This package is based on the Steam VR API and uses its scripts and functions for the actions to work. A basic understanding is required to use SteamVR in Unity but this documentation and the scene provided will make this easy to understand.

Table of Contents:

This package contains: A demo scene containing most of the actions to show them in use. Actions necessary to use SteamVR in playmaker. CC License models, including a gun.obj, bow.fbx arrow.fbx, boxingGloves.obj, and a punchingMachine.obj. Documentation listing the features and functions in this package. Extras folder.

Any questions or comments please contact [*frametaleinfo@gmail.com*]

Understanding SteamVR

[ref.1]

Once the SteamVR package is installed you will have new GameObjects in your scene, including the [CameraRig] as well as the controller and camera.

The Camera Rig contains the play area script as well as the SteamVR Controller manager. The play area script determines the playspace and can be set to *calibrated* or fixed (between 3 options). The SteamVR Controller Manager is required to determine the left and right controller.

Aside from the [CameraRig], [SteamVR] is also inserted into the scene.

This gameObject contains a script(ref.2) which allows you to alter settings in Unity and SteamVR.

[ref.2]

This script is the backbone for the asset and allows you certain features that affect Unity during play, including the masks used on each eye, the tracking space for standing or sitting, physics during system dashboard, and the ability to pause unity (setting the time to 0) when the system button is pressed.

Understanding the Controller and Headset

Inside the [CameraRig] (ref.1) are the 2 controller objects as well as a headset. The controllers contain a 3D model of the same physical controller used in the dashboard menu when booting up SteamVR.

The camera (head) contains 2 child objects, camera (eyes) and camera (ears) which contains the camera and an audio listener respectively.

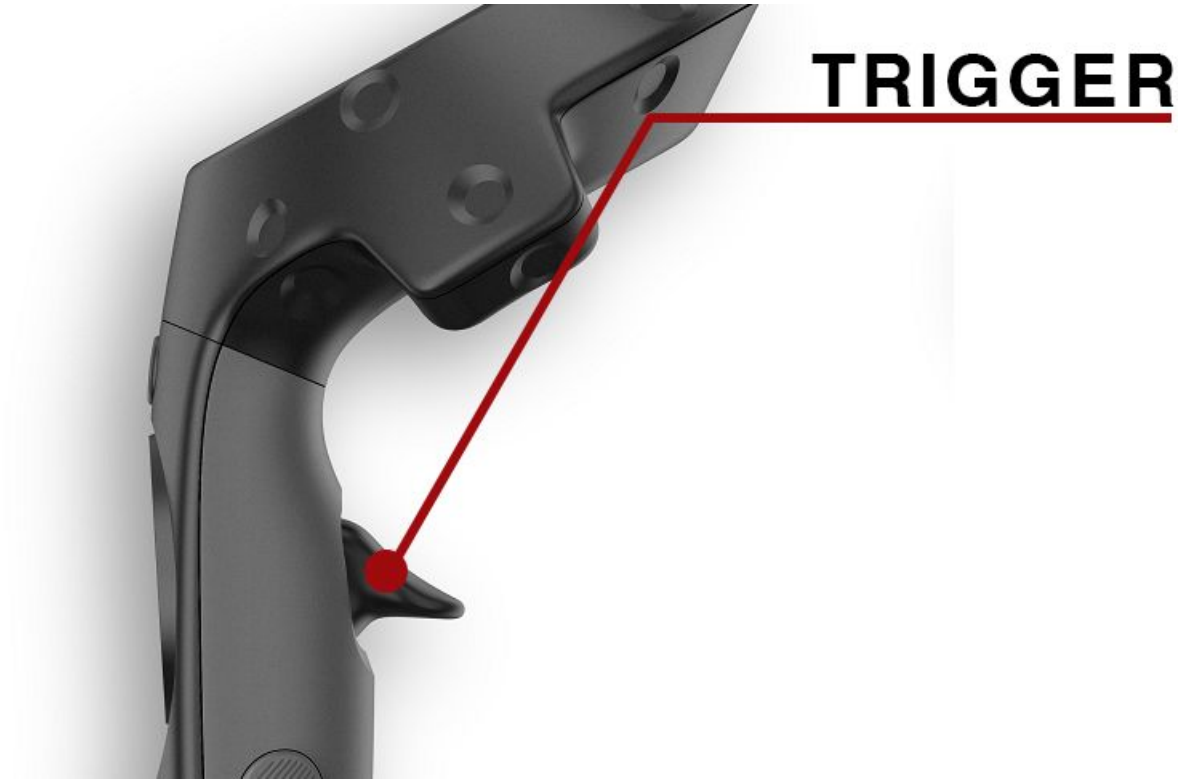
Note: There are **2** cameras in the scene, one for the player and one for anyone viewing the game on the monitor. The camera (head) gameObject has a camera script and the camera (eye) gameObject also has a camera script. The camera (eye) is the display for the headset while the camera (head) is the display for the monitor.

