STEAM VR™

Tracking Training

STEAM® VR

Tracking Training

The Render Model

Key Concepts

- The render model is the image that appears in SteamVR™
- Render models are built in the same coordinate system
- Render models are comprised of at least:
  - Wavefront OBJ mesh
  - Material file MTL
  - Texture file PNG or TGA
- OBJ requirements
  - Single shape
  - Fewer than 65,000 vertices

Caveat Emptor

- The process presented here is simplified
- The goal is a representative render model for tracking evaluation
- Production render models have more features
  - Better UV maps
  - Finer textures
  - React to SteamVR™
  - Expert 3D artists can do this work
- Engineers need tracking results
  - Could reference a application render model
  - Better to make a simple model that represents the object

Process Overview

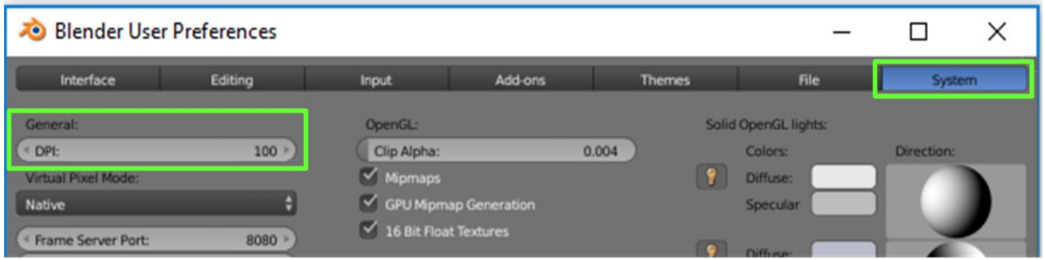
- Create an STL file of the reference controller
- Create an STL file of the object (suitable for the render model)
- Import both into the same coordinate system
- Align your object to the reference model
- Export the aligned STL as the basis for the render model OBJ
- Import the aligned STL into Blender
  - Create a material
  - Create a texture
  - Create a UV map
  - Export the OBJ, MTL, and PNG
- Try the render model in MeshLab
- Try the solid render model in VR!

Start the Exercise

- We need to align our render model to a reference render model
- VR applications replace the controller render model
- Which model to use?
  - The model most content is written to support
  - The HTC Vive controller
- Let’s make an STL of the Vive controller...

Set Preferences in Blender

- Open Blender
- File > User Preferences…
- Adjust: System
  - DPI to increase the size of controls
- Check: Interface
  - Auto Depth
  - Zoom to Mouse Position



General:

DPI: 100

Virtual Pixel Mode:

Native

Frame Server Port: 8080

OpenGL:

Clip Alpha: 0.004

☑ Mipmaps

☑ GPU Mipmap Generation

☑ 16 Bit Float Textures

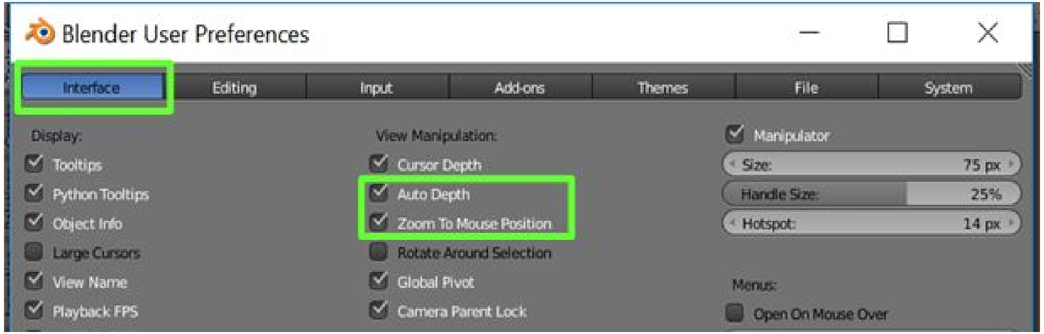
Solid OpenGL lights:

Colors:

Diffuse:

Specular

Direction:



Display:

☑ Tooltips

☑ Python Tooltips

☑ Object Info

☐ Large Cursors

☑ View Name

☑ Playback FPS

View Manipulation:

