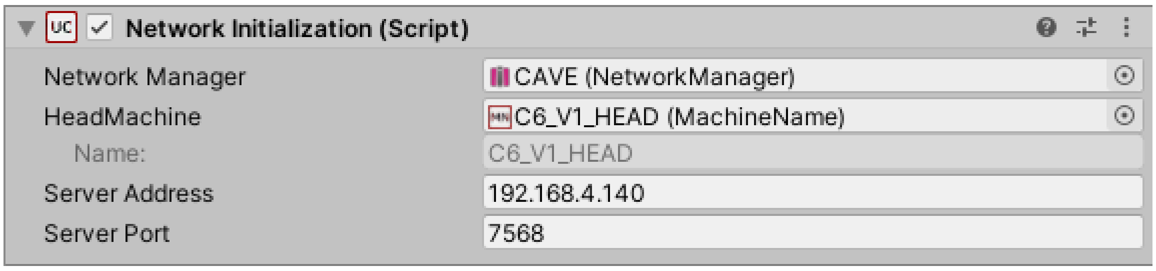
Network Initialization



Network Manager	CAVE (NetworkManager)
HeadMachine	C6_V1_HEAD (MachineName)
Name:	C6_V1_HEAD
Server Address	192.168.4.140
Server Port	7568

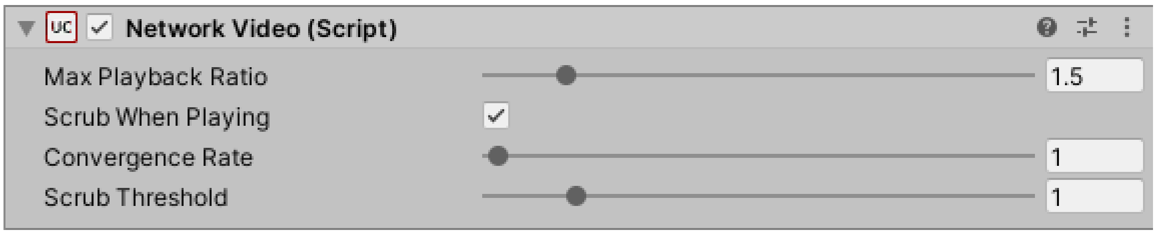
This script automatically starts a server or tries to connect to the server, depending on whether the machine Unity is running on matches the Head Machine option. There are script variables for the server IP (which should be the LAN IP of the head node) and port, both of which can be overridden with command line options:

Command Line Options:

Parameter	Description	Example
serverAddress	The IP address the server runs on. Child nodes will attempt to connect to this address.	`192.168.1.115`
serverPort	Server port.	`7568`
forceClient	Whether or not to force this instance to run as a client, regardless of machine name (useful for running client and server instances on head machine).	`1`

This script needs a reference to a `NetworkManager`, which is the Unity Multiplayer HLAPI package's built in method of managing client server connections. Make sure the `NetworkManager` script is enabled to allow enough connections; under Network Info, Enable *Advanced Configuration* And Ensure *Max Connections* is enough to support all your child nodes.

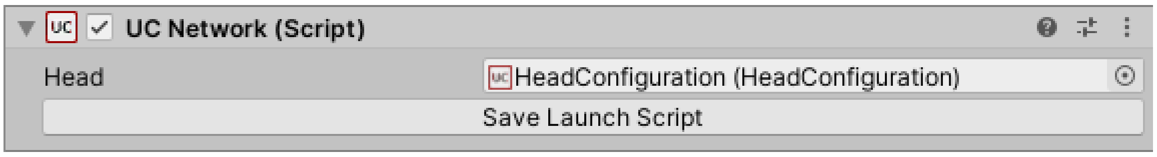
Network Video


Network Video (Script)

Max Playback Ratio	1.5
Scrub When Playing	✓
Convergence Rate	1
Scrub Threshold	1

The `NetworkVideo` script is used to synchronize video playback across multiple machines.

