Desenvolvimento de Robô Quadrúpede com Três Graus de Liberdade (3DOF) em Cada Perna

Gabriel Azenha Fachim, Herysson Rodrigues Figueiredo
Curso de Ciência da Computação
UFN - Universidade Franciscana
Santa Maria - RS
gabriel.afachim@ufn.edu.br, herysson.figueiredo@ufn.edu.br

***Resumo*—Este artigo aborda o desenvolvimento de um robô quadrúpede com três graus de liberdade em cada perna, controlado remotamente através de uma aplicação IoT wireless. O trabalho envolve o projeto dos componentes de hardware e software, aderindo à metodologia ágil Kanban. A estrutura mecânica do robô é projetada para se adaptar em diversos tipos de terrenos, enquanto o software de controle utiliza os cálculos da cinemática inversa para permitir a locomoção. A construção deste robô aborda os desafios da robótica móvel, particularmente em ambientes com solo irregular, onde robôs com rodas tradicionais têm desempenho inferior.**

***Palavras-chave*-robô com pernas, cinemática inversa, robótica móvel, micropython**

I. Introdução

A robótica é a área da ciência que estuda o desenvolvimento e construção do software e *hardware* dos robôs . Com a atual ascensão tecnológica, tornou-se comum o emprego de robôs em diversas tarefas, como pilotagem de veículos e aeronaves, treinamentos e serviços militares, cirurgias, manuseio de materiais de risco, exploração espacial e automação de ambientes de trabalho [1]. Diversas dessas aplicações, no entanto, necessitam que seus respectivos robôs possuam a capacidade de locomoção, o que deu origem à área da robótica móvel.

Na robótica móvel, é fundamental a definição de um método de locomoção eficiente para a tarefa a ser desempenhada pelo robô. Para aplicações que necessitam eficiência de locomoção em solos planos, a roda é o mecanismo mais comumente escolhido, visto que, quando fabricada utilizando o material adequado, possui capacidade de produzir tração suficiente para permitir locomoção em praticamente qualquer solo plano [2].

Contudo, a eficiência da roda diminui consideravelmente quando aplicada em superfícies que possuem grande variação relativa de altura no plano, como, por exemplo, elevações e/ou buracos no terreno. Por esse motivo, foram desenvolvidos diferentes métodos de locomoção robótica, como esteiras e pernas, que garantem às máquinas uma maior adaptação ao terreno [3] e, também, o deslocamento vertical do robô, garantindo opções previamente indisponíveis, como a escalada de objetos.

Tamanha autonomia de locomoção, porém, demanda a construção de uma estrutura mecânica mais complexa, bem como o desenvolvimento de um sistema de controle baseado em cálculos de análise cinemática — campo da matemática que analisa os movimentos de um corpo sem considerar as forças responsáveis por tais movimentos [4]. Além disso, robôs com múltiplas pernas, comparados com robôs tradicionais, demandam mais energia elétrica, devido à quantidade elevada de atuadores elétricos necessários, sendo comumente utilizados um por junta [5].

A capacidade elevada de travessia em terrenos não planos, no entanto, permite a aplicação de robôs com pernas em diversos cenários, como exploração de outros planetas [6] e localização e retirada de minas terrestres [7], além de operações de reconhecimento e resgate em situações de desastres naturais, entre diversos outros exemplos.

Levando em consideração as aplicações e desafios, este trabalho têm como objetivo geral a construção de um robô quadrúpede capaz de locomover-se em diferentes terrenos, além de estudar, aplicar os modelos matemáticos necessários para tal. Já como objetivo específico, esse deve ser controlado remotamente por um usuário, através de uma aplicação *wireless*.

O trabalho é dividido em várias seções que detalham o desenvolvimento dos projetos e protótipos do robô proposto. A seção subsequente é o Referencial Teórico, logo após a seção de Trabalhos Correlatos, em seguida, a seção de Metodologia, seguida pela seção de Resultados e, por fim, a Conclusão e Referências.

II. Referencial Teórico

Esse trabalho aborda conceitos da área da robótica, principalmente focando na movimentação de robôs com pernas e os cálculos de análise cinemática necessários para o controle de ângulo das juntas, permitindo, dessa forma, a locomoção do robô. Cada perna desse possui três juntas, sendo os ângulos entre essas controlados por servomotores, dando ao protótipo 3 graus de liberdade (DOF - *Degrees Of Freedom*), em espaço tridimensional, por perna.

A. Robótica

A robótica é definida como a ciência que estuda a construção de robôs, envolvendo diversas áreas e conceitos da engenharia mecânica, elétrica e mecatrônica, para montagem e funcionamento dos dispositivos, e das ciências, para desenvolvimento de softwares de controle [8].

O termo "*robot*", traduzido para robô, foi cunhado pelo escritor e dramaturgo tcheco Karel Čapek, em 1920, em uma peça de teatro chamada R.U.R (*Rossum's Universal Robots*), e deriva da palavra eslava "rabota", que significa trabalho.

Quanto à locomoção, existem diversos dispositivos efetuadores que permitem movimento [9], incluindo: Rodas, eficientes para movimentar-se em terrenos uniformes; Braços, úteis para balançar, escalar, arrastar-se; Asas e Hélices, possibilitam levantar voo; Nadadeiras, para aplicação submarina; Pernas e esteiras, eficientes para movimentar-se em terrenos não uniformes.

Portanto, nesse trabalho, que possui como objetivo projetar um robô capaz de movimentar-se em terrenos irregulares, foi contemplado apenas o uso de pernas como efetuador da locomoção do robô proposto.

B. Robôs com Pernas

Robôs com pernas são sistemas móveis que utilizam pernas como meio de locomoção. Esse tipo de robô é amplamente utilizado em situações onde é necessário locomover-se em terreno irregular, sendo considerada, inclusive, a implementação no *rover* lunar da missão apolo [10].

Um exemplo desse tipo de locomoção aplicado pode ser observado com o robô quadrúpede Spot [11], desenvolvido pela empresa Boston Dynamics, tendo em vista que o seu sistema de controle complexo, baseado em cálculos de análise cinemática, garante à Spot a habilidade de movimentar-se em diversos tipos de terrenos.

C. Análise Cinemática

A cinemática é a ciência que trata do movimento sem considerar as forças que o causam. Nela se estuda a posição, velocidade, aceleração, e as derivadas de ordem superior das variáveis de posição com relação ao tempo [12]. Devido à isso, implica-se a aplicação de um sistema de coordenadas como referencial para essas variáveis.

