

The Making of Seemore WebGL

Will Eastcott, CEO, PlayCanvas

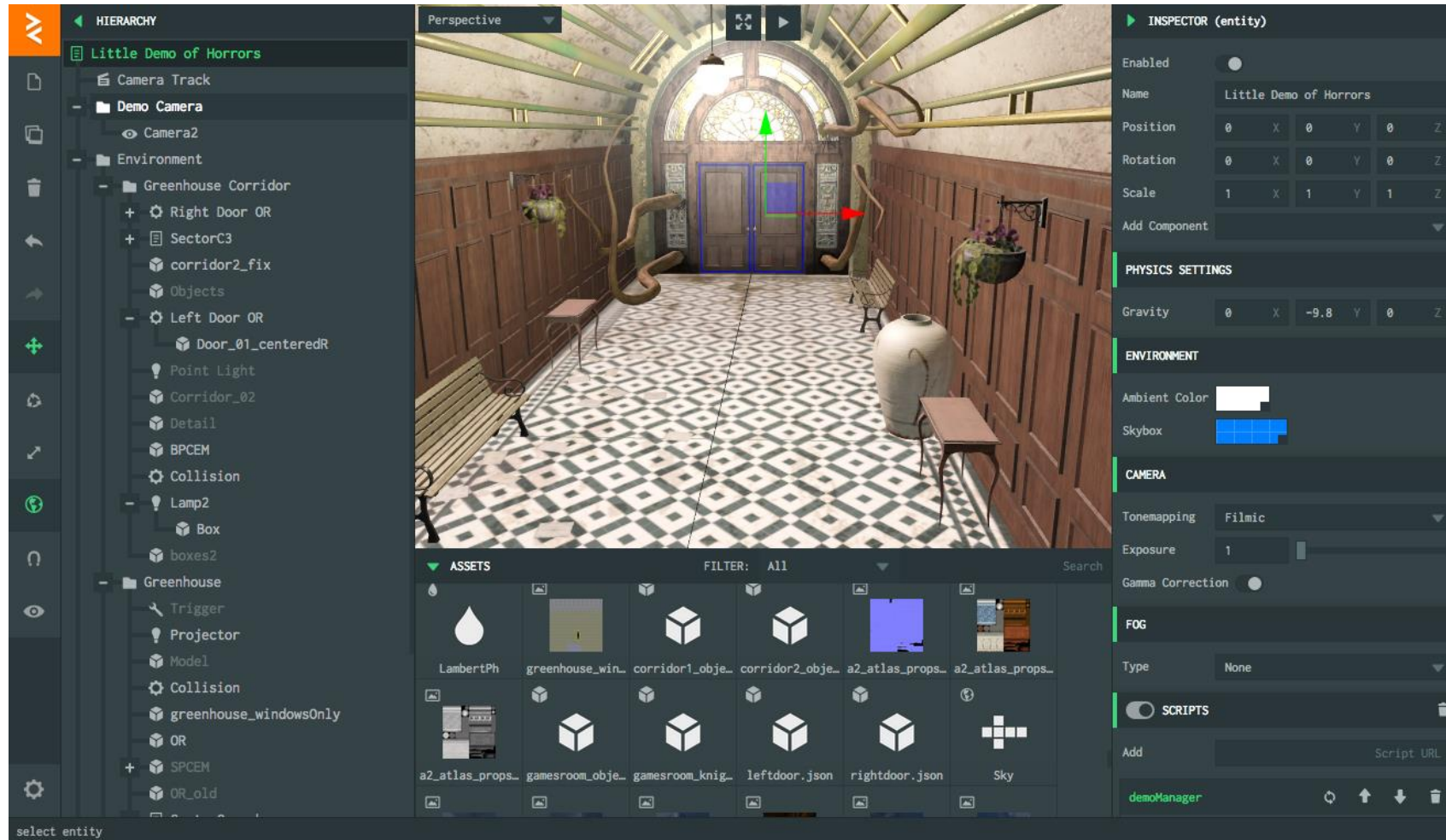
What is Seemore WebGL?

