Variable Penumbra Soft Shadows for Mobile Devices

Alun Evans   Javi Agenjo   Josep Blat

Universitat Pompeu Fabra

Barcelona Spain

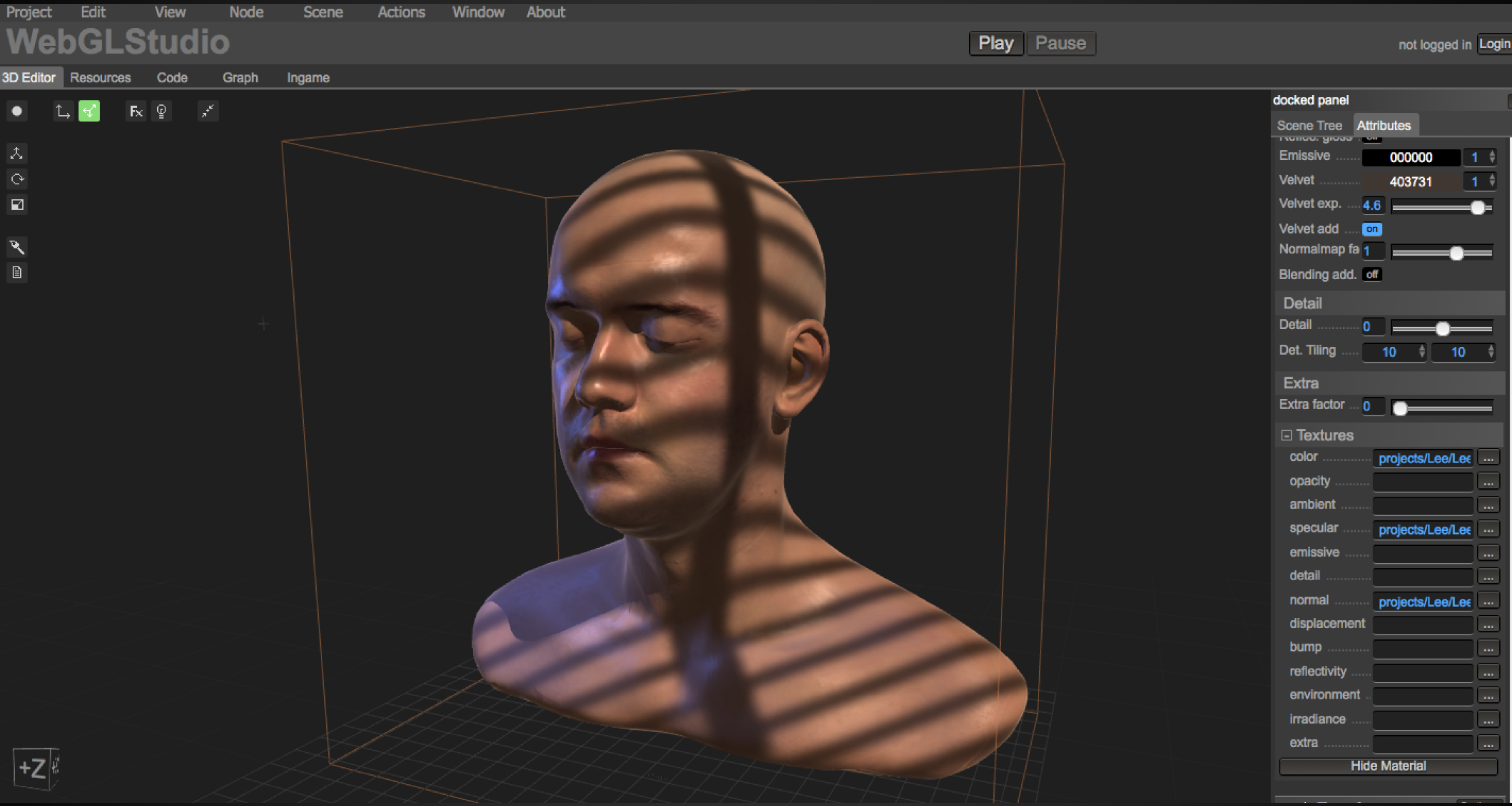
IMP-ART

upf. Universitat Pompeu Fabra Barcelona

GTI Interactive Technologies Group

Oh no, not another talk on shadow maps

Didn’t we solve this all 5 years ago?!



Emissive 000000 1
Velvet 403731 1
Velvet exp. 4.6
Velvet add on
Normalmap fa 1
Blending add. off

Detail

Detail 0
Det. Tiling 10 10

Extra

Extra factor 0

Textures

color	projects/Lee/Lee
opacity	
ambient	
specular	projects/Lee/Lee
emissive	
detail	
normal	projects/Lee/Lee
displacement	
bump	
reflectivity	
environment	
irradiance	
extra	

J.Agenjo, A.Evans,J.Blat.
WebGL Studio - a Pipeline for WebGL Scene Creation.
Web3D 2013, San Sebastian, Spain, (2013)

Uber Shader

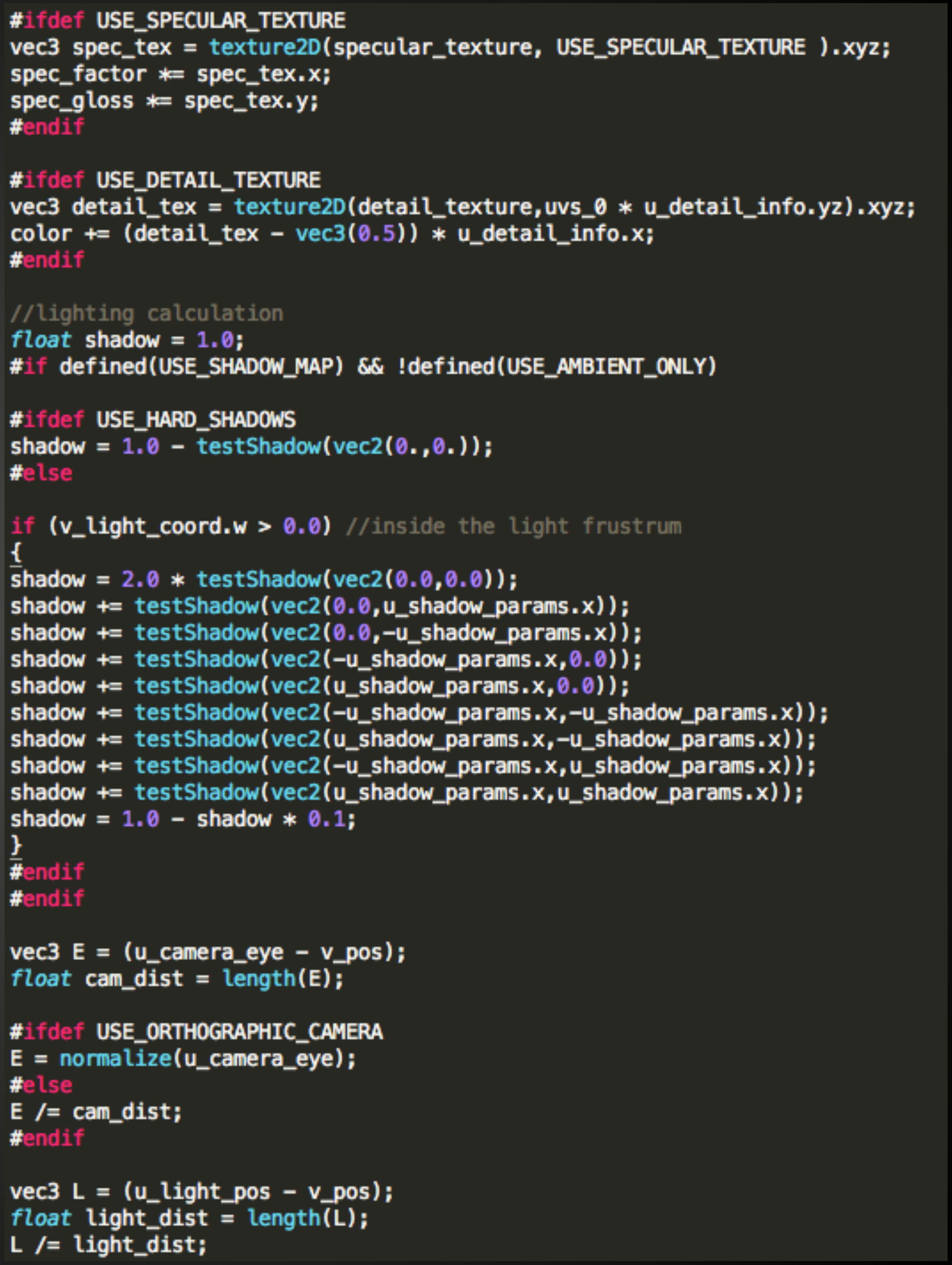
One large shader

Macros wrapping effects code

inject #defines just before compiling

WebGLStudio running on iPad - but crash when enabling soft shadows!

