InstantVR Free Support

Version: 3.2
Date: December 28, 2015

Features

The Free version of InstantVR supports the following input:

- keyboard/mouse: using standard WASD input and mouse look functions
- Xbox controller: standard controller input is supported with left analog stick for walking around and right analog stick for looking around.
- Oculus Rift DK1 & DK2
- Samsung Gear VR
- Google Cardboard for Android.

Prerequisites

- The current distribution only supports development on Windows and Android.
- Unity 5.1.2f1 or higher.
- Oculus Runtime for Windows version 0.7.0.0 or higher is required. The runtime can be downloaded from the Oculus VR site.

Usage

To use InstantVR you need an environment first. The minimum is a flat terrain with a directional light, but you can make it as complex as you want with lots of meshes, rigidbodies and colliders. You are only bound by the limits of Unity and the computer used to drive the game.

You should not include a camera of other first or third person objects in your game. This is fully handled by InstantVR.

To complete the scene you should drag one of the prefabs named 'MH_…' from the folder InstantVR into your scene.

Now you can press play and walk around your wonderful environment.

General configuration

InstantVR

The instantVR contains references to the 6 transforms which move the avatar. These transforms can be placed anywhere within the Hierarchy.

IVR_Walking

The walking script implements walking around using input.

Walking forward and backward is implemented using the standard keys:

- on keyboard: w key = forward, s key = backward
- on gamepad: left analog stick up/down
- on Razer Hydra: left analog stick up/down

The direction in which you walk depends on the body orientation.

Sidestepping is supported using the following keys:

- on keyboard: a key = left, d key = right
- on gamepad: left analog stick left/right
- on Razer Hydra: left analog stick left/right

The direction of stepping depends on the body orientation.

Body rotation uses the same keys as sidestepping:

- on keyboard: a key = left, d key = right
- on gamepad: left analog stick left/right
- on Razer Hydra: left analog stick left/right

The script has the following options:

- Walking: enables forward and backward walking
- Sidestepping: enables left and right sidestepping
- Rotating: enables body rotation
- Rotation speed rate: sets the speed for rotating the body
- Proximity speed: makes the walking speed dependant on the proximity of statis object
- Proximity speed rate: the amount proximity speed influences the walking speed.

  Note that when sidestepping and rotation are selected together, body rotation and sidestepping take place simultaneously.

Without this script physical walking is still possible using the Oculus Rift DK2 or Microsoft Kinect.

IVR_Body Movements

This script translates the movements of the targets to the actual body positions.

It has the following options:

- Enable torso: enables body movements of the upper body
- Enable left: enables leg movements

Extensions

A number of extensions is included which can be added to the gameObject with InstantVR which add support for one or more input devices or other target controllers.

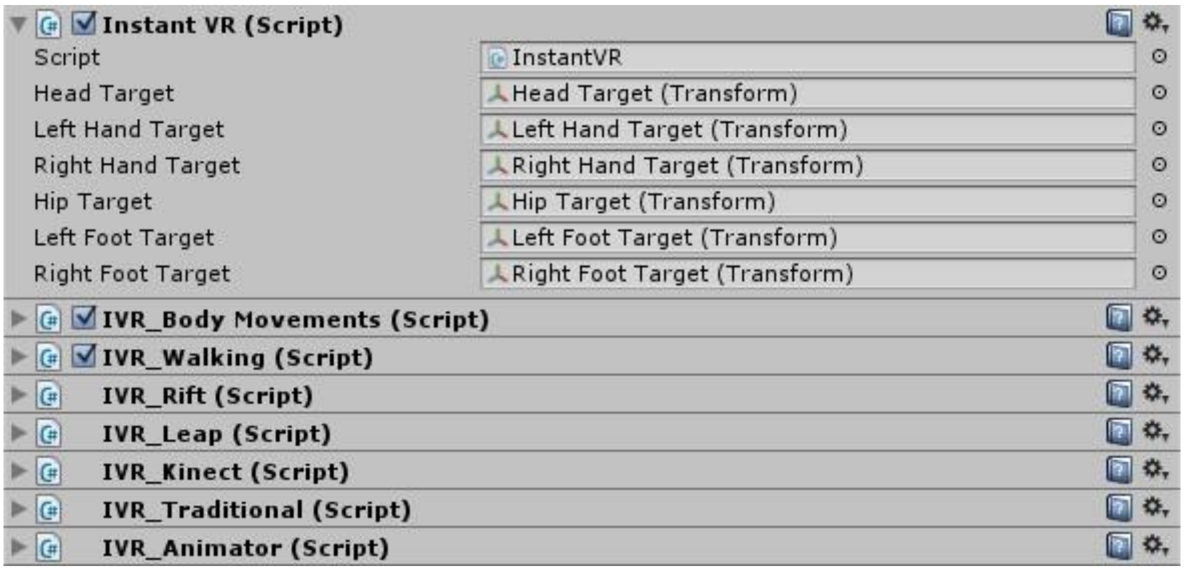
Extensions can be added to every GameObject with an InstantVR script attached to extend the tracking functionality. Extensions will typically implement support for specific input devices like the Oculus Rift or Microsoft Kinect, but also include extensions for animations or networking.

