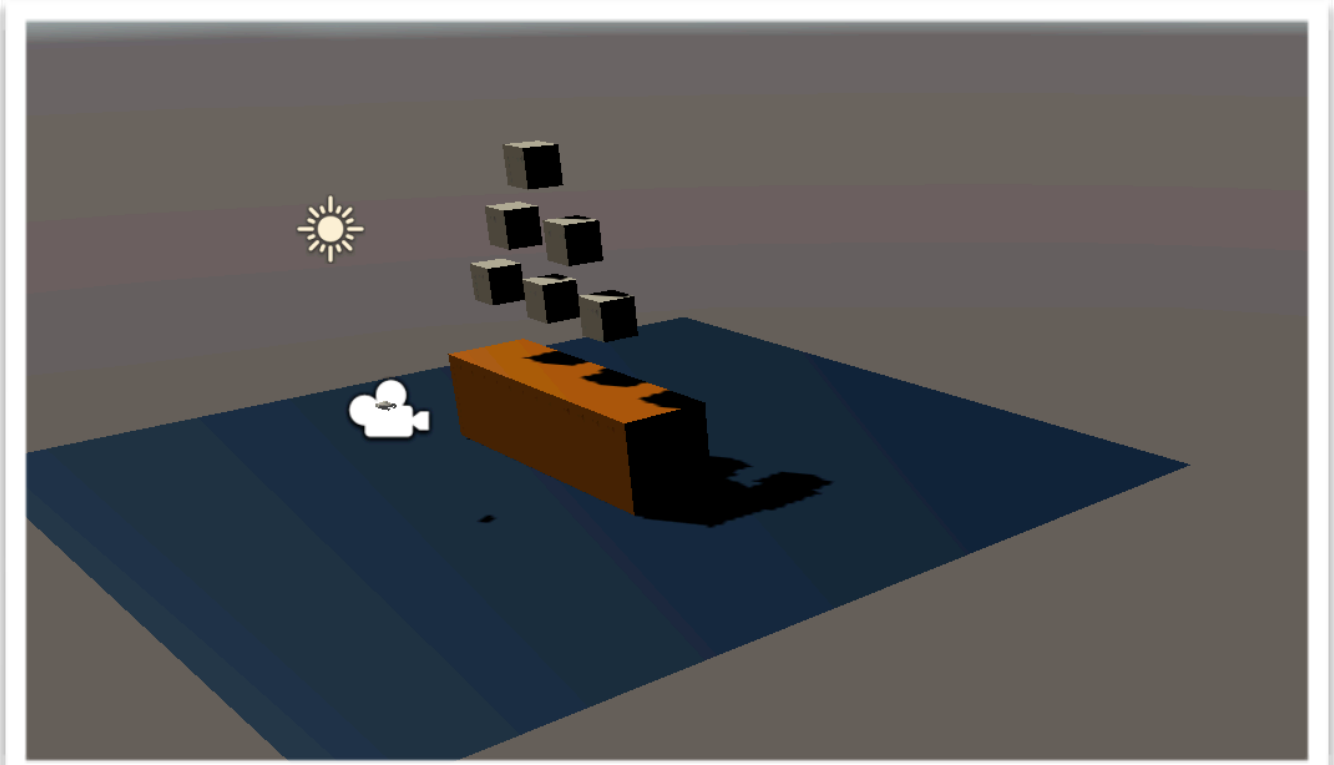
Tutorial Distance Grab(or named Magic Grab)

1. **Distance Grabbing** : The Unity DistanceGrab sample scene demonstrates how you can implement custom hands that can point at distant objects and have them zoom into their hands when the grip trigger is pulled. If an object is close enough to the hand to be grabbed normally, without zooming, it is grabbed in the same way as an Oculus Avatar's hands. The user can also move around using the thumbsticks to demonstrate targeting range. [You can find this scene by search in the assets with the name DistanceGrab. Test and run to see an ideal distance Grabbing you can achieve.]

