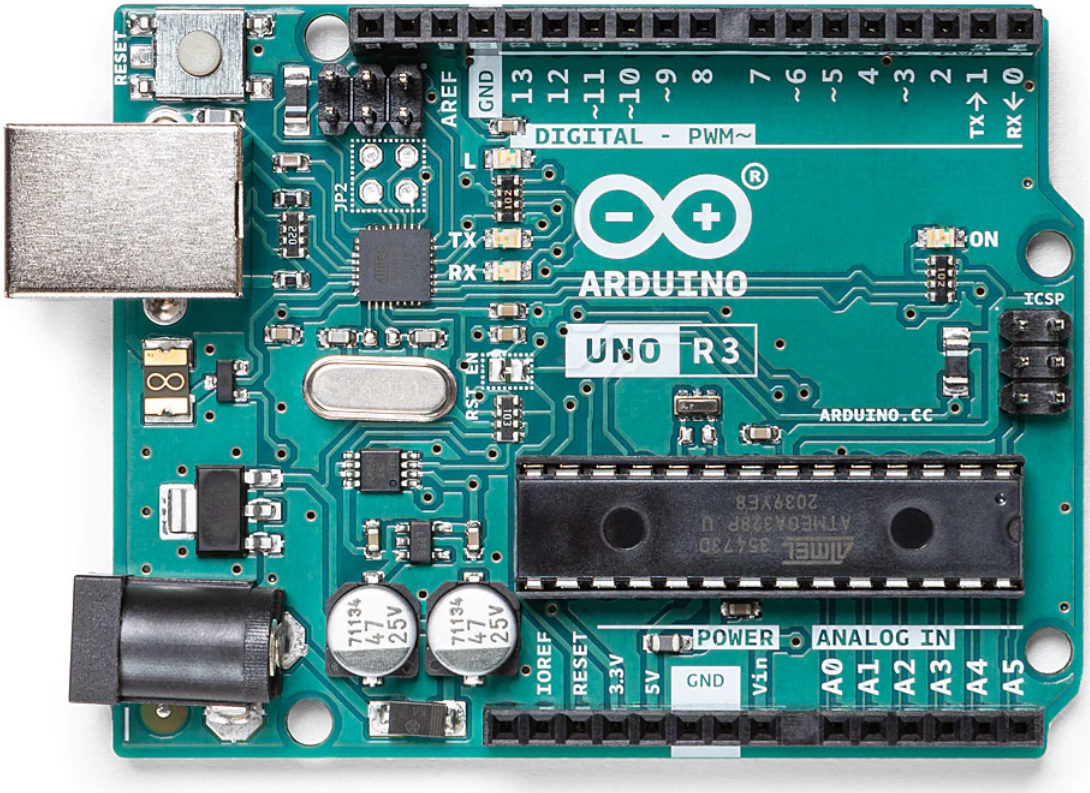
3 METODOLOGIA

O projeto em desenvolvimento contará com algumas etapas importantes, como a montagem do robô com microcontroladores e sensores, e a modelagem cinemática do robô, que inclui testes do controlador projetado.

3.1 Materiais

Para o presente projeto os materiais necessários serão um microcontrolador Arduino como na Figura 8, dois motores DC representados na Figura 9, uma ponte H para controlar os motores como na Figura 10, um chassi para o carrinho que será em material plástico, uma câmera que pode ser um celular, protoboard, câmeras e cabos conectores. A câmera podem ser substituídas sem problemas por um celular, dessa forma o processamento de imagens será feito em um computador a parte, e o Arduíno receberá os dados de entrada através do wi-fi ou bluetooth. Para que isso seja possível deverá utilizar-se de um módulo wi-fi ou módulo bluetooth no microcontrolador.

Figura 8 – Arduino Uno

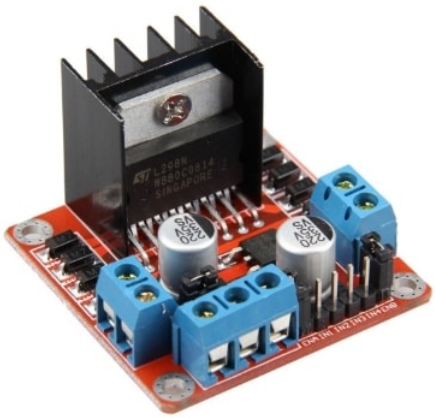


Fonte: (ARDUINO, S/D) .

Figura 9 – Motor DC

Fonte: (PROFILE, S/D) .

Figura 10 – Ponte H



Fonte: (L298N, S/D) .

3.2 Métodos

A modelagem cinemática é a principal parte do estudo, para iniciar esta etapa irá utilizar-se de um carrinho montado com peças em acrílico e os materiais citados anteriormente. Ressalta-se também que, o robô em estudo terá três graus de liberdade, ou seja, pode se mover no eixo x, y e rotacionar no eixo z.

3.2.1 Modelagem Cinemática no MATLAB

A modelagem cinemática, como ja citado anteriormente, consiste em um controle de trajetória do robô, recebendo como entrada o ponto inicial do carrinho e o ponto objetivo do trajeto.

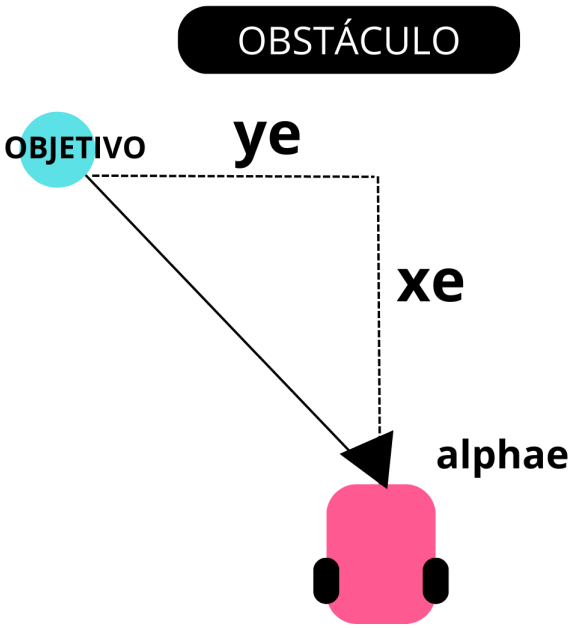
Para iniciar a modelagem utilizamos o software MATLAB (MATLAB, 2023) versão 2023a, cria-se um script e insere-se algumas variáveis e as equações citadas no referencial teórica, assim como Kx, Ky e Ka que serão variáveis de controle impostar para auxiliar no reajuste do modelo, principalmente quando a simulação estiver em prática, para reajustar o robô em

relação as modificações físicas impostas pelo ambiente. Vale ressaltar que o código pode ser encontrado no Apêndice A.

```
% a lei de controle
xe=(xr-xo)*cos(alpha)+(yr-yo)*sin(alpha)
ye=-(xr-xo)*sin(alpha)+(yr-yo)*cos(alpha)
alphar = atan2((yr-yo),(xr-xo)) % angulo da posiçao inicial em relaçã
   o a do objetivo
alphae=alphar-alpha
```

A variáveis xe e ye representam o erro em relação a x e o erro em relaçao a y, e alphae o erro em relação ao ângulo percorrido, como pode ser observado na Figura 11 a seguir.

Figura 11 – xe, ye e alphae



Fonte: Autoria própria, 2023 .

Por conseguinte, a velocidade linear será igual a distância, e por isso as restrições de controle impostas a vor, que é a velocidade linear do robô, e a wd, que é a velocidade angular serão determinadas através dos seguintes cálculos. Para vor, calculou-se para uma roda de 6.5cm de diâmetro a velocidade máxima impondo a máxima tensão, ou melhor, atribuindo o valor máximo ao motor DC de 255, e então fez-se uma marcação na roda e realizou a gravação em camera lenta, isso levou a três voltas em 4.1s, que em radianos é

$$\frac{3 * 2 * \pi}{4.1} \approx 4.59 rad/s \tag{19}$$

mas em m/s é

$$perimetro - da - roda = \frac{2 * \pi * 0.065}{2} \tag{20}$$

$$velocidade - maxima = \frac{perimetro - da - roda * 3}{4.1} \approx 0{,}149 m/s \tag{21}$$

Dessa forma, deve-se impor uma restrição a velocidade angular também, para isso mensura-se a distância entre as rodas, que é igual a 0,13m, e aplica-se a mesma fórmula, temos

$$perimetro = 2 * \pi * 0.13 \tag{22}$$

$$velocidade - maxima = \frac{perimetro - da - roda * 3}{4.1} \approx 0{,}59 m/s \tag{23}$$

Dessa forma, a modelagem possui as seguintes restrições no MATLAB.

```
% velocidade real
vor=sqrt((xr-xo)^2+(yr-yo)^2)
if vor>0.149 %condição de velocidade
    vor=0.149
end
wr=0 % velocidade angular
vd=vor*cos(alphae)+kx*xe % velocidade desejada
wd=wr+vor*(ky*ye+ka*sin(alphae)) % velocidade desejada

 if wd>0.59 %condição de velocidade angular na roda direita
    wd=0.59
elseif wd<-0.59 %condição de velocidade angular na roda esquerda
    wd=-0.59
end
```

Basta agora calculas as velocidades angulares de cada roda, e utilizar a matriz mencionada na subseção 2.3 para recalcular o ponto atual do robô considerando os erros.

```
thetad=vor+b*wd % velocidade angular da roda direita
thetae=vor-b*wd % velocidade angular da roda esquerda
% matriz de controle
Sc=[r/2/b*(b*cos(alpha)-d*sin(alpha))  r/2/b*(b*cos(alpha)+d*sin(
   alpha));...
    r/2/b*(b*sin(alpha)+d*cos(alpha))  r/2/b*(b*sin(alpha)-d*cos(
   alpha));...
    r/2/b                               -r/2/b;...
```

```
    1                                       0;...
    0                                       1]
% q1 recebe o valor de x, y e alpha  em que está
q1=Sc*[thetad thetae]'
```

Vale ressaltar que, utiliza-se das equações de lei de controle como vista na subseção 2.3, o que torna possível a modelagem cinemática do robô. Entretanto, o uso do MATLAB apenas é preferencial para verificar a modelagem e ajustar as variáveis. É importante evidenciar que xr e yr são variáveis que podem ser recebidas via bluetooth pelo Arduino.