Why?



```
#ifdef USE_SPECULAR_TEXTURE
vec3 spec_tex = texture2D(specular_texture, USE_SPECULAR_TEXTURE ).xyz;
spec_factor *= spec_tex.x;
spec_gloss *= spec_tex.y;
#endif

#ifdef USE_DETAIL_TEXTURE
vec3 detail_tex = texture2D(detail_texture,uvs_0 * u_detail_info.yz).xyz;
color += (detail_tex - vec3(0.5)) * u_detail_info.x;
#endif

//lighting calculation
float shadow = 1.0;
#if defined(USE_SHADOW_MAP) && !defined(USE_AMBIENT_ONLY)

#ifdef USE_HARD_SHADOWS
shadow = 1.0 - testShadow(vec2(0.,0.));
#else

if (v_light_coord.w > 0.0) //inside the light frustrum
{
shadow = 2.0 * testShadow(vec2(0.0,0.0));
shadow += testShadow(vec2(0.0,u_shadow_params.x));
shadow += testShadow(vec2(0.0,-u_shadow_params.x));
shadow += testShadow(vec2(-u_shadow_params.x,0.0));
shadow += testShadow(vec2(u_shadow_params.x,0.0));
shadow += testShadow(vec2(-u_shadow_params.x,-u_shadow_params.x));
shadow += testShadow(vec2(u_shadow_params.x,-u_shadow_params.x));
shadow += testShadow(vec2(-u_shadow_params.x,u_shadow_params.x));
shadow += testShadow(vec2(u_shadow_params.x,u_shadow_params.x));
shadow = 1.0 - shadow * 0.1;
}
#endif
#endif

vec3 E = (u_camera_eye - v_pos);
float cam_dist = length(E);

#ifdef USE_ORTHOGRAPHIC_CAMERA
E = normalize(u_camera_eye);
#else
E /= cam_dist;
#endif

vec3 L = (u_light_pos - v_pos);
float light_dist = length(L);
L /= light_dist;
```

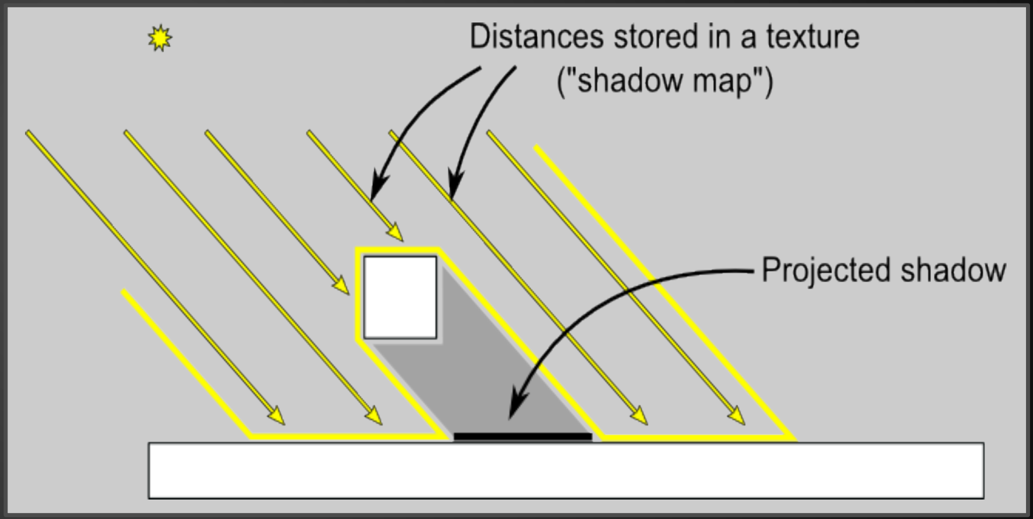
Shadowing 101

1st Pass

Render from point of view of light
Store distance-to-light of object in “Shadow Map”

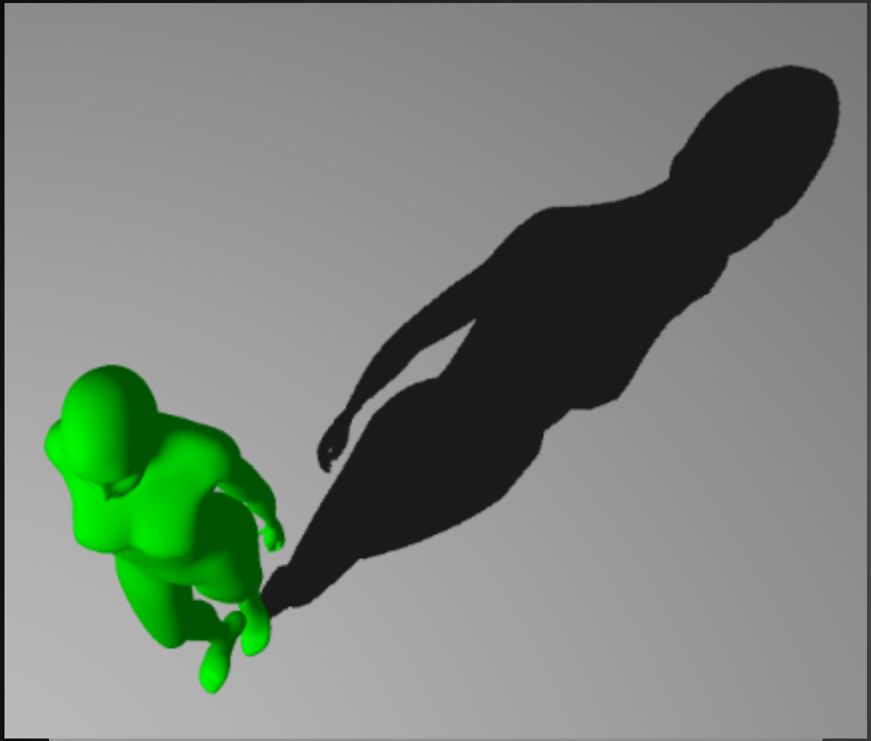
2nd Pass

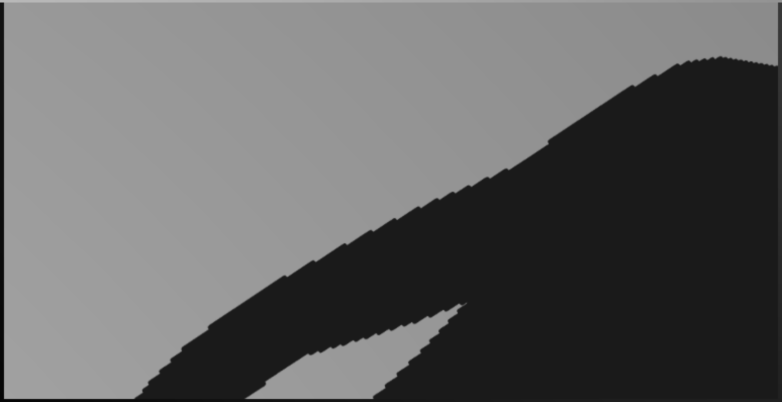
Compare distance-to-light of each fragment to stored distance
If new distance is greater, draw shadow

