**eBook Gratuit**

# APPRENEZ opencv

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#opencv

# Table des matières

# À propos

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

# Chapitre 1: Commencer avec opencv

## Remarques

Cette section fournit une vue d'ensemble de ce qu'est une ouverture, et pourquoi un développeur peut vouloir l'utiliser.

Il devrait également mentionner tous les grands sujets à l'intérieur du forum, et les relier aux sujets connexes. La documentation de opencv étant nouvelle, vous devrez peut-être créer des versions initiales de ces rubriques connexes.

## Versions

# OpenCV 3

| Version | Date de sortie |
|---|---|
| 3.2 | 2016-12-23 |
| 3.1 | 2015-12-18 |
| 3.0 | 2015-06-03 |
| 3.0 RC1 | 2015-04-23 |
| 3.0 bêta | 2014-11-07 |
| 3.0 alpha | 2014-08-21 |

# OpenCV 2

| Version | Date de sortie |
|---|---|
| 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 |

| Version | Date de sortie |
| --- | --- |
| 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

## Charger et afficher une image avec OpenCV

Avec cet exemple, nous verrons comment charger une image couleur à partir du disque et l'afficher à l'aide des fonctions intégrées d'OpenCV. Pour ce faire, nous pouvons utiliser les liaisons C / C ++, Python ou Java.

**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 n'a aucune fenêtre nommée ni équivalent dans opencv java. Utilisez swing ou swt pour afficher l'image.

### Construire et installer OpenCV à partir des sources

Ceci est un guide étape par étape pour installer OpenCV 3 sur un système Linux basé sur Debian à partir de la source. Les étapes doivent rester les mêmes pour les autres distributions, il suffit de remplacer les commandes du gestionnaire de packages pertinentes lors de l'installation des packages pour la génération.

**Remarque:** Si vous n'avez pas envie de perdre du temps à créer des éléments ou à ne pas aimer le terminal, vous pouvez très probablement installer OpenCV à partir de l'interface graphique du gestionnaire de paquets Synaptic. Cependant, ces bibliothèques sont souvent obsolètes.

## Préparez-vous à la construction

Emettez les commandes suivantes dans votre terminal pour installer les packages requis:

```
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
```

Les packages suivants sont facultatifs:

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

Emettez la commande suivante pour obtenir le code source OpenCV et préparer la génération:

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

## Construire et installer

Nous incluons les exemples dans la construction, mais n'hésitez pas à les laisser de côté. N'hésitez pas non plus à définir d'autres drapeaux et à personnaliser votre build comme vous le souhaitez.

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

Si CMake n'a pas signalé d'erreurs ou de bibliothèques manquantes, poursuivez la construction.

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

Si aucune erreur n'a été produite, nous pouvons continuer à installer OpenCV sur le système:

```
sudo make install
```

Maintenant, OpenCV devrait être disponible pour votre système. Vous pouvez utiliser les lignes suivantes pour savoir où OpenCV a été installé et quelles bibliothèques ont été installées:

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

## Test d'installation

Nous construisons d’abord les exemples C ++:

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

Si aucune erreur n'a été produite, exécutez un échantillon, par exemple

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

Si l'exemple s'exécute, les bibliothèques C ++ sont correctement installées.

Ensuite, testez les liaisons Python:

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

Si ces commandes importent OpenCV et impriment la version correcte sans se plaindre, alors les liaisons Python sont correctement installées.

Félicitations, vous venez de construire et d'installer OpenCV. Bonne programmation!

Pour Mac, reportez-vous ici à l' installation d'OpenCV sur Mac OS X

### Bonjour exemple mondial en java

Image OpenCv lue depuis le système de fichiers 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());
    }
}
```

### Obtenez une image de webcam

Affichez un flux vidéo en direct provenant d'une webcam à l'aide de la classe VideoCapture d'OpenCV avec Java, C / C ++ et 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;
}
```

# Python

```
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()
```

## Premiers pas avec OpenCV 3.1 sous Windows

Nous installons OpenCV 3.1.0 sous Windows et commençons. Il existe deux manières d'installer OpenCV sur Windows. L'une consiste à télécharger le programme d'installation et à l'exécuter. L'autre consiste à construire à partir des sources.

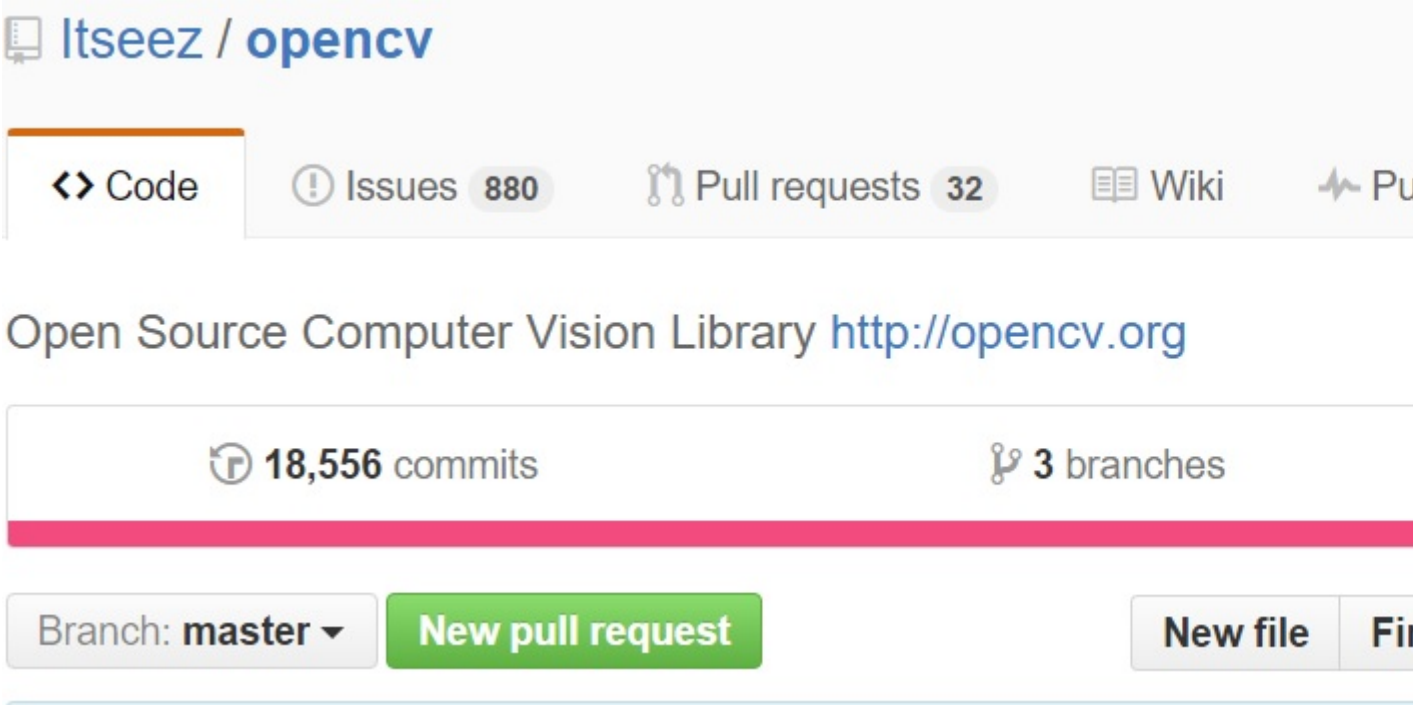
C'est le moyen le plus simple d'installer OpenCV et de commencer. OpenCV fournit des fichiers binaires de pré-compilation à installer sur Windows ici . Une fois le téléchargement terminé, extrayez-le et installez-le sur le chemin choisi.

**ProTip:** Assurez-vous que votre chemin OpenCV n'inclut aucun espace. Donc, c'est mieux si vous l'installez simplement dans le répertoire racine C: \ ou D: \

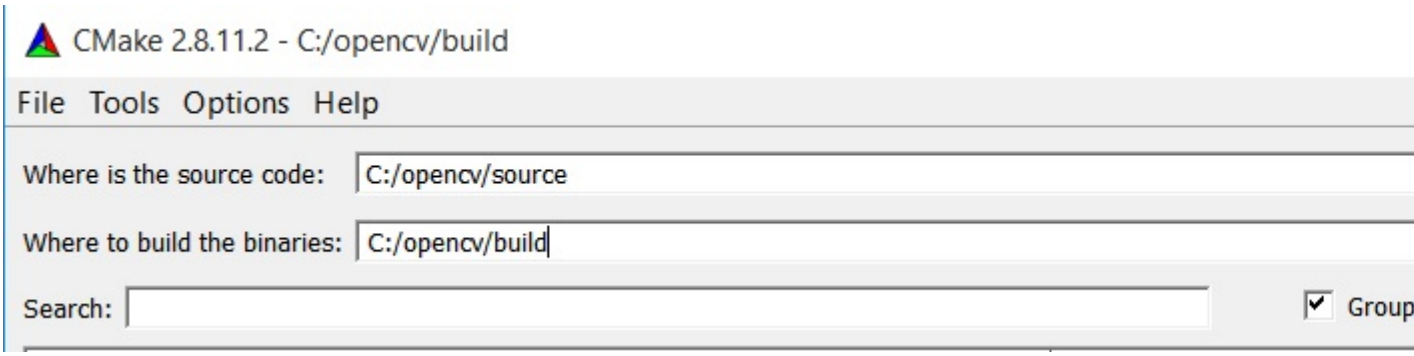
Le problème avec la méthode ci-dessus est que vous ne pouvez pas utiliser les modules opencv_contrib. En outre, il ne vient pas avec tous les outils et bibliothèques tiers. Donc, si vous voulez utiliser tout cela, suivez simplement.

Je vous expliquerai le minimum pour installer OpenCV depuis la source. Pour plus avancé, référez-vous ici .

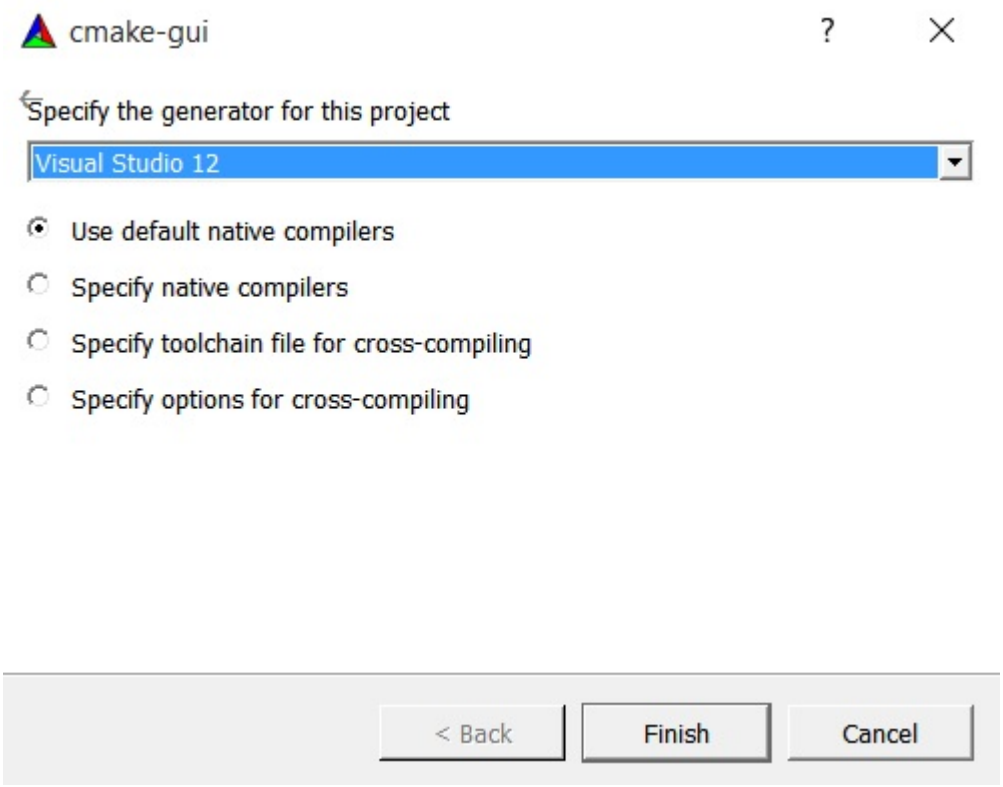
- Installez CMake .
- Clone source OpenCV de https://github.com/Itseez/opencv.git dans un répertoire qui n'a pas d'espaces. Appelons-le "OpenCVdir".




- Maintenant, ouvrez l'interface graphique de CMake et ajoutez votre répertoire source (OpenCVdir) au menu Sources et créez le répertoire dans le menu de génération. **Conseil:** S'il n'y a pas de répertoire de compilation, créez-en un dans votre dossier opencv.

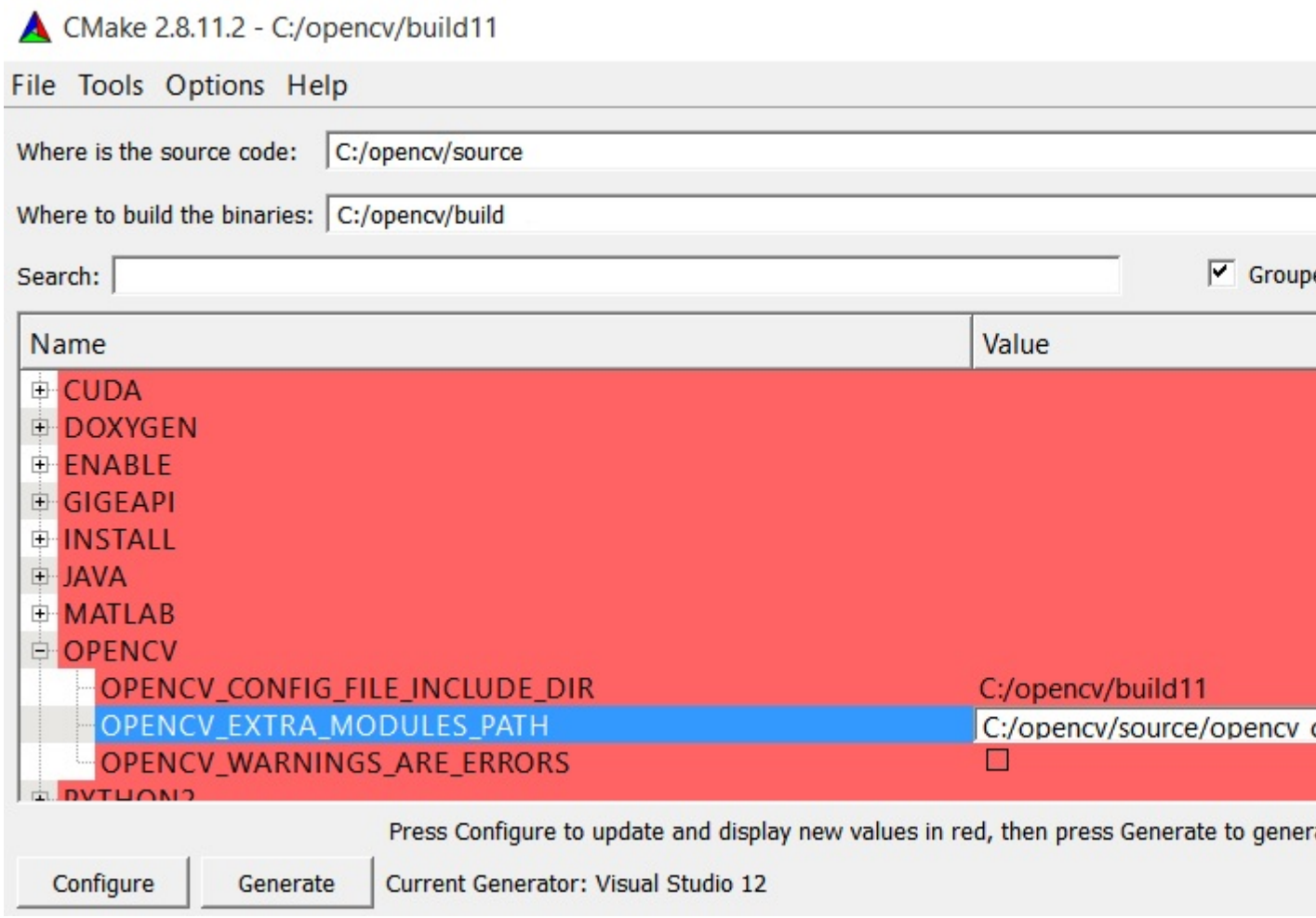



- Cliquez sur Configurer et sélectionnez votre version du compilateur Visual Studio. J'ai eu Visual Studio 2013 Professional 32 bits, j'ai donc choisi le compilateur Visual Studio 12.




