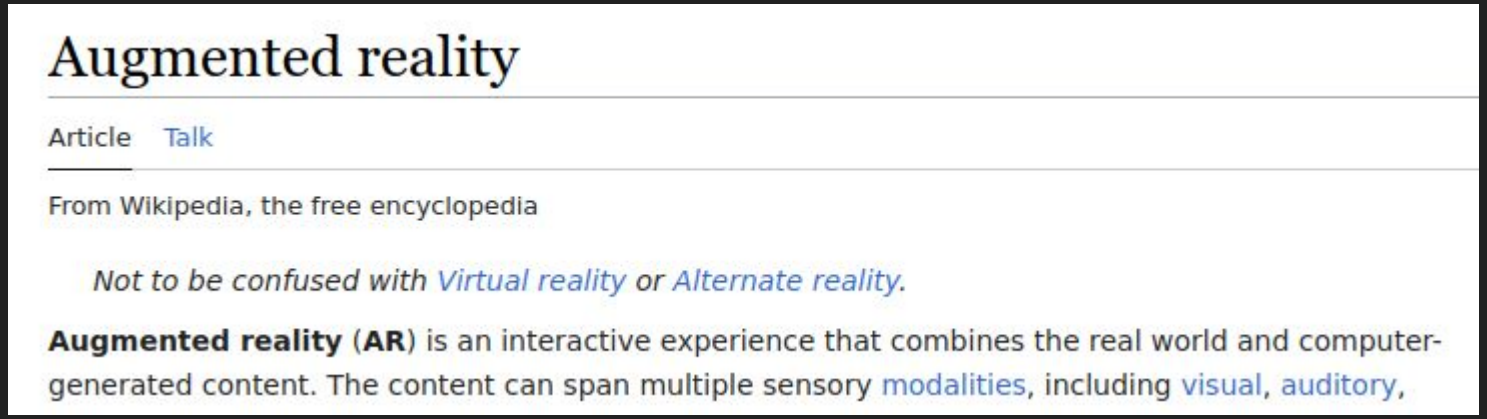
CODE COFFEE

Augmented Reality for Your Posters

What is Augmented Reality?



Augmented reality

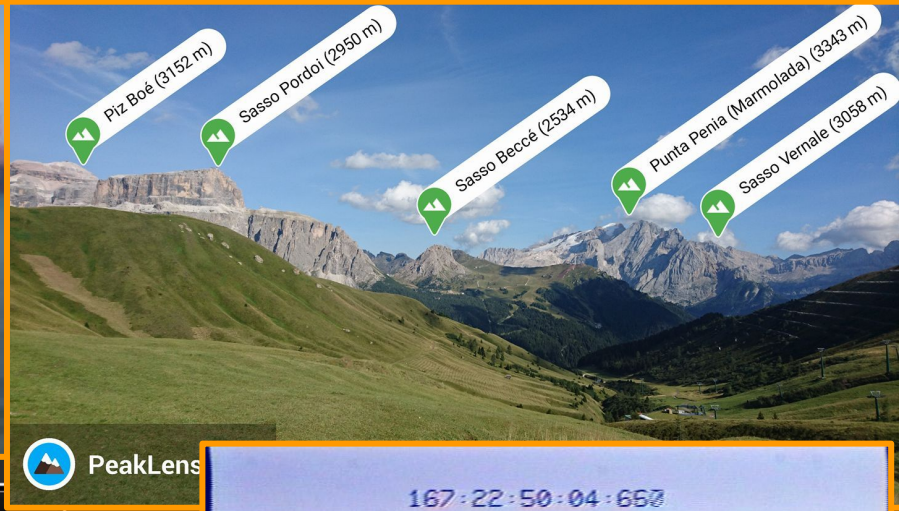
Article Talk

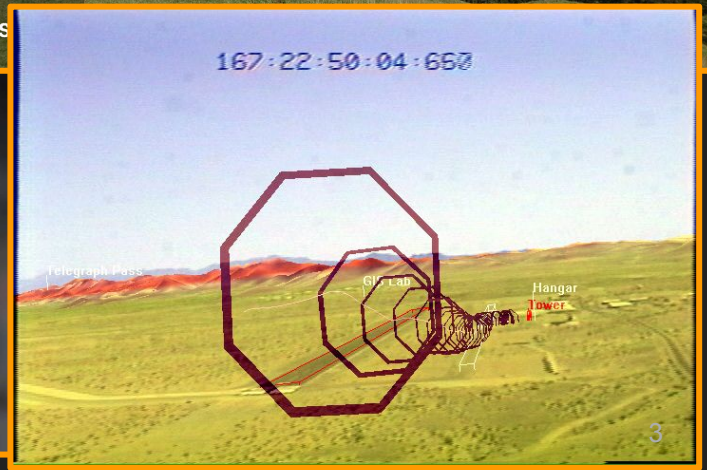
From Wikipedia, the free encyclopedia

Not to be confused with Virtual reality or Alternate reality.

Augmented reality (**AR**) is an interactive experience that combines the real world and computer-generated content. The content can span multiple sensory modalities, including visual, auditory,

What is Augmented Reality?





What is AR.js?