☑ Cursor Depth

☑ Auto Depth

☑ Zoom To Mouse Position

☐ Rotate Around Selection

☑ Global Pivot

☑ Camera Parent Lock

☑ Manipulator

Size: 75 px

Handle Size: 25%

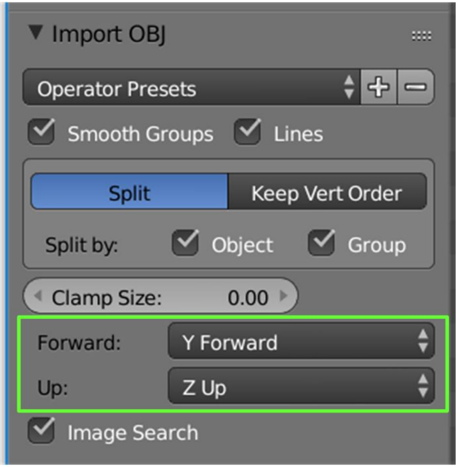
Hotspot: 14 px

Menus:

☐ Open On Mouse Over

Create the Reference STL

- Delete the default cube
- Import the Vive Controller OBJ
  - 150_the_render_model\vr_controller_vive_1_5\vr_controller_vive_1_5.obj
  - Watch out for the import settings
    - Lower left of the import screen
  - Y = Forward, Z = Up



▼ Import OBJ

Operator Presets

☑ Smooth Groups ☑ Lines

Split | Keep Vert Order

Split by: ☑ Object ☑ Group

Clamp Size: 0.00

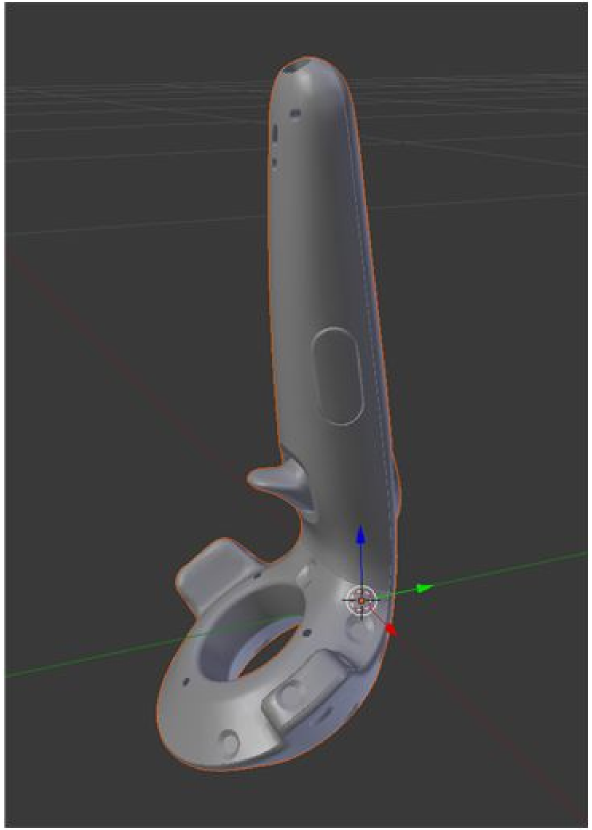
Forward: Y Forward

Up: Z Up

☑ Image Search

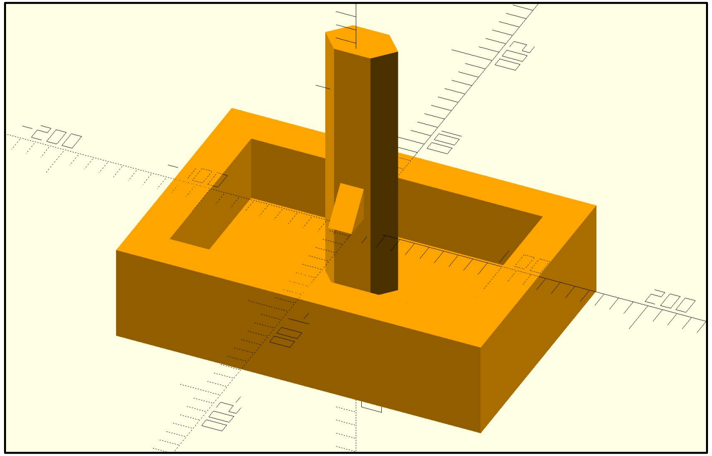
Export the Reference STL

- This is the render model coordinate system
  - What is the origin and orientation relative to a hand?
- Export the STL into the exercise folder
  - 150_the_render_model\htc_vive_controller.stl



