UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

GUILHERME BARROS VIEIRA DE ARAUJO

PROJETO E IMPLEMENTAÇÃO DE UM VEÍCULO ROBÓTICO DE BAIXO CUSTO

QUIXADÁ

2025

GUILHERME BARROS VIEIRA DE ARAUJO

PROJETO E IMPLEMENTAÇÃO DE UM VEÍCULO ROBÓTICO DE BAIXO CUSTO

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Orientador: Prof. Dr. Antônio Joel Ramiro de Castro.

QUIXADÁ

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

A689p Araujo, Guilherme Barros Vieira de.
Projeto e Implementação de um veículo robótico de baixo custo / Guilherme Barros Vieira de Araujo. – 2025.
51 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Computação, Quixadá, 2025.
Orientação: Prof. Dr. Antônio Joel Ramiro de Castro.

1. Robótica móvel. 2. ESP32. 3. Internet das coisas. 4. Navegação autônoma. 5. Baixo custo. I. Título.
CDD 621.39

GUILHERME BARROS VIEIRA DE ARAUJO

PROJETO E IMPLEMENTAÇÃO DE UM VEÍCULO ROBÓTICO DE BAIXO CUSTO

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Computação.

Aprovada em: 16/12/2025.

BANCA EXAMINADORA

Prof. Dr. Antônio Joel Ramiro de Castro (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Thiago Werlley Bandeira da Silva
Universidade Federal do Ceará (UFC)

Profa. Dra. Maria Luiza Miguez
Instituto Federal do Norte de Minas Gerais (IFNMG)

Dedico este trabalho a todos que não soltaram a minha mão quando o peso foi demais e transformaram sua fé em mim no que me sustentou quando a minha acabou.

RESUMO

A robótica móvel e a Internet das Coisas (IoT) impulsionam inovações tecnológicas significativas, porém o elevado custo de aquisição de kits de desenvolvimento comerciais limita a democratização do acesso a essas ferramentas. Neste contexto, este trabalho apresenta o projeto e a implementação de uma plataforma robótica móvel de tração diferencial (4WD) de baixo custo, fundamentada na utilização do microcontrolador ESP32. A metodologia adotada percorreu o ciclo completo de engenharia, abrangendo desde a modelagem esquemática do hardware e seleção de componentes, até a manufatura e integração física dos sistemas mecânicos e eletrônicos. O desenvolvimento do firmware, realizado em linguagem C++, integrou um servidor web embarcado para teleoperação via rede Wi-Fi e algoritmos de leitura de sensores ultrassônicos para a execução de navegação autônoma reativa. Os resultados obtidos nos ensaios práticos validaram a robustez da arquitetura proposta, demonstrando baixa latência no controle remoto e eficácia na lógica de detecção e desvio de obstáculos. A análise econômica revelou um custo final do protótipo inferior a R$ 330,00, representando uma economia superior a 90% em comparação a soluções proprietárias equivalentes. Conclui-se que a plataforma desenvolvida atende aos requisitos de desempenho e acessibilidade, consolidando-se como uma base modular e aberta viável para aplicações em robótica e automação.

Palavras-chave: robótica móvel; ESP32; baixo custo; internet das coisas; navegação autônoma.

ABSTRACT

Mobile robotics and the Internet of Things (IoT) drive significant technological innovations; however, the high acquisition cost of commercial development kits limits the democratization of access to these tools. In this context, this work presents the design and implementation of a low-cost differential drive (4WD) mobile robotic platform, based on the ESP32 microcontroller. The adopted methodology followed the complete engineering cycle, ranging from hardware schematic modeling and component selection to the manufacturing and physical integration of mechanical and electronic systems. The firmware development, carried out in C++ language, integrated an embedded web server for teleoperation via Wi-Fi network and algorithms for ultrasonic sensor reading to execute reactive autonomous navigation. The results obtained in practical trials validated the robustness of the proposed architecture, demonstrating low latency in remote control and effectiveness in obstacle detection and avoidance logic. The economic analysis revealed a final prototype cost lower than R$ 330.00, representing savings of over 90% compared to equivalent proprietary solutions. It is concluded that the developed platform meets performance and accessibility requirements, consolidating itself as a viable modular and open base for robotics and automation applications.

Keywords: mobile robotics; ESP32; low cost; internet of things; autonomous navigation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

4WD	*Four-Wheel Drive*
BMS	*Battery Management System*
EDA	*Electronic Design Automation*
GPIO	*General-Purpose Input/Output*
HTTP	*Hypertext Transfer Protocol*
IoRT	*Internet of Robotic Things*
IoT	*Internet of Things*
MQTT	*Message Queuing Telemetry Transport*
OBR	Olimpíada Brasileira de Robótica
PID	Proporcional-Integral-Derivativo
PWM	*Pulse Width Modulation*
SoC	*System on Chip*
ToF	*Time of Flight*
VCC	*Voltage Common Collector*

LISTA DE SÍMBOLOS

V	Volts
V_{IN}	Tensão de Entrada
V_{med}	Tensão Média
V_{max}	Tensão Máxima
GND	Ground (Terra)
Hz	Hertz
kHz	Quilohertz
MHz	Megahertz
cm	Centímetro
m/s	Metros por segundo
μs	Microssegundos
A	Ampere
v	Velocidade Linear
ω	Velocidade Angular
v_d	Velocidade da Roda Direita
v_e	Velocidade da Roda Esquerda
L	Distância entre as Rodas (Bitola)
d	Distância calculada pelo sensor
v_{som}	Velocidade do Som no ar
t	Tempo
t_{on}	Tempo de sinal em nível alto
T	Período
D	Ciclo de Trabalho (*Duty Cycle*)

SUMÁRIO

1 INTRODUÇÃO

Atualmente, estamos vivendo um momento em que a robótica móvel *Internet of Things* (IoT) estão se misturando cada vez mais. Essa união está mudando completamente a forma como as máquinas funcionam e interagem com o mundo ao redor. Não se trata mais apenas de dispositivos que conseguem se mover; agora, espera-se que esses dispositivos consigam interagir com o ambiente e com as pessoas que o operam, tornando essa interação muito mais dinâmica e útil para o dia a dia.

Tudo isso acontece dentro do cenário da chamada Indústria 4.0. Nesse contexto, criar sistemas que vão além do simples movimento mecânico virou uma necessidade. Hoje, é fundamental que os dispositivos consigam ler dados através de sensores e transmitir essas informações pela rede. Isso tem aplicações práticas enormes, indo desde a logística, onde robôs transportam cargas sozinhos, até a agricultura de precisão e sistemas de vigilância que podem ser monitorados à distância.

Um fator que ajudou muito nessa evolução foi o avanço rápido da microeletrônica. O que antes exigia computadores caros e complexos, hoje pode ser feito com microcontroladores acessíveis, que já possuem um alto poder de processamento e, o mais importante, conectividade sem fio nativa. Isso tornou a tecnologia muito mais democrática, permitindo que componentes poderosos cheguem às mãos de qualquer pessoa que deseje desenvolver novas soluções.

Graças a essas plataformas modernas de desenvolvimento, estudantes e engenheiros conseguem hoje projetar veículos robóticos complexos sem precisar de orçamentos milionários. É possível criar robôs que tomam decisões de navegação sozinhos ou que são controlados direto pelo navegador do celular, sem depender de equipamentos de comunicação proprietários. Isso cria um ambiente perfeito para a pesquisa e incentiva a criação de tecnologias com arquitetura aberta, servindo como base para uma inovação que é, ao mesmo tempo, tecnológica e acessível.

1.1 Motivação

Apesar do crescimento acelerado da robótica, o acesso a plataformas de desenvolvimento completas ainda enfrenta barreiras significativas, especialmente no contexto educacional e de pesquisa em países em desenvolvimento. Kits comerciais de robótica e sistemas de desenvolvimento proprietários (como LEGO Mindstorms, VEX Robotics ou plataformas industriais) oferecem robustez e facilidade de uso, porém, seus custos elevados inviabilizam a aquisição em

larga escala por instituições de ensino públicas e estudantes independentes.

Por outro lado, soluções de baixo custo baseadas em microcontroladores de gerações anteriores (como a família ATmega do Arduino Uno) muitas vezes carecem de recursos nativos essenciais para aplicações modernas, exigindo a integração de múltiplos módulos externos para viabilizar a comunicação Wi-Fi ou Bluetooth, o que eleva a complexidade de montagem e o consumo energético, além de limitar a capacidade de processamento para algoritmos de navegação mais sofisticados.

A motivação deste trabalho reside, portanto, na lacuna existente entre os sistemas profissionais de alto custo e as soluções didáticas obsoletas. Surge a necessidade de validar uma arquitetura intermediária, baseada no microcontrolador ESP32, que combine baixo custo, arquitetura aberta e alto desempenho (processamento *dual-core* e IoT nativa), provando que é possível desenvolver um veículo robótico funcional e robusto capaz de operar nos modos autônomo e remoto, democratizando o acesso a tecnologias de ponta.

1.2 Objetivo Geral

O objetivo geral deste trabalho é projetar, implementar e validar uma plataforma robótica móvel de tração diferencial (*Four-Wheel Drive* (4WD)) de baixo custo, fundamentada no microcontrolador ESP32, capaz de realizar navegação autônoma com desvio de obstáculos e teleoperação via rede Wi-Fi.

1.3 Objetivos Específicos

Para alcançar o objetivo geral proposto, foram definidos os seguintes objetivos específicos:

- Realizar o levantamento bibliográfico sobre arquiteturas de robôs móveis, sensores ultrassônicos e o ecossistema do microcontrolador ESP32;
- Projetar e modelar o esquemático elétrico do sistema, integrando a unidade de controle, drivers de potência, sensores e gerenciamento de energia;
- Construir e integrar fisicamente o protótipo utilizando um chassi 4WD, realizando a montagem mecânica e o cabeamento dos componentes eletrônicos;
- Desenvolver o *firmware* de controle em linguagem C++, implementando algoritmos de leitura de sensores para navegação reativa e lógica de controle de motores via PWM;

– Implementar um servidor *web* embarcado no ESP32 para viabilizar a interface de controle remoto via navegador (IoT);
– Validar o funcionamento do protótipo através de testes práticos de conectividade, autonomia e resposta a obstáculos, além de realizar a análise econômica comparativa com soluções comerciais.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo reúne os principais conceitos teóricos utilizados para o desenvolvimento do projeto. Inicialmente, são abordados os fundamentos da Robótica Móvel, focando nos tipos de locomoção e na mecânica de robôs com tração diferencial. Na sequência, discute-se a Internet das Coisas (IoT) e sua integração com sistemas robóticos, explicando a comunicação necessária para o controle remoto via Wi-Fi. O texto também detalha as características do microcontrolador ESP32, justificando sua escolha como unidade de processamento, e finaliza explorando o funcionamento técnico dos sensores ultrassônicos e dos motores, peças essenciais para a autonomia e movimentação do veículo.

