FidelityFX Super Resolution 2.1

Unreal Engine Plugin

The AMD FidelityFX Super Resolution 2.1 (FSR2) plugin for Unreal Engine is an open source, high-quality solution for producing high resolution frames from lower resolution inputs.

The package also includes the FSR2MovieRenderPipeline plugin which enables use of FSR2 to accelerate rendering when using the Unreal Movie Render Queue.

FSR2's Recommendations & Known Issues *must* be understood and addressed per-project to achieve optimal quality. See the respective section in this guide.

Contents

Known Issues

World-Position-Offset on Static Objects

Static objects that use a material with World-Position-Offset do not generate motion vectors. This affects FSR2's ability to correctly upscale such materials and results in blurring/ghosting of the affected objects.

One option is to enable the console-variables *'r.BasePassOutputsVelocity'* & *'r.BasePassForceOutputsVelocity'* then disable the console variable *'r.SelectiveBasePassOutputs'* which renders velocity for all objects during the base-pass.

For cases where this is too expensive and changing content is infeasible follow the installation instructions to install the correct version of the 'ImproveStaticWPO' engine patch to avoid this issue or ensure that all objects using a WPO material are set to Stationary or Movable. When changing the content grass layers attached to Landscape actors require a new console variable *'r.FidelityFX.FSR2.ForceLandscapeHISMMobility'* be enabled, this will change their mobility to Stationary so they render motion vectors.

Animated Opaque Materials

Animated opaque materials which don't generate motion vectors for the animated content, such as in-world video screens, may also blur or ghost when obscured by static geometry. This can be reduced by ensuring such materials write into the Reactive Mask generated in the plugin.

For the Deferred Renderer this can be achieved by selecting a Shading Model for the *'Reactive Shading Model*' option in the FSR2 section of the project settings or using the *'r.FidelityFX.FSR2.ReactiveMaskForceReactiveMaterialValue'* console variable. Materials that use this Shading Model will be treated as reactive. This means that when this Shading Model is selected in the Material Editor that material will write either the value of the CustomData0.x channel, when exposed by the Shading Model, or the value of the console-variable *'r.FidelityFX.FSR2.ReactiveMaskForceReactiveMaterialValue'* provided it is set to a value greater than 0. Using *'r.FidelityFX.FSR2.ReactiveMaskForceReactiveMaterialValue'* overrides any material specific reactivity specified in the CustomData0.x channel.

Where selecting an existing shading model is unsuitable follow the installation instructions to install the correct version of the 'LitReactiveShadingModel' engine patch which adds a new *'LitReactive'* shading model that can be used specifically for this purpose.

This problem cannot currently be resolved in the Forward Renderer where the Shading Model cannot be determined by the plugin.

PIX & RenderDoc Issues

Then the native FSR2 backends are used within the FSR2 plugin graphics capture tools such as PIX & RenderDoc may be less stable. Disabling the native backend (*'r.FidelityFX.FSR2.UseNativeDX12'* or *'r.FidelityFX.FSR2.UseNativeVulkan'*) and using the RHI-based backend may allow the capture tool to replay captures more reliably.

UE4 Post-Processing Volume Screen-Percentage Overrides

UE4 developers should be aware that FSR2's quality mode (when enabled) will determine the screen-percentage and ignores any screen-percentage override present in a post-processing volume. This will result in different visual and performance results.

Recommendations

Optimizing Translucency Appearance

While the default settings for the FSR2 Reactive Mask should generate reasonable results it is important that developers are aware that the appearance can be altered via the *'r.FidelityFX.FSR2.ReactiveMask'* console-variables. Tuning these variables to suit the content may be necessary to optimise visual results.

Translucent Skyboxes & Background Planes

When using a skybox or a distant background plane it is beneficial for this to be rendered with the Opaque or Masked shading model when using FSR2. If these are rendered with the Translucent shading model they will contribute to the FSR2 translucency and reactive masks, which can result in unnecessary artefacts. This is especially noticeable when other translucent materials are rendered over the top of the skybox/background-plane and the camera moves. This occurs because the plugin cannot distinguish the purpose of individual translucent materials, so they are all treated the same.

