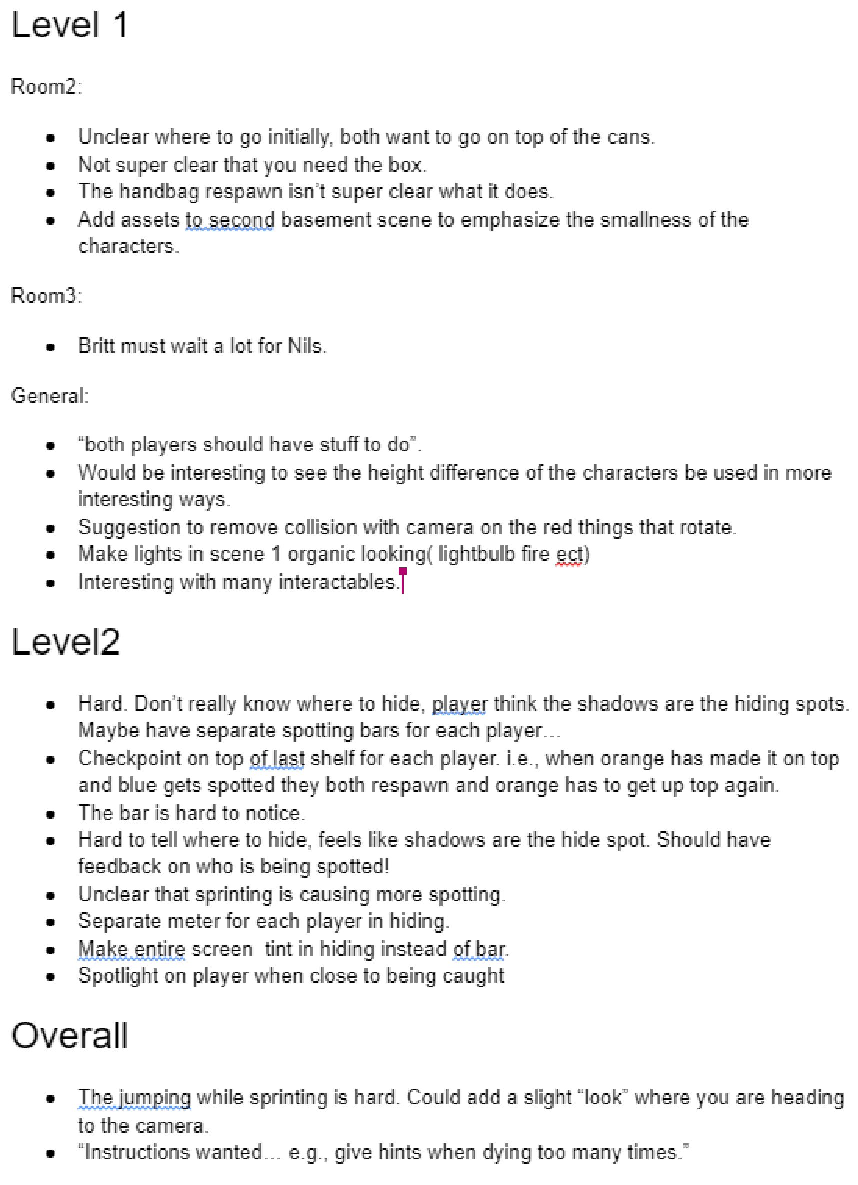
Demo/playtest number 2

The week started with a playtest with all our course mates. I collected data by note-taking while observing and talking with the participants. The un-edited notes can be seen in the image below.



Level 1

Room2:

- Unclear where to go initially, both want to go on top of the cans.
- Not super clear that you need the box.
- The handbag respawn isn't super clear what it does.
- Add assets to second basement scene to emphasize the smallness of the characters.

Room3:

- Britt must wait a lot for Nils.

General:

- "both players should have stuff to do".
- Would be interesting to see the height difference of the characters be used in more interesting ways.
- Suggestion to remove collision with camera on the red things that rotate.
- Make lights in scene 1 organic looking(lightbulb fire ect)
- Interesting with many interactables.

Level2

- Hard. Don't really know where to hide, player think the shadows are the hiding spots. Maybe have separate spotting bars for each player...
- Checkpoint on top of last shelf for each player. i.e., when orange has made it on top and blue gets spotted they both respawn and orange has to get up top again.
- The bar is hard to notice.
- Hard to tell where to hide, feels like shadows are the hide spot. Should have feedback on who is being spotted!
- Unclear that sprinting is causing more spotting.
- Separate meter for each player in hiding.
- Make entire screen tint in hiding instead of bar.
- Spotlight on player when close to being caught

Overall

- The jumping while sprinting is hard. Could add a slight "look" where you are heading to the camera.
- "Instructions wanted... e.g., give hints when dying too many times."

Based on the feedback, together with the group, I made a list of things that I aim to finish for next week. The following text details what and how.

- Make it clearer where to go Level 1 Room 1, or rather that blue and orange have separate ways to go.
- Not have the players push the box from Room 1 to Room 2.
- Make the handbag in L1 R1-2 more "non-interactive".
- Remove the platforms in the glass-shelf in L1 R3.

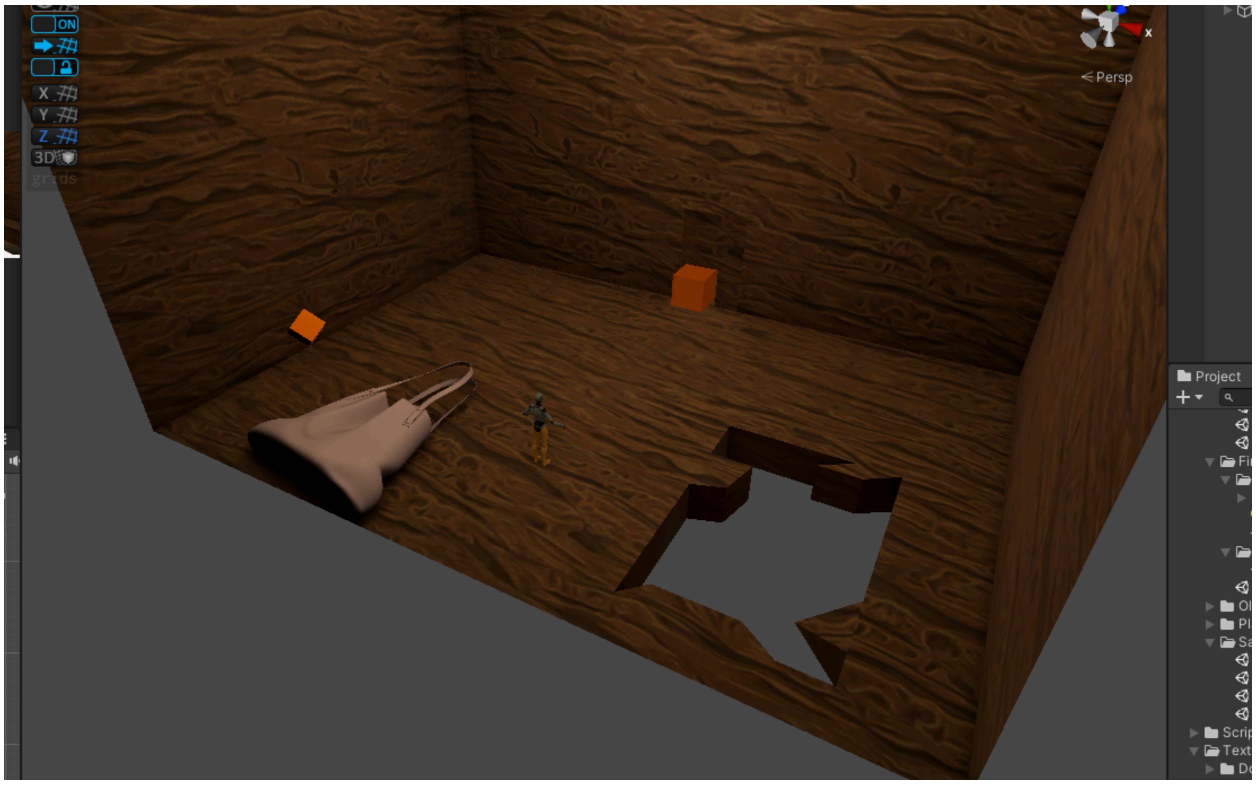
- Stop camera from colliding with rotating obstacles in R3.
- See if camera can be made to slowly position itself behind character when camera is not being manually rotated (maybe only for gamepad).
- Try to give the orange player something more to do in Room3, or minimize waiting.
- Add level transitions.

