Optimized Effects for Mobile Devices

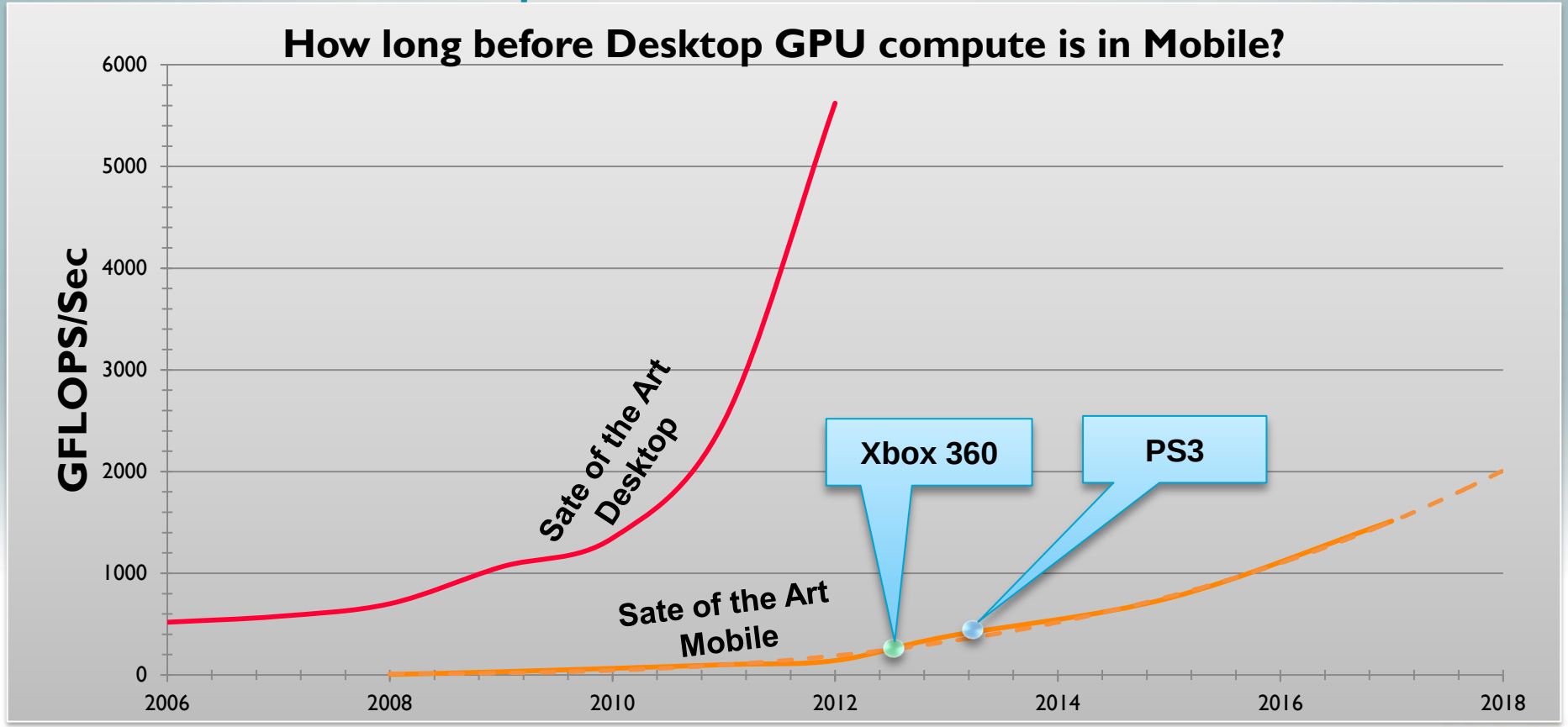
Ed Plowman, Director of Performance Analysis, ARM
Stacy Smith, Senior Software Engineer, ARM

