How to Run the Kinect Demo Scenes

1. Download and install Kinect v2 SDK as described in the next section.
2. Download and import this package.
3. If you want to utilize the included shaders instead of CPU image processing, make sure that Direct3D11 is the first option in the 'Graphics API'-list, in Player Settings / Other Settings / Rendering.
4. If you want to utilize the Kinect speech recognition, download and install the Speech Platform Runtime or SDK, as well as the needed language packs, as described in the next section.
5. Open and run a demo scene of your choice:
    - KinectAvatarsDemo1, located in KinectDemos/AvatarsDemo-folder. Move around to see how the avatars and the cube-man reflect your movements. Try one or more of the suggested gestures.
    - KinectAvatarsDemo2, located in the same folder. See how the first-person avatar (and the cube-man) reflect your movements. Try to see your arms and legs.
    - KinectGesturesDemo1, located in KinectDemos/GesturesDemo-folder. Swipe left, right or up turn the presentation cube left, right or up. You can also try the 'Seated' VGB gesture here.
    - KinectGesturesDemo2, located in the same folder. Use the Wheel-gesture to turn the model left or right, or Zoom-in / Zoom-out to scale the model.
    - DepthColliderDemo, located in KinectDemos/ColliderDemo-folder. Move your body to bounce the falling eggs.
    - ColorColliderDemo, located in the same folder. Touch any of the creatures with your hands to make it jump. Each sound signals a collision event.
    - KinectInteractionDemo1, located in KinectDemos/InteractionDemo-folder. Use your left or right hand to control the hand-cursor on the screen. Grip an object to drag it around. Open your hand to release it. Try to interact with the GUI components, too.
    - KinectInteractionDemo2, located in the same folder. Grip the cube with your hand and turn it in all directions.
    - KinectOverlayDemo1, located in KinectDemos/OverlayDemo-folder. Move your hands. See how the green ball follows the position of your right hand on the screen.
    - KinectOverlayDemo2, located in the same folder. See how the spheres overlay the tracked joints, and the lines – the bones between them.
    - KinectOverlayDemo3, located in the same folder. Close your hand to start drawing. Open it to stop.
    - KinectFaceTrackingDemo1, located in KinectDemos/FaceTrackingDemo-folder. See how the Kinect-generated face model overlays your face on the screen.
    - KinectFaceTrackingDemo2, located in the same folder. See how the hat moves relative to your head.
    - KinectFaceTrackingDemo3, located in the same folder. See how the user face is tracked dynamically.
    - KinectFaceTrackingDemo4, located in the same folder. See how the face expressions (animation units or AUs) affect the rigged face's joints (eyebrows, mouth, etc.)
    - KinectSpeechRecognition, located in KinectDemos/SpeechRecognitionDemo-folder. Say clearly one of the commands to control the robot. Repeat, if needed. Then open the xml-grammar file 'SpeechGrammar.grxml', located in the Assets/Resources folder of the Unity project, and modify the commands, according to your needs. Save the grammar file and run the scene to try them out.
    - KinectBackgroundRemoval1, located in KinectDemos/BackgroundRemovalDemo-folder. See how the cut-out user's image is mixed with the background texture.
    - KinectBackgroundRemoval2, located in the same folder. Check how to set the cut-out user's image as a 2nd background layer and put 3d-objects on top of it.

- Scene0-StartupScene, located in KinectDemos/MultiSceneDemo-folder. Add Scene0, Scene1 and Scene2 to the 'Scenes in Build'-list in 'File / Build Settings'. Then run the startup scene. This set of scenes demonstrates how to use the Kinect-related managers across multiple scenes in a game.
- KinectFittingRoom1, located in KinectDemos/FittingRoomDemo-folder. Stand in T-pose for calibration. See how the clothing model overlays your body on the screen.
- KinectFittingRoom2, located in the same folder. Stand in T-pose for calibration. See how the humanoid model overlays your body on the screen.
- KinectRecorderDemo, located in KinectDemos/RecorderDemo-folder. Say 'Record' to start recording your body movements, or say 'Play' to replay the previously saved movements.