Fixing feedback pain points

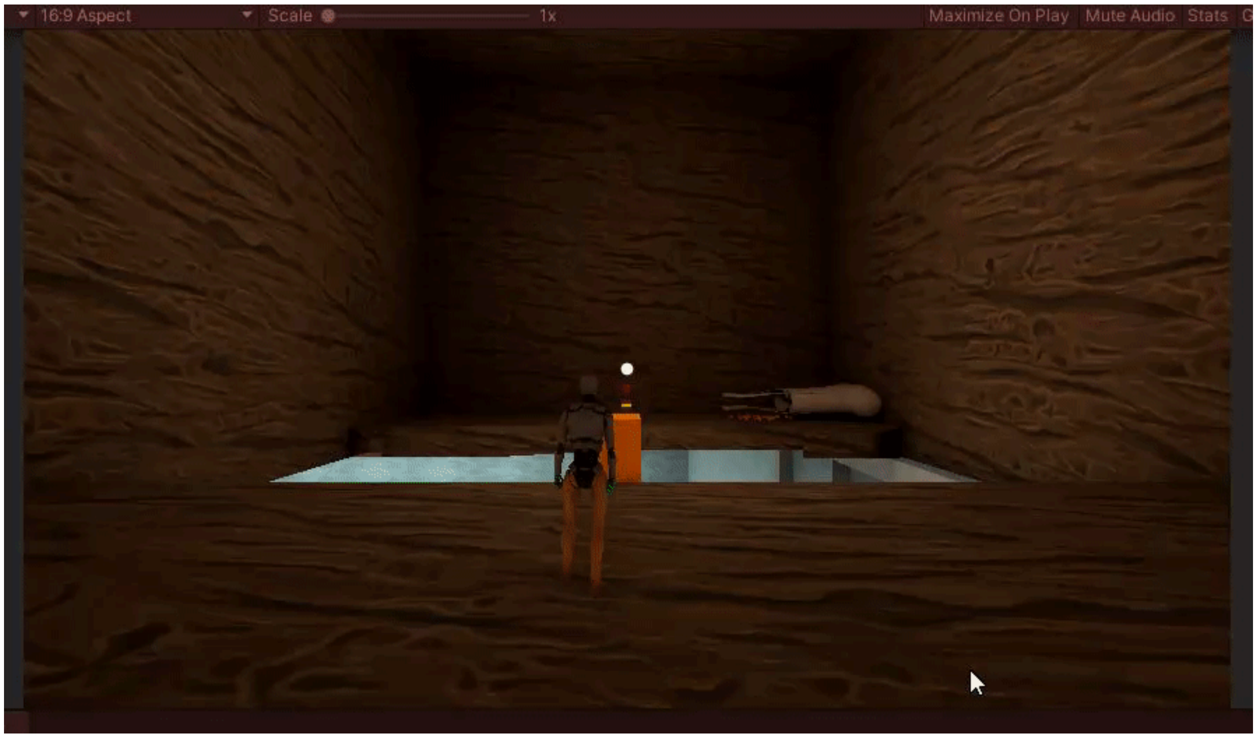
Reducing the "interactive" affordance of the box respawner and teaching the player how it works.

Players felt confused about the handbag that respawns the pushable cubes, that they somehow could interact with it to spawn a cube. I think it mainly was because it had a "lifesize" cube above it, but its functionality could be easily taught by showing how it works if I can force the player to push a cube into a deadzone at the start of the level.

Working on this starting area I also decided to create the environment for the "hole". The first two rooms of Level 1 are supposed to be in a cupboard in a kitchen, and this hole is where the players came from. So having this kitchen visible would be cool.



Before: the starting area hole empty and handbag with floating cube above

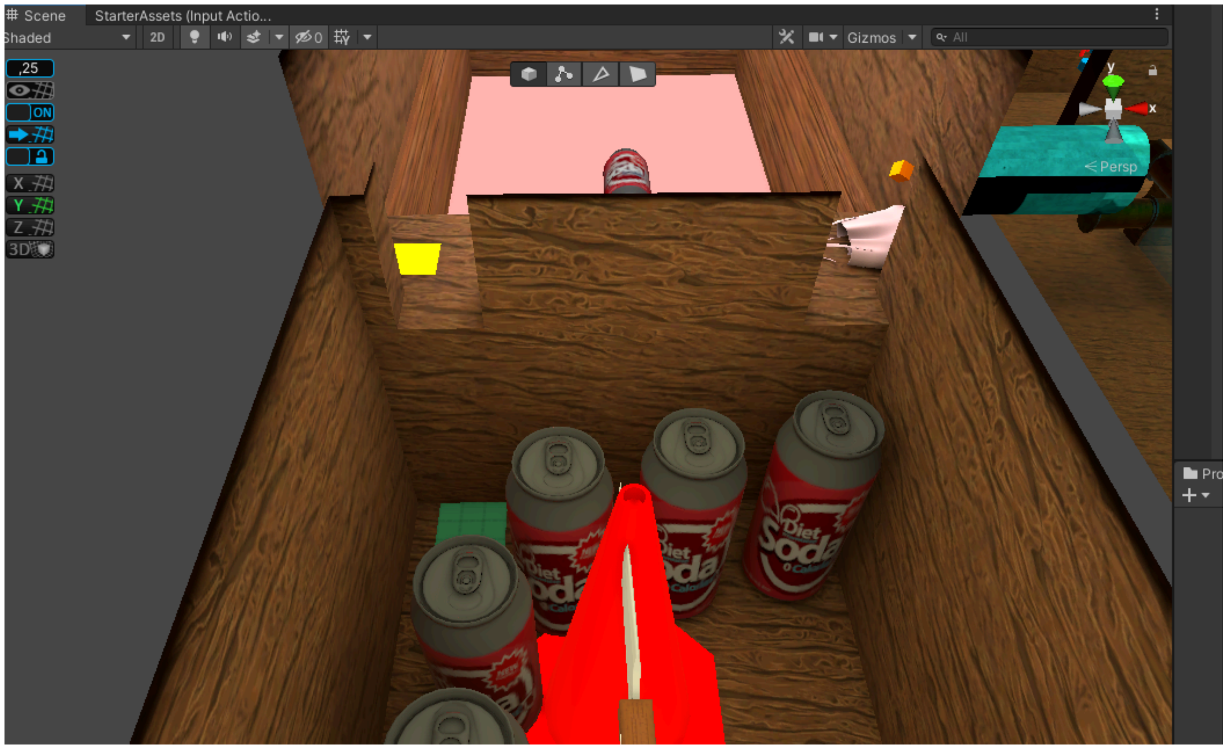


After: forcing the player to push off the box; player sees the box respawn. Kitchen now visible down the hole.

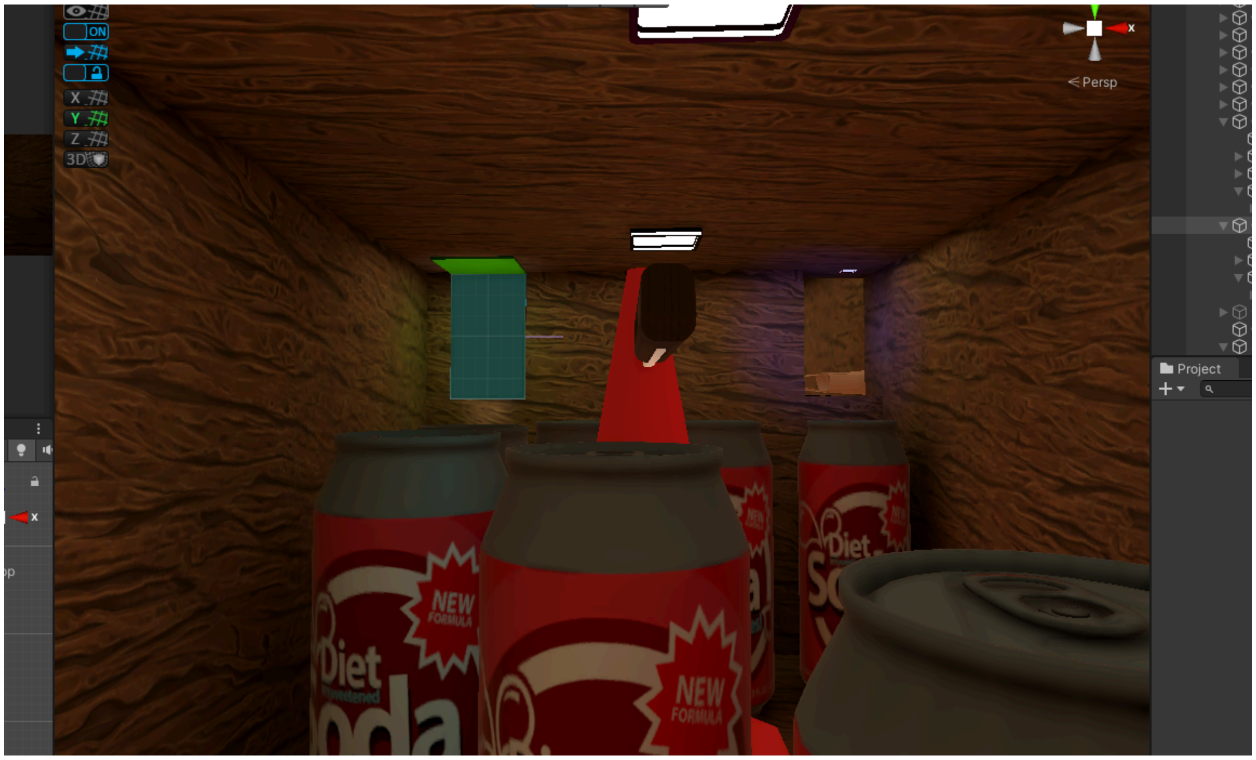
A **challenge** I faced was to make the lighting look good. I was getting weird artifacts in the lightmaps after baking and didn't understand why. Doing some research I learned that lightmapping rays get sort of blocked by backfacing triangles. So the outside of my "cupboard" level was interfering. I also learned how easy it was to enable Occlusion Culling in Unity. Basically just placing some Occlusion volumes and press a button to have Unity calculate a partitioning of the volume.

