

How to use Blender for Augmented Reality UI mockups

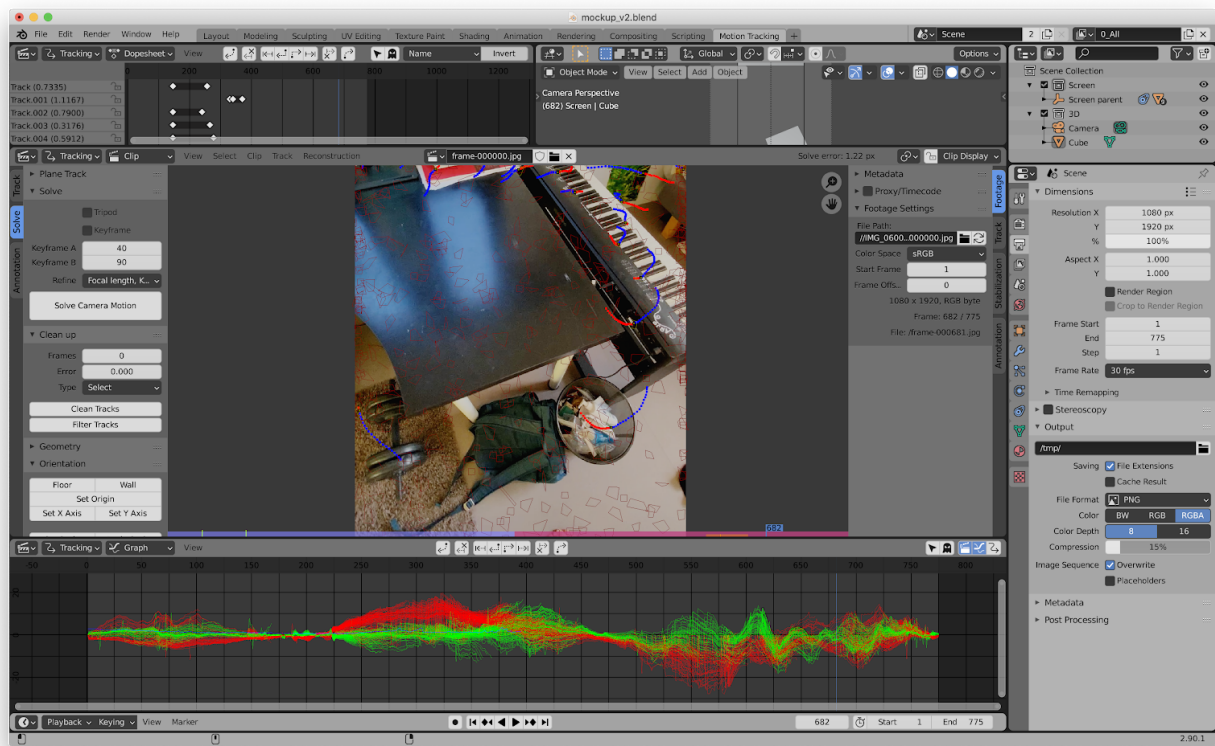
Arjo Nagelhout - 2020 Oct 23

[Blender Motion Tracking - Room Transformation!](#)

First convert the footage to mp4 30fps and trim it using Compressor

Then go to Blender:

1. File > New > VFX
2. Set scene frame rate to match footage
3. Import footage (mp4)
4. Set scene frames
5. Prefetch
6. Motion model: Choose motion model, depending on camera movement
 - a. Perspective
 - b. Affine
 - c. Location, Rotation and Scale
 - d. Location
7. Detect features
 - a. Margin 16
 - b. Threshold 0.010
 - c. Distance 80
8. Track
 - a. Click track forward
 - b. Stop when many features are missing
 - c. Click detect features
 - d. Select all markers
 - e. Go back to a. until you have done the entire scene
9. Filter tracks: track threshold 15.00



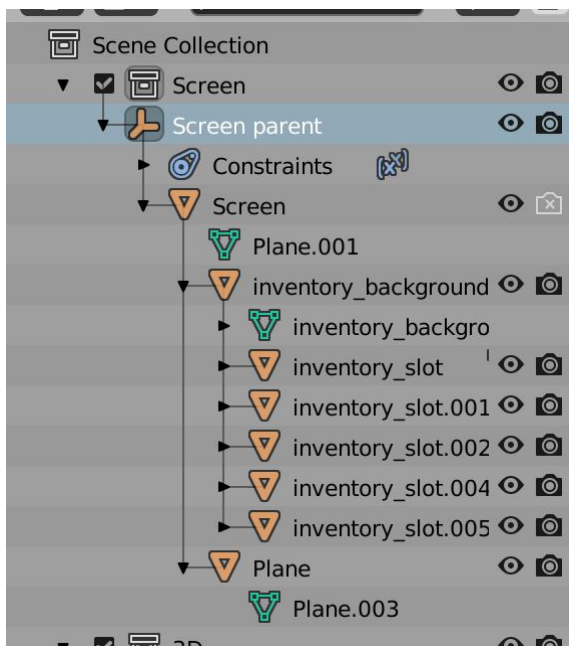
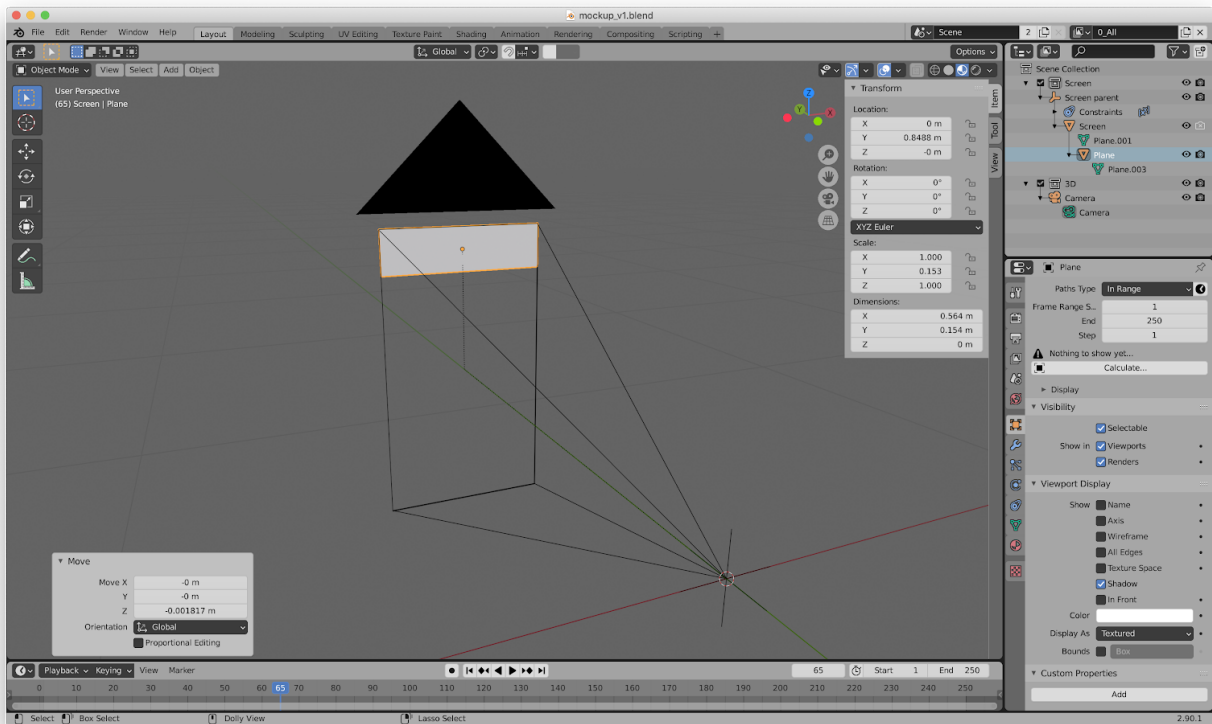
<https://blender.stackexchange.com/questions/39876/movie-clip-on-plane>