2.1 Robótica Móvel

A robótica móvel é a subárea responsável pelo estudo de sistemas capazes de se deslocar fisicamente em um ambiente, diferindo dos manipuladores industriais que operam fixos em uma base. Essa capacidade de locomoção exige que o robô possua um sistema de percepção e controle capaz de interpretar o ambiente e tomar decisões de navegação. Quanto ao nível de autonomia, estes sistemas podem ser classificados desde teleoperados, onde um humano envia comandos diretos, até totalmente autônomos, onde o robô decide sua trajetória com base em sensores. O projeto desenvolvido neste trabalho situa-se em uma categoria híbrida, implementando tanto o controle remoto via rede quanto a autonomia reativa. (Siegwart *et al.*, 2011)

2.1.1 Locomoção e Cinemática Diferencial

A locomoção sobre rodas é a mais eficiente para ambientes planos e estruturados. Dentre as configurações possíveis, a tração diferencial é amplamente adotada em robótica educacional devido à simplicidade mecânica e de controle.

Neste projeto, embora o chassi possua quatro rodas (4WD), a lógica de movimentação é a de *Skid-Steering* (direção por deslizamento). Neste modelo, as rodas do lado esquerdo giram em sincronia, assim como as do lado direito, comportando-se cinematicamente como um robô diferencial de duas rodas virtuais (Dudek; Jenkin, 2024). A velocidade linear v do robô é dada

pela média das velocidades das rodas direita (v_d) e esquerda (v_e), conforme descrito em

$$v = \frac{v_d + v_e}{2}. \tag{2.1}$$

Já a velocidade angular ω, que determina a taxa de giro do robô em torno do seu centro instantâneo de rotação, depende da diferença entre as velocidades das rodas e da distância L entre elas (bitola), dada por

$$\omega = \frac{v_d - v_e}{L}. \tag{2.2}$$

Analisando a Equação 2.2, nota-se que para realizar curvas, basta impor uma diferença de velocidade entre os lados. Se v_d for igual a v_e, a velocidade angular é zero e o robô segue em linha reta. Para rotacionar no próprio eixo, aplicam-se velocidades iguais em módulo, mas com sentidos opostos ($v_d = -v_e$), o que facilita a navegação em espaços confinados (Hamici, 2023).

2.2 Internet das Coisas (IoT) e IoRT

A Internet das Coisas, ou IoT (*Internet of Things*), representa uma mudança de paradigma na computação, onde a conectividade deixa de ser exclusiva de computadores e smartphones para permear objetos do cotidiano. Essencialmente, a IoT descreve uma rede de dispositivos físicos embarcados com sensores, *software* e tecnologias de comunicação que lhes permitem coletar e trocar dados pela internet. No contexto da automação e controle, essa tecnologia permite que dispositivos antes isolados passem a ser monitorados e controlados remotamente, gerando sistemas mais eficientes e responsivos. (Atzori *et al.*, 2010)

2.2.1 Internet of Robotic Things (IoRT)

A união entre a robótica e a IoT deu origem a um novo conceito denominado Internet das Coisas Robóticas (*Internet of Robotic Things* (IoRT)). Enquanto os dispositivos de IoT tradicionais são geralmente passivos, focados apenas no monitoramento e coleta de dados, a IoRT introduz agentes ativos que são os robôs (Ray, 2017).

Nesta arquitetura, o robô não é apenas um sensor móvel que envia dados para a nuvem; ele é um agente capaz de interagir fisicamente com o ambiente baseando-se nas informações que recebe ou processa. O robô desenvolvido neste trabalho exemplifica esse conceito: ele atua como um nó da rede (IoT) que coleta dados de distância (sensor ultrassônico),

mas possui a capacidade mecânica de alterar seu estado e posição no mundo físico (atuadores), fundindo as capacidades de percepção da IoT com as capacidades de ação da robótica.

2.2.2 Arquitetura Cliente-Servidor e Protocolo HTTP

Para viabilizar o controle remoto do robô sem a necessidade de cabos ou controles proprietários, utiliza-se a arquitetura Cliente-Servidor baseada nos protocolos da *World Wide Web*. Neste modelo, o microcontrolador ESP32 é configurado para operar como um Servidor *Web* (*Web Server*), hospedando em sua memória os arquivos da interface gráfica (HTML, CSS e JavaScript) (Kurose; Ross, 2019).

O dispositivo do usuário (smartphone ou computador) atua como o Cliente. A comunicação ocorre através do protocolo*Hypertext Transfer Protocol* (HTTP), seguindo um ciclo de requisição e resposta. Quando o operador aciona um comando na interface, o navegador envia uma requisição (geralmente do tipo GET ou POST) para o endereço IP do robô. O servidor embarcado no ESP32 interpreta essa requisição e executa a ação correspondente nos motores, retornando uma confirmação para a interface. Essa abordagem elimina a necessidade de instalação de aplicativos específicos, garantindo interoperabilidade com qualquer dispositivo que possua um navegador moderno e conexão Wi-Fi (Kurniawan, 2019).

2.3 Sistemas Embarcados e Microcontroladores

Um sistema embarcado é um sistema computacional projetado para realizar uma função dedicada ou um conjunto específico de tarefas, diferindo dos computadores de propósito geral (como PCs e laptops) que são projetados para executar uma vasta gama de aplicações. Geralmente, esses sistemas estão integrados ao hardware que controlam, possuindo restrições de energia, tamanho e custo. O componente central de um sistema embarcado é, na maioria das vezes, um microcontrolador.

Diferente de um microprocessador, que foca apenas no processamento de dados e requer componentes externos (memória RAM, armazenamento e interfaces) para funcionar, o microcontrolador integra todos esses elementos em um único circuito integrado. Essa característica de "computador em um chip"simplifica drasticamente o design de circuitos eletrônicos, reduzindo o custo e a complexidade da montagem, o que os torna ideais para aplicações em robótica e automação (Barr, 1999).

2.3.1 A Plataforma ESP32

Para atender aos requisitos de conectividade e processamento deste projeto, optou-se pela utilização do ESP32, desenvolvido pela Espressif Systems. O ESP32 não é apenas um microcontrolador simples, mas um *System on Chip* (SoC) de baixo custo e baixo consumo de energia que integra, nativamente, tecnologias de comunicação Wi-Fi e Bluetooth.

A arquitetura do ESP32 baseia-se em um processador Xtensa® LX6 de 32 bits, que pode operar em modo *dual-core* com frequências de até 240 MHz. Essa capacidade de processamento representa um salto significativo em relação a plataformas educacionais tradicionais, como o Arduino Uno (baseado no ATmega328P de 8 bits e 16 MHz). Enquanto microcontroladores de 8 bits podem sofrer para gerenciar comunicação de rede e controle de motores simultaneamente, o ESP32 possui desempenho suficiente para executar sistemas operacionais de tempo real e gerenciar múltiplas tarefas concorrentes (ESPRESSIF SYSTEMS, 2024).

Além do processamento, o ESP32 destaca-se pela riqueza de periféricos integrados. Ele oferece conversores analógico-digitais (ADC) de alta resolução, conversores digital-analógicos (DAC) e, crucialmente para este trabalho, controladores de PWM (Modulação por Largura de Pulso) via *hardware*. A presença do transceptor Wi-Fi integrado elimina a necessidade de módulos de comunicação externos (como o ESP8266 ou *shields* Ethernet), simplificando a arquitetura do robô e reduzindo o custo final do protótipo (Hamici, 2023).

2.4 Sensoriamento e Percepção

Para que um robô móvel possa operar de forma autônoma ou semi-autônoma, ele necessita de um sistema de percepção que lhe permita extrair informações sobre o estado do ambiente ao seu redor. Em robótica, o sensoriamento é o processo de transdução, ou seja, a conversão de grandezas físicas (como luz, som, temperatura ou pressão) em sinais elétricos que podem ser processados digitalmente pelo microcontrolador.

A percepção robótica, portanto, não se resume apenas à leitura de dados brutos, mas à interpretação desses sinais para a tomada de decisão. No contexto da navegação autônoma reativa, a capacidade mais crítica é a detecção de proximidade, que permite ao robô identificar obstáculos, estimar a distância até eles e evitar colisões, garantindo a integridade física do sistema e do ambiente (Siegwart *et al.*, 2011).

2.4.1 Princípio de Funcionamento do Ultrassom

Dentre as diversas tecnologias de medição de distância disponíveis (como infravermelho, LiDAR ou visão computacional), os sensores ultrassônicos destacam-se em aplicações de baixo custo pela sua robustez e simplicidade. O princípio físico baseia-se na emissão e recepção de ondas sonoras de alta frequência, inaudíveis ao ouvido humano (geralmente acima de 20 kHz).

O funcionamento desses sensores utiliza o método de "Tempo de Voo"(*Time of Flight* (ToF)). O sensor emite um curto pulso sonoro (chamado de *burst*) que viaja pelo ar, reflete no objeto mais próximo e retorna ao sensor como um eco. Como a velocidade do som no ar é conhecida e relativamente constante (aproximadamente 343 m/s ao nível do mar e a 20°C), é possível calcular a distância do objeto medindo-se o tempo total decorrido entre a emissão e a recepção do sinal (ElecFreaks, 2011).

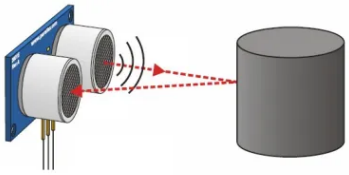
Para o módulo HC-SR04, amplamente utilizado em robótica educacional, o processo de medição inicia-se quando o microcontrolador envia um pulso de nível lógico alto de pelo menos 10 μs para o pino de disparo (*Trigger*). Em resposta, o sensor emite uma série de 8 pulsos ultrassônicos a 40 kHz e aguarda o retorno. Assim que o eco é detectado, o sensor envia um sinal digital para o pino de retorno (*Echo*), cuja duração é proporcional ao tempo de viagem da onda sonora (Venkatesh *et al.*, 2021).

A distância d até o obstáculo é calculada pela metade do produto entre a velocidade do som (v_{som}) e o tempo de duração do pulso de eco (t), conforme descrito na Equação 2.3. A divisão por 2 é necessária porque o tempo medido corresponde ao trajeto de ida e volta da onda.

$$d = \frac{v_{som} \cdot t}{2} \tag{2.3}$$

A Figura 1 ilustra esquematicamente esse processo de emissão e reflexão das ondas sonoras ao encontrar um obstáculo.

Figura 1 – Princípio de funcionamento do sensor ultrassônico HC-SR04



Fonte: https://portal.vidadesilicio.com.br/hc-sr04-sensor-ultrassonico/

Apesar de eficazes, sensores ultrassônicos possuem limitações físicas, como o ângulo de abertura do feixe sonoro e a dificuldade de detectar superfícies que absorvem som (como tecidos) ou que estão inclinadas de forma a refletir a onda para longe do sensor. No entanto, para a detecção de paredes e obstáculos sólidos em ambientes internos, essa tecnologia oferece a precisão necessária para algoritmos de desvio de obstáculos.

2.5 Atuação e Eletrônica de Potência

Enquanto os sensores são responsáveis pela percepção do ambiente, o sistema de atuação é o que permite ao robô interagir fisicamente com o mundo, convertendo energia elétrica em energia mecânica. No entanto, a integração entre a unidade de processamento (ESP32) e os atuadores não é direta. Microcontroladores operam com níveis de tensão e corrente muito baixos (sinais lógicos de 3.3V e correntes na ordem de miliamperes), insuficientes para acionar cargas indutivas que exigem maior potência. Para solucionar essa discrepância, utiliza-se a eletrônica de potência, que atua como uma interface intermediária capaz de modular a entrega de energia da bateria para os motores com base nos comandos lógicos de baixa potência.

