



ROBÔ SEGUIDOR DE LINHA COMO FERRAMENTA DE ENSINO

DOI: 10.37702/2175-957X.COBENGE.2025.6235

Autores: PAULA CAROLINE DOS SANTOS GUEDES, REJANE DE BARROS ARAUJO

Resumo: Este artigo apresenta a utilização de um robô seguidor de linha em ambiente simulado como recurso didático para o ensino de Engenharia, com foco nas áreas de robótica, eletrônica, programação e teoria de controle. Através da plataforma GearsBot, os estudantes podem programar e testar algoritmos de controle em robôs virtuais, compreendendo conceitos fundamentais como sensores, lógica condicional, sistemas embarcados e malha de controle. São exploradas abordagens desde programações em blocos até o uso de linguagens como Python, com implementação de controles proporcionais e Proporcionais-Integrativos-Derivativos (PID). Os resultados demonstram que o uso da simulação favorece o aprendizado ativo, estimula a autonomia e fortalece a ponte entre teoria e prática. Como proposta futura, destaca-se a construção física do robô, ampliando ainda mais a experiência educacional dos alunos.

Palavras-chave: Robótica Educacional, Algoritmo Computacional, Robô Seguidor de Linha

ROBÔ SEGUIDOR DE LINHA COMO FERRAMENTA DE ENSINO

1 INTRODUÇÃO

Para Jaime e Leonel (2024) as últimas décadas trouxeram uma ascensão do uso das tecnologias alinhado ao avanço da ciência, devido ao desenvolvimento de computadores, a disseminação do seu uso, a invenção da Internet e em específico ao desenvolvimento de softwares especializados. Nesse viés, com a crescente demanda por métodos de ensino mais dinâmicos e eficazes nas engenharias, o uso de tecnologias aplicadas tem ganhado destaque como estratégia para facilitar a compreensão de conceitos complexos. Entre essas tecnologias, a simulação computacional de robôs autônomos, como o robô seguidor de linha, tem se mostrado uma ferramenta pedagógica valiosa. Esse tipo de robô, amplamente utilizado em atividades de iniciação à robótica, oferece um ambiente acessível para o estudo prático de temas essenciais como sensores, sistemas embarcados, lógica de controle e eletrônica aplicada.

Ao empregar um robô seguidor de linha em simulação, os alunos têm a oportunidade de projetar e testar algoritmos de controle, analisar respostas do sistema a diferentes cenários e compreender a integração entre hardware e software, tudo isso em um ambiente seguro, flexível e de baixo custo. Estas atividades trazem para dentro da sala de aula a sensação de um ambiente real, onde os acadêmicos podem aplicar, de forma prática, os conceitos apresentados nas aulas expositivas (Silva, 2016). Além disso, essa abordagem favorece o desenvolvimento do raciocínio lógico, da capacidade de resolução de problemas e da familiaridade com ferramentas de modelagem e simulação, cada vez mais presentes no mercado de trabalho. Este artigo propõe a utilização de um robô seguidor de linha em ambiente simulado como recurso didático no ensino de Engenharia, com foco nas áreas de Robótica, Eletrônica, Programação e Controle. A proposta visa demonstrar como a simulação pode potencializar o aprendizado prático, incentivando a experimentação e a autonomia dos estudantes no processo de construção do conhecimento técnico.

2 ROBÔ SEGUIDOR DE LINHA

A robótica educacional emergiu como um recurso tecnológico de aprendizagem, único que pode oferecer o “aprender fazendo”, bem como atividades lúdicas em um ambiente de aprendizagem atrativo, que fomenta o interesse e curiosidade dos alunos (Campos, 2017). Nesse sentido, o robô seguidor de linha permite que os estudantes apliquem, conhecimentos de eletrônica, programação e controle, ao mesmo tempo em que exploram soluções criativas para problemas reais. Sua simplicidade de implementação e clareza no comportamento facilitam a visualização dos efeitos de cada ajuste realizado, promovendo um aprendizado ativo e significativo. Além disso, o caráter lúdico e desafiador da robótica estimula a participação dos alunos e contribui para o desenvolvimento de competências técnicas e cognitivas essenciais para a formação de engenheiros.