Grab Your Crystal Balls

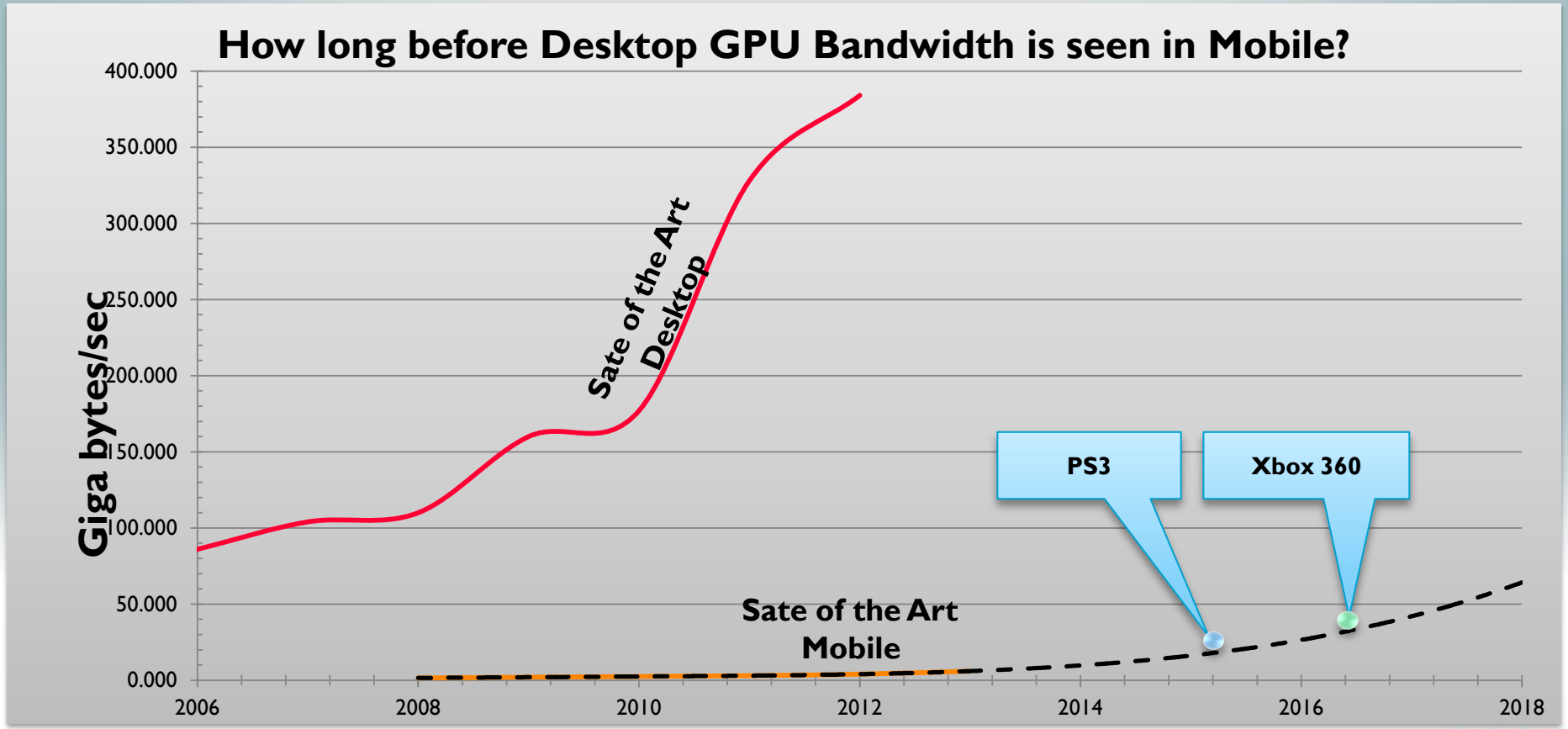
- Top 3 questions I get asked:
  - Q. What does the future of mobile content look like?
  - A. That depends on how much GPU capability you have?
  - Q. How much performance will content developers need?
  - A. As much as you can give them!
  - Q. When will the mobile reach console quality?

- Well, lets take a look at that and see if we can answer the others along the way…

Mobile GPU Compute Year On Year



Mobile GPU BW Growth Year on Year



Why is BW Not Progressing as Fast?

- Simple… Power!
- I did the graph, but desktop GPU power is too horrific!

- Desktop = 170 Watts to >300 Watts… that's just the GPU!
- Console = 80-100 Watts (CPU/GPU/WiFi/Network)
- Mobile Platform = 3 - 7 Watts (CPU/GPU/Modem/WiFi)!

How to Get 100W of Work from 3Watts?

"I believe the sign of maturity is accepting deferred gratification." - Peggy Cahn

- Five main suppliers of GPU tech in mobile
- Three are deferred renderers
- And those three make up >90% of the volume
- This is not a coincidence!
- Deferred rendering is most efficient GPU tech for Mobile
  - Efficiency of BW, HW resource and Power
- Getting the most from it requires slightly different thinking…

Thinking in a Deferred World…

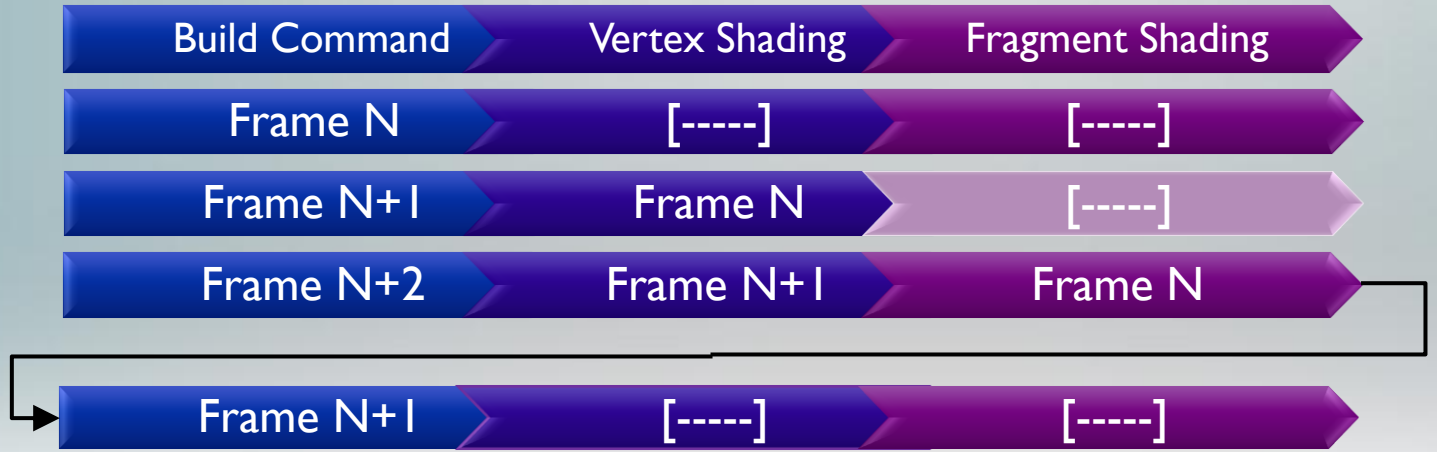
- Minimize draw calls and state changes
  - Draw Calls/API calls are not free use them wisely
  - Grouping draw calls with like state = good
- But… don't go crazy
  - Large object batches with high potential occlusion can be costly
  - Remember those vertices still need processing
- Draw Target Bind/unbind on each draw call = bad
  - Seen (disappointingly) in a lot of commercial engines
  - Can cause flush and reload cycles of tile/cache memory
  - Bind it once, issue all draw calls, unbind it…
  - Hint: Take a look at the use of glDiscardFramebufferEXT()
  - Indicates to driver that render attachment is done with/complete

