

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO
RIO GRANDE DO NORTE - IFRN
CAMPUS SANTA CRUZ**

BYAKUGAN: VISÃO COMPUTACIONAL

DOUGLAS GABRIEL DA SILVA ARAÚJO
ISAAC MARLON DA SILVA LOURENÇO
PAULO VITOR LIMA BORGES

SANTA CRUZ/RN

2020

DOUGLAS GABRIEL DA SILVA ARAÚJO
ISAAC MARLON DA SILVA LOURENÇO
PAULO VITOR LIMA BORGES

BYAKUGAN: VISÃO COMPUTACIONAL

Relatório de pesquisa apresentado à coordenação do curso técnico em Informática e coordenação de pesquisa e inovação do *campus* Santa Cruz como requisito parcial para aprovação e finalização do Trabalho de Conclusão do Curso.

Orientador: Prof. Me. Lennedy Campos Soares

SANTA CRUZ/RN

2020

DOUGLAS GABRIEL DA SILVA ARAÚJO
ISAAC MARLON DA SILVA LOURENÇO
PAULO VITOR LIMA BORGES

BYAKUGAN: VISÃO COMPUTACIONAL

Relatório de pesquisa apresentado à coordenação do curso técnico em Informática e coordenação de pesquisa e inovação do *campus* Santa Cruz como requisito parcial para aprovação e finalização do Trabalho de Conclusão do Curso.

Aprovado em: ___ de _____ de 2020.

BANCA EXAMINADORA

Prof. Me. Lennedy Campos Soares

Orientador – IFRN

Prof. Me. Ericson Norberto Alves Santos

IFRN

Prof. Dr. Paulo Augusto de Lima Filho

IFRN

Prof. Dr. Dênis Keuton Alves

IFRN

AGRADECIMENTOS

Gostaríamos de agradecer ao IFRN, por ajudar no desenvolvimento pessoal e profissional de milhares de pessoas e por acreditar que os robôs desenvolvidos em conjunto pela comunidade acadêmica são de extrema importância para a expansão do uso de tecnologias educacionais. Agradecemos aos nossos familiares, apoiadores, à Olimpíada Brasileira de Robótica e à Mostra Nacional de Robótica.

RESUMO

A Olimpíada Brasileira de Robótica (OBR) propõe que seus participantes desenvolvam robôs autônomos capazes de superar desafios em uma emulação de cenário de desastre. Com a intenção de resolver o desafio específico de resgate de vítimas, este trabalho propõe tecnologias de reconhecimento e tratamento de imagem, uma forma alternativa ao que geralmente é utilizado por outras equipes participantes. Foi adicionada uma câmera em um robô (previamente desenvolvido) com intuito de capturar imagens do ambiente onde são dispostas as vítimas e uma área para resgate. A proposta também utilizou o *Robotic Operating System* (ROS), o que permitiu a realização de algumas tarefas, tais como: a execução de programas do robô de forma distribuída em várias máquinas, a utilização de outros sensores além da câmera, o acionamento de atuadores presentes no robô e o processamento das imagens, feito com a biblioteca *OpenCV*. Os testes realizados indicam resultados promissores na utilização das tecnologias propostas por este trabalho.

Palavras-chave: Competição, Educação, Robótica, Visão Computacional

ABSTRACT

The Brazilian Robotics Olympiad (OBR – Olimpíada Brasileira de Robótica), proposes their participants to develop robots capable to overcome challenges in a disaster scenario autonomously. This present work proposes an implementation of a computer vision system as an alternative way to those proposed by the OBR's challenge. This work added a camera to a robot (previously developed), aiming visual capture of the environment in which the victims and the safe area are disposed of. Then, was used the Robot Operating System (ROS), that allowed the accomplishment of some tasks like: run the robot programs in a distributed way on several machines, the utilization of some others sensors than the camera, the actuation of the actuators on the robot and the image processing, with the OpenCV library. Realized tests indicate promising results in the use of the technologies proposed by this work.

