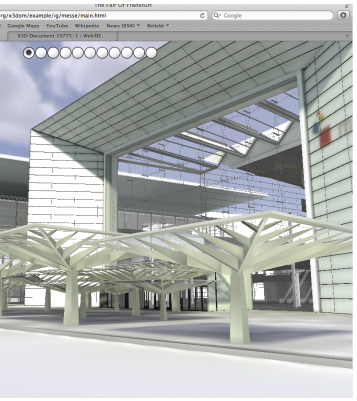
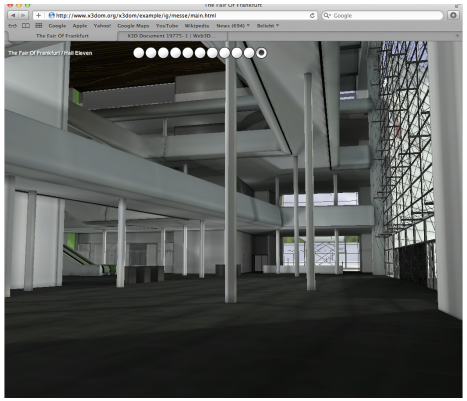
Efficient Binary Meshes in X3DOM refined: Not just images anymore!

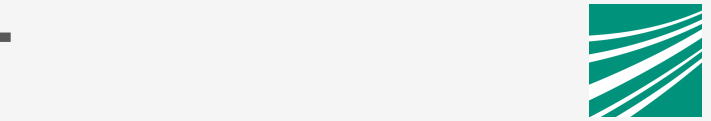
Johannes Behr
Yvonne Jung
Tobias Franke
Timo Sturm

johannes.behr@igd.fraunhofer.de

Fraunhofer IGD

Declarative (X)3D in HTML

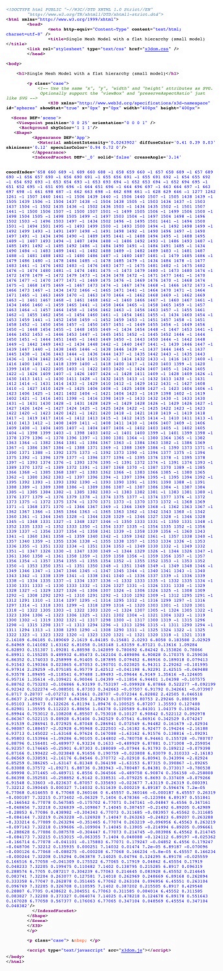
Embed a live scenegraph in the DOM



```
<!DOCTYPE html >
<html >
    <body>
        <h1>Hello X3DOM World</h1>
            <x3d xmlns='…' profile='HTML' >
                <scene>
                    <shape>
                        <box></box>
                    </shape>
                </scene>
            </x3d>
    </body>
</html>
```

Declarative (X)3D in HTML

Large Datasets: Issue of the current approach



- Real 3D applications tend to be huge HTML-files
  - **Unpleasant** non-interactive **user experience**
  - Browser are not build to hold GByte of DOM attribute data (e.g. multiple data copies)
- Reference external sub-trees
  - X3D “Inline” node
  - black/white-box interface?
  - xml/json parser architecture
- Binary XML decompression
  - x3z: (ISO) Decoding on JS-Level
  - x3db: (ISO) Fast Infoset: No UA or JS-lib
  - EXI: (W3C) Even worse

DOM holds structure and data

```
<!DOCTYPE html>
<html>
  <head>
    <link rel='stylesheet' type='text/css' href='http://www.x3dom.org/x3dom/release/x3dom.css'></link>
    <script type='text/javascript' src='http://www.x3dom.org/x3dom/release/x3dom.js'></script>
  </head>
  <body>
    <x3d id='3dstuff' width='400px' height='400px'>
      <scene DEF='scene'>
        <shape>
          <appearance>
            <material diffuseColor='#FF0000'></material>
          </appearance>
          <indexedTriangleSet solid='false' index='0 1 2 1 3 2 1 4 3 5 4 1 0 5 1 0 6 5 6 7 5 5 7 4 7 8 4 7 9 8 7 6 9 6 10 9 10 11
9 10 2 11 10 0 2 6 0 10 11 2 3 8 11 3 4 8 3 11 8 9'>
            <coordinate point='0.447214 0 -0.894427 0.447214 0.850651 -0.276393 1 0 -0 0.447214 0.525731 0.723607 -0.447214
0.850651 0.276393 -0.447214 0.525731 -0.723607 -0.447214 -0.525731 -0.723607 -1 0 0 -0.447214 0 0.894427 -0.447214 -0.850651
0.276393 0.447214 -0.850651 -0.276393 0.447214 -0.525731 0.723607'></coordinate>
            <normal vector='0.447214 0 -0.894427 0.447214 0.850651 -0.276393 1 0 -0 0.447214 0.525731 0.723607 -0.447214 0.850651
0.276393 -0.447214 0.525731 -0.723607 -0.447214 -0.525731 -0.723607 -1 0 0 -0.447214 0 0.894427 -0.447214 -0.850651 0.276393
0.447214 -0.850651 -0.276393 0.447214 -0.525731 0.723607'></normal>
          </indexedTriangleSet>
        </shape>
      </scene>
    </x3d>
  </body>
</html>
```