Thinking in a Deferred World…

- Use Vertex Buffer Objects
  - Client side vertex buffers use copy on write (CoW) on each Draw Call
  - VBO's don't, so they provide a considerable performance increase
  - Avoid dynamic VBO or IBO updates using glBufferSubData()
- Multiple Render Targets (New for OpenGL® ES 3.0)
  - Very efficient on deferred GPU
  - Make sure sum of bits/frag is "do-able" "in tile" for max performance
  - Different criteria for each GPU provider

Avoiding Blocking Behaviours

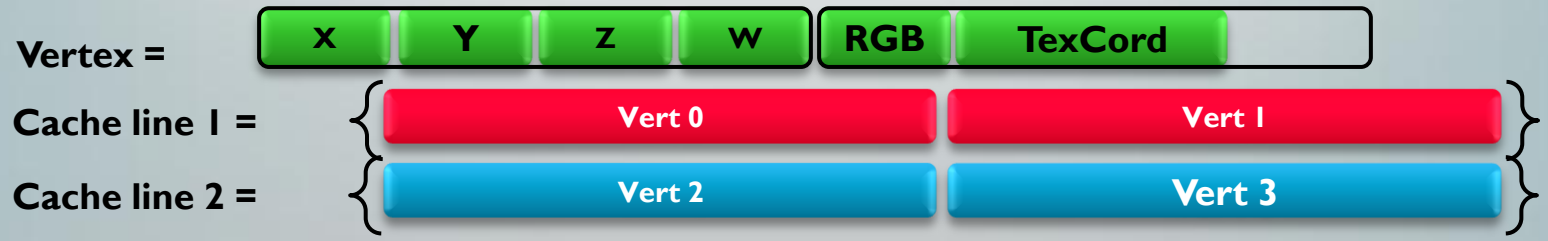
- Deferred GPU's use a pipeline



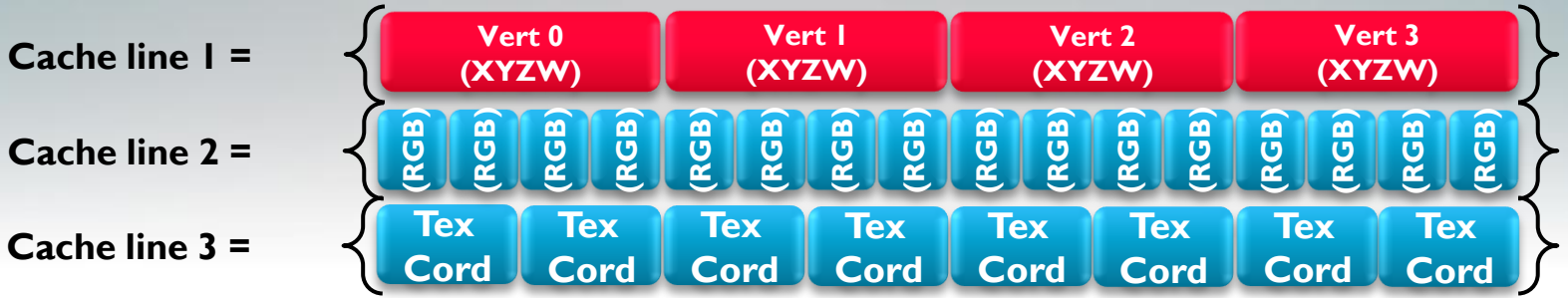
  - glReadPixels(), glCopyTexImage(), glTexSubImage() = bad…
  - If you must use glReadPixels use PBO's
  - Use FBO instead of glCopyTexImage()
  - Also Occlusion Query (OpenGL ES 3.0) - Results delayed by 1-2 frames
  - Busy waiting on OQ bad idea!

