

Interaction helpers

Trackball Animation and Hotspots placement

3DHOP WORKSHOP – 28/09/2015

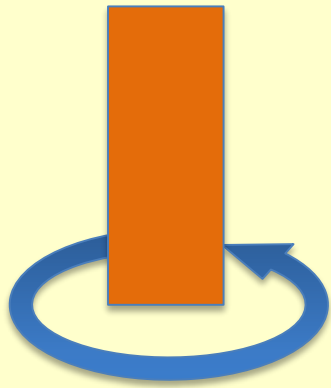
Trackball Animation

Tips and Tricks

Trackball state

Each trackball has a STATE, a set of parameters describing the current position and orientation of the trackball.

By calling the function `presenter.getTrackballPosition()` you will get the current STATE of the trackball, an array `[]` of parameters.

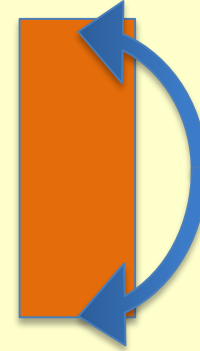
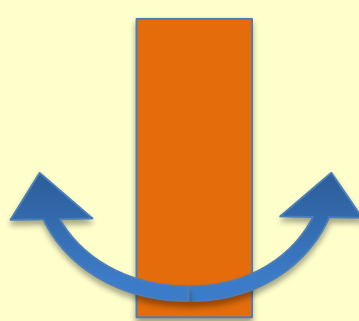
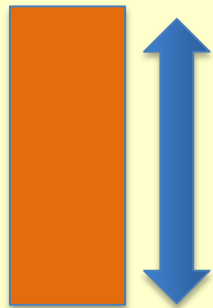
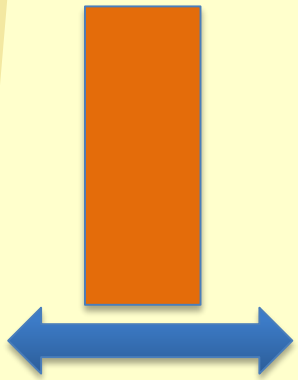


For example, the default TURNTABLE trackball is an array of 3 values: THETA (horizontal rotation), PHI (vertical rotation), DISTANCE (zoom)

Trackball state

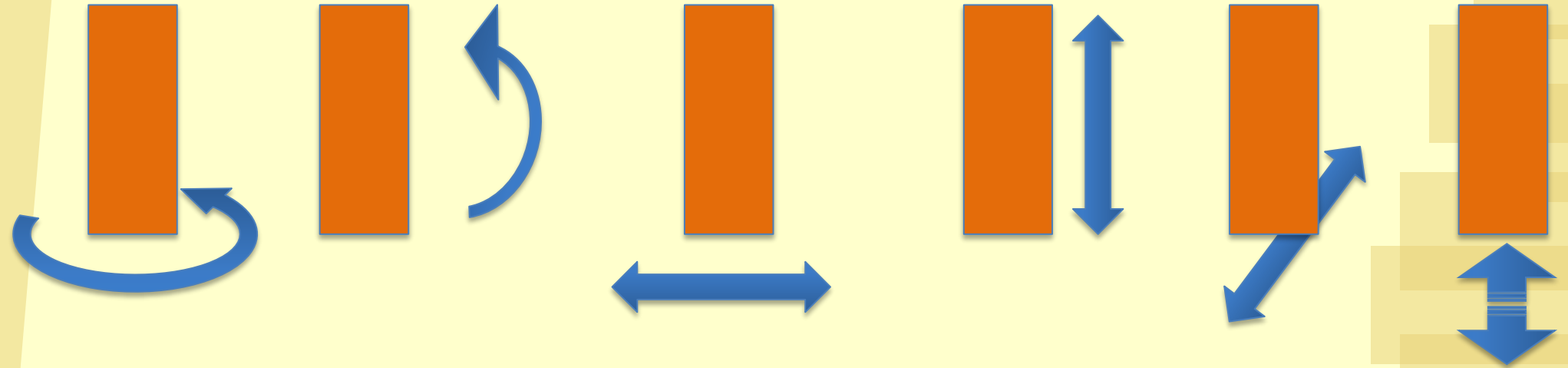
PANTILT trackball is controlled by 5 parameters:

PANX (pan on X axis), PANY (pan on Y axis),
ANGLEX (horizontal tilt), ANGLEY (vertical tilt),
DISTANCE (zoom)



Trackball state

TURNTABLEPAN trackball is controlled by 6 parameters:
THETA (horizontal rotation), PHI (vertical rotation),
PANX (pan on X axis), PANY (pan on Y axis), PANZ (pan on Z axis),
DISTANCE (zoom)



Anim-able trackballs

3DHOP trackballs can be animated using javascript function

`I presenter.setTrackballPosition([newstate])`

Instantly change the trackball position to a new one

`I presenter.animateToTrackballPosition([newstate])`

Animate the trackball from current position to a new one

[newstate] is an array of values, describing a new state. Of course the number of values and their meaning depend on the trackball used.

