SOFTWARE PARA O CONTROLE DE ROBÔ MÓVEL HÍBRIDO COM PROCESSAMENTO DE IMAGEM

VITOR DOS SANTOS SILVA[1], VERA LÚCIA DA SILVA[2]

[1]Aluno do Curso Técnico Integrado ao Ensino Médio em Automação Industrial, PIVICT, IFSP, Câmpus Suzano, santos.vitor@aluno.ifsp.edu.br.
[2] Docente EBTT, IFSP, Câmpus Suzano, verals@ifsp.edu.br

Área de conhecimento (Tabela CNPq): 9.16.00.00-6 Engenharia Mecatrônica

Apresentado no
10° Congresso de Inovação, Ciência e Tecnologia do IFSP
27 e 28 de novembro de 2019 - Sorocaba-SP, Brasil

RESUMO: Robôs móveis estão sendo utilizados para a exploração de ambientes inóspitos e em atividades perigosas. Os robôs móveis são adequados para essas atividades, pois apresentam habilidades como mobilidade e autonomia para reagir a ambientes desconhecidos. A Olimpíada Brasileira de Robótica (OBR) apresenta a competição de busca e resgate, que propõe como desafios aos estudantes, o projeto e a programação de robôs móveis capazes de superar desafios similares aos que os robôs enfrentam em situações reais. Este projeto propõe o desenvolvimento dos softwares para o controle de um robô móvel híbrido, construído com *kit* de robótica Lego EV3 e o Raspberry Pi. O robô proposto deve resolver os desafios da busca e resgate da vítima, utilizando o processamento de imagem. Para isso, o robô deve agir de forma autônoma, para localizar e pegar a vítima (bolinha de isopor) e conduzi-la até uma área segura. O robô deve resgatar o maior número de vítimas possíveis. Para a implementação dos códigos de controle do robô serão utilizadas as Linguagens de Programação Java (LEJOS) para o bloco EV3 e C/C++ e a biblioteca OpenCV para o processamento de imagens no Raspberry Pi. Os softwares embarcados estarão conectados por meio de uma rede local.

PALAVRAS-CHAVE: Robótica; Sistemas Embarcados; Java; C/C++; OpenCV

HYBRID MOBILE ROBOT CONTROLLING SOFTWARE WITH IMAGE PROCESSING

ABSTRACT: Robots are being used to explore inhospitable places and on dangerous activities. The mobile robots are adequate for these activities because they display abilities such as mobility and autonomy to react in unknown environments. The Brazilian Olympiad of Robotics (OBR) provides the search and rescue competition, which proposes as a challenge to students the development of the project and programming of mobile robots capable of overcoming challenges which are similar to the ones faced by the actual robots in real situations. This project proposes the development of applications to control a hybrid mobile robot, built using the robotics kit Lego EV3 and a Raspberry Pi. The robot proposed must solve the challenges of finding and rescuing the victim using image processing. To do this, the robot ought to act in an autonomous way, to locate and retrieve the victim (Styrofoam ball) and transfer it to a safe zone. The robot must rescue the highest number of victims possible. In order to implement the controlling applications, the following programming languages will be used: Java (Lejos) on the EV3 controller and C/C++ and the OpenCV library for image processing on the Raspberry Pi. The embedded applications will be connected by a local network.

KEYWORDS: Robotics; Embedded Systems; Java; C/C++; OpenCV

INTRODUÇÃO

A Olimpíada Brasileira de Robótica é uma competição onde alunos do ensino médio devem desenvolver um robô autônomo capaz de realizar uma série de desafios para a busca e resgate de uma vítima em um ambiente inóspito, como seguir uma linha preta, fazer curvas, ultrapassar obstáculos, subir rampas, dentre outros desafios.

Os projetos de robôs utilizando apenas kits de robótica não mostram-se eficazes para a resolução de todos esses desafios. Destra forma, este trabalho propõe o desenvolvimento de uma arquitetura de robô híbrida para implementar a solução dos vários desafios propostos, visto que as soluções identificas necessitavam de *hardware* mais robusto.

Este trabalho aborda os softwares desenvolvidos para o robô híbrido projetado. O software embarcado visa utilizar-se dos sensores disponíveis no robô para apresentar uma solução lógica em que todos os desafios propostos são contemplados.