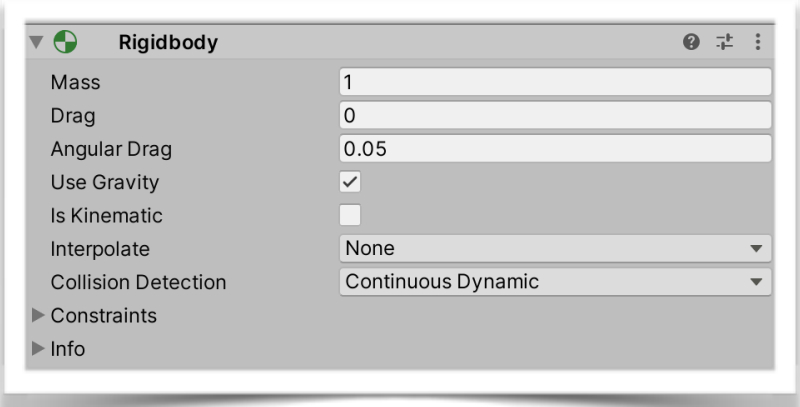
Okay now Let's try implement this by ourselves, it's can be easily achieved by the distance grabs prefabs:

Step 1:

Again, Create a Scene named DistanceGrab. Again drag the OVRPlayerController Into the hierarchy. Create a floor, a desk, and some grabbable cubes. (You can create the scene as you want.). Add a rigid body to the grabbable cubes. Finally, Change the grabbable cubes layer into Grabbable. Remember the number of the layer. The grabbable layer is the one that the raycast physic will interact with.

