Tutorial on using Android for Image Processing Projects

EE368 Digital Image Processing, Spring 2010

Linux Version for SCIEN Lab

Introduction

In this tutorial, we will learn how to set up the Android software development environment and how to implement image processing operations on an Android-based mobile device (e.g., the DROID phone that you have received for the class assignments). Android is an open-source platform developed by Google and the Open Handset Alliance on which interesting and powerful new applications can be quickly developed and distributed to many mobile device users. There is a growing community of Android developers and a growing market for Android-based devices (e.g., Motorola DROID, HTC DROID ERIS, Google Nexus One). Android also comes with a vast library of useful functions, including functions for user interfaces, image/bitmap manipulation, and camera control that we will frequently use in EE368. We look forward to seeing your novel image processing algorithms and applications running on Android-based devices as the quarter progresses.

The tutorial is split into two parts. In the first part, we will explain how to personalize the Android software development environment on a SCIEN computer. Then, in the second part, we will explain how to develop image processing programs that can run on an Android-based mobile device.

Part I: Creating the Software Development Environment

Configuring Eclipse

The necessary software packages are already installed on the SCIEN lab computers. Just a few simple steps need to be taken to personalize the environment.

1. Change to your root directory, e.g., `/afs/ir.stanford.edu/users/j/d/johndoe`
2. Open your .cshrc file in a text editor like emacs.
3. Add the following line to modify your PATH.
   ```
   setenv PATH /users/ee368/EE368-Android/android-sdk-linux_86/tools:$PATH
   ```
4. Add the following line to create an alias to Eclipse.
   ```
   alias ee368eclipse '/users/ee368/EE368-Android/eclipse/eclipse –vm\
   /users/ee368/EE368-Android/jdk1.6.0_19/jre/bin'
   ```
5. Close the text editor and type the following command.
   ```
   source .cshrc
   ```
6. Type the following command to start Eclipse.
   ```
   ee368eclipse &
   ```

7. If asked to choose a workspace at start-up, please choose a personal directory where you would like to store all your Android projects. If not asked at start-up, after Eclipse starts, go to File > Switch Workspace to specify a personal directory as the current workspace. Since the SCIEN computers are public machines, it is important to work in your personal workspace and not modify files in other users' workspaces.

Linking Your Phone to Your Computer

1. Turn on your phone.
2. Go to the home screen.
3. Select Applications > Development and then enable USB debugging.

Part II: Developing Image Processing Programs for Android

Now that the Google Android SDK, the Java Runtime, and the Eclipse IDE are all set up, we are ready to start writing image processing programs that can run an Android-compatible mobile device.

Hello World Example

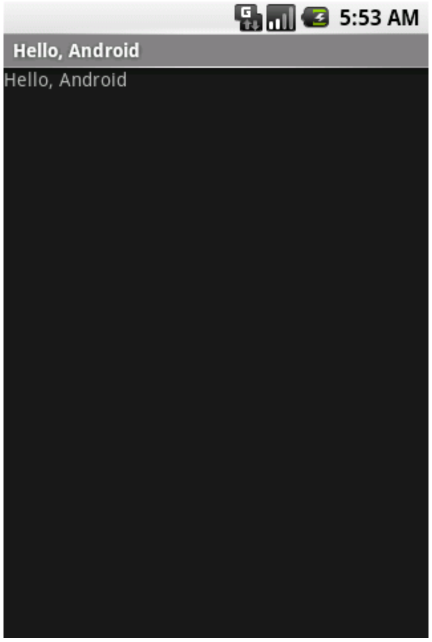
First, we will build a simple Android program in Eclipse. This simple example will also help you to become familiar with how to create an Android project, how to (auto) compile source code, and how to run the generated executable on the mobile device.

Please follow the instructions on this page to develop the "Hello World" program:
`http://developer.android.com/resources/tutorials/hello-world.html`

(Note: In case there are strange errors when the project is created about "R.java", simply add a comment line (e.g., "// dummy comment") at the top of "gen : com.example.helloandroid : R.java" and save the file. This will make the errors disappear.)

In the external "Hello World" tutorial, they only run the "Hello World" program in an emulator. Additionally, we will now also run the program on the actual Android-based phone. Make sure your phone is properly linked to your computer.