2.5.1 Motores de Corrente Contínua (DC)

Os motores de corrente contínua (DC) são os atuadores mais comuns em robótica móvel devido à sua simplicidade de controle e baixo custo. O princípio de funcionamento baseia-se na interação entre o campo magnético gerado por ímãs permanentes no estator e a corrente elétrica que percorre as bobinas do rotor, gerando uma força eletromagnética (Força de Lorentz) que produz o torque rotacional (Hughes; Drury, 2019).

Contudo, motores DC convencionais tendem a ter alta velocidade de rotação e baixo torque, o que é inadequado para a tração direta de veículos. Para adequar essas características à necessidade do robô, acopla-se ao motor um sistema de engrenagens redutoras (caixa de redução). A redução mecânica diminui a velocidade angular final no eixo da roda, mas multiplica proporcionalmente o torque disponível, permitindo que o robô vença a inércia e o atrito com o solo.

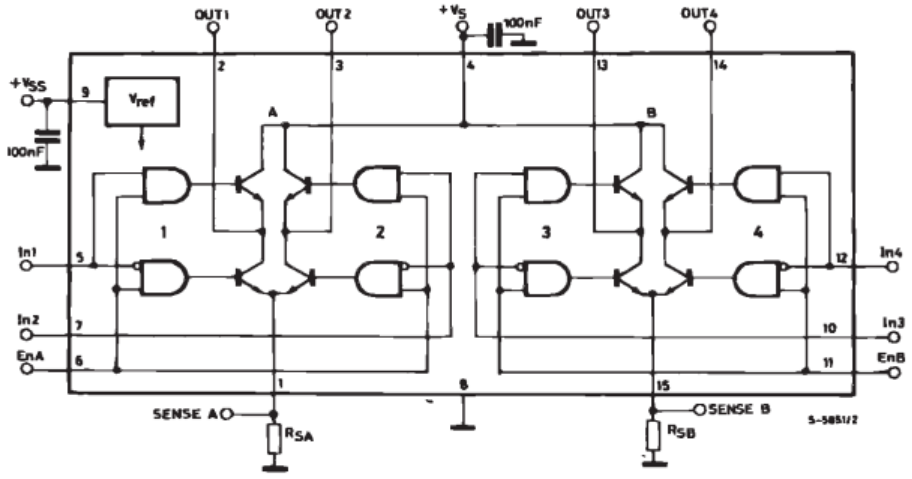
2.5.2 Ponte H e Driver L298N

Para controlar um motor DC, não basta apenas ligá-lo e desligá-lo; é necessário controlar o sentido da rotação para permitir que o robô ande para frente ou para trás. A topologia de circuito utilizada para essa finalidade é a Ponte H.

A Ponte H consiste em um arranjo de quatro chaves eletrônicas (geralmente transistores) dispostas em forma de "H", com o motor no centro. Acionando-se as chaves em pares diagonais opostos, inverte-se a polaridade da tensão aplicada aos terminais do motor, invertendo assim o sentido da corrente e, consequentemente, a rotação (STMicroelectronics, 2000).

Neste projeto, a implementação física dessa topologia é feita pelo circuito integrado L298N. Este *driver* é um duplo Ponte H, capaz de controlar dois canais independentes (lado esquerdo e direito do robô) com correntes de até 2A por canal, isolando galvanicamente a lógica de controle do ESP32 da tensão de alimentação dos motores (7.4V).

Figura 2 – Diagrama de blocos interno do driver L298N mostrando a lógica de controle



Fonte: (STMicroelectronics, 2000)

Analisando o diagrama da Figura 2, observa-se que o controle dos transistores de potência (Q1 a Q4) é intermediado por portas lógicas do tipo *AND* (portas E). O funcionamento ocorre da seguinte forma:

– **Habilitação (Enable/EnA):** Este pino atua como uma chave geral para a ponte. Quando está em nível lógico baixo (LOW), as saídas das portas AND são forçadas a zero, desligando todos os transistores e deixando o motor "solto"(inércia). Quando está em nível alto

(HIGH), permite o acionamento.

- **Entradas (In1 e In2):** Estes pinos definem a direção. Se In1 for Alto e In2 for Baixo, a corrente flui em um sentido (ex: horário). Se invertido, o sentido da corrente inverte. Se ambos forem iguais, ocorre o freio eletrônico.
- **Alimentação Separada:** O diagrama evidencia dois pinos de alimentação distintos: V_{SS} para a lógica interna (5V) e V_S para a potência dos motores (até 46V), garantindo o isolamento necessário entre o microcontrolador e a carga indutiva.

2.5.3 Modulação por Largura de Pulso (PWM)

O controle da velocidade do robô não pode ser feito variando-se a tensão da bateria de forma linear, pois isso dissiparia muita energia em calor e reduziria o torque. A técnica mais eficiente para este fim é a Modulação por Largura de Pulso, ou PWM (*Pulse Width Modulation*).

O PWM consiste em aplicar ao motor um sinal digital de frequência fixa que alterna rapidamente entre ligado (tensão máxima) e desligado (0V). A velocidade média percebida pelo motor depende da razão entre o tempo em que o sinal permanece ligado (t_{on}) e o período total do ciclo (T). Essa razão é chamada de Ciclo de Trabalho ou *Duty Cycle* (D) (Kart, 2001).

A tensão média V_{med} entregue ao motor é dada por

$$V_{med} = V_{max} \cdot D, \tag{2.4}$$

onde V_{max} é a tensão da bateria e D varia de 0 (0%, motor parado) a 1 (100%, velocidade máxima). O ESP32 gera esses sinais via *hardware*, permitindo um controle suave e preciso da aceleração do veículo sem a necessidade de componentes analógicos complexos (Alamsyah *et al.*, 2025).

3 TRABALHOS RELACIONADOS

O avanço da robótica móvel e da Internet das Coisas (IoT) tem impulsionado o desenvolvimento de veículos autônomos e teleoperados para as mais diversas aplicações, desde a exploração de ambientes perigosos até a logística industrial e competições de robótica. Neste cenário, a evolução dos sistemas embarcados, marcada pelo surgimento de microcontroladores de alto desempenho e baixo custo como o ESP32, permitiu que arquiteturas de controle complexas, antes restritas a equipamentos industriais de alto valor, mas que agora podem ser implementadas em plataformas acessíveis. O desenvolvimento de protótipos robóticos com tração diferencial (4WD) e conectividade sem fio nativa surge, portanto, como uma tendência técnica relevante para a validação de algoritmos de navegação e sistemas de controle remoto via *web*.

Este capítulo apresenta e discute trabalhos relevantes da literatura que fundamentam a engenharia do protótipo desenvolvido, estabelecendo um comparativo entre as arquiteturas de *hardware* tradicionais e as soluções modernas baseadas em IoT. A análise a seguir prioriza estudos que abordam o desenvolvimento técnico de robôs móveis, focando em estratégias de construção mecânica, algoritmos de desvio de obstáculos com sensores ultrassônicos e a implementação de interfaces de controle via servidor *web*, destacando como a integração destas tecnologias pode resultar em plataformas funcionais, robustas e economicamente viáveis.

3.1 Rover Controlado por Servidor Web e Com Detecção de Objetos

(Narayana *et al.*, 2024) desenvolveram um *rover* versátil equipado com braço robótico, onde a arquitetura de controle tem como base a integração de microcontroladores ESP32 operando como servidores *web*. O sistema permite a operação remota de motores e servos através de uma interface acessível via navegador em rede local, retirando a necessidade de um aplicativo dedicado. Além do controle de movimento, o projeto possui um módulo de detecção de objetos baseado no algoritmo YOLOv3, processado externamente via Python, demonstrando a capacidade do ESP32 em atuar como nó central de comunicação em aplicações de robótica móvel complexas. A arquitetura proposta por (Narayana *et al.*, 2024). valida a escolha do ESP32 e do protocolo HTTP para o presente trabalho, confirmando que a solução baseada em servidor *web* embarcado oferece a flexibilidade e a interoperabilidade necessárias para o controle eficiente de atuadores em plataformas robóticas de baixo custo.

3.2 Implementação de Um Robô Móvel Autônomo Baseado em ESP-32

(Hamici, 2023) apresenta uma abordagem mais rigorosa para o desenvolvimento de robôs móveis autônomos de tração diferencial, fundamentada na modelagem matemática da cinemática e no uso do microcontrolador ESP32 operando com o sistema operacional de tempo real FreeRTOS. O trabalho detalha a implementação de algoritmos de planejamento de trajetória e localização baseada em odometria, demonstrando que o ESP32 possui capacidade de processamento suficiente para gerenciar tarefas concorrentes críticas, como a leitura de *encoders* e o controle Proporcional-Integral-Derivativo (PID) dos motores, sem latência perceptível. Além da validação do *hardware*, o trabalho propõe uma arquitetura de comunicação baseada no protocolo *Message Queuing Telemetry Transport* (MQTT) para telemetria e controle, reforçando a viabilidade do ESP32 como núcleo de processamento em sistemas robóticos que exigem alta confiabilidade e conectividade. Para o presente projeto, os resultados de Hamici fundamentam a escolha técnica do ESP32, comprovando que sua arquitetura *dual-core* é robusta o bastante para suportar tanto a lógica de navegação autônoma quanto a interface de controle remoto proposta, sem a necessidade de computadores de bordo adicionais.

3.3 Desenvolvimento de Um Robô Móvel com ESP32 Para Olimpíadas de Robótica

A utilização de plataformas robóticas baseadas no microcontrolador ESP32 é validada por (Oliveira *et al.*, 2024) no contexto de competições de alto desempenho, especificamente na Olimpíada Brasileira de Robótica (OBR). O trabalho descreve o desenvolvimento integral de um robô móvel autônomo para a modalidade de resgate, utilizando técnicas de manufatura digital como corte a laser em MDF e impressão 3D para a estruturação mecânica do chassi. A pesquisa detalha a integração de sensores como ultrassônicos no ESP32, além de correção de problemas como a compatibilidade de tensão através de reguladores dedicados e validando a eficiência energética de baterias Li-Ion em regime de competição. Para o presente projeto, os resultados de (Oliveira *et al.*, 2024) contribuíram no desenvolvimento e mostram a viabilidade técnica da construção de chassis personalizados de baixo custo. Além de reforçar a confiabilidade do ESP32 como unidade central de processamento capaz de operar em ambientes dinâmicos que exigem resposta rápida dos sensores e estabilidade mecânica.

3.4 Veículo Robótico para Desvio de Obstáculos com Sensor HC-SR04

A implementação de algoritmos de navegação reativa baseados em sensores ultrassônicos é explorada por (Venkatesh *et al.*, 2021), que desenvolveram um veículo robótico capaz de detectar e desviar de obstáculos de forma autônoma. O sistema utiliza o sensor HC-SR04 montado em um servo motor para realizar a varredura do ambiente, e um microcontrolador Arduino Uno para processar os dados de distância. A lógica de controle descrita opera medindo continuamente a distância frontal; ao detectar um obstáculo a menos de 30 cm, o robô interrompe o movimento, avalia as distâncias laterais e decide a nova trajetória em direção ao caminho livre. Este trabalho é relevante para a presente pesquisa pois valida a eficácia e a simplicidade do algoritmo de desvio de obstáculos baseado em limiares de distância, uma estratégia que foi adaptada e implementada no modo autônomo deste projeto, substituindo o Arduino pelo ESP32 para integrar essa funcionalidade à conectividade remota.