Keywords: Competition, Computer Vision, Education, Robotics

SUMÁRIO

1 INTRODUÇÃO	8
1.1 OBJETIVO GERAL	9
1.2 OBJETIVOS ESPECÍFICOS	9
1.3 ORGANIZAÇÃO DO TRABALHO	10
2 FUNDAMENTAÇÃO TEÓRICA	11
3 METODOLOGIA	13
3.1 CONHECENDO AS FERRAMENTAS	13
3.2 PLATAFORMA ROBÓTICA	14
3.3 PROCESSAMENTO DE IMAGENS	15
3.4 ETAPA FINAL	17
4 RESULTADOS E DISCUSSÕES	18
4.1 IDENTIFICAÇÃO DOS OBJETOS	18
5 CONCLUSÃO	20
REFERÊNCIAS	21

1 INTRODUÇÃO

Sob um ponto de vista educacional, a robótica é uma tecnologia emergente que gera interessantes experiências. Essas experiências demonstram como a robótica é um agente inclusivo e que traz bons resultados em projetos coletivos que utilizam tecnologias para soluções de problemas. (OBR, 2018)

Dentro desse contexto, um importante agente de desenvolvimento da robótica no Brasil é a Olimpíada Brasileira de Robótica (OBR), organização que realiza competições desde 2007 em vários lugares do país. Acreditando que a robótica é fundamental para a expansão tecnológica, a organização incentiva jovens brasileiros a desenvolverem robôs autônomos que consigam superar desafios em uma emulação de um cenário de desastre. O primeiro desses desafios é composto por um caminho previamente estabelecido que o robô deve seguir. Esse caminho pode conter alguns empecilhos, como: redutores de velocidade, obstáculos e encruzilhadas (Figura 1). Logo após seguir esse caminho, o robô deverá encontrar, pegar e salvar as vítimas em um ambiente chamado “sala de resgate”. É nesse local que o robô deve capturar as vítimas que são representadas por bolas (prateadas ou pretas, vivas ou mortas, respectivamente) e em seguida deixá-las na “área de resgate”, representado por um triângulo preto (Figura 2).

Figura 1 – A arena utilizada nas etapas regionais e estaduais da OBR



Fonte: OBR (2019)

Figura 2 – Sala de resgate



Fonte: ARAÚJO (2019)

1.1 OBJETIVO GERAL

O desenvolvimento deste trabalho tem como objetivo geral contribuir para a robótica do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte – *Campus* Santa Cruz (IFRN-SC), pois permitirá que novos alunos desenvolvam projetos baseados em visão computacional, promovendo assim, um avanço no desenvolvimento dos robôs utilizados pelo Laboratório de Robótica do IFRN-SC (LARIS) em competições como a OBR.

1.2 OBJETIVOS ESPECÍFICOS

No contexto da competição da OBR, os trabalhos de Marinato *et al* (2017) e Amorim *et al* (2018) desenvolveram projetos de robôs autônomos baseados em técnicas de visão computacional para realizar os desafios da competição. Este trabalho baseia-se nesses projetos e desenvolve uma proposta semelhante, tendo como objetivos específicos:

1. Utilizar a visão computacional para resolver o desafio do resgate de vítimas com uma maior precisão na identificação dos objetos (vítimas e áreas de resgate);
2. Solucionar alguns problemas encontrados nos trabalhos anteriores, como: a identificação de vítimas de cores distintas e o reconhecimento da área de resgate;
3. Utilizar o *Robot Operating System* (ROS, Sistema Operacional para Robôs), ferramenta que permitirá controlar o ambiente de execução do robô de forma remota, acompanhar os processos em tempo real e visualizar erros no momento de execução.

