



Performance improvements in cocos2d-x v3.0: Lessons learned

Ricardo Quesada

Who am I?



- ✦ Ricardo Quesada
- ✦ Co-founder and main author of cocos2d
- ✦ Chief Architect at Chukong

What is Cocos2d-x?

Cocos2d-x



- ✦ 2D Game engine + 3D extensions
- ✦ Fast and robust
- ✦ Easy to use
 - ✦ Familiar API
- ✦ Open Source
 - ✦ MIT License
- ✦ Multiplatform
 - ✦ Mobile, desktop and web

Cocos2d-x Basic Features

- ✦ Workflow
- ✦ Scene graph
- ✦ Sprites
- ✦ Particles
- ✦ Labels
- ✦ Tile Maps
- ✦ Parallax Scrolling
- ✦ Actions
- ✦ Mesh effects
- ✦ Physics integration
- ✦ 3D extensions
- ✦ Animation support

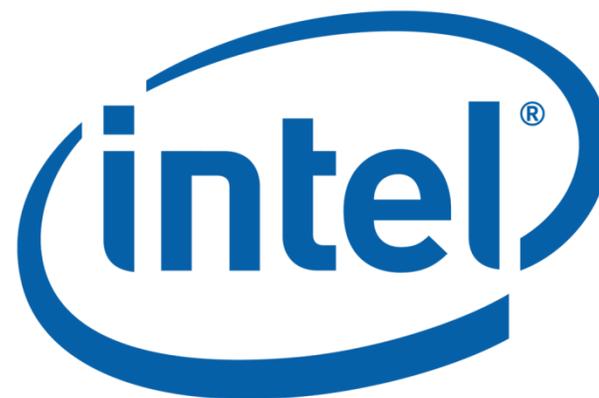
Who is Using Cocos2d-x ?

- ★ Many companies
 - ★ From big companies like Zynga
 - ★ To long-tail / indie companies
- ★ Used by thousands mobile games
 - ★ Many Top #10 games on AppStore and PlayStore
 - ★ 70% of Chinese games are using Cocos2d-x

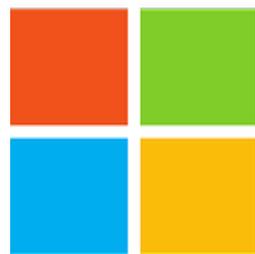
Partners



ARM[®]



Google



Microsoft

Performance Improvements: Renderer

Renderer: Performance improvements

- ✦ Auto Batching
- ✦ Auto Culling
- ✦ Caching Transform

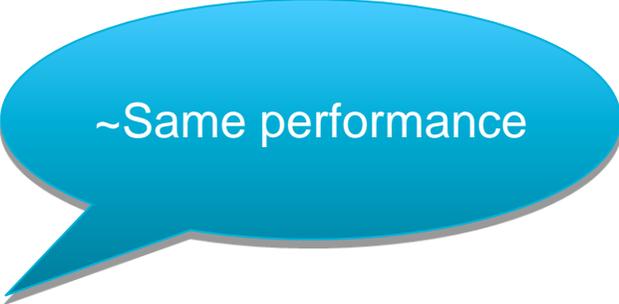


Auto Batching

- ★ v3.0-beta
 - ★ Fast for sprites
 - ★ We profiled GPU. It was very fast.
 - ★ We did basic CPU profiling. It was fast.
 - ★ But some users were reporting it was very slow
- ★ We did more CPU profiling
 - ★ We were copying the Quads 2 times
 - ★ For big Tile Maps it was 5x slower
- ★ Basically our tests suit was incomplete

Auto Culling

- ★ Different strategies:
 - ★ Frustum culling
 - ★ Bounding box testing
 - ★ No culling



~Same performance

- ★ We realized:
 - ★ We weren't doing the correct tests
 - ★ Our culling algorithm was not fast enough



1X~20X

Performance Improvements: visit()

Visit(): Performance improvements

- ✦ Profiled key parts of the code

- ✦ Node::sortAllChildren(): std::sort()



- ✦ Node::visit(): caching + other optimizations



Performance improvements: Other examples

GL_TRIANGLES vs GL_TRIANGLE_STRIP



- ★ A few years ago Apple recommended GL_TRIANGLE_STRIP
 - ★ But in cocos2d, GL_TRIANGLES was much faster
- ★ Certain best practices might not be valid for your game
 - ★ Or not valid for a certain GPU / OpenGL driver
- ★ Profile everything...
 - ★ ...including recommend best practices

