OpenDisplayXR Development Status

By Andrew Hazelden <andrew@andrewhazelden.com>
2023-02-19

This document outlines the OpenDisplayXR project's initial research findings and lists the ongoing development efforts.

Over the last month, I spent a considerable amount of time reviewing the SteamVR, OpenVR, and OpenXR SDKs. I read countless pages of technical documentation to find reference material that covers the creation of virtual device drivers.

I also explored the NVIDIA InstantNGP NeRF toolset which added VR HMD connectivity support, created an initial Blackmagic Fusion Studio settings file for the 3Dconnexion SpaceMouse Enterprise input device, used NewTek NDI IP video streaming technology with software like (NDI Tools, Resolve + Nobe Display NDI, Unity3D + KlackNDI, and TouchDesigner).

I also read about OpenGL and DirectX based quadbuffer stereo rendering, and looked into embedded scripting APIs like LuaJIT.

Virtual Device Drivers

Included below is a detailed summary of HMD (Head Mounted Display) APIs, with information and links that discuss their support for virtual display drivers.

SteamVR SDK

Valve Software | SteamVR | Virtual Display

https://github.com/ValveSoftware/virtual_display

SteamVR supports a virtual device driver through the use of the "IVRVirtualDisplay" interface. This allows OpenVR to function with simulated HMDs. This makes it possible to directly capture the rendered output, and supports wireless transport of the final framebuffer data, too.

OpenVR SDK

OpenVR Driver Sample Code

OpenVR provides source code for a sample HMD driver and controller interface:
https://github.com/ValveSoftware/openvr/tree/master/samples/drivers/drivers/simplehmd

Valve Software | OpenVR | Driver Manifest Wiki Page

The Driver Manifest wiki page explains the syntax of a JSON file named "driver.vrdrivermanifest" that lives inside the driver folder:
https://github.com/ValveSoftware/openvr/wiki/DriverManifest

The JSON based "hmd_presence" attribute defines how the connection of an HMD device is detected. For a passive stereo 3D display, the value "*.*" would be used to return a true logic state back to SteamVR's VR_IsHmdPresent() function. This would indicate the display is enabled and available for use.

Valve Software | OpenVR | Driver Documentation Wiki Page

Driver Documentation wiki page summarizes the steps required to create a new OpenVR driver:
https://github.com/ValveSoftware/openvr/wiki/Driver-Documentation

Valve Software | OpenVR | Input Profiles Wiki Page

Input profiles are used by OpenVR compatible application software to bind user actions to the available hardware input devices.

https://github.com/ValveSoftware/openvr/wiki/Input-Profiles

An input profile supports the definition of a controller type, input source, binding mode, pose. Input profiles can represent inputs such as HMDs and hand controllers, console-style game-pad controllers, joysticks, trackpads, buttons, and haptics.

Valve Software | OpenVR | Issues | IVRVirtualDisplay

A detailed outline is provided on the OpenVR Issues page for the process of creating a custom device driver:
https://github.com/ValveSoftware/openvr/issues/507#issuecomment-450281723

Khronos OpenXR SDK

OpenXR uses extensions to add support for new devices, form factors, and view configurations. This allows vendors to expand upon the core OpenXR API functionality.

The Khronos Group allows non-members to create their custom OpenXR extensions that exist outside of the "Khronos IP Zone". This means no NDAs are required, and Khronos membership fees do not need to be paid by third-party developers.

The OpenXR SDK is accessible via GitHub:
https://github.com/KhronosGroup/OpenXR-SDK

One can register a custom OpenXR extension "author ID" via the OpenXR GitHub repo. An extension ends the author ID with "X" to mark it as an in-development item.

The OpenXR Working Group explains the extension creation process here:
https://registry.khronos.org/OpenXR/specs/1.0/extprocess.html

A description of the registration process is viewable in the wiki:
https://github.com/KhronosGroup/OpenXR-Docs/blob/401caaf0e02e5b9c994dde10e90e77fd5238af03/specification/sources/styleguide/extensions.adoc#registering-extensions

The official registry of OpenXR SDK extensions are visible in the following XML document:
https://github.com/KhronosGroup/OpenXR-SDK-Source/blob/main/specification/registry/xr.xml

An XML based OpenXR registry entry would look like:

```
<tag name="ODX" author="OpenDisplayXR Experimental" contact="Andrew Hazelden"/>
<extension name="XR_ODX_extension_500" number="500" type="instance" supported="disabled">
        <require>
        <enum value="1" name="XR_ODX_extension_500_SPEC_VERSION"/>
        <enum value="&amp;quot;XR_ODX_extension_500&amp;quot;" name="XR_ODX_extension_500_EXTENSION_NAME"/>
        </require>
</extension>
```

OpenXR Forums | Custom Viewport Configurations

A friend, Matthew Dougherty, from the NOAA (National Oceanic and Atmospheric Administration) communicated on the OpenXR community forum back in November 2021. He inquired about the possibility of creating custom OpenXR viewport configurations.

https://community.khronos.org/t/projection-caves-openxr-spec/108002

Matthew Dougherty had his questions answered by the Khronos Group member Ryan Pavlik, the developer of an open-source OpenXR rendering toolset called Monado:
https://monado.freedesktop.org/

