

Rafael Dias Campos

Jogo de Xadrez com Manipuladores Robóticos

Belo Horizonte

2023

Rafael Dias Campos

Jogo de Xadrez com Manipuladores Robóticos

Trabalho de Conclusão de Curso apresentado
ao Curso de Engenharia de Computação do
Centro Federal de Educação Tecnológica de
Minas Gerais, como requisito parcial para a
obtenção do título de Bacharel em Engenharia
de Computação.

Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG

Departamento de Computação

Curso de Engenharia da Computação

Orientador: Ramon da Cunha Lopes

Belo Horizonte

2023

Centro Federal de Educação Tecnológica de Minas Gerais
Curso de Engenharia de Computação
Avaliação do Trabalho de Conclusão de Curso

Aluno: Rafael Dias Campos

Título do trabalho: Jogo de Xadrez com Manipuladores Robóticos

Data da defesa: 30/06/2023

Horário: 14:00

Local da defesa: CEFET-MG Campus II à Avenida Amazonas 7675. Prédio 17 (DECOM), sala 401.

O presente Trabalho de Conclusão de Curso foi avaliado pela seguinte banca:

Professor Ramon da Cunha Lopes - Orientador
Departamento de Computação
Centro Federal de Educação Tecnológica de Minas Gerais

Professora Mara Cristina da Silveira Coelho - Membro da banca de avaliação
Departamento de Computação
Centro Federal de Educação Tecnológica de Minas Gerais

Professor Rogério Martins Gomes - Membro da banca de avaliação
Departamento de Computação
Centro Federal de Educação Tecnológica de Minas Gerais

Professor Tales Argolo Jesus - Membro da banca de avaliação
Departamento de Computação
Centro Federal de Educação Tecnológica de Minas Gerais

Resumo

Atualmente, existe uma grande procura por funcionários especializados em Tecnologia da Informação (TI) e áreas similares, sendo percebida no mundo todo uma carência de profissionais qualificados. Esse problema ocorre parcialmente devido a um desinteresse à capacitação técnica por parte de crianças e jovens, frequentemente devido a uma concepção de que essas áreas são difíceis. Com o foco neste aspecto, esse projeto visa desenvolver um sistema que implementa o jogo de Xadrez utilizando manipuladores robóticos. Este sistema pode ser apresentado para crianças e jovens em feiras educativas e eventos similares com o objetivo de introduzir conceitos básicos e instigar o interesse pelas áreas de computação, elétrica e controle. Sistemas já existentes para mover peças de xadrez com braços mecânicos geralmente apresentam um custo elevado, pois priorizam a velocidade e precisão nos movimentos para que uma máquina jogue contra um jogador humano em campeonatos. Por isso, eles não são muito adequados para usos educacionais e têm-se a necessidade de desenvolver um sistema mais barato e com foco na simplicidade e na capacidade de proporcionar divertimento para os jogadores.

Palavras-chave: Manipuladores Robóticos. Controle Digital. Xadrez.

Abstract

Currently, there is a great demand for specialized professionals in IT and in similar areas, and it is perceived a worldwide shortage of qualified professionals. This problem occurs partially due to a lack of interest in technical training on the part of children and young adults, often due to a conception that these areas are difficult. Focusing on this aspect, this project aims to develop a low-cost system that implements the game of chess using robotic arms. This system can be presented to children and young adults in educational fairs and similar events with the aim of instigating interest in the areas of computing, electrical engineering and control systems engineering. Existing systems to move chess pieces with robotic arms usually have a high cost, as they prioritize speed and precision in movements so that a machine can play against a human player in tournaments. Therefore, they are not very suitable for educational uses and there is a need to develop a cheaper system and with a focus on simplicity and the ability to provide fun for players.

Keywords: Robotic Arms. Digital Control. Chess.

Lista de ilustrações

Figura 1 – Manipulador robótico Mentor	14
Figura 2 – Manipulador robótico RD5NT	14
Figura 3 – Manete para o jogador	15
Figura 4 – Largura do pulso do sinal de controle	16
Figura 5 – Microcontrolador ESP32	17
Figura 6 – Módulo de ponte H	18
Figura 7 – Montagem do sistema	19
Figura 8 – Esquemático simplificado de um CI L293D e um CI 74LS02	20
Figura 9 – Layout da placa de controle	22
Figura 10 – Placa de controle produzida	22
Figura 11 – Movimento das juntas do manipulador	25
Figura 12 – Cálculo do ângulo do torso	26
Figura 13 – Cálculo do ângulo do pulso	27
Figura 14 – Modelo simplificado do manipulador	28

Lista de tabelas

Tabela 1 – Características do manipulador robótico Mentor	14
Tabela 2 – Características do manipulador robótico RD5NT	14
Tabela 3 – Constantes do controlador PID	24
Tabela 4 – Comandos enviados pelo computador para o microcontrolador	29
Tabela 5 – Mensagens enviadas pelo microcontrolador para o computador	29
Tabela 6 – Tempo de execução de cada movimento	37

Lista de abreviaturas e siglas

ADC	<i>Analog to Digital Converter</i> [Conversor Analógico Digital]
CI	Círculo Integrado
CEFET-MG	Centro Federal de Educação Tecnológica de Minas Gerais
STEM	<i>Science, Technology, Engineering and Mathematics</i> [Ciência, Tecnologia, Engenharia e Matemática]
PID	<i>Proportional Integral Derivative</i> [Proporcional Integral Derivativo]
PWM	<i>Pulse Width Modulation</i> [Modulação de Largura de Pulso]

Sumário

1	INTRODUÇÃO	10
2	TRABALHOS RELACIONADOS	11
3	METODOLOGIA	12
4	PLANEJAMENTO	13
4.1	Escolha do equipamento	13
4.1.1	Manipuladores	13
4.1.2	Manete para o jogador	15
4.1.3	Microcontrolador	15
4.1.4	Placa de Controle	17
4.1.5	Computador	18
4.2	Projeto do sistema	18
5	DESENVOLVIMENTO	20
5.1	Desenvolvimento da placa de controle	20
5.2	Leitura da manete	22
5.3	Leitura da posição das juntas	23
5.4	Controle dos motores	23
5.5	Cálculo dos ângulos desejados	24
5.6	Movimentação de peças	28
5.7	Comunicação com o computador	29
5.8	Implementação da lógica de xadrez	30
6	CONCLUSÃO	32
6.1	Performance	32
6.2	Limitações	32
	REFERÊNCIAS	33
	APÊNDICES	34
	APÊNDICE A – ESQUEMÁTICO DA PLACA DE CONTROLE	36
	APÊNDICE B – PERFORMANCE DO SISTEMA	37

1 Introdução

Atualmente, existe uma grande procura por funcionários especializados em Tecnologia da Informação (TI) e áreas similares, sendo percebida no mundo todo uma grande carência de profissionais qualificados para atuar nessas áreas. Apesar do grande crescimento nas áreas de STEM (Science, Technology, Engineering and Mathematics) [Ciência, Tecnologia, Engenharia e Matemática], o número de profissionais qualificados nessas áreas não acompanha esse crescimento e a perspectiva é de que essa situação se acentue ainda mais no futuro (LEPRINCE-RINGUET, 2021).

Devido a essa carência de profissionais, torna-se importante a busca por formas de incentivar o aprendizado e a busca por conhecimento por parte dos jovens. Visando solucionar esse problema, foi decidido realizar um trabalho que incorpore conceitos de robótica, já que seu uso em atividades com crianças consegue influenciar positivamente o desenvolvimento de habilidades da área de STEM (DOROUKA; PAPADAKIS; KALOGIANNAKIS, 2020).

Com base nisso, foi proposto realizar o desenvolvimento de uma plataforma que utilize recursos computacionais passível de ser utilizada para demonstrar conceitos nas áreas de computação, elétrica e controle. Para aumentar o interesse por esta plataforma foi decidido incorporar um jogo de tabuleiro no projeto e foi escolhido o Xadrez, por ser um jogo que exige raciocínio lógico e estratégico, além de ser um jogo popular e bastante conhecido. Correlacionando ambas essas ideias, implementou-se um jogo de Xadrez que pode ser jogado através de braços robóticos.

