Hydrargyrum Chroma-key Webcam-based Object-Tracking Solution

Documentation

Part. A

Installation

Asset Installation:

Step1: Download and Import the "Hydrargyrum Webcam-based Object Tracker" package into your project through the Unity Asset Store;

Step2: Navigate to the "Hydrargyrum Webcam-based Object Tracker \ Prefabs" Folder and drag the Prefab named "HgTracker" into your scene;

Part. B

Usage

Tracker Setup:

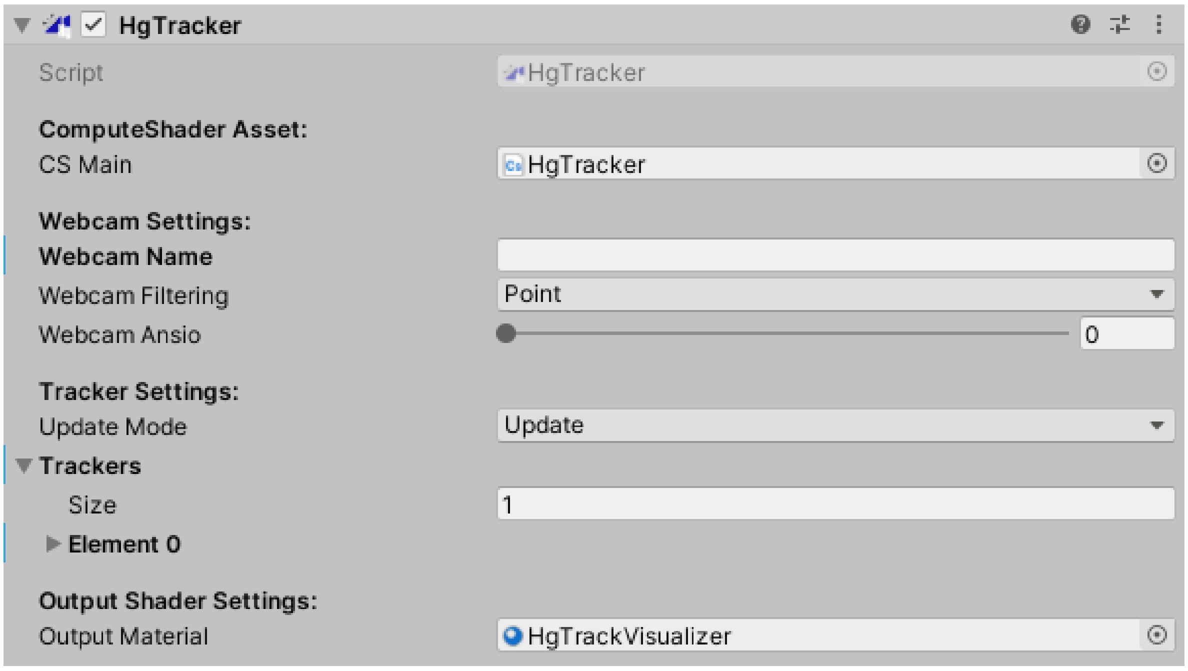
Step1: Click on the “HgTracker” Gameobject on your Hierarchy to open it for inspection in the Inspector;

Step2: Navigate to the “Hydrargyrum Webcam-based Object Tracker \ Shaders & Materials” Folder and drag the Material named “HgTrackVisualizer” onto the GameObject/ Mesh You want the tracker to be visualized on; (This is the Mesh on which the Output of our Tracker will be drawn onto;)

Tracker Setup:

Step3: To Setup your Trackers, Click on the "HgTracker" GameObject on your Hierarchy to inspect it in the Inspector;

Step4: Scroll in the Inspector window to find the "HgTracker" Script/Component assigned to this object;



HgTracker

Script: HgTracker

ComputeShader Asset:

CS Main: HgTracker

Webcam Settings:

Webcam Name:

Webcam Filtering: Point

Webcam Ansio: 0

Tracker Settings:

Update Mode: Update

Trackers

Size: 1

Element 0

Output Shader Settings:

Output Material: HgTrackVisualizer

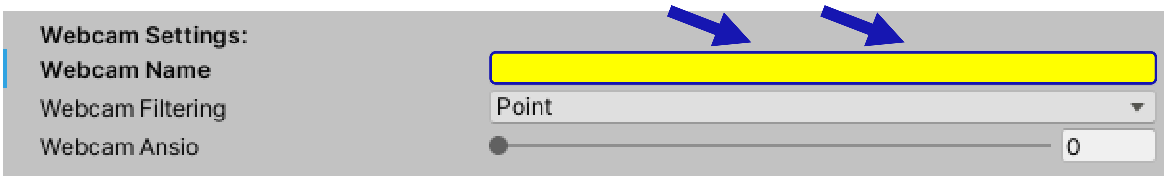
Tracker Setup:

Step5: To verify if your Webcam is being correctly detected and displayed, Press on the "Play" Button; Your webcam feed should be successfully visualized on the Mesh Selected before;

Possible Step5 Errors:

1. If you receive a "This Device is not equipped with a compatible Webcam!" Message in your Console, This means that either your device is not equipped with a webcam, or the Unity Editor is Unable to access your webcam feed;

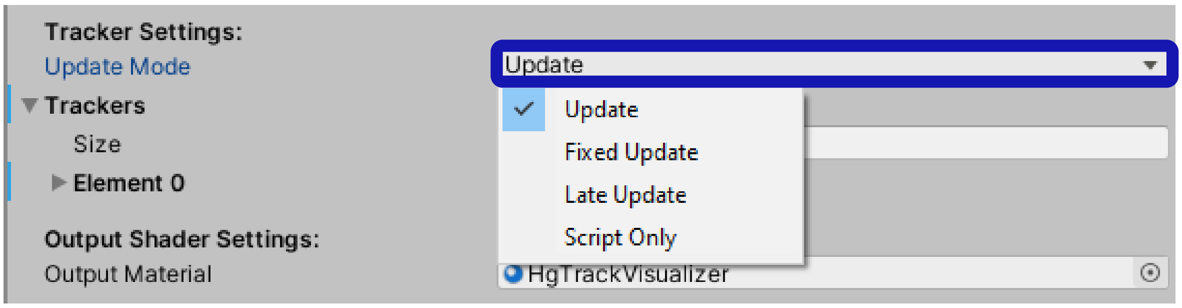
2. If your device is equipped with multiple webcam devices, Consider setting the "Webcam Name" field to either the system defined name, or the index of your desired webcam; Then repeat step 5;



Webcam Settings:	
Webcam Name	
Webcam Filtering	Point
Webcam Ansio	0

Tracker Setup:

Step6: To assure highest compatibility with all game types and genres, this asset allows the user to set the Updating loop on which the asset should be executed;
By setting the **"Update Method"** Variable available just below the **"Webcam Settings"** section, You can run the Tracker on the **Update()**, and **LateUpdate()** methods for games that run on a frame-based basis, or the **FixedUpdate()** method for games that run mainly on the Physics engine;
You can also select the **"ScriptOnly"** option which allows you to call the Tracker anytime You'd like thorough your own scripts;


Tracker Settings:
Update Mode: Update
Trackers
Size
Element 0
Output Shader Settings:
Output Material: HgTrackVisualizer

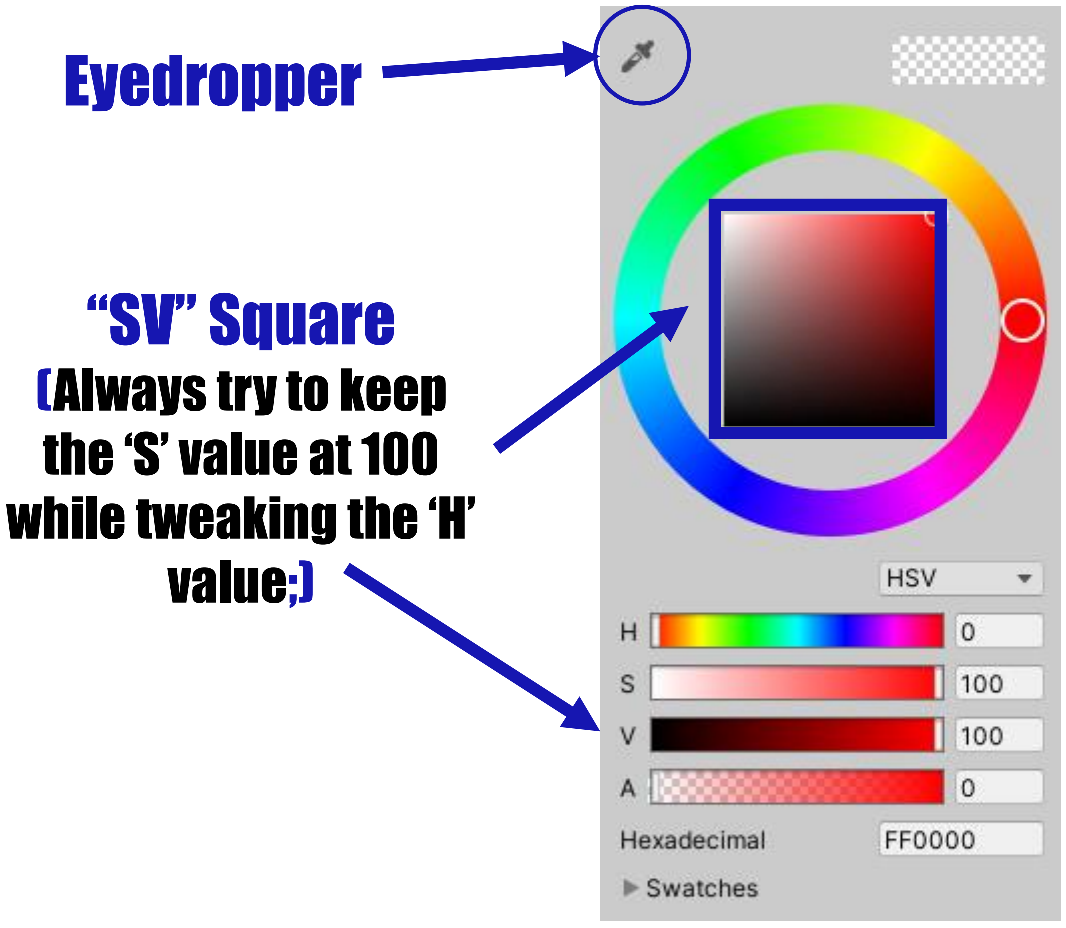
Tracker Setup:

Step7: Now click on the "Trackers" Dropdown to open our "Tracker Array" on the editor; To Inspect any of these trackers, Consider clicking on their title to open the for further inspection as a dropdown;

Step8: As you can see, Each one of our Extremely-Customizable trackers come bundled with a ton of settings; in the next few Page We'll be hopefully going over all these variables and their responsibilities:

Tracker Setup:

Luma: This is the main Color of the Object you want to track that this asset will be looking for in The webcam feed; to set this value, Simply sample the color of the object using the Editor's Eyedropper tool; If the Image becomes extremely noisy, or the trackers doesn't seem to be working properly, Try fiddling with the "SV" Square value in the editor unity you're satisfied; (Read-Write)


Eyedropper
"SV" Square
(Always try to keep the 'S' value at 100 while tweaking the 'H' value;)
HSV
H
S
V
A
Hexadecimal
Swatches

H: 0
S: 100
V: 100
A: 0
Hexadecimal: FF0000

Tracker Setup:

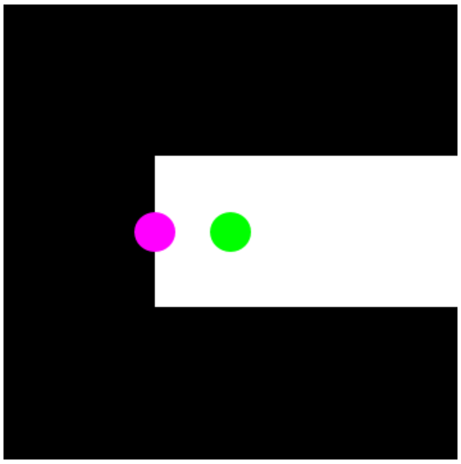
Tracking Mode: With only two options consisting of "Bounding Box" and "Median center", this setting is probably the most important of all;
Each of these setting implement a different tracking method to Track an object as described below:

Bounding Box: Finding the smallest 2D box where all pixels of the object can be stored insides; Improves quality but the smallest noise can reduce accuracy;

Median Center: Finds the average point of all pixels that an object obscures, Still has problems with quality, but noises cannot interfere with this method; (Read-Write)

Green Dot: Bounding Box

Pink Dot: Median center



Tracker Setup:

Threshold: This value sets the minimum blend value for a pixel, which should be detected as our object; reducing this value can improve noise on some occasions: (Default= .5f) (Read-Write)

Smoothing: This value sets the Smoothing factor that will be used for the Chroma-Keying process; Increasing this value can help with noise, but also might introduce extra unwanted noise to the image, as well; (Default= .1f) (Read-Write)

Sensitivity: This value sets the Sensitivity factor that will be used for the Chroma-Keying process; the higher you set this value, the Tracker will be less forgiving in case of color difference between the Luma and Object color; (Default= .1f) (Read-Write)

Tracker Setup:

RawShaderOutput: This value determines if the "Visualization" shader will actually draw the result of the Tracker to your screen or it'll ignore the drawing; (Read-Write)

ScreenSpacePositions: Determines if The Reported Tracker positions will be in Screen Space (Zero to One on X Y axis), or in Pixel space (Zero to Camera resolution on X Y axis) positions; (Read-Write)

Invert_X, Invert_Y: Determines if the Position reported by the Tracker will be inverted on any of the X or Y axis; Useful if you're having problems with inverting values manually; (Read-Write)

Tracker Setup:

IS_Visible: This value reports if the Tracker is able to See/Track the object you have selected; (Readonly)

Phase: Inspired by the Touch.Phase provided Unity.Input class, This variable also reports the Phase the tracker is in: (Readonly)