Make Every Access Count

- Think about "cacheability" of data



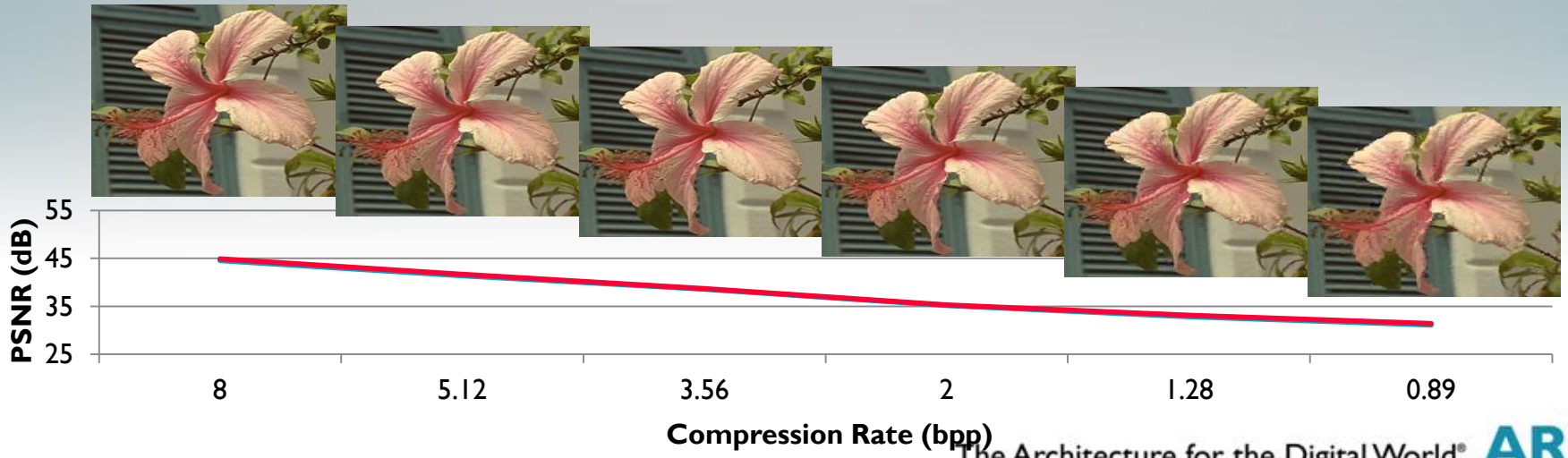
- De-interleave vertex data



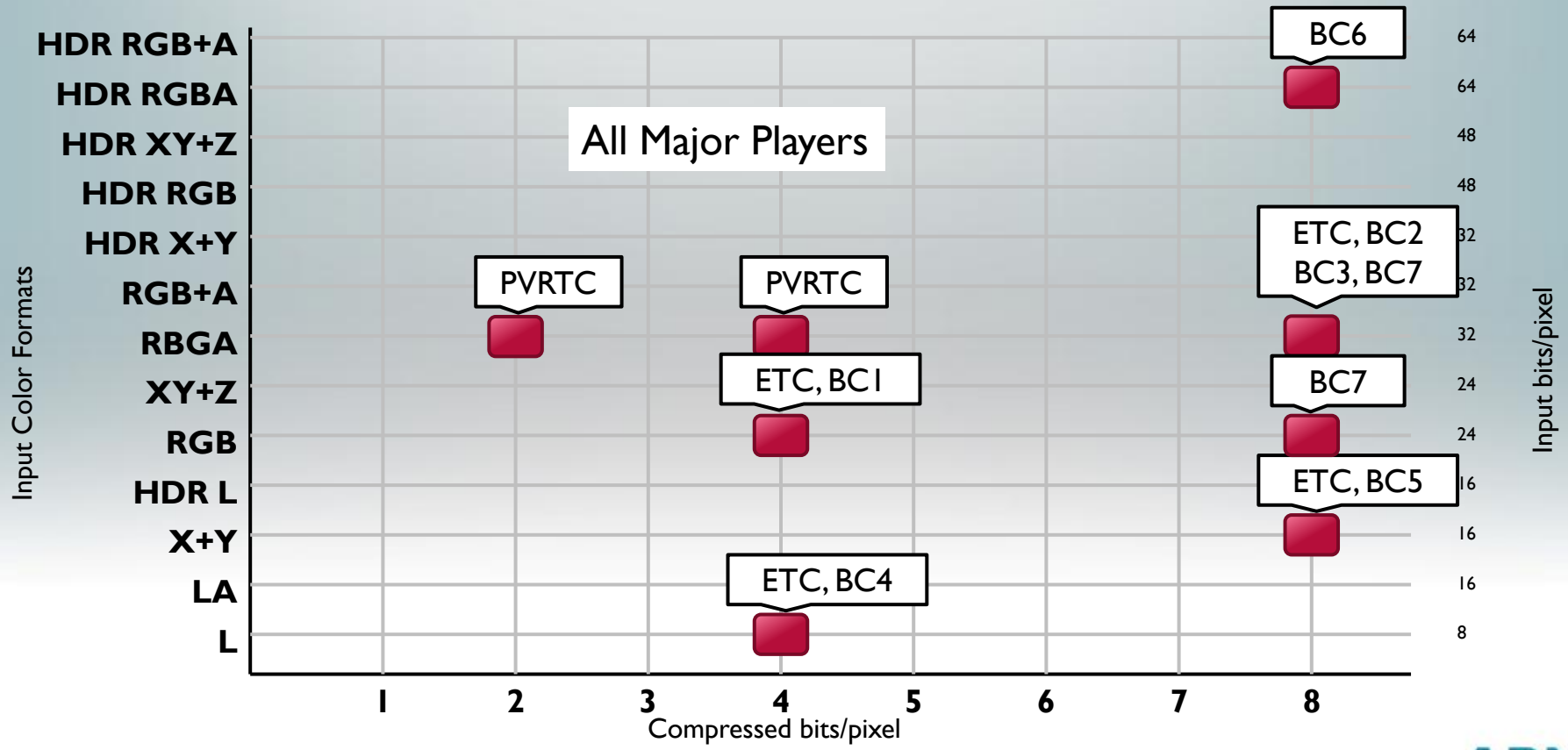
- Think about representation
  - Do you really need a FP32/component for a texture coordinates accessing a 512x512 texture?

Compress, Compress, Compress!