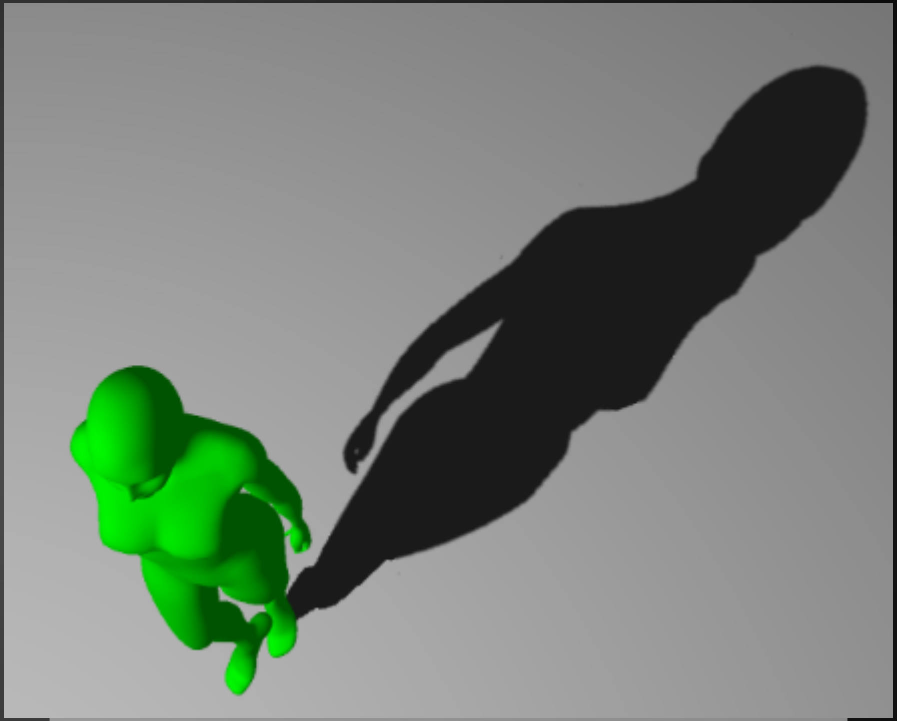


http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/

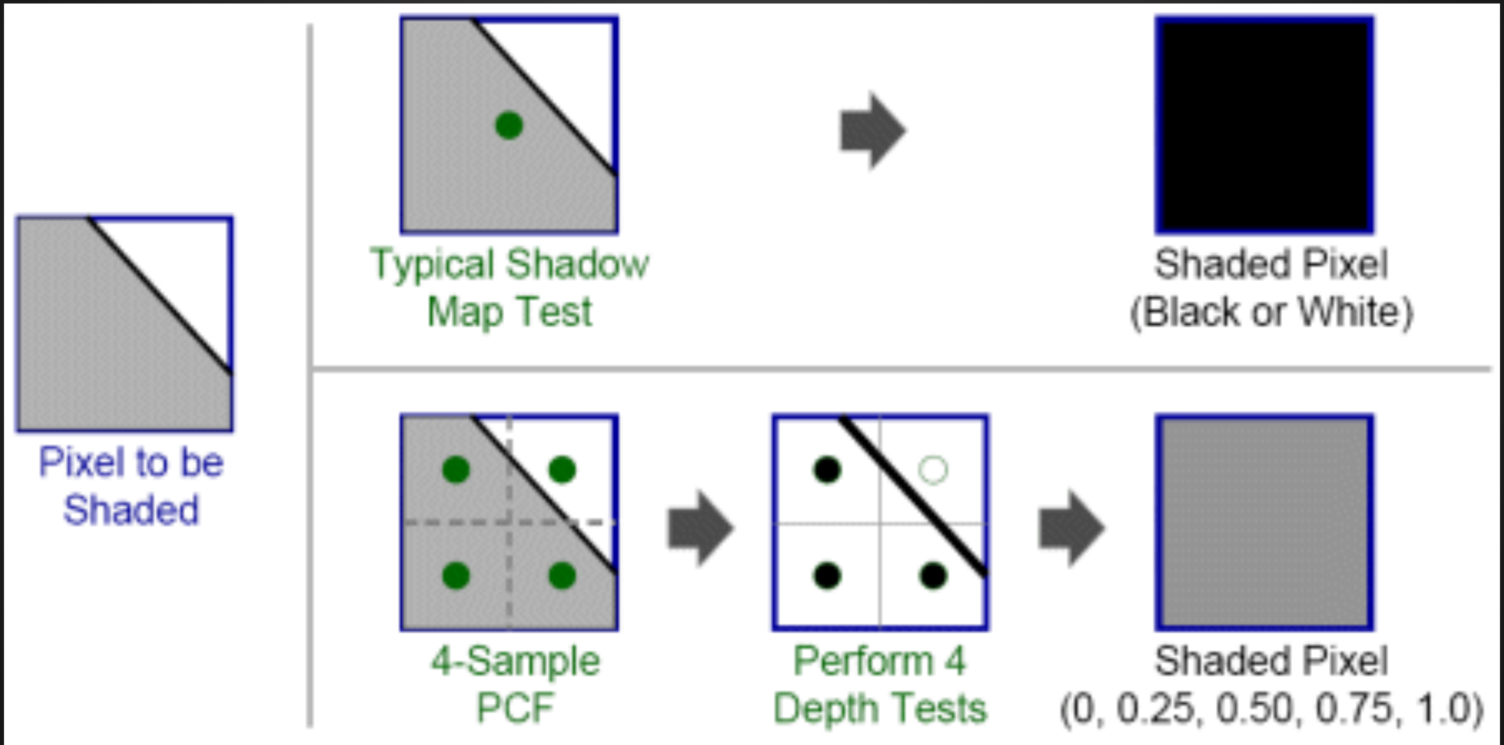
Hard vs Soft Shadows







Percentage Closer Filtering

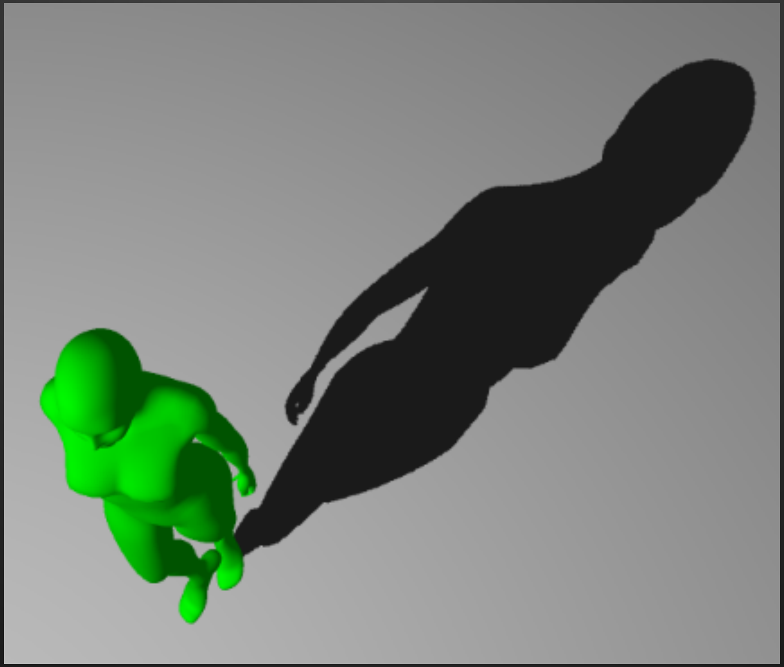


http://www.beyond3d.com/content/reviews/2/4

Poor Performance on iPad (3rd gen)

Native C/Objective-C OpenGL ES 2.0

Technique	Framerate
Hard Shadows	52 fps (19 ms/f)
Soft Shadows (4x4 PCF)	7 fps (142 ms/f)