Xcode® 5.0 vs 5.1beta5



- ★ Xcode 5.0 was about 5% faster than v5.1 beta5
 - ★ Fixed in final v5.1
- ★ Profile everything in the same environment
 - ★ Same device
 - ★ Same operating system version
 - ★ Same toolchain

Performance Improvements: The ones that didn't work

Optimizing $\text{vec3} * \text{mat4}$

- ★ $(\text{vec4} * \text{mat4})$ is super fast with Neon instructions
- ★ $(\text{vec3} * \text{mat4})$ is slower
- ★ Tried:
 - ★ Convert vec3 to vec4
 - ★ $\text{vec4} * \text{mat4}$ using Neon
 - ★ Convert output from vec4 to vec3
- ★ Result: a bit slower than $\text{vec3} * \text{mat4}$ in C due to the conversion to/from $\text{vec3}/\text{vec4}$

Improvements for v3.1
Work in progress

Trying `vec3[] * mat4`

- ★ `(vec3 * mat4)` good speed using Neon
- ★ `(vec3[] * mat4)` much faster using Neon
- ★ Unfortunately cocos2d-x can't use it because it uses an interleaved array:
 - ★ Array = `[vec3,vec2,vec4; vec3,vec2,vec4; vec3,vec2,vec4]`
- ★ We are working with ARM on an interleaved `vec3[] * mat4`

Auto culling in Tile Maps

- ★ v3.0:
 - ★ Sends the whole map all the time
 - ★ Super fast for small maps
 - ★ Super slow for big maps
- ★ v3.1:
 - ★ Only sends visible tiles
 - ★ Fast for both small and big maps
 - ★ Prototype working:
 - ★ 10x faster for big maps
 - ★ about 5% ~ 10% slower for small maps on old devices