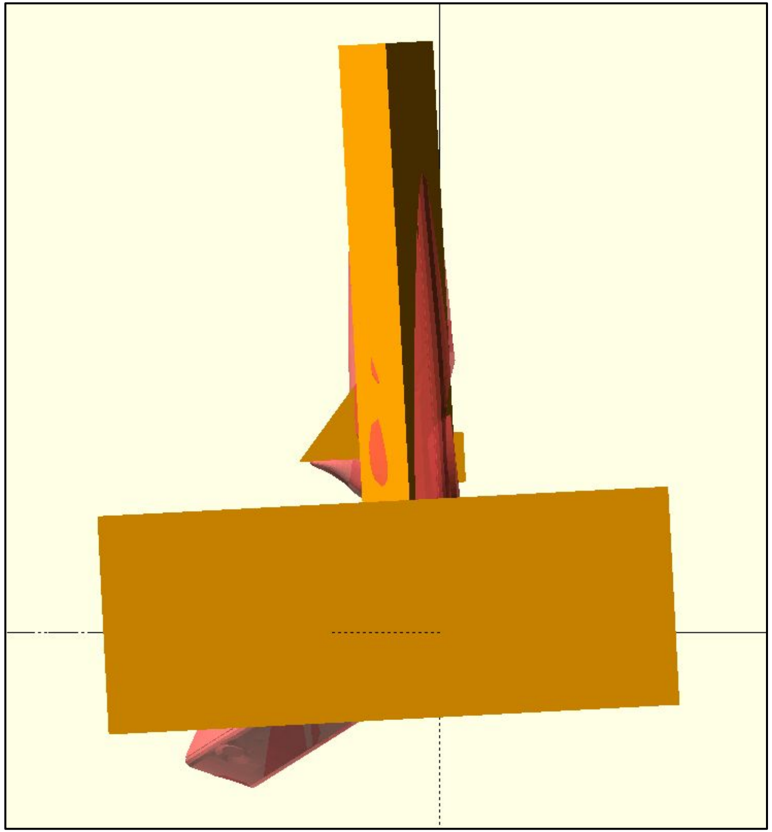
Import Both into OpenSCAD

- thor_hammer.stl is our object’s STL file
  - Simplified for:
  - Lower vertex count
  - Unwrapping the UV map
- Import both STLs into OpenSCAD
- Open in OpenSCAD
  - 150_the_render_model\align_controllers.scad
- **Where is the Vive controller?**



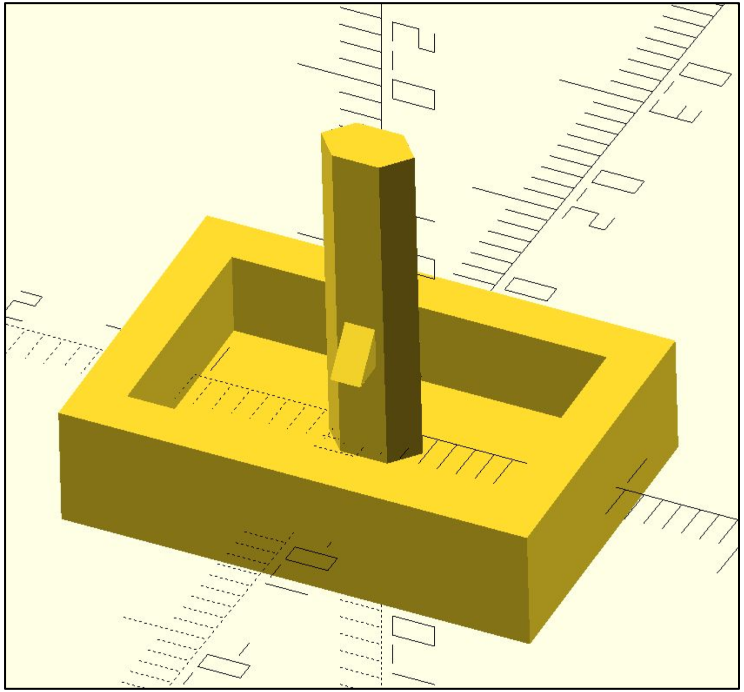
Align the Models

- Adjust the scale to meters
- Make the Vive controller transparent
    - Use '#' before the model
- Adjust the translation and rotation of Thor's hammer
    - Align the triggers
    - Align the trackpad and button
    - Make the handles parallel