Para programar o bloco da Lego EV3 para resolver os desafios das salas 1 e 2 da competição, como seguir linha, desviar de obstáculos e a tomada de decisões de melhores caminhos, será utilizada a Linguagem de Programação Java, através da plataforma LEJOS EV3 (BRITO, GALON, 2016). Dentre os sensores utilizados estão sensores de refletância, cor e ultrassônico, além de uma câmera.

No entanto, uma parte em específico do desafio, a busca e o resgate de vítimas, conhecido como Sala 3, provou-se uma grande dificuldade para os alunos. Nela, o robô deve explorar o ambiente e "resgatar vítimas sem interferência humana" (OBR, 2019). A técnica escolhida para resolver esse desafio é o processamento de imagem, utilizando a Biblioteca OpenCV (PASSARELLI, 2017).

Esse estudo envolveu a utilização de microcontroladores e computadores de placa única, como a Raspberry Pi, além de utilizar o bloco de robótica da empresa Lego, EV3. Para a programação foram utilizadas as linguagens C/C++ e Java.

MATERIAL E MÉTODOS

Materiais

Foi utilizado um controlador Lego EV3 para o controle dos atuadores do robô, uma Raspberry Pi para processamento de imagem e um celular *smartphone* para obtenção das imagens. Para alimentação da Raspberry Pi utilizou-se um *power bank* com capacidade de 10.000mAh. Para a programação foram utilizadas as linguagens de programação C/C++ e Java, a plataforma LEJOS (BRITO, GALON, 2016) e a biblioteca de visão computacional OpenCV (PASSARELLI, 2017). Para a programação foi utilizada a IDE Eclipse (VINICIUS, 2009) devido à facilidade de sua configuração e adequação ao projeto.

Para o desenvolvimento do projeto, foi utilizada a metodologia seguinte, resultando em softwares embarcados no robô para resolver os desafios propostos pela OBR.

Controlador para o robô seguir a linha

Uma das tarefas do robô é seguir uma linha preta em uma superfície de MDF branca, construída por fita isolante. Para que o robô consiga seguir de forma apropriada a linha, sem se perder no percurso, utilizou-se um controlador que possui como variável de processo a diferença entre dois sensores de refletância posicionados lado a lado na parte inferior do robô, para visualizar a linha preta. De acordo com os valores lidos, o processo controla a variável com velocidade dos dois motores. A diferença da leitura dos dois sensores é calculada, em seguida passa por um controlador, e então o valor de saída do controlador é aplicado na velocidade dos motores.

Dois controladores foram testados, o controlador PID (Proporcional Integral Derivativo) e o controlador por lógica Fuzzy. Ambos foram utilizados por um tempo, porém concluiu-se que o controlador PID seria o mais adequado para o projeto. O software desenvolvido em LEJOS foi embarcado no EV3 para controlar o robô seguidor de linha.

Localização da vítima

As vítimas localizadas na Sala 3 consistem em esferas prateadas ou pretas com aproximadamente 5 cm de diâmetro. Para a localização das vítimas utilizou-se o processamento de imagem, que lê uma imagem da câmera do celular, a envia para a Raspberry Pi, que a processa,

passando por diversas camadas de filtros que diminuem o ruído, e enviando a posição do centro da esfera e seu raio para o controlador EV3, para que esse possa centralizar a bolinha e capturá-la.

A imagem recebida é convertida para o esquema de cores preto e branco, depois passa por um desfoque para diminuir seu ruído, e então passa por um algoritmo chamado Transformada de Hough para Círculos, contido na biblioteca OpenCV (PASSARELLI, 2017). A Figura 1 ilustra o resultado da identificação da vítima pela técnica de processamento de imagem.