1. In Eclipse, select Run > Run Configurations > Android Application > HelloWorld > Target. Choose Manual for Deployment Target Selection Mode.
2. Select Run, and in the Device Chooser dialog, select your Android-based phone. The "Hello World" program will be sent to and automatically started on your phone, and you should see the screen similar to Figure 1 on your phone.



```
Hello, Android
```

Figure 1. “Hello World” program running on Android-based phone.

EE368 Viewfinder Example

Now, having grasped the fundamentals of building and running an Android application, we will create a more complicated project involving the onboard camera and real-time image processing.

1. Create a new Android project with the following parameters.

   Project name: ViewfinderEE368
   Check the box for Android 2.0.1
   Application name: Viewfinder EE368
   Package name: com.example.viewfinderee368
   Check the box for Create Activity and enter: ViewfinderEE368
   Min SDK Version: 6

2. Copy the text in the following document into AndroidManifest.xml. This defines the main activities and permissions for this program.

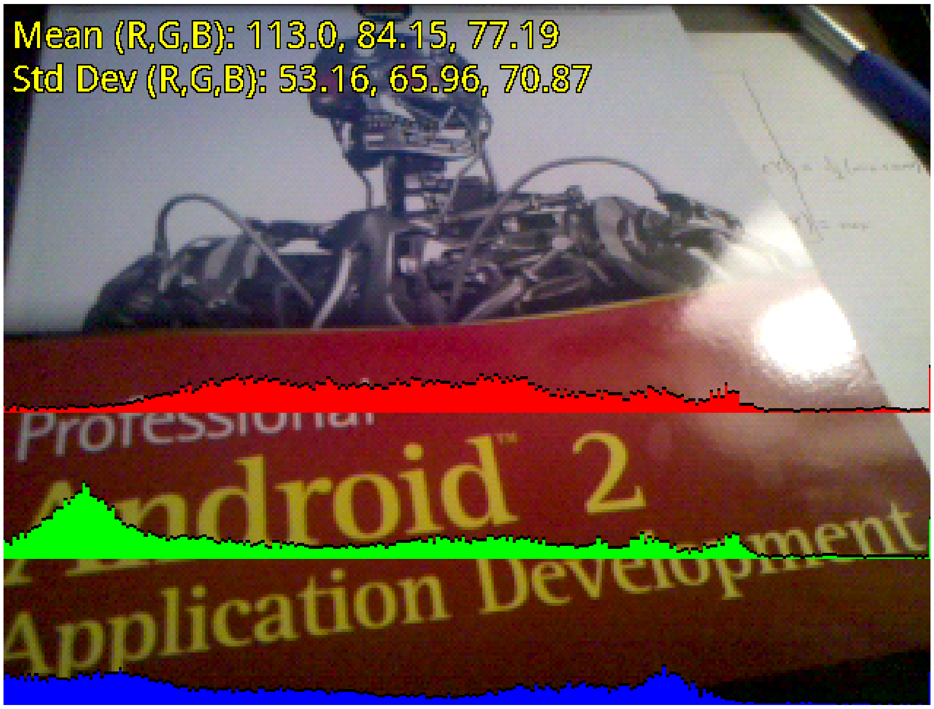
   `http://ee368.stanford.edu/Android/ViewfinderEE368/AndroidManifest.xml`

3. Copy the text in the following document into src : com.example.viewfinderee368 : ViewfinderEE368.java. This defines the classes in this program.

   `http://ee368.stanford.edu/Android/ViewfinderEE368/ViewfinderEE368.java`

4. Check to make sure everything is copied correctly into the project. If there are compilation errors, a red X will appear in the Package Explorer.

5. Select Run and in the Device Chooser dialog, select your phone. You should see something like Figure 2 on your phone. Point the camera at different objects around you to see how the mean, standard deviation, and histogram of each color channel changes dynamically. You are augmenting the viewfinder in real time!



```
Mean (R,G,B): 113.0, 84.15, 77.19
Std Dev (R,G,B): 53.16, 65.96, 70.87
```

Figure 2. “Viewfinder EE368” program running on Android-based phone.

Real-time Phone Debugging in Eclipse

It is actually possible to view real-time messages from the phone in Eclipse, which can be very helpful for debugging and code development.

1. Select Window > Open Perspective > DDMS.
2. A new tab entitled “DDMS” should appear next to the default “Java” tab. Click on the “DDMS” tab.
3. Select Window> Show View > LogCat. The LogCat view shows a sequential list of real-time messages from the phone. In particular, error messages in red can be very useful when trying to debug a problem.