How to Run the Example

1. Download and install Kinect SDK as described in the next section.
2. Open scene 'KinectAvatarsDemo', located in Assets/AvatarsDemo-folder.
3. Run the scene. Move your body and see how the avatars and the cube-man reflect your movements. Both avatars are automatically connected to the 1st player in the example scene.
4. Use your left or right hand to control the hand-cursor on the screen.
5. Try one or more of the suggested gestures and make sure they are detected correctly.
6. Stop the scene. Enable 'Compute User Map' and 'Display User Map'-parameters of KinectManager – a component of 'MainCamera' in the example scene. Then run the scene again.
7. Open and run 'KinectGesturesDemo'-scene, located in Assets/GesturesDemo-folder. Use hand swipes – left and right - to control the presentation cube.
8. Open and run 'KinectOverlayDemo'-scene, located in Assets/OverlayDemo-folder. See how the green ball follows the position of your right hand on the video wall.

Installation of Kinect Sensor with MS SDK

1. Download the Kinect SDK or Kinect Windows Runtime. Here is the download page: http://www.microsoft.com/en-us/kinectforwindowsdev/Downloads.aspx
2. Run the installer. Installation of Kinect SDK/Runtime is simple and straightforward.
3. Connect the Kinect sensor. The needed drivers will be installed automatically.

Why There Are Two Avatars in the Scene

The meaning of the 2 avatars (3D humanoid models) in the scene is to show that you can have both – mirrored and non-mirrored movements.

First, you can have an avatar that mirrors your movement. This is the one facing you in the example scene. This avatar is parented to an empty game object called 'UCharCtrlFront', which has a Y-rotation set to 180 degrees. Also, the AvatarController, attached to the avatar's game object 'U_CharacterFront' has 'Mirrored Movement'-parameter ticked. Mirroring means that when you, for instance, lift your left hand the avatar will lift his right hand and vice versa, like in a mirror.

The second avatar that has his back turned to you is not mirroring, but rather exactly reproduces your movements. Your left is its left and your right is its right. See it that way - you're also staying with your back to the scene's camera. Its control object is called 'UCharCtrlBack'. It has Y-rotation 0 and the 'Mirrored Movement'-parameter of its game object 'U_CharacterBack' is not ticked.

In order to get correct avatar positions and movements in your scene, always create an empty control object and parent your avatar to this object. Set the Y-rotation of the control-object as described above.

Then you can move the avatar around your scene by moving the Ctrl-object, leaving the avatar's local position and rotation set to 0. Pay also attention to 'Mirrored Movement'-parameter of the AvatarController, attached to your avatar's game object.

How to Reuse the Kinect-Example in Your Own Unity Project

1. Copy folder 'KinectScripts' from Assets-folder of the example to the Assets-folder of your project. This folder contains the needed scripts and optional filters.
2. Wait until Unity detects and compiles the new Kinect scripts.
3. Add 'AvatarController'-script to each avatar (humanoid character) in your game that you need to control with the Kinect-sensor.
4. Drag and drop the appropriate bones of the avatar's skeleton from Hierarchy to the appropriate joint-variables (Transforms) of 'AvatarController'-script in the Inspector.
5. Uncheck 'Mirrored Movement', if the avatar should move in the same direction as the user. Check it, if the avatar should mirror user movements
6. Add 'KinectManager'-script to the MainCamera. If you use multiple cameras, create an empty GameObject and add the script to it. KinectManager's Start()-method initializes Kinect sensor, Update()-method updates positions of all Kinect-controlled avatars.
7. Optional: Drag and drop the avatars from Hierarchy to the 'Player 1 Avatars' list.
8. If you need a 2nd Kinect-user to control avatars, check 'Two Users' in the parameters of 'KinectManager'-Script in the Inspector. If this is the case, repeat steps 4-5 for each avatar, controlled by the 2nd user. Then repeat step 7, but this time for 'Player 2 Avatars' list.
9. Check 'Compute User Map' and 'Display User Map'-checkboxes, if you want to see the User-depth Map after the user calibration completes.
10. Use the public functions of 'KinectManager'-script in your scripts. As an example, take a look at 'AvatarController.cs' or at 'GesturesDemoScript.cs', used by KinectGesturesDemo-scene.

Additional Reading

The following how-to tutorials are also located in the Assets-folder of the example Unity-package:

1. Howto-Use-Gestures-or-Create-Your-Own-Ones.pdf
2. Howto-Use-KinectManager-Across-Multiple-Scenes.pdf

References

This example is based on the following two examples from CMU.edu. A big "Thank you" to their authors:

- http://wiki.etc.cmu.edu/unity3d/index.php/Microsoft_Kinect_-_Open_NI
- http://wiki.etc.cmu.edu/unity3d/index.php/Microsoft_Kinect_-_Microsoft_SDK

Support and Feedback

E-mail: rumen.filkov@gmail.com, Skype, Twitter: roumenf, Whats App: on request