OpenCV Tutorial C++

Capture Video from File or Camera

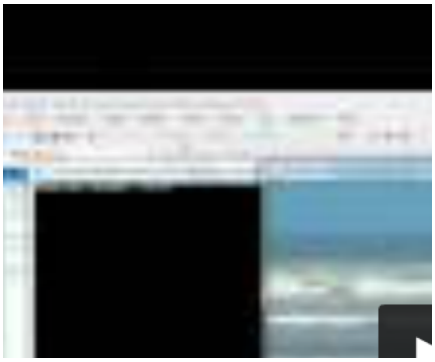
Capture Video From File

In this lesson, I am going to show you how to read a video file. It's fairly simple. You just need to initialize a VideoCapture object which will open a video file and read frame by frame from that opened video. Here is the sample OpenCV code. If you are using Visual Studio, you will need to include "stdafx.h" header file.

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

using namespace cv;
using namespace std;

int main(int argc, char* argv[])
{
   VideoCapture cap("C:/Users/SHERMAL/Desktop/SampleVideo.avi"); // open the video file for reading

   if ( !cap.isOpened() )  // if not success, exit program
   {
        cout << "Cannot open the video file" << endl;
        return -1;
   }

   //cap.set(CV_CAP_PROP_POS_MSEC, 300); //start the video at 300ms

   double fps = cap.get(CV_CAP_PROP_FPS); //get the frames per seconds of the video

    cout << "Frame per seconds : " << fps << endl;

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

   while(1)
   {
      Mat frame;

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

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

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

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

   return 0;

}
////////////////////////////////////////////////////////////////////////////////////////
```

You can download this OpenCV visual c++ project from here. (The downloaded file is a compressed .rar folder. So, you have to extract it using Winrar or other suitable software)

This is how I play a video using my OpenCV program



New OpenCV functions which are not found earlier are explained here

- **VideoCapture::VideoCapture(const string& filename)**

This is one of few constructors available in VideoCapture class. This constructor open the video file and initializes the VideoCapture object for reading the video stream from the specified file.
The destructor of this class will deallocated any associated memory with this object. Therefore you don't need to deallocate memory explicitly in your program.

- **bool VideoCapture::IsOpened()**

If the previous call to VideoCapture constructor is successful, this method will return true. Otherwise it will return false.
It is essential to check that whether the VideoCapture initialize successfully. If it is unsuccessful, program should be exited. Otherwise when you try to read a frame from the VideoCapture object, your program will crash.

- **bool VideoCapture::set(int propId, double value)**

You can change some properties of VideoCapture object. If it is successful, this method will return true. Otherwise it will return false. You should try to change some properties of the video stream in your code. In my code, I have shown you how to change the position of the video, by changing the **CV_CAP_PROP_POS_MSEC** property.

Parameters :

  - **int propID -** This argument specify the property you are going to change. There are many options for this argument. Some of them are listed here.
    - **CV_CAP_PROP_POS_MSEC -** current position of the video in milliseconds
    - **CV_CAP_PROP_POS_FRAMES -** current position of the video in frames
    - **CV_CAP_PROP_FRAME_WIDTH -** width of the frame of the video stream
    - **CV_CAP_PROP_FRAME_HEIGHT -** height of the frame of the video stream
    - **CV_CAP_PROP_FPS -** frame rate (frames per second)
    - **CV_CAP_PROP_FOURCC** - four character code of codec
  - **double value -** This is the new value you are going to assign to the property, specified by the propID

- **doubleVideoCapture::get(int propId)**

This function returns the value of the property which is specified by **propId.** You should try to obtain some properties of the video stream in your code. In my code, I have shown you how to obtain the frame rate (frames per second) of the video, by using the **CV_CAP_PROP_FPS** argument.

Parameters -

  - **int propID -** This argument specify the property you are going to obtain. There are many options for this argument. Some of them are listed here.
    - **CV_CAP_PROP_POS_MSEC -** current position of the video in milliseconds
    - **CV_CAP_PROP_POS_FRAMES -** current position of the video in frames
    - **CV_CAP_PROP_FRAME_WIDTH -** width of the frame of the video stream
    - **CV_CAP_PROP_FRAME_HEIGHT -** height of the frame of the video stream
    - **CV_CAP_PROP_FPS -** frame rate (frames per second)
    - **CV_CAP_PROP_FOURCC** - four character code of codec

- **bool VideoCapture::read(Mat& image);**

The function grabs the next frame from the video, decodes it and stores it in the '**image**' variable. Inside this function, VideoCapture::grap() and VideoCapture::retrieve() will be called. If you want, you can use these two functions instead of VideoCapture::read() function.
If the operation successful, it will return true. Otherwise it will return false.

- **waitKey(30)**

The function waits for 30 milliseconds. If a key was pressed before the specified time, it returns the ASCII value of the pressed key. If that value is 27 (ASCII value of 'esc' key is 27), the program will execute inside the if block. If no key is pressed during that 30ms, the function returns -1 program will continue the while loop.

- **VideoCapture::~VideoCapture()**

Destructor of VideoCapture object will destroy any associated memory of that particular object. This destructor will be called implicitly on exit of the main method of the above program.

Summary

At first, this program captures a video from a file. Then the program enters into a infinite loop. In that loop, it grabs frames from the captured video sequentially, decodes it, shows it in a window and waits for **30** milliseconds. If the video file has no more frames or if the user presses the 'esc' key, the program will break the infinite loop.

Note:
Using **waitKey(int)** function is very important because **imshow(string&, MAT)** function need time to paint the image in the window and **waitKey(int)** will give that necessary time.

Capture Video From Camera

The main difference of the following program from the above program is the argument to the VideoCapture constructor. Here you just need to give the index of your camera to the constructor of the VideoCapture object instead of the filename in the above program. Here is the sample OpenCV code.

