EBook Gratis

APRENDIZAJE opencv

Free unaffiliated eBook created from
Stack Overflow contributors.

#opencv

Tabla de contenido

Acerca de 1

Capítulo 1: Empezando con opencv 2

Observaciones 2

Versiones 2

OpenCV 3 2

OpenCV 2 2

Examples 3

Cargar y mostrar una imagen con OpenCV 3

Construye e instala OpenCV desde la fuente 4

Prepararse para la construcción 4

Construir e instalar 5

Instalación de prueba 5

Hola ejemplo mundial en Java 6

Obtener imagen de webcam 6

Java 6

C ++ 7

Pitón 8

Primeros pasos con OpenCV 3.1 en Windows 8

¿Qué y por qué OPENCV? 14

Capítulo 2: Acceso a píxeles 16

Observaciones 16

Examples 16

Acceda a valores de píxeles individuales con cv :: Mat :: at () 16

Acceso eficiente a píxeles usando cv :: Mat :: ptr puntero 17

Configuración y obtención de valores de píxeles de una imagen gris en C ++ 17

Acceso a píxeles alternativos con Matiterator 19

Acceso a píxeles en el tapete 21

Capítulo 3: Cargar y guardar varios formatos de medios 23

Examples 23

Cargando imagenes 23

Cargando Videos ... 23
Captura en vivo ... 24
Videos de ahorro ... 24
Guardar imágenes ... 25
Capítulo 4: Clasificadores en cascada ... 27
Examples ... 27
Usando los clasificadores en cascada para detectar la cara ... 27
Pitón ... 27
Código ... 27
Resultado ... 27
Clasificadores en cascada para detectar rostro con Java ... 28
Java ... 28
Detección de rostro mediante clasificador haar en cascada ... 29
C ++ ... 29
Capítulo 5: Contraste y brillo en C ++ ... 32
Sintaxis ... 32
Parámetros ... 32
Observaciones ... 32
g (i, j) = .f (i, j) + ... 32
Examples ... 33
Ajuste de brillo y contraste de una imagen en c ++ ... 33
Capítulo 6: Creando un video ... 35
Introducción ... 35
Examples ... 35
Creando un video con OpenCV (Java) ... 35
Capítulo 7: Detección de bordes ... 36
Sintaxis ... 36
Parámetros ... 36
Examples ... 36
Algoritmo Canny ... 36
Algoritmo Canny - C ++ ... 37

Cálculo de umbrales ... 38
Video de Canny Edge de Webcam Capture - Python ... 38
Creación de prototipos de umbrales de Canny Edge con barras de seguimiento ... 38
Capítulo 8: Detección de manchas ... 40
Examples ... 40
Detección de manchas circulares ... 40
Capítulo 9: Detección de objetos ... 42
Examples ... 42
Coincidencia de plantillas con Java ... 42
Código fuente de Java ... 42
RESULTADO ... 42
Capítulo 10: Dibujar formas (línea, círculo, ..., etc.) en C ++ ... 44
Introducción ... 44
Examples ... 44
Muestra de formas de dibujo ... 44
Capítulo 11: Estructuras basicas ... 47
Introducción ... 47
Examples ... 47
Tipo de datos ... 47
Estera ... 47
Vec ... 48
Capítulo 12: Funciones de dibujo en Java ... 50
Examples ... 50
Dibujar el rectángulo en la imagen ... 50
Capítulo 13: Genere y compile opencv 3.1.0-dev para Python2 en Windows usando CMake y Visu 51
Observaciones ... 51
Examples ... 60
Lectura de imagen y conversión a escala de grises ... 60
Capítulo 14: Inicialización de OpenCV en Android ... 62
Examples ... 62
Inicialización asíncrona ... 62

Administrador de OpenCV 63
Inicialización Estática 63
Capítulo 15: Instalación de OpenCV 65
Introducción 65
Examples 65
Instalación de OpenCV en Ubuntu 65
Capítulo 16: Modificación de contenido de imagen 68
Examples 68
Establecer imagen completa a un color sólido 68
Modificación píxel a píxel de las imágenes 68
Modificación del color de la imagen en OpenCV - kmeans (). Para escanear todos los píxeles 69
Capítulo 17: Mostrar imagen OpenCV 70
Examples 70
Lectura básica y visualización de una imagen 70
Leyendo MJPEG desde camara IP 70
Display Image OpenCV Java 71
Capítulo 18: Procesamiento de imágenes 72
Sintaxis 72
Parámetros 72
Observaciones 72
Examples 72
Suavizar imágenes con desenfoque gaussiano en C ++ 72
Umbral 73
Filtrado bilateral 74
Capítulo 19: Usando los clasificadores en cascada en Java 76
Sintaxis 76
Parámetros 76
Examples 77
Obtención de una imagen estática, detección de elementos en ella y salida de los resultado 77
Detectar imágenes desde un dispositivo de video 78
Convertir un objeto Mat en un objeto BufferedImage 79
Detecciones en Detecciones 79

Capítulo 20: Utilizando VideoCapture con OpenCV Python 83
Examples 83
Leyendo cuadros desde un video pre-capturado 83
Utilizando VideoCapture con OpenCV Java 83
Creditos 85

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: opencv

It is an unofficial and free opencv ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official opencv.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con opencv

Observaciones

Esta sección proporciona una descripción general de qué es opencv y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema grande dentro de la apertura, y vincular a los temas relacionados. Dado que la Documentación para opencv es nueva, es posible que deba crear versiones iniciales de los temas relacionados.

Versiones

OpenCV 3

Versión	Fecha de lanzamiento
3.2	2016-12-23
3.1	2015-12-18
3.0	2015-06-03
3.0 RC1	2015-04-23
3.0 beta	2014-11-07
3.0 alfa	2014-08-21

OpenCV 2

Versión	Fecha de lanzamiento
2.4.13	2016-05-19
2.4.12	2015-07-30
2.4.11	2015-02-25
2.4.10	2014-10-01
2.4.9	2014-04-14
2.3.1	2011-08-17

Versión	Fecha de lanzamiento
2.3.0	2011-07-04
2.2.0	2010-12-05
2.1.0	2010-04-06
2.0.0	2009-10-01
1.0.0	2006-10-19

Examples

Cargar y mostrar una imagen con OpenCV

Con este ejemplo, veremos cómo cargar una imagen en color desde el disco y mostrarla utilizando las funciones integradas de OpenCV. Podemos usar los enlaces C / C ++, Python o Java para lograr esto.

En C ++:

```
#include <opencv2/core.hpp>
#include <opencv2/highgui.hpp>

#include <iostream>

using namespace cv;

int main(int argc, char** argv) {
    // We'll start by loading an image from the drive
    Mat image = imread("image.jpg", CV_LOAD_IMAGE_COLOR);

    // We check that our image has been correctly loaded
    if(image.empty()) {
        std::cout << "Error: the image has been incorrectly loaded." << std::endl;
        return 0;
    }

    // Then we create a window to display our image
    namedWindow("My first OpenCV window");

    // Finally, we display our image and ask the program to wait for a key to be pressed
    imshow("My first OpenCV window", image);
    waitKey(0);

    return 0;
}
```

En Python:

```
import sys
import cv2
```

```
# We load the image from disk
img = cv2.imread("image.jpg", cv2.CV_LOAD_IMAGE_COLOR)

# We check that our image has been correctly loaded
if img.size == 0
    sys.exit("Error: the image has not been correctly loaded.")

# We create a window to display our image
cv2.namedwindow("My first OpenCV window")

# We display our image and ask the program to wait until a key is pressed
cv2.imshow("My first OpenCV window", img)
cv2.waitKey(0)

# We close the window
cv2.destroyAllWindows()
```

En Java:

```
import org.opencv.core.Core;
import org.opencv.core.Mat;
import org.opencv.core.CvType;
import org.opencv.highgui.Highgui;
public class Sample{
public static void main (String[] args) {

    //Load native opencv library
    System.loadLibrary(Core.NATIVE_LIBRARY_NAME);

    //Read image from file first param:file location ,second param:color space
    Mat img = imread("image.jpg",CV_LOAD_IMAGE_COLOR);

    //If the image is successfully read.
    if (img.size() == 0) {
        System.exit(1);
    }
}
```

HighGui no tiene ventanas nombradas ni equivalentes en el show en Java abierto. Utilice swing o swt para mostrar la imagen.

Construye e instala OpenCV desde la fuente

Esta es una guía paso a paso para instalar OpenCV 3 en un sistema Linux basado en Debian desde la fuente. Los pasos deben seguir siendo los mismos para otras distribuciones, simplemente reemplace los comandos relevantes del administrador de paquetes al instalar paquetes para la compilación.

Nota: Si no tiene ganas de perder el tiempo creando cosas o le disgusta el terminal, lo más probable es que pueda instalar OpenCV desde la GUI del administrador de paquetes Synaptic. Sin embargo, estas bibliotecas a menudo están desactualizadas.

Prepararse para la construcción

Ejecute los siguientes comandos en su terminal para instalar los paquetes requeridos:

```
sudo apt-get update
sudo apt-get install build-essential
sudo apt-get install cmake git libgtk2.0-dev pkg-config \
                     libavcodec-dev libavformat-dev libswscale-dev
```

Los siguientes paquetes son opcionales:

```
sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev \
                     libpng-devlibtiff-dev libjasper-dev libdc1394-22-dev
```

Ejecute el siguiente comando para obtener el código fuente de OpenCV y preparar la compilación:

```
mkdir ~/src
cd ~/src
git clone https://github.com/opencv/opencv.git
cd opencv
mkdir build && cd build
```

Construir e instalar

Incluimos los ejemplos en la compilación, pero siéntase libre de omitirlos. También siéntase libre de establecer otras banderas y personalizar su construcción como mejor le parezca.

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D INSTALL_PYTHON_EXAMPLES=ON \
      -D INSTALL_C_EXAMPLES=ON ..
```

Si CMake no informó ningún error o faltaron bibliotecas, continúe con la compilación.

```
make -j$(nproc)
```

Si no se produjeron errores, podemos continuar con la instalación de OpenCV en el sistema:

```
sudo make install
```

Ahora OpenCV debería estar disponible para su sistema. Puede usar las siguientes líneas para saber dónde se instaló OpenCV y qué bibliotecas se instalaron:

```
pkg-config --cflags opencv  # get the include path (-I)
pkg-config --libs opencv    # get the libraries path (-L) and the libraries (-l)
```

Instalación de prueba

Primero construimos los ejemplos de C ++:

```
cd ~/src/opencv/samples
cmake .
make
```

Si no se produjeron errores, ejecute una muestra, por ejemplo

```
./cpp/cpp-example-edge
```

Si se ejecuta el ejemplo, entonces las bibliotecas de C ++ se instalan correctamente.

A continuación, prueba los enlaces de Python:

```
python
>> import cv2
>> print cv2.__version__
```

Si estos comandos importan OpenCV e imprimen la versión correcta sin quejarse, entonces los enlaces de Python están correctamente instalados.

Felicidades, acabas de construir e instalar OpenCV. Feliz programacion

Para Mac, consulte aquí la instalación de OpenCV en Mac OS X

Hola ejemplo mundial en Java

Imagen de OpenCv leída del sistema de archivos en Java

```
import org.opencv.core.Core;
import org.opencv.core.Mat;
import org.opencv.imgcodecs.Imgcodecs;

public class Giris {
    public static void main(String[] args) {
        //Load native library
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        //image container object
        Mat imageArray;
        //Read image from file system
        imageArray=Imgcodecs.imread("C:\\Users\\mesutpiskin\\sample.jpg");
        //Get image with & height
        System.out.println(imageArray.rows());
        System.out.println(imageArray.cols());
    }
}
```

Obtener imagen de webcam

Muestra un video en vivo tomado de una cámara web utilizando la clase VideoCapture de OpenCV con Java, C / C ++ y Python.

Java

```
import org.opencv.core.Core;
import org.opencv.core.Mat;
import org.opencv.videoio.VideoCapture;

public class Camera {
    public static void main(String[] args) {
        // Load Native Library
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        // image container object
        Mat imageArray = new Mat();
        // Video device acces
        VideoCapture videoDevice = new VideoCapture();
        // 0:Start default video device 1,2 etc video device id
        videoDevice.open(0);
        // is contected
        if (videoDevice.isOpened()) {
        // Get frame from camera
            videoDevice.read(imageArray);
            // image array
            System.out.println(imageArray.toString());
            // Release video device
            videoDevice.release();
        } else {
            System.out.println("Error.");
        }
    }
}
```

C ++

```
#include "opencv2/opencv.hpp"
#include "iostream"

int main(int, char**) {
    // open the first webcam plugged in the computer
    cv::VideoCapture camera(0);
    if (!camera.isOpened()) {
        std::cerr << "ERROR: Could not open camera" << std::endl;
        return 1;
    }

    // create a window to display the images from the webcam
    cv::namedWindow("Webcam", CV_WINDOW_AUTOSIZE);

    // this will contain the image from the webcam
    cv::Mat frame;

    // capture the next frame from the webcam
    camera >> frame;

    // display the frame until you press a key
    while (1) {
        // show the image on the window
        cv::imshow("Webcam", frame);
        // wait (10ms) for a key to be pressed
        if (cv::waitKey(10) >= 0)
            break;
    }
```

```
    return 0;
}
```

Pitón

```
import numpy as np
import cv2

# Video source - can be camera index number given by 'ls /dev/video*
# or can be a video file, e.g. '~/Video.avi'
cap = cv2.VideoCapture(0)

while(True):
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Our operations on the frame come here
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Display the resulting frame
    cv2.imshow('frame',gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```

Primeros pasos con OpenCV 3.1 en Windows

Instalamos OpenCV 3.1.0 en Windows y empezamos. Hay dos formas de instalar OpenCV en Windows. Una es descargar el instalador y ejecutarlo. Otro es construir desde la fuente.

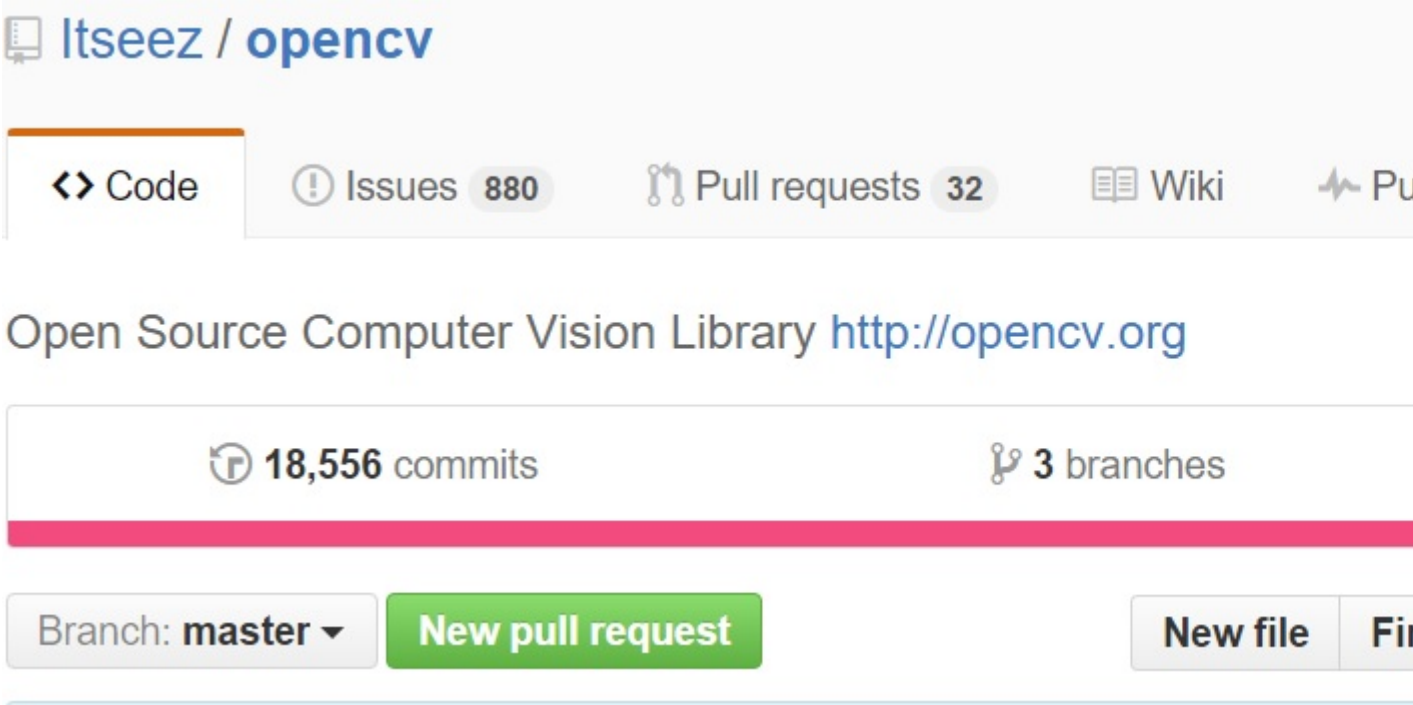
Esta es la forma más fácil de instalar OpenCV y comenzar. OpenCV ofrece binarios precompilados para instalar en Windows aquí . Una vez que finalice la descarga, extráigala e instálela en la ruta elegida.

ProTip: asegúrese de que su ruta OpenCV no incluya ningún espacio. Por lo tanto, es mejor si lo instala en el directorio C: \ o D: \ root

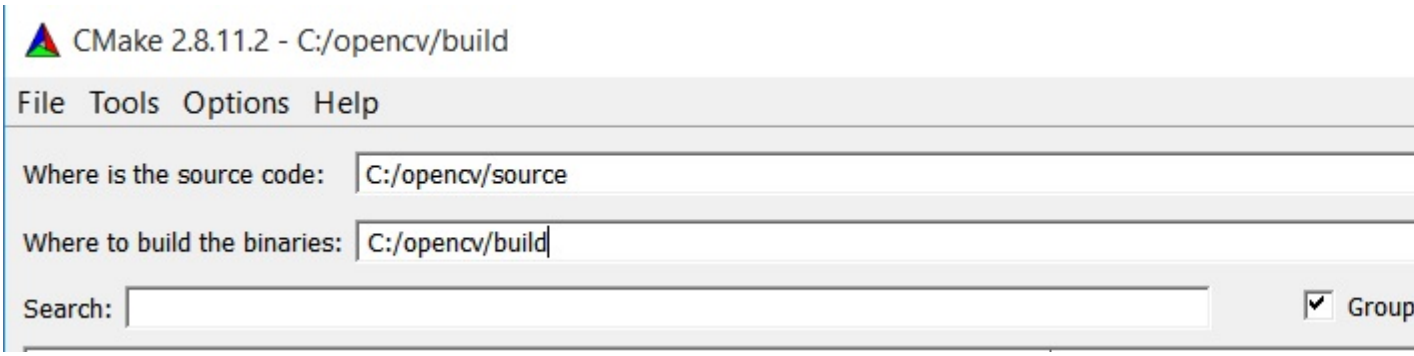
El problema con el método anterior es que no puede usar los módulos opencv_contrib. Además, no viene con todas las herramientas y bibliotecas de terceros. Por lo tanto, si desea utilizar todos estos, simplemente siga adelante.

Explicaré el mínimo mínimo para instalar OpenCV desde la fuente. Para más avanzado, refiérase aquí .

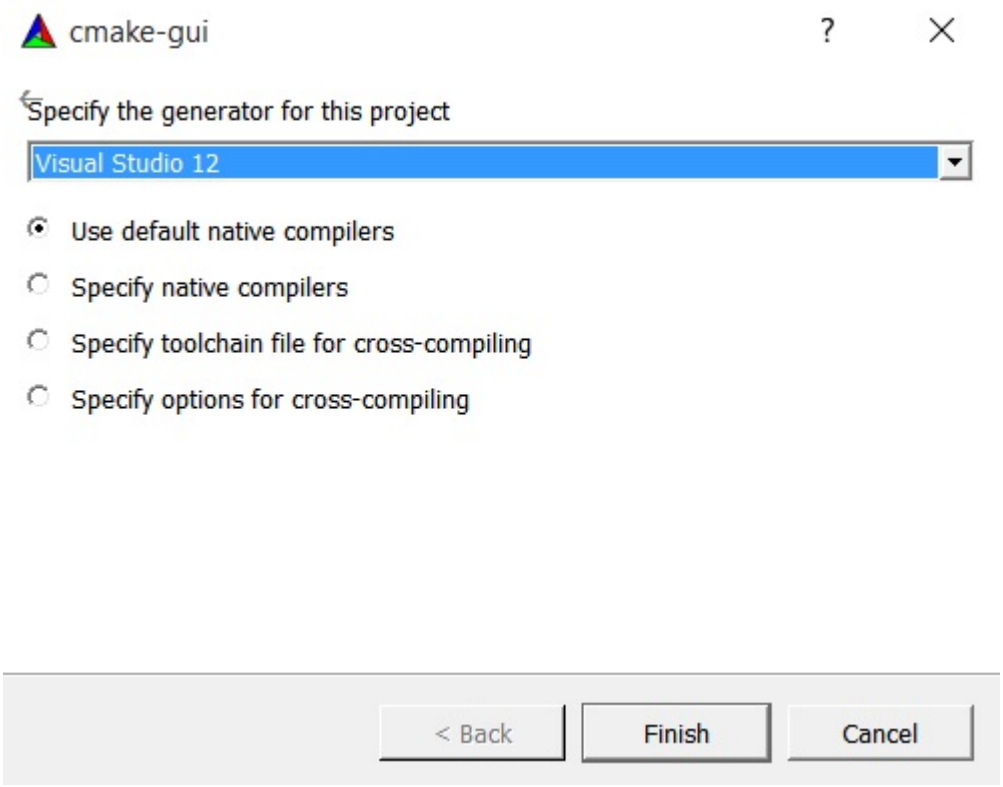
- Instale CMake .
- Clone la fuente de OpenCV desde https://github.com/Itseez/opencv.git en algún directorio que no tenga espacios. Vamos a referirlo como "OpenCVdir".



