Visualization with WebGL

In-Browser 3D with AeroView

Who am I?

- **Matt Viss (matt.l.viss@boeing.com)**
- **Current: Future Production Systems**
- **BS Computer Science at Oregon State**
- **4 years with manufacturing systems**

Why are we exploring WebGL?

- **Current 3d viewers miss the mark**
  - **Clunky or no integration with other systems**
  - **Interface not designed for simple use cases**
  - **Slow startup times (database syncing)**
  - **Software updates and outages**
  - **License costs**
  - **No or limited mobile solutions**

What are we doing?

- **Create a new viewer**
  - **Easy to integrate with**
  - **Worldwide scaling**
    - **Application and content**
  - **Cross-platform (including mobile)**
  - **Meet or exceed performance of existing systems**
  - **Easy collaboration**
- **Select a lightweight format**
  - **No standard formats for loading 3d models (think .jpg for images)**
  - **Current standards are not optimized for read-only, runtime distribution**

What are we doing?

- **High level requirements**
  - **View/interact with lightweight geometry**
  - **Create annotations and views**
    - **Save, update for downstream systems**
  - **Display PMI (Product and Manufacturing Information)**
  - **Support Engineering Captures (Views)**
  - **Take accurate measurements**
- **Explore the possibilities!**

What aren't we doing

- **Loading gigantic models (eg full airplanes)**
  - **Existing Boeing applications for fly-through (IVT)**
- **Replacing existing CAD systems**
  - **Several existing web-based CAD systems**

What is WebGL and why should I care?

- **Javascript API to render 3D graphics directly on the GPU from any compatible web browser**
- **Subset of OpenGL ES**
  - **Supported by the same organization, Kronos Group**
  - **OpenGL is widely supported and hugely popular**
- **Direct access to the GPU**
  - **Faster than previous specs like VRML**
- **No licenses**
- **Wide commercial success**
  - **Unity game engine**
  - **Unreal game engine**
  - **AutoCAD 360**
- **Community Support**
  - **Threejs library**

How do I host it?

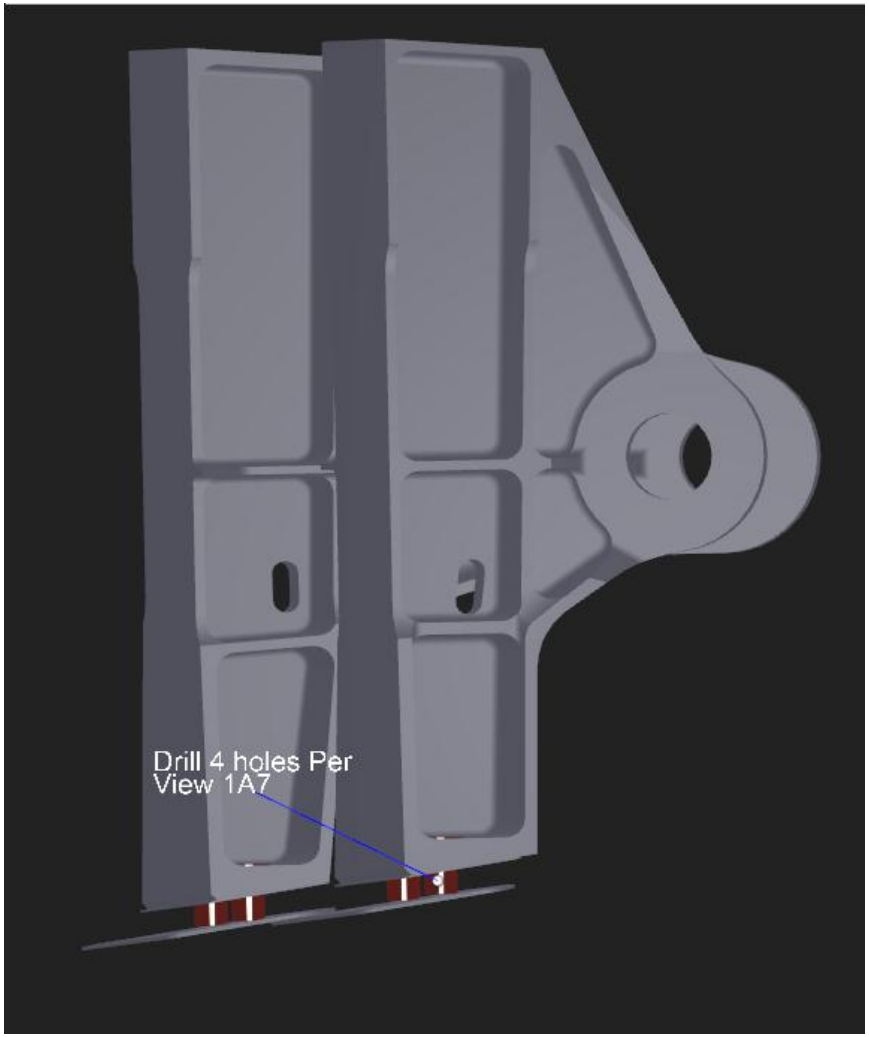
- **WebGL runs entirely on the client**
  - **You can open an html document containing WebGL directly with your browser. No hosting or special libraries required**
- **Use whatever you want for your backend**
  - **We use Flask to serve up html/css/js and standard python libraries for our database bindings**
- **Scaling follows the same procedure as any other web app**
  - **Multiple web servers with load balancing in front**