Em sistemas de efetuadores robóticos é utilizado, também, o conceito de cadeia cinemática, conjunto de elos conectados em cadeia através de juntas individualmente controladas. A extremidade da cadeia é comumente chamada de *end-effector* (efetor final), e sua posição e orientação no espaço podem ser manipuladas através dos cálculos da Cinemática Inversa (*Inverse Kinematics* ou IK), que pode ser calculada utilizando tanto o método algébrico, quanto o método geométrico, de acordo com Akula Umamaheswara [13].

1) IK pelo Método Algébrico: No método algébrico, todos os valores de posição e rotação das juntas são descritos e manipulados utilizando matrizes. Os cálculos são realizados utilizando matrizes de transformação relativas às coordenadas da base e resultam em um conjunto contendo todas as possíveis matrizes de posições e ângulos que levam o *end-effector* à posição desejada.

2) IK pelo Método Geométrico: No método geométrico, a geometria espacial do sistema é decomposta em diversos problemas de geometria plana, permitindo o uso de expressões trigonométricas para encontrar as possíveis soluções para os ângulos. O resultado é o conjunto dos ângulos que cada junta necessita para que o *end-effector* alcance a posição desejada.

Devido à menor complexidade do método geométrico, bem como seu maior desempenho em sistemas com poucas juntas, visto o menor número de dados que devem ser percorridos a fim de encontrar a solução desejada, esse foi escolhido para o desenvolvimento do trabalho. Portanto, é necessário definir uma solução para a IK do robô.

D. Solução Geométrica da IK

Para resolver a IK do robô proposto, de acordo com Umamaheswara Rao [13] e John J. Craig [12], deve-se primeiramente definir o formato do manipulador e um sistema de coordenadas para tal. A Figura 1 (A) mostra a representação matemática desse sistema de referência para uma das pernas do robô.

Após isso é possível determinar o ângulo da junta θ_1 e o valor de H, demonstrados na Figura 1 (B), através das equações (1) e (2) respectivamente.

$$\theta_1 = tan^{-1}(\frac{X}{Y}) \tag{1}$$

$$H = \sqrt{X^2 + Y^2} \tag{2}$$

O valor de H representa a orientação do manipulador entre os eixos horizontais, isso é, os ângulos das outras juntas serão relativos à H. Dessa forma, deve-se substituir o eixo horizontal da representação matemática por H, como na Figura 1 (C).

Assim, na equação (3), torna-se possível calcular o valor de L, que representa a distância do *end-effector* no eixo horizontal.

$$L = \sqrt{Z^2 + H^2} \tag{3}$$

O ângulo θ_3 pode ser calculado diretamente utilizando a lei dos cossenos, como na equação (4).

$$\theta_3 = cos^{-1}(\frac{(l_1)^2 + (l_2)^2 - (L)^2}{2 \times l_1 \times l_2}) \tag{4}$$

Já o ângulo θ_2 deve ser calculado através da diferença entre os ângulos A e B, como mostrado nas equações (5), (6) e (7).

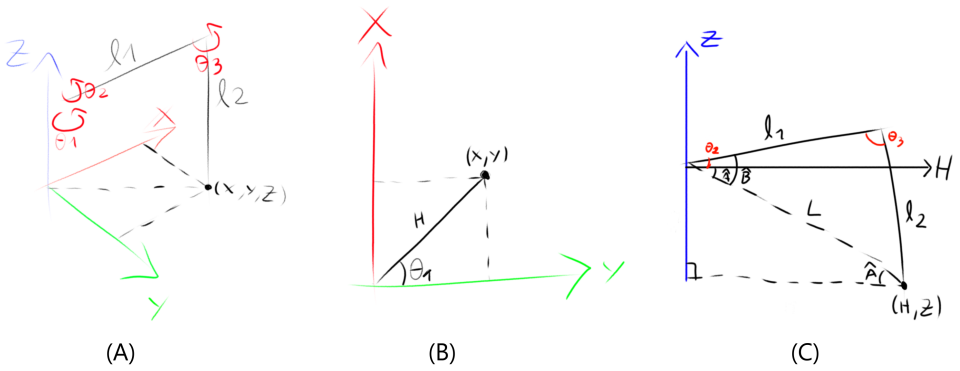


Figura 1. (A) Sistema de coordenadas de uma perna; (B) Representação matemática da visão superior de uma perna; (C) Representação matemática da visão lateral de uma perna;

$$B = cos^{-1}(\frac{(L)^2 + (l_1)^2 - (l_2)^2}{2 \times L \times l_1}) \tag{5}$$

$$A = tan^{-1}(\frac{Z}{H}) \tag{6}$$

$$\theta_2 = B - A \tag{7}$$

Vale ressaltar que os valores X, Y e Z representam a posição desejada para o efetor final no espaço físico e l_1 e l_2 são valores fixos que representam o comprimento de cada segmento da perna. Tendo isso em vista, os valores resultantes das equações (1), (4) e (7) determinam os ângulos necessários em cada junta para posicionar o pé do robô em (X,Y,Z) .

Muitos dos resultados dos cálculos da IK, porém, não podem ser utilizados, pois esses ignoram as possíveis colisões entre partes da cadeia. Dessa maneira, torna-se importante considerar a posição da estrutura física durante a execução do algoritmo de IK.

E. Modelagem e Impressão 3D

A impressão 3D é um método de fabricação aditiva, onde é feito o uso de um software *Computer Aided Design* (CAD), para modelagem 3D de um objeto, em conjunto com uma máquina *Computer Numerical Control* (CNC) capaz de interpretar as instruções de um arquivo *gcode*, nesse caso, uma impressora 3D. Um dos principais tipos dessa CNC, chamado de *Fused Deposition Modelling* (FDM), conduz filamentos de plástico através de uma extrusora, também chamada de *hotend*, aquecida até aproximadamente a temperatura de fusão do material, derretendo-o e depositando-o em camadas, construindo o formato desejado [14].

Atualmente, a tecnologia de impressão 3D tornou-se economicamente acessível, tanto pela disponibilidade de softwares CAD e tutoriais para seu uso quanto pelo surgimento de diversas linhas de impressoras domésticas de baixo custo.

Dentre os materiais utilizados na impressão 3D FDM, foi escolhido para este trabalho o filamento Ácido Polilático (PLA) que será utilizado para constituir a estrutura do robô devido à sua alta resistência mecânica. O projeto levou em consideração a montagem dos servomotores nas pernas do robô, assim como a inclusão das baterias e da unidade central de processamento, um Raspberry Pi Pico W, no interior da estrutura.