NVIDIA CloudXR SDK

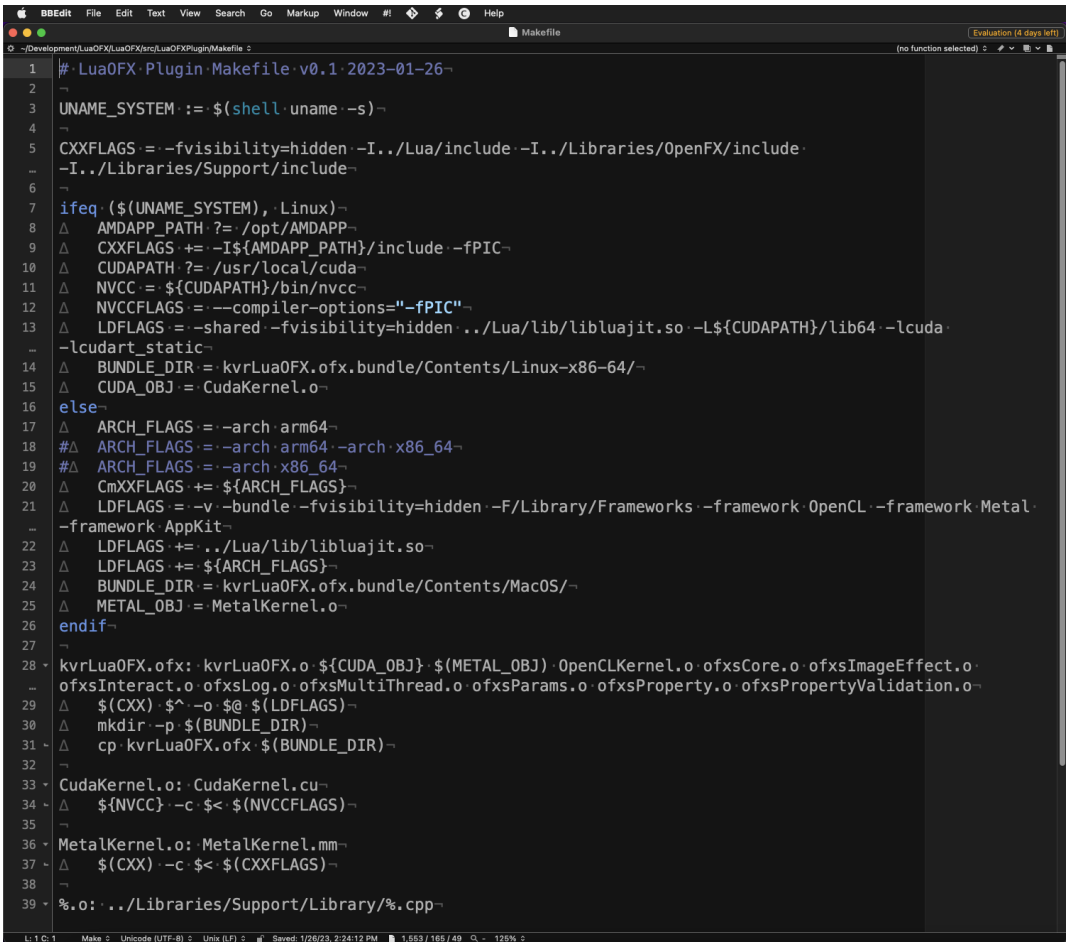
The NVIDIA CloudXR SDK allows a remote cloud-based workstation to host a live rendered stereoscopic 3D HMD session via a virtual device driver:
https://docs.nvidia.com/cloudxr-sdk/index.html

CloudXR is capable of working with SteamVR and OpenVR applications on a remote server. The CloudXR approach to left and right eye view stereo 3D rendering does not require quadbuffer stereo support on Amazon AWS EC2 compute instances.

Embeddable Scripting APIs

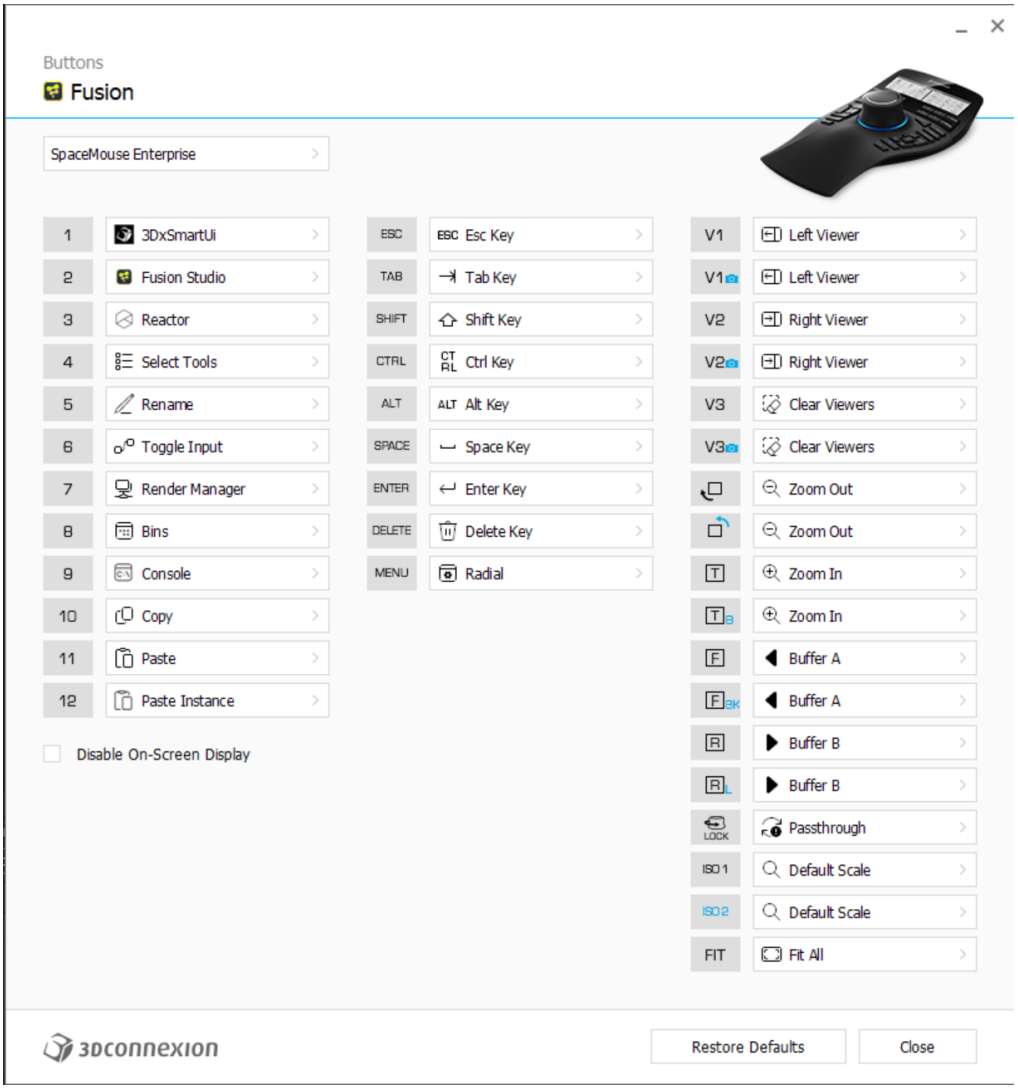
If a virtual device driver wants to support the use of arbitrary input devices in a "VR" like rendering environment, having a high-performance scripting API becomes useful. This allows the use of scripts to act as real-time input value remapping tools that can transform the meaning of raw HID (Human Interface Device) based input data.