To address this issue, the FSR2 plugin assumes that a translucent skybox or background-plane is used and will fade out translucency contribution based on reconstructed distance from the camera. This will cut out all translucency rendered over distant opaque geometry and can be controlled with the *'r.FidelityFX.FSR2.ReactiveMaskTranslucencyMaxDistance'* console variable.

When using an opaque skybox or backplane, adjust the *'r.FidelityFX.FSR2.ReactiveMaskTranslucencyMaxDistance'* console variable to avoid translucency cut-outs.

Hair & Dither Effects

FSR2 does not smooth dither effects in the way other upscalers do. They are retained as thin features which may not be intentional. To avoid this, especially with hair, enable the *'r.FidelityFX.FSR2.DeDither'* console variable which attempts to smooth dither effects prior to FSR2 upscaling.

Release Notes

2.0.1

- Initial Release.

2.0.1b

- Resolved a crash in Unreal Engine 5.

2.1

- Update the FSR2 code to version 2.1.
- Fixed incorrect rendering & crashes with Split-Screen.
- Resolved a crash when changing Scalibility Level in the Editor.
- Resolved a crash when enabling visualisation modes that disable upsampling.
- Resolved a crash when using Shader Model 6 in Unreal Engine 5.
- Resolved a crash opening the Unreal Editor when using the Vulkan RHI.
- Fixed sampling from reduced resolution ScreenSpace Reflections & Separate Translucency.
- Fixed re-enabling World-Position-Offset console variables when toggling FSR2 on & off.
- Added an optional de-dither pass to improve FSR2's handling of dithered materials, especially Hair.
- Disabled FP16 in the RCAS pass to prevent incorrect rendering.
- Added an option to treat a shading model as reactive & use either CustomData0.x or the value of *'r.FidelityFX.FSR2.ReactiveMaskForceReactiveMaterialValue'* as the reactive mask value.
- Added an option *'r.FidelityFX.FSR2.ForceLandscapeHISMMobility'* to force the mobility of Hierarchical Instanced Static Mesh components attached to Landscapes to Stationary so they render motion vectors.
- Added an Engine patch to improve rendering of 'Static' objects that use a material with World-Position-Offset.
- Added an Engine patch which adds the Lit-Reactive ShadingModel that can be used to pass a reactivity value to write into the Reactive Mask for animated materials such as video screens.
- Remove an unnecessary RHI command flush that reduced CPU performance.

2.1.1

- Update the FSR2 code to version 2.1.1.

2.1.2

- Update the FSR2 code to version 2.1.2.

Setup

The FSR2 plugin is intended for Unreal Engine 4.26.2* or later. The FSR2MovieRenderPipeline plugin is intended for Unreal Engine 4.27.2* or later.

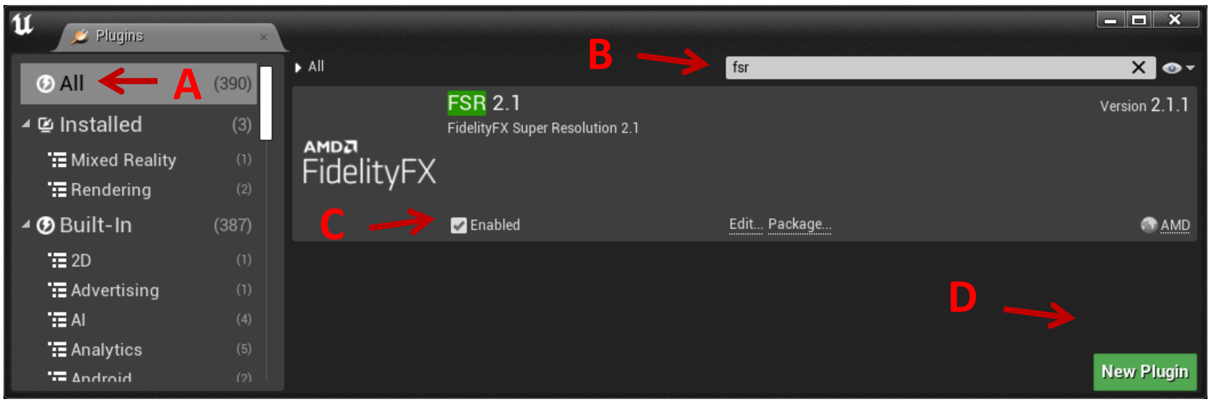
*If you are not a registered Unreal Engine developer, you will need to follow these instructions to register for access to this link.