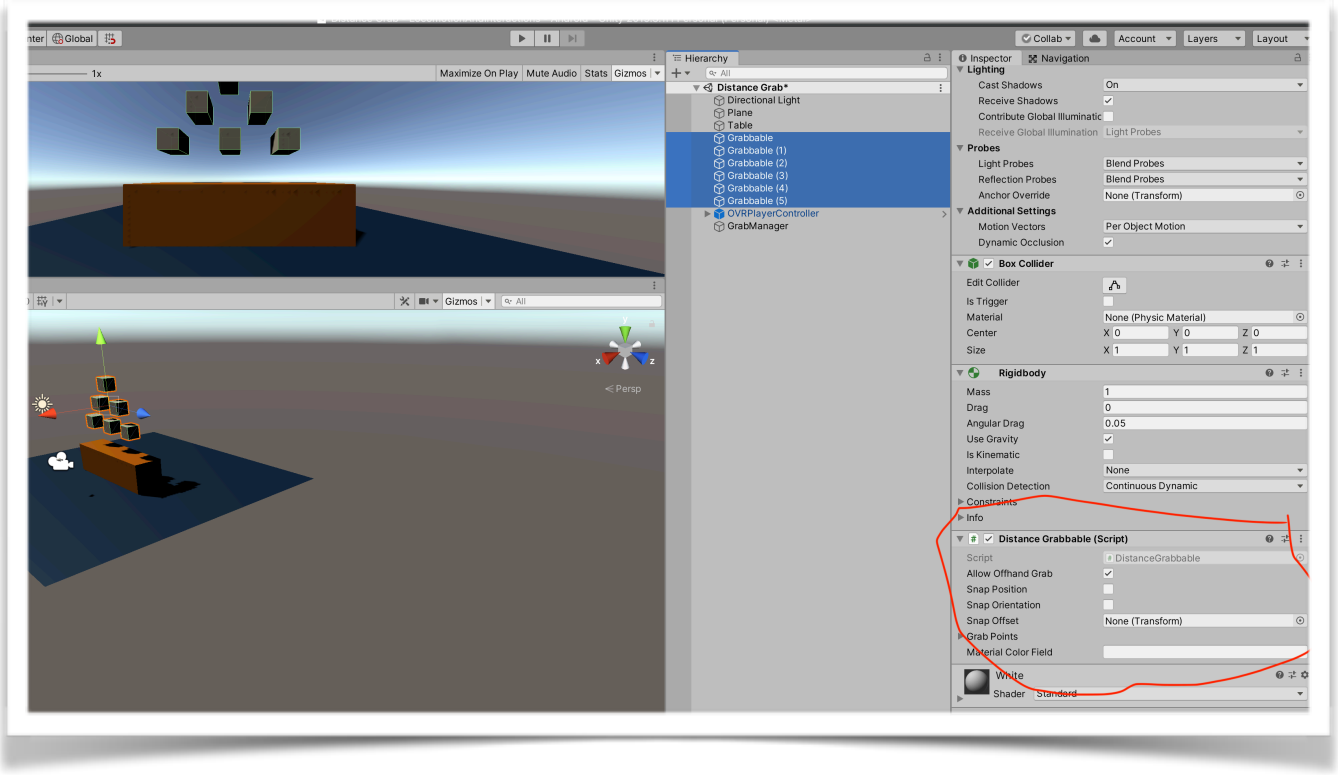


Change the Collision Detection parameter into Continuous Dynamic. This mode can better react to physics. Smoother.



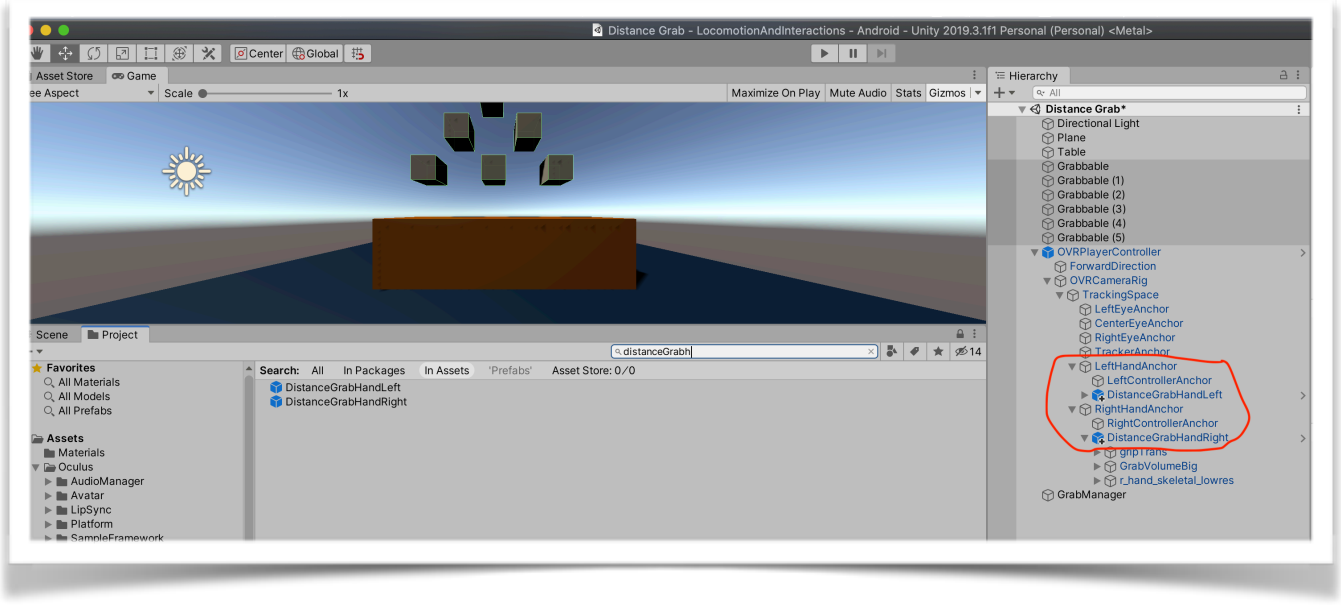
Step 2:

For each grabbable cubes, click add component, use the search to find the script : DistanceGrabbable.