3.5 Plataforma de Aprendizagem de Cinemática Omnidirecional Usando ESP-32

A versatilidade do microcontrolador ESP32 no controle de sistemas robóticos complexos é explorada por (Alamsyah *et al.*, 2025), que desenvolveram uma plataforma para estudo de cinemática aplicada a robôs omnidirecionais. O trabalho detalha a arquitetura de *firmware* necessária para gerenciar múltiplos motores simultaneamente e a implementação de uma interface gráfica para envio de comandos e monitoramento via protocolo sem fio. A pesquisa demonstra a capacidade de processamento do ESP32 para executar cálculos vetoriais de movimento em tempo real, mantendo a estabilidade da comunicação. Para o presente projeto, os resultados de (Alamsyah *et al.*, 2025) fundamentam a escolha do controlador, demonstrando que se este é capaz de gerenciar a complexidade de um sistema holonômico (três eixos de liberdade), apresenta desempenho mais que suficiente para a arquitetura de tração diferencial (4WD) adotada neste trabalho, garantindo uma margem de processamento segura para as rotinas de servidor *web* e lógica autônoma.

3.6 Robô Móvel de Baixo Custo Guiado por Radiofrequência

A evolução da arquitetura de *hardware* para robótica móvel de baixo custo é evidente ao analisar o trabalho de (Honda *et al.*, 2013), que descreve o desenvolvimento de um robô controlado remotamente via módulos de radiofrequência de 433 MHz. O sistema foi projetado

em torno do microcontrolador PIC16F628A, exigindo a construção de circuitos dedicados tanto para a transmissão quanto para a recepção e decodificação dos sinais no robô. Embora os autores tenham atingido o objetivo de baixo custo através da manufatura artesanal do chassi com materiais recicláveis, a complexidade de integração de módulos de RF externos e a necessidade de protocolos de comunicação serial personalizados representam desafios de escalabilidade que são superados pelas tecnologias atuais. O presente projeto se posiciona como uma evolução natural dessa abordagem: ao substituir o conjunto PIC/RF pelo ESP32, unifica-se o processamento e a comunicação em um único *chip* via Wi-Fi, simplificando drasticamente a eletrônica embarcada e eliminando a necessidade de *hardware* transmissor dedicado conectado ao computador.

3.7 Análise Comparativa

Para sintetizar as diferenças entre as abordagens estudadas e a proposta deste TCC, elaborou-se a Tabela 1. A comparação visual destaca a arquitetura de *hardware*, o método de controle e as funcionalidades integradas em cada projeto.

Tabela 1 – Comparativo de recursos entre os trabalhos relacionados e a proposta atual

Trabalhos / Autores	Controlador		Interface		Modo	
	ESP32	Outros	Web	RF/App	Auto	Manual
(Honda *et al.*, 2013)		✓		✓		✓
(Venkatesh *et al.*, 2021)		✓			✓	
(Hamici, 2023)	✓			✓	✓	
(Oliveira *et al.*, 2024)	✓			✓	✓	
(Alamsyah *et al.*, 2025)	✓			✓		✓
(Narayana *et al.*, 2024)	✓		✓			✓
Este Trabalho	✓		✓		✓	✓

Fonte: Elaborada pelo autor.
Nota: Legenda: Outros (Arduino/PIC); RF (Radiofrequência); App (Aplicativo Dedicado).

A Tabela 1 demonstra que, embora existam trabalhos focados em autonomia (como (Venkatesh *et al.*, 2021) e (Hamici, 2023)) e outros focados em controle remoto (como (Honda *et al.*, 2013) e (Narayana *et al.*, 2024)), a proposta deste TCC se diferencia pela integração completa. O protótipo desenvolvido unifica o processamento de alto desempenho (ESP32), a acessibilidade da interface *Web* (sem necessidade de instalação de aplicativos) e a capacidade híbrida de operação (Autônoma e Manual) em uma única plataforma de baixo custo.

4 METODOLOGIA

Esse Capítulo aborda o processo de desenvolvimento e engenharia adotada para a construção do robô. Essa metodologia vai desde o projeto esquemático até o procedimento de testes e validação didática, passando pela fase de escolha de materiais, montagem do protótipo, desafios encontrados e desenvolvimento do *software* embarcado.

4.1 Abordagem Metodológica e Fases de projeto

Para a construção desse protótipo robótico, o processo de desenvolvimento seguiu um processo completamente estruturado de engenharia, onde o foco foi na pesquisa aplicada seguida de prototipagem experimental. A ideia principal foi evitar o máximo de teória e partir para um desenvolvimento prático do projeto.

Para garantir que o projeto fosse robusto, seguro, funcional e entregue dentro dos prazos, a abordagem adotada seguiu o seguinte cronograma:

Tabela 2 – Fases da metodologia de desenvolvimento do protótipo didático

Fase	Etapa	Objetivo Principal
1	Revisão de Requisitos e Seleção de Componentes	Estabelecer os requisitos funcionais e selecionar os componentes de *hardware* (sensores, atuadores, MCU).
2	Projeto e Modelagem Esquemática	Desenvolver e validar o esquemático elétrico completo do sistema utilizando o *software* KiCad.
3	Montagem Física e Integração	Realizar a implementação física do *hardware* (soldagem, montagem mecânica) conforme o projeto esquemático.
4	Desenvolvimento do *Software* Embarcado	Implementar a lógica de controle (*Firmware*), incluindo os modos de operação remoto (IoT) e autônomo.
5	Testes Experimentais e Validação Didática	Executar testes funcionais e experimentos para validar a eficácia do protótipo como ferramenta de ensino.

Fonte: Elaborada pelo autor.
Nota: As fases foram executadas de forma sequencial e iterativa, onde a conclusão de uma etapa serviu como base para a próxima.

As seções seguintes deste capítulo detalharão cada uma dessas fases.

4.2 Materiais e Componentes Utilizados

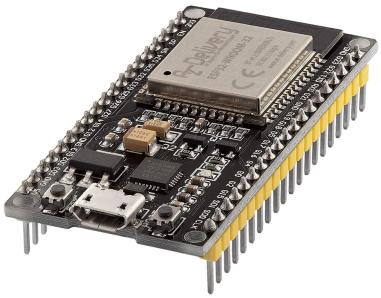
A seleção de componentes para este projeto, correspondente à Fase 1 da metodologia, foi feita abrangendo entre funcionalidade técnica, baixo custo, ampla disponibilidade no mercado

nacional. Pensando em uma apresentação clara, os componentes foram agrupados por sistema funcional, e suas justificativas de escolha são detalhadas a seguir.

4.2.1 Plataforma de Controle e Processamento

A unidade central de controle e processamento de todo o sistema é o microcontrolador ESP32, especificamente a placa de desenvolvimento ESP32 *DevKit* V1, conforme a Figura 3.

Figura 3 – Placa de desenvolvimento, ESP32 *Dev Kit* V1



Fonte: https://www.espressif.com/en/products/devkits

A escolha deste componente foi estratégica, e a principal justificativa para sua seleção é a conectividade Wi-Fi nativa (padrão 802.11 b/g/n). Essa característica é um requisito fundamental do projeto, pois viabiliza o modo de operação remota aplicando conceitos de IoT no demonstrador didático e, permitindo que ele seja controlado a partir de qualquer dispositivo conectado à mesma rede, como um *smartphone* ou computador.

Além da conectividade, esse microcontrolador se destaca por seu poder de processamento, baseado em um processador *dual-core* Tensilica Xtensa LX6. Essa característica é essencial para gerenciar as múltiplas tarefas do protótipo simultaneamente como processamento das leituras dos quatro sensores utilizados; Execução da lógica do modo autônomo para correção da trajetória; Manter a comunicação Wi-Fi e responder a comandos remotos; Gerar os sinais de controle *Pulse Width Modulation* (PWM) para o *driver* dos motores.

A adoção da placa no formato *DevKit* V1 simplificou o processo de prototipagem (Fase 3), pois ela já integra componentes essenciais como um regulador de tensão para alimentar o *chip*, uma interface *USB-Serial* para fácil programação e depuração, e pinos de proposito geral *General-Purpose Input/Output* (GPIO) acessíveis em *headers* padrão. (Ahmed *et al.*, 2021)

4.2.2 Sistema de Sensoriamento

O sistema de sensoriamento é a principal fonte de percepção ambiental do protótipo, sendo o componente-chave para viabilizar o modo de operação autônomo e a análise de movimentos. Para esta finalidade, foram selecionados e empregados quatro módulos sensores ultrassônicos, modelo HC-SR04, posicionados estrategicamente no chassi para cobertura frontal, traseira, lateral esquerda e lateral direita.

Figura 4 – Sensor ultrassônico HC-04

Fonte: https://www.makerhero.com/produto/sensor-de-distancia-ultrassonico-hc-sr04

A justificativa para a escolha deste tipo de sensor baseia-se na sua excelente relação custo-benefício, robustez e simplicidade de interfaceamento com o microcontrolador, requerendo apenas um pino digital de TRIG e um de ECHO por sensor. O princípio de funcionamento consiste na emissão de um pulso sônico e na medição do tempo de retorno do seu eco, permitindo ao *firmware* calcular a distância do obstáculo. (Wang *et al.*, 2018)

O uso de quatro unidades permite que o sistema embarcado execute funções complexas. O sensor frontal é monitorado para prevenção de colisão ao detectar um objeto a 50 cm, que aciona uma parada imediata. Em caso de bloqueio, o *firmware* avalia as leituras dos sensores laterais e traseiro para determinar a rota de fuga mais viável, seja virando ou recuando.

Para o objetivo central deste trabalho, os sensores também permitem a possibilidade de ao ser colocado em um ambiente controlado e pequeno tipo um corredor, o *firmware* meça continuamente sua distância das paredes. Isso habilita a implementação de algoritmos de controle que corrigem a trajetória do carrinho, mantendo-o equidistante das laterais e, assim, executando um movimento retilíneo de forma controlada.

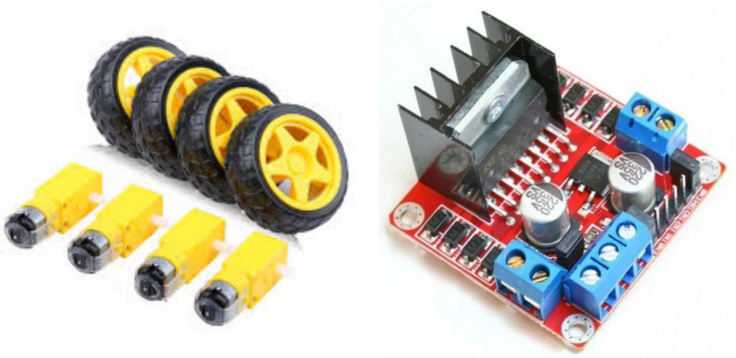
4.2.3 Sistema de Atuação e tração

O sistema de atuação e tração é responsável pelo deslocamento físico do protótipo. A base mecânica consiste em um chassi 4WD de acrílico, um *kit* que já integra os quatro motores DC (tensão nominal de 3-6V) que fornecem tração independente às rodas. (Oltean, 2019)

Como a plataforma de controle opera com sinais lógicos de baixa corrente e tensão de no máximo 3.3V, ela é incapaz de alimentar diretamente os motores. Para solucionar esse problema, teve que ser utilizado um módulo *driver* de potência L298N. Este módulo atua como a interface de potência, recebendo os comandos lógicos do ESP32 e comutando a corrente e tensão, originadas da bateria, para os motores.