Some changes in Level 1 - Room 1 & 2

Based on the feedback I think some players felt kind of tricked when realizing they had to backtrack to get the box from Room 1 to Room 2 as there was no indication that they would need it later. So I just placed another pushable box in Room 2 and changed the "puzzle" in Room 1 per the images below. The puzzle in Room 2 seemed "puzzling" enough for the players without having to figure out they need the box from Room 1. It was fun to see how the players lit up when realizing how to solve the puzzle in Room 2.

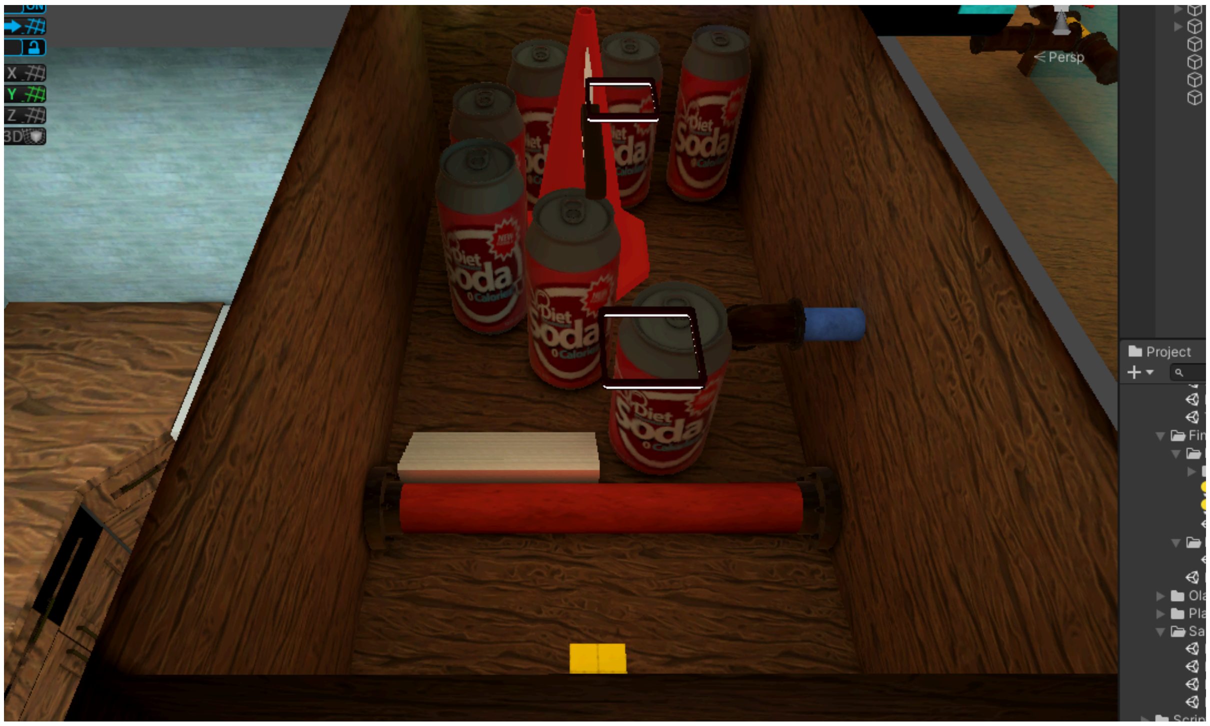


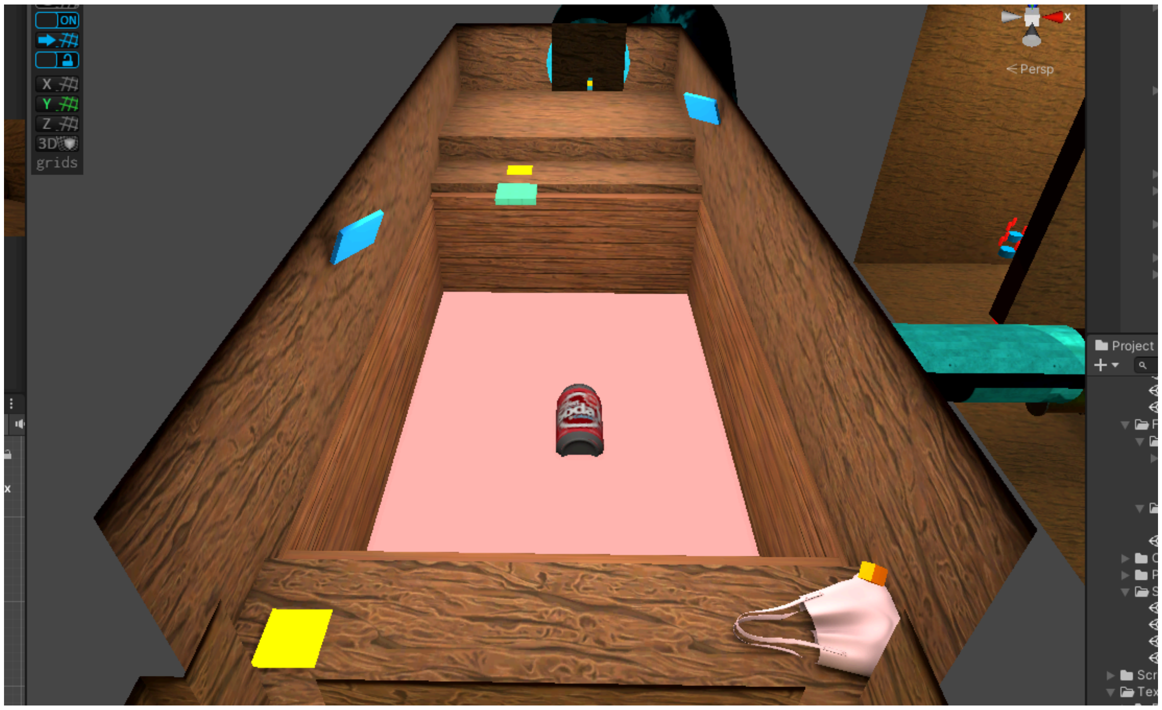
Room 1 before: The green elevator to the left in order to bring the cube. Only the short player can reach the right entrance and activate the elevator on the yellow button.

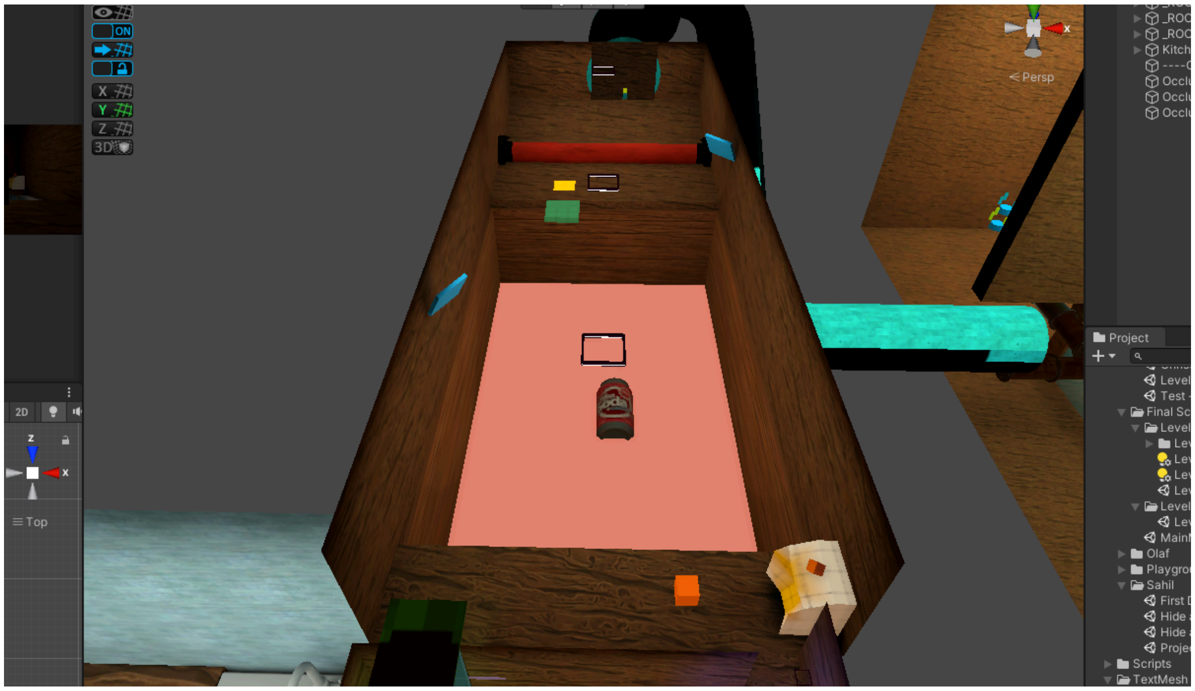


Room 1 after: Elevator removed. Now a "magic movable" blocking the left that the 'blue' player can move. Only 'blue' can enter to the right. 'Orange' has to grapple on the green surface on the ceiling to enter on the left. Added some tinted lighting at these exits.