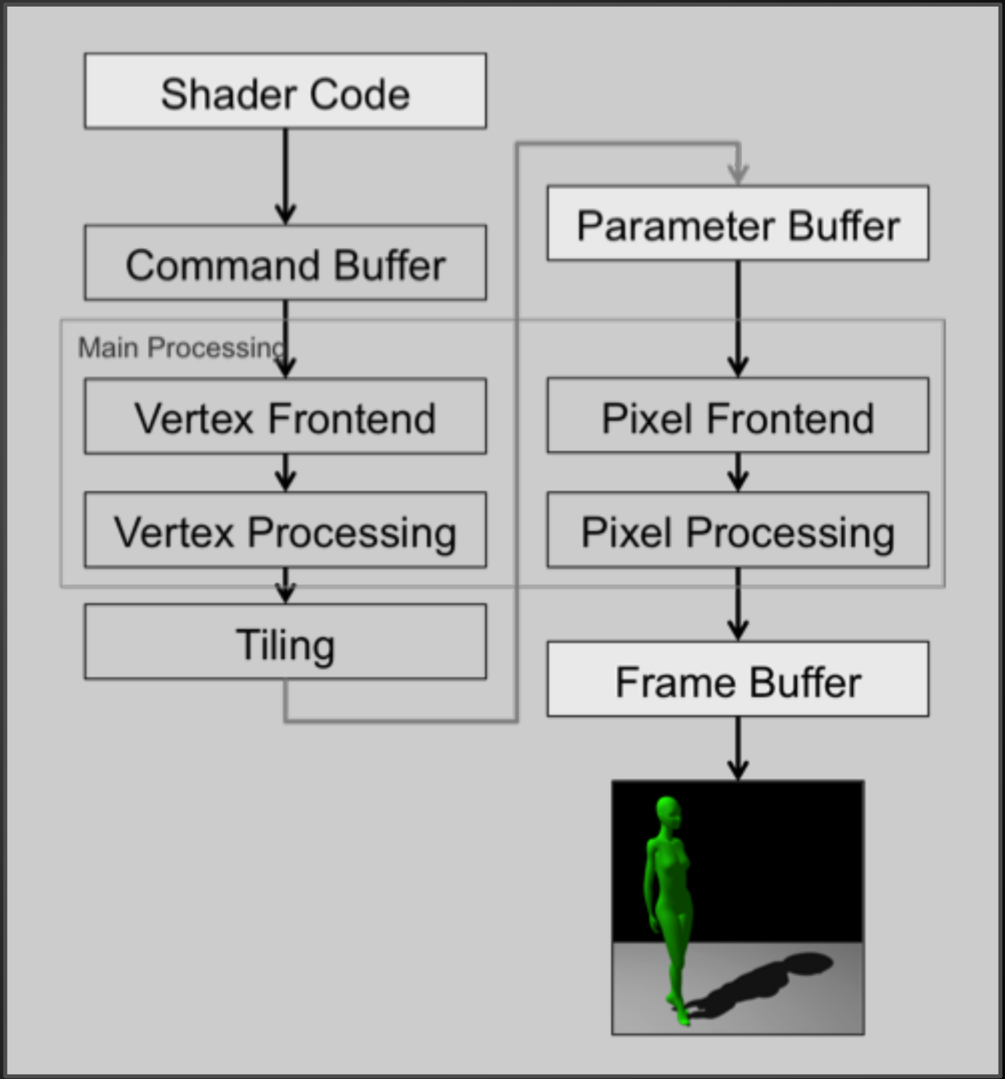
Tile-based (Deferred) Rendering

Mobile GPU architectures usually differ to desktop counterparts

Tile-based rendering renders portions (tiles) of screen separately - typically 16x16 or 32x32

Deferred rendering delays fragment processing until occlusions have been calculated

PowerVR SGX, ARM Mali



Mobile GPUs and shadow mapping

Dependent Texture Reads (where location of lookup in texture is calculated before reading) should be minimised

Calculations of texture coordinates in pixel shader are very expensive

BUT

PowerVR SGX can only pass 32 floats from Vertex to Pixel Shader

Mobile GPUs and vanilla PCF

4x4 kernel PCF involves 16 texture lookups.

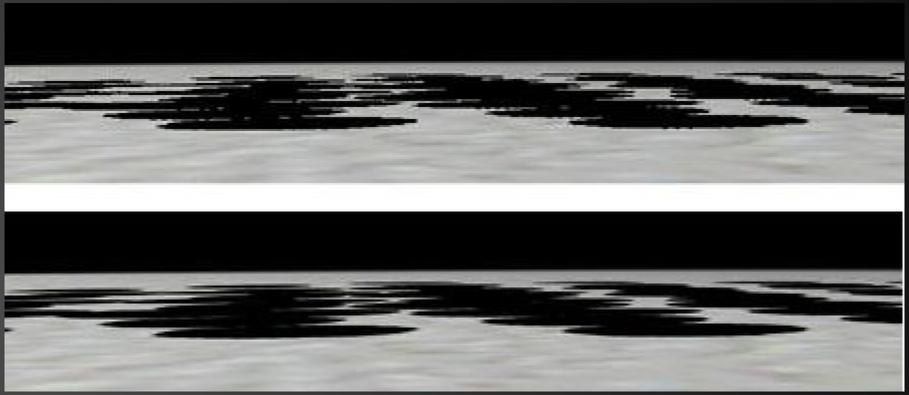
These (mostly dependent) texture-reads slow down the pixel shader to unusable framerates even with simple scenes

5x5 PCF involves 25 lookups and crashes hardware (iPad 3)

So PCF is too slow. What about...

Variance Shadow Maps

Uses hardware filtering and mean/variance of shadow data to smooth shadow edge



http://http.developer.nvidia.com/GPUGems3/gpugems3_ch08.html

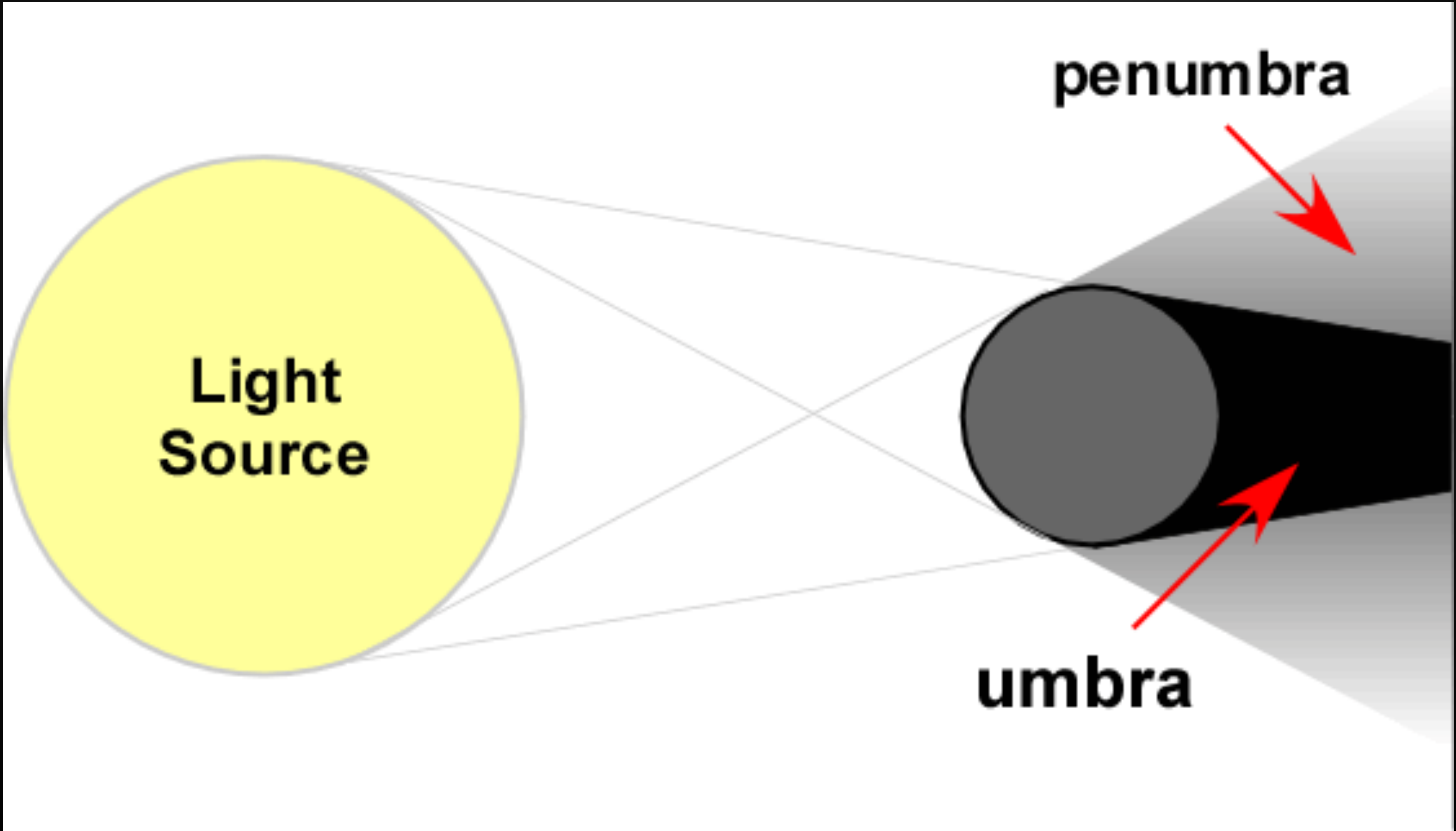
Can get it to work on ipad but with poor performance, and is scene dependent