But how I do I choose the new state I need?

Eyeballing & reasoning

You may “guess” or “eyeball” the position you need to reach

- ◆ Angles are in degree, 0 is *straight ahead*, angles may be positive (clockwise) or negative (counter-clockwise)
- ◆ Pan (on X, Y, Z) is expressed in “scene size”: 1 is the size of the scene (the diagonal of the bounding box). Pan 0 means the object is centered; Pan may be positive or negative.
- ◆ Distance is also expressed in “scene size”, Distance 0 means inside the object; greater Distance means farther from the object

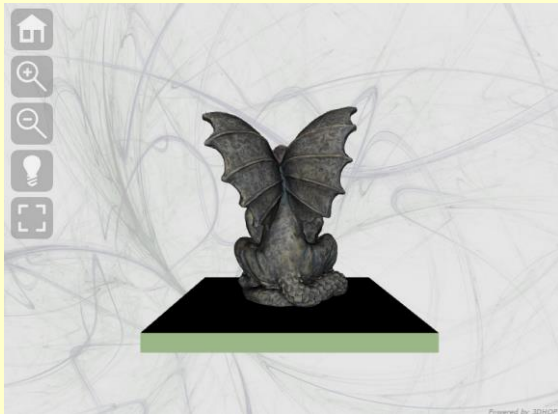
Eyeballing & reasoning

e.g. for the TURNTABLE trackball:

[180.0, 0.0, 3.0] looking at the object from behind, straight, far away

[45.0, 30.0, 1.0] $\frac{3}{4}$ tilted view, up close

[-45.0, 30.0, 1.0] opposite of the above $\frac{3}{4}$ tilted view, up close



Eyeballing & reasoning

BEWARE:

- ◆ If the number of parameters is wrong, nothing will happen (you *may* see an error in the console)
- ◆ If you ask the trackball to reach a position outside its limits, it will try to go there, but will stop at the limits