3.2.2 Modelagem Cinemática no Arduino IDE

A segunda etapa do projeto é transcrever o código feito e testado no MATLAB em linguagem C++ para a plataforma Arduino IDE.

Em primeiro momento deve-se incluir a biblioteca de bluetooth do Arduino, e configurá-la, e em seguida incluir variáveis que serão utilizadas como controle, ou que são portas dos motores DC.

Por conseguinte, adicionamos as variáveis que serão responsáveis pela modelagem cinemática, como foram descritas no MATLAB, e na função void setup, defini-se os motores como saída. Salienta-se que, nesta seção discutirá-se trechos do código, porém o código pode ser encontrado no Apêndice B.

A próxima etapa é receber o dado via bluetooth, que terá o formato de "[número,número]", dessa forma, criou-se um laço que identifica o delimitador vírgula, e armazena apenas os números recebidos em um vetor chamado "comandos". Para realizar esta etapa é necessário verificar se recebe-se algum dado via bluetooth, como explicitado no código a seguir.

```
void loop(){
  char data;
  String num="";
  int cont =0;
  while(SerialBT.available()){
    //verifica se recebe algo do bluetooth
    data=SerialBT.read();
    if(data != '[' && data != ',' && data != ']'){
      num=data+num;
    }else if(data == ','){
      xr = num.toInt();
      num="";
```

```
13      }else if(data == ']'){
14        yr = num.toInt();
15        num="";
16        cont++;
17      }else if(data == '['){
18        num="";
19      }
20      if(cont == 1){
21      //se cont for igual a 1, significa que já possuí-se
22      //um ponto xr e yr e pode-se iniciar a modelagem //cinemática
        para este ponto
23        xr = 0.01*xr;
24        yr = (255 - yr)*0.01;
```

É importante salientar que, o número recebido é divido por cem, pois a escala utilizada na modelagem é em metros. Além disso, inverte-se o eixo y, isso ocorre pois no processamento de imagens o eixo y é invertido, pois quanto menor o número em y mais próximo a camera está do objeto, e por este motivo inverte-se o eixo no Arduino para que ele consiga se localizar como os pontos enviados.

Agora, escreve-se o laço while como no código do MATLAB para iniciar a modelagem cinemática, algumas adaptações foram necessárias, como a identificação do tipo de variável utilizada, além do formato de escrita de alguns métodos matemáticos, como a matriz e potenciações.

```
1       while (tempo == 1){
2       i=i+1;
3       float xe=(xr-xo)*cos(alpha)+(yr-yo)*sin(alpha);
4       float ye=-(xr-xo)*sin(alpha)+(yr-yo)*cos(alpha);
5       float alphar = atan2((yr-yo),(xr-xo)); //angulo da posiçao
        inicial em relação a do objetivo
6       float alphae=alphar-alpha;
7       float vor=sqrt(pow(xr,2)+pow(yr,2));
8       if (vor>0.149){
9           vor=0.149;
10       }
11      float wr=0;
12      float vd=vor*cos(alphae)+kx*xe;
```

```
13      float wd=wr+vor*(ky*ye+ka*sin(alphae));
14       if (wd>0.59){
15          wd=0.59;
16      }
17      else if (wd<-0.59){
18          wd=-0.59;
19      }
20      float thetad=vor+b*wd;
21      float thetae=vor-b*wd;
22      float transptheta[2][1]={{thetad},{thetae}};
23
24      float Sc[5][2] = { // declarando uma matriz [linhas][colunas]
25        {r/2/b*(b*cos(alpha)-d*sin(alpha)), r/2/b*(b*cos(alpha)+d*sin(
    alpha))},
26        {r/2/b*(b*sin(alpha)+d*cos(alpha)), r/2/b*(b*sin(alpha)-d*cos(
    alpha))},
27        {r/2/b,-r/2/b},
28        {1,0},
29        {0,1}
30       };
```

Dessa forma, agora deve-se escrever a equação $q1 = Sc \cdot [thatad\ thetae]^T$, porém a multiplicação de matrizes em linguagem C++ é através de dois laços for aninhados, que possibilitam a multiplicação das matrizes, isso pode ser observado no código a seguir. Além disso, receb-se a atualização do ponto xo e yo.

```
1       float soma =0;
2      //q1=Sc*transptheta;
3      for(int s=0; s<4; s++){
4        for(int s2=0; s2<1; s2++){
5            soma=soma+Sc[s][s2]*transptheta[s2][0];
6        }
7        q1[s][0]=soma;
8      }
9      xo=xo-q1[1][0];
10     yo=yo-q1[2][0];
```

```
11      alpha=alpha-q1[3][0];
```

Finalmente, os motores recebem as velocidades angulares que devem ser acionadas as rodas, com uma adaptação para os motores DC, que possuem velocidade máxima igual a 255. Dessa forma, como a velocidade máxima em tensão é de 255 e a velocidade máxima em radianos é de 4,59rad/s, o ganho para a velocidade angular será de 255/4,59 $\approx$ 55, além disso também introduziu um ganho de 10 para que os motores funcionassem efetivamente sem pouca tensão, o que ocasiona em uma rotação baixa ou nula.

```
1       speedd = 55*10*(thetad); %aumentamos a velocidade para ficar
      proporcional a velocidade analógica dos motores DC
2       speede = 55*10*(thetae);
3
4       analogWrite(motor1Pin2, speedd);
5       analogWrite(motor2Pin2, speede);
6       analogWrite(motor1Pin1, LOW);
7       analogWrite(motor2Pin1, LOW);
8       Serial.println(speedd);
9       Serial.println(speede);;
```

Diante disso, alcança-se a etapa de verificação de erro, que ao passar na verificação os motores DC são desligados, como observado abaixo.

```
1       if((abs(xe)<erro)&&(abs(ye)<erro || i>1000)){
2         tempo = 0;
3         Serial.println("fim");
4         analogWrite(motor1Pin2, LOW);
5         analogWrite(motor2Pin2, LOW);
6         analogWrite(motor1Pin1, LOW);
7         analogWrite(motor2Pin1, LOW);
8       }
9    }
10  }
```

Entretanto, foi observável que se o robô recebesse ângulo e distância de forma dinâmica, ou seja, realimentada, o controlador funcionaria baseado no erro, e assim o trajeto percorrido seria mais factível. Assim sendo, quando o robô realiza a angulação e percorre a distância, fatores como o não balanceamento da roda e o desnível do solo podem ocasionar em erros, o que torna a trajetória não exata.

Dessa forma, a realimentação do controlador fará com que ao enviar um ângulo e uma distância e o robô realizar o percurso, deverá ser enviado o novo ângulo e distância, corrigindo assim os erros. Um fluxograma de funcionamento do projeto pode ser visualizado na subseção seguinte.