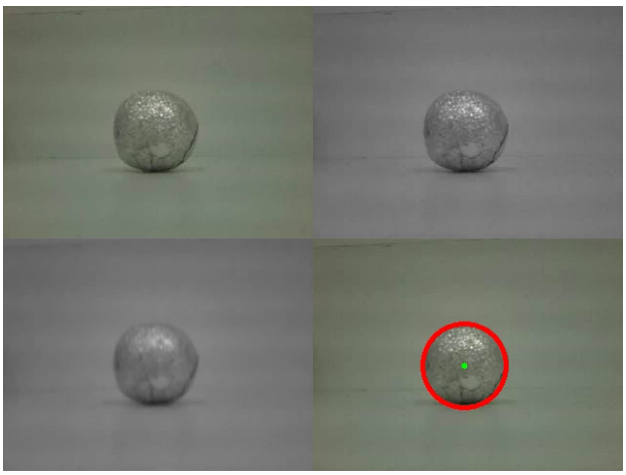

Figura 1: Localização da vítima

Localização da zona de resgate

A zona de resgate é um triângulo preto em um dos cantos da sala 3, no qual o robô deve depositar as vítimas que resgatar. O robô identifica a zona de resgate como um retângulo preto. Para sua localização a imagem é passada por um desfoque para reduzir ruído, então é aplicada uma operação de *thresholding* (limiarização). A imagem de entrada gera uma nova imagem na qual os *pixels* dentro de uma faixa de cores são transformados em brancos, e todos os outros ficam pretos. Essa nova imagem passa por funções da biblioteca OpenCV, que localiza as formas resultantes. Assim como ocorreu com a esfera, a identificação do centro do retângulo é enviado ao controlador Lego EV3. A Figura 2 ilustra o processo de identificação do centro da zona de resgate.

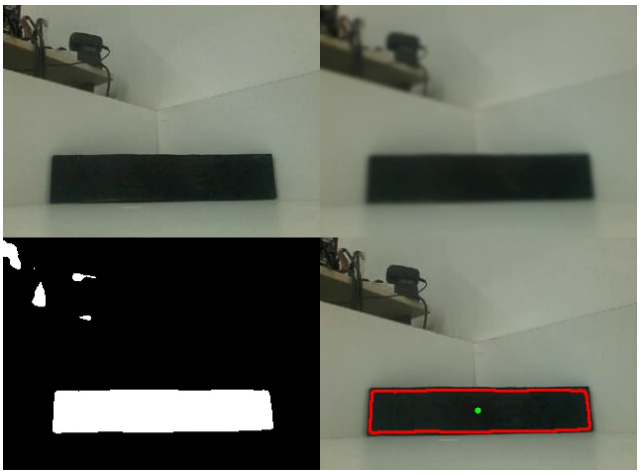

Figura 2: Localização da zona de resgate

Algoritmo Genético

Houve uma tentativa de utilização de um algoritmo genético para a sintonia do processamento de imagem. Para definir os valores que o processamento de imagem deveria retornar, e através dos quais o algoritmo genético avaliaria o desempenho de cada indivíduo, foram capturadas 120 fotos que simbolizavam situações reais, as quais o algoritmo poderia encontrar na competição. A Figura 3 ilustra algumas dessas fotos.

Figura 3: Situações possíveis

Após capturadas essas fotos, foram definidos manualmente os parâmetros que o algoritmo de processamento de imagem deveria retornar. Desse modo, o algoritmo genético poderia identificar o quão distante do valor desejado, o retorno do algoritmo de processamento de imagem estava. As fotos eram utilizadas para que o algoritmo genético pudesse avaliar o desempenho dos indivíduos através da comparação do resultado do algoritmo de processamento de imagem, utilizando-se dos parâmetros daquele indivíduo com o resultado que tinha sido definido como certo manualmente. Na Figura 4 pode-se ver as situações que foram manualmente identificadas como o resultado esperado do algoritmo de processamento de imagem.

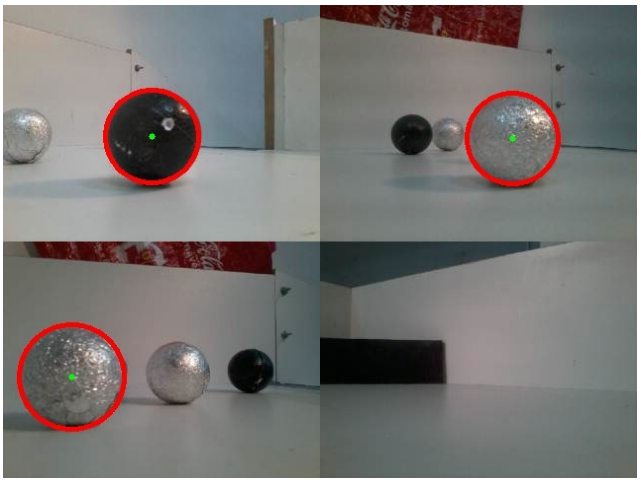

Figura 4: Situações manualmente identificadas

RESULTADOS E DISCUSSÃO

O software desenvolvido já é capaz de localizar as vítimas prateadas e pretas corretamente, no entanto ainda há a necessidade de calibração para situações nas quais mais de uma vítima pode ser visualizada na imagem. A garra do robô então captura a vítima, e é capaz de guardá-la na zona de resgate.