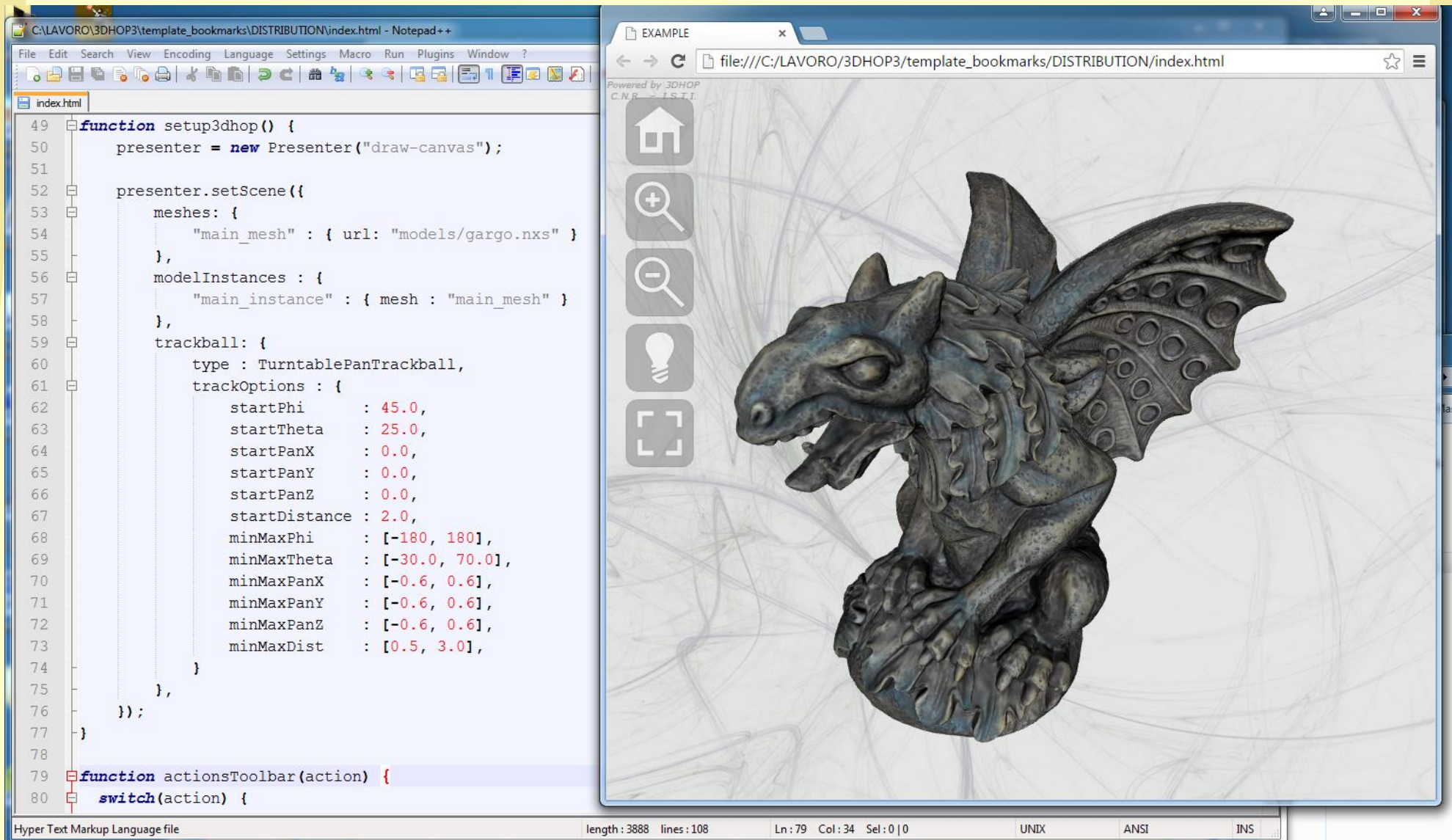
By example

An alternative method to get the new state you need, is to manually go there and ask the trackball its current state.

- ◆ Prepare your 3DHOP page, with the trackball properly set-up
- ◆ Open your page with a browser (Chrome works well for this)
- ◆ Rotate/pan/zoom until you have reached the desired spot you want to use for the trackball animation
- ◆ Open the Javascript Console (on Chrome, CTRS-Shift-I, last tab on the developer window)
- ◆ Write in the console *presenter.getTrackballPosition()*;
- ◆ The array returned is exactly what you need ! Copy it and save it somewhere.
- ◆ Rinse&Repeat for all the position you need
- ◆ The next time you will edit the HTML page, you will have the state arrays you need to specify the animation