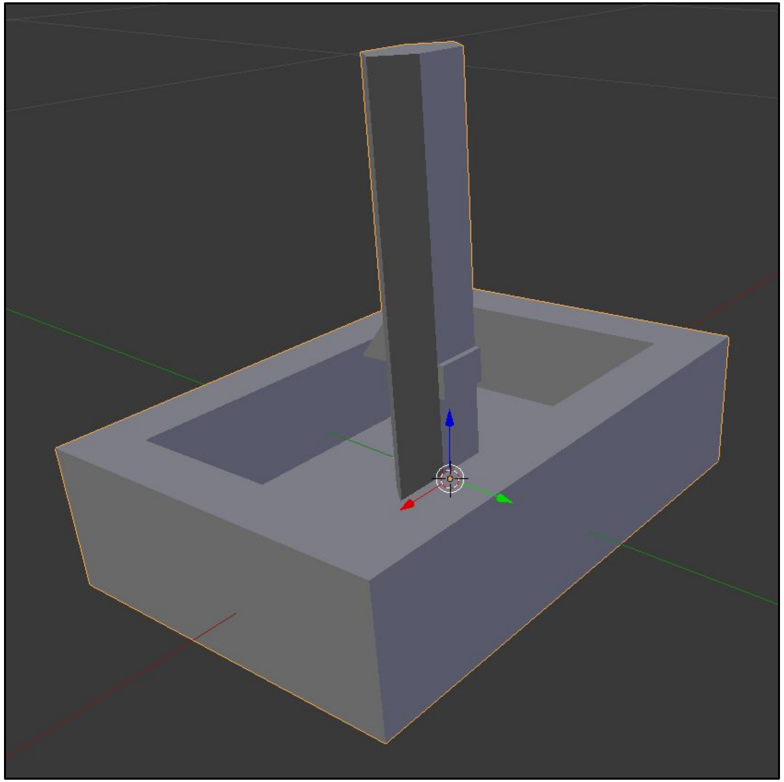
Export the Render Model STL

- Solo Thor’s Hammer
  - Use ‘!’ in front of the hammer
- Build the object
  - Press F6
- A more complicated mesh takes time to build
- File > Export STL…
  - 150_the_render_model\thor_aligned.stl



