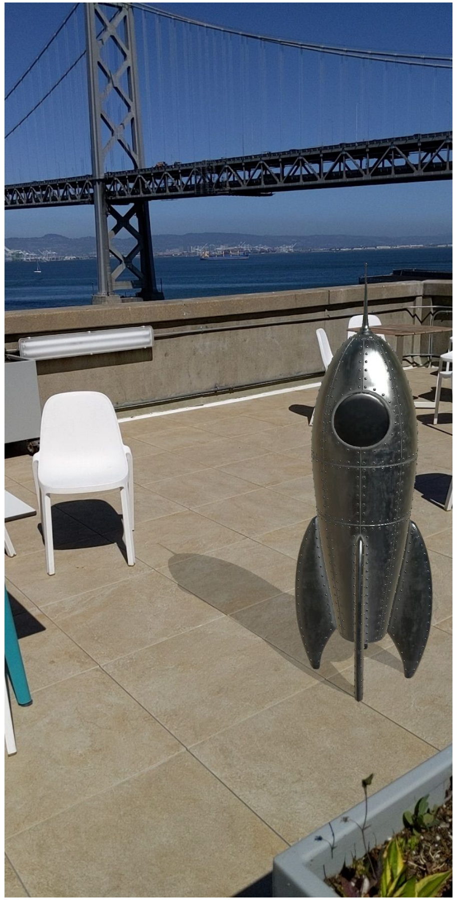
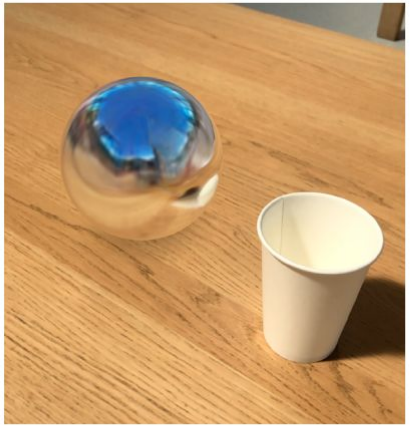
WebXR AR
Light Estimation

Immersive Web Seattle Face to Face
Feb 2020

What we want



QUIET PLEASE



Two known native APIs

ARCore (Android)
ARKit (iOS)

ARCore (Android)

Environmental HDR mode

- Single implicit probe containing:
    - Ambient term (Spherical Harmonics)
    - Single directional light direction/intensity (Float RGB)
    - HDR Environment Cubemap (Half Float arrays)

ARKit (iOS)

Environmental Texturing

- Mutliple potential probe containing:
  - HDR Environment Map (MTLTexture, undocumented format)
  - Includes extents for parallax mapping (Meters)
- Includes an automatic mode that places probe for you

Both APIs

Lower-power, ambient-only mode

- ARCore:
  - Intensity (Float scalar, 0->1)
  - Color correction term? (RGB, gamma color space)
  - Cannot use during Environmental HDR Mode
- ARKit:
  - Intensity (Float scalar, in lumens)
  - Color temperature (Float scalar, in degrees Kelvin)
  - Can be used in parallel to Environmental Texturing

