How to Use Kinect v2 Examples with MS-SDK

1. Download and install Kinect-v2 SDK as described in the next section.
2. If you want to use the Kinect-v2 speech recognition, download and install the Speech Platform Runtime or SDK, as well as the needed language packs, as described in the next section.
3. Import this package into new Unity project.
4. Open 'File / Build settings' and switch to 'PC, Mac & Linux Standalone', Target platform: 'Windows'.
5. Make sure that Direct3D11 is the first option in the 'Graphics API'-list, in 'Player Settings / Other Settings / Rendering'.
6. Open and run a demo scene of your choice from a subfolder of the 'Assets/KinectDemos'-folder. See the short descriptions of all demo scenes below.

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 2.0 (or Kinect Runtime 2.0) is straightforward.
3. Connect the Kinect-v2 sensor. The needed drivers will be installed automatically.
4. If you want to use the Kinect-v2 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-v2 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

Short Descriptions of the Available Demo Scenes

1. KinectAvatarsDemo1, located in KinectDemos/AvatarsDemo-folder. It shows avatars controlled by the user from third person perspective. Move around to see how the avatars reflect your movements.
2. KinectAvatarsDemo2, located in the same folder. This demonstrates the avatar control from first-person perspective. Look around. Move and turn a bit. Look at your arms and legs.
3. KinectAvatarsDemo3, located in the same folder. This scene utilizes the AvatarControllerClassic-component (to control the upper body only), and the OffsetNode-setting (avatar motion is relative to the cube).
4. KinectBackgroundRemoval1, located in KinectDemos/BackgroundRemovalDemo-folder. This scene demoes the BackgroundRemovalManager-component. Look how the user image overlays the background image.
5. KinectBackgroundRemoval2, located in the same folder. See how to set the user's image as a 2nd background layer and put 3d-objects behind it (the halo around user's head), or in front of it (the cubes in the scene).
6. KinectBackgroundRemoval3, located in the same folder. Here you can see how to utilize a separate layer and camera, in order to display images, objects and image effects behind the user's silhouette.
7. KinectBackgroundRemoval4, located in the same folder. Here the background removal gets inverted. The user's silhouette is transparent, while the camera-captured background is opaque (visible).

8. KinectBackgroundRemoval5, located in the same folder. This demo adds depth to the background removal. See how the user's silhouette moves around in the 3D scene. Check the KinectUserVisualizer scene too.
9. ColorColliderDemo, located in KinectDemos/ColliderDemo-folder. It demonstrates 'virtual touch' possibilities – between your hands and the scene objects. Try to touch any of the little guys there with your hands.
10. DepthColliderDemo2D, located in the same folder. It shows how your silhouette can interact with falling objects in a 2D-scene. Raise your hands and try to collect as many balls as possible, then let them fall.
11. DepthColliderDemo3D, located in the same folder. It is similar to the previous one, but in 3D. Try to bounce the falling eggs with your hands or body.
12. KinectFaceTrackingDemo1, located in KinectDemos/FaceTrackingDemo-folder. It shows face overlay. See how the Kinect-generated face model overlays your face on the screen. The model can be textured or not.
13. KinectFaceTrackingDemo2, located in the same folder. It demonstrates how to move an object along with the user's head. Look how the hat model follows your head in real time.
14. KinectFaceTrackingDemo3, located in the same folder. It puts user's face as a texture of 3d object (the FaceView quad in the scene). See how the user's head replaces the model's head in the scene.
15. KinectFaceTrackingDemo4, located in the same folder. Similar to the 1st demo, but it uses rigged head model instead of the internally generated face mesh. See how the head model overlays your face in the scene, and how your face expressions affect the rigged head. The ModelFaceController-component does the job.
16. KinectFittingRoom1, located in KinectDemos/FittingRoomDemo-folder. This is the primary dressing room demo. Stand in T-pose for calibration. Wait a bit for gender and age estimation (this is optional). Then use hand swipes (left or right) to select the next or previous clothing model, or select one from the menu.
17. KinectFittingRoom2, located in the same folder. This is the secondary dressing room demo. See how the humanoid model overlays your body. You are free to replace it with your own humanoid model.
18. KinectGesturesDemo1, located in KinectDemos/GesturesDemo-folder. It demonstrates the detection of discrete gestures. Swipe left, right or up to turn the presentation cube left, right or up.
19. KinectGesturesDemo2, located in the same folder. It demonstrates the detection of continuous gestures. Use the Wheel-gesture to turn the model left or right continuously. Use the Zoom-in gesture, to scale the model. Lower your hands between the gestures, to let the system know when the gesture finishes.
20. VisualGesturesDemo, located in the same folder. Here you can see how the visual gestures (created with Visual Gesture Builder) can be detected and used. Stand up and sit to try the 'Seated'-gesture here.
21. KinectInteractionDemo1, located in KinectDemos/InteractionDemo-folder. It demoes cursor control and hand interactions. See how the hand-cursor on screen follows your right or left hand. Close your hand to grab an object and drag it around. Open your hand to release it. Try to interact with GUI components.
22. KinectInteractionDemo2, located in the same folder. It shows how to use hand interactions to rotate a 3d-object. Grab the cube with your left or right hand. Then hold it and can turn it in all directions.
23. KinectDataServer, located in KinectDemos/KinectDataServer-folder. Actually, this is not a demo scene, but the data server, used in combination with all 'Kinect-v2 VR Examples'-demo scenes.
24. KinectMovieDemo, located in KinectDemos/MovieSequenceDemo-folder. This is to show you how to play a set of movie frames with your body. Move right or left to play the sequence forward or backward.
25. Scene0-StartupScene, located in KinectDemos/MultiSceneDemo-folder. This demonstrates how to use the Kinect-related components across multiple scenes. Open 'File / Build Settings' from the menu and add Scene0, Scene1 and Scene2 to the 'Scenes in Build'-list. Then run the startup scene.
26. KinectOverlayDemo1, located in KinectDemos/OverlayDemo-folder. This is the basic joint-overlay demo. Move your right hand to see how the green ball follows it. This is done by the JointOverlayer-component.
27. KinectOverlayDemo2, located in the same folder. It shows how the sensor sees your body. The green balls overlay the tracked joints of your body. The lines between them represent the bones.
28. KinectOverlayDemo3, located in the same folder. This is a simple 'draw-in-the-air' application. Close your right hand to start drawing. Open it to stop. Press 'U' to undo the last line drawn.