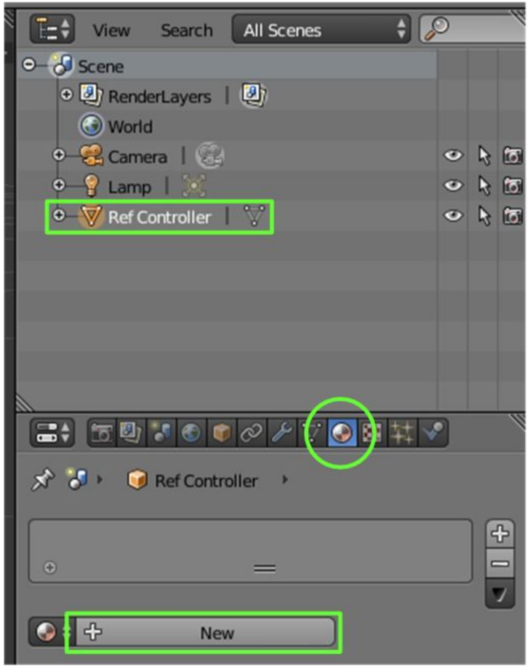
Import STL Into Blender

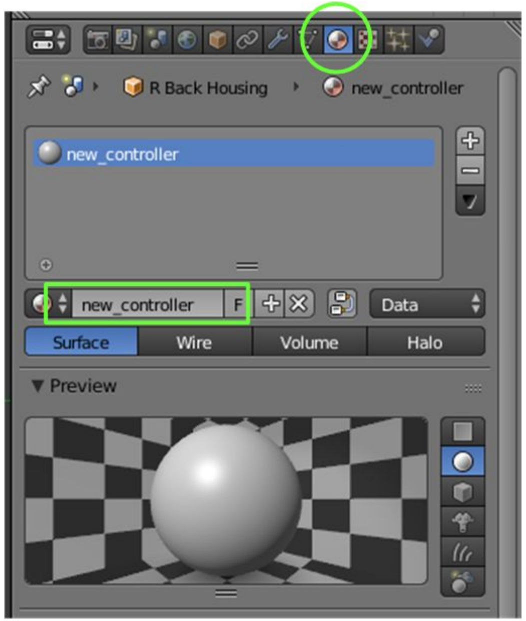
- Open a new Blender file
- Delete that cube
- Import Thor's hammer
  - 150_the_render_model\thor_aligned.stl



Add a Material

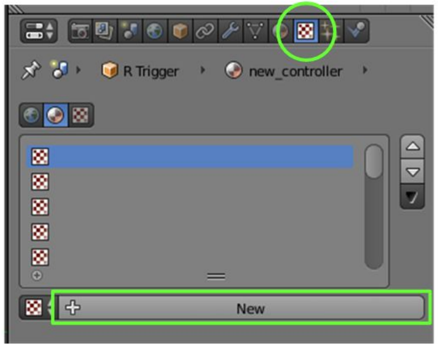
- Select the mesh
- Click the material button
- Give the material a name

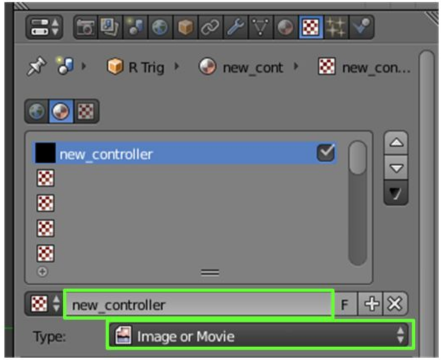




Add a Texture

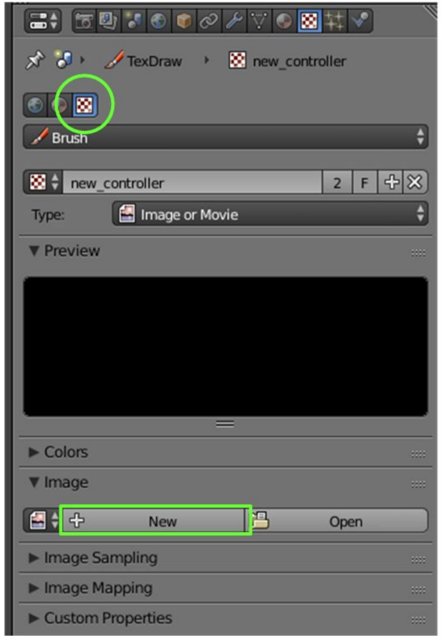
- Click the texture button, +New, Image or Movie, +New
- Give the texture a name and select UV Grid