**Conseil:** vous pouvez télécharger Visual Studio 2013 Professional à partir d'ici. Il est livré avec 30 jours d'essai + 90 jours de prolongation après s'être connecté.

- Appuyez sur Terminer et CMake chargera tous les paquets automatiquement. Vous pouvez ajouter ou supprimer des packages. Appuyez à nouveau sur Configure.
- Si vous voulez construire avec des modules opencv_contrib supplémentaires, vous devez les télécharger ici . Ensuite, extrayez-les et ajoutez le répertoire opencv_contrib / modules à votre CMake comme indiqué ci-dessous.

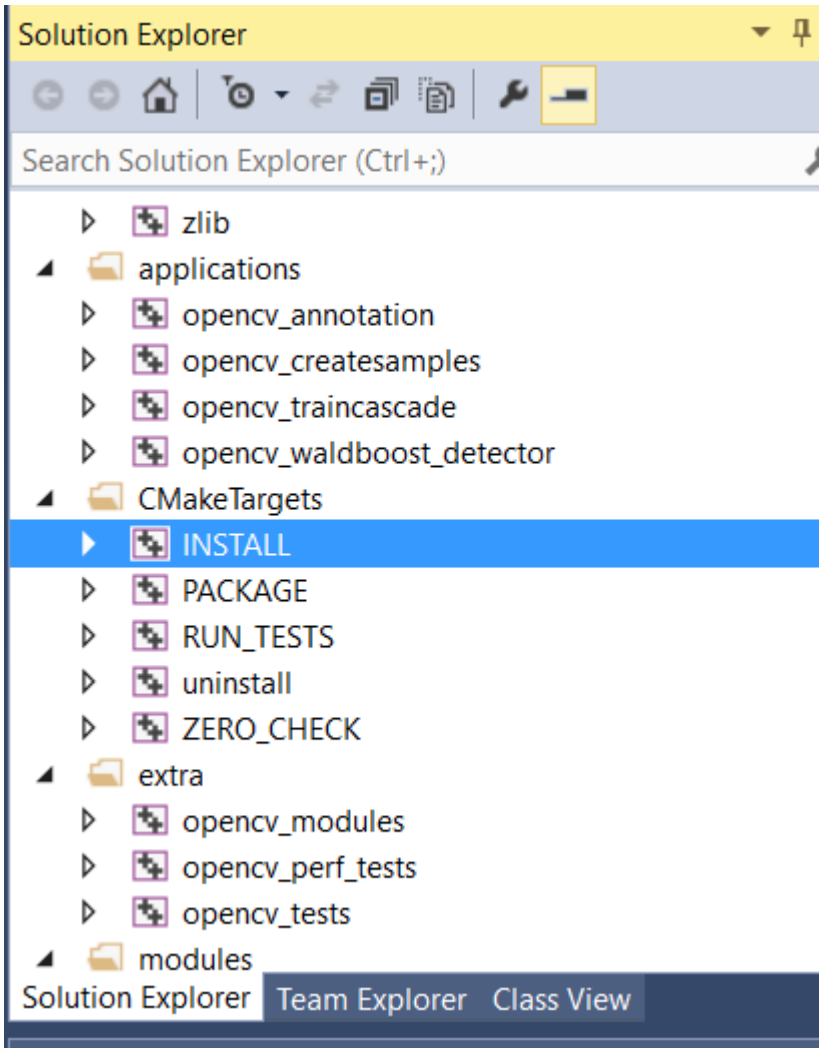



- Maintenant, appuyez à nouveau sur Configurer, puis appuyez sur Générer.
- Fermer CMake. Allez dans le dossier your_opencv \ build et ouvrez le fichier nommé 'OpenCV.sln'. - Il va ouvrir Visual Studio. Maintenant, lancez-le dans les deux Debug


mode et libération


mode.

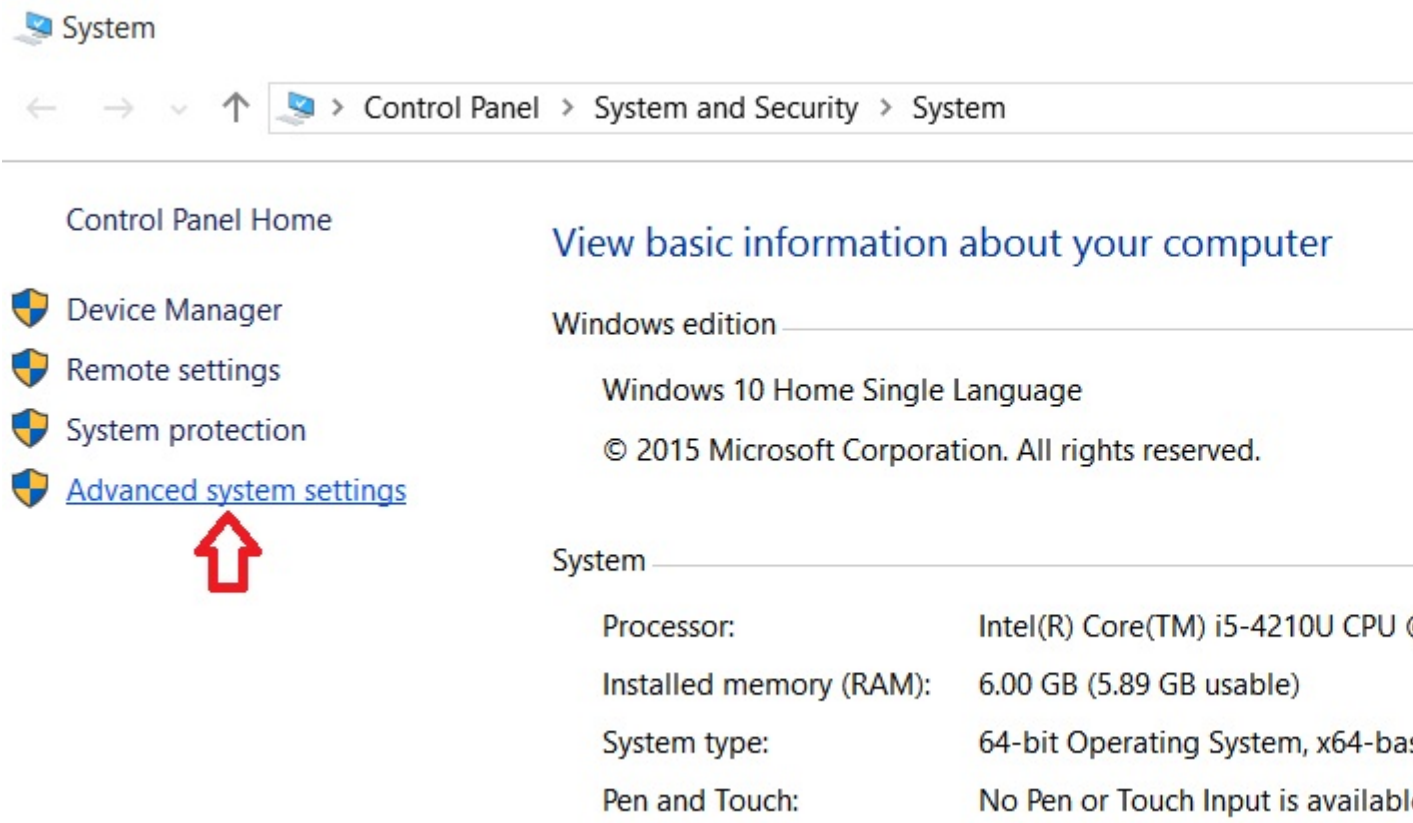
- Maintenant, dans l'explorateur de solutions en haut à droite de Visual Studio, sélectionnez le projet INSTALL et construisez-le.



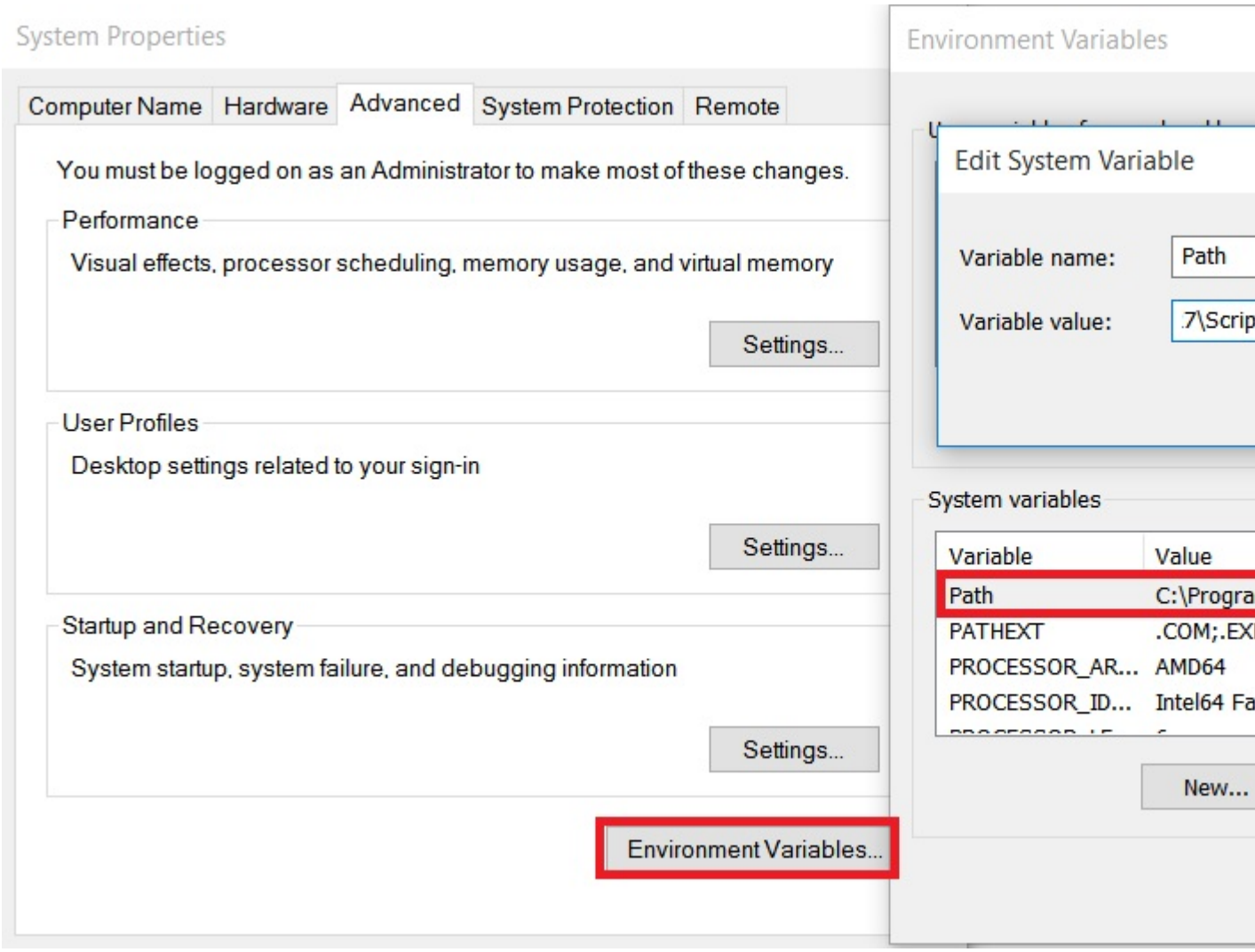

Hourra!! Profitez de votre OpenCV.

**L'ajout du répertoire d'inclusion OpenCV à la variable PATH de la variable d'environnement:**

- Accédez à Propriétés système et cliquez sur Paramètres système avancés.




- Maintenant, cliquez sur Variables d'environnement >> Chemin >> Modifier.




- Ici, ajoutez le dossier bin situé dans votre dossier OpenCVdir / build / install / x86 / vc ** / bin à cette variable. Veillez à ne pas remplacer les valeurs de chemin existantes.
- Après cela, vous devez redémarrer votre système pour que les variables d'environnement changent et que vous êtes maintenant prêt.

## Quoi et pourquoi OPENCV?

OpenCV (Open Source Computer Vision Library) est une bibliothèque de logiciels et de logiciels d'apprentissage automatique. Il a été conçu pour divers objectifs tels que l'apprentissage automatique, la vision par ordinateur, l'algorithme, les opérations mathématiques, la capture vidéo, le traitement d'images, etc. , android, ios). En outre, il a enveloppé dans divers langages de programmation renommés. Dans le cadre du contrat de licence, les entreprises peuvent accéder au code et le modifier.

La bibliothèque contient plus de 2500 algorithmes optimisés, qui ont une excellente précision en termes de performances et de vitesse. Ces algorithmes peuvent être utilisés pour détecter et reconnaître des visages, identifier des objets, classer des actions humaines dans des vidéos, suivre des mouvements de caméra, suivre des objets en mouvement, extraire des modèles 3D d'objets, produire des nuages de points 3D à partir de caméras stéréo image d'une scène entière, trouver des images similaires à partir d'une base de données d'images, supprimer les yeux rouges

des images prises, suivre les mouvements oculaires, reconnaître les décors et établir des marqueurs pour la superposer à la réalité augmentée. , les développeurs et les chercheurs, le nombre est plus de 47 000 et le nombre estimé de téléchargements dépasse 7 millions. La bibliothèque est largement constituée de sociétés professionnelles, de groupes de recherche et d'autres groupes.