The camera (head) and controllers also contain a script named **SteamVR_TrackedObject**. Many custom actions call on functions contained in the Tracked Object Script and is necessary to run the states.

Most actions will show an error (ref.3) if it requires the Tracked Object Script.

[ref.3]

Get Trigger



The Get Trigger can be used to send an event based on a Trigger type.

Controller: The Tracked Object required to run event. [Requires SteamVR_TrackedObject]
Trigger Type: Select between type to trigger an event.
Send Event: Event to send when the specified Trigger Type is used.
Store Result: Set a bool to true when Trigger Type is used.

When using the Get Trigger Type, note that there are 6 different options to choose from in Trigger Type. These types are used primarily for the touchpad, but also work for the trigger. The difference between Get Press and Get Touch is that Get Touch activates slightly faster than Get Press.

Get Touchpad

The Get Touchpad can be used to send an event based on a Touchpad type.

Controller: The Tracked Object required to run event. [Requires SteamVR_TrackedObject]
Touchpad Type: Select between type to trigger an event.
Send Event: Event to send when the specified Touchpad Type is used.
Store Result: Set a bool to true when Touchpad Type is used.

When using the Get Touchpad Type, note that there are 6 different options to choose from in Touchpad Type. Get Touch is enabled when the player touches the touchpad. Get Press is enabled when the player pressed down on the touchpad. This is useful for determining the position of the players touch position using the Touchpad Axis.

Note: Both controllers use 6 different types for each button.
-Get Press
-Get Press Down
-Get Press Up
-Get Touch
-Get Touch Down
-Get Touch Up

Get Touch is primarily used for the touchpad since the touchpad has its circular surface area along with a button for Get Press. However, Get Press and Get Touch can also be used on the Trigger and Grip.

Get Touchpad Axis

The Get Touchpad Axis grabs the X and Y value based on the player's touch position.

Controller: The Tracked Object required to grab axis. [Requires SteamVR_TrackedObject]
Multiplier: Adds a multiplier to the vector2.
Store Vector: Sets the Vector2 touch position into a variable.

This action is used best when combined with Get Touchpad to determine the position the player has pressed on the trackpad to use as a variety of buttons. It can also be used to move objects or the player based on touch position. See the demo scene for the touchpad axis in action.

Get Grip

The Get Grip can be used to send an event based on a Trigger type.

Controller: The Tracked Object required to run event. [Requires SteamVR_TrackedObject]
Grip Type: Select between type to trigger an event.
Send Event: Event to send when the specified Grip Type is used.
Store Result: Set a bool to true when Grip Type is used.

Get Grip has 6 types to use between Get Press and Get Touch, but the 2 are virtually the same with either choice working as expected to send an event.

Get Menu

The Get Menu can be used to send an event based on a Trigger type.

Controller: The Tracked Object required to run event. [Requires SteamVR_TrackedObject]
Menu Type: Select between type to trigger an event.
Send Event: Event to send when the specified Menu Type is used.
Store Result: Set a bool to true when Menu Type is used.

Get Menu has 6 types to use between Get Press and Get Touch, but the 2 are virtually the same with either choice working as expected to send an event. The Get Menu button is most notable used as the options menu for VR titles but this action allows it to be used for any purpose.

Get System

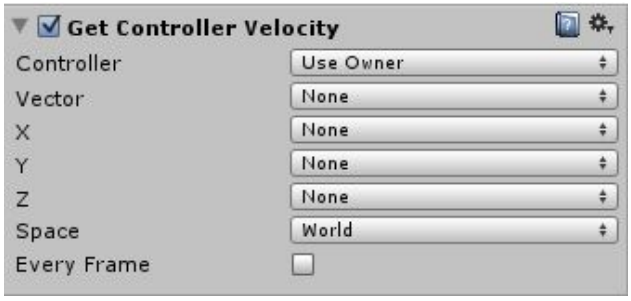
The Get System can send an event or set bool value based on button press.

