Mobile Visual Computing in C++ on Android

Yun-Ta Tsai* NVIDIA
Orazio Gallo† NVIDIA
David Pajak‡ NVIDIA
Kari Pulli§ NVIDIA

Abstract

Based on a tutorial developed by NVIDIA's Mobile Visual Computing team[1], this course will teach the basics to jump-start a visual computing project on Android using native C/C++ code. After explaining how to set up the programming environment and write a simple native application, we will dive into more advanced topics related to Computer Vision (using OpenCV optimized for Android), and high-performance Image Processing (using OpenGL ES2).

1 Introduction

Android is inherently Java-oriented, but native code is often a better choice for computationally heavy algorithms—such as those used in visual computing; among other things, it allows to exploit the power of heterogeneous computing and hardware intrinsics.

This course is targeted to people who are fluent in C/C++, but are not familiar with the native Android development workflow. We divided the tutorial into three sections: Android Native Development, OpenCV, and OpenGL. As the course progresses, we will gradually build a complete visual computing application which can serve as a building block for larger projects.

2 Android Native Development

In this section, we will quickly get people familiar with the development environment using the Tegra Android Development Pack (TADP). TADP is a collection of freely available tools developed by NVIDIA to simplify the native development on Android; it is fully compatible with official Android systems, it is highly optimized for Tegra-based devices, although it can be used with non-Tegra devices as well. The TADP also comes with several examples that make the process of getting started easier.

We will start from project creation, the project's folder structure, the Android Native Activity system, Makefiles, permissions, and the Android Debug Bridge (ADB), which allows to communicate with Android devices. Finally, we will explain how to efficiently debug in the Android native environment.

At the end of this section, the audience should be able to create a C/C++ project for Android.

3 OpenCV

OpenCV is the *de facto* standard vision library; it has been widely used and extended by the computer vision community for years. It is a great tool to evaluate ideas quickly. Moreover, the TADP includes an highly optimized version of OpenCV for Tegra, allowing developers to enjoy the convenient API without sacrificing performance. In this section, we will demonstrate how to integrate OpenCV in our Android project, how to control the camera of the device, how to perform some basic computation, and how to display the results.

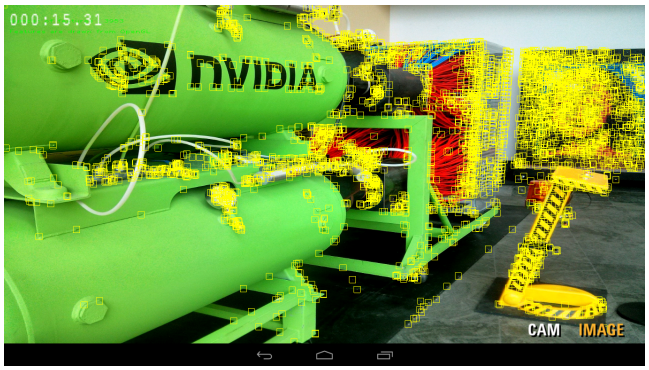

Figure 1: *A screen-host of an NVIDIA-powered tablet performing real-time feature detection. Today's mobile devices are sufficiently powerful to run algorithms that were prohibitive on desktop machines only a few years ago. However, getting started with writing native code for the Android platform can be intimidating even for programmers who are proficient in C/C++.*

4 OpenGL ES2 and GPGPU

OpenGL ES2 is often necessary to render compelling visual content efficiently on Android, as it allows to off-load all the heavy lifting to the GPU. In this section, we will show how to display content using OpenGL ES2, with tips to improve performance in the presence of a heterogeneous pipeline that includes other hardware resources, such as the CPU.

The TADP also provides a set of open-source helpers to reduce the complexity of the management of resources, such as textures and shaders. We will walk the audience through some of the useful functionality that makes OpenGL ES2 programming easier.

Finally, we will demonstrate how to use shaders to perform GPGPU on the device, and we will compare the performance with the optimized implementation for OpenCV.

5 Conclusion

The same algorithms that were prohibitive for desktop machines only a few years ago, are now making their way to mobile devices. This course will simplify the transition from the traditional programming environment to mobile development. After attending this course, researchers and scientists will be able to quickly prototype their ideas on Android platforms.

*e-mail:ytsai@nvidia.com
†e-mail:ogallo@nvidia.com
‡e-mail:dpajak@nvidia.com
§e-mail:karip@nvidia.com

[1] https://developer.nvidia.com/content/native-android-development-tutorial

SIGGRAPH 2013, July 21 – 25, 2013, Anaheim, California.

ACM 978-1-4503-2261-4/13/07