De nombreuses entreprises bien établies telles que Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda et Toyota qui emploient la bibliothèque, ainsi que de nombreuses startups telles que Applied Minds, VideoSurf et Zeitera, utilisent largement OpenCV. Les utilisations déployées d'OpenCV s'étendent de l'assemblage d'images StreetView, à la détection d'intrusions dans la vidéo de surveillance en Israël, à la surveillance des équipements miniers en Chine, à la détection des accidents de noyade en Europe et à l'art interactif. L'Espagne et New York vérifient les pistes en Turquie et inspectent les étiquettes des produits dans les usines du monde entier pour détecter rapidement les visages au Japon. Il possède des interfaces C ++, C, Python, Java et MATLAB et supporte Windows, Linux, Android et Mac OS. OpenCV s'appuie principalement sur les applications de vision en temps réel et tire parti des instructions MMX et SSE lorsqu'elles sont disponibles. Des interfaces complètes CUDA et OpenCL sont en cours de développement. Il existe plus de 500 algorithmes et environ 10 fois plus de fonctions qui composent ou supportent ces algorithmes. OpenCV est écrit nativement en C ++ et possède une interface basée sur des modèles qui fonctionne de manière transparente avec les conteneurs STL.

*Informations collectées sur le site officiel*

Lire Commencer avec opencv en ligne: https://riptutorial.com/fr/opencv/topic/800/commencer-avec-opencv

# Chapitre 2: Accès aux pixels

## Remarques

Veillez à bien connaître le type de `cv::Mat` vous parlez. Par exemple, si vous avez un `cv::Mat` de type `CV_8UC3` , mais que vous y accédez avec `image.at<uchar>(r,c)` aucune erreur ne se produira, mais votre programme aura un comportement inattendu.

## Examples

### Accédez à des valeurs de pixel individuelles avec cv :: Mat :: at ()

Pour accéder aux valeurs de pixels dans un objet OpenCV `cv::Mat` , vous devez d'abord connaître le *type* de votre matrice.

Les types les plus courants sont:

- `CV_8UC1` pour les images en niveaux de gris à 1 canal sur 8 bits;
- `CV_32FC1` pour les images en niveaux de gris à 1 canal à virgule flottante 32 bits;
- `CV_8UC3` pour les images couleur à 3 canaux à 8 bits; et
- `CV_32FC3` pour les images couleur à 3 canaux en virgule flottante 32 bits.

Le paramètre par défaut avec `cv::imread` créera une matrice `CV_8UC3` .

Pour accéder aux pixels individuels, le moyen le plus sûr, mais pas le plus efficace, consiste à utiliser la méthode `cv::Mat::at<T>(r,c)` où `r` est la *ligne* de la matrice et `c` la *colonne* . L'argument du modèle dépend du type de la matrice.

Disons que vous avez une `cv::Mat image` . Selon son type, la méthode d'accès et le type de couleur de pixel seront différents.

- Pour `CV_8UC1` : `uchar pixelGrayValue = image.at<uchar>(r,c)` .
- Pour `CV_8UC3` : `cv::Vec3b pixelColor = image.at<cv::Vec3b>(r,c)` . L'objet `cv::Vec3b` représente un triplet de valeurs `uchar` (entiers compris entre 0 et 255).
- Pour `CV_32FC1` : `float pixelGrayValue = image.at<float>(r,c)` .
- Pour `CV_32FC3` : `cv::Vec3f pixelColor = image.at<cv::Vec3f>(r,c)` . L'objet `cv::Vec3f` représente un triplet de valeurs `float` .

Notez qu'OpenCV représente les images dans l'ordre des ***lignes principales*** , comme par exemple Matlab ou la convention en Algèbre. Ainsi, si vos coordonnées de pixel sont `(x,y)` , vous accéderez au pixel en utilisant `image.at<..>(y,x)` .

Alternativement, `at<>` également prendre en charge l'accès via un seul argument `cv::Point` . Dans ce cas, l'accès se fait en *colonne majeure* :

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

Consultez la documentation OpenCV pour plus de détails sur cette méthode.

## Accès efficace aux pixels en utilisant cv :: Mat :: ptr aiguille

Si l'efficacité est importante, un moyen rapide d'itérer sur des pixels dans un objet `cv::Mat` consiste à utiliser sa méthode `ptr<T>(int r)` pour obtenir un pointeur sur le début de la ligne `r` (index basé sur 0).

Selon le type de matrice, le pointeur aura un modèle différent.

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

Cet objet `ptr` peut alors être utilisé pour accéder à la valeur de pixel sur la ligne `r` et la colonne `c` en appelant `ptr[c]` .

Pour illustrer cela, voici un exemple où nous chargeons une image à partir du disque et inversons ses canaux bleu et rouge, fonctionnant pixel par pixel:

```
#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;
}
```

## Définition et obtention des valeurs de pixels d'une image grise 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;
}
```

## Accès aux pixels alternatif avec le Matiterator

Ce n'est pas la meilleure façon de parcourir les pixels. cependant, c'est mieux que cv :: Mat :: at <T>.

Supposons que vous ayez une image couleur dans votre dossier et que vous souhaitiez itérer chaque pixel de cette image et effacer les canaux vert et rouge (notez que ceci est un exemple, vous pouvez le faire de manière plus optimisée);

```
#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;
}
```

Pour compiler ceci avec Cmake:

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

L'image originale:

L'image traitée:

Notez que nous ne touchons que le canal bleu.

Pour plus d'informations: http://docs.opencv.org/2.4/opencv_tutorials.pdf Page: 145

## Accès aux pixels dans Mat

L'accès individuel aux pixels dans la structure OpenCV Mat peut être réalisé de plusieurs manières. Pour comprendre comment accéder, il est préférable d'apprendre d'abord les types de données.

Structures de base explique les types de données de base. `CV_<bit-depth>{U|S|F}C(<number_of_channels>)` est la structure de base d'un type. Parallèlement à cela, il est important de comprendre les structures `Vec` .

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

où type est l'un des `uchar, short, int, float, double` et les caractères de chaque type sont respectivement `b, s, i, f, d` .

Par exemple, Vec2b indique un `unsigned char vector of 2 channels` .

Considérez `Mat mat(R,C,T)` où R est #rows, C est #cols et T est le type. Voici quelques exemples d'accès à la coordonnée (i, j) de `mat` :

**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
```

Notez qu'il est très important d'entrer un type correct dans `<...>` , sinon vous pouvez avoir une erreur d'exécution ou des résultats indésirables.

Lire Accès aux pixels en ligne: https://riptutorial.com/fr/opencv/topic/1957/acces-aux-pixels

# Chapitre 3: Afficher l'image OpenCV

## Examples

### Lecture de base et affichage d'une image

```
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()
```

Pour contrôler la taille de la fenêtre d'affichage à l'écran, ajoutez les commandes suivantes avant la commande 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)
```

voir les documents openCV pour plus de détails

### Lecture de MJPEG depuis une caméra 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
```

Chaque `JPEG` commence par `0xff 0xd8` et se termine par `0xff 0xd9` . Entre ceux-ci se trouve l'image réelle. Informations détaillées dans cette réponse SO

## Afficher l'image OpenCV Java

Image de lecture de base 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 vous voulez voir des images, vous ne pouvez pas utiliser imshow car OpenCV-java n'a pas cette méthode non plus. Au lieu de cela, vous pouvez écrire la méthode suivante.

```
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;
}
```

Vous pouvez voir l'objet de résultat dans Jframe, Jlabel (icône jlabel), etc.

Lire Afficher l'image OpenCV en ligne: https://riptutorial.com/fr/opencv/topic/3306/afficher-l-image-opencv

# Chapitre 4: Chargement et enregistrement de différents formats de supports

## Examples

### Chargement des images

```
#include <highgui.h>

//...

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

...

### Chargement de vidéos

Montrer comment utiliser `cv::VideoCapture` . Voici l'exemple du chargement de la vidéo à partir d'un fichier:

```
#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;
}
```

## Capture en direct

Montrer comment utiliser `cv::VideoCapture` avec par exemple une webcam. Capturer les images de la webcam et l'afficher. Voici le code exemple:

```
#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;
}
```

## Enregistrement de vidéos

Montrer comment utiliser 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;
```

}

## Enregistrer des images

En fait, l'exemple de Live Capture est utile pour capturer des images. Je l'utilise donc pour capturer des images et les enregistrer dans un dossier.

```
#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;
}
```

Lire Chargement et enregistrement de différents formats de supports en ligne: https://riptutorial.com/fr/opencv/topic/6658/chargement-et-enregistrement-de-differents-formats-de-supports

# Chapitre 5: Classificateurs en cascade

## Examples

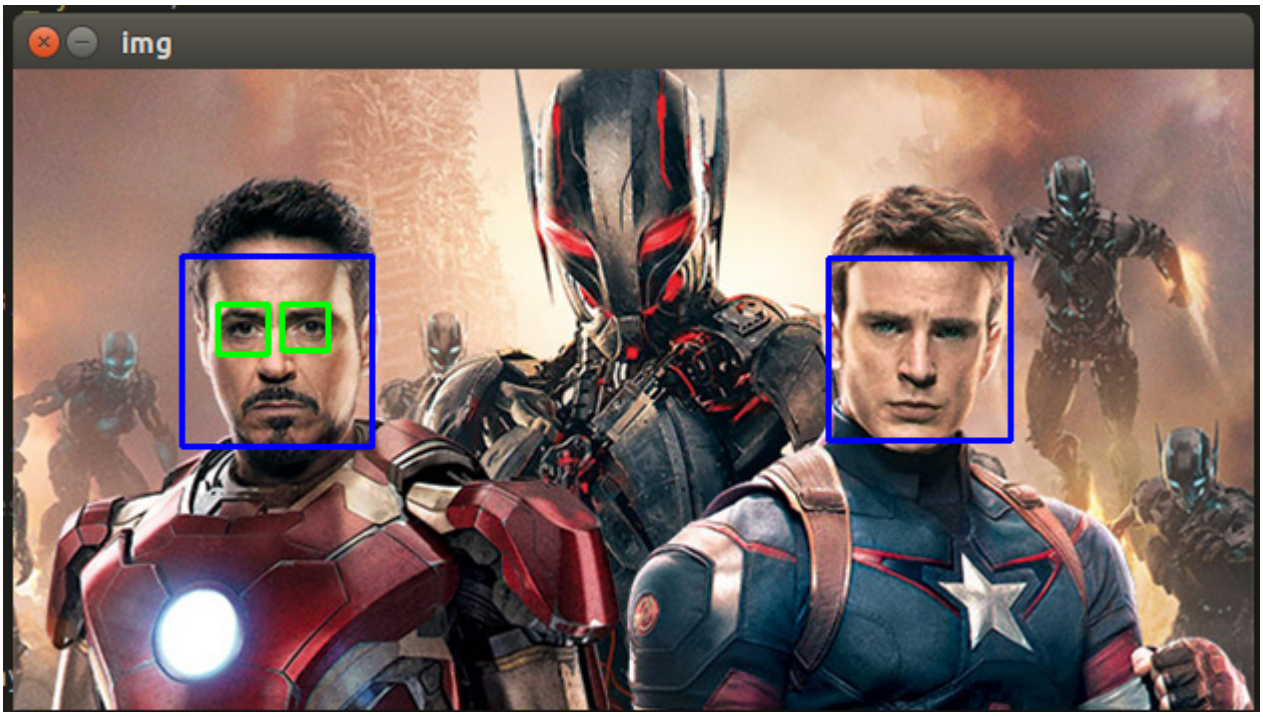
### Utilisation de classificateurs en cascade pour détecter le visage