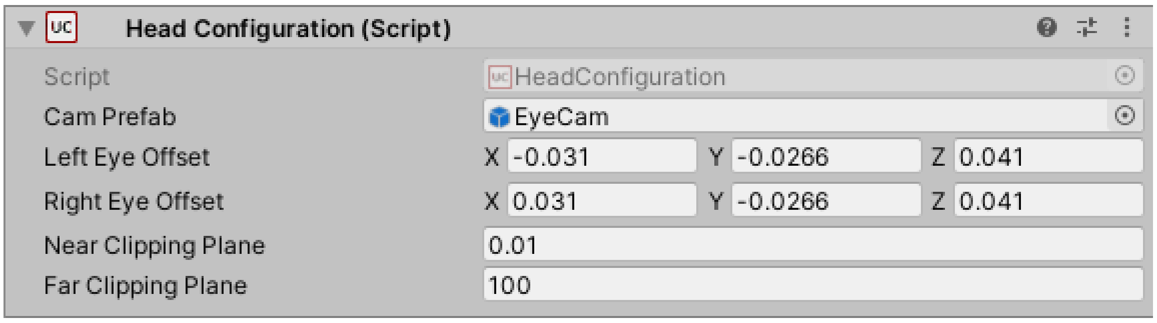
UC Network


UC Network (Script)

Head	HeadConfiguration (HeadConfiguration)

The `UCNetwork` script should be attached to the root GameObject that represents your CAVE setup. You should only need one `UCNetwork` script per scene. This script only runs on the head node machine, and is responsible for sending update information to the child node machines. `UCNetwork` needs a reference to a `HeadConfiguration` script, so that the `HeadConfiguration` GameObject can be updated by the server. The GameObject with the `UCNetwork` script attached can be moved and reoriented throughout the Unity scene, so make sure it is the parent of your `PhysicalDisplay` and `HeadConfiguration` GameObjects.

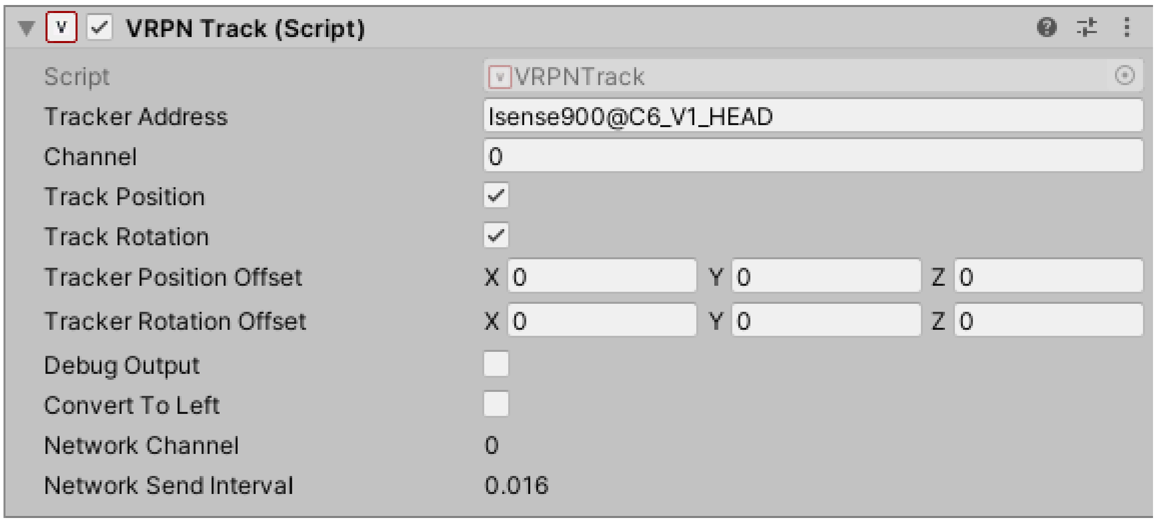
Head Configuration


Head Configuration (Script)

Script	HeadConfiguration
Cam Prefab	EyeCam
Left Eye Offset	X -0.031 Y -0.0266 Z 0.041
Right Eye Offset	X 0.031 Y -0.0266 Z 0.041
Near Clipping Plane	0.01
Far Clipping Plane	100

The `HeadConfiguration` script contains eye separation and camera parameters. You should only need one `HeadConfiguration` script per scene.