Mapping to WebXR

Kip's explainer proposal

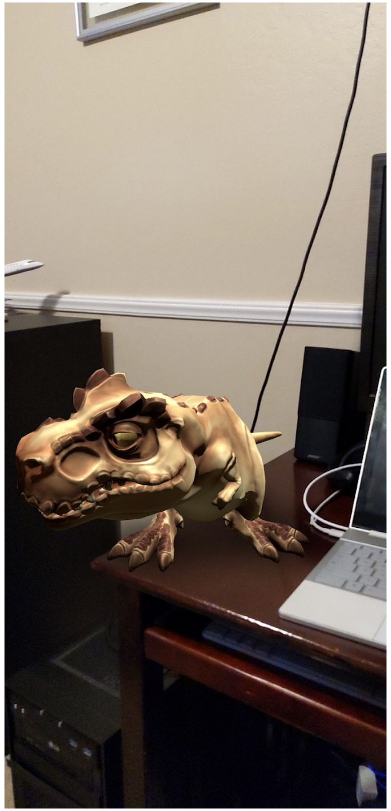
```
partial interface XRFrame {
  Promise<XRLightProbe> getGlobalLightEstimate();
  Promise<XRReflectionProbe> getGlobalReflectionProbe();
};

partial interface XRLightProbe {
  readonly attribute Float32Array indirectIrradiance;
  readonly attribute Float32Array? primaryLightDirection;
  readonly attribute Float32Array? primaryLightIntensity;
  readonly attribute Float32Array? sphericalHarmonicsCoefficients;
  [SameObject] readonly attribute DOMPointReadOnly? sphericalHarmonicsOrientation;
};

partial interface XRReflectionProbe {
  [SameObject] readonly attribute DOMPointReadOnly orientation;
  WebGLTexture? createWebGLEnvironmentCube();
};
```

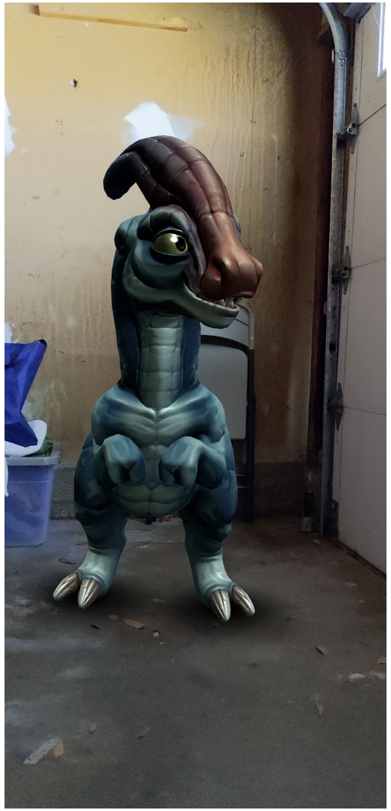
Will’s Chrome implementation (Pt. 1)

```
// Need to initialize
xrSession.updateWorldTrackingState({
  lightEstimationState: { enabled: true } // Whole bunch of supporting IDL for this.
});

// WebIDL
partial interface XRFrame {
   readonly attribute XRWorldInformation worldInformation;
};

interface XRWorldInformation {
  readonly attribute XRLightEstimation? lightEstimation;
};

partial interface XRLightEstimation {
  readonly attribute XRLightProbe? lightProbe;
  readonly attribute XRReflectionProbe? reflectionProbe;
};
```

Will's Chrome implementation (Pt. 2)