A mobile-first, physically rendered game environment powered by HTML5 and WebGL



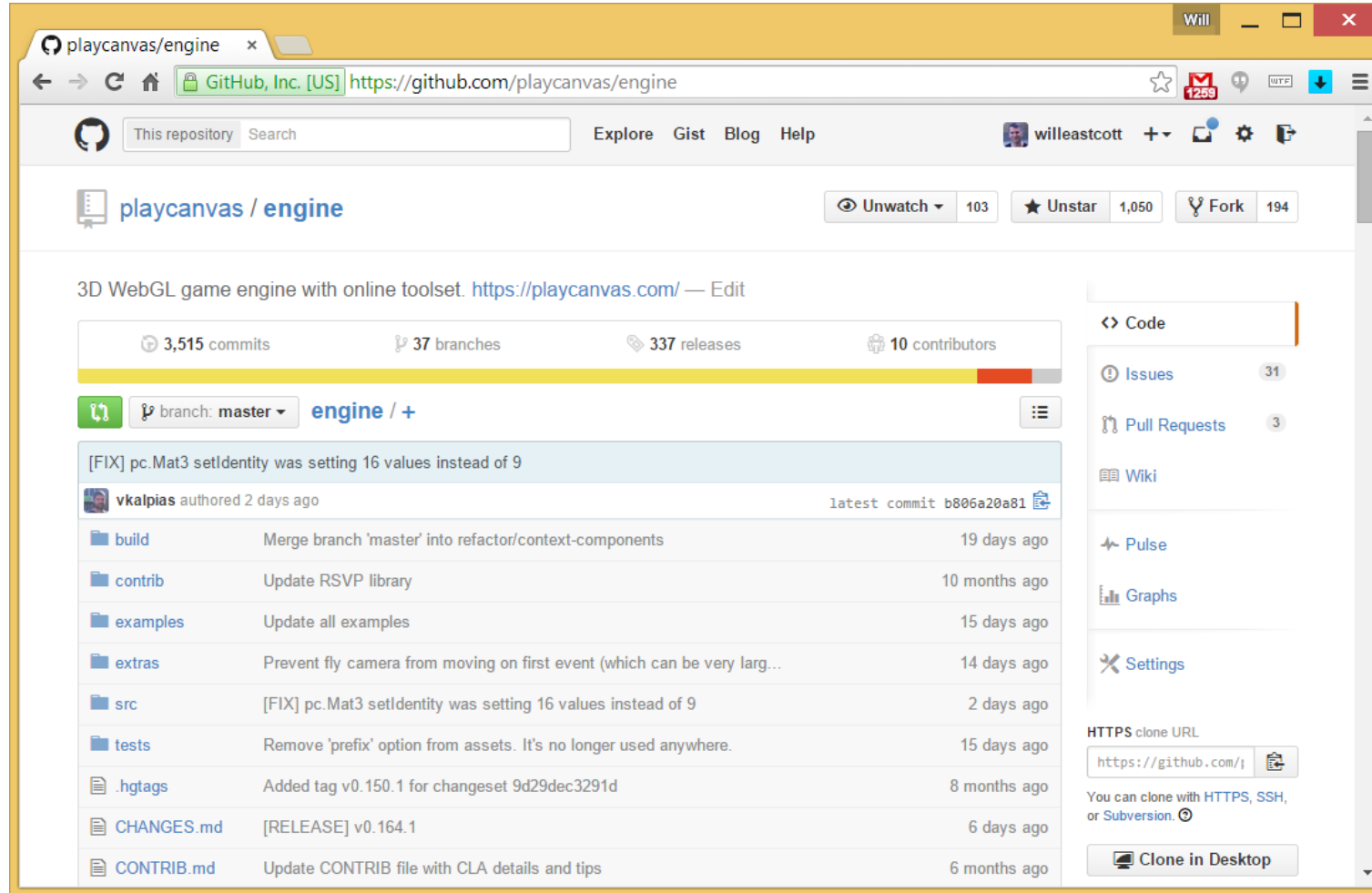
PlayCanvas: Powering Seemore WebGL

Open-source, WebGL game engine with cloud-based visual tools



Did I Mention PlayCanvas is Open Source?

<https://github.com/playcanvas/engine>



The screenshot shows the GitHub repository page for `playcanvas/engine`. The repository is described as a "3D WebGL game engine with online toolset" and provides a link to <https://playcanvas.com/>. It has 3,515 commits, 37 branches, 337 releases, and 10 contributors. The latest commit, `b806a20a81`, was authored by `vkalpias` 2 days ago. The commit message is "[FIX] pc.Mat3 setIdentity was setting 16 values instead of 9". The repository includes several folders and files, such as `build`, `contrib`, `examples`, `extras`, `src`, `tests`, `.hgtags`, `CHANGES.md`, and `CONTRIB.md`. The right sidebar shows options to view the code, issues, pull requests, wiki, pulse, graphs, and settings. It also provides the HTTPS clone URL and a button to clone the repository to the desktop.

playcanvas / engine

3D WebGL game engine with online toolset. <https://playcanvas.com/> — Edit

3,515 commits 37 branches 337 releases 10 contributors

branch: master engine / +

[FIX] pc.Mat3 setIdentity was setting 16 values instead of 9

vkalpias authored 2 days ago latest commit b806a20a81

File	Commit Message	Time Ago
build	Merge branch 'master' into refactor/context-components	19 days ago
contrib	Update RSVP library	10 months ago
examples	Update all examples	15 days ago
extras	Prevent fly camera from moving on first event (which can be very larg...	14 days ago
src	[FIX] pc.Mat3 setIdentity was setting 16 values instead of 9	2 days ago
tests	Remove 'prefix' option from assets. It's no longer used anywhere.	15 days ago
.hgtags	Added tag v0.150.1 for changeset 9d29dec3291d	8 months ago
CHANGES.md	[RELEASE] v0.164.1	6 days ago
CONTRIB.md	Update CONTRIB file with CLA details and tips	6 months ago

Code

Issues 31

Pull Requests 3

Wiki

Pulse

Graphs

Settings

HTTPS clone URL

<https://github.com/>

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Updating a Classic

The original C++ Seemore demo was significantly upgraded in terms of visuals



So How Did We Do It?

