

15º Congresso de Inovação, Ciência e Tecnologia do IFSP - 2024

PROJETO DE ROBÔ SEGUIDOR DE LINHA COM VISÃO COMPUTACIONAL E CONTROLE PROPORCIONAL

Gustavo Nascimento de Sousa¹, Ricardo Pires²

¹Graduando em Engenharia Eletrônica no campus São Paulo do IFSP

sousa.nascimento@aluno.ifsp.edu.br

²Docente no campus São Paulo do IFSP

ricardo_pires@ifsp.edu.br

Rua Pedro Vicente, 625 - CEP 01109-010 - São Paulo (SP) - Brasil

Área de conhecimento (Tabela CNPq): 3.05.05.04-6 Robotização.

RESUMO: Este trabalho aborda o desenvolvimento de um robô seguidor de linha utilizando visão computacional e controle proporcional. O objetivo foi projetar e construir um robô que usasse uma câmera, em vez de sensores de infravermelho, para detectar a linha no chão e antecipar movimentos. Os métodos incluíram a seleção de componentes, a criação de software para processamento de imagens e o controle das rodas do robô, além da integração e teste do sistema. Cada imagem capturada pela câmera foi convertida para tons de cinza, filtrada com desfoque gaussiano e binarizada. Operações de dilatação e erosão foram aplicadas para melhorar os contornos da linha. O controle das rodas, baseado nas imagens processadas e utilizando controle proporcional para medir a diferença entre a posição atual da linha e a posição de referência, permitiu ajustes de direção, resultando em um robô capaz de seguir a linha com precisão. O uso da câmera apresentou resultados promissores na detecção da linha, permitindo que o robô exibisse o comportamento esperado durante os testes. Trata-se de um trabalho em andamento e embora o desempenho tenha sido qualitativamente satisfatório em vários experimentos, ainda é necessário quantificar esses resultados e compará-los aos de outros métodos.

PALAVRAS-CHAVE: visão computacional; processamento de imagens; robótica; sistemas autônomos.

LINE-FOLLOWING ROBOT PROJECT WITH COMPUTER VISION

ABSTRACT: This work focuses on developing a line-following robot using computer vision and proportional control. The goal was to design and build a robot that employs a camera, instead of infrared sensors, to detect the line on the ground and anticipate movements. The methods included component selection, software development for image processing, wheel control, and system integration and testing. Each image captured by the camera was converted to grayscale, filtered with Gaussian blur, and binarized. Dilation and erosion operations were applied to enhance the line contours. Wheel control, based on the processed images and using proportional control to measure the difference between the current line position and the reference position, allowed for directional adjustments, resulting in

a robot capable of accurately following the line. The use of the camera showed promising results in line detection, enabling the robot to exhibit the expected behavior during tests. This is an ongoing project, and while the performance has been qualitatively satisfactory in several experiments, further quantification of these results and comparison with other methods are still required.

KEYWORDS: computer vision; image processing; robotics; autonomous systems.

INTRODUÇÃO

O uso de robôs tem crescido significativamente, refletindo sua integração em processos industriais e seu uso amplo. A robótica é um campo multidisciplinar que abrange o projeto, a construção e o uso de robôs para realizar tarefas que tradicionalmente são executadas por seres humanos. De acordo com Smith (2023), esses robôs são projetados para aumentar a eficiência e a precisão em uma ampla gama de aplicações, desde a manufatura até a medicina. Na indústria automotiva, por exemplo, existem atualmente cerca de um milhão de robôs em operação ao redor do mundo (Heer, 2023), evidenciando a adoção generalizada desta tecnologia.

A visão computacional, por sua vez, é um campo da Inteligência Artificial que permite programas de computador identificar e interpretar objetos em imagens capturadas por câmeras (Gregersen, 2023). Este campo tem avançado rapidamente, oferecendo novas possibilidades para a interação entre robôs e seu ambiente.