Uma decisão de projeto fundamental foi a ligação dos quatro motores. Eles foram associados em pares: os dois motores do lado esquerdo foram ligados em série e conectados a um canal da Ponte H, e os dois motores do lado direito foram ligados em série e conectados ao outro canal. Esta topologia permite o controle de velocidade e direção do tipo "tanque"(*skid-steering*) (Golconda, 2005). O controle direcional frente/ré de cada par é feito por pinos GPIO, enquanto o controle de velocidade é gerenciado por dois sinais PWM independentes, permitindo variar a potência entregue a cada lado do robô.

Figura 5 – Motores DC TT de 6V e ponte H L298n



Fonte: https://www.aliexpress.com/

4.2.4 Sistema de Alimentação e Segurança

O sistema de alimentação do protótipo foi estruturado em torno de um *pack* de baterias customizado, projetado para fornecer a tensão necessária para os motores e, simultaneamente, alimentar a lógica de controle. A fonte primária consiste em duas baterias recarregáveis de íon-lítio, modelo 18650 (3.7V, 6800mAh) (Muenzel *et al.*, 2015). Estas células foram soldadas

em ligação série pelo próprio autor, resultando em uma tensão nominal de 7.4V. Para garantir a segurança operacional e a longevidade das baterias, protegendo-as contra sobrecarga, descarga profunda e curto-circuito, foi integrado ao *pack* um módulo do tipo *Battery Management System* (BMS) circular, modelo BS 2S 5A.

Figura 6 – Baterias, BMS e Regulador utilizados

Fonte: https://www.aliexpress.com/

Este barramento principal de 7.4V atende a duas necessidades distintas do projeto. Primeiramente, ele alimenta diretamente o módulo de potência Ponte H L298N, fornecendo a tensão necessária para que os motores de tração operem com torque suficiente (Kumar *et al.*, 2019). Como medida de segurança e controle, um interruptor geral do tipo chave seletora foi instalado entre a saída do BMS e a entrada de tensão coletor comum *Voltage Common Collector* (VCC) da Ponte H, permitindo o desligamento completo do sistema de tração.

Em segundo lugar, o microcontrolador ESP32, que opera com lógica de 3.3V e possui uma entrada de alimentação V_{IN} tipicamente limitada a 5V, não pode ser alimentado pelos 7.4V da bateria. Para resolver essa incompatibilidade, foi utilizado um regulador de tensão *step-down* (*buck converter*), modelo LM2596. Este módulo foi conectado à saída da bateria e calibrado para fornecer uma tensão de saída estável de 5V (Houari *et al.*, 2023).

A saída de 5V do regulador LM2596 é, então, utilizada para alimentar o ESP32 através de seu pino V_{IN}. Como medida de proteção adicional para o microcontrolador, que é o componente mais sensível do sistema, um fusível foi adicionado entre a saída do regulador *step-down* e a entrada de alimentação do ESP32. Este fusível protege o ESP32 contra picos de corrente ou falha do regulador que pudesse expor o microcontrolador à tensão total da bateria.

4.3 Projeto e Modelagem de Hardware

A fase de projeto e modelagem de *hardware* consiste em uma etapa da fase 2 de modelagem esquemática, e consistiu na concepção detalhada da arquitetura elétrica do sistema, garantindo a integridade das conexões e a viabilidade técnica do protótipo antes de sua montagem física. Para esta etapa, utilizou-se o *software* KiCad, uma ferramenta de automação de *design* eletrônico ou *Electronic Design Automation* (EDA), que permitiu a elaboração do esquema elétrico completo e o mapeamento de todos os pinos e interfaces do sistema. Esta seção detalha a arquitetura desenvolvida, descrevendo a integração entre a unidade de controle, os sensores, atuadores e o gerenciamento de energia, conforme documentado no esquemático do projeto (Dandl *et al.*, 2024).

4.3.1 Seleção de Componentes Principais

A definição da arquitetura eletrônica priorizou a compatibilidade elétrica entre os módulos e a simplificação do diagrama esquemático. A escolha do ESP32 como unidade central, para além de sua conectividade, baseou-se na disponibilidade de pinos de Entrada/Saída GPIO suficientes para gerenciar simultaneamente a leitura dos quatro sensores ultrassônicos e o controle dos dois canais da Ponte H, eliminando a necessidade de expansores de porta ou multiplexadores que complexificariam o projeto do *hardware* (Ahmed *et al.*, 2021).

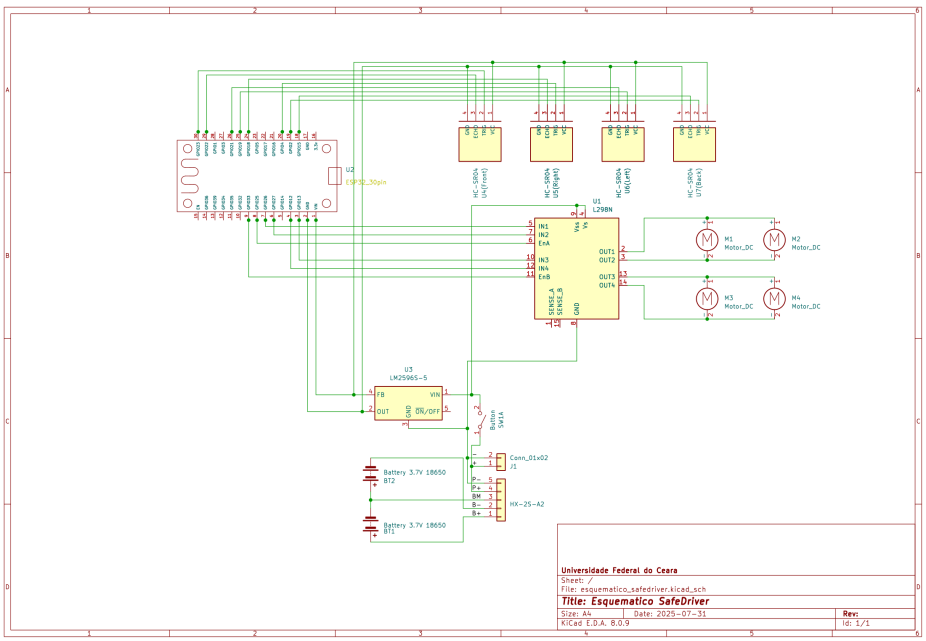
Para o sistema de atuação, a seleção do *driver* L298N foi determinante para o *design* de potência. Sua topologia de ponte H dupla permitiu concentrar o controle de quatro motores em um único módulo, isolando galvanicamente a alta corrente da bateria (7.4V) dos circuitos lógicos sensíveis. Da mesma forma, os sensores HC-SR04 foram escolhidos por operarem com interface digital padrão, o que facilitou o mapeamento de pinos no KiCad e permitiu uma integração direta com o nível lógico do microcontrolador, exigindo apenas o gerenciamento correto das linhas de alimentação de 5V provenientes do regulador (Oltean, 2019).

4.3.2 Esquema Elétrico e Integração

O projeto elétrico completo do sistema foi consolidado em um diagrama esquemático unificado, apresentado na Figura 7. Este diagrama ilustra todas as interconexões físicas entre os módulos descritos na seção anterior, servindo como guia para a montagem do protótipo.

A arquitetura de integração pode ser detalhada em três subsistemas principais de

Figura 7 – Esquemático completo do projeto



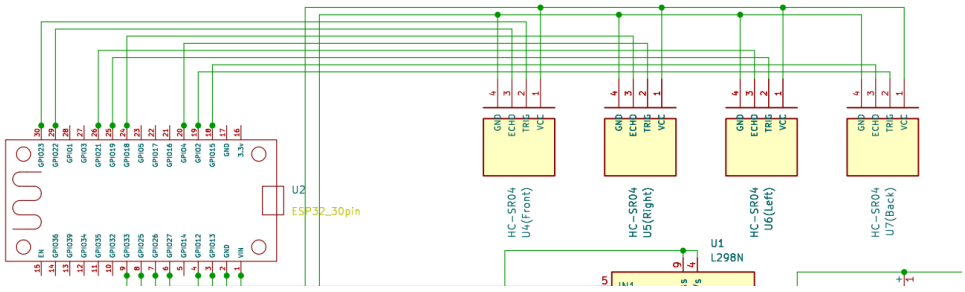
Fonte: Elaborada pelo autor.

conexão, todos convergindo para a unidade central de processamento.

4.3.2.1 Integração do Sistema de Sensoriamento

Conforme detalhado no esquemático da Figura 7 e expandido na Figura 8, os quatro sensores ultrassônicos HC-SR04 (identificados como U4, U5, U6 e U7) são conectados diretamente aos pinos de entrada/saída digital (GPIO) do ESP32. Cada sensor requer quatro conexões: alimentação (VCC ligado a 5V), terra (GND) e dois pinos de dados.

Figura 8 – Sistema de sensoriamento



Fonte: Elaborada pelo autor.

A distribuição de pinos dos sensores teve que ser feita visando evitar erros com funções internas do ESP32, resultando no seguinte mapeamento:

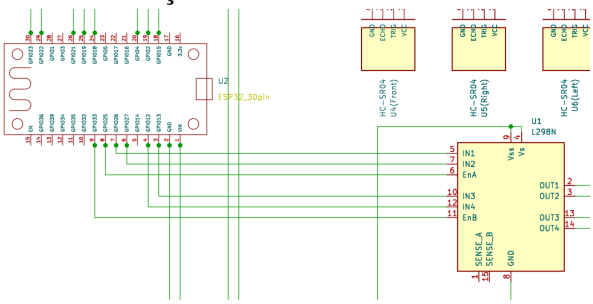
– **Frontal (U4):** O pino de disparo está conectado à GPIO 23 e o pino de eco à GPIO 22.

- **Traseiro (U7):** O pino de disparo está conectado à GPIO 2 e o pino de eco à GPIO 15.
- **Direito (U5):** O pino de disparo está conectado à GPIO 4 e o pino de eco à GPIO 18.
- **Esquerdo (U6):** O pino de disparo está conectado à GPIO 19 e o pino de eco à GPIO 21.

4.3.2.2 Integração do Sistema de Atuação

O controle dos motores é intermediado pelo *driver* L298N (identificado como U1 conforme o esquemático da Figura 7 e expandido na Figura 9). A interface de controle entre o ESP32 e a Ponte H utiliza seis linhas de sinal, divididas em dois grupos (Canal A e Canal B):

Figura 9 – Sistema de Atuação



Fonte: Elaborada pelo autor.