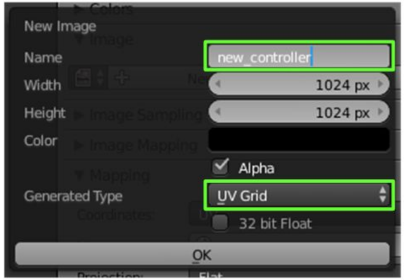
new_controller

Type: Image or Movie



new_controller 2 F

Type: Image or Movie


New Image

Name: new_controller

Width: 1024 px

Height: 1024 px

Color

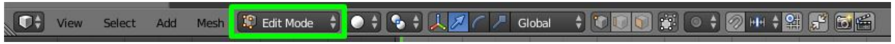
☑ Alpha

Generated Type: UV Grid

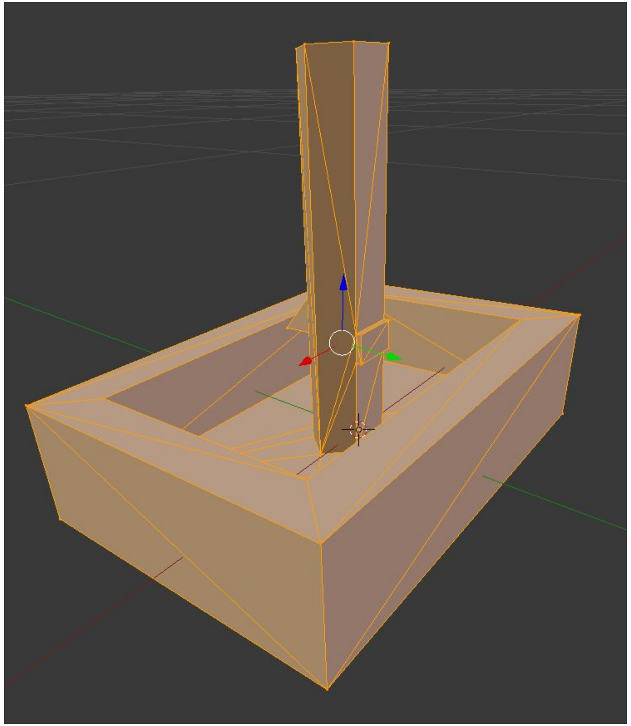
□ 32 bit Float

Create a UV Map

- **Who can explain what a UV map is?**
- This will be a bad UV map…
  - If we don't create one at all the render model will fail to load
- Switch to Edit Mode

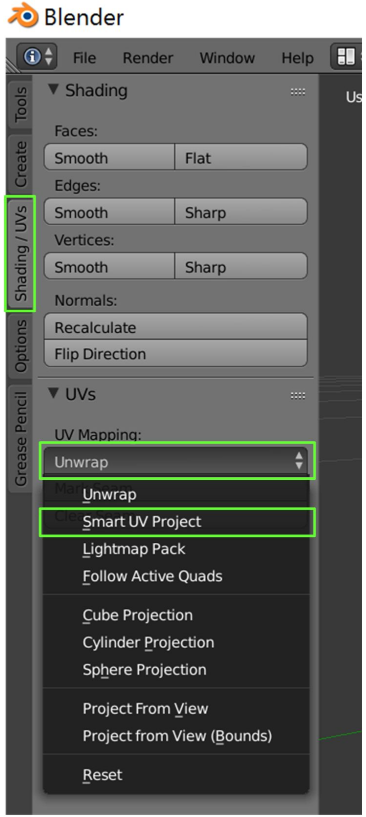


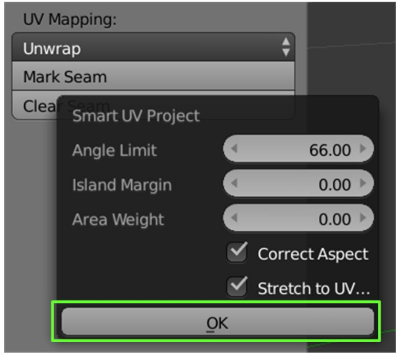
- Press 'a' to select all



Create a UV Map

- Unwrap the object
- Click
  - Shading/UVs
  - Unwrap
  - Smart UV Project
  - OK





