Artigo disponibilizado *on-line*

Revista Ilha Digital

Endereço eletrônico:
http://florianopolis.ifsc.edu.br/~eletronica/

CONTEGEM DE OBJETOS UTILIZANDO FILTRO HSV

Suzi Yousif[1]

Resumo: Este artigo apresenta o desenvolvimento de um contador de objetos utilizando os limiares do do espaço de cor HSV. O seu objetivo principal é poder contar objetos coloridos através da implementação de uma interface de tempo real que permita ao usuário definir quais processamentos são necessários de uma forma didática. Para isso, optou-se pelo uso de Raspberry Pi com um Webcam. Para a leitura e filtragem das imagens, utilizou-se a biblioteca OpenCV na linguagem c++.

Palavras-chave: Processamento de imagem, Filtro HSV, OpenCV, Raspberry Pi.

Abstract*: This paper introduces the development of an object counter using the HSV color space thresholds. The aim of this project is to be able to count colored objects by implementing a real-time interface that allows the user to define which processing is needed in a didactic way. Therefore, it was used Raspberry Pi with a webcam. For reading and filtering the images, OpenCV library was applied in c ++ language.*

Keywords*: Image processing, HSV filter, OpenCV, Raspberry Pi.*

[1] Acadêmica do curso superior de Engenharia Eletrônica do DAELN do IF-SC <suziyousif17@gmail.com>.

1. INTRODUÇÃO

Contagem de objetos é uma tarefa essencial em algumas aplicações de visão computacional, como por exemplo, nas fabricas que contém uma linha de produção constante ou quando há algum tipo de separação de produtos por cor.

Este projeto busca aplicar os conceitos de processamento digital de imagens para contar os objetos em tempo real com base nas cores desejadas.

Há várias formas de detecção de objetos por cor, entre elas há o filtro HSV, o qual foi escolhido para o desenvolvimento por ser mais perceptível ao olho humano que os outros modelos de espaço de cor (ZHANG,2017). O desenvolvimento foi feito com a biblioteca OpenCV, utilizando a plataforma Raspberry Pi.

Para realizar a análise das informações extraídas das imagens, foram aplicados uns conjuntos de algoritmos de funcionalidades diferentes, em que cada uma poderá ser alterada pelo próprio usuário através de uma interface com barras de rolagem.

2. DESENVOLVIMENTO

2.1. Filtro HSV

O modelo HSV, apresentado na Figura 1, é definido pelos parâmetros: matiz (H, *hue*), saturação (S, *saturation*) e luminância (V, value). A matiz define a tonalidade dominante de uma área, cor pura, variando entre 0° e 360°, a saturação mede o quanto esta cor pura está diluída com a cor branca (a pureza da cor) e o valor está relacionado ao brilho.

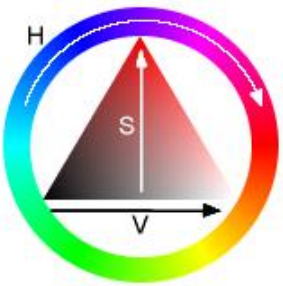


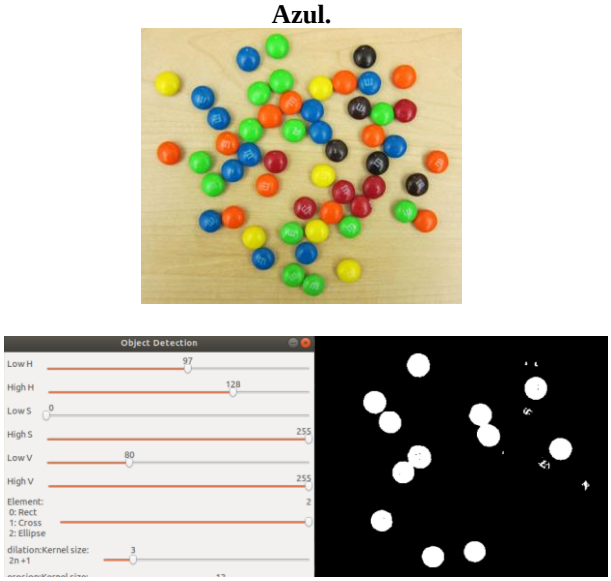
FIGURA 1 – O sistema de cores HSV.