Patch the Engine. For optimal quality it is necessary to use Unreal Engine from source code & apply source code patches.

1. To improve FSR2's handling of World-Position-Offset materials applied to Static objects:
   a. Use: git apply <VERS>-ImproveStaticWPO.patch
      i. Where <VERS> should be the engine-version in use.
2. To improve FSR2's handling of animated opaque materials:
   a. Use: git apply <VERS>-LitReactiveShadingModel.patch
      i. Where <VERS> should be the engine-version in use.

To install the plugin:

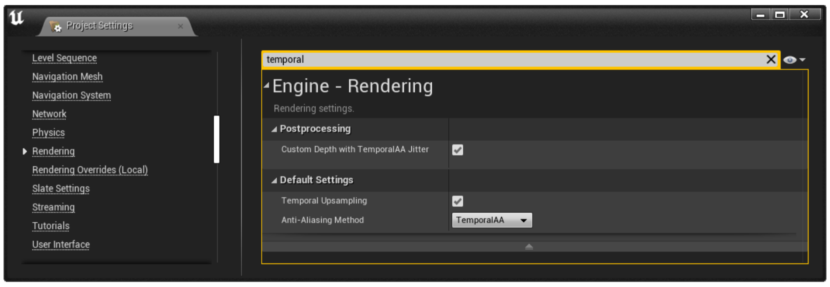
1. Locate the *Engine/Plugins* directory of your Unreal Engine installation.
2. Extract the contents of the FSR2.zip file.
3. Select the sub-folder that corresponds to the Unreal Engine version to be used.
4. Place the **FSR2** folder within your Unreal Engine source tree at: *Engine/Plugins/Runtime/AMD* (for UE4) or *Engine/Plugins/Marketplace* (for UE5)
   a. (Optional) Place the **FSR2MovieRenderPipeline** folder within your Unreal Engine source tree at: *Engine/Plugins/Runtime/AMD*. Only available from Unreal Engine **4.27.2 and later**.
5. Open your Unreal Engine project.
6. Navigate to **Edit > Plugins** in the Unreal Engine toolbar
7. Within the plugin dialog:
   a. Ensure that All is selected on the left side.
   b. Type `fsr` into the search box in the top right corner
   c. Select the **Enabled** checkbox for the **FSR 2.1** plugin.
      i. (Optional) Select the **Enabled** checkbox for the **FSR2MovieRenderPipeline** plugin.
   d. When prompted, click **Restart Now** to apply changes, and restart Unreal Engine.



Plugin Configuration

Usage

Temporal Upsampling must be enabled in the **Project Settings > Rendering** window, accessed via **Edit > Project** Settings in the Unreal Engine toolbar or via the Console Variable *`r.TemporalAA.Upsampling`*.

