In designing this experiment I decided the best way to go about this would be to have different states for it to go through.

- STATES
    - Preview state
        - The first state. Nothing happens. Participant has full visual. This will allow them to see the room they are in and get used to the headset, as well as wait for health and safety warnings Oculus displays on startup to vanish. When any key is pressed they will enter the first Blind state
    - Blind State
        - Participant is blinded. Text will appear on screen to inform them to look ahead and press any button when they are ready.
        - During this state the objects will be set in the scene.
        - When any button is pressed, the view will be set to straight ahead and will move to the Search State.
            - Sim currently CAN'T reset view.
            - NOTE: If they are not looking straight ahead when they leave the blind state, then wherever they are looking will be set to straight ahead. This will be disorienting if they are looking at an extreme angle such as at their feet. The requirement of a button press to indicate readiness over a timer is so they can tell the program when they are looking forward instead of having the program guess and possibly set it to a disorienting angle.
            - IDEA2: If resetting the angle fails to work, we can have the participant try and overlap two crosshairs (one locked to the center of their vision[+], and one locked to a position straight ahead of them[x]). Once the crosshairs overlap, the test will move to Search State with no button press required
            - IDEA3: If the previous is not desirable, the entire world can be reoriented around the Oculus. All measurements will need to have the global offset taken into account, but it is this programmer's assumption that the software will only need to subtract the difference before recording the angles to the output file.
            - See "Notes on Oculus Movement" Below.
    - Search State
        - The phase where the participant will search for the object. Every frame the orientation of the Oculus will be recorded to file. (This is the only state where anything is recorded.) The state ends when the participant presses a button while looking at the object, and enters the lock state.
    - Lock State
        - For a brief moment the user has free view to get a better look at what they found instead of having it swept away the moment they glimpse it. Nothing else happens until they enter the blind state again.

Notes on Oculus Movement:

The code *OVRDevice.ResetOrientation(0)* was found online as a way to reset the direction the Oculus is facing, but only in a web forum 2 years old. All attempts to implement it have failed, as nothing named "OVRDevice" exists in the code package that was downloaded from the Oculus website and imported to this project. (Though many other "OVR-" names exist.) It is assumed this code is outdated. There must be something newer, but searches so far have been fruitless.

I've discovered that my previous assumption that only the OVRCameraRig can send data to the Oculus Rift to be false. Not only can ANY camera in the scene send video to the Oculus, but each camera can be easily set to send video to both eyes, or only the right or left specifically. Even so, the position and rotation of the cameras are in complete control of the Oculus Rift and cannot be modified once the simulation is running. (Any attempts will be overwritten instantly by Oculus input.) However any object it is parented to can still be moved and rotated, offsetting the location and rotation of the camera.

I think it is safe to conclude that any attempt to reset the oculus camera to look perfectly forward in a scene to be impossible without great headache, unless something I haven't found already exists in the OVR code.

Other Updates:

- Sim failed to lock at 60FPS. Also failed to lock at any lower framerate, (Ex: 10FPS) preferring to try for 60 instead. Either the code I'm using to lock it, or the code I'm using to test it, do not seem to be working. EDIT: Found potential issue, will test later
- Upgraded to Unity 5 in order to get extra features that might be needed.
- Discovered reason sim was freezing when exported to full screen (OR windowed regardless) may have been because another program (the Unity Editor) was already using the Oculus.