- Physically Based Rendering (PBR)
- Lightmapping
- Ambient occlusion
- Dynamic shadow mapping
- Box projected cube map reflection mapping
- ...and a serious amount of optimization

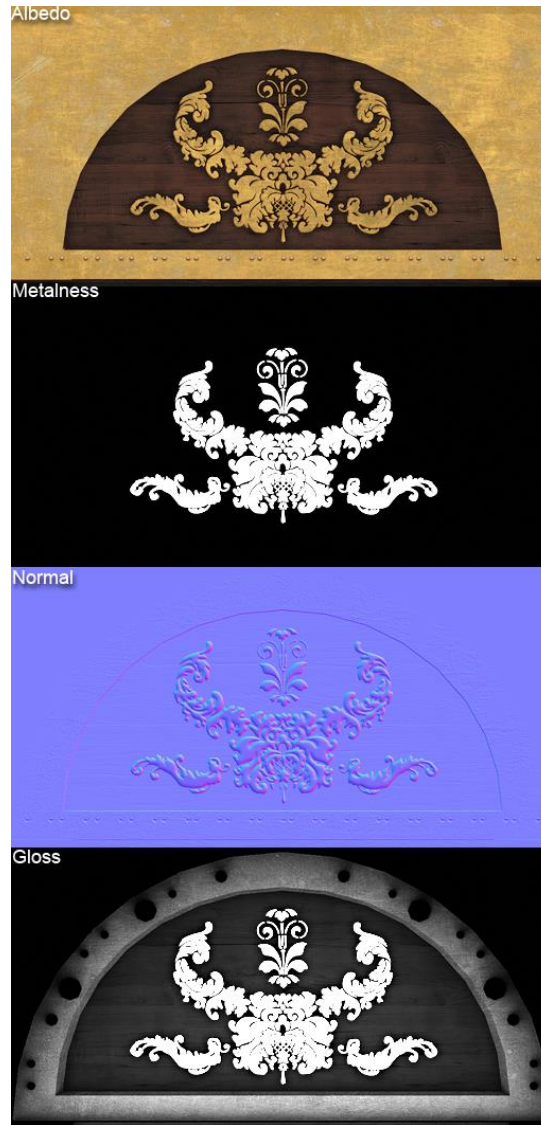
Physically Based Rendering: The Basics

- Energy Conservation
- Meaningful maps
- HDR data
- Tonemapping
- Gamma Correction

Physically Based Rendering: Material Input Data

- Material

- Albedo (color information)
- Normal (surface information)
- Gloss (microsurface information)
- Metalness (optional)
- Opacity (optional)



Physically Based Rendering: Environmental Input Data

- IBL probes – prefiltered, box projected cubemaps
 - Used custom filtering tool
 - PlayCanvas editor users will soon be able to generate these cubemaps in the interface
- Analytical lights (point/directional, energy conserving Blinn-Phong)
- Lightmaps
- Everything must be HDR!
 - Encode HDR to RGBM



Physically Based Rendering: Materials

- Schlick's Fresnel approximation
- Toksvig's factor for specular anti-aliasing



Tonemapping

- Important for getting rid of overly bright or dark spots
- Generally simulates how the eye or film perceives colors
- Seemore uses the 'well known' Uncharted 2 tonemap. See: <http://filmicgames.com/archives/75>



Lightmapping

Single map for scenes with static light sources



Lightmapping

Direct and indirect lightmaps for scenes with slightly moving lights

- Use indirect lightmap as base ambient
- Store directional static shadows in a separate lightmap
 - Can store multiple shadows in a texture's color channels for a slightly moving light
 - Mask real-time diffuse/specular light with shadows
 - Combine with real-time shadows