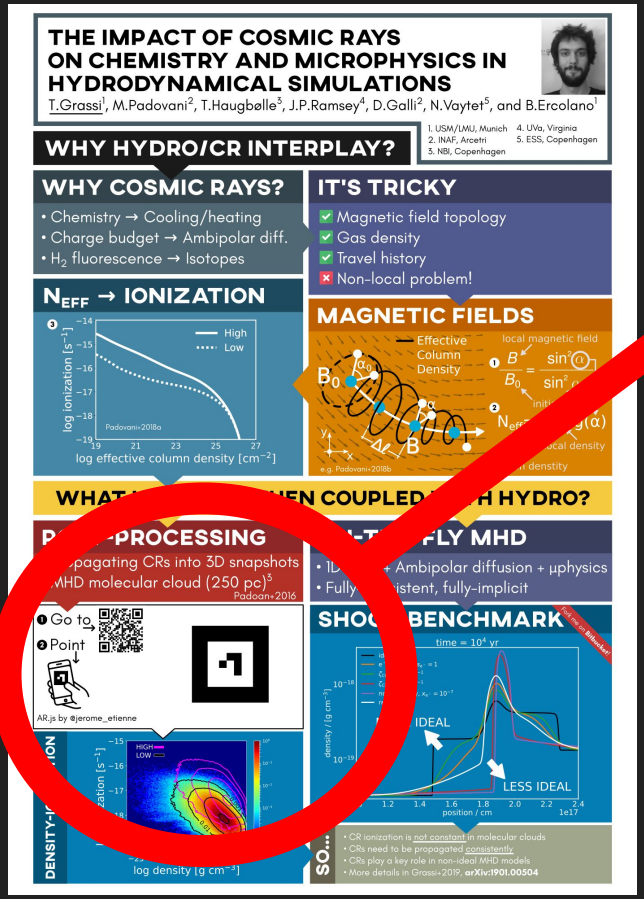
ar-js-org.github.io/AR.js-Docs/

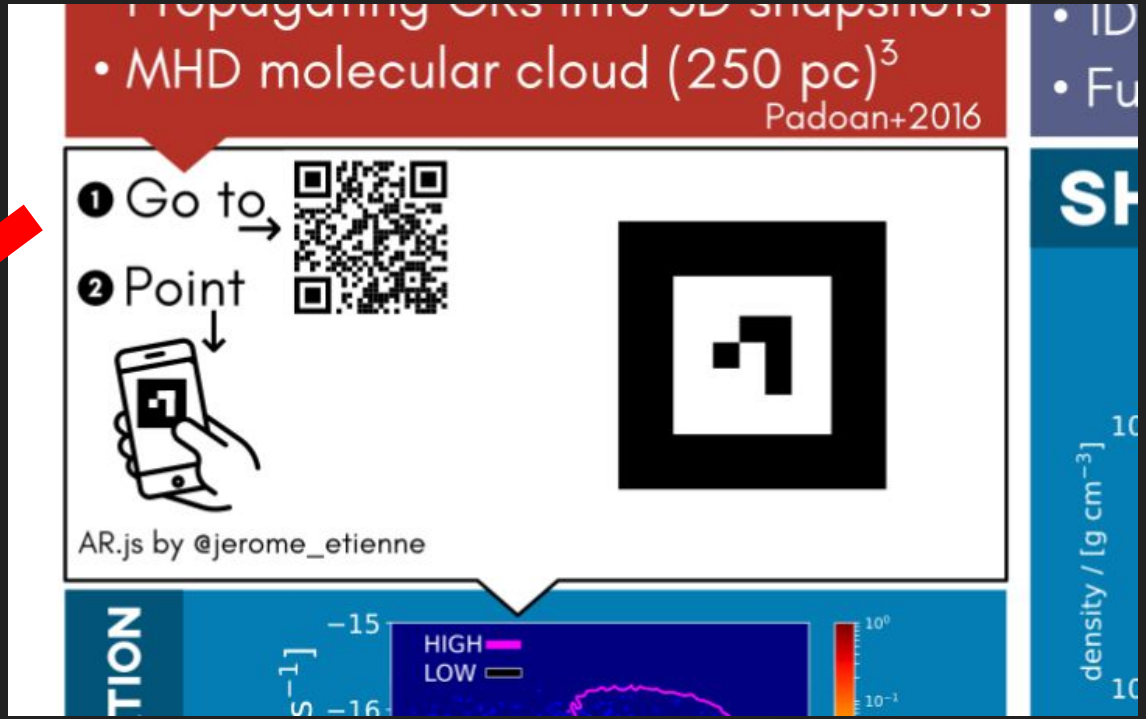
AR.js is a lightweight library for Augmented Reality on the Web, which includes features like:

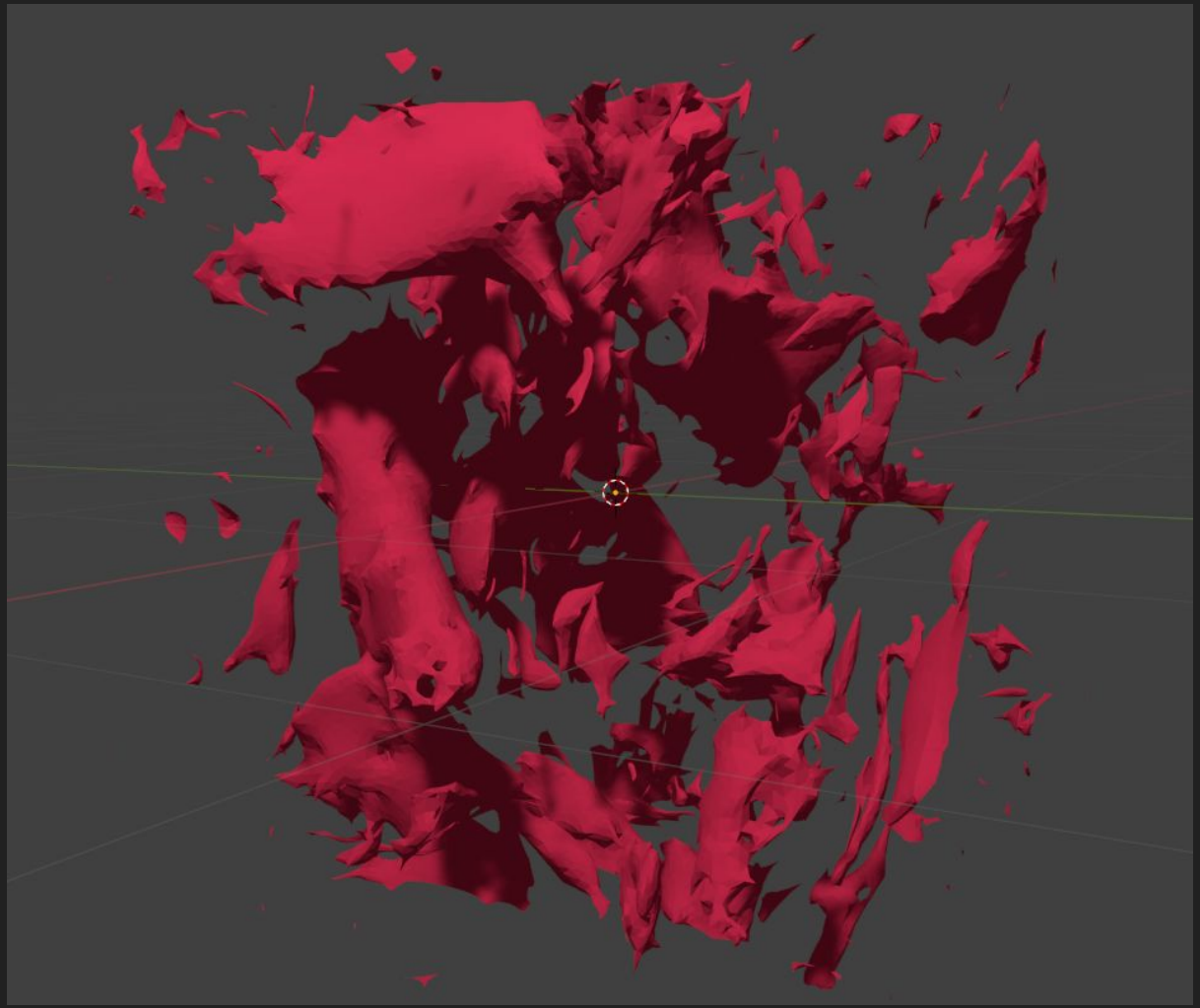
- Image Tracking
- Location based AR
- Marker tracking

- Very Fast: runs efficiently even on phones
- Web-based: pure web solution, so no installation required. Fully JS based, using three.js + A-Frame + jsartoolkit5
- Standards: works on any phone with webgl and webrtc
- Open Source