O controle proporcional é uma técnica amplamente utilizada em sistemas de controle automático, incluindo robótica. Esse método ajusta a saída do sistema de forma proporcional ao erro observado, proporcionando uma resposta precisa e eficaz (Ogata, 2010). Em robôs, o controle proporcional pode melhorar tanto a precisão quanto a estabilidade na execução de tarefas, como o seguimento de linha, ajustando continuamente a posição do robô com base no desvio detectado.

Neste trabalho, propõe-se o projeto e a implementação de um robô que utiliza técnicas de visão computacional para a tarefa de seguimento de linha, em substituição aos sensores de infravermelho tradicionalmente empregados. Este projeto busca explorar e demonstrar a integração entre técnicas de processamento de imagens e controle no aprimoramento das capacidades de robôs em tarefas específicas.

FUNDAMENTAÇÃO TEÓRICA

Um sistema de controle realimentado busca reduzir a diferença entre a sua saída e uma entrada de referência. Neste trabalho, considera-se como referência o posicionamento de uma linha no centro horizontal de uma imagem e como saída a real posição da linha na imagem a cada instante. A diferença entre a saída e a referência, chamada de erro, é constantemente medida, para que sejam realizadas ações de controle buscando reduzi-la a zero. Dentre as ações de controle mais usadas na indústria, estão: a proporcional, a derivativa e a integral. Chamando-se a ação de controle de $m(t)$ e o erro de $e(t)$, tem-se a chamada ação de controle proporcional mais derivativa mais integral (Ogata, 2010):

$$m(t) = K_p \left(e(t) + T_d \frac{de(t)}{dt} + \frac{1}{T_i} \int_0^t e(t) dt \right) \quad (1)$$

em que os valores dos parâmetros K_p , T_d e T_i devem ser ajustados por um projetista.

Em um sistema digital, como um computador, uma imagem é representada na forma de uma matriz de *pixels* (*picture elements*) (Gonzalez; Woods, 2009). Quando uma imagem é colorida, comumente,

cada *pixel* contém três números, para representar os níveis das componentes de sua cor. Quando uma imagem é representada em tons de cinza, cada *pixel* pode ser representado por um único número, proporcional à sua claridade. Em tons de cinza, um sistema muito usado é aquele em que um *pixel* totalmente escuro é representado pelo número 0 e um *pixel* com o máximo de claridade, pelo número 255. Níveis intermediários de claridade são representados por números entre 0 e 255.

Para processar e analisar informações relevantes dessas imagens, diversas técnicas de processamento de imagens são utilizadas (Gonzalez; Woods, 2009).

Uma dessas técnicas é o desfoque gaussiano, que suaviza a imagem. Esse processo é essencial para reduzir ruídos e eliminar detalhes irrelevantes, preparando a imagem para etapas subsequentes de processamento. Após o desfoque, é comum se aplicar um limiar binário invertido à imagem. Nesse método, os *pixels* da imagem em tons de cinza são convertidos em preto ou branco com base em um valor de limiar definido. O limiar é um valor específico que determina o ponto de corte entre os tons de cinza; *pixels* com valores abaixo desse limiar são considerados uma cor (neste caso, branco), e *pixels* com valores acima são considerados outra cor (preto). No limiar invertido, essa lógica é invertida: *pixels* abaixo do limiar tornam-se brancos, enquanto os acima se tornam pretos, destacando de forma mais clara as características de interesse (Gonzalez; Woods, 2009).

Seguindo essa etapa de binarização, são aplicadas operações morfológicas como a erosão, que refina as formas na imagem ao reduzir as bordas dos objetos, eliminando ruídos e pequenos detalhes indesejados. A dilatação, uma operação inversa à erosão, pode ser utilizada em seguida para expandir os objetos e preencher lacunas, consolidando as formas e interconectando *pixels* próximos (Kaehler; Bradski, 2017).

