Camera Lens Calibration Overview

Learn about the tools and algorithms included in the Camera Calibration plugin.

Creating accurate compositions from CG renders and live video requires a virtual camera in Unreal Engine that accurately simulates the physical camera used to capture the real-world video footage. The virtual camera's position and orientation must closely match that of the physical camera, and its tracking information must match the video feed's exact timing to ensure each video frame is accurately synced to the position of the camera for each moment in time.

The **Camera Calibration** plugin provides simplified tools and workflows to calibrate a camera and lens in the Editor. This calibration process generates the data necessary to accurately align the virtual camera with the physical camera's position in space, and to model the lens distortion of the physical camera. The plugin introduces the Lens File asset type which encapsulates all of the calibration data for the camera and lens.

The Camera Calibration plugin also includes a robust lens-distortion pipeline that takes the calibrated distortion data and applies an accurate post-process effect to the CG render. The distortion post-process effect can be applied directly to a CineCamera Actor, which can be used in Movie Render Queue, or to the CG layers of Composure.

The plugin's tools and framework are extensible and flexible to support a wide range of lenses and workflows.

Camera Lens Calibration

Focus and Iris Mapping

The Camera Calibration plugin uses a Lens File asset to convert the raw focus and iris values to physical units used by the CineCamera component. For example, the raw input values could include encoder positions (absolute or normalized) streamed from an external device through Live Link. These values could also be converted to the appropriate units, but this would require fine tuning to achieve better accuracy.

Calibrated Data

Calibrated lens data can vary depending on the camera's focus and zoom. Therefore, to achieve a high-quality calibration, you often need to calibrate using multiple focus and zoom points. Distortion parameters, camera intrinsics, and nodal point offsets are all stored in the Lens File along with their associated focus and zoom value. It is possible to evaluate a calibrated Lens File for any focus and zoom position by interpolating between the stored focus and zoom points.

Distortion Parameters

The Camera Calibration plugin adds support for an accurate lens distortion post-process effect defined for a physical lens model. This lens model defines a set of distortion parameters that are used to compute the distortion for various positions around the lens.

The plugin supports a spherical-distortion model, based on the Brown-Conrady model, that uses five distortion parameters (K1, K2, P1, P2, K3). These parameters are unitless and are computed as part of the calibration process. The plugin is extensible and provides the ability to add support for additional models with different parameters.

Camera Intrinsics

Camera intrinsics include focal length and image center, and define how 3D points in the camera's coordinate system are projected to a 2D image. These intrinsics must also be calibrated to accurately model the camera and lens.

Depending on the calibration method, focal length may be calculated using physical units (millimeters) or pixel count. Focal Length is often represented as a 2D vector (Fx, Fy), which should be roughly equal in length. To generalize the representation, the Lens File stores calibrated focal length as normalized values dividing either by the image sensor width and height (in millimeters), or by the image resolution (in pixels).

After normalization, Fx and Fy will no longer be roughly equal to one another, but (Fy / Fx) should be roughly equal to your aspect ratio.

A calibrated image center takes into account the physical misalignment between the lens and the image sensor of the camera, but it should generally be close to the ideal center. Image center is also typically calculated in physical units (millimeters) or in pixels and is represented by a 2D vector (Cx, Cy). Like focal length, image center is normalized by dividing by the sensor width and height (in millimeters) or by the image resolution (in pixels). After normalization, the image center should be close to (0.5, 0.5).

Nodal Point Offset

The nodal point of a lens is the point at which light rays converge. In Unreal Engine, the virtual camera should be placed in the virtual world at this nodal point to ensure that real objects and CG objects are well-aligned.

Nodal Point Offset calibration is a separate calibration step that finds the pose of your tracked physical camera and computes the offset between that pose and your tracking data. This ensures that as the camera moves, the virtual camera's transform will continue to be set to the nodal point of the physical lens.

ST Maps

An ST Map is an image in which the value of each pixel corresponds to a distorted image coordinate in UV space. ST Maps can be generated using calibration tools outside of Unreal Engine and can be used in post-processing to alter the UVs used when sampling scene textures.