- Ahora, abra CMake GUI y agregue su directorio de origen (OpenCVdir) al menú Orígenes y genere el directorio en el menú de compilación. **Consejo:** si no hay un directorio de compilación, cree uno en su carpeta de opencv.

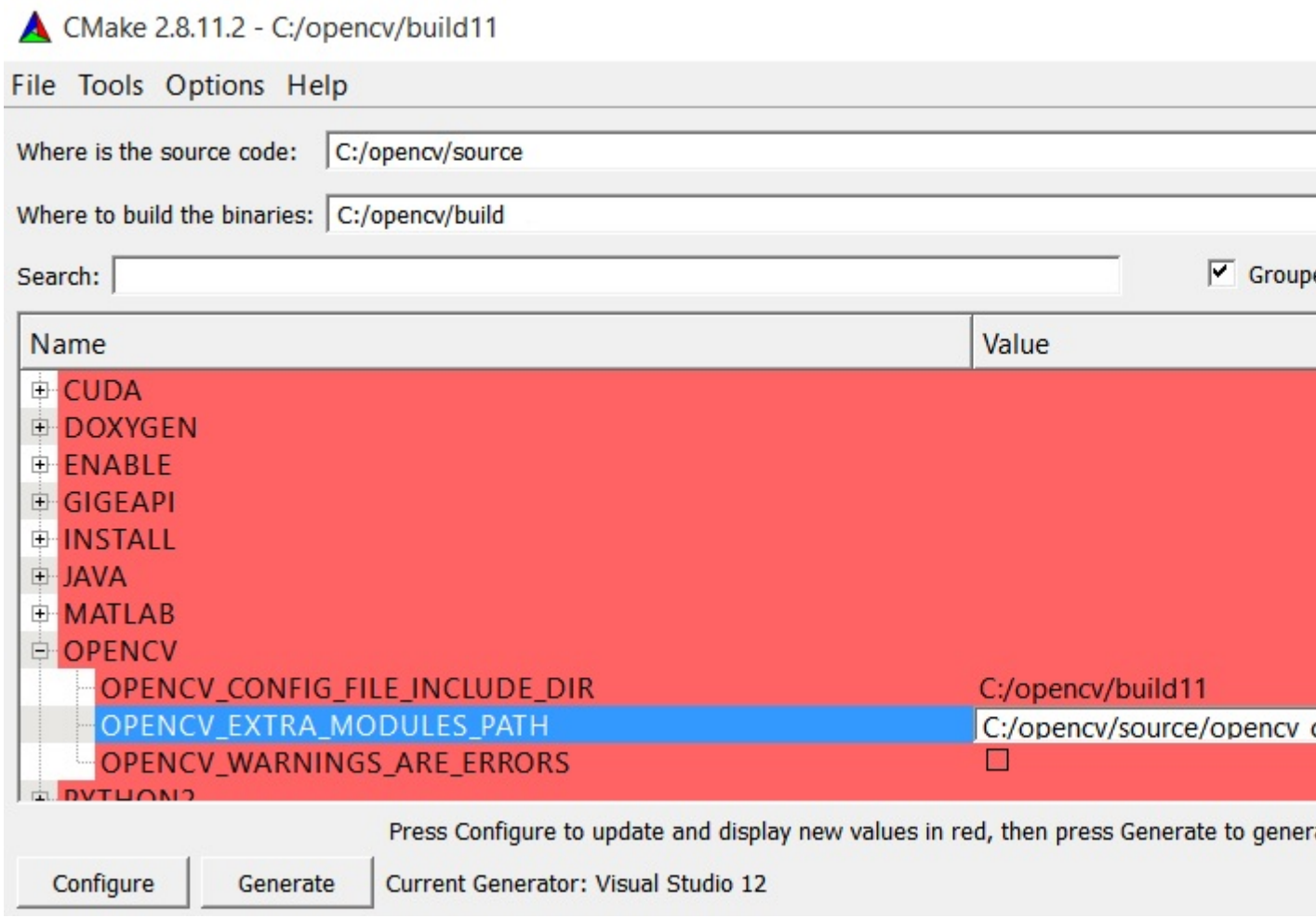


- Haga clic en Configurar y seleccione la versión del compilador de Visual Studio. Tenía Visual Studio 2013 Professional de 32 bits, así que elegí el compilador de Visual Studio 12.



Sugerencia: Puede descargar Visual Studio 2013 Professional desde aquí. Viene con 30 días de prueba + 90 días de seguimiento extendido después de iniciar sesión.

- Presione Finalizar y CMake cargará todos los paquetes automáticamente. Puede agregar o eliminar paquetes. Presiona Configurar de nuevo.
- Si desea compilar con módulos opencv_contrib adicionales, debe descargarlos desde aquí . Luego, extráigalos y agregue el directorio opencv_contrib / modules a su CMake como se muestra a continuación.

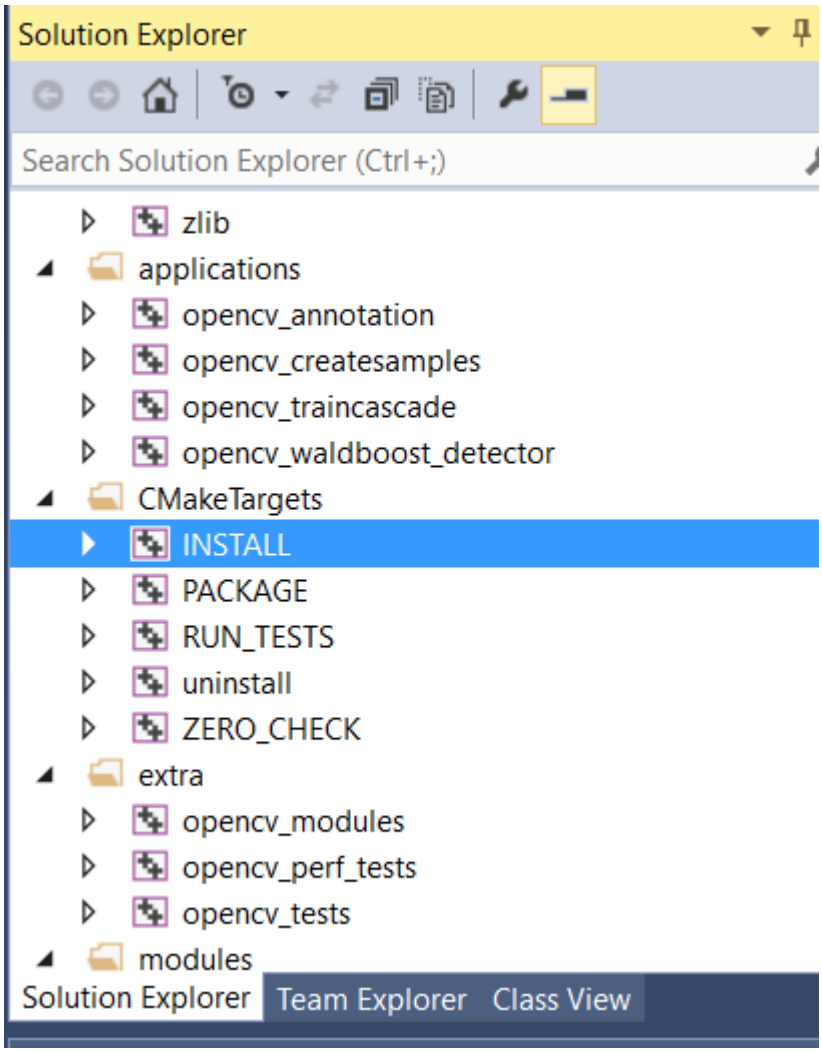


- Ahora presiona Configurar nuevamente y luego presiona Generar.
- Cerrar CMake. Vaya a la carpeta your_opencv \ build y abra el archivo denominado 'OpenCV.sln'. - Se abrirá Visual Studio. Ahora, ejecútalo en Debug

modo y liberación

modo.

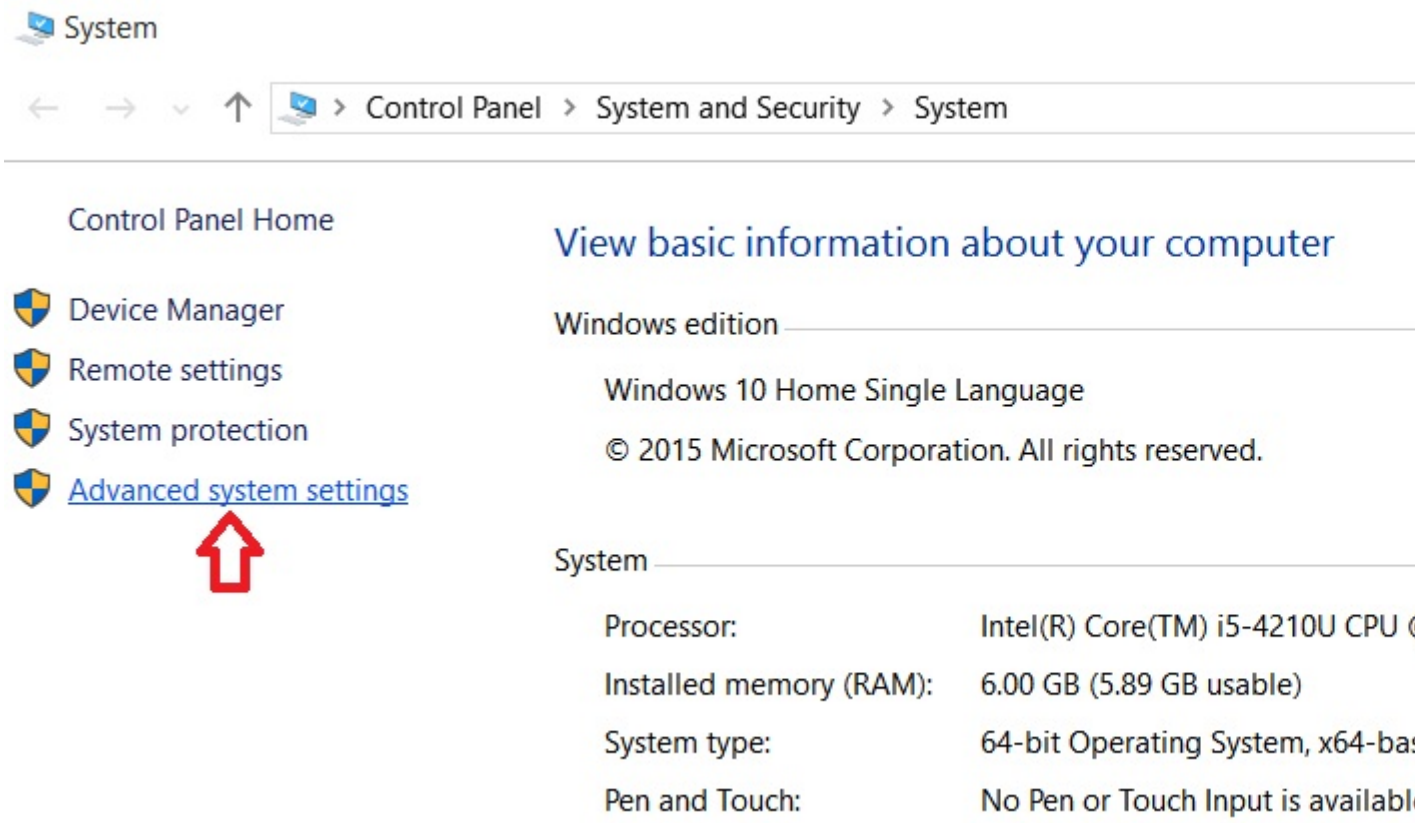
- Ahora, en el explorador de soluciones en la parte superior derecha de su Visual Studio, seleccione INSTALAR proyecto y compílelo.



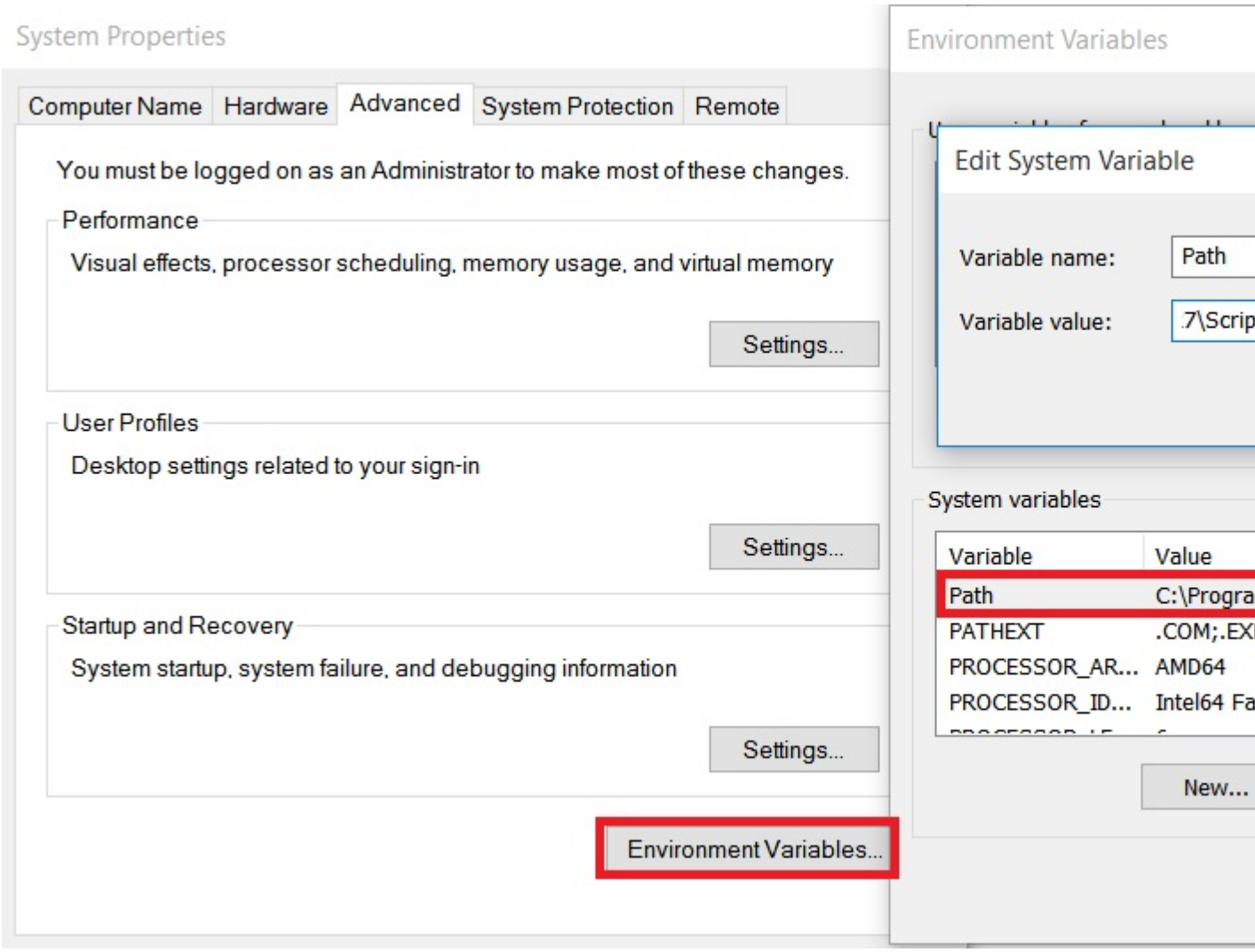
¡¡Hurra!! Disfruta de tu OpenCV.

Agregando OpenCV include directory a la variable PATH de las Variables de Ambiente:

- Vaya a Propiedades del sistema y haga clic en Configuración avanzada del sistema.



- Ahora, haga clic en Variables de entorno >> Ruta >> Editar.



- Aquí, agregue la carpeta bin ubicada en su OpenCVdir / build / install / x86 / vc ** / bin a esta variable. Tenga cuidado de no reemplazar los valores de ruta existentes.
- Después de esto, debe reiniciar su sistema para que las variables de entorno cambien y ahora está listo para comenzar.

¿Qué y por qué OPENCV?

OpenCV (Open Source Computer Vision Library) es una biblioteca de software de visión de computadora y de aprendizaje automático de código abierto. Fue construido para varios propósitos, tales como aprendizaje automático, visión artificial, algoritmo, operaciones matemáticas, captura de video, procesamiento de imágenes, etc. Con el paso de los años, se ha vuelto muy popular entre los investigadores y desarrolladores en cuanto a su soporte en diferentes plataformas (Windows, Linux y Linux). , android, ios). También tiene envoltorio en varios lenguajes de programación de renombre. Según el acuerdo de licencia, tiene acceso para que las empresas utilicen y modifiquen el código.

La biblioteca contiene más de 2500 algoritmos optimizados, que tienen una excelente precisión en el rendimiento y la velocidad. Estos algoritmos se pueden usar para detectar y reconocer rostros, identificar objetos, clasificar acciones humanas en videos, rastrear movimientos de cámara, rastrear objetos en movimiento, extraer modelos 3D de objetos, producir nubes de

puntos 3D desde cámaras estéreo, unir imágenes para producir una alta resolución imagen de una escena completa, encuentre imágenes similares de una base de datos de imágenes, elimine los ojos rojos de las imágenes tomadas con flash, siga los movimientos de los ojos, reconozca paisajes y establezca marcadores para superponerlos con realidad aumentada, etc. OpenCV cuenta con excelentes personas y comunidades involucradas como usuarios , desarrolladores e investigadores, el número es más de 47 mil y el número estimado de descargas supera los 7 millones. La biblioteca se encuentra ampliamente en empresas profesionales, grupos de investigación y otros grupos.

Muchas empresas bien establecidas como Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota que emplean la biblioteca, hay muchas empresas nuevas como Applied Minds, VideoSurf y Zeitera, que hacen un uso extensivo de OpenCV. Los usos desplegados de OpenCV abarcan desde unir imágenes de streetview, detectar intrusiones en el video de vigilancia en Israel, monitorear el equipo de la mina en China, ayudar a los robots a navegar y recoger objetos en Willow Garage, detectar accidentes de ahogamiento en piscinas en Europa, ejecutar arte interactivo en España y Nueva York, revisando las pistas en busca de escombros en Turquía, inspeccionando las etiquetas de los productos en fábricas de todo el mundo hasta la rápida detección de rostros en Japón. Tiene interfaces C ++, C, Python, Java y MATLAB y es compatible con Windows, Linux, Android y Mac OS. OpenCV se inclina principalmente hacia aplicaciones de visión en tiempo real y aprovecha las instrucciones MMX y SSE cuando están disponibles. Actualmente se están desarrollando activamente las interfaces CUDA y OpenCL. Hay más de 500 algoritmos y aproximadamente 10 veces más funciones que componen o soportan esos algoritmos. OpenCV está escrito de forma nativa en C ++ y tiene una interfaz con plantilla que funciona perfectamente con los contenedores STL.

Información recogida de la web oficial.

Lea Empezando con opencv en línea: https://riptutorial.com/es/opencv/topic/800/empezando-con-opencv

Capítulo 2: Acceso a píxeles

Observaciones

Tenga cuidado de conocer el tipo de `cv::Mat` que está tratando. Por ejemplo, si tiene un `cv::Mat` de tipo `CV_8UC3` , pero acceda a él con `image.at<uchar>(r,c)` no se producirá ningún error, pero su programa tendrá algún comportamiento inesperado.

Examples

Acceda a valores de píxeles individuales con cv :: Mat :: at ()

Para acceder a los valores de píxeles en un objeto OpenCV `cv::Mat` , primero debe conocer el *tipo* de su matriz.

Los tipos más comunes son:

- `CV_8UC1` para imágenes de escala de grises de 1 canal de 8 bits;
- `CV_32FC1` para imágenes de escala de grises de 1 canal de punto flotante de 32 bits;
- `CV_8UC3` para imágenes en color de 8 canales y 3 canales; y
- `CV_32FC3` para imágenes de color de 3 canales de punto flotante de 32 bits.

La configuración predeterminada con `cv::imread` creará una matriz `CV_8UC3` .

Para acceder a píxeles individuales, la forma más segura, aunque no la más eficiente, es usar el método `cv::Mat::at<T>(r,c)` donde `r` es la *fila* de la matriz y `c` es la *columna* . El argumento de la plantilla depende del tipo de la matriz.

Digamos que usted tiene un `cv::Mat image` . Según su tipo, el método de acceso y el tipo de color del píxel serán diferentes.

- Para `CV_8UC1` : `uchar pixelGrayValue = image.at<uchar>(r,c)` .
- Para `CV_8UC3` : `cv::Vec3b pixelColor = image.at<cv::Vec3b>(r,c)` . El objeto `cv::Vec3b` representa un triplete de valores `uchar` (enteros entre 0 y 255).
- Para `CV_32FC1` : `float pixelGrayValue = image.at<float>(r,c)` .
- Para `CV_32FC3` : `cv::Vec3f pixelColor = image.at<cv::Vec3f>(r,c)` . El objeto `cv::Vec3f` representa un triplete de valores `float` .

Tenga en cuenta que OpenCV representa imágenes en orden de ***fila mayor*** , como, por ejemplo, Matlab o como la convención en Álgebra. Por lo tanto, si sus coordenadas de píxeles son `(x,y)` , entonces accederá al píxel utilizando `image.at<..>(y,x)` .

Alternativamente, `at<>` también es compatible con el acceso a través de un solo argumento `cv::Point` .
En este caso, el acceso se realiza en la *columna principal* :

```
image.at<..>(cv::Point(x,y));
```

Eche un vistazo a la documentación de OpenCV para obtener más detalles sobre este método.

Acceso eficiente a píxeles usando cv :: Mat :: ptr puntero

Si la eficiencia es importante, una forma rápida de iterar sobre píxeles en un objeto `cv::Mat` es usar su `ptr<T>(int r)` para obtener un puntero al principio de la fila `r` (índice basado en 0).

Según el tipo de matriz, el puntero tendrá una plantilla diferente.