1.3 ORGANIZAÇÃO DO TRABALHO

Este trabalho é constituído por cinco capítulos, que são:

- Capítulo 1: Apresenta-se um breve panorama das competições de robótica no Brasil e os objetivos para o desenvolvimento deste trabalho;
- Capítulo 2: São apresentadas as ideias de alguns autores assim como suas relações com o desenvolvimento do presente trabalho;
- Capítulo 3: Apresentam-se a metodologia de desenvolvimento, as tecnologias envolvidas e a modelagem do sistema;
- Capítulo 4: São apresentadas as discussões dos resultados alcançados;
- Capítulo 5: Apresentam-se as conclusões do trabalho, os pontos positivos, negativos e sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Tendo em vista o avanço tecnológico que a visão computacional trará para o LARIS, é interessante determinar quais componentes são necessários para desenvolver um robô com essa tecnologia.

Dentro do contexto do desenvolvimento de robôs que utilizam visão computacional, Marinato *et al* (2017) e Amorim *et al* (2018) construíram robôs priorizando a estrutura física e lógica em função da utilização de uma câmera, tendo como componentes: motores, servomotores para a garra do robô, as plataformas *Arduino* para o controle dos atuadores e o *Raspberry Pi* para o tratamento de imagens. Tais componentes permitiram comprovar a capacidade de adequação e funcionamento do robô em conjunto com a aplicação de um sistema de visão computacional.

Além da utilização dos componentes descritos, o trabalho de Amorim *et al* (2018) apresenta como os componentes interagem entre si. O primeiro dos componentes, *Raspberry Pi*, ficou responsável pelo tratamento de imagens e identificação de objetos. Após o resultado do tratamento, o *Raspberry Pi* informa para os outros componentes o que deve ser feito. Essa comunicação é realizada através de comandos baseados em caracteres de 8 bits.

Enquanto o *Raspberry Pi* fica responsável pelo processamento das imagens, o *Arduino* fica encarregado de acionar os atuadores do robô de acordo com comandos que venham do *Raspberry Pi*. Existem várias funções preparadas no *Arduino* para serem executadas a partir de comandos, como por exemplo, as funções “*viraDireita()*” ou “*emFrente()*”, que significaria, respectivamente, acionar os motores para virar a direita ou acioná-los para seguir em frente.

A partir dessa necessidade de comunicação entre o *Raspberry Pi* e o *Arduino*, ambos os autores, optaram por conectá-los através da porta serial, tendo em vista que as plataformas utilizadas possuem suporte para esse tipo de comunicação.

O trabalho de Marinato *et al* (2017), ao contrário do de Amorim *et al* (2018), priorizou a explicação do processamento das imagens. Esse processamento deu-se através da diferença de cores dos objetos com relação ao fundo da imagem. O que implicou no uso da cor especificamente vermelha, pois essa possui um maior contraste com o ambiente.

No tratamento de imagens, os autores optaram por utilizar a biblioteca livre *OpenCV*. Essa biblioteca possui vários métodos para processamento das imagens, entregando conjuntos de informações importantes para a identificação dos objetos (OPENCV, 2018). Esses métodos permitem, em outras palavras, a identificação de cores e formas geométricas, o que é necessário para identificar vítimas e a área de resgate na competição da OBR.

Analisando o trabalho de ambos os autores, observa-se que existe uma complexidade inerente ao uso das duas plataformas microcontroladas como o *Raspberry Pi* e o *Arduino*. Além disso, também há uma dificuldade com relação ao tratamento de imagens, que se soma o fato de estar sendo desenvolvida em um sistema embarcado. Em um sistema robótica relativamente complexo como esse, há a necessidade de organizar e monitorar, de forma mais ampla, os processos em execução.

Para tentar solucionar os problemas apresentados anteriormente, o ROS, uma biblioteca computacional, permite organizar, monitorar e programar de maneira flexível, um robô. Essa biblioteca também é considerada um *middleware*, criando uma interface entre o Sistema Operacional e *hardware* do robô (EDNO, 2014).