F. Raspberry Pi

Raspberry Pi é uma fundação europeia fabricante de microcomputadores, que portam o mesmo nome [15]. Em 2021 a fundação desenvolveu uma linha de dispositivos, intitulada pico, voltada especificamente para projetos de Internet das Coisas (*Internet of Things* - IoT).

O Raspberry Pi Pico e seu sucessor, o Raspberry Pi Pico W, lançado em 2022, são plataformas de desenvolvimento e prototipagem eletrônica e possuem 40 pinos, sendo 26 *General Purpose Input/Output* (GPIOs) programáveis, 3 desses conectados a um conversor analógico/digital (*analog/digital converter* ou ADC) de 16 bits, e o restante alimentação [16].

Para o desenvolvimento deste trabalho foi escolhido o modelo Pico W, devido à capacidade de conexão wireless e à integração nativa com a linguagem de programação MicroPython.

G. MicroPython

MicroPython é uma implementação da linguagem de programação de alto nível, Python [17]. Os códigos feitos em Python não são destinados a rodar em microcontroladores, tendo em vista que é uma linguagem interpretada, e, consequentemente, requer grandes quantidades de recursos computacionais, muitas vezes não disponíveis em microcontroladores [18].

No entanto, MicroPython é um compilador de códigos Python, ou seja, um software que transforma um código escrito em linguagem de máquina previamente à execução.

Tornando possível, dessa maneira, o uso de códigos Python em sistemas que possuem apenas 256kB de memória cache e 16kB de memória RAM [19].

Tendo em vista a flexibilidade, agilidade e alta escalabilidade de softwares Python, a compatibilidade nativa de MicroPython com os microcontroladores integrados às placas Raspberry Pi Pico W e a possibilidade de programação de uma aplicação IoT *wireless*, essa linguagem foi escolhida para o desenvolvimento do trabalho.

H. Aplicação IoT

Aplicações IoT são softwares que integram vários dispositivos interconectados através de rede, seja *wireless* ou cabeada. Existem diversos protocolos de comunicação, incluindo o *Hypertext Transfer Protocol* (HTTP), *Bluetooth* e o *Message Queuing Telemetry Transport* (MQQT) [20].

Visando o controle do robô proposto, neste trabalho foi projetada uma aplicação IoT *wireless* simples, utilizando *HyperText Markup Language* (HTML) e *Cascading Style Sheets* (CSS), o, bem como JavaScript (JS) e HTTP para interpretar e enviar ao Rasbperry os inputs do usuário. A partir disso, o microcontrolador deve realizar e aplicar os cálculos necessários para permitir, dessa maneira, a locomoção do robô.

III. Trabalhos Correlatos

Nesta seção serão apresentados, alguns artigos que foram relevantes, de alguma maneira, na concepção deste trabalho, seguido da contribuição dos mesmos. Os trabalhos de Rahul Rathnam [4] e Daniel Kusmenko [21] auxiliaram na descoberta de métodos de resolver a IK, enquanto o artigo de Martin Zoula [5] apresentou aspectos estruturais importantes para a construção do robô.

A. Data Driven Approach for Inverse Kinematics in 2D and 3D

O trabalho de Rahul Rathnam investiga métodos de cinemática inversa (IK) para robôs com 7 e 8 graus de liberdade em 2D e 3D, usando *Machine Learning* — redes neurais, árvores de decisão e florestas aleatórias — para melhorar a precisão com base em dados de cinemática direta. *Forward And Backward Reaching Inverse Kinematics* (FABRIK) foi o método mais eficiente em 2D, enquanto em 3D todos os métodos mostraram limitações de precisão. O artigo contribuiu ao comparar algoritmos consolidados de IK e sua eficácia em diferentes cenários.

B. Design, Construction, and Rough-Terrain Locomotion Control of Novel Hexapod Walking Robot With Four Degrees of Freedom Per Leg

O trabalho de Martin Zoula apresenta o *Hexapod Ant Robot* (HAntR), um robô hexápode com seis pernas e 4 graus de liberdade por perna, projetado para navegação eficiente em terrenos irregulares. O design e os algoritmos de controle permitem estabilidade em diversas superfícies, incluindo inclinadas. O HAntR busca melhorar velocidade, confiabilidade e resistência, operando por mais de uma hora com 85% de seu peso em carga e atingindo 87% de sua velocidade nominal. Esse trabalho contribuiu fornecendo informações valiosas sobre a construção e o aprimoramento de robôs hexápodes.

C. Development of an analytical inverse kinematics for a 5 DOF manipulator

No artigo de Daniel Kusmenko, é apresentada uma abordagem analítica para resolver a cinemática inversa de um braço robótico com 5 DOF, comparando sua precisão e tempo de execução com métodos numéricos. A solução analítica demonstrou ser mais precisa e rápida, devido à menor demanda de processamento e à capacidade de ser aplicada a cadeias cinemáticas de diferentes comprimentos.

Testes com Matlab e um microcontrolador mostraram que essa abordagem supera a numérica em cenários com soluções únicas. A generalização dos cálculos facilita sua aplicação em diversos efetuadores robóticos, como pernas. Esse estudo forneceu a base para o desenvolvimento de um robô quadrúpede, destacando a necessidade de uma metodologia ágil para o gerenciamento do projeto.

IV. Metodologia

Esta seção detalha o processo de desenvolvimento do projeto, incluindo a construção de um modelo geral, levantamento de requisitos, projeto de software e protótipos, começando, porém, pela definição da metodologia ágil.

A. Metodologia Ágil

No desenvolvimento de software, é comum aplicar a filosofia ágil, que prioriza a entrega incremental, a criação de artefatos mínimos de engenharia e, principalmente, a agilidade que, segundo Pressman, é definida como a facilidade de acomodar mudanças no projeto e a simplicidade no desenvolvimento geral [22].

Existem diversos métodos ágeis de desenvolvimento de software, cada qual com especificações variadas, dentre os quais, para o desenvolvimento deste trabalho, foi escolhido o método Kanban devido aos poucos artefatos de software necessários e incentivo à prototipação.

O Kanban nasceu na Toyota, em 1960, como uma prática da engenharia industrial, e consiste em um quadro disposto em um local acessível da fábrica, onde os funcionários escolhiam tarefas, alternando o estado dessas entre "não feito", "fazendo" e "feito", conforme o andamento do projeto. Recentemente essa metodologia foi adaptada para o desenvolvimento de software ágil por David Anderson [23].