In January I received technical assistance from a friend, Marcel Gandriau, a software developer based in France. Marcel helped prepare a makefile and resources that enabled me to compile and embed the LuaJIT interpreter as a shared library inside of a C/C++ based plugin API.



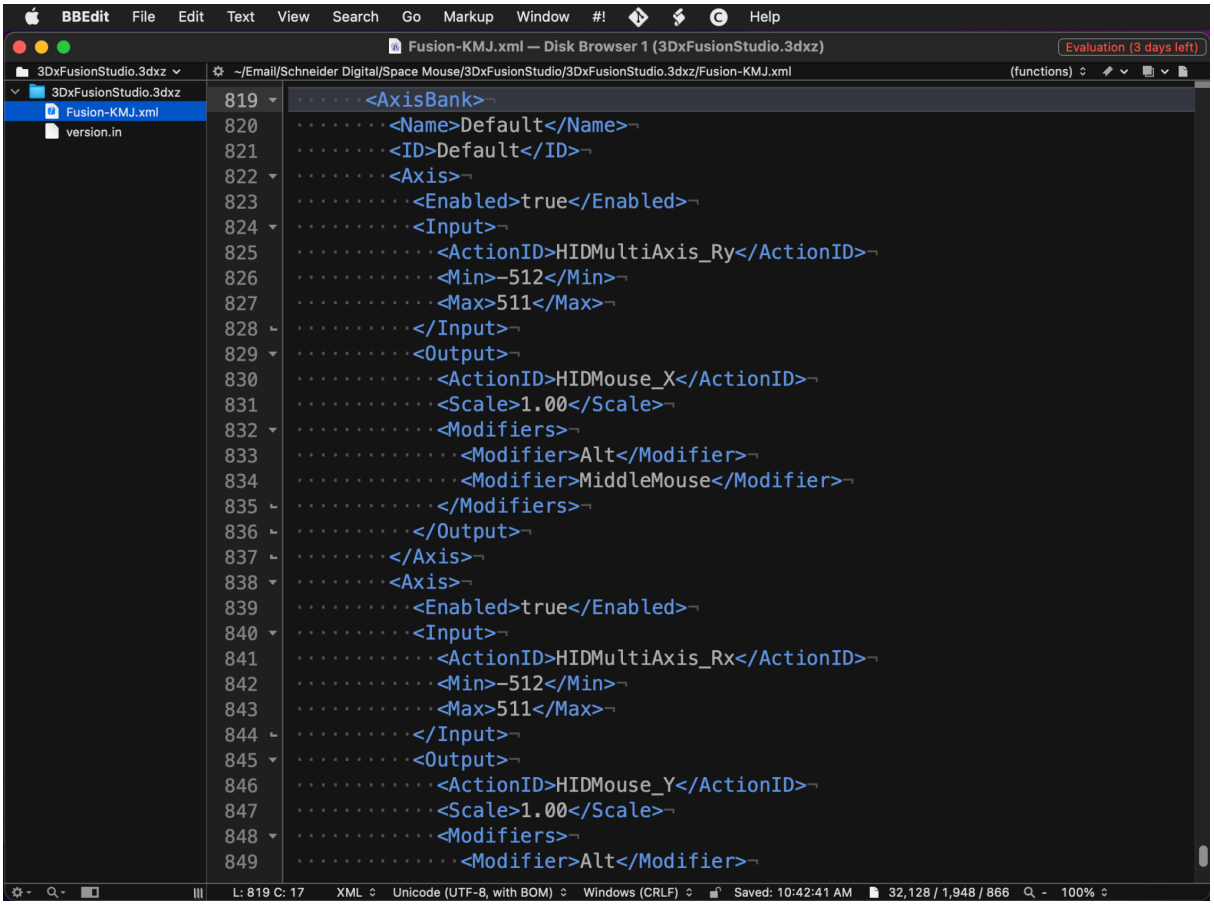
3DConnexion SpaceMouse Enterprise

In January I created a detailed 3DConnection settings file for Blackmagic Design's Fusion Studio compositing software. The configuration file allows one to use the SpaceMouse Enterprise to drive a majority of the tasks one would normally carry out in the node-based visual effects software.



Figuring out the syntax used in the 3DxSmartUI software's zipped XML based settings file took a lot of trial and error based experimentation. The challenge was that not all of the parameters can be configured using the control software's user interface on Windows, and those attributes are also not defined clearly in official end-user delivered documentation.

A lot of hunting on the 3DConnection user forum for posts by the site moderator "@jwick" was required before the exact details on the mouse input parameters for the "AxisBank" tags like "HIDMultiAxis_Ry", "HIDMultiAxis_Rx", and the modifier keys were discovered.