Profiling Tools Used for Cocos2d-x v2.2 and v3.0

Android™ OS: ARM® DS-5 Toolchain

CPU profiling

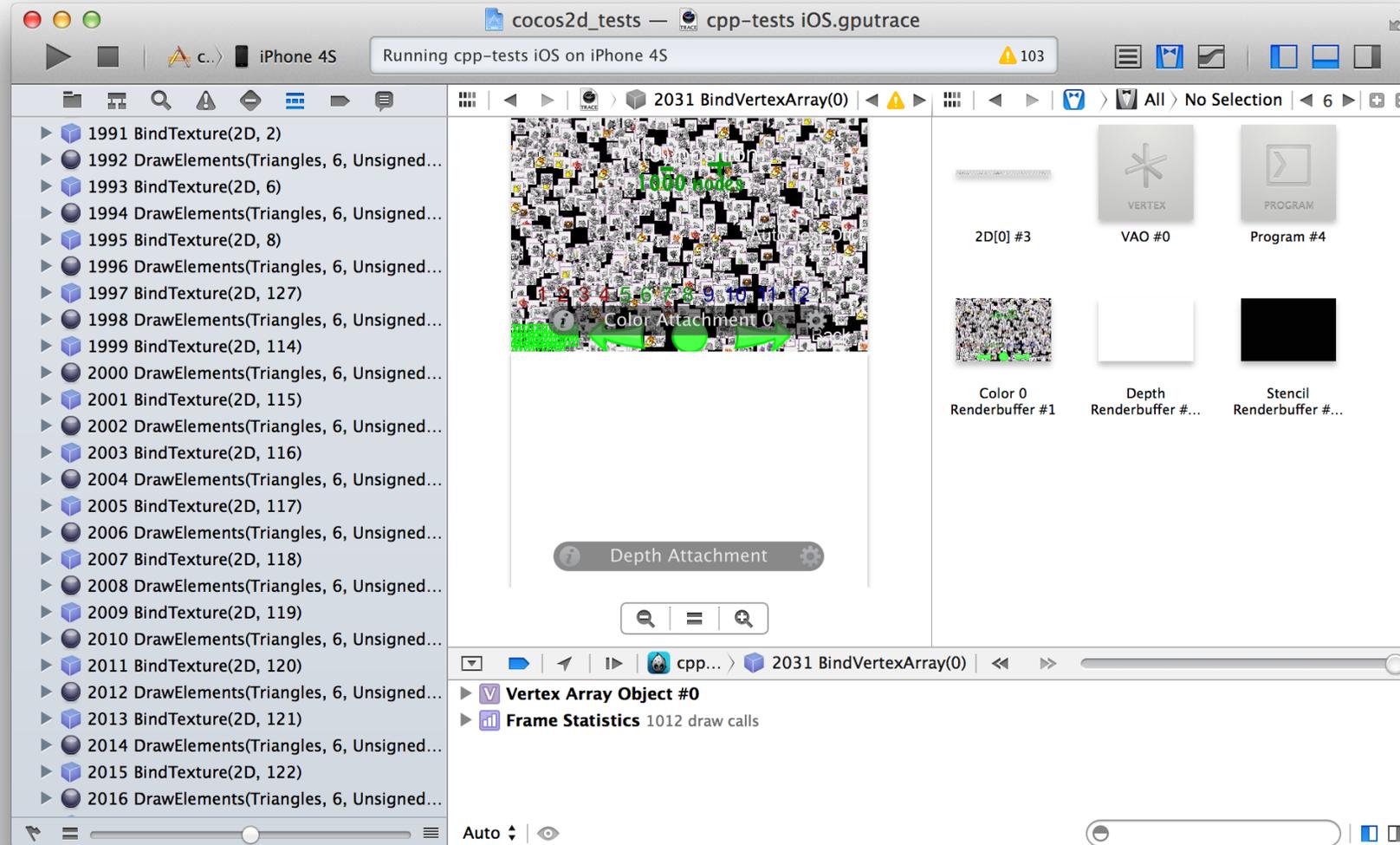


Functions: 2
Samples (Self): 201 (0.48%)

Self	% Self	Instances	Function Name	Location
1,560	3.71%	1	__addsf3	ieee754-sf.S:68
973	2.32%	1	__mulsf3	ieee754-sf.S:452
436	1.04%	1	joy::cocos2d::CCParticleSystem::update(float)	CCParticleSystem.cpp:589
256	0.61%	1	joy::cocos2d::CCObject::~~CCObject_sub_object()	CCObject.cpp:50
239	0.57%	1	joy::cocos2d::CCParticleSystemQuad::updateQuadWithParticle(joy::...	CCParticleSystemQuad.cpp:241
183	0.44%	1	joy::cocos2d::CCObject::CCObject_sub_object()	CCObject.cpp:40
162	0.39%	1	joy::cocos2d::CCPoint::CCPoint(float, float)	CCGeometry.cpp:36
147	0.35%	1	__divsf3	ieee754-sf.S:655
133	0.32%	1	joy::cocos2d::CCScriptEngineManager::sharedManager()	CCScriptSupport.cpp:130
97	0.23%	2	__nesf2	ieee754-sf.S:823
78	0.19%	1	joy::cocos2d::CCPoint::operator =(const joy::cocos2d::CCPoint&)	CCGeometry.cpp:47
77	0.18%	1	joy::cocos2d::CCNode::visit()	CCNode.cpp:783
75	0.18%	2	.plt [libjoygamesdk.so]	libjoygamesdk.so
67	0.16%	1	joy::cocos2d::CCPoint::CCPoint()	CCGeometry.cpp:31
65	0.15%	1	__truncdfsf2	ieee754-df.S:1382
63	0.15%	1	joy::cocos2d::CCNode::nodeToParentTransform()	CCNode.cpp:1106
55	0.13%	1	joy::cocos2d::ccpNormalize(const joy::cocos2d::CCPoint&)	CCPointExtension.cpp:49
52	0.12%	1	joy::cocos2d::CCPoint::setPoint(float, float)	CCGeometry.cpp:53
50	0.12%	1	kmMat4Multiply	mat4.c:218
50	0.12%	1	__aeabi_fsub	ieee754-sf.S:59
40	0.10%	2	__aeabi_dadd	ieee754-df.S:86
39	0.09%	1	__fixunssfsi	ieee754-sf.S:1030
27	0.06%	2	joy::cocos2d::CCParticleSystem::update(float)	CCParticleSystem.cpp:589



iOS®: Xcode® OpenGL® Frame Capture



iOS®: Xcode® Profiler



The screenshot shows the Xcode Instruments interface with the Time Profiler selected. The top bar indicates the target is 'cpp-tests iOS' and the run time is 00:01:29. The main area displays a call tree for the selected function. The table below represents the data shown in the call tree.