DOM holds structure and data

More than 95% are usually unstructured data

```
<!DOCTYPE html>
<html>
  <head>
    <link rel='stylesheet' type='text/css' href='http://www.x3dom.org/x3dom/release/x3dom.css'></link>
    <script type='text/javascript' src='http://www.x3dom.org/x3dom/release/x3dom.js'></script>
  </head>
  <body>
    <x3d id='3dstuff' width='400px' height='400px'>
      <scene DEF='scene'>
        <shape>
          <appearance>
            <material diffuseColor='#FF0000'></material>
          </appearance>
          <indexedTriangleSet solid='false' index='0 1 2 1 3 2 1 4 3 5 4 1 0 5 1 0 6 5 6 7 5 5 7 4 7 8 4 7 9 8 7 6 9 6 10 9 10 11
9 10 2 11 10 0 2 6 0 10 11 2 3 8 11 3 4 8 3 11 8 9'>
            <coordinate point='0.447214 0 -0.894427 0.447214 0.850651 -0.276393 1 0 -0 0.447214 0.525731 0.723607 -0.447214
0.850651 0.276393 -0.447214 0.525731 -0.723607 -0.447214 -0.525731 -0.723607 -1 0 0 -0.447214 0 0.894427 -0.447214 -0.850651
0.276393 0.447214 -0.850651 -0.276393 0.447214 -0.525731 0.723607'></coordinate>
            <normal vector='0.447214 0 -0.894427 0.447214 0.850651 -0.276393 1 0 -0 0.447214 0.525731 0.723607 -0.447214 0.850651
0.276393 -0.447214 0.525731 -0.723607 -0.447214 -0.525731 -0.723607 -1 0 0 -0.447214 0 0.894427 -0.447214 -0.850651 0.276393
0.447214 -0.850651 -0.276393 0.447214 -0.525731 0.723607'></normal>
          </indexedTriangleSet>
        </shape>
      </scene>
    </x3d>
  </body>
</html>
```

Mesh Container in X3DOM

Follow the generic X3DOM approach:

Evaluate the general "**Declarative 3D**" **use cases and requirements** while providing a prototype system which works on **todays W3C/JavaScript/WebGL layer**

General Question: What Container are useful in todays W3C technology stack to support the "Generic Requirements"

- binary
- regular structure
- fast transmission, decoding
- must map to GPU container/buffer

“General Goals”

Increased User experience

User does not have to wait until the document is loaded

Increased Polygon count

From 0.3 Million to 10 Million Polygon

More data can be delivered in acceptable time

Increased Communication speed

Incremental Updates (similar to jpeg decompression)

Separate structure and data

HTML element reference external binary data element

DOM / HTML Document

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
        <head>
                <meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
                <title>Single Mesh Model with a flat hierarchy (small model)
</title>
        <link rel="stylesheet" type="text/css" href="x3dom.css" />
        </head>

<body>

    <h1>Single Mesh Model with a flat hierarchy (small model)</h1>

        <p class="case">
            <!-- Use the same "x", "y", "width" and "height" attributes as SVG.
             Optionally support the "viewBox" and "preserveAspectRatio" just
like SVG -->
                <X3D xmlns="http://www.web3d.org/specifications/x3d-namespace"
id="spheres" showStat="true" x="0px" y="0px" width="400px" height="400px">

  <Scene DEF='scene'>
    <Viewpoint position='0 0 25' orientation='0 0 0 1' />
    <Background skyColor='1 1 1'/>
   <Shape>
          <Appearance DEF='App'>
            <Material ambientIntensity='0.0243902' diffuseColor='0.41 0.39 0.03'
shininess='0.12' specularColor='0.94 0.72 0'/>
          </Appearance>
          <IndexedFaceSet DEF='_0' solid='false' creaseAngle='3.14'

0.048382'/>
          </IndexedFaceSet>
        </Shape>
        </Scene>
        </X3D>
        </p>

