Unity: iOS and Android - Cross Platform Challenges and Solutions

Renaldas Zioma
Unity Technologies
rej@unity3d.com

Aras Pranckevičius
Unity Technologies
aras@unity3d.com

1. Introduction

Unity powers more than a thousand different titles on mobile platforms. One of the main strengths of Unity is the ability to efficiently support a very wide range of mobile devices starting from low-end Android devices running on ARMv6 CPUs with GLES1.1 level GPUs to iPad3, and across 5 different GPU architectures.

In this talk we will share our experience harnessing distinct GPU architectures and lessons learned to maintain an acceptable quality and performance level across multiple devices and operating systems.

2. Elaboration

At present several different GPU architectures are flourishing in the mobile space. Almost every architecture introduces its own texture compression scheme, a set of API extensions, performance analysis tools, and different bugs in graphical drivers.

2.1 Graphical test suite on mobiles

We have adopted a graphics functional test suite for automated testing on mobile platforms. There are a number of gotchas we learned in the process of finding the suitable devices and achieving reproducible results.

2.2 Cross platform shaders

Unity can deploy not only to mobile platforms, but to desktops and web as well. Cg is used as the main cross platform shading language. The cross compilation step is employed to generate a platform specific shader in the HLSL, GLSL or GLSL ES language. An additional optimization step was developed to help the platform driver in achieving the best performance.

2.3 Efficient dynamic geometry submission

The combination of certain API extensions, driver bugs and GPU architectures calls for several practical approaches when submitting dynamic geometry depending on the mobile platform. Currently we employ the following approaches:

- vertex buffer "orphaning"
- double/triple buffering
- queue of preallocated buffers and
- rendering directly from system memory.

We will compare the performance of different approaches on a set of widely used mobile devices and explain why the performance differs so drastically.

2.4 Measuring GPU performance

We will present several "*poor-man*" approaches for coarse GPU profiling we developed so far for platforms which lack related tools (such as iOS) and overview our experience integrating extensions suitable for GPU profiling directly into the engine, when available.

2.5 Dealing with broken paths in drivers

We will share our experience dealing with driver pitfalls on certain platforms and propose workarounds.

2.6 Different texture compressions

We will give a short overview of the different texture compression schemes, their pros and cons, overview available compression libraries and explain why we chose certain approaches ourselves.

2.7 Skinning on CPU

We will present the performance results and explain why we choose to implement skinning on the CPU using VFP or NEON instructions instead of the GPU approach.

2.8 Optimizing shaders and post-process effects

We will present a number of approaches we use to profile and optimize shaders and full screen post-process effects on a variety of mobile platforms.

3. Conclusions

Cross platform challenges and lessons learned while developing genre agnostic technology suitable for a very wide range of devices can be applied by mobile developers working on their custom game engines and can foster further discussions in the field of driver stability and profiling tools on mobile platforms.

SIGGRAPH 2012, Los Angeles, California, August 5 – 9, 2012.
ISBN 978-1-4503-1435-0/12/0008