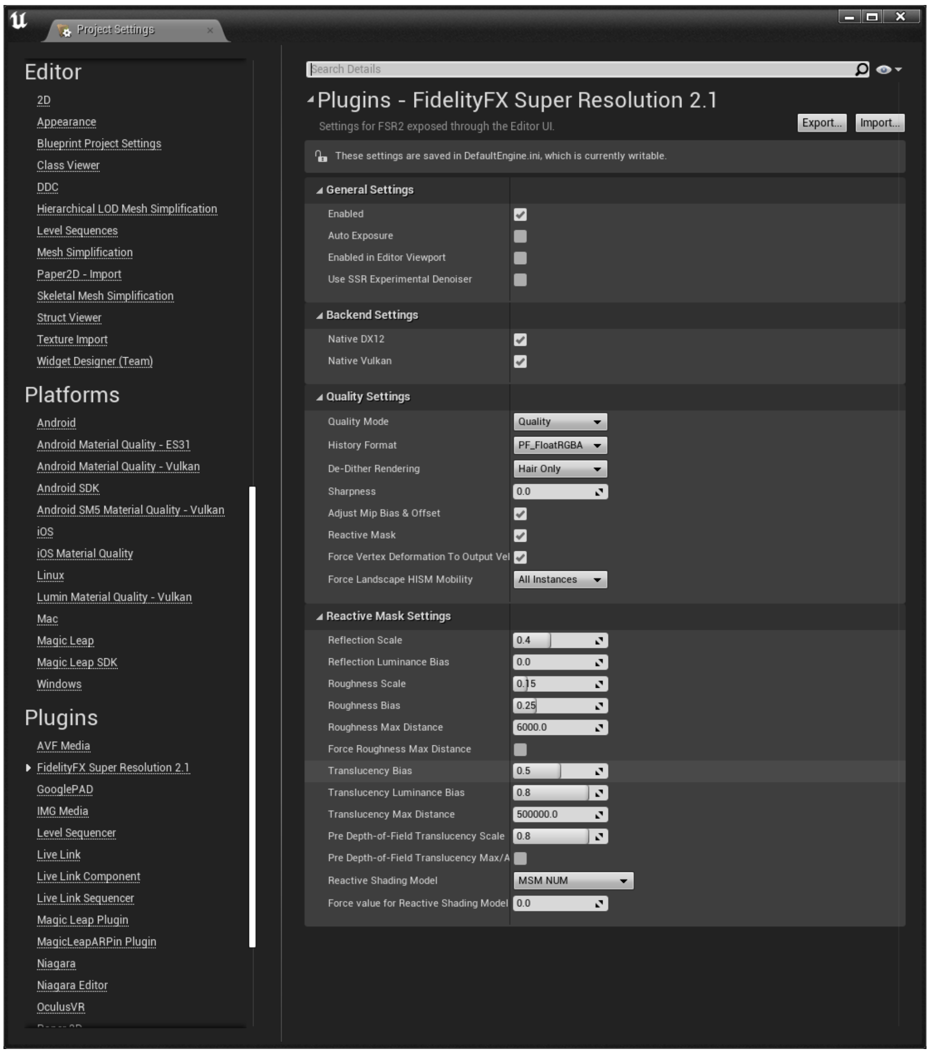


Engine - Rendering
Rendering settings.

Postprocessing
Custom Depth with TemporalAA Jitter ✓

Default Settings
Temporal Upsampling ✓
Anti-Aliasing Method: TemporalAA

FSR 2.1 can be enabled or disabled via the **Enabled** option in the **Project Settings > FidelityFX Super Resolution 2.1** settings window, or with the Console Variable *`r.FidelityFX.FSR2.Enabled`* in the configuration files. The variable can be modified at runtime ***however*** this is not guaranteed to be safe when other third-party upscalers are also enabled.



Plugins - FidelityFX Super Resolution 2.1
Settings for FSR2 exposed through the Editor UI.
These settings are saved in DefaultEngine.ini, which is currently writable.

General Settings	
Enabled	✓
Auto Exposure	☐
Enabled in Editor Viewport	☐
Use SSR Experimental Denoiser	☐
Backend Settings	
Native DX12	✓
Native Vulkan	✓
Quality Settings	
Quality Mode	Quality
History Format	PF_FloatRGBA
De-Dither Rendering	Hair Only
Sharpness	0.0
Adjust Mip Bias & Offset	✓
Reactive Mask	✓
Force Vertex Deformation To Output Vel	✓
Force Landscape HISM Mobility	All Instances
Reactive Mask Settings	
Reflection Scale	0.4
Reflection Luminance Bias	0.0
Roughness Scale	0.15
Roughness Bias	0.25
Roughness Max Distance	6000.0
Force Roughness Max Distance	☐
Translucency Bias	0.5
Translucency Luminance Bias	0.8
Translucency Max Distance	500000.0
Pre Depth-of-Field Translucency Scale	0.8
Pre Depth-of-Field Translucency Max/A	☐
Reactive Shading Model	MSM NUM
Force value for Reactive Shading Model	0.0

Quality Modes

The plugin will use specific quality modes specified via `*r.FidelityFX.FSR2.QualityMode*` overriding `*r.ScreenPercentage*`. The exposed modes are:

- **Quality (1.5x)**: `*r.FidelityFX.FSR2.QualityMode 1*`
  Provides an image quality equal or superior to native rendering with a significant performance gain.
- **Balanced (1.7x)**: `*r.FidelityFX.FSR2.QualityMode 2*`
  Offers an ideal compromise between image quality and performance gains.
- **Performance (2.0x)**: `*r.FidelityFX.FSR2.QualityMode 3*`
  Provides an image quality similar to native rendering with a major performance gain.
- **Ultra Performance (3.0x)**: `*r.FidelityFX.FSR2.QualityMode 4*`
  Provides the highest performance gain while still maintaining an image quality representative of native rendering.