Convolution Shadow Maps, Exponential Shadow Maps & other techniques involve computation in pixel shader too

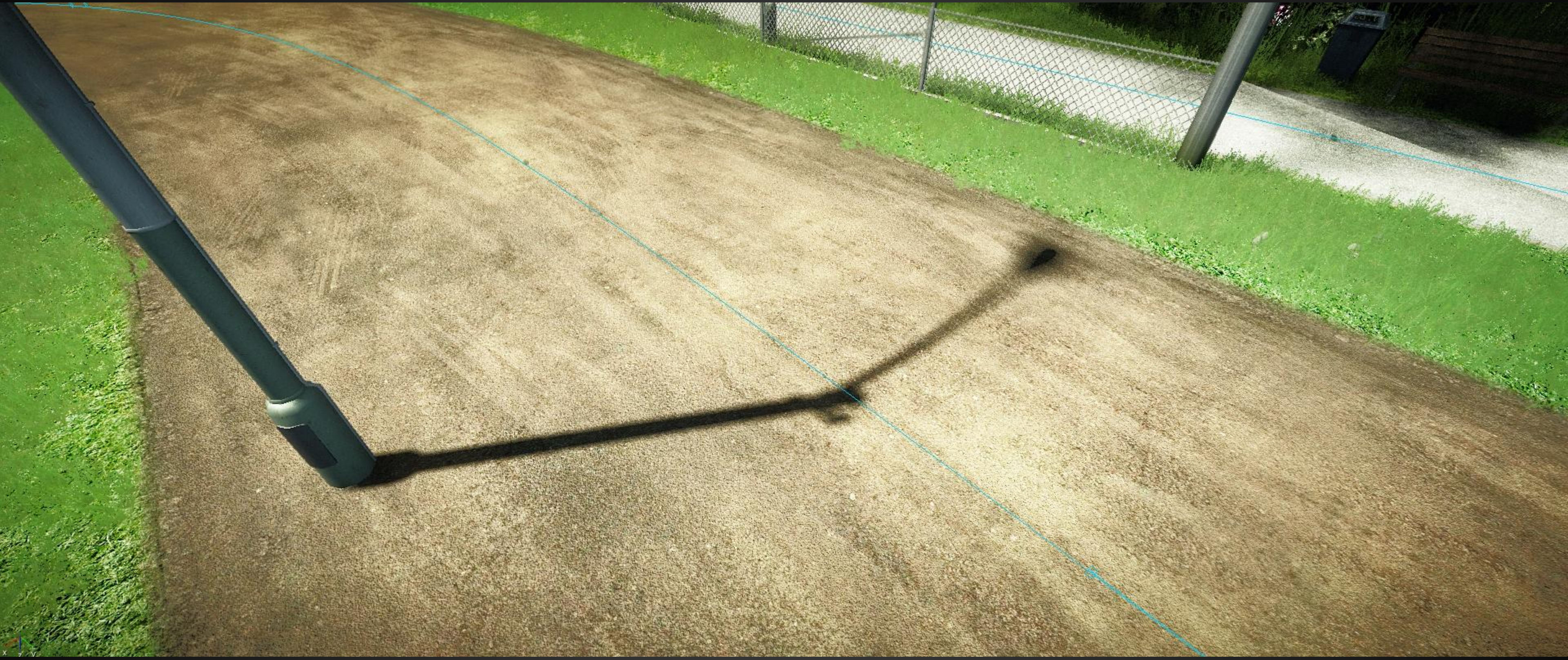
Challenge 1

Get some sort of soft shadows running on a mobile GPU (in this case, the ipad)

Umbra vs Penumbra



Penumbra - Variable vs Fixed



Crysis 2 - Crytek

Width of penumbra depends on width of light source, and distances between light and blocker, and blocker and shaded surface

This looks much more believable than a fixed penumbra!

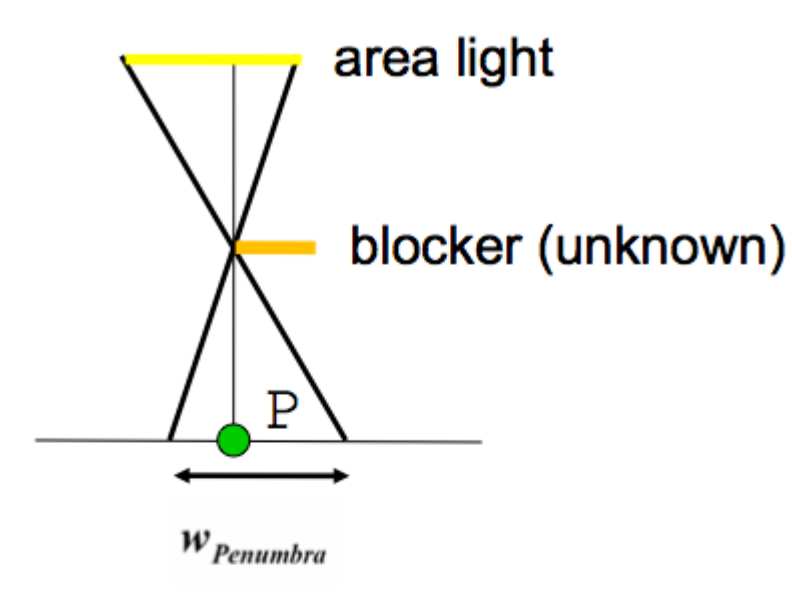
Percentage Closer Soft Shadows

Variable Penumbra Shadowing

Area light, parallel blocker, compute penumbra using similar triangles

Need to search for blockers, usually with a 4x4 or 5x5 kernel radius

Performance hit of blocker phase search



$$w_{Penumbra} = \frac{(d_{Receiver} - d_{Blocker}) \cdot w_{Light}}{d_{Blocker}}$$

Fernando, R., 2005. **Percentage-closer soft shadows.** In *ACM SIGGRAPH 2005 Sketches*

The challenge...