Segundo SGA et al. (2019) o robô seguidor de linha nada mais é que um robô automatizado para realizar o trabalho de seguir uma determinada linha, com sua cor e espessura bem definidas para sua correta leitura. Logo, entende-se que o robô seguidor de linha é um tipo de robô automatizado projetado para detectar e seguir um percurso previamente definido por uma linha, geralmente de cor contrastante (como preta sobre fundo branco) e com espessura padronizada para garantir sua leitura precisa. Utilizando sensores,

normalmente do tipo infravermelho, o robô identifica a linha no chão e ajusta seu movimento em tempo real para permanecer sobre o trajeto. Essa simplicidade de funcionamento torna o robô seguidor de linha uma excelente ferramenta didática, pois permite aplicar conceitos fundamentais de sensoriamento, lógica de controle, programação e atuação de motores, sendo ideal para introduzir estudantes à robótica e à automação de maneira prática e acessível.

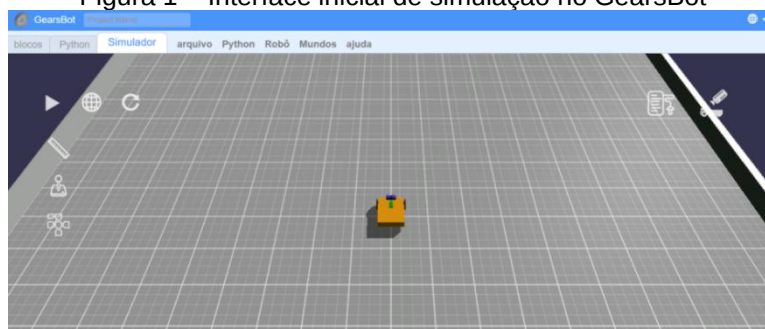
O funcionamento de um robô seguidor de linha está fundamentado no uso de sensores de luz, geralmente sensores infravermelhos, que detectam a presença da linha no chão. Esses sensores captam a diferença de contraste entre a linha (normalmente escura) e o fundo, enviando sinais ao controlador sempre que a linha é identificada. A partir dessas informações, o controlador processa os dados e ajusta os motores de forma a manter o robô alinhado ao trajeto, corrigindo sua direção sempre que necessário. Esse processo se baseia no princípio da realimentação, em que o sistema monitora continuamente seu próprio desempenho e faz ajustes em tempo real para atingir o objetivo desejado — neste caso, permanecer sobre a linha. Trata-se de um exemplo clássico de sistema de controle em malha fechada, no qual a saída do sistema, representada pelo movimento do robô, é constantemente comparada com a referência (a posição da linha), permitindo correções automáticas e precisas

3 METODOLOGIA

A metodologia adotada neste artigo baseia-se na realização de simulações utilizando a plataforma virtual GearsBot (<https://gears.aposteriori.com.sg/>), uma ferramenta educacional gratuita voltada para o ensino de robótica e programação. Por meio desta plataforma, é possível demonstrar o funcionamento de um robô seguidor de linha, explorando tanto a programação em blocos quanto em Python, de acordo com o nível de familiaridade dos estudantes com a lógica computacional. A abordagem prática proporcionada pela GearsBot permite a visualização em tempo real do comportamento do robô frente a diferentes comandos e situações.

Um dos principais diferenciais da plataforma é a possibilidade de personalização. Os usuários podem configurar seu próprio robô, escolhendo os sensores e atuadores mais adequados à tarefa proposta, como sensores infravermelhos para detecção de linha e motores para movimentação. Além disso, a GearsBot permite a criação de mundos personalizados — ou seja, arenas virtuais — que podem variar em tamanho, forma e nível de complexidade, o que torna o ambiente de simulação mais dinâmico e adaptável aos objetivos pedagógicos. Durante a simulação, são testadas diferentes configurações e estratégias de controle, permitindo aos alunos observar, analisar e aprimorar o desempenho do robô seguidor de linha. Essa prática reforça o aprendizado dos conceitos de robótica, eletrônica e controle, ao mesmo tempo em que estimula o raciocínio lógico e a capacidade de solução de problemas. Na Figura 1 é possível observar a tela de simulação da Plataforma GearsBot.