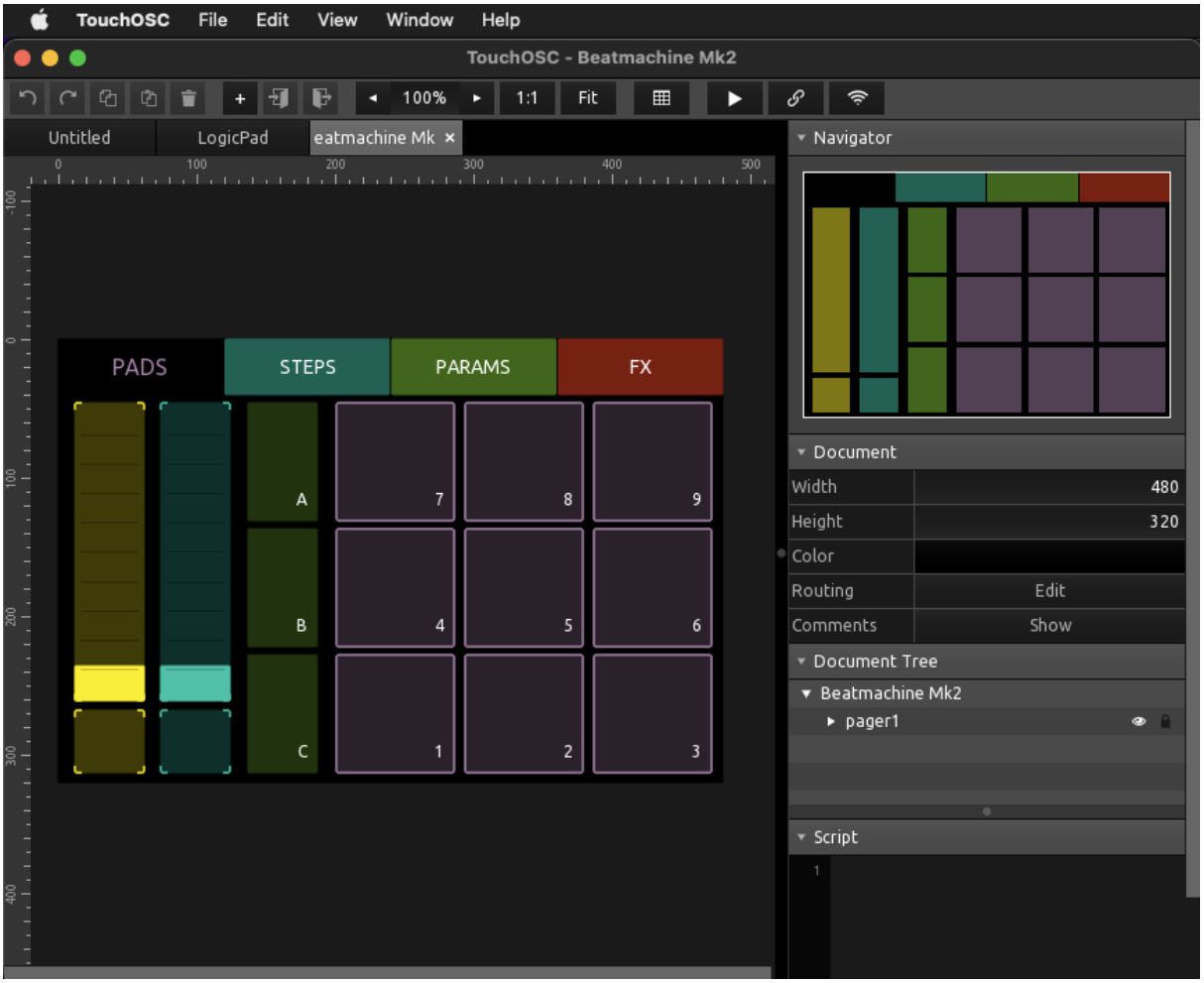
A significant number of customers have commented on the official 3DConnection forums that the current 3DxSmartUI's toolset makes it unnecessarily difficult to define the additional hotkey and mouse button modifiers one needs to activate when driving the viewport in 3rd party software packages.

After going through this experience myself, I think it is worth commenting that a lot of SpaceMouse products go underutilised by end-users after an initial purchase period. This hurts the product's uptake and reputation.

OSC Messaging API

OSC (OpenSoundControl) is a messaging protocol that transmits multimedia centric input data from devices like MIDI hardware over a network. This provides a convenient way to bridge live information from music synthesizers, haptic input devices, and computer systems to a remote host.

Software like TouchOSC makes it possible to create custom touchscreen based GUIs that can be used to drive OSC message passing.



OSC support is an attractive feature to implement in an HMD centric virtual device driver based input profile as OSC can transparently bridge access to a wide range of input hardware.

NewTek NDI IP Video Streaming

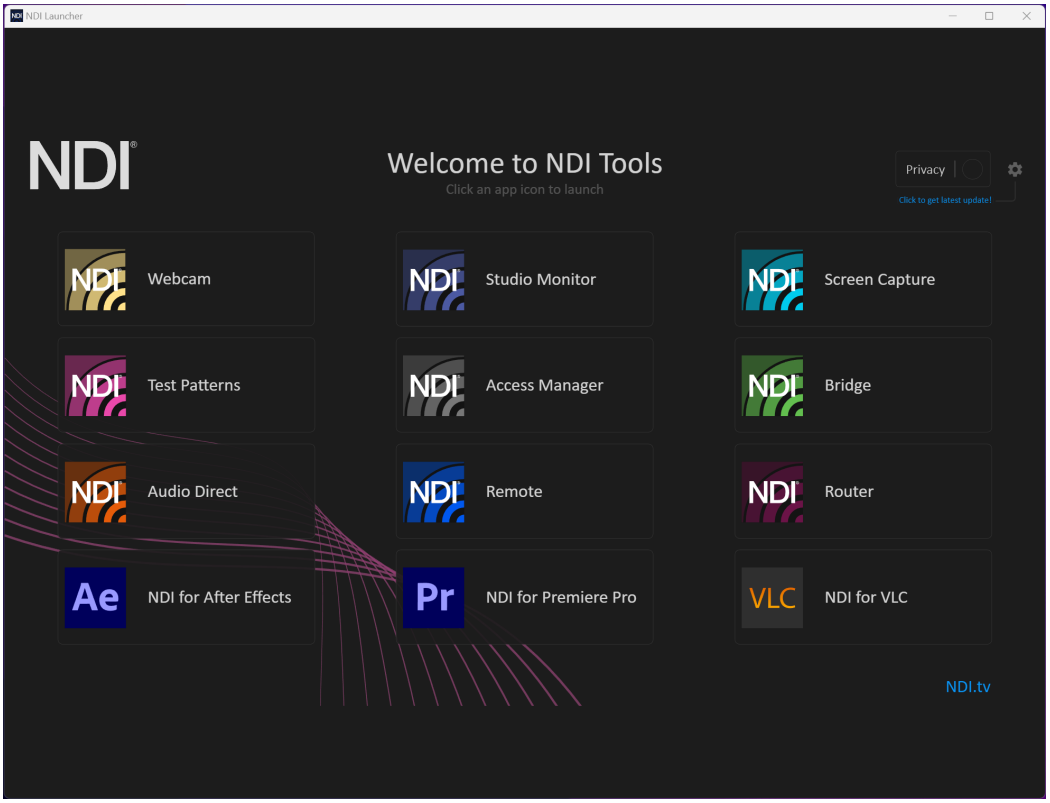
NewTek's [NDI (Network Device Interface) protocol] provides broadcast quality IP video streaming. NDI is available as both a software and hardware based implementation. NDI encoded video can be transported over conventional Ethernet, WiFi, or fibre based networking hardware.

NDI is an attractive IP based video transport technology for streaming immersive content to a multi-display panel centric hardware device like a Schneider Digital [smart VR-Wall].

