Aprimoramento do Sistema de Navegação de Robôs Autônomos Através do Uso de Sistemas Embarcados de Sensoriamento e Visão.

Odivio Caio Santos Matos[1]; Anfranserai Morais Dias[2];

1. Bolsista FAPESB, Graduando em Nome do Curso, Universidade Estadual de Feira de Santana, e-mail: odiviobzr@gmail.com

2. Orientador, Departamento de Tecnologia, Universidade Estadual de Feira de Santana, e-mail: anfranserai@ecomp.uefs.br

PALAVRAS-CHAVE: Visão computacional, Sistemas Inteligentes, Automação.

INTRODUÇÃO

O objetivo deste trabalho foi o aprimoramento do sistema de medição dos robôs de 4 rodas com a adição de uma câmera com processamento embarcado. As informações da imagem foram transmitidas para uma placa Raspberry Pi, uma Single Board Computer de baixo custo, que está integrada ao robô de 4 rodas, e tem a função de realizar operações de alto nível.

Com este aprimoramento, o sistema deve ser capaz de detectar obstáculos retangulares, de diferentes cores e formas, em um ambiente. Para isso foram utilizadas técnicas de processamento de imagens, os quais são algoritmos e métodos de cálculo numérico usados para a obtenção de informações específicas existentes na imagem. A ferramenta escolhida para o desenvolvimento do sistema de visão foi o Opencv (*Open Source Computer Vision*), que disponibiliza diversas bibliotecas de forma gratuita para serem estudas e replicadas.

MATERIAL E MÉTODOS OU METODOLOGIA (ou equivalente)

Com o objetivo de determinar qual o melhor algoritmo a ser utilizado no Raspberry Pi para a identificação de formas, foram escolhidos 3 objetos quadriláteros com diferentes cores e formas para esta análise, como o apresentado a Figura 1. A escolha das cores foi baseada no contraste com o fundo da imagem para verificar o comportamento dos algoritmos em teste.

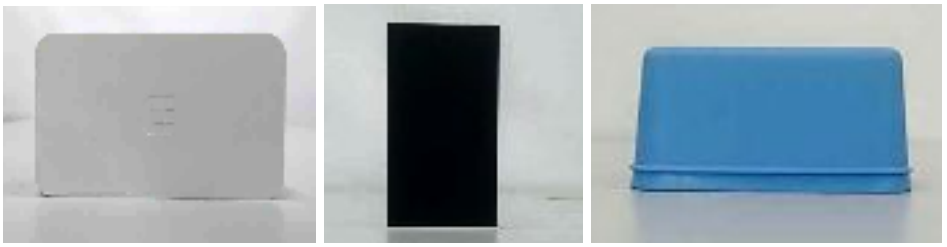

Figura 1: Obstáculos utilizados.

Foram buscados com os testes dois objetivos principais, identificar quais algoritmos possuem o melhor tempo de execução, levando em consideração o tempo de processamento da imagem, e qual apresenta a maior precisão na identificação do obstáculo. Os testes foram realizados para as resoluções de 640x480 e 1280x960,

ambas em 50% da qualidade original, visando mensurar o impacto que a dimensão da imagem representa e com isso definir a melhor resolução para o seu uso.

RESULTADOS E/OU DISCUSSÃO (ou Análise e discussão dos resultados)

Para a coleta das imagens foi escolhido o fundo branco, para evitar possíveis variações no plano de fundo fossem classificados como obstáculos, atrapalhando a análise dos resultados. A cores escolhidas para os obstáculos foram branco, preto e azul.

O primeiro resultado foi a análise do tempo médio de execução de cada método, para determinar qual o método mais veloz e estável para a utilização no Raspberry Pi. A Tabela 1 apresenta os resultados para a resolução de 640X480 e 1280x960.

Tabela 1: Resultados de tempo para resolução 640x480 e 1280x960.

Método	Média (s) 640x480	Média (s) 1280x960
Harris	0,12917	0,44055
FAST	0,03848	0,11827
SIFT	1,62430	6,05179
ORB	0,14461	0,63866

O tamanho da imagem influencia no tempo de execução de todos os algoritmos testados. Analisando as médias individuais dos algoritmos foi observado que o método FAST que apresentou um melhor desempenho, sendo 26,8% mais rápido que o segundo melhor, o método de Harris para a resolução de 640x480. Foi percebido também a diferença do tempo de execução dos métodos para as duas resoluções, onde para o método FAST o tempo aumenta em 32.53%.

O segundo resultado coletado mediu a eficácia dos algoritmos na detecção dos obstáculos nas imagens. Sendos os objetos formas quadriláteras, o objetivo dos métodos é identificar os quatro vértices do objeto e identificar suas posições na imagem, figura 2.

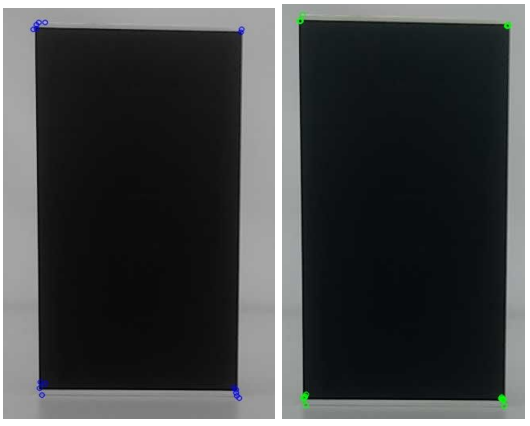

Figura 2: Resultado Objeto Preto em fundo branco para os métodos FAST e ORB.