B. Modelo Geral

A partir da definição do método ágil, foi idealizado um modelo geral do trabalho, isto é, um esboço inicial

contendo a demonstração simplificada do projeto. A Figura 2 (A) mostra um esboço das pernas do robô quadrúpede. O desenho indica a disposição das quatro pernas em relação ao corpo do robô. As pernas estão dispostas de forma simétrica em torno do corpo central, com o ângulo de rotação θ_1 evidenciado.

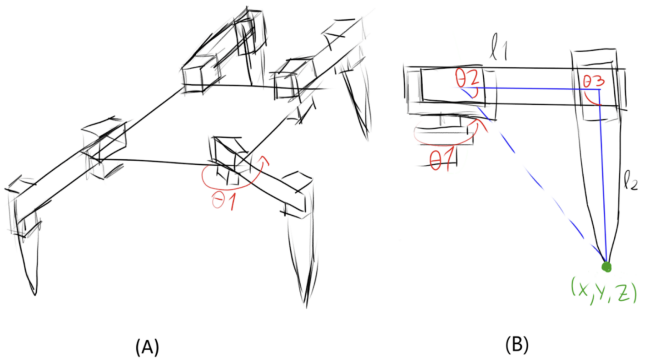


Figura 2. (A) Esboço do *layout* das pernas do robô; (B) Esboço do funcionamento de uma perna do robô.

Já a Figura 2 (B) demonstra o funcionamento de uma única perna, onde l_1 e l_2 representam o comprimento de cada segmento da perna. Além disso, os ângulos θ_1, θ_2 e θ_3 representam as juntas, controladas por atuadores elétricos. Essas juntas formam uma cadeia cinemática, cujo efetor final, nesse caso, o ponto da perna que faz contato com a superfície, é representado pelo ponto (x, y, z). Tendo em vista que em uma perna existem três juntas com ângulos livres, isso é, cuja rotação relativa independe das juntas anteriores, pode ser afirmado que cada perna do robô possui 3DOF.

A construção desse modelo geral permitiu uma visualização mais clara do funcionamento do projeto, o que foi crucial para identificar e entender melhor os componentes e interações do sistema. Essa visualização facilitou a etapa de levantamento de requisitos ao permitir a identificação de necessidades específicas de hardware e software, bem como a antecipação de possíveis desafios e limitações técnicas. Com esta visão do modelo, foi possível detalhar as funcionalidades e características desejadas ao levantar os requisitos.

C. Requisitos

As especificações de um sistema requisitadas pelo cliente podem ser divididas, de acordo com Ian Sommerville [24], entre dois grupos: Requisitos Funcionais (RF), especificações das funções que um sistema deve ser capaz de realizar; e Requisitos Não Funcionais (RNF), restrições e características aplicadas ao sistema como um todo. Para o desenvolvimento desse trabalho foi feito um levantamento de requisitos, assumindo o autor como cliente, cujo resultado pode ser observado na Tabela I.

Após a etapa de levantamento de requisitos, onde o *backlog* do Kanban foi construído, organizado e priorizado, foi iniciado o projeto de software do sistema, a fim de planejar o processo de desenvolvimento em função dos requisitos levantados.

Tabela I
RESULTADO DO LEVANTAMENTO DE REQUISITOS DO ROBÔ QUADRÚPEDE

RF01	Resolver a cinemática inversa para os ângulos de uma perna com 3DOF
RF02	Sincronizar movimento de múltiplas pernas
RF03	Sustentar o próprio peso
RF04	Movimentar-se em todas as direções
RF05	Rotacionar em relação ao próprio centro
RF06	Comunicar-se com sistema IoT de controle
RF07	Realizar calibração de alcance máximo das pernas
RF08	Salvar *log* de calibração
RF09	Ser controlado por aplicação web
RF10	Salvar *log* de movimentação
RNF01	Possuir quatro pernas
RNF02	Cada perna deve possuir 3 DOF
RNF03	O controlador deve ser um Raspberry Pi Pico W
RNF04	O robô deve ser controlado por rede
RNF05	O controlador deve ser programado em MicroPython
RNF06	A estrutura do robô deve ser impressa em 3D
RNF07	O tempo de resposta dos comandos de movimentação deve ser mínimo

D. Projeto de Software

Os diagramas *Unified Modeling Language* (UML) são ferramentas utilizadas para a construção de projetos de software, a fim de apresentar as visões de um dado sistema. A aplicação de metodologias ágeis, como o Kanban, normalmente possuem uma diminuição na quantidade de artefatos gerados, por isso este trabalho contempla apenas a representação de domínio [22] (Figura 3).

O diagrama de domínio do software de controle do robô, contém quatro componentes: Trajetoria, responsável por armazenar informações sobre a trajetória que o robô percorre em cada terreno; Robo, responsável pelo controle do robô como um todo; Perna, que possui os atributos que permitem os cálculos de IK necessários para realizar a movimentação de cada perna individual; e Servo que possui a função de controlar os ângulos dos atuadores elétricos presentes nas pernas do robô.

Dessa maneira, a partir do diagrama na Figura 3, é possível representar de maneira visual o software de controle de um robô quadrúpede que possui 3 DOF por perna e armazena os dados de suas trajetórias. Após a conclusão desta etapa, foi iniciado o desenvolvimento do software do

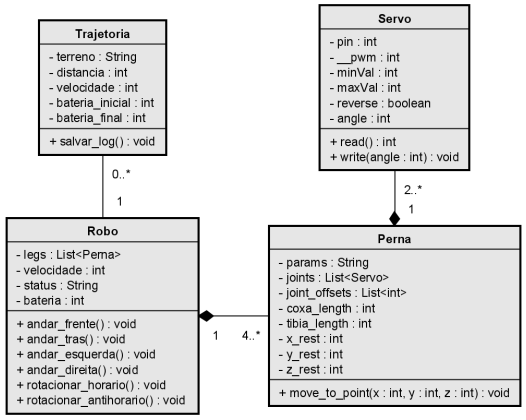


Figura 3. Diagrama de domínio do software do robô.

robô, a fim de possibilitar o controle do protótipo e, dessa forma, testá-lo.

E. Implementação do Software

Para realizar o movimento do robô é necessário implementar os cálculos das equações (1) a (7) a partir de software. O trecho de código apresentado no Listing 1 mostra a implementação em MicroPython da IK na classe Perna do software do robô.