Integration Instructions

RCAS

FidelityFX Super Resolution 2.1 contains a built-in sharpening pass called **Robust Contrast Adaptive Sharpening** that can be configured through the CVar `*r.FidelityFX.FSR2.Sharpness*`, this is disabled by default. If your project has already integrated FidelityFX-CAS, it may be necessary to disable FidelityFX CAS - including any in-game menu options - while `*r.FidelityFX.FSR2.Sharpness*` is enabled to prevent over-sharpening your final renders, and improve integration results.

World-Position-Offset

In order for FSR2 to process materials with World Position Offset and/or World Displacement correctly the `*r.VertexDeformationOutputsVelocity*` (UE4) or `*r.Velocity.EnableVertexDeformation*` (UE5) option must be enabled. The `*r.FidelityFX.FSR2.ForceVertexDeformationOutputsVelocity*` setting is enabled by default and when enabled FSR2 will force `*r.VertexDeformationOutputsVelocity*` (UE4) or `*r.Velocity.EnableVertexDeformation*` (UE5) on.

This option also forces on the console-variable *'r.BasePassForceOutputsVelocity'* to ensure that all objects render velocity during the base-pass when a project enables *'r.BasePassOutputsVelocity'*.

For grass layers attached to Landscape actors it is also necessary to enable a new console variable *'r.FidelityFX.FSR2.ForceLandscapeHISMMobility'* to change their mobility to Stationary so they render motion vectors.

See the Known Issues & Setup sections for more details about an Engine patch that can improve visual quality in some circumstances.

Reactive Mask & Experimental ScreenSapce Reflection Denoiser

When `*r.FidelityFX.FSR2.CreateReactiveMask*` is enabled the FSR2 plugin forces `*r.SSR.ExperimentalDenoiser*` to 1 in order to capture the Screen Space Reflections, to handle this the initial value of `*r.SSR.ExperimentalDenoiser*` will be applied to `*r.FidelityFX.FSR2.UseSSRExperimentalDenoiser*`. Subsequent changes to the value of `*r.FidelityFX.FSR2.UseSSRExperimentalDenoiser*` will override this.

Multiple Upscalers

To switch to or from FSR2 to another temporal upscaler always ensure that only one external temporal upscaler is enabled at a time. Disable the current upscaler before enabling the desired upscaler.

Other Configurations

Console Variable	Default Value	Value Range	Details
r.FidelityFX.FSR2.AdjustMipBias	1	0, 1	Applies negative MipBias to material textures, improving results.
r.FidelityFX.FSR2.Sharpness	0	0.0 – 1.0	When greater than 0.0 this enables **Robust Contrast Adaptive Sharpening** Filter to sharpen the output image.
r.FidelityFX.FSR2.AutoExposure	0	0, 1	Set to 1 to use FSR2's own auto-exposure, otherwise the engine's auto-exposure value is used.
r.FidelityFX.FSR2.HistoryFormat	0	0, 1	Selects the bit-depth for the FSR2 history texture format, defaults to `PF_FloatRGBA` but can be set to `PF_FloatR11G11B10` to reduce bandwidth at the expense of quality.
r.FidelityFX.FSR2.CreateReactiveMask	1	0, 1	Enable to generate a mask from the SceneColor, GBuffer, SeparateTranslucency & ScreenspaceReflections that determines how reactive each pixel should be.
r.FidelityFX.FSR2.ReactiveMaskReflectionScale	0.4	0.0 – 1.0	Scales the Unreal engine reflection contribution to the reactive mask, which can be used to control the amount of aliasing on reflective surfaces.
r.FidelityFX.FSR2.ReactiveMaskReflectionLumaBias	0	0.0 – 1.0	Biases the reactive mask by the luminance of the reflection. Use to balance aliasing against ghosting on brightly lit reflective surfaces.
r.FidelityFX.FSR2.ReactiveMaskRoughnessScale	0.15	0.0 – 1.0	Scales the GBuffer roughness to provide a fallback value for the reactive mask when screenspace & planar reflections are disabled or don't affect a pixel.
r.FidelityFX.FSR2.ReactiveMaskRoughnessBias	0.25	0.0 – 1.0	Biases the reactive mask value when screenspace/planar reflections are weak with the