Além disso, o ROS possui uma coleção de ferramentas essenciais para o desenvolvimento de aplicações na área da robótica, dispondo de vantagens, como: visualização em tempo real de execução de programas em seu ambiente, visualização de erros do robô no *log* do sistema e a modularização dos programas representados por “nós”, permitindo também, a distribuição de programas em várias máquinas. Em seu funcionamento, dois conceitos são primordiais para sua compreensão: “nós” e os “tópicos”. Os “nós”, como citados anteriormente, são representações de programas, o que permite a divisão das tarefas essenciais para a execução do robô e, conseqüentemente, um melhor controle no

desenvolvimento de algumas funcionalidades. Já com relação ao segundo conceito, os “tópicos” são interfaces que permitem a troca de informações entre vários “nós”, o que permite utilizar tipos primitivos de informações, como: *Integers* (números inteiros) ou *String* (conjunto de caracteres). Além disso, o ROS também permite a criação de tipos personalizados de dados.

3 METODOLOGIA

A possibilidade de que um robô autônomo com processamento de imagens pudesse ser uma forma eficiente de resolver o desafio de emulação de resgate de vítimas proposto pela OBR, despertou a ideia de acoplar em uma plataforma robótica um sistema de visão computacional. Esse sistema deve ser capaz de auxiliar o robô na localização e resgate de objetos (vítimas e área para resgate) na sala de resgate.

Inicialmente, o grupo estudou as ferramentas que seriam utilizadas: *OpenCV*, importante para o tratamento e processamento das imagens, e o ROS, para permitir a interação do Sistema Operacional e o *hardware* do robô.

3.1 CONHECENDO AS FERRAMENTAS

Depois do estudo das ferramentas, a primeira etapa de desenvolvimento deste trabalho consistiu em desenvolver dois testes para validar se a proposta do trabalho seria viável. O primeiro teste foi controlar um robô a partir do reconhecimento da movimentação de mãos humanas, utilizando o *OpenCV*. O segundo, controlar um robô remotamente por um *joystick*, com o objetivo de obter experiência no uso do ROS.

O código desenvolvido no primeiro teste utilizou a biblioteca de visão computacional *OpenCV*, o que permitiu aplicar as funções para a mudança de cor da imagem (*cv2.cvtColor()*) e a identificação dos contornos da mão (*cv2.findContours()*). Os códigos

desenvolvidos com essas funções comprovaram o poder computacional da biblioteca no uso de programas para a identificação de objetos em tempo real.

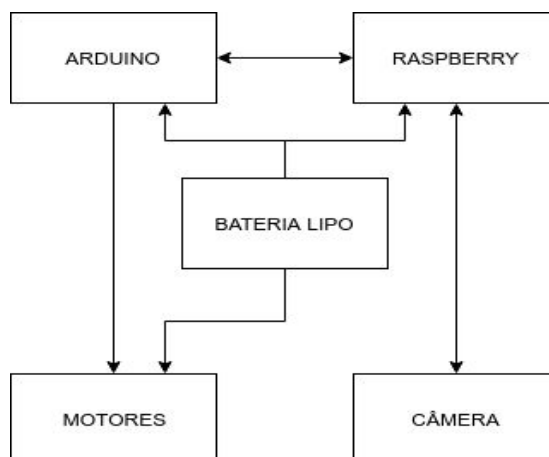
Já o segundo teste permitiu identificar que o ROS é uma opção factível para ser utilizada no robô que o trabalho pretendia utilizar. Foram desenvolvidos alguns “nós” para a comunicação entre duas máquinas, permitindo utilizar programas de forma distribuída. Além disso, foi possível, de forma remota e em tempo real, visualizar a imagem capturada pela câmera do robô disponibilizada por um “nó” no ambiente ROS.

3.2 PLATAFORMA ROBÓTICA

Logo após a realização dos estudos da seção anterior, a próxima etapa foi a aplicação dos conhecimentos adquiridos em uma plataforma robótica. A plataforma escolhida foi o robô desenvolvido pela equipe Wall-E, do LARIS, pois o mesmo já possuía as características necessários para a realização dos desafios da OBR (ARAÚJO, 2019, no prelo). Além de usufruir dessas características, foi possível adequar o robô para as necessidades de um robô que realiza processamento de imagem, similar aos trabalhos de Amorim *et al* (2018) e Marinato *et al* (2017).

