# Projeto Demonstrativo 1
# Explorando OpenCV

Carlos Adir Ely Murussi Leite
150121059
carlos.adir.leite@gmail.com

Departamento de Ciência da Comptutação
Universidade de Brasília
Campus Darcy Ribeiro, Asa Norte
Brasília-DF, CEP 70910-900, Brazil,

**Abstract**

Este documento mostra quatro implementações de algoritmos requeridos pelo Projeto Demonstrativo 1, que propõe 4 requisitos. Os algoritmos são colocados em anexo junto a esse documento. Em suma, o atendimento dos requisitos são apenas para aproximar o contato dos alunos da Disciplina de Princípios de Visão Computacional da Universidade de Brasília com a biblioteca OpenCV. Tais implementações foram feitas em C++ como linguagem principal.

## 1 Introdução

A disciplina Princípios de Visão Computacional da Universidade de Brasilia ensina diversas coisas relacionadas à visão computacional, que em resumo consiste em transformar imagens em dados. Para isso, a disciplina utiliza da biblioteca OpenCV[1] para implementação de algoritmos tanto para auxiliar na aquisição , quanto no tratamento de dados.

Para aplicações futuras na disciplina, é necessário que o aluno se sinta confortavel na utilização das funcionabilidades da biblioteca OpenCV e esse projeto demonstrativo tem por objetivo aproximar o contato do aluno à biblioteca OpenCV.

As especificações do projeto funcionam de forma gradual, ou seja, os requisitos posteriores utilizam boa parte dos requisitos anteriores. As especificações consistem, em resumo de:

1. Fazer o upload de uma imagem armazenada e fazer uma interface interativa com cliques do mouse.
2. Com o procedimento anterior, alterar determinados pixeis e deixa-los vermelhos
3. Semelhante ao requisito anterior, mas com video no lugar de uma imagem.
4. O mesmo que o requisito anterior, mas utilizando uma câmera no lugar do video.

Para implementação dos 4 algoritmos que cumpram os requisitos, utilizou-se C++ como linguagem de implementação. Para aprendizado das funções e biblioteca foi utilizado tanto a página oficial da OpenCV[1], quanto um tutorial disponibilizado na internet[3].

# 2 Metodologia

Primeiro passo foi definir a linguagem de utilização, e foi escolhida a linguagem C++ para cumprir os requisitos. Isso porque um dos objetivos futuros é aplicação de visão computacional em sistemas embarcados, e a utilização de uma linguagem mais veloz embora mais complexa se torna necessário. A metodologia de projeto consistiu em dividir o Projeto Demonstrativo em 4 partes, na mesma divisão dos requisitos. Cada parte foi implementada separadamente:

## 2.1 Requisito 1

Primeira ação foi a criação de uma estrutura de classes para facilitar a implementação e deixar o código versátil. As classes implementadas foram em um total de 3 com seus respectivos atributos e metodos. Com essas classes, planejou-se fazer um código simples. A ideia principal era criar o código inicial que fosse utilizado em todos os outros requisitos.

## 2.2 Requisito 2

Para o requisito 2, a abordagem foi utilizar o código do requisito 1 para armazenamento de imagem e captura de dados, e criar um método dentro da classe da imagem que permita colorir todos os pixeis similares, com uma distância próxima

## 2.3 Requisito 3

A abordagem consistiu em utilizar o mesmo código do requisito anterior, em que colore a imagem a partir do clique, mas que utilize uma referência(um atributo do objeto) para caso o video mude, o algoritmo mantenha o pixel de interesse e não a posição. Outro fator nessa parte é de, diferente de outros casos em que só muda a imagem quando há clique, a todo momento um novo frame do vídeo é carregado e deve ser mostrado.

## 2.4 Requisito 4

A abordagem consistiu em utilizar o código do requisito 3, mas em vez do input ser um arquivo, ser a própria câmera do usuário. Em termos práticos, mudar apenas uma linha.