Portanto, criou se uma interrupção de um segundo, o que possibilita o Arduino verificar neste intervalo de tempo se recebe alguma nova instrução e caso não receba finaliza o percurso. O trecho responsável por receber os dados via bluetooth e a modelagem cinemática ficam dentro da função de interrupção, como pode-se observar no trecho a seguir.

```
  Timer1.initialize(1000000);
  Timer1.attachInterrupt(interruptWalk);
}
void loop() {}

void interruptWalk() {
}
```

Além disso, os dados recebidos possuem o formato de "número(ângulo)/número(distância)x", no qual a '/' caracteriza que o próximo número é uma distância, e o 'x' caracteriza que o número anterior foi totalmente recebido, como observável no trecho seguinte. Vale ressaltar que, a distância recebida está em centímetros, e o Arduino calcula em metros, por isso a variável de distância será dividida por 100.

```
  String num = "";
  char data = '0';
  int count =0;

  while(SerialBT.available()){
    data=SerialBT.read();
    if(data != ',' && data != 'x' && data != '/'){
      num+=data;
    }else if(data == '/'){
      angulo = num.toFloat();
      num="";
    }else if(data == 'x'){
      distancia = num.toFloat();
      distancia = distancia/100;
      num="";
```

```
16        count+=1;
17      }
18    }
19  }
```

Assim sendo, a modelagem cinemática não necessita mais calcular o ponto em que está, pois está sendo realimentada, e com isso considera apenas o ponto em que está em relação ao robô, como se o robô sempre estivesse no ponto (0,0). Dessarte, o alphar será o ângulo recebido, e a velocidade real terá o mesmo valor da distância, caso essa velocidade ultrapasse o valor máximo possível para o motor, a velocidade máxima será de 0.149, assim como a velocidade angular. Além disso, a modelagem só ocorrerá se o contador, ou seja, a variável count for igual a 1, isso significa que o Arduino recebeu o dado inteiro recebido via bluetooth.

```
1   if(count == 1){
2     i=i+1;
3     xe=distancia;
4     alphar = angulo;
5     alphae=alphar-alpha;
6     vor=distancia;
7     if (vor>VEL_MAX){
8       vor=VEL_MAX;
9     }
10    vd=vor*(cos(alphae)+kx*xe);
11    wd=vor*(ka*sin(alphae));
12    if (wd>0.59){
13        wd=0.59;
14    }
15    else if (wd<-0.59){
16        wd=-0.59;
17    }
```

Por conseguinte, temos as velocidades angulares calculadas como no trecho seguinte, e os ganhos para as velocidades como já citados anteriormente. É importante ressaltar que os valores enviados são transformados em string, esse envio é feito apenas para verificação das velocidades quando o robô está percorrendo um circuito, e por conseguinte, após a modelagem cinemática, os motores são desligados esperando nova entradas.

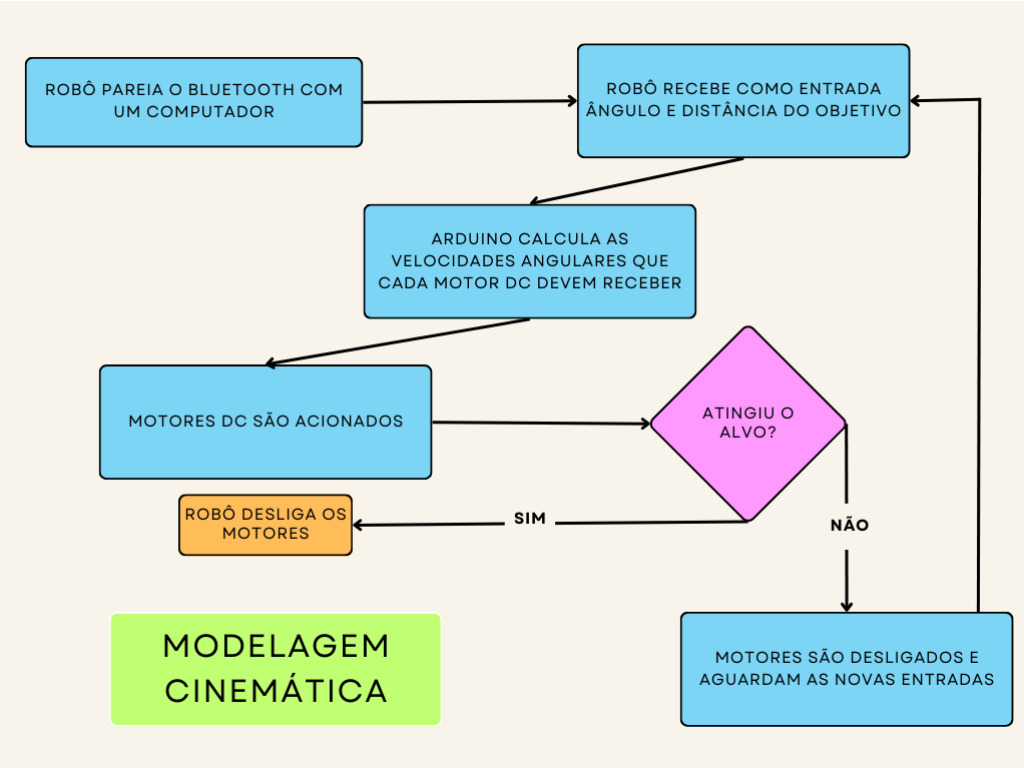
O código pode ser visualizado na íntegra no Apêndice C.

```
    thetad=vor-b*wd;
    thetae=vor+b*wd;
    speedd = 55*9*(thetad);
    speede = 55*9*(thetae);

    String myString = String(speedd);
    String myString2 = String(speede);
    SerialBT.write(myString.c_str());
    SerialBT.write(',');
    SerialBT.write(myString2.c_str());

    analogWrite(motor1Pin2, speedd);
    analogWrite(motor2Pin2, speede);
    analogWrite(motor1Pin1, 0);
    analogWrite(motor2Pin1, 0);
    myString = "";
    myString2 = "";

  }else{
    analogWrite(motor1Pin2, 0);
    analogWrite(motor2Pin2, 0);
    analogWrite(motor1Pin1, 0);
    analogWrite(motor2Pin1, 0);
  }
```

3.2.3 Fluxograma de funcionamento do projeto

De forma a explicitar melhor como a modelagem cinemática do projeto irá funcionar, criou-se um fluxograma para um melhor entendimento, como o da Figura 12

Figura 12 – Fluxograma - Modelagem Cinemática



Fonte: Autor, 2023 .

3.2.4 Projeto do Robô

Para este estudo, projetou-se um modelo 3D de um chassi para o robô, de forma a deixá-lo mais personalizado para o tipo de projeto em questão.

O andar um consiste em um espaço para a ponte H, para a roda deslizante, e para uma raspberry, além de rasgos para suporte das travas das rodas .Além disso, o andar dois possui espaços para o Arduino Mega, para a protoboard e para o celular ou camera que será utilizada. Ambos podem ser visualizados no anexo A.

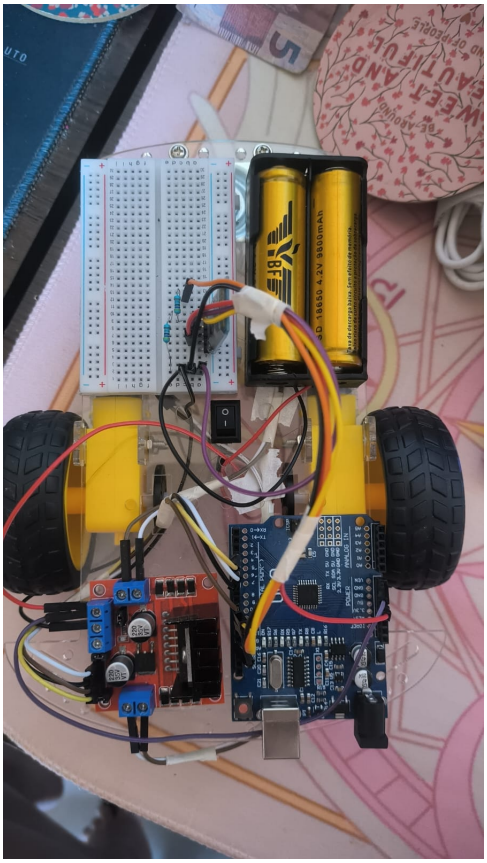
4 RESULTADOS

Neste capítulo será exposto os resultados obtidos das execuções dos cálculos, bem como a modelagem cinemática se comportou de forma simulada e como se comportou de forma prática.

4.1 Escopo do projeto

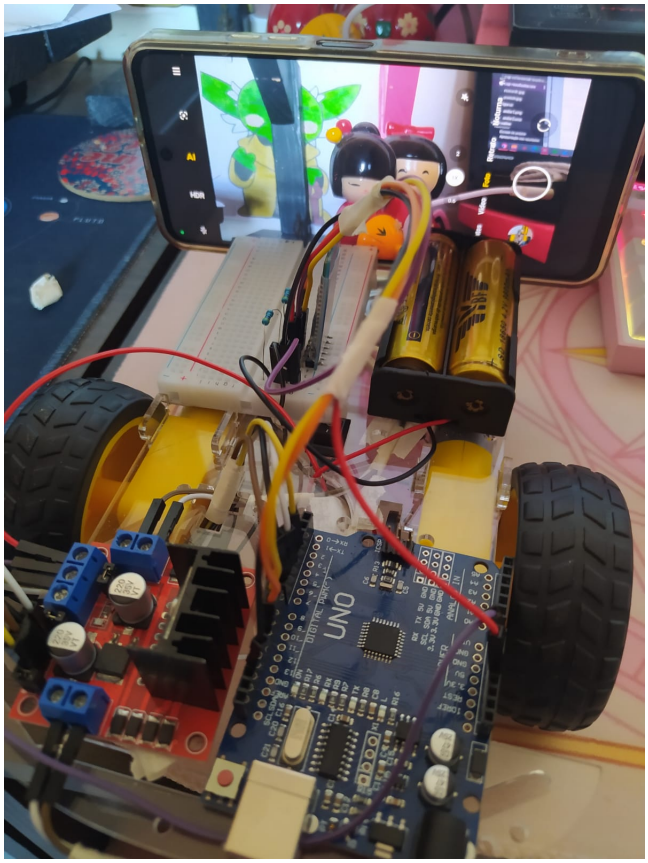
O projeto é composto de um robô construído com material plástico, ponte H, motores DC, Arduino, módulo bluetooth e protoboard. Esse possuí um código no microcontrolador que consiste em cálculos da modelagem cinemática. O robô projetado pode ser observado na Figura 13 e Figura 14, e o circuito elétrico está na Figura 15.