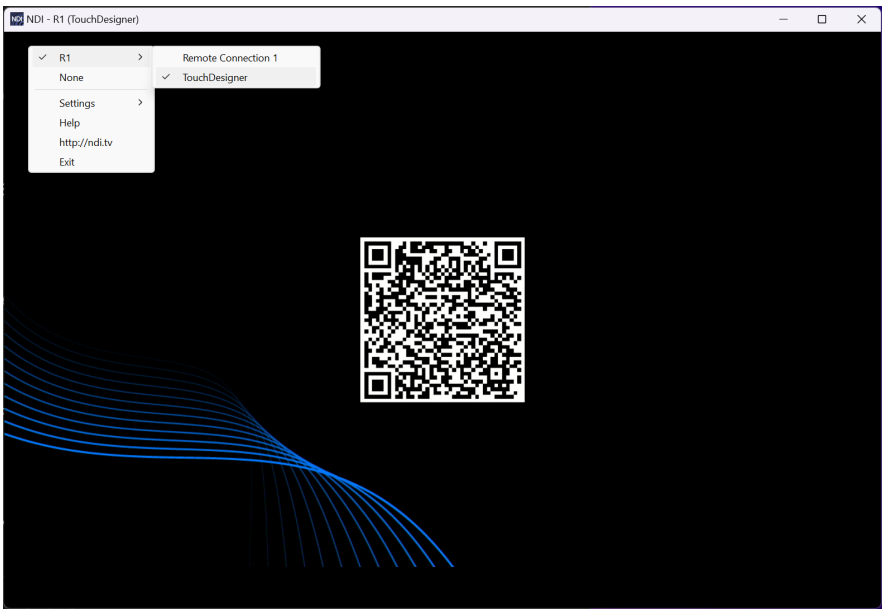
As part of my OpenDisplayXR development process I explored the following NDI implementations:

NDI Tools

The NDI Tools are available as a free download. It provides Windows and macOS users with a common set of base utilities to capture and playback NDI video streams on a conventional desktop system.

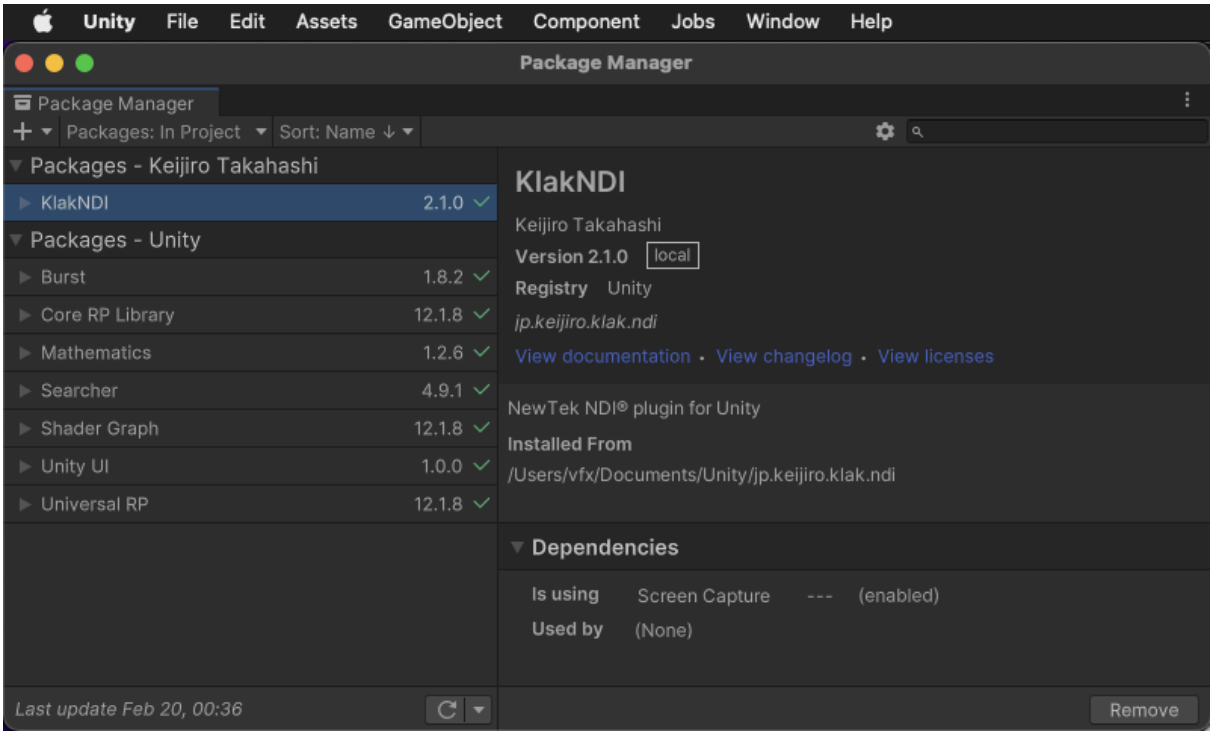


The "Video Monitor" program allows you to display NDI video feeds that are accessible on your local network subnet. Multiple concurrent NDI video streams can be played back on the same workstation. By right-clicking on the contextual menu icon you can select the exact video stream you would like to view.



