How to Run the Example

1. Download and install Kinect SDK 1.8, as described in the next section.
2. Open scene 'KinectAvatarsDemo', located in Assets/AvatarsDemo-folder.
3. Run the scene. Move around to see how the avatars and the cube-man reflect your movements.
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 re-run the scene.
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. Watch how the green ball follows the position of your right hand on the video screen.

Installation of Kinect Sensor with MS SDK 1.8

1. Download the Kinect SDK 1.8 or Kinect Windows Runtime 1.8. Here is the download page: http://www.microsoft.com/en-us/download/details.aspx?id=40278
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 (the 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 – the one that has his back turned to you - is not mirrored. It reproduces your movements in the same direction. Your left side is his left side and your right side is his right side. See it that way – from main camera's point of view, you're also staying with your back to the 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 enabled.

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

the needed avatar rotation in your scene. Then you can move or rotate the avatar within the scene by moving or rotating the control-object, leaving the avatar's local position and rotation at (0, 0, 0). Pay also attention to 'Mirrored Movement'-parameter of the AvatarController, attached to your avatar's 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. Open Unity editor and 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'-component in the Inspector.
5. Disable 'Mirrored Movement', if the avatar should move in the same direction as the user. Enable 'Mirrored Movement', if the avatar should mirror user's movements
6. Add 'KinectManager'-script to the MainCamera. If you use multiple cameras, create an empty GameObject and add the script to it.
7. (Optional) Drag and drop the avatars' game objects from Hierarchy to the 'Player 1 Avatars' list-parameter of KinectManager.
8. If you need a 2nd Kinect-user, enable 'Two Users'-parameter of 'KinectManager'. In this case, repeat steps 4-5 for each avatar, controlled by the 2nd user. Then go back to step 7, but this time use the 'Player 2 Avatars' list-parameter of KinectManager.
9. Enable 'Compute User Map' and 'Display User Map'-checkboxes, if you want to see the depth map on the screen.
10. Use the public functions of 'KinectManager'-script in your scripts. As examples, look at 'GesturesDemoScript.cs' used by KinectGesturesDemo-scene, or 'KinectOverlayer.cs' used by KinectOverlayDemo-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