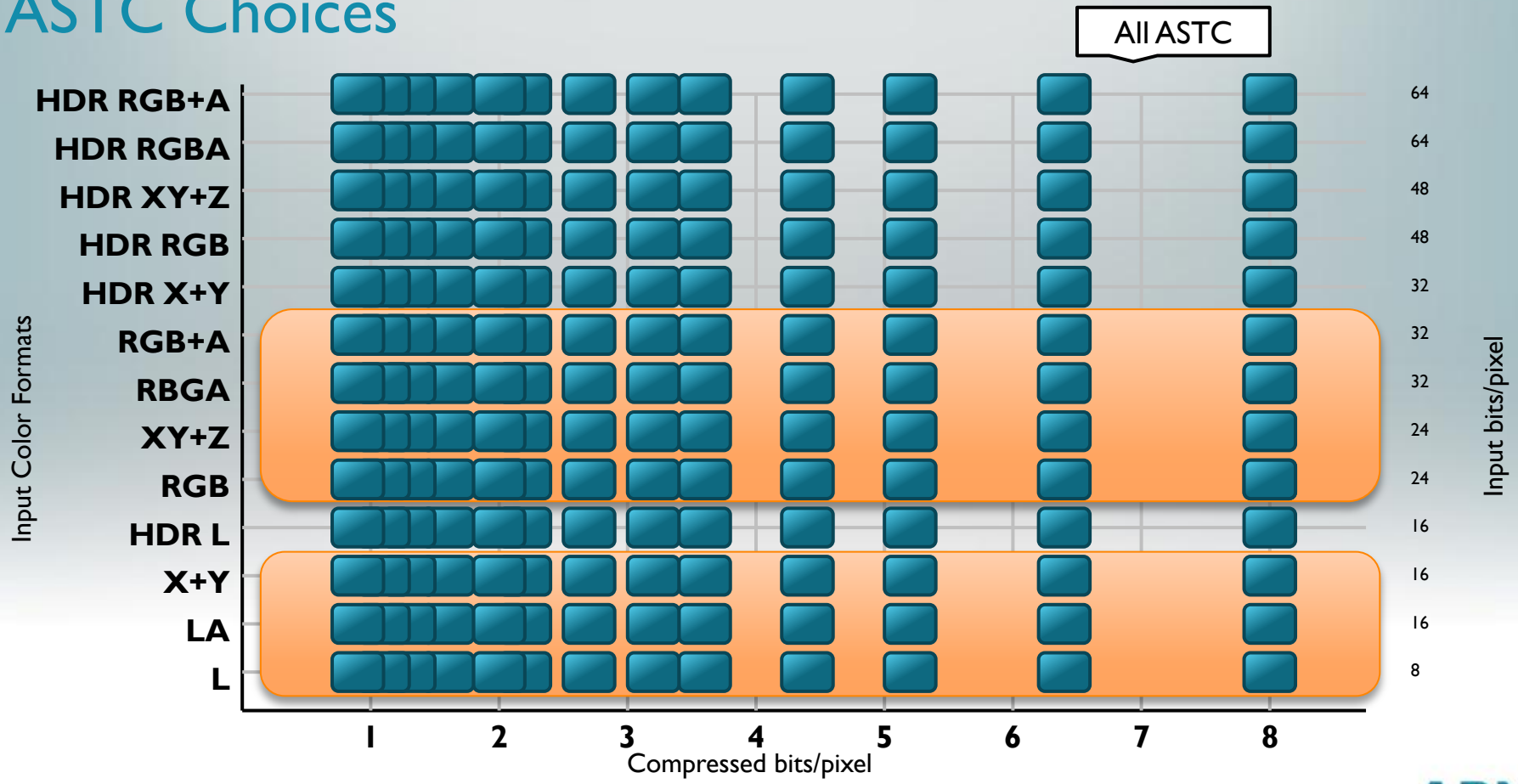
- **ASTC = Adaptive Scalable Texture Compression**
  - New texture compression standard developed by ARM, adopted by Khronos
    - KHR_texture_compression_astc_ldr for OpenGL ES and Open GL
  - Increased quality and fidelity at low bit-rates
  - Expansive range of input formats offers complete flexibility
    - Choice of base format, 2D and 3D plus addition of HDR formats



Compression in the Pre-ASTC World



ASTC Choices



Look it up or Calculate it?

- You would be surprised what you can get done in a cycle…

```
precision mediump float;
varying vec4 detailtc_envtc, bumptrans;
uniform sampler2D dettex, envtex, colormap;
uniform float color_param, bumpstrength;
void main()
{
   vec4 bt = bumptrans;
   vec2 bt_crossmul = bt.xy * bt.wz;
   float diffuse = max(0.0, bt_crossmul.x-bt_crossmul.y);
   vec4 bump_cr = texture2D(dettex,detailtc_envtc.xy);
   vec4 tbump = bt * bumpstrength * bump_cr.xyxy;
   vec2 envtc = tbump.xy + tbump.zw + detailtc_envtc.zw;
   vec4 col = texture2D(colormap, vec2(bump_cr.z, color_param));
   vec4 env = texture2D(envtex,envtc);
   gl_FragColor = col * diffuse + env * bump_cr.w;
}
```

Shader Features:
- **Paletted color mapping**
- **Environment mapping**
- **Bump mapping**
- **Variable reflectance mapping**
- **Diffuse falloff of the texture color**
- **Adjustable bump map strength**
- **Adjustable color table**

Mali-T600 series = 3 Cycles

GDC 2012 Demo: Timbuktu

Asset Conditioning

- **Cross platform** - desktop & mobile
- **Desktop build** - caching
- **Mobile build** - loads caches
- **Asset pipeline** - utility functions

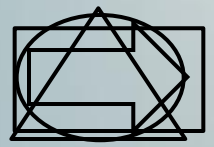
Batching

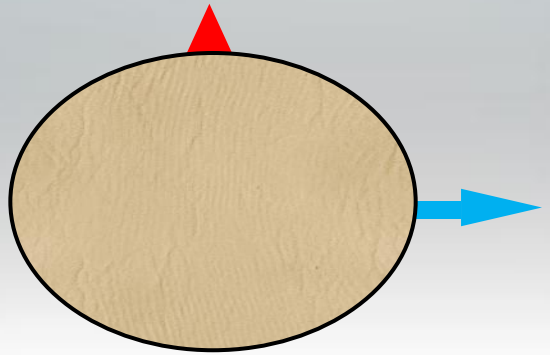
- Deferred immediate mode rendering
- glDrawElements and glDrawArrays have an overhead
- Less draw calls, less overhead.
- DrawCall class stitches multiple objects into one draw
- Macro functions in shaders make batching as simple as:

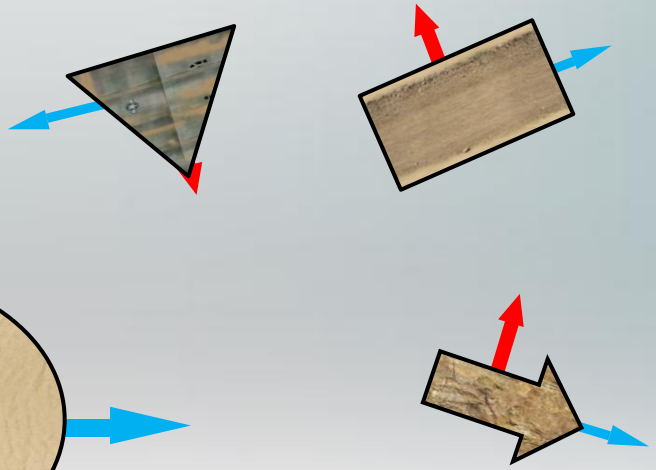
```
vec4 pos=transform[getInstance()]*getPosition();
```

Batching

Batching







uniform mat4 transforms[4];

Object Instancing

- Multiple geometries, or single object instances:

```
for(int i=0;i<50;i++)
  drawbuilder.addGeometry(geo1);
drawbuilder.Build()
```

- Can implement LOD switching, when objects are sorted front to back and correctly culled.
- Seen in TrueForce

Special Effects

Special Effects: Bloom

Special Effects: Bloom

- When considering uses of greater colour resolution the first thought was HDR and Bloom.

- But how to do bloom without the HDR images?

Special Effects: Bloom

Special Effects: Bloom

Render to low res FBO mapped to texture

Value filter and blur in 1st post- processing pass, onto second FBO texture

Sample vertical blur in second pass then apply to full resolution frame buffer

Special Effects: Depth of Field

Special Effects: Depth of Field

- 16bit depth buffers as textures opened the possibility of a variable blur for depth of field

- But how to do it without 16bit textures?