Por fim, para melhorar a representação dos contornos dos objetos na imagem, é frequentemente utilizado o método de aproximação por polígonos. Esse método simplifica os contornos ao reduzir o número de vértices e suavizar as bordas. Essa simplificação não só facilita a análise e interpretação das imagens pelos algoritmos, mas também torna a interpretação visual mais intuitiva para os usuários (Li; Lafarge; Marlet, 2020).

MATERIAIS E MÉTODOS

O robô foi construído utilizando um chassi comumente encontrado em kits didáticos de robótica. Na parte dianteira, foi instalada uma *webcam* (Figura 1) para capturar as imagens, que são processadas por uma placa Orange Pi executando o algoritmo desenvolvido. O controle dos motores é feito por um Arduino, que aciona duas rodas, cada uma conectada a um motor de corrente contínua. Na parte traseira, o robô conta com uma roda passiva centralizada. O sistema utiliza duas fontes de energia: duas baterias recarregáveis, responsáveis por alimentar os motores e o Arduino, e um *powerbank*, que fornece energia à Orange Pi e à *webcam*.

Foram implementadas técnicas de visão computacional utilizando a biblioteca OpenCV, que oferece diversos algoritmos para análises de imagens (Kaehler; Bradski, 2017). A Figura 2 ilustra as etapas executadas pelo algoritmo em forma de fluxograma, estruturadas em três fases principais: captura de imagem, processamento da imagem e envio de comandos para os motores. Inicialmente, a imagem é capturada e processada para identificar contornos e calcular o centro de massa da linha detectada. Com base nesse cálculo, o controle proporcional ajusta a velocidade dos motores, permitindo ajustes conforme a posição da linha. Todo o processamento é realizado em uma placa Orange Pi (Orange Pi, 2024), selecionada por sua capacidade de processamento adequada para essa aplicação e por sua disponibilidade no campus.

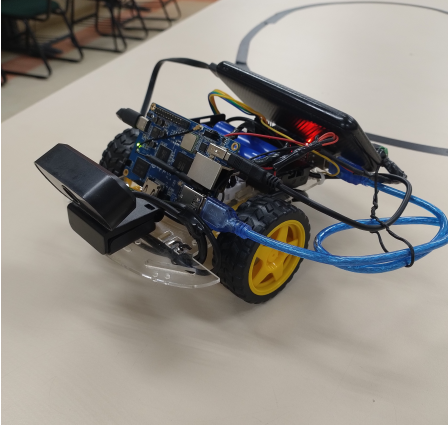


Figura 1: Robô construído para o projeto, equipado com uma *webcam* para captura de imagens e controlado por uma placa Orange Pi e um Arduino.

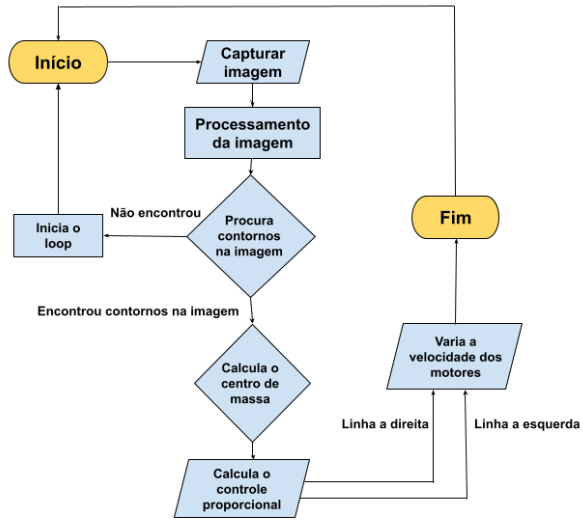


Figura 2: Fluxograma do algoritmo.

Utilizou-se uma *webcam* para obter as imagens. Na primeira etapa do processamento de cada quadro de vídeo, ocorre a conversão para escala de cinza, o que simplifica o processamento e reduz o custo computacional. Além disso, foram aplicados os filtros de desfoque gaussiano e limiar binário invertido para destacar as linhas. O desfoque gaussiano suaviza a imagem, reduzindo ruído e detalhes indesejados, enquanto o limiar binário invertido realça as linhas claras em um fundo escuro, facilitando a detecção de contornos.