I also made the “obstacle” for when the box is needed more consistent by placing the same red pipe in Room 2 as I have in Room 1.


Room 1 with red pipe that the players need a box to get up on


Before: Room 2 with raised wood ledge on the north side that the players need at box to get up on


After: Room 2 ledge replaced with red pipe as obstacle for consistency

Having the camera recenter behind the character when moving

One player during the playtest didn't like the free look camera, but instead wanted it to stay behind the character as they moved. There is a lot of camera work that could be done in the game and I don't have time to get deep into it. But I was interested in figuring out how to code the camera to smoothly stay behind the player.

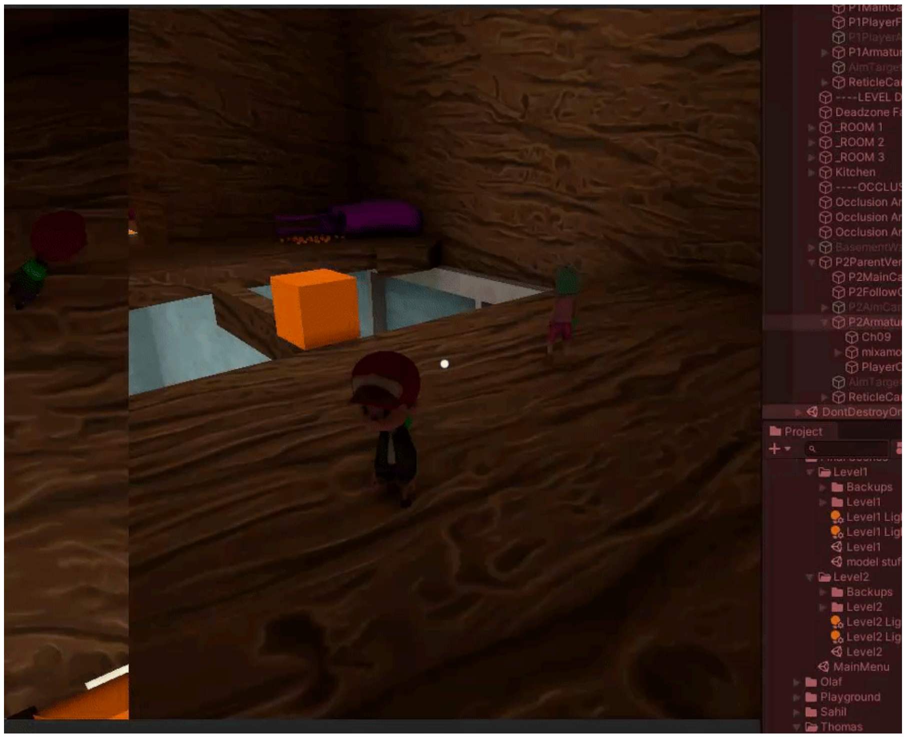
It's basically two lines of code added to the current CameraRotation function, if the player is moving and using a gamepad:

I get the Y-axis angle from a desired quaternion rotation:

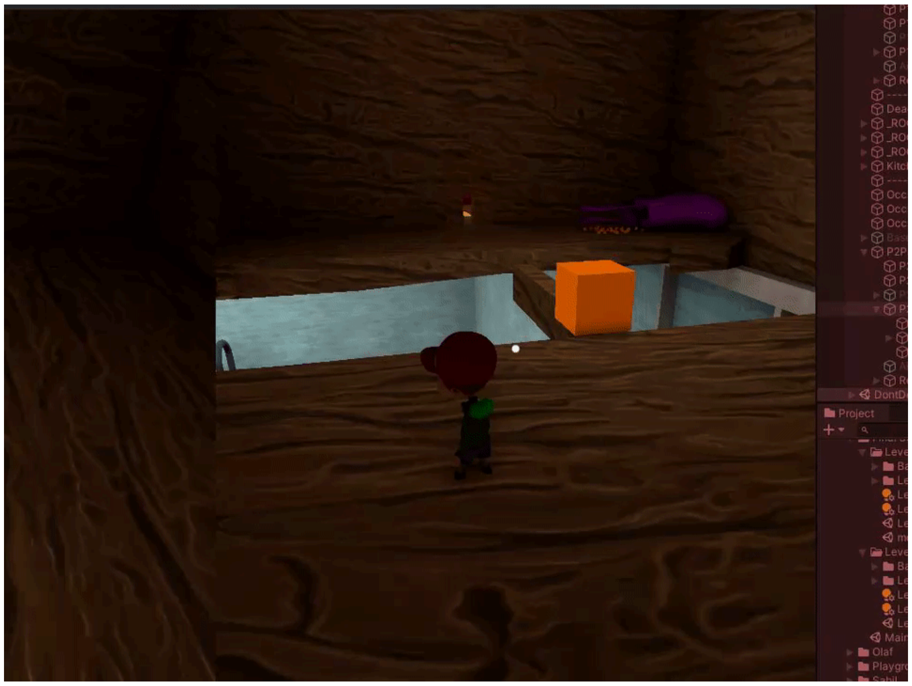
```
        targetRotation =
Quaternion.LookRotation(playerArmature.transform.forward).eulerAngles.y;
```

And apply this to our camera's target Yaw, the SmoothDampAngle helps apply this in a "smooth" fashion faster or slower depending on *recenterToHeadingTime*:

```
        cinemachineTargetYaw = Mathf.SmoothDampAngle(
                                    cinemachineTargetYaw,
                                    targetRotation,
                                    ref cameraRotationHeadingVelocity,
                                    recenterToHeadingTime);
```



The free look camera. Doesn't rotate without player input.



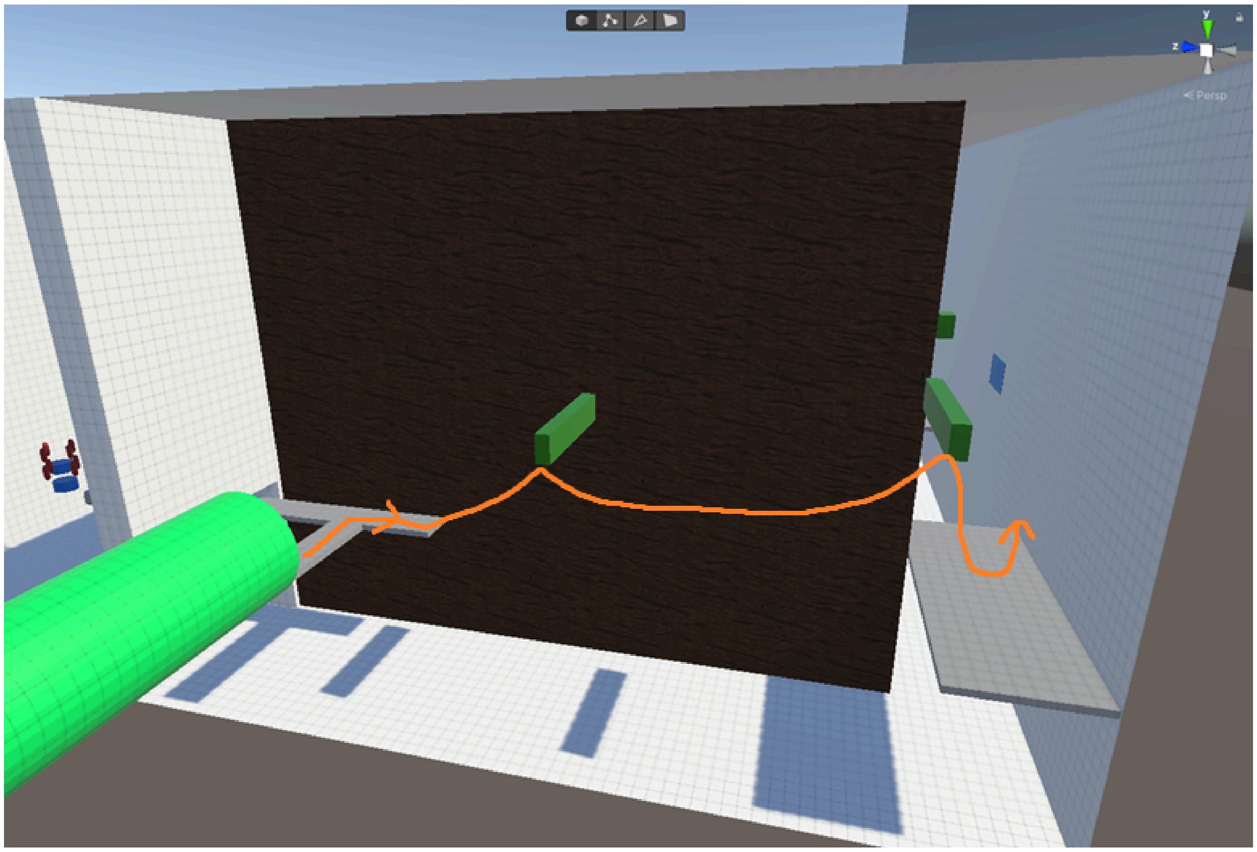
The recenter to heading camera. Rotates to stay behind the character as they move.