Figura 1 – Interface inicial de simulação no GearsBot



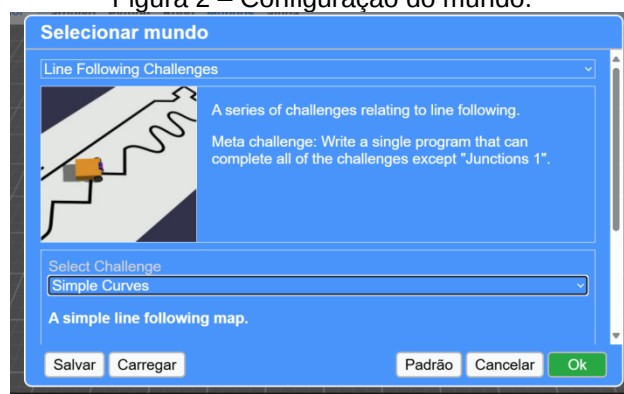
Fonte: Autor

4 RESULTADOS

Nesta seção, são apresentados os resultados obtidos por meio das simulações realizadas com o robô seguidor de linha na plataforma GearsBot. São analisados o comportamento do robô em diferentes configurações de sensores e arenas, bem como o impacto das estratégias de programação utilizadas. Esses resultados demonstram, na prática, como os conceitos de robótica e da teoria de controle se aplicam ao funcionamento do robô, contribuindo para a compreensão e o aprofundamento do aprendizado dos estudantes.

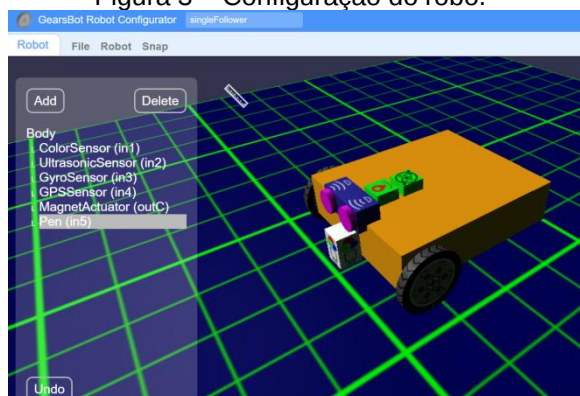
O primeiro passo para utilizar o robô seguidor de linha na plataforma GearsBot é acessar a opção "Mundos" na barra de tarefas e selecionar o cenário "Line Following Challenges". Em seguida, na opção "Select Challenge", o usuário pode escolher entre diversas arenas, que variam desde trajetos simples, com curvas suaves, até percursos mais complexos, com curvas acentuadas e obstáculos. Em relação ao robô utilizado, é importante destacar que o modelo padrão já vem equipado com sensores de cor, sensor ultrassônico, GPS, giroscópio e atuador magnético. No entanto, a plataforma oferece total flexibilidade: por meio das configurações do robô, é possível adicionar ou remover sensores e atuadores conforme os objetivos da simulação. Essa personalização permite adaptar o robô às necessidades específicas de cada experimento, enriquecendo o aprendizado e a experimentação prática. Feito isso, pode-se agora focar na programação do robô que pode ser por meio de blocos ou código Python. Nas Figuras 2 e 3, observa-se as Configurações do mundo e do robô, respectivamente.

Figura 2 – Configuração do mundo.



Fonte: Autor

Figura 3 – Configuração do robô.

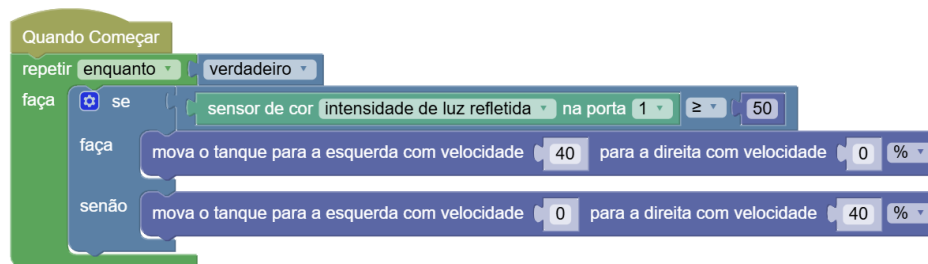


Fonte: Autor

4.1 Seguidor de linha simples

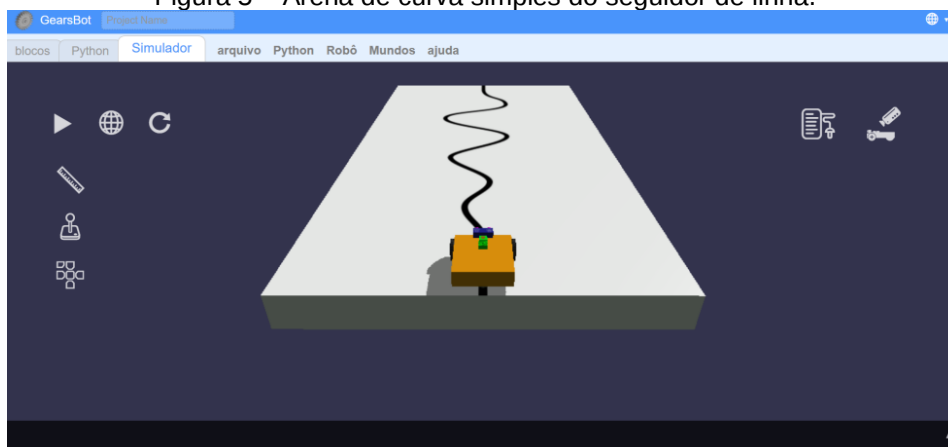
Nesta simulação, é selecionado o cenário “Simple Curve”, disponível em Line Following Challenges, e opta-se por utilizar o robô padrão oferecido pela plataforma. Com essas configurações definidas, inicia-se o processo de programação do robô, primeiramente utilizando a linguagem em blocos para facilitar a visualização da lógica de funcionamento. Nas Figuras 4 e 5 tem-se a programação em bloco simples para o robô andar e a arena de simulação com curva simples do seguidor de linha, respectivamente.