Multiple extensions can be added to a single InstantVR gameObject. The extensions which will be used at play time depends on:

- availability: is the hardware associated with the extension present?
- priority: the extensions with the highest priority are chosen over lower priority extensions.
- tracking: is the hardware currently tracking?

In the example below, we have added 5 extensions: Oculus Rift, Leap Motion, Kinect 2, Traditional (Xbox 360/mouse/keyboard) and Animator.

Note: Leap Motion and Kinect are only supported in InstantVR Advanced, but they are included here for illustration



Instant VR (Script)

Script	InstantVR
Head Target	Head Target (Transform)
Left Hand Target	Left Hand Target (Transform)
Right Hand Target	Right Hand Target (Transform)
Hip Target	Hip Target (Transform)
Left Foot Target	Left Foot Target (Transform)
Right Foot Target	Right Foot Target (Transform)

- ✓ IVR_Body Movements (Script)
- ✓ IVR_Walking (Script)
- IVR_Rift (Script)
- IVR_Leap (Script)
- IVR_Kinect (Script)
- IVR_Traditional (Script)
- IVR_Animator (Script)

Dynamic behavior

During gameplay with all supported hardware available the behavior is as follows:

- Oculus Rift will always be used for head tracking. It has the highest priority and is always tracking.
- Leap Motion is used for tracking the hands when the hands are in the field of view of the Leap Motion cameras. When the hands are outside the view they cannot be tracked by the Leap Motion so it will not be used then. Hand tracking will drop down to the next hand tracker in the hierarchy
- Kinect 2 supports tracking of all 6 tracking targets, but in this case it will not be used for head tracking, because the Oculus Rift has a higher priority. It will be used for

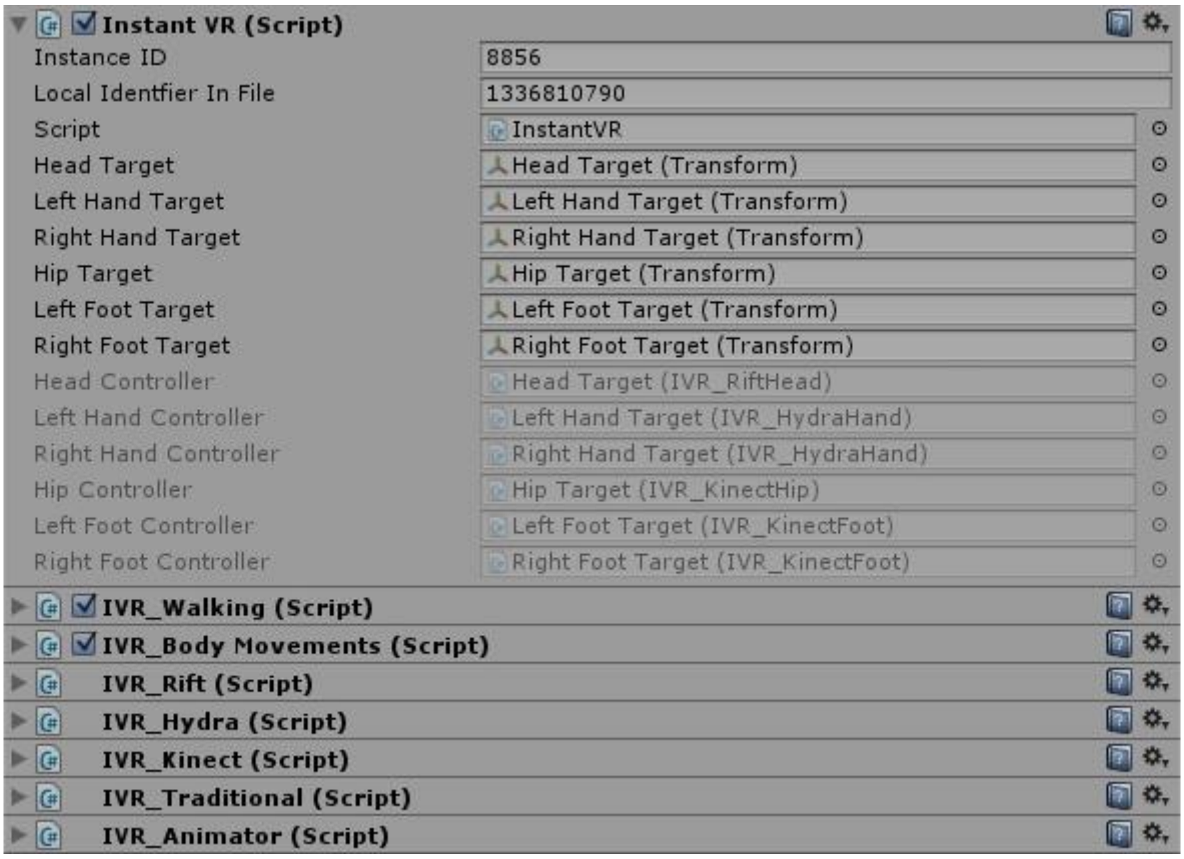
hand tracking when the hands are outside the view of the Leap Motion camera. All other body parts are tracked by Kinect 2 all the time.
- Traditional input is can be used for walking around
- The animator is not used, as all targets are tracked by Kinect.

