ANSEL INTEGRATION GUIDE

1. SETTING UP

1.1 MACHINE CONFIGURATION

In order to use Ansel you need

- Windows PC with Windows 7 (32-bit or 64-bit) or newer
- GeForce GTX 600 series or newer
- Ansel-ready display driver. Any NVIDIA display driver of version 368.81 or higher will meet this requirement
- A game using DirectX 11 that has integrated the Ansel SDK

NOTE:

```
- Support for DirectX 9, 12 and Vulkan is coming soon.
- We currently do not support Ansel for the following NVIDIA GPU / Display configurations
    * Optimus / Hybrid
    * Surround Support
```

This new version of the Ansel SDK whitelists applications for Ansel automatically, so there is no need to use `NvCameraEnable.exe` tool supplied with Ansel anymore.

1.2 UNITY PROJECT CONFIGURATION

To enable Ansel in your projects you need to import the Ansel package which you can download from the Asset Store. Once package is downloaded from the store simply select Assets->Import Package->Custom Package from the menu in the Unity editor in order to import it.

2. INTEGRATING PLUGIN WITH YOUR GAME

2.1 ADDING ANSEL SCRIPT TO YOUR SCENE

Ansel C# script needs to be attached to the main camera in order for Ansel to work properly. Once that is done the following properties will be exposed under Ansel component in the Unity editor:

```
// The speed at which camera moves in the world
[SerializeField] publicfloat TranslationalSpeedInWorldUnitsPerSecond = 5.0f;
// The speed at which camera rotates
[SerializeField] publicfloat RotationalSpeedInDegreesPerSecond = 45.0f;
// How many frames it takes for camera update to be reflected in a rendered frame
[SerializeField] publicuint CaptureLatency = 0 ;
// How many frames we must wait for a new frame to settle
[SerializeField] publicuint CaptureSettleLatency = 0 ;
// Game scale, the size of a world unit measured in meters
[SerializeField] publicfloat MetersInWorldUnit = 1.0f;
// Allows a filter/effect to remain active when the Ansel session is not active
[SerializeField] publicbool IsFilterOutsideSessionAllowed = false;
```

You can adjust these properties as needed in your projects.

NOTE:

```
By default Ansel script sets up left handed coordinate system which is a default
coordinate system in Unity. In an unlikely scenario where your project is using
different coordinate system please edit configuration data in NVIDIA.Ansel.Start
method and adjust right, up and forward vectors as needed.
```

2.2 DETECTING IF ANSEL IS AVAILABLE

NVIDIA.Ansel.IsAvailable property should be used to check if Ansel is available or not on the running system. This is useful in cases where for example UI updates are needed based on Ansel availability.

2.3 CONFIGURING ANSEL SESSION

The time period from when a player successfully starts Ansel and until Ansel is stopped is called a session. A session is collaboratively started and operated between the game and Ansel. When a player requests a session start (for example by pressing ALT+F2) Ansel.IsSessionActive property is set to true. It is however expected that Ansel cannot always be activated. For instance, let's say that a game uses in-engine movie sequences. Ansel could certainly be used to take regular and highres screenshots during those sequences. However, the game developers may wish to prohibit any player controlled camera movement or 360 captures during those sequences since they could expose geometry that was never built because the sequences have been carefully orchestrated. This is how that could be achieved:

```
Ansel.SessionData session = new Ansel.SessionData();
session.isAnselAllowed = true; // set to false to completely disable Ansel
session.isFovChangeAllowed = true;
session.isHighresAllowed = true;
session.isPauseAllowed = true;
session.isRotationAllowed = false;
session.isTranslationAllowed = false;
session.is360StereoAllowed = false;
session.is360MonoAllowed = false;
Ansel.ConfigureSession(session);
```

During an Ansel session the game:

- Must stop drawing UI and HUD elements on the screen, including mouse cursor
- Should not modify camera position, orientation, FOV or any other property which affects camera viewport
- Should not act on any input from mouse and keyboard and must not act on any input from gamepads

Here is one example

```
public class SplineInterpolator : MonoBehaviour
{
    ...
    voidUpdate()
    {
        if(NVIDIA.Ansel.IsSessionActive)
        {
            // Skip an update during Ansel session
            return;
        }
        // Advance current time and do some work
        mCurrentTime += Time.deltaTime;
    }
};
```

2.4 ANSEL CAPTURE AND IMAGE EFFECTS

When Ansel session is active and user starts a capture NVIDIA.Ansel.IsCaptureActive property is set to true. Game should check this flag in order to disable any image effects which may interfere with the capture session (for example if game is using motion blur it needs to disable it during capture).