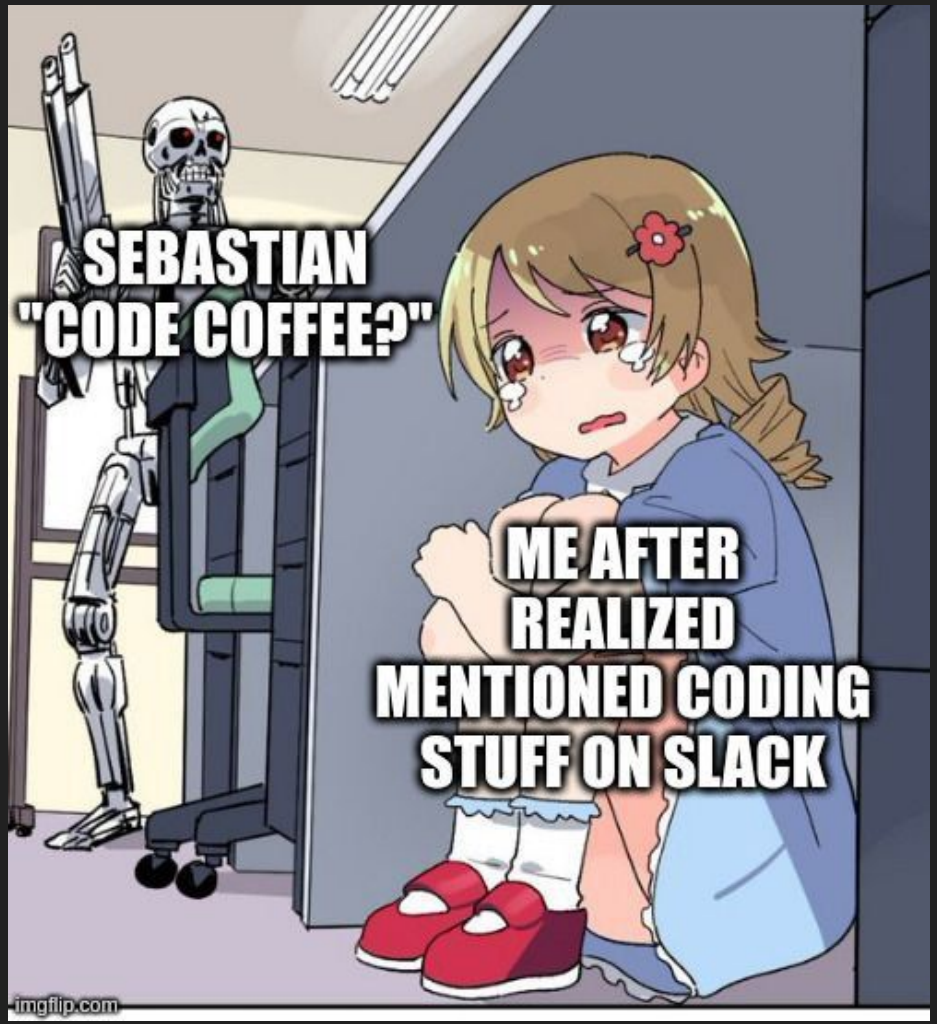
Motivation: 3D visualization in posters







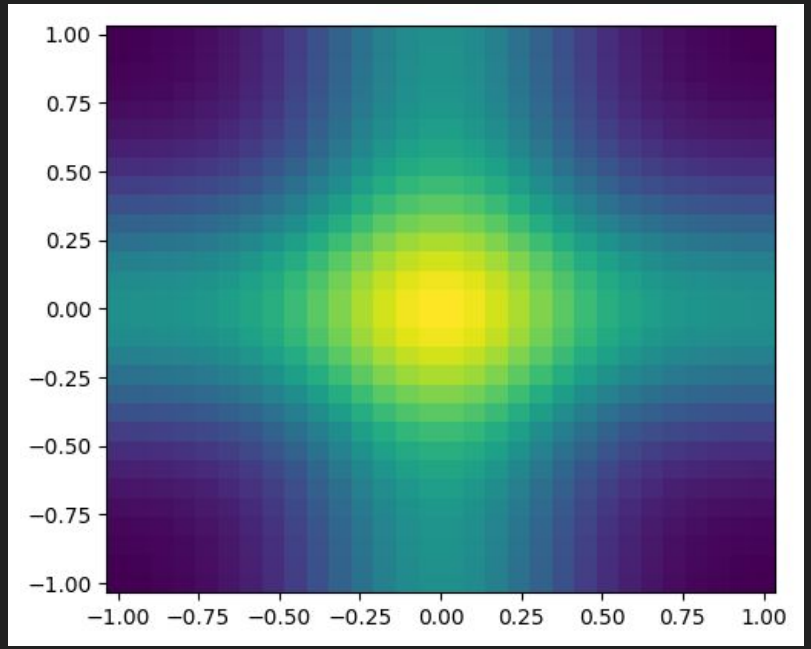
Real Motivation



Step 1: produce some 3D data

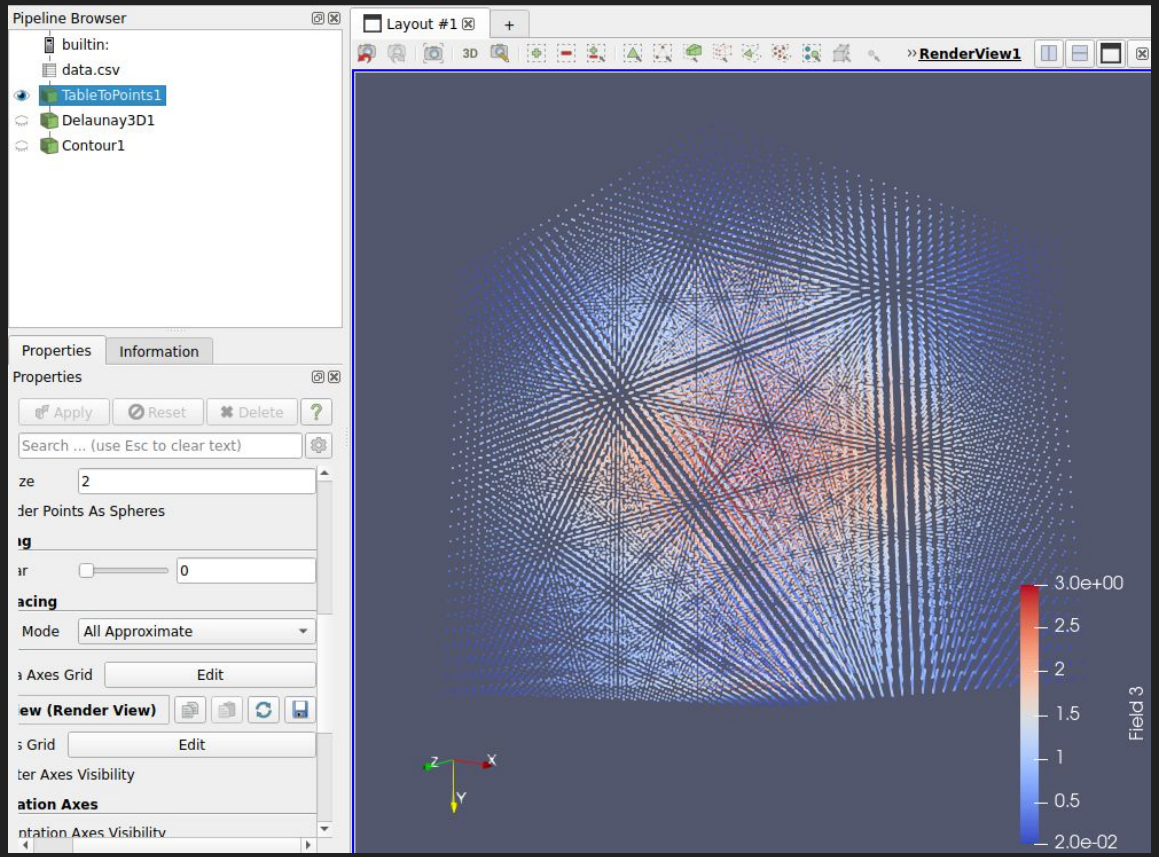
main.py

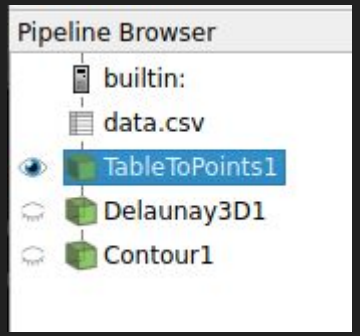
```python
import matplotlib.pyplot as plt
import numpy as np

# number of grid points per side
n = 30

# create grids
x = np.linspace(-1, 1, n)
y = np.linspace(-1, 1, n)
z = np.linspace(-1, 1, n)

# create meshgrid
xx, yy, zz = np.meshgrid(x, y, z, indexing="ij")

# create some 3D function
sx = sy = sz = 0.1
ff = np.exp(-xx**2 / 2 / sx) + np.exp(-yy**2 / 2 / sy) + np.exp(-zz**2 / 2 / sz)

# plot slice to test
plt.pcolormesh(xx[..., n//2], yy[..., n//2], ff[..., n//2])
plt.show()

# prepare for csv file
data = np.array([np.ravel(x) for x in [xx, yy, zz, ff]]).T

# save to csv file
np.savetxt("data.csv", data)
```