Special Effects: Depth of Field

Special Effects: Depth of Field

Mix

Bloom

Additive

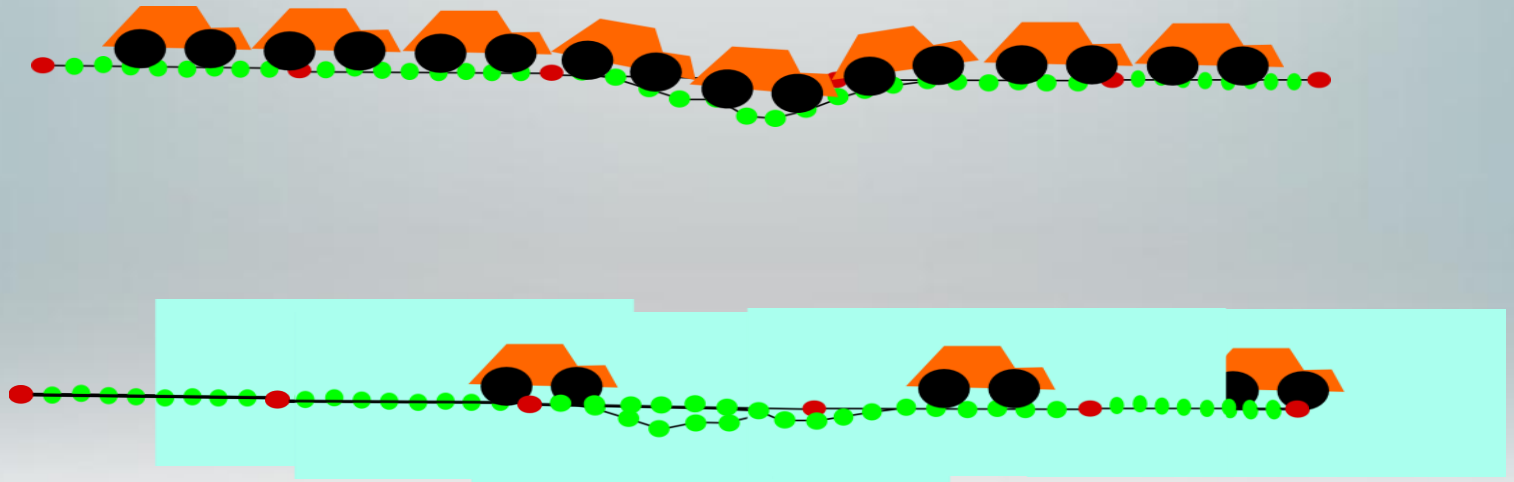
Special Effects: Terrain Mapping

Special Effects: Terrain Mapping

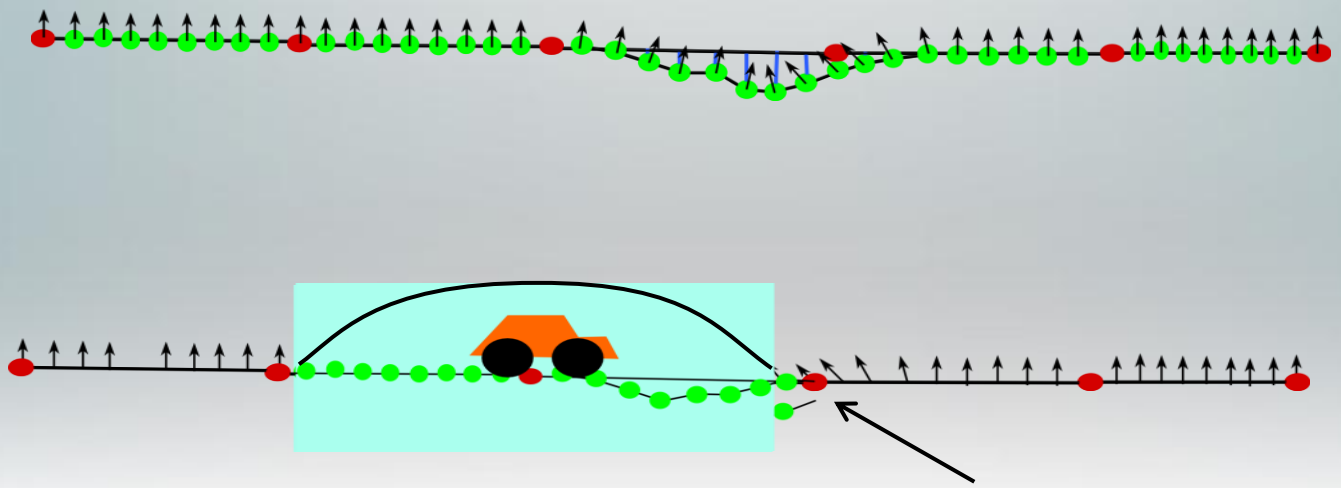
- Uniform Buffers and Vertex IDs can be used to implement tessellated mesh subdivision

- But how can this be approximated without the buffers or IDs?

Special Effects: Terrain Mapping



Special Effects: Terrain Mapping



SIGGRAPH 2012 Demo: Timbuktu 2

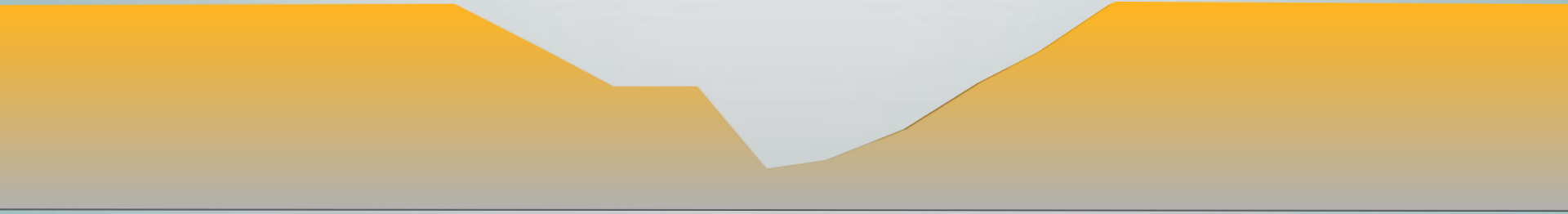
Timbuktu 2: Extended features

- OpenGL® ES 3.0!
- 3D textures
- Shadow comparison
- 16 bit depth textures
- HDR lighting

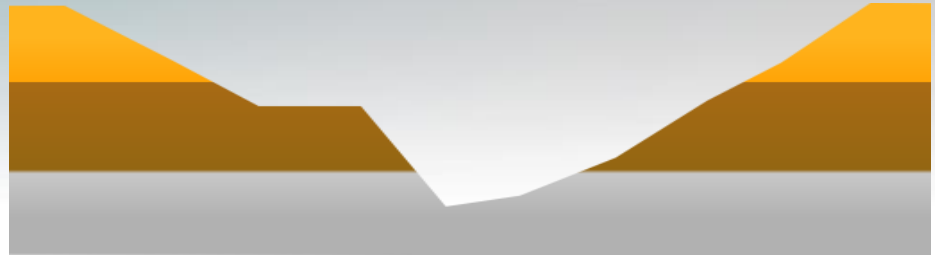
3D Textures

- Give more definition to deformed track
- 3D Textures mipmap in all 3 dimensions
- Instead used 2D Texture arrays

3D Textures



texture2DArray



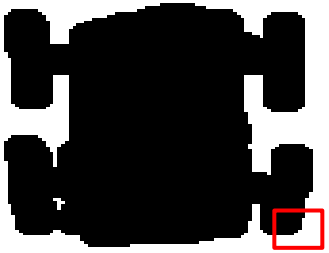
floor(z)
fract(z)
ceil(z)

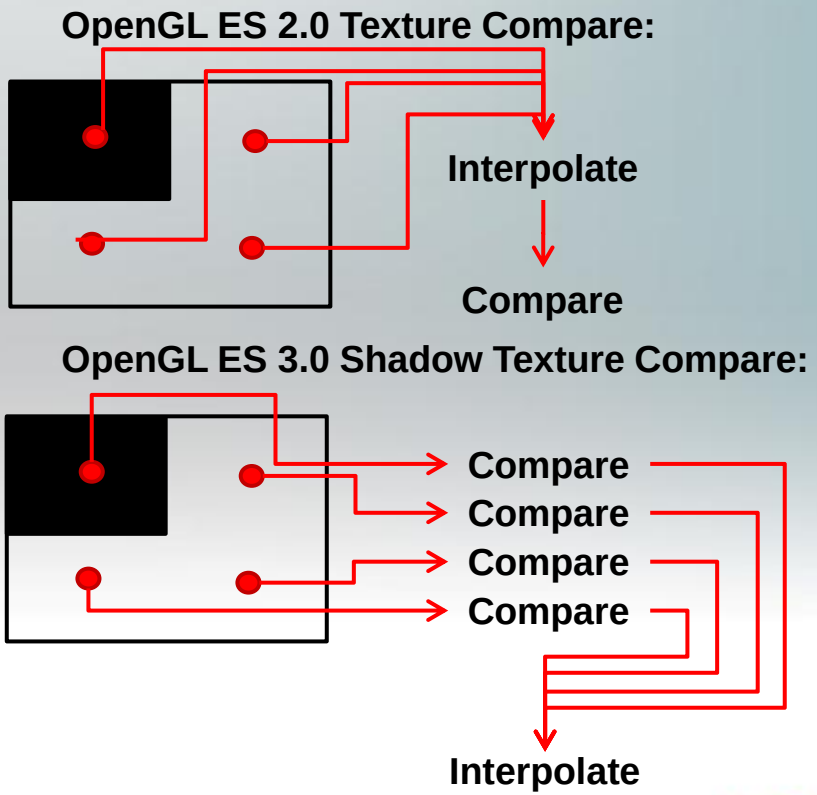
mix(t1, t2, frac)

