Discovery of Devices

Practice objectives:

1. Discover new technologies (Full Body Tracking , Augmented Reality,…).
2. Give Idea for your future project
3. Set these new devices in a Unity Project
4. Example of Unity integration

Since you have to spend 20 minutes for each module and you are four per group, each people have 5 minutes to test one module. This practical work is here just to give you a "flavor" of each device.

Station 1: Full Body Interaction (Kinect 2)

Prepare: Open Unity 5.5.0f3 and open the first on the Unity Wizard list. A window will open and tell you the version is different; you just click continue.

Goal: Real time character animation is not a trivial subject. It contains several steps (modeling, rigging and skinning) and may follow different animation approaches (forward/inverse kinematics). It also involves a tradeoff between latency and quality. In this TP we are going to use a low cost approach, with a Kinect 2 and forward kinematics.

Objective 1: Go through the relevant source code and test the game.

KinectModelControllerV2.cs

Avatar animation: It uses the joint position information provided by Kinect to set the avatar pose (joints orientation). Have a look on the RotateJoint function, called for each joint.

KinectModelPosition.cs

Set the avatar position. It will be written by the students in Objective 3, currently it only contains the variables the students might need.

Tasks:

a. Run the program, and observe the degree of control you have over the avatar (i.e. only the noisy control of the avatar joints)
b. Observe the debugging information drawn in the scene, they inform the RGB image obtained by Kinect, the depth + RGB drawn as a mesh, and the position of the joints of the tracked player. (disable the KinectDebug game object afterwards as it consumes resourses)

Objective 2: Play the multiplayer game!

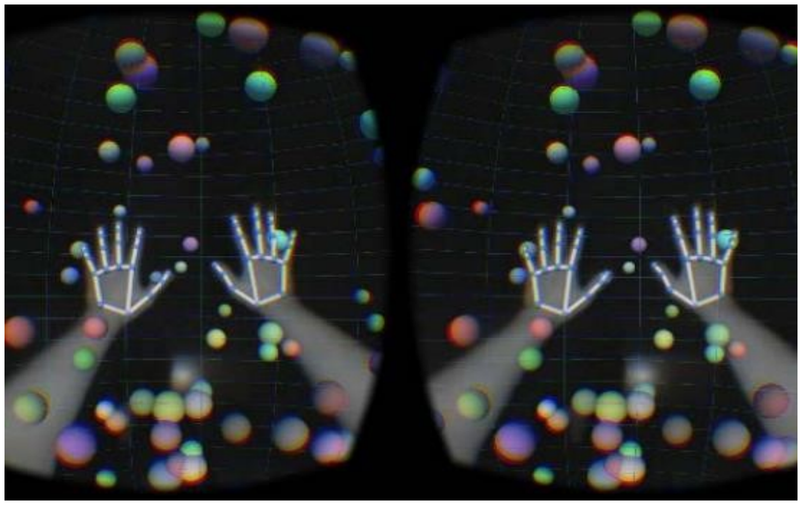
Tired of playing alone? Deactivate the GameObject “SinglePlayer” and activate the GameObject “Multiplayer”

a. You will have to add an **offset** along the X axis so that the players don`t hit each other while sharing the Kinect tracking space
b. Make it a fair game by repositioning the canon so that the balls are thrown from above

Known issue: sometimes the same user gets 2 player IDs and end up controlling both avatars. It this happens, restart the game.

Station 2: Bring Hands into VR (Leap Motion)

Prepare: Open Unity 5.5.2f1 and open the first on the Unity Wizard list. A window will open and tell you the version is different; you just click continue.



Goal: Experiment VR hand tracking and head mounted display with budget equipment. Controlling all the degrees of freedom of a virtual hand is far from trivial, leap motion accomplishes that that with stereo geometry (infrared projection + infrared cameras).

Objective 1: Play the “Jenga” game in VR!

In this task we are going to test the capabilities of the Leap motion, and learn how to set an Oculus VR CV1 camera in a Unity scene. We will also learn how to mount the Leap motion in the Oculus VR CV1.

Open the scene **Jenga**

a. Run the scene (by pressing the “play” button) with the **Leap motion** over the table. Try to remove the pieces without ruining the pile
    a. Quite impossible hum? Do you see why?

b. Set the friction parameters of the **JengaPhysicsMaterial** asset to 0.15, and duplicate the scale of the **Jenga** GameObject (from 0.1 to 0.2) *.
c. Set the fixed timestep to 0.002 (go to edit->project settings->time)
d. Set the default contact offset to 0.0001 (go to edit->project settings->physics)
e. Play the scene again, now it might be possible to make 1 or 2 moves, even though the new physical drag values make it seems quite icy.
f. Now disable the “table” GameObject and enable the “Oculus” GameObject
g. Check the Virtual Reality supported option (go to edit->project settings->player->other settings)
h. Finally, mount the Leap Motion in the Oculus and play the Jenga game in a VR setup!