Parenting things in Blender:

<https://blender.stackexchange.com/questions/3763/parenting-messes-up-transforms-where-is-the-offset-stored>

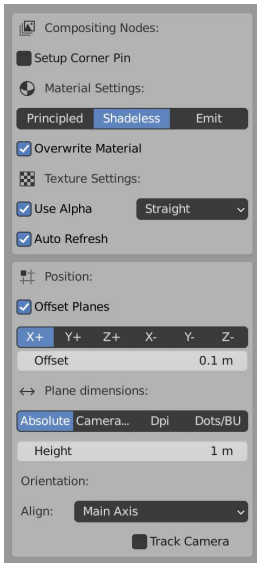
Apparently there is an offset stored that you can't access via the UI, reset it using: Parent > Clear Parent Inverse

I'm currently creating a powerful way to mock up augmented reality applications all from within Blender:



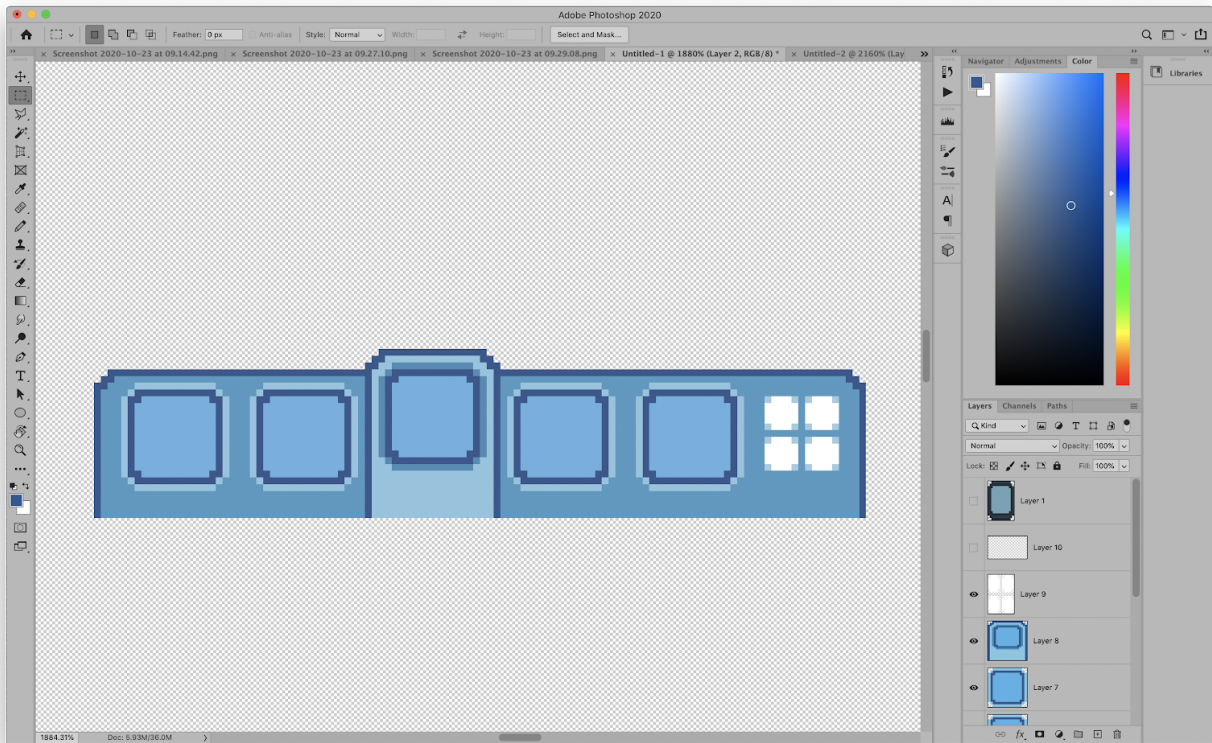
It has two view layers: one is the interface (screen) and the other is the 3D scene. These get rendered and then put on top of each other.

You can import interface design elements with the **Import images as Planes** plugin. (Enable in Preferences > Add-ons).

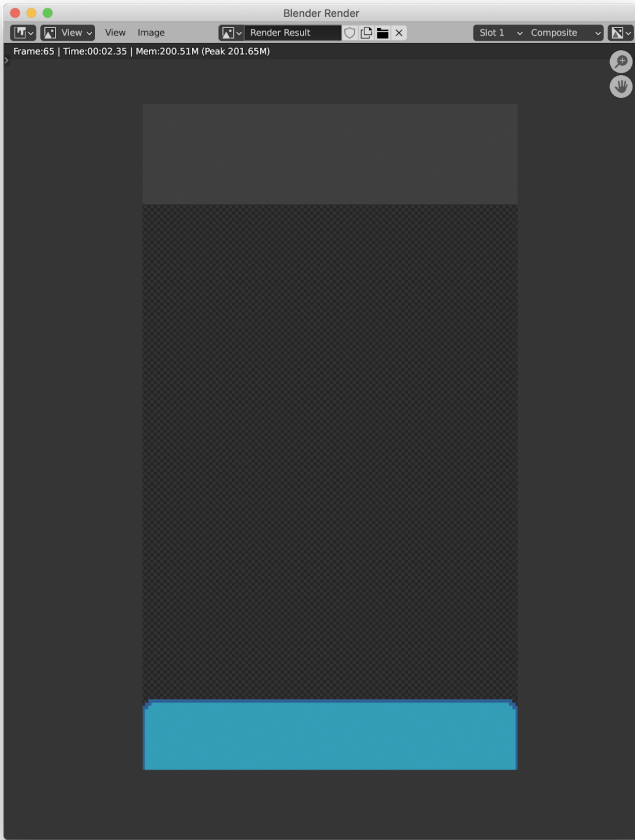


You should change the Blend Mode to “Alpha Clip” and the shadow mode to None.