1) get basic PCF running on a mobile GPU with improved framerate

2) see if we can add a variable penumbra component to the shadow

PCF with better framerate

Edge-based shadow mapping

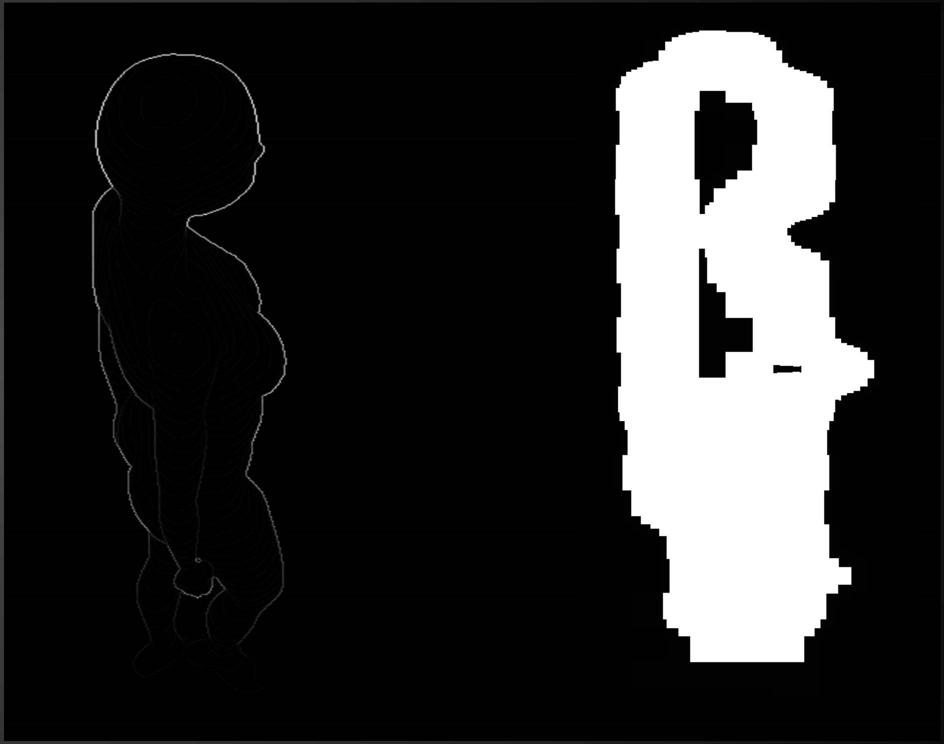
Use simple edge-detection filter on shadow map (GPU based)

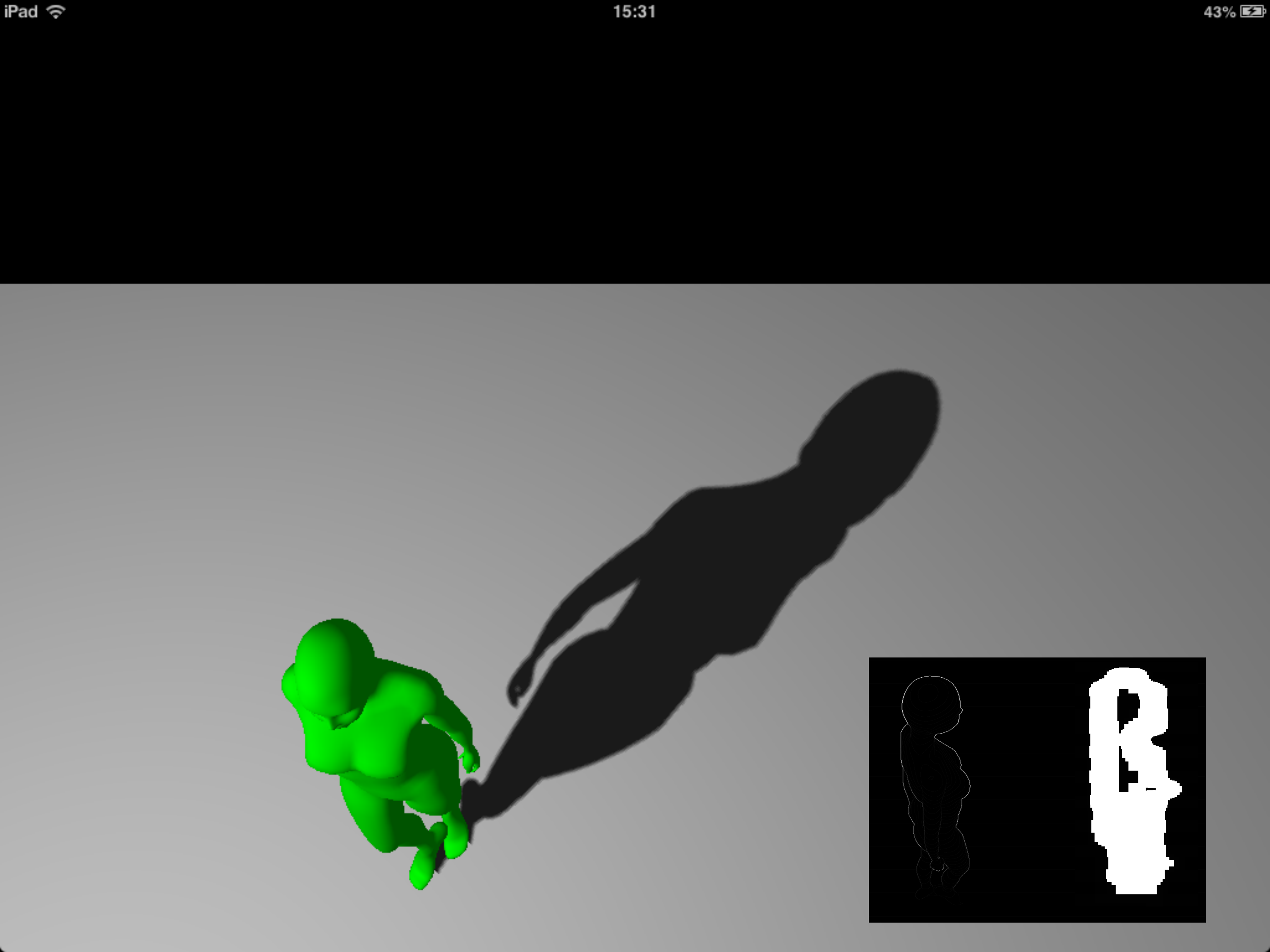
Use *mip-chain dilation* to expand edge by sampling higher level mipmap in shader

Effectively creates a “shadow mask” - only carry out PCF inside mask.

Outside mask still test hard shadows

Requires a conditional




iPad
15:31
43%

Results from test scene

Technique	Framerate
Hard Shadows	52 fps (19 ms/f)
PCF - 4x4 - unoptimised	7 fps (142 ms/f)
PCF - 4x4 - Shadow Mask	25 fps (40 ms/f)
PCF - 7x7 - Shadow Mask	11 fps (91 ms/f)

Not only does shadow mask improve performance, it also permits the use of a wider filter and therefore larger penumbra

Adding a variable penumbra

Quantize shadow edge

Strength of shadow edge is proportional to gradient of image, which is proportional to distance between surfaces

Blocker which is close to shading surface gives weak edge, whereas a blocker further away from shading surface gives strong edge

Mip chain dilation loses this edge strength

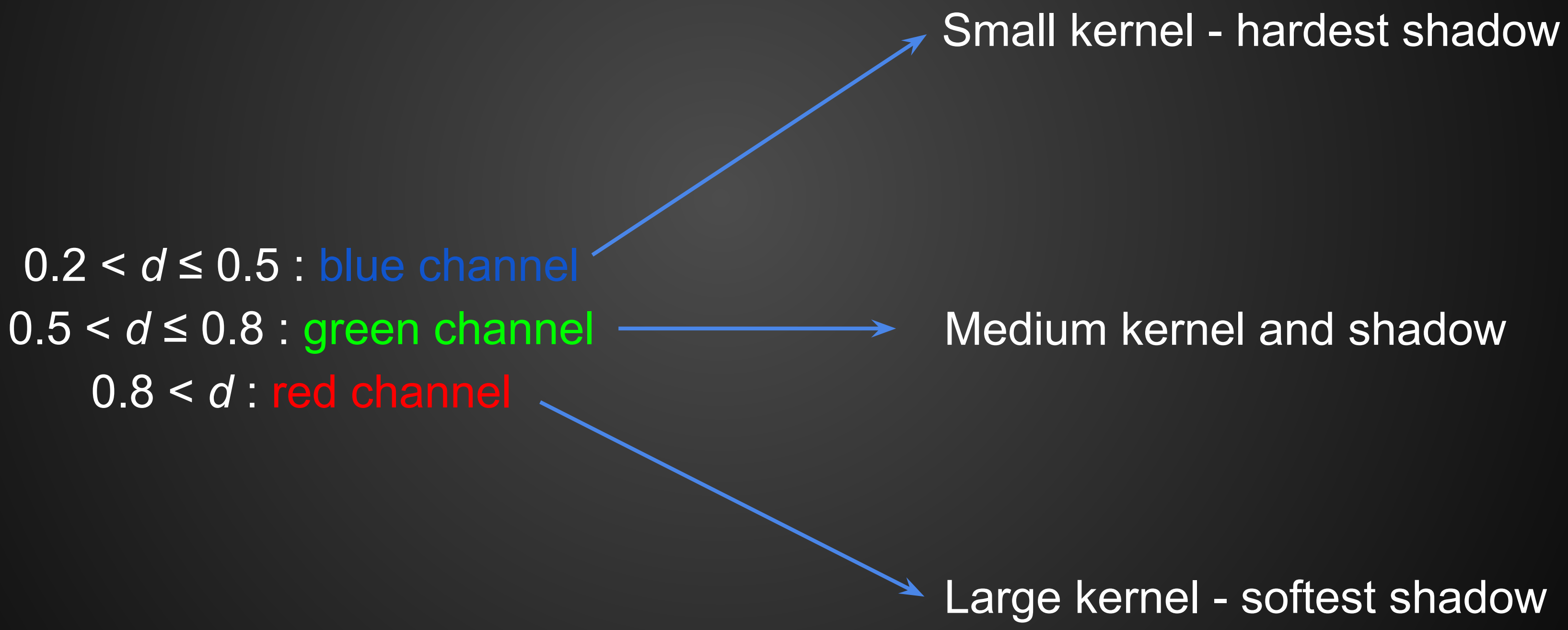
So we quantize it using the RGB channels