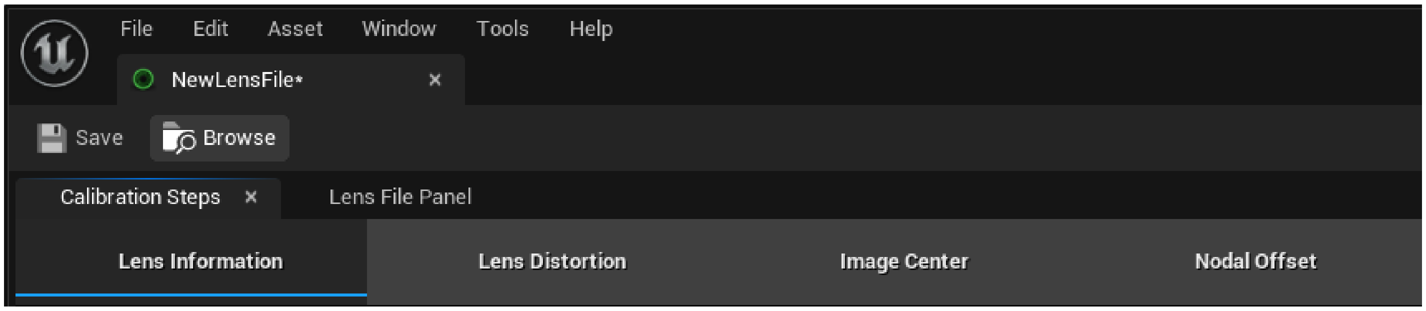
To support users who already generated calibrated data from external tools, the Camera Calibration plugin supports direct input of ST Maps into a Lens File.

Lens File Asset Editor

The Lens File Asset Editor provides multiple calibration tools for automatically populating the Lens File with calibrated data. The editor also provides a curve editor from which you can adjust the computed results.

Calibration Steps

The calibration steps featured in Unreal Engine include a step for entering static lens information, lens distortion calibration, adjusting the image center, and nodal point offset calibration. The list of calibration steps is extensible, meaning programmers can implement additional calibration tools and add them into the editor.



Lens Information

You can input static lens data, such as the lens model name and serial number, as well as camera data, such as the dimensions of the image sensor. Using accurate lens information ensures an accurate calibration of lens distortion and camera intrinsics.

Lens Distortion

You can select the calibration algorithm that is used to compute distortion parameters and camera intrinsics. The plugin includes the following algorithms, which rely on OpenCV to calibrate the lens using a set of 3D-2D point correspondences:

- **Checkerboard:** Uses a rigid checkerboard grid whose corners can be automatically detected.
- **Points Method:** Uses any calibrator object with an identifiable calibration pattern, whose features must be manually clicked by the user

This step is extensible to let programmers implement additional algorithms by inheriting from the UCameraLensDistortionAlgo class.

Image Center

You can manually adjust the location of the image center after you calibrate the lens. This can be useful when the interpolated image center between two calibrated focus or zoom points is not accurate, or when external factors, such as temperature, affect your camera lens slightly.

Nodal Offset

You can select the calibration algorithm and calibrator object that will be used to compute the nodal point offset. The plugin features the following algorithms:

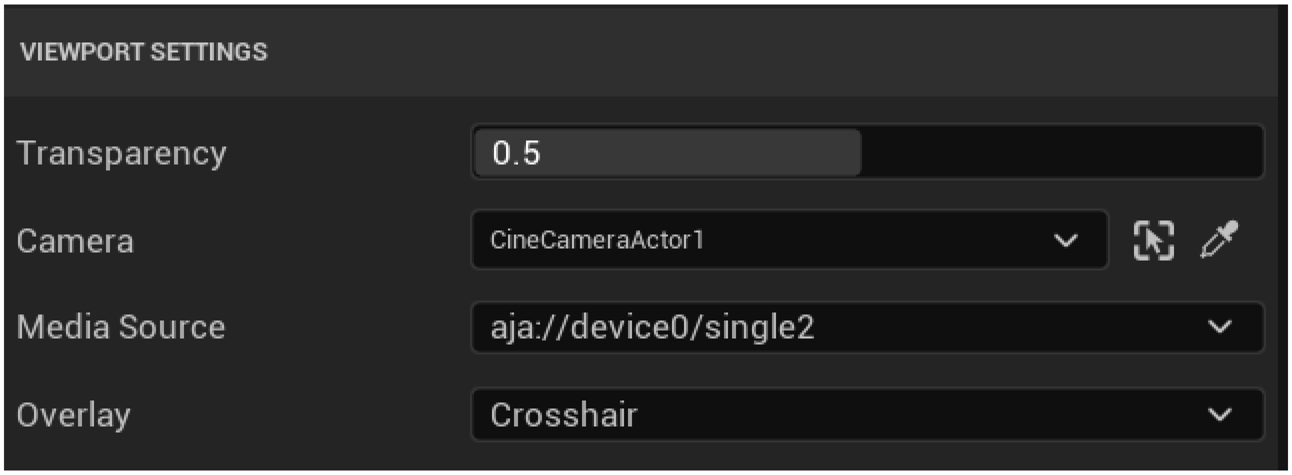
- **Points Method:** Using a tracked calibrator object with one or more identifiable features (such as an LED marker), you can position the calibrator at varying locations in view of their camera and click in the simulcam viewport to capture the current 3D and 2D location of the calibrator. After gathering a sufficient number of points, the nodal offset is estimated by minimizing the reprojection error of the captured points.
- **Aruco Markers:** A specialization of the points method that uses image processing to capture the 2D positions of a tracked calibrator with a printed ArUco pattern on it.
- **Checkerboard:** A specialization of the points method that uses image processing to capture the 2D positions of a tracked calibrator with a printed checkerboard pattern on it.
- **Optical Axis:** Using a calibrator object, you can capture a small number of points at varying distances from the camera that all project to the exact center of the lens. The line between those points represents the optical axis, and the nodal point is found by manually moving the virtual camera along that axis until the entrance pupil position is found.

This step is extensible to let programmers implement additional algorithms by inheriting from the UCameraNodalOffsetAlgo class.

Viewport Settings

You can change what displays in the Lens File Asset Editor's viewport by modifying the Viewport Settings. Viewport Settings contains the following parameters:

- **Transparency:** Slider to adjust the composite of the media and CG layers. 0.0 represents 100% media. 1.0 represents 100% CG.
- **Camera:** The Virtual Camera Actor that is used to render the CG layer. To do any calibration steps, this camera must also have a Live Link Camera Subject driving its focus, iris, and zoom values, and have this Lens File Asset assigned to it.
- **Media Source:** The Media Profile that is used to render the media layer.
- **Overlay:** Options to apply an overlay on the viewport. You can choose **None** or **Crosshair**. You can also add your own custom overlay materials in the plugin's settings.


VIEWPORT SETTINGS

Transparency	0.5
Camera	CineCameraActor1
Media Source	aja://device0/single2
Overlay	Crosshair

Distortion Pipeline

The Lens Distortion pipeline uses camera and lens data to create an accurate distortion effect that is applied to the final render.

There are three primary types of input data that can be used to model the distortion of a real-world lens:

- A Live Link Lens source streaming per-frame distortion parameters and camera intrinsics.
- A Live Link Camera source streaming camera FIZ to evaluate a Lens File asset.
- The evaluation of a Lens File Asset using the current focus and zoom of the Camera Actor.

The underlying object that receives input data and produces the distortion state is called the **lens distortion handler**. Objects that feed the input data into a handler are referred to as **producers** of distortion data. Objects that receive the output displacement maps from the handler and apply them to an image are referred to as **consumers** of distortion data.

The lens distortion handler receives distortion parameters and camera intrinsics as input, and generates a distortion UV displacement map and a post-process material that references that displacement map. The displacement map resolution can be specified in the project settings, and can be relatively small compared to the final render resolution (e.g. 128 × 128).