Smart UV Project

Angle Limit 66.00

Island Margin 0.00

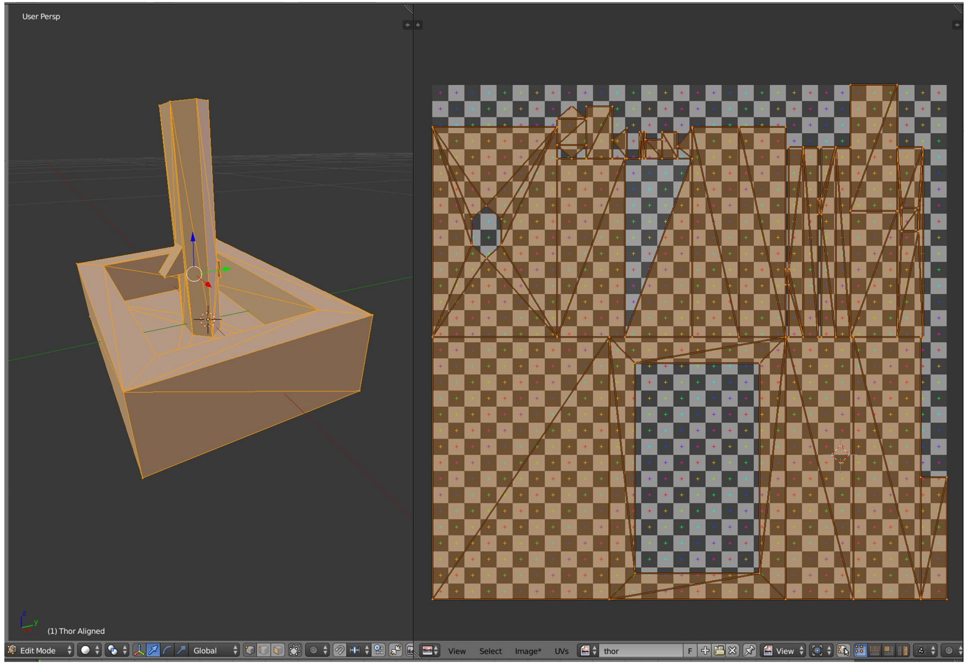
Area Weight 0.00

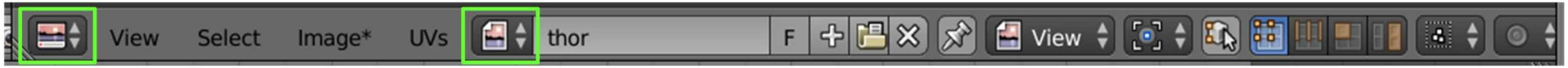
☑ Correct Aspect

☑ Stretch to UV...

Show the Two Side by Side

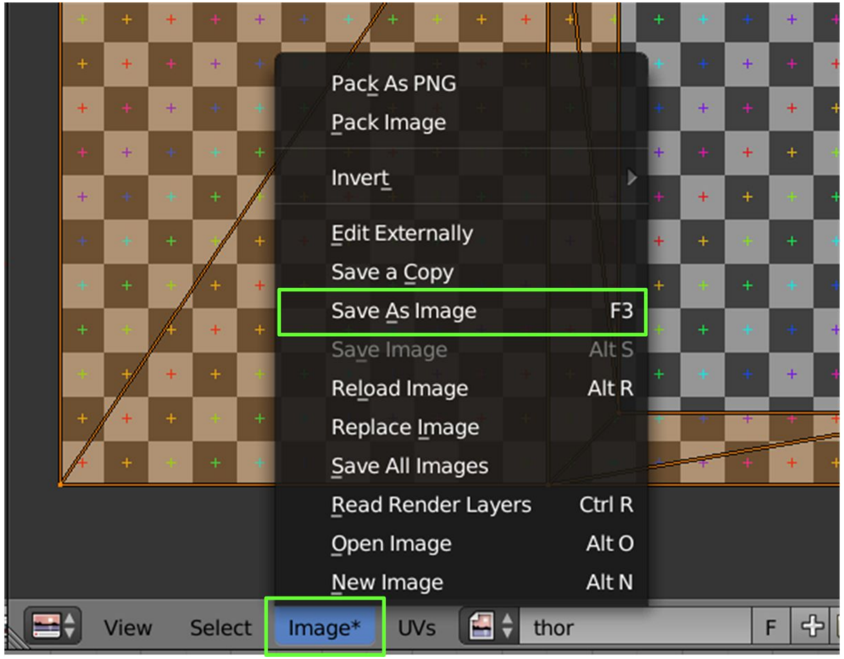
- Drag out a new window
- Select UV/Image Editor
- Select all ‘a’