Send Event: Event to send when the System button is used.
System Enabled: Set a bool to true when System button is pressed.

This action uses the script SteamVR_Render (Ref.2) to determine when to send an event or set bool value based on the system button press. This action is useful for setting a pause screen when the system button has been activated.

Get Controller Velocity

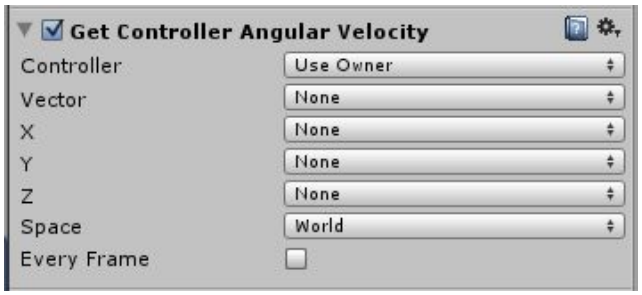
Determines the current speed of the controller.



Controller: The Tracked Object required to obtain velocity. [Requires SteamVR_TrackedObject] and [Rigidbody]
Vector: Vector3 current velocity.
X: Velocity based on the X axis.
Y: Velocity based on the Y axis.
Z: Velocity based on the Z axis.
Space: Set position in local (relative to parent) or world space.
Every Frame: Perform this action every frame.

Get Controller Angular Velocity

Determines the current angular speed of the controller.



Controller: The Tracked Object required to obtain angular velocity. [Requires SteamVR_TrackedObject] and [Rigidbody]
Vector: Vector3 current velocity.
X: Velocity based on the X axis.
Y: Velocity based on the Y axis.
Z: Velocity based on the Z axis.
Space: Set position in local (relative to parent) or world space.
Every Frame: Perform this action every frame.

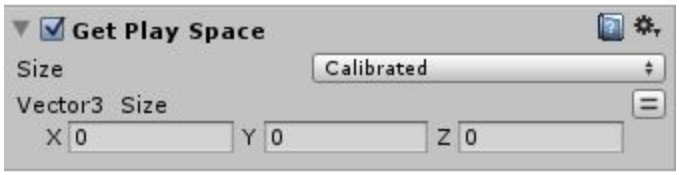
Get Gaze

Bool is set to true if object is in range between Gaze In Cutoff and Gaze Out Cutoff.

Gaze Object: The gameobject used to keep in view of headset.
Is in Gaze: Bool to note when gaze is in range.
Gaze in Cutoff: Cutoff closest to gameObject.
Gaze out Cutoff: Cutoff furthest from gameObject.

Get Playspace

Gets the Play Space determined by size, set to a Vector3.



Size: Select size to grab Vector3.
Vector3 Size: Sets Vector3 size determined by the size option.

Size has a selection between:
-Calibrated (Use to determine players custom playspace.)
-200x150
-300x225
-400x300

Set Rumble

Activates the Haptic Feedback on a controller.

Controller: The Tracked Object required to set haptic feedback. [Requires SteamVR_TrackedObject]
Duration: Set the length of the haptic feedback on the controller.
Strength: Set the strength. Value range from 0 to 1.

Steam Headset Fade

Uses the Steam Fade script to Fade the headset for a determined duration.
**Requires the script SteamVR_Fade to be placed on Camera (Eye) to work.*

Fade Color: The color used to fade the headset.
Duration: The time taken to complete the fade.
Finish Event: Event to send when the fade has completed.

Steam Teleport

Uses steams Teleport script to Teleport the player to the pointed location of the controller using the trigger button.

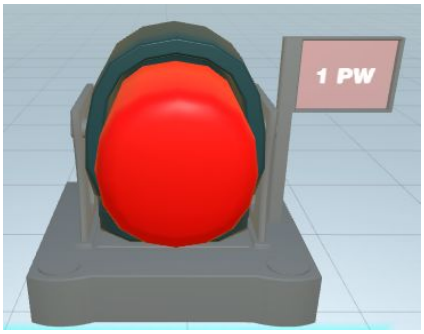
Controller: The Tracked Object required to teleport the player. [Requires SteamVR_TrackedObject]
Teleport Type: Set between the three teleport types to move with the controller.
Send Event: Event to send when the player has teleported. Leave blank to continue teleporting in the state.