Listing 1. Implementação da Cinemática Inversa em MicroPython

```
1  import math
2
3  def cosine_law(a, b, c):
4      nom = ((a**2) + (b**2) - (c**2))
5      dem = (2 * a * b)
6      return math.acos(nom/dem)
7
8  # Classe Perna
9  # ...
10 def move_to_point(self, x=0, y=0, z=0):
11     # Comprimento do segmento l1
12     j2l = self.l1
13     # Comprimento do segmento l2
14     j3l = self.l2
15     # Angulo da junta 1
16     aj1 = math.atan(x/y)
17     h = math.sqrt((y**2) + (x**2))
18     L = math.sqrt((h**2) + (z**2))
19     # Angulo da junta 3
20     aj3 = cosine_law(j2l, j3l, L)
21     b = cosine_law(L, j2l, j3l)
22     a = math.atan(z/h)
23     # Angulo da junta 2
24     aj2 = b + (-a)
```

O método da classe Perna, *move_to_point* (linha 10), espera como parâmetro os valores das distâncias nos eixos x, y e z. As variáveis j2l e j3l (linhas 12 e 14 respectivamente) possuem os valores de l_1 e l_2 (Figura 2 (B)). As variáveis aj1, aj2 e aj3 (linhas 16, 24 e 20) são utilizadas para armazenar os ângulos calculados através da IK e, posteriormente, aplicá-los aos motores. Nas linhas 20 e 21 é utilizada a função *cosine_law* (que calcula a lei dos cossenos a partir dos comprimentos dos segmentos a, b e c de um triângulo), implementada na linha 3.

Listing 2. Método levantar da classe Robo

```
1 # classe Robo
2 # ...
3 def levantar(self):
4     self.perna1.move_to_point(0, 80, 50)
5     self.perna4.move_to_point(0, 80, 50)
6     self.perna2.move_to_point(0, 80, 50)
7     self.perna3.move_to_point(0, 80, 50)
```

Com a implementação da cinemática inversa concluída, foi possível desenvolver os diversos métodos de movimento do robô. O trecho de código do Listing 2 exemplifica o método *levantar*, da classe Robo. Esse método utiliza a função *move_to_point* de classe Perna para aplicar os mesmos valores para x, y e z (0mm, 80mm e 50mm, neste caso) em cada uma das pernas, permitindo a elevação do corpo do robô.

Em seguida, foi iniciada a fase de construção dos protótipos, a fim de validar e testar os conceitos definidos.

F. Protótipos e Montagem

Durante o desenvolvimento do trabalho a prototipagem foi adotada juntamente ao Kanban como método de validação dos requisitos do projeto. Graças a isso, houve múltiplas iterações do robô quadrúpede.

A Figura 4, por exemplo, mostra as duas versões do robô evidenciando as posições dos atuadores, bem como a posição do Rasbperry relativa ao corpo do robô. Cada perna é composta por três atuadores, neste caso, servomotores. Esses motores permitem o controle do ângulo de cada segmento da perna por meio de sinais elétricos, sendo responsáveis pelo movimento preciso dos segmentos conectados.

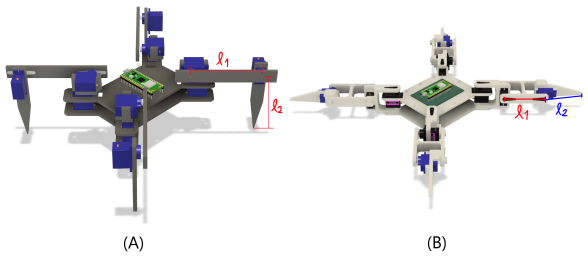


Figura 4. (A) Primeira versão do robô quadrúpede; (B) Segunda versão do robô quadrúpede.

Os comprimentos dos segmentos l_1 e l_2, representados na Figura 2 (B), foram ajustados de maneira que $l_1 \approx l_2$. Dessa forma sendo, na Figura 4 (A), 80 milímetros cada e na Figura 4 (B), l_1 = 61.3 e l_2 = 63.75 milímetros.

Para realizar a montagem e os testes do robô, foi necessário idealizar e construir um circuito elétrico capaz de permitir ao Raspberry o controle de todos os motores, bem como medir a quantidade de bateria restante.

A Figura 5 mostra um diagrama de circuito contendo todas as ligações elétricas entre os doze servomotores e

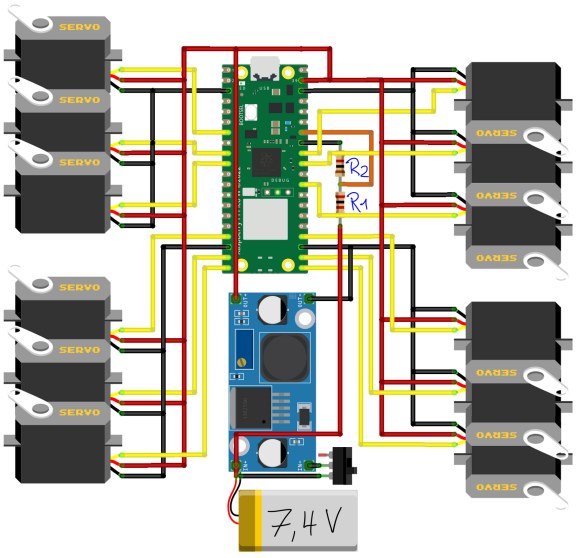


Figura 5. Diagrama do circuito do robô.

seus respectivos pinos GPIO do Raspberry. Além disso, é possível observar uma bateria de 7.4 volts, com o terminal negativo conectado a um interruptor, utilizada para alimentar o circuito. Devido à tensão máxima de alimentação do Raspberry de 5.5 volts, é necessário conectar a bateria aos pinos de entrada de um regulador de tensão (componente azul) ajustado para reduzir o valor da tensão de saída. O terminal de saída negativo do regulador é conectado a qualquer pino *ground* (GND), enquanto o positivo é conectado ao pino VSYS, destinado à entrada positiva de alimentação.