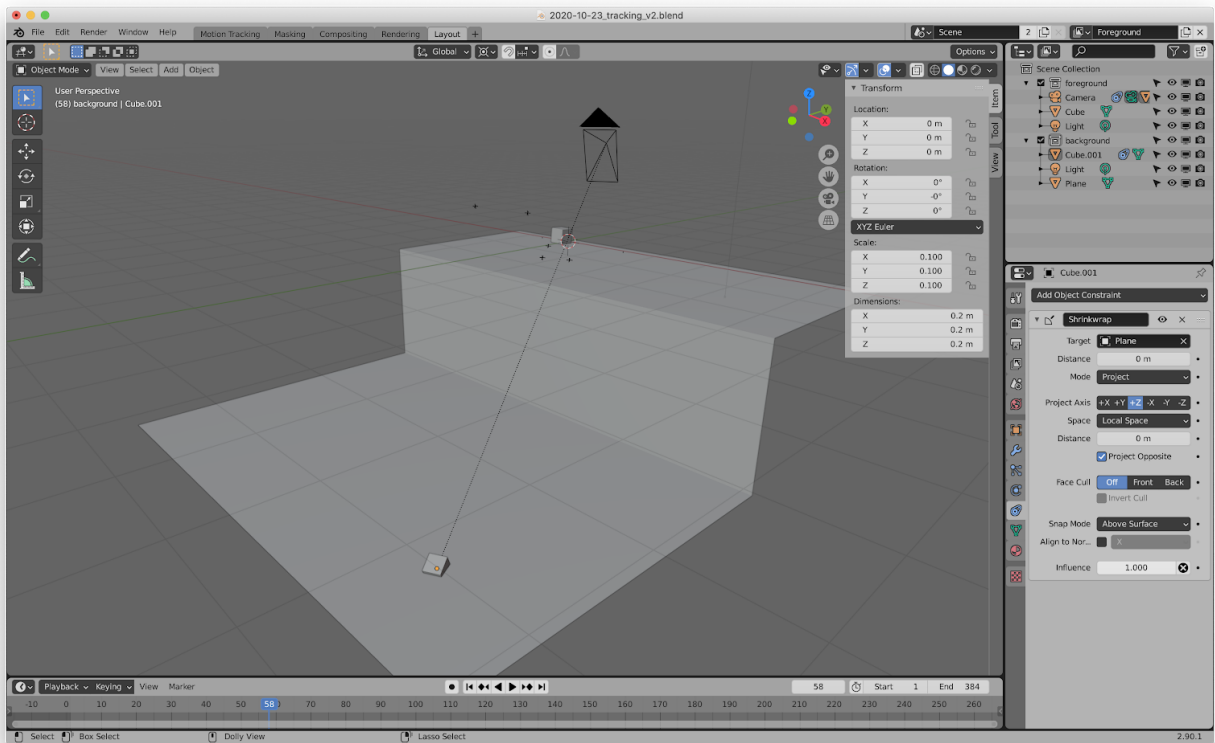
Interface elements can then be created with any design tool of choice, such as Photoshop:



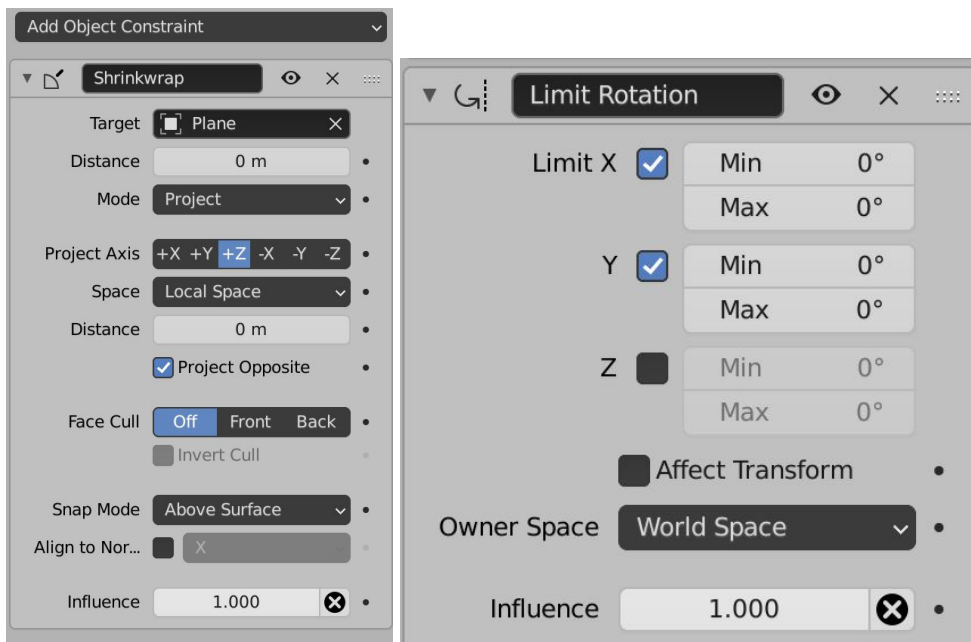
The output will be as follows:



You can now also create animations

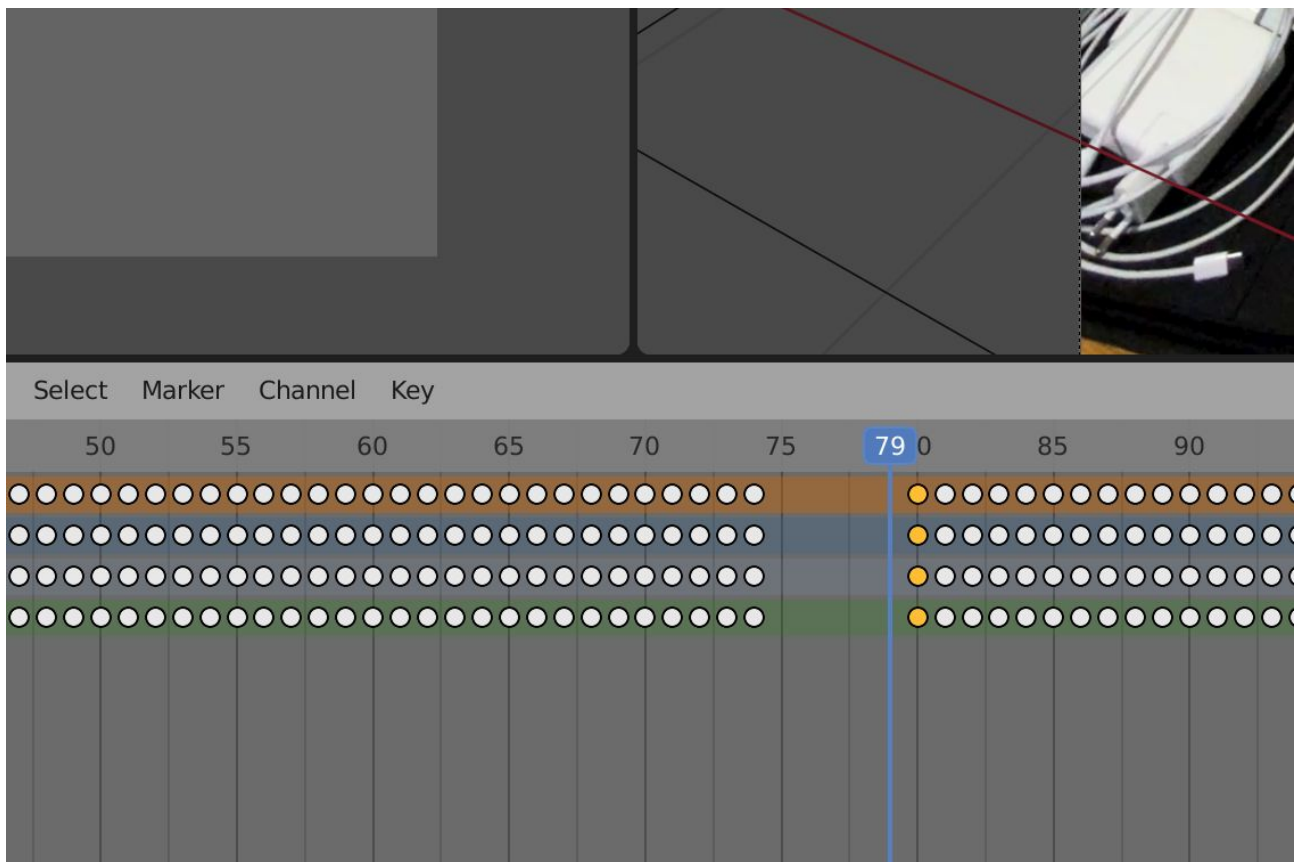


To have the camera point to a specific point in space, use the Shrinkwrap Object Constraint on the object to place in the space with the following settings. Also make the camera the parent of this object.



If you want the object to face the user using just Z rotation, use the limit rotation constraint

The next step is to go to Object > Animation > Bake Action. That way you can change the constraints into keyframes and make sure these work with a certain delay (not instantly following etc.) (Keep the object under the camera parent as otherwise you will get weird interpolation errors due to camera movement)



Find the two keyframes where there is a large jump in between. Remove a set amount of keyframes after the first keyframe. The amount of keyframes removed is dependent on the distance of the jump.

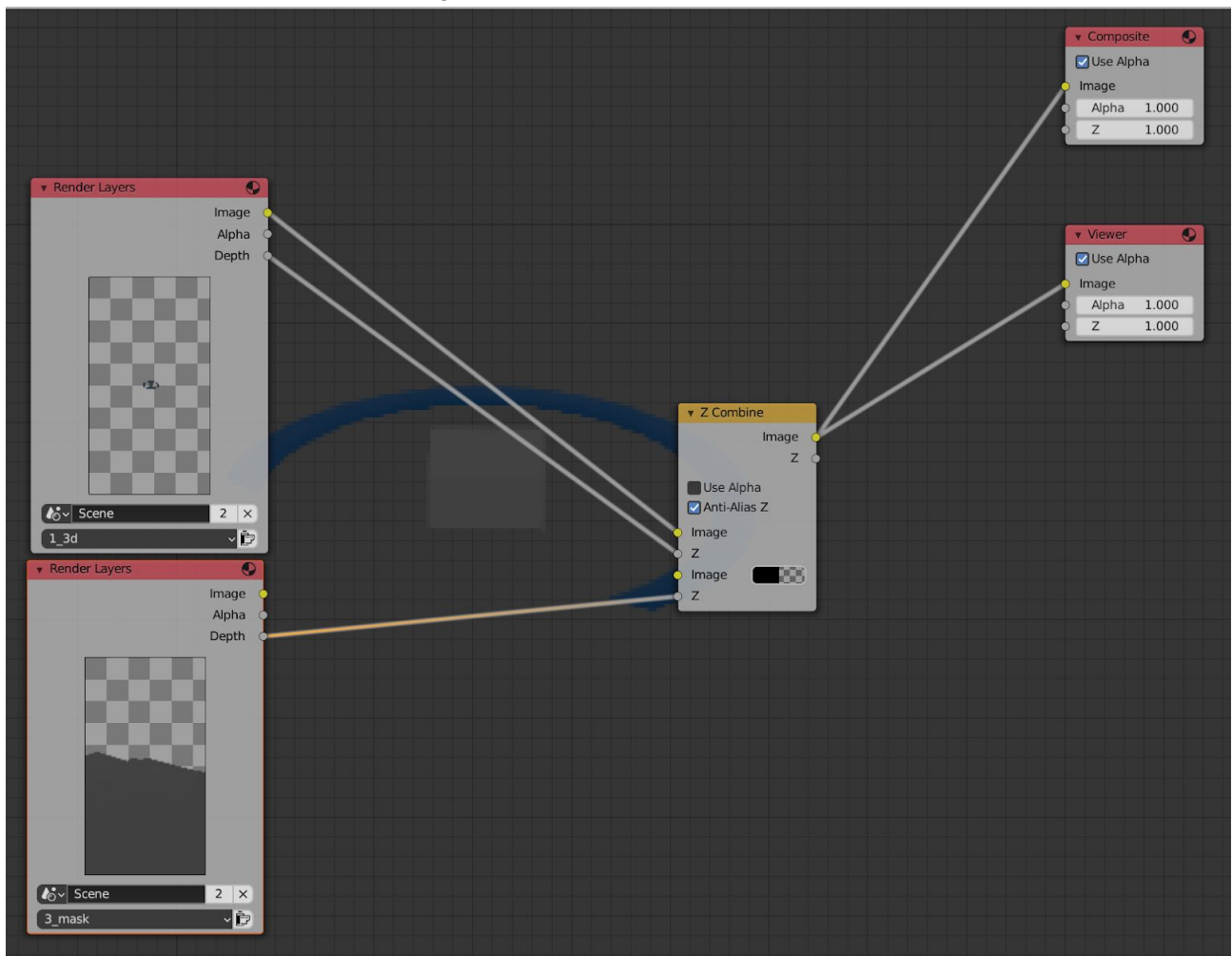
Now for masking the scene:

create two View Layers:

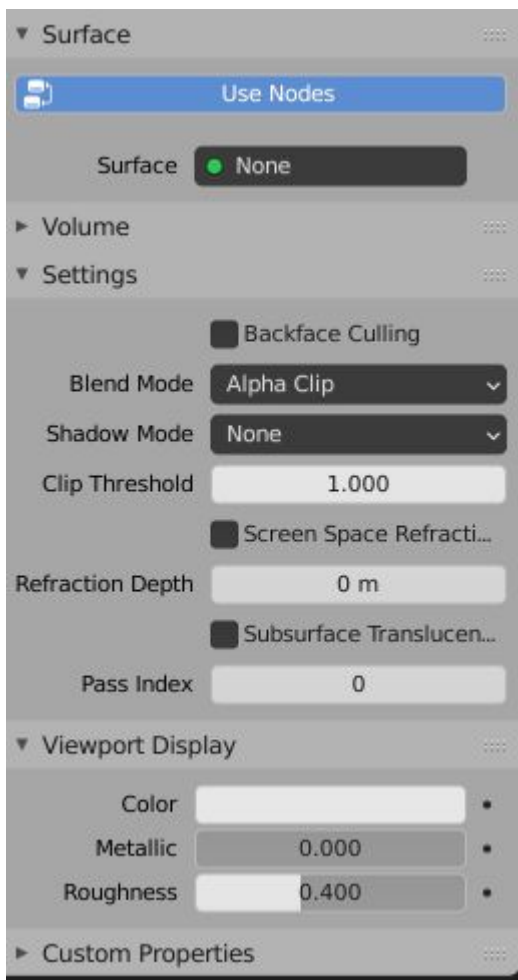
- 3D
- Masking

and create the following node setup:

- Z combine
- Don't connect the mask image, make that transparent.



For the mask objects, add the following material:



And set Properties Panel > Object Properties > Viewport Display > Wireframe to “on”.

New way:

Create one view layer on which both the mask and the 3D scene are visible, set the mask viewport display to **Display as Wire** and set the Mask to Shadow Catcher. (Cycles only).

Also set an HDR or create a sun.

Film > Transparent

Denosing data > composite.

Because performance is a really big issue with creating quick mockups of AR apps, using Eevee is preferred. However, a shadow catcher object for the physical environment should then be added. No -> this is not necessary for previewing app mockups and if you want to render it use Cycles.

Blender additions for better AR mockups:

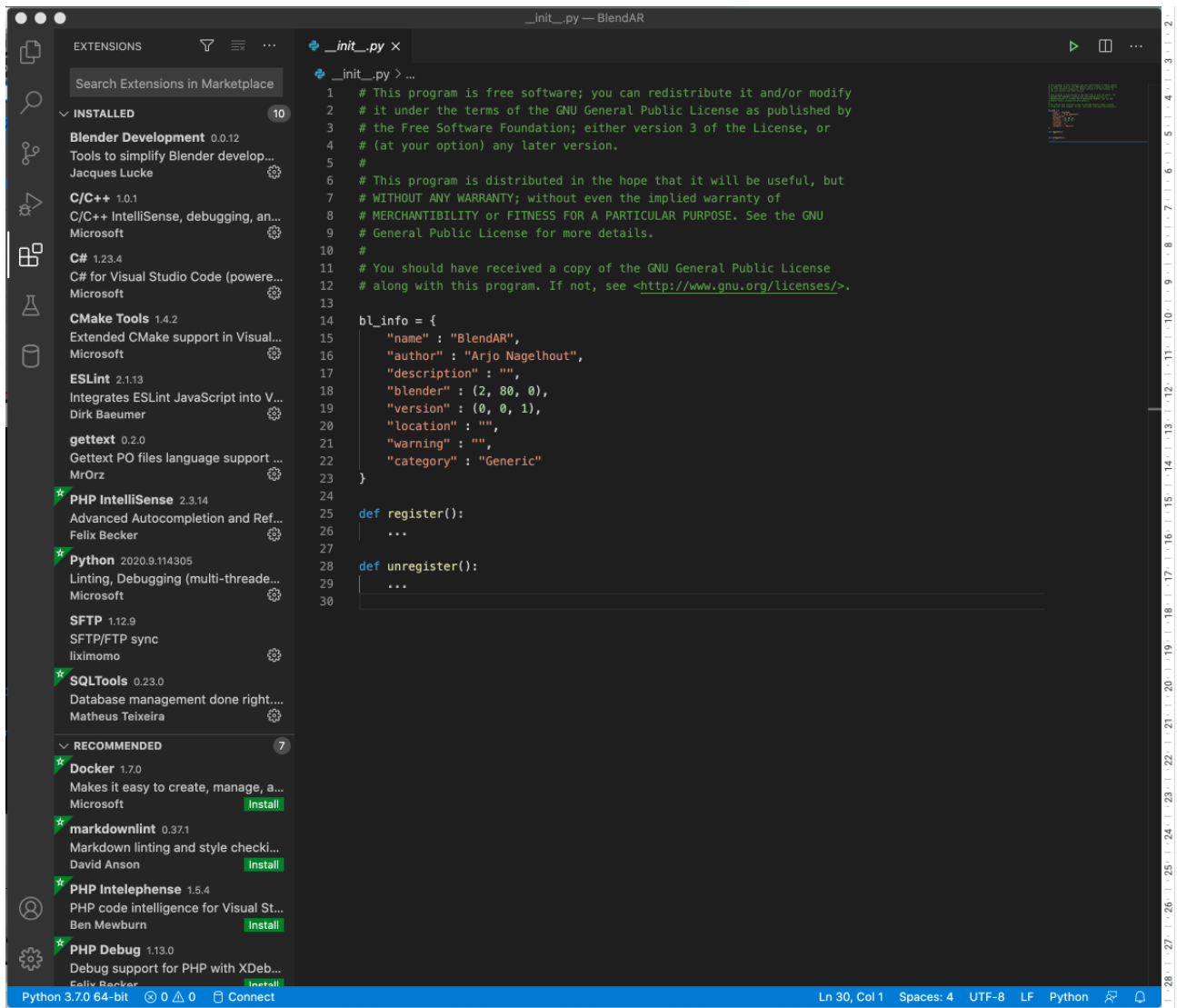
Make planes transformable like rects in Photoshop / Illustrator, along their local axes.

Blender Addon development:

Install Blender + VSCode, Install the Blender Development extension by Jacques Lucke.