Optou-se pelo uso do filtro HSV devido aos seus resultados melhores em relação ao

processamento de imagem que precisa segmentar objetos com base em sua cor. Além disso, como objetivo deste projeto é ser aplicado em tempo real, é necessário o uso de um filtro mais robusto em relação a variação da iluminação externa, de forma que este modelo possui a capacidade de separar a informação iluminação da cor, diferentemente do modelo RGB, em que não poderemos realizar isso.

Para a implementação, a imagem foi convertida de RGB para HSV, aplicando a função *cvtColor* da biblioteca OpenCV. Com a finalidade de deixar a implementação mais genérica, implementou-se seis barras de rolagem (*trackbar*), onde o usuário possa definir os limiares de H, S e V para, posteriormente, extrair um objeto com base nesses valores, aplicando a função *inRange*. Na Figura 2, é apresentada uma imagem com a aplicação do filtro HSV para a obtenção dos objetos azuis.

FIGURA 2 – Filtro HSV para detecção da cor Azul.



2.2. Processamento de Imagem Morfológica

A morfologia matemática é uma ferramenta de extração de componentes de uma imagem, os quais são relevantes para a representação e descrição de uma região, como também para filtragem morfológica. (GONZALEZ; WOODS, 2008).

2.2.1. Dilatação e Erosão

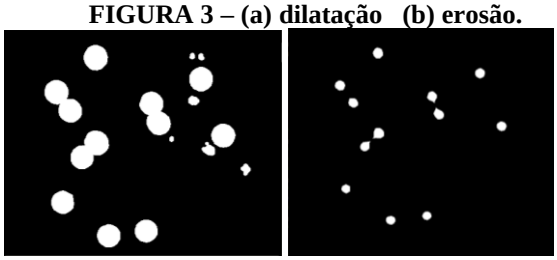
Dilatação e erosão são operações fundamentais para o processamento morfológico. A dilatação é aplicada para aumentar a área de um objeto, sendo que, algumas das suas aplicações são o preenchimento de lacunas e a remoção de ruídos.

A erosão é o contrário da dilatação, ela retira os pixels que se encontram nas bordas de um objeto. Ele é usualmente aplicado para separação de objetos conectados.

Nesta implementação, criou-se três barras de rolagem, uma para definir o elemento estruturante (retangular, cruz ou elíptico), outros dois para definir o tamanho do *kernel* para a aplicação da dilatação e da erosão. Essas foram feitas com as funções *dilate* e *erode* da biblioteca OpenCV.

Geralmente essas duas funcionalidades são aplicadas juntas, conforme apresentado na Figura 3, onde foi necessário dilatar os círculos para posteriormente aplicar a erosão com o objetivo de separar os objetos que estavam conectados.

FIGURA 3 – (a) dilatação (b) erosão.



(a) (b)

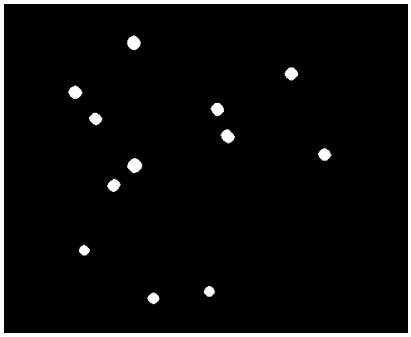
2.3. Filtro Mediana

Filtro mediana é uma tecnologia de processamento de sinais não linear, baseada em estatística. Os valores ruidosos dentro de uma imagem são substituídos pelo valor mediano da vizinhança. (ZHU; HUANG, 2012)

De acordo com Gonzalez e Woods (2008), o filtro da mediana é excelente para remover os ruídos sem reduzir a nitidez da imagem.

Neste projeto, este filtro é aplicado com o uso da função *medianBlur*, após a dilatação e a erosão com a finalidade de remover todo o ruído de alta frequência restante na imagem. A Figura 4 apresenta o resultado deste filtro.

FIGURA 4 – Filtro Mediana.