        <p class="case">   </p>

        <script type="text/javascript" src="x3dom.js"></script>
</body>
</html>
```

Binary asset resources

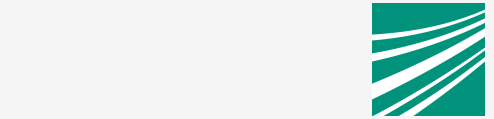
Images and Videos

- Encodes int/float arrays (e.g. coordinate, normal, texCoords, generic-attributes) in RGBA-images
- Multiple images per array
- Multiple images per scene

Explicit Binary Container

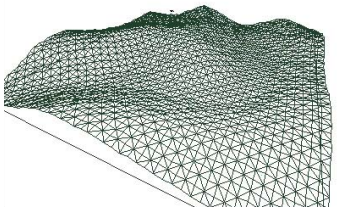
- Directly loaded to TypedArrays
- Data assignment in JS
- Multiple arrays per file
- Multiple files per scene

3D Geometry in Images



HeightMap

2D (semi)regular grid with 1D Height-Data



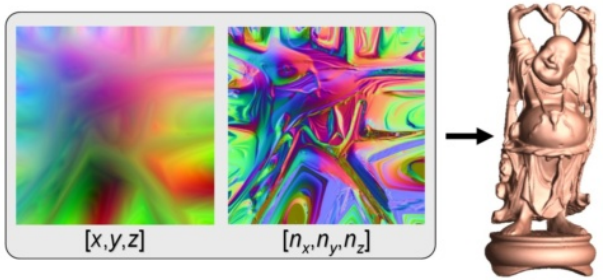
Geometry Images (Hoppe, Siggraph 2002)

Surface usually irregular triangle meshes

=> Remeshing to (semi)regular grid

pro: up/down sampling operation

con: genus-zero surface, **parametrization distortion, border-handling**