Visando a possibilidade de medir a quantidade de bateria restante, o terminal positivo da bateria é conectado a um circuito divisor de tensão, ou seja, uma associação de resistores em série (representados por R_1 e R_2), onde a diferença de potencial entre os resistores e o terminal negativo pode ser calculado, de acordo com Simon Monk [25], a partir da Equação 8.

$$V_{saida} = \frac{V_{entrada} \times R_2}{R_1 + R_2} \tag{8}$$

Tendo em vista que a tensão de entrada máxima dos pinos GPIO do Raspberry é de 3.3 volts e a tensão da bateria é 7.4 volts, é possível determinar os valores das resistências a partir do cálculo mostrado em 9.

$$\begin{aligned} \frac{V_{saida}}{V_{entrada}} = \frac{3.3}{7.4} = \frac{R_2}{R_1 + R_2} \approx 0.44 \\ R_2 = 0.44 \times (R_1 + R_2) = 0.44R_1 + 0.44R_2 \\ R_2(1 - 0.44) = 0.44R_1 \Rightarrow \frac{R_2}{R_1} = \frac{(1 - 0.44)}{0.44} \approx 1.3 \\ R_1 = 1.3R_2 \end{aligned} \tag{9}$$

Considerando possíveis variações na tensão da bateria e a disponibilidade de componentes, a relação dos resistores foi ajustada para $R_1 = 2R_2$. Os valores de R_1 e R_2 foram definidos como 2KΩ e 1KΩ respectivamente. Aplicando esses valores na Equação 8 é possível observar que, com carga de 7.4 volts, de acordo com a Equação 10 a tensão de saída do divisor é de aproximadamente 2.4 volts.

$$V_{saida} = \frac{7.4 \times 1000}{2000 + 1000} \approx 2.4V \tag{10}$$

O terminal entre os resistores foi então conectado ao GPIO 28 do Raspberry, que é conectado ao ADC da placa, permitindo a leitura da tensão da bateria através de software.

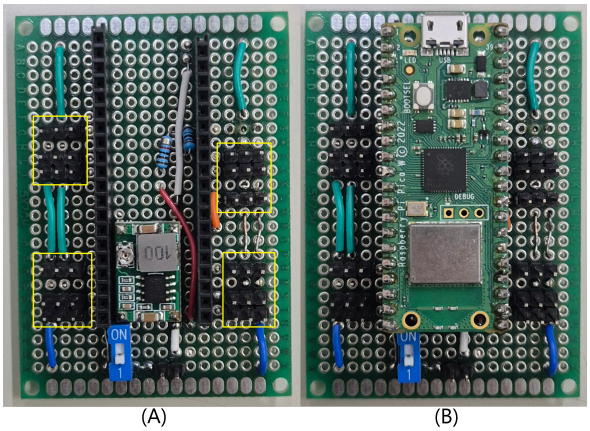


Figura 6. Placa de prototipagem com componentes soldados. (A) Raspberry retirado para melhor visualização; (B) Raspberry adicionado novamente.

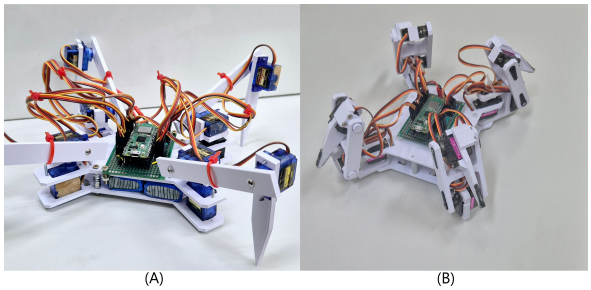
Os componentes foram, então, soldados em uma placa de prototipagem a partir do diagrama da Figura 5, como mostra a Figura 6. Destacado em amarelo estão conectores compatíveis com os servomotores, adicionados para facilitar a manutenção e troca dos atuadores.

Foi construído, então, o protótipo, disposto na Figura 7 (A). O modelo 3D do robô foi impresso com filamento PLA branco, e nele foram montados os atuadores elétricos. Além disso, na placa de prototipagem (Figura 6), que foi disposta ao centro da plataforma superior, foram conectados o Raspberry, atuadores e a bateria, seguindo o esquema eletrônico.

O Raspberry foi então programado para alterar os ângulos dos atuadores para o valor fixo de 90º, permitindo a montagem das pernas nos ângulos corretos, e a bateria foi posicionada acima da plataforma inferior, finalizando a construção do primeiro protótipo do robô.

Durante os testes[1] desse protótipo, o único ponto de sustentação dos seguimentos l_1 e l_2 mostrou-se insuficiente para sustentar o peso do robô, evidenciando a fragilidade da estrutura. Por esse motivo, foi feito o modelo 3D da segunda

[1]Vídeos de testes do robô: https://www.youtube.com/playlist?list=PLtiqwHhrwyDy4FeO3HXZ96Ta0Fswptasy



(A) (B)

Figura 7. (A) Primeiro protótipo do robô quadrúpede; (B) Segundo protótipo do robô quadrúpede.

versão, levando em consideração essa fraqueza estrutural. O processo de montagem descrito anteriormente foi repetido, resultando no protótipo disposto na Figura 7 (B)[2].

Apesar de alguns problemas ocasionados pela escolha de componentes e da geometria da estrutura, os testes com esse protótipo foram promissores, permitindo, dessa forma, o avanço do projeto para a fase de medição dos resultados.

V. Resultados

Nesta seção serão discutidos os resultados obtidos através dos protótipos construídos nas seções anteriores deste trabalho. Tais resultados incluem: Aplicação IoT para controle do robô e gráficos de velocidade e eficiência de bateria por tipo de terreno.

A. Aplicação IoT

Além dos códigos de implementação da cinemática inversa, o Raspberry foi programado como servidor web, que cria uma rede Wi-Fi, permitindo a conexão *wireless* de outros aparelhos. Esses podem, então, enviar requisições HTTP ao endereço IP do servidor, programado para recebê-las e realizar ações correspondentes ao comando desejado.

Para permitir o uso intuitivo pelo usuário, foi construída uma interface utilizando HTML e CSS, exibida na Figura 8, que pode ser acessada através do endereço IP do Raspberry após conectar-se com a rede Wi-Fi do servidor.

Figura 8. Interface de usuário para o software de controle do robô.

[2]Repositório com os modelos 3D e códigos: https://github.com/GabrielAF-Faca/TFG_II

A interface minimalista apresenta diversos elementos: um botão verde no canto superior esquerdo, dois conjuntos de quatro botões com símbolos nos cantos inferiores e um botão com título "Iniciar" acompanhado de um campo de entrada de texto, abaixo do medidor de bateria, ao centro da tela. Cada botão é programado, utilizando JavaScript, para enviar a requisição HTTP correspondente à sua funcionalidade ao servidor.

O botão verde é responsável por ligar e desligar o robô. O conjunto de quatro botões à esquerda da interface controla a movimentação lateral do robô, cada um representando o sentido de movimento indicado pelas setas direcionais: frente, trás, esquerda, direita.

Já o conjunto de botões à direita é responsável tanto pela rotação do robô em torno do próprio eixo nos sentidos horário e anti-horário (botões esquerdo e direito), quanto pela elevação do corpo do robô (botões superior e inferior).