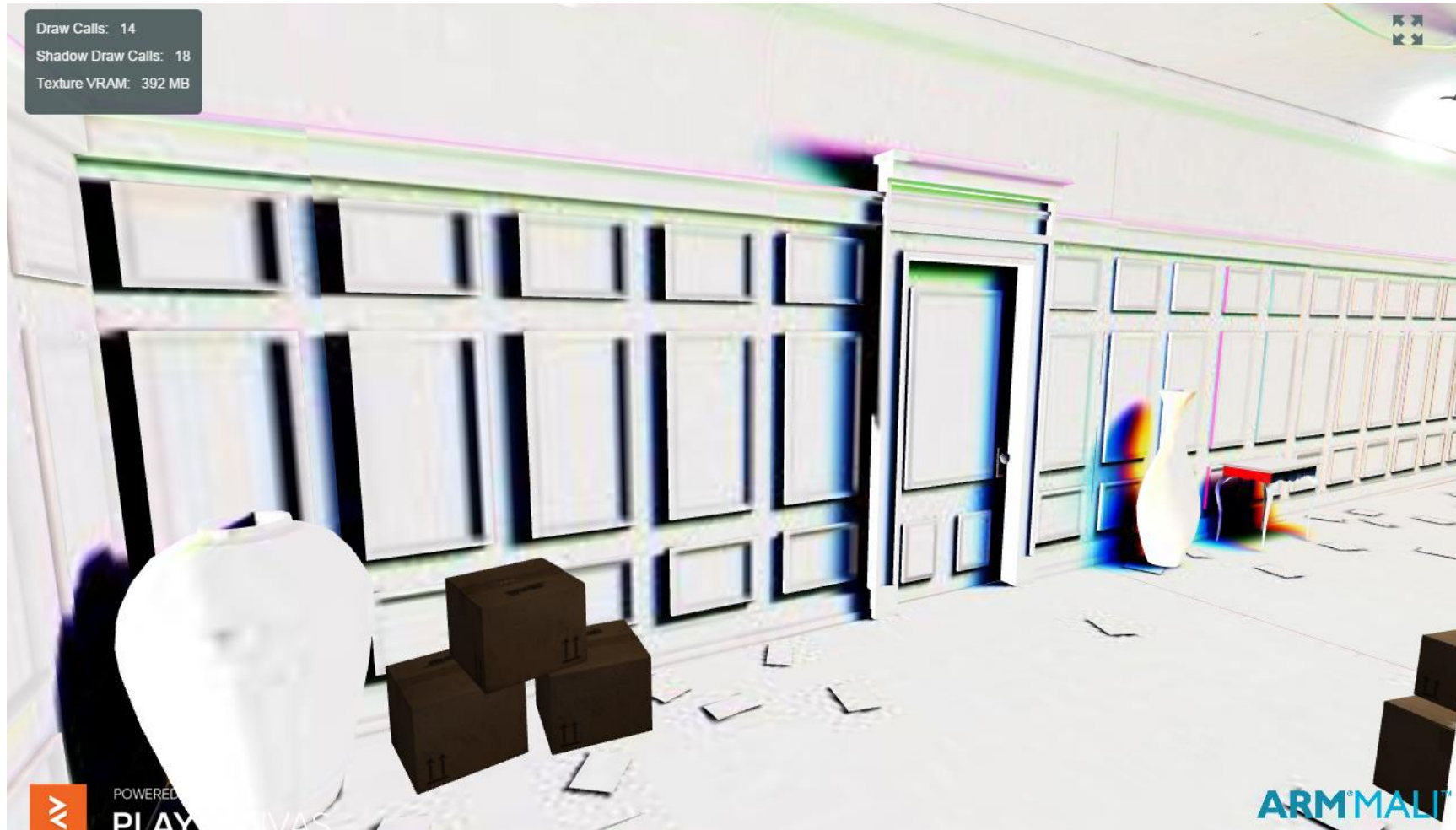
Lightmapping

Indirect lightmap

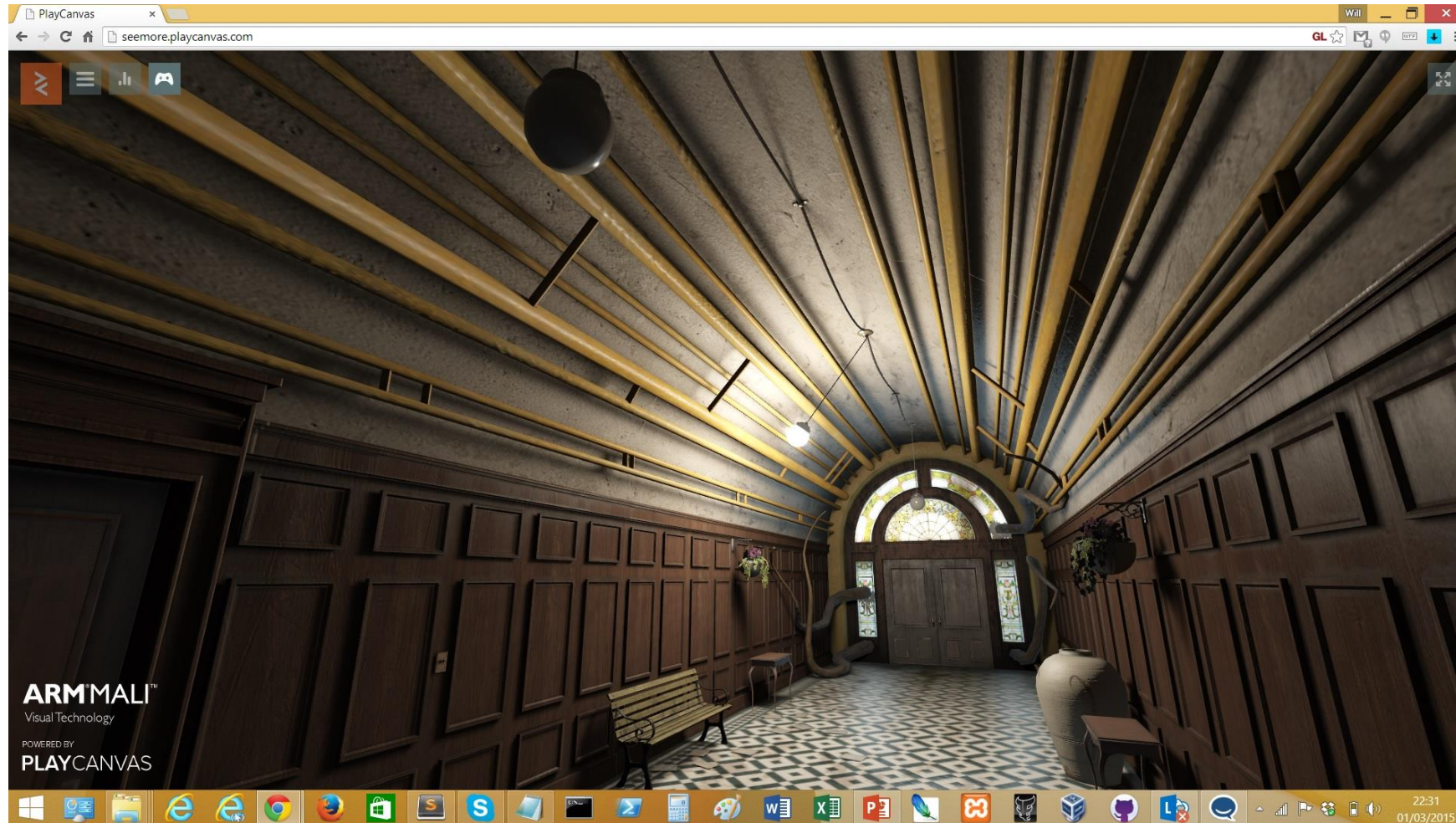


Lightmapping

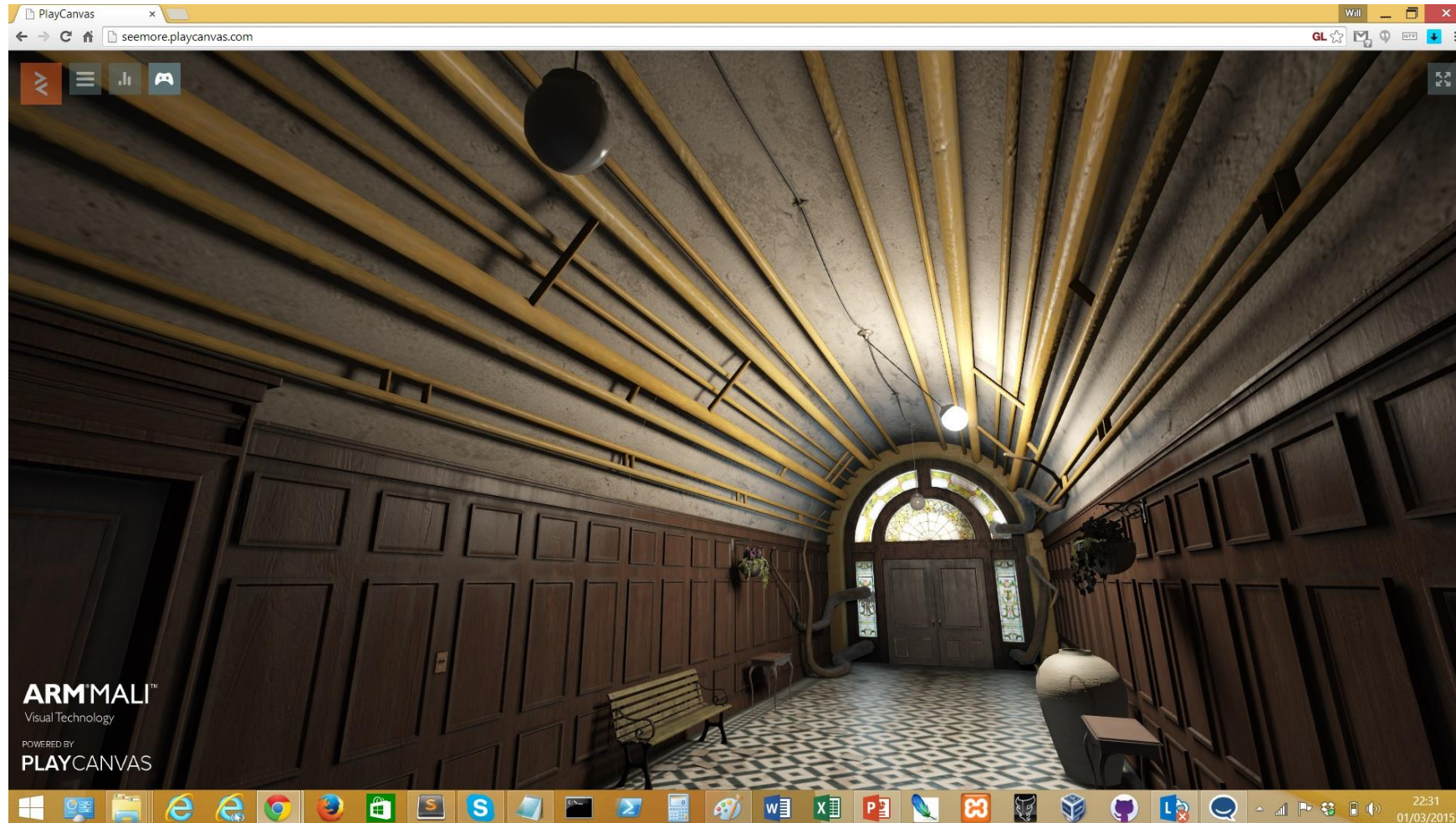
Direct lightmap



Dynamic Shadows: Interpolated Lightmaps



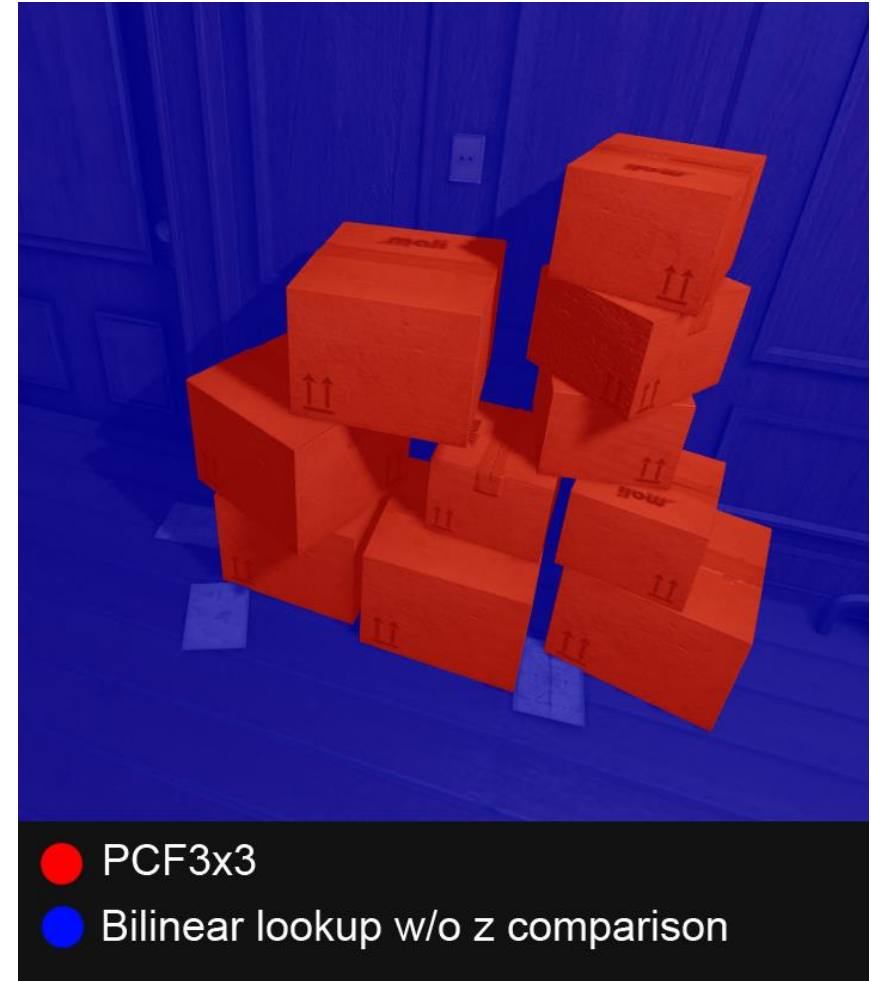
Dynamic Shadows: Interpolated Lightmaps



Dynamic Shadows: Shadow Mapping

- Real-time shadow maps for dynamic objects
 - Very costly on mobile devices (draw calls, projection, reads)