When the game is played with just the Oculus Rift available, it will behave like this:

- Oculus Rift will always be used for head tracking. It has the highest priority and is always tracking.
- Leap Motion and Kinect 2 are not available, so will not be used
- Traditional input is can be used for walking around
- The animator is used to move all targets except the head, which is tracked by the Oculus Rift. This results in leg and arm movements during walking and rotation movements.

Traditional and Animator are typically used when no or limited VR hardware are available and it is good practice to have them as the lowest 2 priority spots as is shown here.

During play mode in the editor it is possible to view which extensions are currently used for each target by switching to debug view:



Instant VR (Script)

Instance ID	8856
Local Identfier In File	1336810790
Script	InstantVR
Head Target	Head Target (Transform)
Left Hand Target	Left Hand Target (Transform)
Right Hand Target	Right Hand Target (Transform)
Hip Target	Hip Target (Transform)
Left Foot Target	Left Foot Target (Transform)
Right Foot Target	Right Foot Target (Transform)
Head Controller	Head Target (IVR_RiftHead)
Left Hand Controller	Left Hand Target (IVR_HydraHand)
Right Hand Controller	Right Hand Target (IVR_HydraHand)
Hip Controller	Hip Target (IVR_KinectHip)
Left Foot Controller	Left Foot Target (IVR_KinectFoot)
Right Foot Controller	Right Foot Target (IVR_KinectFoot)

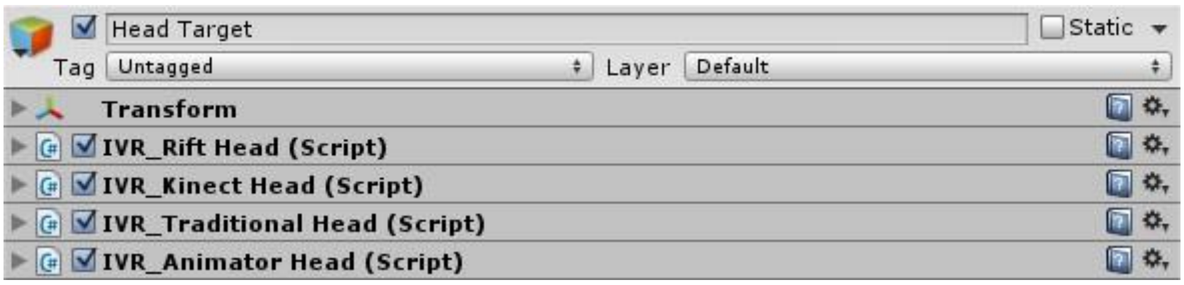
IVR_Walking (Script)
IVR_Body Movements (Script)
IVR_Rift (Script)
IVR_Hydra (Script)
IVR_Kinect (Script)
IVR_Traditional (Script)
IVR_Animator (Script)

In this case we have the extensions for Oculus Rift, Razer Hydra, Microsoft Kinect, the traditional and animator extensions configured. In the debug view of InstantVR you can see the various controllers used: Rift for the Head Target, Hydra for the Hand Targets and Kinect for the Hip and Feet Targets.

Controllers

Every extension implements one or more controllers for targets. The Oculus Rift extension has one controller: the Rift Head Controller which is used solely for the head target. The Leap Motion and Razer Hydra extensions just have a Hand Controller which is used for the left and right hands, while the Kinect extensions have controllers for every target as they can track the full body.

