SRC
A Streamable Format for Generalized Web-based 3D Data Transmission

Max Limper, Maik Thöner, Johannes Behr, Dieter W. Fellner

Fraunhofer IGD / TU Darmstadt
Outline

• Motivation
• The SRC Format
• Integration into X3D
• Applications
• Summary
Motivation

• Many Web-aligned **formats** for **3D mesh data**
  (WebGL-Loader, X3DOM Binary Geometry, glTF, OpenCTM, ...)

• Still no widely accepted, common solution
• Different strengths / weaknesses
Motivation

• 7 Requirements for a common solution (1-5):
  1. Fast, direct / zero copy GPU uploads
  2. Possibility for progressive transmission
  3. #Downloads / #Meshes (#Draws) are decoupled
  4. Simple integration into declarative frameworks
  5. Data reuse and data compositing
...
Motivation

• 7 Requirements for a common solution (6-7):

...  
6. Possibility for different compression methods  
7. GPU-friendly integration of (compressed) texture data
## Motivation

<table>
<thead>
<tr>
<th>Feature</th>
<th>X3DB</th>
<th>gltF</th>
<th>X3DOM Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct / zero copy GPU Upload</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Progressive</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Separation #Downloads / #Meshes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Dec3D Integration</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Data Compositing</td>
<td>DEF/USE</td>
<td>Per File</td>
<td>Yes</td>
</tr>
<tr>
<td>Compression</td>
<td>Yes</td>
<td>Experimental</td>
<td>Quantization</td>
</tr>
<tr>
<td>GPU-friendly Texture Encoding</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
The SRC Format

• SRC = Shape Resource Container
• Structured header + binary file body
  – 3 Words pre-header:
    Format ID, version and encoding, header length
  – Various header encodings (currently: JSON only)
The SRC Format

• Some basic concepts from glTF, additionally:
  – Support for **progressive** transmission
  – **Declarative 3D** integration via X3D
  – Support for **data compositing** via X3D
  – Support for **quantized mesh data**
  – Support for binary (compressed) **texture data**
The SRC Format

- **Chunk** layer instead of (glTF) **Buffer** layer:

  \[ \text{Mesh} \rightarrow \text{Accessor} \rightarrow \text{Buffer View} \rightarrow \text{Chunk} \]

- Accessors in SRC: **IndexView** / **AttributeView**, quantization as basic **compression** via new **decodeOffset** / **decodeScale** attributes
The SRC Format

• Similar concepts for (compressed) **Textures**

\[ \text{Texture} \rightarrow \text{TextureImage} \rightarrow \text{Chunk} \]

• Compressed Textures: Separated transmission of MIP pyramid, 1 TextureImage = 1 MIP level
The SRC Format
The SRC Format

- Chunks enable **interleaved** transmission ...
  - ... of mesh data (e.g., vertex data and indices)
  - ... of mesh and texture data
Integration into X3D

X3DOM BinaryGeometry node:

```xml
<Shape>
  <Appearance>
    <Material diffuseColor='0.6 0.6 0.6'
      shininess='0.00234375'/>
    <ImageTexture url='"duck.png"'/>
  </Appearance>
  <BinaryGeometry vertexCount='12636'
    position='13.44 86.94 -3.70' size='165.47 154.04 115.25'
    primType='"TRIANGLES"' index='binGeo/indexBin.bin'
    coord='binGeo/coordBin.bin+8' normal='binGeo/normalBin.bin+4'
    texCoord='binGeo/texCoordBin.bin+4'
    coordType='Int16' normalType='Int8' texCoordType='Uint16'/>
</Shape>
```
Integration into X3D

*ExternalGeometry* node:

```
<Shape>
  <Appearance>
    <Material diffuseColor='0.6 0.6 0.6' shininess='0.00234375'/>
    <ImageTexture url='"duck.png"'/>
  </Appearance>
  <ExternalGeometry url='"duck.src"'/>
</Shape>
```

- Faster HTML parsing (esp. for large models)
- Other formats could also be used this way
Integration into X3D

X3D Scene

```xml
<X3D>
  ...
  <ExternalGeometry url='foo.src#mesh_1'/>
  ...
  <ImageTexture url='foo.src#tex_1'/>
  ...
  <ExternalGeometry url='foo.src#mesh_2'/>
  ...
<X3D/>
```

File 'foo.src'

<table>
<thead>
<tr>
<th>HEAD</th>
<th>MESH_1</th>
<th>TEX_1</th>
<th>MESH_2</th>
</tr>
</thead>
</table>

Vancouver, Canada 8 - 10 August 2014
Integration into X3D

Mesh data compositing with *Source* node:

```xml
<Shape>
  <Appearance>
    <Material diffuseColor='0.6 0.6 0.6'
      shininess='0.00234375'/>
    <ImageTexture url='duck.png'/>
  </Appearance>
  <ExternalGeometry url='duck.src'>
    <Source name='color'
      url='duckAltColors.src#mesh_1.color'/>
  </ExternalGeometry>
</Shape>
```

→ Details see paper 😊
Integration into X3D

*ExternalShape* node:

```xml
<ExternalShape url="duck.src"
bboxCenter='13.44 86.94 -3.70'
bboxSize='165.47 154 115.25'/>
```

- Even smaller HTML layout
- Inherits bbox fields from *X3DBoundedNode* → load SRC on demand
- No material data in SRC header → use X3D defaults
Applications

• Progressive mesh data representation
Applications

• Siena Cathedral Virtual Walkthrough

– Texture data much larger than mesh data
– Textures 241 MB as PNG, 78 MB compressed
– Direct GPU upload reduces waiting time
Applications

- Automotive & Energy CAD visualization

- More than 10,000 identifiable objects
- #Downloads crucial! (SRC: 1 Request per object)
- Size of HTML page crucial! (*ExternalShape* helps)
Summary

• SRC (Shape Resource Container) = structured header + binary file body

• Container format for mesh data and textures

• Simple, yet flexible, integration into X3D
## Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>X3DB</th>
<th>glTF</th>
<th>X3DOM Formats</th>
<th>SRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct / zero copy GPU Upload</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Progressive</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Separation #Downloads / #Meshes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Dec3D Integration</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Data Compositing</td>
<td>DEF/USE</td>
<td>Per File</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Compression</td>
<td>Yes</td>
<td>Experimental</td>
<td>Quantization</td>
<td>Quantization</td>
</tr>
<tr>
<td>GPU-friendly Texture Encoding</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Thanks for your attention!

Questions?