Na Figura 3, é possível observar a escolha de dois componentes principais: o *Raspberry Pi* e o *Arduino*. Por questões econômicas e simplicidade, em geral, é comum que equipes utilizem somente plataformas como o *Arduino* atuando como o núcleo de todo o processamento do robô. Entretanto, para que este trabalho pudesse usufruir dos processos de visão computacional, o *Arduino* não seria uma boa opção, pois não possui poder computacional suficiente. Portanto, este trabalho optou em utilizar a plataforma *Raspberry Pi* para cuidar do processamento das imagens, já que mostra eficiência nessas tarefas, como mostra Senthilkuma *et al* (2014). Além disso, o *Raspberry Pi* possui capacidade suficiente para processar, além das imagens, vários outros dados dos componentes eletrônicos do robô.

Figura 3 – Arquitetura de *hardware* simplificada do robô



Fonte: ARAÚJO (2019)

Ainda na Figura 3, podemos observar a existência de uma bateria de polímero de lítio (LIPO) de 2,2Ah (ampère-hora), alimentando o robô por cerca de duas horas. Além disso, foi adicionada uma câmera para adquirir imagens do ambiente, e finalmente, também foi utilizado dois motores para o robô.

O controle dos motores é feito a partir do *Raspberry Pi*. No entanto, não existe comunicação direta entre os dois: o comando de movimentação, inicialmente parte do *Raspberry Pi* e em seguida é enviado para o *Arduino*. Logo após essa intermediação, o *Arduino* consegue acionar os motores.

Para que seja possível a utilização do *Arduino* e conjunto com os motores, o presente trabalho optou por utilizar uma biblioteca desenvolvida pelo LARIS, denominada “*robo_hardware*” (SOARES, 2018). Essa biblioteca possui vários módulos para abstrair a comunicação de componentes de robôs, como: motores, sensores de distância, sensores de refletância etc.

Além do “*robo_hardware*”, para permitir a comunicação entre o *Arduino* e o *Raspberry Pi* de forma mais eficiente, é utilizada a biblioteca “*rosserial*” (REIGNIER, 2018). A vantagem do uso dessa biblioteca é a possibilidade de integrar o *Arduino* com o ambiente de execução do ROS. Entretanto, o seu uso consome uma quantidade grande de memória do

Arduino. Logo, este trabalho optou por realizar todas as funções embarcadas de movimentação e controle do robô na plataforma do *Raspberry Pi*. O *Arduino* então, recebe somente pares de informações para o controle de motores (vetores compostos por velocidade do motor esquerdo/direito) e controle da garra do robô (vetores compostos por posição do braço/mão). Essa estratégia de execução assegura um bom funcionamento das plataformas e conseqüentemente, do robô.

3.3 PROCESSAMENTO DE IMAGENS

Na terceira fase de desenvolvimento deste projeto, o grupo dividiu as tarefas em duas etapas: identificar as vítimas (bolas) e identificar a área de resgate (retângulo, na visão do robô). Desenvolvidas essas identificações, foi possível diferenciar as cores das vítimas (pretas e prateadas) e identificar a área de resgate em diferentes posições do robô na sala de resgate.

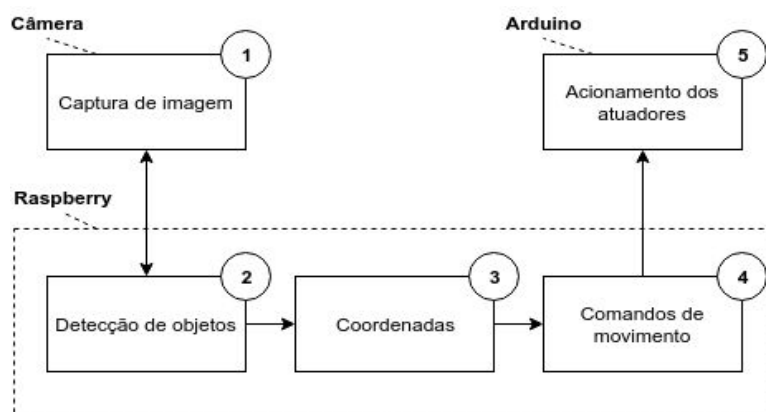
Na identificação das vítimas, foi utilizada uma função do *OpenCV* chamada *cv2.houghCircle()*. Essa função retornava as informações necessárias para o tratamento e tomada de decisões a partir dos dados adquiridos. Para tal, a função utiliza de uma adaptação do método matemático (Transformada de *Hough*), cujo detalhamento foi abstraído pela biblioteca. Essa abstração permite o foco no tratamento específico das coordenadas, cuja composição é formada pelo centro do círculo (x e y) e seu raio. A partir disso, é possível obter a distância e a posição do robô com relação à vítima.