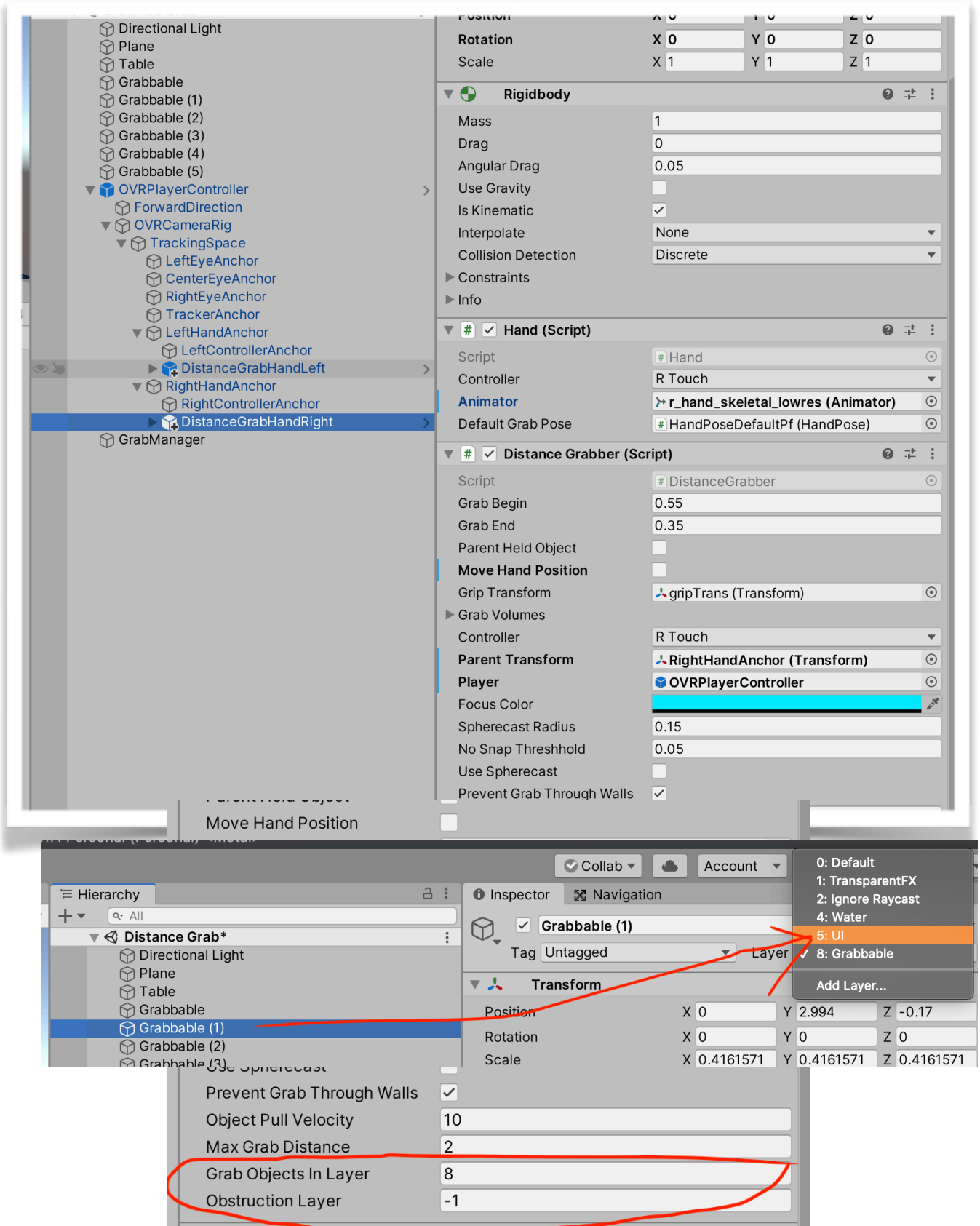
Step 3:

Find two prefabs DistanceGrabHandLeft and DistanceGrabHandRight and add them to the OVRPlayerController Hierarchy as the picture shows: Remember to reset the transform of them into 0.0.0