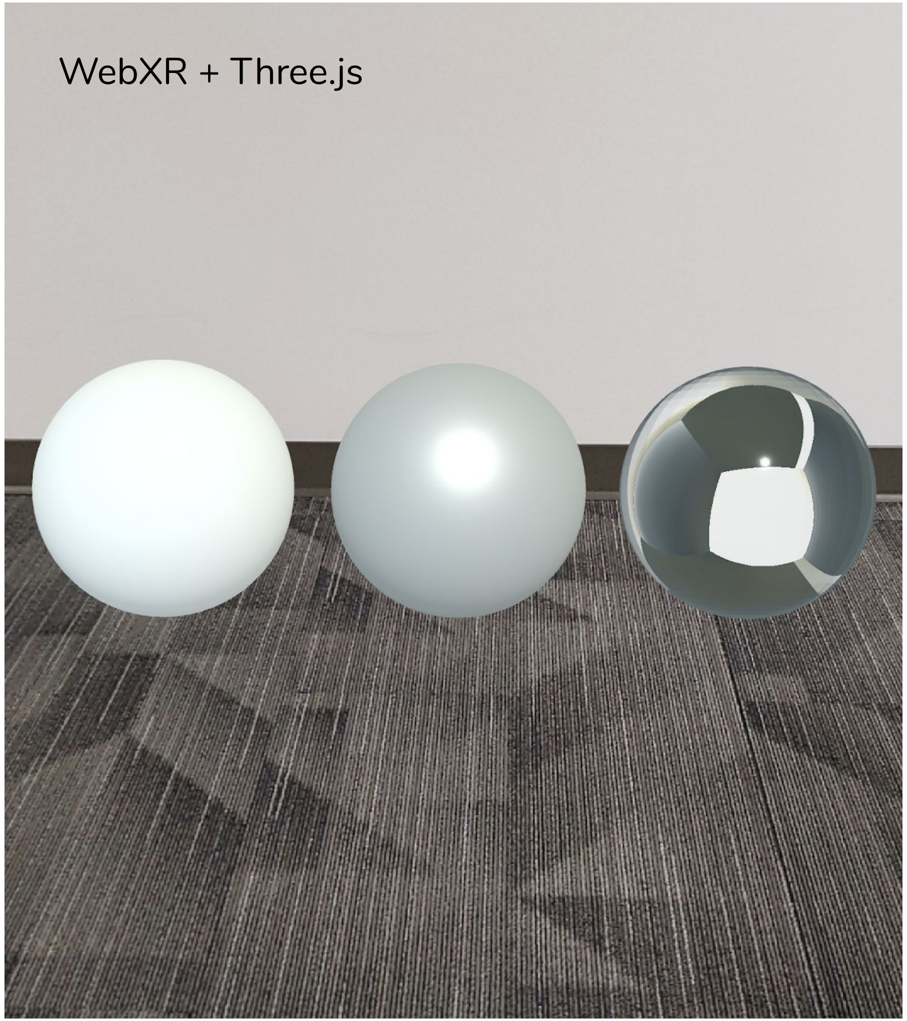
```
interface XRLightProbe {
  readonly attribute XRSphericalHarmonics sphericalHarmonics;
  readonly attribute DOMPointReadOnly mainLightDirection;
  readonly attribute DOMPointReadOnly mainLightIntensity;
};

interface XRSphericalHarmonics {
  readonly attribute DOMPointReadOnly orientation;
  readonly attribute Float32Array coefficients;
};

interface XRReflectionProbe {
  readonly attribute XRCubeMap cubeMap;
};

interface XRCubeMap {
  WebGLTexture updateWebGLEnvironmentCube(WebGL2RenderingContext context,
                                          WebGLTexture? texture);
};
```