- Para `CV_8UC1` : `uchar* ptr = image.ptr<uchar>(r);`
- Para `CV_8UC3` : `cv::Vec3b* ptr = image.ptr<cv::Vec3b>(r);`
- Para `CV_32FC1` : `float* ptr = image.ptr<float>(r);`
- Para `CV_32FC3` : `cv::Vec3f* ptr = image.ptr<cv::Vec3f>(r);`

Este objeto `ptr` se puede usar para acceder al valor de píxel en la fila `r` y la columna `c` llamando a `ptr[c]` .

Para ilustrar esto, aquí hay un ejemplo donde cargamos una imagen del disco e invertimos sus canales Azul y Rojo, operando píxel por píxel:

```
#include <opencv2/core.hpp>
#include <opencv2/imgproc.hpp>
#include <opencv2/highgui.hpp>

int main(int argc, char** argv) {
    cv::Mat image = cv::imread("image.jpg", CV_LOAD_IMAGE_COLOR);

    if(!image.data) {
        std::cout << "Error: the image wasn't correctly loaded." << std::endl;
        return -1;
    }

    // We iterate over all pixels of the image
    for(int r = 0; r < image.rows; r++) {
        // We obtain a pointer to the beginning of row r
        cv::Vec3b* ptr = image.ptr<cv::Vec3b>(r);

        for(int c = 0; c < image.cols; c++) {
            // We invert the blue and red values of the pixel
            ptr[c] = cv::Vec3b(ptr[c][2], ptr[c][1], ptr[c][0]);
        }
    }

    cv::imshow("Inverted Image", image);
    cv::waitKey();

    return 0;
}
```

Configuración y obtención de valores de píxeles de una imagen gris en C ++

```
// PixelAccessTutorial.cpp : Defines the entry point for the console
```

```
// Environment: Visual studio 2015, Windows 10
// Assumptions: Opecv is installed configured in the visual studio project
// Opencv version: OpenCV 3.1

#include "stdafx.h"
#include<opencv2/core/core.hpp>
#include<opencv2/highgui/highgui.hpp>
#include<opencv2/imgproc/imgproc.hpp>
#include<string>
#include<iostream>

int main()
{

    cv::Mat imgOriginal;        // input image
    cv::Mat imgGrayscale;        // grayscale of input image

    std::cout << "Please enter an image filename : ";
    std::string img_addr;
    std::cin >> img_addr;

    std::cout << "Searching for " + img_addr << std::endl;

    imgOriginal = cv::imread(img_addr);            // open image

        if (imgOriginal.empty()) {                                    // if unable to open
image
            std::cout << "error: image not read from file\n\n";        // show error message
on command line
            return(0);                                                // and exit program
        }

    cv::cvtColor(imgOriginal, imgGrayscale, CV_BGR2GRAY);        // convert to grayscale

    const int channels = imgGrayscale.channels();
    printf("Number of channels = %d", channels);

    cv::Mat output ;
    imgGrayscale.copyTo(output); // Just to make sure the Mat objects are of the same size.

    //Set the threshhold to your desired value
    uchar threshhold = 127;

    if (channels == 1)
    {
        for (int x = 0; x<imgGrayscale.rows; x++) {
            for (int y = 0; y<imgGrayscale.cols; y++) {
                // Accesssing values of each pixel
                if (imgGrayscale.at<uchar>(x, y) >= threshhold) {
                    // Setting the pixel values to 255 if it's above the threshold
                    output.at<uchar>(x, y) = 254;
                }
                else if (imgGrayscale.at<uchar>(x, y) < threshhold) {
                    // Setting the pixel values to 255 if it's below the threshold
                    output.at<uchar>(x, y) = 0;
                }
                else {
                    // Just in case
                    printf("The value at (%d, %d) are not right. Value: %d\n", x, y,
imgGrayscale.at<uchar>(x, y));
                }
```

```
            }
        }
    }
    else if (channels == 3)
    {
        // This is only for  gray scale images
        printf("\tThe image has 3 channels. The function does not support images with 3
channels.\n");
    }

    //Create windows to show image
    cv::namedWindow("Gray scale", CV_WINDOW_AUTOSIZE);
    cv::namedWindow("Binary", CV_WINDOW_AUTOSIZE);

    cv::imshow("Gray scale", imgGrayscale);
    cv::imshow("Binary", output);

    cv::waitKey(0);                    // hold windows open until user presses a key

    return 0;
}
```

Acceso a píxeles alternativos con Matiterator

No es la mejor manera de iterar a través de los píxeles; sin embargo, es mejor que cv :: Mat :: at <T>.

Supongamos que tiene una imagen en color en su carpeta y desea iterar cada píxel de esta imagen y borrar los canales verdes y rojos (tenga en cuenta que este es un ejemplo, puede hacerlo de formas más optimizadas);

```
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>


int main(int argc, char **argv)
{

// Create a container
cv::Mat im;

//Create a vector
cv::Vec3b *vec;

// Create an mat iterator
cv::MatIterator_<cv::Vec3b> it;

// Read the image in color format
im = cv::imread("orig1.jpg", 1);

// iterate through each pixel
for(it = im.begin<cv::Vec3b>(); it != im.end<cv::Vec3b>(); ++it)
{
    // Erase the green and red channels
    (*it)[1] = 0;
    (*it)[2] = 0;
}
```

```
// Create a new window
cv::namedWindow("Resulting Image");

// Show the image
cv::imshow("Resulting Image", im);

// Wait for a key
cv::waitKey(0);

return 0;
}
```

Para compilar esto con Cmake:

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

La imagen original:

La imagen procesada:

Tenga en cuenta que no tocamos sólo el canal azul.

Para más información: http://docs.opencv.org/2.4/opencv_tutorials.pdf Página: 145

Acceso a píxeles en el tapete

El acceso de píxeles individuales en la estructura de la matriz OpenCV se puede lograr de múltiples maneras. Para entender cómo acceder, es mejor aprender primero los tipos de datos.

Estructuras básicas explica los tipos de datos básicos. En breve, `CV_<bit-depth>{U|S|F}C(<number_of_channels>)` es la estructura básica de un tipo. Junto con eso, es importante entender las estructuras de `Vec` .

```
typedef Vec<type, channels> Vec< channels>< one char for the type>
```

donde type es uno de `uchar, short, int, float, double` y los caracteres para cada tipo son `b, s, i, f, d` , respectivamente.

Por ejemplo, Vec2b indica un `unsigned char vector of 2 channels` .

Considere `Mat mat(R,C,T)` donde R es #rows, C es #cols y T es tipo. Algunos ejemplos para acceder a la coordenada (i, j) de `mat` son:

2D:

```
If the type is CV_8U or CV_8UC1 ---- //they are alias
mat.at<uchar>(i,j) // --> This will give char value of index (i,j)
//If you want to obtain int value of it
(int)mat.at<uchar>(i,j)

If the type is CV_32F or CV_32FC1 ---- //they are alias
mat.at<float>(i,j) // --> This will give float value of index (i,j)
```

3D:

```
If the type is CV_8UC2 or CV_8UC3 or more channels
mat.at<Vec2b/Vec3b>(i,j)[k] // note that (k < #channels)
//If you want to obtain int value of it
(int)mat.at<uchar>(i,j)[k]

If the type is CV_64FC2 or CV_64FC3
mat.at<Vec2d/Vec3d>(i,j)[k] // note that k < #channels
```

Tenga en cuenta que es muy importante ingresar el tipo correcto en `<...>` , de lo contrario, puede tener un error de tiempo de ejecución o resultados no deseados.

Lea Acceso a píxeles en línea: https://riptutorial.com/es/opencv/topic/1957/acceso-a-pixeles

Capítulo 3: Cargar y guardar varios formatos de medios

Examples

Cargando imagenes

```
#include <highgui.h>

//...

cv::Mat img = cv::imread("img.jpg");
```

...

Cargando Videos

Muestra cómo usar `cv::VideoCapture` . Aquí está el ejemplo de cargar video desde un archivo:

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include "opencv2/core/core.hpp"
#include <iostream>

using namespace cv;
VideoCapture videoSource;
Mat frame;
#define VIDEO_PATH "video.avi"

int main()
{
    //Open video
    if (!videoSource.open(VIDEO_PATH))
    {
        std::cout<<"Video not found at "<<VIDEO_PATH<<std::endl;
        return 1;     // Exit if fail
    }
    videoSource.set(CV_CAP_PROP_CONVERT_RGB, 1);

    int cameraWidth = videoSource.get(CV_CAP_PROP_FRAME_WIDTH);
    int cameraHeight = videoSource.get(CV_CAP_PROP_FRAME_HEIGHT);
    float cameraAspectRatio = cameraWidth / cameraHeight;

    std::cout <<"Camera resolution: " << cameraWidth<<", "<<cameraHeight<<" aspect ratio:
"<<cameraAspectRatio<< std::endl;

    while(true)
    {
        videoSource >> frame;
        if(frame.empty())
            break;
        //Resize frame
```

```
        cv::resize(frame, frame, cv::Size(320, 320 / cameraAspectRatio));
        imshow("frame", frame);
        waitKey(20);
    }
   waitKey(0);
   return 0;
}
```

Captura en vivo

Muestra cómo usar `cv::VideoCapture` con, por ejemplo, una cámara web. Capturar fotogramas desde la webcam y visualizarlos. Aquí está el código de ejemplo:

```
#include <iostream>

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

using namespace cv;
VideoCapture videoSource;
Mat frame;

int main()
{
    if(!videoSource.open(0)) //if more cameras available use 1,2,...
        return 1;

    while(true)
    {
        videoSource >> frame;
        if(frame.empty())
            break;
        imshow("Webcam", frame); //or any kinf of precessing
        if(waitKey(1)==27)
            break;//stop capturing is ESC pressed
    }

   return 0;
}
```

Videos de ahorro

Muestra cómo usar cv :: VideoWriter.

```
#include "opencv2/highgui/highgui.hpp"
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, char* argv[])
{
    VideoCapture cap(0); // open the video camera no. 0

    if (!cap.isOpened())  // if not success, exit program
```

```
    {
        cout << "ERROR: Cannot open the video file" << endl;
        return -1;
    }

    namedWindow("MyVideo",CV_WINDOW_AUTOSIZE); //create a window called "MyVideo"

    double dWidth = cap.get(CV_CAP_PROP_FRAME_WIDTH); //get the width of frames of the video
    double dHeight = cap.get(CV_CAP_PROP_FRAME_HEIGHT); //get the height of frames of the
video

    cout << "Frame Size = " << dWidth << "x" << dHeight << endl;

    Size frameSize(static_cast<int>(dWidth), static_cast<int>(dHeight));

    VideoWriter oVideoWriter ("D:/MyVideo.avi", CV_FOURCC('P','I','M','1'), 20, frameSize,
true); //initialize the VideoWriter object

    if ( !oVideoWriter.isOpened() ) //if not initialize the VideoWriter successfully, exit the
program
   {
        cout << "ERROR: Failed to write the video" << endl;
        return -1;
   }

   while (1)
   {

    Mat frame;

    bool bSuccess = cap.read(frame); // read a new frame from video

    if (!bSuccess) //if not success, break loop
    {
         cout << "ERROR: Cannot read a frame from video file" << endl;
         break;
    }

    oVideoWriter.write(frame); //writer the frame into the file

    imshow("MyVideo", frame); //show the frame in "MyVideo" window

    if (waitKey(10) == 27) //wait for 'esc' key press for 30ms. If 'esc' key is pressed, break
loop
   {
        cout << "esc key is pressed by user" << endl;
        break;
   }
}

return 0;
```

}

Guardar imágenes

En realidad, el ejemplo de Live Capture es bueno para capturar imágenes, así que lo estoy usando para capturar imágenes y guardarlas en una carpeta.

```
#include <fstream>
#include <string>

#include <opencv2/highgui/highgui.hpp>
#include <opencv2/core/core.hpp>
#include <opencv2/imgproc/imgproc.hpp>


int main()
{
    std::stringstream file; // to write the file name

    cv::VideoCapture cap(0); // create a capture object

    int counter = 0; // Create counter

    while(true) // infinite loop
    {
        cv::Mat frame; // Create a object

        cap.read(frame); // read the frame

        file << "/home/user/path_to_your_folder/image" << counter << ".jpg"; // file name

        cv::imwrite(file.str(), frame);

        counter++; // increment the counter
    }

   return 0;
}
```

Lea Cargar y guardar varios formatos de medios en línea:
https://riptutorial.com/es/opencv/topic/6658/cargar-y-guardar-varios-formatos-de-medios

Capítulo 4: Clasificadores en cascada

Examples

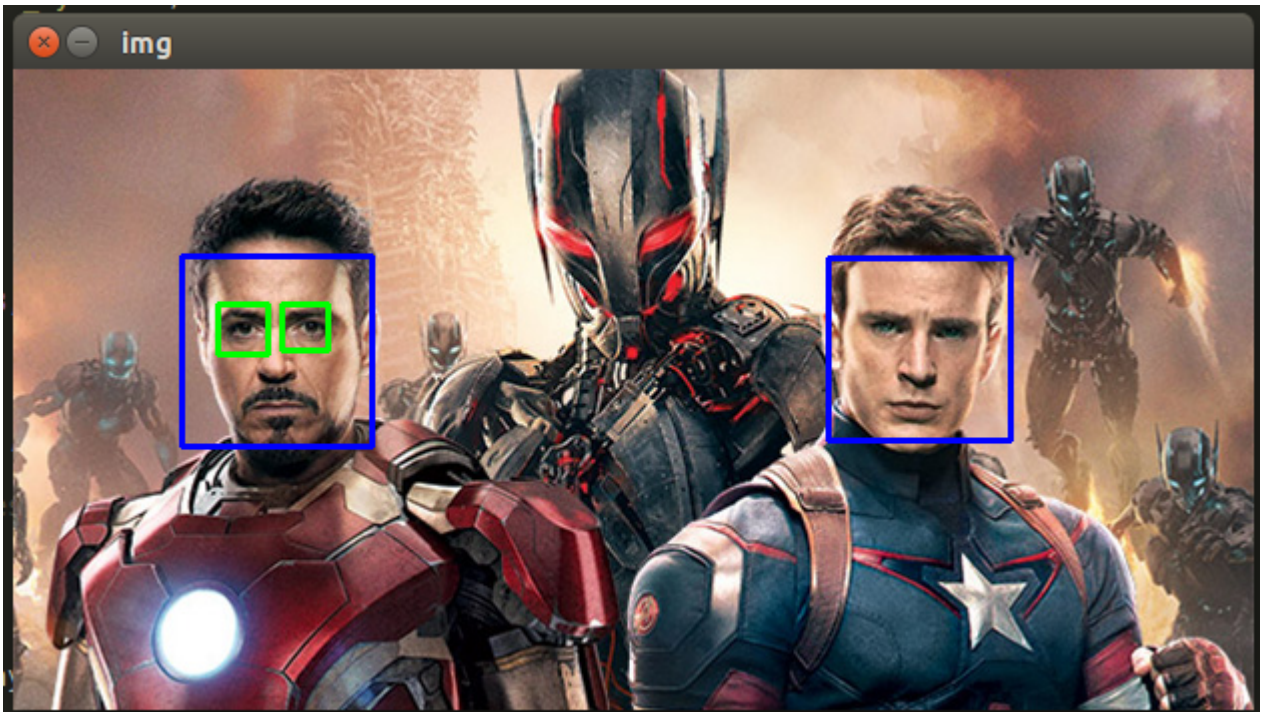
Usando los clasificadores en cascada para detectar la cara

Pitón

Código

```
import numpy as np
import cv2

#loading haarcascade classifiers for face and eye
#You can find these cascade classifiers here
#https://github.com/opencv/opencv/tree/master/data/haarcascades
#or where you download opencv inside data/haarcascades

face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')

#loading the image
img = cv2.imread('civil_war.jpg')

#converting the image to gray scale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

#detecting face in the grayscale image
faces = face_cascade.detectMultiScale(gray, 1.3, 5)

#iterate through each detected face
for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2) #draw rectangle to each detected face

    #take the roi of the face (region of interest)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]

    #detect the eyes
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:

        #draw rectangle for each eye
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

#show the image
cv2.imshow('img',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Resultado



Clasificadores en cascada para detectar rostro con Java

Java

Código

```
import org.opencv.core.Mat;
import org.opencv.core.MatOfRect;
import org.opencv.core.Point;
import org.opencv.core.Rect;
import org.opencv.core.Scalar;
import org.opencv.highgui.Highgui;
import org.opencv.highgui.VideoCapture;
import org.opencv.objdetect.CascadeClassifier;

public class FaceDetector{

    public static void main(String[] args) {

        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        //Create object
        CascadeClassifier faceDetector = new
CascadeClassifier(FaceDetector.class.getResource("haarcascade_frontalface_default.xml").getPath());


        //Read image
        Mat image = Highgui.imread("sourceimage.jpg");

     /*
        //Or read from webcam

         * Mat image=new Mat();
         *VideoCapture videoCapture=new VideoCapture(0);
         *videoCapture.read(image);
     */
```

```
        MatOfRect faceDetections = new MatOfRect();
        //Result list
        faceDetector.detectMultiScale(image, faceDetections);

        for (Rect rect : faceDetections.toArray()) {
            //Draw rectangle on result

            Core.rectangle(image, new Point(rect.x, rect.y), new Point(rect.x + rect.width,
rect.y + rect.height),
                    new Scalar(0, 255, 0));
        }

        //write result
        Highgui.imwrite("result.png", image);
        System.out.println("Succesfull");
    }


}
```

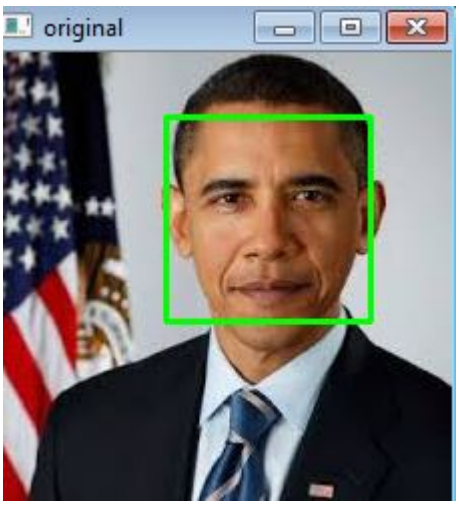
Resultado

Detección de rostro mediante clasificador haar en cascada.

C ++

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

#include <iostream>
#include <stdio.h>

using namespace std;
using namespace cv;

// Function Headers
void detectAndDisplay(Mat frame);

// Global variables
string face_cascade_name = "./data/haarcascade_frontalface_alt2.xml";
CascadeClassifier face_cascade;

// Function main
```

```
int main(void)
{
    // Load the cascade
    if (!face_cascade.load(face_cascade_name)){
        printf("--(!)Error on cascade loading\n");
        return (-1);
    }

    // Read the image file
    Mat frame = imread("d:/obama_01.jpg");

    // Apply the classifier to the frame
    if (!frame.empty())
        detectAndDisplay(frame);
    waitKey(0);
    return 0;
}

// Function detectAndDisplay
void detectAndDisplay(Mat frame)
{
    std::vector<Rect> faces;
    Mat frame_gray;

    cvtColor(frame, frame_gray, COLOR_BGR2GRAY);
    equalizeHist(frame_gray, frame_gray);

