Assignment 4

19126293 林泽鑫
19126316 帖宇

Introduction

For this assignment, we choose to develop an Android App with NDK. The App aims to achieve the functionality of Canny edge detection by invoking OpenCV library via JNI.

Project repository lies in https://github.com/15301074/Android_JNI_opencv

Environment Setup

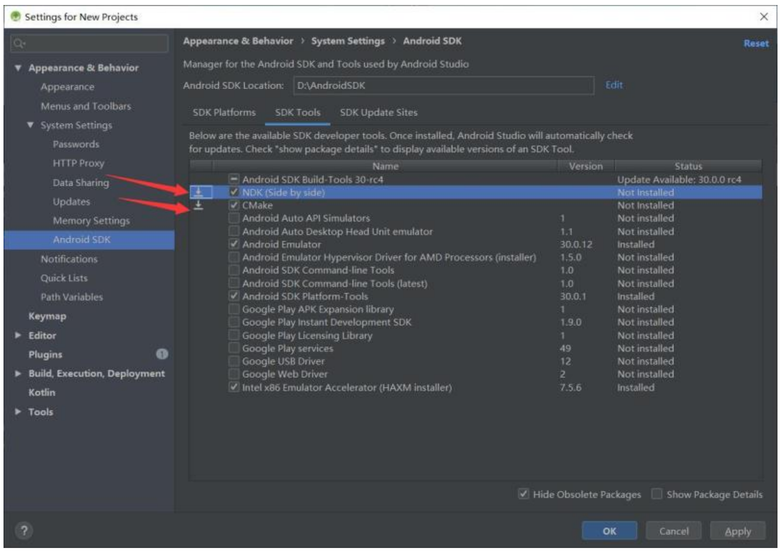
The experiment is conducted under the following environment:

```
Android Studio 3.5.2

Build #AI-191.8026.42.35.5977832, built on October 31, 2019

JRE: 1.8.0_202-release-1483-b03 amd64

JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

Windows 10 10.0
```

1. Install Required Components

   - Download OpenCV-3.4.6-android-sdk from OpenCV official website

   - Install NDK with SDK tools



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: D:\AndroidSDK

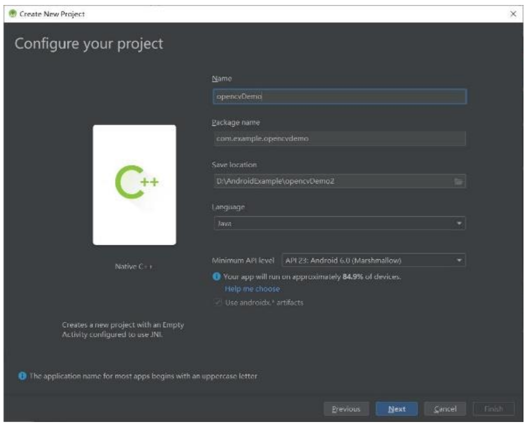
Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

Name	Version	Status
− Android SDK Build-Tools 30-rc4		Update Available: 30.0.0 rc4
✓ NDK (Side by side)		Not Installed
✓ CMake		Not Installed
☐ Android Auto API Simulators	1	Not installed
☐ Android Auto Desktop Head Unit emulator	1.1	Not installed
✓ Android Emulator	30.0.12	Installed
☐ Android Emulator Hypervisor Driver for AMD Processors (installer)	1.5.0	Not installed
☐ Android SDK Command-line Tools	1.0	Not installed
☐ Android SDK Command-line Tools (latest)	1.0	Not installed
✓ Android SDK Platform-Tools	30.0.1	Installed
☐ Google Play APK Expansion library	1	Not installed
☐ Google Play Instant Development SDK	1.9.0	Not installed
☐ Google Play Licensing Library	1	Not installed
☐ Google Play services	49	Not installed
☐ Google USB Driver	12	Not installed
☐ Google Web Driver	2	Not installed
✓ Intel x86 Emulator Accelerator (HAXM installer)	7.5.6	Installed

✓ Hide Obsolete Packages ☐ Show Package Details

2. Create a New Project

The project is created with the template configured to use JNI under C++ 14.