Note:

```
This property is only set to true when multi-part capture is in progress (super resolution, 360).
In contrast, when user is taking a regular screen-shot this property will return false and the
game should apply effects just as it does during the regular game execution.
```

2.5 ANSEL CUSTOM UI

It is possible to add custom checkbox or slider into the Ansel UI, it is a good way to add a control over game specific settings like day/night switch or bloom intensity. Look for `Custom Control example` code line in Ansel.cs file as an example how to use user control functions.

3. TAKING PICTURES WITH ANSEL

3.1 ACTIVATING AND DEACTIVATING ANSEL

Players can start/stop Ansel session by pressing ALT+F2.

3.2 MOVING THE CAMERA

The camera can be moved via keys WASD and XZ for up/down. The camera can also be moved with the left stick on a gamepad and trigger buttons for up/down. Movement can be accelerated by holding SHIFT on keyboard or depressing right stick on gamepad.

3.3 ROTATING THE CAMERA

The yaw and pitch of the camera is directly controlled by mouse or right stick on gamepad. The roll of the camera is controlled via the user interface Roll slider.

3.4 APPLYING A FILTER

A number of filters can be selected via the Filter slider. Some filters, like Custom', have additional settings that can be used to adjust the filter even further.

3.5 TAKING A PICTURE

Ansel offers the following capture types (selected via the Capture type slider):

- Screenshot
- Highres
- 360
- Stereo
- 360 Stereo

Not all of these capture types may be available since it depends on the game integration and the current session (see sections 2.1 and 2.2). Once a type has been chosen the picture is taken by pressing Snap button. Some pictures may take significant time to produce, especially highres shots of large dimensions. If the game uses streaming the streaming performance may be affected when shots involving many parts are being stitched together.

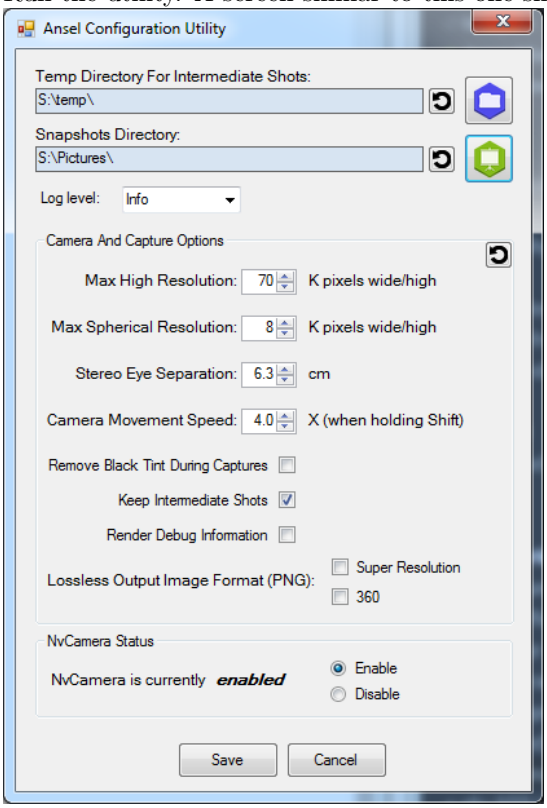
NOTE:

```
Not all filters are valid with multipart Capture types (360 and Highres).
You may therefore see filters (or aspects of a filter) removed in the final picture.
```

4. TROUBLESHOOTING AND DEBUGGING COMMON PROBLEMS

In this section we collected commonly occurring problems we've seen while integrating Ansel with games. This section can hopefully help you resolve a problem or two. It is generally useful to be able to inspect the individual shot tiles that are captured when generating pictures that require multiple shots. Locate the NvCameraConfiguration.exe utility. It can be found inside Program Files\NVIDIA Corporation\Ansel\Tools.

Run the utility. A screen similar to this one should appear:



Check the `Keep Intermediate Shots` option so that you can inspect the individual tiles. You can also pick a different location to store the tiles by changing the `Temp Directory for Intermediate Shots`.

4.1 ARTIFACTS IN MULTIPART SHOTS

This is where we cover the most common errors we've seen while capturing multipart shots in games.

4.1.1 Ghosting everywhere in final picture

Most often this is the result of incorrect field of view being submitted to Ansel - or error made on conversion or usage of value coming back from Ansel. It is recommended that you match the field of view type between game and Ansel to avoid any conversion mistakes. See section 2.1 on how you can configure Ansel to use the game's field of view.

4.1.2 Screen space reflections fade out with increased Highres capture resolution

There is unfortunately no workaround for this problem, it is a limitation of the capture method used.

4.1.3 It's all a blur

Motion blur needs to be disabled during multipart capture.

4.1.4 Streaky reflections

The reason may be that the projection offset and reduced field of view employed by the highres capture method is not being accounted for in the game's reflection code path.