Figura 13 – Robô projetado



Fonte: Autoria própria, 2023 .

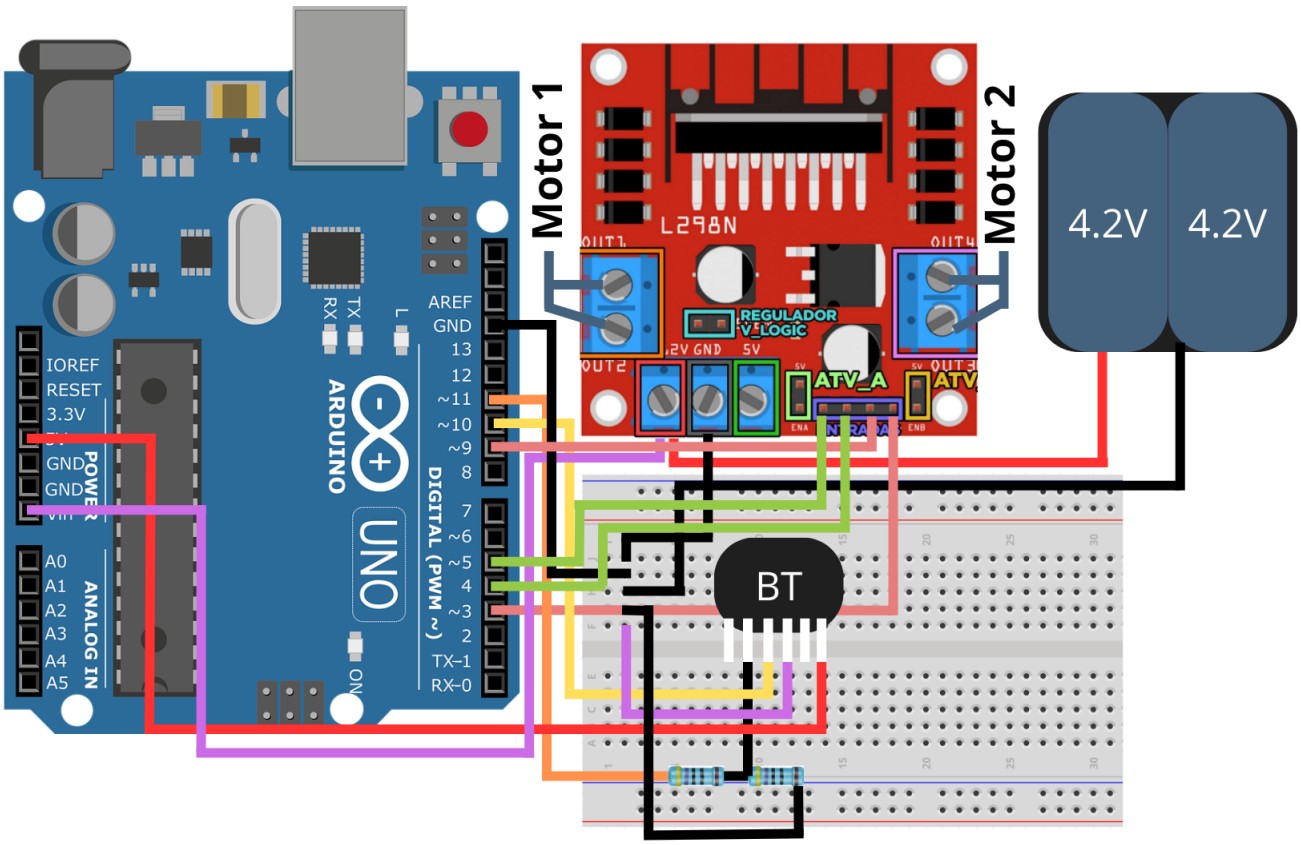
Figura 14 – Robô projetado com câmera



Fonte: Autoria própria, 2023 .

Um robô com rodas sem angulação não é capaz de se locomover no eixo y sem se mover no eixo x, por isso a modelagem cinemática é importante, para que trace rotas factíveis para o robô, e faz isso por meio dos cálculos das velocidades angulares que cada roda deve receber.

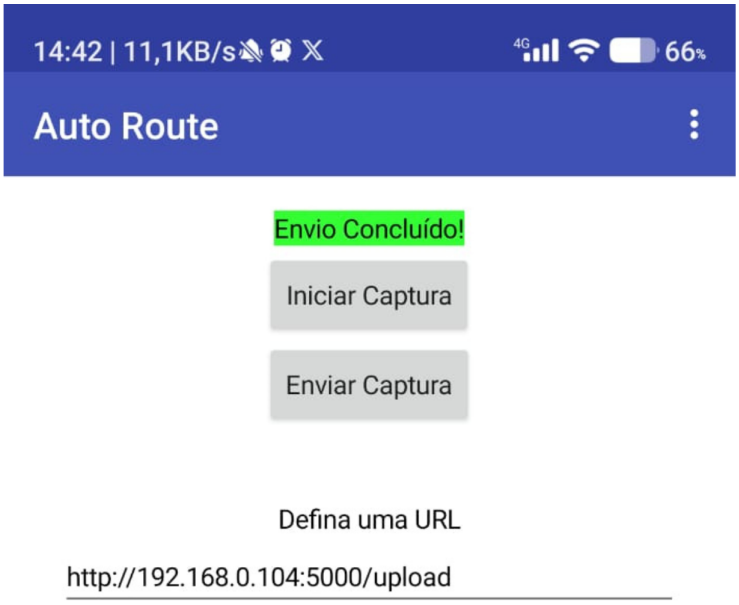
Figura 15 – Esquemático



Fonte: Autoria própria, 2023 .

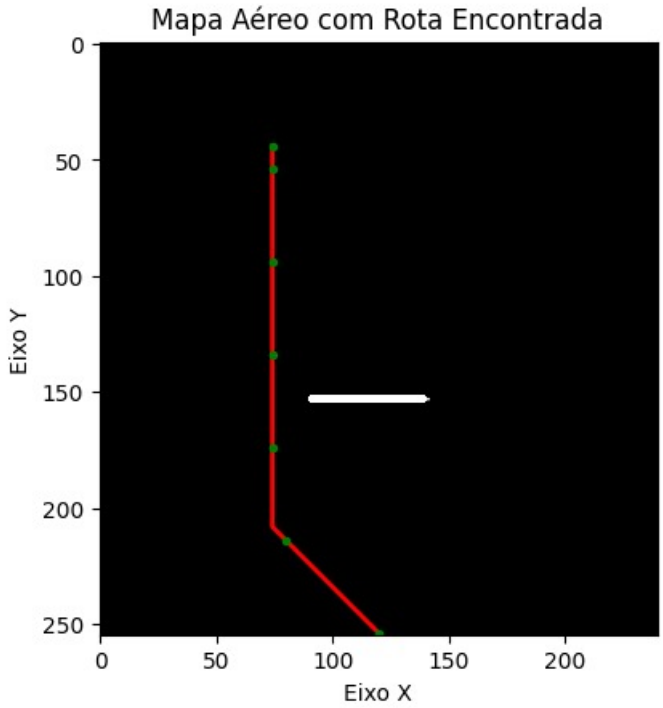
Assim sendo, o robô possui um celular com câmera na sua frente, a câmera captura a foto, ou seja, a visão do robô, e envia a imagem para um servidor que está rodando em um computador, que por sua vez processa a imagem recebida e retorna ao Arduino, que recebe via comunicação bluetooth o ângulo e distância que deve percorrer. A imagem do aplicativo utilizado para realizar a operação de captura e envio da foto pode ser observada na Figura 16. Além disso, a rota produzida pelo algoritmo pode ser visualizada na Figura 17.

Figura 16 – Aplicativo



Fonte: (SILVA, 2023) .