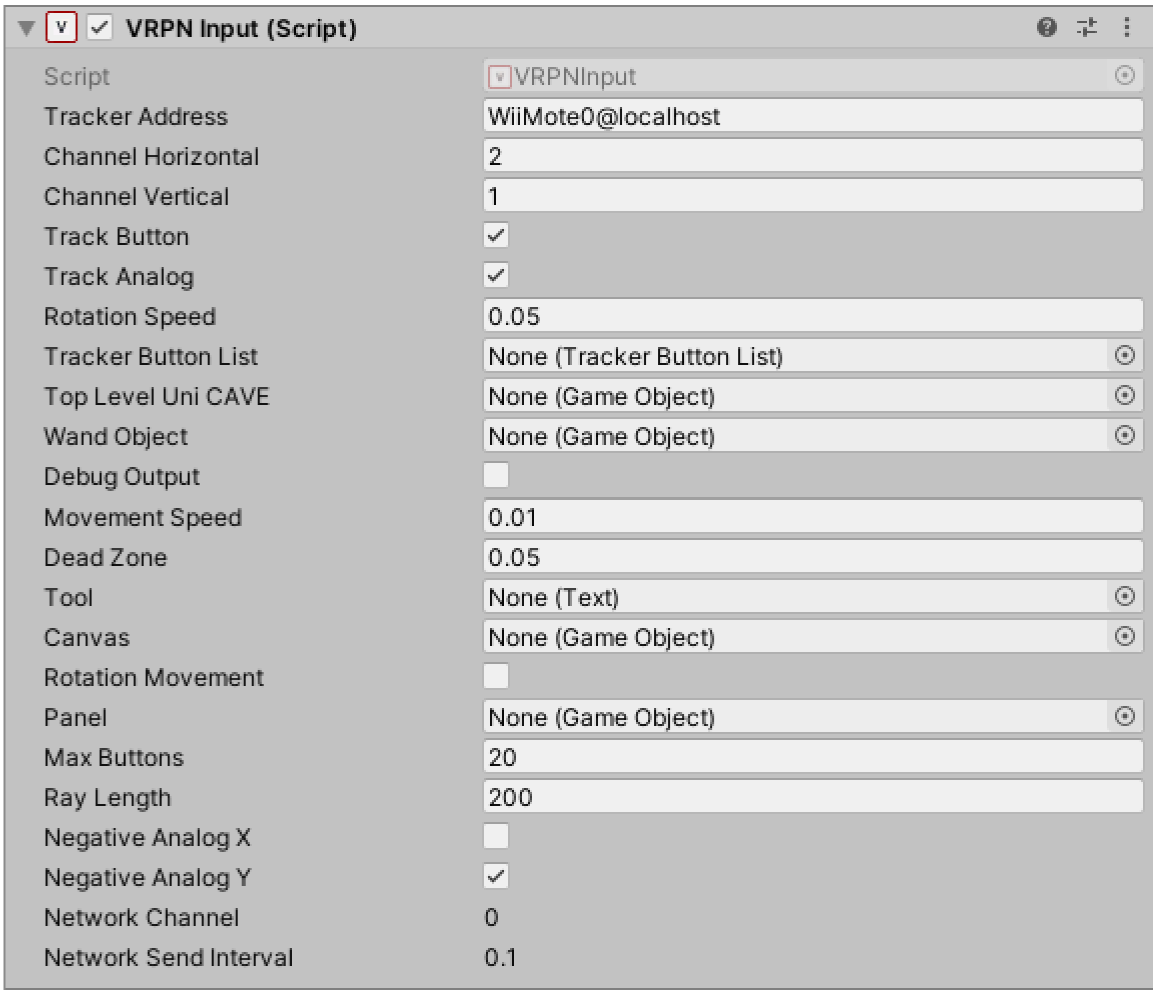
VRPN Track


VRPN Track (Script)

Script	VRPNTrack
Tracker Address	Isense900@C6_V1_HEAD
Channel	0
Track Position	✓
Track Rotation	✓
Tracker Position Offset	X 0 Y 0 Z 0
Tracker Rotation Offset	X 0 Y 0 Z 0
Debug Output	
Convert To Left	
Network Channel	0
Network Send Interval	0.016

Whatever GameObject the `VRPNTrack` script is attached to will be transformed to have the position and orientation of the real life tracker. For example, if you have a VRPN tracker attached to a user's head or glasses, you can put attach a `VRPNTrack` script to your head object.