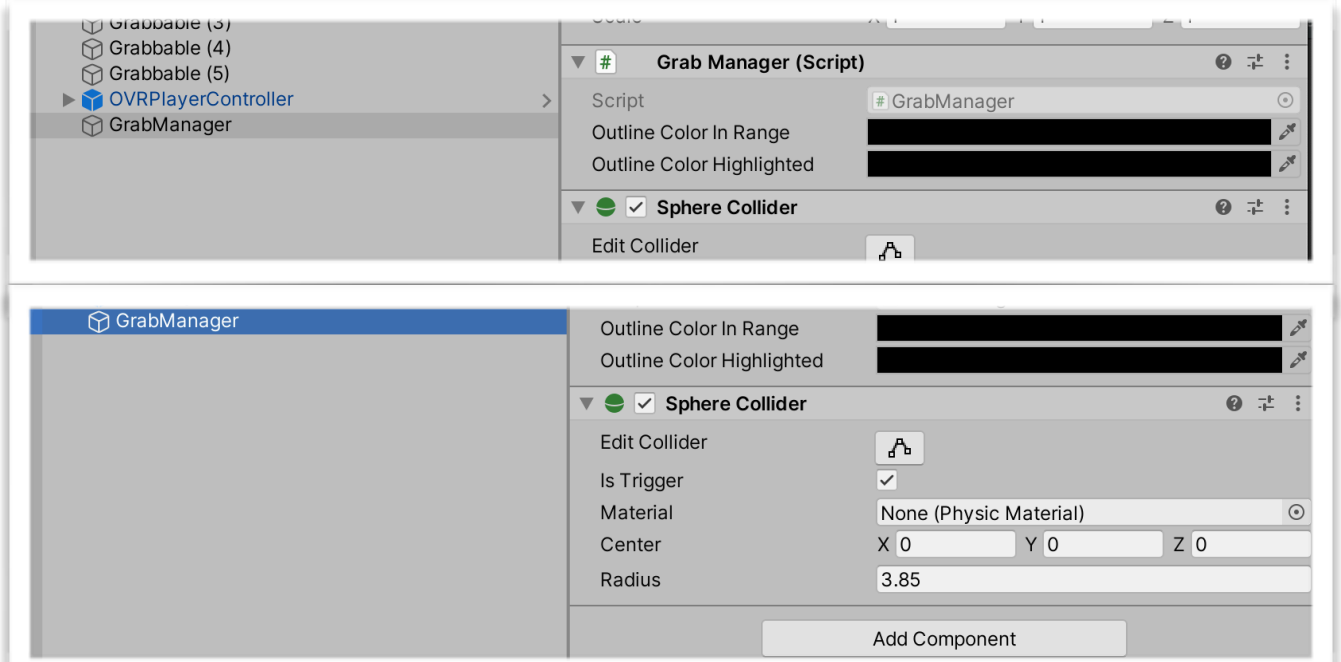
Step 4:

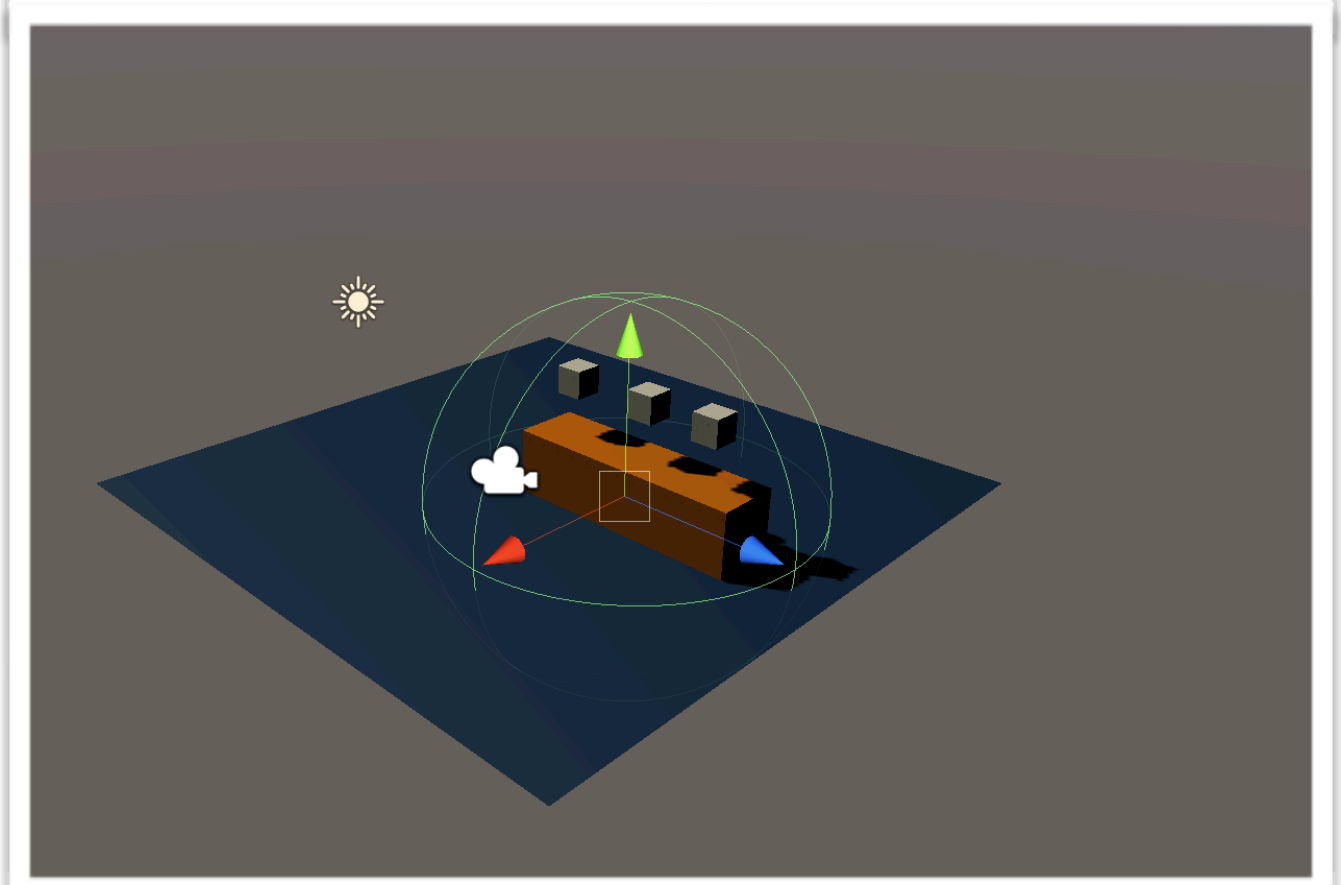
Add the player reference by dragging the OVRPlayerController to the reface field, and assign the parent tranform with the handAnchor. Assign the Grab Objects in Layer into the grabbable layer that you defined. and Obstruction Layer to the one you don't want to grab.



Step 5:

Finally, we need to create an empty object named GrabManager, and by add component, add the script GrabManager to it like the picture below. Then, add a sphere collider and adjust the radius to indicate the grabbable range. Check the IsTrigger Box.
Note: It's better to make the GrabManager object a child of OVRPlayerController and reset the transform to (0,0,0). So you don't need to adjust it manually.





Explanation:

GrabManger Game Object – Child object of OVRPlayerController that notifies grabbable objects when they are within grabbing range, at which point they become outlined in a color (or show a target) to indicate they are grabbable. The grabbing range itself is defined by a Sphere Collider component (as trigger) on this object.
DistanceGrabHandLeft and DistanceGrabHandRight Prefabs – Prefabs that contain the custom hands along with the necessary components and scripts to grab distant and nearby objects.
Grabbable Game Objects - Objects in the scene that have been configured to be grabbed from nearby or a distance.

DistanceGrabber(Script) [You can try some parameters to achieve something different!!]