Já com relação a identificação área de resgate, foram utilizadas outras funções do *OpenCV*, como: *cv2.cvtColor()* para modificar a cor da imagem para cinza; *cv2.threshold()* para transformação da imagem cinza em apenas duas tonalidades (preto e branco), resultando em uma imagem “binarizada”; *cv2.findContours()* para obter as coordenadas dos contornos identificados na imagem “binarizada”. A partir de todas dessas etapas sequenciais e preparatórias, as coordenadas resultantes do *cv2.findContours()* eram interpretadas em vários métodos de tratamento desenvolvidos por este trabalho. Os métodos conseguem distinguir um objeto retangular grande (objeto esperado na identificação da área de resgate) de pequenos contornos resultantes de ruídos na imagem. Por fim, depois de interpretar um conjunto de

coordenadas como sendo a área de resgate, o sistema gera a coordenada central do objeto e a disponibiliza em um “tópico” no ambiente ROS.

Na penúltima fase de desenvolvimento, o grupo conseguiu criar a comunicação necessária entre o *Raspberry Pi* e o *Arduino*, permitindo que o *Raspberry Pi*, em conjunto com o *Arduino*, controle os atuadores e colete informações dos sensores do robô. Essa etapa foi importante para comprovar o poder na estratégia de divisão das tarefas do robô nas duas plataformas: o *Raspberry Pi* ficou responsável pelo processamento todas as informações, enquanto o *Arduino*, atuando como “servo” do *Raspberry Pi*, recebia apenas os comandos de movimentação. A Figura 4 mostra como essas plataformas agem em conjunto.

Figura 4 – Arquitetura lógica do processamento de imagens



Fonte: ARAÚJO (2019)

O processo começa quando o *Raspberry Pi* solicita uma imagem da câmera do robô e a disponibiliza para a próxima etapa. Na seção de detecção de objetos, são utilizadas as funções da biblioteca de visão computacional *OpenCV*. É nessa etapa que imagem alterada é “binarizada”, processo responsável pela transformação das tonalidades em apenas duas cores: preto e branco. Com a imagem “binarizada”, o processo de identificação de objetos se torna menos complexo e mais rápido. Dos contornos identificados nas imagens “binarizadas”, são extraídas as coordenadas importantes para este trabalho: o centro dos objetos e suas posições

na imagem. A partir das coordenadas, o *Raspberry Pi* finalmente consegue enviar os comandos de movimentação para o *Arduino*.

3.4 ETAPA FINAL

Na última fase de desenvolvimento, o grupo desenvolveu uma estratégia para que o robô conseguisse, da melhor maneira possível, executar o desafio do resgate das vítimas. A estratégia inicial foi baseada no trabalho de Barbosa *et al* (2016); entretanto, a complexidade desse desenvolvimento surpreendeu as expectativas e apresentou uma dificuldade para a finalização deste projeto.

4 RESULTADOS E DISCUSSÕES

Nas fases iniciais de desenvolvimento, o uso da visão computacional no robô mostrou-se possível devido o processamento das imagens em tempo real e em paralelo com o uso de motores. Entretanto, em alguns casos foram notados altos níveis de processamento do *Raspberry Pi*, o que evidenciou um ponto crucial na utilização da visão computacional: as imagens deviam ser de baixa resolução (320x240 *pixels*) para que fosse possível uma boa taxa de processamento.