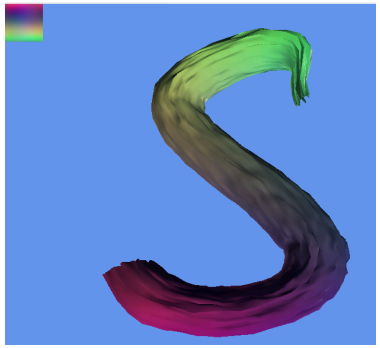


Latest development focus on multi-patch approaches and LOD structures

(see “Adaptive Quad patches” paper)

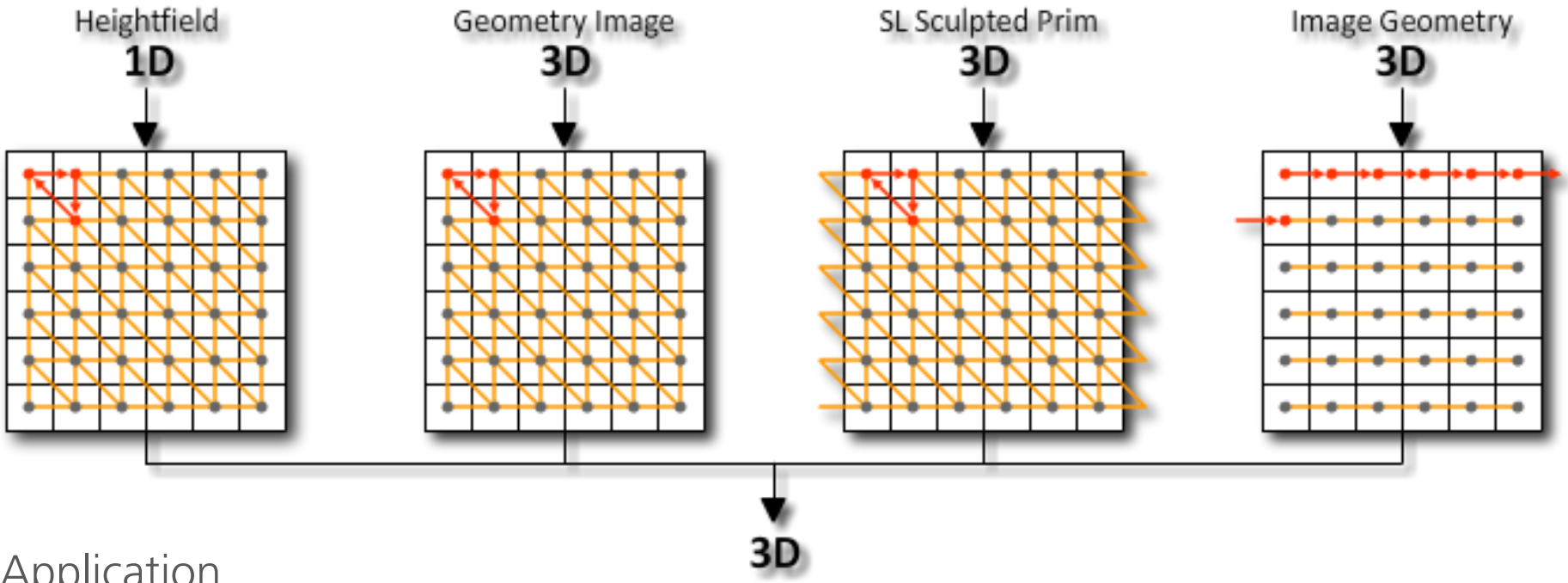
SL Sculpted Prims

Similar properties as Geometry Images



Idea: Sequential Image Geometry

Implicit mesh does not correlate with the mesh topology



Application

- Supports: Transmission, compression (partially), rendering, ...
- Does not support: Scaling of SIG container

Advantage: Works with any mesh type and keeps the original topology

<img>/<video> as generic binary container

Normalization and **linear Quantization** to 2^n Bytes: n is error/user controlled

Uses **multiple images** to distribute precision
(e.g. 1 Image -> 8bit, 2 images -> 16bit, …)

LOD and streaming of precision (e.g. closer objects use higher precision)

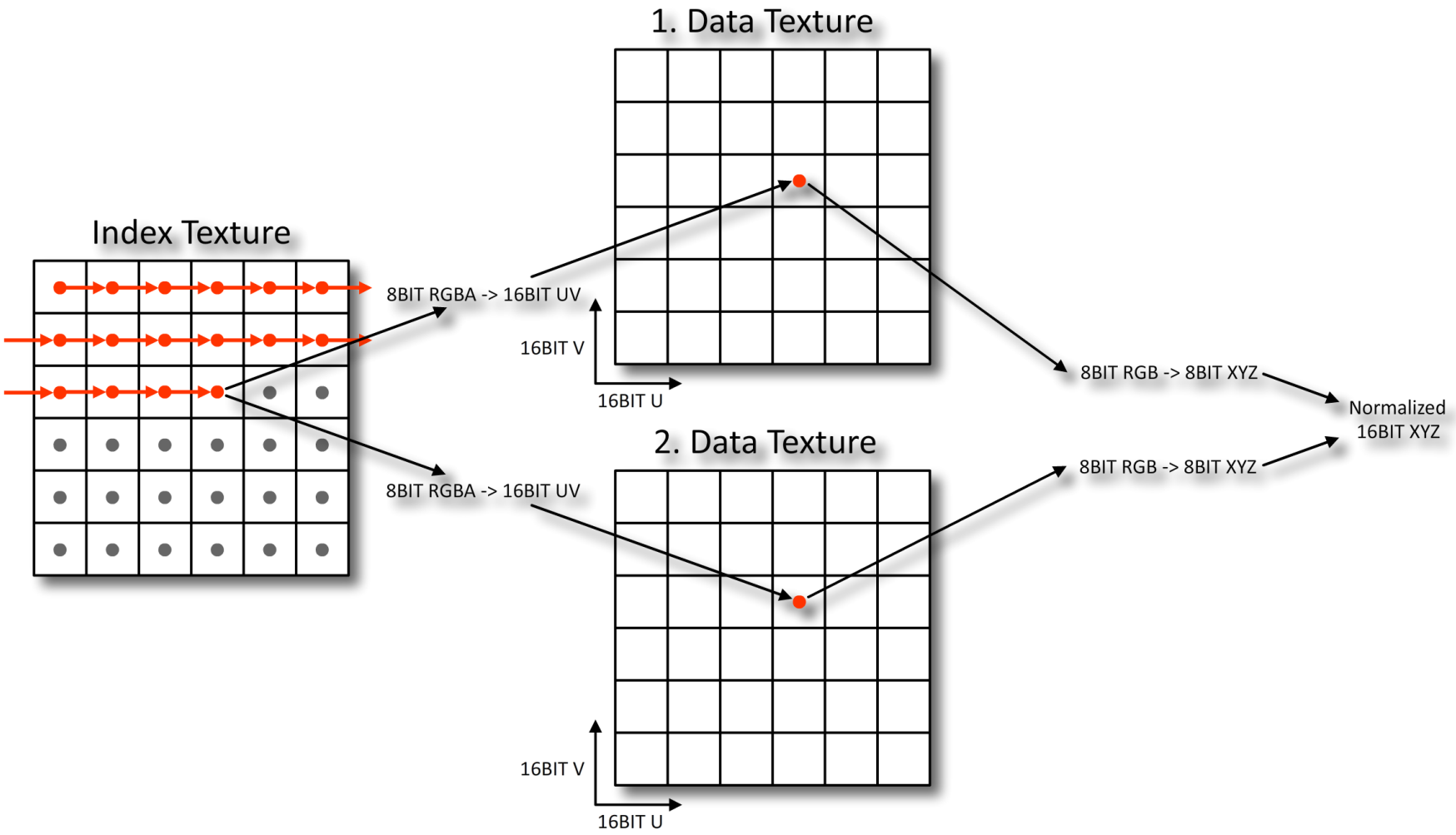