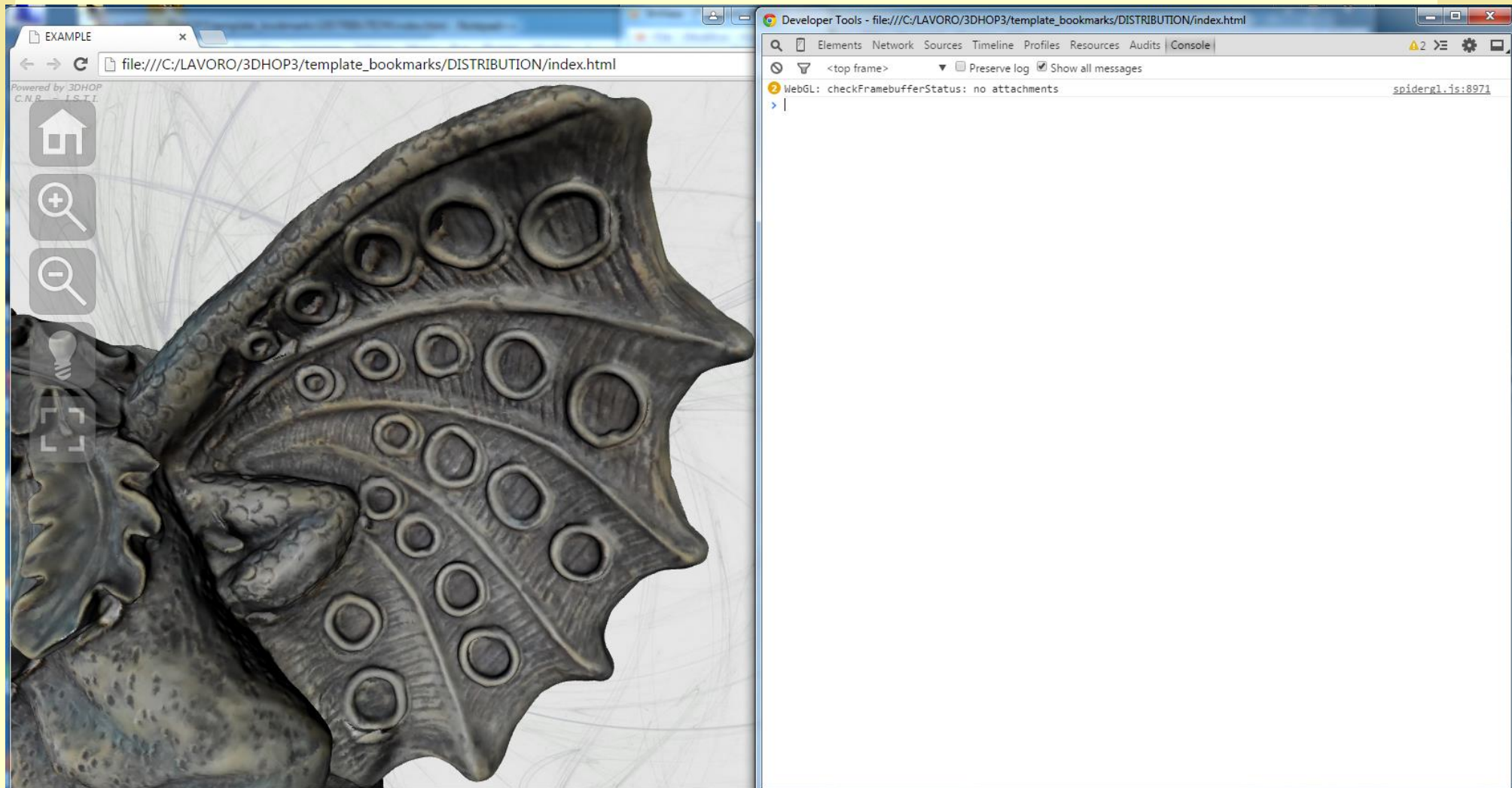
By example

My page, with full scene setup, and trackball properly set (TurntablePan) with starting position (state [45.0, 25.0, 0.0, 0.0, 0.0, 2.0]) loaded in Chrome