Create a new folder

<https://www.youtube.com/watch?v=uahfuypQQ04>



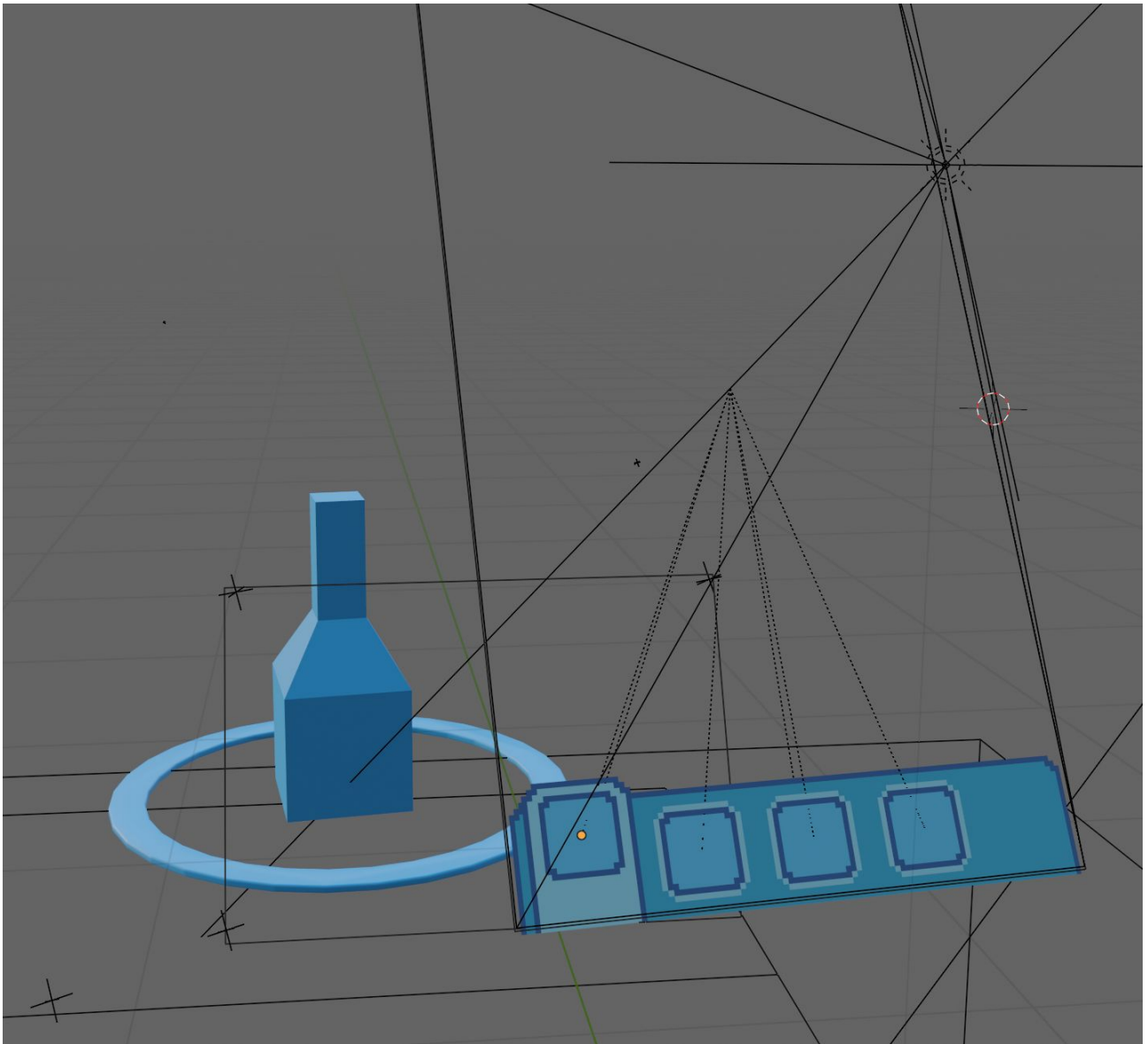
The screenshot shows the Visual Studio Code interface with the Blender Development extension installed. The left sidebar displays the Extensions Marketplace with the following list of installed and recommended extensions:

- INSTALLED (10):**
 - Blender Development 0.0.12** by Jacques Lucke
 - C/C++ 1.0.1** by Microsoft
 - C# 1.23.4** by Microsoft
 - CMake Tools 1.4.2** by Microsoft
 - ESLint 2.1.13** by Dirk Baeumer
 - gettext 0.2.0** by MrOrz
 - PHP IntelliSense 2.3.14** by Felix Becker
 - Python 2020.9.114305** by Microsoft
 - SFTP 1.12.9** by Iximomo
 - SQLTools 0.23.0** by Matheus Teixeira
- RECOMMENDED (7):**
 - Docker 1.7.0** by Microsoft
 - markdownlint 0.37.1** by David Anson
 - PHP Intelephense 1.5.4** by Ben Mewburn
 - PHP Debug 1.13.0** by Felix Becker

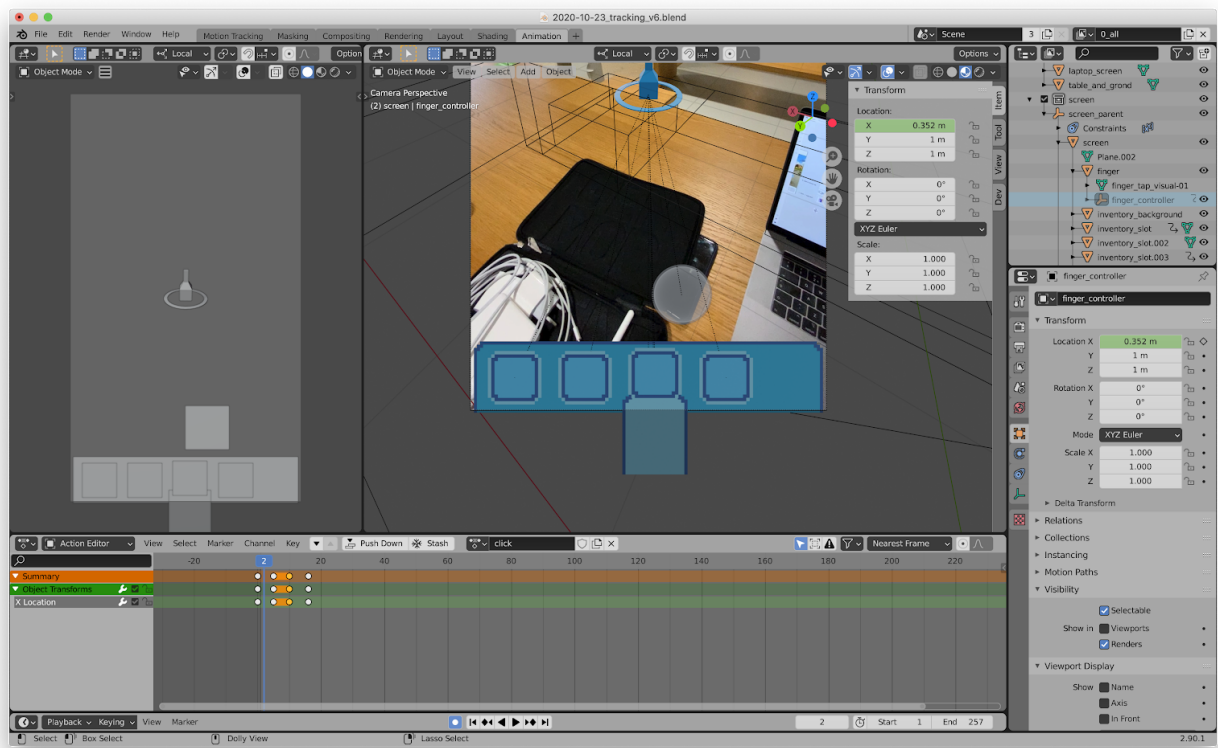
The main editor window shows a Python script named `__init__.py` for a Blender AR mockup. The script includes a license notice and a `bl_info` dictionary:

```
1 # This program is free software; you can redistribute it and/or modify
2 # it under the terms of the GNU General Public License as published by
3 # the Free Software Foundation; either version 3 of the License, or
4 # (at your option) any later version.
5 #
6 # This program is distributed in the hope that it will be useful, but
7 # WITHOUT ANY WARRANTY; without even the implied warranty of
8 # MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
9 # General Public License for more details.
10 #
11 # You should have received a copy of the GNU General Public License
12 # along with this program. If not, see <http://www.gnu.org/licenses/>.
13
14 bl_info = {
15     "name" : "BlendAR",
16     "author" : "Arjo Nagelhout",
17     "description" : "",
18     "blender" : (2, 80, 0),
19     "version" : (0, 0, 1),
20     "location" : "",
21     "warning" : "",
22     "category" : "Generic"
23 }
24
25 def register():
26     ...
27
28 def unregister():
29     ...
30
```

The status bar at the bottom indicates the current environment: Python 3.7.0 64-bit, 0 errors, 0 warnings, and the file is connected to the Blender AR environment.