Installation of Kinect v2 SDK

1. Download the Kinect for Windows SDK 2.0. Here is the download page: http://www.microsoft.com/en-us/download/details.aspx?id=44561
2. Run the installer. Installation of Kinect SDK/Runtime 2.0 is simple and straightforward.
3. Connect the Kinect v2 sensor. The needed drivers are installed automatically.
4. If you want to use the Kinect speech recognition, download and install the MS Speech Platform Runtime v11 (or Speech Platform SDK v11). Install both x86 and x64-packages, to be on the safe side. Here is the download page: http://www.microsoft.com/en-us/download/details.aspx?id=27225
5. For the Kinect speech recognition, you also need to download and install the respective language pack. Here is the download page: https://www.microsoft.com/en-us/download/details.aspx?id=43662

Why Are There Two Avatars in the Scene

The meaning of the two avatars (3D humanoid characters) in the scene is to demonstrate that you can have both – mirrored and non-mirrored movements in your scene, as well as avatars that follow multiple users.

First, you can have an avatar that mirrors your movement. This is the right one, facing you in the example scene. As you can see, its transform has Y-rotation (rotation around Y-axis) set to 180 degrees. Also, there is an AvatarController-component, attached to the avatar's game object and its 'Mirrored Movement'-parameter is enabled. Mirrored movement means that when you, for instance, lift your left hand the avatar lifts his right hand and vice versa, like a mirror.

The left avatar, the one that has his back turned at you, is not mirrored. It reproduces your movements exactly as they are. Your left is his left and your right is his right. Its transform has Y-rotation set to 0 and the 'Mirrored Movement'-parameter of its AvatarController is disabled.

Also note that the left avatar is assigned to Player-index 0 (i.e. the 1st user), while the right one is assigned to Player-index 1 (i.e. the 2nd user). To make it follow the movements of the 1st user as well, change the 'Player index' setting of its AvatarController-component to 0.

In order to get correct avatar position and movement, first position and rotate the avatar's game object in the scene, as needed. Then attach AvatarController-component to the avatar's game object and set its 'Mirrored Movement'-parameter accordingly.

How to Reuse the Kinect-related Scripts in Your Own Unity Project

1. Copy folder 'KinectScripts' from the Assets-folder of the example to the Assets-folder of your project. This folder contains all needed scripts, filters and interfaces.
2. Copy folder 'Resources' from the Assets-folder of the example to the Assets-folder of your project. This folder contains the needed libraries and resources. You may skip the libraries you don't want to use.
3. Copy folder 'Standard Assets' from the Assets-folder of the example to the Assets-folder of your project. It contains the wrapper classes for Kinect v2.
4. Wait until Unity detects and compiles the newly copied resources and scripts.
5. Create a KinectController-object and add the 'KinectManager' as component to it.
6. Enable 'Compute User Map' and 'Display User Map'-parameters, if you want to see the user-depth map on screen. Enable 'Compute Color Map' and 'Display Color Map'-parameters, if you want to see the color camera image on screen. Enable 'Display Skeleton Lines' parameter, if you want to see how Kinect tracks the skeletons on the user-depth map.
7. Add the 'AvatarController'-component to each avatar (humanoid character) in the scene that you need to control with the Kinect-sensor. Enable 'Mirrored Movement'-parameter of the AvatarController, if the avatar should mirror user's movements.
8. You can use the public functions of 'KinectManager', 'InteractionManager', 'FacetrackingManager' and 'SpeechManager' in your scripts, too. Examples: 'CubeGestureListener.cs' and 'CubePresentationScript.cs' used by the KinectGesturesDemo1-scene, 'GrabDropScript.cs' used by the KinectInteractionDemo1-scene, 'KinectOverlayer.cs' used by the KinectOverlayDemo1-scene, 'ModelFaceController.cs' used by the KinectFaceTrackingDemo1-scene or 'BotControlScript .cs' used by the KinectSpeechRecognition-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

More Information, Support and Feedback

Tip and Tricks: http://rfilkov.com/2015/01/25/kinect-v2-tips-tricks-examples/
Troubleshooting: http://rfilkov.com/2014/08/01/kinect-v2-with-ms-sdk/#ki

Web: http://rfilkov.com
Contact: http://rfilkov.com/about/#contact
Twitter: roumenf