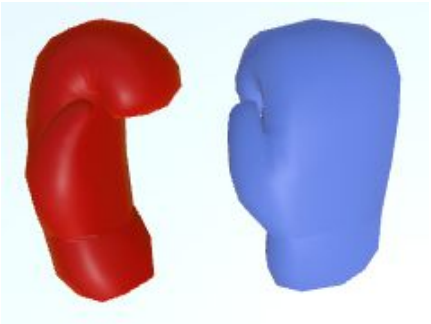
Teleport Types has three options:
Teleport Type use Terrain: Teleports the player using the Terrain Collider.
Teleport Type use Collider: Uses any collider pointed and clicked on to teleport the player.
Teleport Type use Zero Y: Teleports the player on the Y axis.

Demo Scene

When playing the scene, make sure you have SteamVR running.

In the demo scene are examples using most of the actions with detailed FSM's and states as well as 3 custom made objects.





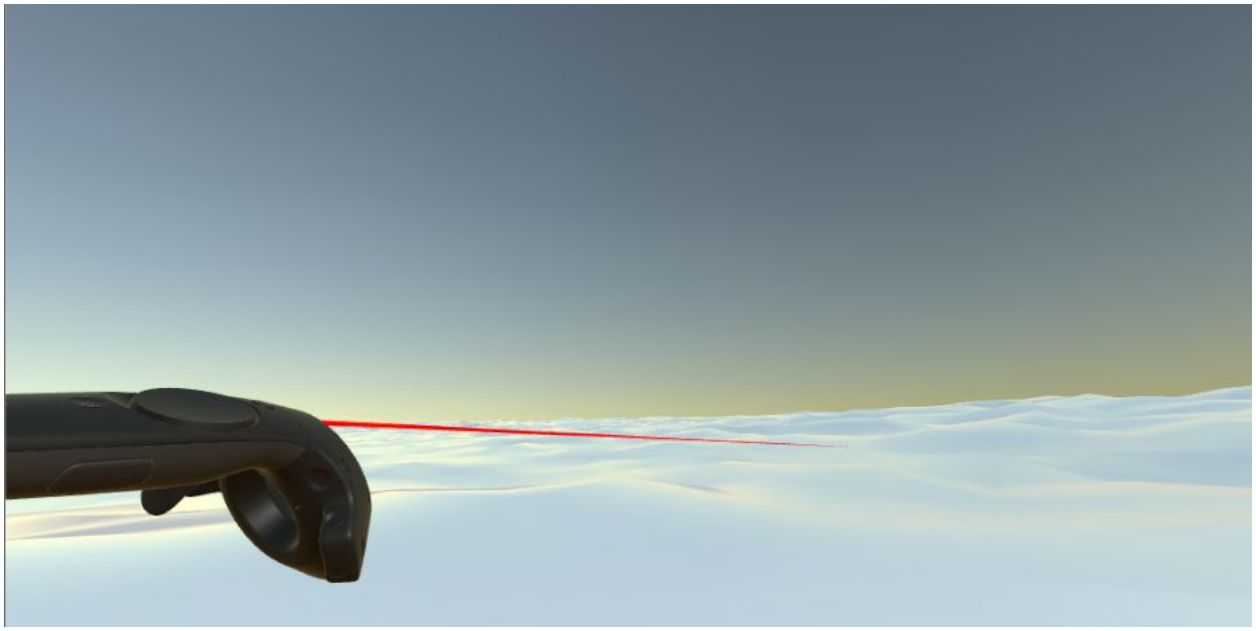
1. The Gun.
2. Boxing Machine.
3. Boxing Gloves.

*Objects are subject to a Creative Commons License.

The scene demonstrates the power of the actions with the Get Trigger used to grab and switch between controllers.
Get Controller Velocity to determine the speed of impact when using the boxing gloves on the machine.
Get Touchpad and Get Touchpad Axis to switch the material color of the sphere between red and blue using the Y axis.
Get Grip to reset the position of the sphere.
Get Gaze to change the material of the giant sphere when gaze is range.
Set Rumble to set the haptic feedback when firing the weapon, using the punching machine and switching between controllers.