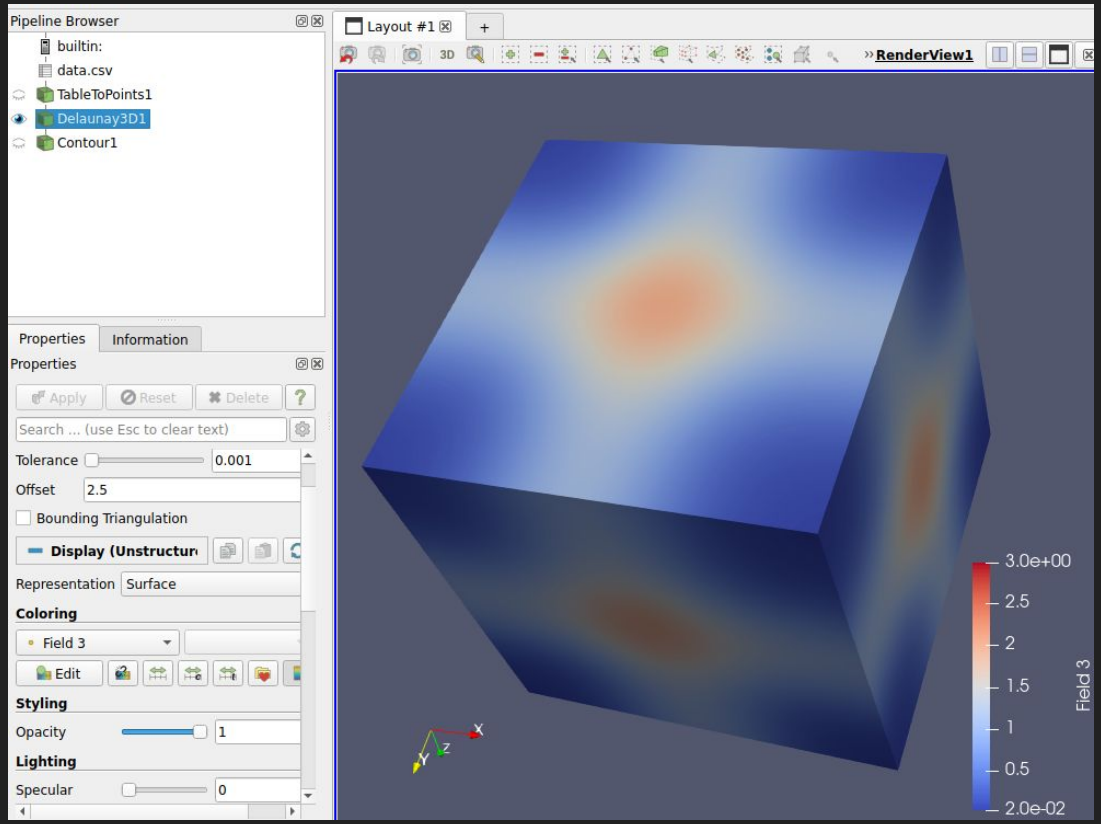
Step 2: produce a 3D object file

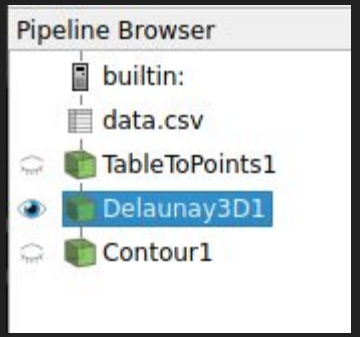
Step 2: produce a 3D object file



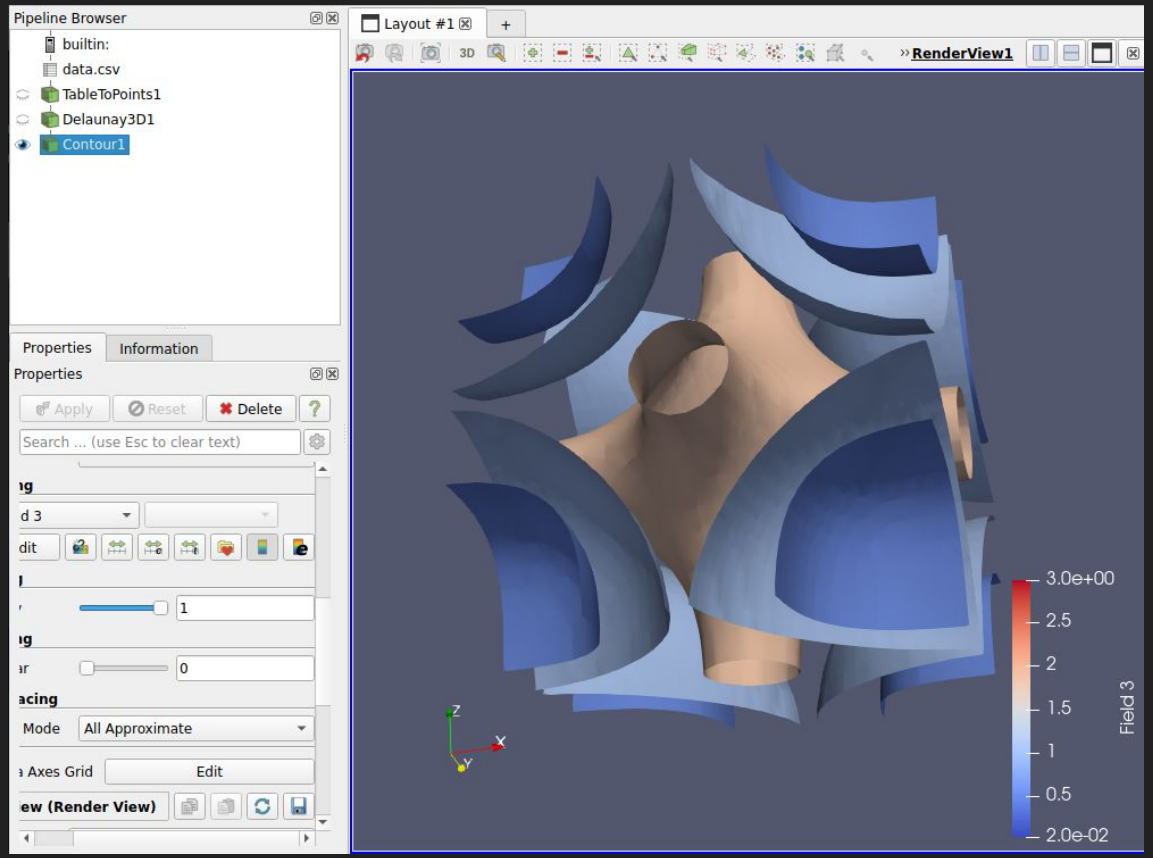


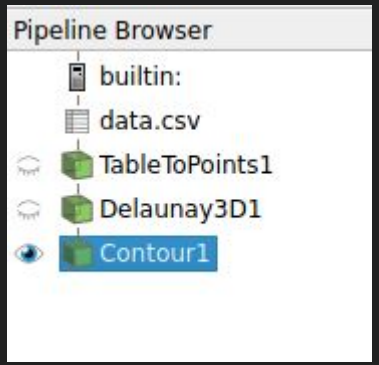