Running Time	Self	Symbol Name	
38889.0ms	54.6%	38889.0	cocos2d::Renderer::drawBatchedQuads() cpp-tests iOS
4817.0ms	6.7%	4817.0	cocos2d::Renderer::render() cpp-tests iOS
3126.0ms	4.3%	3126.0	-[CCEAGLView swapBuffers] cpp-tests iOS
3031.0ms	4.2%	3031.0	cocos2d::QuadCommand::useMaterial() const cpp-tests iOS
2949.0ms	4.1%	2949.0	kmVec3MultiplyMat4 cpp-tests iOS
2464.0ms	3.4%	2464.0	cocos2d::Node::visit(cocos2d::Renderer*, kmMat4 const&, bool) cpp-tests iOS
1811.0ms	2.5%	1811.0	cocos2d::GLProgram::updateUniformLocation(int, void const*, unsigned int) cpp-tests iOS
1464.0ms	2.0%	1464.0	kmGLPushMatrix cpp-tests iOS
1432.0ms	2.0%	1432.0	cocos2d::Sprite::draw(cocos2d::Renderer*, kmMat4 const&, bool) cpp-tests iOS
1139.0ms	1.6%	1139.0	std::_1::vector<cocos2d::Node*, std::_1::allocator<cocos2d::Node*> >::empty() const cpp-tests iOS
837.0ms	1.1%	0.0	main cpp-tests iOS
818.0ms	1.1%	818.0	kmMat4Assign cpp-tests iOS
657.0ms	0.9%	657.0	cocos2d::GLProgram::setUniformsForBuiltin(kmMat4 const&) cpp-tests iOS
652.0ms	0.9%	652.0	cocos2d::QuadCommand::init(float, unsigned int, cocos2d::GLProgram*, cocos2d::BlendFunc, cocos2d::V3F_C4B_T2F_Quad*, long, kmMat4 const&) cpp-tests iOS
545.0ms	0.7%	545.0	cocos2d::GLProgram::setUniformLocationWithMatrix4fv(int, float const*, unsigned int) cpp-tests iOS
377.0ms	0.5%	377.0	cocos2d::Renderer::convertToWorldCoordinates(cocos2d::V3F_C4B_T2F_Quad*, long, kmMat4 const&) cpp-tests iOS
367.0ms	0.5%	367.0	kmGLGetMatrix cpp-tests iOS
354.0ms	0.4%	354.0	DYLD-STUB\$\$glDrawElements cpp-tests iOS
334.0ms	0.4%	334.0	cocos2d::Device::getTextureDataForText(char const*, cocos2d::FontDefinition const&, cocos2d::Device::TextAlign, int&, int&) cpp-tests iOS
324.0ms	0.4%	324.0	cocos2d::QuadCommand::generateMaterialID() cpp-tests iOS
308.0ms	0.4%	308.0	cocos2d::BlendFunc::operator==(cocos2d::BlendFunc const&) const cpp-tests iOS
283.0ms	0.3%	283.0	cocos2d::RenderQueue::push_back(cocos2d::RenderCommand*) cpp-tests iOS
266.0ms	0.3%	266.0	lazyInitialize cpp-tests iOS
258.0ms	0.3%	258.0	km_mat4_stack_push cpp-tests iOS
256.0ms	0.3%	256.0	cocos2d::Renderer::addCommand(cocos2d::RenderCommand*) cpp-tests iOS
248.0ms	0.3%	248.0	cocos2d::Director::drawScene() cpp-tests iOS
165.0ms	0.2%	165.0	cocos2d::GL::blendFunc(unsigned int, unsigned int) cpp-tests iOS

Lessons Learned

Lessons Learned

- ★ Profile everything... including recommended best practices
- ★ Profile everything... in different GPUs / GL drivers
- ★ Profile everything... use the same environment when comparing results
- ★ Profile everything... have a comprehensive test suite

If you can't measure it, you can't tell if it is getting better... or worse.

For further information...



★ <http://www.cocos2d-x.org>



Thank You

Questions?

@ricardoquesada

The trademarks featured in this presentation are registered and/or unregistered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. Any other marks featured may be trademarks of their respective owners