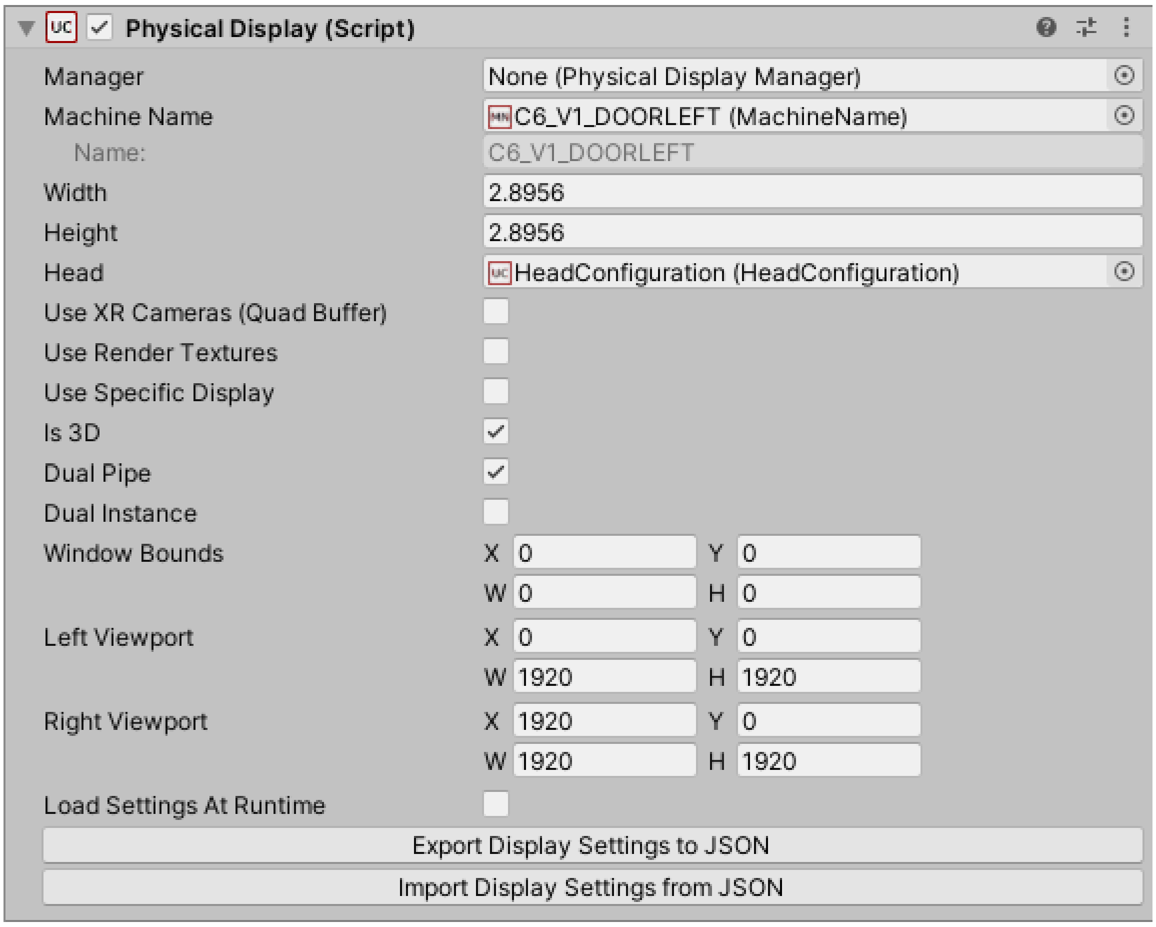
VRPN Input


VRPN Input (Script)

Script	VRPNInput
Tracker Address	WiiMote0@localhost
Channel Horizontal	2
Channel Vertical	1
Track Button	✓
Track Analog	✓
Rotation Speed	0.05
Tracker Button List	None (Tracker Button List)
Top Level Uni CAVE	None (Game Object)
Wand Object	None (Game Object)
Debug Output	
Movement Speed	0.01
Dead Zone	0.05
Tool	None (Text)
Canvas	None (Game Object)
Rotation Movement	
Panel	None (Game Object)
Max Buttons	20
Ray Length	200
Negative Analog X	
Negative Analog Y	✓
Network Channel	0
Network Send Interval	0.1

The `VRPNInput` script is used to receive input from a tracked controller or other device.