 - Only update shadow map every 3rd frame if objects are not moving
 - Only use real self-shadowing with multiple taps on dynamic objects
 - Just use bilinear lookup without depth comparison on static receivers
 - Shadow is encoded as RGB (depth) and A (mask for bilinear lookup) in an 8-bit 1024x1024 texture
 - Use Normal Offset bias to reduce self-shadowing artifacts
 - See <http://www.dissidentlogic.com/old/#Normal%20Offset%20GDC%20Materials>

Dynamic Shadows: Shadow Mapping



Ambient Occlusion

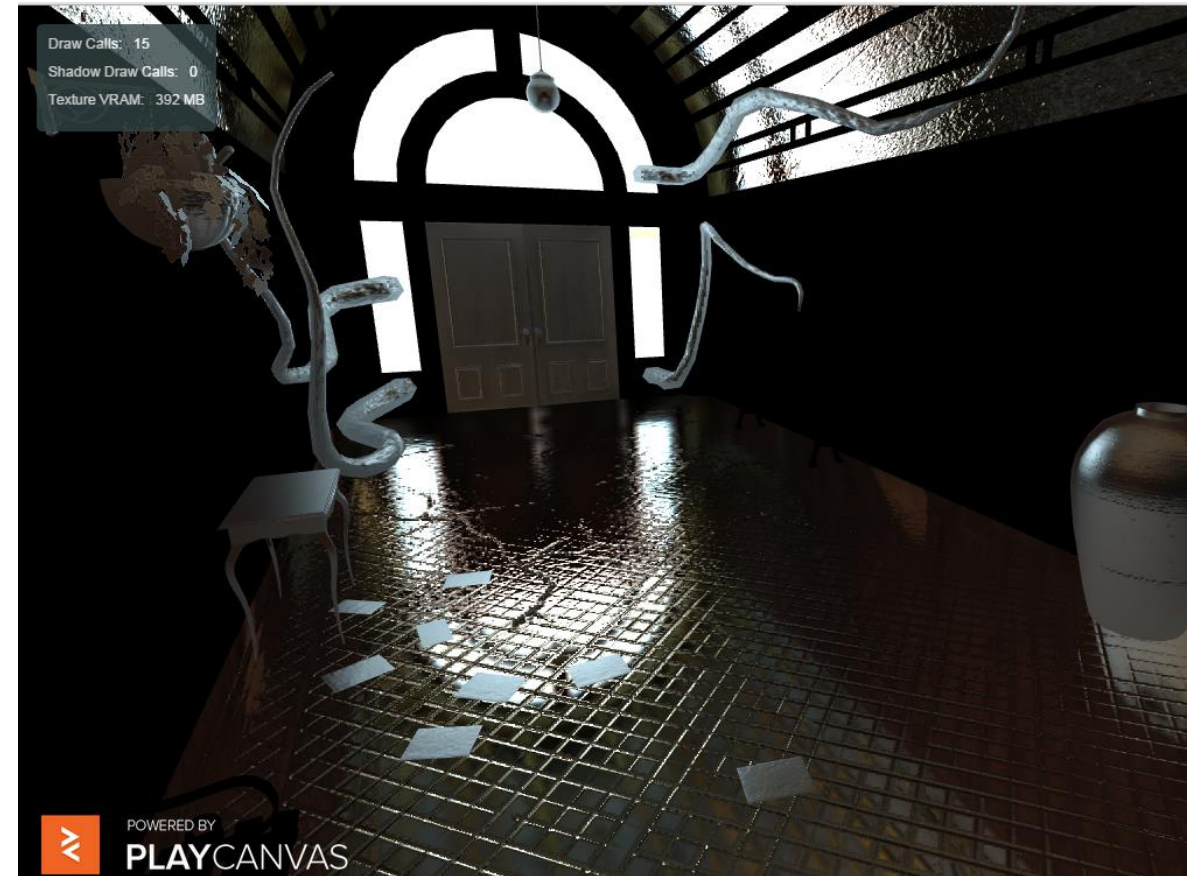
- Occluding IBL lighting (especially specular) is vital
- Specular occlusion should ideally be view-dependent but it's too expensive for mobile
- Baked AO maps (tweak attenuation for specular occlusion)
- Can store multiple AO maps in channels (the plant's mouth, for example)
- Last resort: derive some occlusion factor from single/indirect lightmaps
- Procedural AO for special cases (e.g. plant tentacles)