    // Detect faces
    face_cascade.detectMultiScale(frame_gray, faces, 1.1, 2, 0 | CASCADE_SCALE_IMAGE, Size(30,
30));

    for (int ic = 0; ic < faces.size(); ic++) // Iterate through all current elements
(detected faces)
    {
        Point pt1(faces[ic].x, faces[ic].y); // Display detected faces on main window - live
stream from camera
        Point pt2((faces[ic].x + faces[ic].height), (faces[ic].y + faces[ic].width));
        rectangle(frame, pt1, pt2, Scalar(0, 255, 0), 2, 8, 0);
    }

    imshow("original", frame);
}
```



Lea Clasificadores en cascada en línea:

https://riptutorial.com/es/opencv/topic/6562/clasificadores-en-cascada

Capítulo 5: Contraste y brillo en C ++

Sintaxis

- void cv :: Mat :: convertTo (OutputArray m, int rtype, double alpha = 1, double beta = 0) const

Parámetros

Parámetro	Detalles
metro	matriz de salida; Si no tiene un tamaño o tipo adecuado antes de la operación, se reasigna
rtype	tipo de matriz de salida deseada o, más bien, la profundidad, ya que la cantidad de canales es la misma que la entrada; si rtype es negativo, la matriz de salida tendrá el mismo tipo que la entrada
alfa	Factor de escala opcional. Esto cambia el contraste de una imagen. Los valores por debajo de 1 disminuyen el contraste y por encima de uno aumentan el contraste
beta	Delta opcional agregado a los valores escalados. Los valores positivos aumentan el brillo y los valores negativos disminuyen el brillo.

Observaciones

Contraste :

El contraste es la diferencia de luminancia o color que hace que un objeto (o su representación en una imagen o visualización) sea distinguible. Cuanto mayor sea la diferencia entre un píxel y sus vecinos, mayor será el contraste en esa área.

Brillo :

En otras palabras, el brillo es la percepción provocada por la luminancia de un objetivo visual. En términos de píxeles, cuanto más alto es el valor de un píxel, más brillante es ese píxel.

Ajustes de contraste y brillo:

$$g(i, j) = \alpha . f(i, j) + \beta$$

`f(x)` como píxeles de la imagen de origen y `g(x)` como píxeles de la imagen de salida.

`i` y `j` indican que el píxel está ubicado en la fila i-th y en la columna j-th.

Los parámetros `α > 0` y `β` menudo se denominan parámetros de ganancia y sesgo; a veces se dice que estos parámetros controlan el *contraste* y el *brillo* respectivamente.

Opencv tiene una función llamada convertTo () que puede aplicar estas dos operaciones.

Fuentes:
http://docs.opencv.org/trunk/d3/d63/classcv_1_1Mat.html#adf88c60c5b4980e05bb556080916978b http://opencv-srf.blogspot.ca/2013/07/change-contrast-of-image-or-video .html http://opencv-srf.blogspot.ca/2013/07/change-brightness.html

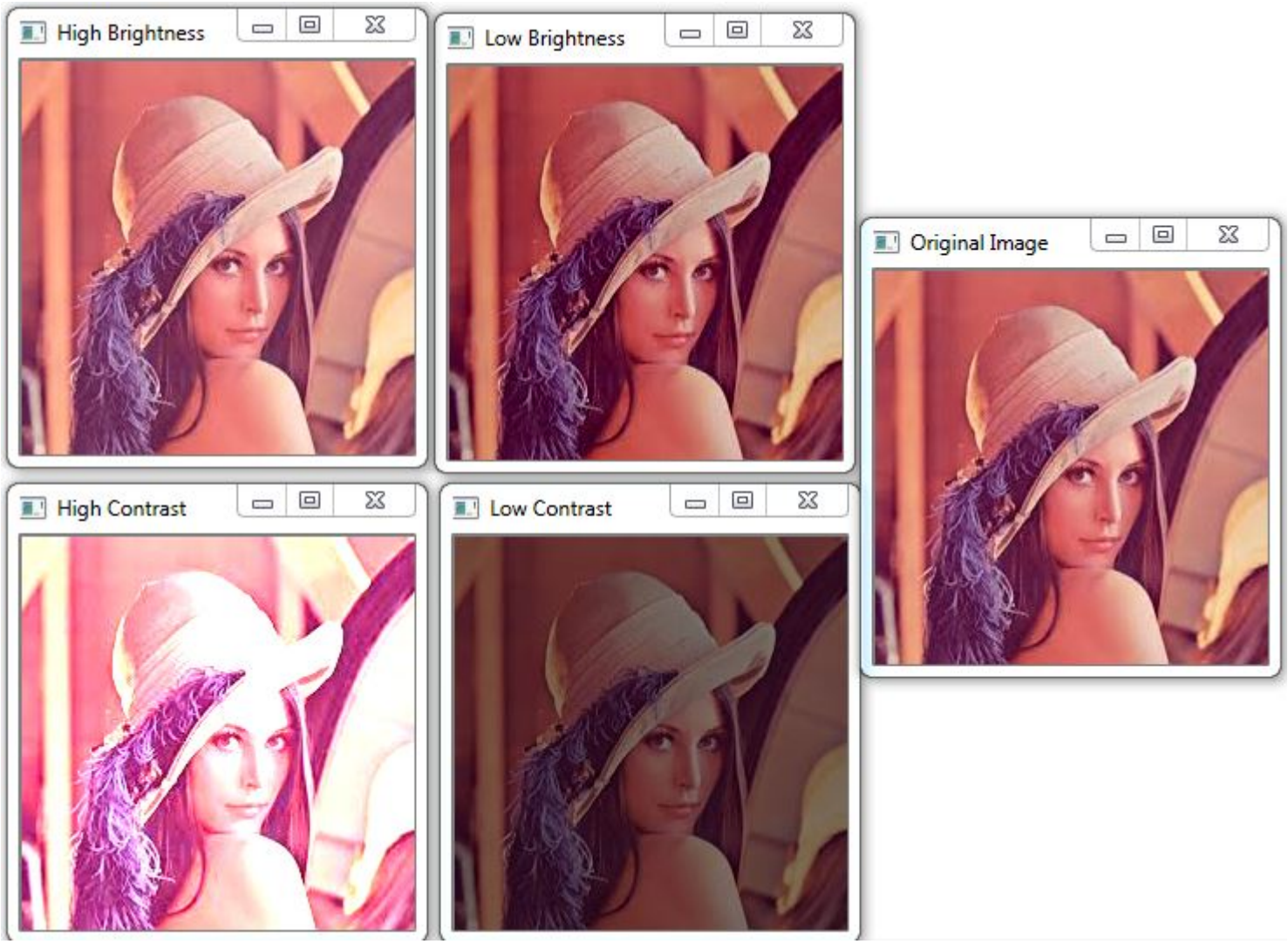
Examples

Ajuste de brillo y contraste de una imagen en c ++

```
// main.cpp : Defines the entry point for the console application.
//
#include "opencv2/highgui/highgui.hpp"
#include <iostream>

using namespace cv;
using namespace std;

int main(int argc, const char** argv)

{

    Mat img = imread("lena30.jpg", CV_LOAD_IMAGE_COLOR); //open and read the image


    if (img.empty())
    {
        cout << "Image cannot be loaded..!!" << endl;
        return -1;
    }

    Mat img_higher_contrast;
    img.convertTo(img_higher_contrast, -1, 2, 0); //increase the contrast (double)

    Mat img_lower_contrast;
    img.convertTo(img_lower_contrast, -1, 0.5, 0); //decrease the contrast (halve)

    Mat img_higher_brightness;
    img.convertTo(img_higher_brightness, -1, 1, 20); //increase the brightness by 20 for each
pixel

    Mat img_lower_brightness;
    img.convertTo(img_lower_brightness, -1, 1, -20); //decrease the brightness by 20 for each
pixel

    //create windows
    namedWindow("Original Image", CV_WINDOW_AUTOSIZE);
    namedWindow("High Contrast", CV_WINDOW_AUTOSIZE);
    namedWindow("Low Contrast", CV_WINDOW_AUTOSIZE);
    namedWindow("High Brightness", CV_WINDOW_AUTOSIZE);
```

```
    namedWindow("Low Brightness", CV_WINDOW_AUTOSIZE);
    //show the image
    imshow("Original Image", img);
    imshow("High Contrast", img_higher_contrast);
    imshow("Low Contrast", img_lower_contrast);
    imshow("High Brightness", img_higher_brightness);
    imshow("Low Brightness", img_lower_brightness);

    waitKey(0); //wait for key press
    destroyAllWindows(); //destroy all open windows
    return 0;
}
```

Salida del programa:



Lea Contraste y brillo en C ++ en línea: https://riptutorial.com/es/opencv/topic/6917/contraste-y-brillo-en-c-plusplus

Capítulo 6: Creando un video

Introducción

Siempre que trabaje con fuentes de video, es posible que desee guardar el resultado del procesamiento de la imagen en forma de un nuevo archivo de video. Para salidas de video simples, puede utilizar la clase VideoWriter incorporada de OpenCV, diseñada para esto. Es útil mirar algunos conceptos antes de usarlos. Estos conceptos son codec, es decir, decodificador y FourCC (código de cuatro caracteres).

Examples

Creando un video con OpenCV (Java)

```
VideoWriter videoWriter;
videoWriter = new VideoWriter(outputFile, VideoWriter.fourcc('x', '2','6','4'),
                fps, frameSize, isRGB);
//We have stated that we will use x264 as codec with FourCC
//For writing, we add the following method and it will write the image we give as parameter in
this call.
 public void Write(Mat frame) {
        if(videoWriter.isOpened()==false){
            videoWriter.release();
            throw new IllegalArgumentException("Video Writer Exception: VideoWriter not
opened,"
                    + "check parameters.");
        }
        //Write video
        videoWriter.write(frame);
    }

//With Video Capture for example, we can read images from the camera and write the same video

VideoCapture videoCapture = new VideoCapture(0);
Size frameSize = new Size((int) videoCapture.get(Videoio.CAP_PROP_FRAME_WIDTH), (int)
videoCapture.get(Videoio.CAP_PROP_FRAME_HEIGHT));
VideoWriter videoWriter = new VideoWriter("test.avi", VideoWriter.fourcc('x', '2','6','4'),
                videoCapture.get(Videoio.CAP_PROP_FPS), frameSize, true);
while (videoCapture.read(mat)) {
            videoWriter.write(mat);
        }
        videoCapture.release();
        videoWriter.release();
```

Lea Creando un video en línea: https://riptutorial.com/es/opencv/topic/9196/creando-un-video

Capítulo 7: Detección de bordes

Sintaxis

- bordes = cv2. Canny (imagen, umbral1, umbral2 [, bordes [, tamaño de abertura [, graduado de L2]]])
- void Canny (imagen de InputArray, bordes de OutputArray, doble threshold1, doble threshold2, int apertureSize = 3, bool L2gradient = false

Parámetros

Parámetro	Detalles
imagen	Imagen de entrada
bordes	Imagen de salida
umbral1	Primer umbral para el procedimiento de histéresis
umbral2	Segundo umbral para el procedimiento de histéresis
Tamaño de apertura	Tamaño de apertura para el operador Sobel
L2gradient	Indicador que indica si se debe usar un algoritmo más preciso para el gradiente de imagen

Examples

Algoritmo Canny

El algoritmo Canny es un detector de bordes más reciente diseñado como un problema de procesamiento de señales. En OpenCV, genera una imagen binaria que marca los bordes detectados.

Pitón:

```
import cv2
import sys

# Load the image file
image = cv2.imread('image.png')

# Check if image was loaded improperly and exit if so
if image is None:
    sys.exit('Failed to load image')
```

```
# Detect edges in the image. The parameters control the thresholds
edges = cv2.Canny(image, 100, 2500, apertureSize=5)

# Display the output in a window
cv2.imshow('output', edges)
cv2.waitKey()
```

Algoritmo Canny - C ++

A continuación se muestra un uso del algoritmo astuto en c ++. Tenga en cuenta que la imagen se convierte primero a imagen en escala de grises, luego el filtro gaussiano se usa para reducir el ruido en la imagen. Luego se usa el algoritmo Canny para la detección de bordes.

```
// CannyTutorial.cpp : Defines the entry point for the console application.
// Environment: Visual studio 2015, Windows 10
// Assumptions: Opecv is installed configured in the visual studio project
// Opencv version: OpenCV 3.1

#include "stdafx.h"
#include<opencv2/highgui/highgui.hpp>
#include<opencv2/imgproc/imgproc.hpp>
#include<string>
#include<iostream>



int main()
{

    //Modified from source:
https://github.com/MicrocontrollersAndMore/OpenCV_3_Windows_10_Installation_Tutorial
    cv::Mat imgOriginal;        // input image
    cv::Mat imgGrayscale;        // grayscale of input image
    cv::Mat imgBlurred;            // intermediate blured image
    cv::Mat imgCanny;            // Canny edge image

    std::cout << "Please enter an image filename : ";
    std::string img_addr;
    std::cin >> img_addr;

    std::cout << "Searching for " + img_addr << std::endl;

    imgOriginal = cv::imread(img_addr);            // open image

    if (imgOriginal.empty()) {                                    // if unable to open image
        std::cout << "error: image not read from file\n\n";        // show error message on
command line
        return(0);                                                // and exit program
    }

    cv::cvtColor(imgOriginal, imgGrayscale, CV_BGR2GRAY);        // convert to grayscale

    cv::GaussianBlur(imgGrayscale,            // input image
        imgBlurred,                            // output image
        cv::Size(5, 5),                        // smoothing window width and height in pixels
        1.5);                                // sigma value, determines how much the image
will be blurred
```

```
    cv::Canny(imgBlurred,            // input image
        imgCanny,                    // output image
        100,                        // low threshold
        200);                        // high threshold


    // Declare windows
    // Note: you can use CV_WINDOW_NORMAL which allows resizing the window
    // or CV_WINDOW_AUTOSIZE for a fixed size window matching the resolution of the image
    // CV_WINDOW_AUTOSIZE is the default
    cv::namedWindow("imgOriginal", CV_WINDOW_AUTOSIZE);
    cv::namedWindow("imgCanny", CV_WINDOW_AUTOSIZE);

    //Show windows
    cv::imshow("imgOriginal", imgOriginal);
    cv::imshow("imgCanny", imgCanny);

    cv::waitKey(0);                    // hold windows open until user presses a key
    return 0;
}
```

Cálculo de umbrales

Cálculo automático de los umbrales bajo y alto para la operación Canny en opencv

Video de Canny Edge de Webcam Capture - Python

```
import cv2


def canny_webcam():
    "Live capture frames from webcam and show the canny edge image of the captured frames."

    cap = cv2.VideoCapture(0)

    while True:
        ret, frame = cap.read()  # ret gets a boolean value. True if reading is successful (I
think). frame is an
        # uint8 numpy.ndarray

        frame = cv2.GaussianBlur(frame, (7, 7), 1.41)
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        edge = cv2.Canny(frame, 25, 75)

        cv2.imshow('Canny Edge', edge)

        if cv2.waitKey(20) == ord('q'):  # Introduce 20 milisecond delay. press q to exit.
            break

canny_webcam()
```

Creación de prototipos de umbrales de Canny Edge con barras de seguimiento

```
"""
CannyTrackbar function allows for a better understanding of
the mechanisms behind Canny Edge detection algorithm and rapid
prototyping. The example includes basic use case.

2 of the trackbars allow for tuning of the Canny function and
the other 2 help with understanding how basic filtering affects it.
"""
import cv2

def empty_function(*args):
    pass

def CannyTrackbar(img):
    win_name = "CannyTrackbars"

    cv2.namedWindow(win_name)
    cv2.resizeWindow(win_name, 500,100)

    cv2.createTrackbar("canny_th1", win_name, 0, 255, empty_function)
    cv2.createTrackbar("canny_th2", win_name, 0, 255, empty_function)
    cv2.createTrackbar("blur_size", win_name, 0, 255, empty_function)
    cv2.createTrackbar("blur_amp", win_name, 0, 255, empty_function)

    while True:
        cth1_pos = cv2.getTrackbarPos("canny_th1", win_name)
        cth2_pos = cv2.getTrackbarPos("canny_th2", win_name)
        bsize_pos = cv2.getTrackbarPos("blur_size", win_name)
        bamp_pos = cv2.getTrackbarPos("blur_amp", win_name)

        img_blurred = cv2.GaussianBlur(img.copy(), (trackbar_pos3 * 2 + 1, trackbar_pos3 * 2 +
1), bamp_pos)
        canny = cv2.Canny(img_blurred, cth1_pos, cth2_pos)
        cv2.imshow(win_name, canny)

        key = cv2.waitKey(1) & 0xFF
        if key == ord("c"):
            break

    cv2.destroyAllWindows()
    return canny

img = cv2.imread("image.jpg")
canny = CannyTrackbar(img)
cv2.imwrite("result.jpg", canny)
```

Lea Detección de bordes en línea: https://riptutorial.com/es/opencv/topic/6099/deteccion-de-bordes

Capítulo 8: Detección de manchas

Examples

Detección de manchas circulares

Este ejemplo muestra cómo encontrar manchas circulares en una imagen en escala de grises. La evaluación de la circularidad de una mancha se realiza utilizando el área y el perímetro (longitud del arco) del contorno. El punto central se evalúa utilizando los momentos del contorno.

```
#include "opencv/cv.h"
#include "opencv/highgui.h"
#include "opencv/cxcore.h"

using namespace cv;

int main(int argc, char** argv)
{
    Mat img = imread("image.jpg", CV_LOAD_IMAGE_GRAYSCALE);
    Mat resultImg;
    cvtColor(img, resultImg, CV_GRAY2BGR);

    // threshold the image with gray value of 100
   Mat binImg;
   threshold(img, binImg, 100, 255, THRESH_BINARY);

    // find the contours
    vector<vector<Point>> contours;
    vector<Vec4i> hierarchy;
    findContours(binImg, contours, hierarchy, CV_RETR_CCOMP, CV_CHAIN_APPROX_SIMPLE);

    if(contours.size() <= 0)
    {
        printf("no contours found");
        return 0;
    }
    // filter the contours
    vector<vector<Point>> filteredBlobs;
    Mat centers = Mat::zeros(0,2,CV_64FC1);
    for(int i = 0; i < contours.size(); i++)
    {
        // calculate circularity
        double area = contourArea(contours[i]);
        double arclength = arcLength(contours[i], true);
        double circularity = 4 * CV_PI * area / (arclength * arclength);
        if(circularity > 0.8)
        {
            filteredBlobs.push_back(contours[i]);

            //calculate center
            Moments mu = moments(contours[i], false);
            Mat centerpoint = Mat(1,2,CV_64FC1);
            centerpoint.at<double>(i,0) = mu.m10 / mu.m00; // x-coordinate
            centerpoint.at<double>(i,1) = mu.m01 / mu.m00; // y-coordinate
            centers.push_back(centerpoint);
        }
```

```
    }

    if(filteredBlobs.size() <= 0)
    {
        printf("no circular blobs found");
        return 0;
    }
    drawContours(resultImg, filteredBlobs, -1, Scalar(0,0,255), CV_FILLED, 8);

    imshow("Blobs",resultImg);
    waitKey(0);
    return 0;
}
```

Lea Detección de manchas en línea: https://riptutorial.com/es/opencv/topic/6589/deteccion-de-manchas

Capítulo 9: Detección de objetos

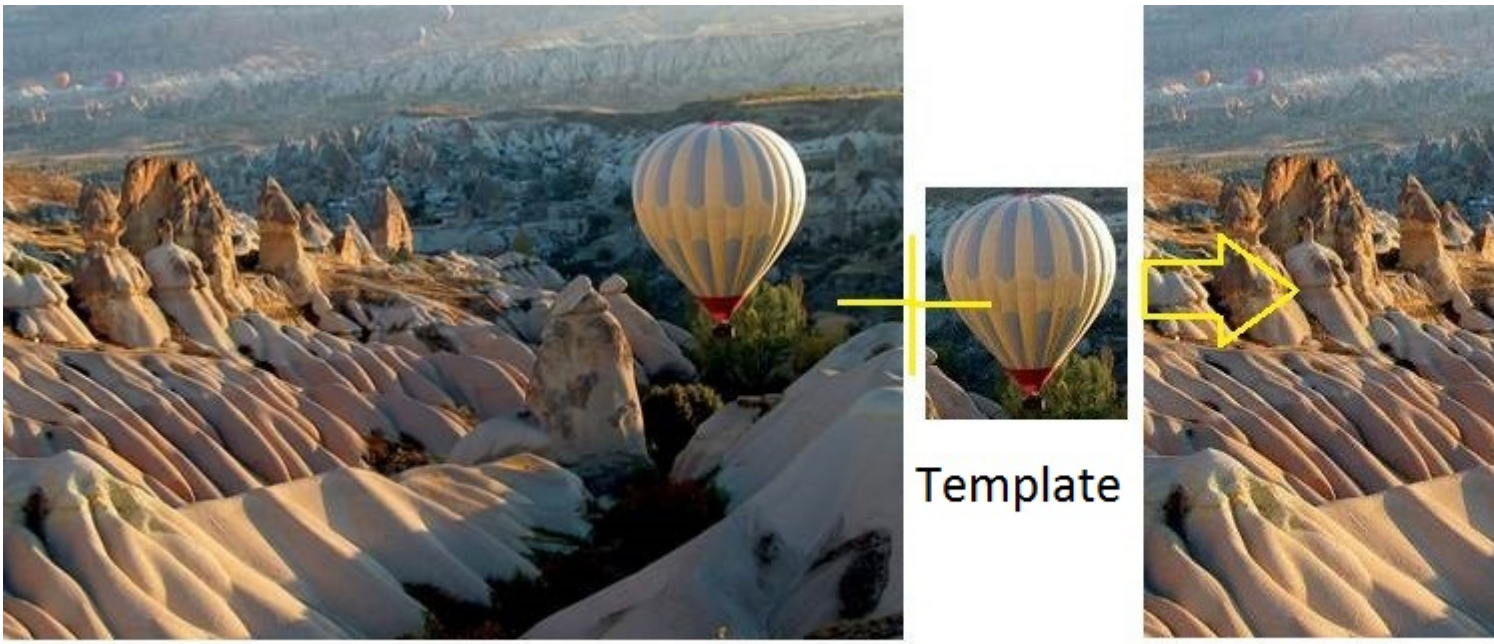
Examples

Coincidencia de plantillas con Java

Código fuente de Java

```
import org.opencv.core.Core;
import org.opencv.core.Core.MinMaxLocResult;
import org.opencv.core.Mat;
import org.opencv.core.Point;
import org.opencv.core.Scalar;
import org.opencv.imgcodecs.Imgcodecs;
import org.opencv.imgproc.Imgproc;

public class TemplateMatching {

    public static void main(String[] args) {

        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        Mat source=null;
        Mat template=null;
        String filePath="C:\\Users\\mesutpiskin\\Desktop\\Object Detection\\Template
Matching\\Sample Image\\";
        //Load image file
        source=Imgcodecs.imread(filePath+"kapadokya.jpg");
        template=Imgcodecs.imread(filePath+"balon.jpg");

        Mat outputImage=new Mat();
        int machMethod=Imgproc.TM_CCOEFF;
        //Template matching method
        Imgproc.matchTemplate(source, template, outputImage, machMethod);


        MinMaxLocResult mmr = Core.minMaxLoc(outputImage);
        Point matchLoc=mmr.maxLoc;
        //Draw rectangle on result image
        Imgproc.rectangle(source, matchLoc, new Point(matchLoc.x + template.cols(),
                matchLoc.y + template.rows()), new Scalar(255, 255, 255));

        Imgcodecs.imwrite(filePath+"sonuc.jpg", source);
        System.out.println("Complated.");
    }