# Python

## Code

```
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()
```

## Résultat




### Classificateurs en cascade pour détecter le visage avec Java

## Java

### Code

```
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");
    }


}
```

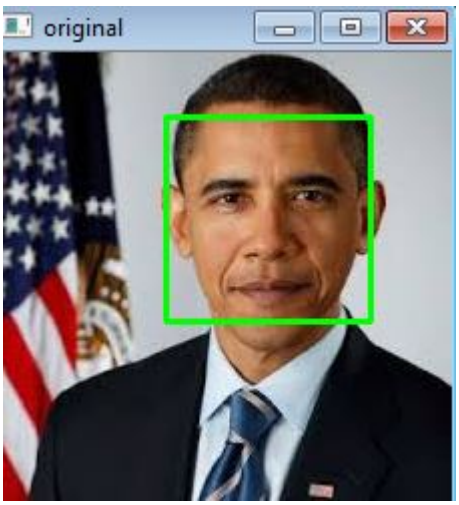
Résultat

### Détection de visage à l'aide d'un classificateur en cascade de haar

## 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);
}
```




Lire Classificateurs en cascade en ligne:

https://riptutorial.com/fr/opencv/topic/6562/classificateurs-en-cascade

# Chapitre 6: Construire et compiler opencv 3.1.0-dev pour Python2 sous Windows en utilisant CMake et Visual Studio

## Remarques

Construire et compiler `opencv 3.1.0-dev` pour obtenir un accès pour **les modules non libres** peut être un casse-tête pour certaines personnes, en particulier sur les machines Windows. Contrairement à Ubuntu, la configuration d'opencv pour Windows prend un certain temps et nécessite deux dépendances de pf pour être installées avant de générer et de compiler.

Les programmes que vous devez télécharger et installer avant d'aller plus loin dans n'importe quelle étape sont les suivants:

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

**Si vous souhaitez télécharger Python pour Win32, vous devez également télécharger CMake pour Win32 même si vous utilisez un ordinateur 64 bits.**

**Il est recommandé de télécharger les programmes 32 bits car certaines bibliothèques Python ne sont prises en charge que par les machines 32 bits. Pour éviter les problèmes, installez tout en version 32 bits.**

3. Communauté Visual Studio 2013

4. Numpy pour Python2.7 Win32

Après avoir installé toutes les dépendances ci-dessus, ***redémarrez*** votre PC et vous serez prêt à passer à l'étape suivante.

---

**Étape 2:**

Si vous n'êtes pas le type de personne qui préfère lire, vous pouvez regarder ce tutoriel . Le tutoriel vous emmène d'ici à la fin de cette documentation.

Vous devrez obtenir **opencv** et **opencv_contrib à** partir de **github** . Vous pouvez trouver les deux à:

1. opencv
2. opencv_contrib

Créez un répertoire nommé opencv-3.1.0 dans lequel vous allez créer deux autres répertoires, l'un pour la *construction* et l'autre pour les *sources* . Vous allez mettre les deux fichiers zip téléchargés dans le fichier source après l'extraction.

Par exemple, votre répertoire opencv-3.1.0 se trouve dans le lecteur C, vous aurez donc trois chemins:

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

Le troisième répertoire comprendra deux chemins:

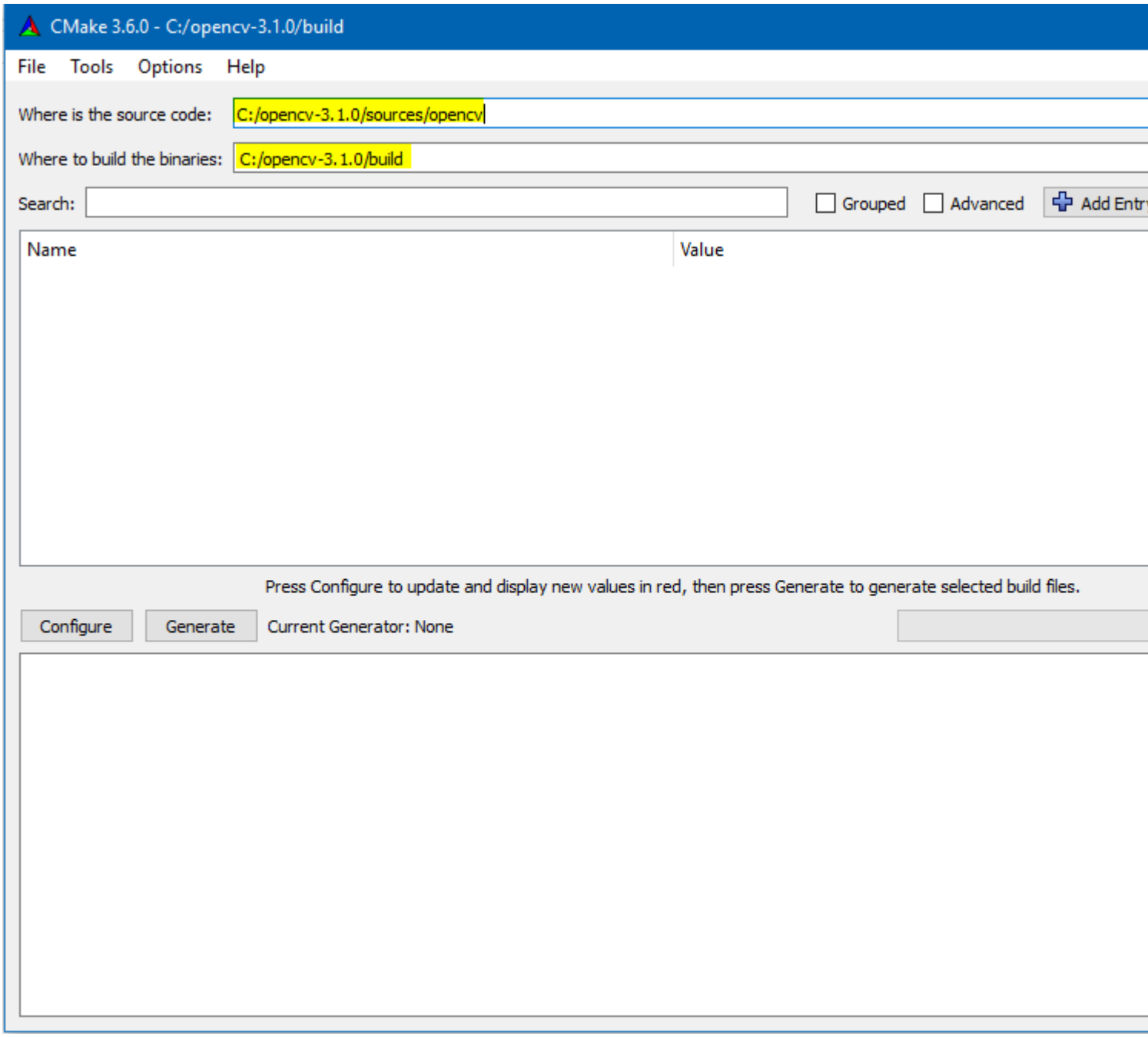
1. `C:\opencv-3.1.0\sources\opencv`
2. `C:\opencv-3.1.0\sources\opencv_contrib`

**Maintenant c'est fait avec la préparation. Permet de faire des choses utiles.**

---

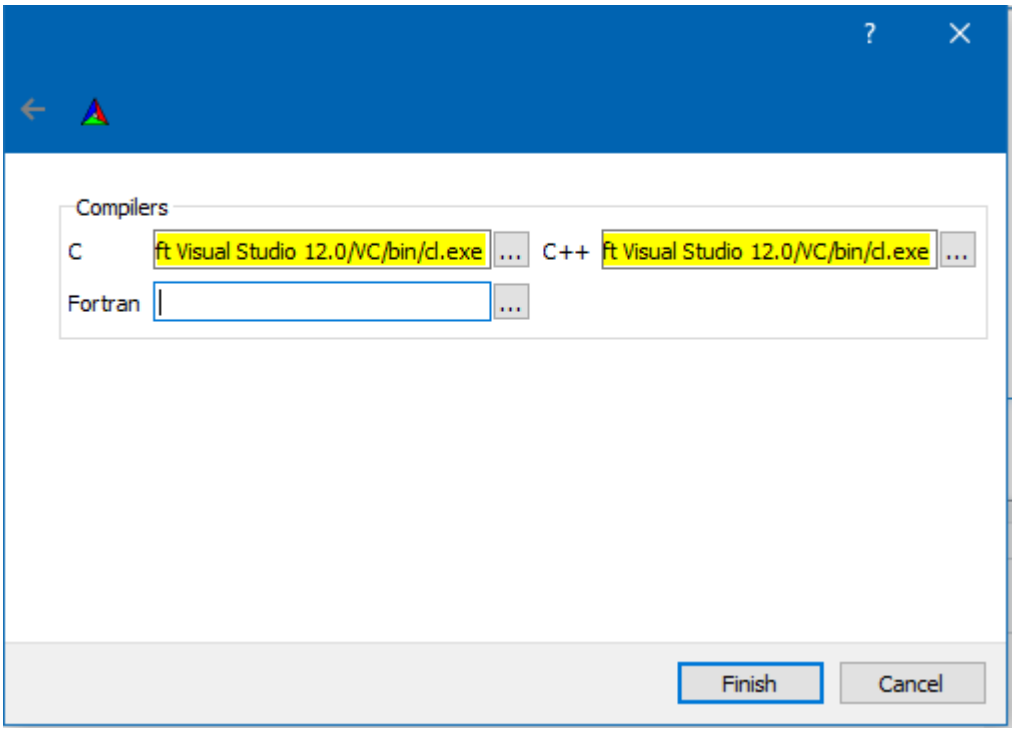
### Étape 3:

Exécutez CMake en tant qu'administrateur. Une fenêtre comme celle-ci apparaîtra et vous devrez fournir deux répertoires, l’un pour les sources et l’autre pour l’endroit où les fichiers seront compilés. L'image ci-dessous peut vous aider mieux que les mots.

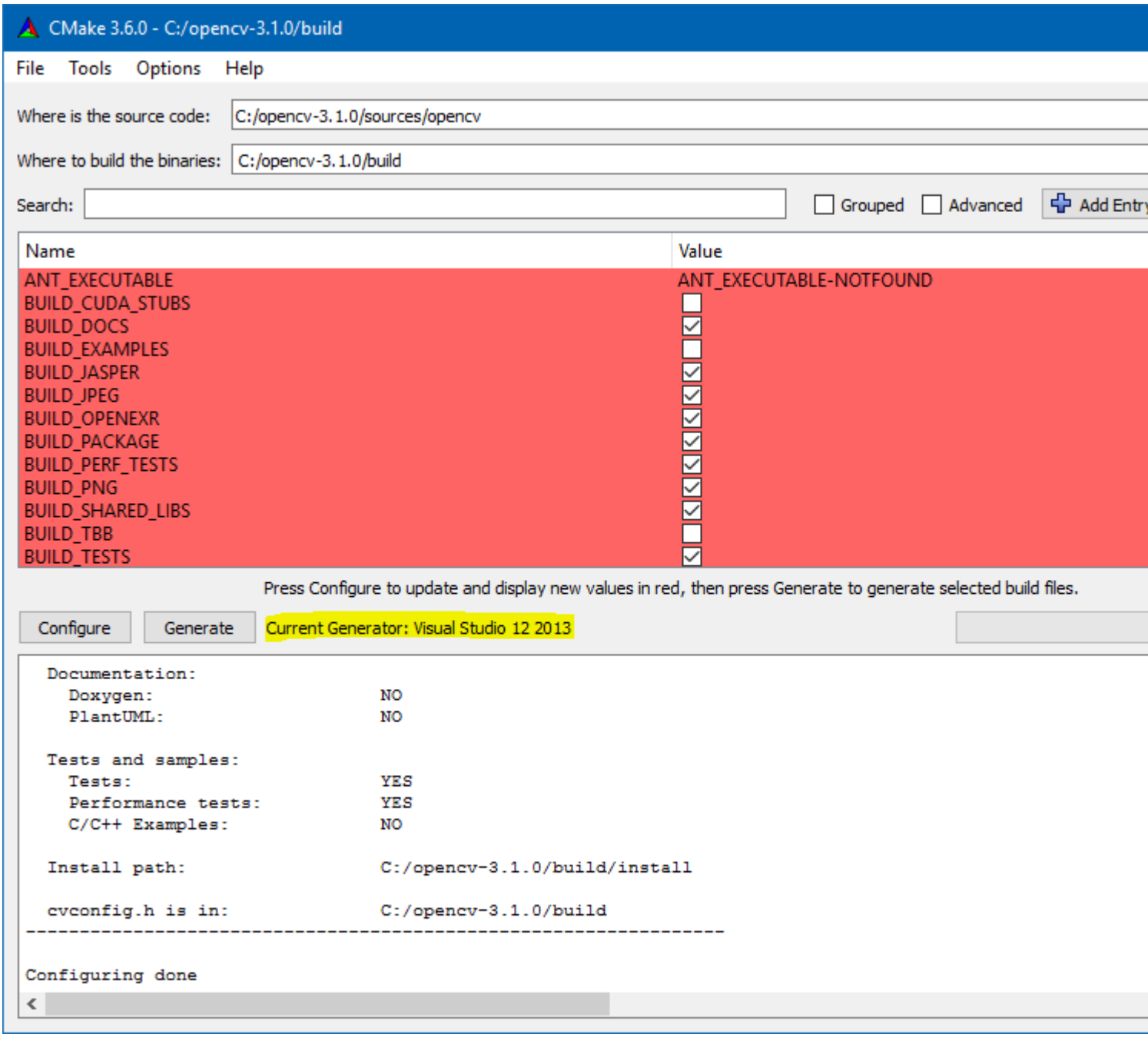



Ensuite, cliquez sur **configurer** et vous serez promu pour fournir les générateurs; c'est-à-dire des compilateurs; pour opencv. Vous devez fournir le `cl.exe` situé dans Microsoft Visual Studio 2013. Cliquez sur **spécifiez les générateurs natifs** et une fenêtre pop-up apparaîtra comme suit:

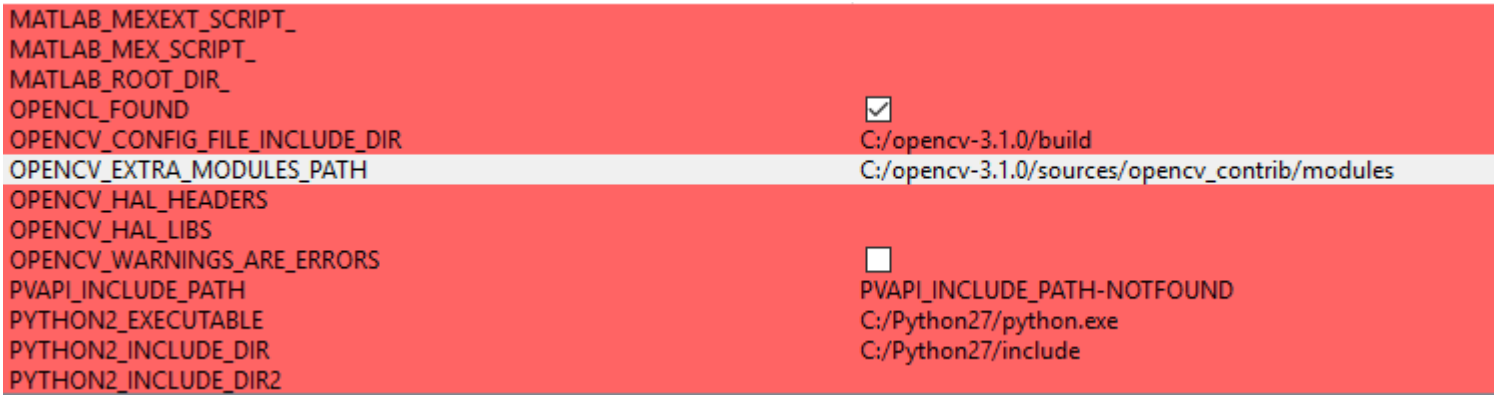
Les chemins seront quelque chose comme ceci: `C:/Program Files (x86)/Microsoft Visual Studio 12.0/VC/bin/cl.exe` . Indiquez votre chemin pour les champs C et C ++. Cliquez sur Terminer et attendez que la configuration soit terminée. Vous devriez obtenir zéro erreur si vous suiviez correctement toutes les étapes précédentes.



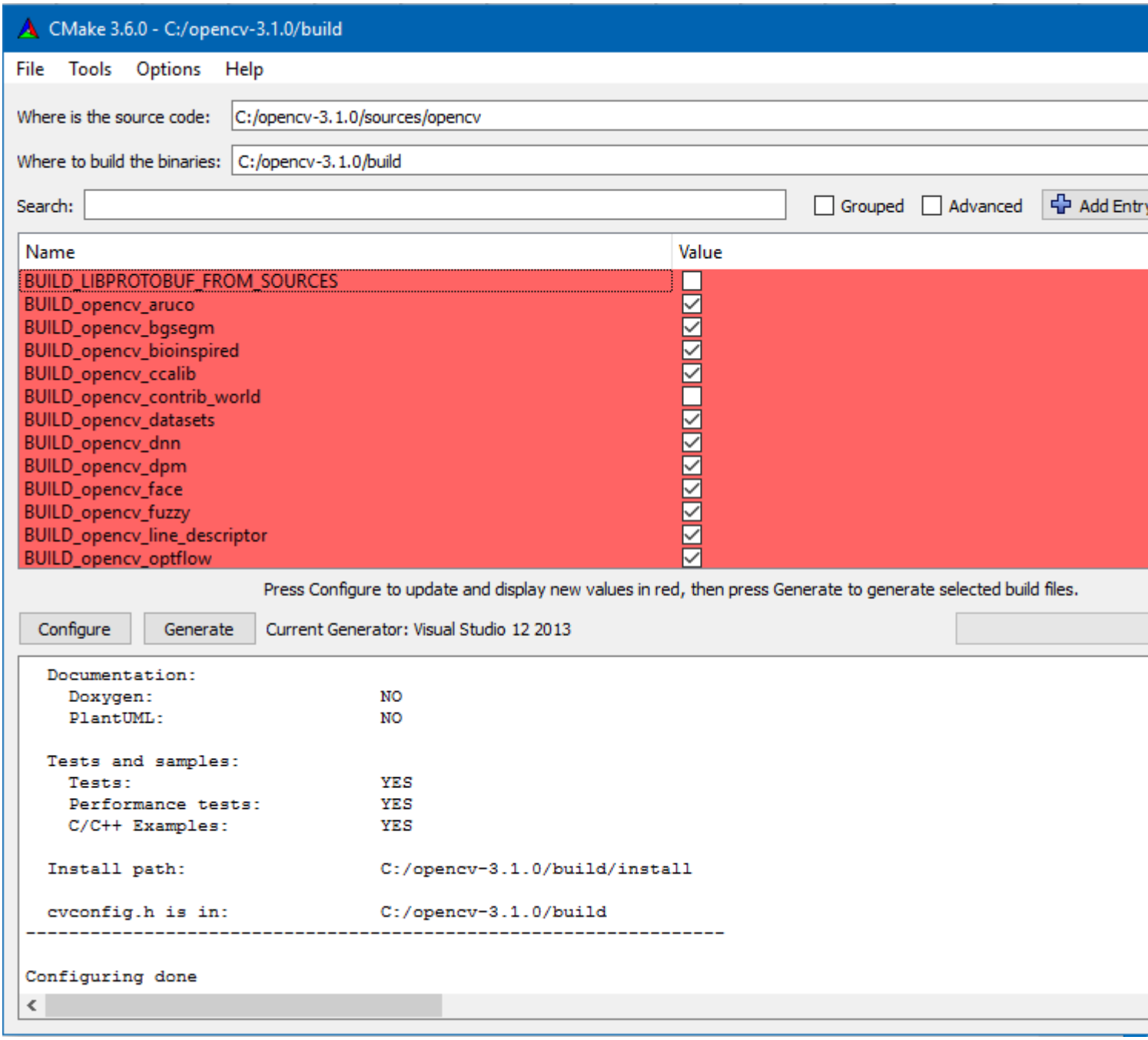

Une fois la configuration de CMake terminée, vous verrez apparaître de nouveaux éléments dans la fenêtre CMake qui sont mis en évidence en rouge. Ce sera quelque chose comme:




Vérifiez les builds dont vous avez besoin en cliquant sur la petite case carrée. Recherchez la ligne `OPENCV_EXTRA_MODULES_PATH` et indiquez le répertoire de modules dans opencv_contrib dans le répertoire des sources.



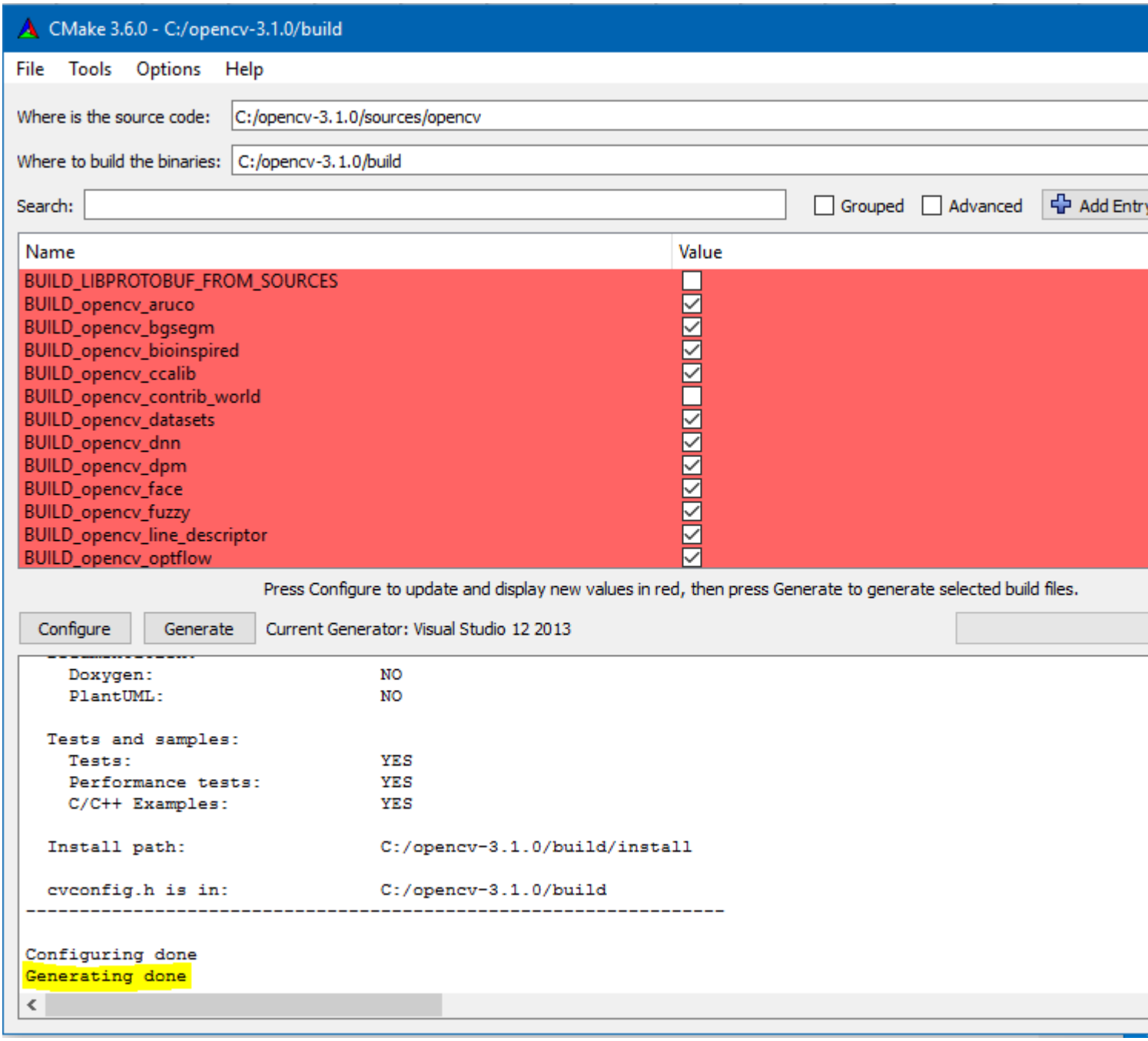
Une fois que vous avez terminé tout ce dont vous avez besoin et que le chemin des modules supplémentaires est activé, appuyez à nouveau sur configure pour mettre à jour. Les lignes précédemment surlignées ne seront plus mises en surbrillance et les nouveaux champs seront surlignés en rouge à la place.




Cochez également les cases pour tout ce que vous devez construire.

> Assurez-vous que ***BUILD_opencv_contrib_world*** et ***BUILD_opencv_world ne*** sont **pas *cochés*** . Il y a probablement un bogue où une erreur se produit quand l'un de ces derniers est vérifié.

À la fin de cette étape, cliquez sur **Générer** et vous aurez terminé avec CMake et vous pourrez le fermer. *S'il n'y a pas d'erreurs, vous obtiendrez un message à la fin du volet inférieur indiquant* ***Génération terminée*** .

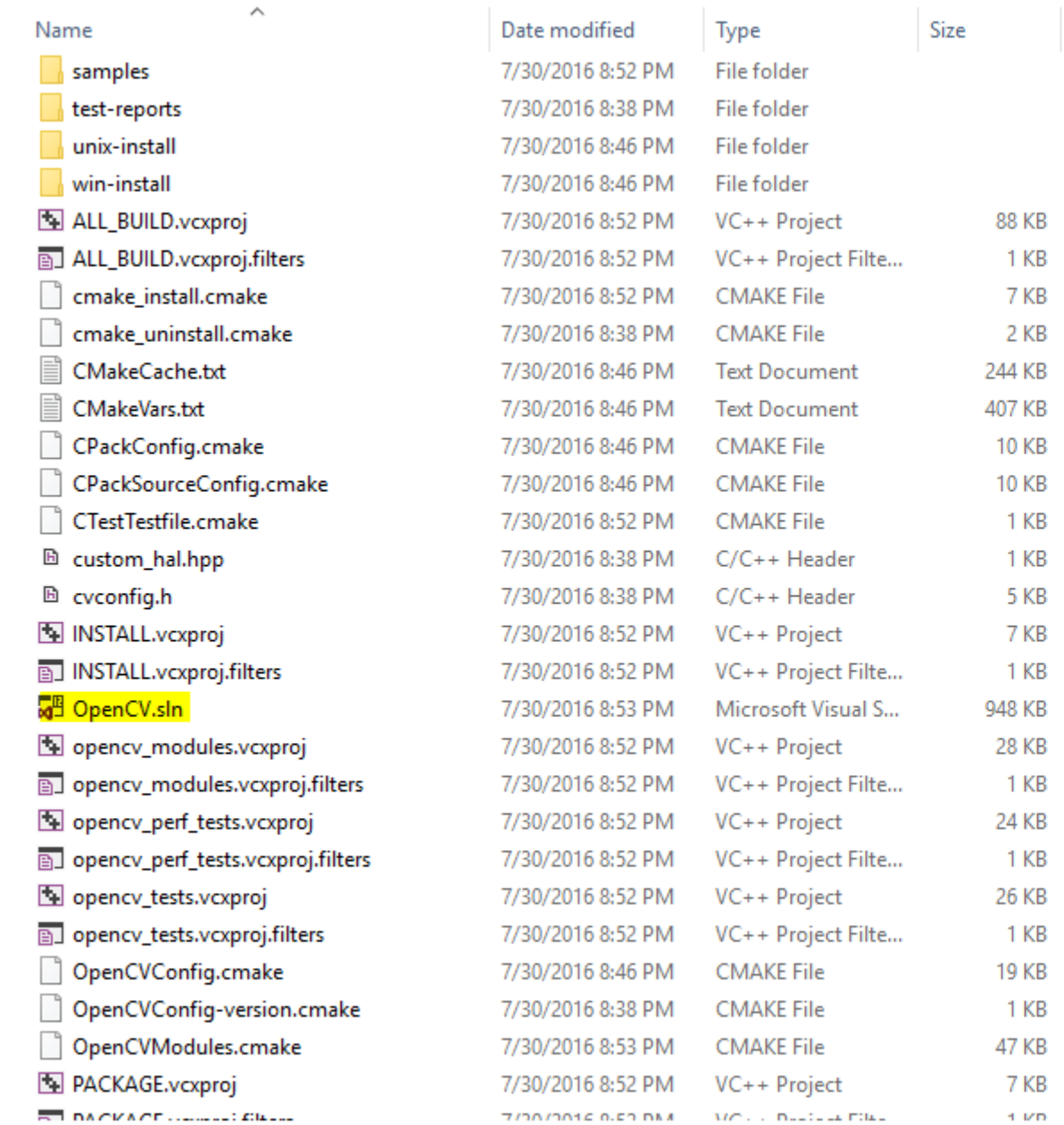



---

**Étape 4:**

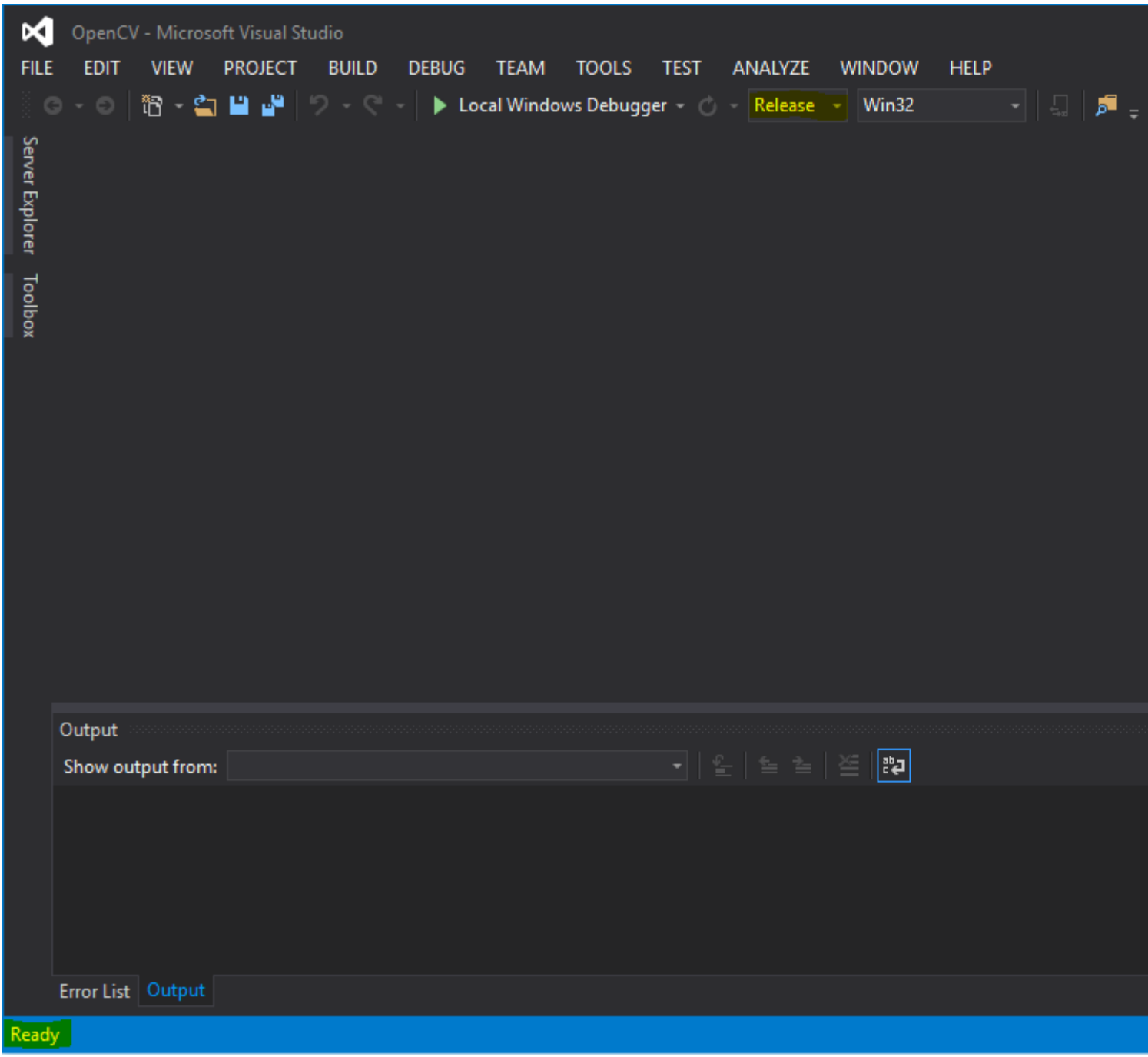
Ouvrez le répertoire de construction situé dans opencv-3.1.0 et vous y trouverez un tas de nouveaux dossiers et fichiers. C'était un dossier vide au début de ce processus.

Vous ne traitez que `OpenCV.sln` fichier `OpenCV.sln` et ne faites rien avec les fichiers restants. Ouvrez ce fichier avec la version utilisée lors de la compilation dans le CMake à l'étape précédente. Ce doit être `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 |

*Lorsque vous ouvrez le fichier .sln, soyez patient car il faut un certain temps pour tout préparer à la construction.* Lorsque **Prêt** est stable (ne change pas), vous pouvez commencer à construire vos cibles. Commencez à construire comme numéroté dans l'image ci-dessous. Assurez-vous également que la `Solution Configuration` la `Solution Configuration` est `Release` pas `Debug` .




---

**Étape 5:**

Lorsque la construction est terminée, vous devrez copier et coller quelques fichiers du répertoire de construction dans le répertoire `Python27` .

Recherchez le fichier `cv2.pyd` et copiez-le dans le répertoire `site-packages` de `Python27` . `cv2.pyd` devrait être présent dans `C:\opencv-3.1.0\build\lib\Release` . Après cela, copiez **seulement** les fichiers **.dll** dans `C:\opencv-3.1.0\build\bin\Release` dans le répertoire parent de `Python27` à cet emplacement `C:\Python27` .

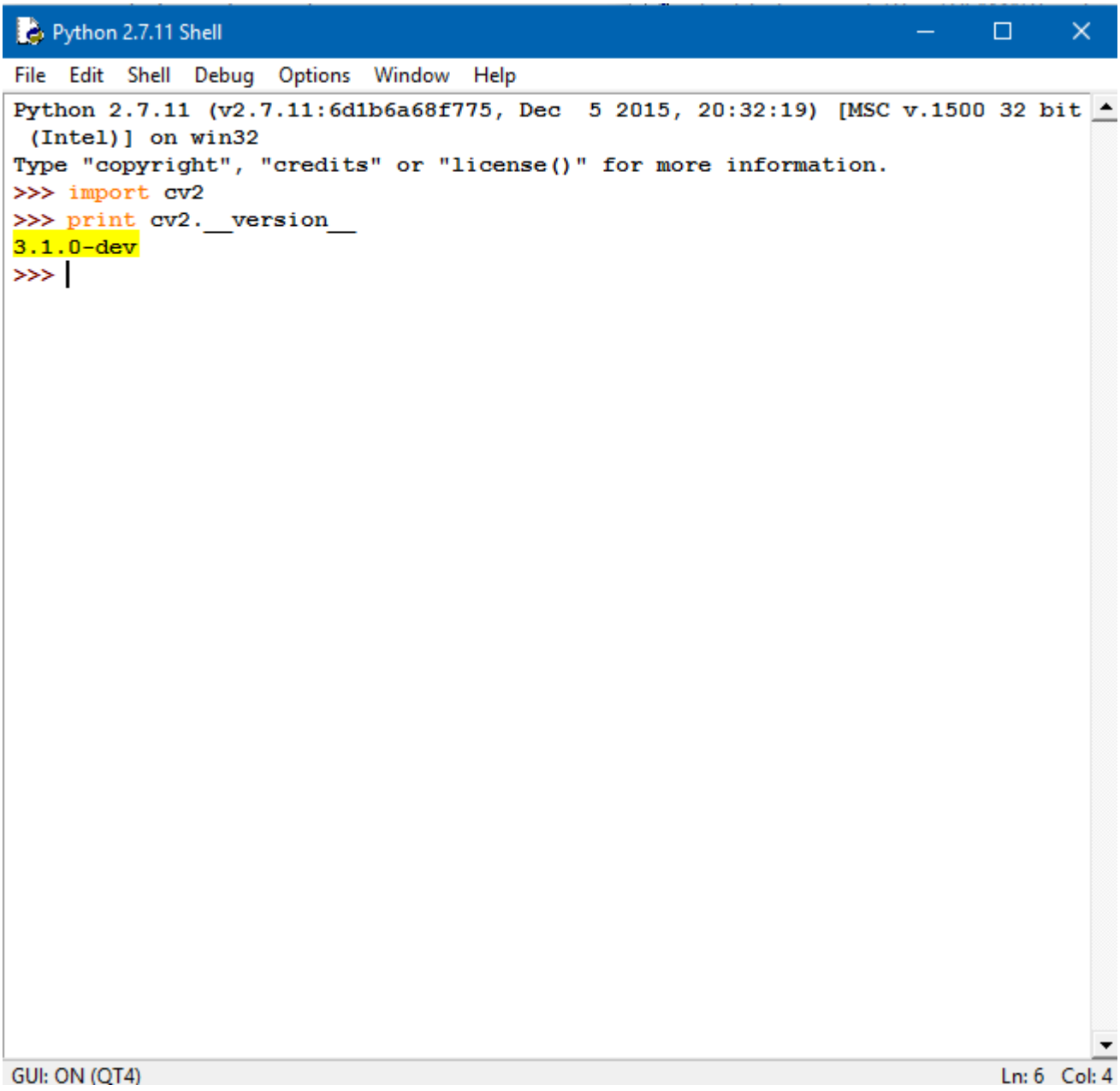
À la fin de cette étape, redémarrez votre PC.

---

**Vérification:**

Ouvrez IDLE et dans le type de shell Python:

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




# Examples

## Lecture de l'image et conversion en niveaux de gris