When playing with a gamepad this might be more comfortable. It's a bit weird when making a sharp 180 turn, the camera should maybe make a fast flip if the input changed that much.

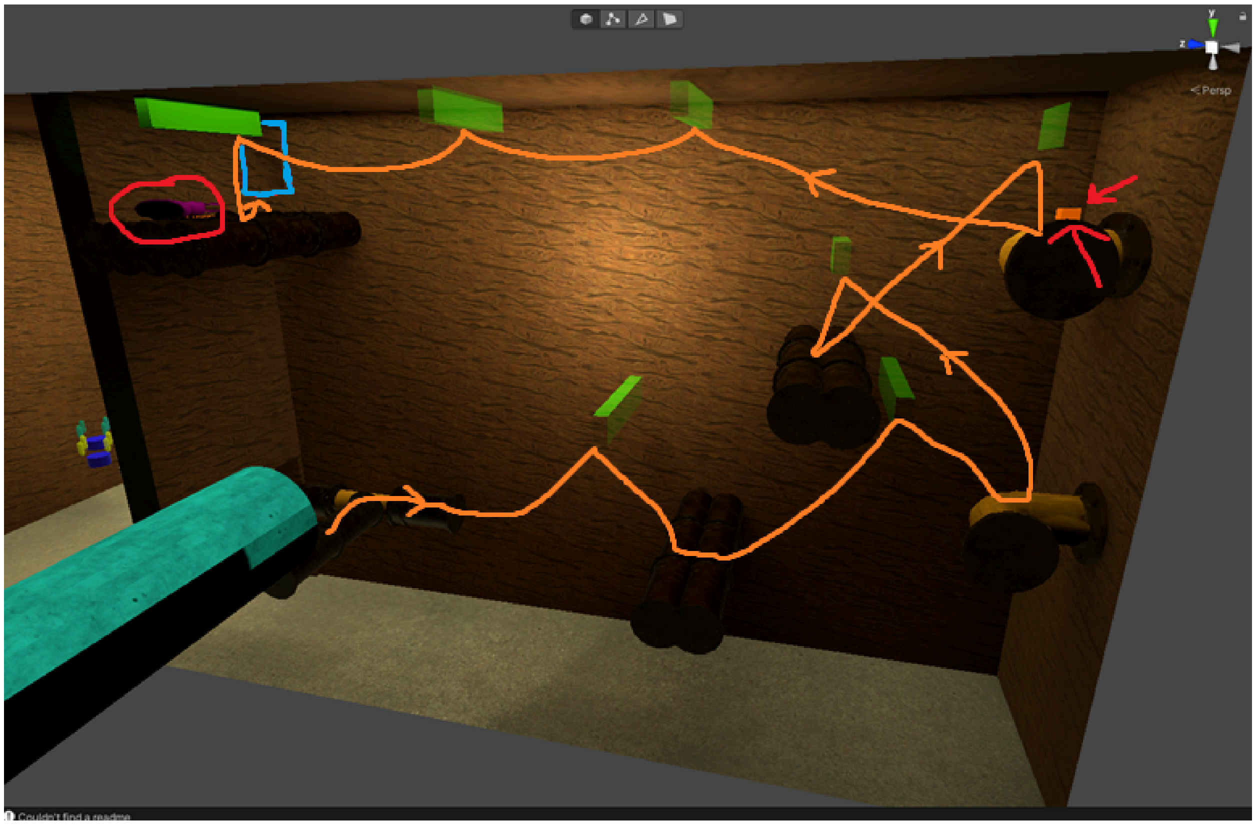
However, since we're closing in on the deadline, I don't have time to implement a UI for controller settings and I don't have time to gather player feedback, so I won't hardcode this into the game right now.

More puzzle design in Level 1 Room 3

From the playtest I noticed that the Orange player had to wait a bit for Blue, especially at the beginning of Room 3. I decided to add an extra puzzle for Orange to do here.



Before: The player just had to swing across from west to east and there wait for the other player to conjure a grappable.



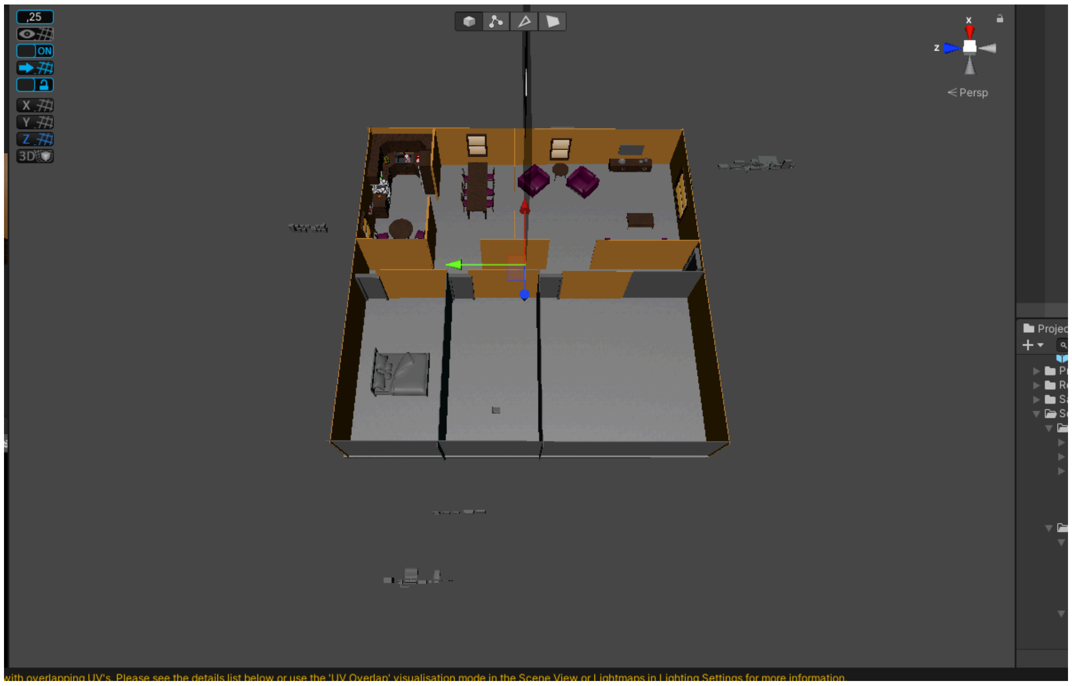
After: The player has to do some more grappling to platforms, and solve the puzzle of getting the box from upper right to upper left.

So the player gets to have some more fun with the grappling hook and figure out how to get the box in the upper right (pointed out with red arrows) to the raised opening in the wall to the upper left (outlined blue). The solution is hopefully apparent as the player sees the distinct "box returner" in the upper left (circled red) and realizes that they can push the box down to the floor (which is a deadzone) causing it to re-appear in the upper left. They can then push the box to the opening in the wall and get up there.