How do I use this thing?

- **Configurable mouse controls (threejs or custom)**
  - **Mimic existing systems like Catia for user comfort**
  - **Use basic orbit controls for ease of use**
  - **Fly-through for overall views**
- **Simple buttons and menus**
  - **Use any web-based UI elements**

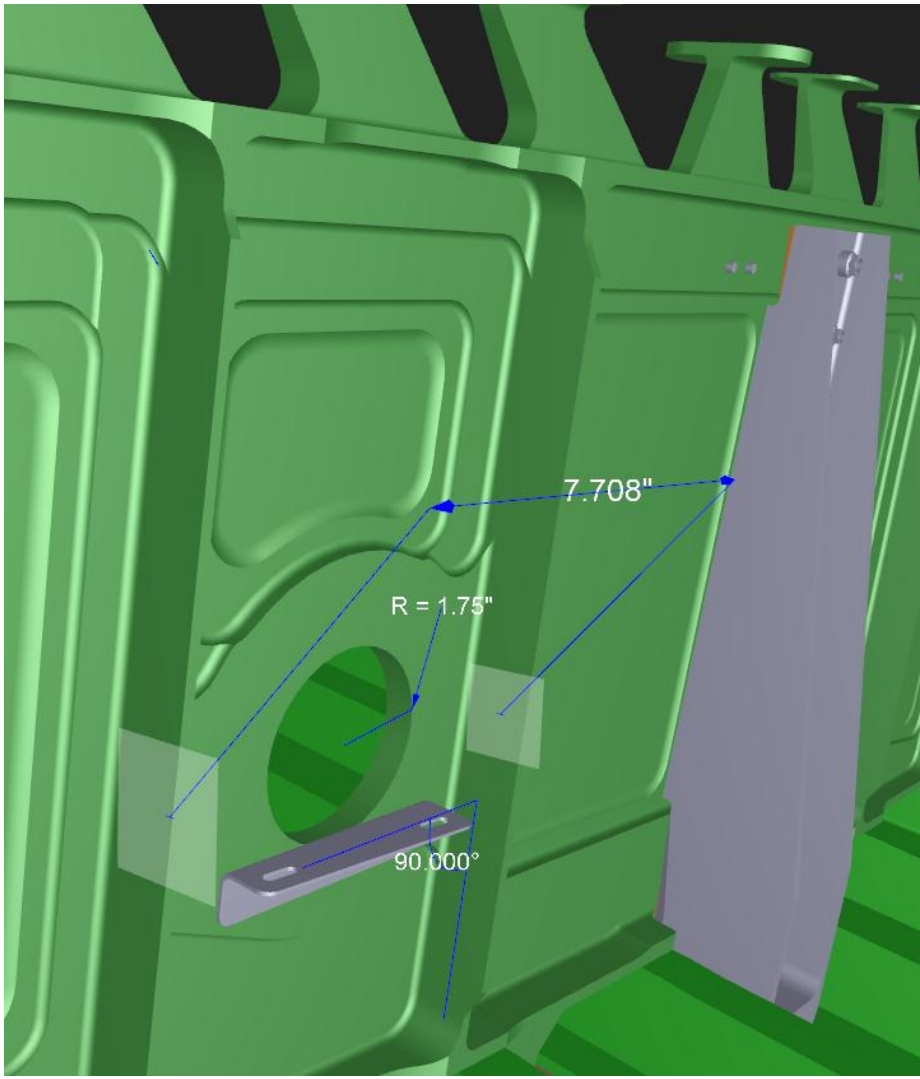
What can I do? Mark-ups.