Step 2: produce a 3D object file





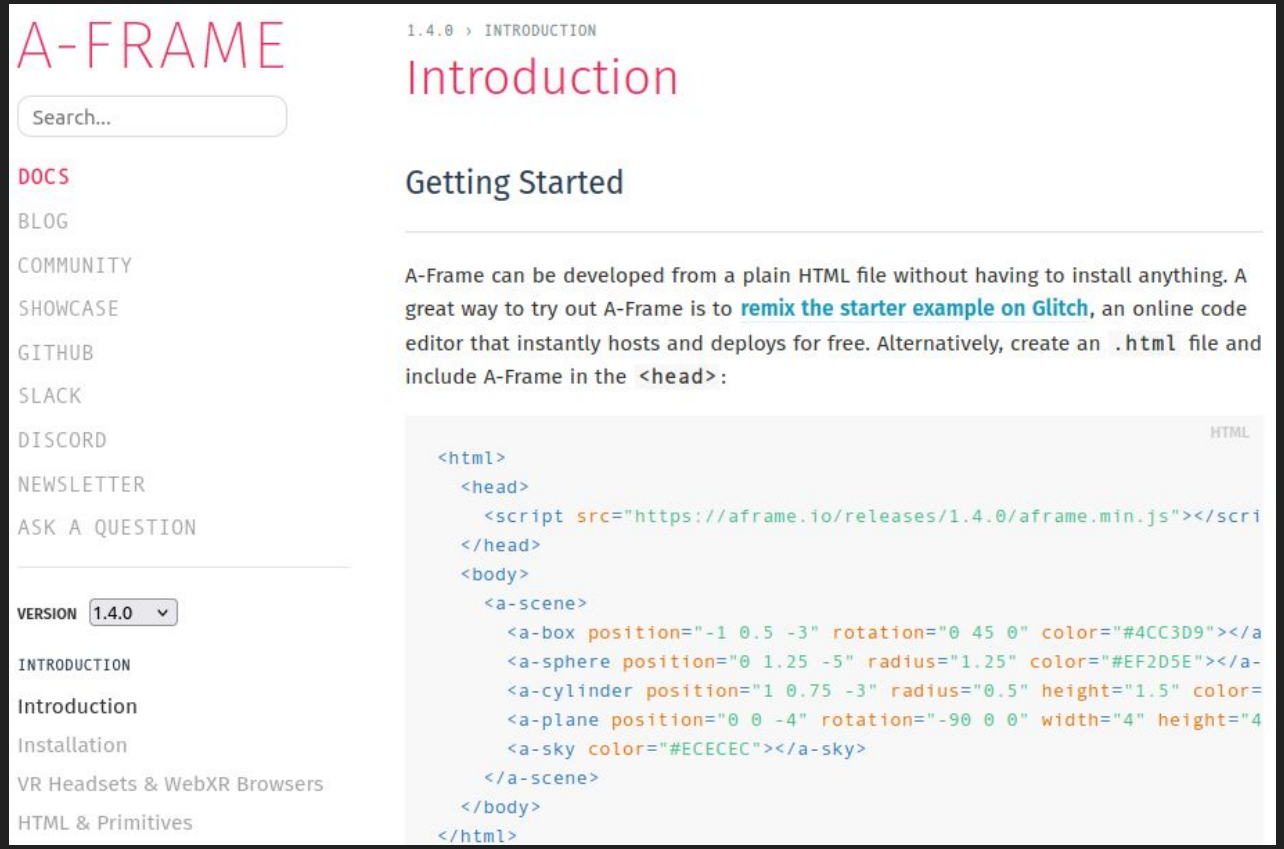
Step 2: produce a 3D object file





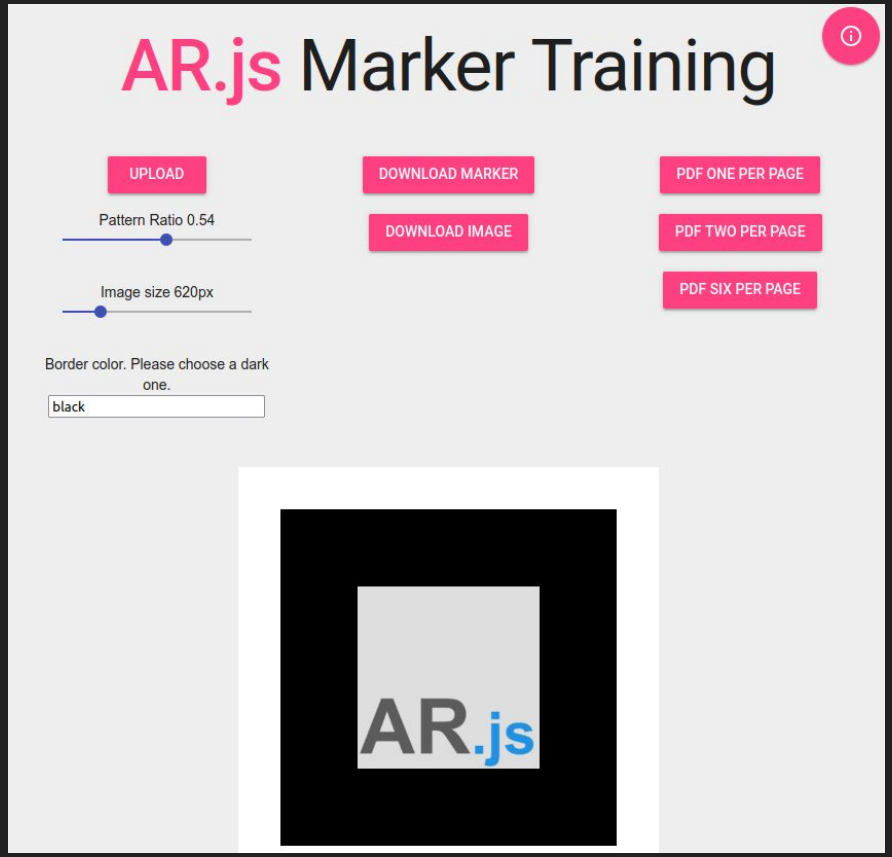
Export Scene -> *.gltf

Before the next step: what is A-FRAME?