2 Trabalhos Relacionados

Durante a pesquisa realizada para o desenvolvimento deste trabalho, foram encontrados alguns trabalhos que apresentam características semelhantes ao que foi proposto. Neste capítulo, serão apresentados alguns desses trabalhos, com o objetivo de mostrar o que foi feito e como foi feito, além de mostrar as diferenças entre eles e o trabalho proposto.

O trabalho apresentado por MORAES (2021) apresenta um sistema para jogar xadrez contra um computador utilizando um tabuleiro físico. Neste trabalho, foi desenvolvido um tabuleiro modificado que utiliza motores de passo e um eletroímã para movimentar as peças. Além disso, foi implementado um sistema de visão computacional para detectar qual peça foi movimentada pelo jogador.

Em outra frente, o trabalho de CARDONA et al. (2012) apresenta uma ferramenta para manipular um braço robótico através de um computador. Essa ferramenta recebe os ângulos desejados de cada junta do braço e realiza o movimento, através de cinemática direta. Através dessa ferramenta, é possível movimentar o braço robótico com base nos ângulos, entretanto não é possível diretamente definir sua posição final.

OLUNLOYO1 et al. (2014) apresenta um estudo sobre cinemática inversa de um braço robótico de cinco juntas. Neste trabalho, é apresentado um método para calcular os ângulos de cada junta, com base na posição final desejada. A partir desse método, é possível movimentar o braço robótico com mais facilidade, pois não é necessário calcular os ângulos manualmente.

A partir do estudo desses trabalhos, foi possível definir o escopo do trabalho proposto e observar as diferenças em relação aos trabalhos relacionados. O trabalho proposto implementa o jogo de xadrez, assim como o trabalho de MORAES (2021), porém utiliza um braço robótico para movimentar as peças. Para isso, foi necessário desenvolver uma ferramenta que permite a comunicação entre um computador e o braço robótico, assim como o trabalho de CARDONA et al. (2012). Por sua vez, controle do braço é feito através de um sistema de cinemática inversa, assim como o trabalho de OLUNLOYO1 et al. (2014), porém com um menor número de juntas.

3 Metodologia

O desenvolvimento do projeto foi dividido em duas etapas: uma de planejamento e outra de execução.

Durante a etapa de planejamento, foram definidos os componentes necessários para a construção do projeto, bem como a forma de integração entre eles. Além disso, foi feito um estudo sobre o controle de um braço robótico e sobre as capacidades do microcontrolador escolhido para o projeto.

Na etapa de execução, foi feita primeiramente a montagem dos componentes físicos do projeto. Em seguida, foi desenvolvido o *software* do microcontrolador, que é responsável por controlar os motores e receber os comandos do usuário. Por fim, foi desenvolvido o *software* do computador para implementar as regras do jogo de xadrez e comunicar com o microcontrolador.

4 Planejamento

Antes de iniciar o desenvolvimento do projeto, foram especificados os recursos necessários para a montagem do sistema e sua utilização. Esta seção descreve a escolha do equipamento e o projeto inicial do sistema.

4.1 Escolha do equipamento

Para o desenvolvimento do projeto, foram disponibilizados, pela instituição CEFET-MG, dois manipuladores robóticos e diversos dispositivos que podem ser utilizados para seu controle. A partir desse equipamento, e de outros disponíveis no mercado, foi decidido como o projeto seria realizado. Os equipamentos utilizados para o projeto estão descritos a seguir:

4.1.1 Manipuladores

Os principais elementos deste trabalho são os manipuladores robóticos, portanto foi feito inicialmente um estudo sobre seu funcionamento e sobre como seu controle pode ser realizado para movimentar as peças de xadrez.

Foi disponibilizado um manipulador robótico de modelo Mentor de cor preta, conforme a figura 1, e um manipulador de modelo RD5NT de cor azul, conforme a figura 2. Eles possuem diversas juntas movimentadas por motores de corrente contínua, e permitem que o manipulador funcione de forma similar a um braço humano. Além disso, cada junta possui um potenciômetro que indica a posição atual do eixo, por meio de um sinal analógico.

O manipulador Mentor apresenta 5 graus de liberdade e uma garra que pode ser utilizada para pegar e soltar objetos. Além disso, ele possui caixas de redução em seus motores, o que permite que ele mantenha sua posição mesmo após o desligamento dos motores. Suas dimensões e faixas de movimento são apresentadas na tabela 1 (OLUNLOYO; AYOMOH; ADEOTI, 2011).

Já o manipulador RD5NT possui apenas 4 graus de liberdade e uma garra. Além de não possuir caixa de redução em seus motores, ele utiliza molas em alguns eixos, o que faz com que perca sua posição quando os motores são desligados. Portanto, seu controle deve ser realizado de forma contínua para que permaneça na posição desejada. Suas dimensões e faixas de movimento são apresentadas na tabela 2 (RIUL; MONTENEGRO; FERREIRA, 2018).

Tabela 1 – Características do manipulador robótico Mentor

Eixo	Movimento angular (graus)	Comprimento (mm)
Eixo 0 (Torso)	210	185
Eixo 1 (Ombro)	180	165
Eixo 2 (Cotovelo)	230	150
Eixo 3 (Esquerdo do Pulso)	320	0
Eixo 4 (Direito do Pulso)	320	0
Pulso <i>Pitch</i>	140	-
Pulso <i>Roll</i>	320	-

Tabela 2 – Características do manipulador robótico RD5NT

Eixo	Movimento angular (graus)	Comprimento (mm)
Eixo 0 (Torso)	293	110
Eixo 1 (Ombro)	107	120
Eixo 2 (Cotovelo)	284	160
Eixo 3 (Pulso)	360	0

Figura 1 – Manipulador robótico Mentor



Fonte: <http://arquivo.eng.br/robotica>

Figura 2 – Manipulador robótico RD5NT



Fonte: <http://arquivo.eng.br/robotica>

Foi decidido utilizar o manipulador de modelo RD5NT para o projeto, pois ele possui menos graus de liberdade na sua garra, o que facilita a implementação do algoritmo de controle. Além disso, ele possui uma garra de tamanho adequado para pegar as peças de xadrez sem derrubar as outras peças do tabuleiro. Apesar de não possuir caixa de redução em seus motores, isso não é um problema, pois o controle do manipulador será realizado de forma contínua.

4.1.2 Manete para o jogador

Para que o jogador possa interagir com o manipulador, foi decidido utilizar uma manete de modelo *batpad*, que possui dois *joysticks*, conforme a figura 3.

Esses *joysticks* devem ser alimentados com 3.3V ou 5V e permitem a leitura de posição em duas dimensões, através de sinais analógicos. Eles também possuem um botão que envia um sinal digital ao ser pressionado.

Figura 3 – Manete para o jogador



Fonte: <https://www.robocore.net/acessorios-robocore/controle-batpad>

4.1.3 Microcontrolador

Para realizar a integração entre a manete e o manipulador, foi decidido utilizar um microcontrolador ESP32, conforme a figura 5. Esse microcontrolador possui um *clock* de 240MHz, 4MB de memória *flash* e 320KB de memória *RAM*. Ele apresenta ao todo 34 pinos que podem ser utilizados como entrada ou saída, e suporta sinais analógicos e digitais (ESPRESSIF, 2023).