- **Controle de Velocidade (PWM):** Os pinos *Enable A* e *Enable B* do *driver* estão conectados, respectivamente, aos pinos D25 e D33 do ESP32. Estes pinos são configurados para saída de sinal PWM, permitindo a modulação da velocidade dos motores.
- **Controle de Direção:** A lógica de rotação (horário/anti-horário) é definida pelos pares de pinos de entrada. O par IN1/IN2 controla o lado esquerdo (conectado a D26 e D27), enquanto o par IN3/IN4 controla o lado direito (conectado a D12 e D13).

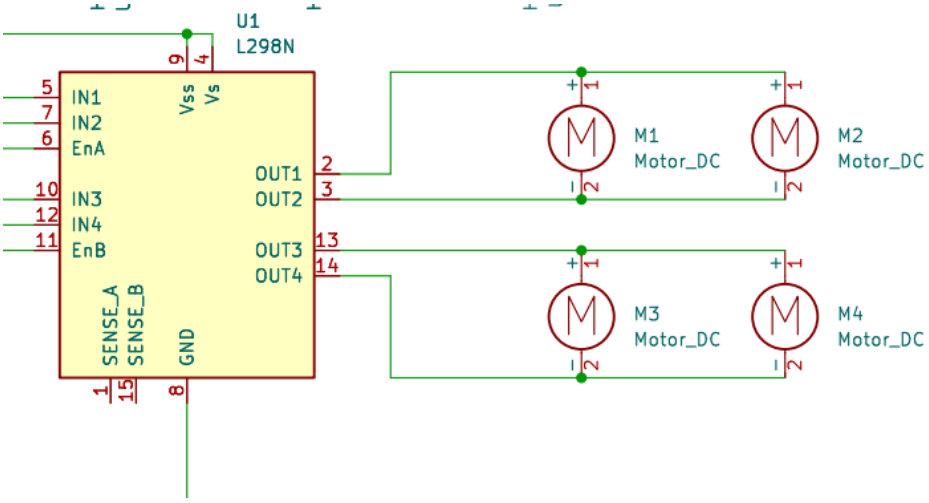
Do lado da potência, as saídas OUT1 e OUT2 alimentam os dois motores do lado esquerdo (ligados em série), e as saídas OUT3 e OUT4 alimentam os motores do lado direito, conforme a topologia de tração 4WD adotada (Golconda, 2005), isso pode ser confirmado no esquemático representado na Figura 7 e expandido na Figura 10.

4.3.2.3 Gerenciamento de Energia e Proteções

O esquemático também detalha o fluxo de energia do sistema. A bateria de 7.4V (BT1+BT2) conecta-se à entrada de alta potência do L298N e, paralelamente, à entrada do regulador LM2596.

Um ponto crítico de segurança ilustrado no projeto é a alimentação do ESP32. A

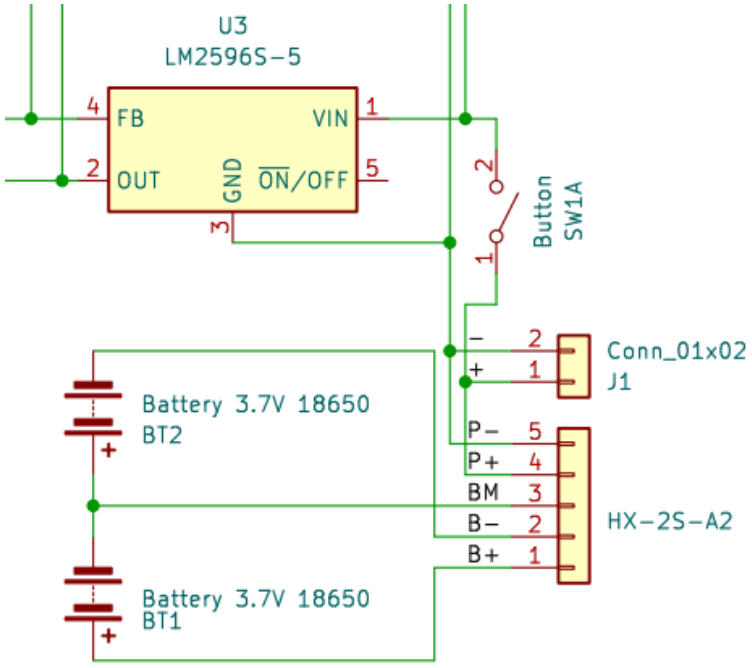
Figura 10 – Ligação dos Motores



Fonte: Elaborada pelo autor.

saída regulada de 5V do LM2596 não é conectada diretamente; ela passa por um fusível de proteção (F1) antes de chegar ao pino VIN do microcontrolador. Além disso, o diagrama evidencia a referência de terra comum (GND) unificando todos os módulos (bateria, *drivers*, sensores e Microcontrolador), condição obrigatória para o funcionamento correto dos sinais lógicos. Esses detalhes de Alimentação e energia podem ser verificados no esquemático completo representado na Figura 7 e expandido na Figura 11.

Figura 11 – Sistema de alimentação e Proteção

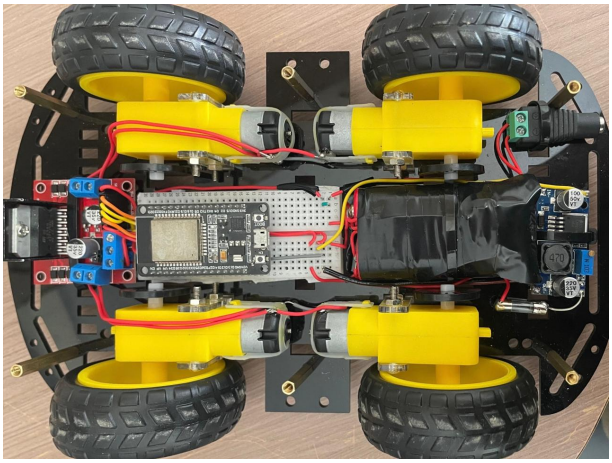


Fonte: Elaborada pelo autor.

4.4 Montagem e Integração Física do Protótipo

A etapa de montagem física, correspondente à Fase 3 da metodologia, consistiu na transição do projeto esquemático para o chassi robótico. A disposição dos componentes, apresentada na Figura 12, não foi aleatória; ela seguiu critérios de distribuição de massa e acessibilidade para manutenção.

Figura 12 – Vista Superior da Montagem Interna dos Componentes no Chassi



Fonte: Elaborada pelo autor.

4.4.1 Disposição Mecânica e Estrutural

A estrutura base utilizada foi o chassi em acrílico do tipo 4WD. A organização espacial dos componentes priorizou o equilíbrio do centro de gravidade do robô. Conforme observável na figura, o pacote de baterias, por ser o componente mais pesado, foi posicionado centralmente na parte inferior do chassi, enquanto o *driver* de motores L298N, que possui dissipador de calor e ocupa volume considerável, foi alocado na extremidade oposta.

O microcontrolador ESP32 foi posicionado no centro geométrico do robô. Esta localização estratégica minimiza o comprimento dos cabos de sinal tanto para os sensores nas extremidades, quanto para o *driver* de potência, reduzindo a impedância e potenciais interferências eletromagnéticas nos fios de comunicação. Todas essas ligações podem ser vistas em detalhes na Figura 13.

Figura 13 – Disposição Mecânica e Estrutral no Chassi

Fonte: Elaborada pelo autor.

4.4.2 Manufatura e Integração Eletrônica

Diferente de uma placa de circuito impresso definitiva, optou-se pela fixação do ESP32 sobre uma matriz de contatos (*mini-protoboard*) adesivada ao chassi. Esta abordagem modular permitiu a fácil remoção do microcontrolador para reprogramação via USB ou substituição em caso de falha, sem a necessidade de dessoldagem.

Um destaque do processo de manufatura foi a confecção do sistema de alimentação. O *pack* de baterias foi montado manualmente, soldando-se as células 18650 em série e integrando o BMS diretamente aos polos, sendo o conjunto final isolado com fita de alta isolação. Logo abaixo do pacote de energia, fixaram-se o regulador de tensão LM2596 e o fusível de proteção em suporte dedicado, garantindo que os componentes de segurança estivessem fisicamente próximos à fonte de energia, protegendo todo o circuito conforme pode ser observado na Figura 14.

Figura 14 – Disposição do Sistema de Alimentação e Proteção

Fonte: Elaborada pelo autor.

4.5 Projeto e Desenvolvimento do Software Embarcado

Nesta etapa, correspondente à Fase 4 da metodologia apresentada, o foco do projeto partiu então para a implementação lógica. O desenvolvimento do *firmware* foi realizado utilizando a linguagem C++ dentro do ambiente de desenvolvimento Visual Studio Code, utilizando a extensão PlatformIO para o gerenciamento de bibliotecas e compilação cruzada para a arquitetura do ESP32. O código foi estruturado para operar em tempo real, sendo responsável por orquestrar simultaneamente o controle de baixo nível dos periféricos e as camadas de alto nível, como a conectividade Wi-Fi e o servidor *web* embarcado.

4.5.1 Arquitetura e Controle de Baixo Nível

A arquitetura do *software* foi construída sobre o ciclo de execução padrão da plataforma Arduino, composta pelas funções de configuração e laço de repetição infinito. Na fase de inicialização, o sistema é responsável por definir os modos de operação dos pinos GPIO, estabelecer a conexão Wi-Fi e configurar os temporizadores necessários para o controle dos motores.

Um ponto crucial no desenvolvimento para o ESP32 é a geração de sinais PWM para o controle de velocidade. Diferente de microcontroladores tradicionais, o ESP32 utiliza o periférico LEDC para gerenciar esses sinais. No código desenvolvido, foram configurados dois canais PWM independentes operando a uma frequência de 1000 Hz com resolução de 8 bits, permitindo valores de controle entre 0 e 255.

O trecho de código a seguir demonstra a implementação desta configuração inicial e a associação dos pinos físicos aos canais lógicos de controle:

Código-fonte 1 – Configuração dos pinos e canais PWM no ESP32

```
1 void setup() {
2 Serial.begin(115200);
3
4 // Configuracao dos pinos de direcao como saida
5 pinMode(frenteMotor1, OUTPUT);
6 pinMode(trasMotor1, OUTPUT);
7 pinMode(frenteMotor2, OUTPUT);
```

```
8  pinMode(trasMotor2, OUTPUT);
9
10 // Associacao do PWM aos pinos de velocidade (Enable A e B)
11 ledcAttachPin(pwmMotor1, 0); // Motor Esquerdo no Canal 0
12 ledcAttachPin(pwmMotor2, 1); // Motor Direito no Canal 1
13
14 // Configuracao da frequencia (1kHz) e resolucao (8 bits)
15 ledcSetup(0, 1000, 8);
16 ledcSetup(1, 1000, 8);
17
18 // ... inicializacao do WiFi e Sensores ...
19 }
```

4.5.2 Algoritmo de Navegação Autônoma

Quando o sistema é comutado para o modo autônomo, o *firmware* deixa de responder aos comandos via Wi-Fi e passa a operar em malha fechada, utilizando os dados do sistema de sensoriamento como *Feedback*. O algoritmo implementado realiza a leitura sequencial dos quatro sensores ultrassônicos e executa uma lógica de decisão baseada em prioridades para evitar obstáculos e manter o movimento.