Physical Display



Physical Display (Script)

Manager	None (Physical Display Manager)
Machine Name	C6_V1_DOORLEFT (MachineName)
Name:	C6_V1_DOORLEFT
Width	2.8956
Height	2.8956
Head	HeadConfiguration (HeadConfiguration)
Use XR Cameras (Quad Buffer)	☐
Use Render Textures	☐
Use Specific Display	☐
Is 3D	✓
Dual Pipe	✓
Dual Instance	☐
Window Bounds	X 0 Y 0 W 0 H 0
Left Viewport	X 0 Y 0 W 1920 H 1920
Right Viewport	X 1920 Y 0 W 1920 H 1920
Load Settings At Runtime	☐

Export Display Settings to JSON
Import Display Settings from JSON

For each real display in your CAVE setup, there should be one GameObject with a `PhysicalDisplay` script attached. The `PhysicalDisplay` script is designed to be extremely flexible so that it can be used on many types of CAVE setups. The `PhysicalDisplay` script contains most of the information about a display as it exists in the real world. Below is a labeled example of a `PhysicalDisplay`.

The main function of a `PhysicalDisplay` is to render perform the calculations necessary so that the display in the real world appears as it should. This is accomplished by changing the projection matrix of the cameras on the `HeadConfiguration`. However, another important function is automatically positioning the window and camera viewports on the screen, so they appear as they should. For example in a dual pipe setup, the left and right eye must appear on different areas of the display for stereo 3D to work correctly; `PhysicalDisplay` takes care of this automatically (and is configured with the *Window Rect*, *Left Viewport*, and *Right Viewport* options).

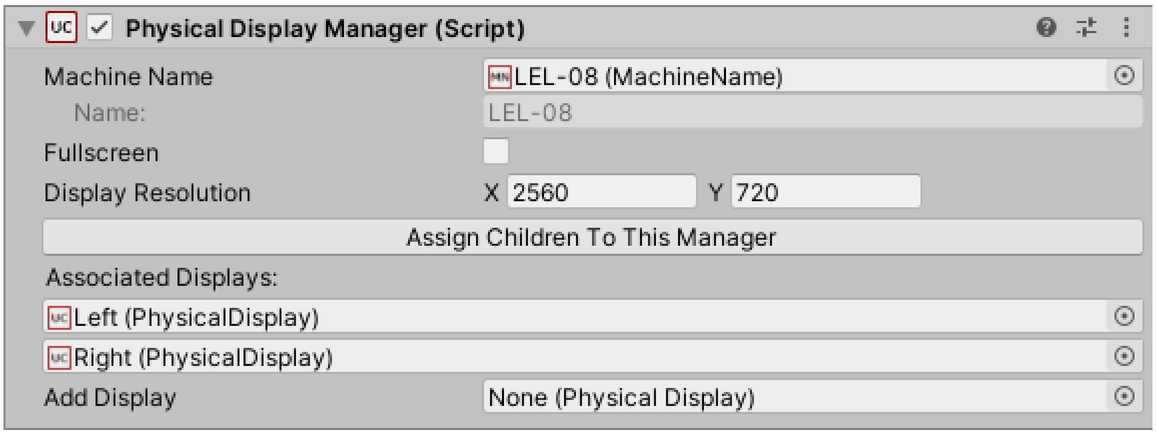
Parameter	Description
Manager	Reference to physical display manager; used for running multiple physical displays in one instance of Unity.
Machine Name	Which machine to run this physical display on.
Activate All Displays	Enable exclusive fullscreen on multiple displays.*
Width	The width of the display in real life (in meters).
Height	The height of the display in real life (in meters).
Head	Reference to the HeadConfiguration script used in the CAVE setup.

Parameter	Description
Use XR Cameras	When rendering in active stereo mode, use Unity's default XR cameras.**
Use Render Textures	Render to textures instead of screen, which can be used for post processing effects.
Use Specific Display	When checked, allows a specific display number to be chosen.
Is 3D	Is this CAVE setup outputting any type of stereo 3D image?
Dual Pipe	Is this program a dual pipe stereo setup (as opposed to a active stereo setup)?
Dual Instance	(If Dual Pipe Enabled) Should each eye render in a separate instance of Unity.
Window Bounds	The bounds where the window should exist on the screen.
Left Viewport	*(If Dual Pipe Enabled)* The bounds where the left eye should render on the screen.
Right Viewport	*(If Dual Pipe Enabled)* The bounds where the right eye should render on the screen.