Writing this afterwards I now realize that the player can use their momentum from grappling to just "fly" to the hole based on the placement of the last green grappable surface in the upper left. Better fix that for the final version…

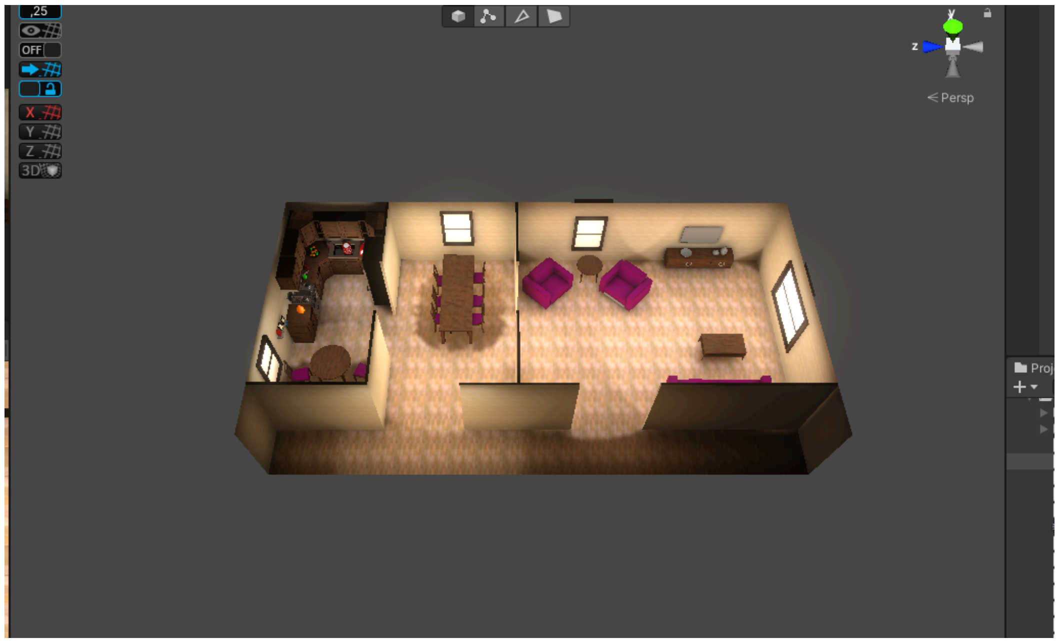
Lighting in Level 2

The team-member working with the environment in Level 2 was struggling with the lighting in Level 2. Since I learned pretty good how it works I took care of fixing it. The idea in this level is that shadows should indicate hiding spots, so it's quite imperative there are good shadows.

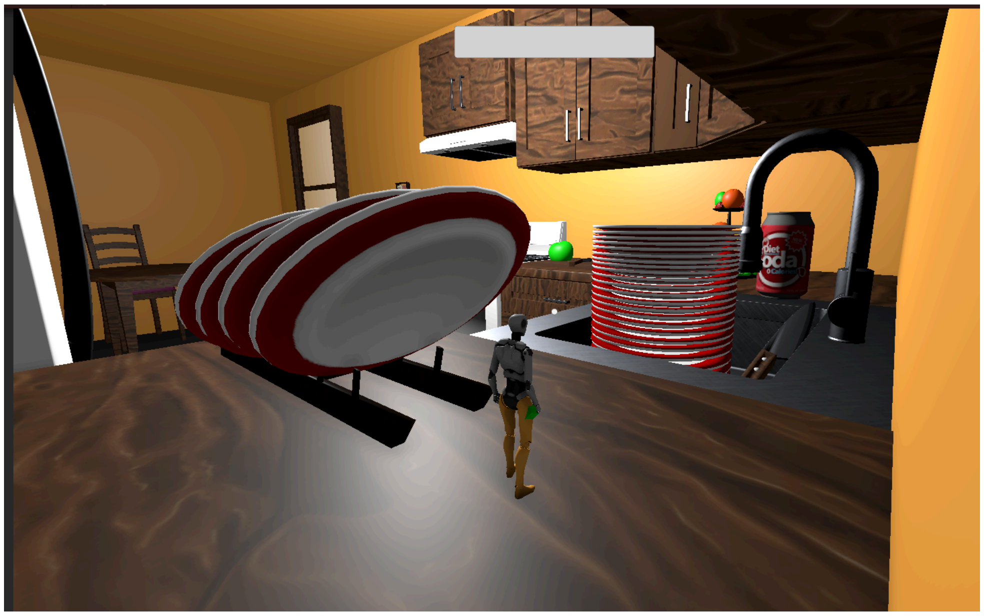


with overlapping UV's. Please see the details list below or use the 'UV Overlap' visualisation mode in the Scene View or Lightmaps in Lighting Settings for more information.

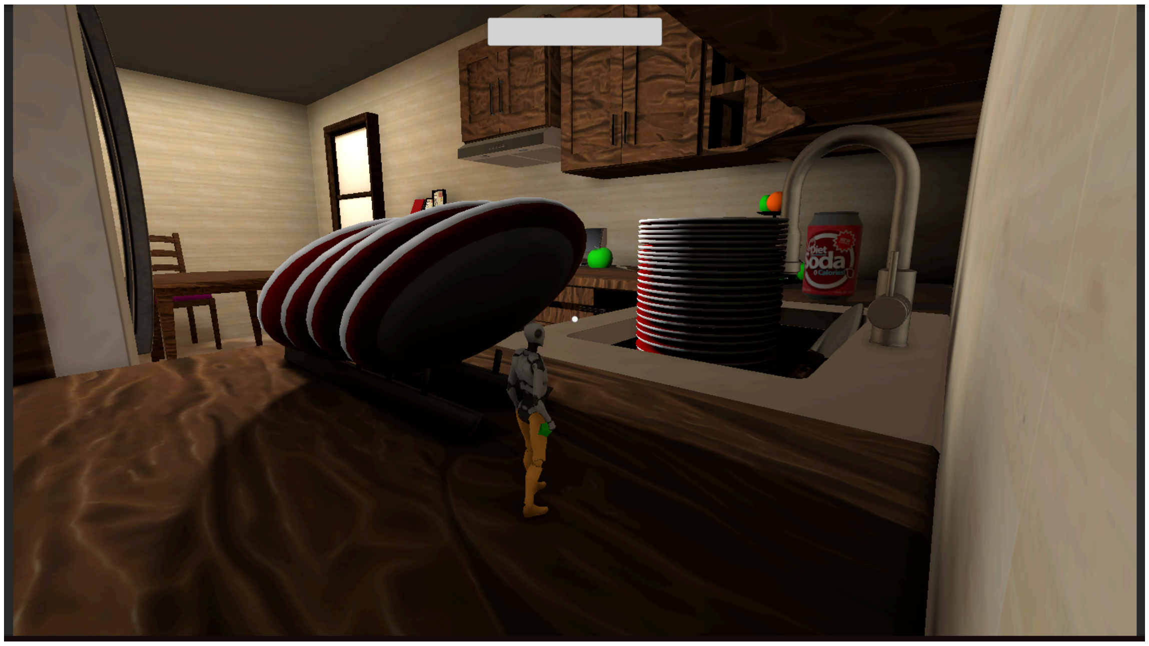
Overview of how it looked before



Overview of how it looked after

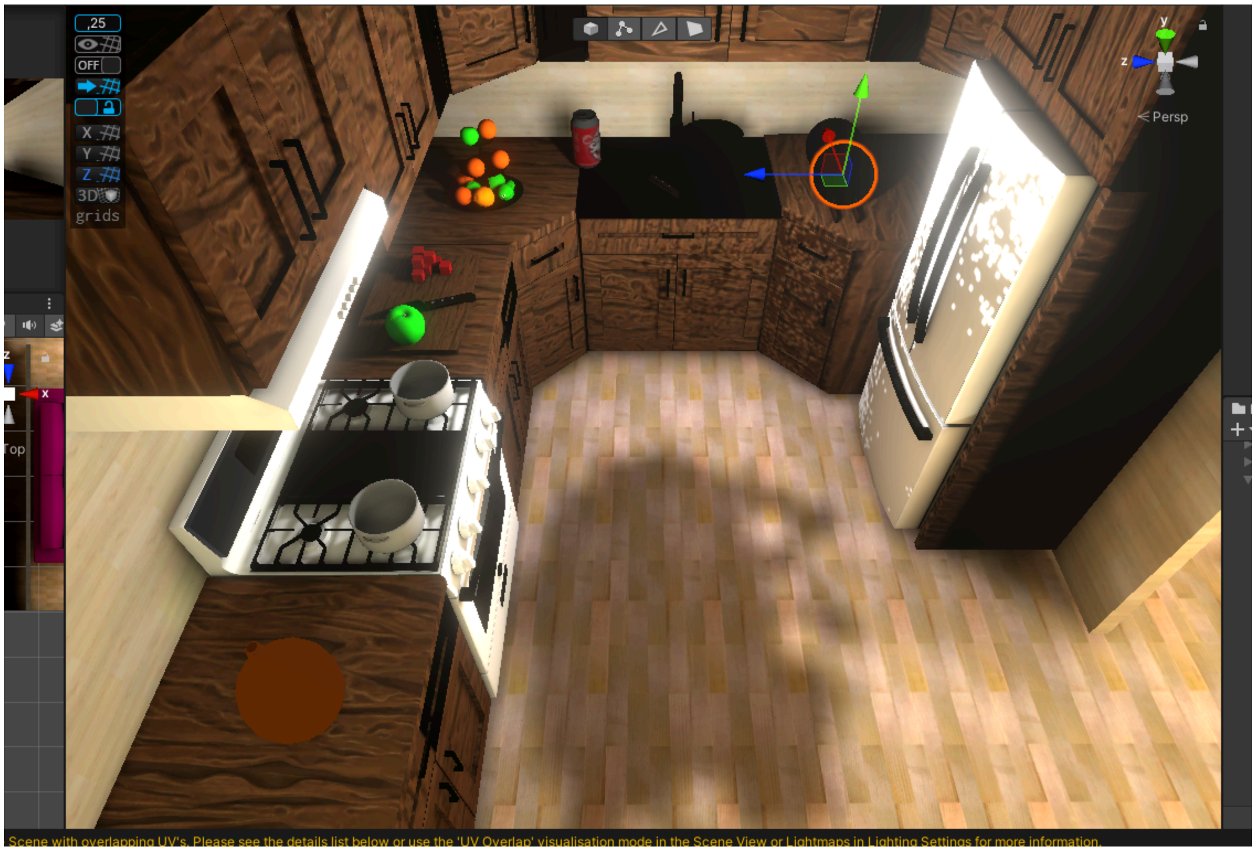


How it looked while playing before. The scene used directional lighting which naturally didn't look good in an indoor environment.