			GBuffer roughness to account for reflection environment captures.
r.FidelityFX.FSR2.ReactiveMaskRoughnessMaxDistance	6000	0.0 – INF	Maximum distance in world units for using material roughness to contribute to the reactive mask, the maximum of this value and View.FurthestReflectionCaptureDistance will be used.
r.FidelityFX.FSR2.ReactiveMaskRoughnessForceMaxDistance	0	0, 1	Enable to force the maximum distance in world units for using material roughness to contribute to the reactive mask rather than using View.FurthestReflectionCaptureDistance.
r.FidelityFX.FSR2.ReactiveMaskTranslucencyBias	1.0	0.0 – 1.0	Scales how much contribution translucency makes to the reactive mask. Higher values will make translucent materials more reactive which can reduce smearing.
r.FidelityFX.FSR2.ReactiveMaskTranslucencyLumaBias	0.8	0.0 – 1.0	Biases the translucency contribution by the luminance of the transparency. Higher values will make bright translucent materials more reactive which can reduce smearing.
r.FidelityFX.FSR2.ReactiveMaskPreDOFTranslucencyScale	1.0	0.0 – 1.0	Scales how much contribution pre-Depth-of-Field translucency color makes to the reactive mask. Higher values will make translucent materials more reactive which can reduce smearing.
r.FidelityFX.FSR2.ReactiveMaskPreDOFTranslucencyMax	0	0, 1	Toggle to determine whether to use the max(SceneColorPostDepthOfField - SceneColorPreDepthOfField) or length(SceneColorPostDepthOfField - SceneColorPreDepthOfField) to determine the contribution of Pre-Depth-of-Field translucency.

r.FidelityFX.FSR2.ReactiveMaskTranslucencyMaxDistance	500000	0.0 – INF	Maximum distance in world units for using translucency to contribute to the reactive mask. This is a way to remove sky-boxes and other back-planes from the reactive mask, at the expense of nearer translucency not being reactive.
r.FidelityFX.FSR2.ReactiveMaskReactiveShadingModelID	MSM_NUM	0-MSM_NUM	Treat the specified shading model as reactive, taking the CustomData0.x value as the reactive value to write into the mask.
r.FidelityFX.FSR2.ReactiveMaskForceReactiveMaterialValue	0.0	0.0-1.0	Force the reactive mask value for Reactive Shading Model materials, when > 0 this value can be used to override the value supplied in the Material Graph.
r.FidelityFX.FSR2.ForceVertexDeformationOutputsVelocity	1	0, 1	Force enables materials with World Position Offset and/or World Displacement to output velocities during velocity pass even when the actor has not moved.
r.FidelityFX.FSR2.ForceLandscapeHISMMobility	0	0, 1, 2	Allow FSR2 to force the mobility of Landscape actors Hierarchical Instance Static Mesh components that use World-Position-Offset materials so they render valid velocities. Setting 1 is faster on the CPU, 2 is faster on the GPU.
r.FidelityFX.FSR2.UseSSRExperimentalDenoiser	0	0, 1	Enable to use *r.SSR.ExperimentalDenoiser* when FSR2 is enabled. This is required when *r.FidelityFX.FSR2.CreateReactiveMask* is enabled as the FSR2 plugin overrides *r.SSR.ExperimentalDenoiser* in order to capture reflection data to generate the reactive mask.
r.FidelityFX.FSR2.UseNativeDX12	1	0, 1	True to use FSR2's native & optimised D3D12 backend, false to use the fallback implementation based on Unreal's RHI.