```
////////////////////////////////////////////////////////////////////////////////////////////////////
#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 << "Cannot open the video cam" << endl;
      return -1;
   }

  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;

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

   while (1)
   {
      Mat frame;

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

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

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

      if (waitKey(30) == 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;

}
////////////////////////////////////////////////////////////////////////////////////////////////////
```

You can download this OpenCV visual c++ project from here. (The downloaded file is a compressed .rar folder. So, you have to extract it using Winrar or other suitable software)

This is how I capture myself with my webcam using the OpenCV program

New OpenCV functions which are not found earlier are explained here

- **VideoCapture::VideoCapture(int device)**

This is one of constructors available in VideoCapture class. This constructor open the camera indexed by the argument of this constructor and initializes the VideoCapture object for reading the video stream from the specified camera.
Here the '0' means the index of the camera to be used. You can use 1,2,3.. instead of 0, if your computer is attached to more than 1 camera.
The destructor of this class will deallocated any associated memory with this object. Therefore you don't need to deallocate memory explicitly in your program.

- **if (!cap.isOpened())**

If the VideoCapture object initialization unsuccessful, the expression inside the 'if 'parentheses will evaluate to true and the statements inside the 'if' block will be executed.
It is a good practice to check this scenario and handle it accordingly because otherwise it may cause the program to crash.

- **double dWidth = cap.get(CV_CAP_PROP_FRAME_WIDTH)**

This function obtain the width (in pixels) of frames of the camera output.

- **double dHeight = cap.get(CV_CAP_PROP_FRAME_HEIGHT)**

This function obtain the height (in pixels) of frames of the camera output.

All other OpenCV functions and their arguments are exactly same as the 1st program.

Next Tutorial : Write Image & Video to File

Previous Tutorial : Read & Display Image

Posted by Shermal Fernando

Is This Helpful : Yes (6) No (0)

100 comments:

LN April 8, 2013 at 6:18 AM

Execellent tutorial! YOU ARE AWESOME!!!!
Made it so easy and fun, thanks a bunch!

Reply

Mesmerizing Ivy May 29, 2013 at 9:11 PM

I need help with accessing TP-Link IP Camera TL-SC3130G in OpenCV. How can I access it wirelessly using WAN IP and port from a remote network?

Reply

Replies

Shermal Fernando May 31, 2013 at 12:38 PM