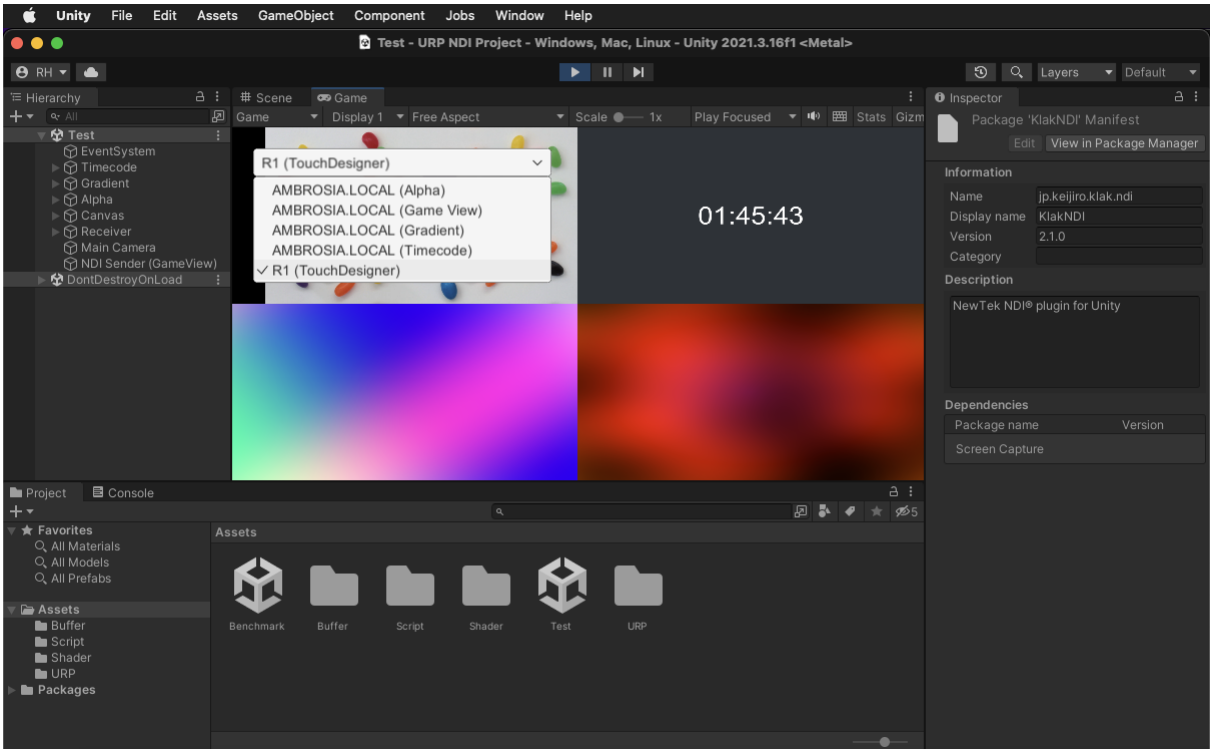
Unity3D Game Engine

Unity3D based game developers can install the free KlakNDI package. This package adds NDI sender and receiver features to Unity scenes.



The KlakNDI plugin makes it possible to transmit a game view in real-time as an NDI IP video stream, or you can capture an external NDI stream and apply it to a Unity based texture map.

The "test" example Unity scene file shows 4 concurrent NDI streams playing back at the same time. A contextual menu allows you to select an external NDI feed that is displayed in the first viewer cell.



NDI streaming approaches can be used to broadcast real-time rendered geospatial output from addons like the Cesium for Unity plugin. The Cesium GitHub page hosts the download for the unity integration addons.

Real-World 3D Geospatial Capability for Unity

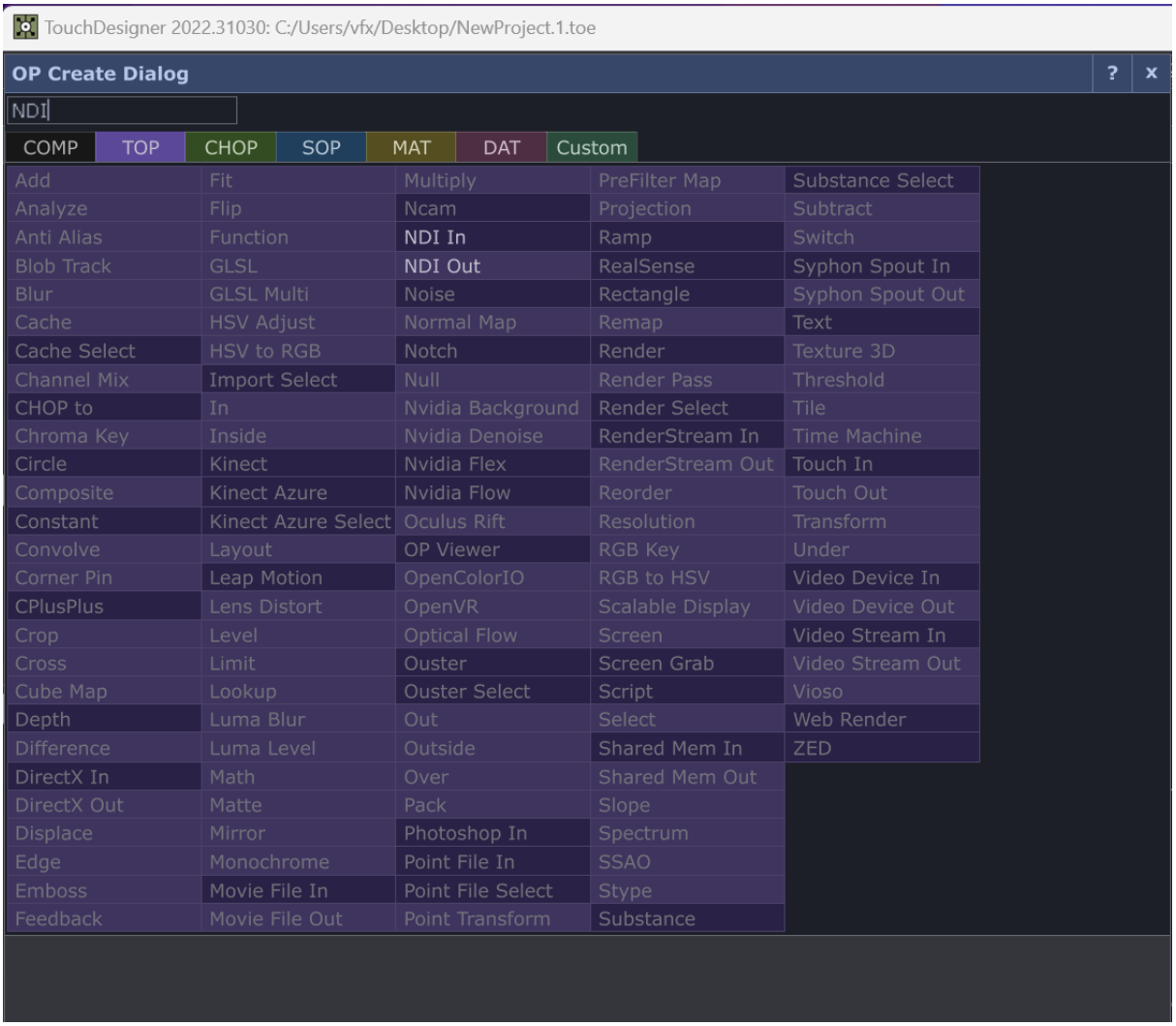
Built on open standards and APIs, Cesium for Unity combines the 3D geospatial capability of Cesium and 3D Tiles with the Unity ecosystem.