Figura 17 – Rota projetada pelo algoritmo A*



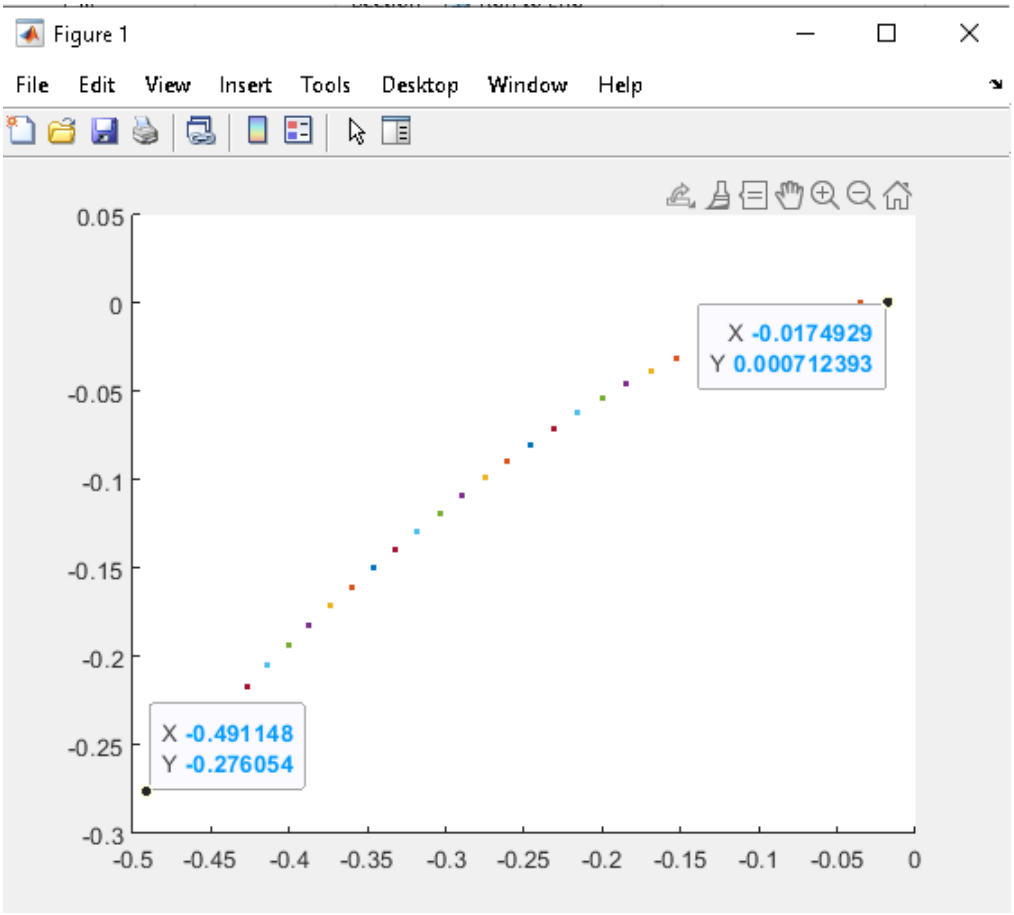
Fonte: (SILVA, 2023) .

4.2 Apresentação dos resultados

Diante do exposto, alguns testes foram executados para validação da modelagem cinemática no MatLab e também diretamente no robô projetado.

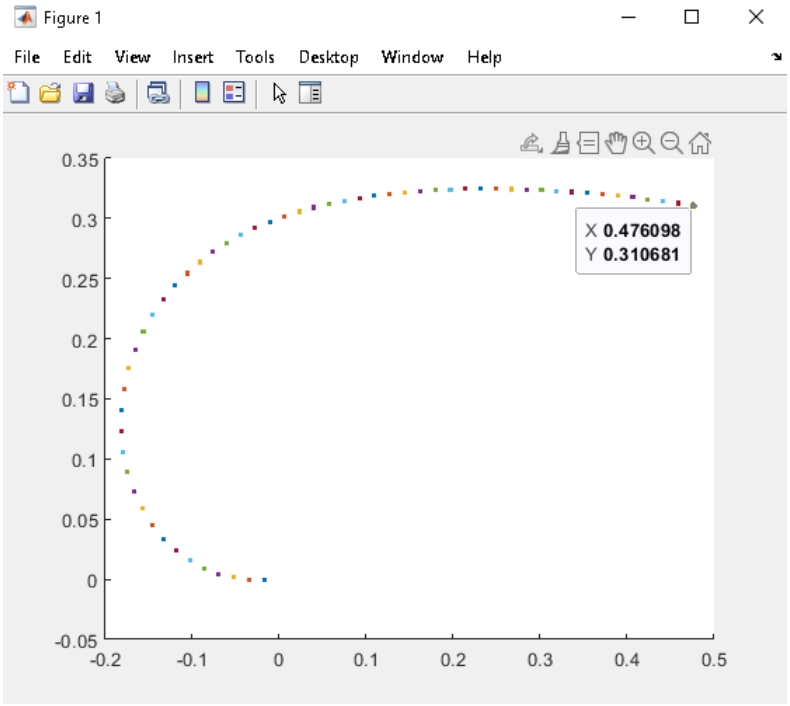
Nas Figuras 18, 19, 20 é possível observar o trajeto percorrido pelo robô, projetado através da modelagem cinemática, alcançando os pontos objetivos xr e yr. Para essas imagens foram feitos testes para os pontos (-0.5, -0.3), (0.5, 0.3) e (-0.9, 0.6).

Figura 18 – Ponto (-0.5,-0.3)



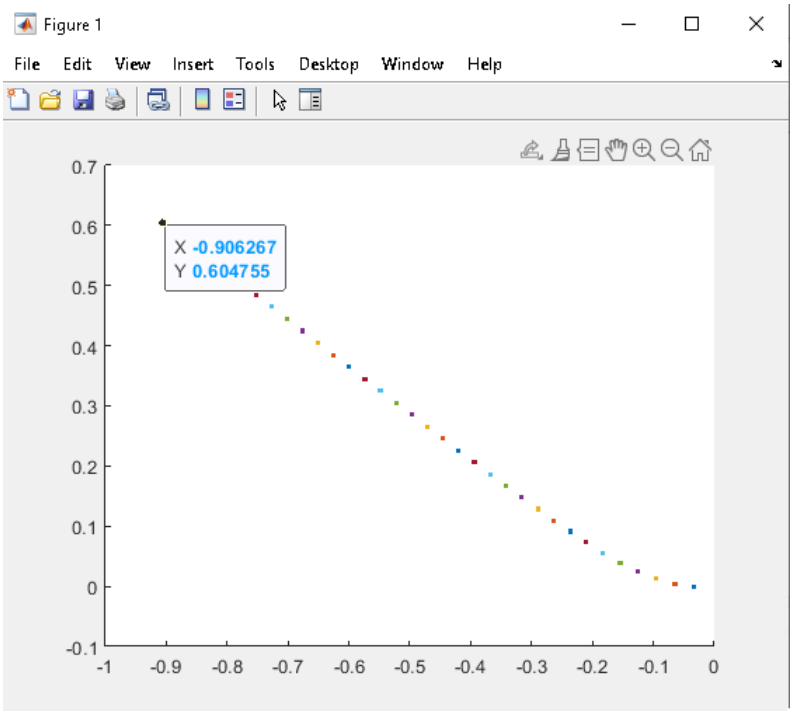
Fonte: Autoria própria, 2023 .

Figura 19 – Ponto (0.5,0.3)



Fonte: Autoria própria, 2023 .

Figura 20 – Ponto (-0.9,0.6)



Fonte: Autoria própria, 2023 .

Dessa forma, é notório que o robô busca fazer uma curva suave até o ponto, ao invés de uma reta, isso acontece devido a modelagem cinemática que propõe que um carro não pode andar de forma lateral, ou seja, no eixo y. Assim sendo, o robô projetado não possui rodas

direcionais, e sim rodas tradicionais, impedindo-o de se mover lateralmente, ou seja, apenas no eixo y, por isso dependendo de sua orientação ele deve fazer uma curva para alcançar o ponto objetivo, como foi exposto nas imagens anteriores.

Também vale ressaltar que, é notório nas imagens o espaçamento entre os pontos da trajetória, quanto mais espaçados significa que o robô pode aumentar a velocidade neste trecho, e quanto menor o espaço entre os pontos significa que o robô se locomoveu de forma mais cautelosa.

Após o carregamento no código no Arduino, pode-se então realizar os testes integrado ao trabalho de conclusão de curso do Anderson (SILVA, 2023), que com uma câmera no robô tira uma foto e processa os objetos da imagem traçando um ponto objetivo. Assim sendo, a Figura 22, Figura 23, Figura 24, e Figura 25 mostram como é a visão do robô e a Figura 21 como é o circuito planejado.

Figura 21 – Circuito Planejado

Fonte: (SILVA, 2023).

Figura 22 – Primeira visão

Fonte: (SILVA, 2023).

Figura 23 – Segunda visão

Fonte: (SILVA, 2023).

Outros dois testes foram realizados, e dessa vez mensurou-se os ângulos percorridos pelo carrinho, as velocidades angulares aplicadas em cada roda, e a distância e ângulos dese-

Figura 24 – Terceira visão

Fonte: (SILVA, 2023).