Ambient Occlusion



Box Projected Cube Map Reflections

- Widely used in games to make cube maps fit in a more correct way with the scene
- Used for both IBL reflections and refractions
- Excellent primer on the topic here:
<http://www.gamedev.net/topic/568829-box-projected-cubemap-environment-mapping/>



Box Projected Cube Map Reflections



Optimization

- ETCI Texture Compression (`WEBGL_compressed_texture_etc1`)
 - Compressing textures is vital for low memory mobile devices
 - WebGL is limited in the formats it exposes but does provide ETCI (and DXT)
 - Good results for diffuse, specular, gloss and lightmaps
 - Bad results for normal maps
 - Compressed the majority of normal maps anyway
 - Remainder were left uncompressed when the results were particularly bad
- Use lowp in fragment shaders where possible
 - Only precision-critical shaders (shadow mapping) retained highp
- Avoid costly math ops (pow, sqrt)
- Cull draw calls
- Sort opaque draw calls front to back

Additional Tricks: Foliage

- Plants are translucent and pass light to the back side
 - Seemore plants are in a hemispherically lit room
 - Just use `-normal.y` as a translucency factor
 - Additional analytical per-vertex translucency occlusion for the main plant



Additional Tricks: Halos

- Avoid full-screen effects
 - Incurs high fill rate costs
 - You lose hardware antialiasing
 - Although it is possible to implement AA as a post process, again, this is expensive
- Use a cheap alternative
 - Transparent, camera aligned sprite



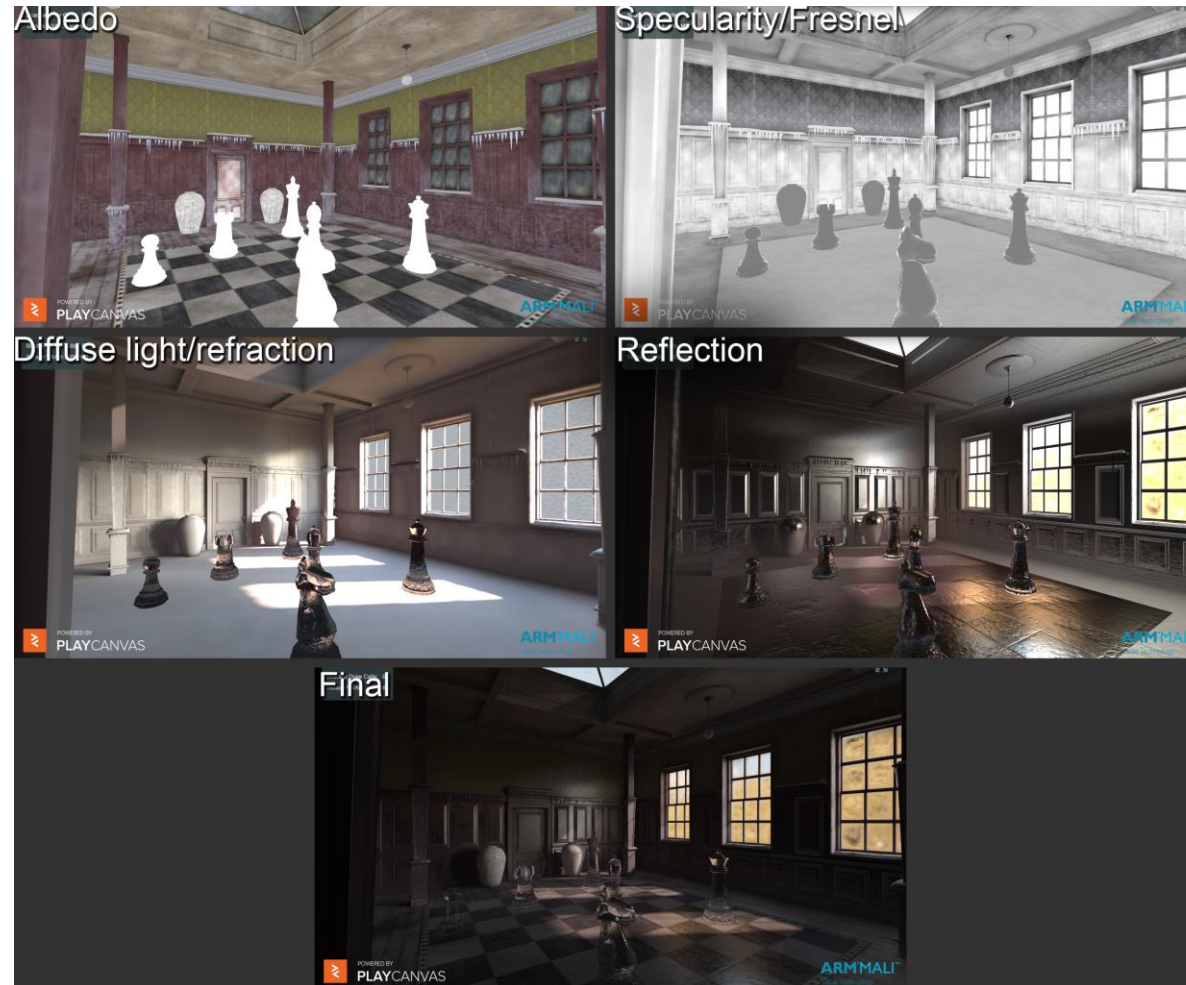
Additional Tricks: Dynamic Exposure

- We avoided an analytical approach
- Exposure controlled through script
- Dependent on location and view vector