This component enables the hands to grab distant and nearby objects that have been configured with a DistanceGrabbable component. This component is key to the main functionality of this sample scene, and its properties are as follows:

- **Grab Begin** and **Grab End** – These define the grip trigger threshold for grabbing and releasing objects. For **Grab Begin**, the closer the value is to 1, the more you must pull in the grip trigger to grab an object. For **Grab End**, the closer the value is to 0, the more you must release the trigger to drop or throw an object. **Grab Begin** must be a value higher than 0 or the user will be unable to grab objects. Always specify a value high enough to require finger pressure on the trigger, such as .2 or higher.
- **Parent Held Object** – Makes the grabbed object a child of the grabber. Enabling this works in this sample scene, but it can create issues in certain physics simulations.
- **Grip Transform** – An attached child transform that indicates where to snap grabbed objects to. This transform is also used to rank objects for grabbing when there are multiple options.
- **Grab Volumes** – Attached child collider objects that detect grabbable object candidates. These child collider objects must have Is Trigger enabled so they can successfully trigger grabbing events. This also prevents the colliders from being affected by physics.
- **Controller** – Indicates the controller that the grabber is tied to.
- **Parent Transform** – Transform of the parent object. You do not have to do anything with this property, but it can be used to attach the grabber to objects other than your Avatar.
- **Focus Color** – Color used to highlight targeted grabbable objects.
- **Spherecast Radius** – Sphere radius when using spherecasting to find target grabbable objects.
- **No Snap Threshhold** – Objects below this distance will not zoom to the hand, but will be grabbed as if nearby. In other words, objects below this are considered nearby.
- **Use Spherecast** – Enables use of spherecasting (rather than the default **Grab Volumes**) to find target grabbable objects.
- **Prevent Grab Through Walls** – Prevents distant and nearby objects from being grabbed through walls.
- **Object Pull Velocity** – Speed at which objects are pulled toward the hand.
- **Max Grab Distance** – Maximum distance from which distant objects could be grabbed. This should be at least equal to (or slightly greater than) the DetectGrabRangeSphere Collider's radius.
- **Grab Objects in Layer** – Only objects from this layer can be grabbed.
- **Obstruction Layer** – Layer that obstructs objects from being targeted and grabbed.
- **Player** – Indicates the OVRPlayerController that the hand is attached to.

DistanceGrabbable[Script]

This key component enables the object to be grabbed by Avatar hands that have been configured with an DistanceGrabber component. This component is key to the main functionality of this sample scene, and its properties are as follows:

- **Allow Offhand Grab** – When enabled, allows an object to be grabbed.
- **Snap Position** – When enabled, the grabbed object's position will snap to match the transform in **Snap Offset**.
- **Snap Orientation** - When enabled, the grabbed object's orientation (rotation) will snap to match the transform in **Snap Offset**.
- **Snap Offset** – A transform that is an offset relative to the DistanceGrabber that the grabbed object's position and/or orientation can snap to.
- **Grab Points** – When several objects are used to make one larger object, the collider for each object is placed in **Grab Point**'s elements, allowing the objects to behave as a single object.
- **Material Color Field** - This is the string that indicates the shader property for outline color to map to, enabling the highlighted outline effect for grabbable objects in range.

Grabbing Range and Targeting [Key parameters you can consider with]

An important implementation consideration for distance grabbing is the range at which objects can be grabbed. To change the grabbing range of your hands, you must change three values:

- The value of the Radius property on DetectGrabRange's Sphere Collider component.
- The values of the Max Grab Range properties on DistanceGrabHandLeft's and DistanceGrabHandRight's DistanceGrabber component.

The Radius property of DetectGrabRange's Sphere Collider creates an area around the OVRPlayerController that informs objects with a DistanceGrabbable component that they are in grabbing range and available as a target to objects with a DistanceGrabber component. In short, DetectGrabRange determines the range of what can be grabbed.

However, it's DistanceGrabHandLeft's' and DistanceGrabHandRight's' DistanceGrabber components that determine what is being actively targeted. The DistanceGrabbers' Max Grab Distances must at least match the Radius of the Sphere Collider. It's recommended that the Max Grab Distance be slightly larger than the radius to compensate for unpredictable real-world hand/body positioning.