**As of 11-2-2018, does not work with OpenGL*

***Requires making changes to Unity player settings*

Physical Display Manager


Physical Display Manager (Script)

Machine Name	LEL-08 (MachineName)
Name:	LEL-08
Fullscreen	☐
Display Resolution	X 2560 Y 720

Assign Children To This Manager

Associated Displays:

Left (PhysicalDisplay)

Right (PhysicalDisplay)

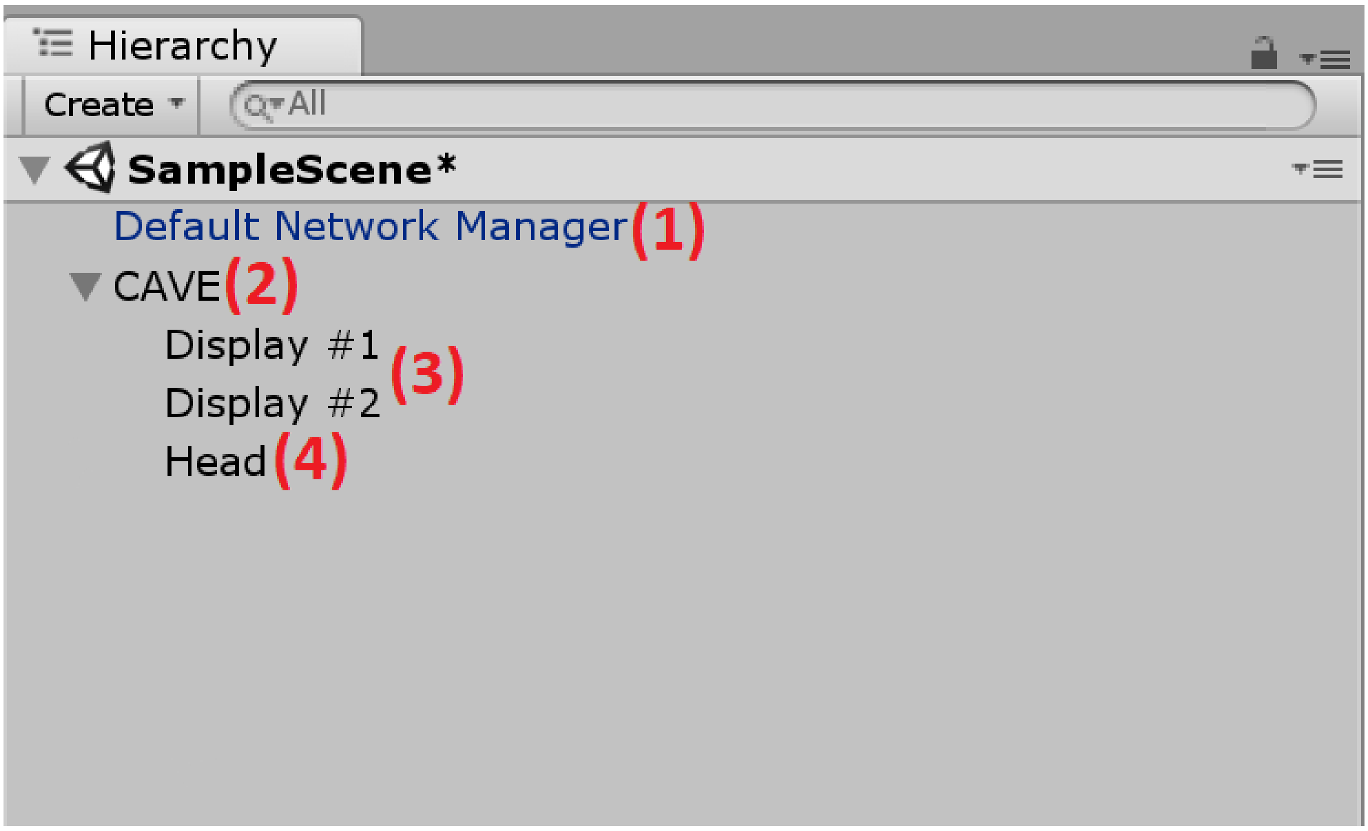
Add Display	None (Physical Display)