Figura 4 – Programação em blocos simples para o robô andar.



Fonte: Autor

Figura 5 – Arena de curva simples do seguidor de linha.



Fonte: Autor.

O código em blocos, utilizado na Figura 4, define o comportamento do robô com base na leitura do sensor de cor, que detecta a intensidade de luz refletida pelo solo. O programa começa com um laço de repetição "enquanto verdadeiro", ou seja, ele é executado continuamente. Dentro do laço, há uma estrutura condicional "se/senão" que toma decisões com base na leitura do sensor conectado à porta 1. Se a intensidade de luz refletida for maior ou igual a 50, o robô interpreta que está sobre uma superfície clara e move o tanque para a esquerda com velocidade 40%, enquanto mantém a direita parada, fazendo o robô virar levemente à esquerda. Caso contrário, se a intensidade for menor que 50 (linha escura), o robô para a esquerda e move a direita com velocidade 40%, fazendo-o virar para o lado oposto. Esse movimento alternado permite que o robô se mantenha alinhado à linha, corrigindo sua trajetória constantemente.

Esse tipo de programação pode ser facilmente adaptado para diferentes objetivos. Por exemplo, é possível incluir comandos para parar o robô ao atingir determinada condição (como o tempo de execução, a leitura de um obstáculo, ou a chegada ao fim da linha), inserindo uma lógica personalizada no código. Essa flexibilidade é justamente um dos grandes benefícios de se utilizar robôs em simulações como ferramenta educacional: ela permite que os alunos explorem e testem diferentes estratégias de controle e programação na prática,

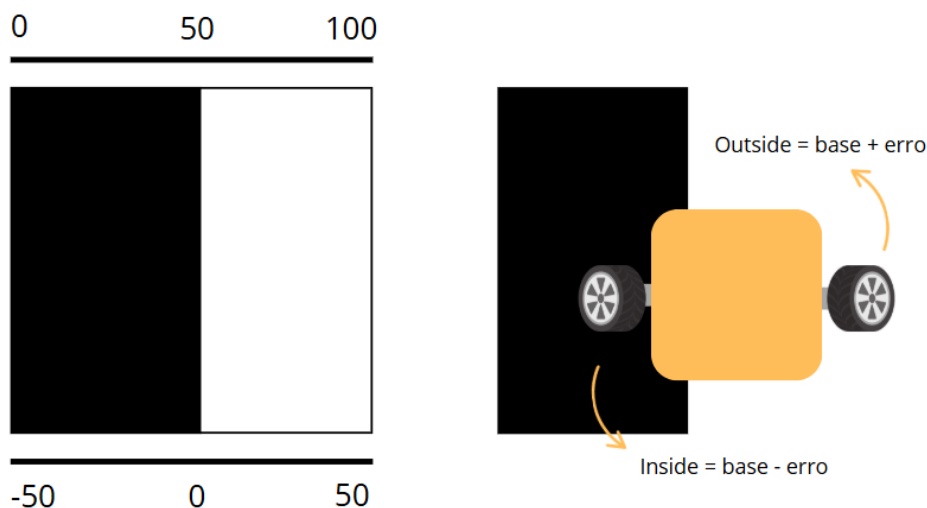
compreendendo de forma aplicada conceitos que, muitas vezes, ficam restritos ao campo teórico.