2.4. Detecção do Objeto

Após a filtragem da imagem, aplicou-se a função *connectedComponentsWithStats* para

detectar todos os objetos conectado. Essa função retorna uma matriz das estatísticas que a função calcula. Uma dessas informações é a área de cada objeto detectado, o qual foi utilizado como limiar para evitar a detecção de objetos muito pequenos, os quais são na maioria das vezes ruídos. Além disso, ela fornece as coordenadas do objeto, o qual foi utilizado para desenhar um retângulo em cada objeto detectado. Na Figura 5 é possível visualizar o resultado final da implementação, onde foram contados todos os objetos azuis.

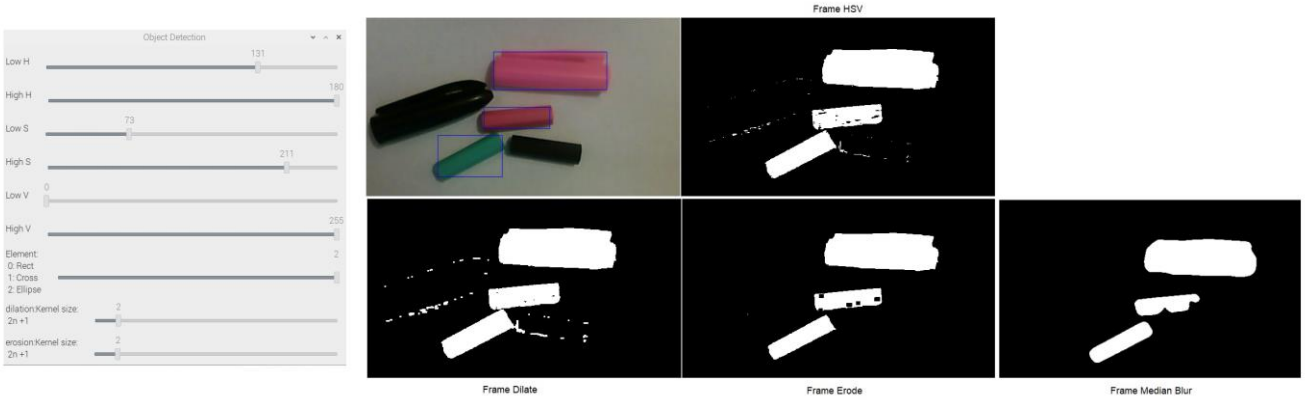
FIGURA 5 – Contagem dos objetos detectados.

3. IMPLEMENTAÇÃO NO RASPBERRY PI

A implementação na plataforma Raspberry Pi necessitou um cartão SD para instalação do sistema operacional Raspbian, Webcam para detecção de objetos em tempo real, monitor para visualização dos resultados, um teclado e um mouse. Com toda a estrutura montada, instalou-se as dependências da biblioteca OpenCV e gravou o código do arquivo .cpp, que contém a implementação, dentro da Raspberry Pi.

A figura 6 apresenta os passos de filtragem para detectar objetos com tons mais claros, com o uso do Webcam.

FIGURA 6 – Contagem dos objetos utilizando Webcam em tempo real.



4. CONSIDERAÇÕES FINAIS

O uso do filtro HSV se mostrou adequado para a detecção de objetos em uma imagem que tenha fundo branco, devido às suas características de poder separar a iluminação da cor. Com isso, foi possível contar os objetos com e sem o uso de uma cor determinada. Além disso, o uso da interface com barras de rolagem fez com que o algoritmo seja o mais genérico possível, o qual é uma tarefa complexa nas aplicações de visão computacional, devido a vários fatores do meio ambiente que possam afetar o processamento, como por exemplo, a iluminação externa.

REFERÊNCIAS

GONZALEZ, Rafael C.; WOODS, Richard E.. **Digital Image Processing.** 3. ed. New Jersey: Pearson, 2008.

Open Source Computer Vision. Disponível em: <https://docs.opencv.org/3.4/index.html>. Acesso em: 4 dez. 2019.

ZHANG, Y.-J. Image Processing. Berlin, [Germany]: De Gruyter, 2017. ISBN 9783110520323. Acesso em: 10 dez. 2019.

ZHU, Youlian; HUANG, Cheng. An Improved Median Filtering Algorithm for Image Noise Reduction. **Elsevier**, Changzhou, p.609-616, 2012.