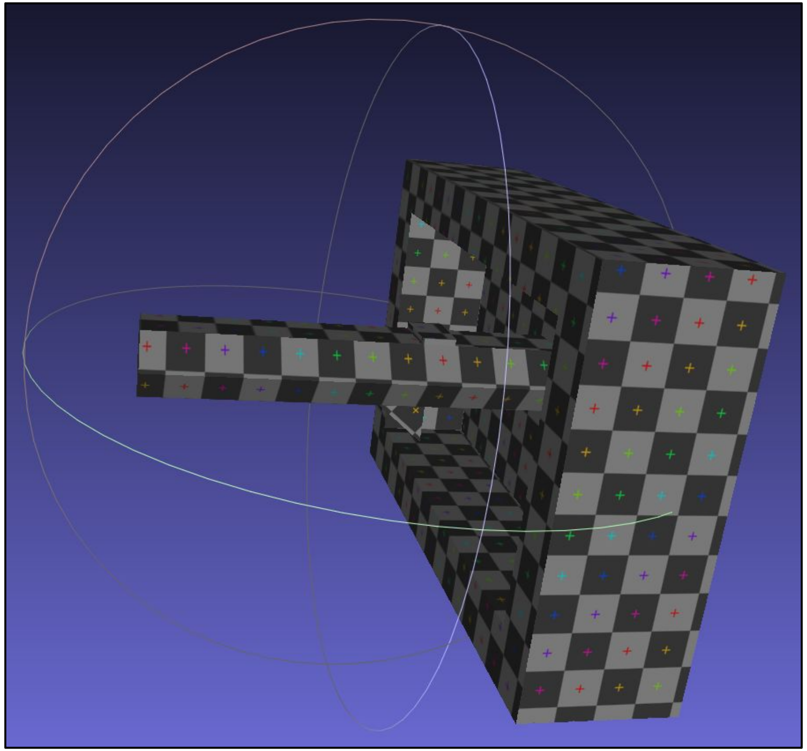
Save the Texture Image

- Click
  - Image
  - Save As Image
- Save the file in:
  - 150_the_render_model\



Export the OBJ

- Save the Blender File
  - File > Save As...
- Export the OBJ
  - File > Export > Wavefront (.obj)
  - Y = Forward, Z = Up
  - Save into: 150_the_render_model
- Open the OBJ in Meshlab!



SteamVR™!

- Render models are stored in the following directory
  - `C:\Program Files (x86)\Steam\steamapps\common\SteamVR\resources\rendermodels`
- Create a folder with your name
  - Add the OBJ, MTL, and PNG
  - The OBJ name must match the folder name
  - If you change other file names, watch out for the references
- Bring it up to the front and we will try it out!
- **How accurately is it tracking the real object?**
- **How could we align it better?**

Troubleshooting the Render Model

- The render model may not load
- Look in the compositor log file for clues…
  - C:\Program Files (x86)\Steam\logs\vrclient_vrcompositor.txt
  - Search for your render model name
- More than one shape? Too many vertices? No UV map?

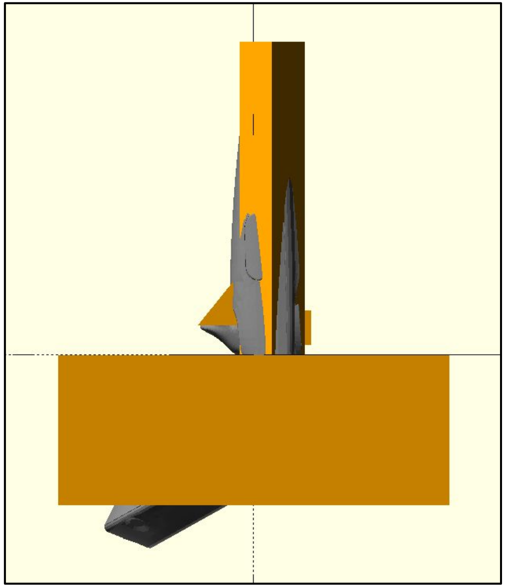
```
<date> - Render model from <rendermodels>\ref_controller\ref_controller.obj contained
more than one shape. We only support one.
```

```
<date> - Render model <rendermodels>\ref_controller\ref_controller.obj has 313189
vertices. Only 65k are supported
```

```
<date> - Render model <rendermodels>\ref_controller\ref_controller.obj has 0 texture
coordinates, expected 82994
```

Dialing in the Head Variable

- **Remember what the “head” variable means?**
- Align the render model with the original STL
  - Reverse the translation and rotation from before
- Use the position in the “head” variable
- Turn the rotation into unit vector components
- Upload the new JSON file
- **Now how does the render model match reality?**



Summary

- The render model is the image that appears in SteamVR™
- All render models are built in the same coordinate system
- Requirements
  - Single shape
  - Fewer than 65,000 vertices
  - OBJ with UV map, MTL, PNG
- Deploy it in the rendermodels folder
- Reference it in the JSON file
- Check vrclient_vrcompositor.txt for errors
- Talk to a 3D artist to get real material properties and textures!