4.2 THE VIEW OF THE WORLD "POPS" WHEN ENTERING AND EXITING ANSEL MODE

This is typically due to incorrect field of view being passed on the first frame or due to a screen space effect being disabled when Ansel mode is activated. For the latter it is preferred to deactivate troublesome effects only during multipart captures (via the capture callback).

4.3 ANSEL CANNOT BE ACTIVATED

Please consult the configuration requirements for Ansel listed in section 1.2. You can verify that Ansel is enabled by using the `NvCameraConfiguration.exe` utility that was introduced at the beginning of this chapter. You can also disable whitelisting as outlined in section 1.2. Finally, setting a breakpoint on the startSessionCallback can be used to verify that Ansel is trying to start a session.

Also it is important for Ansel to be loaded before D3DDevice was created. So the `AnselSDK32.dll/AnselSDK64.dll` must be loaded on the very start of an application, which is not the case by default in Unity. To achieve that we use `isPreloaded` flag in dll's meta file. It should be set when you import the package, but double check it by openning `AnselSDK64.dll.meta` and looking for `isPreloaded: 1`. Unity sets this flag automatically if dll name start with `audioplugin` (Audio Plugins have the same requirement), other than that this is not mentioned anywhere in Unity docs.

4.4 CAMERA ROTATION OR MOVEMENT IS INCORRECT

Incorrect rotation is best observed with a gamepad - i.e. pushing the joystick left doesn't rotate the view towards the left or pushing the joystick up doesn't rotate the view up. Incorrect movement can be verified with either keyboard or gamepad.
This problem is usually rooted in incorrect axes provided for right, up, and down directions. See note in section 2.1.

4.5 CAMERA ROTATION OR MOVEMENT IS TOO SLOW / TOO FAST

The speed for rotation is set via the RotationalSpeedInDegreesPerSecond property. The default value is 45 degrees/second.
The speed for movement is set via the TranslationalSpeedInWorldUnitsPerSecond property. The default value is 1 world unit/second.

4.6 ALL ANSEL CAPTURES PRODUCE BLACK IMAGES

This is usually caused by an unsupported backbuffer format. Ansel currently supports the following backbuffer formats:

Supported formats

```
DXGI_FORMAT_R8G8B8A8_UNORM
DXGI_FORMAT_R8G8B8A8_UNORM_SRGB
DXGI_FORMAT_B8G8R8A8_UNORM
DXGI_FORMAT_B8G8R8A8_UNORM_SRGB
DXGI_FORMAT_R10G10B10A2_UNORM
```

Additionally, multisampling is supported for all the above formats. If your game is using a format that is not on the list Ansel
will produce images with zero for every pixel - i.e. black.

APPENDIX A

Notice

The information provided in this specification is believed to be accurate and reliable as of the date provided. However, NVIDIA Corporation ("NVIDIA") does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This publication supersedes and replaces all other specifications for the product that may have been previously supplied.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and other changes to this specification, at any time and/or to discontinue any product or service without notice. Customer should obtain the latest relevant specification before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer. NVIDIA hereby expressly objects to applying any customer general terms and conditions with regard to the purchase of the NVIDIA product referenced in this specification.

NVIDIA products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on these specifications will be suitable for any specified use without further testing or modification. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to ensure the product is suitable and fit for the application planned by customer and to do the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this specification. NVIDIA does not accept any liability related to any default, damage, costs or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this specification, or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this specification. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA. Reproduction of information in this specification is permissible only if reproduction is approved by NVIDIA in writing, is reproduced without alteration, and is accompanied by all associated conditions, limitations, and notices.

ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the NVIDIA terms and conditions of sale for the product.

VESA DisplayPort

DisplayPort and DisplayPort Compliance Logo, DisplayPort Compliance Logo for Dual-mode Sources, and DisplayPort Compliance Logo for Active Cables are trademarks owned by the Video Electronics Standards Association in the United States and other countries.

HDMI

HDMI, the HDMI logo, and High-Definition Multimedia Interface are trademarks or registered trademarks of HDMI Licensing LLC.

ROVI Compliance Statement

NVIDIA Products that support Rovi Corporation's Revision 7.1.L1 Anti-Copy Process (ACP) encoding technology can only be sold or distributed to buyers with a valid and existing authorization from ROVI to purchase and incorporate the device into buyer's products.

This device is protected by U.S. patent numbers 6,516,132; 5,583,936; 6,836,549; 7,050,698; and 7,492,896 and other intellectual property rights. The use of ROVI Corporation's copy protection technology in the device must be authorized by ROVI Corporation and is intended for home and other limited pay-per-view uses only, unless otherwise authorized in writing by ROVI Corporation. Reverse engineering or disassembly is prohibited.

OpenCL

OpenCL is a trademark of Apple Inc. used under license to the Khronos Group Inc.

Trademarks

NVIDIA and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

www.nvidia.com