Chapter 12 OpenCV For Android

12.1 Introduction

OpenCV (Open Source Computer Vision Library) is a popular open source software library designed for computer vision application and machine learning. Its official web site is at:

http://opencv.org/

Since it is released under a BSD license, it is free for both academic and commercial use. It has a wide-range of interfaces, including C++, C, Python and Java, supporting Windows, Linux, Mac OS, iOS and Android platforms. OpenCV has been designed for computational efficiency with a strong focus on real-time applications and rich graphics features. The software has been adopted all around the world with more than forty-seven thousand people of user community and an estimated number of downloads exceeding 7 million.

OpenCV provides a common infrastructure for computer vision applications and helps accelerate the development of machine perception applications in commercial products. The library has more than 2500 optimized algorithms, including a comprehensive set of both classic and contemporary computer vision and machine learning algorithms, which can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements and moving objects, extract 3D models of objects, and recognize scenery, etc. The library has been used extensively in companies, research groups and governmental bodies, such as Google, Yahoo, IBM, Sony and many startup companies.

12.2 OpenCV4Android SDK

12.2.1 Getting the SDK

OpenCV supports the Android platform by providing the OpenCV4Android SDK package. Detailed information about the package, including download links, installations, and tutorials for various systems, can be found at the OpenCV site at:

http://docs.opencv.org/doc/tutorials/introduction/android_binary_package/O4A_SDK.html

The download link for version 2.4.9 is at

http://sourceforge.net/projects/opencvlibrary/files/opencv-android/2.4.9/

Suppose we have downloaded the package. We can unzip it into a directory, say, */apps/android*, by the commands,

```
$ cd /apps/android/
$ unzip ~/Downloads/OpenCV-2.4.9-android-sdk.zip
```

Suppose we get into the OpenCV directory and list its files with the commands,

```
$ cd OpenCV-2.4.9-android-sdk
$ ls
```

We should the see the following files and directories listed:

```
apk  doc  LICENSE  README.android  samples  sdk
```

The package has the following file structure:

```
OpenCV-2.4.9-android-sdk
|_ apk
|    |_ OpenCV_2.4.9_binary_pack_armv7a.apk
|    |_ OpenCV_2.4.9_Manager_2.18_XXX.apk
|
|_ doc
|_ samples
|     |_ 15-puzzle
|     |_ camera-calibration
|     |_ color-blob-detection
|     .....
|     |_ example-15-puzzle.apk
|     .....
|_ sdk
|     |_ etc
|     |_ java
|     |_ native
|           |_ 3rdparty
|           |_ jni
|           |_ libs
|                |_ armeabi
|                |_ armeabi-v7a
|                |_ x86
|
|_ LICENSE
|_ README.android
```

12.2.2 Import OpenCV Library and Samples

It is better to start working with OpenCV for Android from a new clean workspace. Suppose we choose our workspace to be */apps/android/opencv-work*. As usual, we use Eclipse IDE for development. We can **import** the OpenCV library into our workspace using Eclipse:

1. Click **File** > **Import..**
2. Select **Android** > **Existing Android Code Into Workspace** and click **Next**.
3. Enter */apps/android/OpenCV-2.4.9-android-sdk* for the **Root Directory**.
4. Check all the projects to import and check *Copy projects into workspace* as shown in Figure 12-1. Check **Finish** to import the projects.

5. You may or may not need to do the following, depending on how you have installed your Eclipse IDE.
   (a) If you do not have a copy of Android NDK, you need to download it from the site:
   https://developer.android.com/tools/sdk/ndk/index.html
   Unzip and unpack the package. Install it in Eclipse IDE:
   Click **Window** > **Preferences** > **Android** > **NDK**. Then enter the root directory of the NDK for **NDK Location** (e.g. */apps/android/android-ndk-r10*), and click **Apply** > **OK**. (You may need to restart Eclipse.)
   (b) You may need to install the OpenCV Manager manually. In our example, the command to install it is:
   $ adb install /apps/android/OpenCV-2.4.9-android-sdk/
   apk/OpenCV_2.4.9_Manager_2.18_armeabi.apk