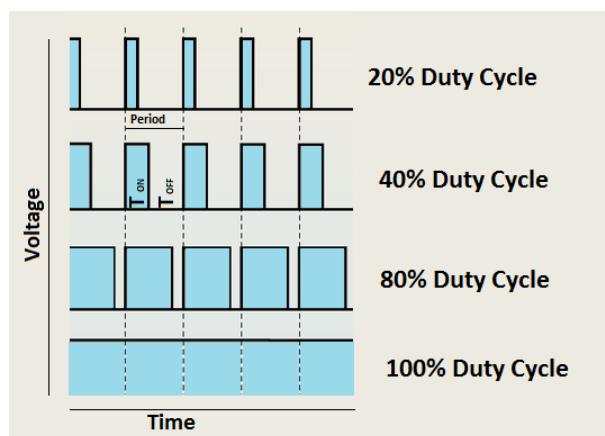
Para permitir o controle manual do manipulador robótico, o microcontrolador faz a leitura dos sinais analógicos e digitais provenientes da manete com alguns de seus 18 canais de *ADC* (*Analog to Digital Converter*) [Conversor Analógico Digital]. E para permitir o controle automático, ele recebe sinais de controle de um computador através de um cabo

USB. A partir desses sinais, o ESP32 realiza o cálculo das posições desejadas de cada junta do manipulador.

Para controlar os motores, o microcontrolador primeiramente faz a leitura dos sinais analógicos provenientes dos potenciômetros de cada junta. A partir desses sinais, é possível determinar a posição atual de cada junta. Com base na posição atual e na posição desejada, o microcontrolador envia um sinal de controle para os motores do manipulador robótico.

Os sinais de controle são digitais do tipo *PWM* (*Pulse Width Modulation*), que variam de 0V a 3.3V e possibilitam o controle da velocidade de rotação dos motores através da variação da tensão média. Valores maiores de *Duty Cycle* [Ciclo de Trabalho] resultam em maior velocidade de rotação do motor, pois o sinal permanece em nível alto por um período maior de tempo. Por outro lado, valores menores de *Duty Cycle* resultam em menor velocidade de rotação do motor, pois o sinal permanece em nível alto por um período menor de tempo. Para definir o sentido de rotação do motor, foi utilizado um sinal digital que indica se ele deve girar no sentido horário ou anti-horário.

Figura 4 – Largura do pulso do sinal de controle



Fonte: <https://create.arduino.cc/projecthub/muhammad-aqib/arduino-pwm-tutorial-ae9d71>

Figura 5 – Microcontrolador ESP32



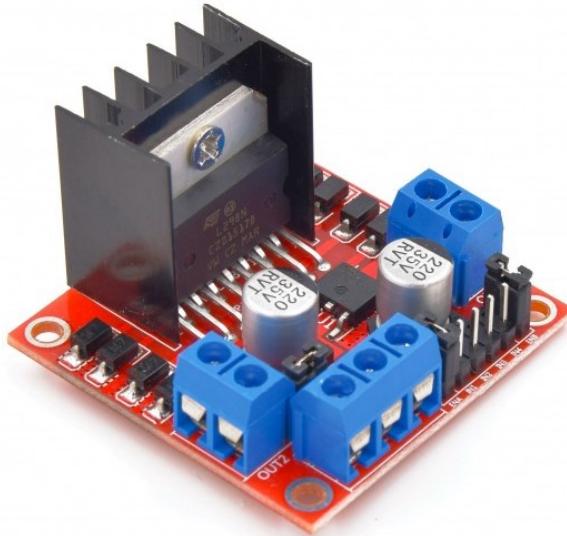
Fonte: <https://www.robobuilders.com.br/nodemcu-esp32-38-pinos-devkit>

4.1.4 Placa de Controle

Para controlar os motores do manipulador, é necessário o uso de uma placa de controle para converter os sinais de baixa potência provenientes do microcontrolador em sinais de maior potência que movimentam as juntas do manipulador robótico. Essa placa é alimentada com 12v e recebe os sinais digitais de direção e de *PWM* do ESP32 e movimenta os motores de acordo.

Para obter essa funcionalidade, foi decidido utilizar módulos de ponte H, que são circuitos integrados (CI) utilizados para aplicar uma tensão variável a um componente através de um sinal de *PWM*. Eles também permitem alterar a direção em que a corrente é aplicada no componente, o que possibilita inverter o sentido de rotação de um motor (DIGILENT, 2012). A figura 6 apresenta um módulo de ponte H disponível no mercado, que utiliza o CI L298N.

Figura 6 – Módulo de ponte H



Fonte: <https://www.smart-prototyping.com/L298N-Dual-H-bridge-Motor-Driver-Board>

4.1.5 Computador

Para gerenciar o jogo de xadrez, foi decidido utilizar um computador. Ele é responsável por implementar a lógica do jogo de xadrez e enviar sinais de controle para o ESP32, que por sua vez controla o manipulador robótico.

Para realizar a comunicação entre o computador e o ESP32, pode ser feito uso do protocolo serial, através de um cabo USB. Através dele o computador envia sinais indicando para qual casa do tabuleiro o manipulador deve mover, e quando ele deve pegar ou soltar uma peça. Por outro lado, o ESP32 envia sinais indicando quando o movimento foi finalizado e, no modo manual, qual movimento o jogador deseja realizar.

Para implementar as regras do jogo, é necessário desenvolver um software para interagir com uma *engine* de xadrez. Através dele, é possível verificar os movimentos válidos e identificar quando o jogo terminou. Além disso, ele também é responsável por converter um movimento em uma sequência de comandos que devem ser enviados ao ESP32.

4.2 Projeto do sistema

Após a definição de todos os equipamentos a serem utilizados, foi feito o projeto do sistema, indicando como os componentes devem ser conectados.

A manete deve ser alimentada com 3.3V e ser conectada ao ESP32 para permitir a leitura dos sinais de entrada. Essa conexão é realizada com 10 cabos, sendo 5 para cada

joystick com seu respectivo botão.

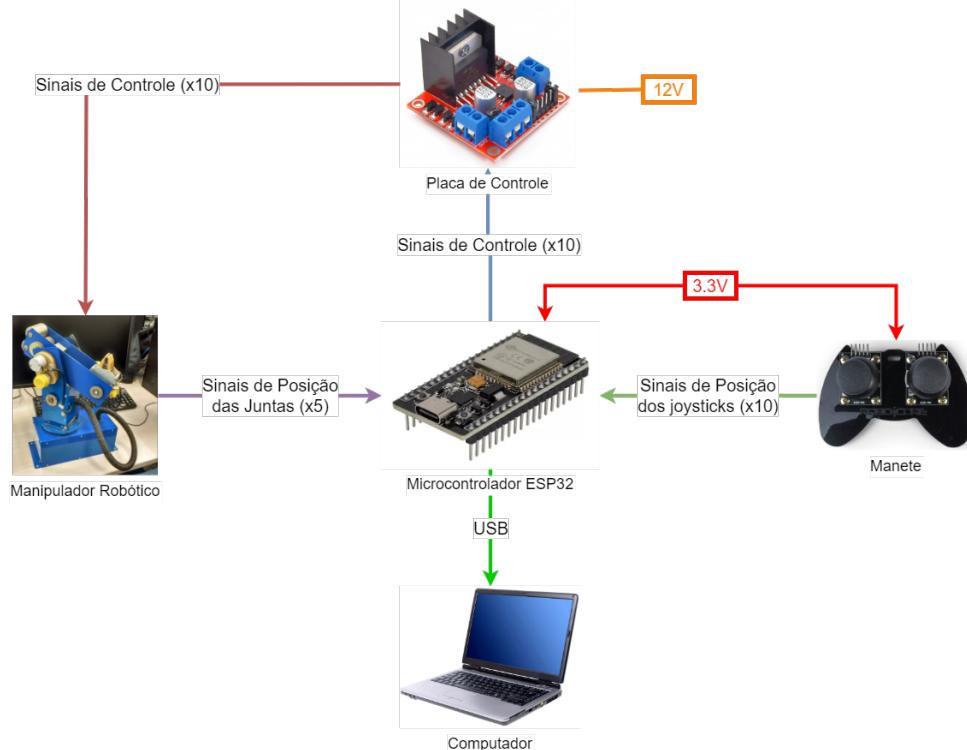
A placa de controle deve ser alimentada com 12V e também deve ser conectada ao ESP32 para receber os sinais de controle. Essa conexão é realizada com 10 cabos, 2 para o controle de cada junta do manipulador robótico. Essa placa também deve ser conectada aos motores do manipulador robótico, também com 10 cabos, 2 para cada junta.