If we look into the Targets themselves we can see the controllers currently associated with that target. Here we have an example of the Head Target:



You can see the Rift and Kinect Head Controllers. Like the priority for extensions the order in which the controllers appear determines the priority. This order is in fact controlled by the ordering of the extensions. So you need to reorder the extension when you want to change the priority of the Head Controllers. Direct changes in the order of controllers will be undone automatically.

Note: the controller priorities and positions are not updated when the inspector is in Debug mode. This is a known issues and is reported to Unity. The priorities and ordering will be corrected when the inspector is switched back to Normal mode.

It is possible to disable specific controllers. For instance: if you do not want to use Kinect Head tracking when the Rift is not available, but you do want to use Kinect for the rest of the body, you can disable the Kinect Head Controller. This is done by disabling the IVR_Kinect Head by unchecking the script in the example above. If you want to disable all controllers of an extension, it is better to remove the extension altogether.

Calibration

Certain extensions need calibration to work correctly, like the Oculus Rift.

The calibration information can be configured in the extensions, but it is also possible to calibrate during gameplay.

Calibration configuration is done by setting the Tracker Position in the extensions. The tracker position is the position of the tracker relative to the player’s position. For instance if the tracker is 1 meter in front of you at a height of 1.8 meter from the ground, 20cm to the left, the Tracker Position should be set to x = 0, y = 1.8, z = -0.2

For the Oculus Rift DK2, the tracker position is the position of the IR camera.

The calibration can also be done manually during gameplay using the calibration keys implemented the various input controllers:

- Pressing Tab on the keyboard
- Pressing Back and Start simultaneously on an Xbox controller
- Pressing both option buttons on the Hydra

IVR_Animator

The animator implements procedural animations for legs and arms. It is typically used as the last extension in the list of extensions as a fallback when these are not driven by a input device

Hand Targets

Follow head: when enabled, the hands will follow the X/Z position of the hips. Needs to be enabled for physical walking using the Rift and when using the IVR_Walking script

Hip Target

Follow head: when enabled, the hip will follow the X/Z position of the head. Needs to be enabled for physical walking using the Rift.

Rotation method determins how the rotation along the Y axis is determined:

- NoRotation: speaks for itself
- LookRotation: the body rotates to the direction in which the player is looking. Good option when using the Oculus Rift.
- HandOrientation: is not supported in the free version. It will behave the same as LookRotation
- Auto: chooses HandOrientation with the Hydra is used and LookRotation otherwise.

IVR_Cardboard

Google Cardboard enables a wide range of mobile phones to be used as a head mounted display. Cardboard is only supported on Android.

Important: the option ‘Virtual Reality Supported’ needs to be unchecked in the Edit menu, Project Settings, Player. This is in contrast to the IVR_Rift setting.

Head Target

Rotational tracking is supported for all mobile phones compatible with Google Cardboard.

IVR_RiftGearVR

Adds support for the Oculus Rift DK1 or DK2 on Windows or Gear VR on Android.

Important: the option ‘Virtual Reality Supported’ needs to be checked in the Edit menu, Project Settings, Player.

Head Target

Rotational tracking is supported for all versions of the Oculus Rift and the Gear VR. Positional tracking is only supported for the Oculus Rift DK2. Note that positional tracking is only possible when the headset is in view of the Oculus IR camera.

IVR_Traditional

Adds support for Xbox controllers or mouse/keyboard.