I added a finger press visualisation, controlled using Actions and drivers: (Similar to the look of Adobe XD's visualisation).`



I added some objects that the user is going to be able to create using the build system.



Now, for moving the object in space, it's centered in the middle of the screen. It can be rotated with one finger dragging. Placement can be done by selecting "Place".

It rotates around its center point. Dragging horizontally results in global Z-rotation. Dragging vertically results in local Y-rotation.



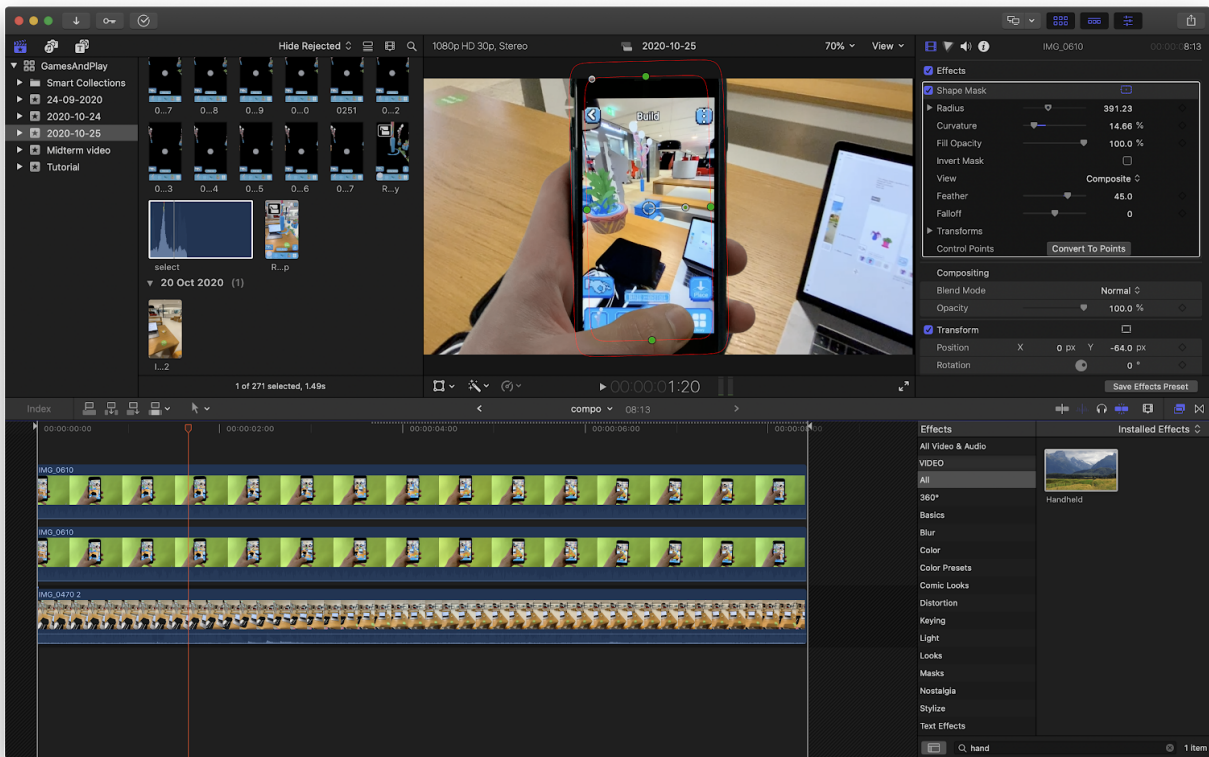
Adding text using Blender's text rendering system.



A greenscreen is used for showing how a user might interact with it.



Final Cut Pro is used for final compositing:



Learning points:

Use Guided Access on iPhone to disallow touches. An alternative is to put a piece of tape on your fingertip, but this can fall off and could become visible in the shot.

Repeat the footage around 6 times and practice it over and over again to perform the exact movements that were done in the 3D render.

When creating a mockup with a finger, don't create a digital fingertip. This will show in the final footage and discourage natural finger movements.

For tracking,

- a light environment is preferable.
- Make sure there is not that much noise.
- Using a high frame rate is preferred, but not that necessary
- Make sure you have enough detail in the environment (high contrast points such as these trackers:)