O algoritmo com melhor taxa média de acerto foi o Harris, apresentando um desempenho de 90% de identificação nas 45 diferentes imagens analisadas, na resolução 640x480, tabela 2, e seguido dele o método FAST, que já havia apresentado um melhor desempenho em termos de velocidade. Com os resultados coletados ficou claro que o melhor método a ser utilizado foi o método FAST.

Tabela 2: Taxa média de identificação das imagens para ambas resoluções.

Método	640x480	1280x960
Harris	90%	73%
FAST	73.4%	69.95%
SIFT	66.6%	53.3%
ORB	69.8%	76.5%

Com os resultados dos testes, foi possível determinar que o método FAST, com resolução 640x480, era o mais indicado para a aplicação proposta. Assim o próximo passo foi o desenvolvimento do algoritmo de desvio, com o objetivo de detectar o obstáculo e determinar o melhor ângulo no trajeto a ser feito para que o robô desvia-se do obstáculo.

O algoritmo foi desenvolvido assumindo um comportamento reativo do robô. Uma vez detectado o obstáculo um desvio deve ser realizado. O valor resultante do algoritmo implementado é um ângulo em relação a trajetória atual, que deverá ser realizado de forma a gerar um desvio seguro em relação ao obstáculo.

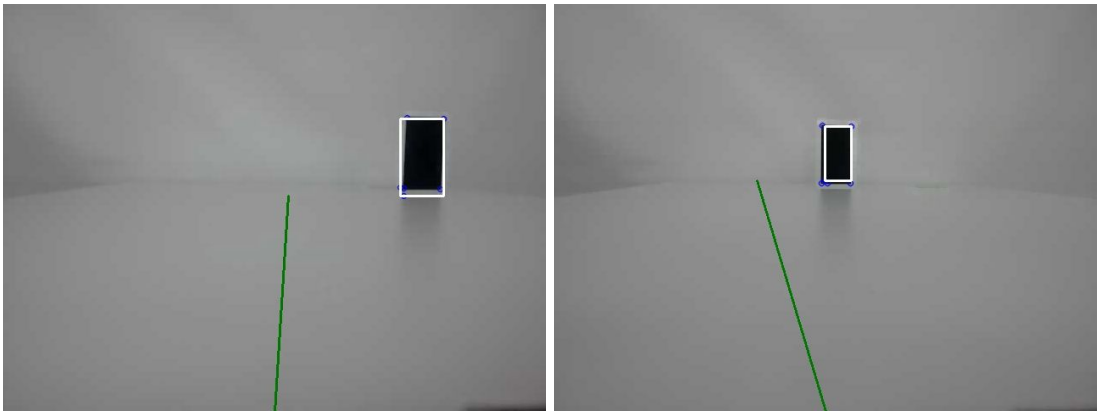

Figura 5:Representação dos Desvios Calculados.

Como eram conhecidos as dimensões dos objetos de teste, foi possível realizar o cálculo de desvio levando em consideração as proporções reais entre o obstáculo e robô. Porém em uma situação onde o tamanho do obstáculo não é conhecido, outros métodos devem ser utilizados para a determinação das duas dimensões.

CONSIDERAÇÕES FINAIS (ou Conclusão)

A abordagem escolhida para o sistema de visão computacional foram os descritores de imagens, algoritmos capazes de descrever e identificar determinadas características em imagens. Dentre as características, a identificação de cantos em imagens torna possível a detecção de obstáculos de diferentes cores e formas. A proposta de identificar e determinar os algoritmos de visão foi realizada e apresentaram resultados satisfatórios.

Contudo não possível colocar em prática a execução dos desvios de trajetória, uma vez que a integração dos algoritmos na arquitetura do robô e a realização dos testes não foram realizados. Porém com os resultados obtidos de execução dos diferentes algoritmos é possível melhor indicado-los para futuros projetos e com a obtenção de uma outra câmera, continuar as pesquisas de visão computacional.

REFERÊNCIAS

LOWE, D. G.; *Distinctive Image Features from Scale-Invariant Keypoints*. International Journal of Computer Vision 60(2), 91–110, 2004, Disponivel em: <https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>

Drummond, T.; PORTER, R.; ROSTEN, E.; *Faster and Better: A Machine Learning Approach to Corner Detection*. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, 2010. Disponivel em: <https://ieeexplore.ieee.org/document/4674368/>

STEPHENS, M.; HARRIS, C.; *A Combined Corner and Edge Detector*, Plessey Research Roke Manor, United Kingdom, 1988, Disponivel em: <https://www.cis.rit.edu/~cnspci/references/dip/feature_extraction/harris1988.pdf>

VEDALDI, A.; *Scale Invariant Feature Transform (SIFT)*. Opencv. Disponivel em: <https://docs.opencv.org/3.1.0/db/d27/tutorial_py_table_of_contents_feature2d.html> Acesso em: 2 de fervereiro de 2018.

SILVA, A. T.; *Descritores de Imagem*. 2014. Disponivel em: <www.joinville.udesc.br/portal/professores/andretavares/materiais/descritores2.pdf> Acesso em: 2 de fervereiro de 2018.

GONÇALVES, I. S.; *Desenvolvimento de um sistema reativo para desvio de obstáculos.* Dissertação(Trabalho de Conclusão de Curso) Universidade Estadual de Feira de Santana, Feira de Santana, BA, 2011.

SZELISKI, R.; *Computer Vision: Algorithms and Aplications.* Vol. 1. Springer 2010.