29. KinectPhotoBooth, located in the same folder. This is a photo-booth application, courtesy of Sophiya Siraj. Swipe left or right to change the 2d-models. Make a right-hand grip above the shoulder to make a photo.
30. KinectPhysicsDemo, located in KinectDemos/PhysicsDemo-folder. This is a basic ball physics demo. Raise a hand to get a virtual ball on the screen. Throw the ball and try to hit the barrel.
31. KinectRecorderDemo, located in KinectDemos/RecorderDemo-folder. This is a simple body-data recorder and player. Say 'Record' to start recording your body movements, 'Stop' to stop it, or 'Play' to replay the previously saved movements. You can copy the data files between machines or replay them later.
32. KinectSpeechRecognition, located in KinectDemos/SpeechRecognitionDemo-folder. This is basic speech-recognition demo. Say clearly one of the listed commands to control the robot. Repeat it, if needed. The grammar file 'SpeechGrammar.grxml' is located in the Assets/Resources folder.
33. KinectHandObjectChecker, located in KinectDemos/VariousDemos-folder. It shows how the Kinect can detect an object in your hands. For instance get a book in one of your hands, then pass it to the other.
34. KinectHeightEstimator, located in the same folder. This is a simple measurement demo. See how the height and certain widths of the user's body can be estimated, according to the provided depth information.
35. KinectHolographicViewer, located in the same folder. This is simple holographic-view demo, based on the SimpleHolographicCamera-component and script, courtesy of Davy Loots. See how the cube perspective changes, when you move left or right to the sensor.
36. KinectPoseDetector, located in the same folder. This is visual pose-detection demo. Check how the differences between bone angles of the model and avatar are summed up to provide pose matching estimation. This is done by the PoseDetectorScript-component.
37. KinectSceneVisualizer, located in KinectDemos/VisualizerDemo. It converts the scene, as seen by the sensor, to a mesh and overlays it over the camera image. This is done by the SceneMeshVisualizer-component.
38. KinectUserVisualizer, located in the same folder. It converts the user, as seen by the sensor, to a mesh and puts it into the 3D-scene. This is done by the UserMeshVisualizer-component.

Why Are There Two Avatars in KinectAvatarsDemo1-scene

The presence of the two avatars (humanoid characters) in the scene is to demonstrate different options of their AvatarController-components. AvatarController is the component that controls the motion of the avatar model.

Look at the right avatar. It mirrors your movements (your left arm is his right one, etc.), and moves with respect to the main camera the same way as you move with respect to the sensor. Its transform's Y-rotation is set to 180 degrees, hence the model is turned to you (just as in a mirror). The AvatarController-component of this avatar's game object has its 'Mirrored Movement'-setting enabled, as well. Its 'Pos relative to camera'-setting references the main camera, to make it the origin of avatar movements' coordinate system.

The left avatar (the one that is turned with his back at you) is not mirrored. It follows your movements as they are. Your left is its left and your right is its right. Its transform Y-rotation is set to 0 (it is turned in your direction) and the 'Mirrored Movement'-parameter of its AvatarController is disabled. Its 'Pos relative to camera'-setting is None. This makes it move with respect to its initial position.

Pay special attention to the 'Player index'-setting of both avatars. The same setting you will find in many other user-related components. It specifies the user index of the user, who controls the avatar. 0 means the 1st user, 1 – the 2nd one, 2 – the 3rd one, etc. If you want the avatar to be controlled by another user, just change the 'Player index'-setting of its AvatarController-component accordingly.

How to Reuse the Kinect-related Scripts in Your Own Unity Project (Windows Builds)

1. Copy folder 'KinectScripts' from the Assets-folder of this package to the Assets-folder of your project. This folder contains all needed components, scripts, filters and interfaces.
2. Copy folder 'Resources' from the Assets-folder of this package to the Assets-folder of your project. This folder contains the needed libraries and resources. You may skip copying the libraries you don't need.
3. Copy folder 'Standard Assets' from the Assets-folder of this package to the Assets-folder of your project. It contains the wrapper classes for Kinect-v2.
4. Wait until Unity detects, imports and compiles the newly detected resources and scripts.
5. In your scene, create a KinectController-game object and add the 'KinectManager'-component to it. This is a required component for all Kinect-related scenes.
6. Add the 'AvatarController'-component to each avatar in the scene, who needs to be controlled by the Kinect-sensor. See the previous section for more information about the AvatarController-component.
7. Feel free to use the public API of 'KinectManager', 'InteractionManager', 'FacetrackingManager', 'SpeechManager', etc. Kinect-related components in your scripts. Look here for more information, tips and tricks: http://rfilkov.com/2015/01/25/kinect-v2-tips-tricks-examples/

Additional Reading

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

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

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

Contact: http://rfilkov.com/about/#contact (please mention your invoice number)
Twitter: https://twitter.com/roumenf