* Assets\udp\_camera\_stream.py  
  streams the camera over udp  
  You can change the Quality and Size of the JPG you are sending to test tradeoffs on packet size
* Assets\RobotController.cs  
  Script used by Odile robots, it lets you control with Keyboard or F710 Logitec Gamepad, also has a gui to change controller button mapping
* Assets\PoseControllerV2.cs  
  this manager receives the 34 body landmarks detected by the depthai camera and positions a sphere at each landmark. it centers the pose to the vertical axis of the character controller
* Assets\UDPCameraViewer.cs  
  receives camera feed on udp socket and shows it on a RawImage, also sends camera angles via udp using key protocol
* Assets\\_Game\\_Scripts\Player\PlayerControllerVR.cs  
  VR version of the demo player controller, best effort to have some working controls, almost made Fede throw up after 5 minutes of testing
* Assets\\_Game\\_Scripts\Misc\NetworkButtons.cs  
  gui to start host, server or client, has public booleans to skip it and start host or client directly
* Assets\udp\_camera\_receive.py  
  show incoming udp video stream
* ASSETS
* TODO properly merge assets from youtube tutorial
* Assets/Scenes
* OdileOnline - multiplayer videogame with simulated robot and human pose tracking
* Odile - Offline simulator of odile
* FPVRobot - first person robot control, shows an image always facing the vr that catches the incoming video stream, also sends via udp the tracked vr angle
* Level - very first offline demo of human robot interaction
* Assets/\_Game/Scenes/Main.unity - first multiplayer game demo with robot and human
* Assets/\_Game/Prefabs
* PlayerSpawner - differentiates prefabs between players (unity netcode wants the same prefab for all players (maybe))
* HumanNetwork - character for multiplayer without pose tracking (Assets/\_Game/Scenes/Main.unity)
* RobotNetwork - same for robot
* PlayerSpawnerOdile - differentiates prefabs between players, also changes robot visualization client side
* HumanPoseComplete - character for OdileOnline with network functionalities, listens udp for poses, each sphere has a Client Network Transform syncing their position
* OdileOnline - odile with network functionalities, each joint also has a Client Network Transform syncing the rotation
* GamepadControlsOnline - Scriptable Objects with gamepad controls for odile online
* Assets/OdileNetworkPrefabs.asset - probably wrong, check this out
* Assets/\_Game/Prefabs/PlayerSO.asset - SO for initial multiplayer demo
* HOW TO MULTIPLAYER
* Put a network manager in the scene, give it a player prefab (spawner) and put in network prefabs lists all the objects you will sync over network
* To network prefabs add Network Object script and Client Network Transform (also to children) to sync transform
* extension ParallelSync opens a second unity that stays syncronized
* ROBOTS
* Odile
* To control odile start main setting camera and/or controller to 0 or 1
* if camera doesn’t work unplug it and plug it back in
* DepthAI camera
* Run DepthAICamera\_standalone.py on Jetson Nano with camera plugged