O botão intitulado "Iniciar" tem a função de iniciar a medição de tempo (em milissegundos) e armazenar a bateria inicial do trajeto, bem como o nome do trajeto especificado através do campo de texto acima. Após clicado, o título do botão é alterado para "Parar" e o texto do trajeto é removido. Ao clicar novamente no botão, o servidor registra o tempo e a bateria final do trajeto e a distância (em centímetros) especificada, também, pelo campo de texto. Todos esses dados são adicionados a um arquivo, seguindo a estrutura: *Nome do trajeto, Distância (cm), Tempo (ms), Bateria Inicial (%), Bateria Final (%).*

Por fim, é possível observar um medidor de bateria formado por dez círculos coloridos no centro da interface. Os círculos verdes representam a porcentagem da quantidade restante de bateria, aproximadamente 70% no caso da Figura 8, onde há 7 círculos verdes e 3 vermelhos. A medida da bateria é calculada a partir da leitura do GPIO 28, onde o divisor de tensão está conectado.

Os dados obtidos através dos testes em diversos tipos de terrenos foram, posteriormente, utilizados para medir a velocidade média, bem como a eficiência de bateria em cada tipo de solo testado. Essas informações são exibidas na próxima subseção, permitindo uma análise visual do desempenho do robô.

B. Desempenho do Robô

A partir dos dados coletados pelos testes de movimentação do robô em cada tipo de terreno é possível realizar a análise de desempenho deste em cada situação. A Figura 9 mostra cada tipo de superfície onde a movimentação do robô foi testada: (A) Porcelanato Áspero; (B) Lajota Rugosa; (C) Grama Plano; (D) Lajota Pedregulho; (E) Lajota Lisa; (F) Concreto Inclinado (11°); (G) Concreto Plano; (H) Pedregulho Solto.

Já a Figura 10 mostra um gráfico da distância (cm) percorrida pelo tempo (s) de trajeto. Cada reta no gráfico representa

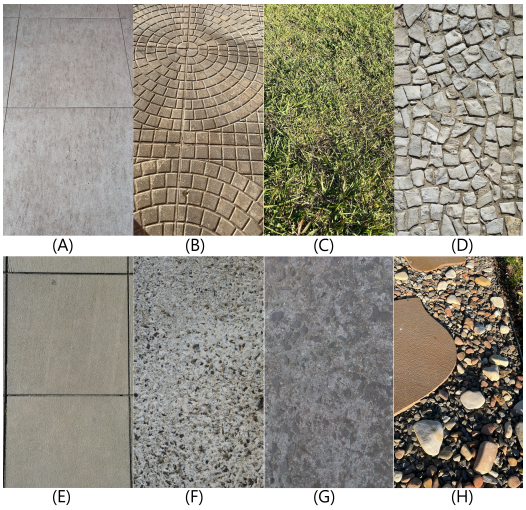


Figura 9. Terrenos testados com o robô.

a velocidade média, em centímetros por segundo (cm/s), de deslocamento do robô em cada terreno atravessado.

Neste teste, é possível observar a grande redução de velocidade nas superfícies não planas (F) ou muito irregulares (D), devido à maior dificuldade de travessia do robô. O oposto ocorre nos terrenos planos com poucas irregularidades ((B), (E) e (G)), onde há um grande aumento de desempenho.

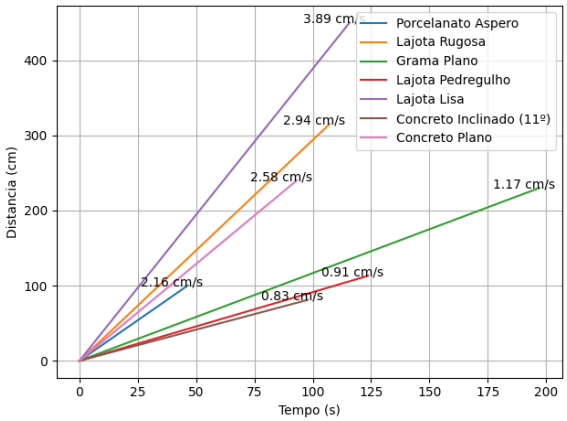


Figura 10. Gráfico de distância percorrida por tempo em cada terreno.

Por fim, a Figura 11 mostra um gráfico do consumo de bateria (%) a cada 10 segundos em cada tipo de superfície percorrida. É notável em ambos os gráficos o desempenho reduzido das superfícies (C), (D) e (F), causado, principalmente, pela inclinação (em (F)) e geometria da extremidade da perna do robô que, graças a seu formato estreito, prendia-se nas irregularidades do terreno, aumentando o tempo de trajeto e, por consequência, diminuindo a velocidade. A superfície (B), no entanto, foi a única que apresentou alta velocidade média e alto consumo de bateria simultaneamente, possuindo segunda colocação em ambos os quesitos.

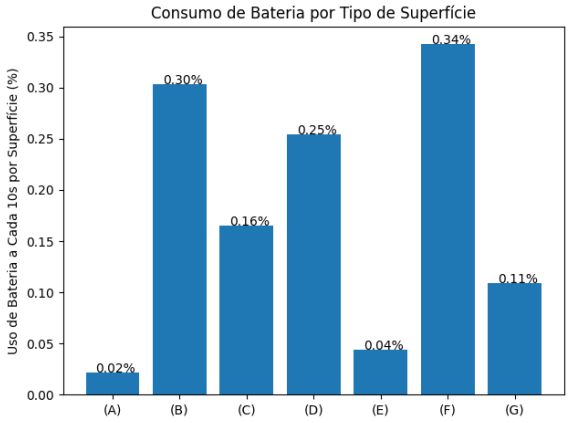


Figura 11. Gráfico de consumo de bateria a cada 10 segundos em cada terreno.

Vale ressaltar, também, que a superfície (H) não está colocada em nenhum dos gráficos. Isso se dá pois o robô não foi capaz de locomover-se em superfícies com grandes quantidades de elementos móveis independentes, como britas, também devido à geometria da perna. Além disso, os movimentos realizados em tais superfícies ocasionavam grandes estresses mecânicos nas engrenagens dos atuadores, causando, em alguns casos, a quebra deste componente.

Apesar disso, o desempenho nas superfícies (A), (B), (E), (F) e (G) mostrou-se surpreendentemente satisfatório, alcançando velocidade e autonomia de bateria relativamente altas, exceto nas superfícies (B) e (F). A superfície (F), apesar de seu baixo desempenho comparada às outras, mostrou a capacidade do robô de locomoção em terrenos inclinados e, por consequência, pôde ser considerada como resultado bem sucedido.