4.2 Seguidor de linha com controle

Embora a programação em blocos apresentada na Figura 4 seja suficiente para fazer o robô seguir a linha de forma básica, ela não garante uma operação totalmente eficaz. A eficiência pode ser comprometida dependendo da posição do sensor, da largura da linha ou de mudanças bruscas no percurso. Nesses casos, o robô pode apresentar movimentos imprecisos ou sair do trajeto. Para resolver esse problema, a implementação de um controle se torna uma solução eficaz, pois permite ajustes contínuos na movimentação do robô com base no erro da intensidade de luz refletida, garantindo maior estabilidade, precisão e adaptação ao percurso. Em primeira instância, é considerada a aplicação de um controle proporcional, o princípio se baseia na leitura da intensidade de luz refletida pelo sensor do robô, que varia entre 0% (superfície totalmente preta) e 100% (superfície totalmente branca). O ponto ideal de leitura é 50%, que representa a borda entre a linha escura e o fundo claro. A partir disso, calcula-se o erro como a diferença entre a leitura atual e esse valor ideal de 50 que é então utilizado para ajustar proporcionalmente a velocidade dos motores do robô.

A Figura 6 mostra como isso é aplicado: o motor do lado interno da curva tem sua velocidade reduzida em proporção ao erro (base - erro), enquanto o motor do lado externo tem sua velocidade aumentada (base + erro). Essa diferença de velocidade faz com que o robô corrija sua trajetória automaticamente, mantendo-se sobre a linha. Essa abordagem permite um movimento mais suave e eficiente, em comparação com controles simples baseados apenas em condições lógicas (ligado/desligado). A Equação (1) e a Equação (2) mostram o cálculo para o erro e o controle proporcional, respectivamente.

Figura 6 – Funcionamento do controle com base na intensidade da luz

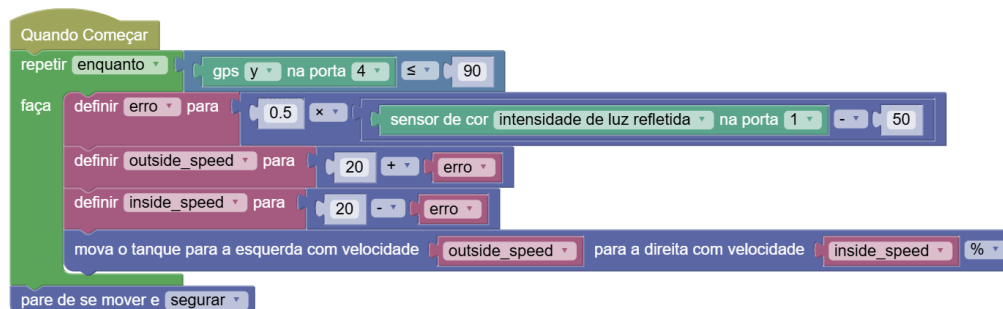


Fonte: Autor

$$Erro = Intensidade - 50 \quad (1)$$

$$G_c(s) = K_c \times Erro \quad (2)$$

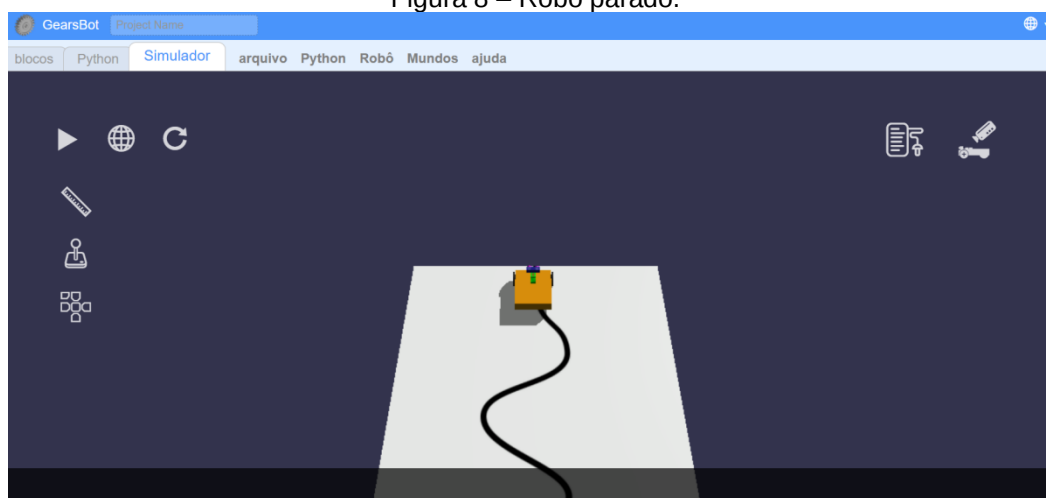
Figura 7 – Programação com uma parada e o controle proporcional.