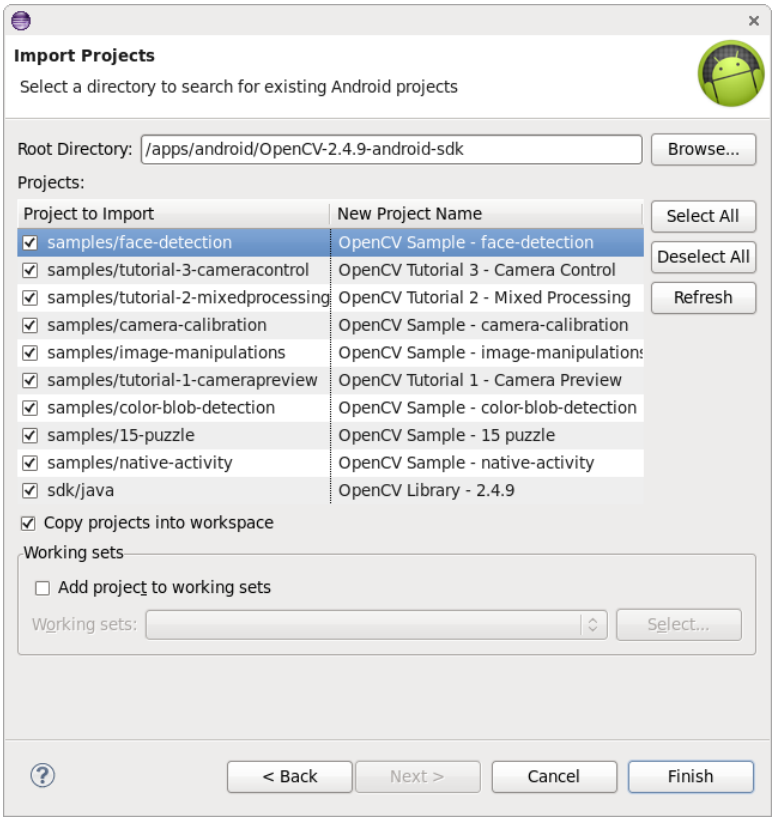

Figure 12-1 Importing OpenCV Projects From SDK

12.2.3 Run OpenCV Samples

After the import and the setup, we should be able to run the Open CV samples. (You may need to do some configuration of the Eclipse IDE to run them, depedning on how your Eclipse environement has been setup.) For example, to run the *15 puzzle*, we do the following:

1. Click on the first project, *OpenCV Sample - 15 puzzle*, in the *Package Explorer*. You may need to first clean the project by clicking **Project** > **Clean..** ... If you still see red error indicators at the import statements of the source code, you can click on an error indicator and choose **Fix project setup ..**, and then click **Add project 'OpenCV Library - 2.4.9' to build path ...**
2. Add the OpenCV library in Eclipse: Click **File** > **Properties** > **Android** > **Library** > **Add**. Select **OpenCV Library - 2.4.9** and click **OK**.
3. Include 'Emulated Camera' in AVD: Click **Window** > **Android Virtual Device Manager**. Then click the tab **Android Virtual Devices** of the Android Virtual Device Manager dialog, select the AVD you are going to run the project, click **Edit**. The *Edit Android Virtual Device (AVD)* panel appears. Choose **Emulated** for both of the entries *Front Camera* and *Back Camera*. Click **OK** to go back to the *Android Virtual Device Manager*. Press the escape key *Esc* to return to the Eclipse editing screen.
4. Now we can run the application by clicking **Run** > **Run** > **Android Application**. We should see a moving pattern as shown in Figure 12-2.

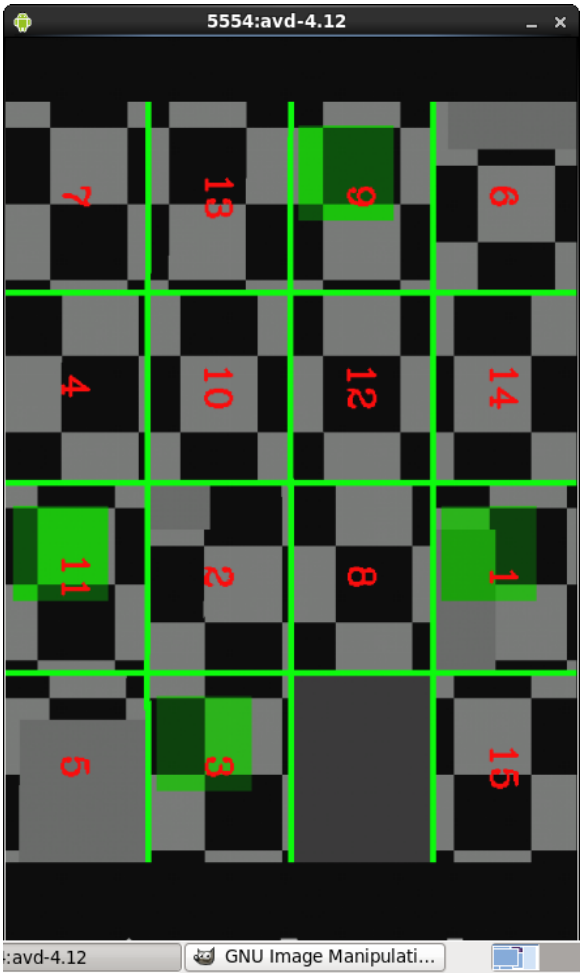

Figure 12-2 OpenCV Samples Outputs

If you "C/C++ Build" is not in the *Build Property* of your project (right-click project and select **Properties** to check) and you need it in the project, you can add it by:

1. Close all projects.
2. Open the library project and right-click on it. Select **Android Tools** > **Add Native Support..**
3. Enter a unique library name that does not contain any space and click **Finish**.
4. Right-click the project (at package explorer) and select **Properties**. The *C/C++ Build* should be there. Change the build command to ${*NDKROOT*}*/ndk-build*, assuming *NDKROOT* has been setup properly.
5. Build the project by clicking **Project** > **Build Project**.