VI. Conclusão

Neste trabalho foram estudados e implementados os cálculos de cinemática inversa a fim de permitir a locomoção de um robô quadrúpede com três graus de liberdade (DOF) por perna, controlado através de uma aplicação IoT *wireless*. Começando com uma revisão detalhada dos métodos de locomoção robótica, destacando a escolha das pernas como o dispositivo efetuador ideal para terrenos irregulares. Foi utilizada a metodologia ágil Kanban, juntamente a ciclos de prototipagem, para realizar a construção de um protótipo funcional buscando aplicar os conceitos estudados e medir o desempenho do método de locomoção escolhido, o que possibilitou a execução dos testes e coleta dos resultados.

Os resultados obtidos ao longo deste trabalho demonstraram-se satisfatórios, atendendo os objetivos gerais e específicos propostos. O robô quadrúpede desenvolvido foi capaz de realizar locomoção em diversos tipos de terreno, utilizando os cálculos de cinemática inversa para

controle de suas juntas e sendo controlado remotamente por meio de uma aplicação IoT. Esses resultados validaram tanto o projeto mecânico quanto a solução implementada para controle e navegação.

No entanto, alguns desafios foram identificados, como a fragilidade das engrenagens dos servomotores em terrenos mais exigentes e a dificuldade de locomoção em superfícies com elementos móveis. Dessa forma, para trabalhos futuros, recomenda-se a utilização de servomotores de maior qualidade, com engrenagens inteiramente metálicas, para aumentar a robustez e a durabilidade do sistema, possibilitando um desempenho ainda melhor em terrenos desafiadores.

Essa melhoria não só amplia as capacidades do robô, mas também garantia maior confiabilidade e eficiência em suas aplicações práticas, permitindo explorar novos cenários e desafios na área de robótica móvel.

Referências

[1] S. L. Dantas. *História da robótica na educação*. Contentus, 2020.

[2] M. A. Elsheikh. "Design of a special rigid wheel for traversing loose soil". Em: *Scientific Reports* 13 (2023). Disponível em <https://doi.org/10.1038/s41598-022-27312-6>.

[3] R. Biradar A. Olivier e G. Devanagavi. "Conceptual design of a quadruped wheel-legged robot for an autonomous terrestrial locomotion". Em: (2022). Disponível em https://doi.org/10.1109/ICAECC54045.2022.9716642.

[4] R. Rathnam e W. Godfrey. "Data Driven Approach for Inverse Kinematics in 2D and 3D". Em: (2023). Disponível em <https://doi.org/10.1109/CICT59886.2023.10455509>.

[5] M. Zoula P. Čížek e J. Faigl. "Design, Construction, and Rough-Terrain Locomotion Control of Novel Hexapod Walking Robot With Four Degrees of Freedom Per Leg". Em: (2021). Disponível em https://doi.org/10.1109/ACCESS.2021.3053492.

[6] F. Cordes T. M. Roehr e F. Kirchner. "RIMRES: A Modular Reconfigurable Heterogeneous Multi-Robot Exploration System". Em: (2012). Disponível em https://robotics.estec.esa.int/i-SAIRAS/isairas2012/Papers/Session%202A/02A_02_Cordes.pdf.

[7] E. Garcia D. Sanz-Merodio e P. Gonzalez-de-Santos. "Analyzing energy-efficient configurations in hexapod robots for demining applications". Em: (2012). Disponível em http://dx.doi.org/10.1108/01439911211227926.

[8] T. Z. G. Dias. *Cinesiologia, biomecânica e robótica*. Contentus, 2021.

[9] M. Mataric. *Introdução à robótica*. Blucher, 2014.

[10] M. G. Bekker. "Mechanics of Locomotion and Lunar Surface Vehicle Concepts". Em: *SAE Transactions* 72 (1963). Disponível em https://www.jstor.org/stable/44562978, pp. 549–569.

[11] Boston Dynamics. *Spot® - The Agile Mobile Robot*. 2020. URL: https://bostondynamics.com/products/spot/ (acesso em 03/04/2024).

[12] J. J. Craig. *Robótica*. Pearson, 2013.

[13] A. U. Rao e A. Himanshu. "A Review of Methods Used for Kinematic Modeling of A Manipulator". Em: Disponível em <https://www.researchgate.net/publication/351357733_A_Review_of_Methods_Used_for_Kinematic_Modeling_of_A_Manipulator>. 2017.

[14] T. D. Ngo et al. "Additive manufacturing (3D printing): A review of materials, methods, applications and challenges". Em: *Composites Part B: Engineering* 143 (2018). Disponível em <https://doi.org/10.1016/j.compositesb.2018.02.012>, pp. 172–196. ISSN: 1359-8368.

[15] Raspberry Pi. *We are Raspberry Pi. We make computers.* URL: https://www.raspberrypi.com/about/ (acesso em 25/04/2024).

[16] Raspberry Pi. *Raspberry Pi Pico and Pico W*. URL: https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html (acesso em 25/04/2024).

[17] Python. *About*. URL: https://www.python.org/about/ (acesso em 26/04/2024).

[18] A. Ernest, M. Ezekiel e G. Abilimi. "Qualitative Assessment of Compiled, Interpreted and Hybrid Programming Languages". Em: *Communications on Applied Electronics* 7 (2017). Disponível em <http://dx.doi.org/10.5120/cae2017652685>, pp. 8–13.

[19] George Robotics Limited. *MicroPython*. URL: https://micropython.org/ (acesso em 26/04/2024).

[20] A. M. Santos. *Arduino: do básico à internet das coisas*. Brasport, 2023.

[21] D. Kusmenko e K. Schmidt. "Development of an analytical inverse kinematics for a 5 DOF manipulator". Em: (2020). Disponível em <https://doi.org/10.1109/REM49740.2020.9313880>, pp. 1–5.

[22] R. S. Pressman e B. R. Maxim. *Engenharia de Software: UMA ABORDAGEM PROFISSIONAL*. AMGH, 2021.

[23] D. J. Anderson e A. Carmichael. *Essential Kanban Condensed*. Kanban University Press, 2016.

[24] I. Sommerville. *Engenharia de software*. Pearson, 2018.

[25] S. Monk. *Hacking Electronics: Learning Electronics with Arduino and Raspberry Pi*. 2nd Ed. McGraw Hill Education, 2017.