Phase.Began: This means that the Tracker has started tracking the Object in the current Cycle/ Frame;

Phase.Moved: This means that the Tracker is able to Detect the object and the object has moved since the last frame;

Phase.Stationary: This means that the Tracker is able to Detect the object However, the object has not moved since the last frame;

Phase.Ended: This means that either the Tracker is unable to see the object anymore, or the object has been moved out of the Webcam's view;

Tracker Setup:

RawPosition: This value reports the current raw position of the object in either Screen-Space or Pixel-space position; Note that this output can be jittery as it has not been smoothed at all;

DeltaPosition: This value reports the amount which the Tracker has moved since the Last Cycle/ Frame;

SmoothedPosition: This value reports the Smoothed position of the object in either Screen-Space or Pixel-space position; It's recommended to use this for trackers that use the "Median Center" method for tracking;

Tracker Setup:

OnTrackerEnter():

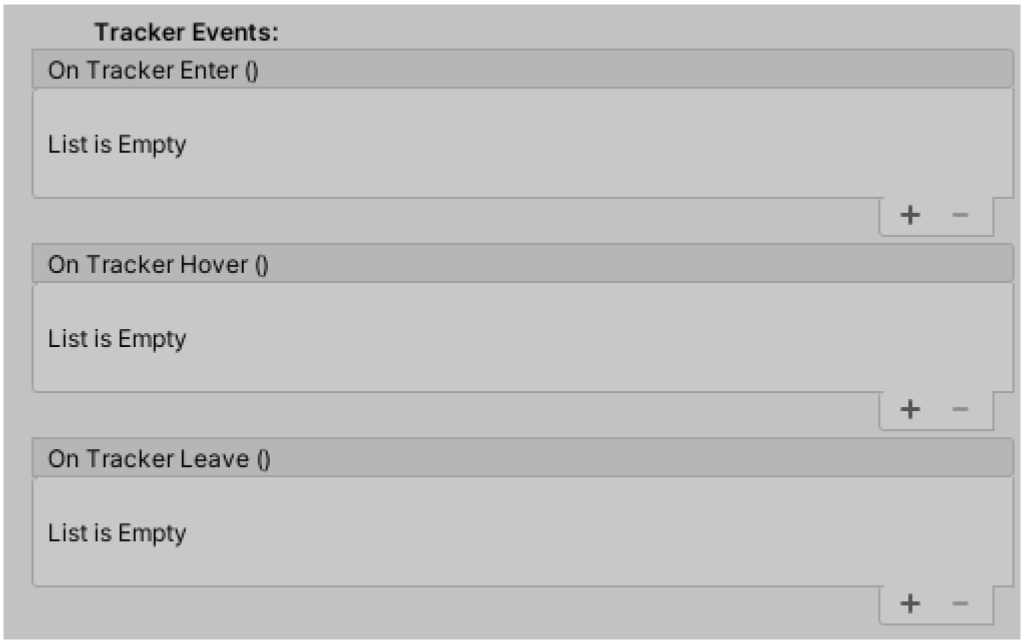
Called when the Tracker begins successfully tracking/ seeing an object;

OnTrackerHover():

Called when the Object that is being Tracked is still visible on the screen;

OnTrackerLeave():

Called when the Trackers is no more able to Track/ See the object as it moved out of the Webcam's view;


Tracker Events:
On Tracker Enter ()
List is Empty
On Tracker Hover ()
List is Empty
On Tracker Leave ()
List is Empty

Part. c

Coding Reference

CO

Coding Reference:

Unfortunately, As I am unable to register this asset as a global Input method in the Unity Game Engine, We'll have to use some dirty trick to access input variables from this asset; Here's how to do so:

Coding Reference: How to Access Variables through your Script:

Step1: Open The Script You want to access the variable through in the Code editor of your choice;

Step2: Make A reference to our Tracker object in the top of your Script: (This Object is named "HgTracker" So we can use a simple GameObject. Find() to find it)

```
HgTracker OBJTracker;
void Start()
{
OBJTracker = GameObject.Find("HgTracker")
.GetComponent<HgTracker>();
}
```

Coding Reference: How to Access Variables through your Script:

Step3: To access any of our variables, we simply have to read it from the "HgTracker" object we just referenced in our script:

```
HgTracker OBJTracker;
void Start()
{
OBJTracker = GameObject.Find("HgTracker")
.GetComponent<HgTracker>();
}

Void Update()
{
   if(OBJTracker.Trackers.Length > 0)
   {
        //Accessing Position
        Vector2 Position =
        OBJTracker.Trackers[0]
        .SmoothedPosition;
   }
}
```

End of Document:

Thank you for purchasing Hydrargyrum Game’s Chroma-key Webcam-based Object-Tracking Solution!

As a teen asset developer, It’s a huge achievement for me to sell my first asset on the Unity asset store;