Figura 25 – Quarta visão

Fonte: (SILVA, 2023).

jados, estes que foram obtidos do algoritmo de planejamento de rota. Estas informações são

Tabela 1 – Ângulos e distâncias desejados e ângulo real

Ângulo desejado	Distância desejada	Ângulo real
-0.79	200	+/- 0.39
-0.49	200	-0.49
0.49	200	>0.32 e <0.78
0	200	0.26
0	200	0
-0.35	200	0.17
-0.65	200	0.32
0.45	200	>0.32 e <0.78
0.45	200	>0.32 e <0.78

Tabela 2 – Velocidades angulares das rodas do robô

Velocidade roda direita	Velocidade roda esquerda
89.47	58.04
84.17	63.34
63.34	84.17
73.76	73.76
73.76	73.76
81.34	66.17
87.15	60.36
64.13	83.38
64.13	83.38

explícitas na tabela 1, que expõe Ângulo desejado, distância desejada e ângulo real, e na tabela 2 estão os dados das velocidades de cada roda. Vale ressaltar, que as medidas de âmgulos reais foram realizadas com o auxilio de dois esquadros, um de 45 graus e outro de 30 graus, e com isso as medidas não são precisas.

O robô então possui a capacidade de modelar cinematicamente o trajeto proposto, isso foi notório nos testes, nos quais o carrinho percorreu o circuito planejado de forma a não colidir com outros obstáculos.

Além disso, como mencionado na Subseção 3.2.4, projetou-se um chassi 3D, entretanto, o prazo de entrega do trabalho não possibilitou essa execução, visto que a modelagem cinemática consumiu grande parte do tempo para ser exposto em prática. Ademais, várias mudanças de componentes foram realizadas durante o desenvolvimento do projeto, o que inviabiliza a impressão do chassi, que não comportará os componentes como raspberry e Arduino Mega, e sim um Arduino Uno e um celular com câmera.

5 CONCLUSÃO

Neste presente trabalho, buscou-se desenvolver um robô de baixo custo com navegação autônoma, no qual a modelagem cinemática desempenhou um papel crucial na otimização dos processos de navegação. Os objetivos específicos incluíram a criação de um modelo cinematicamente correto e a integração com as estratégias estudadas e implementadas pelo Anderson (SILVA, 2023), de processamento de imagens e predição de rotas para permitir uma navegação autônoma eficiente .

Ao longo deste trabalho, discutiu-se o conceito de modelagem cinemática em robótica, explorou-se a importância da configuração de um veículo e sua relação com a restrição não holonômica, que influencia a capacidade de um robô de se mover diretamente de uma configuração para outra.

Analisou-se a teoria por trás de veículos não holonômicos e sua descrição matemática. Esses sistemas requerem abordagens específicas para planejamento de trajetória e controle de movimento, o que é essencial para alcançar a navegação autônoma. Discutiu-se como as equações cinemáticas são fundamentais para desenvolver modelos de controle eficazes e para avaliar o espaço de trabalho do robô.

A partir do referencial teórico apresentado, pode-se estabelecer equações que descrevem a velocidade do robô, o ângulo do robô em relação ao objetivo e as velocidades angulares das rodas. Essas equações são cruciais para mover o robô em direção a um ponto objetivo e garantir que ele siga uma trajetória desejada.

Ademais, os testes práticos mostraram que o robô conseguiu se comunicar via bluetooth com um sistema de planejamento de rota e detecção de obstáculos, e assim modelou cinematicamente a rota com sucesso de forma factível.

Diante do exposto, houve também limitações, o sistema integrado ao robô tem um tempo grande de processamento, e também precisa de interferência humana para capturas de imagem e envio, visto que não houve tempo para automatizar este processo. Assim sendo, o trajeto percorrido pelo robô não está completamente dinâmico, mas possui capacidade para ser otimizado e trabalhar de forma autônoma.

Ainda assim, o projeto cumpre com o objetivo de criar um robô que possui navegação autônoma, visto que os sistemas não recebem valores estáticos, mas os calculam de acordo com as condições expostas, sem interferência humana e sem colidir com obstáculos.

Em suma, o presente trabalho contribui para o avanço da robótica, especialmente na área de navegação autônoma de robôs de baixo custo. A modelagem cinemática desempenha um papel fundamental na otimização dos processos de navegação, permitindo que robôs se movam de forma eficiente e segura em ambientes complexos. Espera-se que este trabalho seja útil para instituições científicas e empresas que buscam desenvolver tecnologias acessíveis e avançadas nessa área.

REFERÊNCIAS

ALMEIDA, R. Z. H. de. **Modelagem dinâmica e controle de robô manipulador de arquitetura paralela assimétrica de três graus de liberdade.** Set 2014. 179 p. Tese (Doutorado) — Escola Politécnica, São Paulo, Set 2014.

ARDUINO. **Arduino Uno Rev3**. S/D. Disponível em: https://store.arduino.cc/products/arduino-uno-rev3. Acesso em: 28 mar. 2023.

BANZI, M. **Primeiros Passos com o Arduino**. [*S.l.*]: Novatec Editora Ltda, 2011. v. 1.

CORKE, P. **Robotics, vision and control**: Fundamental algorithms in matlab. [*S.l.*]: Springer-Verlag Berlin Heidelberg, 2011. v. 73. 588 p. (ISBN 978-3-642-20143-1, v. 73).

CORREA, D. S. O. **Navegação autônoma de sensores de robôs móveis e detecção de intrusos em ambientes internos utilizando 2D e 3D**. Jun 2013. 96 p. Tese (Doutorado) — Instituto de Ciências Matemáticas e de ComputaçãoÁrea de ConhecimentoCiência da Computação e Matemática Computacional, São Carlos, Jun 2013.

GTP. **Geek+ Automated Forklift**. 2020. Disponível em: https://www.youtube.com/watch?v=pFsamGUITlo>,Acessoem:25mar.2023. Acesso em: 27 mar. 2023.

L298N, M. D. M. ponte H. **Ponte-H - L298N**. S/D. Disponível em: https://files.comunidades.net/mutcom/Driver_Motor_ponteH__L298N.pdf. Acesso em: 26 mar. 2023.

MATLAB. **Matemáticas. Gráficas. Programación.** 2023. Disponível em: https://la.mathworks.com/products/matlab.html. Acesso em: 1 set. 2023.

NAKAI, M. E. **Controle robusto tolerante a falhas na formação de robôs heterogêneos baseado em sistemas lineares sujeitos a saltos markovianos**. Jun 2018. 132 p. Tese (Doutorado) — Escola de Engenharia de São Carlos, São Carlos - SP, Jun 2018.

OGATA, K. **Engenharia de controle moderno**. São Paulo: Pearson Prentice Hall, 2010.

PENIDO, R. S. T. Édilus de C. C. **Microcontroladores**. Jan 2013. 80 p. — Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais, Ouro Preto - MG, Jan 2013.

PISSARDINI DANIEL CHIN MIN WEI, E. S. d. F. J. Rodrigo de S. VeÍculos autÔnomos: Conceitos, histÓrico e estado-da-arte. São Paulo, p. 13, jan 2013.

PROFILE, J. electric company. **Standard DC Series Motors**. S/D. Disponível em: https://www.contact-evolution.ch/files/DC_motors.pdf. Acesso em: 07 abr. 2023.

SEBORG D.E., T. E.; MELLICHAMP, D. **Process Dynamics and Control**. New York: Wiley, 2004.

SIEGWART ILLAH R. NOURBAKHSH, e. D. S. R. **Introduction to Autonomous Mobile Robots**. [*S.l.*]: Library of Congress Cataloging-in-Publication Data, 2011. (ISBN 978-0-262-01535-6).

SILVA, A. S. da. **Sistema autônomo de predição de rota para veículos terrestres**. Out 2023 — Universidade Tecnológica Federal do Paraná, Apucarana, Out 2023.

TESLA. **Gráficos de mortes de Tesla**. 2023. Disponível em: https://www.tesladeaths.com/deaths-chart. Acesso em: 14 set. 2023.

TSAI, L. W. **Robot Analysis**: The mechanics of serial and parallel manipulators. [*S.l.*]: John Wiley Sons, 1999.

UM método de controle de rastreamento estável para um robô móvel autônomo. *In*: PROCEEDINGS., Conferência Internacional IEEE sobre Robótica e Automação. [*S.l.*: *s.n.*].