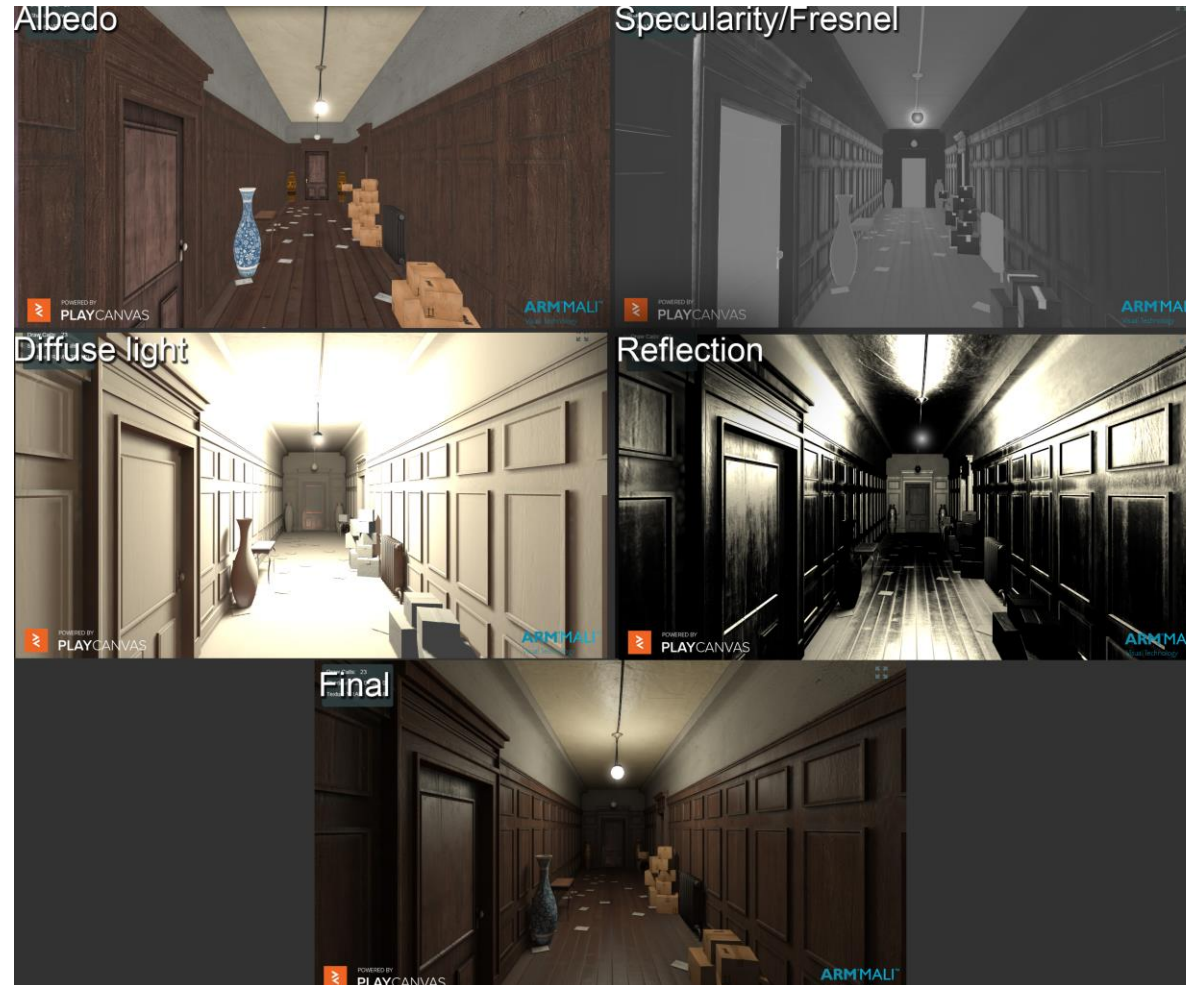
Things That Didn't Make The Cut

- Instancing (`ANGLE_instanced_arrays`)
 - Seemore was limited more on the GPU (fragment operations)
 - The demo didn't render enough instances to make a significant difference
- Variance Shadow Mapping
 - Works well on desktop machines but a bug in Chrome's float texture extension was problematic
 - Encoding the shadow map to RGBA8 lost too much precision

Putting It All Together



Putting It All Together



Some Links...

- The Open Source PlayCanvas project: <https://github.com/playcanvas/engine>
- The Cloud-Hosted PlayCanvas toolset: <https://playcanvas.com>
- The Seemore Demo: <http://seemore.playcanvas.com>
- ARM Mali Developer Center: <http://malideveloper.arm.com>

tanx.playcanvas.com