Shadow Mapping

- Depth rendered to FBO from viewpoint of light
- Projected onto scene to compare to distance
- Doing this in the shader yields some interesting results

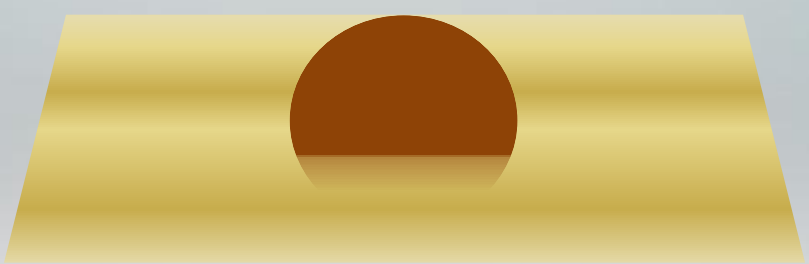
Shadow Comparison Texture





16 Bit Depth Buffers

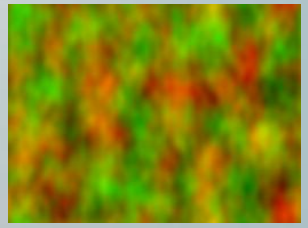
- Also used for:
- Soft Particles



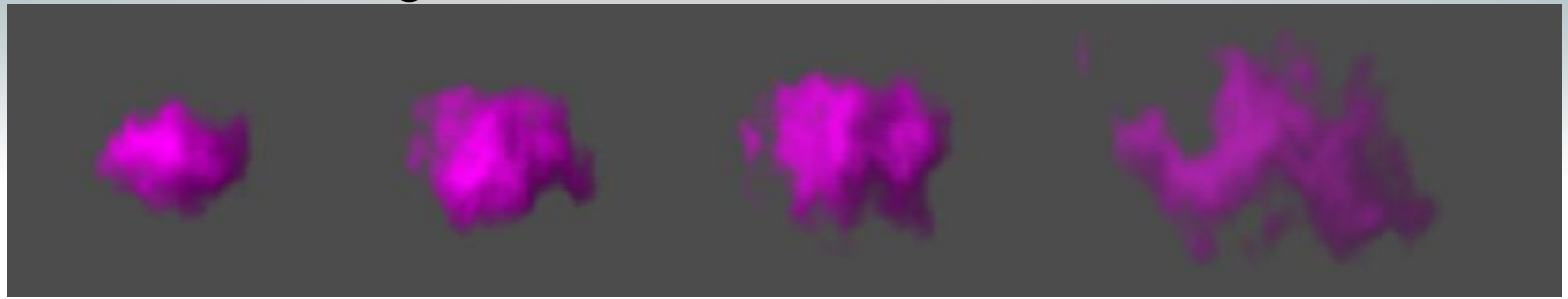
- Better fidelity of DOF

Particle Lighting

- Displacement Mapping:

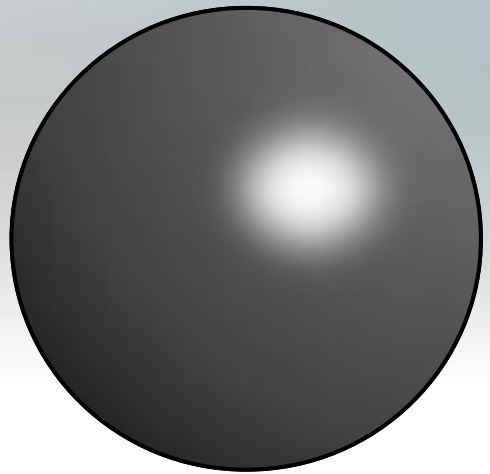


- Texture offset to X and Y coords
- Distortion strength increases over time



HDR Lighting

- RGBA 10 10 10 2 format used
- Everything gets normalised
- Bright spots need to stand out
- Make everything else darker!
- Set exposure in post processing



Main conference ARM Sponsored Sessions

Thursday, March 28th	
10:00 – 11:00am **Room 3022**	**Optimized Effects for Mobile Devices** Stacy Smith, Senior Software Engineer, ARM Ed Plowman, Director of Performance Analysis, ARM
1:00 – 2:00pm **Room 3016**	**The Future of Mobile Gaming** **PANEL SESSION** Moderator: Jason Della Rocca, Co-founder of Execution Labs Baudouin Corman, Vice President of Publishing, Americas for Gameloft David Helgason, CEO, Unity Dr Chris Doran, Founder & COO, Geomerics Niccolo De Masi, CEO, Glu Jasper Smith, CEO, PlayJam Nizar Romdhane, Director of Ecosystem, ARM

ARM #1124 in-booth Educational Theater

- ✓ Over 30 talks from ARM and partners such as EpicGames, Havok, PlayJam, Geomerics, Softkinetics, Metaio, Marmalade
- ✓ 20 minute length talks with Q&A at the end
- ✓ An Android tablet prize draw at each session
- ✓ Summary and videos of all Educational Theater Talks at http://malideveloper.arm.com/gdc2013

Thank you
Any questions?

malideveloper.arm.com