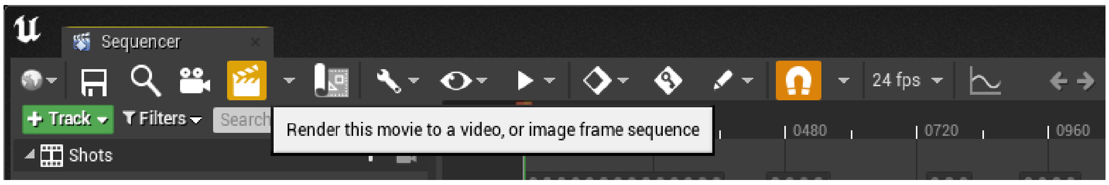
r.FidelityFX.FSR2.UseNativeVulkan	1	0, 1	True to use FSR2's native & optimised Vulkan backend, false to use the fallback implementation based on Unreal's RHI.
r.FidelityFX.FSR2.EnabledInEditorViewport	0	0, 1	Enable FidelityFX Super Resolution for Temporal Upscale in the Editor viewport by default.
r.FidelityFX.FSR2.DeDither	2	0, 1, 2	Enable an extra pass to de-dither rendering before handing over to FSR2 to avoid over-thinning. Can be set to Full for all pixels or to just around Hair for Deferred Renderer. Defaults to Hair Only.

Movie Render Pipeline Plugin

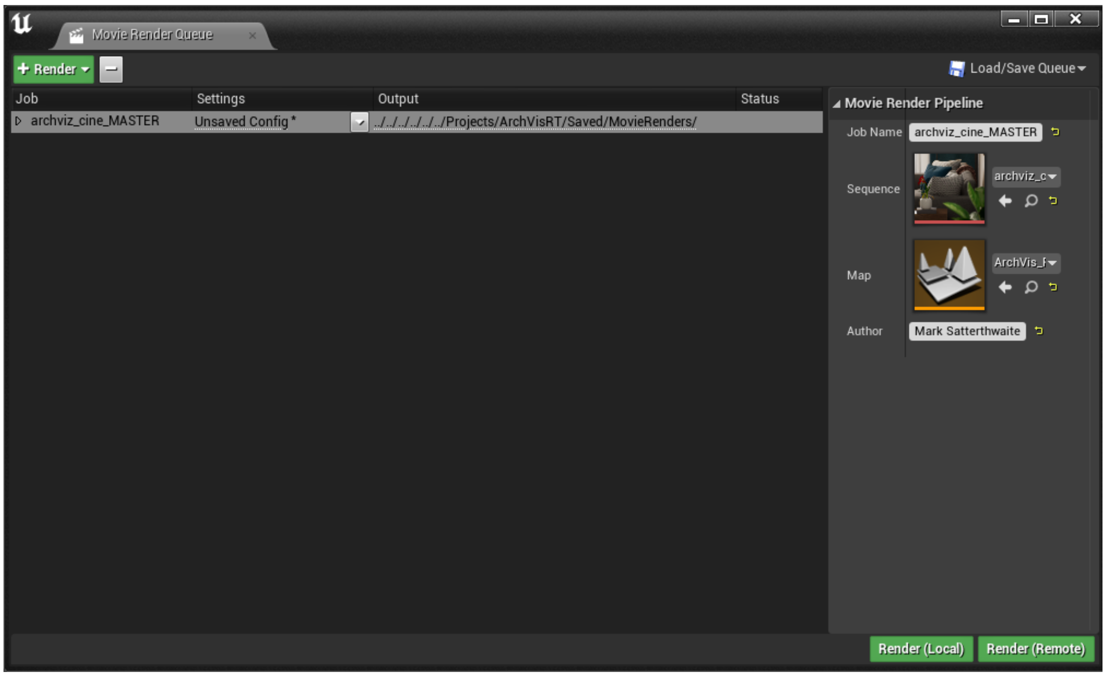
The FSR2MovieRenderPipeline plugin once enabled allows using FSR2 to accelerate rendering of Sequencer cinematics using Unreal's Movie Render Queue.

To use the plugin:

1. Open your Unreal project.
2. Open a **Sequencer** cinematic through the Editor.
3. Select the movie output in the toolbar.



