Tutorial de OpenCV em C++

Curso de Linguagem de Programação Específica para IA
PPI Fotografia Computacional

INSTITUIÇÃO EXECUTORA

Centro de Informática UFPE

FADE UFPE

COORDENADORA

MCTI FUTURO

Softex

APOIO

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

Introdução

Instalação

➔ Para trabalharmos com OpenCV em C++, primeiro precisamos assegurar que o ambiente esteja devidamente configurado e que a biblioteca esteja instalada.

- Tutorial para Windows: https://docs.opencv.org/4.x/d3/d52/tutorial_windows_install.html
- Tutorial para Linux: https://docs.opencv.org/4.x/d7/d9f/tutorial_linux_install.html

*Obs: É interessante adicionar as flags abaixo durante a instalação

-DOPENCV_GENERATE_PKGCONFIG=ON -DBUILD_opencv_python3=OFF

API de OpenCV para C++

Headers

➔ Quando planejamos utilizar OpenCV com um programa em C++, apenas precisamos incluir o arquivo header “opencv2/opencv.hpp”, pois ele incluirá todos os outros arquivos header necessários para executar as aplicações.

```
#include <iostream>
#include <opencv2/opencv.hpp>
```

Namespace

- Todas as classes e funções de OpenCV estão no *namespace* cv, então temos duas possibilidades:
  - Adicionar a instrução ***using namespace cv;*** logo depois de incluir os arquivos header.
  - Acrescentar o especificador ***cv::*** na declaração de cada classe, função ou estrutura de dados de OpenCV no código.

Namespace

```cpp
#include <opencv2/opencv.hpp>
using namespace cv;

int main(int argc, char* argv[]) {
    // Leitura da imagem
    Mat image = imread("C:/Images/Eagle.jpg");
    return 0;
}
```

Exemplo de código com *using namespace cv*

```cpp
#include <opencv2/opencv.hpp>

int main(int argc, char* argv[]) {
    // Leitura da imagem
    cv::Mat image = cv::imread("C:/Images/Eagle.jpg")
    return 0;
}
```

Exemplo de código com o especificador *cv::*

Tipos de dado de um array

- ➔ O tipo de dado de um array define o número de canais, o número de bits alocados para cada elemento e como o valor de um elemento é representado usando esses bits.

- ➔ Se um array for uma imagem, cada elemento representa um pixel da imagem.

Tipos de dado de um array

- ➔ Qualquer array com canais deve pertencer a um dos tipos de dado abaixo:
  - CV_8U - inteiro de 8 bits sem sinal
  - CV_8S - inteiro de 8 bits com sinal
  - CV_16U - inteiro de 16 bits sem sinal
  - CV_16S - inteiro de 16 bits com sinal
  - CV_32S - inteiro de 32 bits com sinal
  - CV_32F - ponto flutuante de 32 bits
  - CV_64F - ponto flutuante de 64 bits

Carregar e Exibir uma Imagem

Carregar e exibir uma imagem

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    // Leitura da imagem
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    // Verificação de erro
    if (image.empty()) {
        cout << "Não foi possível abrir ou encontrar a imagem" << endl;
        cin.get(); // Espera qualquer tecla ser pressionada
        return -1;
    }

    String windowName = "Janela"; // Nome da janela
    namedWindow(windowName); // Criando uma janela
    imshow(windowName, image); // Exibindo a imagem na janela criada
    waitKey(0); // Espera uma tecla ser pressionada
    destroyWindow(windowName); // Destrói a janela criada
    return 0;
}
```

Carregar e exibir uma imagem

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    // Criando uma imagem com:
    // 3 canais
    // 8 bits de profundidade
    // resolução 800 x 600 (largura x altura)
    // cada pixel inicializado com valores (100, 250, 30) para os canais
    // B, G e R, respectivamente
    Mat image(600, 800, CV_8UC3, Scalar(100, 250, 30));

    String windowName = "Janela"; // Nome da janela
    namedWindow(windowName); // Criando uma janela
    imshow(windowName, image); // Exibindo a imagem na janela criada
    waitKey(0); // Espera uma tecla ser pressionada
    destroyWindow(windowName); // Destrói a janela criada
    return 0;
}
```

Salvar uma Imagem

Salvar uma imagem

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    /*
    Mudanças feitas na imagem
    */

    bool isSuccess = imwrite("D:/MyImage.jpg", image); // Escreve a imagem como um arquivo JPG
    if (isSuccess == false) {
        cout << "Falha em salvar a imagem" << endl;
        cin.get();
        return -1;
    }

    cout << "Imagem salva com sucesso" << endl;