Foi estabelecida no código uma constante de segurança fixada em 50 cm. A lógica de navegação opera monitorando continuamente o sensor frontal; caso a leitura seja inferior a este limiar, o robô interrompe o movimento imediatamente e inicia a sub-rotina de desvio (Almasri *et al.*, 2016).

O trecho de código a seguir ilustra a estrutura de decisão implementada no laço principal, onde o sistema compara as distâncias laterais disponíveis para decidir a rota de fuga mais segura:

Código-fonte 2 – Lógica de decisão e desvio de obstáculos

```
1 // Verifica se ha obstaculo frontal dentro da margem de
     seguranca (50cm)
2 if (currentDistanceFrontCm <= DISTANCIA_PARADA_AUTONOMO_CM)
```

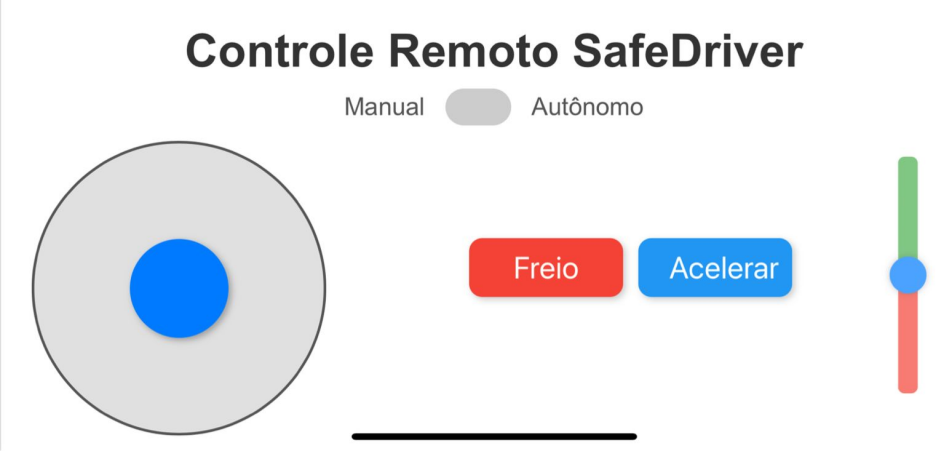
```
    {
parar(); // Parada imediata de seguranca

// Avalia qual lado possui mais espaco livre
// Se a esquerda tiver mais espaco (ou igual), vira a
   esquerda
if (distLeft > 0 && distLeft >= distRight) {
virarEsquerda();
}
// Caso contrario, vira a direita
else if (distRight > 0 && distRight > distLeft) {
virarDireita();
}
// Se ambas as laterais estiverem bloqueadas
else {
tras(); // Recua
}

} else {
frente(); // Caminho livre, mantem movimento retilineo
}
```

4.5.3 Interface Web e Conectividade IoT

Para viabilizar a operação remota (Modo Manual), o ESP32 foi configurado para atuar como um servidor *web*, hospedando uma interface gráfica interativa. Esta página, desenvolvida utilizando HTML, CSS e JavaScript, é armazenada na memória *flash* do microcontrolador e pode ser acessada por qualquer dispositivo conectado à mesma rede Wi-Fi através do endereço IP do robô.

A interface, ilustrada na Figura 15, apresenta um *joystick* virtual para controle direcional, um controle deslizante *slider* para ajuste de velocidade e botões de ação imediata (Freio e Acelerar), além de uma chave seletora para alternar entre os modos Manual e Autônomo.

Figura 15 – Interface Web de Controle



Fonte: Elaborada pelo autor.

A comunicação entre a interface do usuário (*Front-end*) e o *firmware* (*Back-end*) ocorre de forma assíncrona, utilizando a tecnologia AJAX/Fetch API. Quando o usuário interage com os controles na tela, o código JavaScript envia requisições HTTP *GET* em segundo plano, contendo os parâmetros de velocidade e direção, sem a necessidade de recarregar a página.

O código abaixo demonstra como o servidor embarcado intercepta essas requisições, extrai os parâmetros enviados pela *URL* e atualiza as variáveis globais que comandam os motores:

Código-fonte 3 – Tratamento de requisições HTTP para controle remoto

```
1  // Rota para processar comandos de controle vindos da
       interface web
2  server.on("/controle", HTTP_GET, []() {
3  // Verifica se o parametro de velocidade foi recebido
4  if (server.hasArg("vel")) {
5  // Converte o argumento da URL para inteiro
6  velocidadeCrua = server.arg("vel").toInt();
7
8  // Mapeia o valor recebido para o range PWM dos motores
       (0-255)
9  if (modoAutonomoAtivo) {
10 velocidadeMapeada = mapValue(velocidadeCrua, 0, 255, 90,
       255);
11 } else {
12 velocidadeMapeada = mapValue(abs(velocidadeCrua), 0, 255,
```

```
    90, 255);
}
}

// Envia resposta de sucesso ao navegador
server.send(200, "text/plain", "OK");

});
```

Esta arquitetura IoT elimina a necessidade de desenvolvimento de aplicativos nativos específicos para cada sistema operacional, garantindo que esse demonstrador didático posssa ser utilizado em qualquer dispositivo.

4.6 Procedimentos de Teste e Validação Didática

A etapa final da metodologia consistiu na validação experimental do protótipo, realizada através de ensaios funcionais em ambiente controlado. O objetivo principal desta fase foi verificar a integridade da montagem e também a capacidade do algoritmo de tomar decisões corretas diante de obstáculos reais, validando o sistema como uma ferramenta demonstradora de movimentos.

Os testes foram conduzidos submetendo o robô, já completamente montado conforme a Figura 16, a diferentes cenários de obstrução física. Inicialmente, avaliou-se a resposta do sensor frontal, posicionando objetos a distâncias variadas para confirmar a parada automática no limite de segurança de 50 cm. Em seguida, foram simuladas situações de "beco sem saída"e bloqueios parciais (apenas esquerda ou apenas direita bloqueadas) para observar se a lógica de prioridade implementada no *firmware* acionava os motores corretamente para a direção livre (Cañadas-Aránega *et al.*, 2024).

Observou-se que o protótipo respondeu de forma consistente aos estímulos externos, alternando entre o movimento retilíneo (quando o caminho estava livre) e as manobras de correção de trajetória (quando detectava barreiras), cumprindo assim os requisitos funcionais estabelecidos para a análise de movimentos.

5 RESULTADOS

Este capítulo apresenta os resultados obtidos com a implementação do veículo robótico, consolidando as etapas de engenharia descritas na metodologia. A discussão a seguir aborda a validação funcional do sistema integrado em ambiente controlado, avaliando desde a robustez da montagem mecânica e a eficiência da arquitetura eletrônica até a resposta prática dos algoritmos de controle remoto via interface *web* e de navegação autônoma. Além disso, é apresentada uma análise econômica detalhada, demonstrando a viabilidade financeira do projeto frente às soluções comerciais existentes no mercado educacional.

5.1 Consolidação do Protótipo e Hardware

Após a montagem física do projeto, obteve-se uma plataforma experimental robusta e funcional. A integração entre o chassi de acrílico e os componentes eletrônicos seguiu o que foi apresentado no projeto esquemático, não havendo grandes diferenças dimensionais ou elétricas durante a construção do protótipo.

A estrutura final, apresentada na Figura 16, mostra que o projeto possui rigidez mecânica adequada para suportar os quatro motores DC e o pacote de baterias sem apresentar flexões ou vibrações excessivas durante a operação. O peso total do protótipo esteve dentro da margem prevista para o torque oferecido pelos motores TT, garantindo uma mobilidade fluida tanto em superfícies lisas quanto em carpetes ou superfícies mais irregulares.

Figura 16 – Resultado Final

Fonte: Elaborada pelo autor.

Um dos principais resultados da etapa de *hardware* foi a organização interna dos componentes. Conforme detalhado na Figura 12, a disposição estratégica dos módulos permitiu:

- **Estabilidade Dinâmica:** O posicionamento das células de *Li-Ion* (componente de maior densidade de massa) no eixo central inferior resultou em um centro de gravidade baixo, prevenindo tombamentos em paradas bruscas ou acelerações rápidas.
- **Manutenibilidade:** A fixação do ESP32 sobre uma mini *protoboard*, em vez de solda direta, provou ser uma decisão acertada, facilitando o acesso aos pinos e também atualizações de *firmware* via USB. Além de medições de tensão durante os testes.
- **Segurança Elétrica:** O isolamento adequado das conexões de potência e a fixação firme dos cabos evitaram interferências mecânicas com as partes móveis e curtos-circuitos acidentais.

5.2 Análise Econômica e Viabilidade de Custo

Um dos pilares centrais deste trabalho é a validação da viabilidade econômica da plataforma robótica, em contraste aos elevados custos de aquisição de *kits* proprietários. Para atingir esse objetivo, foi realizado o levantamento detalhado de todos os custos envolvidos na aquisição de componentes eletrônicos, mecânicos e insumos de montagem utilizados na versão final do protótipo.

A Tabela 3 detalha a lista de materiais, indicando a quantidade, a origem da aquisição e o custo efetivo pago.

Tabela 3 – Detalhamento de custos dos componentes utilizados no protótipo

Componente	**Qtd.**	**Origem**	**Preço (R$)**
ESP32 DOIT DevKit V1	1	Nacional	32,88
Chassi 4WD com 4 motores DC	1	Nacional	104,75
Sensores Ultrassônicos HC-SR04	4	Nacional	55,98
Fios *jumper* e terminais	1	Nacional	28,90
Ponte H L298N	1	Nacional	21,01
Baterias *Li-Ion* 18650 3.7V	2	Nacional	16,50
Conversor *Step-Down* LM2596	1	Nacional	19,00
Botão Liga/Desliga	1	Nacional	8,10
Conector P4 Fêmea	1	Nacional	8,00
Módulo BMS 2S 5A	1	Nacional	30,09
Protoboard pequena	1	Nacional	3,90
Fusível 1A	1	Nacional	0,40
Custo Total do Protótipo	-	-	**329,51**

Fonte: Elaborada pelo autor com base nos dados de aquisição.
Nota: Os valores referem-se aos preços praticados no mercado nacional no período de desenvolvimento do projeto.

O custo total de construção do veículo robótico foi de **R$ 329,51**. A estratégia de

aquisição priorizou fornecedores nacionais para todos os componentes, o que, embora possa apresentar um custo unitário ligeiramente superior à importação direta da China, garantiu agilidade no desenvolvimento, garantia sobre os componentes e facilidade de reposição imediata em caso de falhas.

5.2.1 Comparativo com Soluções Comerciais

Para contextualizar a economia gerada, é interessante comparar o custo deste projeto com plataformas comerciais equivalentes frequentemente utilizadas em ambientes acadêmicos, como o *kit LEGO Mindstorms EV3* ou soluções baseadas em *VEX Robotics*.

Considerando que um *kit* básico de robótica proprietário com capacidades similares (controlador programável, sensores de distância e motores) possui um custo médio de mercado no Brasil superior a R$ 3.500,00, o protótipo desenvolvido neste trabalho apresenta um custo mais de 10 vezes menor.