```
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()
```

Lire Construire et compiler opencv 3.1.0-dev pour Python2 sous Windows en utilisant CMake et Visual Studio en ligne: https://riptutorial.com/fr/opencv/topic/6100/construire-et-compiler-opencv-3-1-0-dev-pour-python2-sous-windows-en-utilisant-cmake-et-visual-studio

# Chapitre 7: Contraste et luminosité en C ++

## Syntaxe

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

## Paramètres

| Paramètre | Détails |
|---|---|
| m | matrice de sortie; s'il n'a pas une taille ou un type correct avant l'opération, il est réaffecté |
| type | le type de matrice de sortie souhaité ou plutôt la profondeur puisque le nombre de canaux est identique à celui de l'entrée; si rtype est négatif, la matrice de sortie aura le même type que l'entrée |
| alpha | facteur d'échelle facultatif. Cela change le contraste d'une image. Les valeurs inférieures à 1 diminuent le contraste et au dessus on augmente le contraste |
| bêta | delta facultatif ajouté aux valeurs mises à l'échelle. Les valeurs positives augmentent la luminosité et les valeurs négatives les diminuent |

## Remarques

**Contraste** :

Le contraste est la différence de luminance ou de couleur qui permet de distinguer un objet (ou sa représentation dans une image ou un affichage). Plus la différence entre un pixel et ses voisins est élevée, plus le contraste est élevé dans cette zone.

**Luminosité** :

En d'autres termes, la luminosité est la perception provoquée par la luminance d'une cible visuelle. En termes de pixels, plus la valeur d'un pixel est élevée, plus le pixel est lumineux.

**Réglages du contraste et de la luminosité:**

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

`f(x)` comme image source pixels et `g(x)` comme image de sortie pixels.

`i` et `j` indiquent que le pixel est situé dans la i-ème ligne et la j-ième colonne.

Les paramètres $\alpha > 0$ et $\beta$ sont souvent appelés paramètres de gain et de biais; On dit parfois que ces paramètres contrôlent respectivement le *contraste* et la *luminosité* .

Opencv a une fonction appelée convertTo () qui peut appliquer ces deux opérations.

Sources:
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

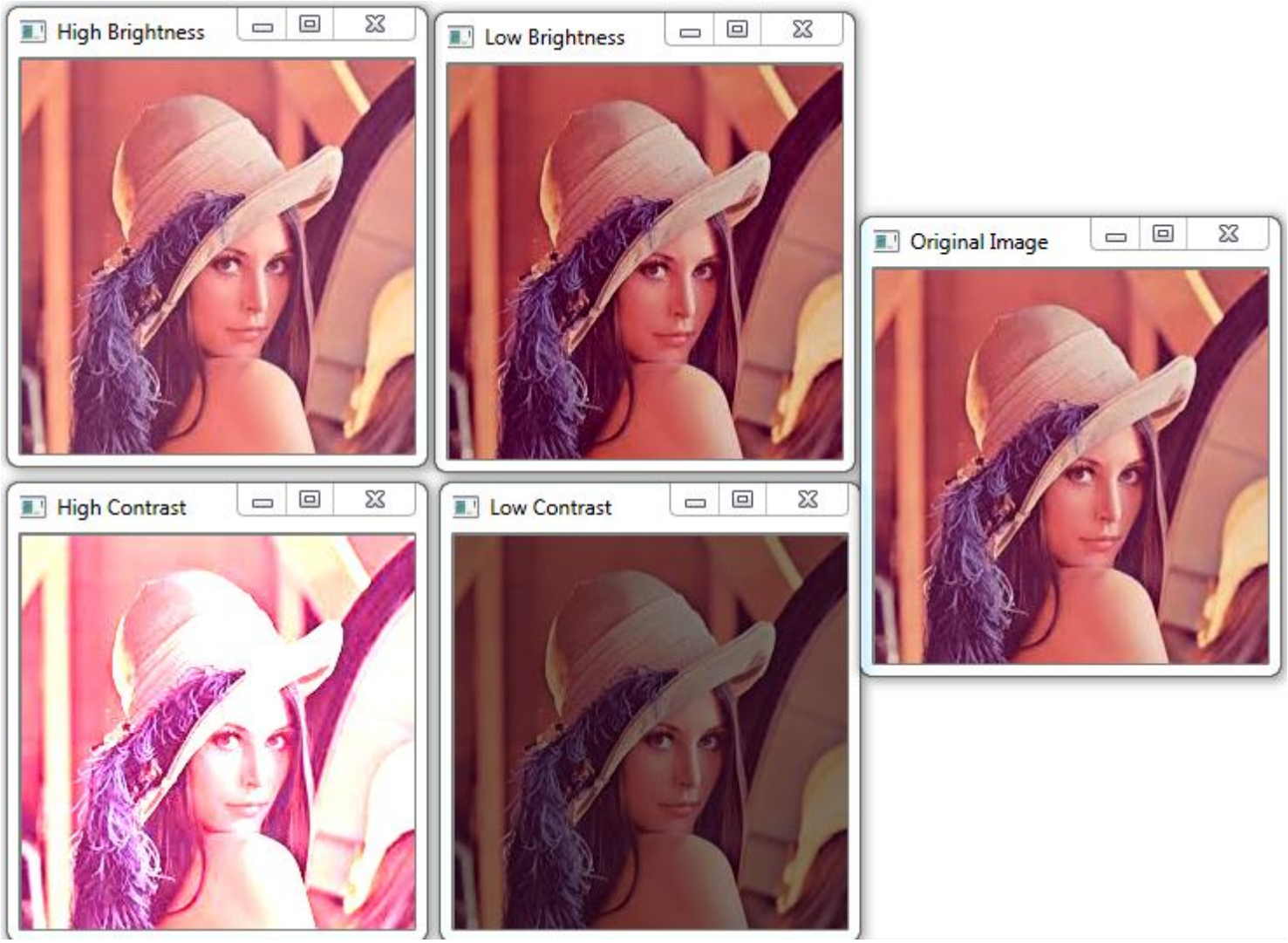
# Examples

## Réglage de la luminosité et du contraste d'une image 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;
}
```

Sortie du programme:




Lire Contraste et luminosité en C ++ en ligne:
https://riptutorial.com/fr/opencv/topic/6917/contraste-et-luminosite-en-c-plusplus

# Chapitre 8: Créer une vidéo

## Introduction

Chaque fois que vous travaillez avec des flux vidéo, vous pouvez éventuellement enregistrer votre résultat de traitement d'image sous la forme d'un nouveau fichier vidéo. Pour des sorties vidéo simples, vous pouvez utiliser la classe VideoWriter intégrée à OpenCV, conçue pour cela. Il est utile d'examiner certains concepts avant de les utiliser. Ces concepts sont codec, c'est-à-dire décodeur et FourCC (code à quatre caractères).

## Examples

### Créer une vidéo avec 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();
```

Lire Créer une vidéo en ligne: https://riptutorial.com/fr/opencv/topic/9196/creer-une-video

# Chapitre 9: Dessin de formes (ligne, cercle, ..., etc.) en C ++

## Introduction

Dans OpenCV, on peut dessiner de nombreuses formes telles que point, ligne, cercle, ..., etc. Il existe une option pour remplir une forme. Le code suivant est explicite et montre comment les formes sont dessinées.

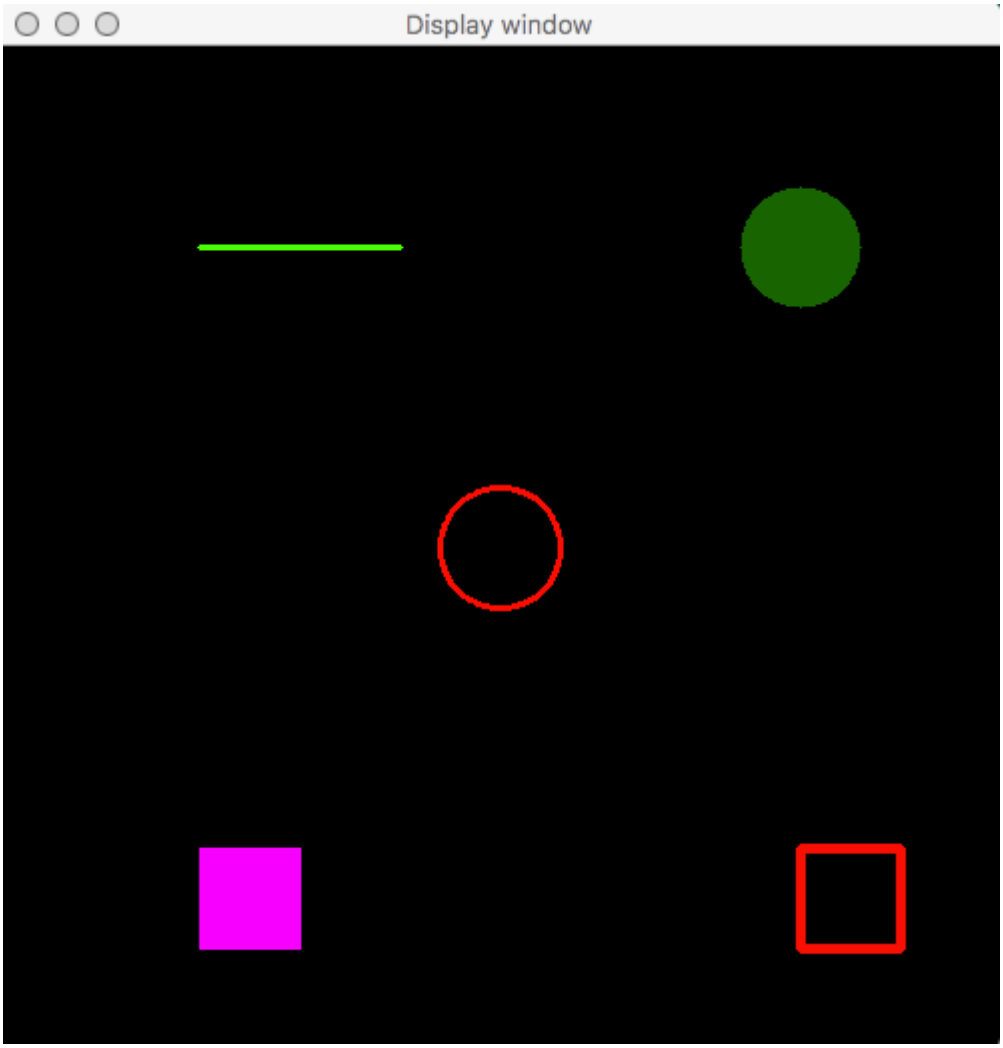
## Examples

### Exemple de formes de dessin

```
#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 sortie est



OpenCV 3.2 Mac avec compilateur g ++

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

Lire Dessin de formes (ligne, cercle, ..., etc.) en C ++ en ligne:
https://riptutorial.com/fr/opencv/topic/9749/dessin-de-formes--ligne--cercle-------etc---en-c-plusplus

