LEVEZ, Felipe Bertelli; BENINI, Fabriciu Alarcão Veiga; GOMES, Gabriel Henrique Faustini. Aplicando opencv no raspberry pi. In: WORKSHOP DE INOVAÇÃO, PESQUISA, ENSINO E EXTENSÃO, 3., 2018, São Carlos, SP. **Anais**... São Carlos, SP: IFSP, 2018. p. 17-20. ISSN 2525-9377.

APLICANDO OPENCV NO RASPBERRY PI

FELIPE BERTELLI LEVEZ; FABRICIU ALARCÃO VEIGA BENINI; GABRIEL HENRIQUE FAUSTINI GOMES

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, São Carlos, Brasil

RESUMO: O presente trabalho aborda a aquisição de imagens orientada a sistemas embarcados. Para isso, foi utilizado a Raspberry Pi 2 e um módulo de câmera CF5647CM-V1 de 5MP. Ao longo da publicação, são explicados conceitos sobre Visão Computacional e uma das bibliotecas *open source* mais utilizadas para aquisição de imagem, o OpenCV. Ademais, é descrito como instalar e configurar corretamente as bibliotecas necessárias e como realizar uma captura e leitura de imagem utilizando C++.
PALAVRAS-CHAVE: OpenCV. Sistema Embarcado. Visão computacional. Processamento de imagem.

ABSTRACT: The present work deals with the acquisition of images oriented to embedded systems. For this, the Raspberry Pi 2 and a camera module CF5647CM-V1 of 5MP were used. Throughout the publication, we explain concepts about Computational Vision and one of the most used open source libraries for image acquisition, OpenCV. In addition, it describes how to correctly install and configure the required libraries and how to perform an image capture and reading using C ++.
KEYWORDS: OpenCV. Embedded system. Computer vision. Image processing.

INTRODUÇÃO

Com o passar do tempo os sistemas embarcados vêm ganhando grande popularidade. De forma que na atualidade a sua capacidade de processamento pode ser equiparada com a de um computador de mesa em alguns aspectos. Para a aquisição de imagens ela está cada vez mais simples, pois as ferramentas de desenvolvimento estão sendo direcionadas para esse propósito, ainda que inicial, incluindo as plataformas de auxílio à programação. Estas últimas são fartas e grandes chegando ao ponto da capacidade de programar e executar um código remotamente, através do terminal de um PC (ANDRADE, 2011).

Uma das várias dificuldades encontradas na atualidade está em como capturar uma imagem. Para isso foi desenvolvido um estudo, concluindo que o melhor caminho é através de uma biblioteca denominada *OpenCV*. Esta, é conhecida por ser uma das principais bibliotecas *open source* para visão computacional, tratamento de imagens e aprendizagem de máquina. Além disso, apresenta aceleração de GPU para auxiliar em operações em tempo real (EMBARCADOS, 2017).

A visão computacional também é conhecida como visão de máquinas. Com ela, torna-se possível capturar as informações das imagens, desde astronômicas até microscópicas. Através de algoritmos consegue-se descrever e analisar a informação de qualquer imagem digitalizada (GOMES; BENINI, 2016).

Trabalhar com imagens em sistemas embarcados não é uma tarefa fácil. Ele demanda muitas pesquisas e testes. Todavia, com o auxílio da internet e a crescente utilização de sistemas embarcados, a documentação está cada vez mais farta.

MATERIAL E MÉTODOS

O sistema embarcado é o Raspberry Pi 2, utilizado conjuntamente com um módulo de câmera CF5647CM-V1 de 5MP, criado especialmente para esse fim. O Raspberry Pi é a mistura de minicomputador e microcontrolador, de baixo custo, que possui um tamanho reduzido. Dessa forma, pode ser utilizado para realização de automação residencial, projetos científicos, confecção de robôs, entre outros.

Um ponto interessante, atualmente, é a quantidade de documentações e artigos encontrados devido ao aumento da popularidade dos sistemas embarcados, motivo este que resulta em crescentes números de projetos, publicações e documentações.

Concluiu-se que a biblioteca de visão computacional, denominada *OpenCV*, é a que apresenta melhores recursos. Esta, será usada como solução para uma das grandes dificuldades dos sistemas embarcados, a aquisição e tratamento de imagens. Sua complexidade atinge níveis avançados de técnica de processamento, com funções capazes de aplicar a *transformada rápida de Fourier*, que se baseia em decompor uma imagem em seus componentes, senos e cossenos. Ainda, transforma uma imagem de seu domínio espacial para o domínio da frequência. Contudo, o ponto mais delicado do projeto é a instalação da biblioteca *OpenCV*, pois é demorada, durando cerca de uma hora e meia. Além disso, é sensível, podendo até mesmo travar o processador central.

Para a instalação e configuração do *OpenCV*, deve-se seguir os seguintes passos:

- cd ~
- git clone https://github.com/Itseez/opencv.git
- cd ~/opencv
- mkdir build
- cd build
- cmake -D CMAKE_BUILD_TYPE=Release -D CMAKE_INSTALL_PREFIX=/usr/local ..
- make -j<jobs em parallel>
- make install

A linguagem de programação utilizada é a C++. Esta é conhecida por ser multi-paradigma e possui boa flexibilidade, além de seu alto desempenho relacionado a softwares científicos (Microsoft; 2016).