$0.2 < d \leq 0.5$: blue channel

$0.5 < d \leq 0.8$: green channel

$0.8 < d$: red channel

Choose PCF Kernel in scene pass



$0.2 < d \leq 0.5$: blue channel

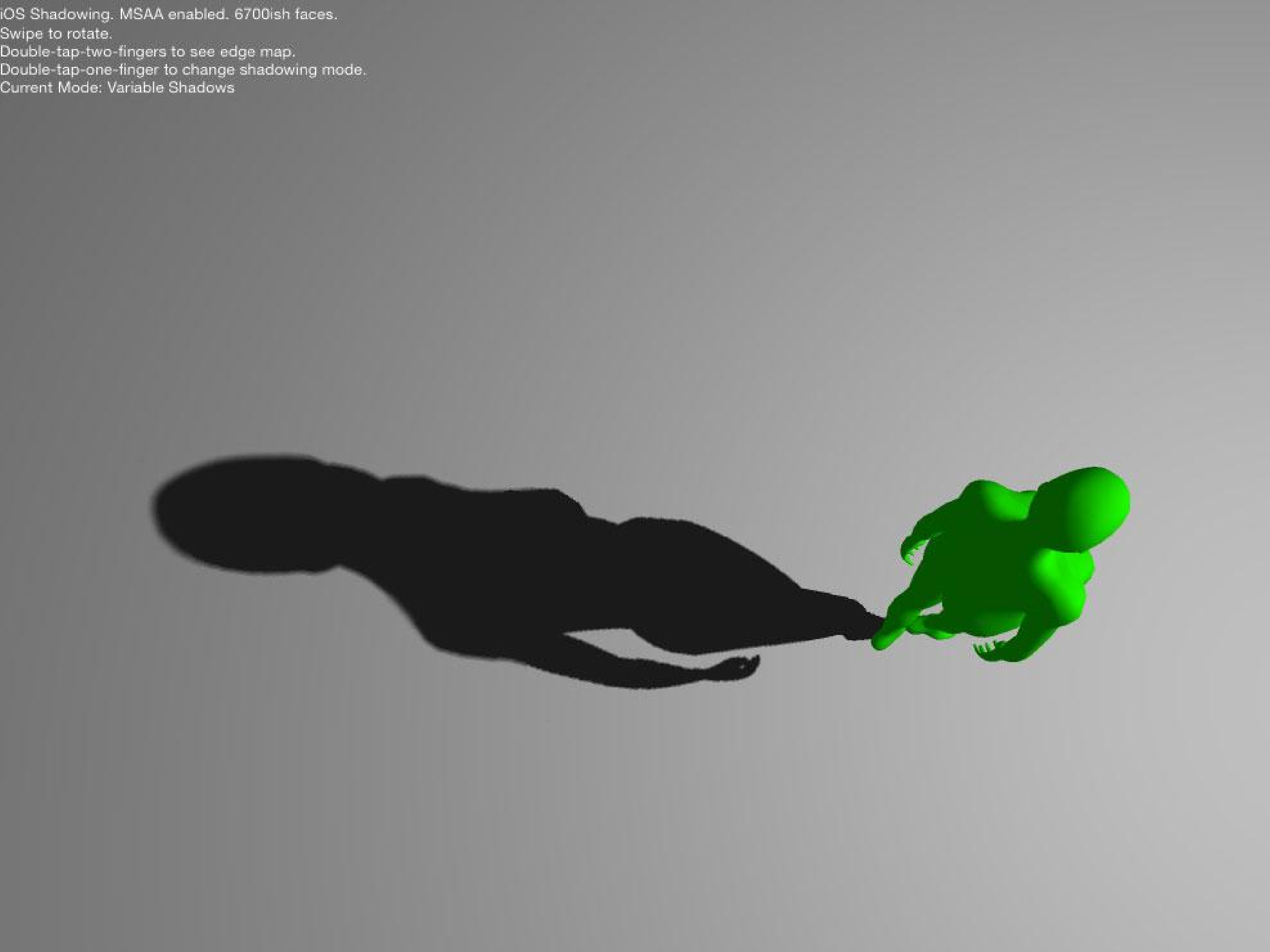
$0.5 < d \leq 0.8$: green channel

$0.8 < d$: red channel

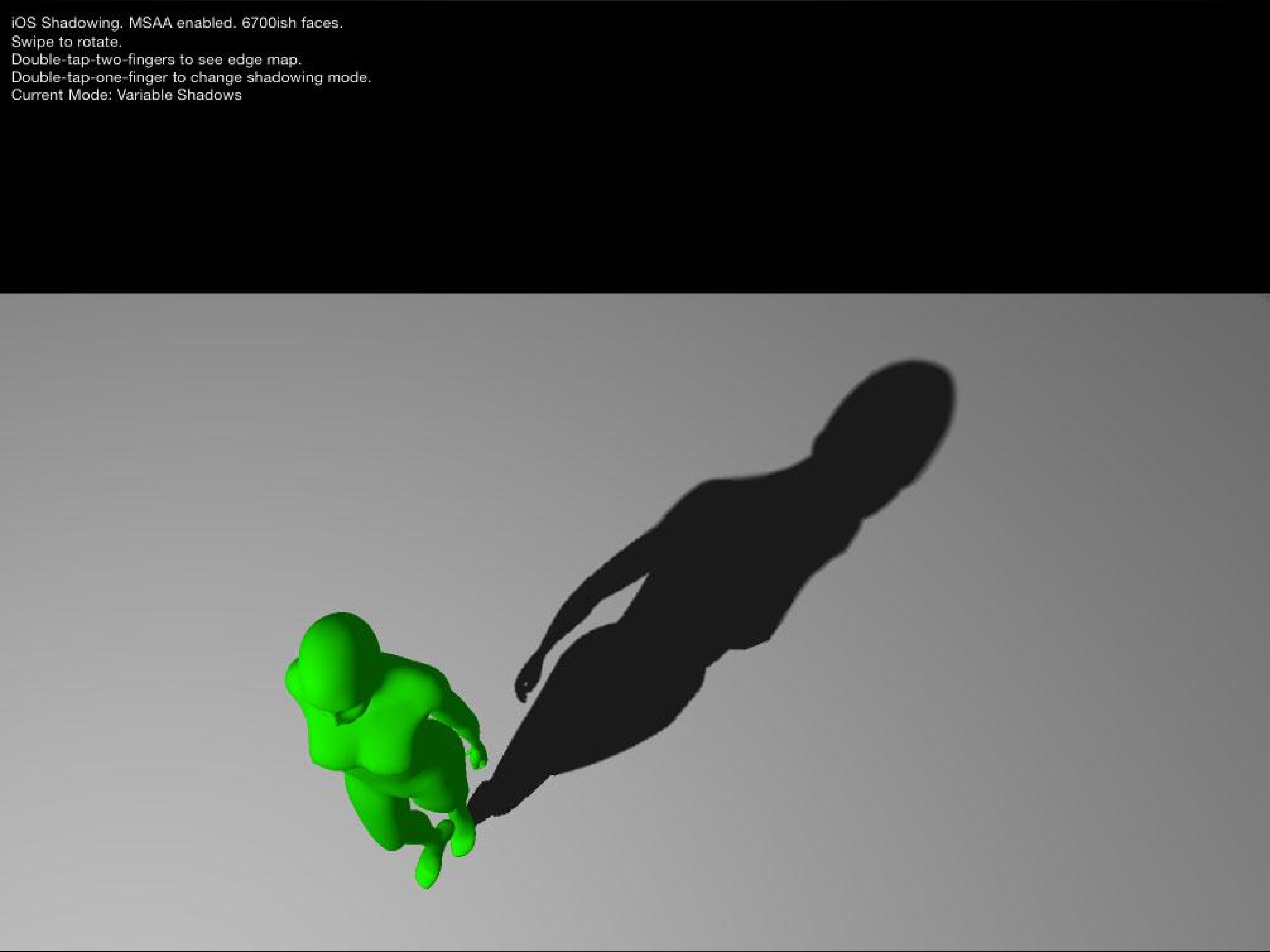
Small kernel - hardest shadow

Medium kernel and shadow

Large kernel - softest shadow



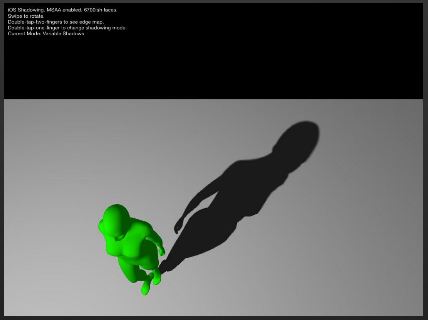
iOS Shadowing. MSAA enabled. 6700ish faces.
Swipe to rotate.
Double-tap-two-fingers to see edge map.
Double-tap-one-finger to change shadowing mode.
Current Mode: Variable Shadows



iOS Shadowing. MSAA enabled. 6700ish faces.
Swipe to rotate.
Double-tap-two-fingers to see edge map.
Double-tap-one-finger to change shadowing mode.
Current Mode: Variable Shadows

Results (iPad 3rd Gen)

Technique	**Framerate**
Hard Shadows	52 fps (19 ms/f)
PCF - 4x4 - unoptimised	7 fps (142 ms/f)
PCF - 4x4 - Shadow Mask	25 fps (40 ms/f)
PCF - 7x7 - Shadow Mask	11 fps (91 ms/f)
Quantized Shadow Mask (2x2, 3x3, 5x5)	**20 fps (50 ms/f)**
Quantized Shadow Mask (3x3, 5x5, 7x7)	**14 fps (71 ms/f)**



iOS Shadowing. MSAA enabled. 6700ish faces.
Swipe to rotate.
Double-tap-two-fingers to see edge map.
Double-tap-one-finger to change shadowing mode.
Current Mode: Variable Shadows