Create New Project

Configure your project

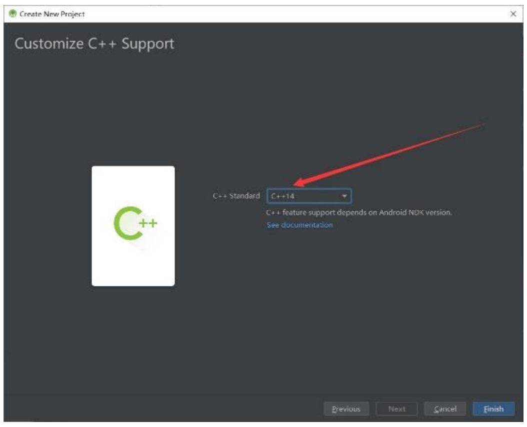
Name: opencvDemo
Package name: com.example.opencvdemo
Save location: D:\AndroidExample\opencvDemo2
Language: Java
Minimum API level: API 23: Android 6.0 (Marshmallow)

Your app will run on approximately **84.9%** of devices.

Use androidx.* artifacts

Creates a new project with an Empty Activity configured to use JNI.

The application name for most apps begins with an uppercase letter


Create New Project

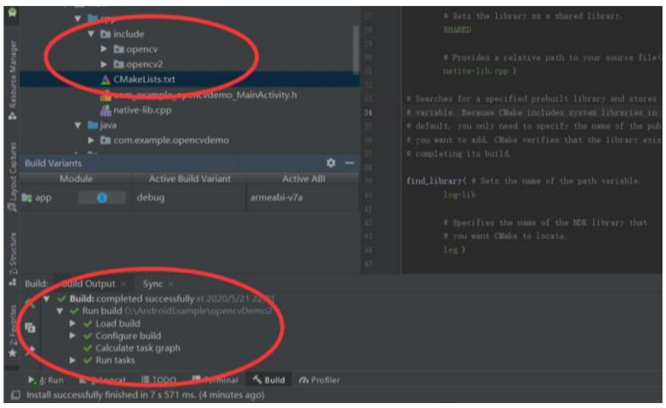
Customize C++ Support

C++ Standard: C++14

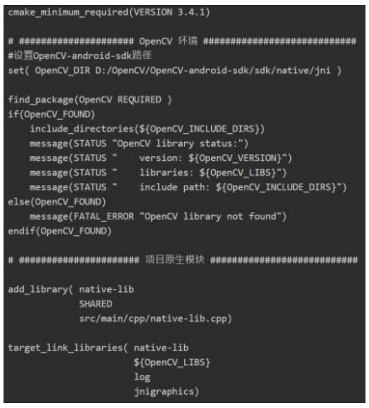
C++ feature support depends on Android NDK version.

3. Configure CMakeList.txt

Specify path for OpenCV-3.4.6-android-sdk, and setup library registration and target link.



```
Build: completed successfully at 2020/5/21 22:11
  Run build D:\AndroidExample\opencvDemo2
    Load build
    Configure build
    Calculate task graph
    Run tasks
Install successfully finished in 7 s 571 ms. (4 minutes ago)
```



```
cmake_minimum_required(VERSION 3.4.1)

# #################### OpenCV 环境 ###########################
#设置OpenCV-android-sdk路径
set( OpenCV_DIR D:/OpenCV/OpenCV-android-sdk/sdk/native/jni )

find_package(OpenCV REQUIRED )
if(OpenCV_FOUND)
    include_directories(${OpenCV_INCLUDE_DIRS})
    message(STATUS "OpenCV library status:")
    message(STATUS "    version: ${OpenCV_VERSION}")
    message(STATUS "    libraries: ${OpenCV_LIBS}")
    message(STATUS "    include path: ${OpenCV_INCLUDE_DIRS}")
else(OpenCV_FOUND)
    message(FATAL_ERROR "OpenCV library not found")
endif(OpenCV_FOUND)

# ##################### 项目原生模块 ##########################

add_library( native-lib
             SHARED
             src/main/cpp/native-lib.cpp)

target_link_libraries( native-lib
                       ${OpenCV_LIBS}
                       log
                       jnigraphics)
```

Project Structure

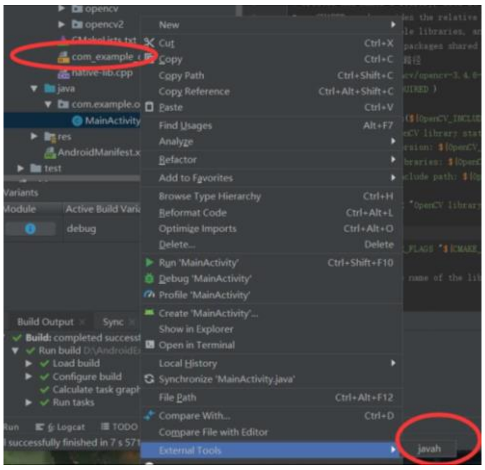
- cpp
  - include: header files for OpenCV-android-sdk
  - CMakeList.txt: building instructions for headers files of OpenCV-android-sdk and associated native library
  - com_example_opencvdemo_MainActivity.h: generated header file for declared native methods in MainActivity
  - native-lib.cpp: implementation of JNI calls defined in the application
- java
  - MainActivity: the activity which implements the functionalities of image display and Canny edge detection