- A full-scale high-accuracy WGS84 globe for Unity
- Visualize massive high-resolution real-world photogrammetry and 3D geospatial content at runtime using 3D Tiles
- Free and open source visualization plugin
- Integrated with Unity's Game Objects, Components, Character Controllers, and more
- Optional subscription to Cesium ion for one-click access to global curated 3D content including terrain, imagery, 3D cities, and photogrammetry
- Support for multiple platforms including Windows, macOS, Android, and VR platforms such as Quest 2 and Quest Pro

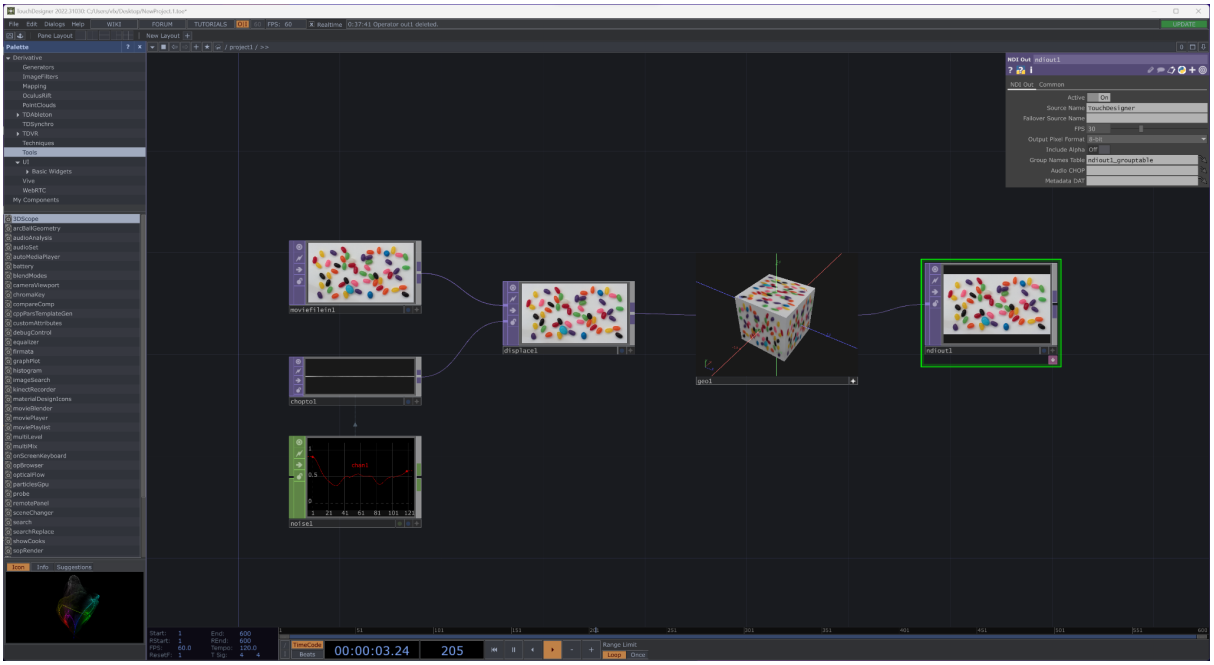
GET STARTED

Derivative TouchDesigner

The TouchDesigner real-time node based graphics environment allows you to work with NDI in/NDI out nodes with the same ease and elegance as movie/image sequences, traditional video capture devices, or HDMI display outputs.



"NDI in" and "NDI out" nodes can be added to a Touchdesigner project by pressing the TAB key. Then you can click on the TOP node and it is inserted into the node graph. You can then customize the name of the IP video stream in the node's settings.

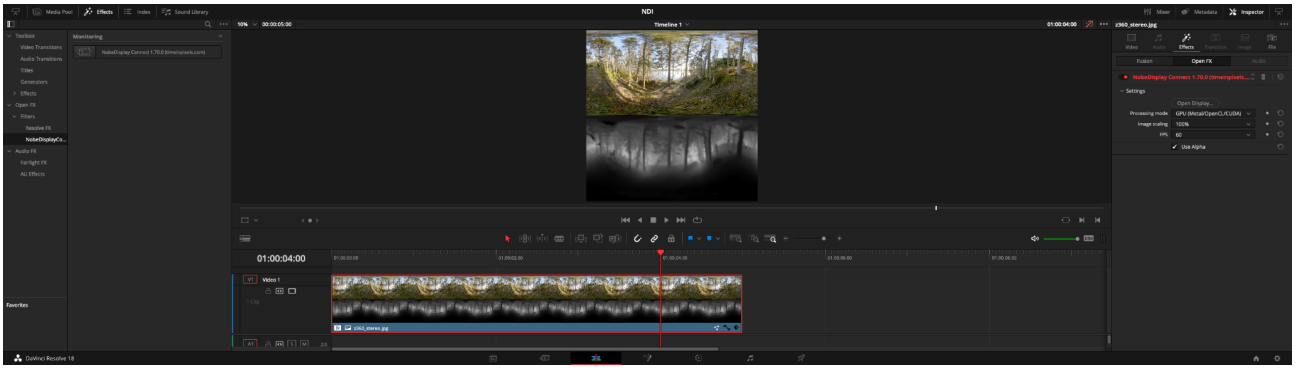


This image shows the default sample TouchDesigner project, with an NDI out node added to the end of the node graph.