Fonte: Autor

O programa apresentado na Figura 7 utiliza um controle proporcional para guiar um robô seguidor de linha com base nas leituras de um sensor de cor. Esse tipo de controle ajusta a velocidade das rodas de forma proporcional ao desvio do robô em relação à linha. O desvio, chamado de erro, é calculado pela Equação 1. Esse erro é multiplicado por um fator de correção ($K_c = 0,5$) para suavizar a resposta. A partir disso, são definidas duas velocidades: uma para a roda do lado de fora da curva (*outside_speed*) e outra para a roda do lado de dentro (*inside_speed*). Dessa forma, quanto maior o erro, maior é a diferença entre as velocidades das rodas, o que permite ao robô corrigir sua trajetória e continuar seguindo a linha. Na Figura 8, observa-se a tela de simulação do robô seguidor de linha com o controlador proporcional.

Figura 8 – Robô parado.



Fonte: Autor.

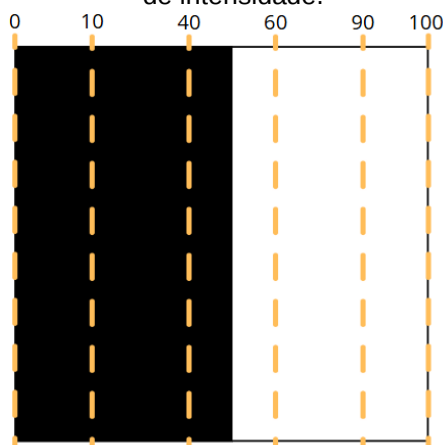
Uma melhoria importante implementada neste programa é a parada do robô com base em sua posição geográfica. Isso é feito por meio de um sensor GPS conectado à porta 4, que verifica constantemente a coordenada Y. O robô executa o controle proporcional enquanto o valor de Y for menor ou igual a 90 cm. Quando essa condição deixa de ser verdadeira, o laço de repetição é encerrado e o comando de parada é ativado, fazendo com que o robô interrompa seu movimento de forma controlada como pode ser visto na Figura 8. Antes dessa modificação, o robô não possuía nenhum critério para encerrar a execução, seguindo a linha indefinidamente. Com a inclusão do sensor GPS, ele agora consegue identificar o momento exato de parar com base em sua localização.

Outro tipo de simulação possível, é um controle baseado em faixas de intensidade de luz refletida detectadas pelo sensor de cor. O programa implementa uma lógica de decisões

utilizando uma estrutura condicional encadeada (vários "se-senão") para ajustar o movimento do robô conforme o valor lido pelo sensor. O algoritmo lê a intensidade de luz refletida na porta 1 e armazena esse valor na variável "Intensidade". Com base no valor da intensidade, o robô executa diferentes ações de movimento.

Esse tipo de controle, embora não utilize cálculos matemáticos contínuos como o controle proporcional clássico, representa uma forma simplificada de controle proporcional por faixas. Ele se baseia na variação da intensidade da luz refletida, lida pelo sensor, que varia de 0% (preto absoluto) a 100% (branco total). O robô interpreta esse intervalo como uma escala que indica sua posição em relação à linha. O algoritmo divide essa faixa de 0 a 100% em zonas ou intervalos de decisão: 0%, 10%, 40%, 60%, 90% e 100%. Cada uma dessas zonas está associada a um comportamento específico de correção, ajustando a velocidade das rodas de forma diferente para que o robô possa retornar ou manter-se sobre a linha. Valores mais próximos de 0% indicam que o robô está muito sobre a faixa preta, portanto, precisa virar à direita; valores mais próximos de 100% indicam que ele está sobre a superfície branca, exigindo uma curva para a esquerda. A Figura 9 mostra a divisão da faixa para análise de intensidade.

Figura 9 – Divisão da faixa para análise de intensidade.



Fonte: Autor

Essa divisão em faixas permite ao robô responder gradualmente ao desvio da linha, com curvas mais acentuadas nas extremidades (0% a 10% e 90% a 100%) e correções mais suaves nas regiões centrais (em torno de 40% a 60%), funcionando assim como uma forma de controle proporcional escalonado.

Diante disso, nesta simulação agora é utilizada a arena Sharp Turns, que apresenta curvas mais acentuadas e desafiadoras para o sensor seguir com precisão. Isso pode ser constatado ao aplicar a mesma programação em blocos da Figura 4 para essa nova arena: ao tentar seguir essa nova pista, o robô acaba se perdendo da linha, evidenciando as limitações desse controle simples em cenários mais complexos. Nesse caso, o controle com base na variação proporcional da linha se torna mais eficaz. Nas Figuras 10 e 11, observa-se a programação do novo controle proporcional e a simulação do comportamento do robô em uma arena mais complexa, respectivamente.