The `PhysicalDisplayManager` script allows multiple `PhysicalDisplay` objects to exist within one instance of Unity. For example, if your CAVE setup consists of three displays connected to one machine, your Unity scene would have three `PhysicalDisplay` objects. Ordinarily one would need to launch three instances of Unity to manage these three displays (this technique can be used for Multi-GPU setups to run each instance of Unity on a different GPU). If these `PhysicalDisplay` objects are attached to a `PhysicalDisplayManager`, only one instance of Unity can run which will control all three displays.

A good example implementation of `PhysicalDisplayManager` can be found in the **DualMonitorOneMachine** prefab under *Prefabs/Basic*.

Basic Custom Setup

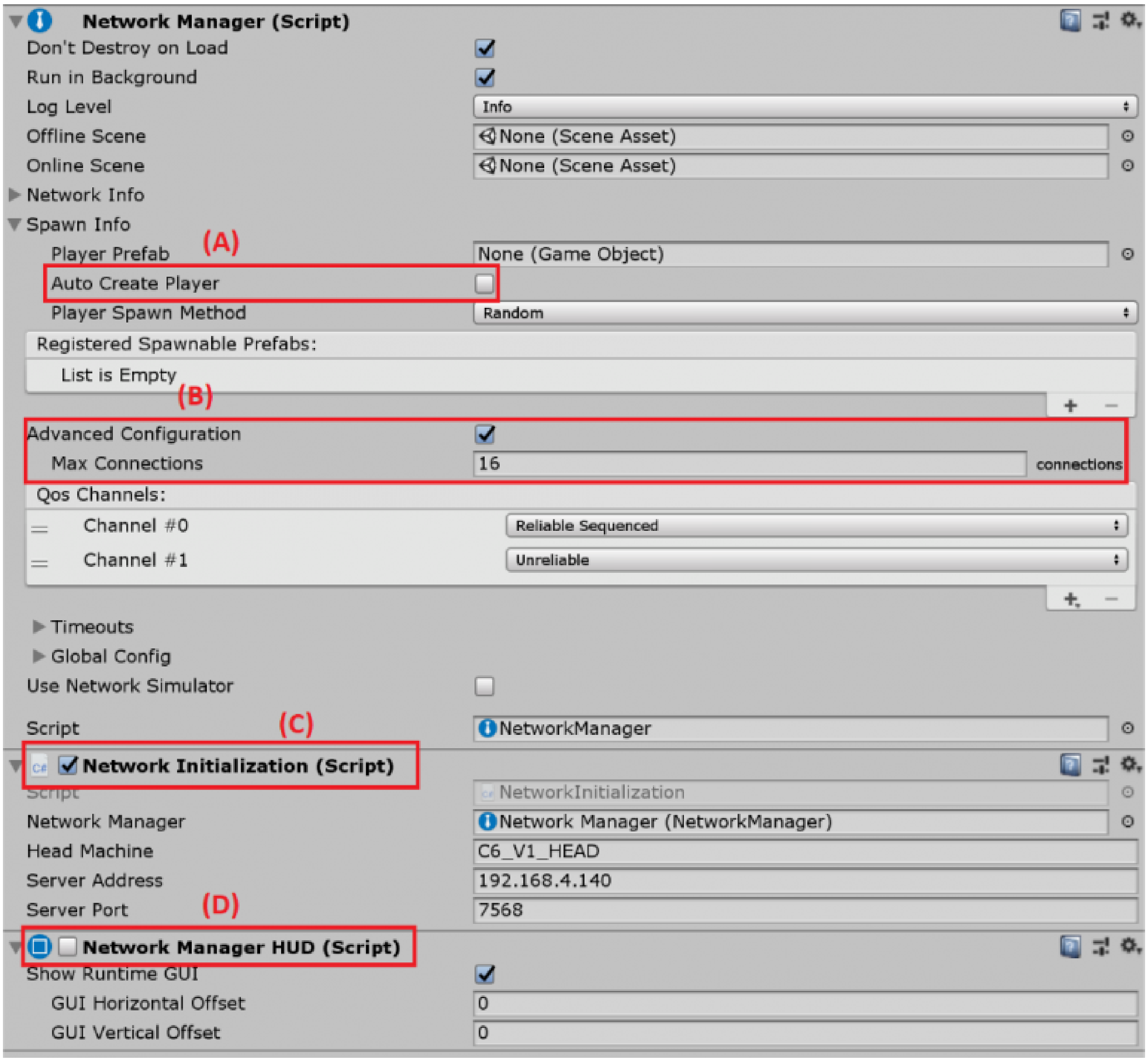
A basic setup consists of four parts:



1. A GameObject with Network Manager script and Network Initialization script (see below)
2. A GameObject with UCNetwork script attached
3. An assortment of GameObjects with PhysicalDisplay scripts attached
4. A GameObject with HeadConfiguration script attached (And often a VRPNTrack script)

Parts **2-4** involve simply configuring the respective scripts as described in their sections, but Part **1** involves a few more scripts and configuration.

Object **1** in the Basic Setup:



Network Manager (Script)	
Don't Destroy on Load	✓
Run in Background	✓
Log Level	Info
Offline Scene	None (Scene Asset)
Online Scene	None (Scene Asset)
Network Info	
Spawn Info	
Player Prefab	None (Game Object)
Auto Create Player	☐
Player Spawn Method	Random
Registered Spawnable Prefabs:	
List is Empty	
Advanced Configuration	✓
Max Connections	16 connections
Qos Channels:	
Channel #0	Reliable Sequenced
Channel #1	Unreliable
Timeouts	
Global Config	
Use Network Simulator	☐
Script	NetworkManager
Network Initialization (Script)	✓
Script	NetworkInitialization
Network Manager	Network Manager (NetworkManager)
Head Machine	C6_V1_HEAD
Server Address	192.168.4.140
Server Port	7568
Network Manager HUD (Script)	☐
Show Runtime GUI	✓
GUI Horizontal Offset	0
GUI Vertical Offset	0

A. Disable “Auto Create Player” on the Network Manager Script

B. Enable “Advanced Configuration”, ensure that “Max Connections” is enough for your setup

C. Configure your Network Initialization script as described

D. For debugging, it is useful to use a `NetworkManagerHUD` script to manually start a server or client, which is normally done by the `NetworkInitialization` script based on the machine name Unity is currently running on.

Additional Command Line Options:

Parameter	Description	Valid Options	Example
`overrideMachineName`	Launch your application as if it was running with a certain machine name	Any string	`overrideMachineName MY_HEAD_MACHINE`

Parameter	**Description**	**Valid Options**	**Example**
`appendMachineName`	Launch your application as if it was running with a string appended to its machine name (useful if you have two instances of Unity running on the same machine but want a different display for each of them)	Any string	`appendMachineName _left`
`eye`	Which eye a certain instance of your application should render (useful if you have one instance of Unity running per eye)	`left`, `center`, `right`	`eye right`
`serverAddress`	(Valid only for NetworkInitialization scripts) What network address the server should start on or connect to	Any IPV4	`serverAddress 192.168.5.8`
`serverPort`	What network address the server should start on or connect to (Valid only for `NetworkInitialization` scripts)	Any integer ∊[1,65535]	`serverPort 1234`
`-screen-fullscreen`	Whether to launch the application in exclusive fullscreen mode	`0`, `1`	`-screen-fullscreen 0`
`-popupWindow`	Whether to launch the application without borders	None	`-popupWindow`
`-show-screen-selector`	Whether to launch the screen selector window (enabled by default, can be disabled in *Edit->Project Settings->Player->Resolution and Presentation->Display Resolution Dialog*)	None	`-show-screen-selector`
`-screen-width`	Width of the window (may be overridden at runtime by `PhysicalDisplay`)	Positive, nonzero integers	`-screen-width 1280`
`-screen-height`	Height of the window (may be overridden at runtime by `PhysicalDisplay`)	Positive, nonzero integers	`-screen-width 720`

Additional Player Settings – Enabling Active Stereo

To enable active stereo setups in Unity: go to *Edit->Project Settings->Player* and enable *Virtual Reality Supported*, add *Stereo Display (non head-mounted)* to the list of SDKs, and remove the other SDKs.