Decompression for free (only lossless png is useful right now)

Streaming updates for free: WebGL/X3DOM support <video>

Browser/Server well optimized to handle **large number of images and parallel downloads** of image => Great user experience

Developer, Browser, Server and W3C love images and video:
Content is HTML + image/video-resource data

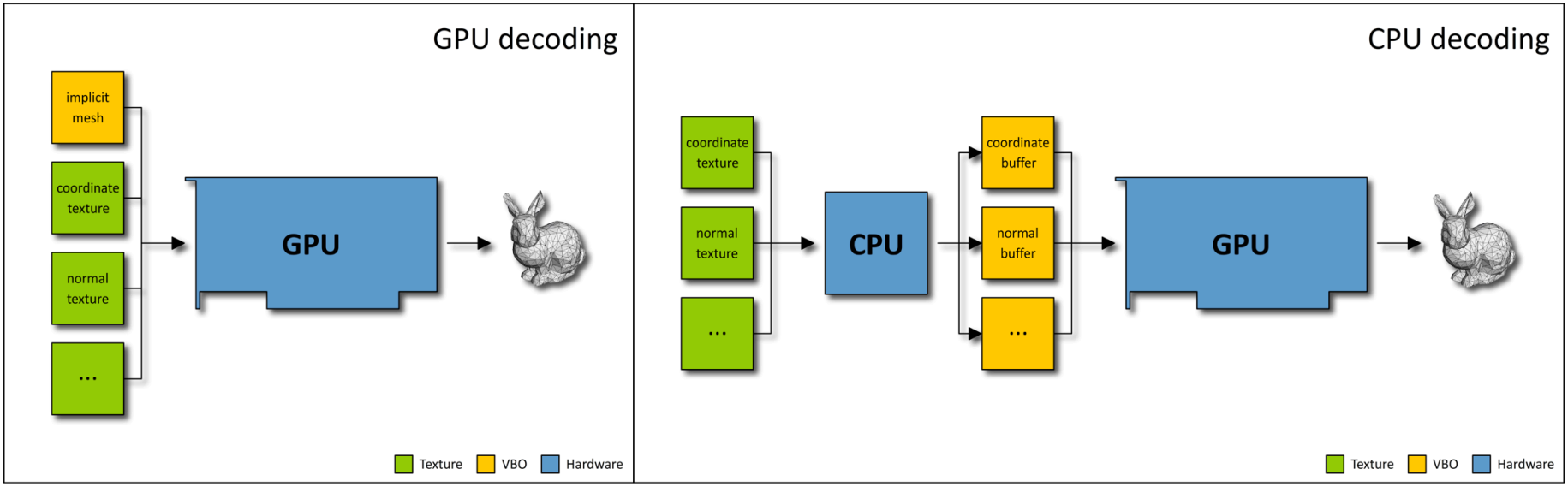
Multi image vertex property encoding



Data decoding and rendering

GPU: Single VBO, Extremely fast visualization with Vertex Textures Units, precision grows until vertex texture limit is reached

CPU/GPU: WebGL without Vertex Texture Unit support/ Flash 11



Binary Container

Powerful abstraction for efficient data encoding for Web-apps

Uses new **XHR ability to load binary ArrayBuffer**

Maps to **TypedArray/GPU buffer**

No **JS-Interaction for decoding**

Could be used for RESTful mesh attribute access

