OpenCv Kata

!nombres!
Alejandro Salgado G

6 de Febrero de 2016

Tabla de contenidos

Partes de OpenCv

OpenCv se divide en 8 partes.

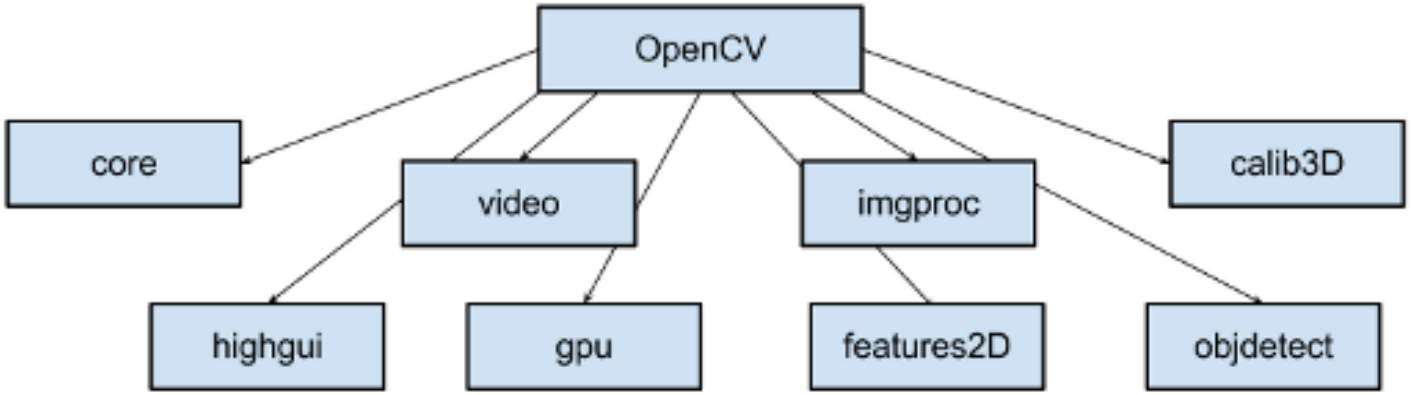


Imagen tomada de:
https://www.packtpub.com/sites/default/files/Article-Images/8812_03_01.png

Leer y mostrar una imagen

En este ejemplo se necesitan 2 modulos de OpenCv

1. Core para poder leer la imagen
2. Highgui para poder mostrarla

En Codigo se ve asi:

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/core/core.hpp"
```

Adicionalmente se debe usar el namespace o espacio de nombre de OpenCv:

```
using namepace cv;
```

Leer y mostrar una imagen

Por ultimo esta el codigo fuente:

```
int main(int argc, char *argv[]){

    Mat image = imread( /* ruta de la imagen */ );
    namedWindow( "Window", CV_WINDOW_AUTOSIZE );
    imshow( "Window", image );
    waitKey(0);

    return 0;

}
```

Cambiar medida

En este ejemplo se necesitan 3 modulos de OpenCv

1. Core para poder leer la imagen
2. Highgui para poder mostrarla
3. Imgproc para poder modificarla

En Codigo se ve asi:

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/core/core.hpp"
#include "opencv2/imgproc/imgproc.hpp"
```

Adicionalmente se debe usar el namespace o espacio de nombre de OpenCv:

```
using namepace cv;
```

Cambiar medida

Por ultimo esta el codigo fuente:

```
int main(int argc, char *argv[]){

    Mat original, resized, saved;

    original = imread( /* ruta de la imagen */ );
    namedWindow("Original image", CV_WINDOW_AUTOSIZE );
    imshow("Original image", original );

    resize(original, resized, Size(), 0.5, 0.5, INTER_LINEAR);
    namedWindow("Resized image", CV_WINDOW_AUTOSIZE);
    imshow("Resized image", resized );

    waitKey(0);

    return 0;

}
```

Guardar imagen

Al igual que en el ejemplo anterior aqui se necesitan 3 modulos de OpenCv

1. Core para poder leer la imagen
2. Highgui para poder mostrarla
3. Imgproc para poder modificarla

En Codigo se ve asi:

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/core/core.hpp"
#include "opencv2/imgproc/imgproc.hpp"
```

Adicionalmente se debe usar el namespace o espacio de nombre de OpenCv:

```
using namepace cv;
```

Guardar imagen

codigo fuente:

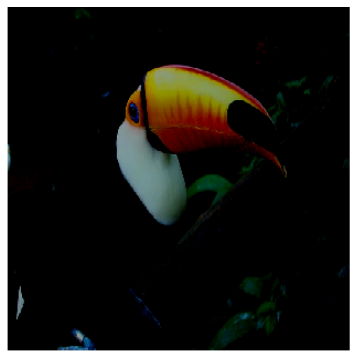
```
int main(int argc, char *argv[]){