Changing shadow map resolution

Technique	Framerate
512x512	24fps (42 ms/f)
1024x1024	20fps (50 ms/f)
2048x2048	12fps (84 ms/f)

Quantized Shadow Mask (2x2, 3x3, 5x5)

Artefacts

Irregular edge detection results in “wrong” filter being applied in a region, which leads to inconsistent edges. Causes:

1) poor quality edge filter and/or low resolutions shadow map

2) ‘internal’ edges of object (more serious)

Also ‘jumping’ between PCF filters can look bad if widths are too different

Conclusions

Even basic soft-shadowing is slow on mobile GPUs such as those used in the iPad

Creating a Shadow Mask, using an edge detector and mip-chain dilation, improves performance of PCF greatly

Variable penumbra shadows are possible with OK framerates

Future work

More tests with different scenes (especially large scenes)

Links to automatic light frustrum sizing

Might be able to combine with PCSS in the ‘desktop’ world upf. Universitat Pompeu Fabra Barcelona

GTI Interactive Technologies Group

IMPART

SEVENTH FRAMEWORK PROGRAMME

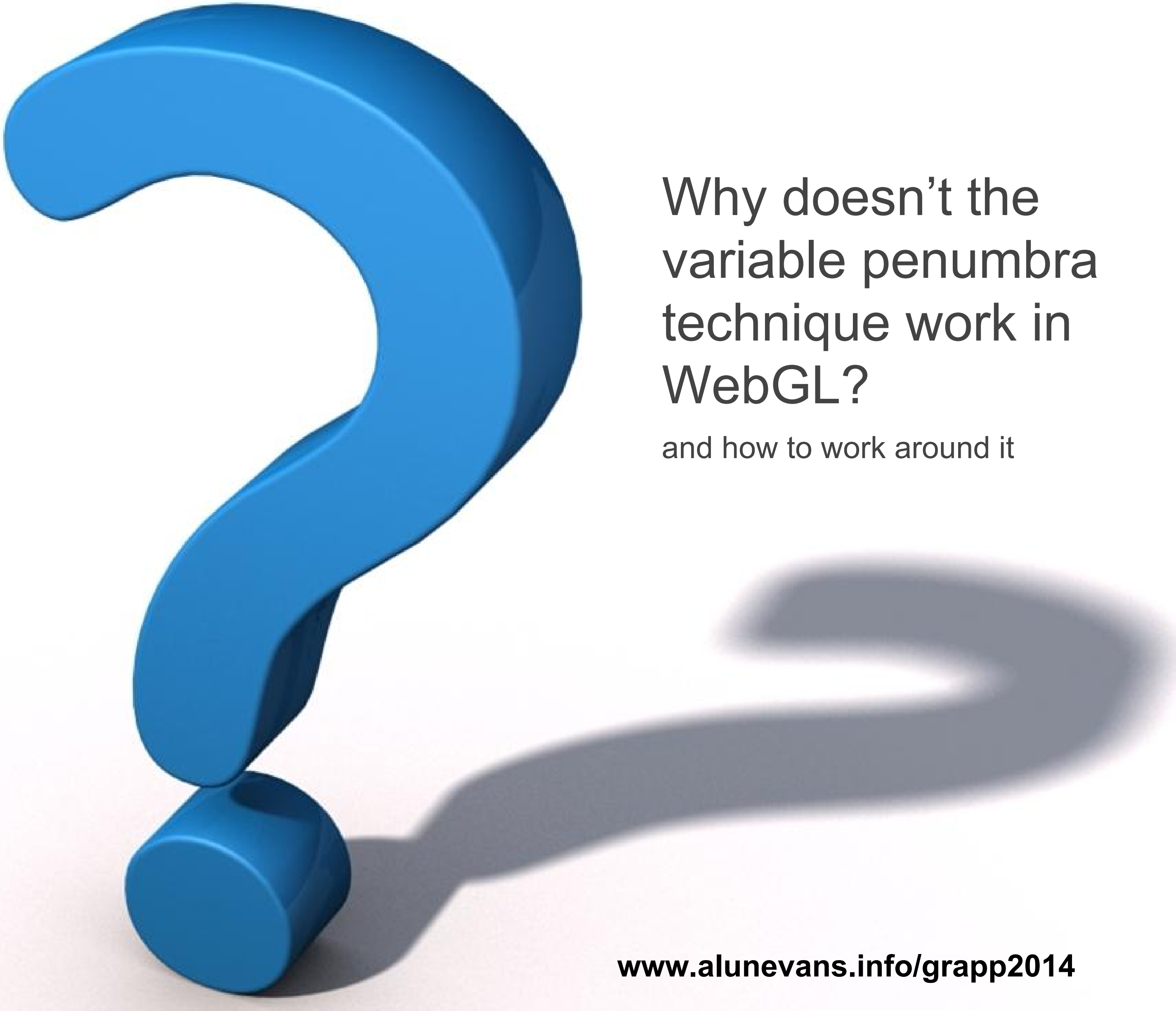
http://impart.upf.edu/

links and source code:

www.alunevans.info/grapp2014

alun.evans@upf.edu

A question for you!



Why doesn’t the variable penumbra technique work in WebGL?

and how to work around it

www.alunevans.info/grapp2014

alun.evans@upf.edu