Before next step: what is a marker?

Before the next step: what is a marker?



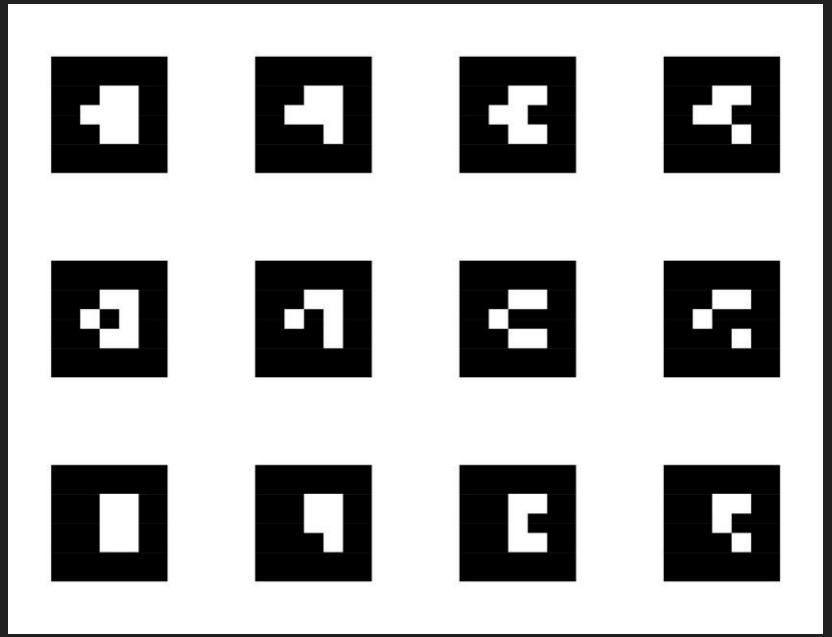
.patt file

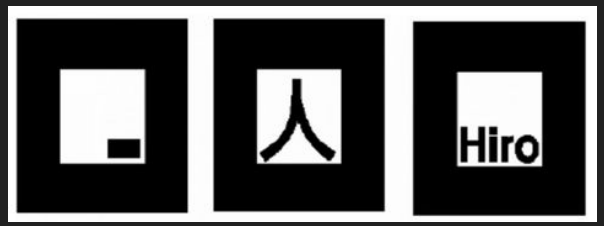
```
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255   0 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255   0 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255   0 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255   0 249 255 255 255 255 255 255 255
255 255 255 255 255 255 251   0   0 255 255 255 255 255 255 255
255 255 255 255 255 255  20   0   0 255 255 255 255 255 255 255
255 255 255 255 255 255   0   0   0 255 255 255 255 255 255 255
255 255 255 255 255 255   0 194   0   0 255 255 255 255 255 255
255 255 255 255 255   0   0 255 255   0 253 255 255 255 255 255
255 255 255 255 212   0 254 255 255   0   0 255 255 255 255 255
255 255 255 108   0   5 255 255 255 255   0   0 255 255 255 255
255 255   0   0   0 255 255 255 255 255 255   0   0  23 255 255
254   0   0  79 255 255 255 255 255 255 255 255   0   0   0 255
255  13 255 255 255 255 255 255 255 255 255 255 255 209 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
```

Before the next step: what is a marker?

Some tips:

- Avoid overly complicated patterns
- Use rotationally asymmetrical markers
- From portrait to landscape and back
- Marker size
- Proportional scale
- Print the marker
- Proper border size around the marker
- Chrome on iOS: we have a problem
- Change camera resolution





Step 3: Prepare JS code (actually just HTML)

index.html

```
<!DOCTYPE html>
<html>
    <script src="js/aframe.min.js"></script>
    <!-- we import arjs version without NFT but with marker +
    <script src="js/aframe-ar.js"></script>
    <body style="margin : 0px; overflow: hidden;">
        <a-scene embedded arjs>
        <a-marker type='pattern' url='kanji.patt'>
            <!-- we use cors proxy to avoid cross-origin prol
            <a-entity
            position="0 0 0"
            scale="1 1 1"
            gltf-model="models/testx.gltf"
            ></a-entity>
        </a-marker>
        <a-entity camera></a-entity>
        </a-scene>
    </body>
</html>
```

import necessary JS libraries

embed in a a-scene

define marker

assign gltf model to marker

add a camera

Step 4: Load on some server

important: https

Step 5: Test it!

marker on slide 14

Final remarks

- Use relatively light 3D models (it has to be downloaded)
- Animations can be used, but with Paraview is complicated
- A-frame animations are available (e.g. constant rotation)
- VR goggles
- Note that sometimes Paraview produces “wrong” gltf models (solved with x3d and Blender)
- Scaling is important
- Light conditions are important (reflections on black prints)
- Fancy but not impressive
- A-Frame allows interactions and triggered events
- It is possible to use textures and advanced lighting
- Also location-based and image-based tracking is available