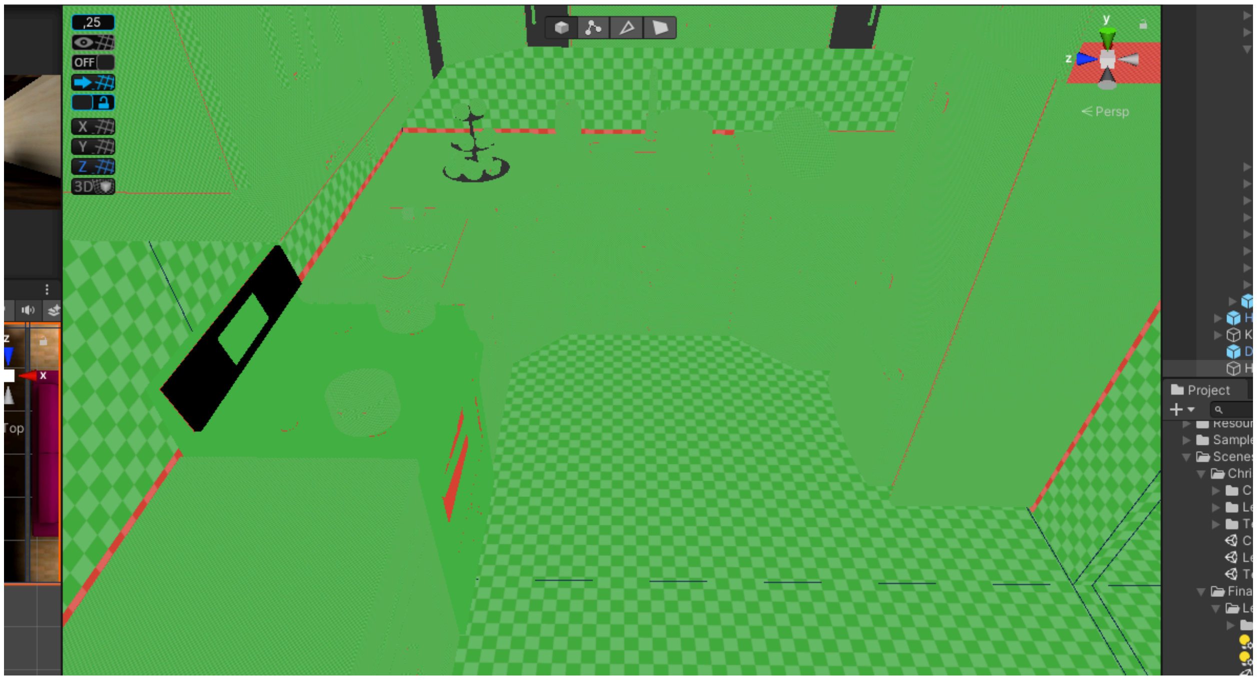
How it looked while playing afterwards, with baked lighting.

In the original scene the rooms were built out of Unity cubes. A lot of the geometry was intersecting and non-perpendicular, resulting in a bad lightmapping. So I rebuilt the rooms and made sure all objects were placed well, not intersecting etc.

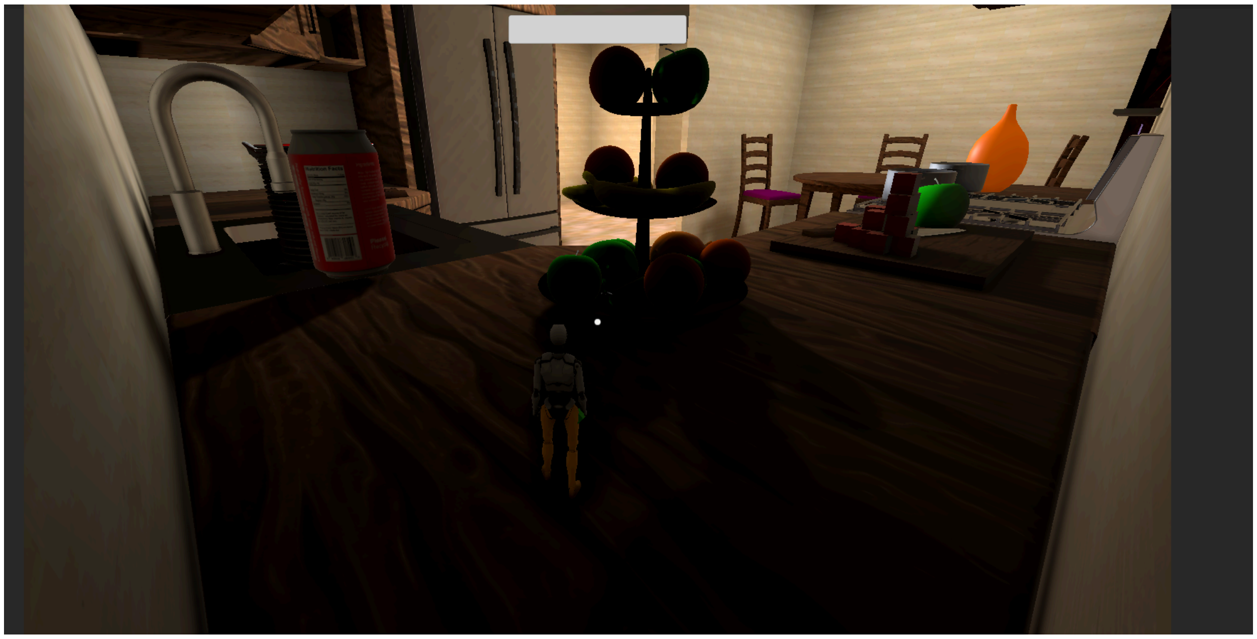


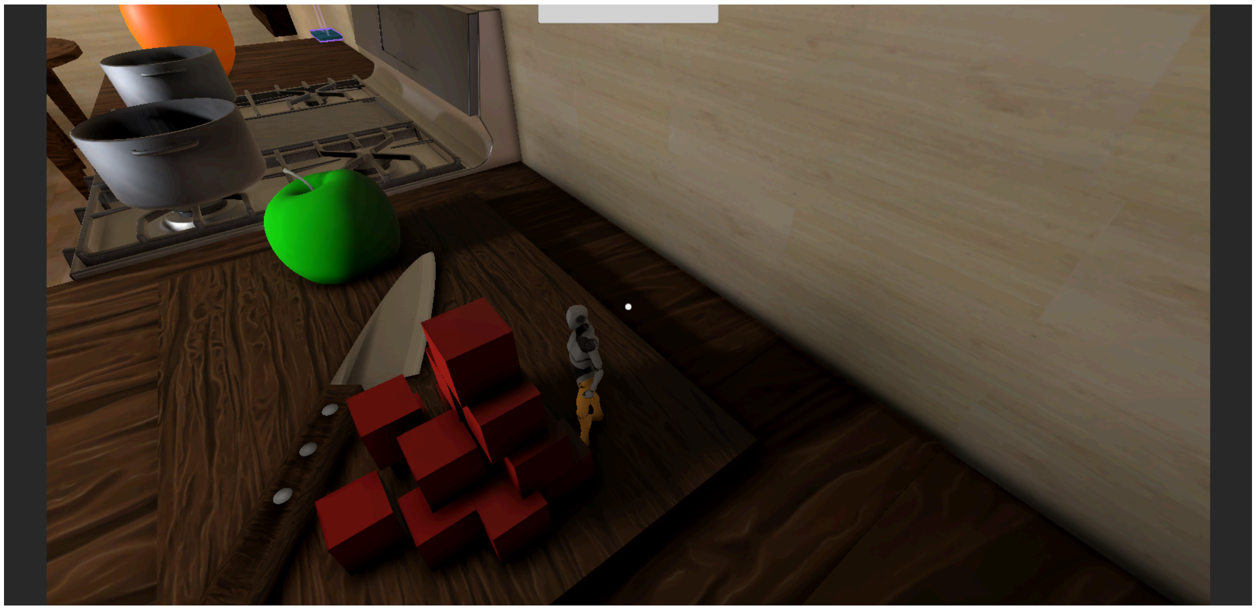
Scene with overlapping UV's. Please see the details list below or use the 'UV Overlap' visualisation mode in the Scene View or Lightmaps in Lighting Settings for more information.