Após sucessivos testes, estabeleceu-se algoritmos que quando combinados e executados são capazes de capturar uma imagem e armazená-la em um local previamente estabelecido. No final do processo, 3 arquivos foram criados, são eles:

- DisplayImage2.cpp;
- CMakeLists.txt;
- camera.sh.

O arquivo *bash*, *camera.sh*, é responsável por capturar a imagem, nomeá-la com a data e o horário da captura e chamar o executável *DisplayImage* passando como parâmetro o nome da imagem capturada no momento. Em um arquivo *bash*, há um script shell que é um grupo de comandos variáveis, ou qualquer outro texto que pode ser utilizado em um terminal linux. E, é utilizado para automatizar tarefas, a fim de poupar tempo e trabalho (NEGUS; BRESNAHAN, 2014).

O arquivo em C++,*DisplayImage2.cpp*, tem como finalidade verificar se a captura da imagem foi realizada com sucesso retornando uma mensagem de erro caso negativo. Também tem a função de implementar funções que podem alterar os dados da imagem, para ser aplicado em algoritmos de visão computacional.

O arquivo *cmake*, *CMakeLists.txt*, é responsável por compilar o arquivo DisplayImage2.cpp, assim gerando um executável denominado DisplayImage.

RESULTADOS E DISCUSSÃO

Ao longo do projeto, foi encontrado um meio de implementar uma base de funcionamento para visão computacional. Utilizando-se da biblioteca *OpenCV*, que oferece uma quantidade significativa de recursos para processamento de imagens.

Com a *OpenCV*, foi realizada a implementação de um algoritmo, usando a *linguagem C++* e arquivos *Bash*, que tem como finalidade capturar a imagem e armazená-la em um arquivo de extensão .jpg (*Joint Photographic Experts Group*).

Abaixo podem ser vistos os arquivos criados e utilizados no projeto.

camera.sh

```
#!/bin/bash
DATE=$(date '+%d-%m-%Y_%H-%M')
raspistill -o /home/pi/camera/$DATE.jpg
./DisplayImage /home/pi/camera/$DATE.jpg
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 2.8)
project( DisplayImage )
find_package( OpenCV REQUIRED )
add_executable( DisplayImage DisplayImage2.cpp )
target_link_libraries( DisplayImage ${OpenCV_LIBS} )
```

DisplayImage2.cpp

```
#include <stdio.h>
#include <opencv2/opencv.hpp>
#include <vector>
#include <iostream>

using namespace cv;

int main(int argc, char** argv )
{
    if ( argc != 2 )
    {
        printf("usage: DisplayImage.out <Image_Path>\n");
        return -1;
    }

    Mat image;
    image = imread( argv[1], 1 );

    if ( !image.data )
    {
        printf("No image data \n");
        return -1;
    }

    std::vector<uchar> buf;
    imencode(".jpg", image, buf, std::vector<int>());

    printf("%lu\n", buf.size());

    namedWindow("Display Image", WINDOW_NORMAL );
    imshow("Display Image", image);

    waitKey(0);

    return 0;
}
```

CONCLUSÕES

Utilizar sistemas de aquisição de imagens é uma tarefa complexa, como dito anteriormente. Contudo, ao atingir o objetivo inicial que era a captura de imagem utilizando a biblioteca *OpenCV*, verificamos que apesar da dificuldade, é tangível. Também, foi possível capturar imagens

remotamente utilizando os mesmos recursos. Conclui-se que este projeto pode e deve ser disseminado para que auxilie os demais pesquisadores, estudantes ou qualquer pessoa que possua interesse em sistemas embarcados.

REFERÊNCIAS

ANDRADE, D. F. **Sistema Embarcado para Aquisição de Imagens Astronômicas**. São Paulo, 2011. Dissertação de mestrado apresentada à Escola Politécnica da Universidade de São Paulo, 2011.

EMBARCADOS. **Aplicação de visão computacional com OpenCV**. Disponível em: <https://www.embarcados.com.br/aplicacao-de-visao-computacional-com-opencv/>. Acesso em: 02 jan. 2018.

GOMES, G. H. F.; BENINI, F. A. V. Plataforma de aquisição de imagem para sistemas embarcados. In: WORKSHOP DE INOVAÇÃO, PESQUISA, ENSINO E EXTENSÃO, 2., 2016, São Carlos, SP. **Anais...** São Carlos, SP: IFSP, 2016. p. 80-83. ISSN 2525-9377.

MICROSOFT. Bem-vindo ao C++ (C++ Moderno). Disponível em:< https://msdn.microsoft.com/pt-br/library/hh279654.aspx >. Acesso em: 24 mar. 2018.

NEGUS, C.; BRESNAHAN, C. **Linux A Bíblia**. 8. ed. Rio de Janeiro: Editora Alta Books, 2014.