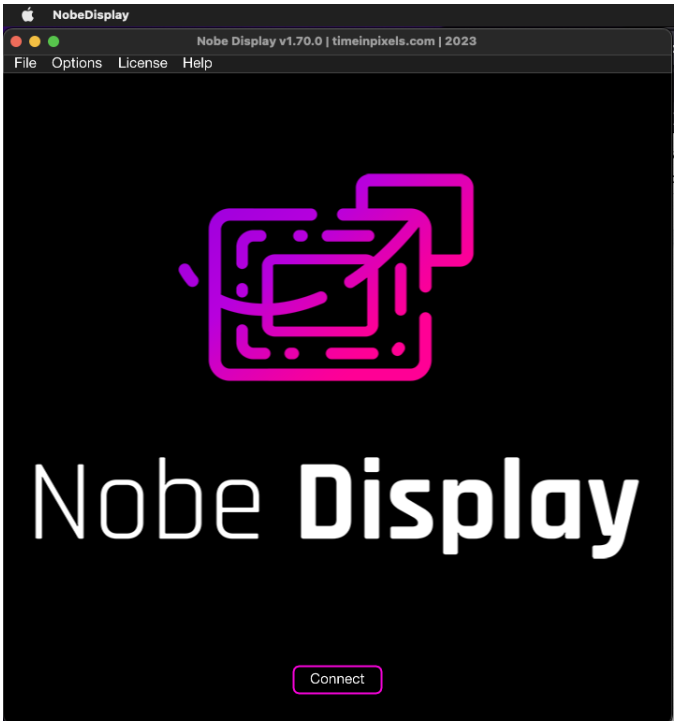
TouchDesigner's node graph provides a visual interface to apply post-processing effects to real-time media that passes through the composite. This makes it possible to reformat media, apply visual effects, render 3D meshes, perform warping or panoramic image stitching, or to process stereoscopic 3D footage on-the-fly.

DaVinci Resolve

Blackmagic Design's DaVinci Resolve Studio software is a video editing and color correction environment. Resolve's Edit and Color pages can be used inside an NDI based workflow with the addition of the OpenFX based Nobe Display NDI plugin that is dragged onto a timeline based item in the Edit page.

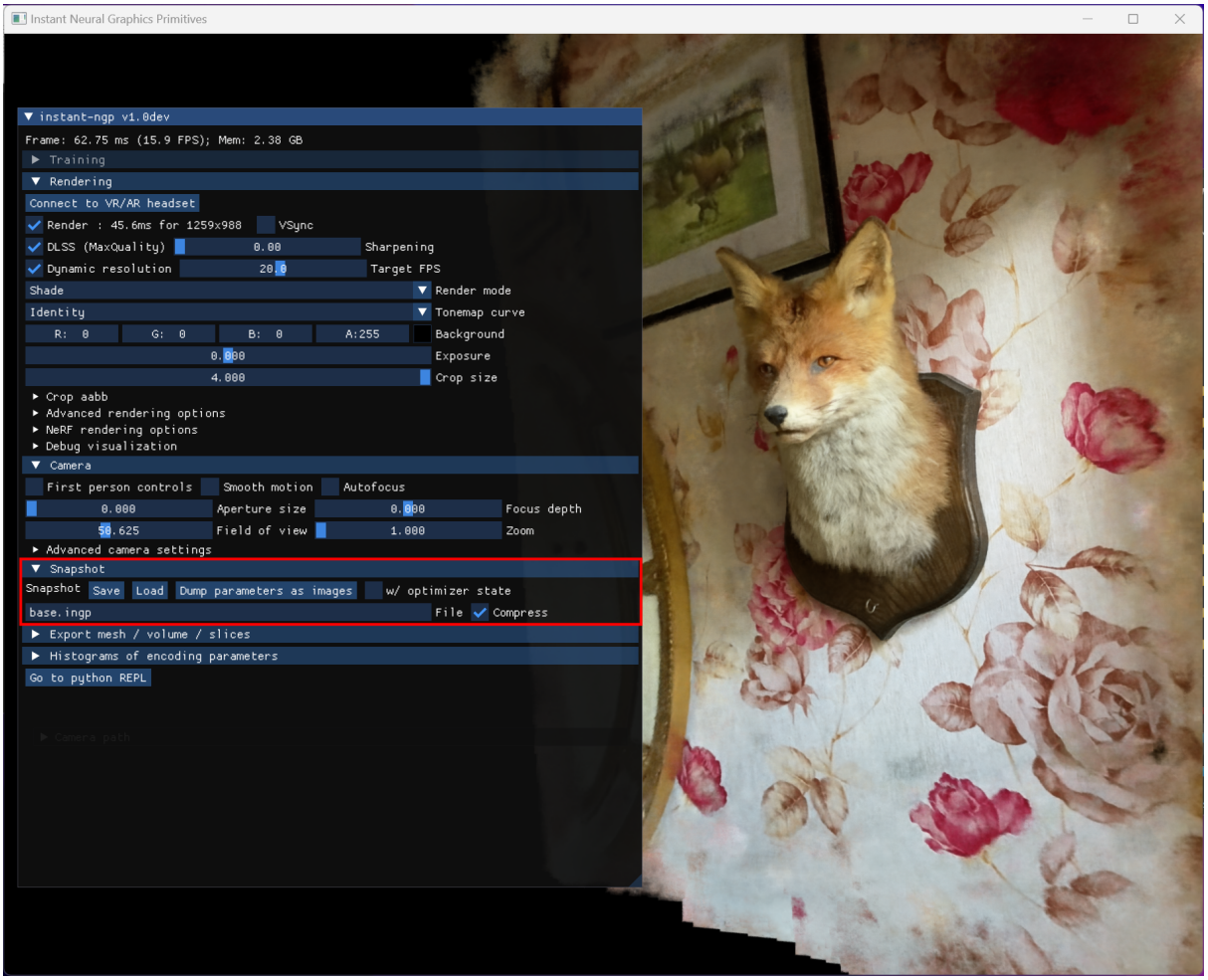


The NobeDisplay program runs in the background. It is used to connect to the DaVinci Resolve timeline based OpenFX plugin.



NVIDIA InstantNGP (NeRF)

NVIDIA's InstantNGP software has recently added support for stereoscopic 3D rendering of NeRF (Neural Radiance Field) scenes. This is possible via newly added HMD bindings. It is also possible to generate rectilinear, fisheye, and equirectangular image projection based offline movie renderings from the toolset.



A recent update has made it possible for InstantNGP projects to be imported and exported using a single file based redistributable format called an ".ingp" file. This document format allows users to deliver compact and efficiently compressed 3D explorable NeRF datasets. One can drag and drop an .ingp file or a "transform.json" file from an Explorer based desktop folder into the InstantNGP window and it will be instantly opened.

Neural radiance field rendering approaches work well for reconstructing and rendering scenes that contain high amounts of vegetation with thin branches and leaf structure. It also excels with outdoor environments that have details like reflective & refractive water, and for architectural use cases with office towers that have glass windows that are both transparent, and also reflective with a mirror like finish at certain viewing angles.