O manipulador robótico deve ser conectado ao ESP32 para o envio dos sinais de posição de cada junta. Essa conexão é realizada com 5 cabos, um para cada junta.

Por fim, o ESP32 deve ser alimentado com 3.3V e deve ser conectado a um computador para a implementação da lógica do jogo.

A montagem do sistema é mostrada na Figura 7.

Figura 7 – Montagem do sistema



Fonte: Do próprio autor

5 Desenvolvimento

Após o planejamento do projeto, foi feito o desenvolvimento de cada etapa. Inicialmente, foi feito o desenvolvimento da placa de controle para poder acionar os motores. Em seguida, o *software* para o microcontrolador foi desenvolvido para realizar a leitura dos dados da manete e controlar o manipulador robótico. Por último, foi desenvolvido o *software* para o computador, que implementa a lógica do jogo de xadrez.

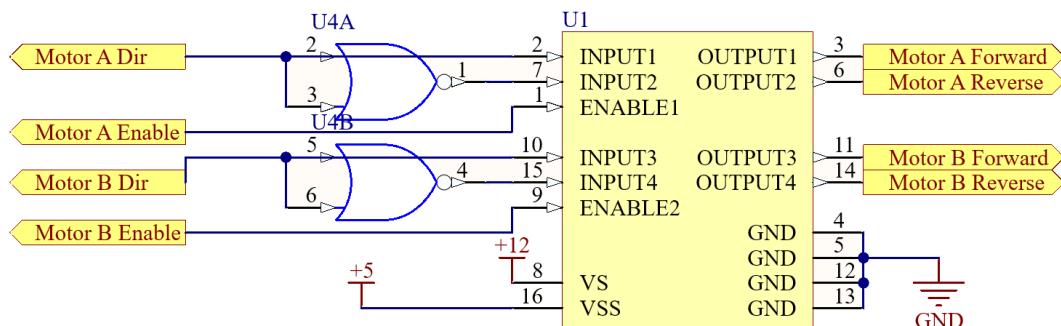
5.1 Desenvolvimento da placa de controle

Primeiramente, foi feito o desenvolvimento da placa de controle dos manipuladores, pois ela é necessária para as próximas etapas do projeto. Para isso, foi necessário definir quais componentes utilizar e como conectá-los.

Conforme descrito na subseção 4.1.4, a placa de controle deve utilizar uma ponte H para o controle de cada junta. Para isso, foi escolhido o CI L293D, que possui duas pontes H e suporta tensões de 12V. Como é necessário controlar 6 motores, foram utilizados 3 CI L293D.

Para simplificar o controle e evitar problemas de acionamento duplo das entradas das pontes H, foi utilizado o CI 74LS02 como um inversor lógico. Dessa forma, a placa de controle possui para cada junta uma entrada de *Enable* para ligar/desligar o acionamento da junta, e uma porta de *Direction* para definir a direção de movimentação dela. A partir dessas entradas, o CI L293D é acionado e o motor é controlado. A Figura 8 mostra o esquemático simplificado de um CI L293D e um CI 74LS02.

Figura 8 – Esquemático simplificado de um CI L293D e um CI 74LS02



Fonte: Do próprio autor

Com os componentes principais definidos foi feito o desenvolvimento do esquemático da placa de controle com o auxílio do software *Altium Designer*. O Apêndice A mostra o

esquemático completo da placa de controle. Nesse esquemático foram adicionados resistores de *pulldown* para garantir que as entradas dos CI L293D e 74LS02 permaneçam em nível lógico baixo caso não estejam conectadas ao microcontrolador. Também foram adicionadas *LEDs* para indicar a alimentação de 5V e 12V da placa.

Após o desenvolvimento do esquemático, foi feita a montagem da Placa de Circuito Impresso (PCB), ainda com o auxílio do software *Altium Designer*. Para isso, os componentes foram posicionados no *layout* da placa, tendo em vista a economia de espaço e a necessidade de manter os componentes próximos para facilitar sua conexão. Em seguida, as trilhas e vias que realizam a conexão dos componentes foram desenhadas. Para permitir a conexão de todos os componentes, foi necessário utilizar uma placa com 2 camadas. A Figura 9 mostra o *layout* final da placa de controle.

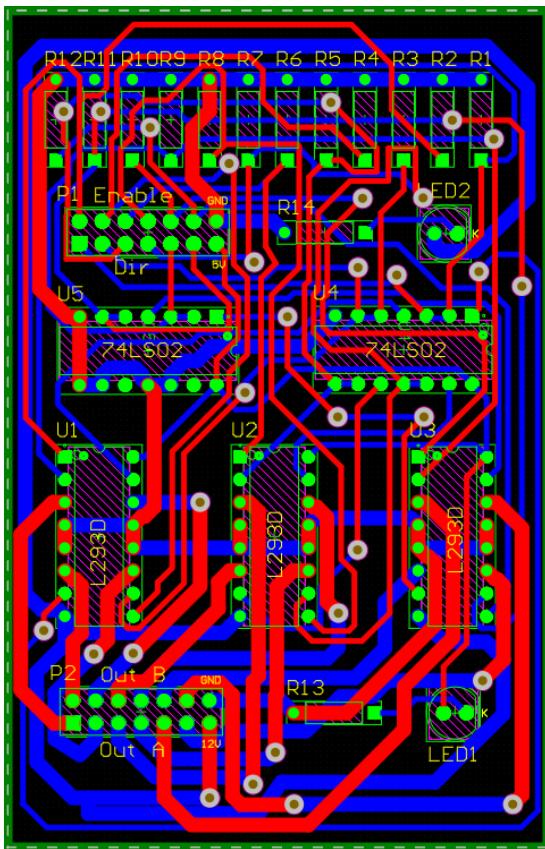
Com o *layout* finalizado, foi feita a produção da PCB de forma manual. Para isso, o negativo do *layout* foi impresso em uma folha de transparência. Depois, uma placa de circuito impresso de duas camadas foi cortada no tamanho desejado.

Em seguida, foi feita a transferência do *layout* para a placa. Para isso, uma fina camada de tinta fotossensível destinada para PCB foi aplicada sobre a placa. Essa tinta foi pré-curada à 75°C durante 15 minutos com o auxílio de uma base de aquecimento, para garantir que ela não se descolasse da placa. Após a pré-cura, a tinta foi exposta à luz ultravioleta por 3 minutos, utilizando a transparência com o *layout* como máscara. Em seguida, a placa foi submersa em uma solução de carbonato de sódio para realizar a revelação do *layout*. Após a placa ser revelada, a tinta foi curada à 85°C durante 30 minutos.

Esse processo de transferência do *layout* foi repetido para a segunda camada da placa. Em seguida, foi utilizada uma solução de percloreto de ferro para corroer as áreas de cobre que não receberam tinta. Após a corrosão, a placa foi mergulhada em uma solução de hidróxido de sódio para remover a tinta.