Figura 10 – Programação com o novo controle proporcional



Fonte: Autor

Figura 11 – Comportamento do Robô em uma arena mais complexa.



Fonte: Autor

O controle proporcional baseado na variação de faixas é fundamental para garantir que o robô não se desvie da linha em trajetos com curvas e dobras acentuadas. Ao dividir a transição entre o preto e o branco em diferentes faixas de intensidade, o sistema consegue ajustar a direção com diferentes níveis de correção, conforme a posição estimada do robô. Esse tipo de controle por zonas permitiu respostas rápidas e direcionadas, tornando o robô mais eficiente em percursos complexos.

Segundo Coelho, Jeronymo e Araújo (2019), o controlador proporcional é um tipo de controle instantâneo e sem memória, isto é, o valor atual depende apenas do valor instantâneo do erro. Além do controle proporcional, também é possível implementar os termos integral e derivativo, tanto separadamente como em conjunto. Nos casos em que se busca uma melhora tanto na resposta transitória como na resposta de regime permanente, dois zeros e um polo na origem são adicionados, formando assim um controle PID (Proporcional, Integral, Derivativo), (Fadali & Visioli, 2014). Logo, esse controlador oferece ainda mais precisão e estabilidade ao movimento do robô. Para quem se sente mais familiarizado com a linguagem Python, existe a possibilidade de desenvolver esse tipo de controle diretamente por meio de código em texto, o que permite maior flexibilidade nos ajustes e testes. Com o objetivo de

exemplificar essa abordagem, é realizado um teste na linguagem Python utilizando um controle PID, que apresenta um desempenho superior em relação ao controle puramente proporcional, especialmente na suavidade das correções e na estabilidade durante o trajeto.

Para Visioli (2006), uma das formas de implementar o controle PID é pela forma paralela, neste caso, as três ações (P, I e D) são completamente separadas, como mostra a Equação (3) com a função de transferência de um controlador PID paralelo, sendo K_c , o ganho proporcional, K_i , o ganho Integral e K_d o ganho derivativo. Na Figura 12 têm-se a programação do controle PID desenvolvido na linguagem Python para a simulação, os parâmetros sintonizados por tentativa e erro, a saber: $K_c = 0,5$, $K_i = 0,8$ e $K_d = 0,01$;

$$\frac{U(s)}{E(s)} = G_c(s) = K_c + \frac{K_i}{s} + K_d s \quad (3)$$

Figura 12 – Programação em python de um controle PID

```
81 while True:
82     # Obtém o tempo atual para cálculos precisos
83     current_time = time.time()
84     delta_time = current_time - last_time
85
86     # Lê o sensor e calcula o erro atual
87     intensidade = color_sensor.in1.reflected_light_intensity
88     error = intensidade - TARGET
89
90     # Calcula o termo Proporcional (P)
91     P = KC * error
92
93     # Calcula o termo Integral (I) com prevenção de windup
94     integral += error * delta_time
95     I = KI * integral
96
97     # Calcula o termo Derivativo (D)
98     derivative = (error - last_error) / delta_time if delta_time > 0 else 0
99     D = KD * derivative
100
101     # Calcula a saída total do PID
102     pid_output = P + I + D
103
104     # Atualiza o último erro e tempo para próxima iteração
105     last_error = error
106     last_time = current_time
107
108     # Calcula as velocidades dos motores (limitando entre -100 e 100)
109     outside = max(min(25 + pid_output, 100), -100) # Motor externo
110     inside = max(min(25 - pid_output, 100), -100) # Motor interno
111
112     # Aplica as velocidades aos motores
113     tank_drive.on(outside, inside)
114     time.sleep(0.01)
```

Fonte: Autor

O código da Figura 12 implementa um controle PID na arena da Figura 5 de curva simples. Com testes para a sintonia por tentativa e erro foi identificado que um dos valores ideais para o ganho proporcional (K_c) começa em 0,5. Valores menores fazem com que o robô não responda de forma adequada às variações da linha, prejudicando o desempenho. Já para o ganho integral (K_i), o valor de 0,8 mostrou-se eficaz, sendo observado que quanto maior o K_i , melhor o robô corrige erros acumulados ao longo do trajeto, mantendo-se mais alinhado à linha. Em contrapartida, o ganho derivativo (K_d) tem melhor resultado com o valor de 0,01, pois valores mais altos deixam o robô instável e com movimentos excessivamente

oscilantes. Esses ajustes demonstram a importância da sintonia fina dos parâmetros para garantir um controle eficiente e preciso no seguidor de linha.