APÊNDICE A – Modelagem Cinemática desenvolvida no ambiente de software MatLab versão 2023a

```
clear all
close all
hold on

xr=-0.5; % posicao x do objetivo
yr=-0.3; % posicao y do objetivo
xo=0; % posicao x inicial
yo=0; % posicao y inicial

tempo = 1;
erro = 0.05
i=0
r= 0.030 % raio da roda
eixo=0.12 % tamanho do eixo traseiro do robo
b=eixo/2 % metade do eixo
d = 0.01 % distancia entre o centro de massa e o centro do eixo
kx=0.005 % peso do erro em x
ky=0.5 % peso do erro em x
ka=0.5 % peso do erro em alpha
alpha=0 % orientação inicial do robo
% q1=[xc yc alpha thetad thetae]
% q2 = [ thetad thetae]
while tempo %enquanto o tempo for igual a 1
    i=i+1 % incrementa uma unidade em i, esta variavel é
    % para controle de quantidade de iterações

    % erro no eixo x, y e no angula alpha de acordo com
    % a lei de controle
    xe=(xr-xo)*cos(alpha)+(yr-yo)*sin(alpha)
    ye=-(xr-xo)*sin(alpha)+(yr-yo)*cos(alpha)
    alphar = atan2((yr-yo),(xr-xo)) % angulo da posiçao inicial em
   relação a do objetivo
    alphae=alphar-alpha

```

```
34      % velocidade real
35      vor=sqrt((xr-xo)^2+(yr-yo)^2)
36      if vor>0.149 %condição de velocidade
37          vor=0.149
38      end
39      wr=0 % velocidade angular
40      vd=vor*cos(alphae)+kx*xe % velocidade desejada
41      wd=wr+vor*(ky*ye+ka*sin(alphae)) % velocidade desejada
42
43       if wd>0.59 %condição de velocidade angular na roda direita
44          wd=0.59
45      elseif wd<-0.59 %condição de velocidade angular na roda esquerda
46          wd=-0.59
47      end
48
49      thetad=vor+b*wd % velocidade angular da roda direita
50      thetae=vor-b*wd % velocidade angular da roda esquerda
51
52      % matriz de controle
53      Sc=[r/2/b*(b*cos(alpha)-d*sin(alpha))  r/2/b*(b*cos(alpha)+d*sin(
    alpha));...
54          r/2/b*(b*sin(alpha)+d*cos(alpha))  r/2/b*(b*sin(alpha)-d*cos(
    alpha));...
55          r/2/b                               -r/2/b;...
56          1                                    0;...
57          0                                    1]
58
59      % q1 recebe o valor de x, y e alpha  em que está
60      q1=Sc*[thetad thetae]'
61      xo=xo-q1(1) %atualiza o ponto inicial x para o
62      % ponto em que se encontra
63      yo=yo-q1(2) %atualiza o ponto y inicial para o
64      % ponto em que se encontra
65      alpha=alpha-q1(3) %atualiza o angulo inicial para o
```

```
66      % angulo em que se encontra
67
68      hold on
69      figure(1)
70      % imprime de forma visual o trajeto percorrido pelo
71      % robo do seu ponto inicial até o objetivo
72      plot(xo,yo,'.')
73
74      % verificação do erro, como xe e ye podem retornar a
75      % distancia em que x/y está do x/y objetivo, utiliza-se
76      % estas variáveis para comparar com o erro que
77      % queremos. Além disso, se passar de 1000 iterações,
78      % a variável tempo recebe zero e o laço de repetição
79      % while termina.
80      if(abs(xe)<erro)&&(abs(ye)<erro || i>1000)
81          tempo = 0
82          display("fim")
83      end
84  end
```

APÊNDICE B – Código da Modelagem Cinemática desenvolvida para o hardware de Arduino na plataforma Arduino IDE

```
1
2  ////////ARDUINO////////////////////////////////
3  #include <SoftwareSerial.h>
4
5  //#if !defined(CONFIG_BT_ENABLED) || !defined(
       CONFIG_BLUEDROID_ENABLED)
6  //#error Bluetooth is not enabled! Please run `make menuconfig` to
       and enable it
7  //#endif
8  SoftwareSerial SerialBT(10, 11); // RX, TX do Arduino
9
10 int speedd = 0;
11 int speede = 0;
12 int motor1Pin1 = 5;
13 int motor1Pin2 = 6;
14 int motor2Pin1 = 9;
15 int motor2Pin2 = 3;
16 int led =13;
17 //char comandos[1];
18 ////////////////////////////////////////////
19
20 ////////MATLAB////////////////////////////////
21 float xr; //posicao do objetivo
22 float yr; //posicao do objetivo
23 float xo=0.120; //posicao inicial
24 float yo=0.254; //posicao inicial
25
26 int tempo = 1;
27 float pi = 3.14159;
28 float erro = 0.005;
29 int i=0;
30 float r= 0.032; //raio da roda
31 float eixo=0.13;
32 float b=eixo/2;
```

```
33 float d = 0.01; //distancia entre o centro de massa e o centro do
      eixo
34 float kx=0.005;
35 float ky=5;
36 float ka=5;
37 float alpha=0;
38 float q1[5][1]={ // declarando uma matriz [linhas][colunas]
39       {0},
40       {0},
41       {0},
42       {0},
43       {0}
44     };
45
46 ////////////////////////////////////////
47
48 void setup() {
49   Serial.begin(9600);
50   SerialBT.begin(9600);
51   Serial.println("The device started, now you can pair it with
      bluetooth!!!!");
52
53   pinMode(motor1Pin1, OUTPUT);
54   pinMode(motor1Pin2, OUTPUT);
55   pinMode(motor2Pin1, OUTPUT);
56   pinMode(motor2Pin2, OUTPUT);
57   pinMode(led, OUTPUT);
58   delay(100);
59   digitalWrite(led, HIGH);
60   delay(200);
61   digitalWrite(led, LOW);
62 }
63
64 void loop(){
```

```
65  char data;
66  String num="";
67  int cont =0;
68  while(SerialBT.available()){
69    // [ 23, 45 ]
70    data=SerialBT.read();
71    if(data != '[' && data != ',' && data != ']'){
72      num=data+num;
73    }else if(data == ','){
74      xr = num.toInt();
75      num="";
76    }else if(data == ']'){
77      yr = num.toInt();
78      num="";
79      cont++;
80    }else if(data == '['){
81      num="";
82    }
83    if(cont == 1){
84      xr = 0.001*xr;
85      yr = (255 - yr)*0.001;
86      while (tempo == 1){
87        i=i+1;
88        float xe=(xr-xo)*cos(alpha)+(yr-yo)*sin(alpha);
89        float ye=-(xr-xo)*sin(alpha)+(yr-yo)*cos(alpha);
90        float alphar = atan2((yr-yo),(xr-xo)); //angulo da posiçao
   inicial em relação a do objetivo
91        float alphae=alphar-alpha;
92        float vor=sqrt(pow((xr-xo),2)+pow((yr-yo),2));
93         if (vor>0.149){
94            vor=0.149;
95         }
96        float wr=0;
97        float vd=vor*cos(alphae)+kx*xe;
```

```
98          float wd=wr+vor*(ky*ye+ka*sin(alphae));
99          if (wd>0.59){
100             wd=0.59;
101         }
102         else if (wd<-0.59){
103             wd=-0.59;
104         }
105         float thetad=vor+b*wd;
106         float thetae=vor-b*wd;
107         float transptheta[2][1]={{thetad},{thetae}};
108
109         float Sc[5][2] = { // declarando uma matriz [linhas][colunas]
110           {r/2/b*(b*cos(alpha)-d*sin(alpha)), r/2/b*(b*cos(alpha)+d*
    sin(alpha))},
111           {r/2/b*(b*sin(alpha)+d*cos(alpha)), r/2/b*(b*sin(alpha)-d*
    cos(alpha))},
112           {                  r/2/b           ,                 -r/2/b
              },
113           {                    1             ,                    0
              },
114           {                    0             ,                    1
              }
115         };
116
117         float soma =0;
118         for(int s=0; s<4; s++){
119           for(int s2=0; s2<1; s2++){
120               soma=soma+Sc[s][s2]*transptheta[s2][0];
121           }
122           q1[s][0]=soma;
123         }
124         xo=xo-q1[1][0];
125         yo=yo-q1[2][0];
126         alpha=alpha-q1[3][0];
```

```
127
128         speedd = 55*10*(thetad);
129         speede = 55*10*(thetae);
130
131         analogWrite(motor1Pin2, speedd);
132         analogWrite(motor2Pin2, speede);
133         analogWrite(motor1Pin1, LOW);
134         analogWrite(motor2Pin1, LOW);
135
136         if((abs(xe)<erro)&&(abs(ye)<erro || i>1000)){
137           tempo = 0;
138           Serial.println("fim");
139           analogWrite(motor1Pin2, LOW);
140           analogWrite(motor2Pin2, LOW);
141           analogWrite(motor1Pin1, LOW);
142           analogWrite(motor2Pin1, LOW);
143         }
144       }//while(tempo==1)
145     }//if
146     cont = 0;
147   }//while(serialbt)
148 }//void loop
```