Objective 2: Setup an augmented reality scene using infrared cameras overlay

The Leap motion has two infrared cameras, which allow for a sort of monochromatic augmented reality. In this task we are going to explore this possibility.

a. Open the scene **infraredCameraOverlay**
b. Press play, the infrared images are shown, and virtual hands are drawn over your own hands.

What about interacting with virtual objects over the table in front of you? To do so you need to create an invisible surface collocated with the real table.

a. Create a new GameObject – GameObject->3D Object->Cube
b. Iteratively: Move the cube down and test the scene, until it feels aligned with the real table (while moving the head, the motion parallax – which is an important depth cue – will make the table and cube appear to be located at the same space, i.e. they will not move relative to each other).
c. Disable the **renderer** Component of the cube GameObject
d. Drag the **Jenga** prefab to the scene and hit play.

Now that you see virtual objects over your table, why don’t you expand the scene? What about having a virtual object collocated with the monitor, and other objects of the real environment so that the Jenga pieces can interact with them?

Extra

Load the **Joggling** scene and test your joggling skills in VR. The scene has only one ball, you can duplicate it as much as you want (you can also cheat by lowering the gravity force under the **edit->project settings->physics** menu).

* Objective 2-e.a- besides the dubious and noisy tracking the Leap motion may provide at times, there are two other reasons that make playing the game almost impossible: the collision is not computed with the mesh you see rendered, usually simpler primitives are used to compute collisions to reduce its impact in performance, if you run on the editor you can select the collision GameObject to see the collision mesh; there is a strong friction between the pieces, this can be addressed by reducing both friction parameters in the **JengaPhysicsMaterial** asset.

Station 3: Walk in VR (Katwalk VR)

Prepare: Open Unity 5.5.2f1 and open the first on the Unity Wizard list. A window will open and tell you the version is different; you just click continue.

Goal: Experiment treadmill in VR with head mounted display and a consumer tracking system. Locomotion is a real challenge in VR. Indeed, you might have to travel a whole world in your living room. If the user uses the Joystick to move it might have cyber sickness. You can teleport yourself but you will be less immersed.

Objective: Play in VR

This is a semester project made by a previous master student. The goal was to implement a new way to move in VR thanks to the treadmill. The Vive Trackers have been used to track the user feet and compute the user acceleration in order to give a realistic speed to the avatar.

a. **Respect the order for turn on the devices!** Turn on first the left controller then the right. The torso on the KatwalkVR has to be the third. Finally turn on the Vive tracker for the left feet and end with the right feet.
b. One helps the other to set the KatwalkVR (2 persons at least). First, attach the Vive trackers around the ankles, then grab the safety bar to enter in the KatwalkVR. Then, attach the belt around your belly and your legs. Finally, put the headset and grab the controllers from your comrade.
c. Go to “Level” Folder then “launcher” and finally select the scene called “launcher”.
d. Hit play and then “C” to calibrate the feet of the avatar. **Note:** You can read the README.txt if you want more commands.

There are 4 different levels you can try. Each level exploits one or several features exclusives to the KatwalkVR. The 1st level is challenging the precision of the speed and the movement during walking.

The second level lets you jump for real. The third lets you jump, walk and crouch. Finally, the last level lets you slide. All these levels are demonstrating new kind of locomotion you might want to use in the future.

Station 4: Augment Reality (HoloLens V1)

Prepare: Open Unity 2018.3.8, and load the project *HoloLens* (find it in the Unity Hub launcher).

Goal: Experiment with the elements of augmented reality in HoloLens. The additional information that can be overlaid on your real-world view can be informative, but also distracting. The input methods, especially with this model HoloLens, are very limited and quickly can become frustrating, even if they show future promise.

Objective: Experience Basic AR

All of the assets you need have already been loaded into this project. You can now test different scenes by loading them, connecting remotely to the HoloLens, and then using emulation mode to play them in real time.

a. Turn on the HoloLens, go into the apps, and start the *Holographic Remoting Player* app.
b. You will see an IP address in the main window, take note of this address.
c. In Unity, go to example scenes, UX, and choose one of the categories. Load a scene that you'd like to try (ideally one that involves interactivity).
d. In Unity, go to Window>XR>Holographic Emulation.
e. In the popup window, set *Emulation Mode* to *Remote to Device*.
f. Enter the HoloLens IP address in the field *Remote Machine*, click connect.
g. Once the device is connected (green), click play in Unity. Do not close this popup window.
h. You should now see the game playing in the HoloLens.
i. Try one or two scenes to experiment with different interaction and grasping methods. (Note, you may need to "disconnect" the HoloLens to get it to "forget" your previous emulation when changing scenes).

If you have any questions during the session, do not hesitate to ask the TAs!
Good Luck!