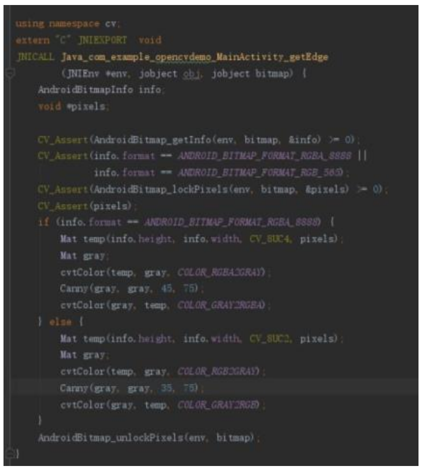
OpenCV Integration

Native Layer

1. Generate the header file for declared native methods in the MainActivity, in our case the only native method defined is: native void getEdge(Object bitmap).

2. Implement defined JNI call in corresponding .cpp file.





```cpp
using namespace cv;
extern "C" JNIEXPORT void
JNICALL Java_com_example_opencvdemo_MainActivity_getEdge
        (JNIEnv *env, jobject obj, jobject bitmap) {
    AndroidBitmapInfo info;
    void *pixels;

    CV_Assert(AndroidBitmap_getInfo(env, bitmap, &info) >= 0);
    CV_Assert(info.format == ANDROID_BITMAP_FORMAT_RGBA_8888 ||
              info.format == ANDROID_BITMAP_FORMAT_RGB_565);
    CV_Assert(AndroidBitmap_lockPixels(env, bitmap, &pixels) >= 0);
    CV_Assert(pixels);
    if (info.format == ANDROID_BITMAP_FORMAT_RGBA_8888) {
        Mat temp(info.height, info.width, CV_8UC4, pixels);
        Mat gray;
        cvtColor(temp, gray, COLOR_RGBA2GRAY);
        Canny(gray, gray, 45, 75);
        cvtColor(gray, temp, COLOR_GRAY2RGBA);
    } else {
        Mat temp(info.height, info.width, CV_8UC2, pixels);
        Mat gray;
        cvtColor(temp, gray, COLOR_RGB2GRAY);
        Canny(gray, gray, 35, 75);
        cvtColor(gray, temp, COLOR_GRAY2RGB);
    }
    AndroidBitmap_unlockPixels(env, bitmap);
}
```

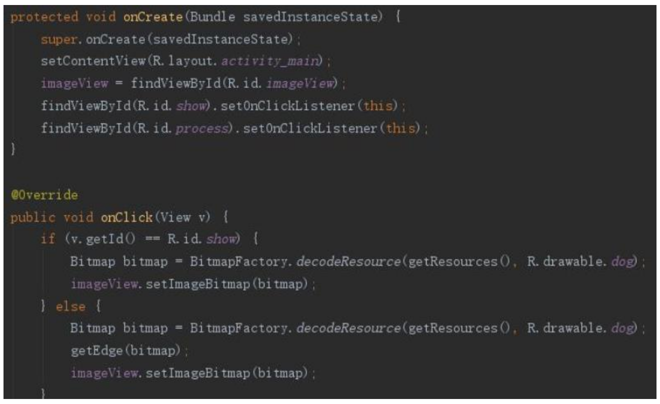
Application Layer

Our application just contains one simple activity called MainActivity and associated view declaration. The functionalities implemented include:

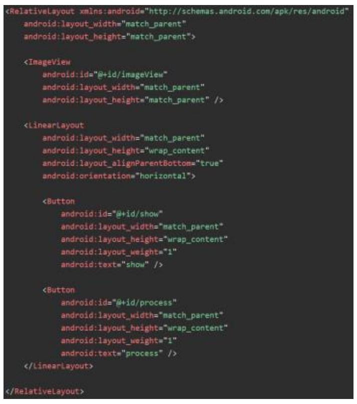
- Image display

- Canny edge detection via JNI call depending on OpenCV

There're two buttons "SHOW" and "PROCESS" and an ImageView control in the MainActivity. The click handler bound to "SHOW" button implements the logic of image display, while the handler bound to "PROCESS" simply invoke the JNI call "getEdge(bitmap)" to achieve Canny edge detection.