Em seguida, foram aplicados filtros morfológicos de erosão e dilatação. A erosão foi utilizada para remover pequenos detalhes e ruído que poderiam comprometer a interpretação correta da imagem, enquanto a dilatação preencheu lacunas, refinando a detecção de contornos. A função *approxPolyDP* da OpenCV utiliza polígonos para suavizar os contornos detectados, proporcionando uma boa precisão tanto em segmentos retos quanto em curvas. Após essas etapas, o centro de massa do contorno aproximado foi calculado para determinar a posição da linha em relação ao centro da imagem. Testes iniciais foram realizados para verificar o funcionamento do algoritmo, ajustando parâmetros para melhorar a acurácia na detecção da linha e na navegação do robô.

Para a interação da Orange Pi com os motores do robô, foi utilizado um Arduino, uma plataforma de hardware e software de código aberto (Arduino, 2024). A comunicação entre a Orange Pi e o Arduino é realizada através da comunicação serial, permitindo que a Orange Pi envie comandos ao Arduino. O Arduino, por sua vez, controla os motores com base na posição detectada da linha pela Orange Pi. Assim, o robô é instruído a seguir em frente, virar à esquerda ou à direita, dependendo da posição da linha em relação ao centro da imagem.

O controle proporcional foi utilizado para ajustar as velocidades dos motores com base no desvio da linha em relação ao centro da imagem capturada. O erro (e) foi definido como a diferença entre a posição atual do centro de massa da linha (c_x) e uma posição de referência, que corresponde ao centro da imagem (c_{ref}). Isso pode ser expresso matematicamente como:

$$e = c_x - c_{ref} \quad (2)$$

O c_x é calculado a partir dos momentos da imagem binarizada, que representa a linha detectada. A posição de referência c_{ref} é a coordenada do centro da região de interesse (*ROI*) na imagem.

Com base nesse erro, o ajuste necessário nas velocidades dos motores foi calculado. A velocidade dos motores foi determinada por:

$$v_{esq} = v_{base} + K_p e \quad (3)$$

$$v_{dir} = v_{base} - K_p e \quad (4)$$

onde v_{esq} e v_{dir} são as velocidades dos motores esquerdo e direito, respectivamente, v_{base} é uma velocidade base definida para ambos os motores (para sair da inércia), K_p é o ganho proporcional que determina a intensidade da correção com base no erro, e é o erro definido como a diferença entre a posição atual do centro de massa da linha (c_x) e o centro da imagem (c_{ref}).

Esse ajuste faz com que, quanto maior o desvio da linha em relação ao centro da imagem, maior seja a diferença de velocidade entre os dois motores, resultando em uma correção mais significativa na trajetória do robô. Quando o desvio é pequeno, a correção aplicada é menor, resultando em um movimento mais suave. Os valores de v_{esq} e v_{dir} foram limitados e podem ser ajustados conforme necessário, por exemplo, em uma pista com mais curvas, a velocidade pode ser reduzida para melhorar a precisão do controle e evitar saídas bruscas da trajetória.

RESULTADOS E DISCUSSÃO

Durante os testes práticos realizados com o robô, observou-se que o algoritmo de visão computacional desenvolvido foi eficaz na identificação da linha. O robô foi capaz de detectar e seguir a linha de maneira satisfatória.

Na Figura 3, à esquerda, é possível verificar a imagem já processada, mas sem os filtros, com a linha identificada e as instruções de direção para o robô. À direita, vê-se o resultado do processamento da imagem, onde são realizadas as etapas de conversão para escala de cinza, seguida pela aplicação de desfoque gaussiano, limiar binário invertido e filtros morfológicos (erosão e dilatação).