}
```

RESULTADO



Resource Image

Result Im

Lea Detección de objetos en línea: https://riptutorial.com/es/opencv/topic/6735/deteccion-de-objetos

Capítulo 10: Dibujar formas (línea, círculo, ..., etc.) en C ++

Introducción

En OpenCV, se pueden dibujar numerosas formas como puntos, líneas, círculos, ..., etc. Hay una opción para rellenar una forma. El siguiente código se explica por sí mismo y muestra cómo se dibujan las formas.

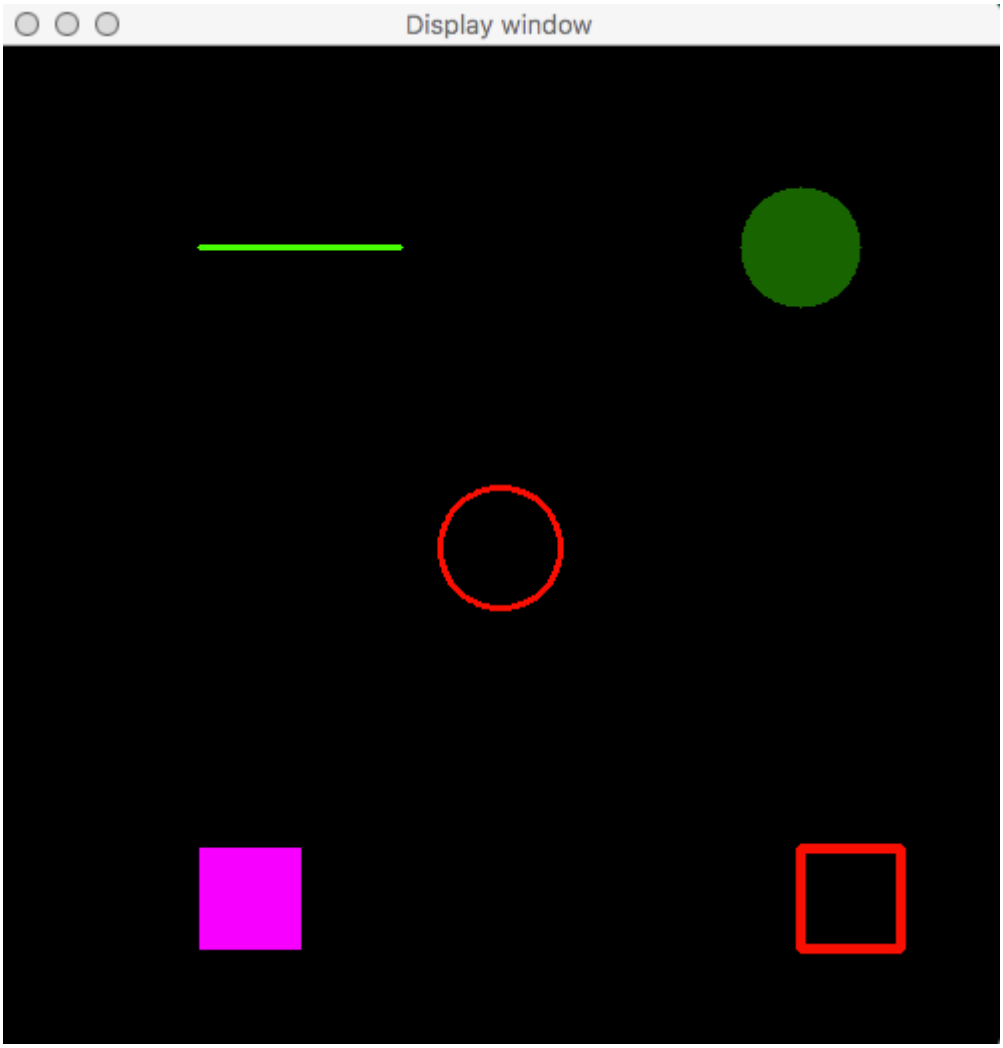
Examples

Muestra de formas de dibujo

```
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc.hpp> // drawing shapes
#include <iostream>


int main( int argc, char** argv )
{
    // First create a black image.
    cv::Mat image(500,500, CV_8UC3, cv::Scalar(0,0,0));

    // Check if the image is created successfully.
    if( !image.data ){
        std::cout <<  "Could not open or find the image" << std::endl ;
        exit(EXIT_FAILURE);
    }


    //####################(  Draw Line  )#########################
    cv::Point p1(100,100), p2(200,100);
    cv::Scalar colorLine(0,255,0); // Green
    int thicknessLine = 2;

    cv::line(image, p1, p2, colorLine, thicknessLine);

    //####################(  Draw Circle  )#######################
    // unfilled circle
    cv::Point centerCircle1(250,250);
    int radiusCircle = 30;
    cv::Scalar colorCircle1(0,0,255);
    int thicknessCircle1 = 2;

    cv::circle(image, centerCircle1, radiusCircle, colorCircle1, thicknessCircle1);

    // filled circle
    cv::Point centerCircle2(400,100);
    cv::Scalar colorCircle2(0,100,0);

    cv::circle(image, centerCircle2, radiusCircle, colorCircle2, CV_FILLED);
```

```
    //####################(  Draw Rectangle  )#######################
    // unfilled
    cv::Point p3(400,400), p4(450,450);
    cv::Scalar colorRectangle1(0,0,255);
    int thicknessRectangle1 = 3;

    cv::rectangle(image, p3, p4, colorRectangle1,thicknessRectangle1);

    //   filled
    cv::Point p5(100,400), p6(150,450);
    cv::Scalar colorRectangle2(255,0,255);

    cv::rectangle(image, p5, p6, colorRectangle2, CV_FILLED);


    //#################( Draw Shapes on Image )######################
    cv::namedWindow( "Display window", cv::WINDOW_AUTOSIZE );
    cv::imshow( "Display window", image );

    cv::waitKey(0);


    return 0;
}
```

La salida es



OpenCV 3.2 Mac con compilador g ++

```
g++ main2.cpp -o main `pkg-config --cflags --libs opencv`
```

Lea Dibujar formas (línea, círculo, ..., etc.) en C ++ en línea:
https://riptutorial.com/es/opencv/topic/9749/dibujar-formas--linea--circulo-------etc---en-c-plusplus

Capítulo 11: Estructuras basicas

Introducción

Este tema cubre estructuras básicas en OpenCV. Las estructuras que se tratarán en este tema son `DataType` , `Point` , `Vec` , `Size` , `Rect` , `Scalar` , `Ptr` y `Mat` .

Examples

Tipo de datos

Los tipos primitivos en OpenCV son `unsigned char, bool, signed char, unsigned short, signed short, int, float, double` . Cualquier tipo de datos en OpenCV se define como `CV_<bit-depth>{U|S|F}C(<number_of_channels>)` donde `U: unsigned` , `S:signed` y `F:floating point` .

Por ejemplo, `CV_32FC2` es una `CV_32FC2` 32 bits, punto flotante y 2 canales. y la definición de básicos, un tipo de canal son

```
#define CV_8U   0
#define CV_8S   1
#define CV_16U  2
#define CV_16S  3
#define CV_32S  4
#define CV_32F  5
#define CV_64F  6
#define CV_USRTYPE1 7
```

Los otros tipos con canal más alto se producen a partir de estos mediante la siguiente definición:

```
#define CV_MAKETYPE(depth,cn) (CV_MAT_DEPTH(depth) + (((cn)-1) << CV_CN_SHIFT))
```

Usando estos tipos de datos se pueden crear otras estructuras.

Estera

`Mat` (Matrix) es una matriz n-dimensional que se puede utilizar para almacenar diversos tipos de datos, tales como RGB, HSV o las imágenes en escala de grises, los vectores con valores reales o complejos, otras matrices etc.

A `Mat` contiene la siguiente información: `width` , `height` , `type` , `channels` , `data` , `flags` , `datastart` , `dataend` y así sucesivamente.

Tiene varios métodos, algunos de ellos son: `create` , `copyTo` , `convertTo` , `isContinious` etc.

Hay muchas formas de crear una variable Mat. Considere que quiero crear una matriz con 100 filas, 200 columnas, escriba CV_32FC3:

```
int R = 100, C = 200;
Mat m1; m1.create(R,C,CV_32FC3);//creates empty matrix
Mat m2(cv::Size(R, C), CV_32FC3); // creates a matrix with R rows, C columns with data type T
where R and C are integers,
Mat m3(R,C,CV_32FC3); // same as m2
```

Matriz de inicialización:

```
Mat m1 = Mat::zeros(R,C,CV_32FC3); // This initialized to zeros, you can use one, eye or
cv::randn etc.
Mat m2(R,C,CV_32FC3);
for (int i = 0; i < m2.rows; i++)
    for (int j = 0; j < m2.cols; j++)
        for (int k = 0; k < m2.channels(); k++)
            m2.at<Vec3f>(i,j)[k] = 0;
//Note that, because m2 is a float type and has 3 channels, we used Vec3f, for more info see
Vec

Mat m3(3, out, CV_32FC1, cv::Scalar(0));
```

Vec

`Vec` (Vector) es una clase de plantilla para valores numéricos. A diferencia de `c++ vector` s, generalmente almacena vectores cortos (solo algunos elementos).

La forma en que se define un `Vec` es la siguiente:

```
 typedef Vec<type, channels> Vec< channels>< one char for the type>;
```

donde type es uno de `uchar, short, int, float, double` y los caracteres para cada tipo son `b, s, i, f, d` , respectivamente.

Por ejemplo, `Vec3b` indica un vector de caracteres sin firmar de 3 canales. Cada índice en una imagen RGB está en este formato.

```
Mat rgb = imread('path/to/file', CV_LOAD_IMAGE_COLOR);
cout << rgb.at<Vec3b>(0,0); //The output is [r g b] values as ASCII character.
// To print integer values of RED value
cout << (int)rgb.at<Vec3b>(0,0)[0]; //The output will be an integer in [0, 255].
```

En la clase `Vec` se definen los siguientes operadores

```
v1 = v2 + v3
v1 = v2 - v3
v1 = v2 * scale
v1 = scale * v2
v1 = -v2
v1 += v2 and other augmenting operations
v1 == v2, v1 != v2
```

Para más información, ver el enlace.

Lea Estructuras basicas en línea: https://riptutorial.com/es/opencv/topic/9099/estructuras-basicas

Capítulo 12: Funciones de dibujo en Java

Examples

Dibujar el rectángulo en la imagen