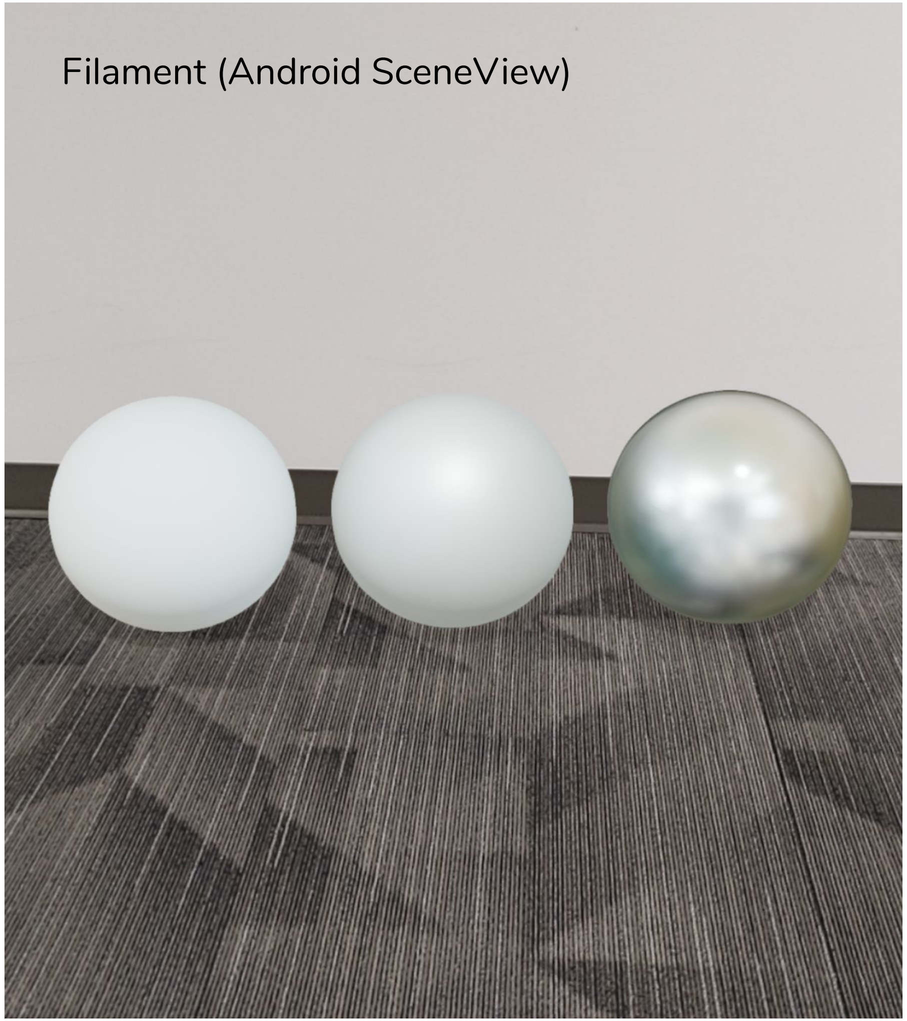




Results!

But also not without problems…


WebXR + Three.js


Filament (Android SceneView)

Proposed adjustments

Simplify structure

- Closer to Kip’s original suggestion
- No Promises on XRFrame methods.
- Define Light Probe structure and expose a single “Global” or “Automatic” probe to allow for potential expansion into manually placed probes in the future
- Drop dynamic enabling of light estimation in favor of a session feature names.

Should we enable low power ambient-only mode?

- May be necessary on iOS
- Can encode single value ambient into Spherical Harmonics.
- Zero-intensity directional light
- No Environment Map

Cubemap handling questions

- ARCore gives half float arrays.
- No idea what format ARKit uses, but returns a MTLTexture
- Float/Half float not supported in WebGL 1.0 core.
  - Limit to WebGL 2.0 only?
  - Require extension to be enabled first?
  - Force internal conversion when support not available?
  - Return data arrays and let dev deal with it? (*Blech*)
- To mipmap or not to mipmap?

Cubemap handling (continued)

- Need a way to allow texture reuse.
- Need to notify developer when it changes.
- Doesn’t necessarily need to be frame-aligned, because it will (ideally) change at a slower cadence.
- As a result, reflections at least may want to be a separate object.

Brandon’s proposal

```
partial interface XRFrame {
  XRLightProbe? globalLightProbe;
};

interface XRLightProbe {
  readonly attribute Float32Array sphericalHarmonicsCoefficients;
  readonly attribute DOMPointReadOnly primaryLightDirection;
  readonly attribute DOMPointReadOnly primaryLightIntensity;
};
```

Brandon's proposal (Pt. 2)

```
partial interface XRSession {
  Promise<XRWebGLReflectionProbe> requestGlobalWebGLReflectionProbe(
    WebGLRenderingContext context);
};

interface XRWebGLReflectionProbe extends EventTarget {
  WebGLTexture? cubeTexture;
  attribute EventHandler onupdate;
};
```

Open Questions

- How many guarantees can/should we give about data?
- What’s the appropriate way to communicate the space the data is reported in?
- Feature names: Worth separating light/reflection probe features?