Producers of Distortion Data

LiveLink streaming per-frame distortion parameters and camera intrinsics

There are third party camera tracking vendors that provide calibrated lens distortion information in addition to basic camera position and orientation using their own protocols. This lens data, including distortion parameters and camera intrinsics, is streamed for each frame and can be used to compute distortion without the need of a Lens File asset. In order to stream this data into Unreal Engine, we provide a LiveLink Lens Role and Lens Controller.

The Lens Controller will instantiate a new lens distortion handler and stream the data it receives from the LiveLink source. The type of handler that is instantiated is dependent on the lens model, which must also be specified by the LiveLink source as static data.

LiveLink streaming camera FIZ to Evaluate a Lens File asset

For users who perform camera calibration within Unreal Engine, we have added a Lens File asset to the LiveLink Camera Controller. When a Lens File is present, the camera controller will evaluate that Lens File at the focus and zoom positions streamed through LiveLink. This evaluation interpolates between the closest calibrated points in the Lens File to generate distortion parameters, camera intrinsics, and nodal point offset for the current frame.

Alternatively, if the Lens File contains ST Maps, those will also be interpolated based on incoming focus and zoom streamed through LiveLink.

Static Camera FIZ and LensFile asset

Some users may not stream live-camera data through LiveLink, so the pipeline also supports evaluating a Lens File using the virtual camera's current focus and zoom settings.

The Camera Calibration plugin introduces a Lens Distortion component that can be added to a CineCamera Actor. This component can evaluate a Lens File directly based on the current camera settings.

Blueprints

You can create a lens distortion handler object using Blueprints and drive its distortion state without the need to specify a Lens File. This workflow is not designed for compositing CG elements with live video, but could be used for CG-only projects in which a distortion look is preferred.

Consumers of Distortion Data

CineCamera with Lens Distortion Component

A Lens Distortion component can be added to a CineCamera Actor to receive distortion data from any producer in the scene.

You can select a distortion source from the component settings. When a distortion source is selected, a post-process material will automatically be added to the target camera component, which receives the distortion displacement map generated by the chosen distortion source.

> Applying a distortion effect to a camera will not affect any Composure CG layers that may be targeting that camera.

Composure CG Layers

You can specify a distortion source within the Composure CG layer settings. The list of available sources is populated based on the producers in the Level that are currently producing distortion data for the camera that the CG layer is targeting.

When a distortion source is selected, a post-process material will automatically be added to the CG layer's scene capture component. This material receives the distortion displacement map generated by the chosen distortion source. Note that the target camera must be a CineCamera Actor.

Applying a distortion effect to a CG layer will not affect the rendered output of the target camera.

Overscan

In order to properly distort a CG image, it is often the case that more input pixels are required than are available in the original undistorted render. In order to ensure that the distorted image has valid data at every pixel, we must **overscan** the undistorted render to produce a small amount of additional render from the scene.

The percentage of extra pixels needed in the overscan is computed by the lens distortion handler based on the current distortion state. The **Oversan Factor** is then used by the distortion consumers to augment the field of view (FOV) of the undistorted render before applying the post-process distortion material. This produces an undistorted image with a wider view of the scene, albeit at the same target image resolution as the original undistorted image.

An overscan multiplier can also be specified from any of the distortion producer settings, from which you can specify how much of the computed overscan should be applied (anywhere from 0x to 2x).

Undistortion

The lens distortion handlers compute an undistortion displacement map, which is accessible from Blueprints. This gives the ability to create custom post-process materials and apply them to reverse distortion from incoming media sources.

OpenCV

The Camera Lens Calibration tools use OpenCV for its algorithms. You can use OpenCV directly through the **OpenCV** plugin, which contains OpenCV version 4.5.4 and adds new Blueprint nodes, such as for ArUco marker tracking and chessboard tracking. Refer to the Blueprint API Reference for more details on the Blueprint nodes:

- Blueprint API Reference for OpenCV Chessboard Functions
- Blueprint API Reference for OpenCV ArUco Functions