```
public class DrawRectangle {

    public static void main(String[] args) {
    //Load native library
    System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
    //image container object
    Mat goruntuDizisi=new Mat();
    //Read image in file system
    goruntuDizisi=Imgcodecs.imread("C:\\image.jpg");
    //Draw rectangle
    //Parameters: mat object for drawing, point coordinates (x1,y1,x2,y2) and color BGR
    Imgproc.rectangle(goruntuDizisi, new Point(10,100), new Point(100,200),new
Scalar(76,255,0));
    Imgcodecs.imwrite("C:\\Yeni_kiz_kulesi.jpg", goruntuDizisi);
    System.out.println("Writed");
    }
}
```

Lea Funciones de dibujo en Java en línea: https://riptutorial.com/es/opencv/topic/6153/funciones-de-dibujo-en-java

Capítulo 13: Genere y compile opencv 3.1.0-dev para Python2 en Windows usando CMake y Visual Studio

Observaciones

La construcción y `opencv 3.1.0-dev` para obtener acceso para **módulos no libres** puede ser un dolor de cabeza para algunas personas, especialmente en la máquina con Windows. A diferencia de Ubuntu, la configuración de opencv para Windows toma algo de tiempo y requiere que se instalen un par de dependencias pf antes de compilar y compilar.

Los programas que debe descargar e instalar antes de continuar en cualquier paso son:

1. Python 2.7.x o Python 3.xx
2. CMake

Si va a descargar Python para Win32, también debe descargar CMake para Win32 incluso si está utilizando una máquina de 64 bits.

Se recomienda descargar los programas de 32 bits porque algunas bibliotecas de Python solo son compatibles con máquinas de 32 bits, así que para evitar problemas, solo instale todo en la versión de 32 bits.

3. Visual Studio Community 2013

4. Numpy para Python2.7 Win32

Después de instalar todas las dependencias anteriores, ***reinicie*** su PC y estará listo para continuar con el siguiente paso.

Paso 2:

Si no eres el tipo de persona que prefiere leer, puedes ver este tutorial . El tutorial lo lleva desde aquí hasta el final de esta documentación.

Necesitará obtener **opencv** y **opencv_contrib** de **github** . Puedes encontrar ambos en:

1. abierto
2. opencv_contrib

Cree un directorio llamado opencv-3.1.0 donde, en este director, creará otros dos directorios, uno para la *compilación* y otro para las *fuentes* . Pondrá los dos archivos zip descargados en el archivo de fuentes después de la extracción.

Por ejemplo, su directorio opencv-3.1.0 se encuentra en la unidad C, por lo que tendrá tres rutas:

1. `C:\opencv-3.1.0`
2. `C:\opencv-3.1.0\build`
3. `C:\opencv-3.1.0\sources`

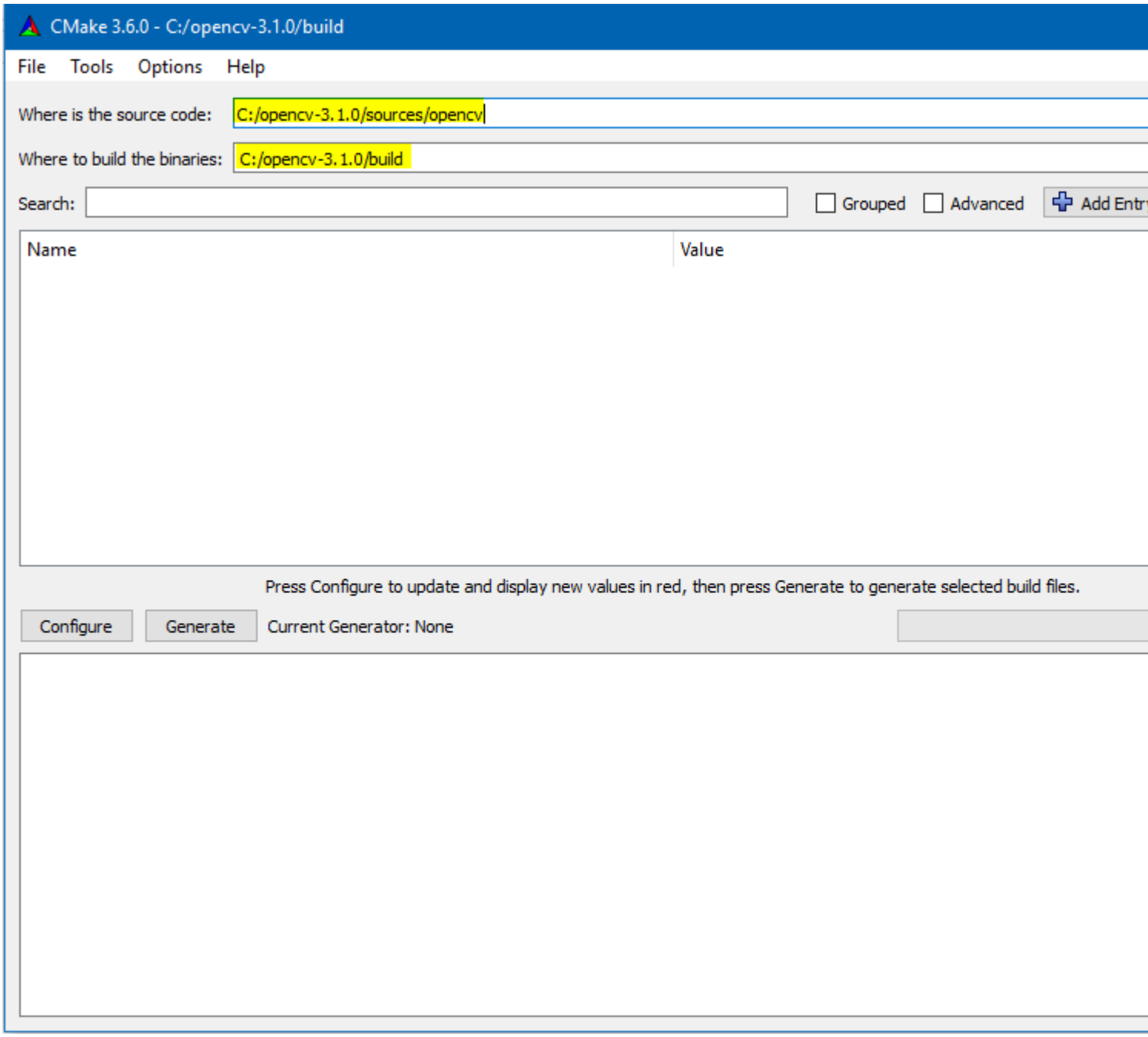
El tercer directorio incluirá dos rutas:

1. `C:\opencv-3.1.0\sources\opencv`
2. `C:\opencv-3.1.0\sources\opencv_contrib`

Ahora se hace con la preparación. Vamos a hacer algunas cosas útiles.

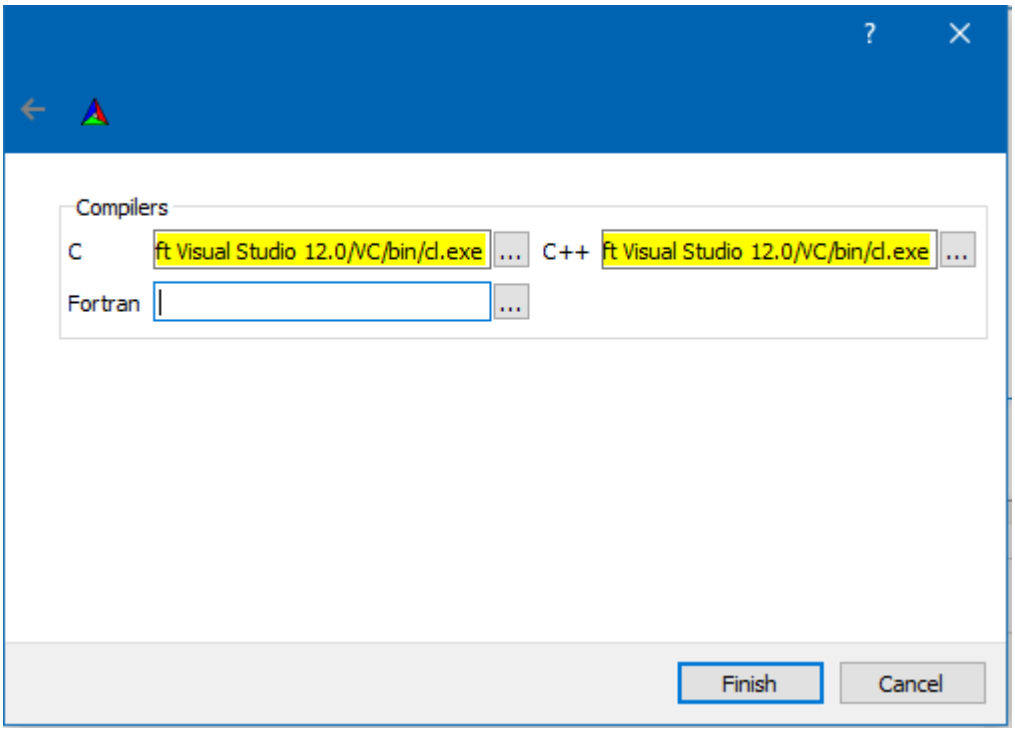
Paso 3:

Ejecutar CMake como administrador. Aparecerá una ventana como esta y tendrá que proporcionar dos directorios, uno para las fuentes y otro para el lugar donde se compilará el opencv. La imagen de abajo puede ayudarte más que las palabras.

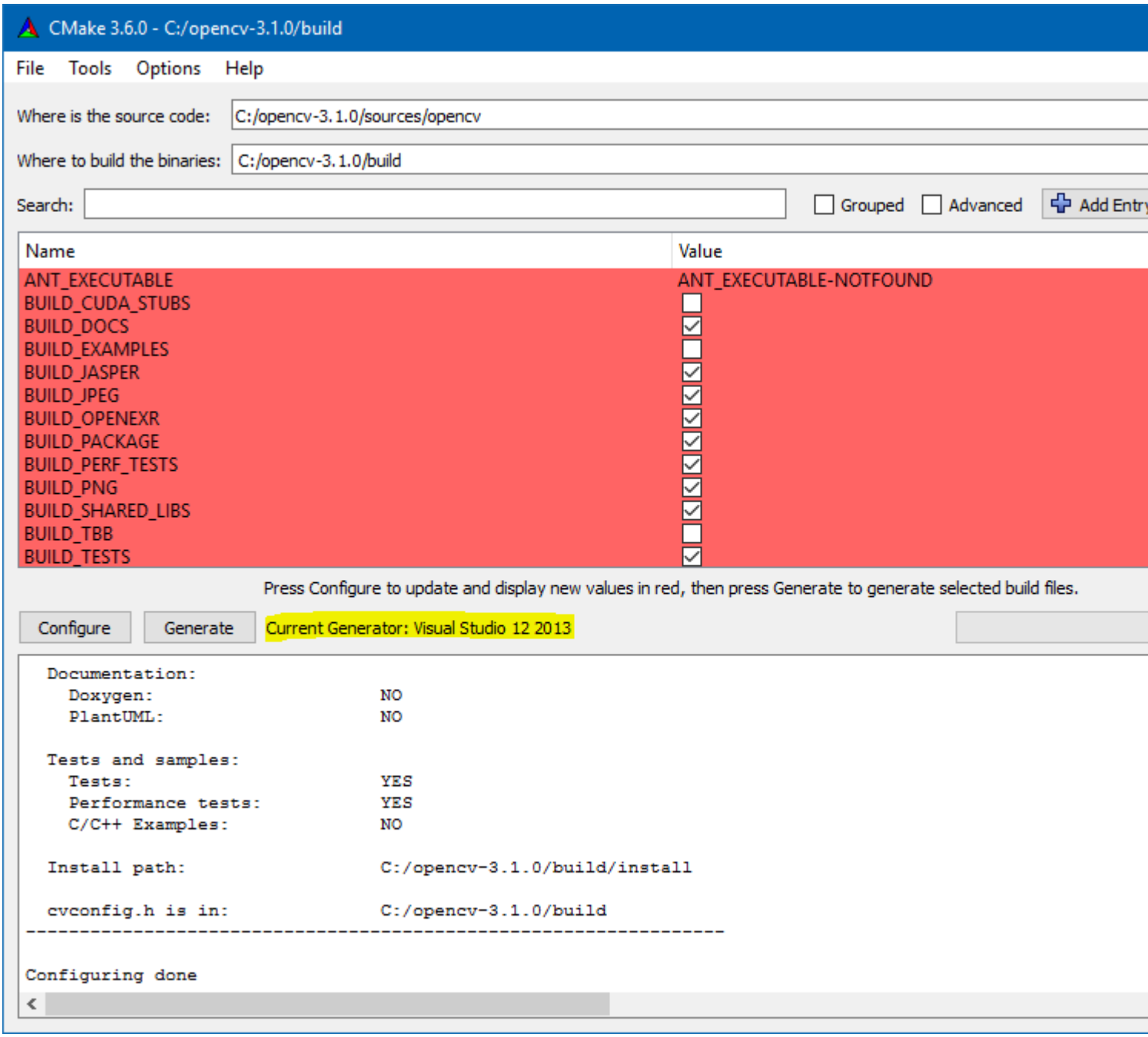


A continuación, haga clic en **Configurar** y será promovido para proporcionar los generadores; es decir, compiladores; para opencv. `cl.exe` proporcionar el `cl.exe` ubicado en Microsoft Visual Studio 2013. Haga clic en **especificar generadores nativos** y aparecerá una ventana emergente como la siguiente,

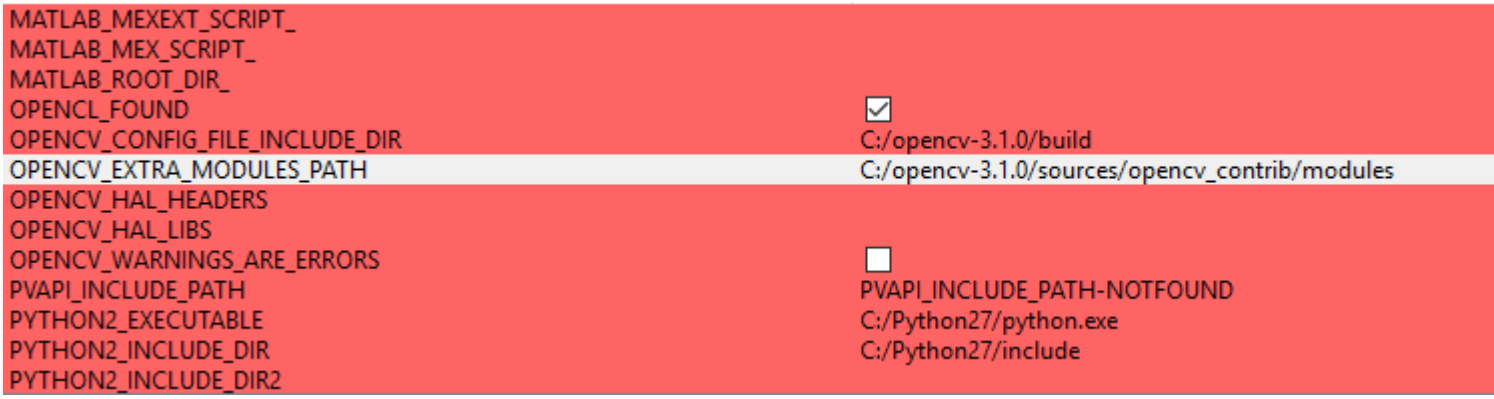
Las rutas serán algo así: `C:/Program Files (x86)/Microsoft Visual Studio 12.0/VC/bin/cl.exe` . Proporcione su ruta para los campos C y C ++. Haga clic en Finalizar y espere hasta que finalice la configuración. Debería obtener cero errores si estaba siguiendo todos los pasos anteriores correctamente.



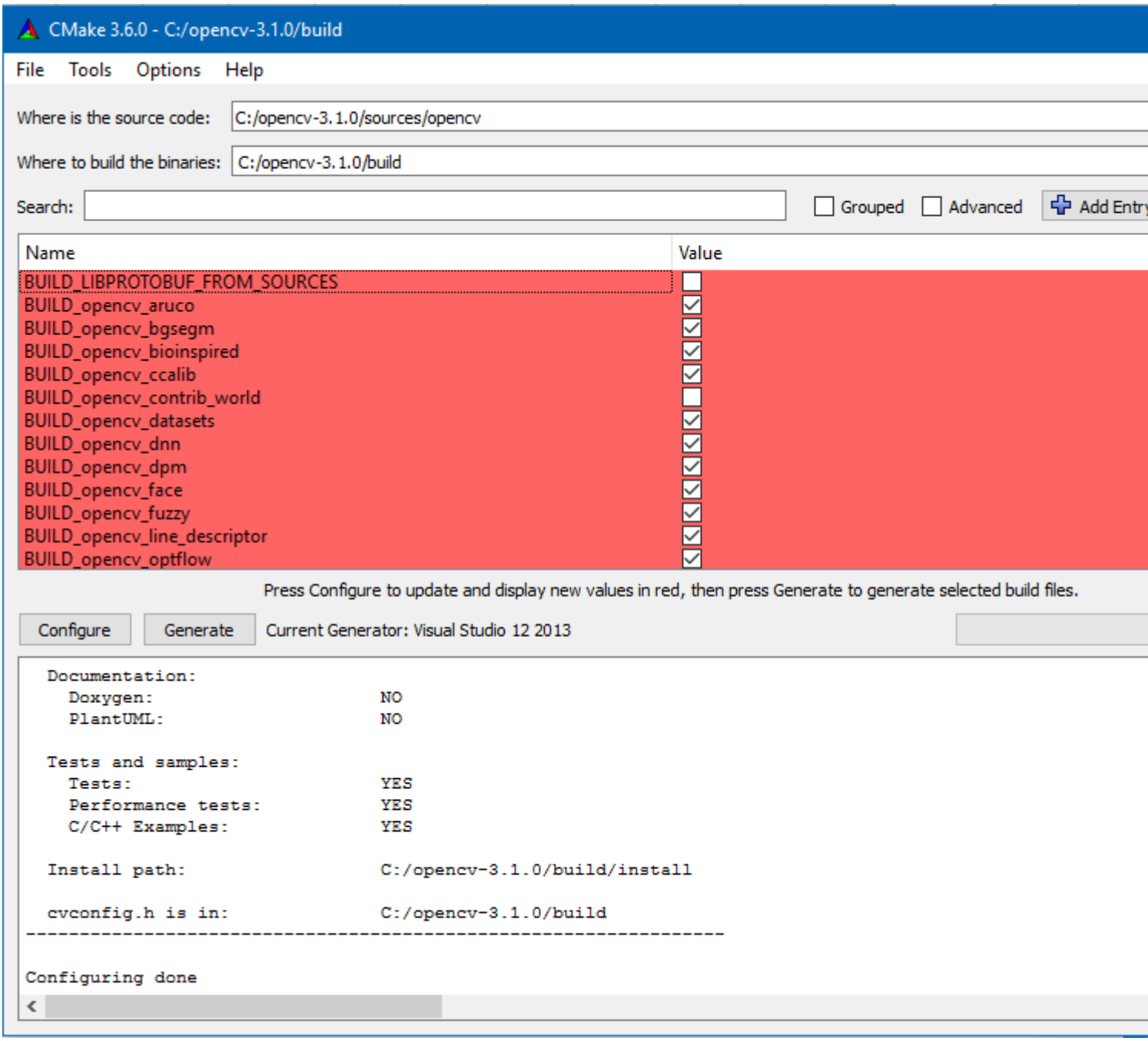
Una vez que CMake termina de configurar, verá que aparecen nuevos elementos en la ventana de CMake que están resaltados en rojo. Será algo como:



Compruebe las construcciones que necesita haciendo clic en la pequeña casilla cuadrada. Busque la línea `OPENCV_EXTRA_MODULES_PATH` y proporcione el directorio de módulos dentro de opencv_contrib dentro del directorio de fuentes.



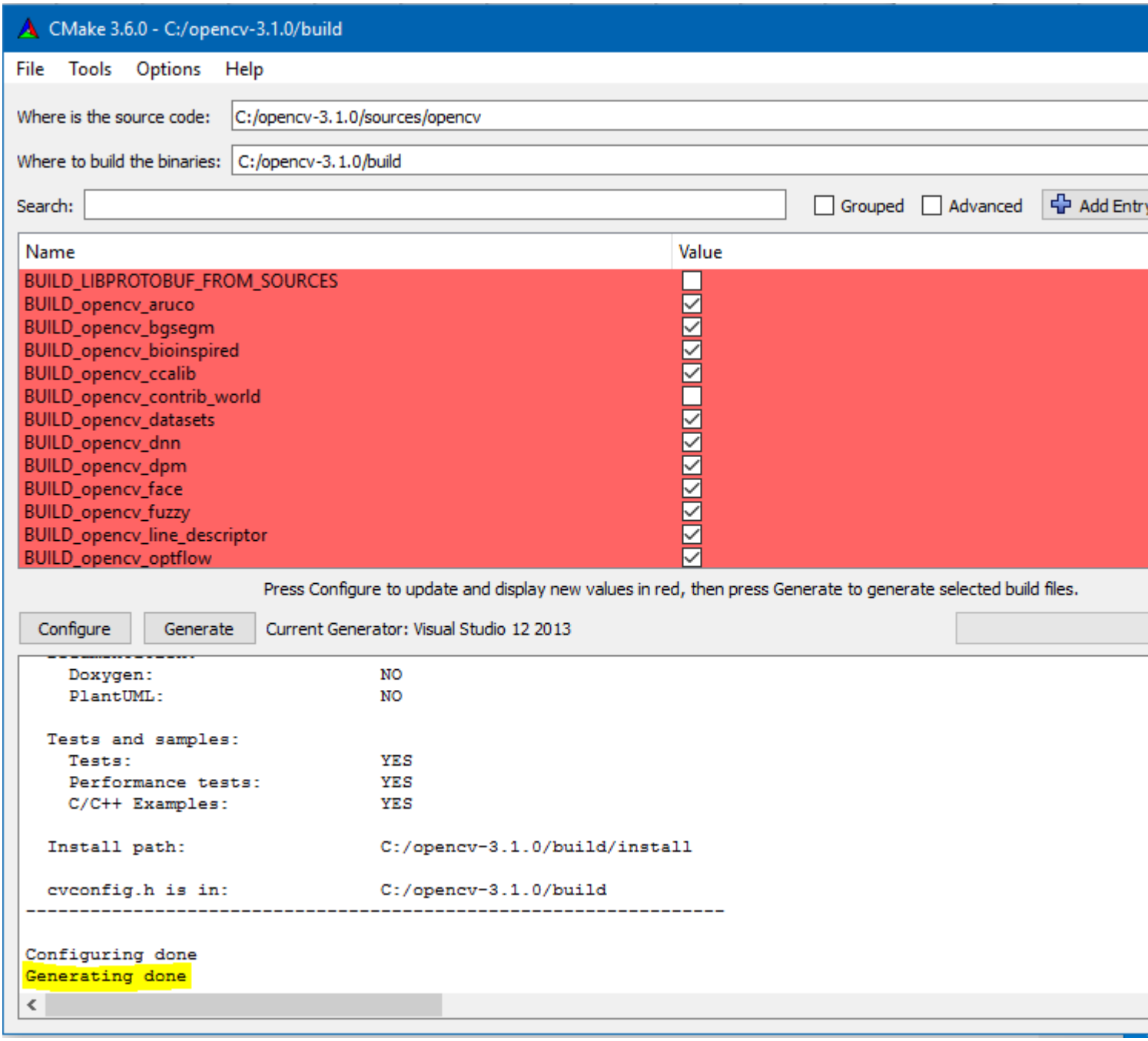
Una vez que haya terminado todo lo que necesita y provisto la ruta para los módulos adicionales, presione configurar nuevamente para actualizar. Las líneas resaltadas anteriormente ya no se destacarán y los campos nuevos se destacarán en rojo en su lugar.



También marque las casillas para cualquier cosa que necesite construir.

> Asegúrese de que ***BUILD_opencv_contrib_world*** y ***BUILD_opencv_world*** estén **sin *marcar*** . Probablemente hay un error donde se produce un error cuando se comprueba alguno de estos últimos.

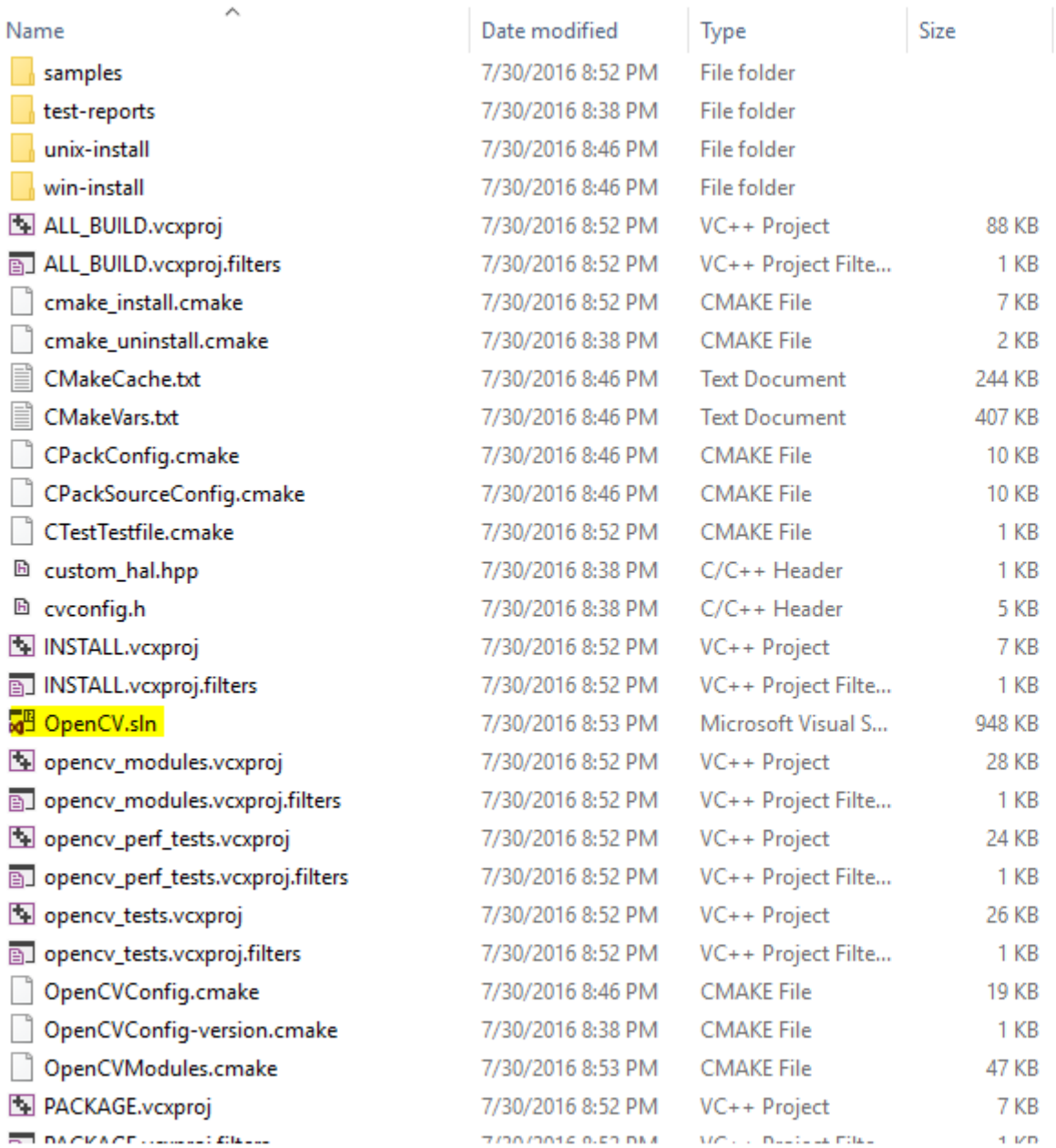
Al final de este paso, haga clic en **Generar** y terminará con CMake y podrá cerrarlo. *Si no hay errores, aparecerá un mensaje al final del panel inferior que dice "* ***Generación finalizada"*** *.*



Etapa 4:

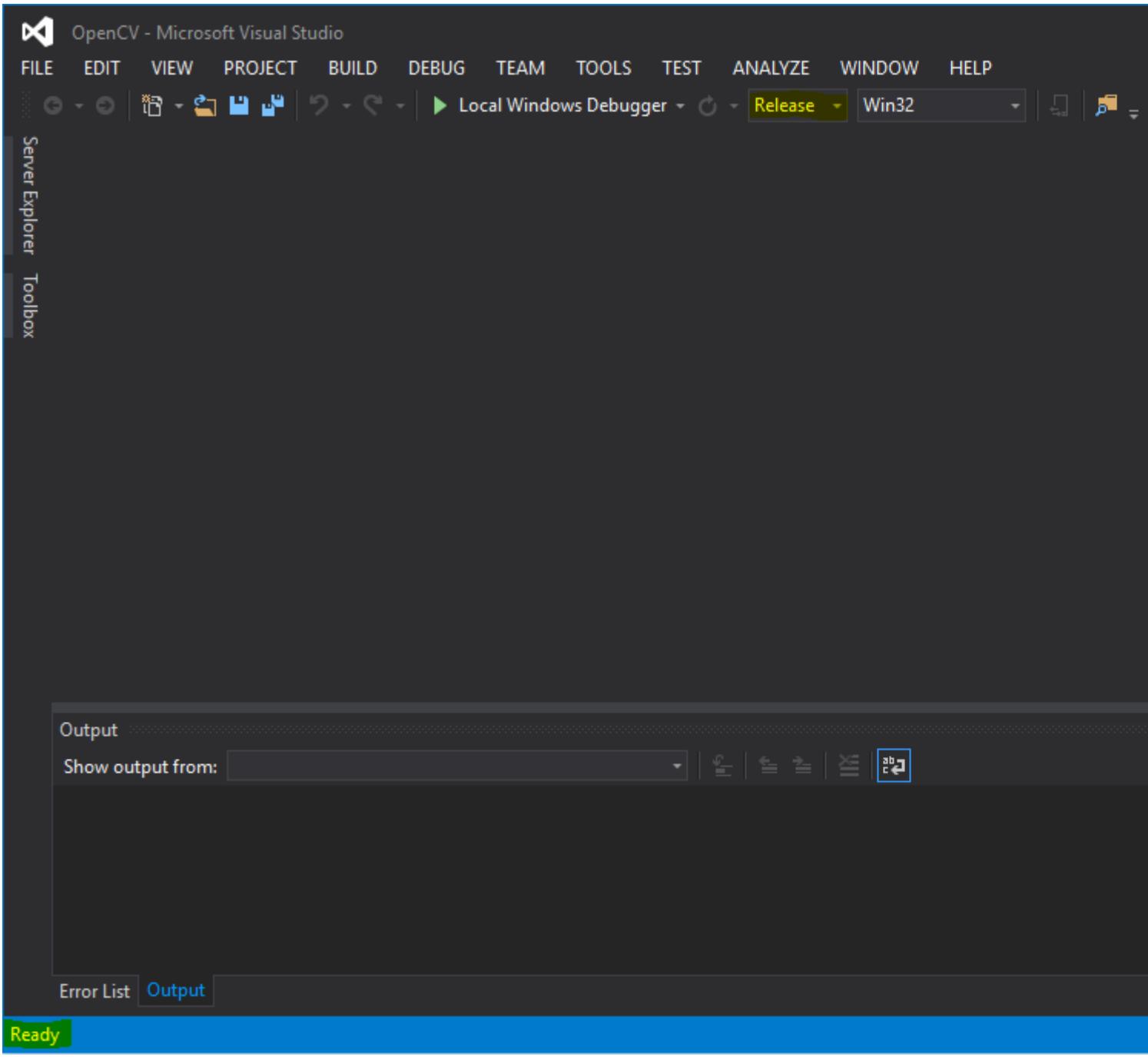
Abra el directorio de compilación ubicado en opencv-3.1.0 y encontrará un montón de nuevas carpetas y archivos dentro de él. Era una carpeta vacía al comienzo de este proceso.

Solo tratará con el archivo `opencv.sln` y no hará nada con los demás archivos. Abra este archivo con la versión que usó al compilar en el CMake en el paso anterior. Tiene que ser `Visual Microsoft 2013`.



Name	Date modified	Type	Size
samples	7/30/2016 8:52 PM	File folder	
test-reports	7/30/2016 8:38 PM	File folder	
unix-install	7/30/2016 8:46 PM	File folder	
win-install	7/30/2016 8:46 PM	File folder	
ALL_BUILD.vcxproj	7/30/2016 8:52 PM	VC++ Project	88 KB
ALL_BUILD.vcxproj.filters	7/30/2016 8:52 PM	VC++ Project Filte...	1 KB
cmake_install.cmake	7/30/2016 8:52 PM	CMAKE File	7 KB
cmake_uninstall.cmake	7/30/2016 8:38 PM	CMAKE File	2 KB
CMakeCache.txt	7/30/2016 8:46 PM	Text Document	244 KB
CMakeVars.txt	7/30/2016 8:46 PM	Text Document	407 KB
CPackConfig.cmake	7/30/2016 8:46 PM	CMAKE File	10 KB
CPackSourceConfig.cmake	7/30/2016 8:46 PM	CMAKE File	10 KB
CTestTestfile.cmake	7/30/2016 8:52 PM	CMAKE File	1 KB
custom_hal.hpp	7/30/2016 8:38 PM	C/C++ Header	1 KB
cvconfig.h	7/30/2016 8:38 PM	C/C++ Header	5 KB
INSTALL.vcxproj	7/30/2016 8:52 PM	VC++ Project	7 KB
INSTALL.vcxproj.filters	7/30/2016 8:52 PM	VC++ Project Filte...	1 KB
OpenCV.sln	7/30/2016 8:53 PM	Microsoft Visual S...	948 KB
opencv_modules.vcxproj	7/30/2016 8:52 PM	VC++ Project	28 KB
opencv_modules.vcxproj.filters	7/30/2016 8:52 PM	VC++ Project Filte...	1 KB
opencv_perf_tests.vcxproj	7/30/2016 8:52 PM	VC++ Project	24 KB
opencv_perf_tests.vcxproj.filters	7/30/2016 8:52 PM	VC++ Project Filte...	1 KB
opencv_tests.vcxproj	7/30/2016 8:52 PM	VC++ Project	26 KB
opencv_tests.vcxproj.filters	7/30/2016 8:52 PM	VC++ Project Filte...	1 KB
OpenCVConfig.cmake	7/30/2016 8:46 PM	CMAKE File	19 KB
OpenCVConfig-version.cmake	7/30/2016 8:38 PM	CMAKE File	1 KB
OpenCVModules.cmake	7/30/2016 8:53 PM	CMAKE File	47 KB
PACKAGE.vcxproj	7/30/2016 8:52 PM	VC++ Project	7 KB

Cuando abra el archivo .sln, tenga paciencia, ya que toma algo de tiempo preparar todo para la construcción. Cuando **Listo** es estable (no cambia), puedes comenzar a construir tus objetivos. Comienza a construir como se indica en la imagen de abajo. También asegúrese de que la `Solution Configuration` la `Solution Configuration` sea `Release` no `Debug` .



Paso 5:

Cuando finalice la construcción, deberá copiar y pegar un par de archivos del directorio de construcción en el directorio de `Python27` .

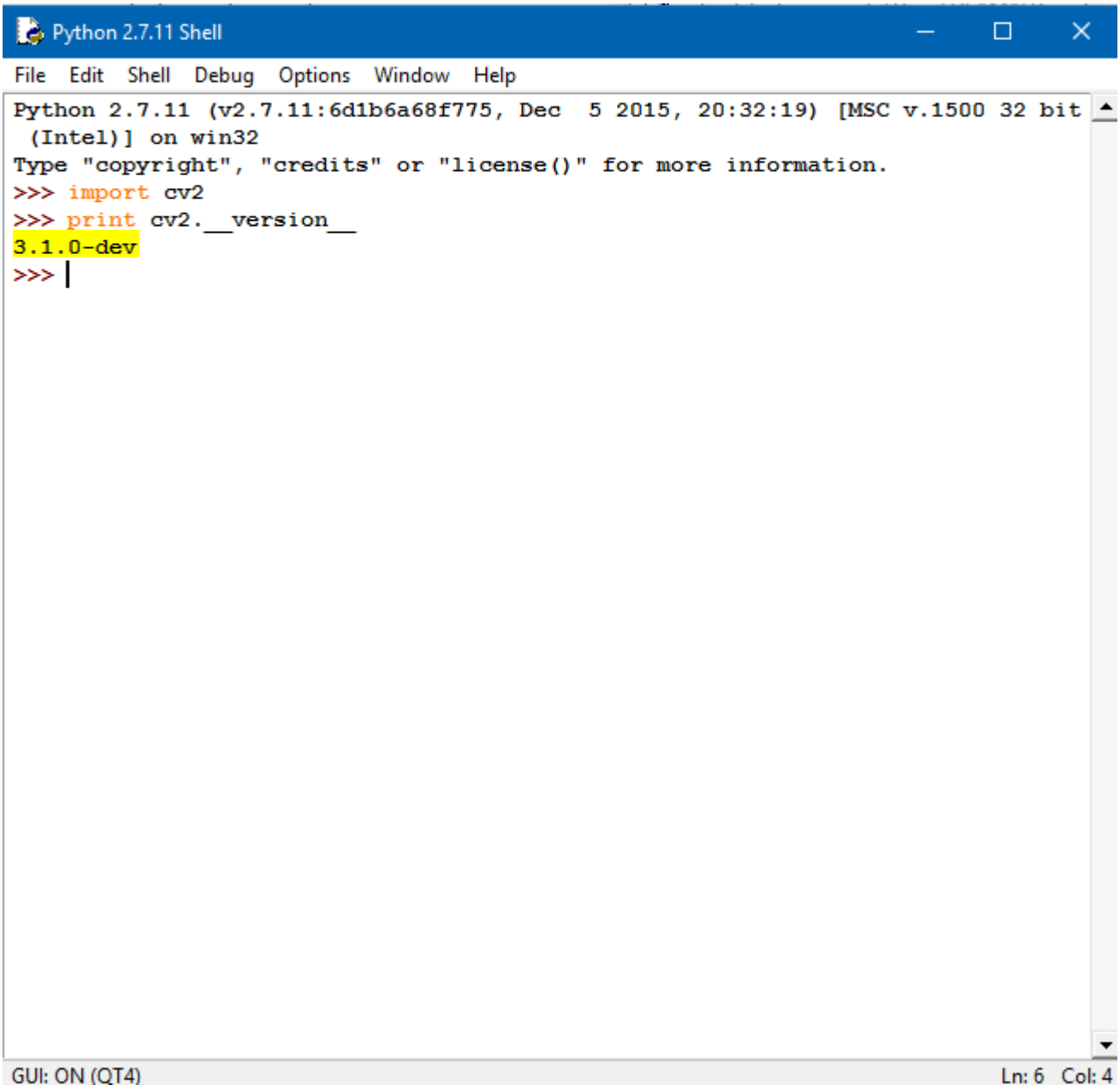
Busque el archivo `cv2.pyd` y cópielo en el directorio de `site-packages` en `Python27` . El `cv2.pyd` debe estar presente en `C:\opencv-3.1.0\build\lib\Release` . Después de eso, copie **solo** los archivos **.dll** dentro de `C:\opencv-3.1.0\build\bin\Release` en el directorio principal de `Python27` en esta ubicación `C:\Python27` .

Al final de este paso, reinicie su PC.

Verificación:

Abre IDLE y dentro del tipo de shell de Python:

```
>>> import cv2
>>> print cv2.__version__
3.1.0-dev
```



Examples

Lectura de imagen y conversión a escala de grises

```
import cv2
import numpy as np

img = cv2.imread('<your_image>')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

cv2.imshow('image', img)
```

```
cv2.imshow('gray', gray)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Lea Genere y compile opencv 3.1.0-dev para Python2 en Windows usando CMake y Visual Studio en línea: https://riptutorial.com/es/opencv/topic/6100/genere-y-compile-opencv-3-1-0-dev-para-python2-en-windows-usando-cmake-y-visual-studio

Capítulo 14: Inicialización de OpenCV en Android

Examples

Inicialización asíncrona

El uso de la inicialización asíncrona es una forma recomendada para el desarrollo de aplicaciones. Utiliza el Administrador de OpenCV para acceder a las bibliotecas de OpenCV instaladas externamente en el sistema de destino.

Fragmento de código implementando la inicialización asíncrona:

```
public class MainActivity extends Activity implements CvCameraViewListener2 {

    private BaseLoaderCallback mLoaderCallback = new BaseLoaderCallback(this) {
        @Override
        public void onManagerConnected(int status) {
            switch(status) {
                case LoaderCallbackInterface.SUCCESS:
                    Log.i(TAG,"OpenCV Manager Connected");
                    //from now onwards, you can use OpenCV API
                    Mat m = new Mat(5, 10, CvType.CV_8UC1, new Scalar(0));
                    break;
                case LoaderCallbackInterface.INIT_FAILED:
                    Log.i(TAG,"Init Failed");
                    break;
                case LoaderCallbackInterface.INSTALL_CANCELED:
                    Log.i(TAG,"Install Cancelled");
                    break;
                case LoaderCallbackInterface.INCOMPATIBLE_MANAGER_VERSION:
                    Log.i(TAG,"Incompatible Version");
                    break;
                case LoaderCallbackInterface.MARKET_ERROR:
                    Log.i(TAG,"Market Error");
                    break;
                default:
                    Log.i(TAG,"OpenCV Manager Install");
                    super.onManagerConnected(status);
                    break;
            }
        }
    };

    @Override
    public void onResume() {
        super.onResume();
        OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_3_1_0, this, mLoaderCallback);
    }

    ...
}
```

En este caso, nuestra aplicación funciona con OpenCV Manager de forma asíncrona. `OnManagerConnected` devolución de llamada `OnManagerConnected` se llamará en el hilo de la interfaz de usuario, cuando finalice la inicialización.

> Tenga en cuenta que no está permitido usar llamadas OpenCV o cargar bibliotecas nativas dependientes de OpenCV antes de invocar esta devolución de llamada. Cargue sus propias bibliotecas nativas que dependen de OpenCV después de la inicialización exitosa de OpenCV.

La implementación predeterminada de `BaseLoaderCallback` trata el contexto de la aplicación como `Activity` y llama al método `Activity.finish()` para salir en caso de un error de inicialización. Para anular este comportamiento, debe anular el método `finish()` de la clase `BaseLoaderCallback` e implementar su propio método de finalización.

Administrador de OpenCV

OpenCV Manager es un servicio de Android destinado a administrar binarios de bibliotecas de OpenCV en dispositivos de usuarios finales. Permite compartir las bibliotecas dinámicas de OpenCV entre aplicaciones en el mismo dispositivo.

El Administrador proporciona los siguientes beneficios:

- Menos uso de memoria (alrededor de 40MB). Todas las aplicaciones utilizan los mismos binarios del servicio y no mantienen las librerías nativas dentro de ellas.
- Optimizaciones específicas de hardware para todas las plataformas soportadas.
- Fuente de confianza de la biblioteca OpenCV. Todos los paquetes con OpenCV se publican en el mercado de Google Play.
- Actualizaciones regulares y correcciones de errores.

La única desventaja es que se le pide al usuario que descargue una aplicación adicional, por lo que la experiencia del usuario disminuye ligeramente.

Más información: Android OpenCV Manager

> **Actualizado 18/10/16:**
> Hay un error en la versión de OpenCV Manager distribuida en Play Store (actualizado el 21/09/15).
> Sólo afecta a la versión OpenCV 3.1.0. Cuando ejecutas algunas funciones OpenCV, obtienes un error `SIGSEGV` . La versión distribuida con Android SDK funciona bien (`OpenCV-android-sdk/apk/OpenCV_3.1.0_Manager_3.10_{platform}.apk`). Se puede descargar desde la web de OpenCV .
> Más información: Edición # 6247 .

Inicialización Estática

De acuerdo con este enfoque, todos los binarios de OpenCV están incluidos en su paquete de aplicación. Está diseñado principalmente para propósitos de desarrollo y depuración. Este enfoque está en **desuso** para el código de producción, se recomienda la inicialización asíncrona.

Si el proyecto de su aplicación no tiene una parte JNI, simplemente copie las bibliotecas nativas de OpenCV correspondientes desde `OpenCV-3.1.0-android-sdk/sdk/native/libs` a su directorio de proyecto en la carpeta `app/src/main/jniLibs` .

En el caso del proyecto de aplicación con una parte JNI, en lugar de copiar las bibliotecas manuales, necesita modificar su archivo `Android.mk` : agregue las siguientes dos líneas de código después de `"include $(CLEAR_VARS)"` y antes de `"include path_to_OpenCV-3.1.0-android-sdk/sdk/native/jni/OpenCV.mk"` :

```
OPENCV_CAMERA_MODULES:=on
OPENCV_INSTALL_MODULES:=on
```

El resultado debe ser similar al siguiente:

```
include $(CLEAR_VARS)
# OpenCV
OPENCV_CAMERA_MODULES:=on
OPENCV_INSTALL_MODULES:=on
include ../../sdk/native/jni/OpenCV.mk
```

Después de eso, las bibliotecas de OpenCV se copiarán a la carpeta `jniLibs` su aplicación durante la compilación JNI.

El último paso para habilitar OpenCV en su aplicación es el código de inicialización de Java antes de llamar a la API de OpenCV. Se puede hacer, por ejemplo, en la sección estática de la clase de actividad:

```
static {
    if (!OpenCVLoader.initDebug()) {
        // Handle initialization error
    }
}
```

Si su aplicación incluye otras bibliotecas nativas dependientes de OpenCV, debe cargarlas después de la inicialización de OpenCV:

```
static {
    if (!OpenCVLoader.initDebug()) {
        // Handle initialization error
    } else {
        System.loadLibrary("my_jni_lib1");
        System.loadLibrary("my_jni_lib2");
    }
}
```

> **Nota: el** método `initDebug()` está en desuso para el código de producción. Está diseñado solo para fines de desarrollo experimental y local. Si desea publicar su aplicación, utilice el enfoque con inicialización asíncrona.

Lea Inicialización de OpenCV en Android en línea:
https://riptutorial.com/es/opencv/topic/7545/inicializacion-de-opencv-en-android

Capítulo 15: Instalación de OpenCV

Introducción

Instalación de OpenCV en Linux, Mac OS y Windows

Examples

Instalación de OpenCV en Ubuntu

Enlace de fuente

Abre la Terminal y escribe los siguientes comandos.

1-Actualizar y actualizar el paquete de su sistema ubuntu:

```
sudo su

sudo apt-get -y update

sudo apt-get -y upgrade

sudo apt-get -y dist-upgrade

sudo apt-get -y autoremove
```

2-Instalación de dependencias:

```
sudo apt-get install libopencv-dev
```

3-Build Tools para OpenCV Código fuente:

```
sudo apt-get install build-essential checkinstall cmake pkg-config
```

Bibliotecas de E / S de 4 imágenes para OpenCV:

```
sudo apt-get install libtiff5-dev libjpeg-dev libjasper-dev libpng12-dev zlib1g-dev
libopenexr-dev libgdal-dev
```

Bibliotecas de E / S de 5 vídeos para OpenCV:

```
sudo apt-get install libavcodec-dev libavformat-dev libmp3lame-dev libswscale-dev
libdc1394-22-dev libxine2-dev libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev libv4l-
dev v4l-utils libfaac-dev libopencore-amrnb-dev libopencore-amrwb-dev libtheora-dev libvorbis-
dev libxvidcore-dev libx264-dev x264 yasm
```

6-Librerías de paralelismo y álgebra lineal:

```
sudo apt-get install libtbb-dev libeigen3-dev
```

Bibliotecas de interfaz de usuario de 7 gráficos:

```
sudo apt-get install libqt4-dev libgtk2.0-dev qt5-default

sudo apt-get install libvtk6-dev
```

8 – Instalación de Java:

```
sudo apt-get install ant default-jdk
```

Instalación de 9-Python:

```
sudo apt-get install python-dev python-tk python-numpy python3-dev python3-tk python3-numpy
python-matplotlib

sudo apt-get install python-opencv

sudo apt-get install doxygen
```

10-Descargar el código fuente de OPENCV desde Github:

```
wget https://github.com/opencv/opencv/archive/3.2.0.zip
```

11-Descomprimir archivo Zip OPENCV:

```
unzip 3.2.0.zip
```

12-Eliminar el archivo Zip OPENCV:

```
rm 3.2.0.zip
```

13-Build OPENCV:

```
mv opencv-3.2.0 opencv

cd opencv

mkdir build

cd build

cmake -D CMAKE_BUILD_TYPE=RELEASE -D CMAKE_INSTALL_PREFIX=/usr/local -D WITH_TBB=ON -D
BUILD_NEW_PYTHON_SUPPORT=ON -D WITH_V4L=ON -D INSTALL_C_EXAMPLES=ON -D
INSTALL_PYTHON_EXAMPLES=ON -D BUILD_DOC=ON -D BULD_EXAMPLES=ON -D WITH_QT=ON -D WITH_OPENGL=ON
-D WITH_EIGEN=ON -D FORCE_VTK=TRUE -D WITH_VTK=ON ..

make -j4
```

sudo make install

```
sudo sh -c 'echo "/usr/local/lib" > /etc/ld.so.conf.d/opencv.conf'

sudo ldconfig
```

14-Terminado Verifique su número de versión de OpenCV:

```
pkg-config --modversion opencv

pkg-config --cflags opencv
```

Lea Instalación de OpenCV en línea: https://riptutorial.com/es/opencv/topic/8934/instalacion-de-opencv

Capítulo 16: Modificación de contenido de imagen

Examples

Establecer imagen completa a un color sólido

Dada una `cv::Mat img` no vacía de algún tamaño, podemos rellenarla de un color sólido de varias maneras:

```
img = cv::Scalar(blueVal,greenVal,redVal);
```

o, cuanto más general, soporte de máscara, `cv::Mat::setTo()` :

```
img.setTo(cv::Scalar(blueVal,greenVal,redVal));
```

Si está utilizando la API más antigua de OpenCV C con `IplImage* img` :

Utilizar:

```
cvSet(img, CV_RGB(redVal,greenVal,blueVal));
```

Modificación píxel a píxel de las imágenes.

En OpenCV, las imágenes pueden ser RGB / BGR, HSV, escala de grises, blanco y negro, etc. Es crucial conocer el tipo de datos antes de tratar con imágenes.

Los tipos de datos de imagen son principalmente `CV_8UC3` (Matriz de uchar con 3 canales) y CV_8U (Matriz de uchar con 1 canal), sin embargo, la conversión a otros tipos como CV_32FC3, CV_64F también es posible. (ver tipos de datos)

Considere, la imagen es una imagen RGB que se lee con la función `imread` .

```
Mat rgb = imread('path/to/rgb/image', CV_LOAD_IMAGE_COLOR);
//to set RED pixel value of (i,j)th to X,
rgb.at<Vec3b>(i,j)[0] = X;
```

Del mismo modo, si la imagen está en escala de grises,

```
gray.at<uchar>(i,j) = X;
```

Tenga en cuenta que, en OpenCV, las imágenes en blanco y negro se almacenan como tipo CV_8U con los valores 0 y 255. Por lo tanto, el cambio de imágenes BW es igual que en las imágenes grises.

Modificación del color de la imagen en OpenCV - kmeans (). Para escanear todos los píxeles de una imagen y reemplazar los valores de píxeles con colores genéricos.

```
#include opencv2/opencv.hpp> #include vector> using namespace std; using namespace cv; int main() { Mat3b img = imread("test.jpg"); z }
```

```
imshow("Original", img);

// Cluster

int K = 8;
int n = img.rows * img.cols;
Mat data = img.reshape(1, n);
data.convertTo(data, CV_32F);

Mat labels;
Mat1f colors;
kmeans(data, K, labels, cv::TermCriteria(), 1, cv::KMEANS_PP_CENTERS, colors);

for (int i = 0; i < n; ++i)
{
    data.at<float>(i, 0) = colors(labels.at<int>(i), 0);
    data.at<float>(i, 1) = colors(labels.at<int>(i), 1);
    data.at<float>(i, 2) = colors(labels.at<int>(i), 2);
}

Mat reduced = data.reshape(3, img.rows);
reduced.convertTo(reduced, CV_8U);



imshow("Reduced", reduced);
waitKey(0);

return 0;
```

#include opencv2/opencv.hpp> #include vector> using namespace std; using namespace cv; int main() { Mat3b img = imread("test.jpg"); z }

Lea Modificación de contenido de imagen en línea:
https://riptutorial.com/es/opencv/topic/6307/modificacion-de-contenido-de-imagen

Capítulo 17: Mostrar imagen OpenCV

Examples

Lectura básica y visualización de una imagen.

```
import cv2

image_path= #put your image path here

#use imread() function to read image data to variable img.
img = cv2.imread(image_path)

#display image data in a new window with title 'I am an image display window'
cv2.imshow('I am an image display window',img)

#wait until user hits any key on keyboard
cv2.waitKey(0)

#close any windows opened by opencv
cv2.destroyAllWindows()
```

Para controlar el tamaño de la ventana de visualización en la pantalla, agregue los siguientes comandos antes del comando cv2.imshow:

```
window_width=800 #size of the display window on the screen
window_height=600

#open an empty window with a title.
#The flag cv2.WINDOW_NORMAL allows the window to be scaleable.
cv2.namedWindow('I am an image display window', cv2.WINDOW_NORMAL)

#scale the image display window to desired size
cv2.resizeWindow('I am an image display window', window_width, window_height)
```

ver documentos de openCV para más detalles

Leyendo MJPEG desde camara IP

```
import cv2
import numpy as np
import urllib

stream=urllib.urlopen('http://96.10.1.168/mjpg/video.mjpg')
bytes=''
while True:
    bytes+=stream.read(1024)
    a = bytes.find('\xff\xd8') # JPEG start
    b = bytes.find('\xff\xd9') # JPEG end
    if a!=-1 and b!=-1:
        jpg = bytes[a:b+2] # actual image
        bytes= bytes[b+2:] # other informations
```

```
        # decode to colored image ( another option is cv2.IMREAD_GRAYSCALE )
        img = cv2.imdecode(np.fromstring(jpg, dtype=np.uint8),cv2.IMREAD_COLOR)
        cv2.imshow('Window name',img) # display image while receiving data
        if cv2.waitKey(1) ==27: # if user hit esc
            exit(0) # exit program
```

Cada `JPEG` comienza con `0xff 0xd8` y termina con `0xff 0xd9` . Entre esas hay imagen real. Información detallada en esta respuesta SO

Display Image OpenCV Java

Imagen de lectura básica de java.

```
import org.opencv.core.Core;
import org.opencv.core.Mat;
import org.opencv.imgcodecs.Imgcodecs;

//Load native library
System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
//Mat object used to host the image
Mat imageArray;
//Read image file from vile system
imageArray=Imgcodecs.imread("path/to/image");
```

Si desea ver imágenes, no puede usar imshow porque OpenCV-java tampoco tiene este método. En su lugar, puede escribir el siguiente método.

```
private static BufferedImage ConvertMat2Image(Mat imgContainer{
    MatOfByte byteMatData = new MatOfByte();
    //image formatting
    Imgcodecs.imencode(".jpg", imgContainer,byteMatData);
    // Convert to array
    byte[] byteArray = byteMatData.toArray();
    BufferedImage img= null;
    try {
        InputStream in = new ByteArrayInputStream(byteArray);
        //load image
        img= = ImageIO.read(in);
    } catch (Exception e) {
        e.printStackTrace();
        return null;
    }
    return img;
}
```

Puede ver el objeto de resultado en Jframe, Jlabel (icono de jlabel), etc.

Lea Mostrar imagen OpenCV en línea: https://riptutorial.com/es/opencv/topic/3306/mostrar-imagen-opencv

Capítulo 18: Procesamiento de imágenes

Sintaxis

1. **Sintaxis de Blur Gaussian C ++:** void GaussianBlur (InputArray src, OutputArray dst, Tamaño ksize, doble sigmaX, double sigmaY = 0, int borderType = BORDER_DEFAULT)

Parámetros

Parámetros del Desenfoque Gaussiano	Detalles
src	Imagen de entrada, la imagen puede tener cualquier número de canales, que se procesan de forma independiente, pero la profundidad debe ser `CV_8U` , `CV_16U` , `CV_16S` , `CV_32F` o `CV_64F` .
dst	Imagen de salida del mismo tamaño y tipo que `src`
ksize	Tamaño del núcleo gaussiano. `ksize.width` y `ksize.height` pueden diferir, pero ambos deben ser **positivos e impares** . O bien, pueden ser cero y luego se calculan a partir de sigma *.
sigmaX	Desviación estándar del núcleo gaussiano en la **dirección X.**
sigmaY	Desviación estándar del núcleo gaussiano en **dirección Y.** si `sigmaY` es cero, se establece como igual a `sigmaX` , si ambos sigmas son ceros, se calculan desde `ksize.width` y `ksize.height` . Para controlar completamente el resultado, independientemente de las posibles modificaciones futuras de toda esta semántica, se recomienda especificar todo el `ksize` de `ksize` , `sigmaX` y `sigmaY` .
borderType	Método de extrapolación de píxeles.