The scene also features targets to shoot at when the gun is available, a giant sphere to *gaze* and a physics sphere to grab and throw.

Teleport Scene



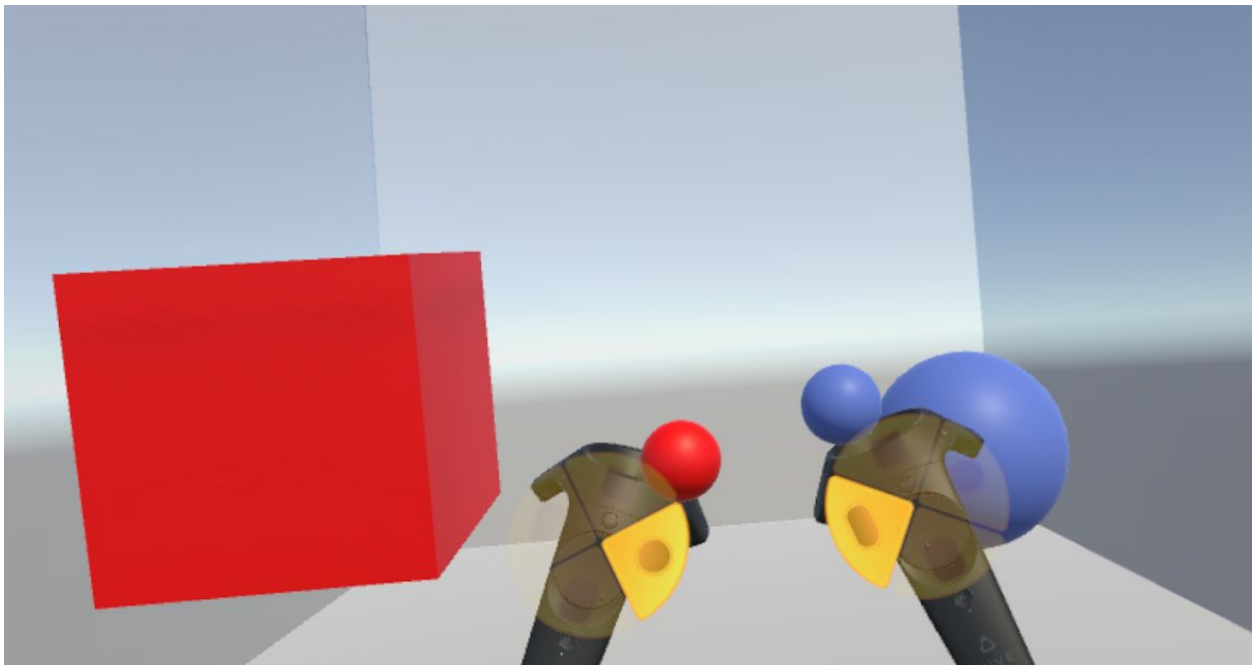
This scene contains the new actions, Teleport and Headset Fade.

Located in the scene is a Teleport Manager that uses the Right Controller to teleport across the terrain. It also uses the fade function to smooth the movement in the teleport function.

It is important to note: When using the Headset Fate action, the Camera (eye) shown in (Ref.4), must contain the SteamVR_Fade script attached. The example scene demonstrates this.

[ref.4]