    Mat original, resized, saved;

    original = imread( /* ruta de la imagen */ );
    namedWindow( "Original image", CV_WINDOW_AUTOSIZE );
    imshow( "Original image", original );

    resize( original, resized, Size(), 0.5, 0.5, INTER_LINEAR );

    imwrite( /* ruta de la imagen */, resized);

    namedWindow( "Image saved", CV_WINDOW_AUTOSIZE);
    saved = imread( /* ruta de la imagen */ );
    imshow("Image saved", saved);

    waitKey(0);

    return 0;
}
```

Que es la manipulacion de pixeles?

Utilizar los elementos que componen una imagen con el fin de obtener informacion de esta o modificarla para un fin en especifico.

Original

Suma C = 100



Resta C = 100

Imagen tomada de:
https://yetanotherlog.wordpress.com/2011/11/18/log-0-manipulacion-basica-de-imagenes/

Que es Thresholding?

Es una herramienta que nos permite conocer determinadas posiciones de pixeles con el objetivo de efectuar cambios especificos en una imagen.

Imagen tomada de:
http://docs.opencv.org/2.4/doc/tutorials/imgproc/threshold/threshold.h

Como funciona el Thresholding para imagenes a color?

Como funciona el Thresholding para imagenes grises?

Manipulacion de pixeles (Thresholding)

Igual que en el ejemplo anterior se necesitan 3 modulos de OpenCv

1. Core para poder leer la imagen
2. Highgui para poder mostrarla
3. Imgproc para poder modificarla

En Codigo se ve asi:

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/core/core.hpp"
#include "opencv2/imgproc/imgproc.hpp"
```

Adicionalmente se debe usar el namespace o espacio de nombre de OpenCv:

```
using namepace cv;
```

Manipulacion de pixeles (Thresholding)

Este codigo fuente esta dividido en 4 partes

1. Convertir una imagen de color a escala de grises
2. Thresholding para grises
3. Thresholding para color
4. Main

Manipulacion de pixeles (Thresholding)

Convertir a gris:

```
Mat convertGray(Mat &color){

    Mat gray; gray.create(color.rows, color.cols, CV_8UC1);
    cvtColor(color, gray, CV_BGR2GRAY);

    namedWindow("Gray image", CV_WINDOW_AUTOSIZE);
    imshow("Gray image", gray);

    return gray;

}
```

Manipulacion de pixeles (Thresholding)

Thresholding para grises:

```
Mat thresholdingGray(Mat &image, uchar thresholdValue){

    for(int i=0; i < image.rows; i++){
        for(int j=0; j<image.cols; j++){

            uchar value = image.at<uchar>(i,j);
            if(value > thresholdValue) image.at<uchar>(i,j)=255;

        }
    }

    return image;
}
```

Manipulacion de pixeles (Thresholding)

Thresholding para color:

```
Mat thresholdingColor(Mat &image, uchar thresholdValue){

    for(int i=0; i < image.rows; i++){
        for(int j=0; j<image.cols; j++){

            int sum = image.at<Vec3b>(i,j)[0] +
                      image.at<Vec3b>(i,j)[1] +
                      image.at<Vec3b>(i,j)[2];

            uchar average = sum/3;

            if(average > thresholdValue){

                image.at<Vec3b>(i,j)[0] = 255;
                image.at<Vec3b>(i,j)[1] = 255;
                image.at<Vec3b>(i,j)[2] = 255;

            }
        }
    }

    return image;
}
```

Manipulacion de pixeles (Thresholding)

Main:

```
int main(int argc, char *argv[]){

    Mat color = imread( /* ruta de la imagen */ );

    namedWindow("Normal image", CV_WINDOW_AUTOSIZE);
    imshow("Normal image", color);

    Mat gray = convertGray(color);

    Mat colorConverted = thresholdingColor( color, 100 );

    namedWindow("Color image after thresholding", CV_WINDOW_AUTOSIZE);
    imshow("Color image after thresholding", colorConverted);

    Mat grayConverted = thresholdingGray( gray, 100 );

    namedWindow("Gray image after thresholding", CV_WINDOW_AUTOSIZE);
    imshow("Gray image after thresholding", grayConverted);

    waitKey(0);

    return 0;

}
```

Mas informacion

Todos los ejemplos que se mostraron estan implementados en

https://github.com/AlejandroSalgadoG/ImageProcessing

Alli encontraran la implementacion en C++ y en Python algunos ejemplos adicionales

Para correrlos solo se necesita entrar a la carpeta del ejemplo deseado y ejecutar los comandos

1. make
2. make exe