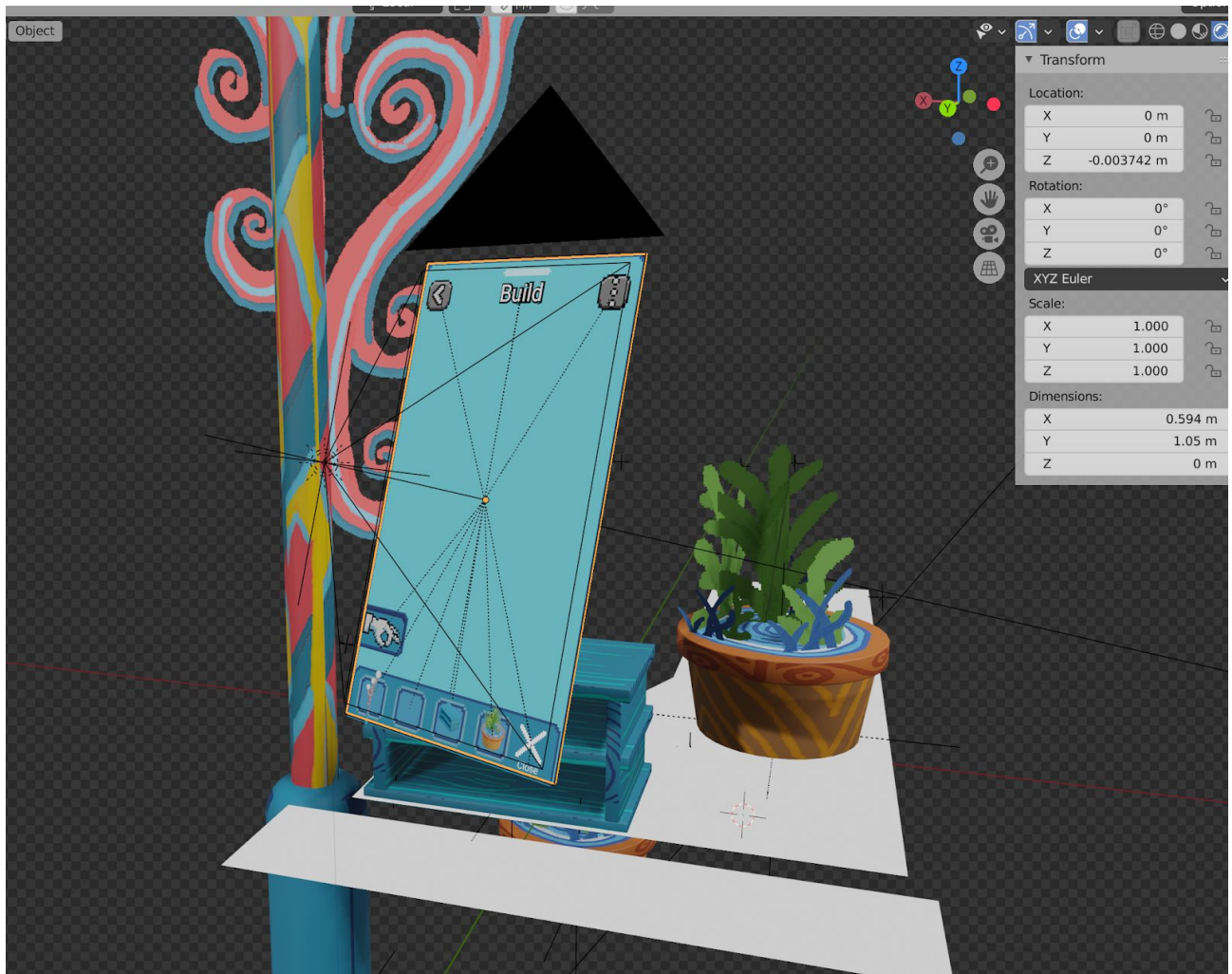
- These trackers can be removed, but that is not necessary, because they are not that distracting.
- Make sure you have trackers far away from the camera, as well as close. Make movements around a position, don't stand at a single position, by doing this, tracking won't work.
- Use plane tracking if there is not much movement in the scene

Before shooting, plan your camera movements. Determine what the interaction is that you want to show. If the interaction is only shortly showing the real world, take advantage of that.

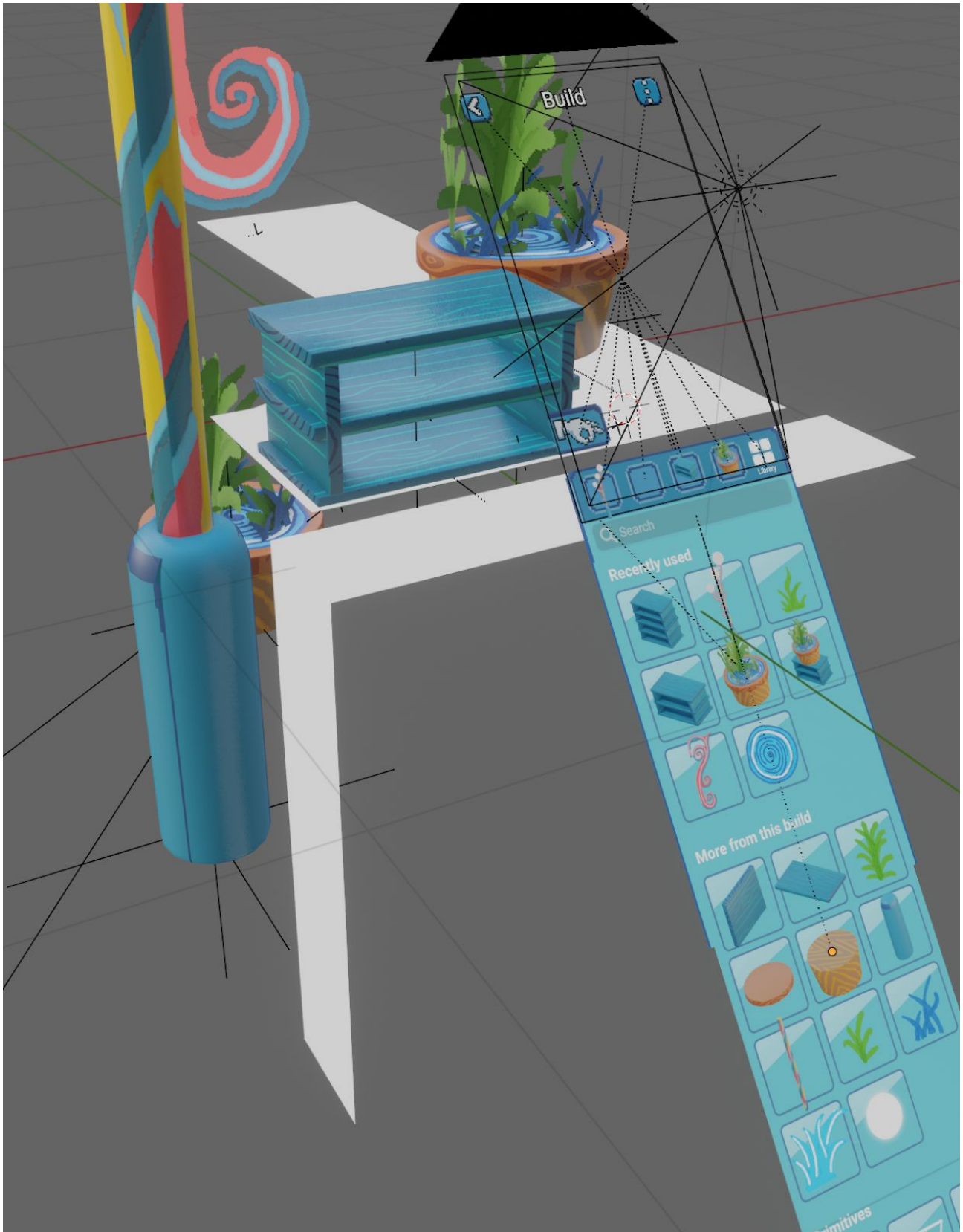
Make a lo-fi mockup using Adobe XD to quickly mockup the screen layout.

Create high quality screen assets using Photoshop and Illustrator.





The current challenge is how a scrollable surface can be constructed. Maybe by translating the UV coordinates? Maybe by adding a view layer? A mask? A video playing?



One thing that can dramatically improve performance is removing the 3D rendering when it is behind the sheet. This can be achieved by separating the rendering into two parts.

Maybe it's an idea to first plan the shot, perform the movements on a blank green screen, then import this footage into Blender and composite it with the tracked camera shot.

Issues with using Blender for this: colors are off.

Maybe use After Effects / Apple Motion? -> no integrated animation / timing. Maybe just keep track of the frame numbers?