4. Click on **'Unsaved Config'** to open the settings for the Movie Render Queue.

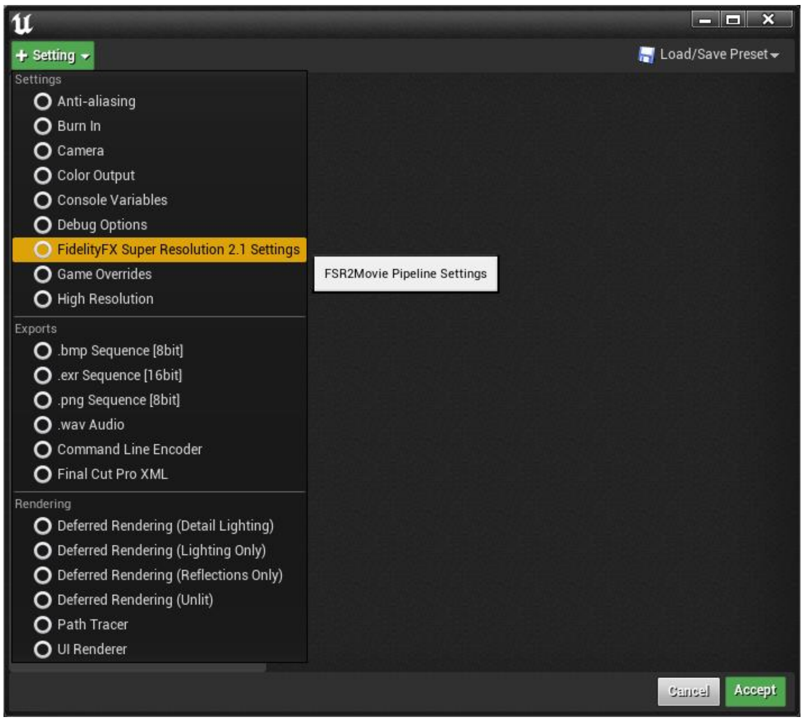


Job	Settings	Output	Status
archviz_cine_MASTER	Unsaved Config *	../../../../../Projects/ArchVisRT/Saved/MovieRenders/	

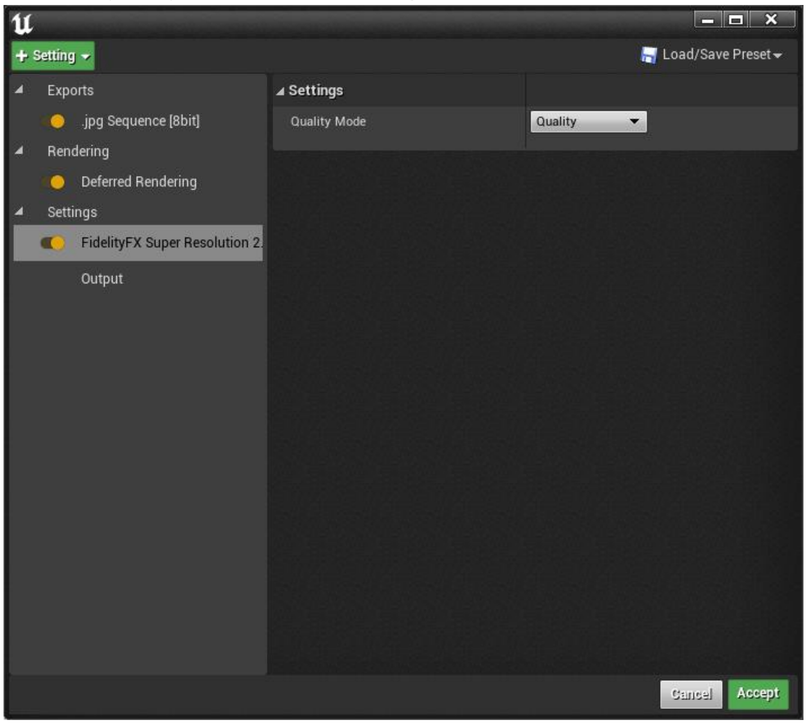
Movie Render Pipeline

Job Name	archviz_cine_MASTER
Sequence	archviz_c
Map	ArchVis_I
Author	Mark Satterthwaite

5. Select the **'+ Setting'** option and enable **'Fidelity FX Super Resolution 2.0 Settings'**.



6. Then select the new **'Fidelity FX Super Resolution 2.0 Settings'** in the list and choose the desired quality mode for rendering.



Quality Mode: Quality

7. Click Accept & then Render (Local).
8. The output will be rendered using FSR2 to upscale the output to the target resolution.