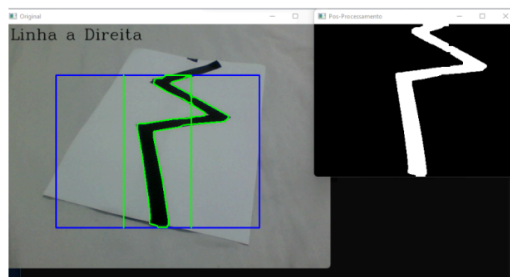


Figura 3: Exemplo do algoritmo identificando a linha.

Os testes demonstraram que o sistema de controle das rodas do robô respondeu corretamente às imagens processadas. Quando a linha estava centralizada na imagem, o robô continuava em frente. Quando a linha se desviava para a esquerda ou direita, o robô ajustava sua direção para seguir a linha. Vale destacar que este trabalho encontra-se em andamento e que novos experimentos serão realizados para a obtenção de resultados quantitativos.

CONCLUSÕES

Por meio dos testes práticos realizados, verificou-se que a visão computacional, aliada ao controle proporcional, é uma abordagem viável para o desenvolvimento de robôs seguidores de linha. O controle

proporcional ajusta a ação de controle com base na diferença entre a posição atual da linha e o centro da imagem capturada pela câmera. Esse método permite ajustes dinâmicos de acordo com a curva: se a linha está levemente desviada para a direita ou esquerda, o controle proporcional ajusta a velocidade dos motores de forma a corrigir o desvio de maneira suave. Caso a linha esteja mais fora do centro, a ação de ajuste é de maior intensidade, permitindo ao robô realizar curvas de maneira mais precisa e aumentar a velocidade em trechos retos, otimizando seu desempenho geral. No entanto, foram observadas limitações, como dificuldades em curvas fechadas, especialmente de 90 graus, e alguns desafios relacionados à interferência causada na câmera, que pode captar elementos fora da área de interesse. Essas questões destacam áreas para desenvolvimento adicional. Pretende-se, também, expandir a implementação e os experimentos para incluir uma versão com controle PID (Proporcional, Integral e Derivativo), com o objetivo de avaliar a eficácia de métodos de controle mais sofisticados.

CONTRIBUIÇÕES DOS AUTORES

Ricardo Pires contribuiu com a concepção e o escopo do estudo. Gustavo Nascimento desenvolveu a parte prática. Ambos participaram da metodologia, experimentos e redação do trabalho.

AGRADECIMENTOS

Agradecimento ao Instituto Federal de São Paulo pelo suporte e infraestrutura fornecidos, que foram essenciais para a realização deste projeto.

REFERÊNCIAS

- ARDUINO. *What is Arduino?* 2024. Disponível em: <https://www.arduino.cc/>. Acesso em: 29 ago. 2024.
- GONZALEZ, R. C.; WOODS, R. E. *Digital image processing*. Índia: Pearson Education, 2009.
- GREGERSEN, E. *Encyclopaedia Britannica, Inc. - Computer Vision*. 2023. Disponível em: <https://www.britannica.com/technology/computer-vision>. Acesso em: 29 ago. 2024.
- HEER, C. *International Federation of Robotics - One Million Robots in Car Industry Worldwide - New Record*. 2023. Disponível em: <https://ifr.org/>. Acesso em: 29 ago. 2024.
- KAEHLER, A.; BRADSKI, G. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. Sebastopol, CA, USA: O'Reilly Media, 2017.
- LI, M.; LAFARGE, F.; MARLET, R. Approximating shapes in images with low-complexity polygons. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2020. p. 8633–8641.
- OGATA, K. *Engenharia de Controle Moderno*. São Paulo (SP), Brasil: Pearson Universidades. 5ª edição, 2010.
- Orange Pi. *Orange Pi*. 2024. Disponível em: <http://www.orangepi.org/>. Acesso em: 29 ago. 2024.
- SMITH, R. J. *Encyclopaedia Britannica, Inc. - Robotics*. 2023. Disponível em: <https://www.britannica.com/technology/robotics>. Acesso em: 29 ago. 2024.