e.g. http://meshLand.com/mesh/32/coordinate.bin

Support **quantization with GPU based decoder**

(WebGL can handle 8 and 16 bit TypedArrays)

Standard rendering and shader handling

(Does not need support for Vertex Textures for GPU decoding as SIG)

Support also **incremental updates** through bit distribution over multiple files

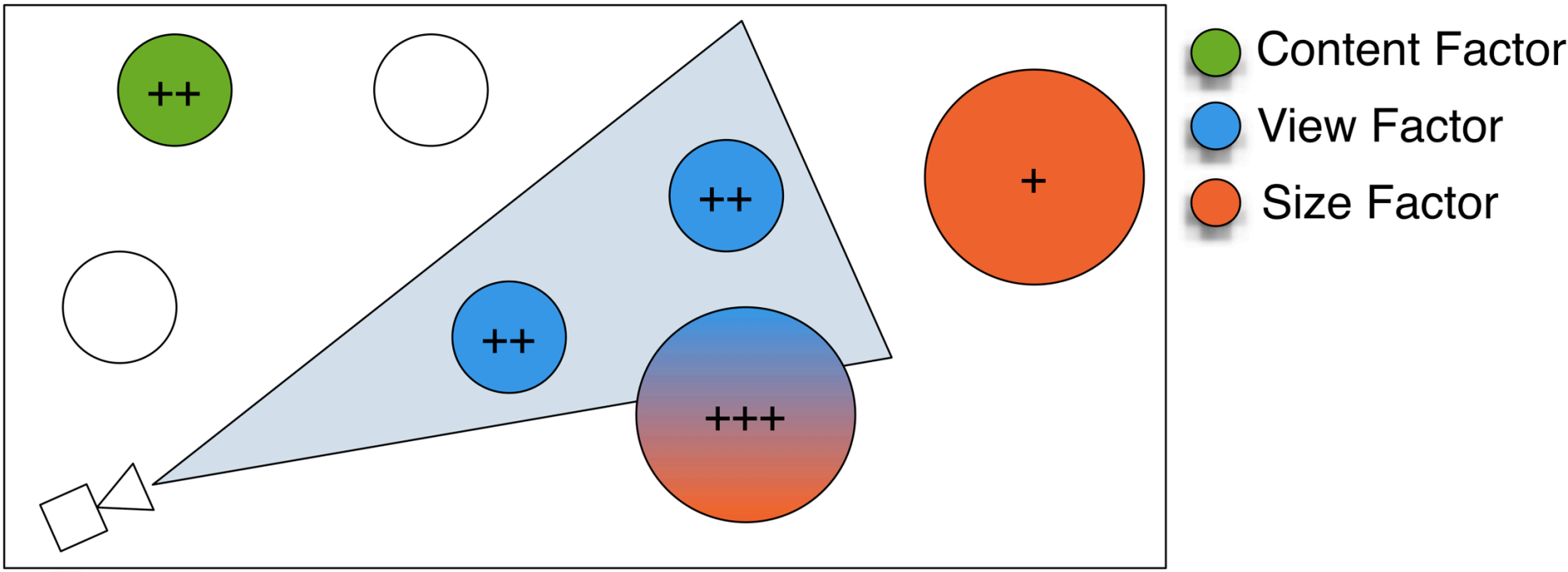
Need **bit-compositing in JS**

WebWorker and **Transferables** can help to decode in parallel

Priority Controlled Rendering

Priority controlled **download manager and renderer**

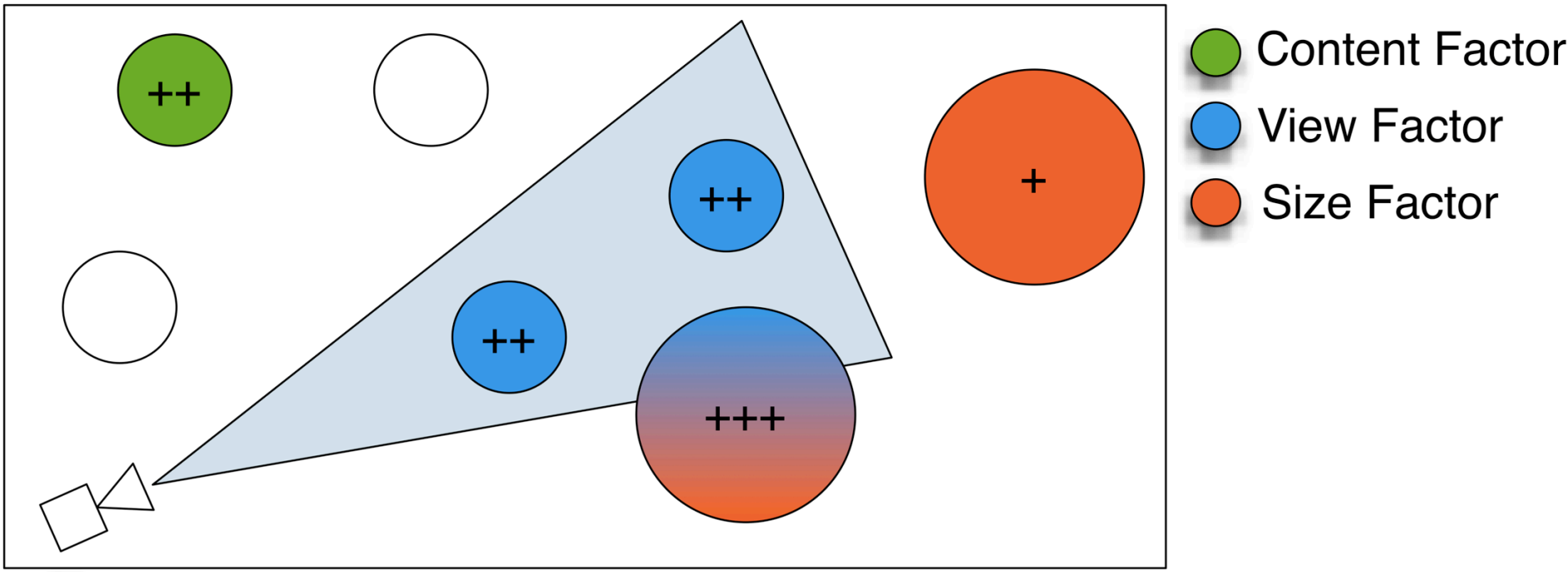
- **Content**: Use/Application given to focus on specific objects
- **View**: Objects which are in the view frustum
- **Size**: Objects which are bigger in world space
- **Data-Level**: Data which represents a more basic level get higher priority
- **External**: External Culling/Visibility service controls priority