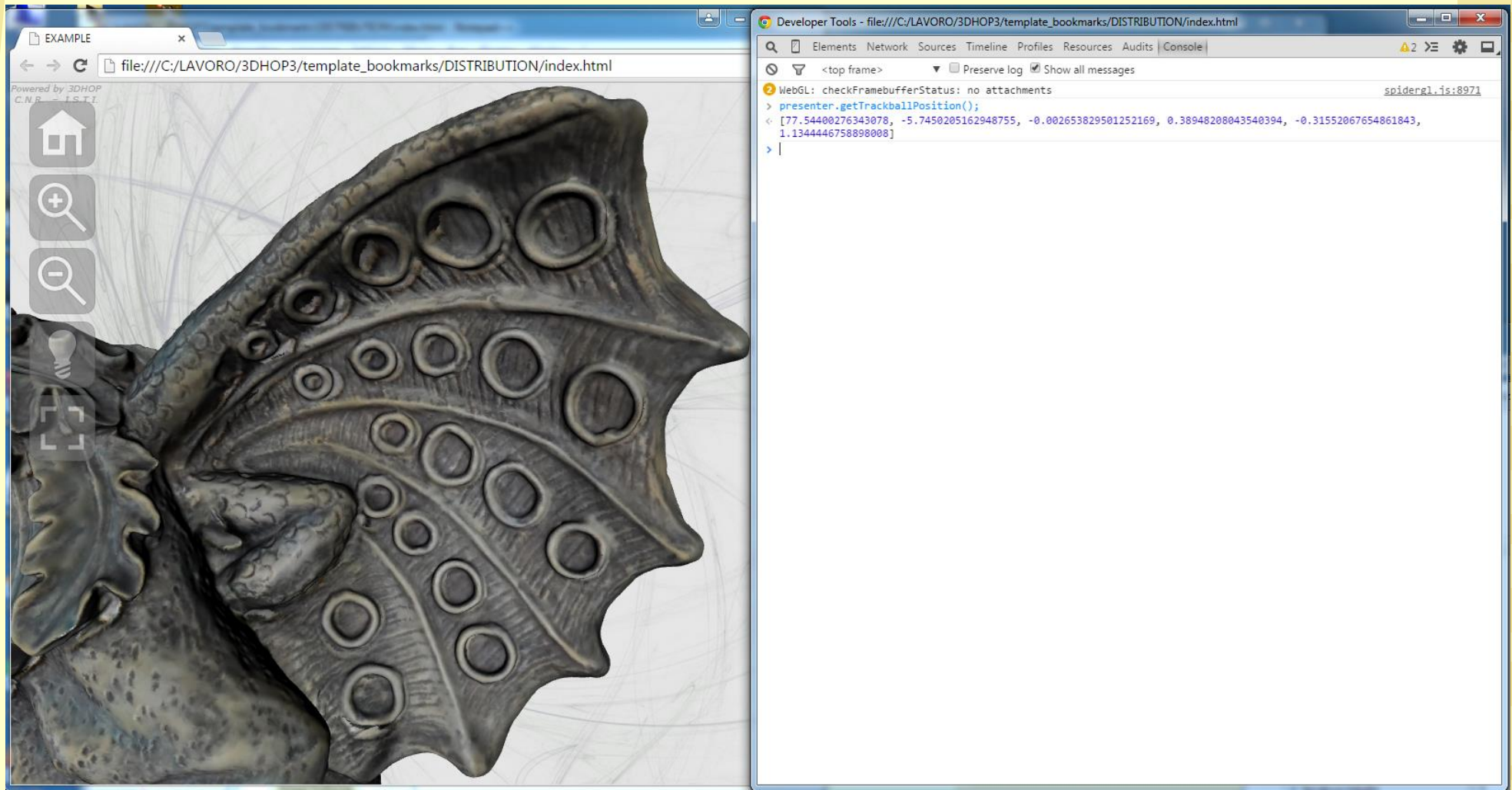
By example

I have rotated, panned and zoomed to reach the position I need. CTRL-Shift-I to open developer tools, select Console from the upper tab... Now you are in the Javascript Console and may interact directly with 3DHOP and the other page scripts



By example

In the console, write `presenter.getTrackballPosition();` then press Enter. The system will print the current state of the trackball, which is `[77.54400276343078, -5.7450205162948755, -0.002653829501252169, 0.38948208043540394, -0.31552067654861843, 1.1344446758898008]`