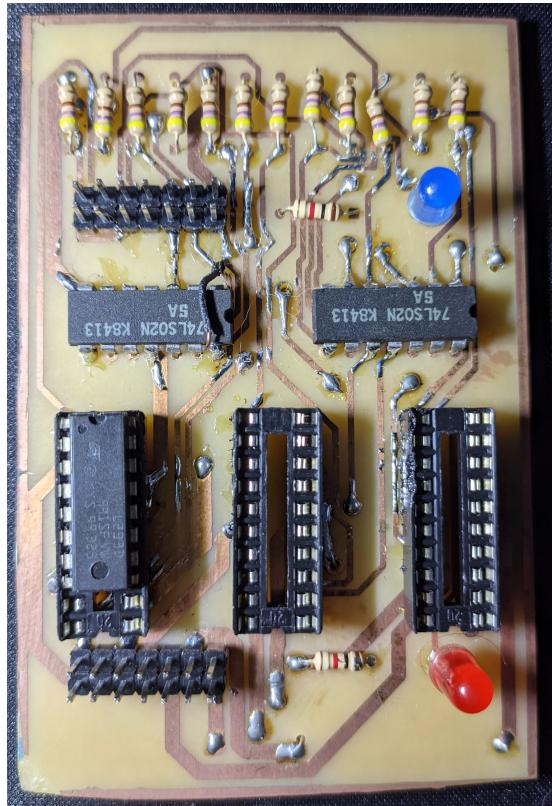
Após a corrosão da PCB, foi feita a perfuração das vias e dos furos para os componentes. Por fim, foi feita a soldagem dos componentes na placa e a conexão das vias. A Figura 10 mostra a placa de controle montada.

Figura 9 – Layout da placa de controle



Fonte: Do próprio autor

Figura 10 – Placa de controle produzida



Fonte: Do próprio autor

5.2 Leitura da manete

Após a montagem da placa de controle, foi iniciado o desenvolvimento do *software* para o microcontrolador, utilizando o *software PlatformIO*.

Inicialmente, foi implementada uma funcionalidade para realizar a leitura dos dados da manete que, conforme descrito na subseção 4.1.2, possui dois *joysticks* com dois eixos e um botão cada.

Para realizar a leitura dos eixos dos *joysticks*, foram utilizadas as entradas analógicas do microcontrolador. Como o ESP32 utiliza um conversor analógico-digital de 12 bits, os valores lidos variam de 0 a 4095. Valores próximos de 2048 representam a posição central do *joystick*, enquanto valores próximos de 0 ou 4095 representam as posições extremas. Para aprimorar a usabilidade da manete, foi implementada uma área de *deadzone*, na qual o valor lido é considerado como zero, para evitar que o manipulador se movimente sem a intenção do usuário.

Para realizar a leitura do botão, foi utilizada uma entrada digital. Esses botões possuem um *pull-up* interno, o que significa que o valor lido é 1 quando o botão não está

pressionado e 0 quando o botão está pressionado.

Os valores lidos são armazenados em uma variável, que é utilizada por outras funcionalidades do *software*.

5.3 Leitura da posição das juntas

Para ler a posição das juntas do manipulador robótico, foi utilizado como base o *software* desenvolvido anteriormente para ler os dados das manetes.

Primeiramente, foi utilizado um multímetro para medir o valor de tensão de saída do potenciômetro de cada junta em diferentes ângulos. A partir disso, foi observado que a tensão de saída altera de forma aproximadamente linear com o ângulo, e foi criada uma equação linear para cada junta, que relaciona a leitura realizada pelo microcontrolador com seu ângulo atual.

Com base nessas equações, foi implementada uma funcionalidade para calcular os valores que devem ser lidos pelo ESP32 quando o manipulador robótico estiver em uma determinada configuração.

5.4 Controle dos motores

O controle dos motores, responsável por mover o manipulador robótico para que ele atinja uma determinada configuração, foi incorporado no software desenvolvido na seção anterior.

Para isso, foi implementado um controlador digital PID no microcontrolador ESP32. Esse algoritmo realiza a leitura dos valores de tensão de cada junta e obtém o erro a partir da diferença entre o valor desejado e o valor atual. A partir desse erro, é calculada a integral dos erros até o momento e a derivada entre o erro atual e o último erro. Por fim, esses valores (erro atual, integral dos erros e derivada do erro) são multiplicados, respectivamente, por três constantes (K_p , K_i e K_d) e os resultados são somados.

Esse valor final é utilizado para definir os sinais de saída do microcontrolador para a placa de controle. O módulo desse valor de saída do PID define o Duty Cycle do sinal de PWM, enquanto o seu sinal (positivo ou negativo) define a direção de movimentação do motor.

Depois da implementação do controlador PID foram feitos diversos testes para encontrar valores adequados para as constantes K_p , K_i e K_d . Para isso, foi enviado um comando para movimentar uma junta para um determinado ângulo. Após a junta estabilizar na posição desejada, foi enviado um outro comando para movimentá-la para um ângulo diferente, distante do primeiro. Para alguns valores das constantes, foi observado

um comportamento muito lento do sistema, enquanto para outros valores foi observado um comportamento muito instável, com *overshoot* e oscilações. Depois de diversos testes, foram encontrados os valores apresentados na Tabela 3, que possibilitam um movimento relativamente rápido e estável do manipulador robótico.

Tabela 3 – Constantes do controlador PID

Junta	K_p	K_i	K_d
Torso	400	10	5
Ombro	500	40	0
Cotovelo	500	30	0
Pulso	100	2	0.1

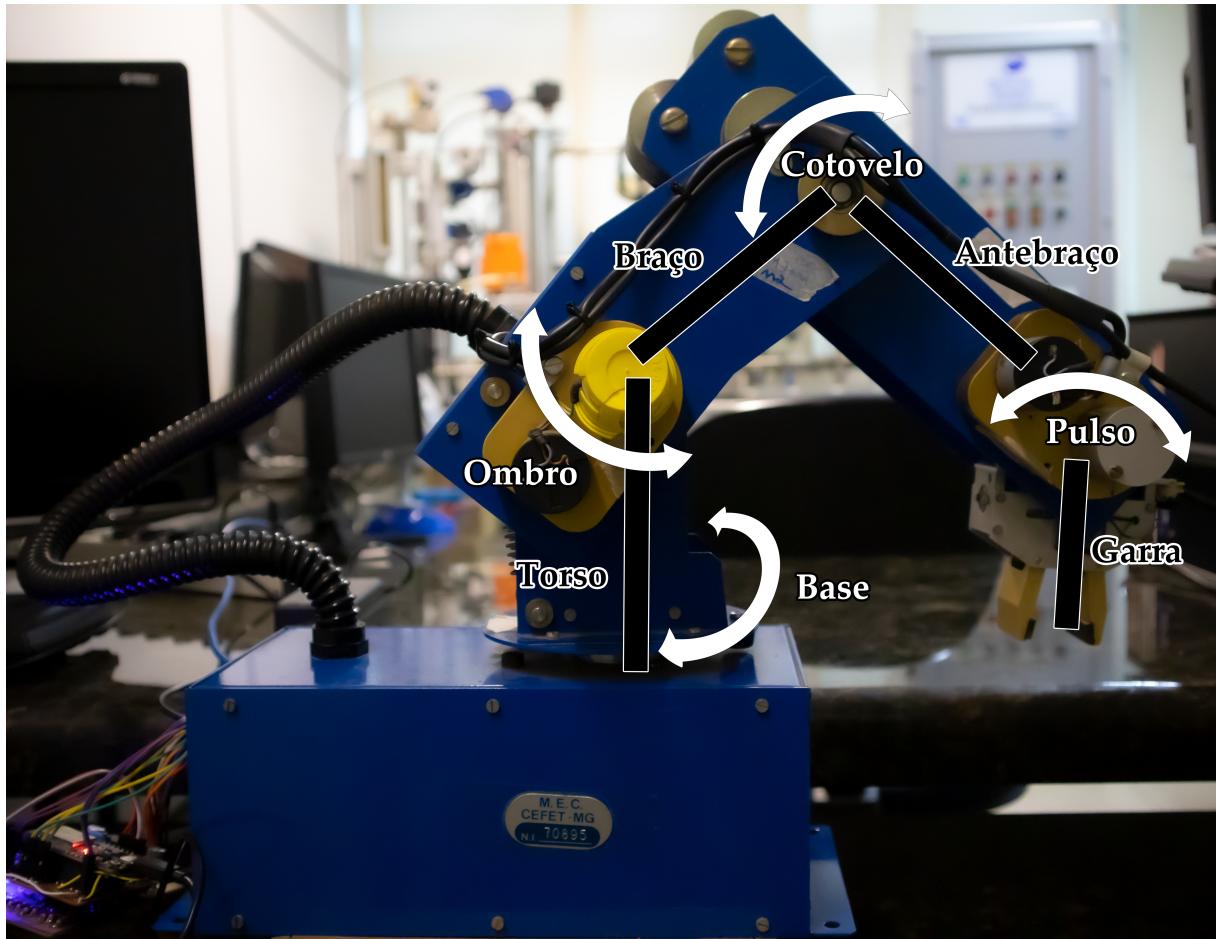
Fonte: Do próprio autor

5.5 Cálculo dos ângulos desejados

Após implementar o controle do manipulador a partir dos ângulos desejados para cada junta, foi necessário desenvolver o código que realiza o cálculo da configuração necessária para que o manipulador robótico alcance uma determinada posição no espaço.