# Chapitre 10: Détection d'objets

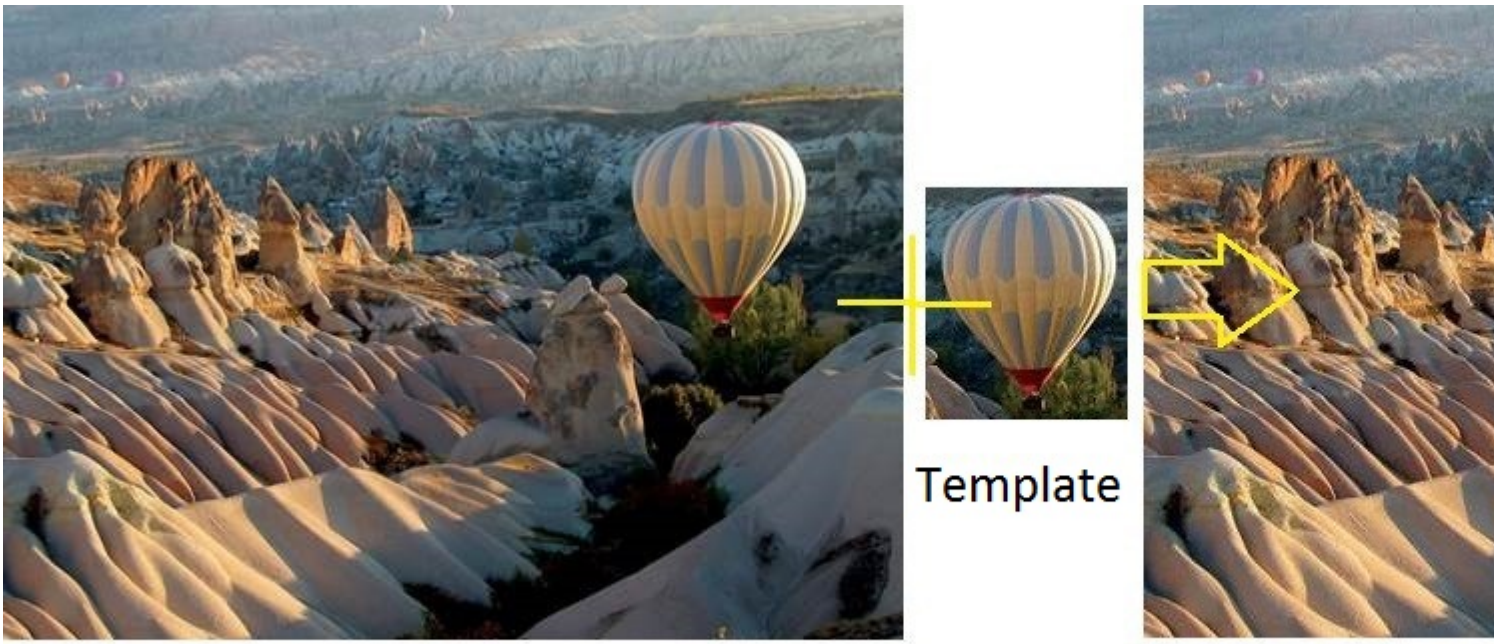
## Examples

### Correspondance de modèle avec Java

## Code source 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.");
    }

}
```

## RÉSULTAT




Resource Image

Result Im

Lire Détection d'objets en ligne: https://riptutorial.com/fr/opencv/topic/6735/detection-d-objets

# Chapitre 11: Détection de blob

## Examples

### Détection circulaire de blob

Cet exemple montre comment trouver des gouttes circulaires dans une image en niveaux de gris. L'évaluation de la circularité d'un blob se fait à l'aide de l'aire et du périmètre (longueur de l'arc) du contour. Le point central est évalué à l'aide des moments du contour.

```
#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;
}
```

Lire Détection de blob en ligne: https://riptutorial.com/fr/opencv/topic/6589/detection-de-blob

# Chapitre 12: Détection de bord

## Syntaxe

- bords = cv2. Canny (image, seuil1, seuil2 [, bords [, apertureSize [, L2gradient]]])
- void Canny (image InputArray, bords OutputArray, double threshold1, double threshold2, int apertureSize = 3, bool L2gradient = false

## Paramètres

| Paramètre | Détails |
|---|---|
| image | Image d'entrée |
| bords | Image de sortie |
| seuil1 | Premier seuil pour la procédure d'hystérésis |
| seuil2 | Deuxième seuil pour la procédure d'hystérésis |
| ouvertureTaille | Taille d'ouverture pour l'opérateur Sobel |
| L2gradient | Indicateur indiquant si un algorithme plus précis pour le dégradé d'image doit être utilisé |

## Examples

### Algorithme Canny

L'algorithme de Canny est un détecteur de bord plus récent conçu comme un problème de traitement du signal. Dans OpenCV, il génère une image binaire marquant les arêtes détectées.

**Python:**

```
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()
```

## Canny Algorithm - C ++

Vous trouverez ci-dessous une utilisation de l'algorithme canny en c ++. Notez que l'image est d'abord convertie en image en niveaux de gris, puis que le filtre gaussien est utilisé pour réduire le bruit dans l'image. Ensuite, l'algorithme de Canny est utilisé pour la détection des contours.

```
// 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;
}
```

## Calcul des seuils de Canny

Calcul automatique des seuils bas et haut pour l'opération Canny en ouverture

## Canny Edge Video 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()
```

## Prototypage de Canny Edge Thresholds à l'aide de trackbars

```
"""
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)
```

Lire Détection de bord en ligne: https://riptutorial.com/fr/opencv/topic/6099/detection-de-bord

# Chapitre 13: Fonctions de dessin en Java

## Examples

### Dessine un rectangle sur l'image

```
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");
    }
}
```

Lire Fonctions de dessin en Java en ligne: https://riptutorial.com/fr/opencv/topic/6153/fonctions-de-dessin-en-java

# Chapitre 14: Initialisation OpenCV sous Android

## Examples

### Initialisation asynchrone

L'initialisation asynchrone est une méthode recommandée pour le développement d'applications. Il utilise OpenCV Manager pour accéder aux bibliothèques OpenCV installées en externe dans le système cible.

Extrait de code implémentant l'initialisation asynchrone:

```
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);
    }

    ...
}
```

Dans ce cas, notre application fonctionne avec OpenCV Manager de manière asynchrone. `OnManagerConnected` rappel `OnManagerConnected` sera appelé dans le thread d'interface utilisateur lorsque l'initialisation sera terminée.

> Veuillez noter qu'il est interdit d'utiliser des appels OpenCV ou de charger des libs natives dépendantes d'OpenCV avant d'appeler ce rappel. Chargez vos propres bibliothèques natives qui dépendent d'OpenCV après l'initialisation réussie d'OpenCV.

Par défaut `BaseLoaderCallback` mise en œuvre gâterie contexte d'application comme `Activity` et demande `Activity.finish()` méthode de sortie en cas d'échec d'initialisation. Pour remplacer ce comportement, vous devez remplacer la méthode `finish()` de la classe `BaseLoaderCallback` et implémenter votre propre méthode de finalisation.

# OpenCV Manager

OpenCV Manager est un service Android destiné à gérer les fichiers binaires de bibliothèque OpenCV sur les périphériques des utilisateurs finaux. Il permet de partager les bibliothèques dynamiques OpenCV entre les applications sur le même périphérique.

Le gestionnaire offre les avantages suivants:

- Moins d'utilisation de la mémoire (environ 40 Mo). Toutes les applications utilisent les mêmes fichiers binaires du service et ne conservent pas les bibliothèques natives à l'intérieur.
- Optimisations spécifiques au matériel pour toutes les plates-formes prises en charge.
- Source de bibliothèque OpenCV fiable. Tous les packages avec OpenCV sont publiés sur le marché Google Play.
- Mises à jour régulières et corrections de bugs.

Le seul inconvénient est que l'utilisateur est invité à télécharger et à appliquer une application supplémentaire, ce qui réduit légèrement l'expérience utilisateur.

Plus d'infos: Android OpenCV Manager

> **Mis à jour 18/10/16:**
> Il y a un bogue dans la version d'OpenCV Manager distribuée sur Play Store (mise à jour le 21/09/15).
> Cela n'affecte que la version OpenCV 3.1.0. Lorsque vous exécutez certaines fonctions OpenCV, vous obtenez une erreur `SIGSEGV` . La version distribuée avec Android SDK fonctionne `OpenCV-android-sdk/apk/OpenCV_3.1.0_Manager_3.10_{platform}.apk` ( `OpenCV-android-sdk/apk/OpenCV_3.1.0_Manager_3.10_{platform}.apk` ). Il peut être téléchargé à partir du site Web OpenCV .
> Plus d'infos: Numéro 6247 .

## Initialisation statique

Selon cette approche, tous les fichiers binaires OpenCV sont inclus dans votre package

d'application. Il est conçu principalement à des fins de développement et de débogage. Cette approche est **déconseillée** pour le code de production, l'initialisation asynchrone est recommandée.

Si votre projet d'application ne comporte pas de partie JNI, copiez simplement les librairies natives OpenCV correspondantes d' `OpenCV-3.1.0-android-sdk/sdk/native/libs` dans le répertoire de votre projet dans le dossier `app/src/main/jniLibs` .

Dans le cas du projet d'application avec une partie JNI, au lieu de copier manuellement des bibliothèques, vous devez modifier votre fichier `Android.mk` : ajoutez les deux lignes de code suivantes après `"include $(CLEAR_VARS)"` et avant `"include path_to_OpenCV-3.1.0-android-sdk/sdk/native/jni/OpenCV.mk"` :

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

Le résultat devrait ressembler à ceci:

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

Après cela, les bibliothèques OpenCV seront copiées dans le dossier `jniLibs` de votre application lors de la construction de JNI.

La dernière étape de l'activation d'OpenCV dans votre application est le code d'initialisation Java avant d'appeler l'API OpenCV. Cela peut être fait, par exemple, dans la section statique de la classe Activité:

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

Si votre application inclut d'autres bibliothèques natives dépendantes d'OpenCV, vous devez les charger après l'initialisation d'OpenCV:

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

> **Remarque:** `initDebug()` méthode `initDebug()` est obsolète pour le code de production. Il est conçu à des fins de développement expérimental et local uniquement. Si vous

souhaitez publier votre approche d'utilisation de l'application avec l'initialisation asynchrone.

Lire Initialisation OpenCV sous Android en ligne:
https://riptutorial.com/fr/opencv/topic/7545/initialisation-opencv-sous-android

# Chapitre 15: Installation OpenCV

## Introduction

Installation d'OpenCV sous Linux, Mac OS et Windows

## Examples

### Installation d'OpenCV sur Ubuntu

Lien source

Ouvrez le terminal et écrivez les commandes suivantes.

1-Mettre à jour et mettre à jour le paquet de votre système 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-Installation de Dependenices:

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

3-Build Tools pour OpenCV Code source:

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

Bibliothèques d'E / S à 4 images pour OpenCV:

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

Bibliothèques d'E / S 5 vidéos pour 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-Bibliothèques de parallélisme et d'algèbre linéaire:

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

Bibliothèques d'interface utilisateur à 7 graphiques:

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

sudo apt-get install libvtk6-dev
```

8 – Installation Java:

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

Installation en 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-Télécharger le code source d'OPENCV depuis Github:

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

Fichier Zip OPENCV à décompression 11:

```
unzip 3.2.0.zip
```

12-Supprimer le fichier 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-Terminé Vérifiez votre numéro de version OpenCV:

```
pkg-config --modversion opencv

pkg-config --cflags opencv
```

Lire Installation OpenCV en ligne: https://riptutorial.com/fr/opencv/topic/8934/installation-opencv

# Chapitre 16: Modification du contenu de l'image

## Examples

### Définir l'image entière sur une couleur unie

Étant donné une taille de `cv::Mat img` non vide, nous pouvons la remplir de plusieurs manières:

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

ou, plus général, mask support, `cv::Mat::setTo()` :

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

Si vous utilisez l'ancienne API OpenCV C avec `IplImage* img` :

Utilisation:

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

### Modification des pixels par pixel des images

Dans OpenCV, les images peuvent être RVB / BGR, HSV, niveaux de gris, noir et blanc, etc. Il est essentiel de connaître le type de données avant de traiter les images.

Les types de données d'image sont principalement `CV_8UC3` (matrice de uchar à 3 canaux) et CV_8U (matrice de uchar à 1 canal), mais la conversion à d'autres types tels que CV_32FC3, CV_64F est également possible. (voir types de données )

Considérez que l'image est une image RVB qui est lue par la fonction de `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;
```

De même, si l'image est en niveaux de gris,

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

Notez que, dans OpenCV, les images en noir et blanc sont stockées en tant que type CV_8U avec les valeurs 0 et 255. Par conséquent, la modification des images BW est identique à celle des images grises.

## Modification de la couleur de l’image dans OpenCV - kmeans (). Pour analyser tous les pixels d'une image et remplacer les valeurs de pixels par des couleurs génériques.

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

```
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;
```

}

Lire Modification du contenu de l'image en ligne:
https://riptutorial.com/fr/opencv/topic/6307/modification-du-contenu-de-l-image

# Chapitre 17: Structures de base

## Introduction

Cette rubrique couvre les structures de base dans OpenCV. Les structures qui seront abordées dans cette rubrique sont `DataType` , `Point` , `Vec` , `Size` , `Rect` , `Scalar` , `Ptr` et `Mat` .

## Examples

### Type de données

Les types primitifs dans OpenCV sont `unsigned char, bool, signed char, unsigned short, signed short, int, float, double` . Tout type de données dans OpenCV est défini comme `CV_<bit-depth>{U|S|F}C(<number_of_channels>)` où `U: unsigned` , `S:signed` et `F:floating point` .

Par exemple, `CV_32FC2` est une `CV_32FC2` 32 bits, à virgule flottante et 2 canaux. et la définition de base, les types de canal sont

```
#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
```

Les autres types avec un canal plus élevé sont produits à partir de ceux-ci par la définition suivante:

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

En utilisant ces types de données, d'autres structures peuvent être créées.

### Tapis

`Mat` (Matrix) est un tableau à n dimensions qui peut être utilisé pour stocker différents types de données, telles que des images RVB, HSV ou en niveaux de gris, des vecteurs avec des valeurs réelles ou complexes, d'autres matrices, etc.

Un `Mat` contient les informations suivantes: `width` , `height` , `type` , `channels` , `data` , `flags` , `datastart` , `dataend` , etc.

Il a plusieurs méthodes, certaines sont: `create` , `copyTo` , `convertTo` , `isContinious` etc.

Il existe plusieurs façons de créer une variable Mat. Considérez que je veux créer une matrice avec 100 lignes, 200 colonnes, tapez 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
```

Tapis d'initialisation:

```
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) est une classe de modèle pour les valeurs numériques. Contrairement au `c++ vector` , il stocke généralement des vecteurs courts (quelques éléments seulement).

La façon dont un `Vec` est défini est la suivante:

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

où type est l'un des `uchar, short, int, float, double` et les caractères de chaque type sont respectivement `b, s, i, f, d` .