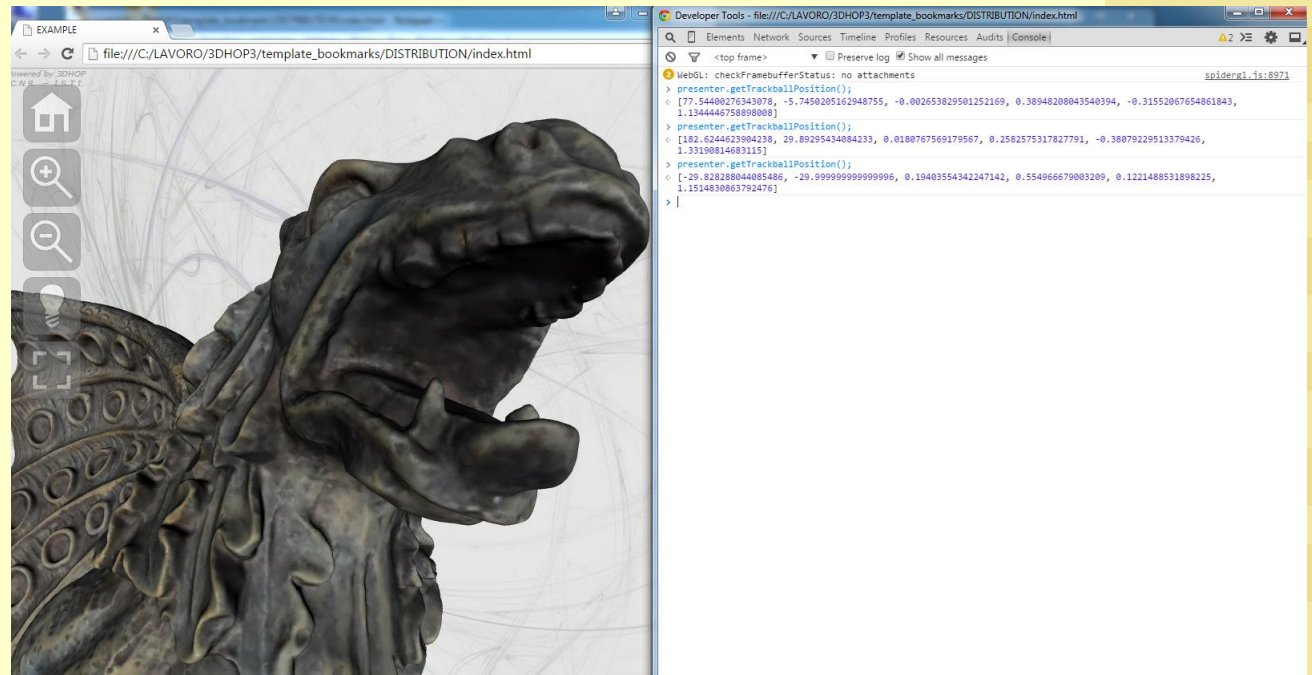


By example

You may safely reduce the number of figures after the decimal point... from this:
[77.54400276343078, -5.7450205162948755, -0.002653829501252169,
0.38948208043540394, -0.31552067654861843, 1.1344446758898008]

To this: [77.54, -5.74, 0.0, 0.38, -0.31, 1.13]

Now you may move the object to the next position you need, and call again `presenter.getTrackballPosition();` to get the next state



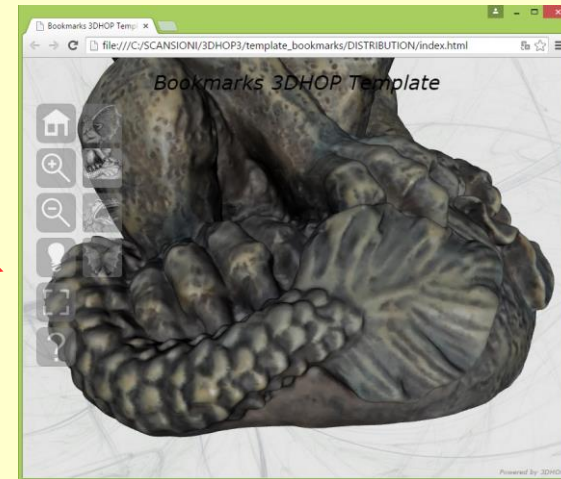
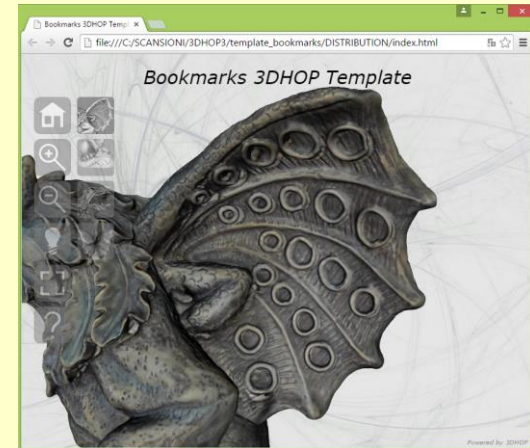
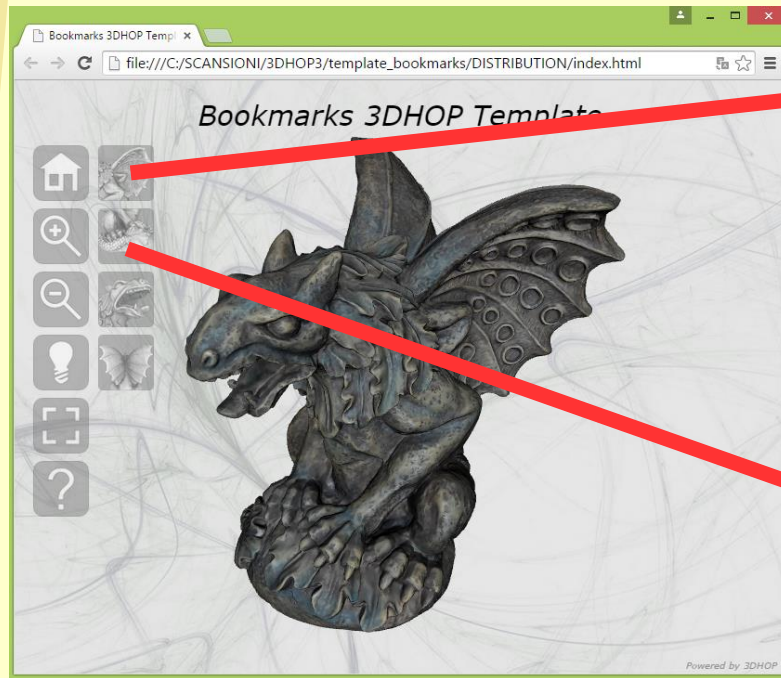
By example