Primeiramente, foi feita uma análise de como cada junta do manipulador robótico se movimenta. Como pode ser observado na Figura 11, o torso se movimenta horizontalmente, perpendicularmente ao eixo Z. Por outro lado, o ombro, cotovelo e pulso se movimentam verticalmente, paralelamente à base do manipulador.

Figura 11 – Movimento das juntas do manipulador

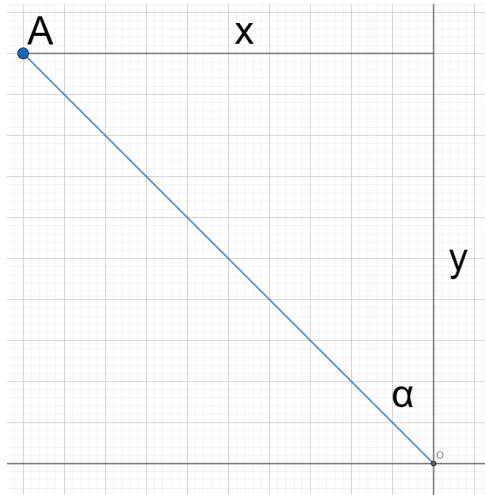


Fonte: Do próprio autor

Em seguida, foi feita uma equação para o ângulo da primeira articulação (torso). Como essa é a única junta capaz de rotacionar perpendicularmente ao eixo Z, seu ângulo pode ser calculado de forma independente das outras juntas, através apenas dos valores de x e y da posição desejada, ignorando o valor de z. Como é possível observar na Figura 12, o ângulo dessa junta deve formar um triângulo retângulo com catetos de comprimento x e y. A partir disso, o ângulo pode ser calculado como:

$$\hat{\text{angulo}}\text{Torso} = \arctan\left(\frac{x}{y}\right) \quad (5.1)$$

Figura 12 – Cálculo do ângulo do torso



Fonte: Do próprio autor

Em seguida, foi feito o cálculo dos ângulos restantes (ombro, cotovelo e pulso). Para que seja possível pegar as peças sem interferir nas outras em casas próximas, é indispensável que a garra do manipulador esteja sempre perpendicular com o tabuleiro. Dessa forma, apenas é necessário calcular os ângulos das duas juntas restantes de forma que o ombro termine nas posições x e y desejadas e a uma altura igual ao z desejado mais o comprimento da garra.

Com base nisso, podemos simplificar os dados necessários para calcular os ângulos do ombro e do cotovelo. Essa parte do manipulador deve ter um comprimento total definido pela hipotenusa do triângulo retângulo apresentado na Figura 12, com catetos x e y. E sua altura total deve ser igual ao z desejado, subtraído dos comprimentos da garra e do torso do manipulador. Portanto, temos as seguintes equações:

$$\text{comprimento} = \sqrt{x^2 + y^2} \quad (5.2)$$

$$\text{altura} = z - \text{comprimentoTorso} - \text{comprimentoGarra} \quad (5.3)$$

A partir desses valores de comprimento e altura, o cálculo desses ângulos se reduz a um problema de cinemática inversa de um manipulador com dois graus de liberdade. Existem duas soluções para esse tipo de problema, uma com o ângulo do segundo eixo positivo em relação ao primeiro, e outra com esse ângulo negativo. Devido às limitações da movimentação do ombro do manipulador utilizado, foi escolhida a solução na qual o ângulo do cotovelo é negativo, descrita pelas equações abaixo. (TORRES-REYES, 2018)

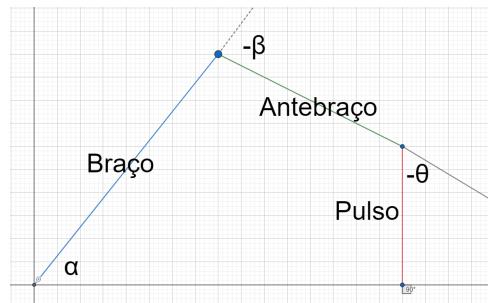
$$\begin{aligned} \text{ânguloCotovelo} = & \quad (5.4) \\ & - \arccos \left(\frac{\text{comprimento}^2 + \text{altura}^2 - \text{comprimentoBraço}^2 - \text{comprimentoAntebraço}^2}{2 \cdot \text{comprimentoBraço} \cdot \text{comprimentoAntebraço}} \right) \end{aligned}$$

$$\begin{aligned}
 \hat{\text{ânguloOmbro}} &= \arctan \frac{\text{altura}}{\text{comprimento}} \\
 &+ \arcsin \left(\frac{\text{comprimentoAntebraço} \cdot \sin(\hat{\text{ânguloCotovelo}})}{\text{comprimentoBraço} + \text{comprimentoAntebraço} \cdot \cos(\hat{\text{ânguloCotovelo}})} \right)
 \end{aligned} \tag{5.5}$$

Por último, resta calcular o ângulo do pulso, que deve ficar perpendicular ao tabuleiro. Para isso, pode ser descrito um quadrilátero envolvendo o braço, antebraço, pulso e o tabuleiro, conforme representado na Figura 13. Como a soma dos ângulos internos de um quadrilátero deve ser igual a 360° , pode-se obter a equação abaixo:

$$\hat{\text{ânguloPulso}} = 90 - \hat{\text{ânguloOmbro}} - \hat{\text{ânguloCotovelo}} \tag{5.6}$$