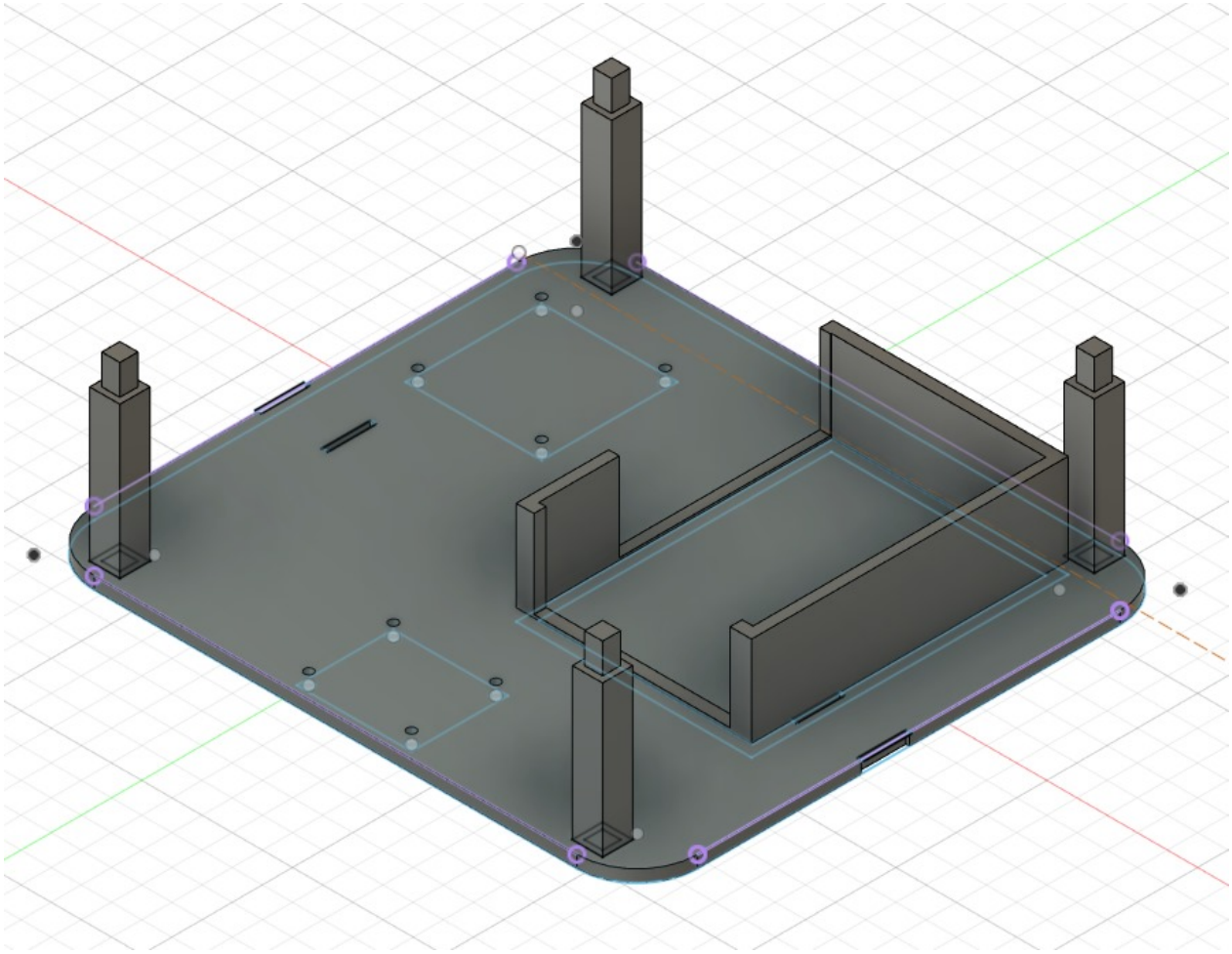
APÊNDICE C – Código da Modelagem Cinemática desenvolvida para o hardware de Arduino na plataforma Arduino IDE recebendo como entrada ângulo e distância

```
1  if(count == 1){
2     i=i+1;
3     xe=distancia;
4     alphar = angulo; //angulo da posiçao inicial em relação a do
     objetivo
5     Serial.println(alphar);
6     Serial.println(xe);
7     alphae=alphar-alpha;
8     Serial.println(alphae);
9     //vor=array_distancias[ad];
10    vor=distancia;
11    if (vor>VEL_MAX){
12      vor=VEL_MAX;
13    }
14    wr=0;
15    Serial.println(vor);
16    vd=vor*(cos(alphae)+kx*xe);
17    Serial.println(vd);
18    wd=wr+vor*(ka*sin(alphae));
19    if (wd>0.59){
20        wd=0.59;
21    }
22    else if (wd<-0.59){
23        wd=-0.59;
24    }
25    Serial.println(wd);
26    thetad=vor-b*wd;
27    thetae=vor+b*wd;
28    Serial.println(thetad);
29    Serial.println(thetae);
30    speedd = 55*9*(thetad);
31    speede = 55*9*(thetae);
32
33    String myString = String(speedd);
```

```
34      String myString2 = String(speede);
35      Serial.println(speedd);
36      Serial.println(speede);
37      SerialBT.write(myString.c_str());  // Converte a String em um
     array de caracteres
38      SerialBT.write(',');
39      SerialBT.write(myString2.c_str());
40
41      analogWrite(motor1Pin2, speedd);
42      analogWrite(motor2Pin2, speede);
43      //analogWrite(motor1Pin2, 0);
44      //analogWrite(motor2Pin2, 0);
45      analogWrite(motor1Pin1, 0);
46      analogWrite(motor2Pin1, 0);
47      myString = "";
48      myString2 = "";
49      /*if(vor == VEL_MAX && distancia > VEL_MAX){
50        angulo = 0;
51        distancia = distancia - VEL_MAX;
52      }else{
53        distancia = 0;
54      }*/
55
56    //}else{
57      //if((abs(xe)<erro)){
58        //Serial.println("FIM");
59        SerialBT.write("f");
60      //}
61    }else{
62      analogWrite(motor1Pin2, 0);
63      analogWrite(motor2Pin2, 0);
64      analogWrite(motor1Pin1, 0);
65      analogWrite(motor2Pin1, 0);
66    }
```

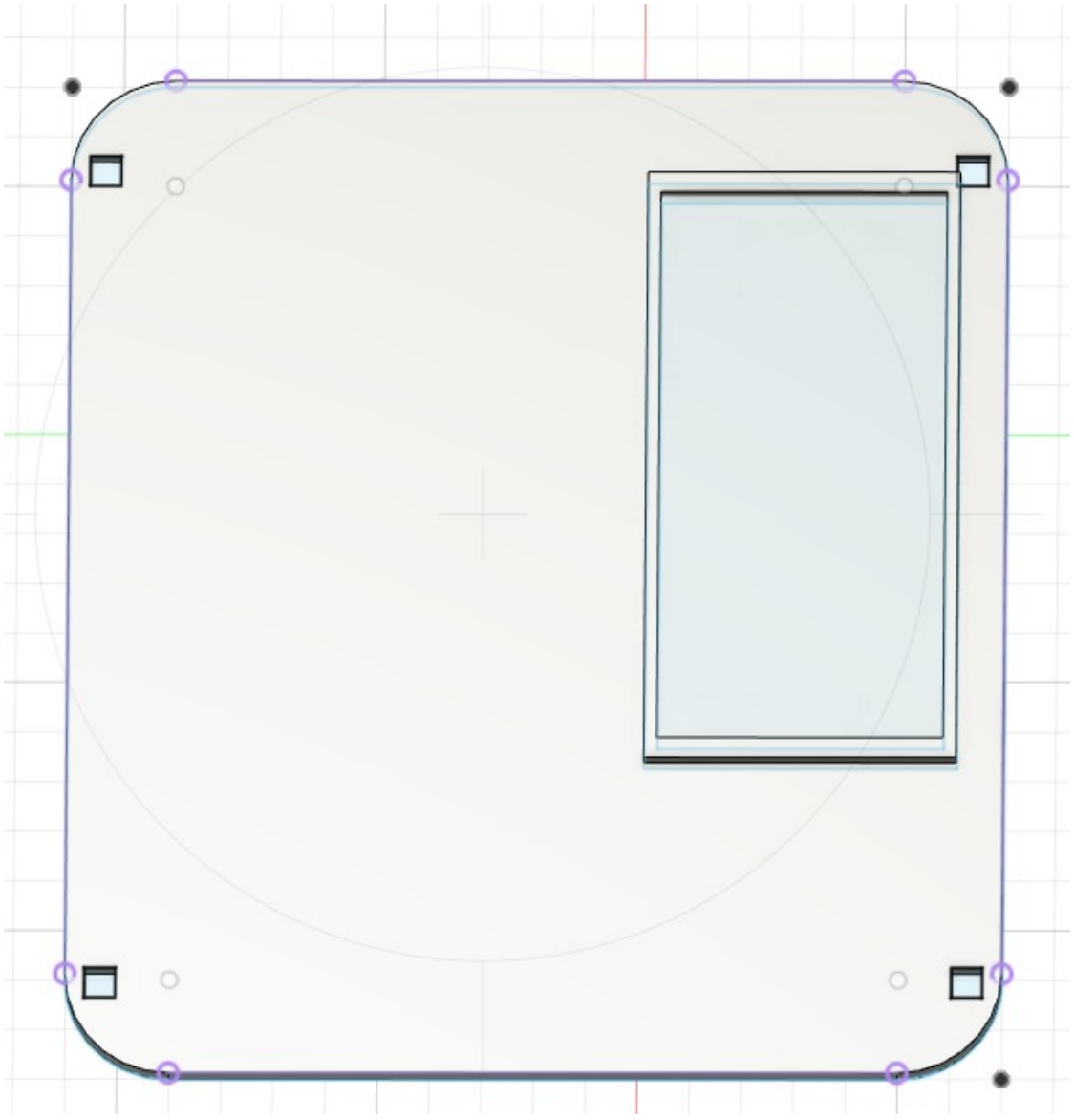
ANEXO A – Modelagem 3D

Figura 26 – Andar um do robô



Fonte: Autor, 2023 .

Figura 27 – Andar dois do robô



Fonte: Autor, 2023 .