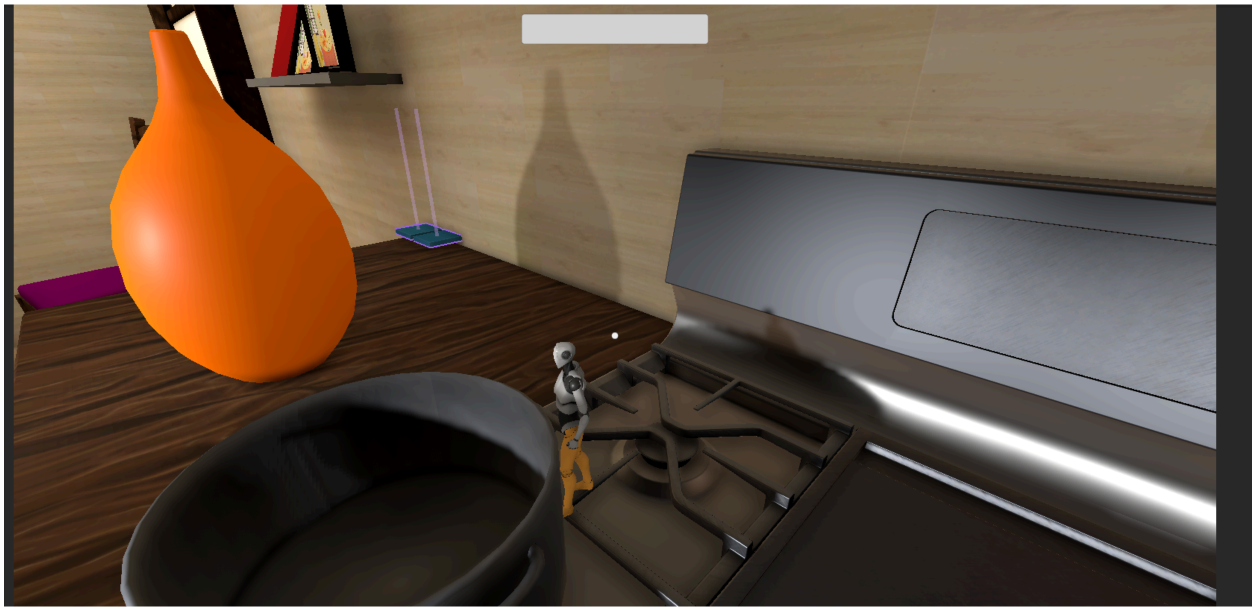
In the first bake, some objects were creating artifacts and became black because of poor shaders.



Investigating the texel validity. Mainly the window on the stove door was creating issues, so I just removed it.


Was afraid it would be too dark in the corner, but it felt alright when actually playing.


The cubes and apple giving off clear shadows.

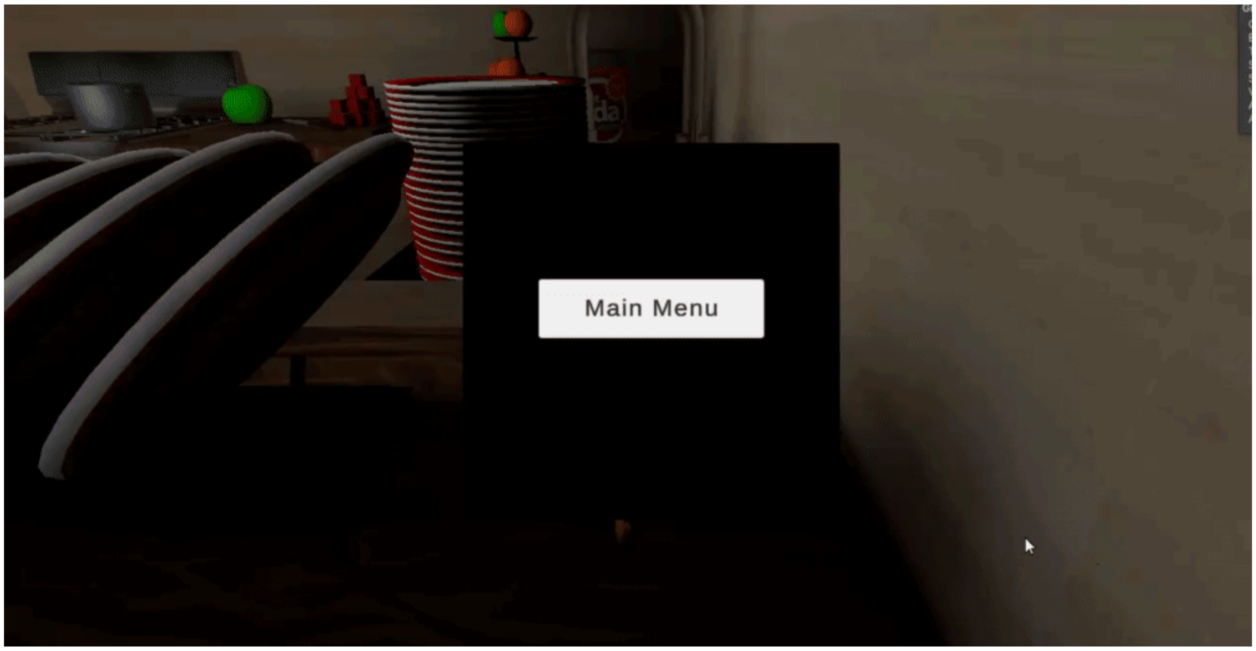


It was hard to get distinct shadows on the metallic stove surface. I used a real-time spotlight to get the shadow effect here and for the dynamic movable orange vase. When profiling it hardly affected the frame rate.

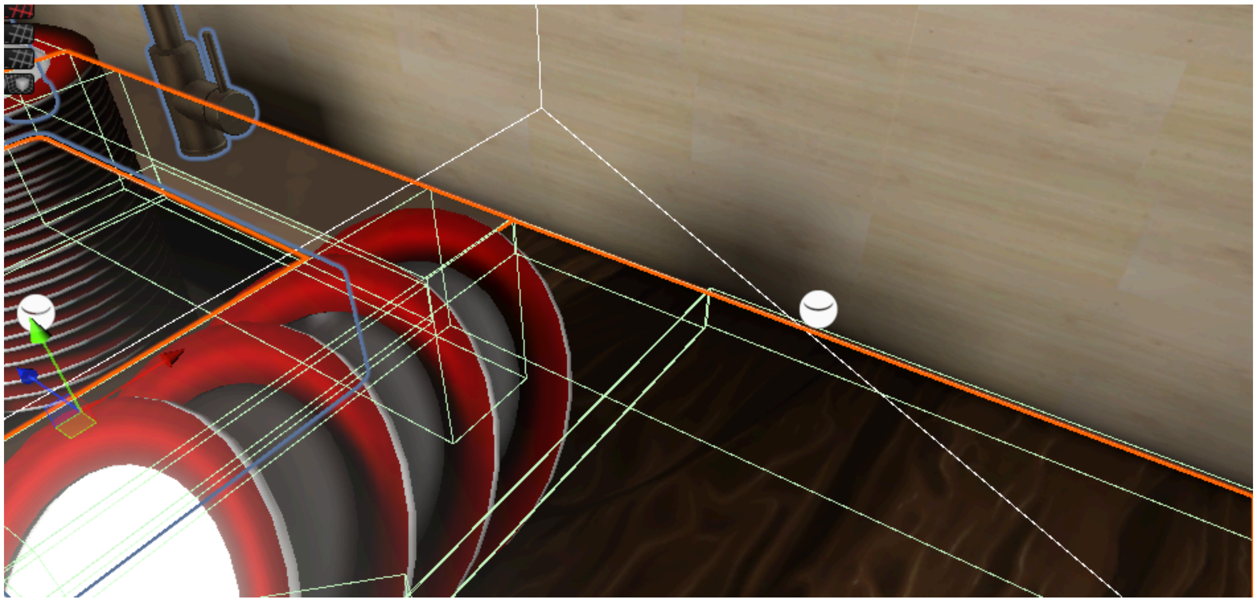
It was quite challenging trying to get clear shadows baked the direction you wanted it and having it look natural. I estimate I spent a lot of time experimenting with different ways. Using too many light sources the shadows became too bright, so to speak, not casting a dark distinct shadow, and the lights had to be quite strong and close up to create distinct shadows. But if they were too bright, or too close, the object casting the shadow would light up like the sun. I eventually found a good spot for a single point light to cast shadows in good enough directions for all the objects and areas that had been programmed to be “hiding spots”.

More sliding improvements

I noticed that the character was getting stuck weirdly sometimes, which I quickly guessed was another issue for my “sliding” code. Between the seams of two colliders when walking slowly it detects the vertical face of the collider you’re approaching.



Character getting “stuck” between two colliders.



The colliders at that area.

The problem was that I was applying sliding from several hits, which I felt was the correct logic at the time I wrote it. I changed the logic to only consider the ONE collider that is closest to the character, which would in these cases be the flat ground and not include the vertical face beneath. So far it works fine.