- **Active billboarding**
- **Text is a separate layer that always appears on top**
- **Automatically size text based on camera distance**
- **Drag, edit, associate highlighted parts**



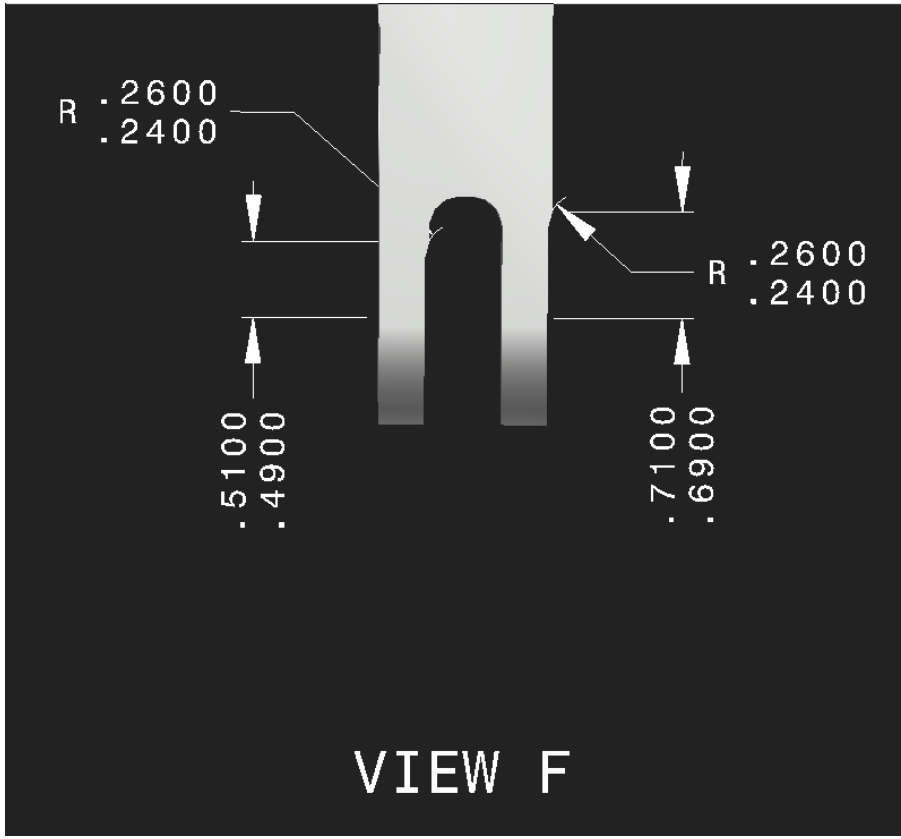
What can I do? Measure.

- **Auto-detect measurement based on selected feature (1 or 2 point)**
- **Distance**
  - **Between points, radii, planes**
- **Radius**
  - **High-accuracy method- patent pending**
- **Point position**
- **Angle between planes**



What can I do? View Captures.

- **Load engineering defined captures**
  - **PMI**
  - **Visible/hidden parts**
  - **camera location and target**
  - **Camera type (orthographic or perspective)**
- **Loaded from view dropdown menu (or spec tree)**



What can I do? Save views.

- **Create your view**
  - **Start from captures (or not)**
  - **Hide parts you don't want visible**
  - **Markup however you want**
  - **Save it!**
- **WebGL allows screen capturing**
  - **When saving views, we capture a screenshot for downstream systems. Use as preview or to load quickly on the shop floor**

What format is being used for geometry?

- **Custom JSON optimized for WebGL**
  - **Parse 3dxml/smgxml with python script**
  - **Convert all mesh types to triangles (from triangle strips, fans)**
  - **Combine small meshes, divide meshes over 16-bit limit**
  - **Compresses well with gzip**
    - **Browsers natively support gzip decompression, no javascript!**
- **Future move to glTF**
  - **Contains everything we strived for manually**
  - **CAD -> COLLADA -> glTF**
  - **Open standard by same group maintaining WebGL**
  - **Build with WebGL and OpenGL in mind**
  - **Threejs loader**
  - **.jpg for 3d graphics!**

What aren't you telling me?

- **16-bit indices buffer limit**
  - **Divide larger meshes in conversion process**
- **Javascript performance**
  - **Surprisingly good, constantly optimized but currently single-threaded**
- **JSON file size**
  - **256MB in firefox**
  - **Easy enough to break large geometries into multiple files**
- **32-bit process memory limit**
  - **4 GB memory limit for any 32-bit processes**
  - **Developer editions of firefox have 64-bit builds**

- **Questions?**