# 3 Resultados

## 3.1 Requisito 1

Implementou-se 3 classes no total e consistem em:

- Ponto: Uma alternativa em vez de passar as coordenadas x e y frequentemente, com risco de troca entre os termos.
- Pixel: Para armazenar as componentes RBG de um pixel. Fez-se essa classe para evitar passar as 3 componentes frequentemente
- Imagem: Para armazenar a imagem e abrigar metodos de pintar certos pixeis, a de pintar todos de mesma cores, e descobrir se imagem é na escala cinza.

A partir dessas classes, a implementação do algoritmo que atendesse ao requisito 1 consistiu em: A cada momento que fosse clicado em um ponto difente, a função responsável pelo mouse retornava as coordenadas $(x,\ y)$ do ponto clicado. Em um loop, sempre que o ponto era diferente do ponto clicado anteriormente, passa-se o ponto $(x,\ y)$ para o objeto da imagem e captura as informações e imprime na tela.

Os resultados são mostrados pelas Figuras (3.1) e (3.1)

Figure 1: Requisito 1 - Imagem colorida

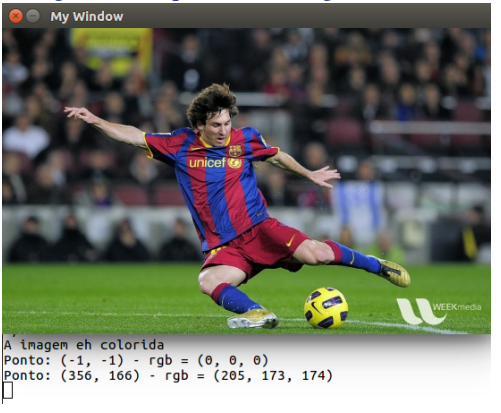


Figure 2: Requisito 1 - Imagem escala cinza


## 3.2 Requisito 2

Para o requisito 2, bastava apenas pegar a informação coletada no requisito 1 e colorir pixeis próximos. Para isso, bastou implementar a função *PintaDistancia* em que colore todos os pixeis próximos de acordo com a distância. Mudando o parâmetro *distancia*, na linha 54 dos algoritmos 2.cpp, 3.cpp e 4.cpp, obtém-se diferentes resultados.

Um fator a ser comentado é que sempre que há mudança do ponto clicado, a imagem é recarregada e colorida conforme o pixel clicado. Dessa maneira, sempre se espera um evento de clique para recarregar a imagem e pintar. O resultado é mostrado pela Figura (3.2)

Figure 3: Requisito 1 - Imagem escala cinza


### 3.3 Requisito 3

Conforme comentado na metodologia, de que o objetivo de implementação desse requisito era fazer a mudança apenas do frame e utilização de uma referência de pixel, obteve-se o resultado esperado.

Infelizmente não foi possivel fazer a troca do video para preto e branco pois a função testada e mostrada pela literatura[2] não funcionada embora diversas tentativas de adaptação.

### 3.4 Requisito 4

Fazendo os processos descritos na metodolia, obteve-se o código implementando no arquivo 4.cpp em que se utiliza a câmera para captura de imagens.

## 4 Discussão e Conclusões

Como se pode observar, o objetivo de familiarizar o aluno com a biblioteca OpenCV e a implementação de alguns algoritmos foi obtida com êxito, fazendo o aluno aprender a lidar com as funções mais básicas e trabalhar com a imagem. Infelizmente nos algoritmos 3 e 4 não foi possivel converter a imagem para escala cinza, provavelmente por desatualização do código fonte. Próximo passo é a implementação dos outros projetos demonstrativos.

Por ser um assunto simples, não tem muito conteudo. O código referente ao relatório bem como imagens e outras informações são encontrados no GitHub[4], e uma descrição mais detalhada encontrada em seu Wiki[5].

## References

[1] OpenCV Community. Opencv, 2018. URL http://ieeexplore.ieee.org/document/940586/.

[2] OpenCV Community. Opencv - miscellaneous image transformations, 2018. URL https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html.

[3] Shermal Fernando. Opencv tutorial c++, 2018. URL www.opencv-srf.com/.

[4] Carlos Adir Ely Murussi Leite. Github - pvc, 2018. URL https://github.com/CarlosAdir/PVC.

[5] Carlos Adir Ely Murussi Leite. Github - pvc wiki, 2018. URL https://github.com/CarlosAdir/PVC/wiki.