Touchpad Axis



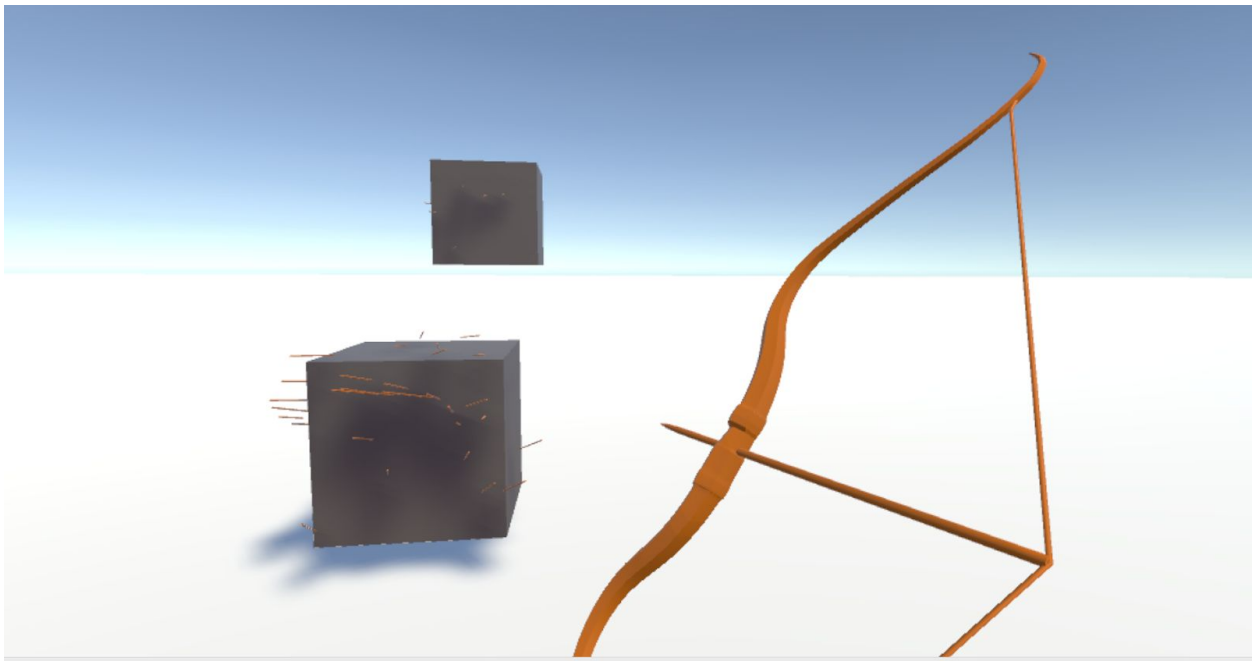
Scene demonstrates further use of the touchpad axis.

It uses four different axis points on the controller to move around the spherical object as well as set its material based on location.

Ideal use for switching weapons or an inventory selection.

Contains two unique FSMs that display the axis in action.

Archery



Scene demonstrates Bow and Arrow mechanics using Playmaker and Steam VR.

The main FSMs are located under *Bow* gameObject and *Arrow* gameObject.

The Set String state contains comments to further explain the process. Uses The position of both controllers to set the rotation and position of the string.

Bow gameObject is imported as an fbx with joints controlling the string mesh on the bow.

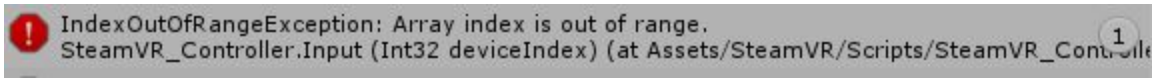
Bow is set on Left controller and Arrows are set on Right Controller. To implement personal bow, view Arrow Manager FSM as an example. Be aware of the forward pivot based on the 3D model which may affect most of the actions in the state. I.e. If your bow has a forward direction of 1, set the Z Transform Direction to 1.

Arrow has FSM on ArrowPrefab which enables gravity on launch and smooth follow based on speed and trajectory.

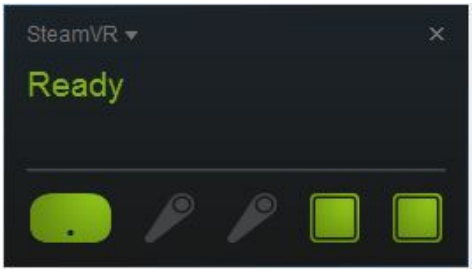
Errors

Errors that you may encounter in this package and how to resolve them.

Index OutOfRangeException: Array index is out of range.



If this error appears in the console, an action or script can not find the controller it is expecting to find. In most cases a controller is not found in the steamVR status.



Expression denotes a `type', where a `variable', `value' or `method group' was expected.

This may refer to the SetRumble.cs script. As coroutines in playmaker do not work in version 1.8.0, you must get beta version 1.8.1 to use the SetRumble.cs script.

The type or namespace name `FsmStateAction' could not be found.

If this error appears, Playmaker can not be found in the assets folder.

The type or namespace name `SteamVR_Controller' could not be found.

If this error appears, SteamVR can not be found in the assets folder.

Any questions or comments please contact [frametaleinfo@gmail.com]