I added to the HTML another toolbar with buttons that animate the trackball to the specific position I chose...

```
70 <!--BOOKMARKS TOOLBAR-->
71 <div id="bookmarksToolbar">
72   <br/>
73   <br/>
74   <br/>
75   <br/>
76 </div>
```

```
119 function actionsToolbar(action) {
120   switch(action) {
121     <!--COMMANDS-->
122     case 'home' : presenter.animateToTrackballPosition([45.0, 25.0, 0.0, 0.0, 0.0, 3.0]); break;
123     case 'zoomin' : presenter.zoomIn(); break;
124     case 'zoomout' : presenter.zoomOut(); break;
125     case 'light' : presenter.enableLightTrackball(true); lightSwitch(); break;
126     case 'light_on' : presenter.enableLightTrackball(false); lightSwitch(); break;
127     case 'full' : fullscreenSwitch(); break;
128     case 'full_on' : fullscreenSwitch(); break;
129     case 'help' : helpSwitch(); break;
130
131     <!--BOOKMARKS-->
132     case 'pos1' : presenter.animateToTrackballPosition([81.89, -3.07, 0.04, 0.35, -0.28, 1.70]); break;
133     case 'pos2' : presenter.animateToTrackballPosition([-52.64, 6.67, -0.02, -0.59, 0.14, 1.22]); break;
134     case 'pos3' : presenter.animateToTrackballPosition([-46.67, -35.27, 0.27, 0.58, 0.32, 1.26]); break;
135     case 'pos4' : presenter.animateToTrackballPosition([178.72, 23.62, 0.06, 0.15, 0.07, 1.94]); break;
136
137   }
```

By example



Hotspot Positioning

Tips and Tricks

Anim-able trackballs

In 3DHOP a hotspot is defined by using a mesh, like a normal instance. The mesh is defined, as usual, starting from a file.

To have the hotspot geometry in the correct position, there are two options:

- ◆ I move/rotate/scale a generic geometry using the transformations inside the declaration of the 3DHOP scene
- ◆ The mesh I use is already in the correct position with respect to the instances in the scene

Place a predefined shape

The shape of a spot may be specified by using a simple geometry and move/scale it

In the 3DHOP distribution, there are two simple shapes, a sphere and a cube, centered in the origin, of 1 unit size (sphere.ply, box.ply).

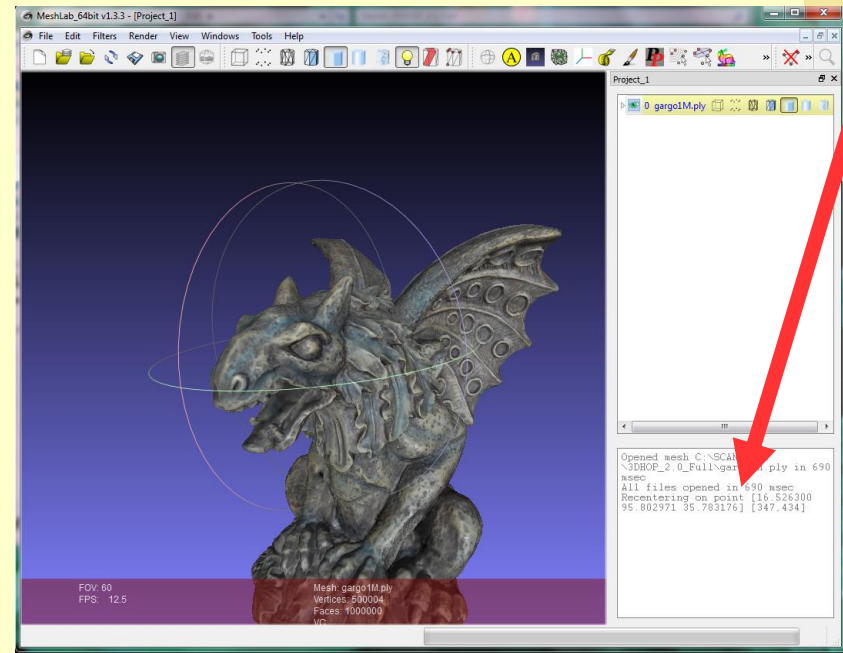
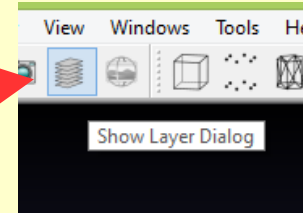
You may use them, and place them in the correct position with a translation, and scale according to the size of the area of interest. But how can I know where to place them?