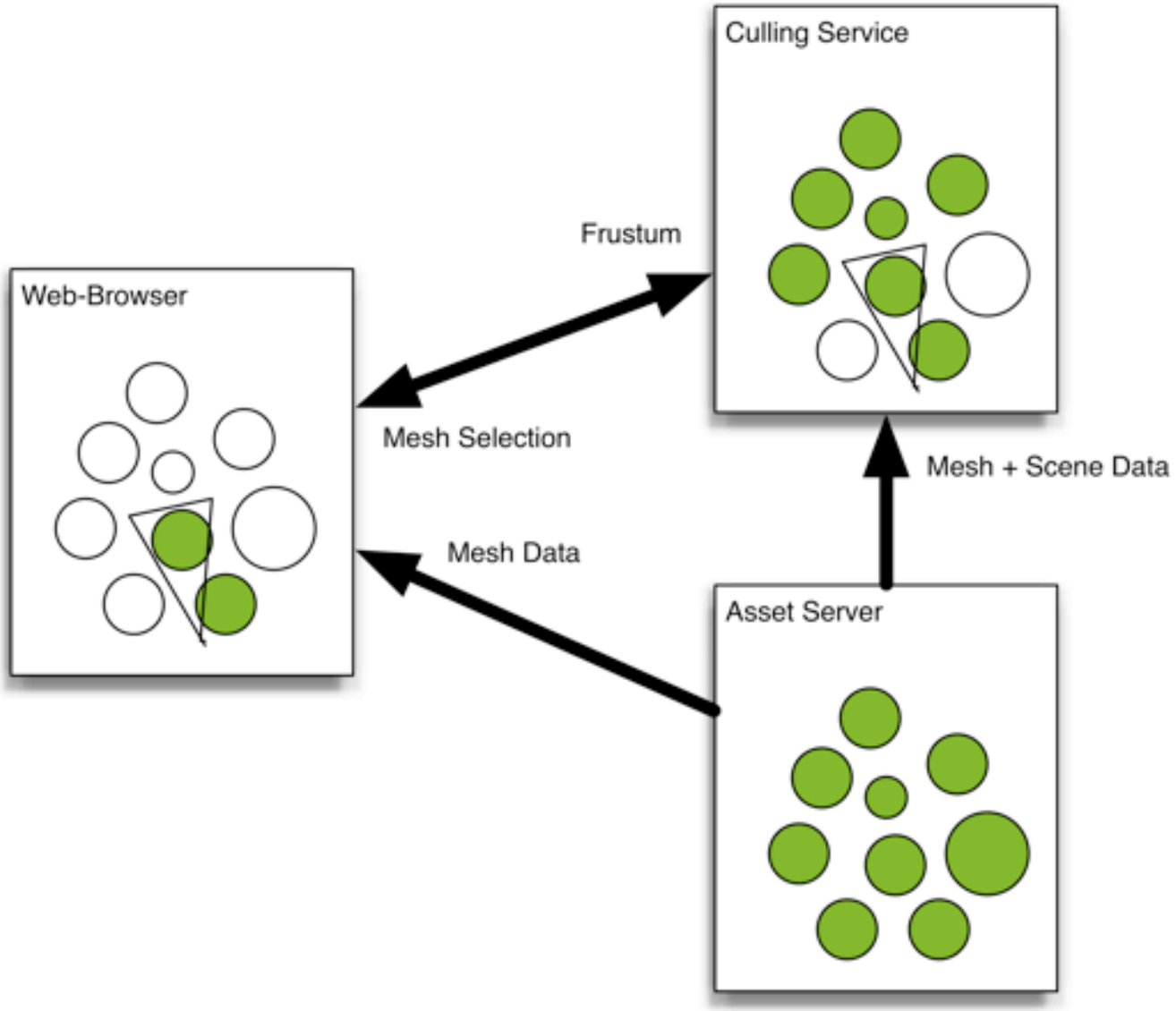
Priority Controlled Rendering

Priority controlled **download manager and renderer**

- **Content**: Use/Application given to focus on specific objects
- **View**: Objects which are in the view frustum
- **Size**: Objects which are bigger in world space
- **Data-Level**: Data which represents a more basic level get higher priority
- **External**: External Culling/Visibility service controls priority

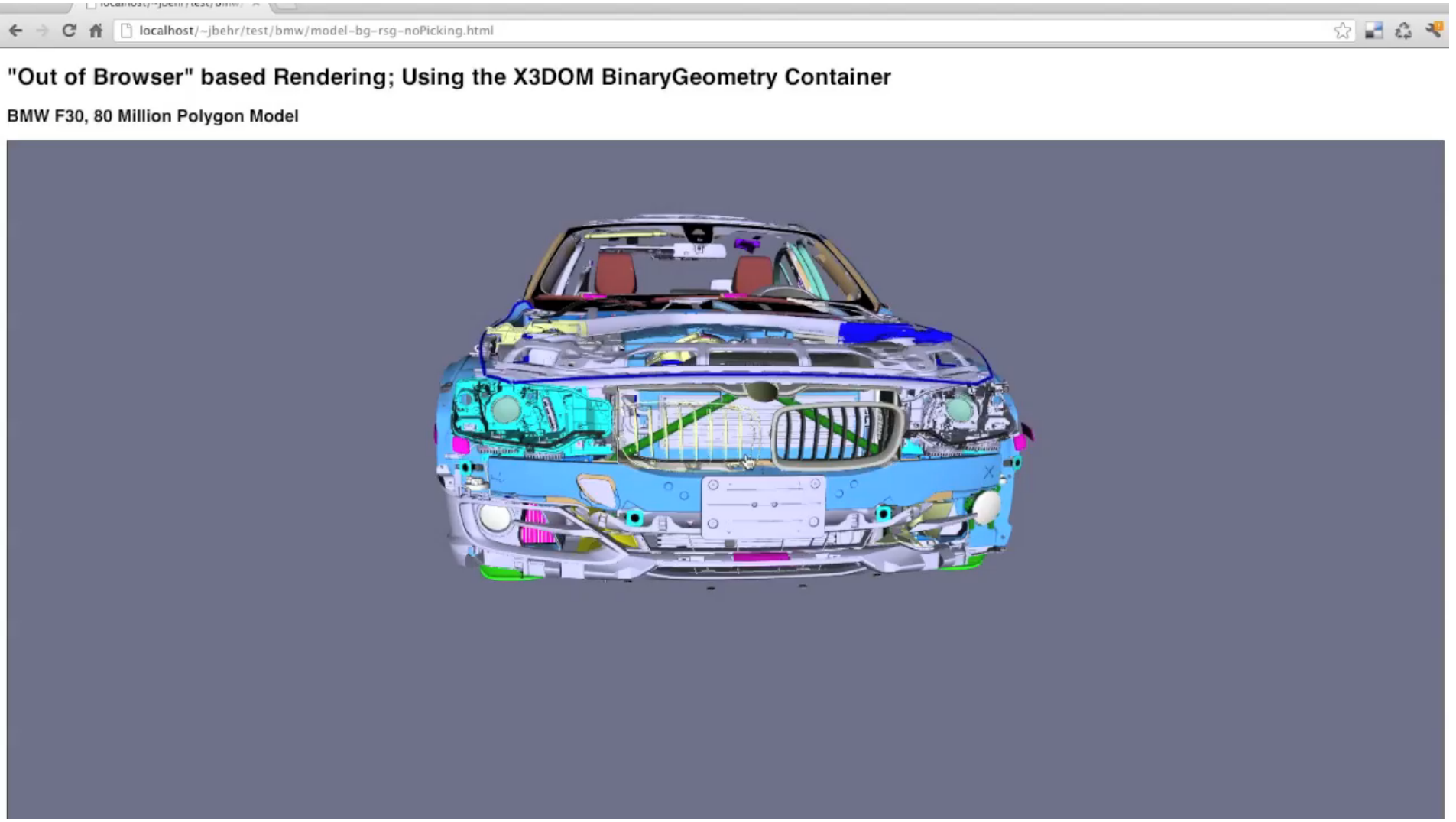


“Out of Core” Rendering with PCR

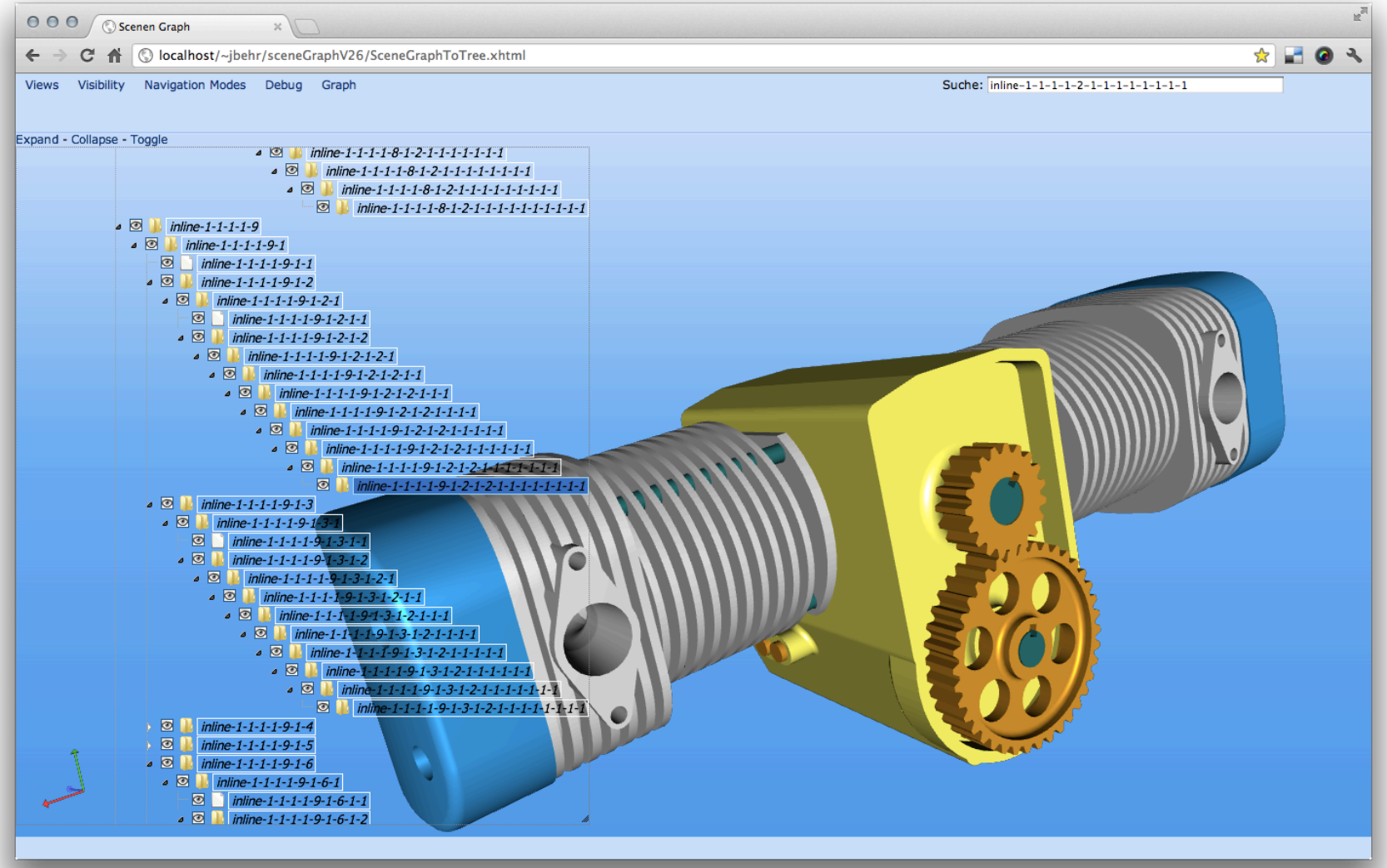


Service Controlled PCR

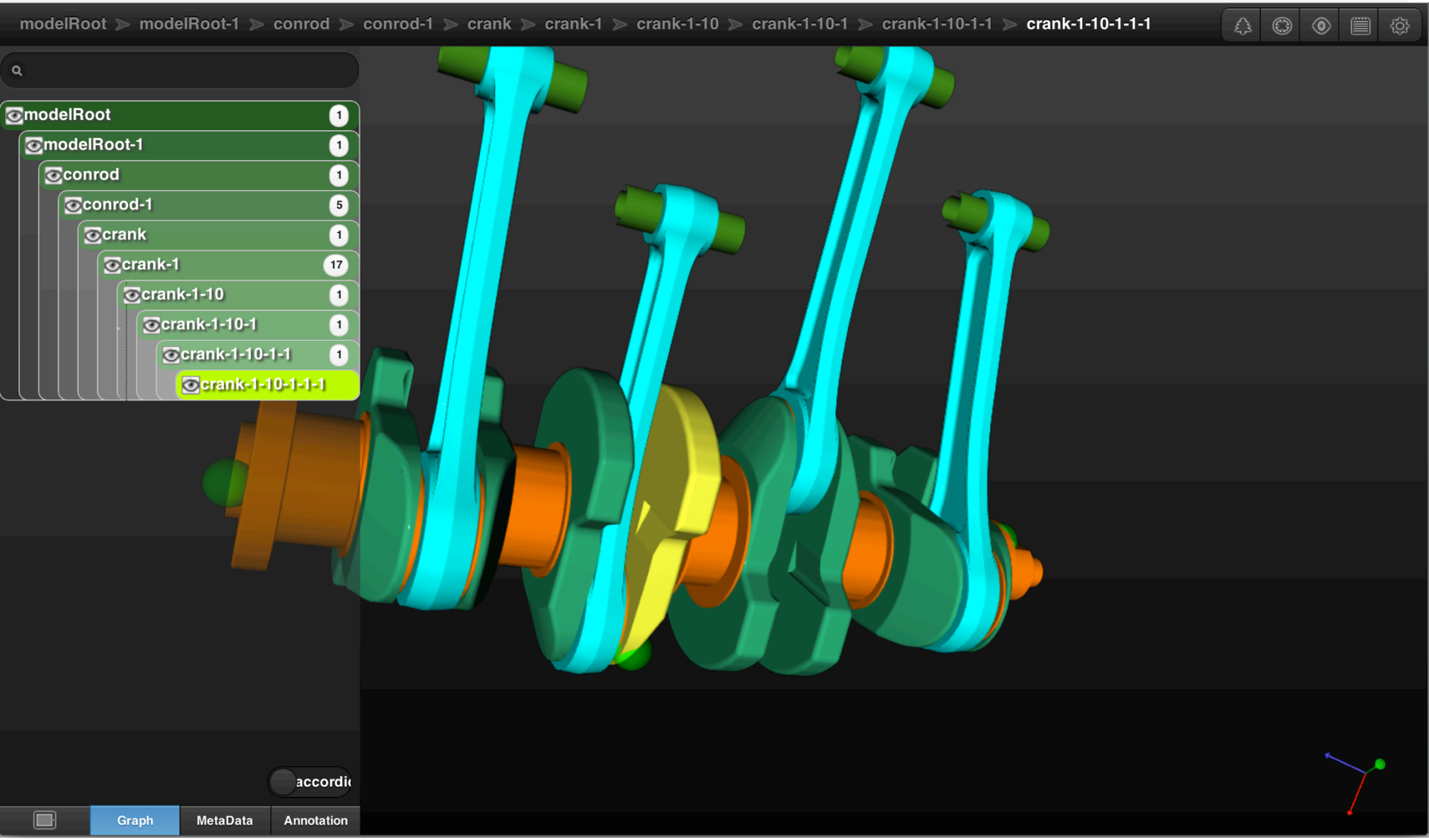
Uses bidirectional WebSocket connection to distribute computation



Application Example – Desktop



Application Example – Mobile

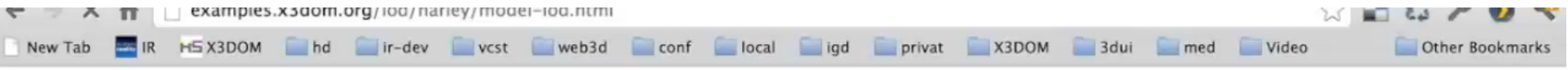


Combination with textures

Single container type can minimize Donwload-Management

Low Bandwidth / Mobile device

Online BG-LOD Examples over 3G



Implementation

Decoding & Rendering:

Open source and Part of X3DOM, available on github

http://www. x3dom.org

Patch creation and encoding:

Closed source aopt/instantReality 2.2 (release 3. August 2012)

Windows, Mac & Linux

http://www.instantreality.org

New “Large Datasets” tutorial on x3dom.org page

Free for “non commercial use”

Patch creation and encoding
Using the instantReality/aopt tool

Scene/Mesh statistics

aopt –I foo.x3d –p

aopt –I foo.x3d –J

Patch creation:

aopt –I foo.x3d –u –F subtree:”maxtris(20000)” –N foo-opt.x3d

subtree: Single Node (DEF/id), Node-Type or “Scene”

BinaryGeometry from PrimitiveSet

mkdir binGeo

aopt –i foo-opt.x3d –G binGeo/:saI –x foo-bg.x3d –N foo-bg.html

Tutorial online after siggraph!

Thanks

Demos: http://examples.x3dom.org