Par exemple, `Vec3b` indique un vecteur char non signé de 3 canaux. Chaque index d'une image RVB est dans ce format.

```
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].
```

Dans la classe `Vec` , les opérateurs suivants sont définis

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

Pour plus d'informations, voir le lien

Lire Structures de base en ligne: https://riptutorial.com/fr/opencv/topic/9099/structures-de-base

# Chapitre 18: Traitement d'image

## Syntaxe

1. **Syntaxe du flou gaussien C ++:** GaussianBlur vide (InputArray src, OutputArray dst, Taille ksize, double sigmaX, double sigmaY = 0, int borderType = BORDER_DEFAULT)

## Paramètres

| Paramètres du flou gaussien | Détails |
|---|---|
| src | Image d'entrée, l'image peut avoir n'importe quel nombre de canaux, qui sont traités indépendamment, mais la profondeur doit être `CV_8U` , `CV_16U` , `CV_16S` , `CV_32F` ou `CV_64F` . |
| dst | Image de sortie de la même taille et du même type que `src` |
| ksize | Taille du noyau gaussien. `ksize.width` et `ksize.height` peuvent différer mais ils doivent tous deux être **positifs et impairs** . Ou, ils peuvent être zéro et ils sont calculés à partir de sigma *. |
| SigmaX | Écart type du noyau gaussien dans la **direction X** |
| sigmaY | Écart type du noyau gaussien dans la **direction Y** Si `sigmaY` est à zéro, il est défini pour être égal à `sigmaX` , si les deux sigmas sont des zéros, ils sont calculés à partir de `ksize.width` et de `ksize.height` . Pour contrôler complètement le résultat indépendamment des modifications futures possibles de toute cette sémantique, il est recommandé de spécifier tous les `ksize` , `sigmaX` et `sigmaY` . |
| borderType | Méthode d'extrapolation de pixels. |

## Remarques

Je ne pense pas qu'il soit logique de mettre la syntaxe et les paramètres spécifiques au flou gaussien dans ce lieu, car le sujet est tellement vaste qu'il devrait inclure de nombreux autres exemples.

## Examples

### Lissage d'images avec flou gaussien en C ++

Le lissage, également appelé **flou** , est l'une des opérations les plus couramment utilisées dans le traitement d'images.

L'utilisation la plus courante de l'opération de lissage consiste à **réduire le bruit** dans l'image pour un traitement ultérieur.

Il existe de nombreux algorithmes pour effectuer le lissage.

Nous allons examiner l'un des filtres les plus couramment utilisés pour brouiller une image, le **filtre gaussien** utilisant la fonction de bibliothèque OpenCV `GaussianBlur()` . Ce filtre est conçu spécifiquement pour supprimer *le bruit haute fréquence* des images.

```
#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;
}
```

Pour la définition mathématique détaillée et d'autres types de filtres, vous pouvez vérifier la documentation d'origine .

## Seuillage

**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()
```

## Filtrage Bilatéral

Dans les applications de traitement d'image, les filtres bilatéraux constituent un type particulier de **filtres non linéaires** .

Il y a un compromis entre la perte de structure et la suppression du bruit, car la méthode la plus populaire pour éliminer le bruit est le flou gaussien qui ne connaît pas la structure de l'image; par conséquent, il supprime également les bords. La plupart du temps, les bords contiennent des

informations précieuses sur la scène et nous ne voulons pas les perdre. Le **filtre bilatéral** connaît la structure de la scène et a tendance à agir comme un filtre de flou classique lorsqu'il se trouve sur une zone sans arêtes. cependant, quand il voit un bord, il change de comportement; de sorte que le flou ne fonctionne pas sur tous les bords, mais fonctionne le long des bords, ce qui signifie qu'ils sont **des filtres préservant** les bords.

```
#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);
}
```

Lire Traitement d'image en ligne: https://riptutorial.com/fr/opencv/topic/2032/traitement-d-image

# Chapitre 19: Utilisation de classificateurs en cascade en Java

## Syntaxe

- CascadeClassifier cascade = new CascadeClassifier ("cascade.xml"); // Crée un classificateur en cascade à partir de cascade.xml
- Mat image = Imgcodecs.imread ("image.png"); // Convertit image.png en un objet Mat (Matrix)
- Détections MatOfRect = new MatOfRect (); // Crée un fichier MatOfRect (Matrix of Rectangles) vide, utilisé comme sortie pour nos classes de détection
- detections.toArray (); // Retourne un tableau d'objets Rect pouvant être itéré sur
- Imgproc.rectangle (image, nouveau point (rect.x, rect.y), nouveau point (rect.x + rect.width, rect.y + rect.height), nouveau scalaire (0, 255, 0)); // Dessine un rectangle à contour vert depuis les emplacements x et y du premier point vers l'emplacement x et y du second point sur l'objet Mat "image". "rect" est un objet Rect, généralement fourni par detections.toArray (). Utilise la classe de points OpenCV.
- Imgcodecs.imwrite ("output.png", image); // Ecrit l'objet Mat modifié "image" dans le fichier "output.png"
- CascadeClassifier.detectMultiScale (image, détections); // Détecte tout objet dans l'objet Mat "image" et affiche les détections dans l'objet MatOfRect "détections"
- CascadeClassifier.detectMultiScale (image, détections, **scaleFactor** , **minNeighbors** , **flags** , **minSize** , **maxSize** ); // Effectue une détection avec des paramètres supplémentaires. Voir les détails ci-dessous.
- Imgproc.ellipse (image, centre, **axes** , 0, 0, 360, nouveau Scalar (255, 0, 255), **épaisseur** , **lineType** , 0); // Dessine une ellipse sur l'image au `center` du point. Utilise la classe de points OpenCV.

## Paramètres

| Paramètre | Détails |
|---|---|
| facteur d'échelle | Combien la taille de l'image est réduite à chaque échelle d'image. Par défaut = `1.1` |
| minNeighbors | Combien de voisins un rectangle candidat doit avoir avant de le sélectionner comme objet détecté. Par défaut = `4` |
| drapeaux | Drapeaux hérités Dans la plupart des cas, cela devrait être mis à `0` . Par défaut = `0` |
| minSize | La taille minimale d'un rectangle candidat peut être. Cela utilise la classe de `Size` OpenCV. Peut être utilisé pour réduire le temps de détection et l'utilisation du processeur, ainsi que pour réduire les faux positifs. |

| Paramètre | Détails |
|---|---|
| taille max | Taille maximale que peut contenir un rectangle candidat. Cela utilise la classe de `Size` OpenCV. Peut être utilisé pour réduire le temps de détection et l'utilisation du processeur, ainsi que pour réduire les faux positifs. |
| les axes | Utilise la classe de taille OpenCV. Définit la largeur et la hauteur de l'ellipse. |
| épaisseur | Épaisseur de la ligne, en pixels. |
| type de ligne | A divers paramètres. `0` correspond à la ligne `CV_AA` , `8` ligne à 8 connexions, `4` ligne à 4 connexions et `CV_AA` à la ligne antialiasée. Par défaut = `8` |

# Examples

### Obtenir une image statique, détecter les éléments et afficher les résultats.

Veuillez noter que cet exemple utilise 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);

    }
}
```

Le `Rect[]` renvoyé par `diceDetections.toArray()` peut être itéré sur. Chaque `Rect` à l'intérieur du tableau aura quatre propriétés principales: `x` , `y` , `width` et `height` . `x` et `y` définissent la position supérieure gauche du rectangle et `width` et `height` renvoie un `int` de la largeur et de la hauteur du rectangle. Ceci est utilisé pour dessiner des rectangles sur des images. Les paramètres requis minimaux de la fonction `Imgproc.rectangle` sont les suivants:

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

Les deux `Point` sont utilisés pour les positions du coin supérieur gauche et du coin inférieur droit. Ces positions sont *à la fois absolues* sur l'image fournie comme premier paramètre et non sur l'autre. Vous devez donc ajouter la position `x` ou `y` du rectangle en plus de la `width` ou de la `height` pour définir correctement le point de `end` .

Notez que la classe `Point` utilisée dans ces paramètres n'est **pas** la classe `Point` la bibliothèque standard de Java. Vous devez importer la classe de `Point` OpenCV à la place!

## Détection d'images à partir d'un périphérique vidéo

Cet exemple présente la classe `VideoCapture` , où nous l'utilisons pour prendre une image à partir d'une webcam et l'enregistrer sur une image.

```
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.

    }
}
```

## Conversion d'un objet Mat en objet BufferedImage

Cet exemple de Daniel Baggio a été tiré directement de cette réponse StackExchange , mais a été republié pour la visibilité.

Cette classe prend un objet Mat et renvoie l'objet BufferedImage utilisé par les bibliothèques `javax.swing` . Cela peut être utilisé par un objet `Graphics` pour dessiner l'image.

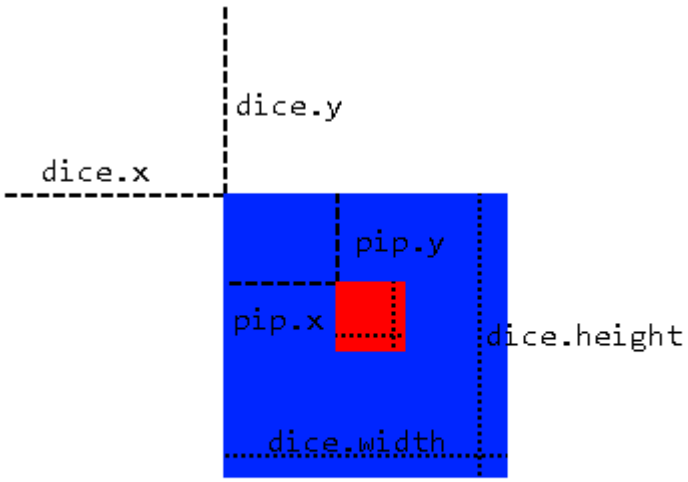
```
private BufferedImage toBufferedImage(Mat m) {
    if (!m.empty()) {
        int type = BufferedImage.TYPE_BYTE_GRAY;
        if (m.channels() > 1) {
            type = BufferedImage.TYPE_3BYTE_BGR;
        }
        int bufferSize = m.channels() * m.cols() * m.rows();
        byte[] b = new byte[bufferSize];
        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;
}
```

## Détections dans les détections

Cet exemple utilise Dice et les points noirs sur les dés (les pips) comme objet. Comme l'exemple est assez long, expliquer d'abord certains concepts clés est essentiel pour comprendre l'exemple.

Comprendre le premier exemple, "Obtention d'une image statique, détection des éléments et sortie des résultats". est essentiel pour comprendre cet exemple, en particulier comment OpenCV dessine des rectangles.

Regardez l'image suivante:




Nous utiliserons la méthode de sous-application, où nous utiliserons une zone détectée comme base pour appliquer davantage de détections. Ceci n'est possible que si un objet se trouve toujours dans un autre objet que nous pouvons détecter, comme nos pips sur nos dés. Cette méthode présente plusieurs avantages:

- Au lieu de scanner l'image entière, il suffit de scanner la zone dans laquelle nous savons que l'objet se trouvera.
- Supprime toute possibilité de faux positifs en dehors de la zone de détection.

Nous faisons cela en appliquant d'abord un balayage classificateur en cascade sur l'image entière pour nous donner un objet `MatOfRect` contenant nos grands objets (dés, dans ce cas). Nous parcourons ensuite le tableau `Rect[]` donné par la fonction `toArray()` de l'objet `MatOfRect` . Cet objet `Rect` est utilisé pour créer un objet `Mat` temporaire qui est "recadré" aux propriétés de l'objet `Rect` ( `x, y, width, height` ) à partir de l'image d'origine, où nous pouvons alors effectuer des détections sur l'objet `Mat` temporaire. En d'autres termes, nous demandons au classificateur de ne faire que des détections sur les parties de dés de l'image, et nous spécifions la position de chaque dé en utilisant les objets `Rect` obtenus en effectuant une détection sur l'image entière.

Cependant, les objets `Rect` (pips) ont leurs propriétés par rapport à leurs dés, et non à l'image elle-même. Pour résoudre ce problème, lorsque nous voulons dessiner des rectangles à l'image réelle montrant l'emplacement des pépins, nous ajoutons à la fois `dice.x` et `dice.y` au `Point` départ.

```
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 fonction `getNextImage()` renvoie un objet `Mat` , qui, associé aux autres exemples publiés, peut

être appelé en permanence et peut être converti en `BufferImage` pour fournir un flux de diffusion affichant les détections.

Lire Utilisation de classificateurs en cascade en Java en ligne:
https://riptutorial.com/fr/opencv/topic/6377/utilisation-de-classificateurs-en-cascade-en-java

# Chapitre 20: Utiliser VideoCapture avec OpenCV Python

## Examples

### Lecture d'images d'une vidéo pré-capturée

```
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
```

### Utiliser VideoCapture avec OpenCV Java

Il n'y a pas de solution dans Java, vous devez écrire une méthode pour cela. Cette méthode est un Mat2bufferedImage. Prend l'objet mat comme paramètre et renvoie l'image.

```
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();

        }
    }
}
```

Lire Utiliser VideoCapture avec OpenCV Python en ligne:
https://riptutorial.com/fr/opencv/topic/6803/utiliser-videocapture-avec-opencv-python

# Crédits

| S. No | Chapitres | Contributeurs |
|---|---|---|
| 1 | Commencer avec 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 | Accès aux pixels | Adi Shavit, brian, cagatayodabasi, Ehsan Ab, Elizabeth, smttsp, Sunreef |
| 3 | Afficher l'image OpenCV | Aleksandar, Elizabeth, jlarsch, mesutpiskin, smttsp |
| 4 | Chargement et enregistrement de différents formats de supports | Adi Shavit, cagatayodabasi, Jav_Rock, Lakshya Kejriwal, MD. Nazmul Kibria |
| 5 | Classificateurs en cascade | Arijit, MD. Nazmul Kibria, mesutpiskin |
| 6 | Construire et compiler opencv 3.1.0-dev pour Python2 sous Windows en utilisant CMake et Visual Studio | Tes3awy |
| 7 | Contraste et luminosité en C ++ | Ehsan Ab, MD. Nazmul Kibria |
| 8 | Créer une vidéo | mesutpiskin |
| 9 | Dessin de formes (ligne, cercle, ..., etc.) en C ++ | CroCo |
| 10 | Détection d'objets | K D, mesutpiskin |
| 11 | Détection de blob | MD. Nazmul Kibria, Sebastian |
| 12 | Détection de bord | cmastudios, Ehsan Ab, K D, m3h0w, Sounak |
| 13 | Fonctions de dessin | mesutpiskin |

| | | |
|---|---|---|
| | en Java | |
| 14 | Initialisation OpenCV sous Android | David Miguel |
| 15 | Installation OpenCV | amorenew |
| 16 | Modification du contenu de l'image | Adi Shavit, DivyaMaheswaran, smttsp |
| 17 | Structures de base | smttsp |
| 18 | Traitement d'image | cagatayodabasi, Dan Mašek, Elizabeth, jlarsch, Shubham Batra , Sunreef, Utkarsh Sinha |
| 19 | Utilisation de classificateurs en cascade en Java | Edward Shen |
| 20 | Utiliser VideoCapture avec OpenCV Python | jlarsch, mesutpiskin |