Esta redução drástica de custos, superior a 90%, valida a hipótese inicial do trabalho, demonstrando que a utilização de *hardware* aberto e componentes genéricos permite a democratização do acesso a ferramentas avançadas de robótica e IoT. Diferente dos *kits* fechados, a solução baseada no ESP32 não apenas reduz a barreira de entrada financeira, mas também oferece maior flexibilidade de manutenção e expansão, uma vez que a reposição de uma peça danificada (como um sensor ou motor) é trivial e de baixo custo se comparada à reposição de componentes proprietários.

5.3 Validação Funcional e Desempenho Operacional

Os testes ao final, integraram as funcionalidades de controle e autonomia para mensurar e validar o comportamento dinâmico do veículo em condições reais de operação. A avaliação foi feita tendo em vista a latência da comunicação sem fio, a precisão da lógica de desvio de obstáculos e a estabilidade do sistema de alimentação.

Em relação ao controle remoto, a interface *web* desenvolvida (apresentada anteriormente na Figura 15) demonstrou alta responsividade. O servidor embarcado no ESP32 gerenciou as requisições HTTP de forma estável, resultando em uma latência perceptível mínima entre o comando do usuário na tela e o acionamento dos motores. Testes de alcance realizados em ambiente interno confirmaram a manutenção da conexão Wi-Fi estável em distâncias superiores

a 10 metros, validando a arquitetura IoT para teleoperação em redes locais.

Quanto à navegação autônoma, o algoritmo de segurança se mostrou preciso. Ao detectar obstáculos frontais dentro do limite programado de 50 cm, o sistema foi capaz de interromper a tração instantaneamente e executar a rotina de decisão, utilizando os sensores laterais para redirecionar o veículo para áreas livres. O comportamento do robô provou que a leitura sequencial dos quatro sensores ultrassônicos fornece dados suficientes para uma navegação reativa básica, evitando colisões frontais e laterais em ambientes controlados.

Por fim, o desempenho energético do protótipo conseguiu superar as expectativas iniciais. O sistema de alimentação baseado nas células de *Li-Ion* 18650 garantiu uma autonomia de operação contínua superior a 40 minutos. O regulador de tensão LM2596 manteve a estabilidade da linha de 5V para o microcontrolador mesmo durante os picos de corrente exigidos pelos motores nas partidas e reversões, sem apresentar aquecimento excessivo, comprovando o dimensionamento adequado da eletrônica de potência.

6 CONCLUSÕES E DISCUSSÕES

Este trabalho teve como objetivo projetar e implementar uma plataforma robótica móvel de tração diferencial (4WD) de baixo custo, fundamentada na utilização do microcontrolador ESP32 para aplicações de controle remoto via IoT e navegação autônoma. A metodologia percorreu o ciclo completo de engenharia, desde a modelagem esquemática do *hardware* utilizando o software KiCad até a manufatura e integração física dos componentes eletrônicos no chassi. O desenvolvimento do *firmware* abrangeu a implementação de um servidor *web* embarcado para teleoperação e a codificação de algoritmos de leitura de sensores ultrassônicos para a tomada de decisão em tempo real, visando validar a capacidade de processamento multitarefa do sistema escolhido.

Os resultados obtidos nos ensaios práticos demonstraram a viabilidade técnica e econômica da solução proposta. O principal êxito constatado foi a robustez da arquitetura baseada no ESP32, que gerenciou simultaneamente a pilha de comunicação Wi-Fi e o controle dos motores sem latência perceptível, garantindo uma resposta imediata aos comandos da interface *web*. O algoritmo de autonomia mostrou-se eficaz na detecção de obstáculos dentro do limite de segurança de 50 cm, executando manobras de desvio com precisão. Do ponto de vista econômico, a análise de custos revelou que o protótipo, orçado em aproximadamente R$ 329,51, representa uma alternativa acessível frente aos *kits* proprietários, oferecendo funcionalidades equivalentes com uma redução de custo superior a 90%, o que valida a hipótese de democratização do acesso à tecnologia proposta inicialmente.

Por fim, este trabalho superou desafios relacionados à gestão de energia e estabilidade mecânica através da otimização do posicionamento das baterias e do uso de reguladores de tensão dedicados, resultando em um sistema confiável e com autonomia energética satisfatória. A plataforma desenvolvida contribui para a área de robótica móvel ao fornecer uma base de *hardware* e *software* aberta e modular, apta a ser replicada ou expandida em trabalhos futuros. O projeto deixa como legado um protótipo funcional que pode ser aprimorado com a integração de sensores mais complexos, como câmeras para visão computacional ou algoritmos de mapeamento, servindo como ferramenta de pesquisa e aprendizado prático.

REFERÊNCIAS

AHMED, A. S.; MARZOG, H. A.; ABDUL-RAHAIM, L. A. Design and implement of robotic arm and control of moving via iot with arduino esp32. **International Journal of Electrical & Computer Engineering (2088-8708)**, v. 11, n. 5, 2021.

ALAMSYAH, S. A.; WIDAYAKA, P. D.; PUSPITANINGAYU, P.; IKHSAN, T. H.; PRADANA, R. A.; PALMA, R. R. H.; ACHMAD, I. Z. Design of omniwheel kinematics learning platform using esp32 and microsoft visual studio. **Best: Journal of Applied Electrical, Science and Technology**, v. 7, n. 2, p. 105–113, 2025.

ALMASRI, M. M.; ALAJLAN, A. M.; ELLEITHY, K. M. Trajectory planning and collision avoidance algorithm for mobile robotics system. **IEEE Sensors journal**, IEEE, v. 16, n. 12, p. 5021–5028, 2016.

ATZORI, L.; IERA, A.; MORABITO, G. The internet of things: A survey. **Computer networks**, Elsevier, v. 54, n. 15, p. 2787–2805, 2010.

BARR, M. **Programming embedded systems in C and C++**. Sebastopol, CA: O'Reilly Media, Inc., 1999.

CAÑADAS-ARÁNEGA, F.; MORENO, J. C.; BLANCO-CLARACO, J. L.; GIMÉNEZ, A.; RODRÍGUEZ, F.; SÁNCHEZ-HERMOSILLA, J. Autonomous collaborative mobile robot for greenhouses: Design, development, and validation tests. **Smart Agricultural Technology**, Elsevier, v. 9, p. 100606, 2024.

DANDL, K.; TÓTH, K.; BITZENBAUER, P. From simulation to experiment: Using kicad to design electric circuits in the physics classroom. **Eurasia Journal of Mathematics, Science and Technology Education**, Modestum, v. 20, n. 10, p. em2510, 2024.

DUDEK, G.; JENKIN, M. **Computational principles of mobile robotics**. Cambridge: Cambridge university press, 2024.

ElecFreaks. **Ultrasonic Ranging Module HC-SR04**. [*S. l.*], 2011. Datasheet. Disponível em: https://www.elecfreaks.com. Acesso em: 10 dez. 2025.

ESPRESSIF SYSTEMS. **ESP32 Series Datasheet**. [*S. l.*], 2024. Disponível em: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf. Acesso em: 05 dez. 2025.

GOLCONDA, S. **Steering control for a skid-steered autonomous ground vehicle at varying speed**. Dissertação (Mestrado) – University of Louisiana at Lafayette, Lafayette, LA, 2005.

HAMICI, N. **Design and Implementation of an Autonomous Mobile Robot Based on ESP-32**. Dissertação (Mestrado) – University of Guelma, Guelma, Algeria, 2023.

HONDA, C. K.; ARRUDA, M. C. A.; MATTOS, E. P. de; FARIA, G. Projeto e desenvolvimento de um robô móvel de baixo custo guiado por radiofrequência. **Anais do Computer on the Beach**, v. 4, p. 118–127, 2013.

HOUARI, B.; AISSA, B.; LAKHDAR, L.; YAZID, B. E. Dc/dc buck converter prototype for educational nanosatellite power sub-system. In: INTERNATIONAL CONFERENCE ON ADVANCES IN ELECTRONICS, CONTROL AND COMMUNICATION SYSTEMS (ICAECCS). [*S. l.*], 2023. p. 1–6.

HUGHES, A.; DRURY, B. **Electric motors and drives**: fundamentals, types and applications. Oxford: Newnes, 2019.

KART, D. S. B. **Pulse width modulation**. Palkulam, India: Rohini College of Engineering & Technology, 2001.

KUMAR, R.; PRASAD, U.; VISHWAKARMA, K. A study and an analysis of battery management system for electric vehicle. **World Journal of Engineering and Technology**, Scientific Research Publishing, v. 7, 2019.

KURNIAWAN, A. **Internet of Things Projects with ESP32**: Build exciting and powerful iot projects using the all-new espressif esp32. [*S. l.*]: Packt Publishing Ltd, 2019.

KUROSE, J. F.; ROSS, K. W. **Computer networking**: A top-down approach. [*S. l.*]: Pearson Harlow, England Boston, 2019.

MUENZEL, V.; HOLLENKAMP, A. F.; BHATT, A. I.; HOOG, J. de; BRAZIL, M.; THOMAS, D. A.; MAREELS, I. A comparative testing study of commercial 18650-format lithium-ion battery cells. **Journal of The Electrochemical Society**, IOP Publishing, v. 162, n. 8, p. A1592, 2015.

NARAYANA, S. P.; SANDEEP, S.; PATEL, S.; YOGESH, M.; NATH, K. B. Web-server controlled rover with robotic arm and object detection. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL SYSTEM AND INFORMATION TECHNOLOGY FOR SUSTAINABLE SOLUTIONS (CSITSS), 8. [*S. l.*], 2024.

OLIVEIRA, A. A. de; FRANCO, J. P. E. de L.; SANTOS, N. O. D.; ALVES, W. V. C.; SILVA, V. L. da. Desenvolvimento de robô móvel com esp32 para olimpíadas de robótica. In: CONGRESSO DE INOVAÇÃO, CIÊNCIA E TECNOLOGIA DO IFSP. [*S. l.*]: [*s. n.*], 2024. v. 15, n. 3.

OLTEAN, S.-E. Mobile robot platform with arduino uno and raspberry pi for autonomous navigation. **Procedia Manufacturing**, Elsevier, v. 32, p. 572–577, 2019.

RAY, P. P. Internet of robotic things: concept, technologies, and challenges. **IEEE access**, IEEE, v. 4, p. 9489–9500, 2017.

SIEGWART, R.; NOURBAKHSH, I. R.; SCARAMUZZA, D. **Introduction to autonomous mobile robots**. Cambridge, MA: MIT press, 2011.

STMicroelectronics. **L298 - Dual Full-Bridge Driver**. [*S. l.*], 2000. Datasheet. Disponível em: http://www.st.com. Acesso em: 10 dez. 2025.

VENKATESH, S.; KUNDAN, S.; SRIJA, A. Obstacle avoidance robotic vehicle using hc-sr04 ultrasonic sensor. **Turkish Journal of Computer and Mathematics Education**, Ninety Nine Publication, v. 12, n. 12, p. 2358–2362, 2021.

WANG, R.; CHEN, L.; WANG, J.; ZHANG, P.; TAN, Q.; PAN, D. Research on autonomous navigation of mobile robot based on multi ultrasonic sensor fusion. **2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)**, p. 720–725, 2018. Disponível em: https://api.semanticscholar.org/CorpusID:195223783. Acesso em: 05 dez. 2025.