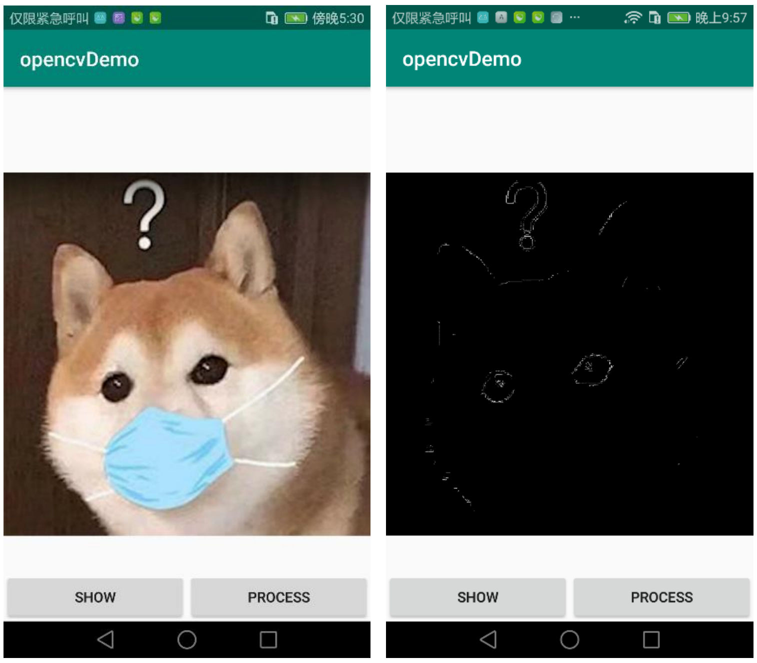


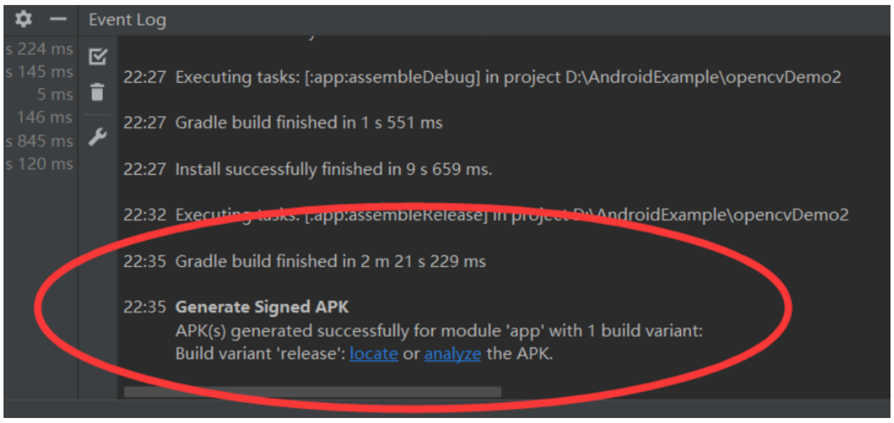
```java
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    imageView = findViewById(R.id.imageView);
    findViewById(R.id.show).setOnClickListener(this);
    findViewById(R.id.process).setOnClickListener(this);
}

@Override
public void onClick(View v) {
    if (v.getId() == R.id.show) {
        Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.dog);
        imageView.setImageBitmap(bitmap);
    } else {
        Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.dog);
        getEdge(bitmap);
        imageView.setImageBitmap(bitmap);
    }
```



```xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:orientation="horizontal">

        <Button
            android:id="@+id/show"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="show" />

        <Button
            android:id="@+id/process"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="process" />
    </LinearLayout>

</RelativeLayout>
```

The execution results and signed APK are shown as follows:





```
s 224 ms
s 145 ms
    5 ms
  146 ms
s 845 ms
s 120 ms

22:27  Executing tasks: [:app:assembleDebug] in project D:\AndroidExample\opencvDemo2

22:27  Gradle build finished in 1 s 551 ms

22:27  Install successfully finished in 9 s 659 ms.

22:32  Executing tasks: [:app:assembleRelease] in project D:\AndroidExample\opencvDemo2

22:35  Gradle build finished in 2 m 21 s 229 ms

22:35  Generate Signed APK
       APK(s) generated successfully for module 'app' with 1 build variant:
       Build variant 'release': locate or analyze the APK.
```

References

1. 《基于 Android studio3.6 的 JNI 教程之 opencv 实例详解》

2. 《OpenCV On Android 最佳环境配置指南(Android Studio 篇)》

3. 《NDK 开发笔记之 1：AndroidStudio3.0+ 配置 Ndk 和集成 OpenCV4.0 一个简单例子》

4. 《通过 CMake 方式生成动态库 so 文件》

5. 《CMake 方式导入第三方.a 静态库以及编译库时出现问题》

6. 《有关 Opencv Undefined Reference to cv::Mat::updateContinuityFlag()编译问题的解决》