Attention: This is based on preliminary software and hardware, subject to change.

How to Run the Example

1. Download and install Kinect v2 SDK as described in the next section.
2. Open scene ‘KinectAvatarsDemo’, located in Assets/AvatarsDemo-folder.
3. Run the scene. Move your body and look how both avatars (normal/mirrored) and the cubeman 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 – 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 turn the presentation cube left or right.
8. Open and run ’KinectInteractionDemo’-scene, located in Assets/InteractionDemo-folder. Try to grip and then drag an object on the screen. Release the object. Try dragging and dropping objects with your right hand and with your left hand.

Installation of Kinect v2 SDK

1. Download the Kinect SDK or Kinect Windows Runtime.
2. Run the installer. Installation of Kinect SDK/Runtime is simple and straightforward.
3. Connect the Kinect sensor. The needed drivers are 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 named '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 enabled. 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 at you is not mirrored. It reproduces your movements as they are. 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 enabled.

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 needed. Then you can move or rotate the avatar within the scene by moving or rotating the Ctrl-object, leaving the avatar's local position and rotation at 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.
7. (Optional) Drag and drop the avatars' game objects from Hierarchy to the 'Avatar Controllers'-list parameter of KinectManager component.
8. Enable 'Compute User Map' and 'Display User Map'-parameters, if you want to see the user-depth map. Enable 'Compute Color Map' and 'Display Color Map'-parameters, if you want to see the color camera image. Enable 'Display Skeleton Lines' parameter, if you want to see how Kinect tracks the skeletons on the user-depth map.
9. Use the public functions of 'KinectManager'-script in your scripts. For examples, see the 'AvatarController.cs' or 'GesturesDemoScript.cs'-scripts.

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

Support and Feedback

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