4.1 IDENTIFICAÇÃO DOS OBJETOS

No contexto da identificação de objetos utilizando visão computacional, Marinato *et al* (2017) e Amorim *et al* (2018) fazem uso de vítimas de coloração avermelhada pela facilidade de tratamento, o que evidencia a dificuldade de reconhecimento de vítimas prateadas, cores de pouco contraste com a sala de resgate. O presente trabalho conseguiu a identificação de vítimas com as cores prateadas e pretas. Para observar o percentual de acerto da identificação, foram realizados testes (ARAÚJO, 2019, no prelo). De acordo com esses testes, as vítimas pretas são mais fáceis de identificar do que as prateadas, provavelmente devido ao contraste entre o branco e preto. A tabela 1 mostra os resultados específicos do experimento de identificação.

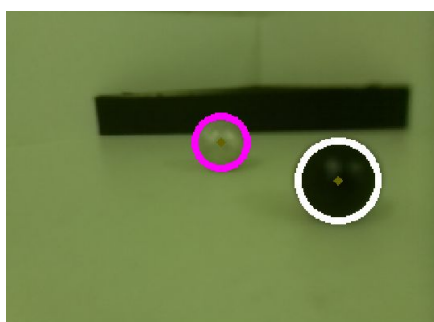
Tabela 1 – Percentual de identificação dos objetos

Cor	Percentual
Prateada	28%
Preta	76%

Fonte: ARAÚJO (2019)

Apesar da aparente facilidade de identificação das vítimas mortas, essas podem se tornar quase que impossíveis de se detectar quando estão em uma mesma perspectiva da área de resgate (de cor totalmente preta), o que não acontece com as vítimas brancas, que ficam muito evidentes com relação às tonalidades escuras. Também foi notada uma facilidade de identificação quando as vítimas estavam mais próximas da câmera, o que permite uma discussão futura sobre qual seria a distância ideal para identificar as vítimas. A Figura 5 mostra a identificação das vítimas na visão do robô.

Figura 5 - Identificação das vítimas prateadas e pretas

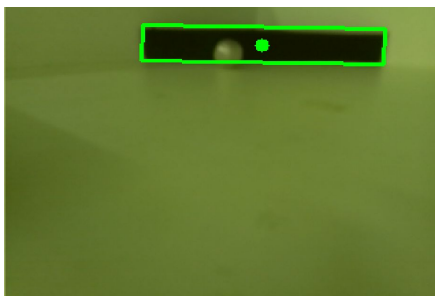


Fonte: ARAÚJO (2019)

Marinato *et al* (2017) deixa claro também a necessidade de um sistema de identificação da área de resgate (região segura para as vítimas). A detecção da área de resgate desenvolvida por este trabalho mostra que mesmo uma vítima prateada obstruindo a visão completa da área, essa é identificada (Figura 6) por um grande retângulo preenchido preto (na perspectiva do robô) circundado por um retângulo verde. Em testes quantitativos foi observado que o *software* consegue identificar a área de resgate em 56% dos casos

(ARAÚJO, 2019, no prelo), o que evidenciou a dependência de melhorias no *software* e na calibração da câmera do robô.

Figura 6 - Identificação da área de resgate



Fonte: ARAÚJO (2019)

5 CONCLUSÃO

Na OBR 2019, a utilização da visão computacional para a resolução de um dos desafios da competição assegura a intenção deste trabalho: a equipe Wall-E (objeto de trabalho deste projeto) recebeu o prêmio de inovação, evidenciando que a proposta possui relevância tecnológica e que pode inspirar outras equipes participantes a utilizarem robôs desenvolvidos com base no reconhecimento e tratamento de imagens.

Apesar de algumas dificuldades e algumas tarefas incompletas, este trabalho vem contribuindo de forma significativa para o desenvolvimento do LARIS, permitindo que as tecnologias propostas possam ser desenvolvidas por novos alunos. A proposta apresentada abre direções para trabalhos futuros que poderão superar os problemas deste trabalho e aperfeiçoar suas capacidades.