Observaciones

No creo que tenga sentido poner sintaxis y parámetros específicos para el desenfoque gaussiano en este lugar, ya que el tema es tan amplio que debería incluir muchos otros ejemplos.

Examples

Suavizar imágenes con desenfoque gaussiano en C ++

El suavizado, también conocido como **desenfoque** , es una de las operaciones más utilizadas en

el procesamiento de imágenes.

El uso más común de la operación de suavizado es **reducir el ruido** en la imagen para su posterior procesamiento.

Hay muchos algoritmos para realizar operaciones de suavizado.

Veremos uno de los filtros más utilizados para desenfocar una imagen, el **Filtro Gaussiano que** utiliza la función de biblioteca OpenCV `GaussianBlur()` . Este filtro está diseñado específicamente para eliminar *el ruido* de *alta frecuencia* de las imágenes.

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

using namespace std;
using namespace cv;

int main(int argc, char** argv){

    Mat image , blurredImage;

    // Load the image file
    image = imread(argv[1], CV_LOAD_IMAGE_COLOR);

    // Report error if image could not be loaded
    if(!image.data){
        cout<<"Error loading image" << "\n";
        return -1;
    }

    // Apply the Gaussian Blur filter.
    // The Size object determines the size of the filter (the "range" of the blur)
    GaussianBlur( image, blurredImage, Size( 9, 9 ), 1.0);

    // Show the blurred image in a named window
    imshow("Blurred Image" , blurredImage);

    // Wait indefinitely untill the user presses a key
    waitKey(0);

    return 0;
}
```

Para la definición matemática detallada y otros tipos de filtros, puede consultar la documentación original .

Umbral

En Python:

```
import cv2
image_path= 'd:/contour.png'
img = cv2.imread(image_path)

#display image before thresholding
cv2.imshow('I am an image display window',img)
cv2.waitKey(0)

#convert image to gray scale - needed for thresholding
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)


#apply threshold to gray image to obtain binary image

threshold=150 #value above which pixel values will be set to max_value
max_value=255  #value to which pixels above threshold will be set
threshold_stype=cv2.THRESH_BINARY #default threshold method

ret, img_binary = cv2.threshold(img_gray, threshold, max_value, threshold_stype)

#display image after thresholding
cv2.imshow('image after applying threshold',img_binary)
cv2.waitKey(0)

#save the binary image
cv2.imwrite('d:/binary.png',img_binary)
cv2.destroyAllWindows()
```

Filtrado bilateral

En las aplicaciones de procesamiento de imágenes, los filtros bilaterales son un tipo especial de **filtros no lineales** .

Existe un intercambio entre la pérdida de estructura y la eliminación de ruido, porque el método más popular para eliminar el ruido es el desenfoque gaussiano, que no es consciente de la estructura de la imagen; Por lo tanto, también elimina los bordes. La mayoría de las veces, los

bordes contienen información valiosa sobre la escena y no queremos perderla. El **filtro bilateral** es consciente de la estructura de la escena y tiende a actuar como un filtro borroso clásico cuando está en un área sin bordes; sin embargo, cuando ve una ventaja, cambia su comportamiento; de modo que, el desenfoque no funciona a lo largo de los bordes, sino que funciona a lo largo de los bordes, lo que significa que son **filtros que conservan** los bordes.

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

void main(int argc, char* argv[]) {
    if(argc==1) {
        std::cout << argv[0] << " <image>" << endl;
        return;
    }

    cv::Mat image, output;
    image = cv::imread(argv[1]);
    if(image.empty()) {
        std::cout << "Unable to load the image: " << argv[1] << endl;
        return;
    }

    cv::bilateralFilter(image, output, 3, 5, 3);
}
```

Lea Procesamiento de imágenes en línea:
https://riptutorial.com/es/opencv/topic/2032/procesamiento-de-imagenes

Capítulo 19: Usando los clasificadores en cascada en Java

Sintaxis

- CascadeClassifier cascade = new CascadeClassifier ("cascade.xml"); // Crea un clasificador en cascada desde cascade.xml
- Mat image = Imgcodecs.imread ("image.png"); // Convierte image.png en un objeto Mat (Matrix)
- Detecciones de MatOfRect = nuevo MatOfRect (); // Crea un archivo vacío MatOfRect (Matriz de rectángulos), utilizado como salida para nuestras clases de detección
- detections.toArray (); // Devuelve una matriz de objetos Rect que se pueden iterar sobre
- Imgproc.rectangle (imagen, nuevo punto (rect.x, rect.y), nuevo punto (rect.x + rect.width, rect.y + rect.height), nuevo escalar (0, 255, 0)); // Dibuja un rectángulo con contorno verde desde las ubicaciones x e y del primer punto hasta las ubicaciones x e y del segundo punto en la "imagen" del objeto Mat. "rect" es un objeto Rect, generalmente proporcionado por detections.toArray (). Utiliza la clase de puntos de OpenCV.
- Imgcodecs.imwrite ("output.png", imagen); // Escribe el objeto Mat modificado "imagen" en la "salida.png"
- CascadeClassifier.detectMultiScale (imagen, detecciones); // Detecta cualquier objeto en la "imagen" del objeto Mat y genera las detecciones en el objeto "detecciones" MatOfRect
- CascadeClassifier.detectMultiScale (imagen, detecciones, **scaleFactor** , **minNeighbors** , **flags** , **minSize** , **maxSize**); // Realiza una detección con parámetros adicionales. Vea los detalles abajo.
- Imgproc.ellipse (imagen, centro, **ejes,** 0, 0, 360, nuevo Scalar (255, 0, 255), **espesor, tipo de línea,** 0); // Dibuja una elipse en la imagen en el `center` punto. Utiliza la clase de puntos de OpenCV.

Parámetros

Parámetro	Detalles
factor de escala	Cuánto se reduce el tamaño de la imagen en cada escala de imagen. Predeterminado = `1.1`
minNeighbors	Cuántos vecinos debe tener un rectángulo candidato antes de seleccionarlo como un objeto detectado. Predeterminado = `4`
banderas	Banderas heredadas. En la mayoría de los casos, esto se debe establecer en `0` . Predeterminado = `0`
tamaño minimo	Tamaño mínimo que un rectángulo candidato puede ser. Esto utiliza la clase de `Size` de OpenCV. Puede usarse para disminuir el tiempo de detección y el uso de la CPU, así como para reducir los falsos positivos.

Parámetro	Detalles
tamaño máximo	Tamaño máximo que puede ser un rectángulo candidato. Esto utiliza la clase de `Size` de OpenCV. Puede usarse para disminuir el tiempo de detección y el uso de la CPU, así como para reducir los falsos positivos.
hachas	Utiliza la clase de tamaño de OpenCV. Define el ancho y la altura de la elipse.
espesor	Grosor de la línea, en píxeles.
tipo de línea	Tiene varios parámetros. `0` es la línea continua, `8` es para una línea conectada a 8, `4` es para una línea conectada a 4 y `CV_AA` es para una línea suavizada. Predeterminado = `8`

Examples

Obtención de una imagen estática, detección de elementos en ella y salida de los resultados.

Tenga en cuenta que este ejemplo utiliza OpenCV 3.1.

```
import org.opencv.core.Mat;
import org.opencv.core.MatOfRect;
import org.opencv.core.Point;
import org.opencv.core.Rect;
import org.opencv.core.Scalar;
import org.opencv.imgcodecs.Imgcodecs;
import org.opencv.imgproc.Imgproc;
import org.opencv.objdetect.CascadeClassifier;

public class Classifier {
    private CascadeClassifier diceCascade = new
        CascadeClassifier("res/newMethod/diceCascade.xml");
    private Mat image;
    private String loc = "path/to/image.png";
    private String output = "path/to/output.png";

    public void detImg() {

        Mat image = Imgcodecs.imread(loc); // Reads the image

        MatOfRect diceDetections = new MatOfRect(); // Output container
        diceCascade.detectMultiScale(image, diceDetections); // Performs the detection

        // Draw a bounding box around each detection.
        for (Rect rect : diceDetections.toArray()) {
            Imgproc.rectangle(image, new Point(rect.x, rect.y),
                new Point(rect.x + rect.width, rect.y + rect.height),
                new Scalar(0, 255, 0));
        }

        // Save the visualized detection.
        Imgcodecs.imwrite(output, image);
```

```
    }
}
```

El `Rect[]` devuelto por `diceDetections.toArray()` puede `diceDetections.toArray()` . Cada `Rect` dentro de la matriz tendrá cuatro propiedades principales: `x` , `y` , `width` y `height` . `x` e `y` definen la posición superior izquierda del rectángulo, y el `width` y la `height` devuelven un `int` del ancho y la altura del rectángulo. Esto se utiliza al dibujar rectángulos en imágenes. Los `Imgproc.rectangle` mínimos requeridos de la función `Imgproc.rectangle` son los siguientes:

```
Imgproc.rectangle(Mat image, Point start, Point end, Scalar color);
```

Ambos `Point` se utilizan para las posiciones de la esquina superior izquierda y la esquina inferior derecha. Estas posiciones son *ambas absolutas* para la imagen proporcionada como primer parámetro, no entre sí. Por lo tanto, debe agregar la posición `x` o `y` del rectángulo además del `width` o la `height` para definir correctamente el punto `end` .

Tenga en cuenta que la clase `Point` utilizada en estos parámetros **no** es la clase `Point` la biblioteca estándar de Java. ¡Debes importar la clase `Point` de OpenCV en su lugar!

Detectar imágenes desde un dispositivo de video

Este ejemplo introduce la clase `VideoCapture` , donde la usamos para tomar una imagen de una cámara web y guardarla en una imagen.

```
import org.opencv.core.Mat;
import org.opencv.core.MatOfRect;
import org.opencv.core.Point;
import org.opencv.core.Rect;
import org.opencv.core.Scalar;
import org.opencv.imgcodecs.Imgcodecs;
import org.opencv.imgproc.Imgproc;
import org.opencv.objdetect.CascadeClassifier;
import org.opencv.videoio.VideoCapture;

public class Classifier {
    private CascadeClassifier diceCascade = new
        CascadeClassifier("res/newMethod/diceCascade.xml");
    private Mat image;
    private String loc = "path/to/image.png";
    private String output = "path/to/output.png";
    private VideoCapture vc = new VideoCapture();

    public void detImg() {
        vc.open(0); // Opens the video stream

        Mat image = new Mat(); // Creates an empty matrix
        vc.read(image); // Reads the image from the video stream and
            writes it to the image matrix.

        MatOfRect diceDetections = new MatOfRect(); // Output container
        diceCascade.detectMultiScale(image, diceDetections); // Performs the detection

        // Draw a bounding box around each detection.
```

```
        for (Rect rect : diceDetections.toArray()) {
            Imgproc.rectangle(image, new Point(rect.x, rect.y),
                new Point(rect.x + rect.width, rect.y + rect.height),
                new Scalar(0, 255, 0));
        }

        // Save the visualized detection.
        Imgcodecs.imwrite(output, image);

        vc.release(); // Closes the stream.

    }
}
```

Convertir un objeto Mat en un objeto BufferedImage

Este ejemplo de Daniel Baggio se tomó directamente de esta respuesta de StackExchange , pero se ha vuelto a publicar para visibilidad.

Esta clase toma un objeto Mat y devuelve el objeto BufferedImage utilizado por las bibliotecas `javax.swing` . Esto puede ser utilizado por un objeto `Graphics` para dibujar la imagen.

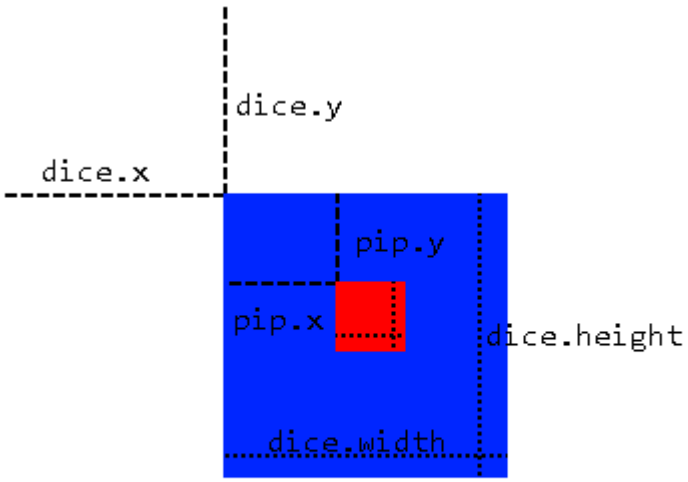
```
private BufferedImage toBufferedImage(Mat m) {
    if (!m.empty()) {
        int type = BufferedImage.TYPE_BYTE_GRAY;
        if (m.channels() > 1) {
            type = BufferedImage.TYPE_3BYTE_BGR;
        }
        int bufferSize = m.channels() * m.cols() * m.rows();
        byte[] b = new byte[bufferSize];
        m.get(0, 0, b); // get all the pixels
        BufferedImage image = new BufferedImage(m.cols(), m.rows(), type);
        final byte[] targetPixels = ((DataBufferByte)
image.getRaster().getDataBuffer()).getData();
        System.arraycopy(b, 0, targetPixels, 0, b.length);
        return image;
    }

    return null;
}
```

Detecciones en Detecciones

Este ejemplo utiliza los dados y los puntos negros en los dados (los puntos) como nuestro objeto. Como el ejemplo es bastante largo, primero es esencial explicar algunos conceptos clave para entender el ejemplo.

Comprensión del primer ejemplo, "Obtención de una imagen estática, detección de elementos en ella y salida de los resultados". Es fundamental para entender este ejemplo, especialmente cómo OpenCV dibuja rectángulos.

Echa un vistazo a la siguiente imagen:



Usaremos el método de subimagen, donde usamos un área detectada como nuestra base para aplicar más detecciones. Esto solo es posible si un objeto siempre estará dentro de otro objeto que podamos detectar, como nuestros pips en nuestros dados. Este método tiene varios beneficios:

- En lugar de escanear toda la imagen, solo necesitamos escanear el área donde sabemos que estará el objeto.
- Elimina cualquier posibilidad de falsos positivos fuera del área de detección.

Hacemos esto aplicando primero un escaneo clasificador en cascada sobre la imagen completa para darnos un objeto `MatOfRect` que contiene nuestros objetos grandes (dados, en este caso). Luego `Rect[]` sobre la matriz `Rect[]` dada por la función `toArray()` del objeto `MatOfRect` . Este objeto `Rect` se usa para crear un objeto `Mat` temporal que se "recorta" a las `Rect` objeto `Rect` (`x, y, width, height`) de la imagen original, donde luego podemos realizar detecciones en el objeto `Mat` temporal. En otras palabras, le decimos al clasificador que solo realice detecciones en las partes de la imagen de los dados, y especificamos la posición de cada dado usando los objetos `Rect` que obtuvimos al realizar una detección en toda la imagen.

Sin embargo, los objetos `Rect` (pips) tienen sus propiedades en relación con sus dados, y no la imagen en sí. Para resolver este problema, cuando deseamos dibujar rectángulos en la imagen real que muestra las ubicaciones de los pips, agregamos `dice.x` y `dice.y` al `Point` inicio.

```
import org.opencv.core.Mat;
import org.opencv.core.MatOfRect;
import org.opencv.core.Point;
import org.opencv.core.Rect;
import org.opencv.core.Scalar;
import org.opencv.core.Size;
import org.opencv.imgcodecs.Imgcodecs;
import org.opencv.imgproc.Imgproc;
import org.opencv.objdetect.CascadeClassifier;
import org.opencv.videoio.VideoCapture;

public class Classifier {
```

```
    private CascadeClassifier diceCascade =
        new CascadeClassifier("res/newMethod/diceCascade.xml");
    private CascadeClassifier pipCascade =
        new CascadeClassifier("res/newMethod/pipCascade6.xml");
    private VideoCapture vc = new VideoCapture();
    private Mat image;

    public void openVC(int index) {
        vc.open(index);
    }

    public void closeVC() {
        vc.close();
    }

    public Mat getNextImage() {
        image = new Mat();
        vc.read(image); // Sets the matrix to the current livestream frame.

        MatOfRect diceDetections = new MatOfRect(); // Output container

        // See syntax for explainations on addition parameters
        diceCascade.detectMultiScale(image, diceDetections, 1.1, 4, 0, new Size(20, 20),
            new Size(38, 38));

        // Iterates for every Dice ROI
        for (int i = 0; i < diceDetections.toArray().length; i++) {
            Rect diceRect = diceDetections.toArray()[i];

            // Draws rectangles around our detected ROI
            Point startingPoint = new Point(diceRect.x, diceRect.y);
            Point endingPoint = new Point(diceRect.x + diceRect.width,
                diceRect.y + diceRect.height);
            Imgproc.rectangle(image, startingPoint, endingPoint, new Scalar(255, 255, 0));

            MatOfRect pipDetections = new MatOfRect();

            pipCascade.detectMultiScale(image.submat(diceRect), pipDetections, 1.01, 4, 0,
                new Size(2, 2), new Size(10, 10));

            // Gets the number of detected pips and draws a cricle around the ROI
            for (int y = 0; y < pipDetections.toArray().length; y++) {
                // Provides the relative position of the pips to the dice ROI
                Rect pipRect = pipDetections.toArray()[y];

                // See syntax explaination
                // Draws a circle around our pips
                Point center = new Point(diceRect.x + pipRect.x + pipRect.width / 2,
                    diceRect.y + pipRect.y + pipRect.height / 2);
                Imgproc.ellipse(image, center, new Size(pipRect.width / 2, pipRect.height /
2),
                     0, 0, 360, new Scalar(255, 0, 255), 1, 0, 0);
            }
        }

        return image;
    }
}
```

La función `getNextImage()` devuelve un objeto `Mat` , que junto con los otros ejemplos publicados, se

puede llamar constantemente y se puede convertir en un `BufferImage` , para proporcionar detecciones de visualización en vivo.

Lea Usando los clasificadores en cascada en Java en línea:
https://riptutorial.com/es/opencv/topic/6377/usando-los-clasificadores-en-cascada-en-java

Capítulo 20: Utilizando VideoCapture con OpenCV Python

Examples

Leyendo cuadros desde un video pre-capturado

```
import numpy as np
import cv2

#access a video from your disk
#to use the GIF in this example, convert to avi!
cap = cv2.VideoCapture('eg_videoRead.avi')


#we are going to read 10 frames
#we store the frames in a numpy structure
#then we'll generate a minimum projection of those frames

frameStack=[]
numFrames=10

for fr in range(numFrames):
    cap.set(cv2.CAP_PROP_POS_FRAMES,fr) #specifies which frame to read next
    frame=cap.read() #read the frame
    #gray = cv2.cvtColor(frame[1], cv2.COLOR_BGR2GRAY) #convert to gray scale
    frameStack.append(frame[1]) #add current frame to our frame Stack

minProjection=np.min(frameStack,axis=0) #find the minimum across frames
cv2.imshow("projection", minProjection) #show the result
```

Utilizando VideoCapture con OpenCV Java

No hay imagen en el java, necesitas escribir un método para esto. Este método es un Mat2bufferedImage. Toma el objeto mat como parámetro y devuelve la imagen.

```
public static void main(String[] args) {
    Mat frame = new Mat();
    //0; default video device id
    VideoCapture camera = new VideoCapture(0);
    JFrame jframe = new JFrame("Title");
    jframe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
    JLabel vidpanel = new JLabel();
    jframe.setContentPane(vidpanel);
    jframe.setVisible(true);

    while (true) {
        if (camera.read(frame)) {

            ImageIcon image = new ImageIcon(Mat2bufferedImage(frame));
            vidpanel.setIcon(image);
            vidpanel.repaint();

        }
    }
}
```

Lea Utilizando VideoCapture con OpenCV Python en línea:
https://riptutorial.com/es/opencv/topic/6803/utilizando-videocapture-con-opencv-python

Creditos

S. No	Capítulos	Contributors
1	Empezando con opencv	Arijit, bburns.km, Berriel, Community, Elizabeth, hackhisass, jlarsch, John Hany, K D, MD. Nazmul Kibria, mesutpiskin, snb, StephenG, Sunreef, winseybash, Yassie, zeeshan khan
2	Acceso a píxeles	Adi Shavit, brian, cagatayodabasi, Ehsan Ab, Elizabeth, smttsp, Sunreef
3	Cargar y guardar varios formatos de medios	Adi Shavit, cagatayodabasi, Jav_Rock, Lakshya Kejriwal, MD. Nazmul Kibria
4	Clasificadores en cascada	Arijit, MD. Nazmul Kibria, mesutpiskin
5	Contraste y brillo en C ++	Ehsan Ab, MD. Nazmul Kibria
6	Creando un video	mesutpiskin
7	Detección de bordes	cmastudios, Ehsan Ab, K D, m3h0w, Sounak
8	Detección de manchas	MD. Nazmul Kibria, Sebastian
9	Detección de objetos	K D, mesutpiskin
10	Dibujar formas (línea, círculo, ..., etc.) en C ++	CroCo
11	Estructuras basicas	smttsp
12	Funciones de dibujo en Java	mesutpiskin
13	Genere y compile opencv 3.1.0-dev para Python2 en Windows usando CMake y Visual Studio	Tes3awy
14	Inicialización de	David Miguel

	OpenCV en Android	
15	Instalación de OpenCV	amorenew
16	Modificación de contenido de imagen	Adi Shavit, DivyaMaheswaran, smttsp
17	Mostrar imagen OpenCV	Aleksandar, Elizabeth, jlarsch, mesutpiskin, smttsp
18	Procesamiento de imágenes	cagatayodabasi, Dan Mašek, Elizabeth, jlarsch, Shubham Batra , Sunreef, Utkarsh Sinha
19	Usando los clasificadores en cascada en Java	Edward Shen
20	Utilizando VideoCapture con OpenCV Python	jlarsch, mesutpiskin