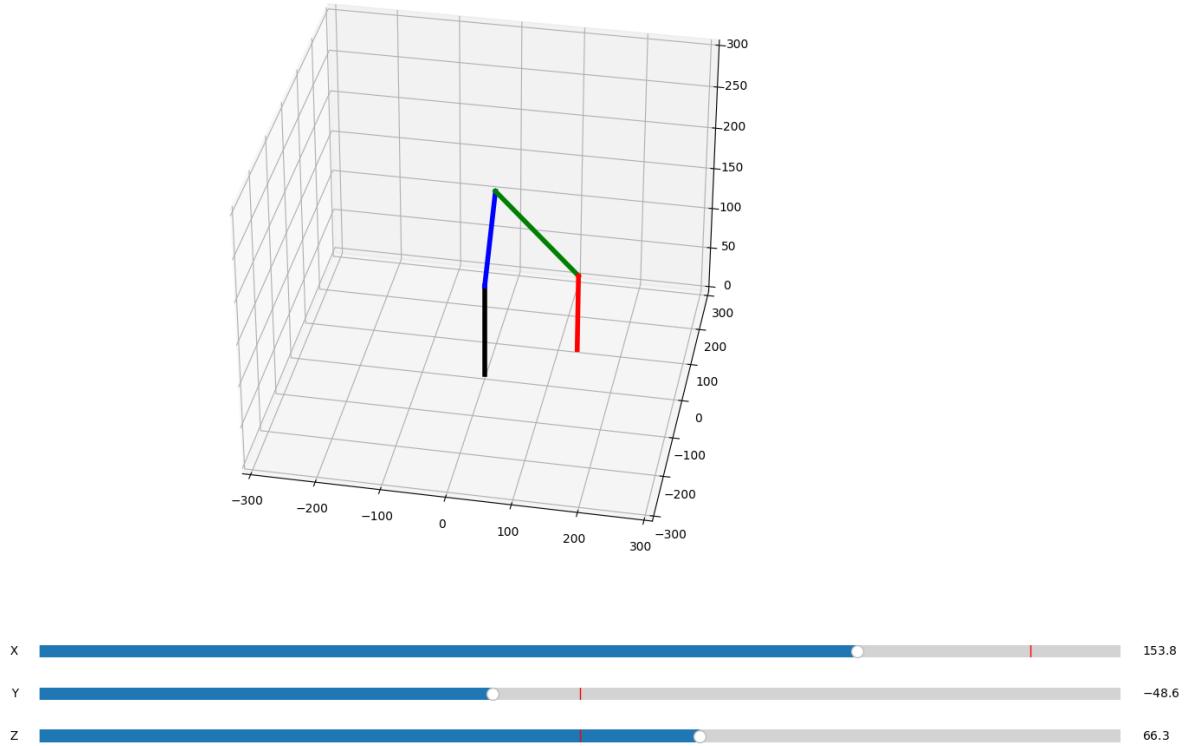
Figura 13 – Cálculo do ângulo do pulso



Fonte: Do próprio autor

Para facilitar a visualização dos ângulos e permitir uma comparação visual entre a posição desejada do manipulador e sua posição real, foi desenvolvido um simples *script* em Python que implementa um modelo simplificado do manipulador. Esse *script* permite movimentar o modelo para uma determinada posição x, y e z, e exibe os ângulos calculados para cada junta. A Figura 14 mostra o modelo desenvolvido.

Figura 14 – Modelo simplificado do manipulador



Fonte: Do próprio autor

5.6 Movimentação de peças

Depois de desenvolvida a funcionalidade de cálculo dos ângulos para movimentar o manipulador para uma determinada posição, foi necessário adicionar uma funcionalidade para utilizar o manipulador para pegar, mover e soltar as peças de xadrez.

Para isso, foi primeiramente definido de forma prática uma altura de movimentação do manipulador, que permite que ele move livremente sem derrubar nenhuma peça e uma altura para pegar as peças, na qual a garra consegue alcançar e pegar qualquer peça. Em seguida, foram mapeadas as coordenadas x e y de todas as casas do tabuleiro. Devido à uma limitação no tamanho do manipulador utilizado, não foi possível alcançar algumas casas, portanto as peças nessas casas devem ser movimentadas manualmente pelo jogador.

A partir disso, foi implementada uma funcionalidade para movimentar o manipulador para uma determinada casa do tabuleiro, apenas nos eixos x e y. Em seguida, foi adicionada uma funcionalidade para pegar e outra para soltar a peça que está na casa atual do manipulador, movimentando-o apenas no eixo z e acionando a garra. Por fim, foi feita uma funcionalidade para movimentar o manipulador para a posição de repouso e outra para movimentá-lo para a posição de descarte de peças.

Devido à relação não linear entre os ângulos das juntas e a posição do manipulador,

foi observado que o manipulador não se movimentava adequadamente entre as casas do tabuleiro, frequentemente derrubando peças. Para solucionar esse problema, cada movimento do manipulador foi dividido em diversos movimentos menores, de forma que o manipulador se movimente de forma mais suave e controlada.

5.7 Comunicação com o computador

Após finalizar a movimentação das peças, foi desenvolvido o módulo responsável por realizar a comunicação entre o microcontrolador e o computador.

Conforme descrito na subseção 4.1.5, essa comunicação é feita através do protocolo serial, por meio de um cabo USB. Entretanto, ainda era necessário definir o formato dos dados que serão enviados e recebidos.

Para isso, foram definidos alguns comandos que podem ser enviados pelo computador para o microcontrolador, conforme descrito na Tabela 4.

Tabela 4 – Comandos enviados pelo computador para o microcontrolador

Comando	Descrição	Formato
move	Movimentar para casa	move x y
grab	Pegar peça	grab
release	Soltar peça	release
discard	Movimentar para posição de descarte	discard
reset	Movimentar para posição de repouso	reset
wait	Aguarde por x milisegundos	wait x

Fonte: Do próprio autor

Ao final da execução de cada comando, o microcontrolador envia uma mensagem de confirmação para o computador, informando se o comando foi executado com sucesso (OK) ou não (FAIL).

Para possibilitar o controle manual do manipulador, foram definidas algumas mensagens que podem ser enviadas pelo microcontrolador para o computador, conforme descrito na Tabela 5.

Tabela 5 – Mensagens enviadas pelo microcontrolador para o computador

Mensagem	Descrição	Formato
joystick	Joystick foi movimentado	joystick <up down left right>
select	Botão do joystick foi pressionado	select

Fonte: Do próprio autor

5.8 Implementação da lógica de xadrez

Depois de finalizar a comunicação entre o microcontrolador e o computador, foi desenvolvido o *software* responsável por implementar a lógica do jogo de xadrez.

O *software* foi desenvolvido em Python, utilizando a biblioteca stockfish (ZHELYA-BUZHISKY, 2022) para realizar comunicação com a *engine* de xadrez Stockfish 15.1 (ROMSTAD; COSTALBA; KIISKI, 2022). Essa *engine* é responsável por verificar quais movimentos são válidos e calcular qual movimento o computador deve jogar em cada turno.

Além da utilização da biblioteca stockfish, foi necessário implementar algumas funcionalidades manualmente que não existem nela. Em primeiro lugar, foi implementada uma função para adquirir quais movimentos são válidos para a posição atual do tabuleiro, utilizando o comando *go perft 1* da *engine* Stockfish. Em seguida, foi desenvolvida uma funcionalidade para detectar quando a partida acabou, verificando se não existe mais nenhum movimento válido para o jogador atual. Por último, foi feita uma função para verificar se o jogo terminou empatado, verificando a avaliação da posição atual do tabuleiro pela *engine* Stockfish.

Para comunicar com o microcontrolador, foi utilizada a biblioteca PySerial (LIE-CHTI, 2020). Através dessa biblioteca, é possível enviar e receber as mensagens descritas na seção 5.7. Após enviar uma mensagem para o microcontrolador, o *software* aguarda uma mensagem de confirmação para indicar o sucesso ou falha da execução do comando.

O programa desenvolvido suporta dois modos de jogo: jogar contra o computador ou contra outra pessoa. No primeiro modo, o *software* Stockfish 15.1 é utilizado para calcular qual movimento o computador irá jogar, enquanto no segundo modo, o jogador que estiver controlando o manipulador pode movimentá-lo pelas casas do tabuleiro e pegar peças sem restrição, entretanto ao tentar soltar uma peça, é feita uma consulta ao Stockfish 15.1 para validar se a jogada desejada é válida. Em ambos os modos de jogo, os movimentos realizados pelo jogador humano que não controla o manipulador robótico devem ser digitados manualmente no computador.

Ao detectar uma captura de peça, o programa primeiro controla o manipulador para movimentar a peça capturada para a posição de descarte. Em seguida é feita a movimentação da peça que capturou para a casa de destino. Foi feito um tratamento especial para capturas do tipo *en passant*, pois nesses movimentos a peça capturada não está na casa de destino.

Um outro tratamento especial foi feito para movimentos de roque, pois nesses movimentos duas peças são movimentadas ao mesmo tempo.

Por fim, caso o manipulador robótico esteja jogando com as peças pretas, o programa

realiza uma conversão da posição do tabuleiro para realizar o movimento adequado.

6 Conclusão

A partir do desenvolvimento deste trabalho, foi possível obter uma plataforma simples e de baixo custo que permite o jogo entre duas pessoas ou entre uma pessoa e o computador. Essa plataforma pode ser apresentada para crianças e jovens em feiras e eventos com o objetivo de introduzir conceitos básicos de computação, engenharia elétrica e controle de sistemas, além de instigar o conhecimento nessas áreas.