Para trabalhos futuros, propõe-se:

1. Aperfeiçoamento das estratégias de utilização dos algoritmos de tratamento de imagem;
2. Utilização desta proposta na Olimpíada Brasileira de Robótica (OBR);
3. Desenvolvimento de uma documentação para uso de técnicas de visão computacional em robôs utilizados na OBR.

REFERÊNCIAS

AMORIM, Junior et al. **Implementação de um robô para competição baseado em visão computacional**. Anais da Mostra Nacional de Robótica – MNR 2018, Sistema Olimpo, 2018. Disponível em: <http://sistemaolimpo.org/midias/uploads/74a53560dc4f4644cae153091d1e116e.pdf>. Acesso em: 16 abr. 2019.

ARAÚJO, Douglas Gabriel; LOURENÇO, Isaac Marlon; BORGES, Paulo Vitor. **Byakugan: Visão Computacional**. Anais da Mostra Nacional de Robótica - MNR 2019, 2019. No prelo.

ARRUDA, Gabriel Augusto; MARIANO, Vinícius Barbosa Pereira Mariano; SALES, Vinicius Gambi. **Otimização de detecção de rotas e reconhecimento de objetos para Robocup Junior Rescue por meio de processamento digital de imagem e visão computacional**. MNR 2017, Londrina – PR, 2017. Disponível em: <http://www.mnr.org.br/wpcontent/uploads/2019/06/MNR-Anais2017.pdf>. Acesso em: 11 ago. 2019.

BARBOSA, André; FAGUNDES, Elizandro Gomes; DARÓS, Renan Rocha. **Uma metodologia para a resolução do desafio da olimpíada brasileira de robótica – O robô Volk**. 5º Simpósio de Integração Científica e Tecnológica do Sul Catarinense – SICT-Sul, 2016. Disponível em: <http://docente.ifsc.edu.br/werther/publicacoes/2016SITCSUL-2a.pdf>. Acesso em: 10 ago. 2019.

EDNO, Francisco (ed.). **Tutoriais sobre ROS**. 14 dez. 2014. Disponível em: http://wiki.ros.org/pt_BR/ROS/Tutorials. Acesso em: 15 mar. 2019.

G. Senthilkuma, K. Gopalakrishnan, V. Sathish Kumar. **Embedded Image Capturing System using Raspberry Pi System**. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS). Volume 3, Issue 2, March – April 2014.

MARINATO, Gabriela P.; SOARES, Joyce A. P.; AMARAL, Eduardo M. A. **Sistema de detecção e resgate de vítima para um robô autônomo seguidor de linha baseado em visão computacional**. Mostra Nacional de Robótica (MNR), Serra - ES - Brasil, 2017. Disponível em: <http://sistemaolimpo.org>. Acesso em: 16 abr. 2019.

OPENCV, Dev Team. **OpenCV 2.4.13.7 documentation**, 12 jul. 2018. Disponível em: <https://docs.opencv.org/2.4.13.7/>. Acesso em: 15 mar. 2019.

ORGANIZAÇÃO OBR. **Por que uma olimpíada de robótica?** OBR, 2018. Disponível em: <http://www.obr.org.br/>. Acesso em: 16 abr. 2019.

ORGANIZAÇÃO OBR. **Manual de Regras e Instruções – Etapa Regional/Estadual**. Manuais OBR, 2019. Disponível em: http://www.obr.org.br/manuais/OBR2019_MP_ManualRegionalEstadual.pdf. Acesso em: 1 nov. 2019.

REIGNIER, Romain (ed.). **Rosserial: Documentation**. ROS Wiki, 2018. Disponível em: <http://wiki.ros.org/rosserial>. Acesso em: 24 nov. 2019.

SOARES, Lennedy; SILVA, Anderson. **Biblioteca robo_hardware**. 17. ed. Github, 2018. Disponível em: https://github.com/IFRN-SC/robo_hardware/wiki. Acesso em: 24 nov. 2019.