- ◆ If I already know the XYZ position of my spot, I'm ok, that value is the translation
- ◆ Or, I can pick the spot using MeshLab

Place a predefined shape

- ◆ Load the 3D model inside MeshLab
- ◆ Open the layer dialog
- ◆ Move the object until you see the area of the model you want to use for the spot
- ◆ DOUBLE CLICK on the area
- ◆ Meshlab will re-center the model where you have clicked, and the XYZ coordinates will appear in the log area of the layer dialog

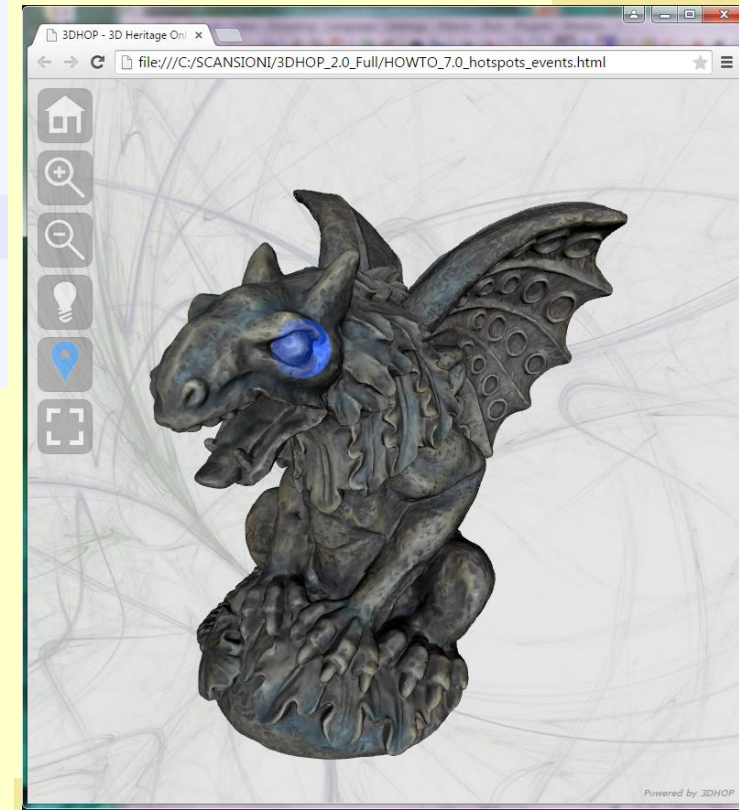
I double-clicked the left eye, and
I get:
[16.526300 95.802971 35.783176]



Place a predefined shape

I create a spot using the sphere.ply geometry, and apply a translation to the point I picked, plus a scaling to make it the correct size

```
47 presenter.setScene({
48   meshes: {
49     "Gargoyle" : { url: "models/multires/gargo.nxs" },
50     "Sphere" : { url: "models/singleres/sphere.ply" },
51   },
52   modelInstances : {
53     "Gargo" : {
54       mesh : "Gargoyle"
55     },
56   },
57   spots : {
58     "Sphere" : {
59       mesh : "Sphere",
60       transform : {
61         matrix: SglMat4.mul(SglMat4.translation([16.52, 95.80, 35.78]),
62                               SglMat4.scaling([ 5.0, 5.0, 5.0]))
63       },
64       color : [0.0, 0.25, 1.0, 0.2]
65     },
66   },
67 },
```



Modelled geometries

3DHOP trackballs can be animated using javascript function

- ◆ Load your model inside Blender (or another modeller), but DO NOT MOVE/SCALE/ROTATE IT
- ◆ Create the geometry of the hotspots directly on the imported 3D model
- ◆ Export the newly created geometries

You now have perfectly fitting hotspot geometries



Modelled geometries

If you need a very simple shape, you can do it inside MeshLab

- ◆ Create a new layer with *filters->Create New Mesh Layers->* .
Something simple like Sphere or Box
- ◆ Move the new layer geometry in the correct position, rotate and scale it using manipulators <https://www.youtube.com/watch?v=FGaNv23Xvtw>
- ◆ Freeze current matrix, export the layer geometry as a PLY