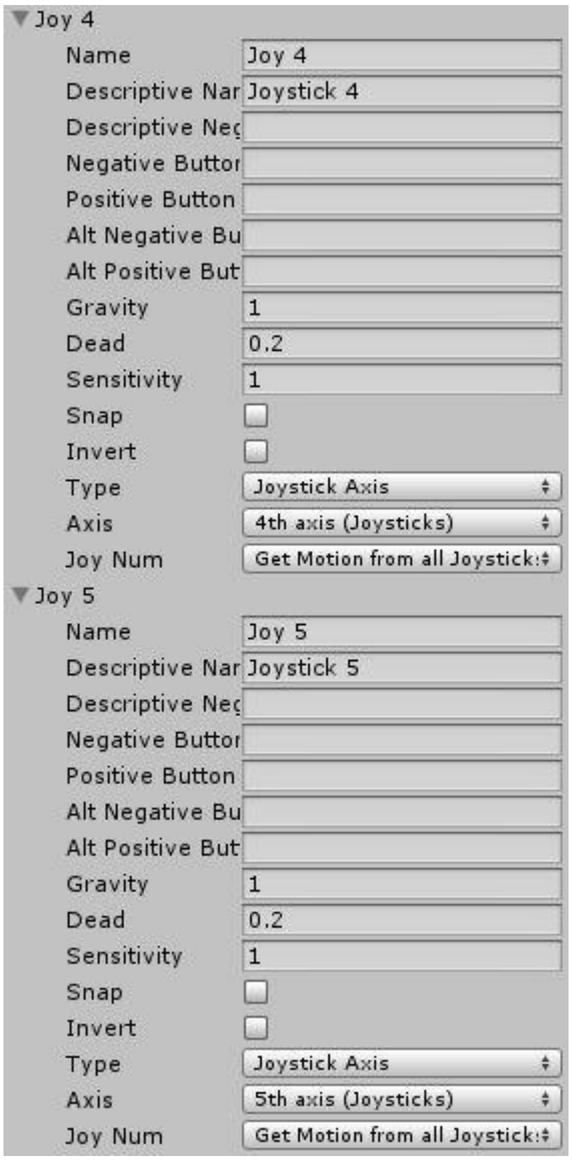
Left Hand Movements

	Xbox controller	Mouse/Keyboard
Thumb	DPad down/right	-
Index finger	left bumper	-
Middle finger	left trigger	-
Ring finger	left trigger	-
Little finger	left trigger	-

Right Hand Movements

Thumb	A/X buttons	-
Index finger	right bumper	left mouse button
Middle finger	right trigger	right mouse button
Ring finger	right trigger	right mouse button
Little finger	right trigger	right mouse button

To support Xbox controllers, you have to add two entries in the InputManager. You can open this from the Edit – Project Settings – Input menu option. The necessary values can be found in the picture below.



▼ Joy 4

Name	Joy 4
Descriptive Nar	Joystick 4
Descriptive Neç	
Negative Buttor	
Positive Button	
Alt Negative Bu	
Alt Positive But	
Gravity	1
Dead	0.2
Sensitivity	1
Snap	☐
Invert	☐
Type	Joystick Axis
Axis	4th axis (Joysticks)
Joy Num	Get Motion from all Joystick:

▼ Joy 5

Name	Joy 5
Descriptive Nar	Joystick 5
Descriptive Neç	
Negative Buttor	
Positive Button	
Alt Negative Bu	
Alt Positive But	
Gravity	1
Dead	0.2
Sensitivity	1
Snap	☐
Invert	☐
Type	Joystick Axis
Axis	5th axis (Joysticks)
Joy Num	Get Motion from all Joystick:

Walking

Physical walking is supported in combination with the Oculus Rift DK2 or Microsoft Kinect.

Known issues and limitations

The following issues are known to the current version of InstantVR:

- The priority of the controllers is not updated from the extensions priority when the inspector is in Debug Mode. This is a Unity issue which has been reported. The ordering will be corrected again when the inspector is switched back to Normal mode.
- When the Razer Hydra is not working and/or when you see the message that the sixense.dll is missing, although it is in the plugins folder, please update to version 1.0.6 of the Sixense Unity Plugin. This is a known issue with version 1.0.4 of the Sixense Unity Plugin.

InstantVR Version history

Version 1.0

- Initial release

Version 1.1

- Oculus Rift SDK 0.4.3 implemented with Unity Free support
- Included options to enable/disable controllers
- Look rotation using Rift is now working properly
- Improved forearm IK
- Leg orientation improved with high foot positions
- Free and Advanced code bases merged

Version 2.0

- Oculus Rift SDK 0.4.3.1
- Physical drift correction
- Free walking support
- Side stepping
- Removed Character Motor and Character Controller
- Hands no longer collide with body if they use the same layer

Version 2.1

- Proximity based walking speed
- Arm swing animation added
- Improved leg animation with thumbstick walking
- Improved Rift calibration. No need to face the Rift camera anymore.
- Oculus Rift SDK 0.4.4

Version 3.0

- Unity 5 support
- Modular architecture
- Body movements in editor
- Realtime switching between input controllers
- Oculus Rift SDK 0.5.0.1

Version 3.1

- Unity 5.1 support
- Google Cardboard support

Version 3.2

- Gear VR support

- Redesigned calibration

More information

The latest and detailed support information can be found at the Passer VR website:
http://serrarens.nl/passervr/support/instantvr-support/

Contact

You can contact me via email: support@passervr.com or use the contact form on my website: http://serrarens.nl/passervr/contact/

Thank you for supporting my work!
Pascal.