O processamento de imagem ainda pode ser melhorado, para que haja um maior contraste entre o fundo e as vítimas prateadas.

A parte do software responsável por resolver as primeiras salas do desafio possui métodos funcionais, que, no entanto, estão bem suscetíveis a mudanças devido a diferentes condições de ambiente. Com a devida calibração, os métodos inclusos devem resolver os desafios sem grandes problemas. Na Figura 5 pode-se ver o robô utilizado, com a garra e o celular cuja câmera foi necessária para o processamento de imagem.

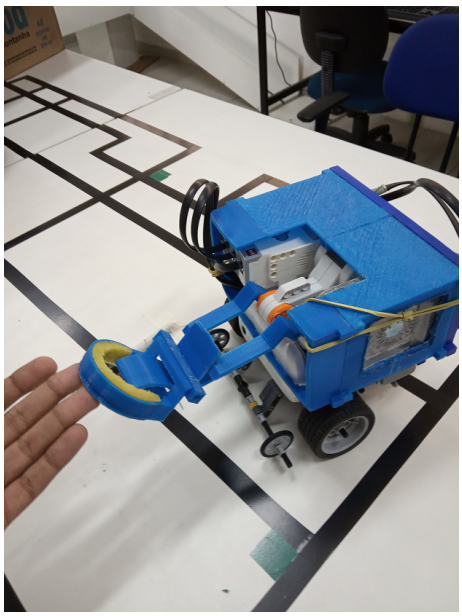

Figura 5: Robô utilizado

CONCLUSÕES

Pôde-se concluir que sistemas dinâmicos, como os sistemas de controle de um robô móvel, possuem muitas variáveis que devem ser levadas em conta, e podem alterar a resposta do algoritmo, mesmo em condições controladas em laboratório. O contraste do fundo da sala com a vítima por exemplo, se tornou um grande dificultador, levando em conta que o algoritmo deve aplicar um desfoque para reduzir ruído, o que acaba gerando perda de contraste, dificultando a identificação correta de vítimas com cor parecida ao fundo.

Muitas variáveis referentes à estrutura do robô também dificultam o cálculo do deslocamento necessário baseado nos valores de retorno dos sensores ao algoritmo. Quando existe um maior atrito por exemplo, pode interferir no deslocamento do robô, que não responde aos comandos imediatamente com os resultados esperados. Tais variáveis, no entanto, se encontram fora do escopo desse trabalho, porém interferem diretamente no desenvolvimento dos softwares de controle do robô.

AGRADECIMENTOS

Agradeço primeiramente a Deus e aos meus pais pelo apoio no desenvolvimento desse projeto, que embora seja muito trabalhoso, como qualquer projeto de pesquisa, tem possibilitado o meu crescimento pessoal e a possibilidade de contribuição à comunidade científica e aos futuros estudantes que se deparem com problemas parecidos.

Agradeço também a todos os meus professores, mas em especial aos professores Vera Lúcia e Wagner Garo que têm prestado uma assistência mais próxima.

REFERÊNCIAS

OBR, Modalidade Prática. Manual de Regras e Instruções, Etapa Regional/Estadual, 2019. Disponível em: <http://www.obr.org.br/manuais/OBR2019_MP_ManualRegionalEstadual.pdf>. Acesso em: 01 set. 2019.

BRITO, R.C.; GALON, H.E. Introdução aos ambientes de programação NXT-G e leJOS para o Lego Mindstorms: 23. ed. Paraná: Editora UTFPR, 2016.

PASSARELLI, Leandro. Aplicação de visão computacional com OpenCV. 2017. Embarcados. Disponível em: <https://www.embarcados.com.br/aplicacao-de-visao-computacional-com-opencv>. Acesso em: 19 jul. 2019.

VINICIUS, Caio. Trabalhando com a IDE Eclipse. 2009. Oficina da Net. Disponível em: <https://www.oficinadanet.com.br/artigo/1571/trabalhando_com_a_ide_eclipse>. Acesso em: 19 jul. 2019.