6.1 Performance

Após finalizar o projeto, foi feita uma análise da performance do sistema, com o objetivo de verificar quanto tempo o manipulador robótico leva para realizar um movimento.

Para isso, foi jogada uma partida contra o computador, anotando o tempo gasto para cada movimento realizado pelo manipulador robótico. O resultados obtidos estão apresentados no Apêndice B.

6.2 Limitações

O projeto apresenta algumas limitações que podem ser melhoradas em trabalhos futuros.

Em primeiro lugar, o manipulador robótico escolhido não apresenta tamanho suficiente para alcançar todas as casas do tabuleiro, o que limita o jogo, principalmente quando uma pessoa está controlando o dispositivo. A escolha de um modelo de manipulador robótico com maior alcance pode resolver esse problema.

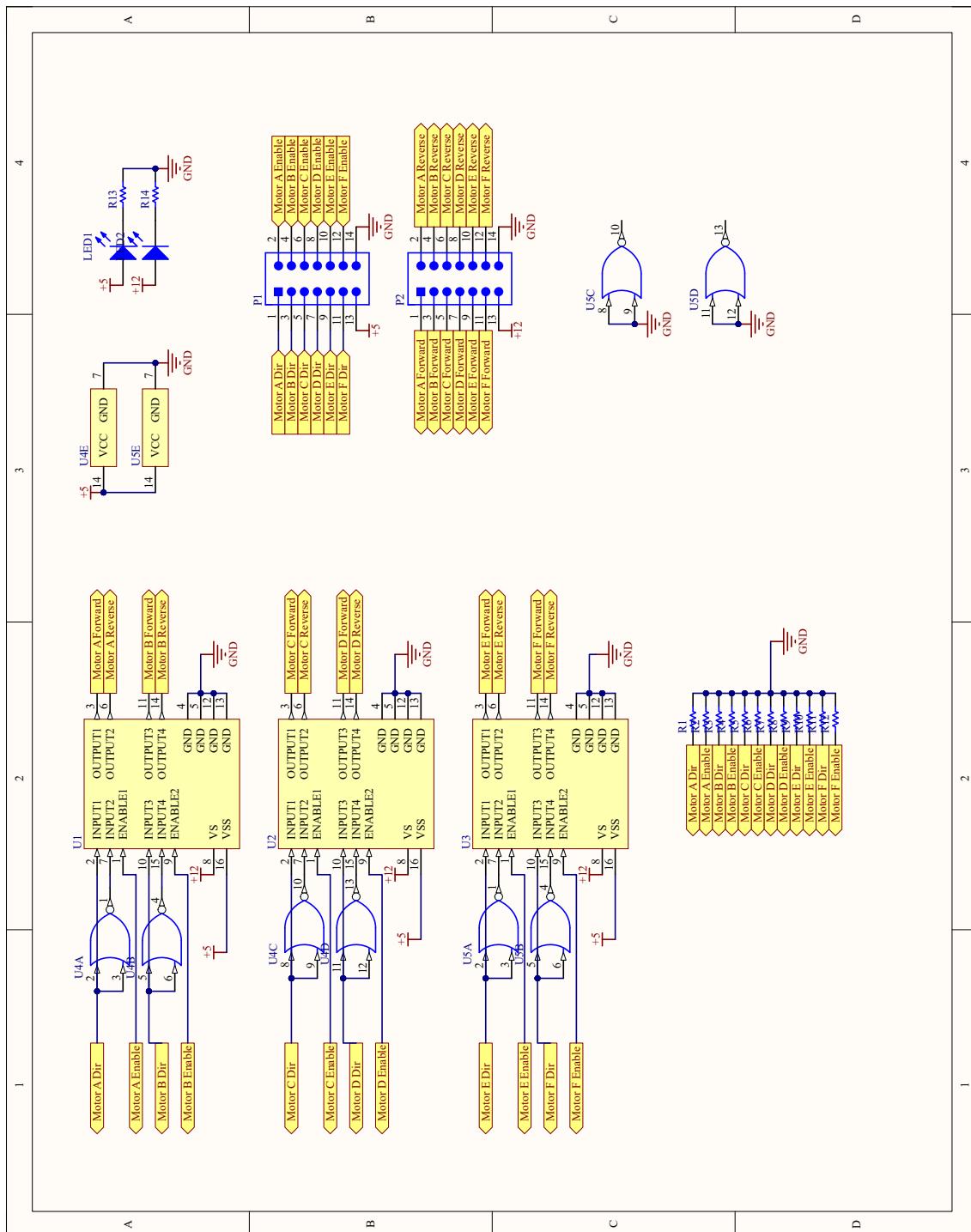
Além disso, o projeto não apresenta um sistema de detecção de peças no tabuleiro, o que faz com que o usuário tenha que informar ao computador qual peça foi movida. A implementação de um sistema de visão computacional pode tornar o sistema muito mais interativo e intuitivo.

Referências

- CARDONA, A. et al. Uma ferramenta computacional para manipulação de um braço robótico. 2012.
- DIGILENT. *H-Bridges*. 2012. Disponível em <<https://learn.digilentinc.com/Documents/325>>. Acesso em 10/12/2022.
- DOROUKA, P.; PAPADAKIS, S.; KALOGIANNAKIS, M. Tablets and apps for promoting robotics, mathematics, stem education and literacy in early childhood education. *Int. J. Mobile Learning and Organisation*, Vol. 14, No. 2, 2020, Rethymnon and Heraklion, Crete, Greece, 2020.
- ESPRESSIF. *ESP32 Series Datasheet*. 2023. Disponível em <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>. Acesso em 02/06/2023.
- LEPRINCE-RINGUET, D. *The shortage of tech workers is about to become an even bigger problem for everyone*. [S.l.]: ZDNET, 2021. Disponível em <<https://www.zdnet.com/article/the-shortage-of-tech-workers-is-about-to-become-an-even-bigger-problem-for-everyone>>. Acesso em 10/11/2022.
- LIECHTI, C. *Pyserial*. 2020. Disponível em <<https://pypi.org/project/pyserial>>. Acesso em 22/05/2023.
- MORAES, L. D. R. de. Jogador de xadrez robótico com visão computacional. 2021.
- OLUNLOYO, V. O.; AYOMOH, M.; ADEOTI, I. On the mentor arm position placement problem: A forward kinematics analysis. 2011.
- OLUNLOYO1, V. et al. Inverse kinematics analysis of a five jointed revolute arm mechanism. 2014.
- RIUL, J. A.; MONTENEGRO, P. H. de M.; FERREIRA, G. de S. Controle neural de três elos de um robô de cinco graus de liberdade. 2018.
- ROMSTAD, T.; COSTALBA, M.; KIISKI, J. *Stockfish*. 2022. Disponível em <<https://stockfishchess.org>>. Acesso em 22/05/2023.
- TORRES-REYES, N. *2-Link Kinematics of Planar Robotic Arm*. 2018. Disponível em <https://www.dashhub.org/unlv/wiki/doku.php?id=2_link_kinematics>. Acesso em 26/05/2023.
- ZHELYABUZHHSKY, I. *Stockfish*. 2022. Disponível em <<https://pypi.org/project/stockfish>>. Acesso em 22/05/2023.

Apêndices

APÊNDICE A – Esquemático da placa de controle



APÊNDICE B – Performance do sistema

Tabela 6 – Tempo de execução de cada movimento

	Humano	Computador	Tempo (s)
1	e4	c5	20.7
2	Nf3	e6	19.0
3	Nc3	a6	23.2
4	Bc4	b5	21.5
5	Bb3	c4	20.9
6	Bxc4	bxcc4	41.8
7	d3	cx d3	41.8
8	Qxd3	Bb7	21.6
9	Be3	Nf6	18.8
10	Ne5	Nc6	21.2
11	O-O	Nxe5	40.0
12	Qe2	Qc7	19.8
13	f3	Bc5	19.8
14	Bxc5	Qxc5+	41.4
Tempo médio			26.5
Tempo médio movimento			20.7
Tempo médio captura			41.3