Conseguir simular e observar na prática as mudanças no comportamento do robô seguidor de linha a partir da variação dos ganhos do controlador PID foi fundamental para o aprendizado. Essa experiência permitiu compreender de forma concreta como cada parâmetro influencia diretamente na estabilidade, precisão e desempenho do sistema, reforçando a importância da aplicação prática dos conceitos teóricos no desenvolvimento de soluções de controle.

5 CONSIDERAÇÕES FINAIS

A utilização de um robô seguidor de linha como ferramenta de ensino na engenharia tem se mostrado extremamente valiosa por promover uma experiência prática e interativa na assimilação de conceitos fundamentais. Ao programar o robô, os estudantes aplicam diretamente conhecimentos teóricos de lógica de programação, desenvolvendo o pensamento computacional e compreendendo, na prática, como estruturas condicionais, laços de repetição e sensores influenciam o comportamento de sistemas automatizados. Além disso, o projeto permite a introdução e experimentação de técnicas de controle como o PID, fundamentais para entender o funcionamento de sistemas de controle em malha fechada, muito importante ao engenheiro de controle e automação e amplamente aplicados na indústria.

A possibilidade de simular diferentes trajetórias e ajustar parâmetros em tempo real torna o processo de aprendizagem mais dinâmico, estimulante e orientado à resolução de problemas, permitindo que os alunos explorem cenários variados e desenvolvam soluções eficazes para manter o robô na rota correta. A personalização do robô e do ambiente também favorece o desenvolvimento da criatividade e da autonomia técnica. Vale destacar que toda a metodologia aplicada e os resultados obtidos foram utilizados na disciplina de Introdução à Robótica do curso de Engenharia de Controle e Automação do IFPA, onde foi possível perceber, de forma evidente, uma melhora significativa no desempenho e na compreensão dos alunos ao longo da atividade.

Como continuidade do projeto, está prevista a construção física do robô, o que proporcionará aos alunos o enfrentamento de desafios concretos como a montagem, a calibração de sensores e a aplicação dos algoritmos em condições reais. Essa transição do ambiente simulado para o mundo físico consolida o aprendizado e evidencia a importância da robótica educacional como uma ponte entre teoria e prática na formação de futuros engenheiros.

REFERÊNCIAS

CAMPOS, Flavio Rodrigues. Robótica educacional no Brasil: questões em aberto, desafios e perspectivas futuras. **Revista Ibero-Americana de Estudos em Educação**, Araraquara, v. 12, n. 4, p. 2108-2121, 2017.

COELHO, A. A. R.; JERONYMO, D. C.; ARAUJO, R de B. **Sistemas Dinâmicos: Controle clássico e preditivo discreto**, UFSC, 1ª Edição, 2019

FADALI, M. Sami; VISIOLI, Antonio. **Digital Control Engineering: Analysis and Design**. St. Louis: Elsevier Science, 2014.

JAIME, D.; LEONEL, A. Uso de simulações: Um estudo sobre potencialidades e desafios apresentados pelas pesquisas da área de ensino de física. **Revista Brasileira De Ensino De Física**, v. 46, e20230309, 2024.

SGAI, Leonardo Machado *et al.* **Projeto de um robô seguidor de linha**. Disponível em: https://www.academia.edu/41278427/PROJETO_DE_UM_ROB%C3%94_SEGUIDOR_DE_LINHA. Acesso em: 20 de maio de 2025.

SILVA, R. R. L. da *et al.* O uso de jogos e simulação como métodos alternativos de ensino em Engenharia no Brasil: uma revisão bibliográfica. **Revista Espacio**, v. 37, n. 5, 2016.

VISIOLI, A. **Practical PID Control**. London: Springer-Verlag, 2006.

LINE FOLLOWING ROBOT AS A TEACHING TOOL

Abstract: *This article presents the use of a line-following robot in a simulated environment as a teaching resource for Engineering education, focusing on the areas of Robotics, Electronics, Programming, and Control. Through the GearsBot platform, students can program and test control algorithms on virtual robots, understanding fundamental concepts such as sensors, conditional logic, embedded systems, and control loops. Approaches range from block programming to the use of languages such as Python, with the implementation of proportional and Proportional-Integral-Derivative (PID) controls. The results demonstrate that the use of simulation promotes active learning, stimulates autonomy, and strengthens the bridge between theory and practice. As a future proposal, the physical construction of the robot is highlighted, further enhancing the educational experience of the students.*

Keywords: *Educational Robotics, Computational Algorithm, Line Following Robot.*