    String windowName = "Imagem Salva";
    namedWindow(windowName);
    imshow(windowName, image);
    waitKey(0);
    destroyWindow(windowName);

    return 0;
}
```

Alterar o Brilho de uma Imagem

Alterar o brilho de uma imagem

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    Mat imageBrighnessHigh50;
    image.convertTo(imageBrighnessHigh50, -1, 1, 50); // aumenta o brilho em 50

    Mat imageBrighnessHigh100;
    image.convertTo(imageBrighnessHigh100, -1, 1, 100); // aumenta o brilho em 100

    Mat imageBrighnessLow50;
    image.convertTo(imageBrighnessLow50, -1, 1, -50); // reduz o brilho em 50

    Mat imageBrighnessLow100;
    image.convertTo(imageBrighnessLow100, -1, 1, -100); // reduz o brilho em 100

    String windowName = "Realce de Brilho";
    namedWindow(windowName);
    imshow(windowName, image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Alterar o Contraste de uma Imagem

Alterar o contraste de uma imagem

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    Mat imageContrastHigh2;
    image.convertTo(imageContrastHigh2, -1, 2, 0); // aumenta o contraste em 2

    Mat imageContrastHigh4;
    image.convertTo(imageContrastHigh4, -1, 4, 0); // aumenta o contraste em 4

    Mat imageContrastLow0_5;
    image.convertTo(imageContrastLow0_5, -1, 0.5, 0); // reduz o contraste em 0.5

    Mat imageContrastLow0_25;
    image.convertTo(imageContrastLow0_25, -1, 0.25, 0); // reduz o contraste em 0.25

    String windowName = "Realce de Contraste";
    namedWindow(windowName);
    imshow(windowName, image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Equalização

Equalização de uma imagem em escala de cinza

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    // mudar a cor da imagem para escala de cinza
    cvtColor(image, image, COLOR_BGR2GRAY);

    // equalizar o histograma
    Mat hist_equalized_image;
    equalizeHist(image, hist_equalized_image);

    String windowName = "Histogram Equalized Image";
    namedWindow(windowName);
    imshow(windowName, hist_equalized_image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Equalização de uma imagem colorida

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    //Convertendo a imagem de BGR para o espaço de cores YCrCb
    Mat hist_equalized_image;
    cvtColor(image, hist_equalized_image, COLOR_BGR2YCrCb);
    //Dividindo a imagem em 3 canais e a armazenando em um vector
    vector<Mat> vec_channels;
    split(hist_equalized_image, vec_channels);
    //Equalizando o histograma do canal Y
    equalizeHist(vec_channels[0], vec_channels[0]);
    //Combinando 3 canais no vector para formar a imagem colorida no espaço de cores YCrCb
    merge(vec_channels, hist_equalized_image);
    //Convertendo a imagem equalizada do espaço de cores YCrCb de volta para BGR
    cvtColor(hist_equalized_image, hist_equalized_image, COLOR_YCrCb2BGR);

    String windowName = "Histogram Equalized Image";
    namedWindow(windowName);
    imshow(windowName, hist_equalized_image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Filtros de Imagem

Blur Homogêneo

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    //Aplicando blur na imagem com kernel 3x3
    Mat image_blurred_with_3x3_kernel;
    blur(image, image_blurred_with_3x3_kernel, Size(3, 3));

    //Aplicando blur na imagem com kernel 5x5
    Mat image_blurred_with_5x5_kernel;
    blur(image, image_blurred_with_5x5_kernel, Size(5, 5));

    String window_name = "Imagem com Blur";
    namedWindow(window_name);
    imshow(window_name, image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Blur Gaussiano

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    //Aplicando blur na imagem com kernel Gaussiano 3x3
    Mat image_blurred_with_3x3_kernel;
    GaussianBlur(image, image_blurred_with_3x3_kernel, Size(3, 3), 0);

    //Aplicando blur na imagem com kernel Gaussiano 5x5
    Mat image_blurred_with_5x5_kernel;
    GaussianBlur(image, image_blurred_with_5x5_kernel, Size(5, 5), 0);

    String window_name = "Imagem com Blur Gaussiano";
    namedWindow(window_name);
    imshow(window_name, image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Erosão de imagens

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    //Aplicando erosão na imagem com kernel 3x3
    Mat image_eroded_with_3x3_kernel;
    erode(image, image_eroded_with_3x3_kernel, getStructuringElement(MORPH_RECT, Size(3, 3)));

    //Aplicando erosão na imagem com kernel 5x5
    Mat image_eroded_with_5x5_kernel;
    erode(image, image_eroded_with_5x5_kernel, getStructuringElement(MORPH_RECT, Size(5, 5)));

    String window_name = "Imagem com Erosao";
    namedWindow(window_name);
    imshow(window_name, image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Dilatação de imagens

```cpp
#include <opencv2/opencv.hpp>
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char** argv) {
    Mat image = imread("/Path/Para/A/Imagem.jpg");

    //Dilatando a imagem com kernel 3x3
    Mat image_dilated_with_3x3_kernel;
    dilate(image, image_dilated_with_3x3_kernel, getStructuringElement(MORPH_RECT, Size(3, 3)))

    //Diatando a imagem com kernel 5x5
    Mat image_dilated_with_5x5_kernel;
    dilate(image, image_dilated_with_5x5_kernel, getStructuringElement(MORPH_RECT, Size(5, 5)))

    String window_name = "Imagem Dilatada";
    namedWindow(window_name);
    imshow(window_name, image);
    waitKey(0);
    destroyAllWindows();

    return 0;
}
```

Obrigado pela atenção!

INSTITUIÇÃO EXECUTORA

Centro de Informática UFPE

FADE UFPE

COORDENADORA

MCTI FUTURO

Softex

APOIO

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO