Viz Virtual Studio Administrator Guide

Version 1.9



Viz Virtual Studio

Copyright ©2026 Vizrt. All rights reserved.

No part of this software, documentation or publication may be reproduced, transcribed, stored in a retrieval system, translated into any language, computer language, or transmitted in any form or by any means, electronically, mechanically, magnetically, optically, chemically, photocopied, manually, or otherwise, without prior written permission from Vizrt. Vizrt specifically retains title to all Vizrt software. This software is supplied under a license agreement and may only be installed, used or copied in accordance to that agreement.

Disclaimer

Vizrt provides this publication “as is” without warranty of any kind, either expressed or implied. This publication may contain technical inaccuracies or typographical errors. While every precaution has been taken in the preparation of this document to ensure that it contains accurate and up-to-date information, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained in this document.

Vizrt’s policy is one of continual development, so the content of this document is periodically subject to be modified without notice. These changes will be incorporated in new editions of the publication. Vizrt may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time. Vizrt may have patents or pending patent applications covering subject matters in this document. The furnishing of this document does not give you any license to these patents.

Antivirus Considerations

Vizrt advises customers to use an AV solution that allows for custom exclusions and granular performance tuning to prevent unnecessary interference with our products. If interference is encountered:

- **Real-Time Scanning:** Keep it enabled, but exclude any performance-sensitive operations involving Vizrt-specific folders, files, and processes. For example:
  - C:\Program Files\[Product Name]
  - C:\ProgramData\[Product Name]
  - Any custom directory where [Product Name] stores data, and any specific process related to [Product Name].
- **Risk Acknowledgment:** Excluding certain folders/processes may improve performance, but also create an attack vector.
- **Scan Scheduling:** Run full system scans during off-peak hours.
- **False Positives:** If behavior-based detection flags a false positive, mark that executable as a trusted application.

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Created on

2026/11/19

Contents

1 New in Viz Virtual Studio 10
1.1 Video Wall Editor 10
1.2 Engine Color Correction 11
1.3 Data Viewer & Diagnostics 11
1.4 Other Improvements 11
2 Introduction 12
2.1 Document Structure 12
2.2 Feedback and Suggestions 12
3 Overview 13
3.1 Introduction and Terminology 14
3.1.1 Key Technologies and Procedures 15
3.1.2 Software Components 15
3.1.3 Documentary 16
3.2 Tracking Hub 17
3.2.1 Protocols supplied by a plug-in 18
3.2.2 Protocols and certified systems are not the same thing 18
3.3 Studio Manager 19
3.4 WIBU-based Licensing System 20
3.4.1 Important Pre-installation Information 20
3.4.2 Key Features and Workflow of the New Licensing System 20
3.4.3 Notable Limitations and Known Issues 20
3.4.4 Basic Setup 20
3.5 WIBU License System in Tracking Hub 22
3.5.1 One License Consumed 22
3.5.2 Two Licenses Consumed 24
3.5.3 Copy Rig Functionality 26
3.6 Tracking Hub Structural Design 28
3.6.1 Protocol 28
3.6.2 Parameters 29
3.6.3 Rig Objects 29
3.6.4 Basic Camera 44
3.6.5 Timing 47
3.6.6 Delay 48

3.6.7 Video Delay 48
3.6.8 Tracking Delay 49
4 Installation and Startup 51
4.1 Important Checklist Before Installation 52
4.1.1 System Requirements 52
4.2 Viz Virtual Studio Folders 55
4.2.1 Installation Folders 55
4.3 Viz Virtual Studio Installation 57
4.3.1 Prerequisites 57
4.3.2 Applications 57
4.3.3 .NET Framework 57
4.3.4 To Install Tracking Hub and Studio Manager Using the Bundle Installer 58
4.3.5 To Repair or Remove Tracking Hub 59
4.3.6 Unattended Installation of Viz Virtual Studio Applications 59
4.4 Start Viz Virtual Studio 61
4.4.1 Manual Configuration of the Network Adapter IP Addresses 63
4.4.2 Start the Studio Manager 63
4.4.3 Activating Automatic Logon 65
4.4.4 License Configuration 66
4.4.5 Closing Viz Tracking Hub 67
5 Studio Manager User Interface 68
5.1 Parameter Panel 69
5.2 Topology Panel 70
5.2.1 Tracking Systems 70
5.2.2 Rigs 72
5.2.3 Services 75
5.2.4 Topology Connection Lines 76
5.2.5 Topology Colors 77
5.2.6 Motion Capturing with Motion Analysis 77
5.2.7 Multi Object Tracking 79
5.3 Log Panel 80
5.4 Timing Analysis Window 82
5.5 Preview Panel 84
5.5.1 Interface Configuration 85
5.5.2 View Menu 87
5.5.3 Navigation 87
5.5.4 HUD 88

5.5.5 Cyclorama Editor 89
5.5.6 Videowall Editor 89
5.5.7 Cyc 90
5.5.8 CoCyc 90
5.5.9 Objects Menu 90
5.5.10 Selectable Display Modes 91
5.5.11 Camera Handles 92
5.6 Post System 97
5.6.1 Create a Post Session 97
5.6.2 Load a Post Session 98
5.6.3 Configure a Post Session 99
5.6.4 Selecting the Recording and Replay Sources 100
5.6.5 Camera Channels 101
5.6.6 Post Offset and Delay 102
5.6.7 Post Data Storage 103
5.7 Data Inspector 104
5.7.1 1 What the Data Inspector is for 104
5.7.2 2 Post data mode 105
5.7.3 3 Live mode 108
5.8 Configuration Panel 113
5.9 Target Detection 115
5.9.1 Viz Engine Scene Preparation 117
5.9.2 Target Detection Wizard 118
5.9.3 Automated Target Localization 119
5.9.4 Target Coverage 119
5.9.5 Target Preview 120
5.9.6 Target Localization Procedure 120
5.9.7 Example Script 129
6 Studio Manager Configuration 131
6.1 Configure the Studio 132
6.2 Tracked Cameras and Viz Engine 133
6.3 Router Control 134
6.3.1 To Add a Router Control Preset 134
6.3.2 GPI IO Device Switching 141
6.4 Backup Configuration 143
6.4.1 Master and Slave 144
6.4.2 What Synchronize transfers 144

6.4.3 Sending to Viz Engine while a backup is configured 144
6.5 Replay Tool 146
6.5.1 Prerequisites 146
6.5.2 Preparations 147
6.6 Post Data Inspector 162
6.6.1 Controlling the Data-plot 166
6.7 NDI Output Service 167
6.7.1 Embedding UDP Tracking into an NDI Stream 167
6.7.2 Viz Engine Configuration 167
6.7.3 Camera Configuration 168
6.7.4 Studio Manager Configuration 171
6.7.5 Viz Arc Camera Switching 174
6.8 Embedded NDI+ Tracking Data 176
6.8.1 Introduction 176
6.8.2 Setting up the Studio 176
6.8.3 Setting up the Tracker 177
6.8.4 Setup Rig Topology 181
6.8.5 Setup NDI Service 181
6.9 Configure Topology 184
6.9.1 Configure a Tracking System 184
6.9.2 Configure a Rig 188
6.9.3 Configure a Service 190
6.9.4 Set a Delay 192
6.9.5 Set an Offset 193
6.9.6 Set Center Shifts 194
6.9.7 Analyze Time 194
6.9.8 Lens Range Calibration 200
6.9.9 Modify a Rig 202
6.10 Use of Templates 206
6.10.1 Using Existing Templates 206
6.10.2 Creating New Templates 207
6.11 Studio Manager Camera Filter 209
6.11.1 Quick start 209
6.11.2 Setting up a filter 210
6.11.3 File format 212
6.11.4 What is displayed and what is hidden 214
6.11.5 How you can tell a filter is active 214

6.11.6 Commands disabled while a filter is active....215
6.11.7 Limitations....215
6.11.8 Troubleshooting....215
6.12 Engine Color Correction....217
6.12.1 1. Overview....217
6.12.2 2. Prerequisites....218
6.12.3 3. Running a calibration (operator workflow)....218
6.12.4 4. Applying / switching LUTs....223
6.12.5 5. The second layer (AR refine) — optional....223
6.12.6 6. Output files....224
6.12.7 7. Configuration file....225
6.12.8 8. Troubleshooting....229
6.13 Videowall Editor....230
6.13.1 Setup....230
6.13.2 The editor panel....233
6.13.3 Wall geometries....235
6.13.4 The settings panel....241
6.13.5 Advanced mode....243
6.13.6 Example setup....248
6.14 Videowall Editor — Example Setup....249
6.14.1 Step 1 — Add the videowall service....250
6.14.2 Step 2 — Choose the GFX channel....251
6.14.3 Step 3 — Add the back wall....251
6.14.4 Step 4 — Add the LED floor....252
6.14.5 Step 5 — Define the Screen mapping....253
6.14.6 Step 5 — Create the walls on the engine....254
6.14.7 Step 6 — Verify with the test room....255
6.14.8 Step 7 — Show real content....256
6.14.9 Step 8 — Add a second engine with wall groups....257
6.14.10 Troubleshooting....258
6.14.11 Checklist....259
6.15 Cyclorama Editor....260
6.15.1 Workflows....260
6.15.2 Common Cyc-Editor Elements....262
6.15.3 Back-Wall....263
6.15.4 One Corner....265
6.15.5 Two Corner....267

6.15.6 Masks 268
6.15.7 See Also 270
7 Lens File Editor 271
7.1 New Lens File Format 272
7.2 Lens File Editor 272
7.2.1 Camera Selection 273
7.2.2 Lens Area 274
7.2.3 Group Portlet 276
7.2.4 Engine Control 277
7.2.5 Calibration Scene 277
7.2.6 Parameter Control and Views 278
7.2.7 3D View 279
7.2.8 Point Matrix 280
7.2.9 2D View 281
7.2.10 Shortcut System 283
7.3 Automated Lens Calibration 285
7.3.1 Preparations 285
7.3.2 Start Process 285
7.3.3 Prepare Targets 288
7.3.4 Start Calibration 290
7.3.5 Lens Calibration Panel 291
7.3.6 Lens Calibration Wizard 299
7.3.7 Lens Calibration Process 310
8 Troubleshooting 332
8.1 General 332
8.1.1 Syncing 332
8.2 Tracking 332
8.2.1 Tracking Networks 332
8.2.2 Receive Timing tool 332
8.3 Lenses 336
8.4 Delay 336
8.5 NDI 336
8.5.1 Requirements and Limitations 336
8.6 Chroma Keying 338
8.7 Remote Connection 338
9 Appendices 339

9.1 Words and Terminology 340
9.2 Supported Protocols 341
9.2.1 Protocols supplied by a plug-in 342
9.2.2 Protocols and certified systems are not the same thing 342
9.3 Supported NDI Cameras 343
9.4 Description of the FreeD Protocol 344
9.4.1 Start Element “vizth_xml_tracking” 344
9.4.2 Checksum Element 344
9.4.3 Extraction 345
9.4.4 Axis Element 345
9.5 Motion Analysis Integration 347
9.5.1 The CORTEX System 347
9.5.2 CCD Calibration 347
9.5.3 Sync and FPS (Frames per Second) 347
9.5.4 Timing 347
9.5.5 Network Connection 347
9.5.6 Cortex Precision Limit 348
9.5.7 Motion Capturing 348
9.6 TH and SM Ports 349

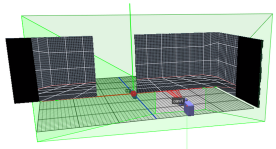
1 New in Viz Virtual Studio



The Viz Virtual Studio 1.9.0 release focuses on new tools that assist the operator during setup, especially when setting up XR sets.

The main new features in this release are:

1.1 Video Wall Editor

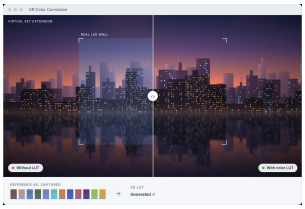


Viz Virtual Studio 1.9.0 introduces a new Video Wall Editor for setting up video wall geometries, especially for XR sets. It creates the video wall scenes in Viz Engine, as well as test scenes for easy alignment adjustments.

- **Wall layouts:** One-Corner, Two-Corner, Curved and Back Wall layouts, including right-aligned walls.
- **Vertex-based setup:** Walls are defined by their vertices. Measured points can be mapped directly to the wall geometry, and auto-alignment fits the wall to them.
- **Test scene:** A generated test scene shows on the engine right away whether geometry and alignment are correct.
- **Wall groups:** Walls can be grouped and each group routed to a different engine.
- **NDI preview:** The engine output can be previewed through NDI inside the editor, cropped to the area of the wall.

Copyright © 2026 Vizrt

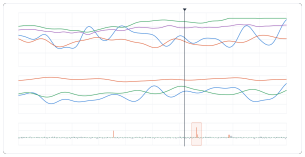
1.2 Engine Color Correction



A new tool creates automated color correction look-up tables (LUTs) for Viz Engine scenes. XR engines can adapt to the color space of the real video wall, giving a seamless transition between the real and the virtual extended set.

- **Automatic calibration:** The engine output is captured, compared with reference colors, and a 3D LUT is generated to correct the difference.
- **Calibration log:** A separate window shows each step of the calibration and its results.

1.3 Data Viewer & Diagnostics



Tracking analysis and troubleshooting tools are now combined in the Data Viewer.

- **Tracking flow:** The tracking flow view has moved into the Data Viewer, so live and recorded tracking data are inspected in one place.
- **FBX export:** Tracking data can be exported as FBX for use in other 3D and post-production applications.
- **Receive Timing:** A new tool in the Tools menu shows when tracking data is received, to help find timing and delay problems.
- **Communication logging:** Communication between Studio Manager and its services can be logged for troubleshooting.

1.4 Other Improvements

- **Lens calibration:** An in-progress calibration can be saved and loaded again later. Reference points can be adjusted and individual lens points reset. The Lens Tool has four more targets, and printed target sizes are kept. Lens data dumps are on by default, cleaned up automatically, and can be deleted manually.
- **Services:** Camera services can be renamed and filtered. Hostnames are shown in the service views, and the service status display is clearer.
- **Perspective view:** The camera can be controlled with the mouse, including scroll to translate.
- **Stability:** Loading a configuration shows a busy indicator, and removing services is faster. Fixed a crash when deleting presets, plus several smaller UI fixes.

Copyright © 2026 Vizrt

2 Introduction

This Administrator Guide gives the information required to understand **Tracking Hub** and **Studio Manager**, two important software components in the Viz Virtual Studio solution. The term *virtual studio* in general normally refers to tools which seek to simulate a physical television and/or movie studio. The **Viz Virtual Studio** solution is a set of software and technologies to create virtual studios.

2.1 Document Structure

Title	Content
Introduction	Gives details on related documentation and how to contact Vizrt for Customer support. Details are also given for Customers to provide feedback on their Vizrt product.
Overview	Gives an overview of both the Tracking Hub and the Studio Manager.
Installation and Startup	Describes the system requirements, planning for installation and steps to install and start Viz Virtual Studio.
Lens File Editor	Provides information about the Lens File Editor and calibration of lens calibration.
Studio Manager GUI	Describes the Studio Manager, skills needed to be familiar with the Studio Manager and how it operates.
Studio Manager Configuration	Describes how to configure Studio Manager and its components.
Troubleshooting	Lists some common pitfalls and how to solve them and guidance on optimizing your Virtual Studio workflow.
Appendices	Lists any additional information about the Viz Virtual Studio.

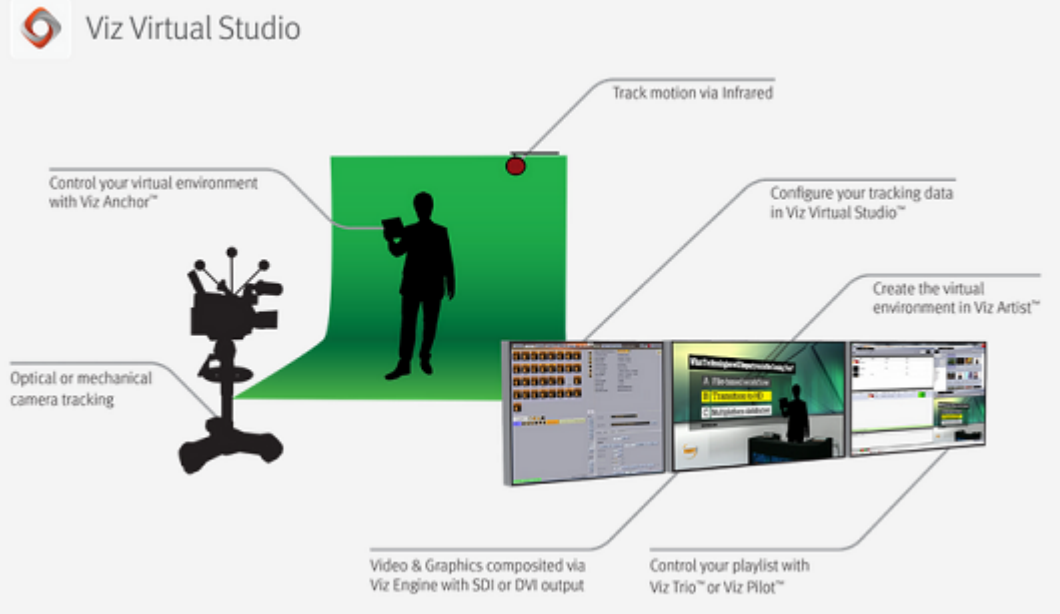
2.2 Feedback and Suggestions

We encourage suggestions and feedback about our products and documentation. To give feedback and/or suggestions, please contact your local Vizrt customer support team at www.vizrt.com.

Copyright © 2026 Vizrt

3 Overview

The Viz Virtual Studio provides a perfect match between real and the virtual scene elements. The reality is a compromise between the technical possibilities and the perceptual ability of the audience.



This manual gives a description of Tracking Hub and Studio Manager.

This release gives a limited feature set. Not all tracking devices, routers and GPI devices are implemented yet. The features of the Cyclorama (Cyc) Editor are limited. If more complex geometries are needed, or obstacles should be masked out, this needs to be done in the front layer of the Viz Engine.

Note: The Tracking Hub is intended for use with Viz Artist and Viz Engine versions 3.9.0 and later. The recommended versions are Viz Engine 4 or later.

If you are new to virtual studio technology or need a very short refresher, see the introduction in the Introduction and Terminology section.

This section contains information about the following topics:

- Introduction and Terminology
- Tracking Hub
- Studio Manager
- WIBU-based Licensing System
- WIBU License System in Tracking Hub
- Tracking Hub Structural Design

Copyright © 2026 Vizrt

3.1 Introduction and Terminology

This chapter gives a brief introduction to Viz Virtual Studio (VS). It introduces a few key concepts intended for readers who are not familiar with Virtual Studio. The term *virtual studio* in general normally refers to tools which seek to simulate a physical television and/or movie studio. Viz Virtual Studio is a set of software and technologies to create virtual studios.

A virtual studio is a television studio that allows the real-time combination of people or other real objects and computer generated environments and objects in a seamless manner. A key point of a virtual studio is that the real camera can move in 3D space, while the image of the virtual camera is rendered in real-time from the same perspective. The virtual scene has to adapt at any time to the camera settings such as zoom, pan, angle, traveling and more. This is what differentiates a virtual studio from the traditional technique of Chroma keying.

A major difference between a virtual studio and the bluescreen special effects used in movies is that the computer graphics are rendered in real-time, removing the need for any post production work, allowing it to be used in live television broadcasts. The virtual studio technique is also different from Augmented Reality (AR) which is a live direct or indirect view of a physical, real-world environment with elements augmented (or supplemented) by computer-generated sensory input such as sound, video, graphics or GPS data.



Copyright © 2026 Vizrt



3.1.1 Key Technologies and Procedures

The background (typically green) is masked away in real-time with a virtual software-created background. The physical objects in front (persons, desks, etc.) are merged with the virtual background creating the illusion of physical presence. The background is created and prepared in advance, typically using Viz Artist or by importing 3D-scenes into Viz Artist and saving them as scenes in Graphic Hub. Viz Engine, a real-time compositing engine, is responsible for combining the physical and virtual objects and to display the results (send the signal to a display, an IP stream or to send the signal downstream to a video mixer).

An important part of the virtual studio is the camera tracking that uses either optical or mechanical measurements to create a live stream of data describing the perspective of the camera. Exact camera and sensor tracking with as little delay as possible is the key difficulty to solve in a virtual studio.

3.1.2 Software Components

- **Tracking Hub:** A service process (runs as a console program without a GUI) responsible for receiving and forwarding measurements from cameras, trackers and other sensors. Tracking Hub must be kept running at all times.
- **Studio Manager:** A GUI for setting up, defining and controlling a Tracking Hub and the Virtual Studio environment.

Copyright © 2026 Vizrt

Viz Virtual Studio uses the Viz Engine to render the output signal, whereas a client application such as Viz Trio, Viz Mosart or Viz Pilot is normally used to control the overall broadcast.

3.1.3 Documentary

If reading this on an Internet-connected device, you can view the Viz Virtual Studio documentation (see links below). See also Words and Terminology in the Appendix chapter for common words and their meaning.

The secrets of virtual set production is divided into five parts:

1. What is virtual set and why should broadcasters use them?
2. Virtual sets versus augmented reality
3. Essential Vizrt tools
4. Virtual set tools
5. Design your story

Copyright © 2026 Vizrt

3.2 Tracking Hub

Tracking Hub is a console program for Viz Virtual Studio, it collects data from tracking systems (cameras and objects) and stores this information. This tracking data is provided to Viz Engine at a configurable time or after a delay. To achieve exact tracking, many configurations are stored and provided to the user.

The Tracking Hub currently has these tracking drivers installed:

Tracking Hub has native support for the following protocols. The name in brackets is the protocol identifier as it appears in the Studio Manager tracking system dropdown.

- FreeD (`freed`) — see Description of the FreeD Protocol
- FreeD A0 (`freeda0`)
- RTHead (`rthead`)
- XML Tracking (`xmltracking`) — a self-definable protocol for special solutions
- Motion Analysis (`motionanalysis`) — it is possible to work with two MA instances
- mo-sys (`mo-sys`) — with lens information
- Trackmen (`trackmen`)
- Trackmen Mocap (`trackmen-objtr`)
- Vicon (`vicon`)
- Libero (`libero`)
- Libero TC (`liberotc`)
- Spidercam (`spidercam`)
- Spidercam Fielddolly (`spidercamfd`)
- Spidercam FrameB (`spidercam-frameb`)
- Flair (`flair`)
- Thoma (`thoma`)
- Stype A5 (`stypea5`)
- Stype HF (`stypehf`)
- Camreginfo (`camreginfo`)
- Technodolly (`technodolly`)
- Skycam (`skycam`)
- HD Skycam (`hd_skycam`)
- GSS Geopoint (`gssgeopoint`)
- Kuper (`kuper`)
- Kronomav (`kronomav`)
- Telemetrics (`telemetrics`)
- Arraiy (`arraiy`) — only with a Vizrt lens file
- Eagle Eye (`eagleeye`)
- GeoPro (`geopro`)

Copyright © 2026 Vizrt

- Cable Cam (`cablecam`)
- Cable Cam 2 (`cablecam2`)
- Sony (`sony`)
- Sony D0 (`sonyd0`)
- Vips / Airpixel (`vipsairp`)

OpenTrackIO is implemented but not yet part of a released build. It will appear in this list once the feature is merged.

3.2.1 Protocols supplied by a plug-in

Besides the native protocols above, Tracking Hub loads protocol drivers as DLL plug-ins written with the plug-in API. NCam and Cincraft reach Tracking Hub this way, so they are not listed as native protocols.

3.2.2 Protocols and certified systems are not the same thing

Tracking Hub parses these protocols and sends the translated data to Viz Engine. A protocol listed here does not mean a tracking system using the same name is certified — see Certified Tracking Systems for the systems that are. Conversely, a certified system does not appear here under its product name: Shotoku and NCam are certified systems that speak one of the protocols above, and XPecto is the protocol implementation behind the `trackmen` entry.

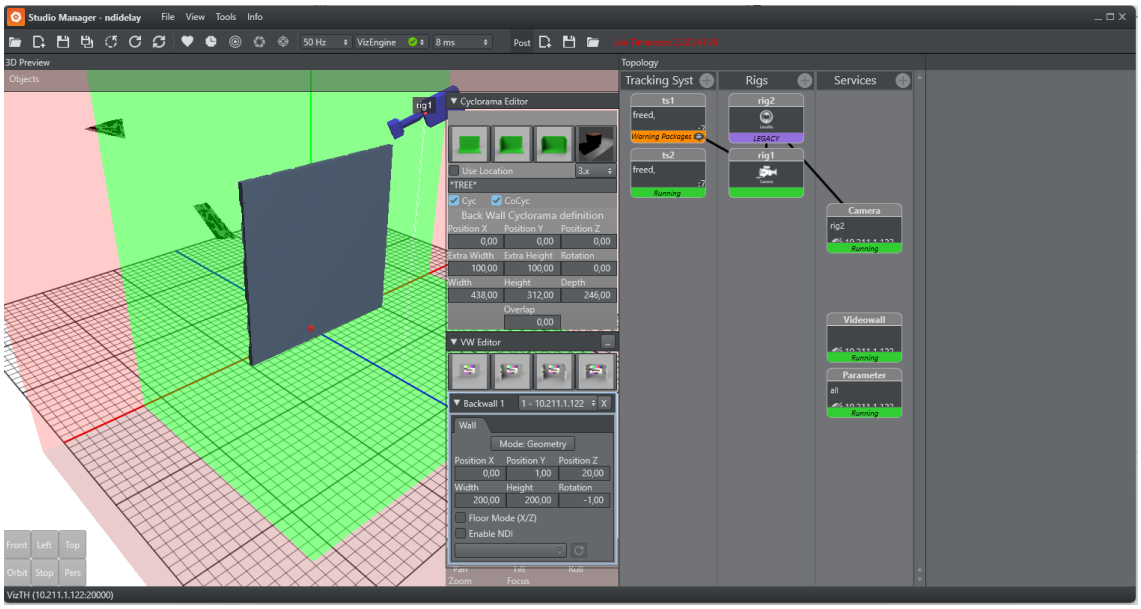
See Also

- Tracking Hub Structural Design
- Studio Manager

Copyright © 2026 Vizrt

3.3 Studio Manager

The Studio Manager is a Graphical User Interface (GUI) for the Tracking Hub. When the Studio Manager opens, it presents a login window where the user can select the network interface and the Tracking Hub to control. With few exceptions, every part of the Tracking Hub can be controlled by Studio Manager.



The Studio Manager application is described in the following chapters, in particular Studio Manager GUI and Studio Manager Configuration.

See Also

- Tracking Hub
- Tracking Hub Structural Design
- Configure the Studio
- Configure Topology

Copyright © 2026 Vizrt

3.4 WIBU-based Licensing System

This chapter describes management and usage of the new licensing system based on CodeMeter from WIBU Systems available in Viz Virtual Studio 1.1.1 and later. It replaces the previous VALID/Sentinel/Hardlock Dongle licensing system and allows Viz Virtual Studio to be used without a physical dongle on each machine by allocating licenses from a license server on the network.

The support for the old VALID/Sentinel/Hardlock Dongle was removed.

In this section, you find the following information:

- Important Pre-installation Information
- Key Features and Workflow of the New Licensing System
- Notable Limitations and Known Issues
- Basic Setup

3.4.1 Important Pre-installation Information

The WIBU licensing system requires the installation of the CodeMeter Runtime Software. It is not part of the Viz Virtual Studio Bundle installer and must be installed separately.

When the license should be retrieved from a dedicated license server, it must configured in the VizrtLicensing Service (see the **Installation** section of the Viz Licensing Administrator Guide) or the CodeMeter WebAdmin.

Please refer to the Viz Licensing Administrator Guide for further detailed information.

- There is an auto discovery if no license server is configured in the server search list of CodeMeter.
- On network disconnect and reconnect, it may happen that a license is checked out twice. In this case, it must be released manually on the CodeMeter service on the license server or the license server can be restarted.

3.4.2 Key Features and Workflow of the New Licensing System

- Dongle-less operation on the clients with monitoring and logging capabilities.
- Grace periods for allocated licenses to avoid immediate expiration on short network interruptions.
- Configurable WIBU license container location (local, network).

3.4.3 Notable Limitations and Known Issues

- Only one license system can be used.
- Uninstall any CodeMeter Runtime prior. And afterwards install the Virtual Studio and the newer CodeMeter Runtime.

3.4.4 Basic Setup

These are the steps to set up Viz Virtual Studio licensing with WIBU:

1. Install Viz Virtual Studio with the bundle installer. Configure CodeMeter with the VizrtLicensing Service or the CodeMeter WebAdmin (can be opened from the CodeMeter Control Center).
2. Configure the license system in Viz StudioManager.
    a. When using a WIBU license server:

Copyright © 2026 Vizrt

i. Open the CodeMeter WebAdmin and add the license server to the server search list.
ii. Configure Studio Manger LIC_LOC to `network` or `network or any`.

b. When using a WIBU dongle:
i. Attach the WIBU dongle to any USB port of the machine.
ii. Configure to use legacy licensing. In the Studio Manager under **Tools Settings**, select **License**. Set LIC_LOC to `local`.

3. Click **Save** and restart.License Information

Tip: Information about the license can be found in the Studio Manager under **Info License ...**.

See Also

- **Client Configuration** page in the **License Server Setup and Administration** section of the Viz Licensing Administrator Guide.

Copyright © 2026 Vizrt

3.5 WIBU License System in Tracking Hub

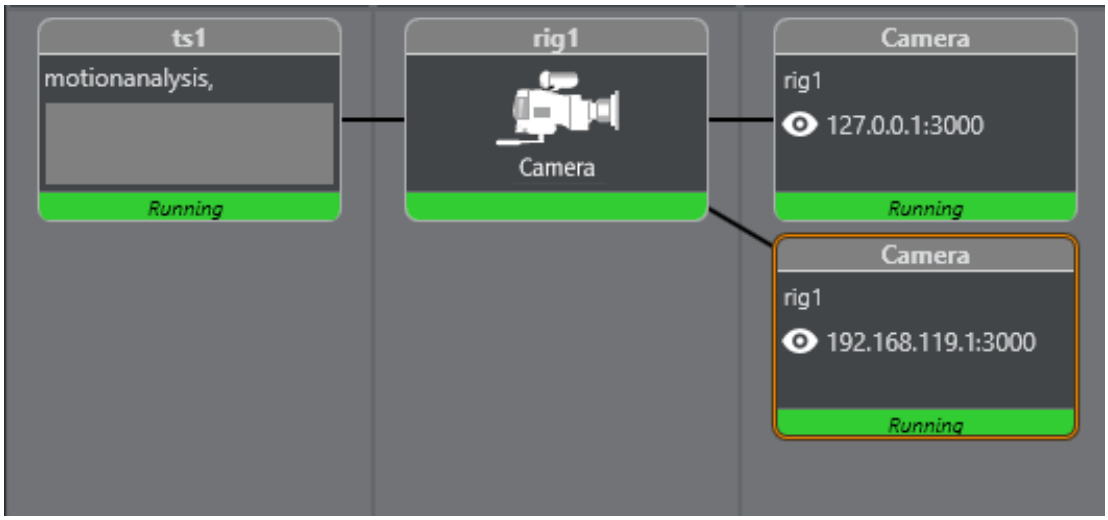
Tracking Hub supports the WIBU license system. This has the advantage that licenses can be held on a central server and licenses can be acquired by any machine on the company network. It is still possible to use WIBU Dongles on a Tracking Hub machine. The WIBU driver and the CodeMeter control application must be installed separately; they are not part of the Tracking Hub or Viz Virtual Studio Bundle installers.

To run a Virtual Studio, two licenses are needed: The first one is called *Viz Virtual Studio*. Without this license, Tracking Hub starts up, but does not create any running services. This is the base license, used to run the Tracking Hub and the Studio Manager. The second license is for cameras. This license is called *VS_CAMERA* or *Viz Virtual Studio Camera.*

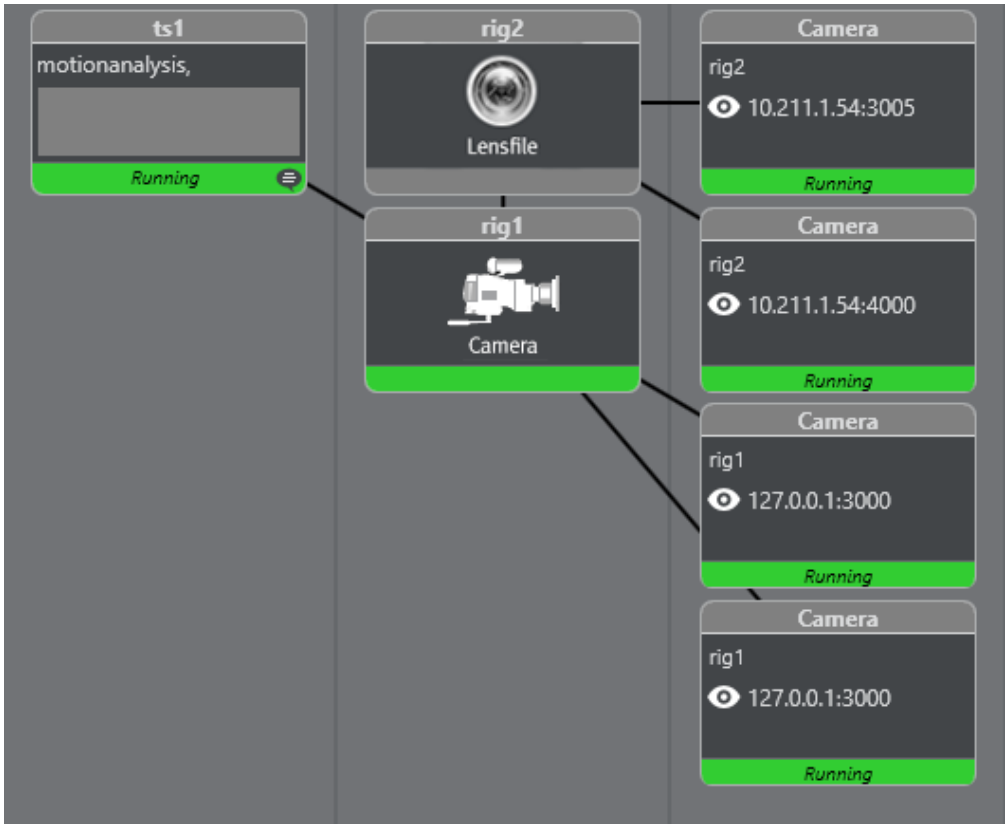
The *VS_CAMERA* license limits the amount of cameras which can be distributed by the Tracking Hub to the Engines or other Control Applications. It does NOT limit the camera services, or the virtual cameras, which are used in Viz. In Tracking Hub, a camera is represented as a connected combination of rig elements which calculate the CCD Position of a camera. It is possible to create one tracked camera out of two or more tracking systems, but one camera is always represented as one combination of rigs.

The *VS_CAMERA* license also limits the amount of rigs which can be connected to active Camera Services. It is still possible to create any amount of rig structures in the Studio Manager. Limited is the amount of rigs, which can be sent from Tracking Hub to the Engines in Virtual Studio.

3.5.1 One License Consumed

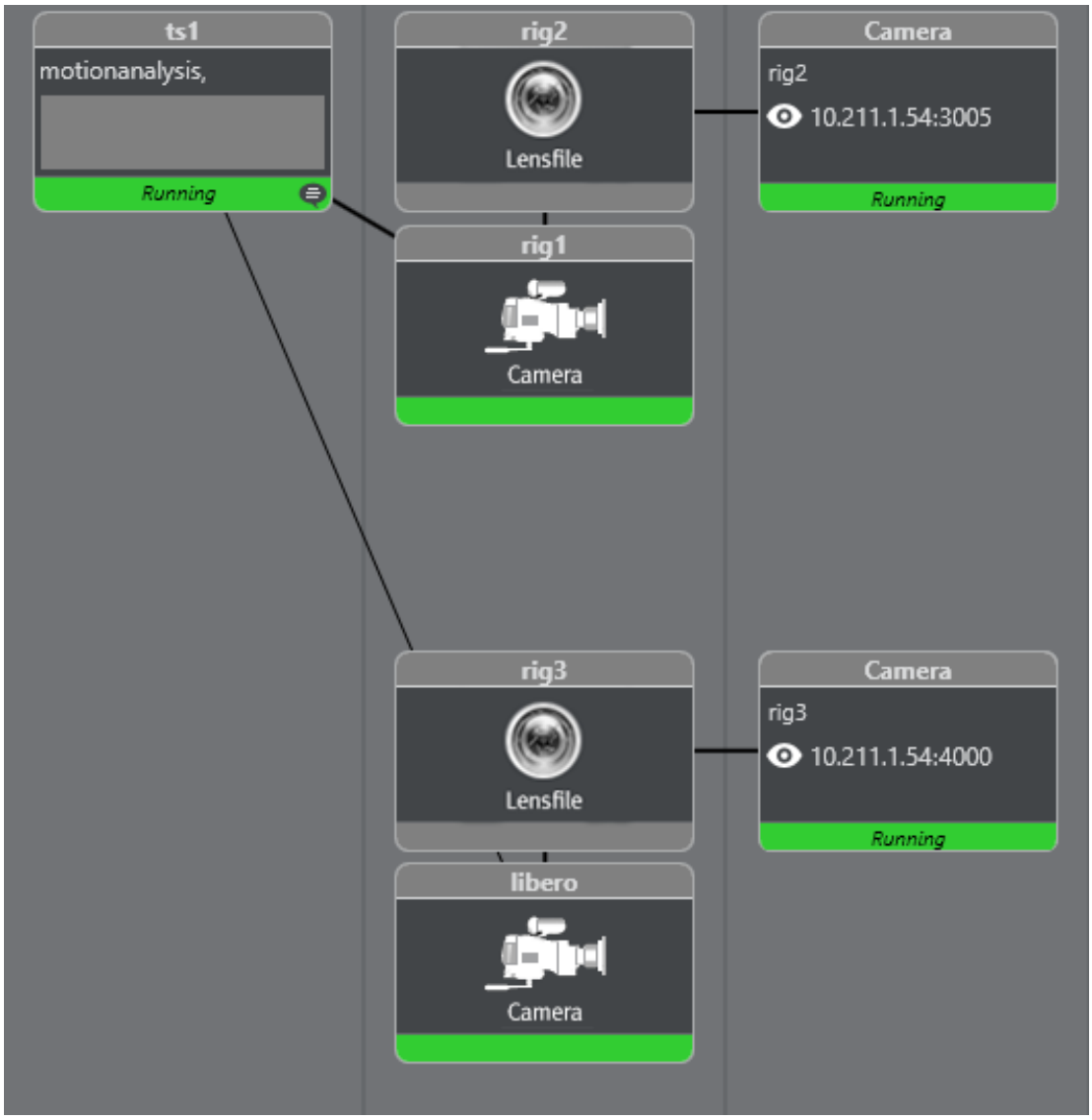


Copyright © 2026 Vizrt

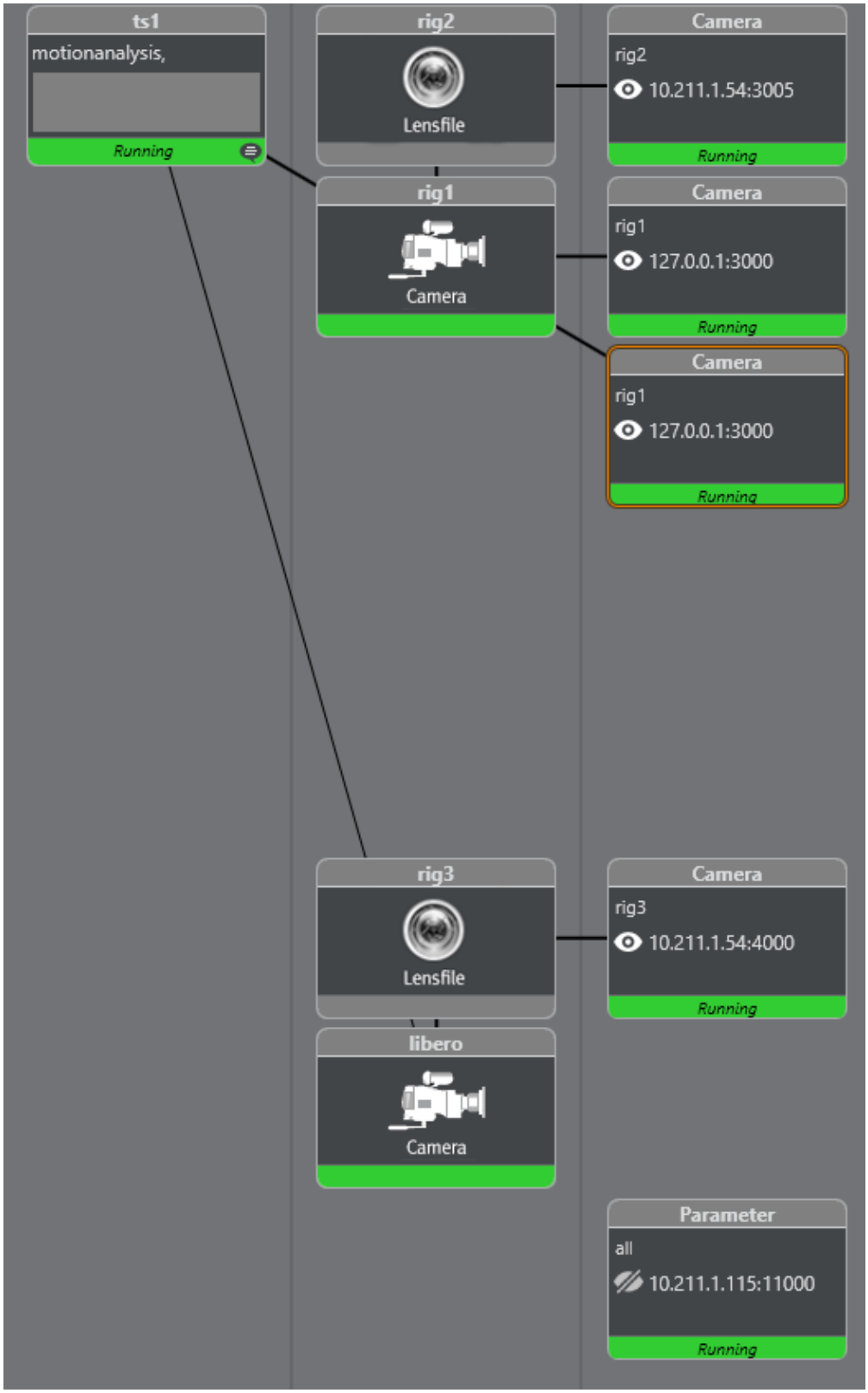


Copyright © 2026 Vizrt

3.5.2 Two Licenses Consumed



Copyright © 2026 Vizrt

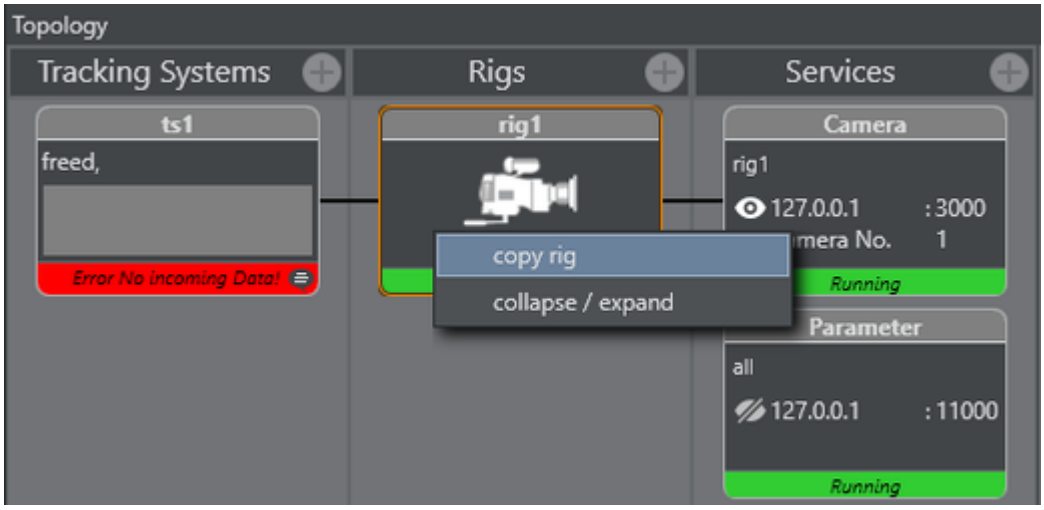


Copyright © 2026 Vizrt

3.5.3 Copy Rig Functionality

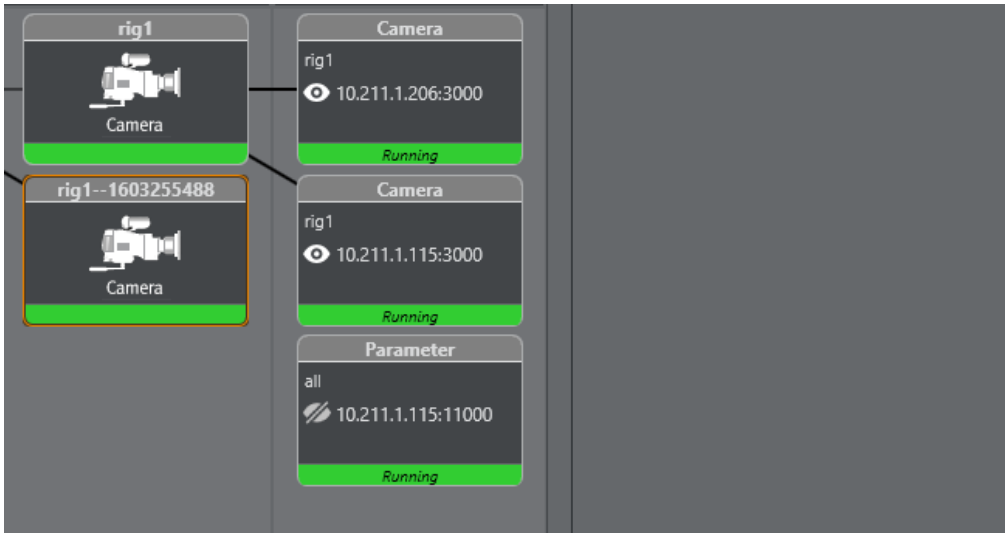
In some special cases, two identical rigs with the same parameters are needed, but with different delays. One example is the Engine/Unreal Engine 4 (UE4) configuration, where the camera data sent to the Unreal Engine camera has more delay the the Engine camera. In former Viz versions, two licenses were consumed.

The ability to copy a rig setup was introduced in Tracking Hub version 1.3.0 to make the UE4 configuration easier and only consume one license.



To create a rig copy, right click the rig you want to duplicate and click on the menu item **copy rig**. A duplicate of the rig is created. On this copy, all the parameter fields are empty. The only value which can be changed or modified is the overall delay.

Every change made to the original rig setup is automatically reflected in the copy.



Copyright © 2026 Vizrt

This copy is not counted in the license system.

Copyright © 2026 Vizrt

3.6 Tracking Hub Structural Design

This section covers the following topics:

- Protocol
- Parameters
    - Parameter Pool
- Rig Objects
    - Pivot Rotation
    - Location Pivot Rotation
    - Location Pivot Triple Rotation
    - Arm Rig
    - Translation
    - Crane Arm
    - Crane Head
    - Lens File
    - Camera Rig
    - Position and Rotation
    - Zoom and Focus
    - Center Shift
    - Mounting Offsets
    - Rotation Order
    - CameraXZY Rig
    - CameraYXZ Rig
    - CameraFD Rig
    - GimbalCam Rig
    - Extended Lens Parameters Rig
    - Object Rig
    - MoCap Rig
- Basic Camera
    - Nodal Point
    - Mounting Offsets
- Timing
- Delay
- Video Delay
- Tracking Delay

3.6.1 Protocol

The Protocol defines the data format sent from a tracking system to the Tracking Hub. It is independent from the transmission media and the geometry of the tracking system. The protocol generates tracked and named parameters, which are collected in the parameter pool. Protocols can be hard-coded in the Tracking Hub (like Motion Analysis), or as DLL files written with the plug-in API (available from version 2.0).

The XML Protocol Description was introduced in Tracking Hub 1.1. This protocol driver reads an XML file from either the hard drive or Graphic Hub. Out of this description, a parameter parser is generated which is able to read the

Copyright © 2026 Vizrt

data stream from the tracking system and translate it to named parameters, which are then sent to the Parameter Pool.

3.6.2 Parameters

A Parameter is a measured value of an encoder. It is extracted by the Protocol classes out of the Protocol data stream, the parameter is created by the Protocol. Its name is taken from the protocol description. The parameters in Tracking Hub are then converted from raw encoder values to human readable values, for example, the tick counts of a pan or tilt head are converted to degrees. The position tick counts are converted into centimeters.

Tracking delay and smoothing filters are applied to the parameter and can be adjusted per parameter.

Parameter Pool

All parameters are collected in the parameter pool. In the parameter pool, you find every encoder or optical tracked value.

3.6.3 Rig Objects

Rig objects (sometimes called Lattice objects) are used to represent geometry. The Tracking Hub separates *tracking* from *geometry*. All tracked parameters (axis) are collected in the *Parameter Pool*, as described in the Parameters section. Rig objects are built from hierarchical arranged sub elements where every sub element represents a mechanical or optical part of a tracked camera and its mechanical rigging.



Currently, the following rig elements are covered:

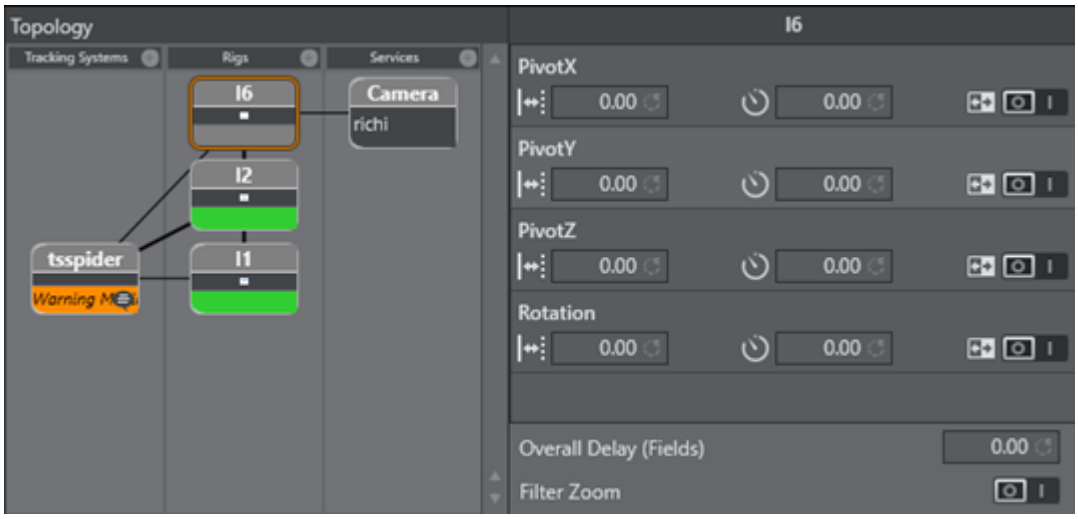
- • Pivot Rotation

Copyright © 2026 Vizrt

- Location Pivot Rotation
- Location Pivot Triple Rotation
- Camera Rig
    - Position and Rotation
    - Zoom and Focus
    - Center Shift
    - Mounting Offsets
- CameraXZY Rig
- CameraYXZ Rig
- CameraFD Rig
- GimbalCam Rig
- Extended Lens Parameters Rig
- Object Rig
- MoCap Rig
- Arm
- Translation
- CraneArm
- CraneHead
- Lensfile

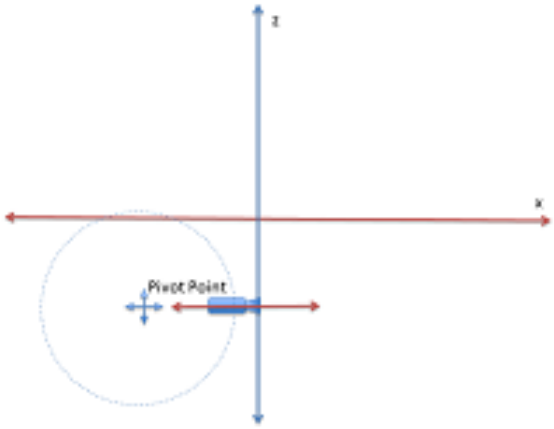
Pivot Rotation

Rotates the coordinate system of a rig around a free definable point in space. The camera Rig is bound to the pivot point in a parent-child relationship.



PivotX, PivotY and PivotZ define the location of where the rotation takes place. Rotation specifies the rotation in degrees around the Y Axis of the pivot point.

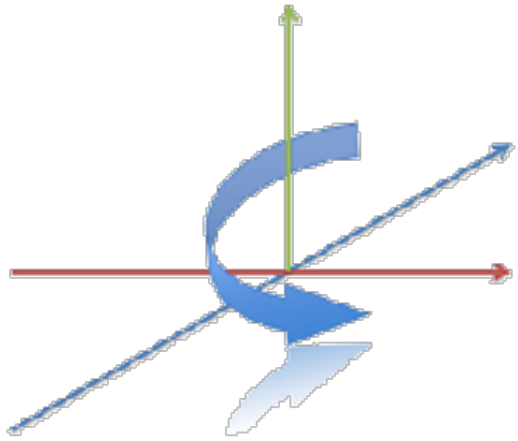
Copyright © 2026 Vizrt



After the rotation, the coordinate system of the Rig is rotated around the Pivot Point. This is useful if, for example, a rail tracker needs to be adjusted in angle and offset to a studio coordinate system.

Copyright © 2026 Vizrt

Location Pivot Rotation



The location pivot rotation rig is similar to the Pivot Rotation. The difference is that this rig takes the pivot point coordinates from the child camera's position offset, which is attached to the camera position. This rig accepts only a rotation angle, as the position is taken from the child position offsets.

Location Pivot Triple Rotation

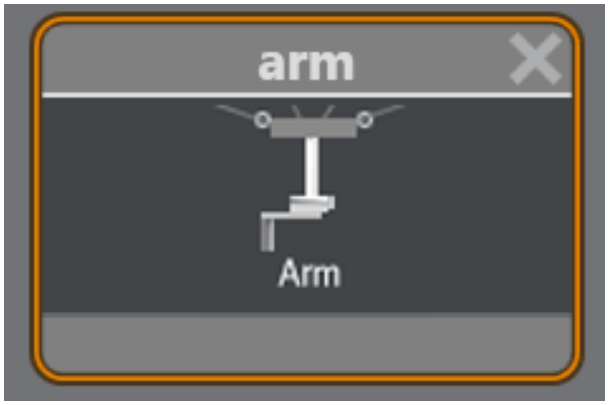
The Location Pivot Triple Rotation rig behaves like the Location Pivot Rotation rig, taking its pivot point from the child camera's position offset, but it accepts three rotation angles instead of one. RotationX, RotationY and RotationZ are applied in the order Z, X, Y.

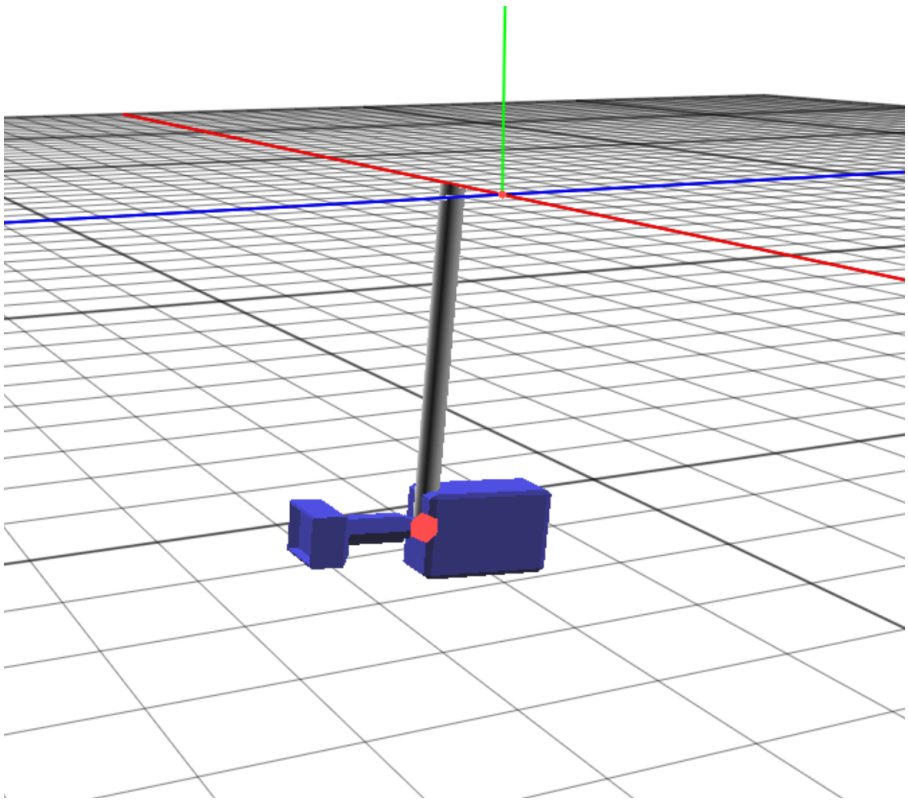
Use this rig where a mount needs to be aligned to the studio coordinate system in more than one axis, for example a rail or a rig plate that is not level.

Arm Rig

The Arm rig is mainly used for Spidercam setup, or anywhere a mounted arm follows a rotation axis. The rig does not include any head length or angle adjustments to the next child. All rotations that go into the Tilt, Pan and Roll axis are given to the next sibling. It is possible to set JointX, JointY and JointZ values to adjust mounting offsets between the position giving element and the arm.

Copyright © 2026 Vizrt





The arm length coordinates define the direction and length of the arm.

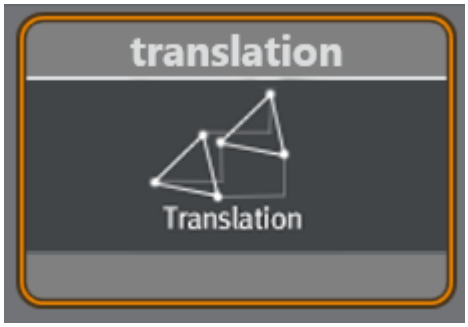
Copyright © 2026 Vizrt



Translation

The Translation rig is mostly used as parent rig to an Arm or CraneArm Rig. It lets you modify the position of all child rigs. **PosX**, **PosY** and **PosZ** can be used to track the position. It is a usual element in a Spidercam setup.

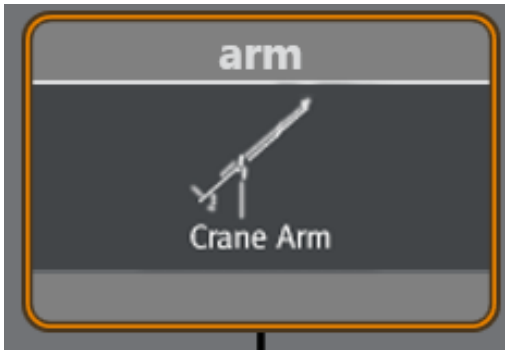
Copyright © 2026 Vizrt





Crane Arm

The Crane Arm rig is used to represent a crane arm. The difference from the normal Arm Rig is that the direction fields define a normalized direction of the arm, and the length is defined by the **ArmLength** field. In the arm rig, the **Length** parameters can be separated by axis.



You can set **JointX**, **JointY** and **JointZ** values to adjust mounting offsets between the position giving element and the arm.

Copyright © 2026 Vizrt



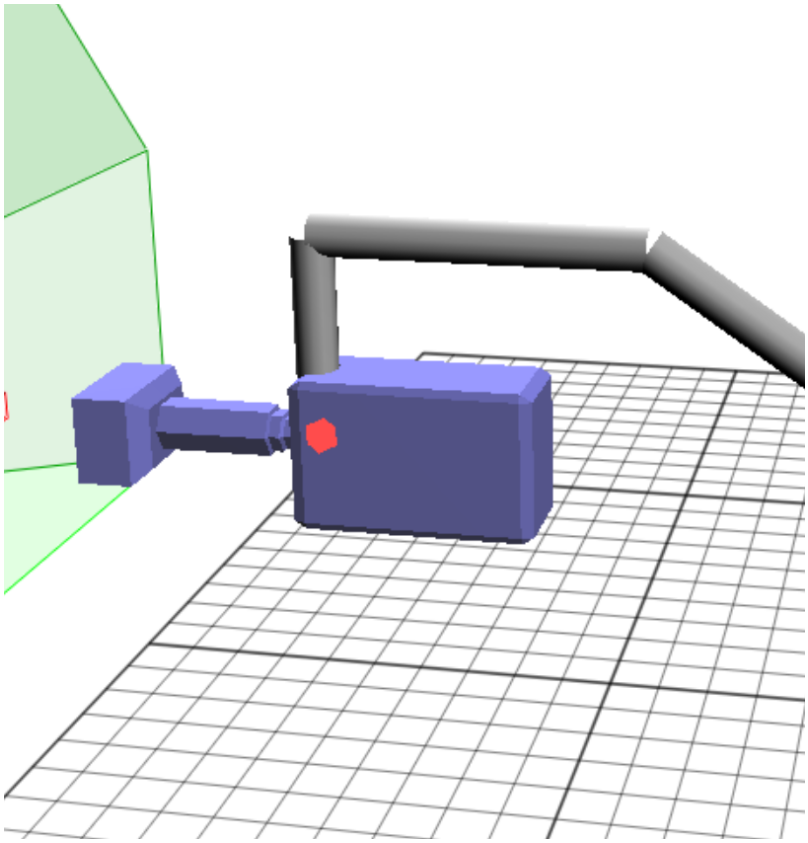
ArmdirectionX, **ArmdirectionY** and **ArmdirectionZ** are normalized to `1` and define the initial direction of the Arm. Usually, **ArmdirectionZ** is set to `-1`.

Crane Head

A Crane Head represents a perpendicular element, which can be attached to a crane arm, or to a simple arm. The **Headlength** defines the horizontal length in -Z direction after the perpendicular element. The **Headheight** defines the Height in -Y direction after the perpendicular element.

Copyright © 2026 Vizrt





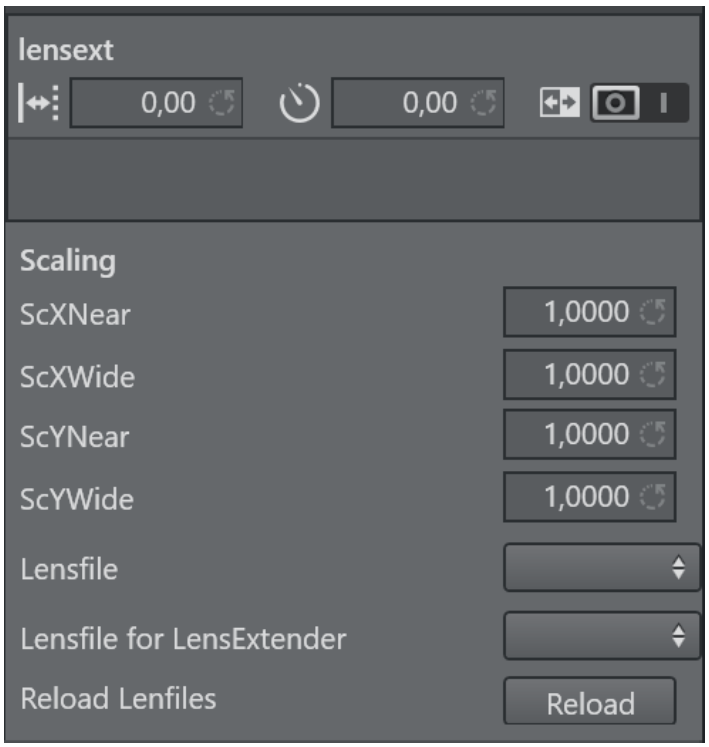


Copyright © 2026 Vizrt

Lens File



The Lens File rig must always be the first in a rig hierarchy. It is able to select one of the lens files, which must be located in the **Lensfile** folder in the Tracking Hub configuration folder in *%ProgramData%*. Once in the hierarchy, the lens file interprets the incoming raw zoom and raw focus values and sends FieldOfView, K1, K2, CenterShift and NodalPoint to the engine. The camera service object must be targeted to the lens file rig.



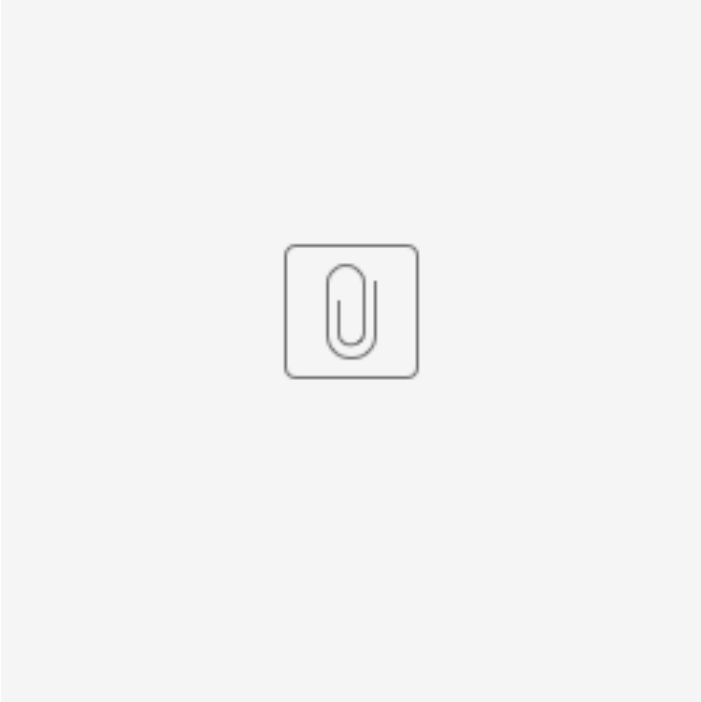
The only trackable parameter in the rig is the **lensext** parameter. In some protocols, such as **FreeD**, the lens extender is tracked. The value of **lensext** can be `0` or `1`. If it is `0`, the first lens file is used. If set to `1`, the second lens file is used. The scaling values define a scale factor for the field of view with respect to the actual zoom value. **ScYNear** and **ScXNear** are the scaling factor when the lens is completely zoomed in. **ScYWide** and **ScXWide** define the scaling factor when the lens is completely wide. Between Wide and Near, the values are interpolated. If you do

Copyright © 2026 Vizrt

any changes inside the lens file or add some new, you must press **Reload** to make them available in your configuration.

Camera Rig

The Camera Rig represents the simplest camera known in the Tracking Hub. It holds the rotation and position parameters of a real camera in the studio. Zoom and Focus (in raw format coming from an encoder) and the Front, Height and Right lens shifts which are coming from the mount of the camera.



Position and Rotation

Tilt, Pan and Roll defining the rotation of the camera around the X,Y and Z Axis. PosX, PosY and PosZ define the position of the camera in space. Each of these values might be tracked. In the case of a Pan/Tilt camera head which does not provide position information, the position can be adjusted in the offset values.

Zoom and Focus

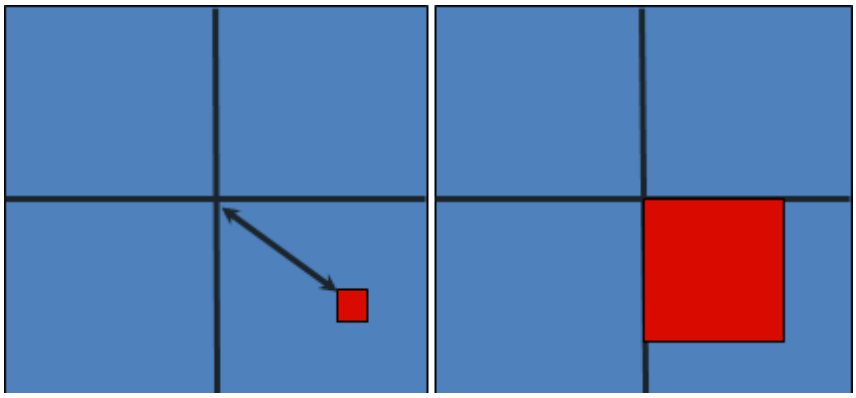
Zoom and Focus come from the Lens encoders (whether internal or external). In the system and in Viz Engine, these values are always normalized and go from `0` to `1`. Therefore, it is most important that the Lens Ranges are calibrated correctly in the tracking system.

Copyright © 2026 Vizrt

Center Shift

Whenever you mount a lens to a camera body, the connection is never 100% the same every time; hence, there is always an offset between the camera body and the lens which shifts the angle of the light onto to charge-coupled device (CCD). In the Viz Virtual Studio software, this effect is called the Center Shift.

Note that every lens shows its own center shift and can for the most part be ignored as it is much smaller than the center shift caused when mounting the lens to the camera body. The most visible center shift is caused by the actual mounting of the lens and is more and more visible as the mount connection between the camera body and the lens becomes damaged/worn.



To calibrate the center shift, the Camera Rig handles the *CenterX* and *CenterY* offsets the following way: Completely zoomed out the full offset is applied and sent to Viz Engine. When zooming in, these values go in the zero direction until they reach absolute zero (completely zoomed in). It is **absolutely necessary** to calibrate this center-shift to get an acceptable tracking result. Whenever you see a movement of virtual objects when zooming in and out, you need to check if the center shift has changed.

The center shift can be corrected in the lens file only!

Note: Always check the center shift after mounting your lens to the camera body!

To calibrate the center shift, follow the algorithm below:

1. Reset the CenterX and CenterY values to zero.
2. Look for a corner or easy to identify point in the camera view.
3. Enable the Center Shift Cross in Viz Artist, see **Scene Settings > Virtual Studio**.
4. Zoom in completely to this point and adjust focus. This is also a good point in time to check the back focus of the camera.
5. Pan and tilt the camera until the corner or point aligns with the Center Shift Cross.
6. Zoom out completely and see that the point slides away from the center.
7. Adjust the CenterX and CenterY values until the corner or point aligns with the Center Shift Cross again.
8. Zoom out completely. If the Center Shift Cross is no longer in the corner or point, repeat from step 5.

Note: Repeat the steps until the Center Shift Cross is on the corner or point at all times. After two or three iterations, this is usually the case.

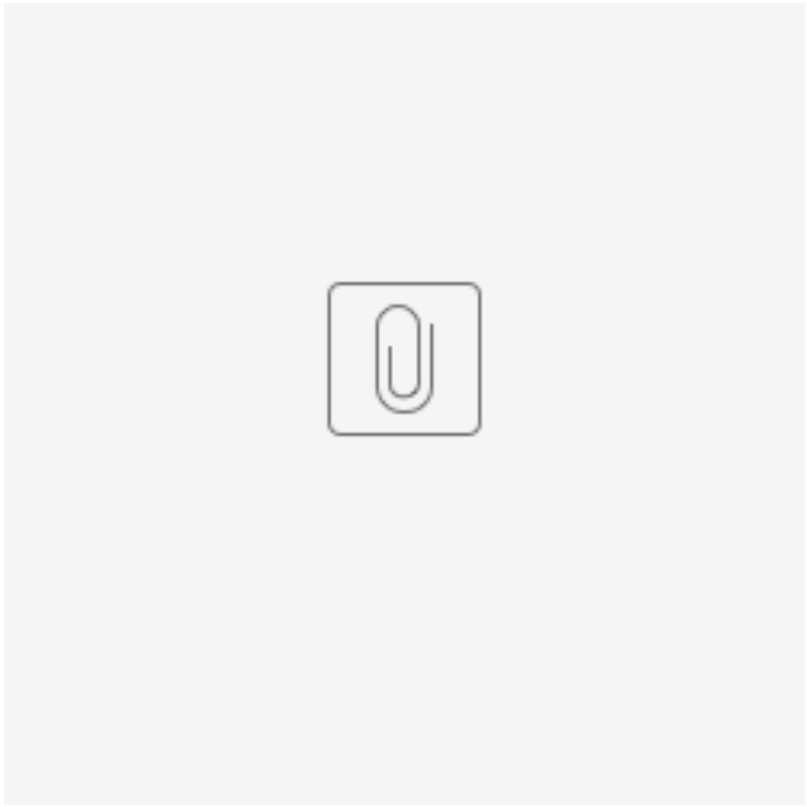
Copyright © 2026 Vizrt

Mounting Offsets

In order to get an acceptable result, we need to know the exact position of the camera's charge-coupled device (CCD) position. The CCD is (after the optical nodal point) the position of the Viz Engine remote camera. Not every tracking system gives that position. More likely, the position of the last rotation axis on the mechanical head is given, which in most cases is the tilt axis. If the send position is not clear, you need to contact the vendor of the tracking system.

After mounting a camera to a head, there are always offsets between the tracked point and the CCD of the camera. Many cameras have a mark on their body which marks the position of the CCD. If no mark is visible, you can assume that the CCD is 2 cm behind the mounting point of the lens.

In Tracking Hub, we know three mounting offsets. The *FrontLensShift*, *HeightLensShift* and the *RightLensShift*. The following drawings show the Front and Height Lens Shift and how these values change the behavior of the CCD.



Copyright © 2026 Vizrt



Rotation Order

A tracking system delivers Pan, Tilt and Roll as three separate angles. Three angles alone do not define an orientation: the result depends on the order in which they are applied, because rotations are not commutative. The order is not a matter of taste, it mirrors how the axes of the real head are mounted inside each other. The axis applied last is the outermost one, the one closest to the pedestal or mount.

This is why the Tracking Hub offers several camera rigs which differ only in rotation order. Choosing the wrong one produces a result that looks correct as long as at most one angle is non-zero, and drifts as soon as the operator tilts and pans at the same time. A tilting horizon when panning is the typical symptom.

The table lists the order in which the rotations act, starting at the camera. Read the first entry as the rotation applied to the camera first, and the last entry as the outermost axis.

Rig	Rotation order	Typical mechanics
Camera	Roll, Tilt, Pan	Standard pan/tilt head: the base pans, the head tilts, the camera rolls
CameraXZY	Tilt, Roll, Pan	Roll axis mounted between the pan and the tilt axis
CameraYXZ	Pan, Tilt, Roll	Pan axis mounted innermost, for example a head on a nodding arm

Copyright © 2026 Vizrt

Rig	Rotation order	Typical mechanics
CameraFD	Tilt, Roll, Pan	As CameraXZY, but roll and pan also displace the camera, see CameraFD Rig

All four rigs expose exactly the same parameters. They differ in rotation order only, with the exception of CameraFD, which additionally couples rotation to position.

CameraXZY Rig

The CameraXZY Rig holds the same parameters as the Camera Rig and differs only in its rotation order, which is X Z Y: Tilt is applied first, then Roll, then Pan. Use this rig when the roll axis of the head is mounted between the pan and the tilt axis.

Tilt, Pan, Roll, PosX, PosY, PosZ, Zoom, Focus, CenterX, CenterY and the Front, Height and Right lens shifts behave exactly as described under Camera Rig.

CameraYXZ Rig

The CameraYXZ Rig holds the same parameters as the Camera Rig and differs only in its rotation order, which is Y X Z: Pan is applied first, then Tilt, then Roll. This places the pan axis innermost, which is the case when the head itself is carried by a tilting element such as an arm or a crane.

Tilt, Pan, Roll, PosX, PosY, PosZ, Zoom, Focus, CenterX, CenterY and the Front, Height and Right lens shifts behave exactly as described under Camera Rig.

CameraFD Rig

The CameraFD Rig was developed for the Spidercam fielddolly (FD). Its rotation order is X Z Y, as in the CameraXZY Rig, but it differs from every other camera rig in one important respect: the roll and pan axes are not treated as pure rotations of the camera. They are treated as arms, so rotating them also displaces the camera position. The rig applies a fixed offset of 3.5 cm between the tilt axis and the pan axis to model the gap of the physical dolly head.

Because of this coupling, CameraFD is not interchangeable with CameraXZY even though both use the same rotation order. Use CameraFD only for a fielddolly setup.

GimbalCam Rig

The GimbalCam Rig represents a gimbal-mounted camera, as used for aerial and cable-suspended systems. Instead of Pan, Tilt and Roll it exposes a five-axis chain following the GSS Geopoint System ICD: an inner gimbal (In_Elevation, In_Azimuth), a Roll axis, and an outer gimbal (Elevation, Azimuth).

Zoom, Focus, CenterX, CenterY and the Front, Height and Right lens shifts behave as described under Camera Rig.

Extended Lens Parameters Rig

The Extended Lens Parameters rig holds parameters which are rarely used and delivered by TrackMen and Libero only.

Copyright © 2026 Vizrt

Object Rig



The Object rig is used to position an object in Viz Engine with the use of shared memory and the object service. It can be used to mask out pedestals, which would be in view of other cameras or to track objects with the Motion Analysis tracking system. Pan/Tilt/Roll are the rotation angles of the object and *PosX*, *PosY* and *PosZ* are the position in centimeters (cm). Connected to an Object-Service the coordinates are sent to Viz Engine and can be applied to any objects in the Viz Artist scene tree by a script.

MoCap Rig

The MoCap Rig is the entry point for motion capture data. Unlike the other rig objects it does not calculate any geometry: it carries a single mocap axis and passes the skeleton data through from the parameter pool, including the body definition and the per-bone transforms. The rig is used together with the mocap service, which forwards the data to Viz Engine.

Because no geometry is calculated, the MoCap Rig has no visual representation in the Studio Manager 3D view.

3.6.4 Basic Camera

The Basic Camera rig object has to be attached at the end of every rig structure. The most basic and commonly used parameters required for exact tracking are stored in this rig object. This section shows how the Tracking Hub needs to place the idealized OpenGL point camera to create an approximation of the physical studio camera.

Copyright © 2026 Vizrt

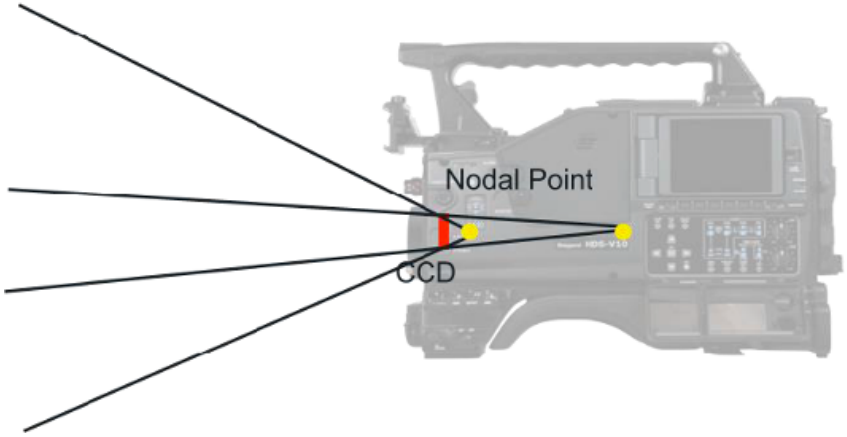


Item	Description
1	Clip space
2	Light Space
3	Camera Space
4	Image Space
5	Object Space

By default, the OpenGL camera is a point in space. All transformations (perspective and model view) are linear. It does not matter how much the zoom is increased in the perspective, or how near the camera is to an object in the model view transformation: A straight line in the model space is a straight line in the image projection. When the camera is rotated, it rotates around its position.

Copyright © 2026 Vizrt

Nodal Point



Let us assume the OpenGL camera is placed on the charge-coupled device (CCD) of the camera body. In the image, above, it completely depends on the field of view, where the light rays reflected from an object conjure to a point and create a sharp picture on the CCD. For sports, where the real camera is far away from the observed objects (we assume distances of more than ten meters), there is no influence from the adjusted focus. Therefore, in the image above we only observe the zoom effects on the nodal point of the objects watched.

Zooming in and out, causes the light rays, going through the lens, to conjure on a moving point behind the lens. In our Viz Virtual Studio software, we call this point the *Nodal Point*, and this point marks the position where we have to place our idealized OpenGL camera. The Nodal Point is defined during lens calibration, which is still happening in the Viz Engine. Therefore it is not an error, if you zoom in and out, and the position of the camera is moving towards and away from the Look-At direction of the camera. This is intended and must be done to provide an accurate result.

Note: Currently the Nodal Point comes from the lens file only.

Copyright © 2026 Vizrt

Mounting Offsets



Item	Description
1	CCD
2	Height Lens Shift
3	Tilt Axis
4	Front Layer

3.6.5 Timing

As a prerequisite for a virtual studio, every part of the equipment must be synchronized. This includes the Viz Engines, the tracking systems and the Tracking Hub.

The Tracking Hub can be synchronized in two ways:

- Using a synchronized Viz Engine as sync source, or
- A Plura PCL-PCI card.

Copyright © 2026 Vizrt

It is the possible to switch the Tracking Hub to freerun mode. This is not intended for production purposes, rather it is meant for emergencies and experiments. **Never** under any circumstances use freerun mode for production. Even in freerun mode, the Tracking Hub generates a smooth result, but a change in delay becomes visible periodically. This occurs when the free running Tracking Hub swaps over the field border of a synchronized Viz Engine.

It is possible to run Tracking Hub in low latency mode. This is the case when the overall delay is set to a value below `1` in the rig. If you see missing data packages (indicated by warning displayed on the tracking system) you can increase the delay to values over `1`. At that point, Tracking Hub starts to interpolate missing tracking packages. If, from time to time, one data package is missing, a delay value between `1` and `2` is enough to smooth the data. When more than one package is missing (two successive missing packages), the delay must be increased accordingly. Even if this is possible with the Tracking Hub now, we want to achieve as low delay as possible. So if you see missing packages, you need to check the connection to the tracking system. Check if the system is synchronized in a correct way or find out what is causing this wrong behavior.

Use the *timing monitor* in Studio Manager to check the timing behavior of a tracking system.

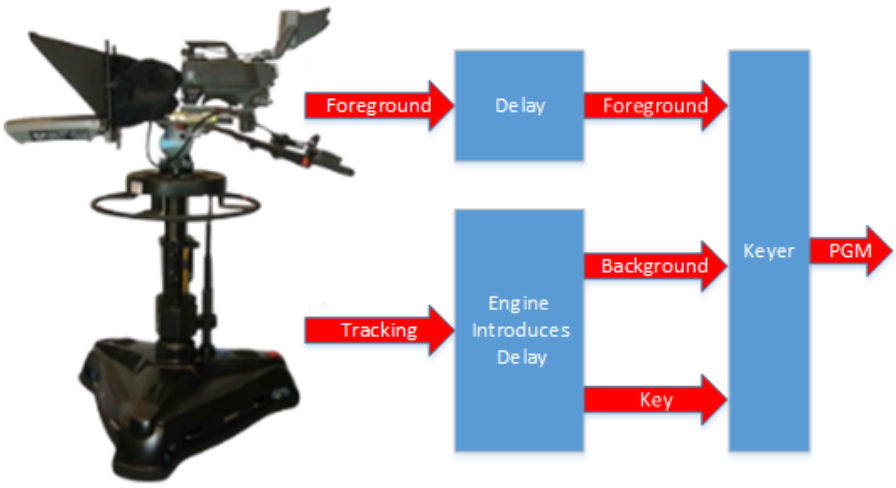
3.6.6 Delay

The responsible technicians are confronted with delays between the incoming tracking data and the incoming video signals when a virtual studio is set up. We do not handle audio delays here, because audio is, in most cases, handled by the TV station’s audio department.

We handle two different kinds of delays in the virtual studio:

- Video Delay
- Tracking Delay

3.6.7 Video Delay



It is assumed that the camera and the tracking system are connected to the same sync signal. When the camera records an image, the tracking system acquires the camera's position, does some calculations and then sends this data to the Tracking Hub. One field later, Tracking Hub sends this tracking information to Viz Engine. Unfortunately,

Copyright © 2026 Vizrt

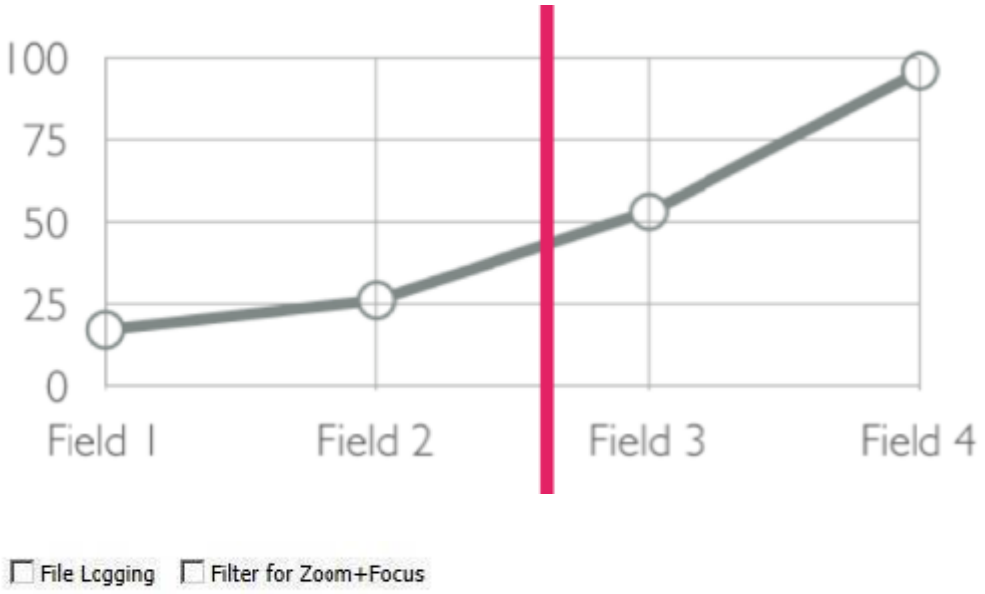
Viz Engine needs time to first render the image and additional time to output this image through the video card to its output. Normally, this process needs about two or three frames, depending on the video hardware used.

If we route the image from the camera directly to the keyer, the video is presented before the rendered image from Viz Engine. Therefore, delaying the video for a specific time is required to achieve a close match between the real image (i.e. the foreground in the virtual set) and the rendered image (i.e. the background in the virtual set). To apply this video delay, it is (or was) required to introduce a delay line which delays the video for a specific amount of frames.

Note: In interlaced modes, video can not be delayed for fields. Video delays can only be applied on a frame base.

Modern Keyers, such as the Ultimatte 11 or the Saphire 3 have built in delay lines. An extra delay line must be installed when using older hardware.

3.6.8 Tracking Delay



After reading the last section about Video Delay, you may have the impression that tracking systems and cameras sample the position or record the image exact at the flank of the sync signal (blackburst). This is not the case. Depending on the *exposure time*, *iris* and *gain* we do not know when the camera is recording the image. We only know for certain that the camera sends the image out at the next sync. But we don't know at what time during the field the image was recorded. Still worse, if you have a long exposure time (for example, two milliseconds) where is the point in time, the image was recorded?

The same problem occurs with tracking systems. Mechanical tracking systems are always near the sync flank, but they are never exactly at the flank. They always need a few milliseconds to read all the encoders and calculate the position from them. Using optical tracking systems compounds these problems. At what point in time are the cameras shooting the targets? We do not know. The only thing we know for certain is that there is always a fraction of a field delay between the tracking data and the video data, even when the video delay is set as accurately as possible.

Copyright © 2026 Vizrt

From the last chapter we know, that video delay can only be set in whole values of frames. How do we set the fraction of a field?

> **Note:** When the delay differs 0.1 field from the video, it is visible to the eye!

The only way to solve this problem is to **interpolate** using the tracking data received by Tracking Hub. There is an Overall delay setting in the Studio Manager Rig control that can be used to interpolate the tracking data in fractions of fields. In addition, every axis can have its own delay bias. This is useful, when mixing different tracking systems. When the overall delay is set to `0.5`, the Tracking Hub interpolates a half field back in time.

But what can you do if the delay must be set to -0.5 to achieve a good result? This is not possible. We cannot attempt to extrapolate the data, because this probably results in ugly jumps. In this case, the only solution is to increase the video delay.

Copyright © 2026 Vizrt

4 Installation and Startup

This section details the installation requirements, the installation and system startup and shutdown. Before you install and configure the software, you should plan the installation.

This section contains the following:

- Important Checklist Before Installation
- Viz Virtual Studio Folders
- Viz Virtual Studio Installation
- Start Viz Virtual Studio

Copyright © 2026 Vizrt

4.1 Important Checklist Before Installation

The items listed below must be checked before an installation can be done on site. Before an installation can be started make sure:

- Camera adjustments/shifts should always be set by camera personnel before any lens calibration is done. After any adjustments/shifts are set, the camera cannot be adjusted without being reconfigured.
- Cables: Who is in charge of the cabling before doing an installation?
- Tracking: What are the tracking system protocols?
- Transfer of tracking data: How is tracking data transferred and what kind of cables are used?
- Synchronization sources: What type of synchronization sources used?
    - Blackburst
    - Tri-level
    - In case Viz Engine is selected as the sync source, Tracking Hub must run on the same machine. Then the following sources can be selected in Viz Engine:
        - SDI input 1 or 2
        - NDI video source (input 1 or input 2)
        - PTP
- Location of the Viz Engines: Do you have physical access to the machines?
- Coordinate system: What are the coordinate systems for the tracking system(s) used?
- Make sure that there are no firewalls enabled.

4.1.1 System Requirements

This section details the system requirements.

Software

Wibu Codemeter

- The Virtual Studio Bundle installer does not contain Wibu Codemeter. It needs to be installed seperately.

Synchronization

The Tracking Hub needs some sort of synchronization method on the machine it is installed. This can be:

- A running Viz Engine installation, or
- An installed Plura synchronization card.

Note: If sync sources is not found, Tracking Hub can still be used in free running mode. Free running is **not** intended and should not be used for production. It can, however, be used to experiment with tracking on a laptop. For use in production, either Viz Engine or Plura synchronization is necessary.

Note: 32-bit Operating Systems are no longer supported. For versions 1.2 and later, Tracking Hub and Studio Manager are available in 64-bit only.

Copyright © 2026 Vizrt

Minimum Hardware Configuration

- Minimum 16 GB Ram
- Dual-Core Intel CPU
- 10 GB of free disk space
- NVIDIA P2000 Graphics Card
- OpenGL Hardware-accelerated Graphics Card

Info: Minimal Hardware Configuration provides sufficient resources for running a Tracking Hub configuration.
The automated lens file calibration requires additional GPU processing power. To benefit from this feature, use the Recommended Hardware Configuration.

Recommended Hardware Configuration

- HP Z4 G4 - 12-Core CPU
- 64 GB Ram
- 2x Gbit RJ45
- 30 GB of free disk space
- NVIDIA RTX 4000 Ada

Setup Required for NDI Workflow

- HP Z4 G4 - 6-Core CPU
- 32 GB Memory
- 2x Gbit RJ45
- 256 GB SSD for OS
- 512 GB SSD for Data
- NVIDIA RTX A4000 /16 GB GDDR6

Supported Hardware and Software

- Windows 10 (64-bit) LTSC
- Viz Engine 3.9.0 (either 32-bit or 64-bit) or later, recommended Viz Engine 4.2 or higher
- Plura PCL-PCI and PCL-PCIe (L, LV, 3G) timecode reader cards (Driver 5.45; 5.34 also tested)

Important: The PCL-PCI L card can only read timecodes and can NOT be used for synchronization.

- Studio Manager requires .NET framework 4.7.2 or higher.

Copyright © 2026 Vizrt

NVIDIA Graphics Driver Setting

Warning: On some machines with an NVIDIA graphics card, the memory (RAM) used by Studio Manager increases steadily while its 3D preview is shown, by several hundred MB per hour, until the system runs low on memory. The cause is the NVIDIA driver setting Threaded optimization. It does not happen on every machine or on every start. On machines with an NVIDIA graphics card, the Studio Manager installer sets Threaded optimization to Off for StudioManager.exe, and Studio Manager shows a warning at start-up when the setting is on.

To turn Threaded optimization off for Studio Manager by hand, for example when that warning appears:

1. Right-click the desktop and open the NVIDIA Control Panel.
2. Go to 3D Settings > Manage 3D settings and open the Program Settings tab.
3. Under Select a program to customize, click Add and select StudioManager.exe (by default C:\Program Files\Vizrt\StudioManager\StudioManager.exe).
4. In the list of settings, set Threaded optimization to Off.
5. Click Apply, then restart Studio Manager. The setting only takes effect after a restart.

See Also

- Viz Virtual Studio Installation
- Start Viz Virtual Studio
- Supported Protocols
- Supported NDI Cameras
- Troubleshooting

Copyright © 2026 Vizrt

4.2 Viz Virtual Studio Folders

4.2.1 Installation Folders

The default installations folders are

- *C:\Program Files\vizrt\VizTH*
- *C:\Program Files\vizrt\Studio Manager*

Files which are created or can be modified are located in `%ProgramData%\vizrt\VizTH`. Tracking Hub creates the following subfolders when it starts, so they are present on every installation:

Folder	Contents
Cfg\	Studio configurations. Each configuration has its own subfolder, holding the cyc, video wall and router XML files that belong to it.
Lattices\	Rig definitions, one XML file per rig, named `XML_LA_Cfg_<rig name>.xml`.
Lensfiles\	Lens files. A Lensfile rig references a file in this folder by name. This is the folder the Lens File Editor reads and writes.
Lutfiles\	Look-up tables used for color correction.
Post\	Recorded tracking data written and read back by the Post system.
Targetcalibration\	Reserved for target calibration data.
Templates\	Studio templates, one XML file per template.
XMLTracker\	XML protocol descriptions used by the XML Tracking protocol.

The root of `%ProgramData%\vizrt\VizTH` itself holds:

- `VizTH.conf` — the main configuration file. If it is missing, Tracking Hub writes a new one with default settings on the next start.
- Crash dump files.
- Debug log files (`VizTH_send*.log` , `VizTH_rcv*.log` , `VizTH_timing*.log`) when the corresponding logging is switched on.

Copyright © 2026 Vizrt

A second location, `%LOCALAPPDATA%\vizrt\VizTH`, holds the startup configuration. It is created on startup in the same way.

Temporary files are located at: *%TMP%\vizrt\VizEngine*, which usually resolves to *C:\Users\<user name>\AppData\Local\Temp\vizrt\VizEngine*. This folder is referenced as *<viz temp folder>* throughout this User Guide.

Copyright © 2026 Vizrt

4.3 Viz Virtual Studio Installation

There are several setup applications available to install the components of Viz Virtual Studio, as indicated in the table below:

Setup Application	Content
VizVirtualStudioBundle-x64-<VERSION>.exe	Bundle with Tracking Hub and Studio Manager
StudioManager-x64-<VERSION>.msi	Studio Manager installer
TrackingHub-x64-<VERSION>.msi	Tracking Hub installer

It is recommended to install Studio Manager and/or Tracking Hub using the Virtual Studio Bundle setup application, as this installs the prerequisite software as well.

The bundle installs the packages listed below. A package that is already present on the machine is detected, and its check box is disabled.

4.3.1 Prerequisites

- Microsoft Visual C++ 2005 Redistributable (x64)
- Microsoft Visual C++ 2008 Redistributable (x64)
- Microsoft Visual C++ 2010 Redistributable (x64)
- Microsoft Visual C++ 2012 Redistributable (x64)
- Microsoft Visual C++ v14 Redistributable (x64), which covers the 2015 to 2022 releases

Note: CodeMeter Runtime Kit, for WIBU licensing is not part of the Bundle staring with version VVS 1.9.0. It needs to be installed seperately.

4.3.2 Applications

- Tracking Hub
- Studio Manager
- Viz AI Core
- Viz AI Core Plugin
- Viz AI Lens Calibration Plugin

4.3.3 .NET Framework

Studio Manager requires Microsoft .NET Framework 4.7.2 or higher. The Studio Manager installer offers to download it if it is not already present.

Copyright © 2026 Vizrt

The bundle setup application itself requires .NET Framework 4.6.2 or higher in order to run, so on a machine that does not meet the Studio Manager requirement the bundle cannot be started either.

4.3.4 To Install Tracking Hub and Studio Manager Using the Bundle Installer

1. Log on to the computer as a Computer Administrator and start *VizVirtualStudioBundle-x64-<VERSION>.exe* to run the setup application.
2. By default, the Tracking Hub and Studio Manager options are selected in the installer. If required, remove the options.

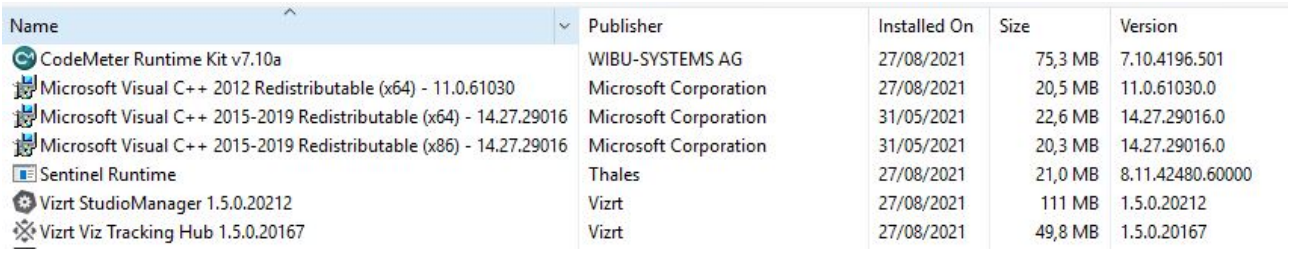


3. Click **Install** and follow the on-screen instructions.

> **Note:** The packages to install can be selected by check boxes. If a package is already installed, the check box is disabled.

After installation, the **Programs and Features** panel shows entries for each installed package

Copyright © 2026 Vizrt



Name	Publisher	Installed On	Size	Version
CodeMeter Runtime Kit v7.10a	WIBU-SYSTEMS AG	27/08/2021	75,3 MB	7.10.4196.501
Microsoft Visual C++ 2012 Redistributable (x64) - 11.0.61030	Microsoft Corporation	27/08/2021	20,5 MB	11.0.61030.0
Microsoft Visual C++ 2015-2019 Redistributable (x64) - 14.27.29016	Microsoft Corporation	31/05/2021	22,6 MB	14.27.29016.0
Microsoft Visual C++ 2015-2019 Redistributable (x86) - 14.27.29016	Microsoft Corporation	31/05/2021	20,3 MB	14.27.29016.0
Sentinel Runtime	Thales	27/08/2021	21,0 MB	8.11.42480.60000
Vizrt StudioManager 1.5.0.20212	Vizrt	27/08/2021	111 MB	1.5.0.20212
Vizrt Viz Tracking Hub 1.5.0.20167	Vizrt	27/08/2021	49,8 MB	1.5.0.20167

Note: Issues have been reported with failing redistributable updates on Windows 10. This can lead to Tracking Hub not starting properly. In case this happens:

- Ensure there is no Microsoft update in queue or in progress (if you are not sure, reboot Windows).
- Manually uninstall any MS Visual C++ 2015-2022 Redistributables (both x86 and x64) and restart.
- Manually install the MS Visual C++ V14 Redistributable (both x86 and x64) and restart.
- Install *VizVirtualStudioBundle-x64-1.9.0*.

Note: Issues have been reported with failing redistributable updates on Windows 10 and Windows 11 related to Microsoft Visual C++ v14 Redistributable 14.50.35719. The Bundle installer will print an error after installation if this redistributable is already installed on the PC. Options to prevent this error:

- Uncheck redist option in bundle if it is already installed.
- Uninstall Microsoft Visual C++ v14 Redistributable 14.50.35719 before starting the bundle installer.

4.3.5 To Repair or Remove Tracking Hub

Once the Bundle has been installed, you can run the Tracking Hub Bundle setup application again to enter **Repair** or **Uninstall** modes. Select **Repair** or **Uninstall** as required:

- Uninstalling the bundle removes the applications it installed: Tracking Hub, Studio Manager, Viz AI Core, Viz AI Core Plugin and Viz AI Lens Calibration Plugin. The prerequisites — the Microsoft Visual C++ redistributables and the CodeMeter Runtime Kit — are intended to remain on the PC. You can remove them separately, if desired.
- After Repairing, you must restart the computer. After restarting, you can proceed to Start Viz Virtual Studio.

4.3.6 Unattended Installation of Viz Virtual Studio Applications

Installation Parameters

Parameter Name	Default Value	Available Values
Tracking Hub MSI		
INSTALLFOLDER	*%\Program Files%\Vizrt\VizTH*	

Studio Manager MSI		
INSTALLFOLDER	*%\Program Files%\Vizrt\StudioManager*	
Virtual Studio Bundle		
INSTALLFOLDER	*%\Program Files%\Vizrt\VizTH* *%\Program Files%\Vizrt\StudioManager*	

Example

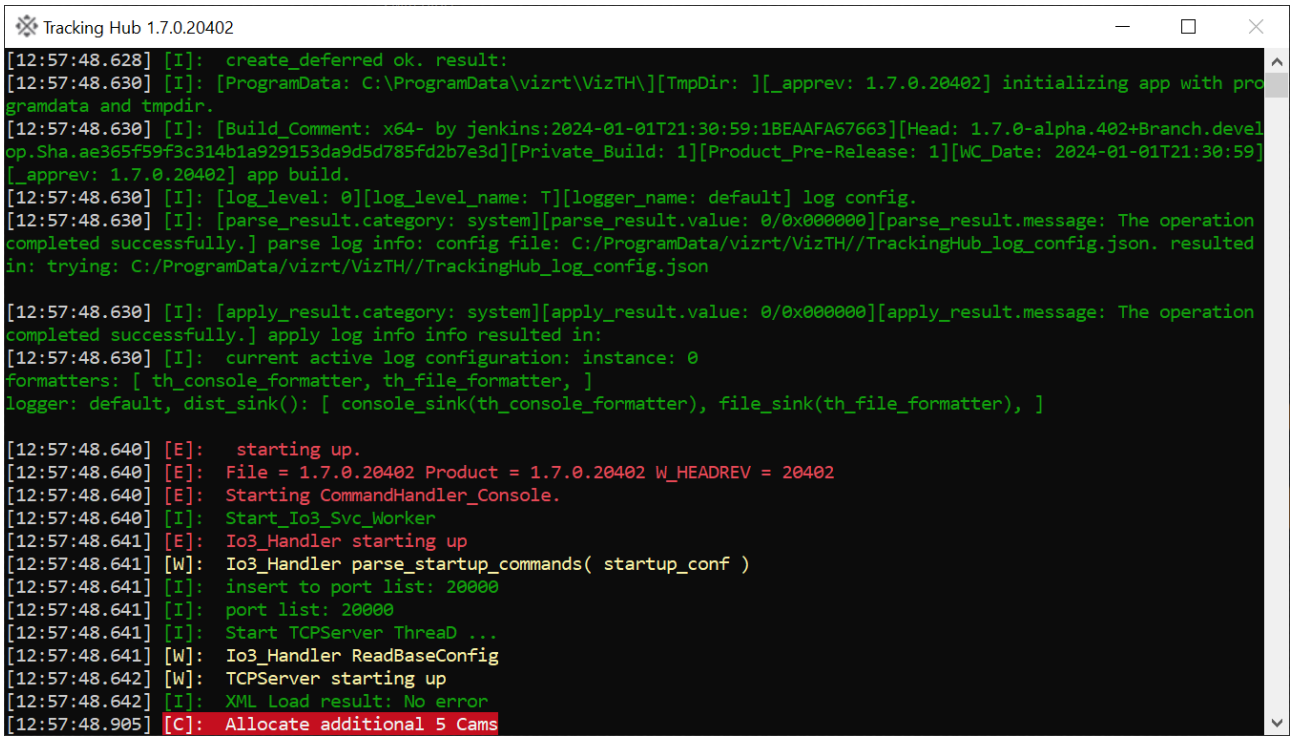
Description	Command
Install Tracking Hub in unattended mode with only progress bar using the *msi installer* and installation path is *C:\TH*	`msiexec /i TrackingHub.msi /passive INSTALLFOLDER="C:\TH"`
Install Studio Manager silently with no UI using the *msi installer* and installation path is *C:\SM*	`msiexec /i StudioManager.msi /qn INSTALLFOLDER="C:\SM"`
Install Tracking Hub in C:\TH and Studio Manager in C:\SM and all necessary MS Visual C++ redistributables and CodeMeter runtime kit using Virtual Studio Bundle	`VizVirtualStudioBundle-x64.exe --silent --trackinghub=INSTALLFOLDER='C:\TH' --studiomanager=INSTALLFOLDER='C:\SM'`

Copyright © 2026 Vizrt

4.4 Start Viz Virtual Studio

- Manual Configuration of the Network Adapter IP Addresses
- Start the Studio Manager
- Activating Automatic Logon
- License Configuration
- Closing Viz Tracking Hub

First start Tracking Hub, then start Studio Manager. Tracking Hub starts as a console program:

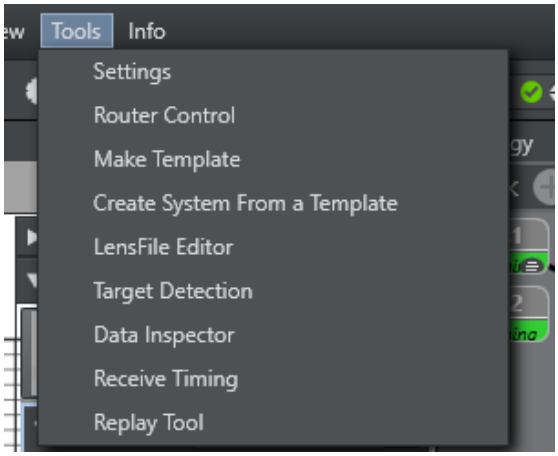


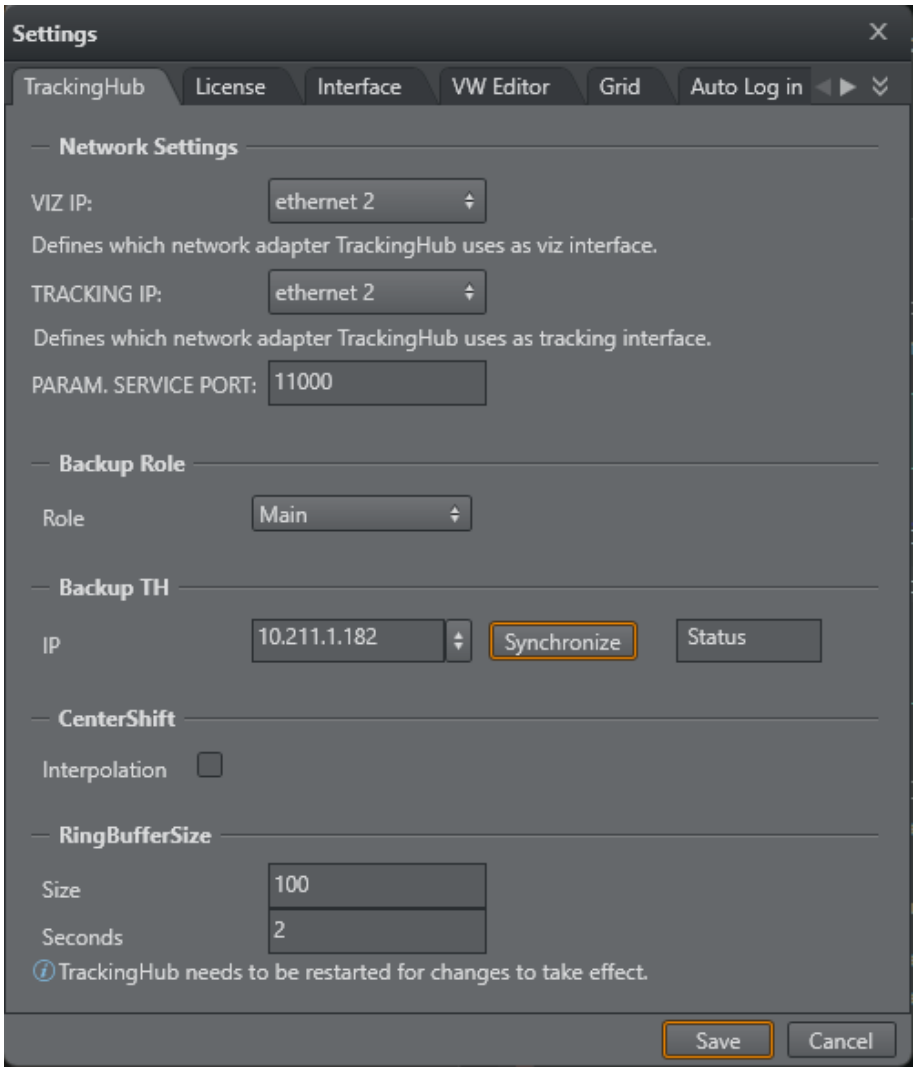
Tracking Hub keeps its configuration file `VizTH.conf` at `C:\ProgramData\vizrt\VizTH`. If the file is missing, Tracking Hub writes a new one with default settings the next time it starts.

Usually, the tracking network is separated from the Engine network. This makes sense when Tracking Systems need their own IP range, and when the amount of tracking data should not interfere with regular network traffic. Tracking Hub can be set to a Viz adapter, which communicates with the Engines, and a tracking adapter which communicates with the tracking system. On the first startup of the Tracking Hub, the local loop back adapter is selected. Changing the network settings is mandatory after the first installation on a new machine.

To change the adapters used by Tracking Hub, select **Settings** from the **Tools** menu in Studio Manager.

Copyright © 2026 Vizrt





Copyright © 2026 Vizrt

IMPORTANT! If changing the network adapter, Tracking Hub needs to be restarted for any changes to take effect.

4.4.1 Manual Configuration of the Network Adapter IP Addresses

1. Start the Tracking Hub and then immediately close it.
2. Go to the configuration folder (the default path is *C:\ProgramData\vizrt\VizTH\Cfg*) and edit the file *BaseConfig.xml*. *BaseConfig.xml* is a XML-formatted text-file. When editing the file, be sure to use a text editor such as Notepad(++) or similar, and save it as a text-formatted file.
3. In the fields `VizAdapter="xxx" TrackingAdapter="xxx"`, specify the adapter names or IP addresses.

Tip: You can view the IP-addresses in use by the server by opening a command prompt and entering the command `ipconfig`.

4. **VizAdapter** and **TrackingAdapter** can have the same setting.

This needs to be done only once. After the command is executed, *BaseConfig.xml* is updated.

IMPORTANT! You can minimize the Tracking Hub console program, but it should be kept running at all time.

4.4.2 Start the Studio Manager

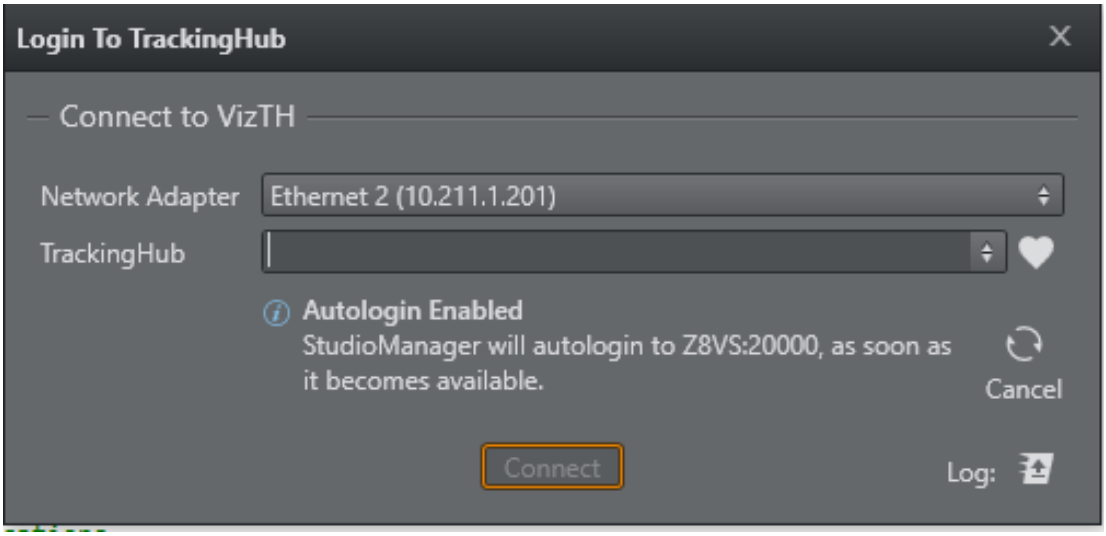
1. Start the Studio Manager:
2. In the Tracking Hub login window enter the required details:
   - **Network Adapter**: Select a network adapter from the Network Adapter List.
   - **Tracking Hub**: Select a Host from the drop-down list. Under certain conditions, such as the server being located on a different network sub-net, the host does not appear in the list. The host name or IP address can then be entered manually.

Copyright © 2026 Vizrt



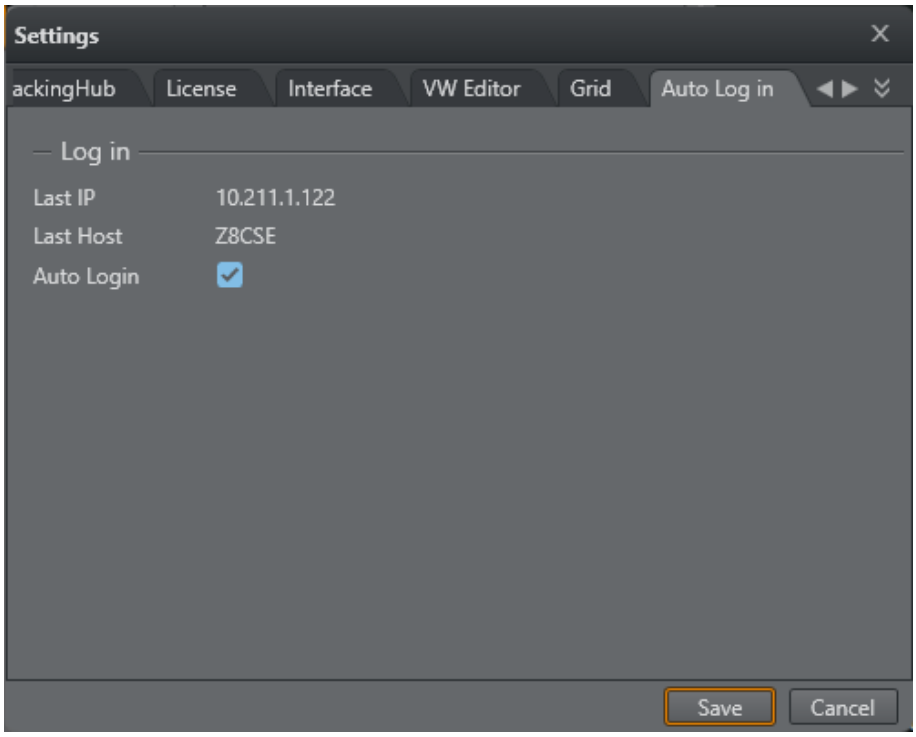
- The dialog remembers the last Adapter which was selected. If auto logon is enabled, the Studio Manager connects to the last selected Tracking Hub.

Copyright © 2026 Vizrt



3. Click **Connect**.

4.4.3 Activating Automatic Logon



Automatic logon can be a activated and deactivated in the **Auto Log in** Tab of the Settings Dialog.

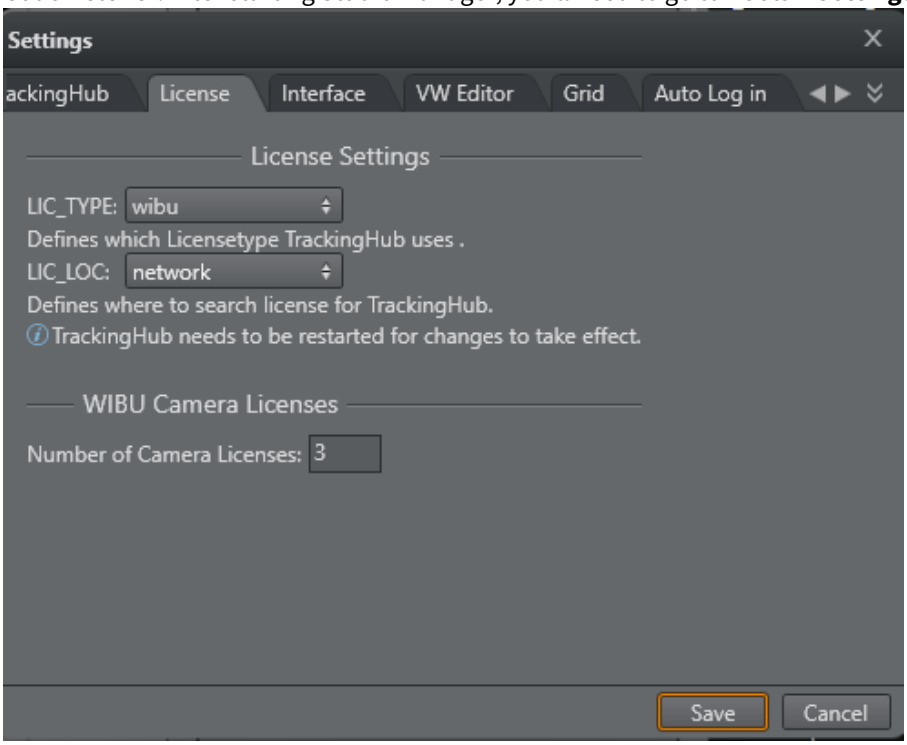
Copyright © 2026 Vizrt

4.4.4 License Configuration

Tracking Hub requires a valid license to work with. This license can be allocated via the WIBU system.

For WIBU licenses, you have different options: You can either use a dongle based license or a network license.

- If a network server is going to be used, make sure the server is listed in the server search list of the Codemeter Web Interface. This can be accessed by browsing to http://localhost:22352/configuration/server_search_list.html. Add the server hosting the license into the server search list.
- A dongle can be either a physical dongle one or a software based one. Both of them need to show up in the Codemeter UI. After starting Studio Manager, you'll need to go to **Tools > Settings and License**.



- **LIC_Type:** Selects what kind of license system you are going to use.
  - **wibu:** For the WIBU Codemeter based licensing system.
- **LIC_LOC:** Defines where Viz Virtual Studio acquires a license. Possibilities are:
  - **local:** Only to be used for WIBU dongle or software containers.
  - **network:** If a network server is hosting your licenses.
  - **local + network:** Either a local or a network license is used, whichever the licensing system finds. This is the default setting.

> ⚠ **Note:** The Sentinel hardlock option has been removed with version 1.7.0. If you still use a Sentinel dongle, please contact Customer Success.

After the license has been set correctly, please close Studio Manager, restart Tracking Hub and restart Studio Manger to check if the license got acquired correctly.

Details can be found in chapter WIBU-based Licensing System and WIBU License System in Tracking Hub.

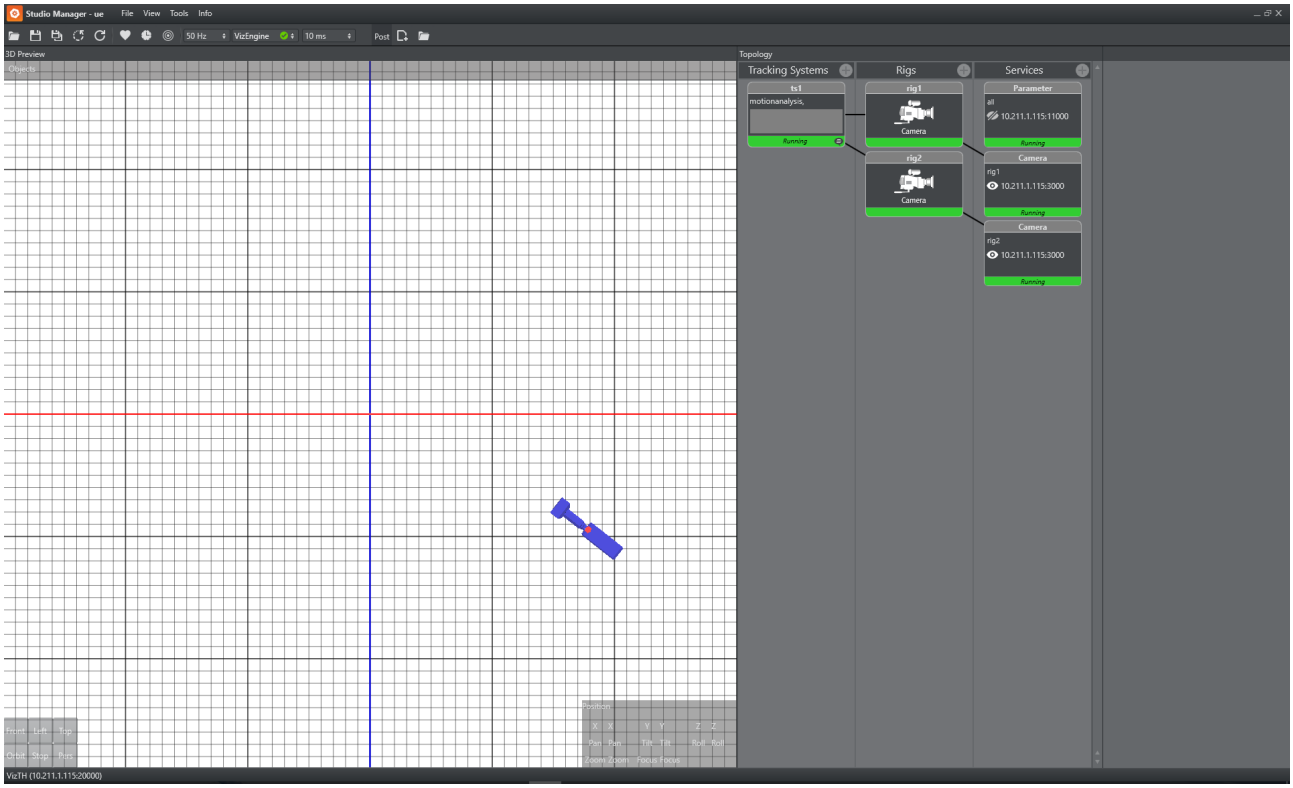
Copyright © 2026 Vizrt

4.4.5 Closing Viz Tracking Hub

Note: To close Tracking Hub, type `exit` or `quit` into the console and press ENTER.

Copyright © 2026 Vizrt

5 Studio Manager User Interface



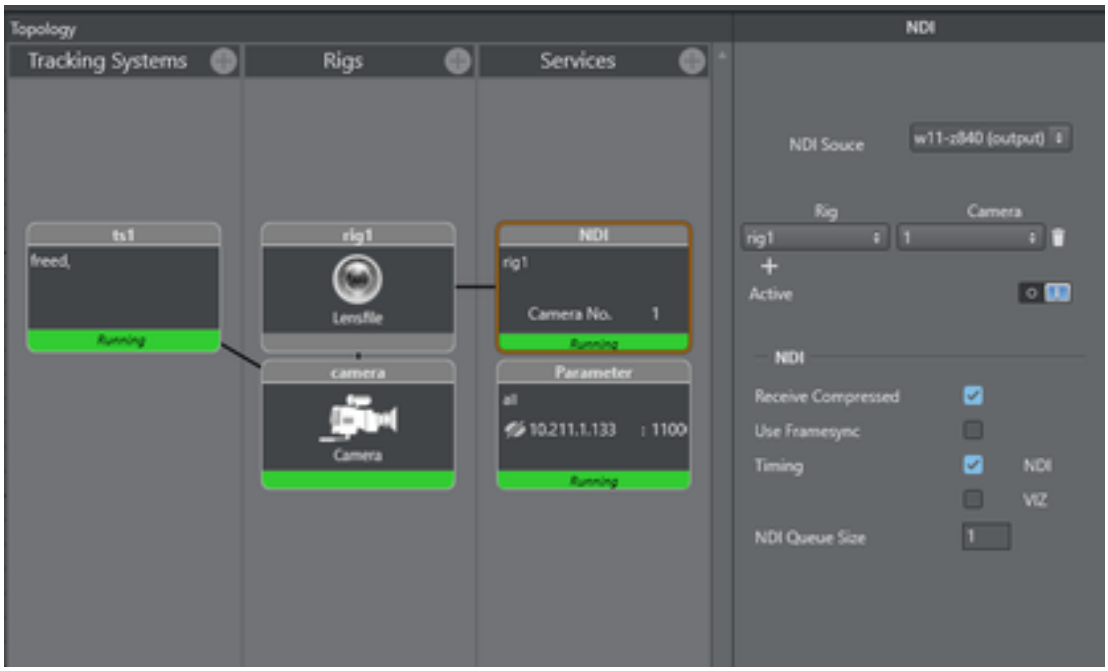
This section details each of the Studio Manager panels and their properties:

- Parameter Panel
- Topology Panel
- Log Panel
- Timing Analysis Window
- Preview Panel
- Post System
- Data Inspector
- Configuration Panel
- Target Detection

Copyright © 2026 Vizrt

5.1 Parameter Panel

The Parameter Panel shows the configuration parameters for the selected Topology item: Tracking Systems, Rigs and Services.

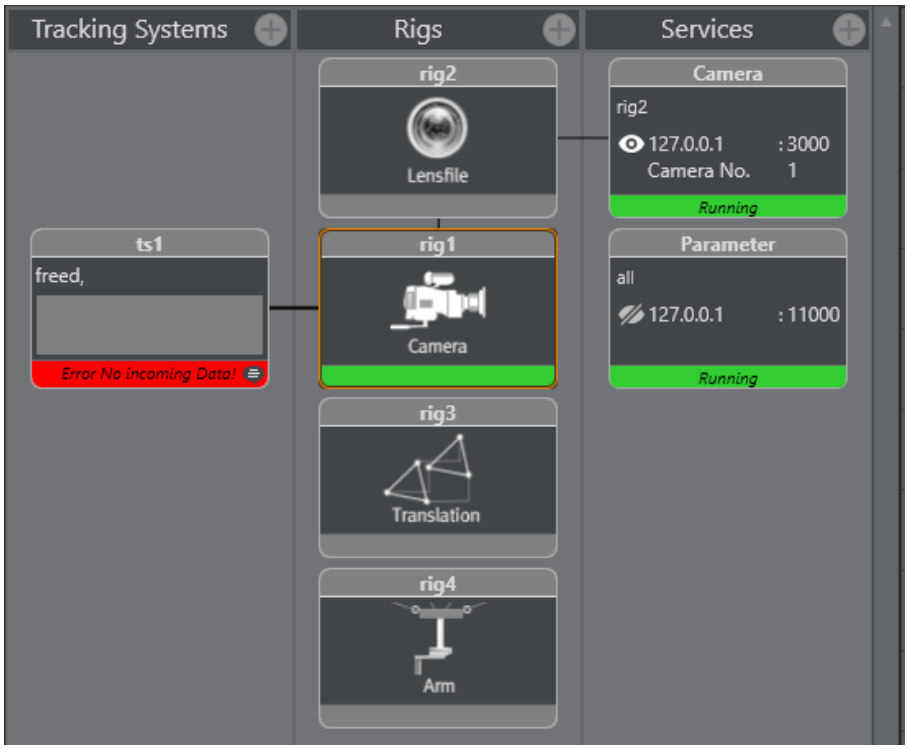


Click on a Tracking System, a Rig, or a Service, to view or change its parameters as required.

5.2 Topology Panel

- Tracking Systems
- Rigs
- Services
- Topology Connection Lines
- Topology Colors
- Motion Capturing with Motion Analysis
- Multi Object Tracking

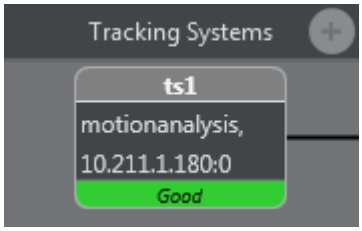
The Topology Panel shows a representation of the tracking systems, the connection to a Viz Engine and which Viz Engine services are in use.



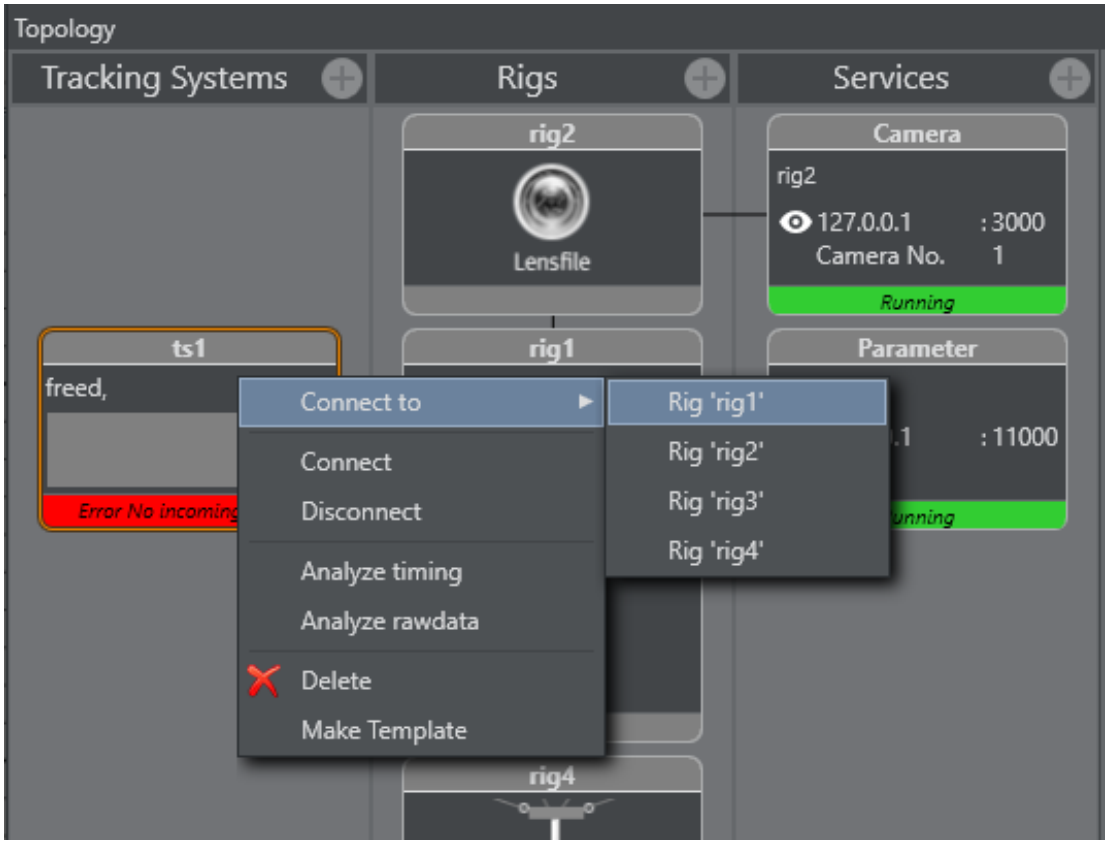
5.2.1 Tracking Systems

The Tracking System panel represents the tracking software in the studio. Click on a tracking system icon to view its properties (see Parameter Panel). A line from the tracking system represents a connection to a Rig.

Copyright © 2026 Vizrt



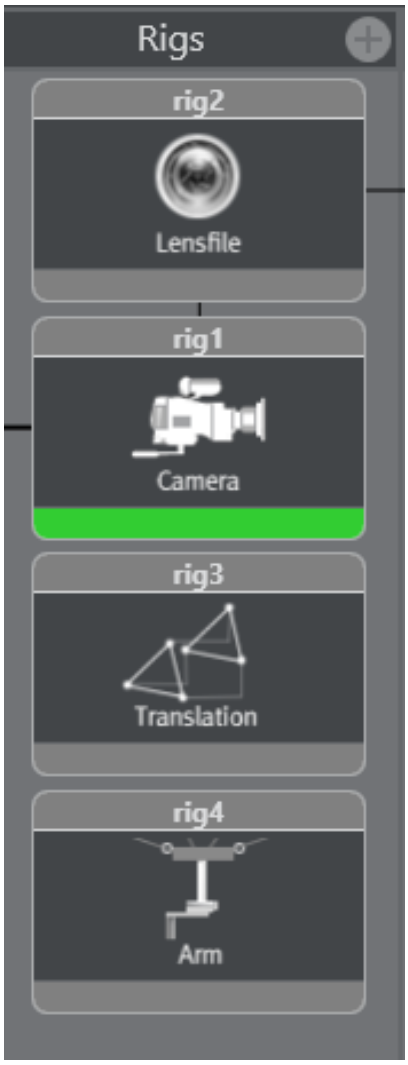
Right-click Menu:



- **Connect to:** Connects to a Rig.
- **Connect/Disconnect:** Connects to or disconnects from the tracking system.
- **Analyze timing:** Shows time analysis.
- **Analyze rawdata:** Enables or disables the collection of raw data for analytical purposes.
- **Delete:** Removes the Tracking System.

Copyright © 2026 Vizrt

5.2.2 Rigs



A Rig is the interface between the Tracking System and Viz Engine. Click on any Rig icon to view its properties (see Parameter Panel). A line from a Rig on the left represents a connection to a tracking system (see Tracking Systems). A line from a Rig on the right represents a connection to a Service (see Services).

Rigs are defined as the following types:

- Camera
- CameraFD
- CameraXZY
- CameraYXZ
- GimbalCamera
- Extended Lens Parameters
- LensFile
- LensFile-FocalDistance

Copyright © 2026 Vizrt

- Pivot Rotation
- Location Pivot Rotation
- Location Pivot Triple Rotation
- Arm
- CraneArm
- CraneHead
- Translation
- Object
- MoCap

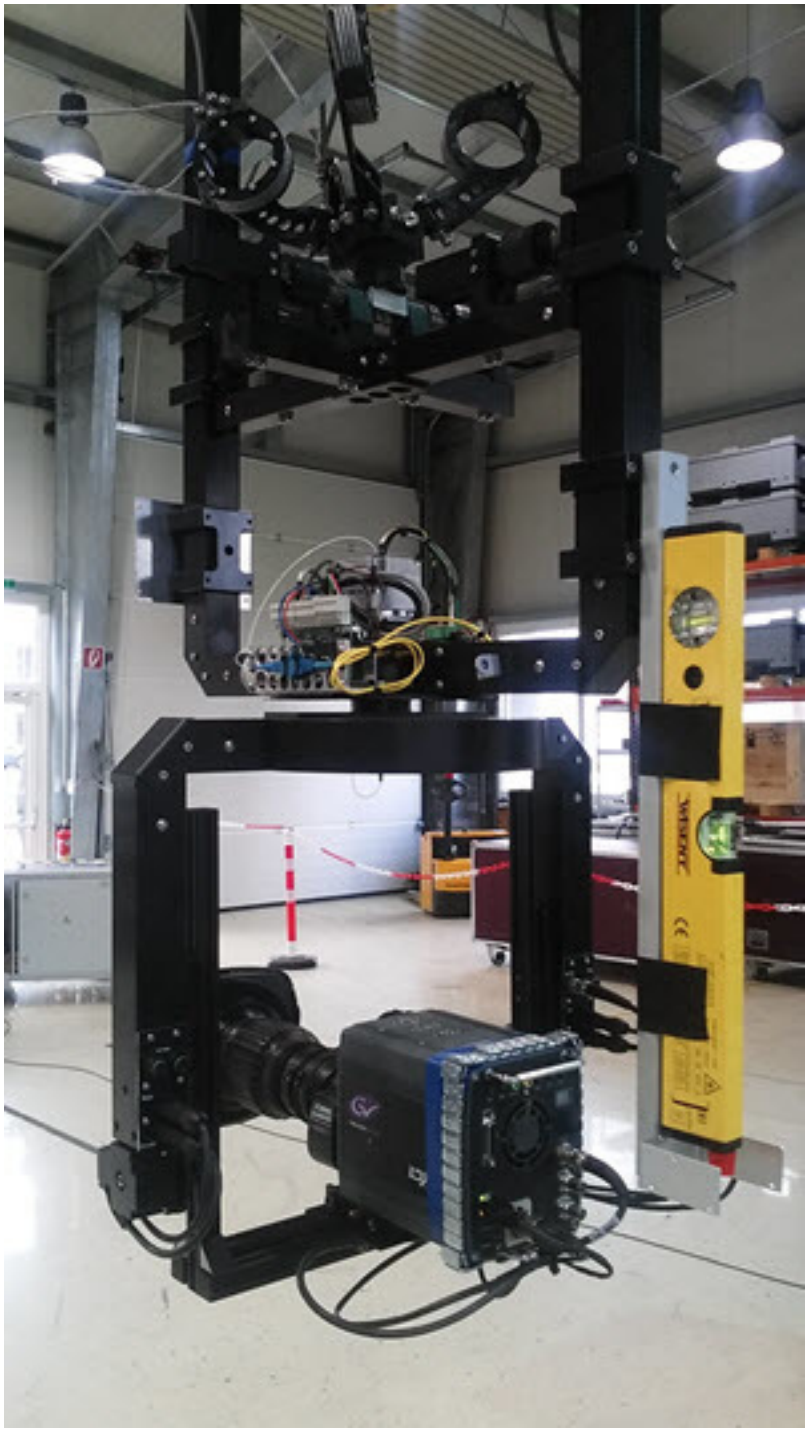
The four camera rigs hold the same parameters and differ in the order in which Pan, Tilt and Roll are applied. CameraFD additionally couples rotation to position, for the Spidercam fielddolly. For the order of each rig, and what it means for the mechanical setup of the head, see Rotation Order in Tracking Hub Structural Design.

The Translation rig is for position data only, and provides the Tracking Hub with the following parameters:

- PosX
- PosY
- PosZ

The Arm rig can be rotated in three dimensions, and starts with a swivel. It has a defined length with a defined direction, and offsets between swivel and length. Both the Translation and Arm rigs are used for camera systems such as Spidercam:

Copyright © 2026 Vizrt



The Location Pivot Rotation rig rotates around the Y axis, which moves points in the X-Z plane. Its pivot point is not fixed: it is taken from the child camera's position offset. Pivot Rotation is a rotation rig with a freely definable center point, and provides the Tracking Hub with the following coordinates:

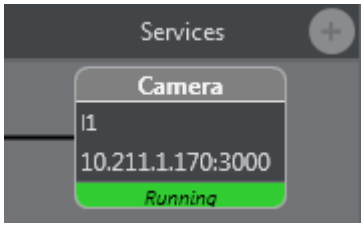
- PivotX
- PivotY
- PivotZ

Copyright © 2026 Vizrt

Camera is a basic rig for tracked cameras. Object is a basic object for tracked objects.

5.2.3 Services

A Service is the element of a Viz Engine which is connected to the Tracking System through the Rig. Click on a Service icon to view its properties (see Parameter Panel). A line from the a Service represents a connection to a Rig (see Rigs).



Available Services are:

- **Camera:** Tracks a camera for Viz Engine.
- **Object:** Tracks an object with Viz Engine.
- **Parameter:** Provides all required data for the creation of the 3D View.
- **MoCap:** Captures motion for Viz Engine.
- **Communication Timing:** Sends timing information which are measured during tracking data acquisition.
- **Tracking Timing:** Sends timing information which are measured during sending the data as service to Viz Engines.
- **Timecode Service:** Used for post and replay.
- **Multi Object Service**: For the TrackMen object tracker. **See Multi Object Tracking.**
- **Tracking Flow and Tracking Flow Raw Service:** Used to analyze incoming tracking data.
- **NDI Service:** Used for embedding tracking data into NDI stream.
- **TC Camera:** Used by the Viz Libero System.
- **Target Service:** Sends target data to Viz Engine. See Target Detection.
- **Video Wall Service:** Sends tracking data for a video wall. It is handled like a camera service.

Right-click display object for additional menu:

Copyright © 2026 Vizrt



- **Analyze timing:** See Analyze Time.

5.2.4 Topology Connection Lines

The Topology Connection Lines show the connections between each item in the Topology. Double-clicking a Topology connection line between a Tracking System and a Rig opens the Tracking parameter routing window for the connection:



5.2.5 Topology Colors

The colors of the icons in the Topology view show the working condition:

- **Green:** Everything is OK and in regular working condition.
- **Grey:** The tracker is not connected or deactivated.
- **Orange:** The Service or Tracker state is in warning condition. See the log or message for more information.
- **Red:** The Service or Tracker state has a serious error. See log or message for more information.

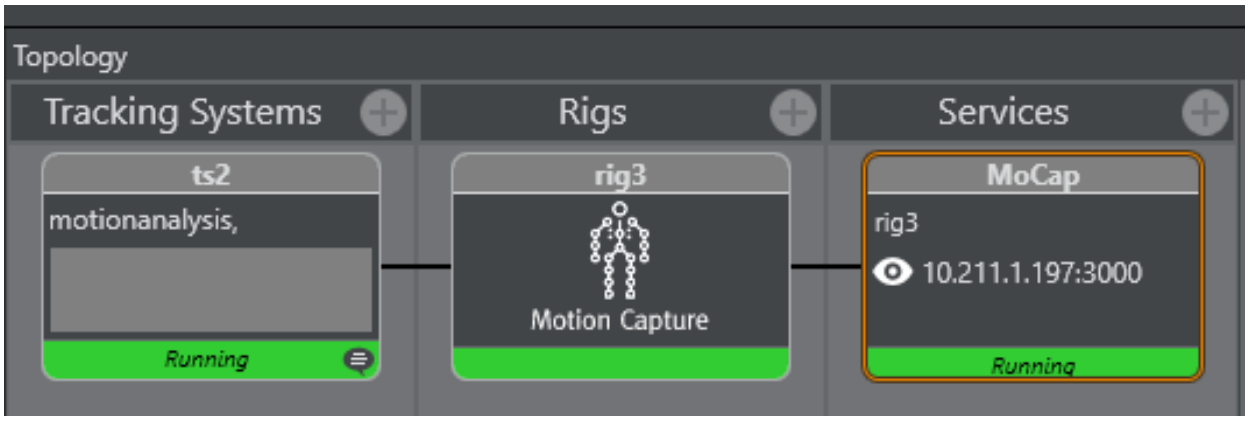
5.2.6 Motion Capturing with Motion Analysis

Motion Capturing is a special way of collecting and forwarding data to an Engine. This mode transmits movements of a person or people instead of camera positions. For these special procedures, there is a new rig and a new service available.

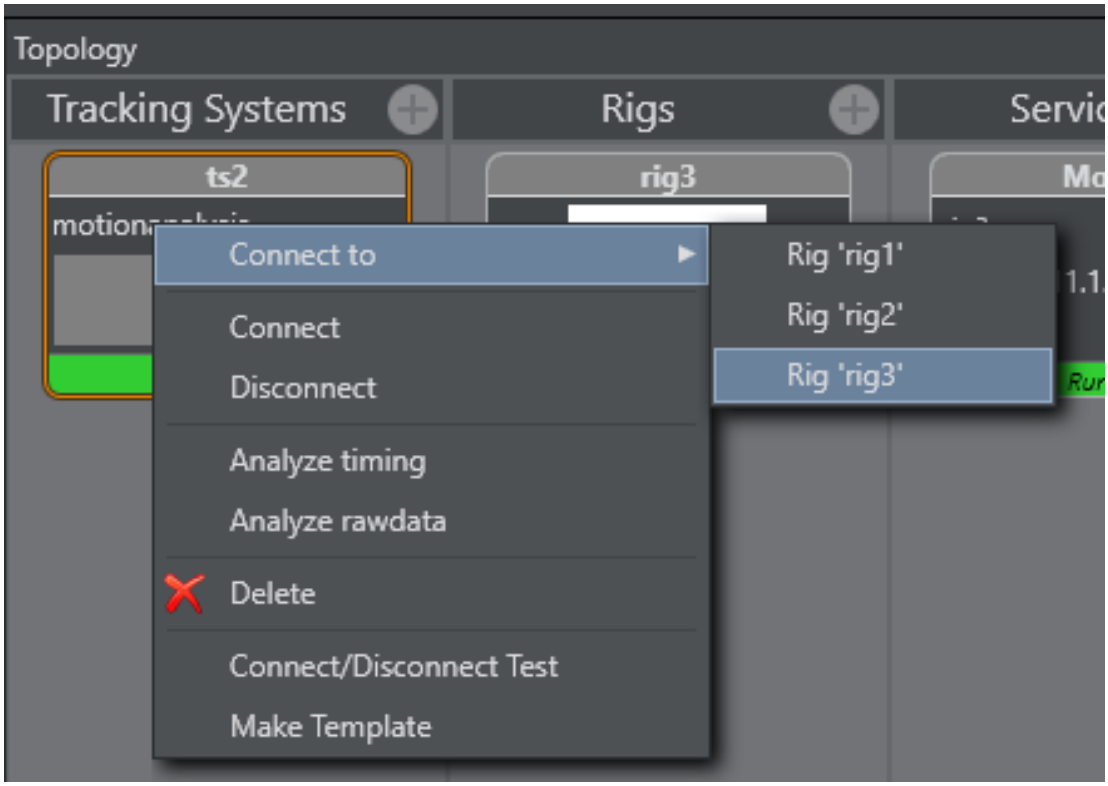
Copyright © 2026 Vizrt

Selectable via the menu:

- Rigs > Add MoCap
- Services > Add MoCap service

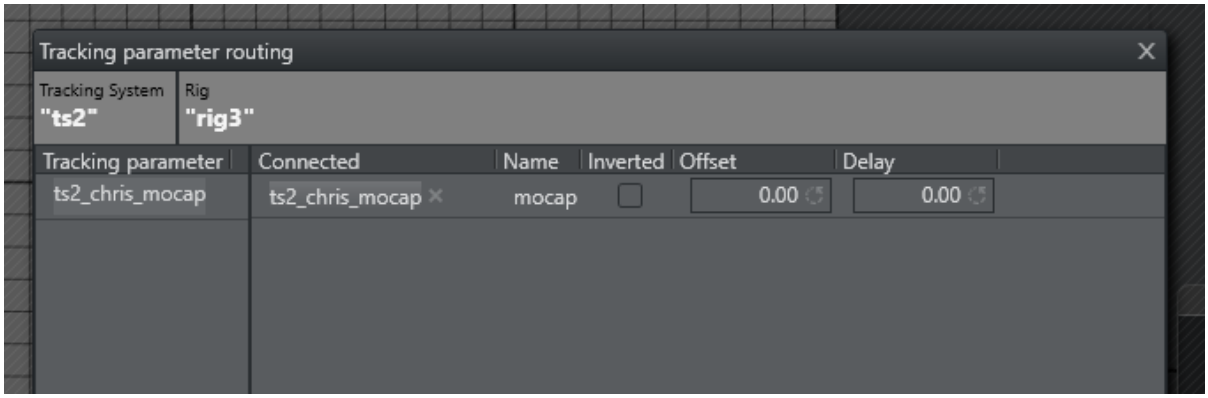


Connect the Motion Analysis Tracking System to the MoCap Rig as usual.



In this case, there is just one axis available.

Copyright © 2026 Vizrt



The name of the axis depends on the used name in MA's Cortex. Mocap Service can be used as other services.

5.2.7 Multi Object Tracking

Multi object tracking forwards a whole group of tracked rigid objects to Viz Engine in one stream, instead of one object per service. It is used with the TrackMen object tracker, which is selected on the tracking system as the `trackmen-objtr` protocol.

The tracking system sends one packet per group. A packet contains:

- A cluster number, which identifies the group.
- A list of objects, each with an id, a position and a rotation.
- A timecode.
- The axis assignment for pan and tilt, and a length exponent that defines the unit of the position values — for example -2 for centimetres.

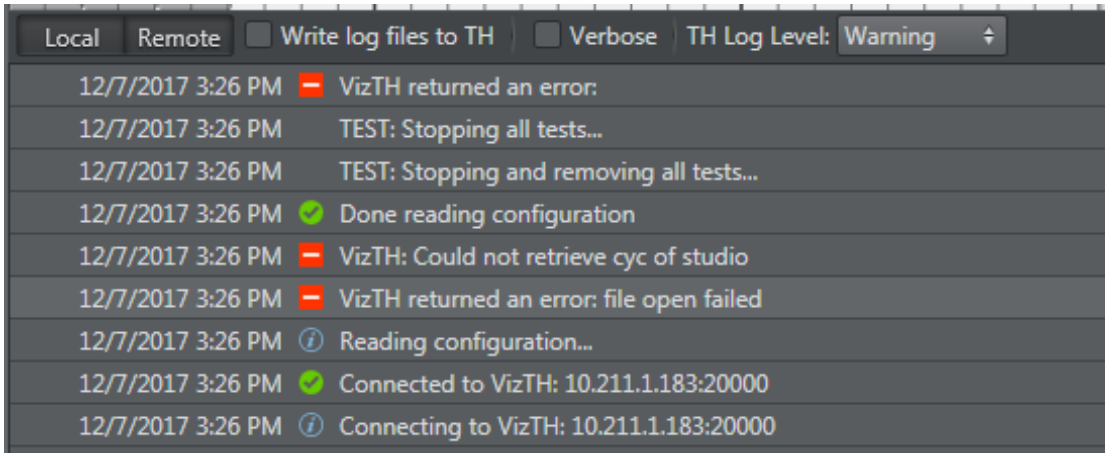
Each cluster becomes one named entity in the parameter pool, called `tmot_<cluster number>`. The objects of a cluster travel through the same channel as motion capture data, so the group arrives as one set of transforms rather than as separate object parameters. This is why the protocol appears as Trackmen Mocap under Supported Protocols.

Every object carries a visibility flag. If a cluster stops sending for more than 60 milliseconds, Tracking Hub marks it as missing and publishes the cluster with all of its objects set to invisible. The objects therefore disappear in Viz Engine instead of freezing at their last known position.

Copyright © 2026 Vizrt

5.3 Log Panel

The Log panel provides messages and notifications from the Tracking Hub. The bottom line displays information about which Tracking Hub the Studio Manager is connected to. You can open and close the Log panel by clicking **View Log** in the Configuration Panel.



You can copy each entry in the Log Panel to the clipboard by right-clicking the entry and selecting **Export message to clipboard**. The information can be copied to the clipboard as plain text or as XML encoded text.

- **Export Log to a file...:** Exports the log file as XML or as plain text.
- **Delete Log Entries:** Clears all log messages from the window.
- **Write Log File on VizTH:** Makes the Tracking Hub write a time-stamped log file for all tracking data received and sent to Viz Engine when selected. Additional timing information is logged as well. The log files are:
  - `VizTH_rcv<n>.log`
  - `VizTH_send<n>.log`
  - `VizTH_timing<n>.log`

  They are located in `C:\ProgramData\vizrt\VizTH`. When either the received or the sent log reaches 1 MB, all three files are closed and a new set is started with an incremented number.
- **:** Changes the view of the debug output between the Studio Manager or the connected Tracking Hub.
- **Verbose:** Enables verbose logging when the check-box is selected.
- **TH Log level:** Use the drop-down menu on the right side to select the log level:
  - **Fatal:** Displays only fatal errors (lowest log level).
  - **Error:** Displays Fatal and Error messages.
  - **Warning:** Displays Fatal, Error and Warning messages.
  - **Information:** Displays Fatal, Error, Warning and Information messages.
  - **Debug:** Displays Fatal, Error, Warning, Information and Debug messages.
  - **Verbose:** Displays all messages (highest log level).

  The log level decides which messages Tracking Hub generates. Where those messages end up is configured separately, in `Trackinghub_log_config.json`, and the two limits do not match:
  - The console shows **Information** and above. With the delivered configuration, Debug and Verbose messages never appear in the console.

Copyright © 2026 Vizrt

- The log file records **Debug** and above, so Verbose messages are not written either.

To see Debug or Verbose output, lower the corresponding sink level in `Trackinghub_log_config.json`. Tracking Hub prints a reminder in the console whenever a level below Information is selected.

Messages are written by a rotating file sink, configured in `Trackinghub_log_config.json` as `logs/th_<n>_default`, with up to 1024 MB per file and up to 10 files.

Copyright © 2026 Vizrt

5.4 Timing Analysis Window

The Timing Analysis Window can be opened by clicking the View Timing Analysis button in the Configuration Panel. The Timing Analysis Panel contains two tabs: **Tracking** and **Engine**.

- **:** Starts or pauses the timing analysis.
- **:** Refreshes the data.

The **Tracking** tab analyzes tracking systems and camera services. In the top half of the panel, timing events are displayed in form of one timing track per tracking system and camera service. The timing events between the latest two sync signals are displayed (i.e. the length of the timing tracks is determined by the studio frequency).

The sync signals are represented by *orange circles*. The next-to-last sync signal rests at the left border of each track, without moving. The latest sync signal may flicker at the right border, and may even be temporarily out of sight, because of timing variations.

- A *blue bar* represents the time needed to receive tracking data.
- A *purple bar* represents the time needed to store the received tracking data.
- A *green bar* represents the time needed for computations on the tracking data to be sent next.
- A *red circle* marks the moment when Viz Tracking Hub has finished sending the tracking data to the connected Viz Engine(s).

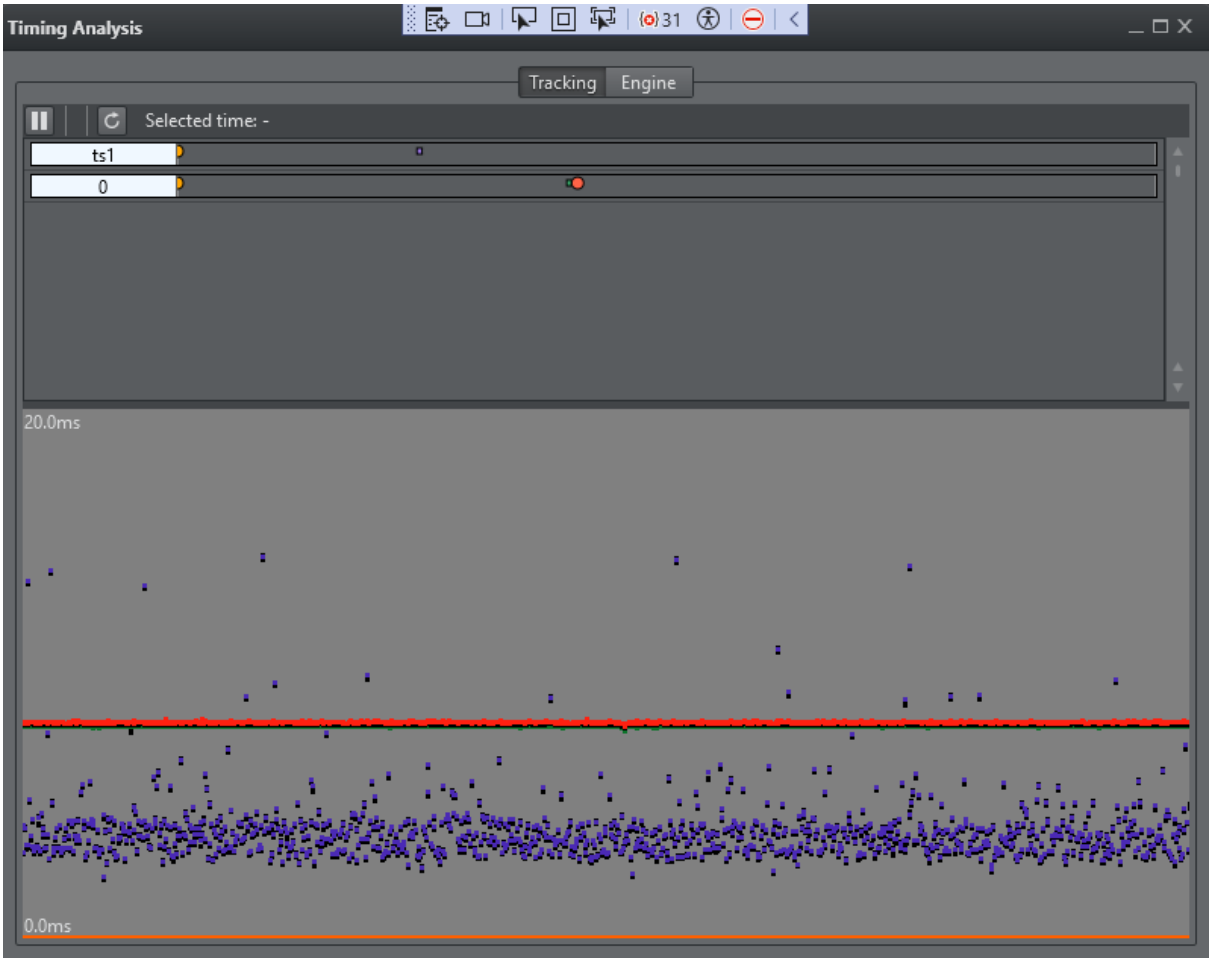
In the lower half of the panel, some of the events are plotted over time. The newest events are displayed at the right border and constantly move towards the left border as time passes. Colors correspond to those in the timing tracks in the top half of the panel.

For a specific tracker, timing analysis can be started right clicking the tracker or selecting **Analyze timing** in the open context menu. A service is created to send the timing information to the Studio Manager. To stop the service, either close the created service or deselect **Analyze timing** in the context menu.

The **Engine** tab analyzes the timing reported by Viz Engine (requires Viz Engine version 3.8 or later) on the selected network.

- The *blue bar* represents the minimum (light blue) and maximum (dark blue) time which passed between two sync signals on the Engine.
- The start of the *red bar* is the point in time when the tracking data arrived at the Engine. The end of the red bars indicates the minimum (light red) and maximum (dark red) time between two incoming tracking packages.

Copyright © 2026 Vizrt



See Also

- Configure the Studio
- Configuration Panel
- Parameter Panel
- Topology Panel
- Preview Panel

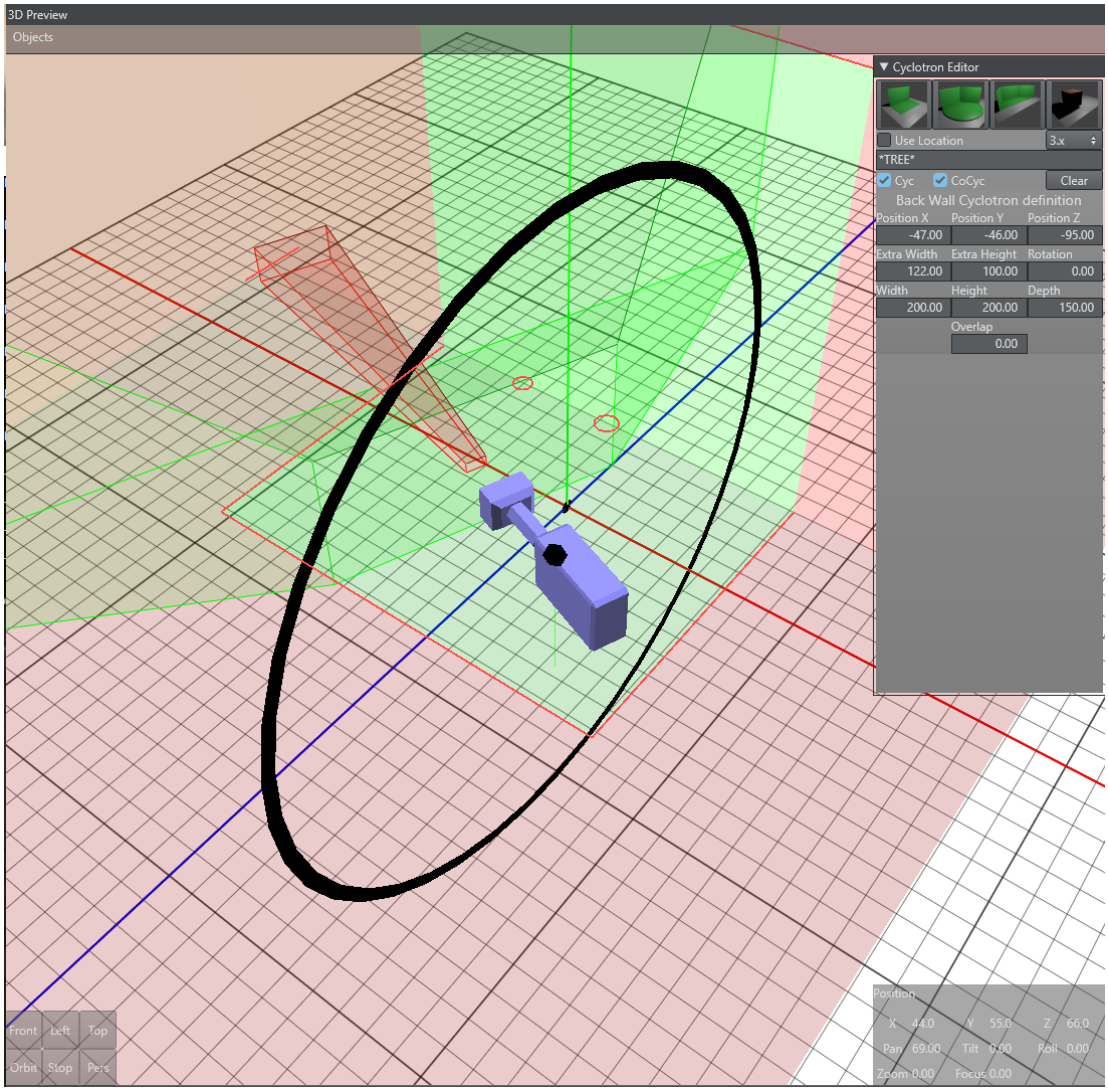
5.5 Preview Panel

This page contains information about the following topics:

- Interface Configuration
- View Menu
- Navigation
- HUD
- Cyclorama Editor
- Videowall Editor
- Cyc
- CoCyc
- Objects Menu
- Selectable Display Modes
- Camera Handles

The 3D Preview panel gives a preview of the tracked cameras, which should be as close to the real studio as possible. When the Studio Manager starts up, a Parameter Service is established, which sends the positions of all tracked rigs to the preview panel.

Copyright © 2026 Vizrt



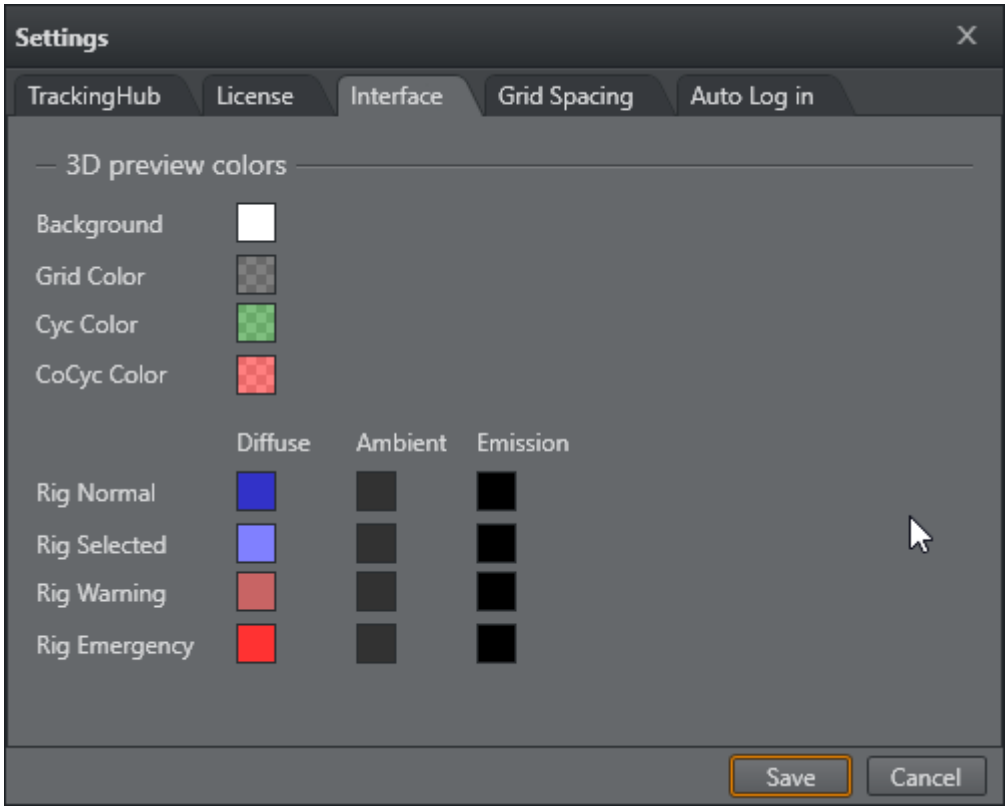
The preview panel consists of the following parts:

- The main menu.
- The navigation bar.
- The HUD display.

5.5.1 Interface Configuration

To adjust the colors to a users need, open the Settings window in Tools. In this window, the user can adjust the render colors for the visible objects in the preview panel. Click in a drop-down list to open the color selection window. The RGB and Alpha values can be adjusted for all colors by clicking the **Advanced** button.

Copyright © 2026 Vizrt

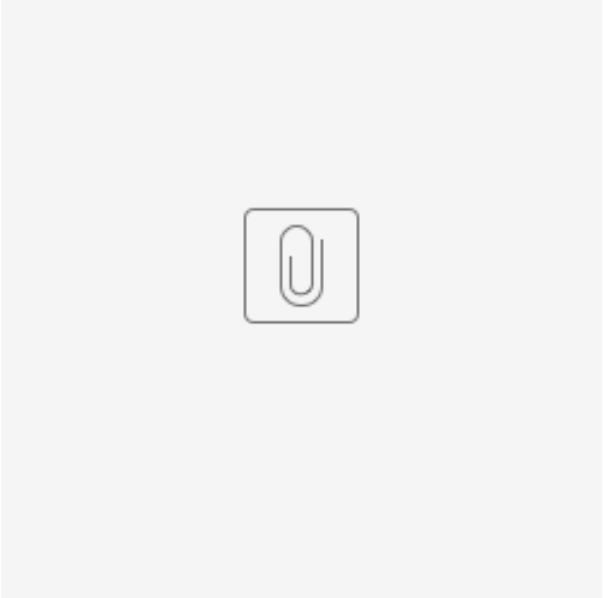


For the Background, Grid, Cyc and CoCyc colors, only diffuse color can be selected. These objects are rendered without lighting.

- **Rig Normal:** Shows colors used when the Rig is not selected. The user can define Diffuse, Ambient and Shininess colors.
- **Rig Selected:** Shows colors used when the Rig is selected in the Rig Editor.
- **Rig Warning:** Shows when a connected Parameter service reports a problem which is handled by the system (interpolation of missing packages).
- **Rig Emergency:** Shows when a Parameter service reports a problem which cannot be handled by the system, and immediate action from the operator is needed.

Copyright © 2026 Vizrt

5.5.2 View Menu



5.5.3 Navigation

Activates or deactivates the navigation bar in the bottom left corner of the 3D Preview window:



- **Front:** Activates an orthographic camera, which looks in Z-Direction.
  - **Right Mouse:** Moves the camera in X and Y.
  - **Mouse Wheel:** Zooms in and out.
  - **Middle Mouse:** Moves the camera in X and Y.
- **Left:** Activates an orthographic camera which looks in X-Direction.
  - **Right Mouse:** Moves the camera in Z and Y.
  - **Mouse Wheel:** Zooms in and out.
  - **Middle Mouse:** Moves the camera in Z and Y.
- **Top:** Activates an orthographic camera which looks in Y-Direction
  - **Right Mouse:** Moves the camera in X and Z.

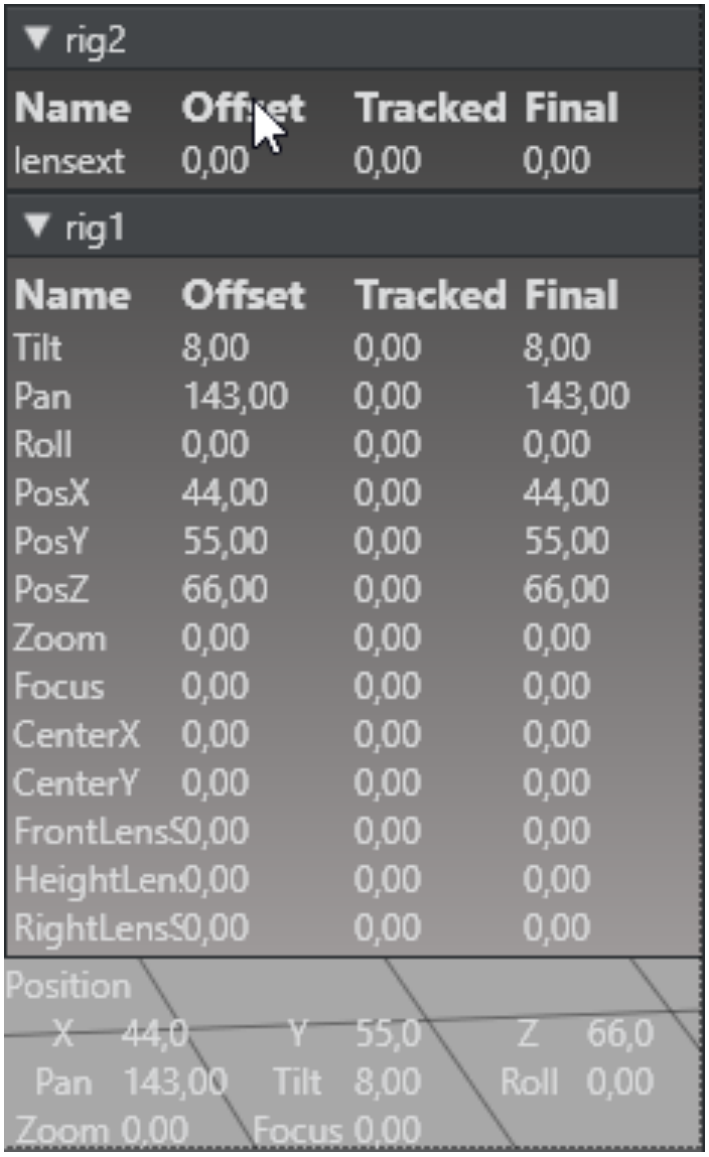
Copyright © 2026 Vizrt

    - **Mouse Wheel:** Zooms in and out.
    - **Middle Mouse:** Moves the camera in X and Z.
- **Orbit:** Activates a perspective camera which is fixed to the selected tracking system. This enables the user to modify handles even when the camera is moving.
    - **Right Mouse:** Rotates the camera around the rig.
    - **Mouse Wheel:** Zooms in and out.
    - **Middle Mouse:** Moves the camera in rig direction or away from the rig.
- **Stop:** Disables the 3D View to improve performance on low level systems.
- **Pers:** Activates a perspective camera which is not bound to a Rig.
    - **Right Mouse:** Pans or tilts the camera.
    - **Mouse Wheel:** Zooms in and out.
    - **Middle Mouse:** Moves the camera in X and Z direction.

5.5.4 HUD

Activates the head up display, which displays the tracked raw coordinates of the selected rig.

Copyright © 2026 Vizrt



5.5.5 Cyclorama Editor

The Cyc Editor is located in the top-right corner of the preview panel and can be expanded by clicking the arrow on its left side. It is used to create the Cyc and CoCyc masks which are needed to mask out the regions of the studio not covered by the green/blue screen. The CoCyc geometry is stored on the Tracking Hub and is distributed from there to all connected Engines.

For the full description of the editor, its Cyc types and masks, see Cyclorama Editor.

5.5.6 Videowall Editor

The Videowall Editor describes the LED walls of the studio: where each wall stands, how big it is, what shape it has, and which Viz Engine renders it. Walls are edited in the same 3D studio view as the cyclorama and the cameras.

Copyright © 2026 Vizrt

For the full description, see Videowall Editor.

5.5.7 Cyc

Defines what parts of the Cyc the preview panel renders. The selectable elements are Wall and Floor. All these settings are stored for local use in the user settings folder.

5.5.8 CoCyc

Defines what parts of the CoCyc the preview panel renders.

5.5.9 Objects Menu

Every rig present in Tracking Hub has its own display menu. These options are handled in the Object menu item. The menu lists the rigs of the current configuration by name, and each entry offers the display modes available for that rig type.

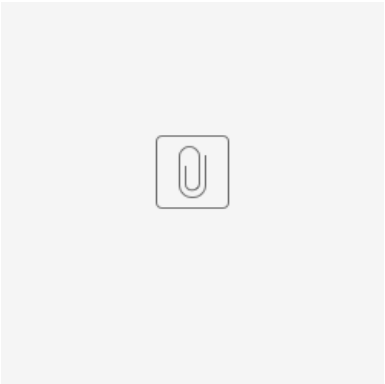


Copyright © 2026 Vizrt

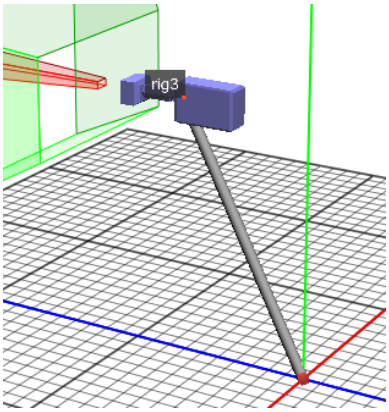
5.5.10 Selectable Display Modes

Which display modes a rig offers depends on its type:

Rig type	Available modes
Camera rigs	Simple, Pedestal, Crane
Arm, CraneArm, CraneHead	Simple, Arm, Fork
Pivot Rotation, Location Pivot Rotation, Location Pivot Triple Rotation, Object	Simple only

Mode	Description	
Simple	The tracked point is displayed as a selectable circle in the view. A rendered camera shows the offset settings.	
Pedestal	The rig is displayed as a tracked pedestal. The tracking point is always on top of the pan/tilt head. The rendered camera shows the offsets added to the rig.	

Copyright © 2026 Vizrt

Mode	Description	
Crane	The rig is displayed as a small jib. The origin of the base is the X/Z offset of the rig. The tracked point is a result of offset and height.	
Arm	The rig is drawn as an arm.	

A rig of type Object is a tracked object which is not a camera, for example a Motion Analysis object. No camera is rendered and only the tracked point is displayed.

5.5.11 Camera Handles

Whenever the user clicks in the rig editor to adjust a specific value, the preview panel displays the values as a handle that can be dragged. Handles can be arrows (position) or disks (rotation). Whenever the user touches a Handle with the mouse, the handle goes from its base color to yellow. This indicates that only this handle is active at the moment. Clicking with the left mouse button and moving the mouse in handle or in the opposite direction modifies the handle's value.

The following list shows all available handles and the corresponding rig values.

Copyright © 2026 Vizrt

Rig Value	Description	Example
Position Offset	Every axis is color coded and only visible if modifiable. • **Green:** Y Axis • **Red:** X Axis • **Blue:** Z Axis	
Pan Offset	**Green:** Modifies the pan offset of the camera.	

Copyright © 2026 Vizrt

Rig Value	Description	Example
Tilt Offset	**Blue:** Modifies the tilt offset of the camera.	
Roll Offset	**Red:** Modifies the roll of the rig.	

Copyright © 2026 Vizrt

Rig Value	Description	Example
Front Lens Shift	**Blue:** Shows the distance from the CCD to the rotation axis of the tracking system.	
Height Lens Shift	**Green:** Displays the distance from the CCD to the tilt axis of the tracking system.	

Copyright © 2026 Vizrt

Rig Value	Description	Example
Right Lens Shift	**Red:** Displays the distance from the CCD to the rotation axis in X direction.	

See Also

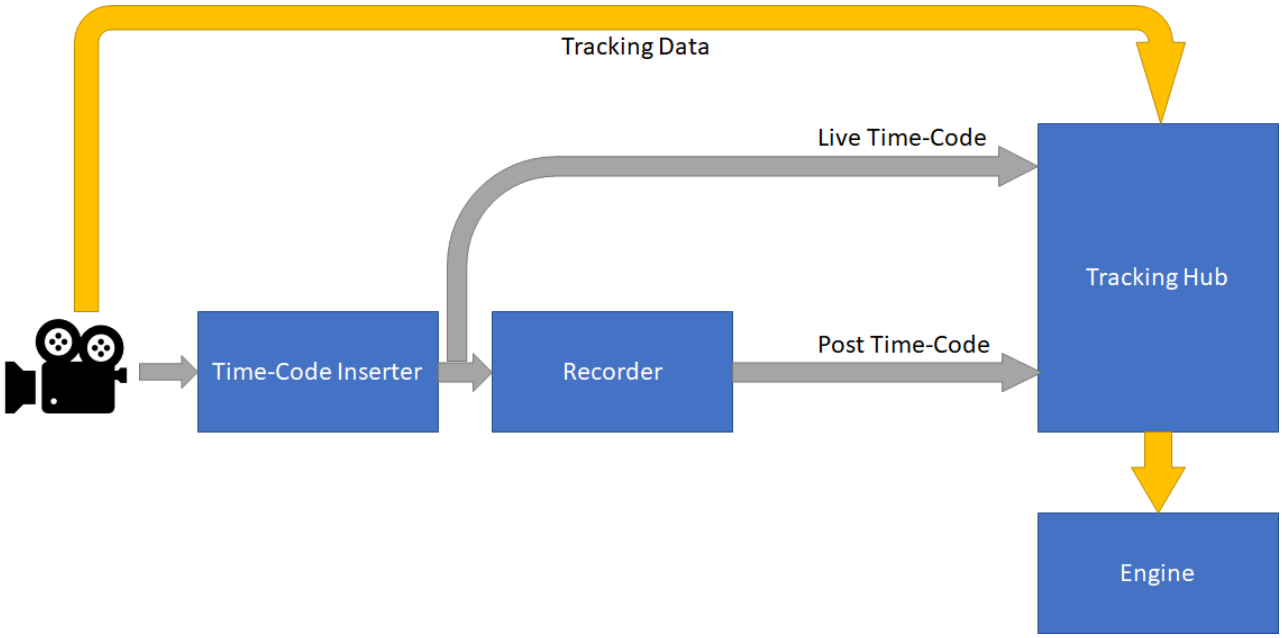
- Configure the Studio
- Configuration Panel
- Parameter Panel
- Topology Panel
- Log Panel
- Cyclorama Editor
- Videowall Editor

Copyright © 2026 Vizrt

5.6 Post System

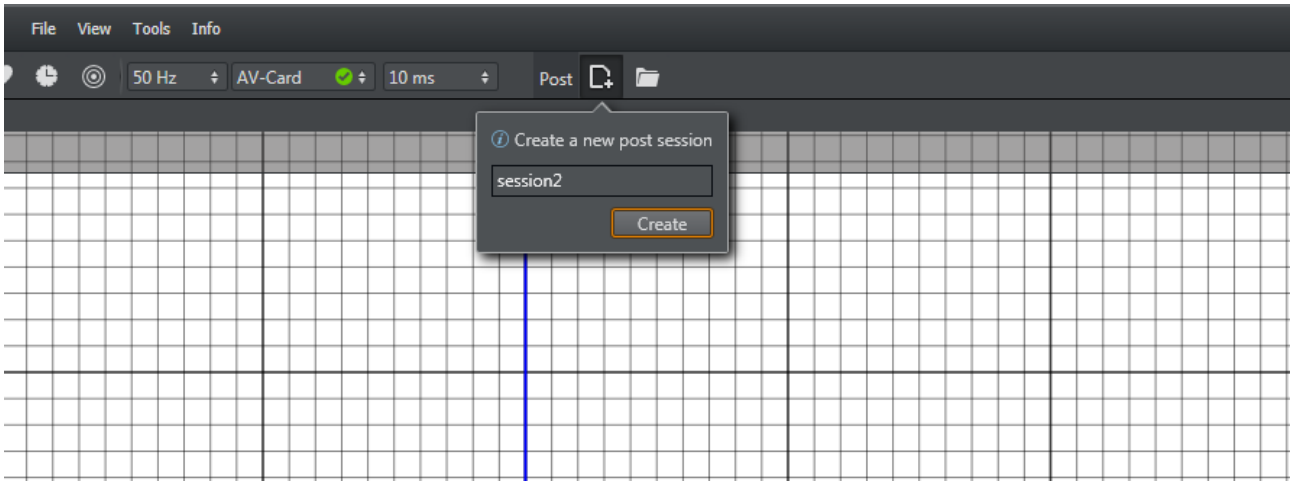
The Tracking Hub Post system lets you record tracking data and play the data back at a later time. To use this feature, every field in the video and every tracking data must be stamped with a timecode. The timecode is read by Plura timecode reader card (PCLPCIe 3G).

> **Note:** While the Live timecode can be LTC based, the Post timecode must be field based (ATC VITC). If a frame based timecode is used for live, the systen increases the field count on every field change. This is not possible for post, as the video can be shuttled back and forward, and therefore the direction is not given for sure.



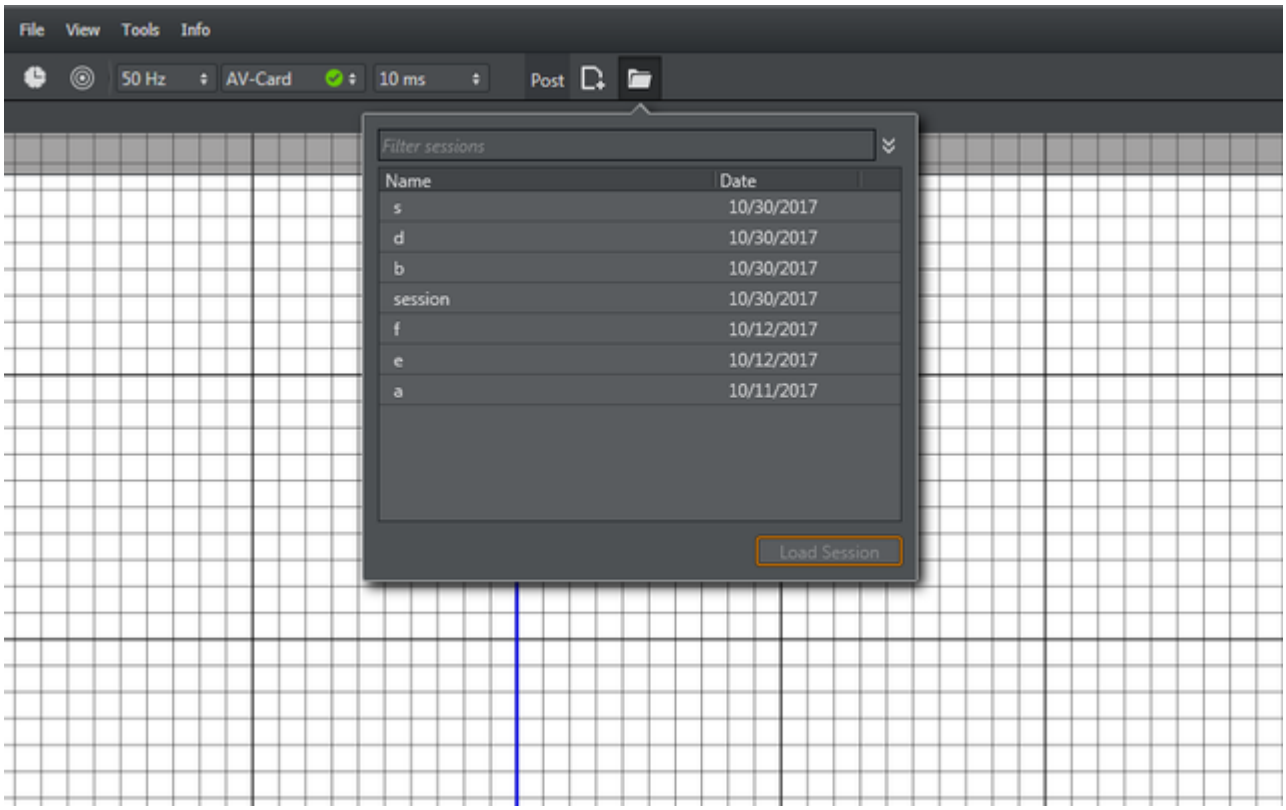
5.6.1 Create a Post Session

Click the **New Session** button to create a new session. Enter the session name in the pane that opens, then click **Create**.



5.6.2 Load a Post Session

To load a previously stored session, click the **Load** button in the **Post** section of the window. A pane with a list of previously recorded sessions appears. Select one of the sessions and click **Load Session**.



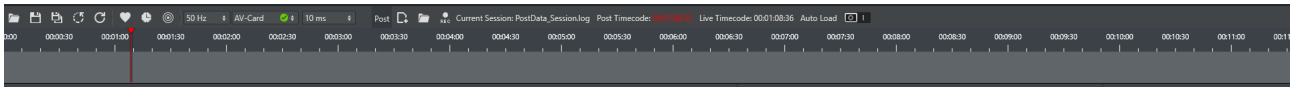
Copyright © 2026 Vizrt

5.6.3 Configure a Post Session

After creating or loading a session, the timeline and timecode labels are visible. A green bar in the time line display indicates time spans where tracking data was recorded and is in memory. The red timecode arrow shows the actual post timecode. The white arrow shows the live timecode. Hold **Ctrl** and scroll the mouse wheel to modify the timeline scale. You can change the position of the timeline using the scroll bar at the bottom.

Hover the mouse over the **Post Timecode** or **Live Timecode** labels to display a tooltip that shows the Post timecode source or the Live timecode source.

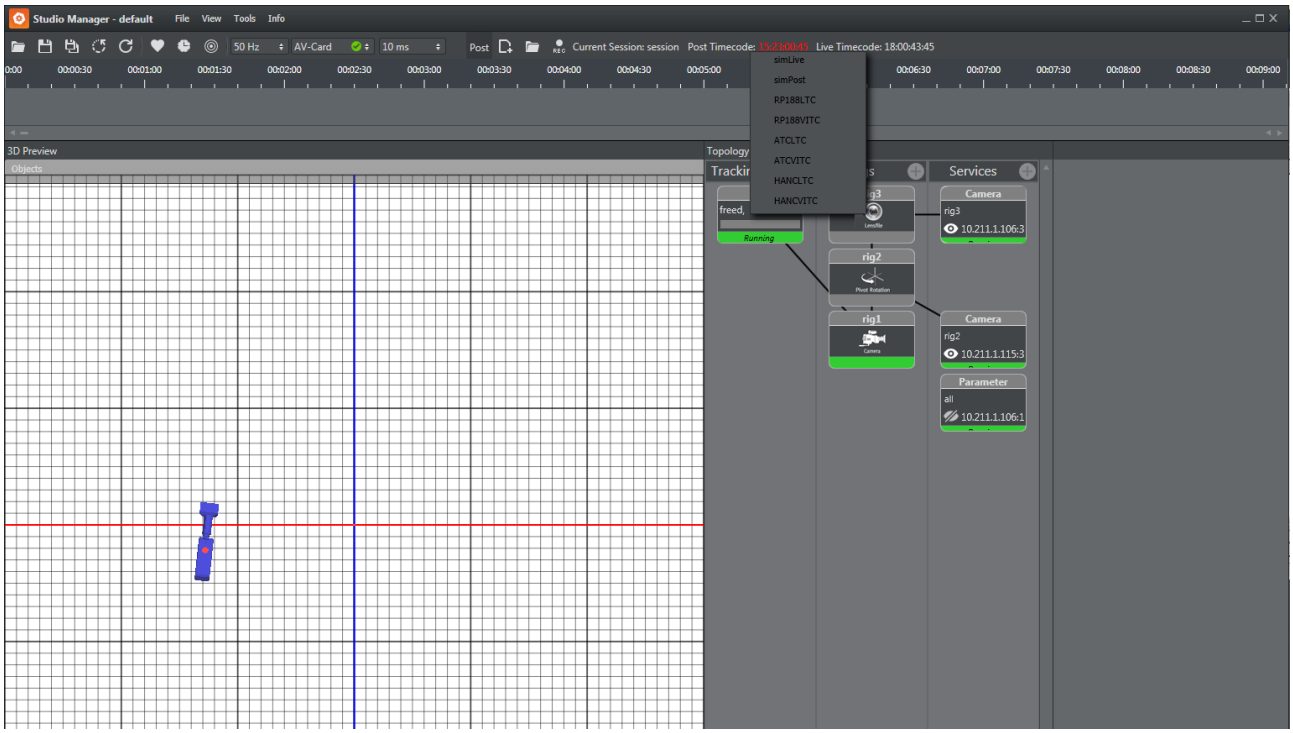
You can automatically load a saved session after Tracking Hub restarts. Activate the **Auto load** switch to enable this feature.



Right click the **Live** or **Post Timecode** labels to select the actual timecode source. A small window appears, which shows the available timecode sources of Tracking Hub.

> **Note:** Tracking Hub must be run in AV-Card mode to recognize the installed Plura timecode card. Otherwise, the timecode sources are not displayed.

SimLive and **SimPost** are used to record tracking data without timecode reader card. This is for analysis purposes only and should never be used for production recording!

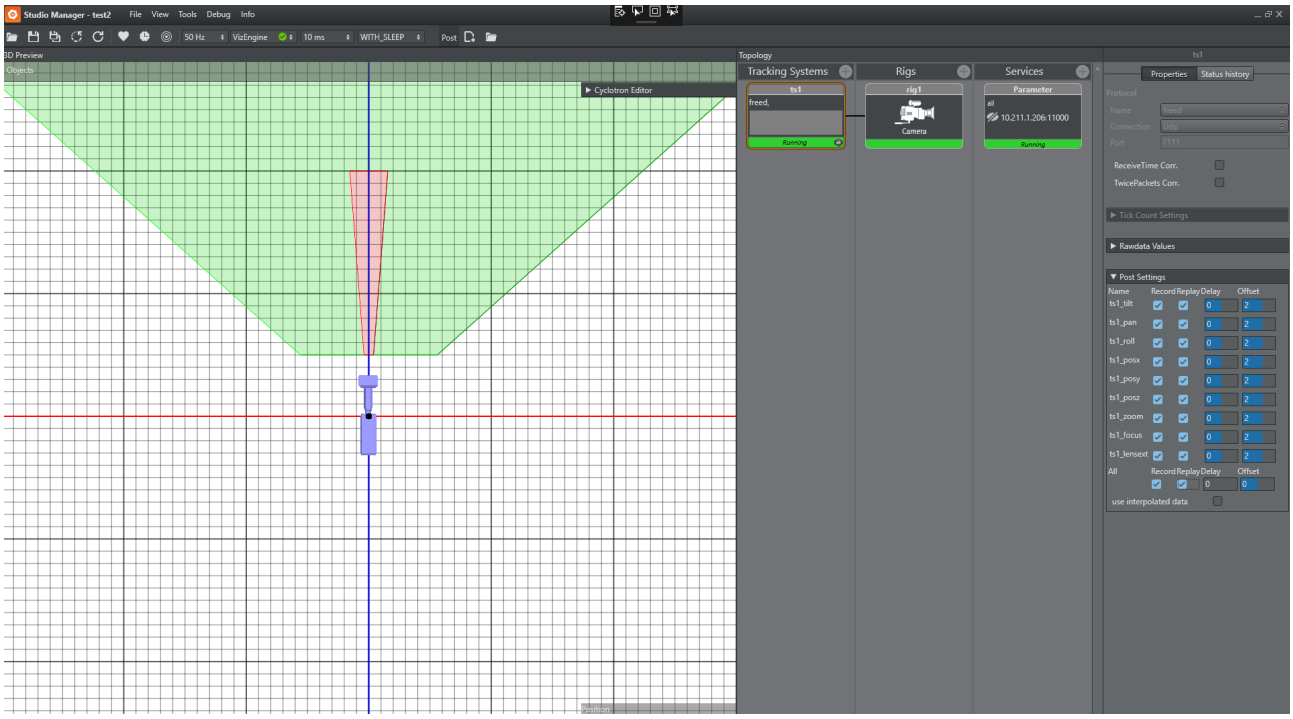


Copyright © 2026 Vizrt

5.6.4 Selecting the Recording and Replay Sources

Tracking Hub can record two sources of data. The parameters coming from the tracking system can be recorded separately, and are sent to the rig during replay. This enables correcting offsets and rig setup after the data was recorded. The second source is the finally baked camera matrix, which is recorded after the rig calculations have taken place. This data can not be modified.

The following screenshot shows the parameter selection for a tracking system. For every parameter, recording and replay can be checked in a check box. An individual delay can be given for every parameter. Replay for a parameter is activated when replay is checked and the post timecode differs by more than ten fields from the live timecode.

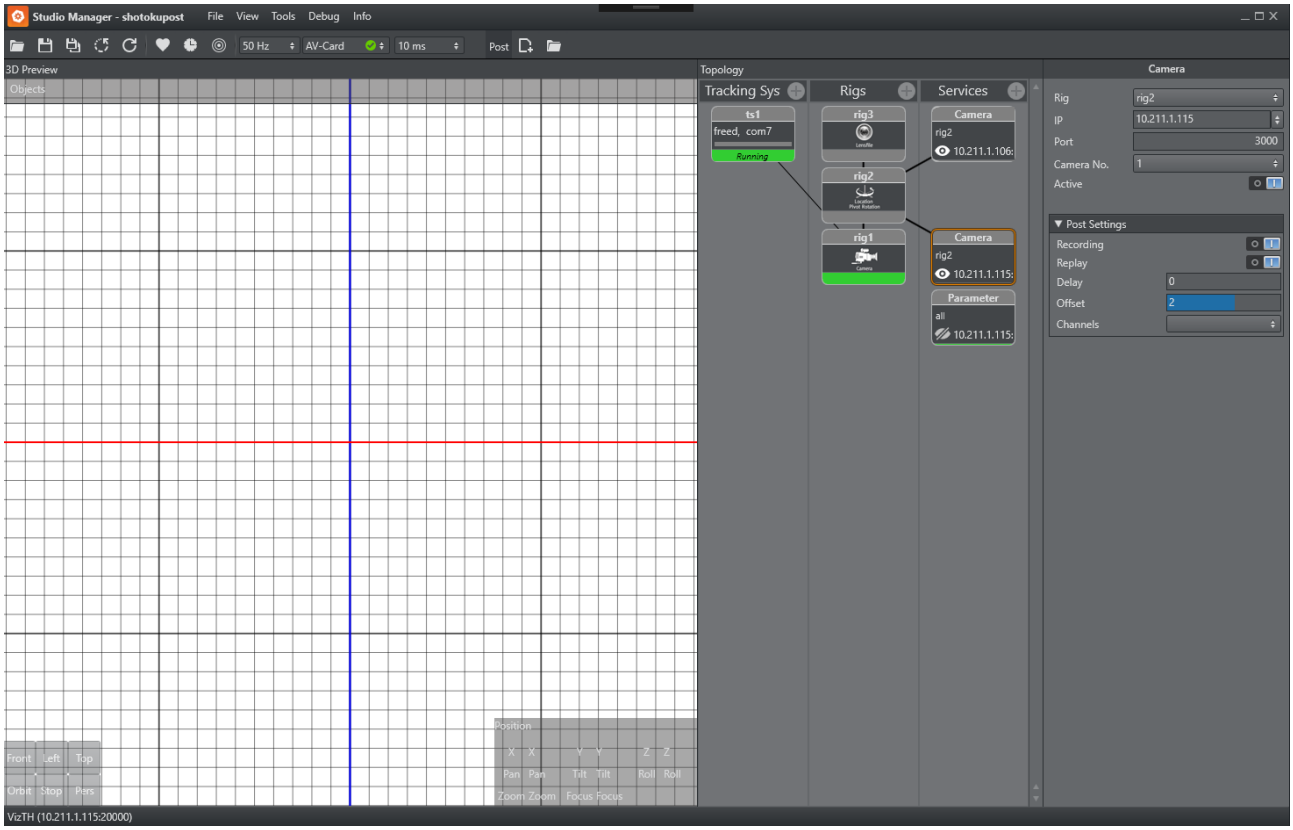


When recording tracking data, Tracking Hub records one package per field. Usually, this is the desired behavior. In rare cases (recording Motion Analysis Data), the tracking system sends more than one package per field. This can happen if a tracking system is running at 60 Hz and the studio is running at 50 Hz. In this case, Tracking Hub interpolates the data and no jitter is visible. In the recorded session, a jitter will be visible. When this happens, the flag **use interpolated data** can be switched on and the post system records the interpolated data.



You can activate recording and replay of the camera matrix in the **Camera Service** settings. There is also an option for the camera delay.

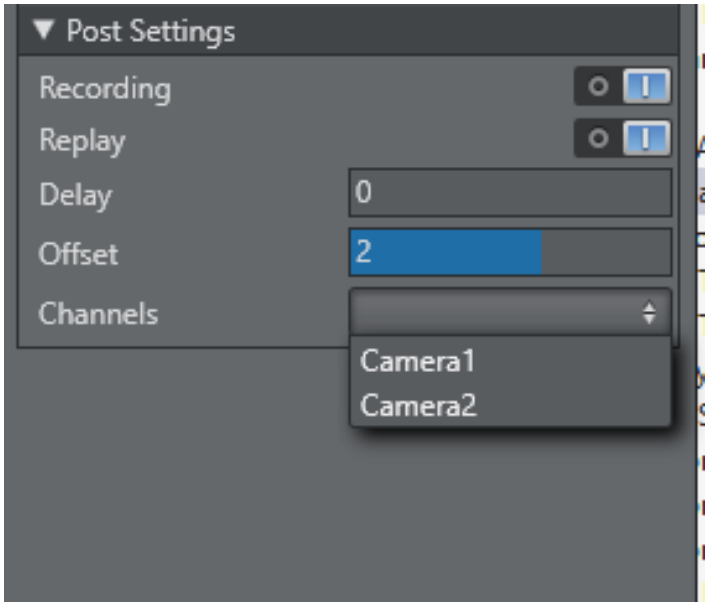
Copyright © 2026 Vizrt



5.6.5 Camera Channels

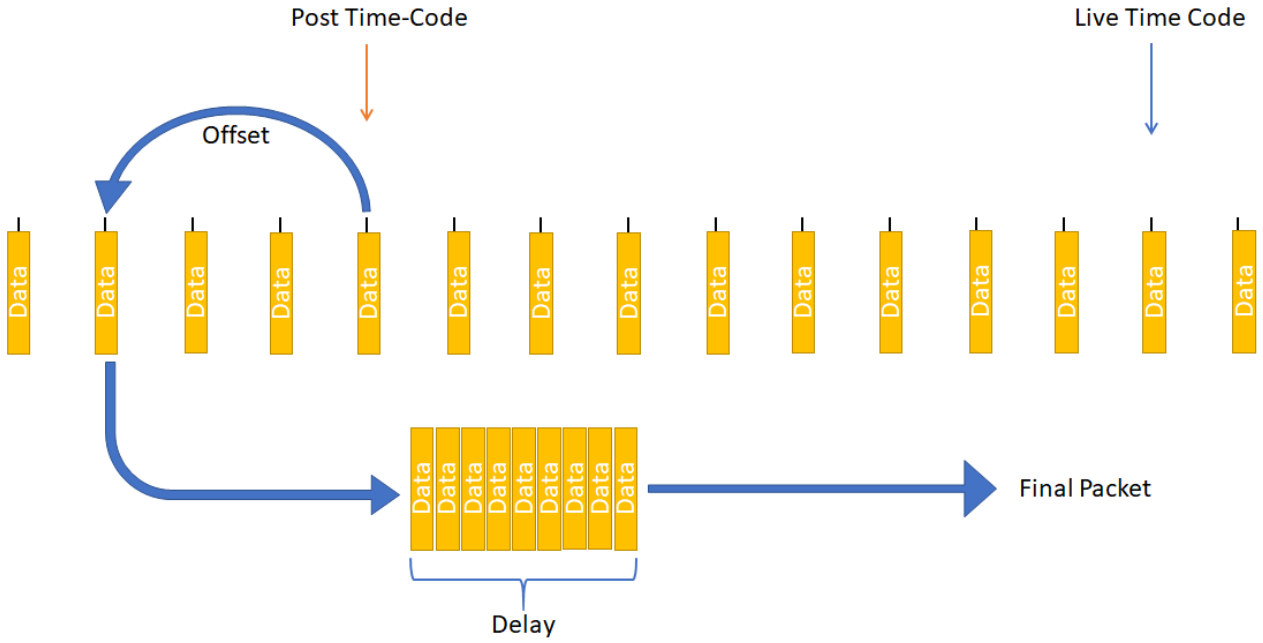
When loading a session that has more than one camera, the camera channels are named by the number in creation order of the service. You can assign any channel that has been loaded from a former session to any camera service. To assign a channel to a service, choose one of the available channels from the drop down menu. If nothing is selected, the channel with the number of the camera service is used.

Copyright © 2026 Vizrt



5.6.6 Post Offset and Delay

Tracking Hub offers two types of delays. One is an offset that is added or subtracted (depending on the sign) from the actual Post timecode. The typical offset for Parameter recording is `-2` fields, and for a camera matrix recording it is usually two fields. The second delay setting is a ring buffer. This is useful in configurations where shuttling back and forward is used.



Copyright © 2026 Vizrt

5.6.7 Post Data Storage

The post data is written to file immediately after being received. In addition, Tracking Hub keeps the most recent packages of each post channel in memory — one package per field, up to the number set by `PostBufferSize`. With its default of 1,000,000 packages that is roughly five and a half hours at 50 Hz, or four and a half hours at 60 Hz.

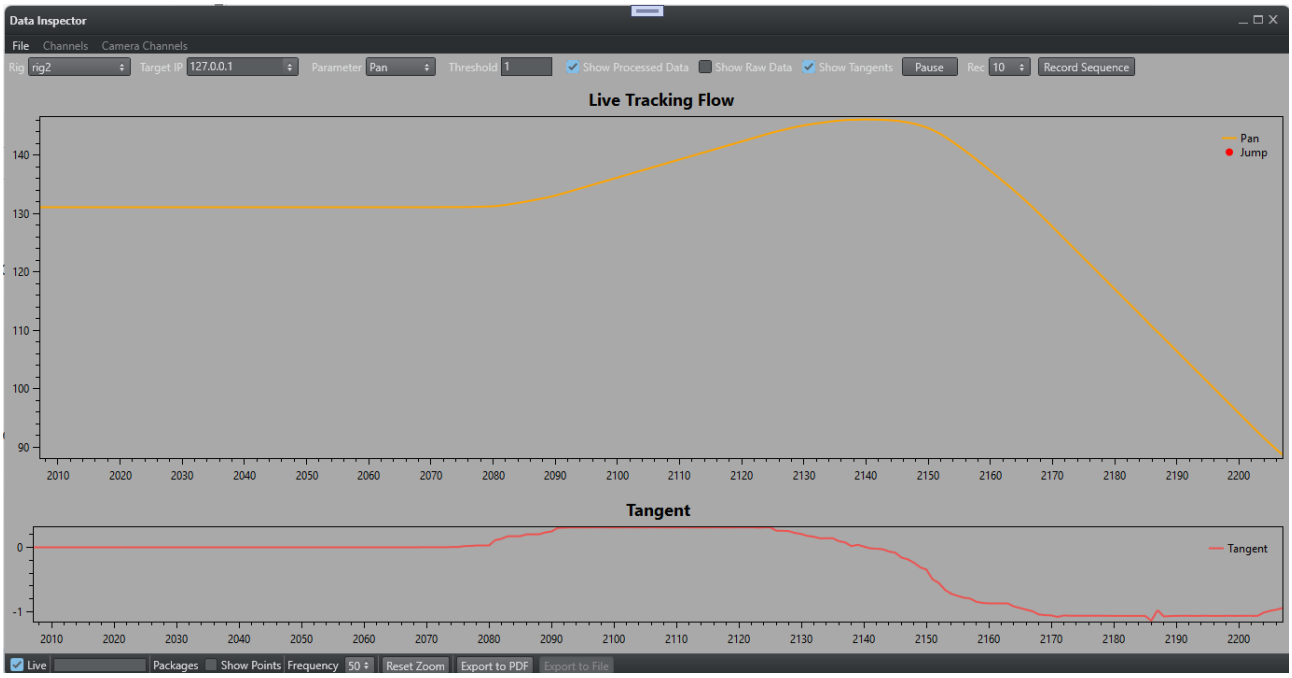
- The storage location is *C:\ProgramData\vizrt\VizTH\Post*.
- The ring buffer size of the post channels can be changed in the *BaseConfig.xml*.
- *PostBufferSize* is the number of packages stored before the oldest is deleted.
- *PostFileCutTime* is the time when Tracking Hub creates a new save file for the PostData. The *PostFileCutTime* gives the **Time** in minutes.

Example: `PostBufferSize="1000000" PostFileCutTime="0"`

Copyright © 2026 Vizrt

5.7 Data Inspector

5.7.1 1 What the Data Inspector is for



The Data Inspector shows tracking data as curves, so you can see what the system is actually delivering instead of inferring it from the picture. It answers questions such as: is the camera data smooth or does it jitter, does a value jump, does the delay setting fit, and does what leaves the TrackingHub match what the tracker sends in.

It has two modes:

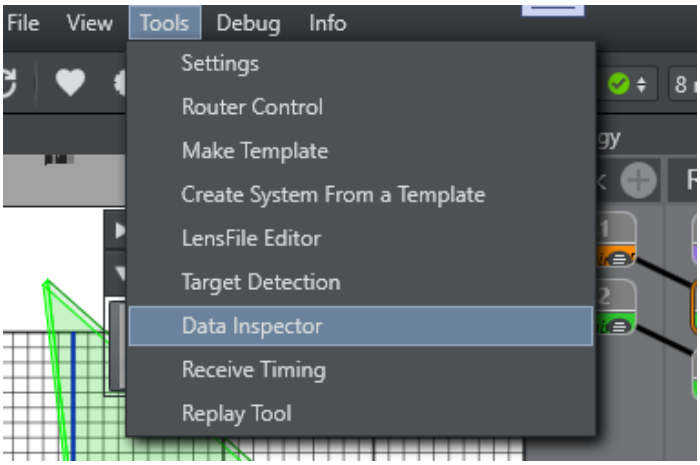
· **Post data mode** plots a recording from disk. Use it to examine a take after the fact and to export it, including as FBX for a 3D application.

· **Live mode** plots the data arriving right now, as a rolling window. Use it while setting up, or while hunting a problem on the floor.

Note Live mode replaces the former Data Flow Inspector window. That separate tool is gone; everything it did is now part of the Data Inspector.

Copyright © 2026 Vizrt

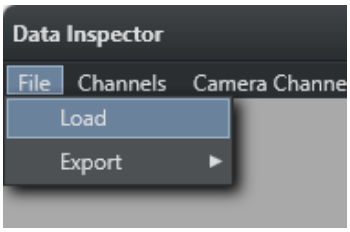
1.1 Opening it



In StudioManager, choose **Tools** → **Data Inspector**. The window opens on the post data view.

5.7.2 2 Post data mode

2.1 Loading a recording



Choose **File** → **Load** and pick a .VPD recording. Recordings live under C:\ProgramData\vizrt\VizTH\Post by default.

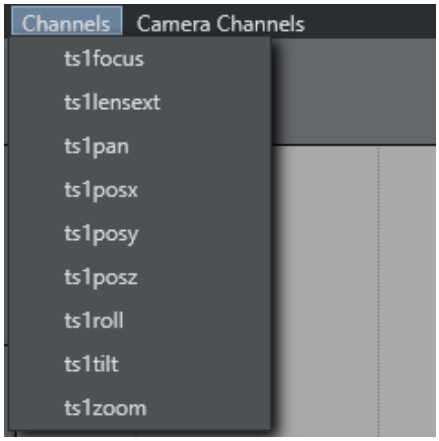
The progress bar in the status bar fills while the file is read, and **Packages** then shows how many packets the recording holds.

2.2 Choosing what to see

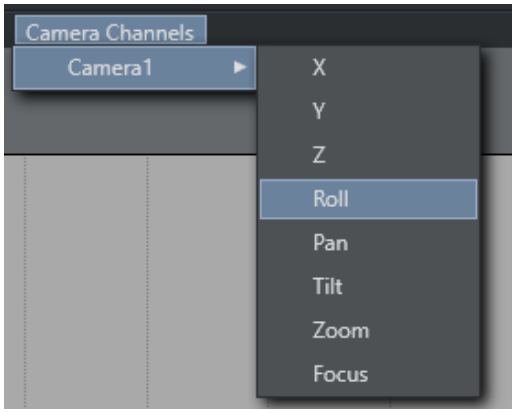
The two menus at the top list what the recording contains:

- **Channels** – the individual parameter channels.

Copyright © 2026 Vizrt



· **Camera Channels** – the camera recordings, grouped per camera, with one entry per value: pan, tilt, position, lens and so on.



Each entry is a toggle: tick it to add its curve to the plot, untick it to remove it.

2.3 Reading the plot

Control	What it does
Show Points	Marks every individual sample with a triangle. Use it to see whether a curve really has one sample per frame, or whether packets are missing.
Frequency	The capture rate of the recording: 59.94, 60 or 50. It sets the time base of the horizontal axis, so pick the rate the system was running at.
Mouse wheel	Zooms. Dragging pans the view.

Copyright © 2026 Vizrt

Reset Zoom	Restores the default scaling of all charts. A middle-mouse double click does the same.

2.4 Exporting

Export to PDF writes the plot as you currently see it.

Export to File opens the Export Data dialog. Pick what to export – **Cameras Only**, **All Data** or **Selected** – and then a format.

Format	Contents
XML	The selected channels with their values and timecodes.
CSV	One row per packet, for a spreadsheet or your own analysis. Decimal points are always "." regardless of the Windows region setting, so the file opens the same way everywhere.
FBX	The camera recordings as an animated 3D camera, for Blender, Maya, 3ds Max and similar. Only camera channels are exported.

2.5 FBX export options

Three settings appear once you choose FBX:

· **FBX channels** – tick the cameras that should go into the file. This is an explicit choice because video wall, TC camera and NDI services record under the same CameraN names as the real camera services and cannot be told apart from the recording alone.

· **FBX source unit** – the length unit the recording is in. StudioManager works in centimetres, which is the default; choose metres only if the recorded positions are in metres.

· **Axis mapping** – collapsed and inactive by default. It lets you invert or re-assign individual axes before export, for recordings that already carry a swapped or inverted axis, so an old take can still be exported without re-recording it. **Reset to recorded order** returns it to the untouched state.

Tip A correctly configured rig needs no axis mapping. If you find yourself changing it for every export, the rig configuration is what should be fixed.

The exported camera carries animated position, rotation and focal length, and the animation is as long as the recording.

Copyright © 2026 Vizrt

5.7.3 3 Live mode

3.1 Switching it on

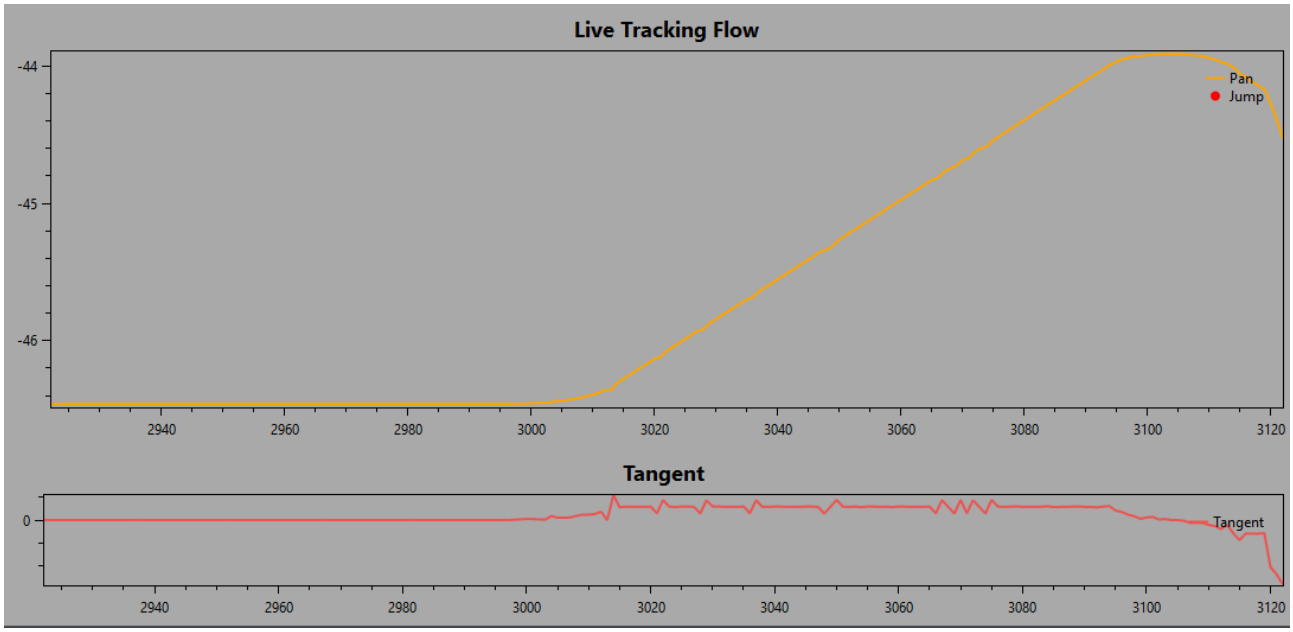
Tick **Live** at the bottom left. The post plot is replaced by two charts and a toolbar.



Live mode needs a connection to the TrackingHub. Without one the tick is refused with a message and the checkbox springs back.

Note While live mode is on, the channel menus and Export to File are greyed out. They work on a loaded file, which live mode does not have.

3.2 The two charts



· **Upper chart** – the value itself, as a window of the last few hundred packets that scrolls along as data arrives.

· **Lower chart** – the tangent, that is the change from one packet to the next. A clean movement gives a smooth tangent; a spike in the tangent is a step in the data that the picture will show as a jolt.

Copyright © 2026 Vizrt

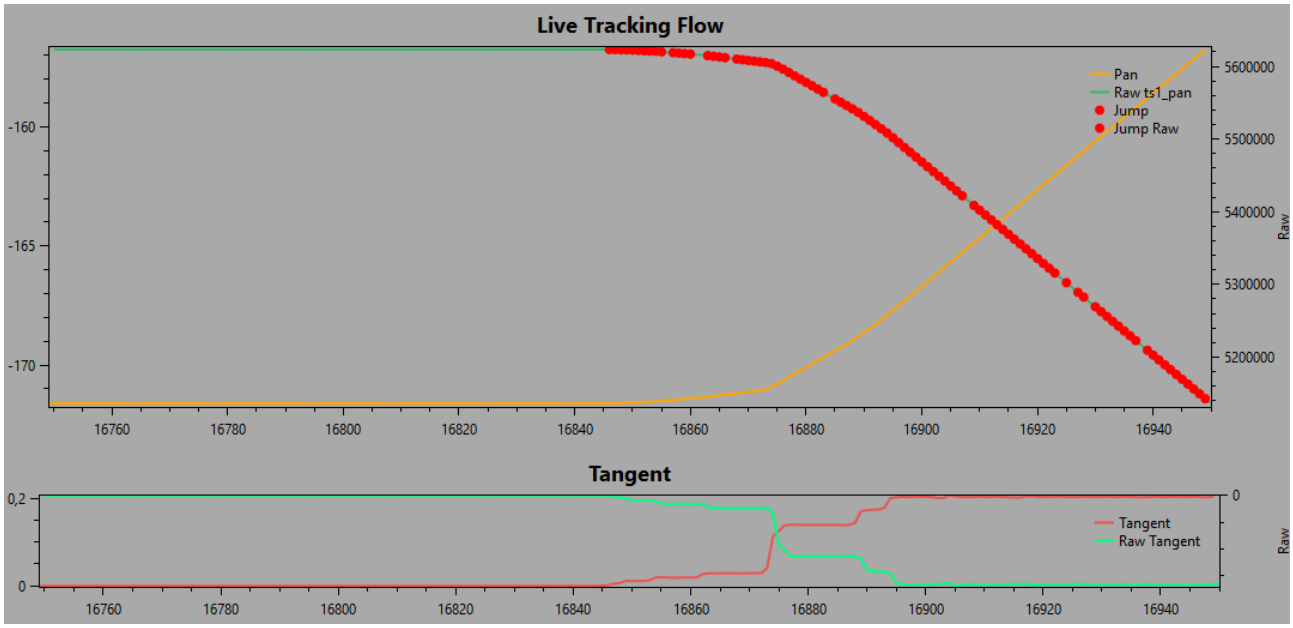
Show Tangents hides and restores the lower chart. Both charts always share the same horizontal range, so a spike in the tangent sits directly under the sample that caused it.

3.3 Choosing what to plot



Control	What it does
Rig	The camera rig whose output you want to see.
Target IP	Where the TrackingHub sends the stream. Leave it at 127.0.0.1 when StudioManager and the hub are on the same machine; enter this machine's address when the hub runs elsewhere.
Parameter	Which value to plot – pan, tilt, roll, the three positions, zoom, focus, field of view and the lens values.

3.4 Processed and raw data



Two curves can be shown at the same time, and the difference between them is the point of the exercise:

Copyright © 2026 Vizrt

· **Processed** (orange) is the finished value the TrackingHub sends to the render engine, after the rig calculation, the lens file and all offsets and delays.

· **Raw** (green) is the value the tracker delivers, before that calculation. It is drawn against its own axis on the right, because it is in the tracker's own units – usually counts, not degrees or centimetres.

Comparing the two tells you where a problem sits. If the raw curve is clean and the processed one is not, the cause is in the configuration. If both are equally bad, the tracker or the cabling is.

Switch them on and off with **Show Processed Data** and **Show Raw Data**.

3.5 Where the raw curve comes from

When you switch the raw curve on, a **Raw source:** label appears and names the tracker axis being read, for example Raw source: ts1_pan. The same name appears in the chart legend.

The Data Inspector works this out from the rig itself, by looking up which tracker axis the rig reads for the parameter you selected. That matters, because a rig may well be wired across: a rig can take its Y position from the tracker axis named "posz". Following the rig gets this right where guessing by name would not.

Some values have no tracker axis at all, because they are calculated rather than measured – field of view, the lens distortion values K1 and K2, and the scale factors. For those the label says so, instead of pairing the curve with something unrelated.

3.6 Finding jumps

A sample is marked with a red circle when the change between two samples differs from the previous one by more than **Threshold**. That is a quick way to spot a dropout or a spike without staring at the curve.

The label states the unit of whatever you are plotting: Threshold [°] for the rotations and the field of view, Threshold [cm] for the positions, and plain Threshold for the values that have no unit.

Important The same number is applied to the raw curve as well, but raw values are in tracker counts. With 32768 counts per degree, a threshold that is sensible for the processed curve is exceeded by the raw curve almost always. Judge the raw markers with that in mind.

3.7 Freezing the picture

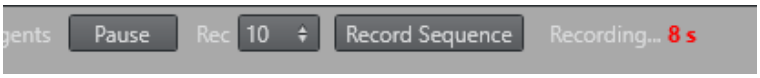
Press **Pause** to freeze both curves so you can study a detail. While frozen you can zoom with the wheel, drag the view and pull a zoom rectangle; both charts move together. **Resume** starts the curves again and returns the view to the scrolling window.

Zooming is available only while paused. On a running plot a manual zoom would switch the automatic scaling off, and the curve would walk out of view while the chart appeared to be stuck.

Note Data keeps arriving during a pause, and a running recording is not affected. What the pause suppresses is the drawing: the packets from that period do not enter the curves and are not scanned for jumps.

Copyright © 2026 Vizrt

3.8 Recording a sequence



1. Set **Rec** to the length you want, between 10 and 60 seconds.
2. Press **Record Sequence**. The status shows Recording... and a red countdown runs down.
3. When it finishes, a save dialog offers a CSV file. The default location is C:\ProgramData\Vizrt\StudioManager\ and the name is TF_<date>_<time>.csv.

The file holds one row per packet with all values of the camera package – rotations, positions, zoom, focus, field of view, centre shift, the lens values and the timestamp.

The length is derived from the **Frequency** setting in the status bar, so ten seconds really are ten seconds at 50, 59.94 and 60 Hz alike.

Controlling the Data-plot

Action	Shortcut
Pan	Right mouse button, **ALT** + Left mouse button, **UP/DOWN/LEFT/RIGHT ARROW** keys, **CTRL** + **ARROW** keys for fine pan
Zoom	Mouse wheel, **CTRL** + Mouse wheel for fine zoom
Zoom in	Mouse extra button 1, **+** (Plus), **PAGE UP**, **CTRL** + **+** (Plus)/**PAGE UP** for fine zoom in
Zoom out	Mouse extra button 2, **-** (Minus), **PAGE DOWN**, **CTRL** + **-** (Minus)/**PAGE DOWN** for fine zoom out
Zoom by rectangle	**CTRL** + Right mouse button, Middle mouse button, **CTRL** + **ALT** + Left mouse button
Reset	**CTRL** + Right mouse button double-click, Middle mouse button double-click, **CTRL** + **ALT** + Left mouse button double-click
Reset axes	**A**, **HOME**
Copy bitmap	**CTRL** + **C**

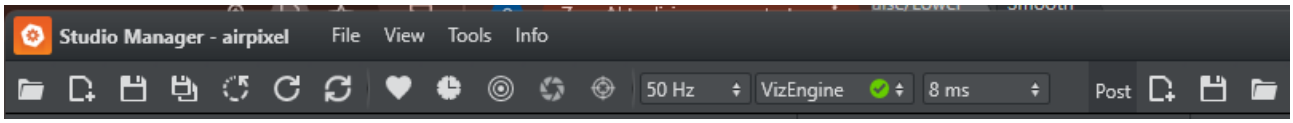
Copyright © 2026 Vizrt

Action	Shortcut
Tracker	Left mouse button, **SHIFT +** Left mouse button for points only tracker, **CTRL +** Left mouse button for free tracker (show mouse coordinates basically),

Copyright © 2026 Vizrt

5.8 Configuration Panel

Use the **Configuration Panel** to save and load Studio configuration profiles, select modes of operation including frequency, and to set some user interface options.



- - **Open:** Opens a Tracking Hub configuration file. The name of the currently loaded configuration file is always displayed in the top bar, between the application name and main menu.
  - **New Configuration:** Creates a new, empty Studio configuration.
  - **Save:** Saves the current Studio configuration.
  - **Save As:** Saves the current Studio configuration with a new name.
  - **Reset Configuration:** Resets the current Studio configuration. Note that the loaded configuration file is only changed if you click Save after resetting the configuration.
  - **Refresh UI:** Reloads the actual configuration. Use this to apply any changes third party software has made to the Tracking Hub parameters, so that the changes can be visualized in the Studio Manager.
  - **Synchronize Backup:** Writes the current configuration to the Tracking Hub. This button appears only when a backup rule, Master or Slave, is configured. Close the Lens File Editor first — while it is open, the command is refused with a message.
  - **View Log:** Opens or closes the Log Panel.
  - **View Timing Analysis:** Opens the Timing Analysis Window.
  - **LensFile Editor:** Opens the Lens File Calibration Tool.
  - **Target Detection / Send Target Data:** Opens the Target Detection dialog. You are asked to select the rig to work with before the dialog opens. The Lens File Editor is not available while the Target Detection dialog is open.

  Note: New Configuration and Reset Configuration are disabled while a camera filter is active. The tool tip of the greyed out button states the reason.
- **Frequency:** Selects the production frequency:
  - `50 Hz`, `60 Hz` or
  - `59.94 Hz`, `29.97Hz`, `30 Hz`, `25 Hz`, `24 Hz`.
- **Synchronization Mode:** Selects synchronization mode for the Tracking Hub:
  - **Freerun:** Does not synchronize. The Tracking Hub uses its own time base, corresponding with the configured frequency.
  - **AV-Card:** Uses an installed *Plura* card to synchronize.
  - **Viz Engine:** Synchronizes with a Viz Engine running on the same computer.
  - **NDI:** Please see: Embedded NDI Tracking Data.

Tip: If the selected synchronization mode is followed by a red box with a white horizontal line, this indicates synchronization issues.

Copyright © 2026 Vizrt

- **Send Delay:** Sets the time Tracking Hub waits until it sends tracking data to Viz Engine. This setting is especially important if your tracking delay is less than one field. With the help of the Timing Analysis, the transmission time can be set after the arrival of the tracking data. If your tracking delay is larger than one field, set the send delay value to **No Delay**. This reduces the CPU usage.

See Also

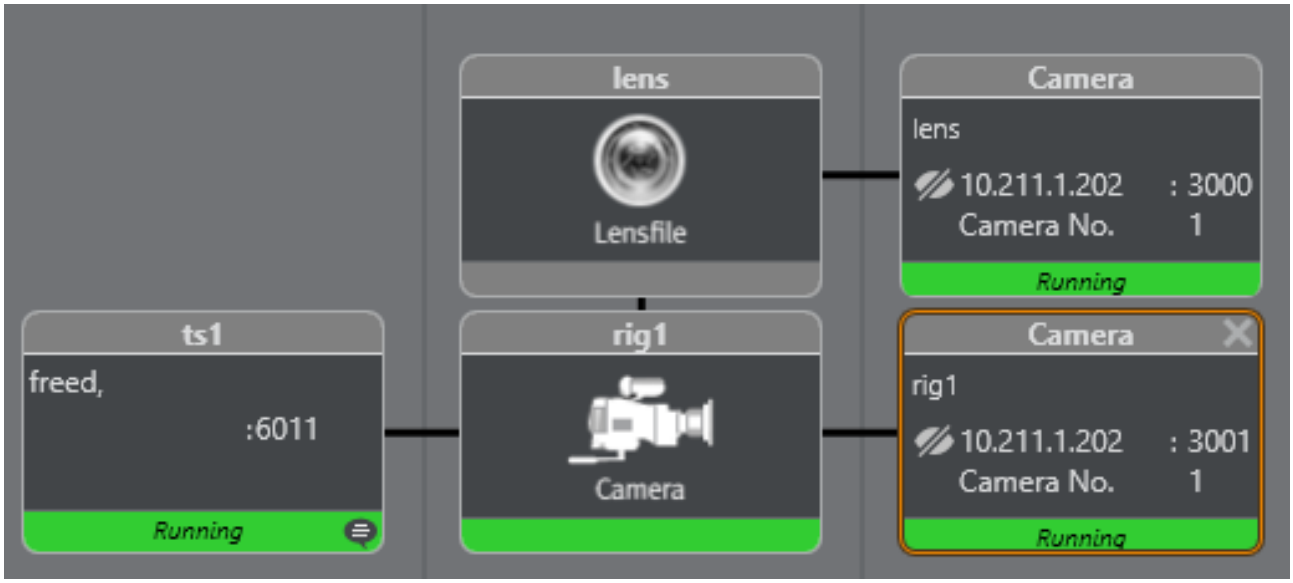
- Parameter Panel
- Topology Panel
- Log Panel
- Preview Panel

Copyright © 2026 Vizrt

5.9 Target Detection

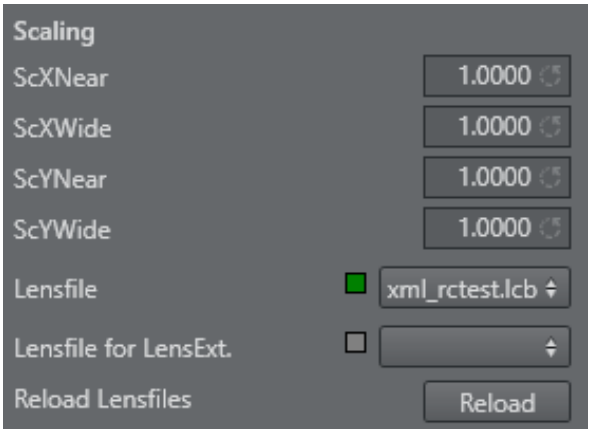
- Viz Engine Scene Preparation
- Target Detection Wizard
- Automated Target Localization
- Target Coverage
- Target Preview
- Target Localization Procedure
    - Target Service
    - Studio Manager Preview

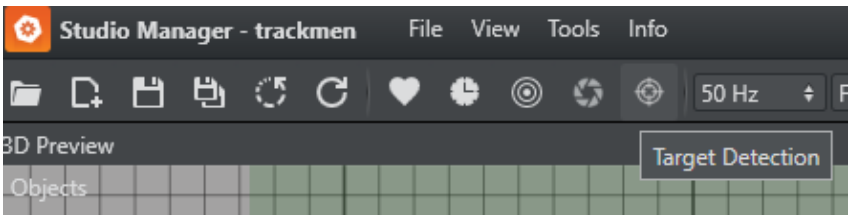
The Target Detection feature can be used to receive the coordinates of a physically placed target inside the studio. This can help start an existing working setup.



- Create a service for the engine, and a service for target detection.
- Assign the appropriate lens file for the used camera, and lens (the one created in the automated lens calibration process).
- During the lens calibration process, a lens file is created along with an intrinsics file (*.dat*). The lens file and the intrinsics file have the same name, for instance *xml_rctest.lcb* and *xml_rctest.dat*.

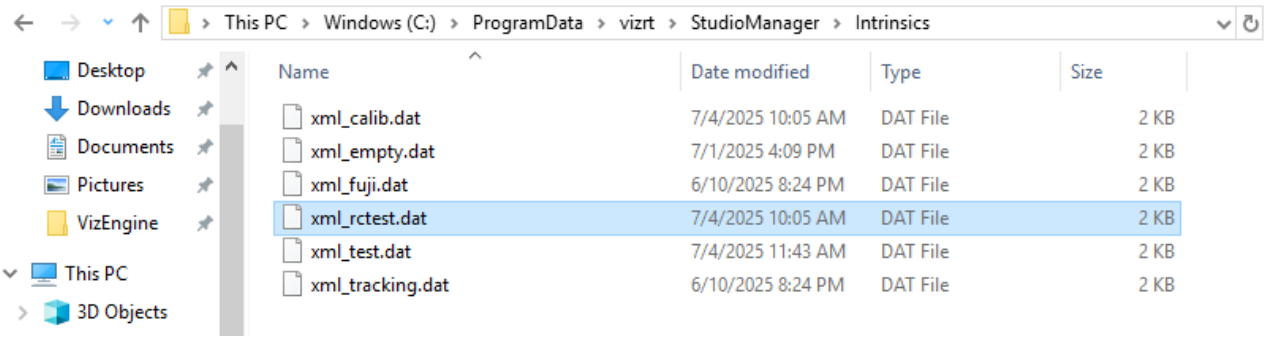
Copyright © 2026 Vizrt





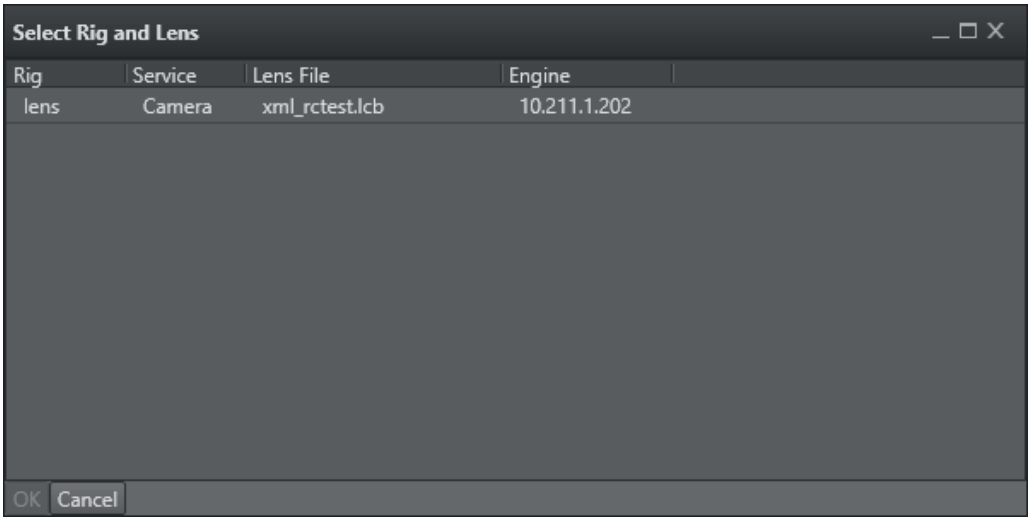
- Intrinsics files contain additional lens information that is required for subsequent target detection. The lens file loaded in the studio manager references the appropriate intrinsics file (for example, in the info field *intrinsics_file="xml_rctest"*).

> **Note:** These files can be found in the Studio Manager folder, and **must not be renamed**.



- Lens files containing a related intrinsics file, appears in the file picker dialog.

Copyright © 2026 Vizrt



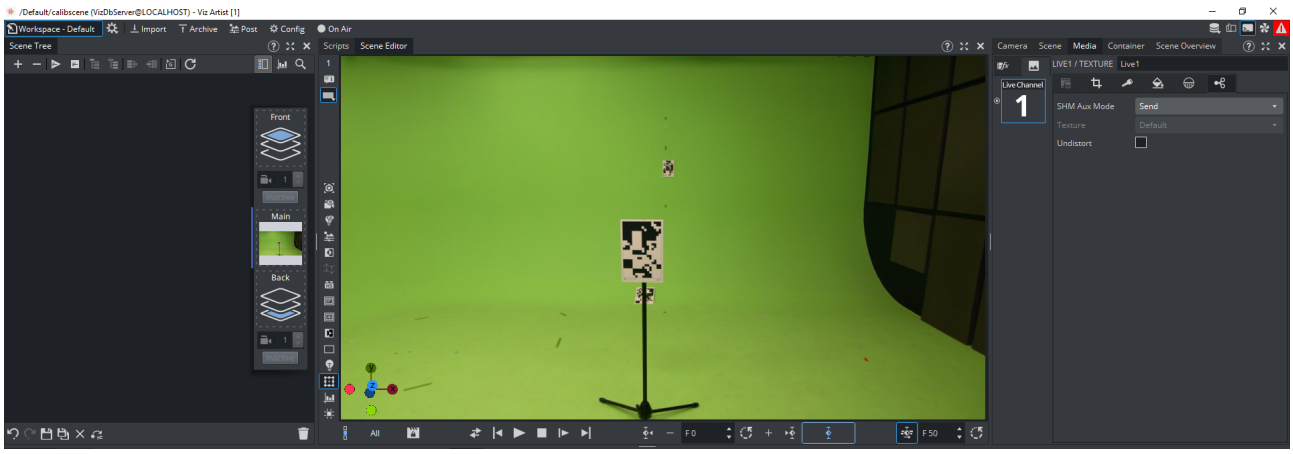
- Loading a file, starts searching for target calibration results stored in a separate file (*C:\ProgramData\vizrt\StudioManager\Calibrationresults)*. This file is created once the target calibration process is running. Like the lens file calibration, the target detection can be done by using the input of a Viz Engine scene, or any existing NDI stream.

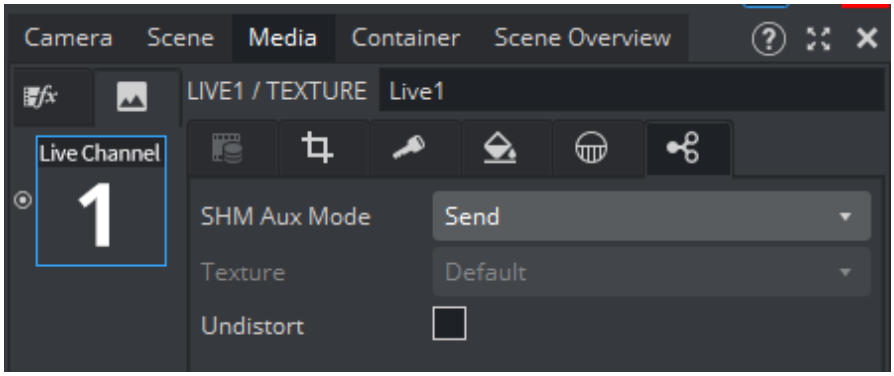
5.9.1 Viz Engine Scene Preparation

If the input on a local Viz Engine is used, a simple scene needs to be created. Therefore, the input needs to be added to the scenes Media texture assets.

- Add it to the media panel or,
- Drag the media asset to the scene settings background image (the texture asset is created automatically).

The media asset **SHM Aux Mode**, needs to be set to **Send**, to share the input with the local calibration tool.





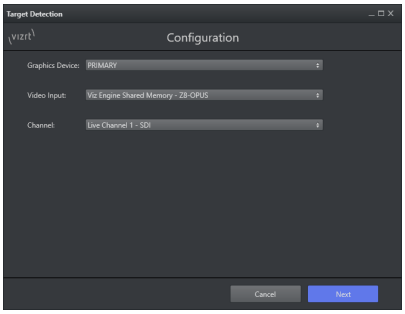
- Set Engine to On Air.

5.9.2 Target Detection Wizard

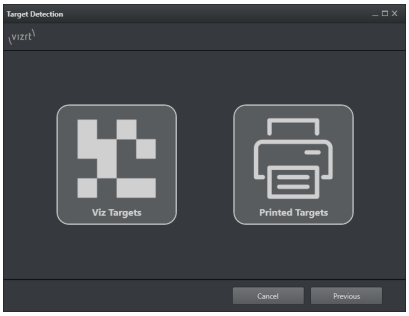
Similar to the Calibration Wizard, the target detection needs to be configured by choosing the used input, and the targets set up in the studio.



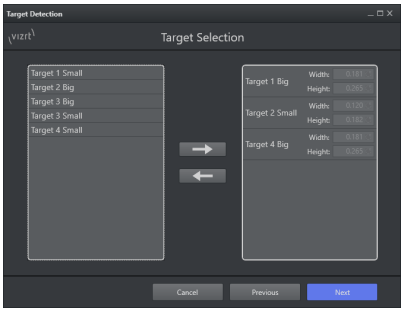
Launch detection.



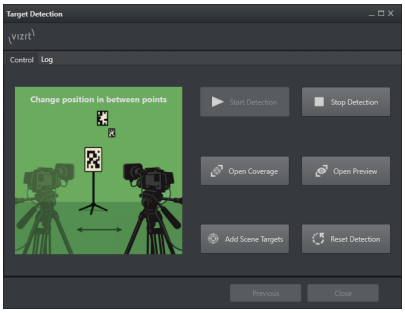
Setup the use case configuration.



Choose the target mode.



Choose the targets used in setup.



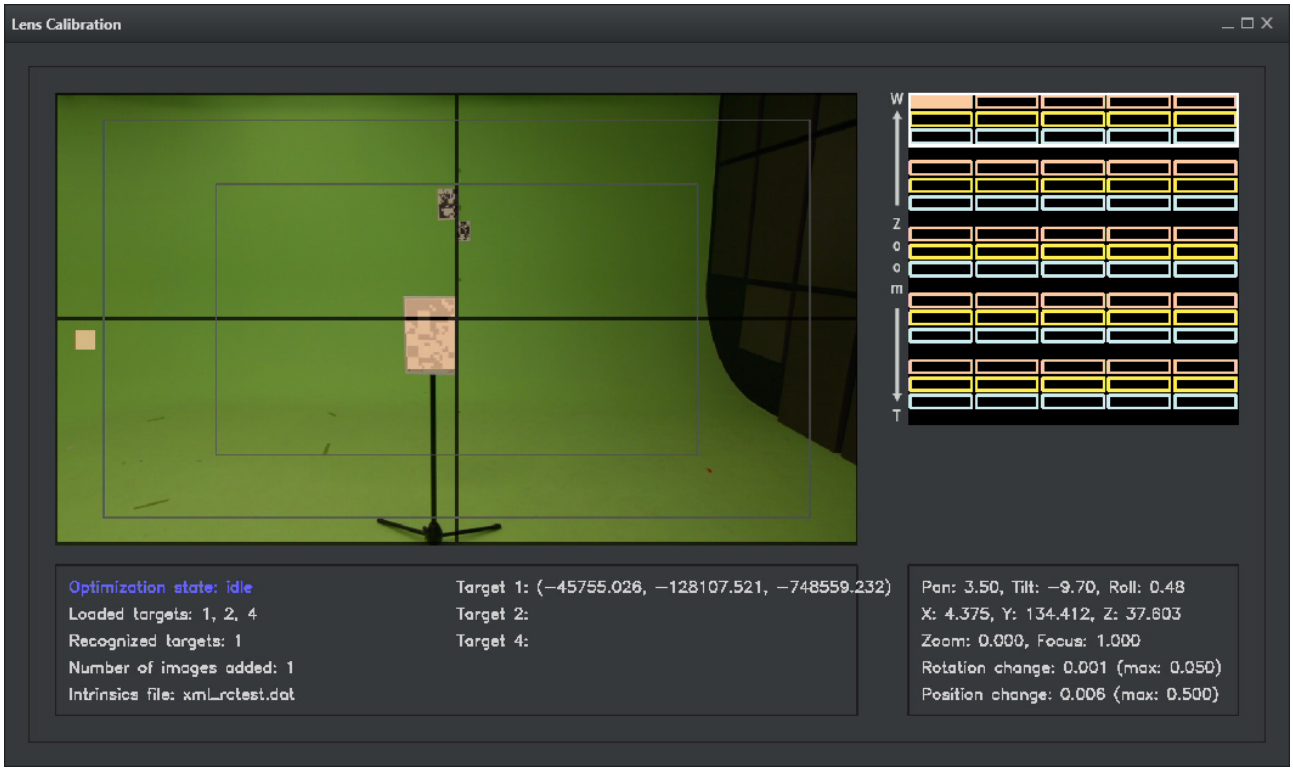
Start detection and open coverage view.

5.9.3 Automated Target Localization

The target localization process uses the pre-calibrated lens parameters from the intrinsics file, to automatically determine target positions in the coordinate system of the camera tracking system. Compared to the full lens calibration, the target localization requires far less images.

For optimal results, the chosen targets should be captured in different image regions, and across several zoom levels. The focus can be set to infinity, or to a level that provides sharp images.

5.9.4 Target Coverage



The input image is split into four regions. The top left corner of each image indicates which target was captured in this region. Every target can be captured once in every image region.

The graph on the right shows how well the zoom levels and the different camera positions are covered. The zoom is split into five levels (vertical axis), and every zoom level is split into five camera positions (horizontal axis). Targets are color coded, and there is one row for each target for each zoom level.

The panel at the bottom left shows the current positions computed for the targets.

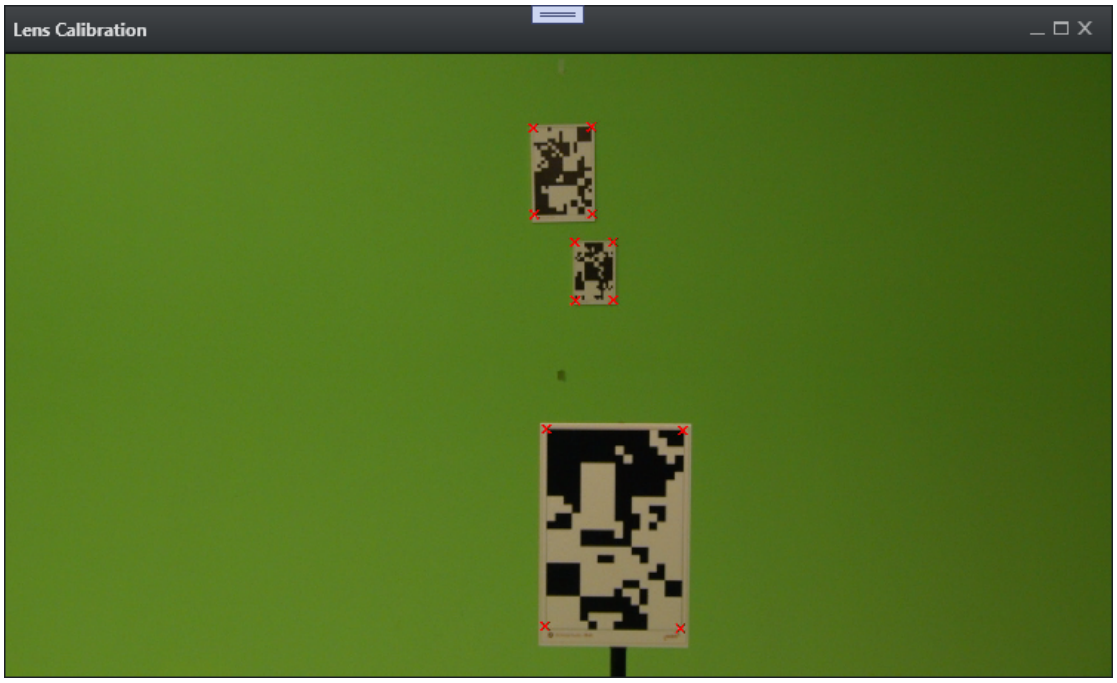
For accurate results, it is usually not necessary to cover the full zoom and position range. You may capture the targets in different image regions on several zoom levels, from different camera positions, and validate the result using the target preview, or, the Viz Engine Scene by moving the camera around and changing its zoom. If it is accurate enough, the process can be stopped. Otherwise, you should try to capture more images.

Before stopping the process, wait for the optimization state to become idle.

Copyright © 2026 Vizrt

5.9.5 Target Preview

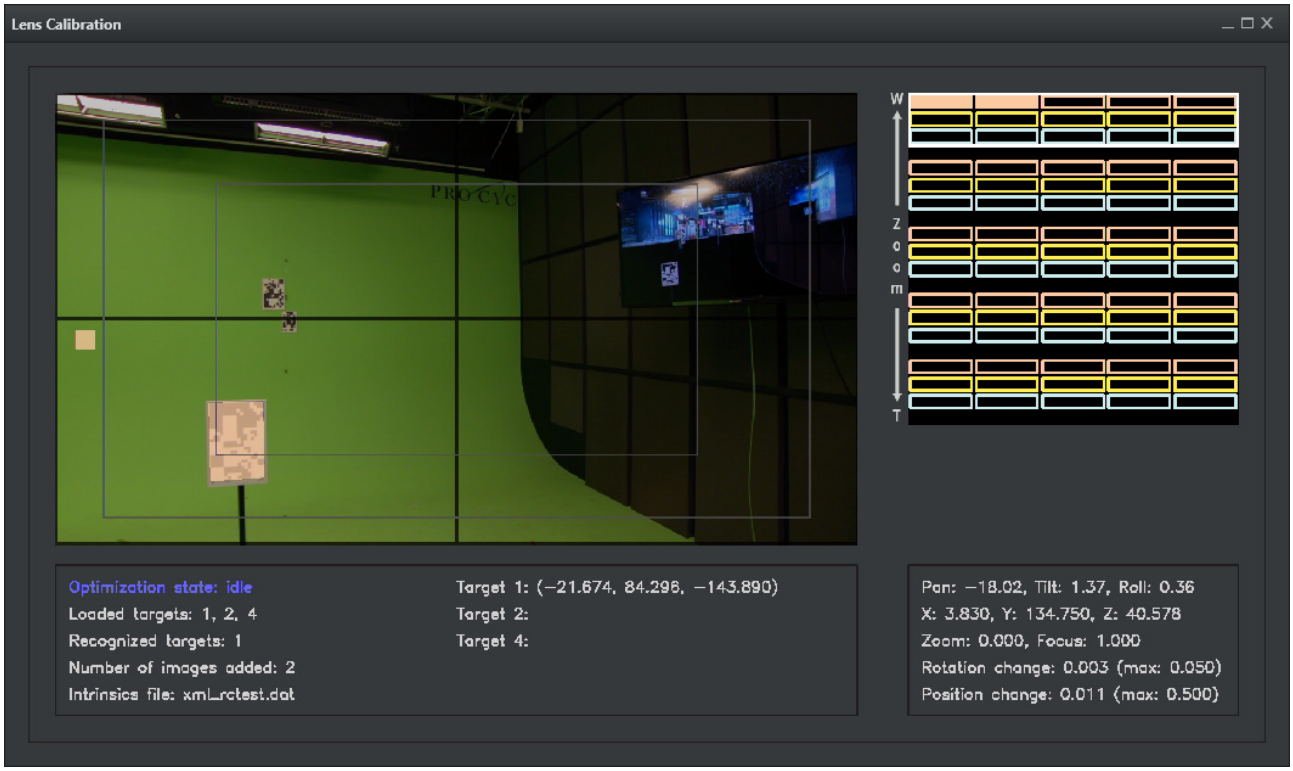
Similar to the lens calibration, the current results can be observed in the target preview. The preview and the Viz Engine Scene can be used to check the accuracy of the target positions, and to determine when to stop the localization process.



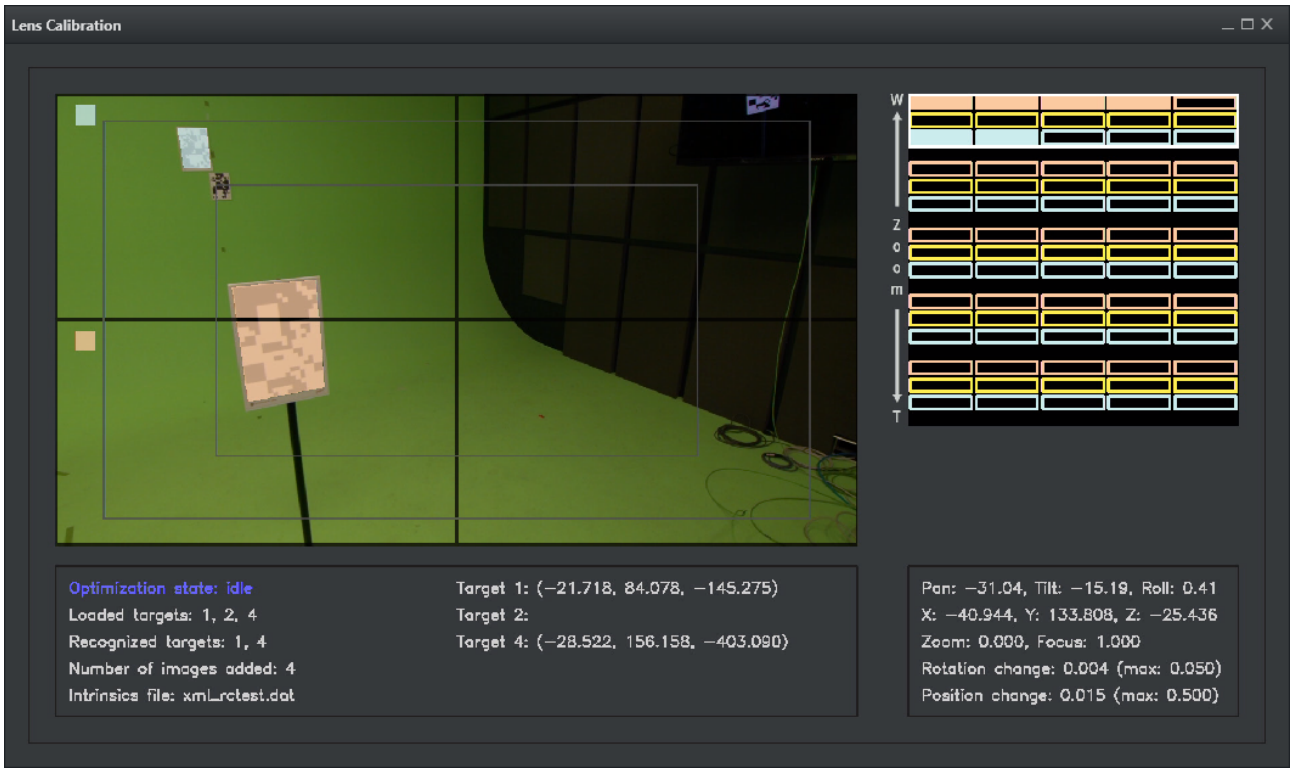
5.9.6 Target Localization Procedure

Start with the first camera position. Pan and tilt, to detect targets in all four areas.

Copyright © 2026 Vizrt

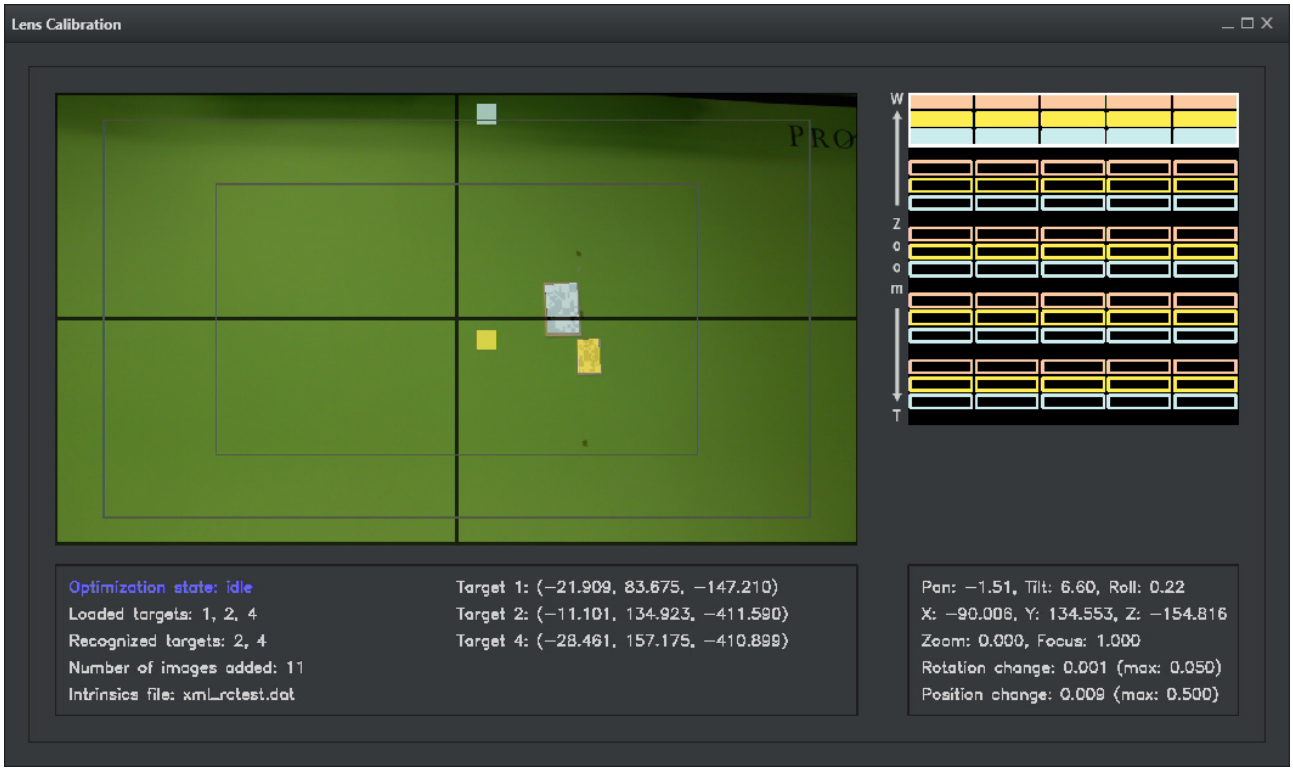


Try to successfully capture all targets in the different image regions. Repeat this step for different camera positions.



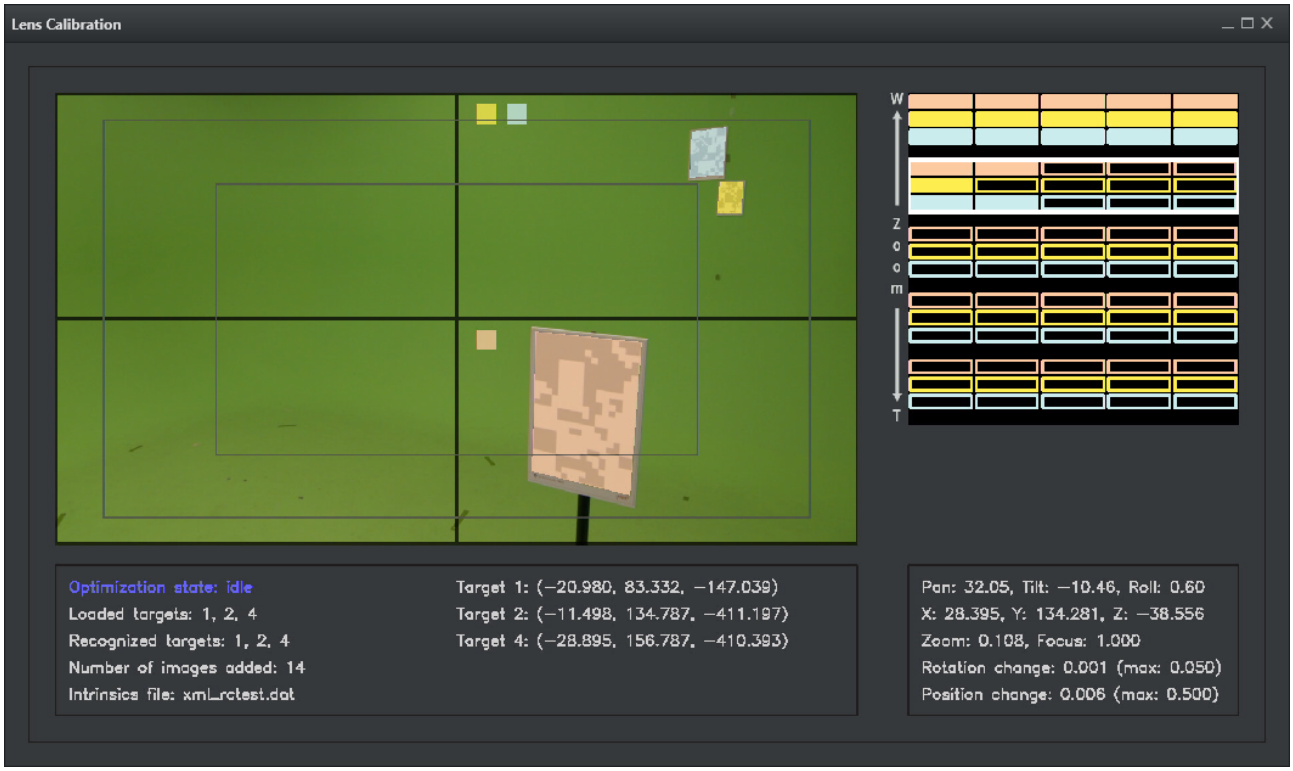
Copyright © 2026 Vizrt

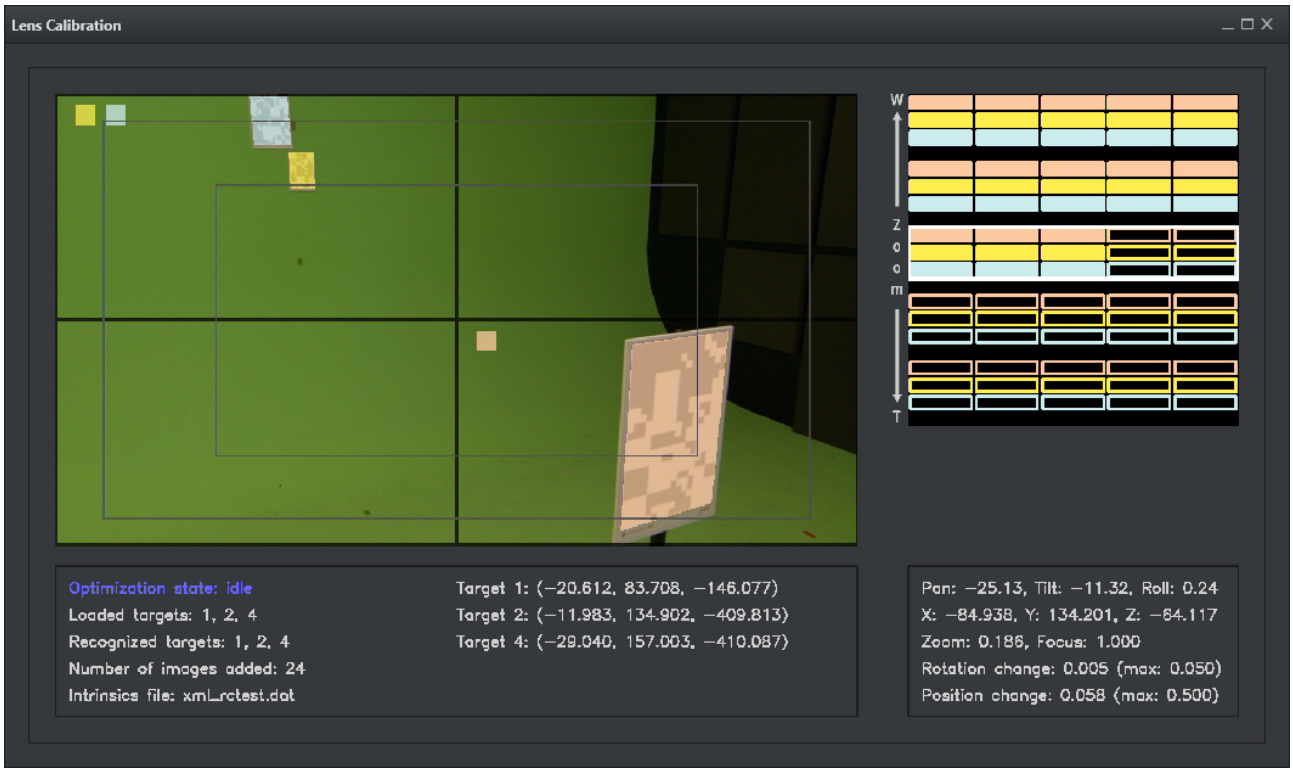
Up to five camera positions can be stored for a chosen zoom level. It is not mandatory to fill out all five positions, but by choosing more positions, the results improve.



Repeat the steps above for other zoom levels.

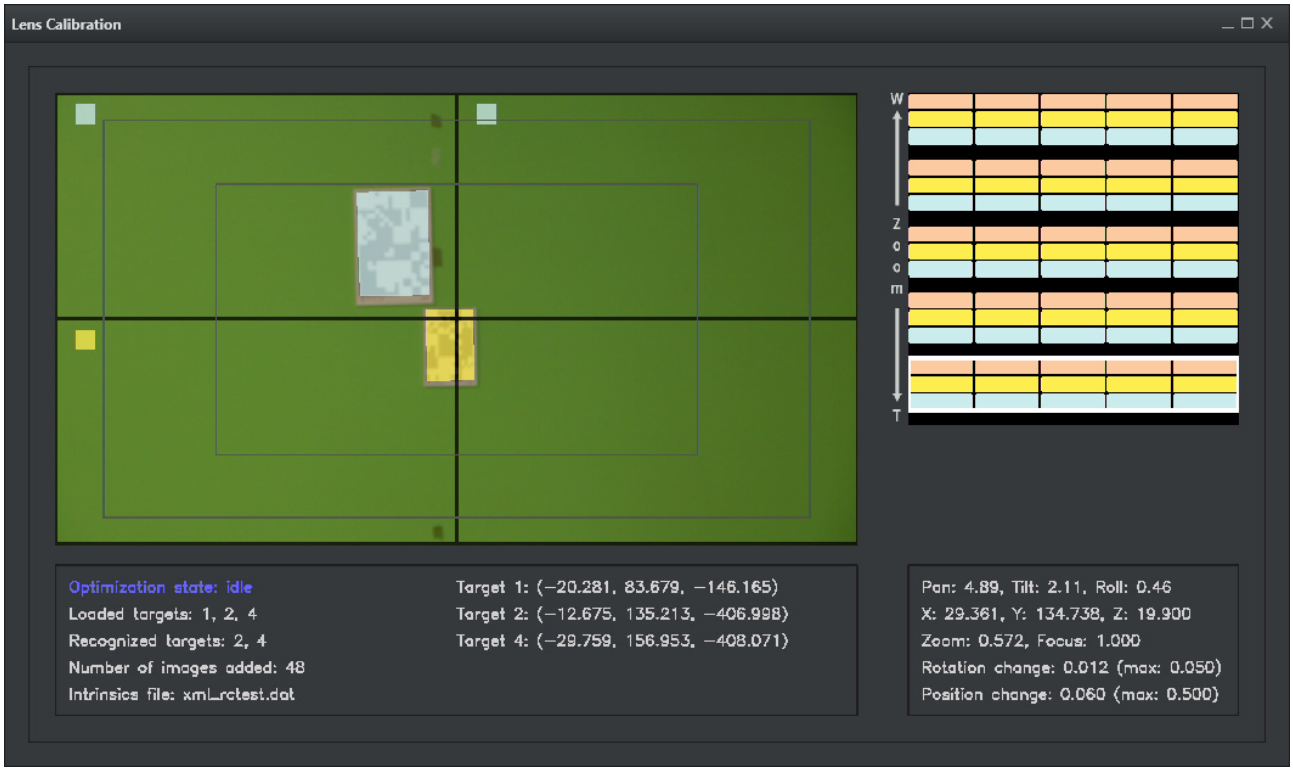
Copyright © 2026 Vizrt





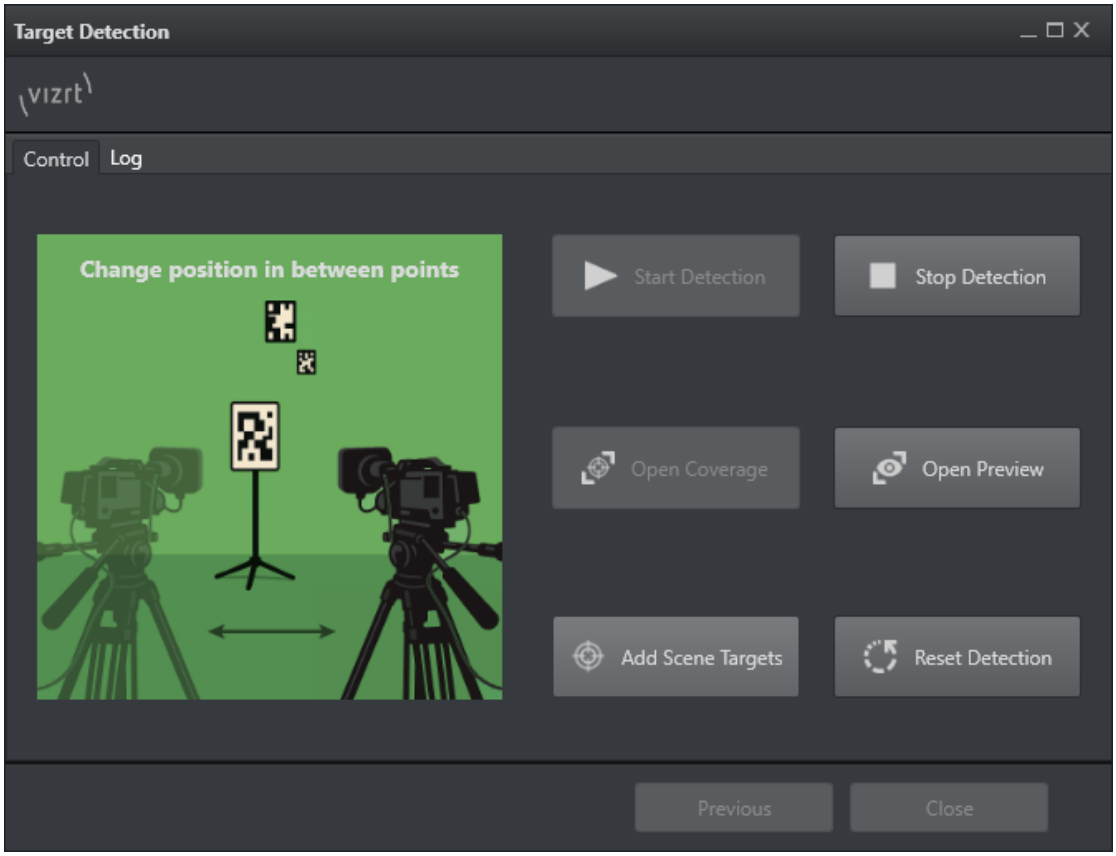
The detected target position is automatically streamed stored.

Copyright © 2026 Vizrt



In general, the more zoom levels and camera positions per target, and the more image regions for each of those combinations are covered, the more accurate the results. However, in practice, it is usually not necessary to cover everything. You can use the preview, and the Viz Engine Scene to check its quality. If it is accurate for all the positions and zoom level, you may operate. The process can be stopped earlier, but wait for the optimization state to become idle.

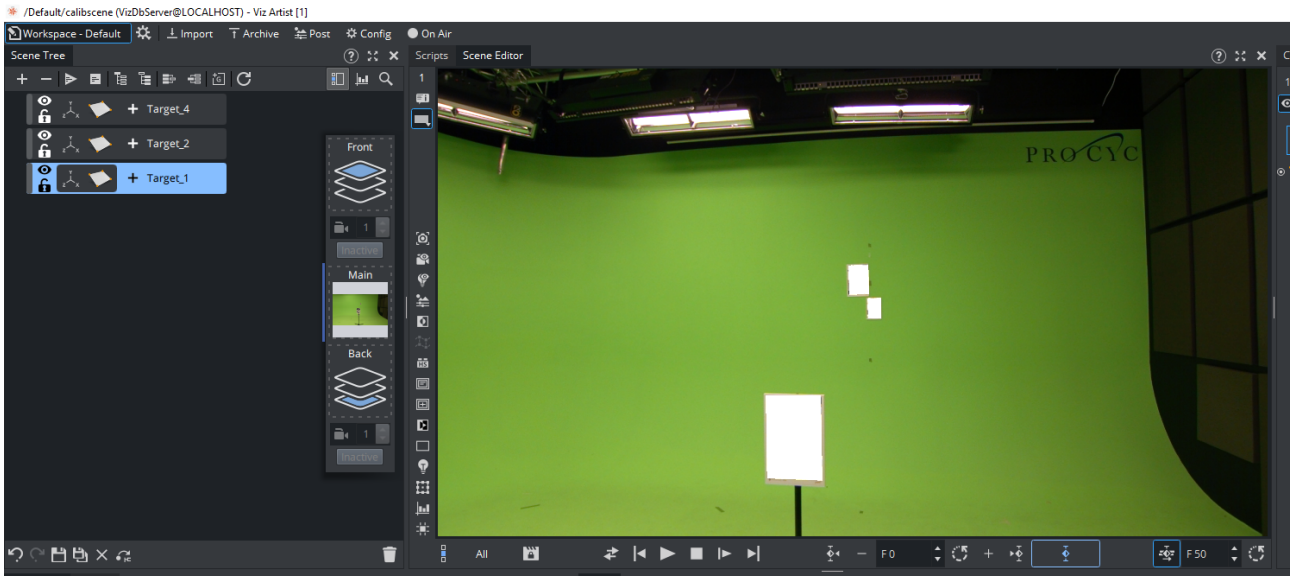
Copyright © 2026 Vizrt



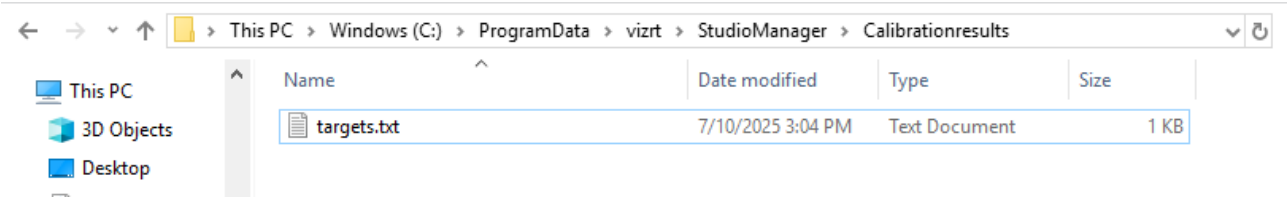
The Wizard offers the option to reset the detection, and start from scratch.

The **Add Scene Targets** button creates containers in the existing, currently loaded Viz Engine scene.

RectXYZ is copied to a separate container, and filled with the detected coordinates for each target. This allows to show the targets in any existing VS/AR scene, without creating them manually.

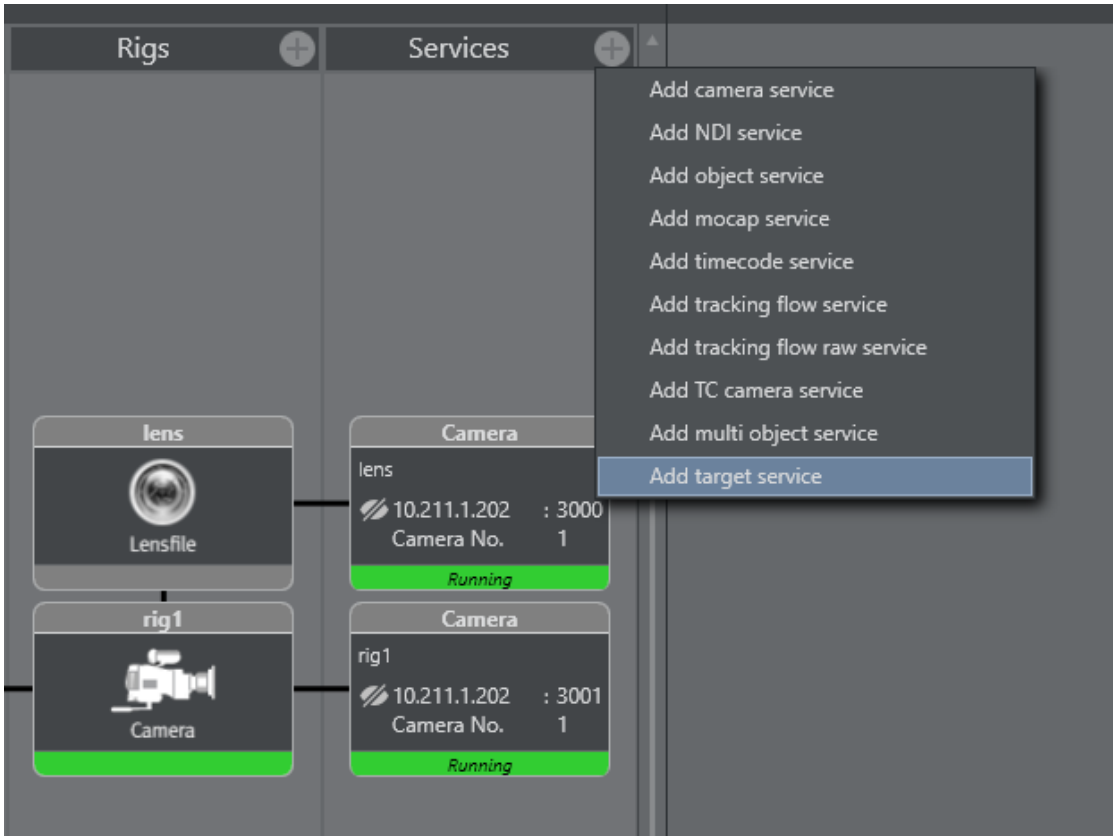


The targets coordinates are also stored in a separate file at *C:\ProgramData\vizrt\StudioManager\Calibrationresults*. The file gets reset every time a new detection starts.



Target Service

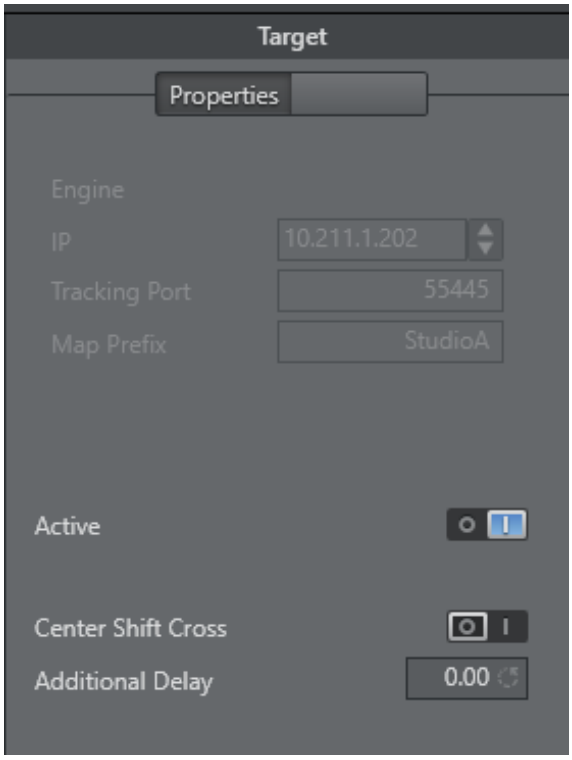
In order to stream the targets result into the engines shared memory, a **target service** can be used. This allows the usage of the detected coordinates within a scene (for example, by using scripting to implement an alignment logic of containers/objects according to a targets position).



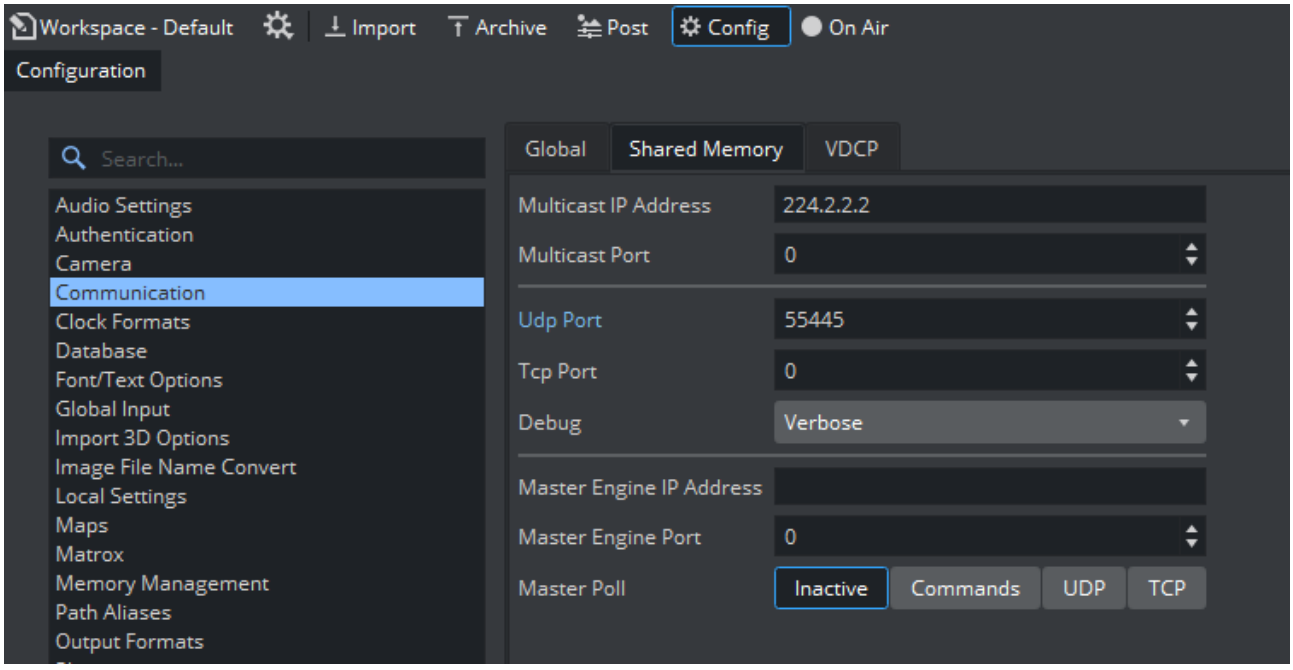
Set up the port the engine is using for shared memory configuration (setting can be found in **Communication > Share Memory Tab**). By default, no port is set in the Viz Engine configuration (see image below).

> **Note:** The Engine needs a restart when the port is set or changed.

Copyright © 2026 Vizrt



A **Map Prefix** can be added to each shared memory map to avoid conflicts with other studios/scenes.

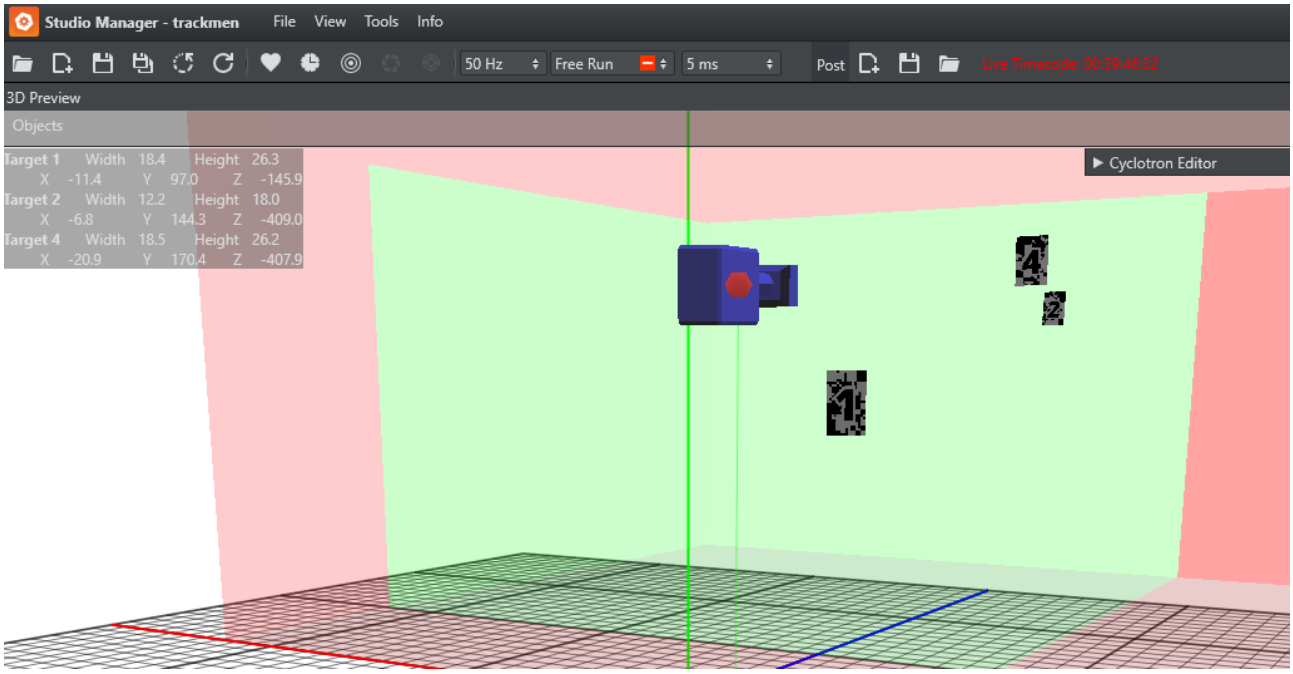


The **Debug** section prints incoming packages when set to **Verbose**.

Copyright © 2026 Vizrt

Studio Manager Preview

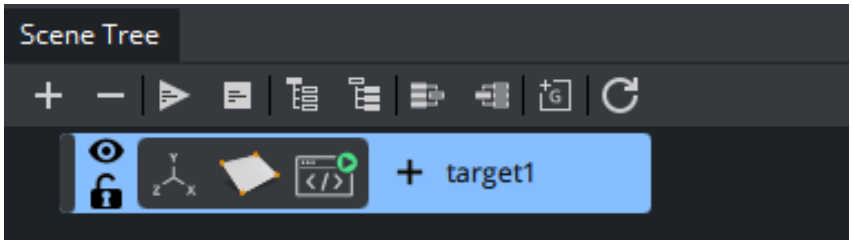
Within the Studio Manager, a Target HUD can be activated in the **View** settings. This shows the currently detected center of each physical target. The 3D preview renders the targets as well.



Copyright © 2026 Vizrt

5.9.7 Example Script

A simple scene can be created to have the RectXYZ plug-in updated within a Viz Engine Scene.



Create a container, add the RectXYZ and a script plug-ins. Copy the following script into the script editor and compile.

```
dim coordinates as Array[String]
dim targets as Array[String]

targets.Push("target1")
targets.Push("target2")
targets.Push("target3")
targets.Push("target4")

sub OnInitParameters()
    RegisterParameterString("prefix", "Prefix","", 50,255, "")
    RegisterParameterDropDown("my_target","Choose Target",1, targets, 100, 50)
    RegisterParameterBool("debug", "Print Changes" , false)
end sub


sub OnParameterChanged(parameterName As String)
    Vizcommunication.Map.RegisterChangedCallback(GetParameterString("prefix") &
targets[GetParameterInt("my_target")])
end sub


sub OnSharedMemoryVariableChanged(map As SharedMemory, mapKey As String)

    if GetParameterBool("debug") then
        println 3, mapKey & " : " &map[mapKey]
    end if

    coordinates.clear()
    cStr(Vizcommunication.Map[mapKey]).Split(",", coordinates)

    this.Geometry.SetParameterDouble("top_right_x",cDbl(coordinates[0]))
    this.Geometry.SetParameterDouble("top_right_y",cDbl(coordinates[1]))
    this.Geometry.SetParameterDouble("top_right_z",cDbl(coordinates[2]))
```

Copyright © 2026 Vizrt

```
    this.Geometry.SetParameterDouble("top_left_x",cDbl(coordinates[3]))
    this.Geometry.SetParameterDouble("top_left_y",cDbl(coordinates[4]))
    this.Geometry.SetParameterDouble("top_left_z",cDbl(coordinates[5]))

    this.Geometry.SetParameterDouble("bottom_left_x",cDbl(coordinates[6]))
    this.Geometry.SetParameterDouble("bottom_left_y",cDbl(coordinates[7]))
    this.Geometry.SetParameterDouble("bottom_left_z",cDbl(coordinates[8]))

    this.Geometry.SetParameterDouble("bottom_right_x",cDbl(coordinates[9]))
    this.Geometry.SetParameterDouble("bottom_right_y",cDbl(coordinates[10]))
    this.Geometry.SetParameterDouble("bottom_right_z",cDbl(coordinates[11]))
end sub
```

Copyright © 2026 Vizrt

6 Studio Manager Configuration

This section details how to configure the Studio Manager. Make sure to configure the Studio first, then the Topology.

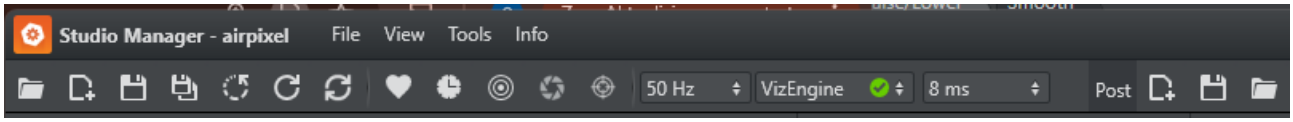
This section contains the following topics:

- Configure the Studio
- Tracked Cameras and Viz Engine
- Router Control
- Backup Configuration
- Replay Tool
- Post Data Inspector
- NDI Output Service
- Embedded NDI+ Tracking Data
- Configure Topology
- Use of Templates
- Studio Manager Camera Filter
- Engine Color Correction
- Videowall Editor
- Videowall Editor — Example Setup
- Cyclorama Editor

Copyright © 2026 Vizrt

6.1 Configure the Studio

This section details how to configure Viz Virtual Studio.



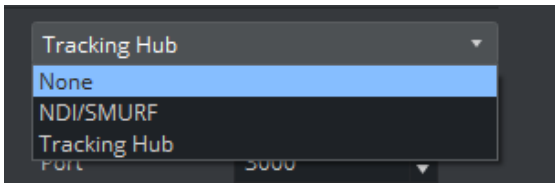
1. Click **Open** to select a previously saved studio configuration, or **New Configuration** to start an empty one.
   Save As stores the configuration you are working on under a new name.
   Set the **Frequency** frame rate. Available options are:
   - 50 Hz
   - 60 Hz
   - 59.94 Hz
   - 29.97 Hz
   - 30 Hz
   - 25 Hz
   - 24 Hz

   Set the **Synchronization** base:
   - **Free Run:** Does not synchronize. Tracking Hub runs using its own time base, corresponding with the configured frequency. This option should not be used in production environments.
   - **AV-Card:** Synchronizes using an installed Plura card. Note that the PCL-PCI L card can only read timecode and cannot be used for synchronization — see Important Checklist Before Installation.
   - **VizEngine:** Synchronizes with a Viz Engine running on the same computer.
   - **NDI:** Synchronizes on an NDI stream. See Embedded NDI+ Tracking Data.
2. Go to Configure Topology.

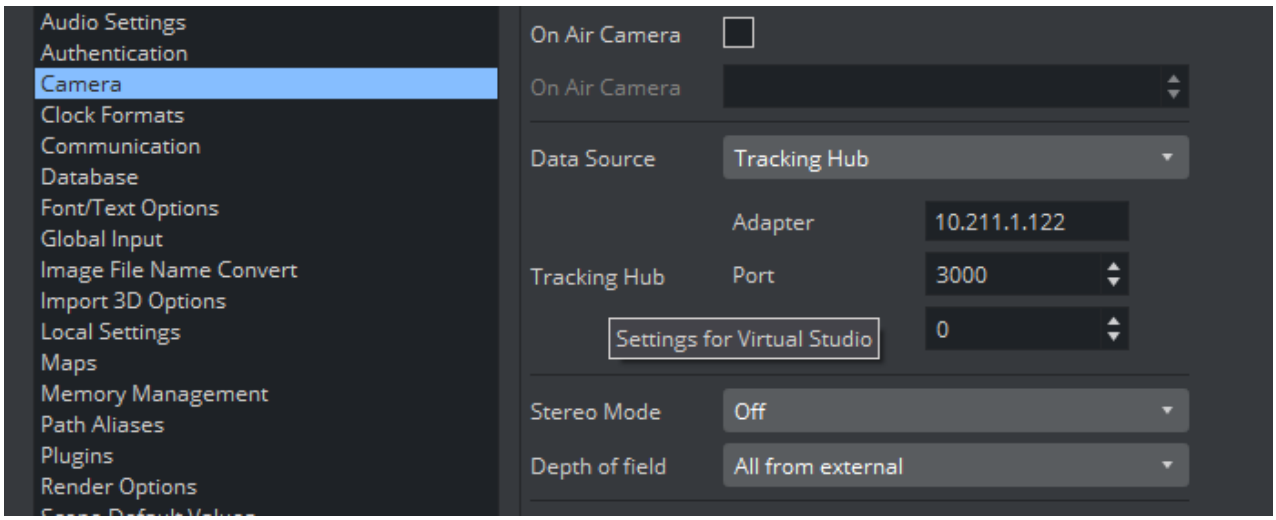
Copyright © 2026 Vizrt

6.2 Tracked Cameras and Viz Engine

Viz Engine needs to be correctly configured to receive tracking data from the Tracking Hub. For the correct configuration open the engine configuration and select the camera section. In “Data Source” select “Tracking Hub” in case you use UDP for tracking data transport. In case of NDI select “NDI/SMURF”.



When your machine comes with more than one network adapter you can add the adapter address and the port in the Adapter Box and Port number box.



IMPORTANT! The Viz Engine must be restarted after changing the configuration.

Copyright © 2026 Vizrt

6.3 Router Control

Camera cuts can be controlled from within the Studio Manager by using the **Router Control**, accessible from the **Tools** menu. The Router Control can control camera cuts by using presets, or manually. By adding presets for the most commonly used inputs and outputs, this interface allows for quick and easy camera cuts on demand. When Router Control is in use, an additional configuration file is used. This file is located in *%ProgramData%\vizrt\VizTH\Cfg\<Studio config name>*, and the file is named *XML_TH_RouterCfg.xml*. At any given time, configured connection or preset parameters can be deleted independently from the overall router control configuration by clicking the corresponding trash can icon.

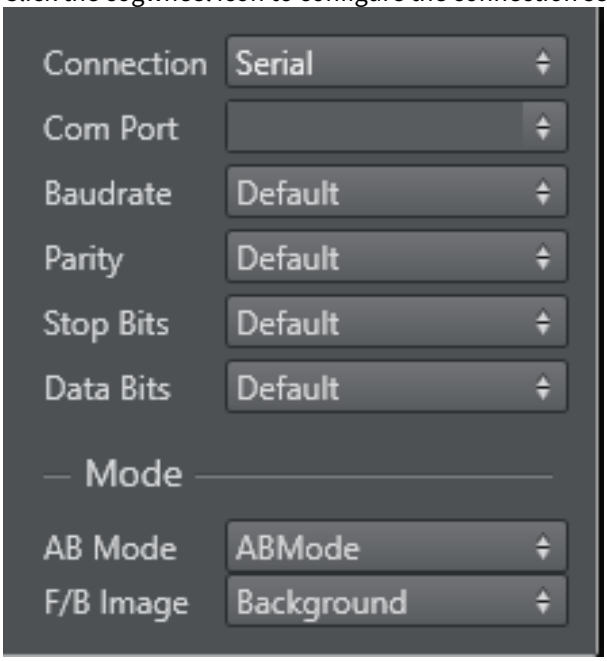
6.3.1 To Add a Router Control Preset

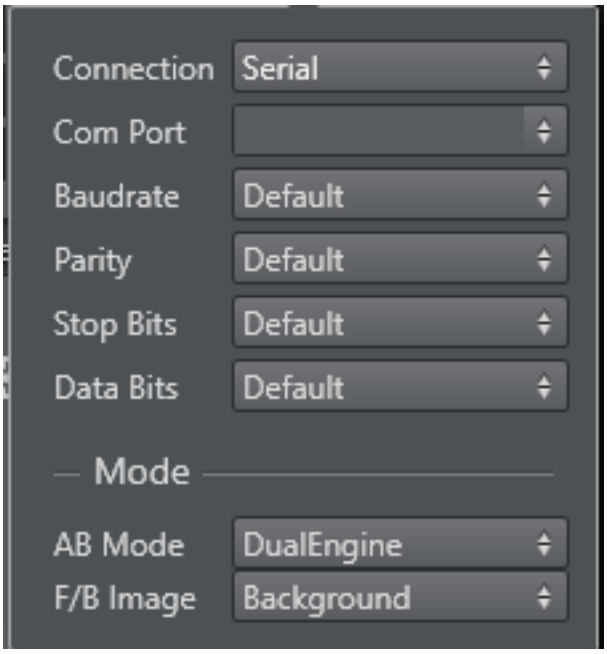


1. Open the **Tools** menu and select **Router Control**.
2. In the dialogue window that opens, select a **Model** and enter a **Name** for the router, then click **Add**. The selected model defines the communication protocol which is used, and is not vendor dependent. The available protocols are:
    - Black Magic (network)
    - Nevion MRP (network)
    - Nevion NCB (serial)

Copyright © 2026 Vizrt

3. Click the plus icon (**+**) in the upper right corner to add a router (1). If a router has been added to the system previously, it can be reused by selecting it from the **Router** drop-down list.
4. Click the cogwheel icon to configure the connection settings and **AB** mode for the router (2).



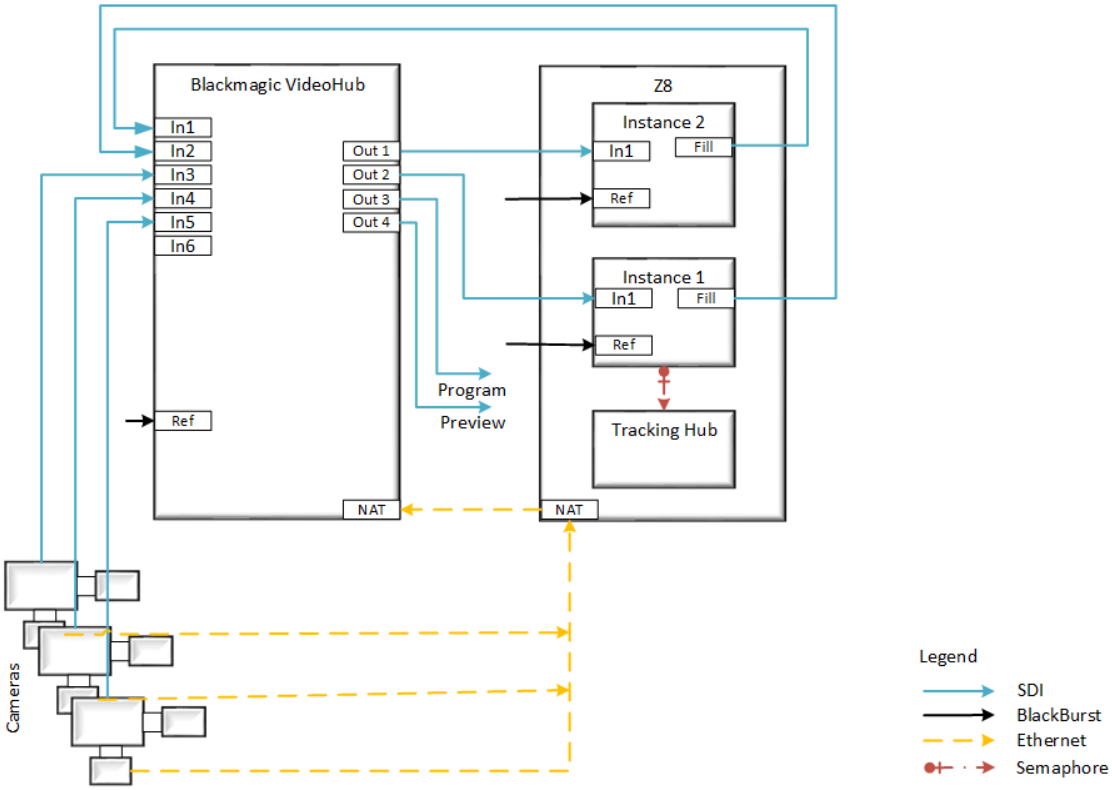


- **Connection:** Select the connection protocol. Available options are **Serial**, **UDP** and **TCP**. For serial connections, the **Com Port** needs to be set in all cases. The other parameters may be left to their **default** settings, or specified based on the requirements of the specific hardware connected. Please refer to the vendor documentation for connection details.
- **No Mode selection:** If no mode is selected, Tracking Hub changes the switch points on the router with no internal logic. The following diagram shows a "one machine with two Engine instances and

Copyright © 2026 Vizrt

internal keyer" solution which implements a program / preview setup. The two instances hold the same scene. When switching a camera is intended, first the camera on the engine and the video going into the preview engine is switched. When all signals are stable and the camera is switched, the roles of the Engines are modified by changing the signals on Out 3 and Out 4. Therefore, the preview and program role of the Engines change with every switch.

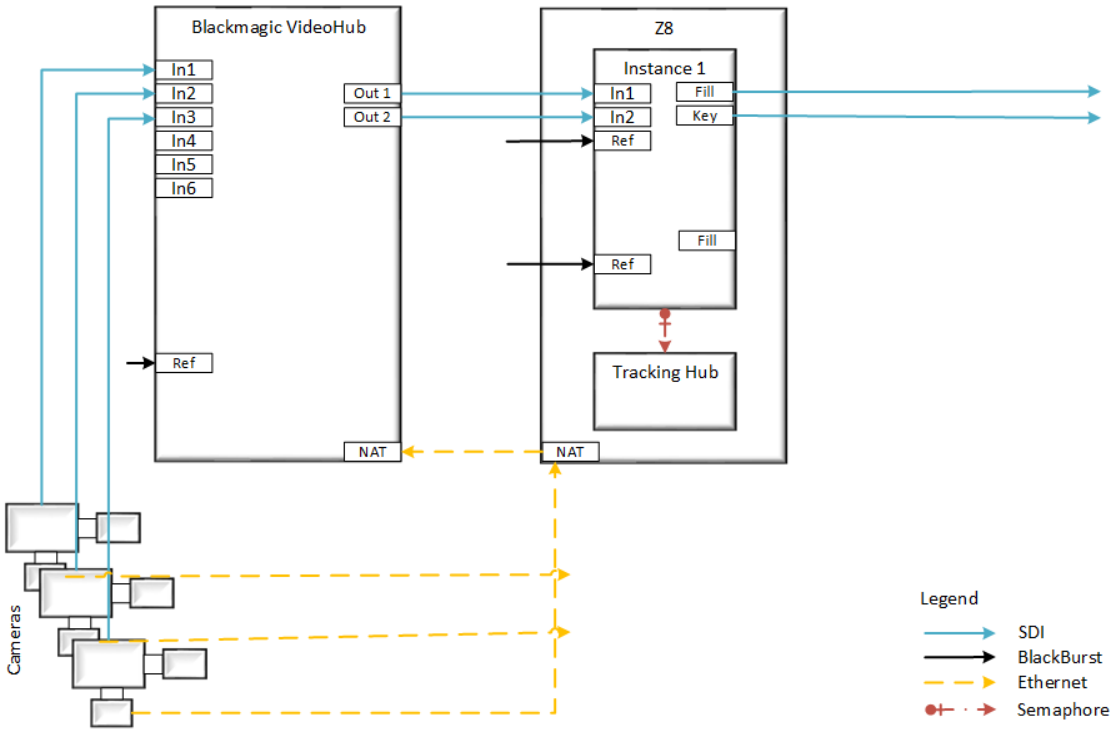
Sync Switching 1 Machine 2 Viz instances with internal chroma keyer



- **AB Mode:** AB mode is used to deal with situations with unreliable router switching times. When running in AB mode, two SDI cables are connected from the router to the Viz Engine inputs. Upon switching, the router first switches the signal to the unused line. If the signal is valid, the command to switch the input and camera is sent to the Engine. This way, the output from Viz Engine is not interrupted even if the router switching has a slight delay. This mode can be used, when the video is whether linear keyed from the background or chroma keyed with one of the internal Chroma Keyers of Viz.

Copyright © 2026 Vizrt

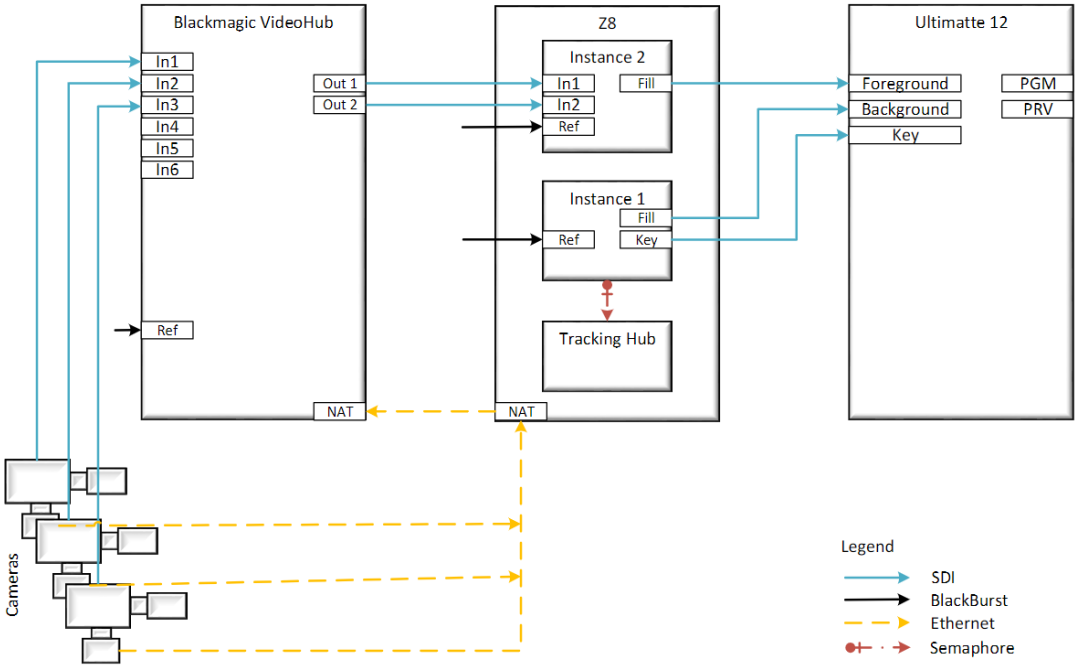
Sync Switching 1 Machine 1 / Viz instance internal keyer / 1 router



- **Dual Engine**: Dual Engine mode is used, when **ONE** and only **ONE** physical Engine machine should handle multiple cameras with a switcher. In the last few years, it became more difficult to find cheap routers which are able to switch with predictable timing. The method described here makes it possible to use the already implemented A/B mode in Tracking Hub with an external keyer, like the Ultimatte 12. The goal is to create a low cost virtual set with an arbitrary number of cameras with a one machine solution.
  On this machine (preferably a Z8 with an x.mio3 and two graphics adapters) a dual Engine is installed with a Tracking Hub. Let's also assume, that a BlackMagic Video Hub (or any other cheap router) and an Ultimatte 12 are present. In this scenario, the Video Hub is connected via Ethernet to the VSS machine. In Tracking Hub, a new switching method is added to the existing A/B mode. In this mode, the user can enter the IP and port of the first and second instance of the Engines. The router is managed by the selected protocol driver.
  When selecting this mode, the following steps occur:
  Tracking Hub takes over instance two and generates a basic scene with two inputs by using the command interface. Using this method ensures that instance two, which is responsible for the synchronized switching, is in place and ready. Tracking Hub also checks if the instance is present and On Air, that all inputs are enabled and that the sync source is set and the instance is locked.
  Tracking Hub contacts the video hub and make sure that it is reacting.
  Then the user is able to enter any amount of presets in the matrix controls of Tracking Hub. Switching can, like before, happen by GPI, commands or any future interface.
  In case of switching the following occurs:

Copyright © 2026 Vizrt

Status: Tracking Hub knows the actual status of the active output of instance two. There is always an active and inactive input present on instance two.
When a new camera is activated by a preset, Tracking Hub commands the router to switch this video feed to the inactive input of instance two. After that Tracking Hub waits a defined (by user) timespan, until the signal is stable switched and valid on the Matrox input. If necessary, Tracking Hub can check if everything is OK at the Engine by command interface, and no error messages are coming in.
After that period, Tracking Hub triggers a synchronized command to instance one and instance two (with the necessary delays) to switch camera on one and video on instance two.
This workflow can then be repeated an unlimited number of times.



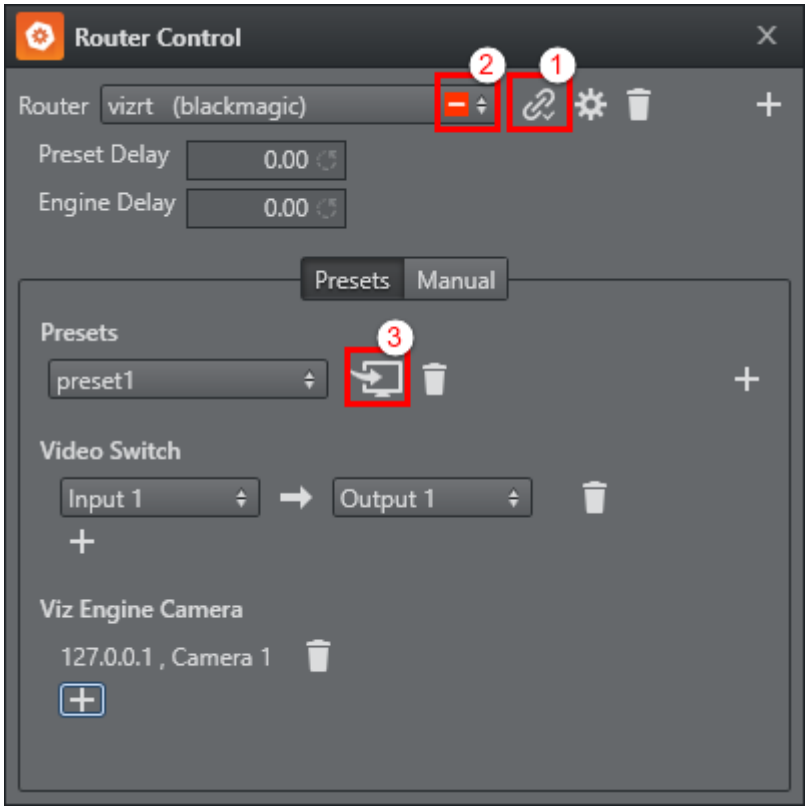
5. If required, a delay can be added in the **Preset Delay** or **Engine Delay** fields.
6. Next, click the plus icon (**+**) to the right in the **Preset** section to add a preset (3). Enter a **Name** and click **Add**.

Copyright © 2026 Vizrt



7. In the **Video Switch** section, select the **Input** and **Output** on the router from the left and right drop-down lists, respectively.
8. In the **Viz Engine Camera** section, select which Engine and camera to switch to.

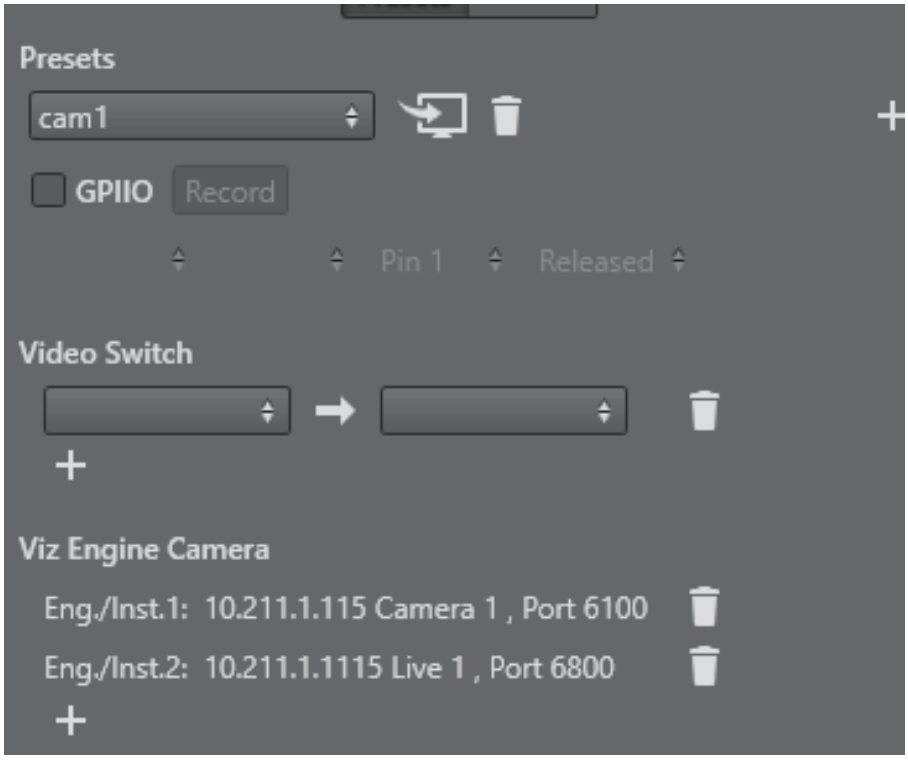
A router connection with a preset has now been configured, and the Router Control window should look like this:



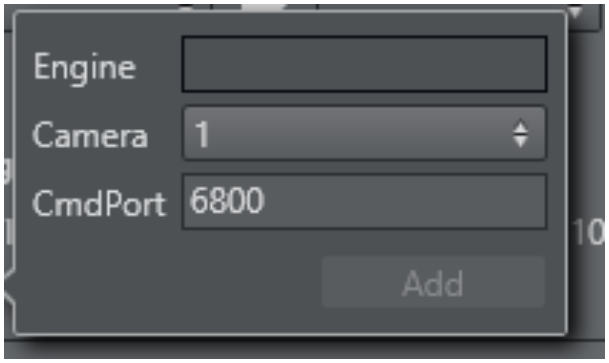
Click the **Connect** button (1) to connect to the configured router. If there is a connection problem, this is indicated by a red warning box in the **Router** drop-down list (2). To issue the take command to the router and Engine, click the **Take** button (3).

When using *DualEngine* mode:

Copyright © 2026 Vizrt



The first Engine is the Renderer, and the Engine camera is switched on this instance. The second Engine is the Video instance, and the video is switched on that. In the picture above you can see, that Engine one is Titled as *Camera Nr.* while Engine two is marked as *Live Nr.*. In Dual Engine mode, it only makes sense to configure two Engines. Please note that in the Engine instance settings, the field *Port* has been added!

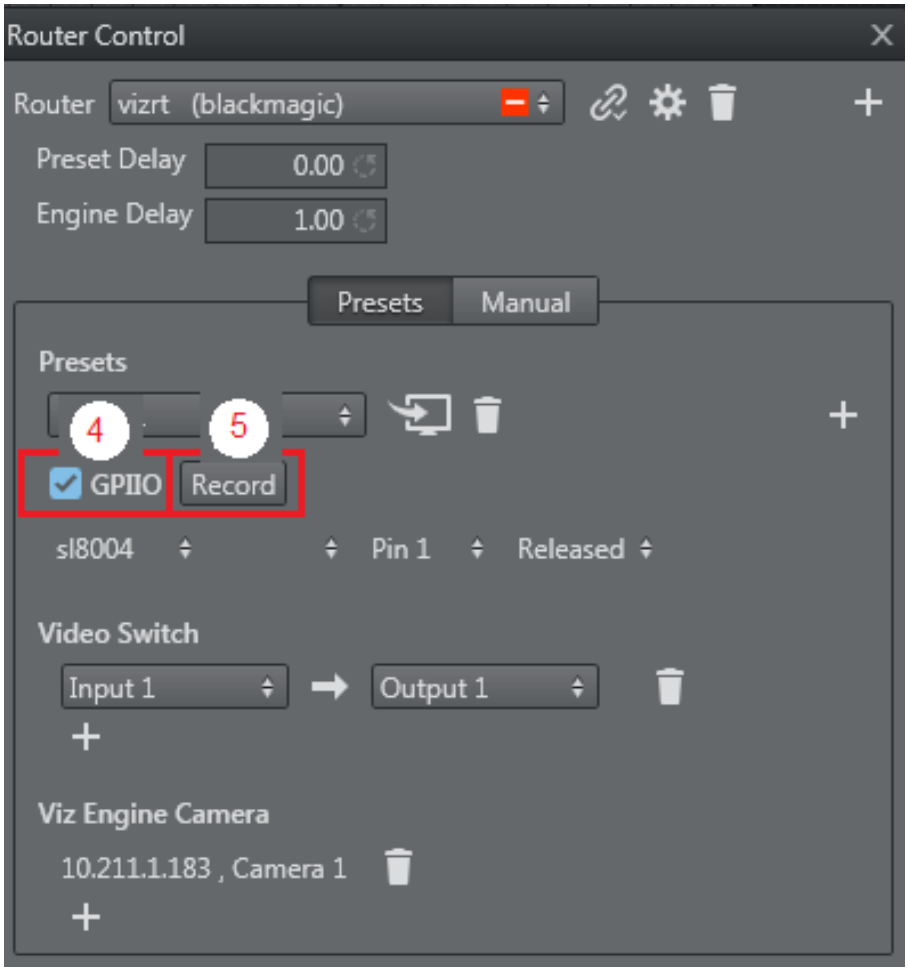


By default the port for instance one is: `6100` and the port for instance two is: `6800` . But this settings can differ from installation to installation. Please check the Engine configuration for the correct port settings.

6.3.2 GPI IO Device Switching

In addition to the standard configuration, you can add a **GPI IO** device for switching (4). You can select this device if it was detected during startup. This detection is done automatically.

Copyright © 2026 Vizrt



To configure the device, you have two options:

- **Manual:** Select the device, the port (not available for all devices), and the pin. In addition, you can select if the switching should react on pressing or releasing the button.
- **Automatic:** Click **Record** (5) and then select the button on the GPI IO device. The button is recorded and connected to the actual preset.

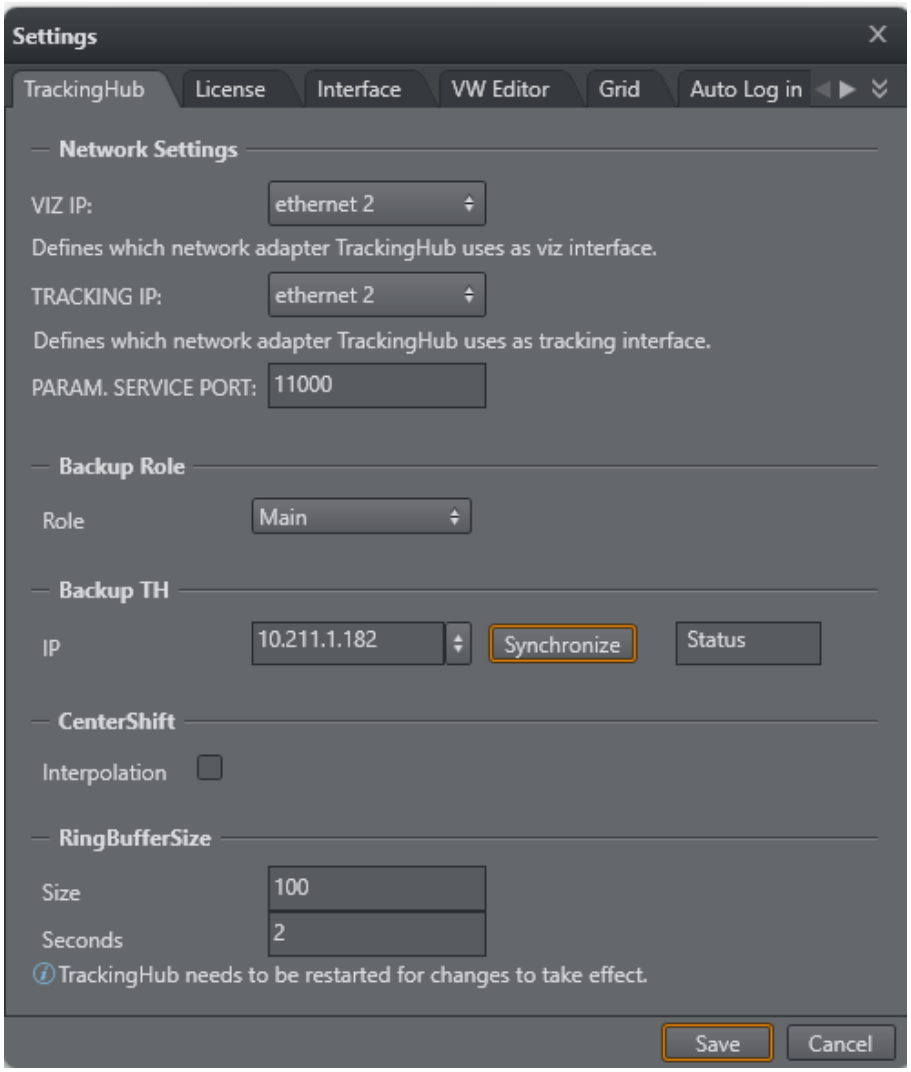
Copyright © 2026 Vizrt

6.4 Backup Configuration

In the case of a backup operation, both Tracking Hubs send tracking data to the Viz Engine(s). To prevent the continuity warning message appearing repeatedly on the Viz Engine console, set the following configuration flag in the Viz Engine configuration file:

```
trackinghub_warning_level = 1
```

You must configure the **Network settings** on both Tracking Hubs (see Start Viz Virtual Studio). Go to **Tools > Settings** to set up a backup Tracking Hub.



Click the drop down menu to open a selection of available Tracking Hubs. You must select the respective other for both Tracking Hubs. For example:

Copyright © 2026 Vizrt

TH1 has IP: 10.211.1.153	TH2 has IP: 10.211.1.183
Backup TH: 10.211.1.183	Backup TH: 10.211.1.153

Click **Synchronize** to synchronize the configuration from one Tracking Hub to another.

6.4.1 Master and Slave

A backup pair is not symmetric. Each Tracking Hub carries a backup rule of its own, which is either **Master** or **Slave**, and the two must be set accordingly:

- **Master:** holds the configuration that is distributed. Synchronize is started here.
- **Slave:** receives the configuration, receives the Master's lifesign, and monitors it.

The Slave watches the Master's lifesign continuously:

- Missing for more than **one day**: the Slave sends a notification, repeated once an hour.
- Missing for more than **two days**: the Slave reconfigures itself as a standalone Tracking Hub. Its backup rule is set to none, the backup address is cleared, the change is written to `BaseConfig.xml`, and the license is released and requested again for standalone operation.

The standalone switch is permanent. When the Master comes back, the pair has to be configured again on both machines.

6.4.2 What Synchronize transfers

Synchronize is started on the Master and sends the following to the Slave, in this order:

- The Cyc file
- The lens files
- The XML tracking description, if one is configured
- The LUT files, if any are in use
- The video wall file
- The router configuration, if one is configured
- The current state
- The studio configuration file

Close the Lens File Editor before synchronizing. While it is open, the command is refused with a message.

6.4.3 Sending to Viz Engine while a backup is configured

Without a backup partner, Tracking Hub sends the data of every rig. As soon as a backup partner is configured, both hubs restrict their output: a rig is sent only while its tracking is good. The hub that currently holds valid data therefore supplies the Engine.

Both hubs still address the same Engines, which is why the continuity warning has to be suppressed with `trackinghub_warning_level = 1`.

Copyright © 2026 Vizrt

Important: When using serial interfaces, the COM ports might be different on these PCs. Always inspect the selected COM ports after synchronization, and if necessary, modify them.

Copyright © 2026 Vizrt

6.5 Replay Tool

- Prerequisites
  - Software
  - Video Hardware
- Preparations
  - Viz Engine Configuration
  - Service Host Configuration
  - Channel Recorder Client
  - Tracking Hub Configuration
    - Service Host Section
    - Clips Section
  - Replay Tool
    - Replay Group
      - Clip Section
      - Play Section
      - Feedback Section
      - Replay Live Section
    - Record Group

The Replay Tool can be used to record time code stamped clips with the help of Service Host and a Channel Recorder plug-in. Any time code which is readable by Viz Engine can be used. Most cameras already insert at least one readable time code.

To use the tool, some prerequisites must be met and some configuration settings modified. In the following sections, a step-by-step description of the preparations are given.

6.5.1 Prerequisites

Software

The Replay Tool was tested with the following software versions:

- Viz Engine 4.3
- Viz Artist 4.3
- Service Host 2.0
- Tracking Hub 1.5
- Studio Manager 1.5

Please install the packages above in any order. If older versions are already installed, uninstall them first.

Video Hardware

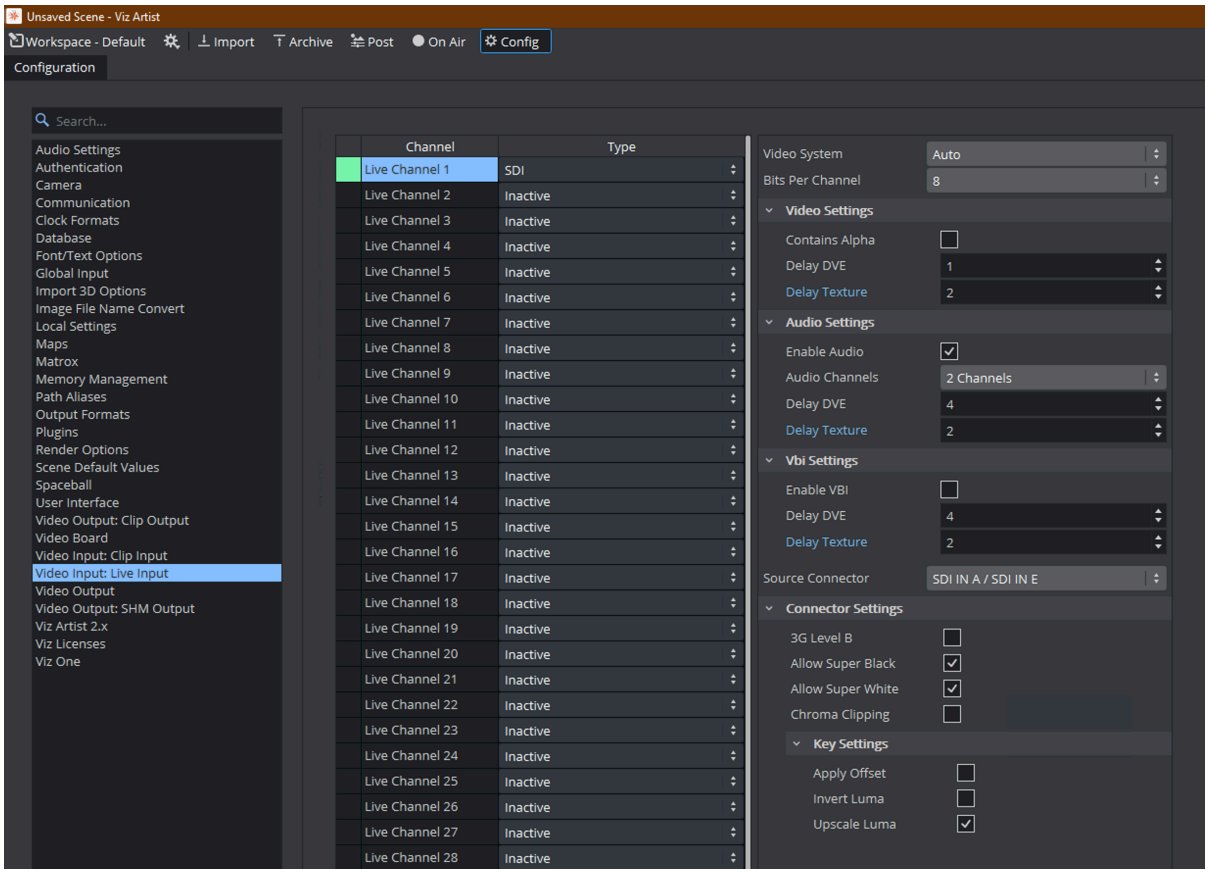
- Matrox XMIO 3/5
- Matrox DSXLE 4

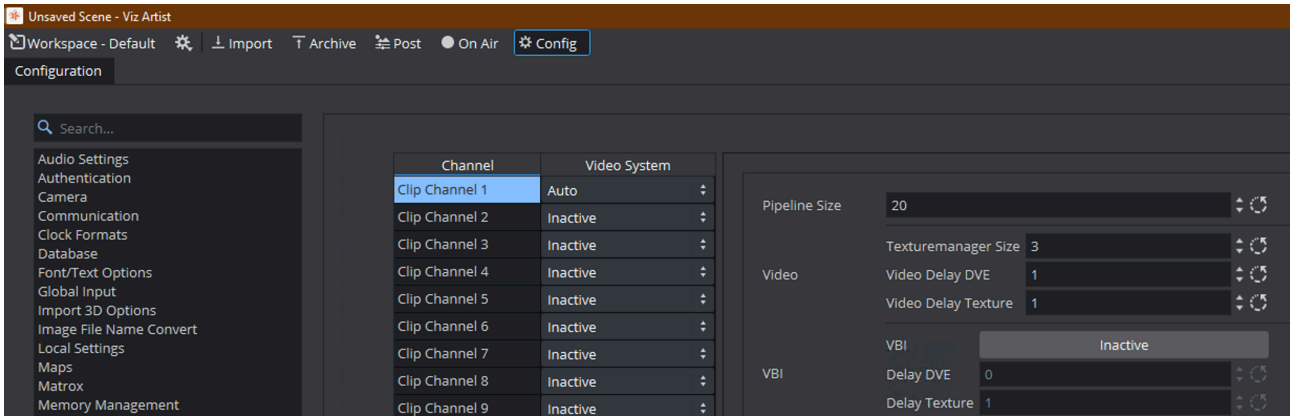
Copyright © 2026 Vizrt

6.5.2 Preparations

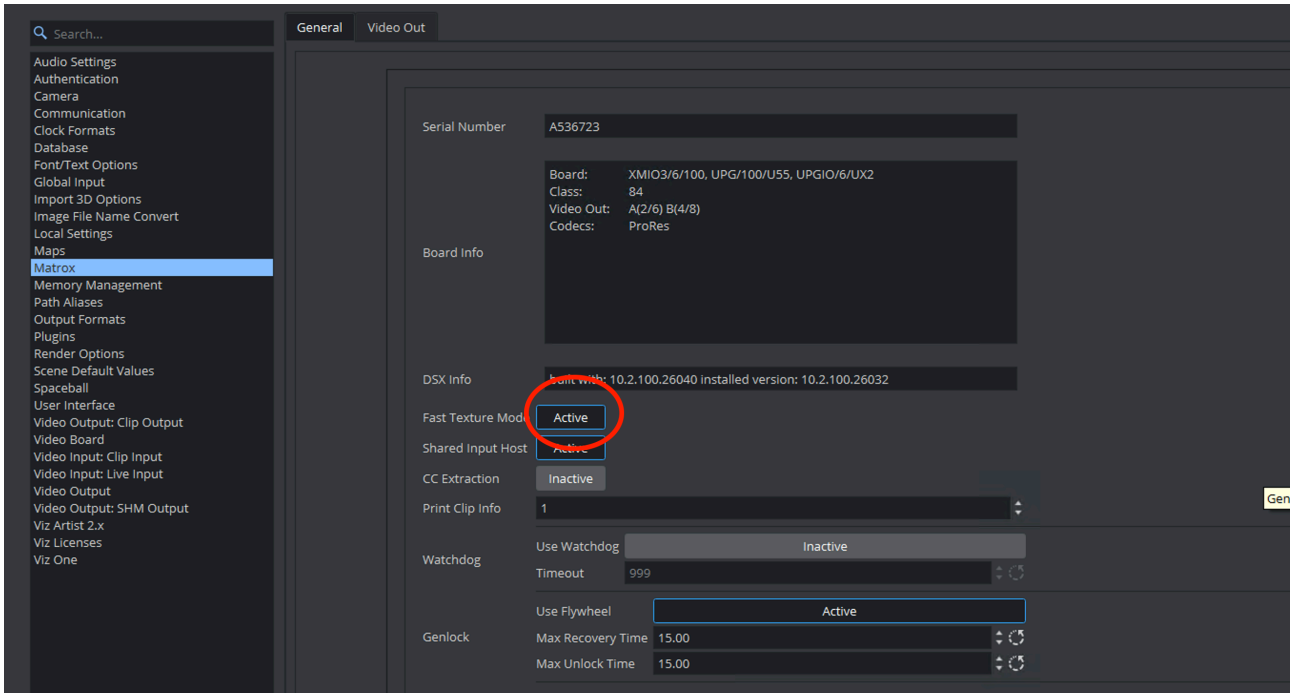
Viz Engine Configuration

After initial installation of Viz Engine, launch the Viz Engine configuration and activate Clip Channel 1 and Live Video In 1.





Make sure that in the Matrox section *Fast Texture Mode* is activated. This setting is mandatory for Virtual Set installations.



After these first steps close Viz and open the config file. Please make sure, that in the section "MATROX_CONFIG" *Matrox.FastTextureMode* is set to `1` and that *Matrox.InputHost* is set to `1`.

```
############################
SECTION MATROX_CONFIG
############################
##
## matrox devices common
##
Matrox.Devices = A536723
## Enable fast texture input mode.
## ATTENTION: in this mode DVE will not work.
#* Matrox.FastTextureMode: Default=0
Matrox.FastTextureMode = 1
## Set this to true, if the physical inputs on this engine are to be used as shared
resources in additional engine instances.
#* Matrox.InputHost: Default=0
Matrox.InputHost = 1
```

After that, go to the *Clip in channel: 1* section.

```
##
## Clip in channel: 1
```

Copyright © 2026 Vizrt

```
##
```

Make sure that the selected clip player is set to `0: Matrox`. The VML clip player does not work with time code!

```
## Default clip player to be used for this channel. This is only relevant when Matrox
board is used and io_mode is configured to V4. 0=Matrox 1=VML
#* ClipIn1.Player: Default=0 Min=0 Max=1
ClipIn1.Player = 0
```

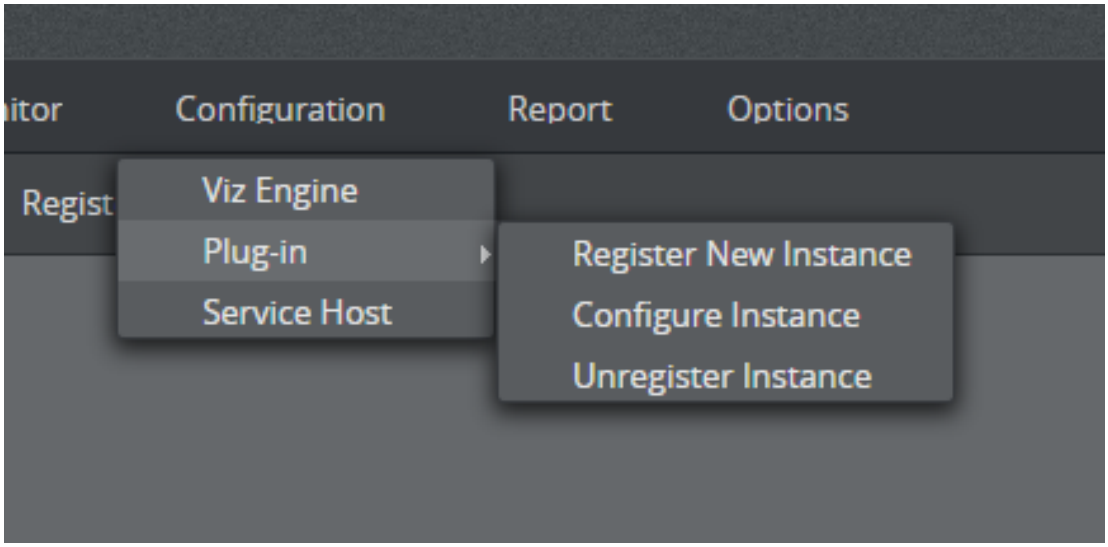
Service Host Configuration

If there is an older version of the Service Host installed, it must be uninstalled first. It is recommended to use the default connection settings for the installation, as the ports are preconfigured in the Replay Tool.

Please refer to the Service Host Administrator Guide for further information.

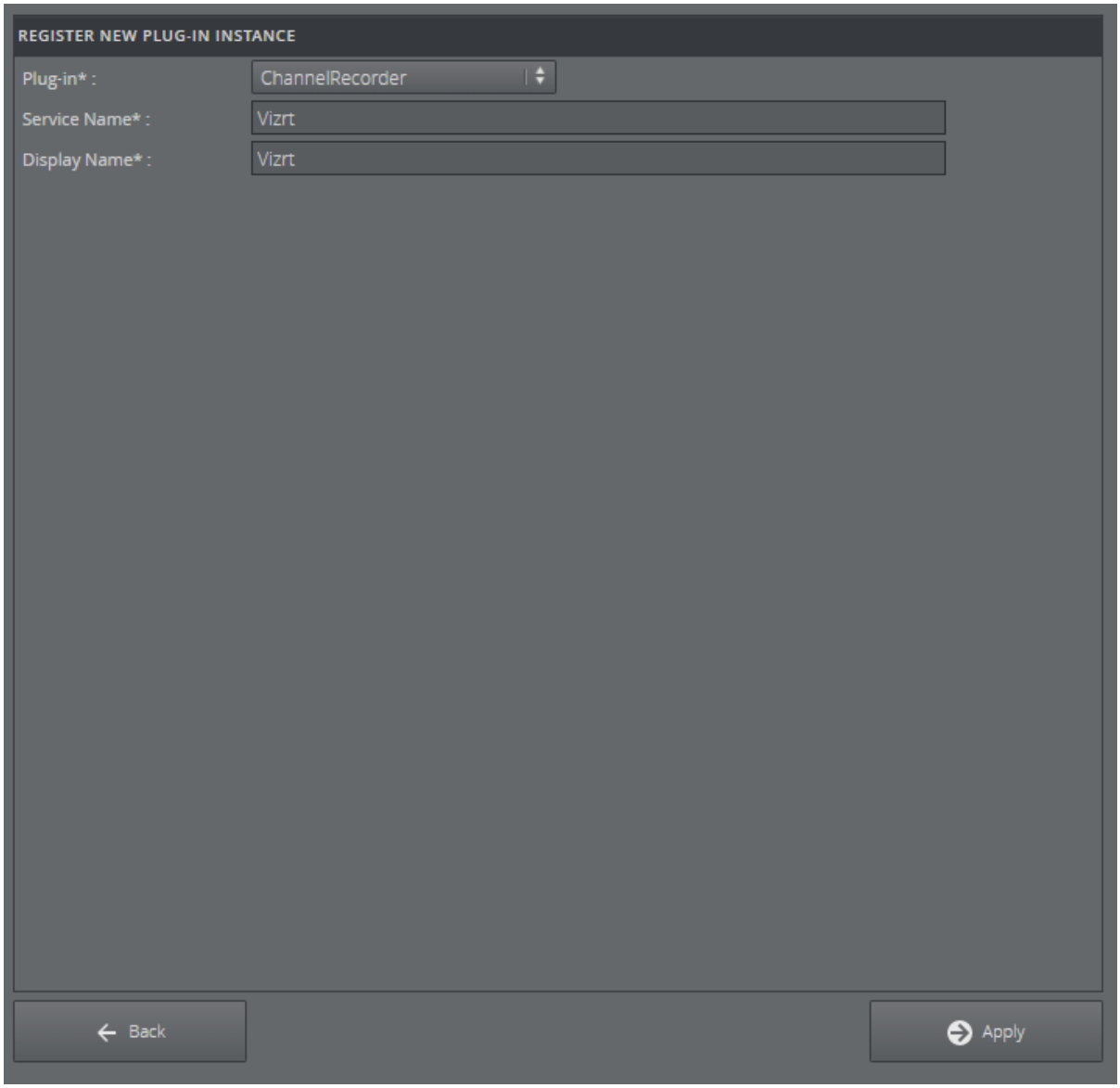
Channel Recorder Client

Channel Recorder is a plug-in for Service Host. It enables recording of video inputs at the same time as Viz is using it for rendering. A detailed documentation of all settings is given in the Service Host Administrator Guide. A brief example which is needed for the Replay Tool is provided in the following Section.



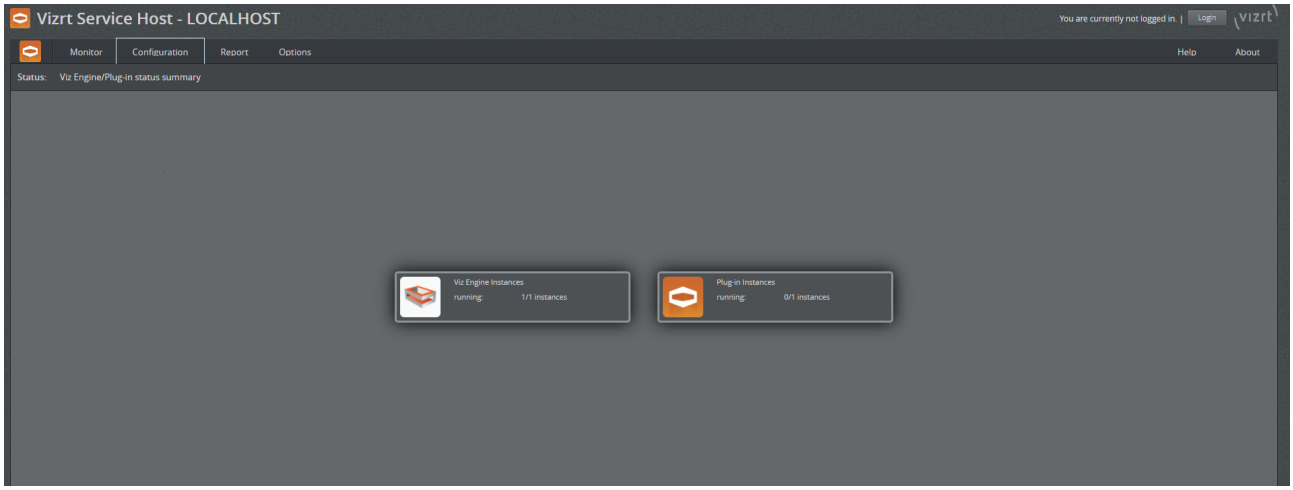
Open the Service Host UI in a web browser. Click on **Configuration > Plugin > Register New Instance**. Select *ChannelRecorder* and then click on **Apply**.

Copyright © 2026 Vizrt

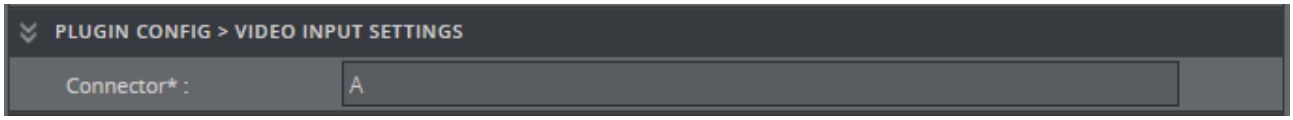


After this step, a new plug-in can be found within Service Host.

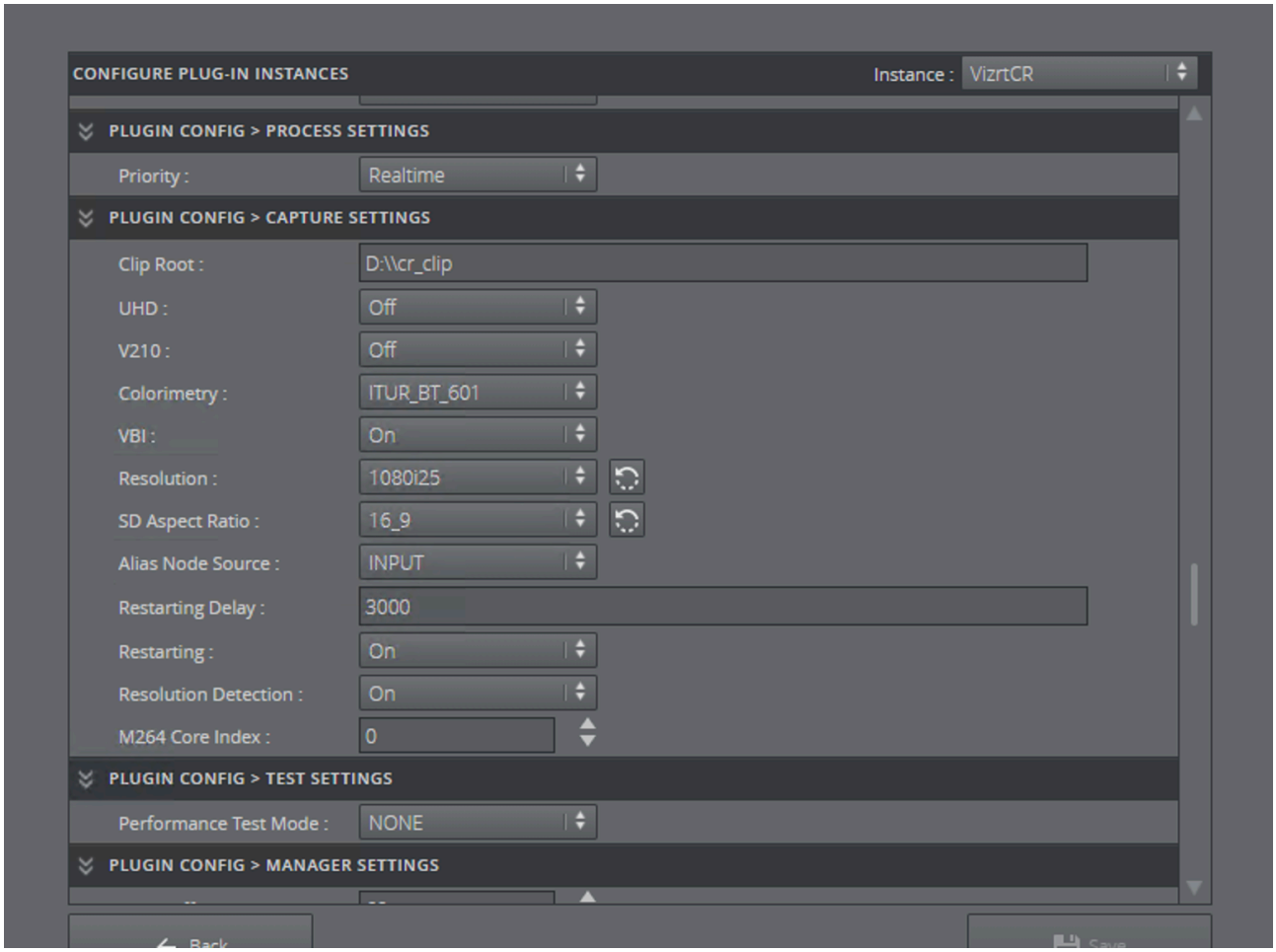
Copyright © 2026 Vizrt



The next step is configuring the plug-in. In this example, Input connector A is used for recording. The clip folder is located at *D:/cr_clip*. The plug-in stores all recorded clips in this folder. Select the video resolution and aspect ratio needed for recording record.

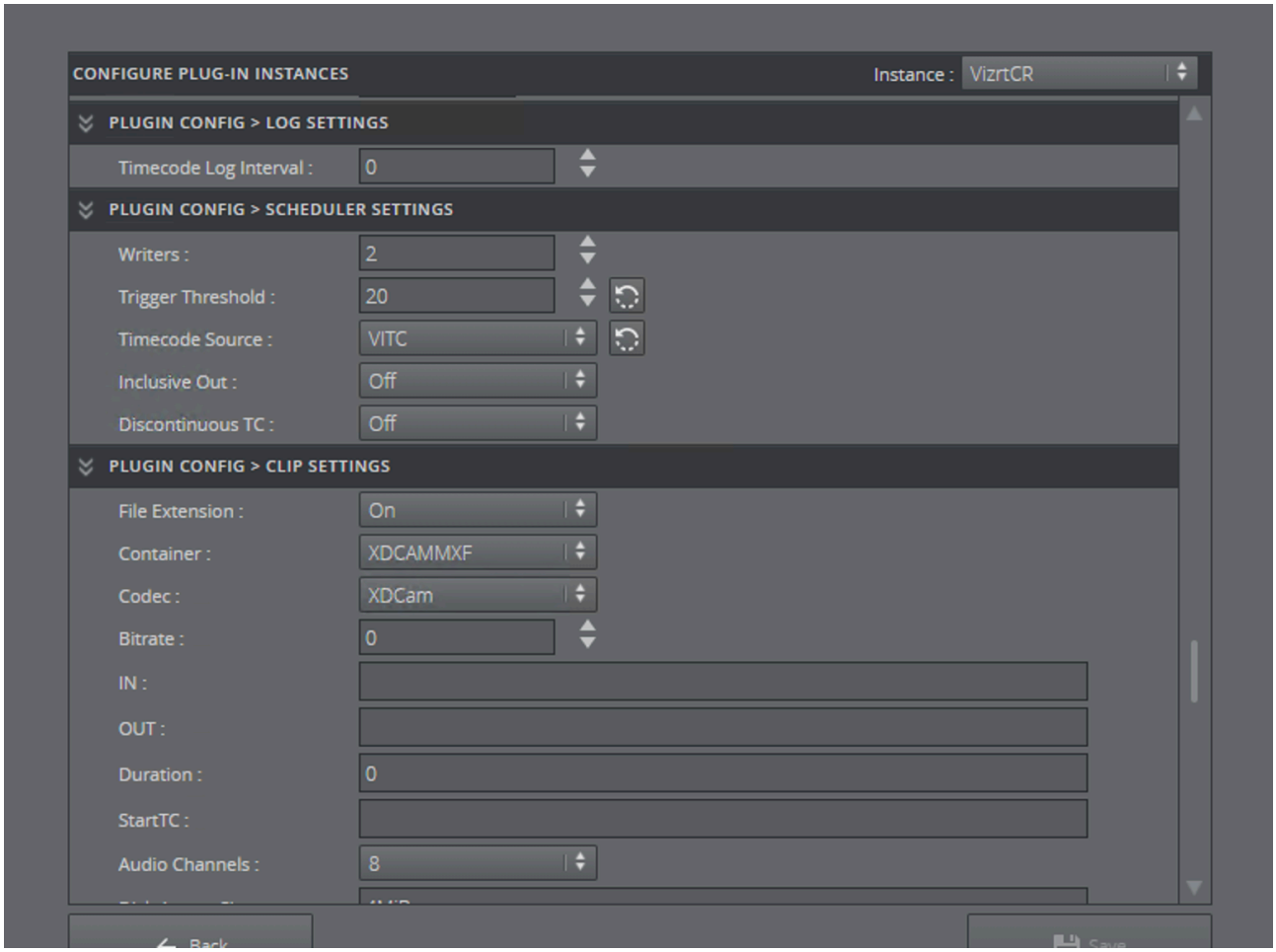


Copyright © 2026 Vizrt



Scroll down to the **Scheduler Settings** and select VITC as Timecode Source. In the clip settings the codec and container can be selected. For more information on what codec and container is usable for the selected resolution in Viz Engine, please refer to **Channel Recorder > References and Specifications > Supported Codecs** in the Service Host Administrator Guide. If recording does not start, check the log file of the Channel recorder and follow the instructions there.

Copyright © 2026 Vizrt

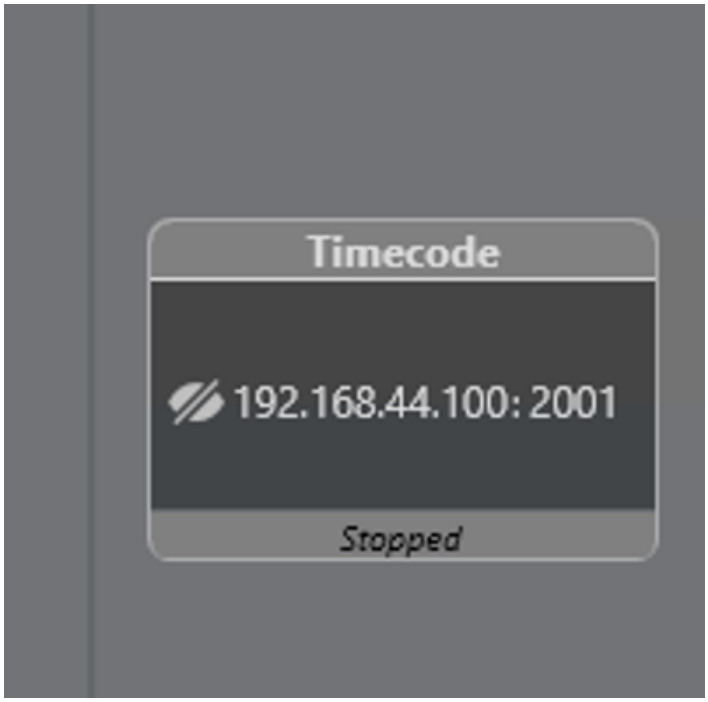


Make sure that Viz Engine is running before starting the instance. Otherwise, the service is not able to record properly and the recorded clip will be black.

Tracking Hub Configuration

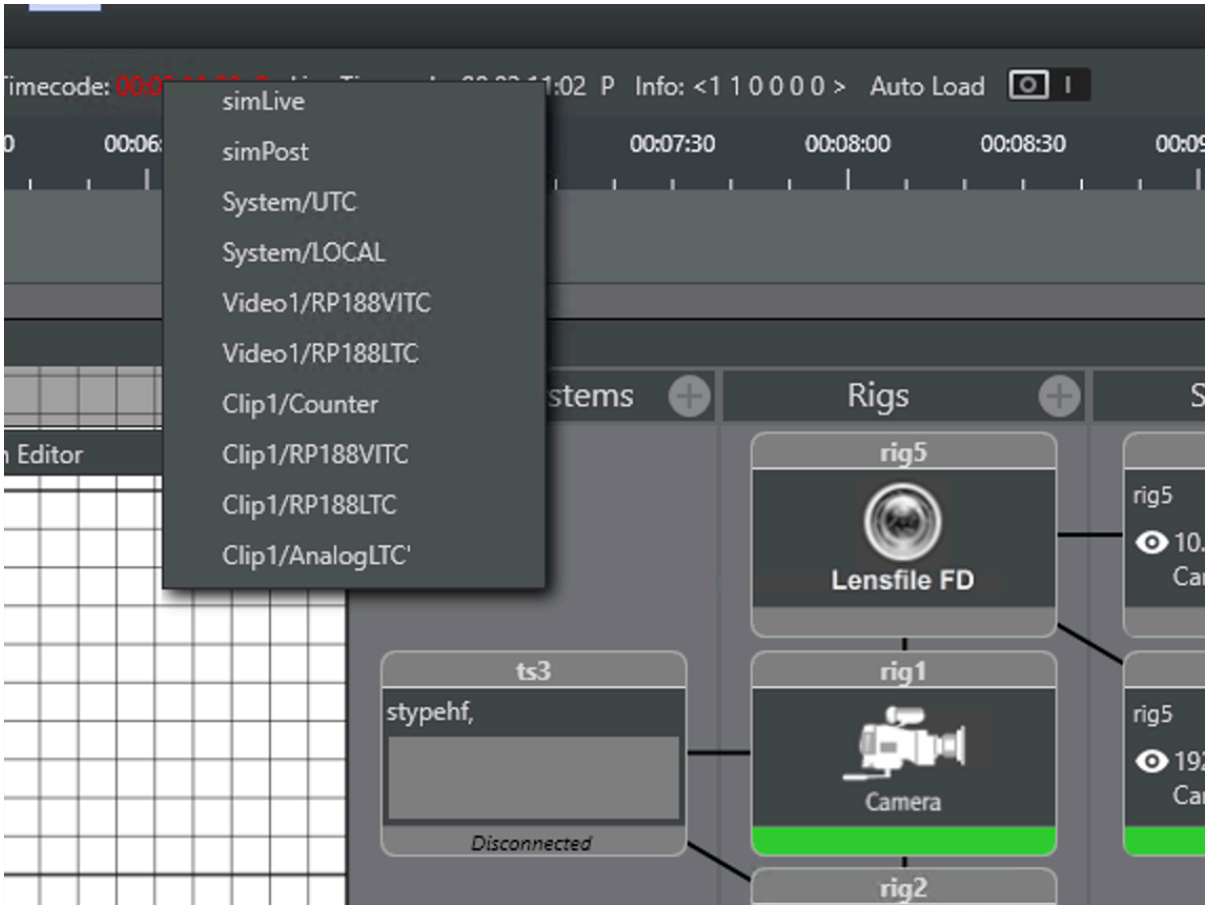
Please refer to the section Post System and configure the setup for recording. To receive time codes from the Engines, add a Timecode service and connect it to the Engine. Tracking Hub requests all time code sources from the Engine and adds them as sources in the post settings. The Engine must be On Air when the service is activated, otherwise the commands to receive the time code services fail.

Copyright © 2026 Vizrt



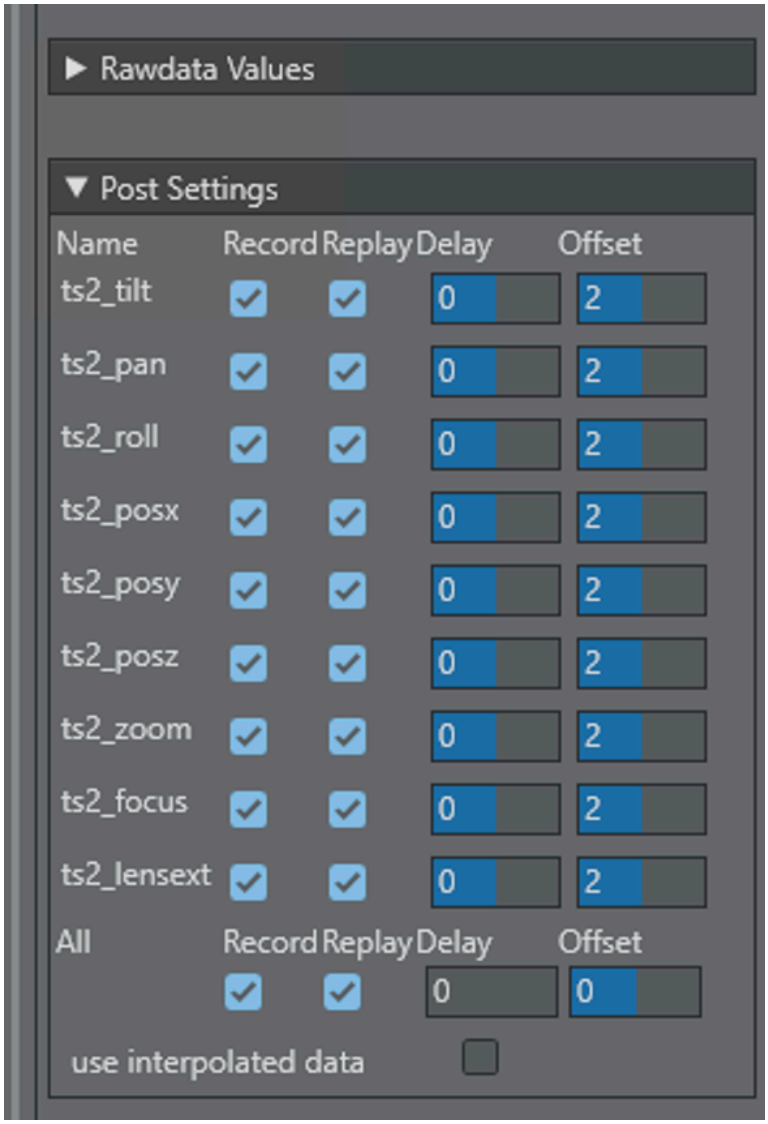
As post time code select *Clip1/RP188VITC* as live time code select *Video1/RP166VITC*. The live time code should be taken immediately.

Copyright © 2026 Vizrt

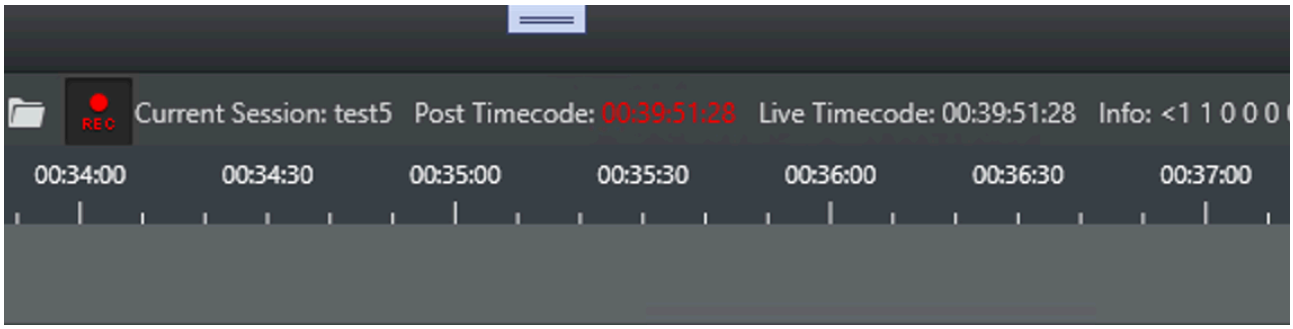


Select the parameters which should be recorded.

Copyright © 2026 Vizrt

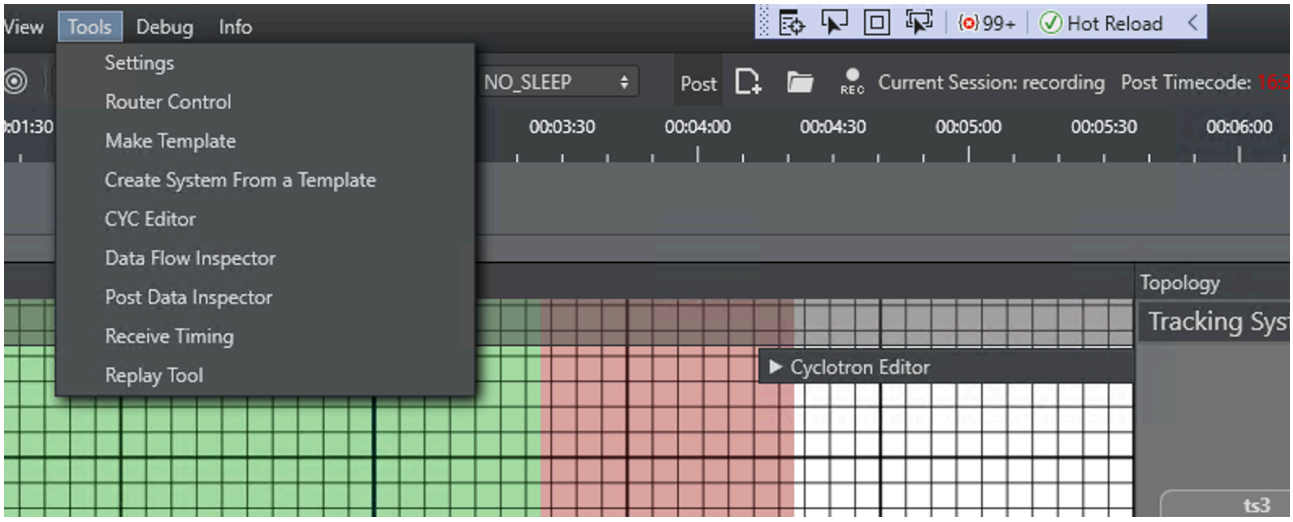


The press **Record** in the Post Tab.



After the recording session has started, open the Replay Tool. The tool is located under **Tools > Replay Tool**.

Copyright © 2026 Vizrt



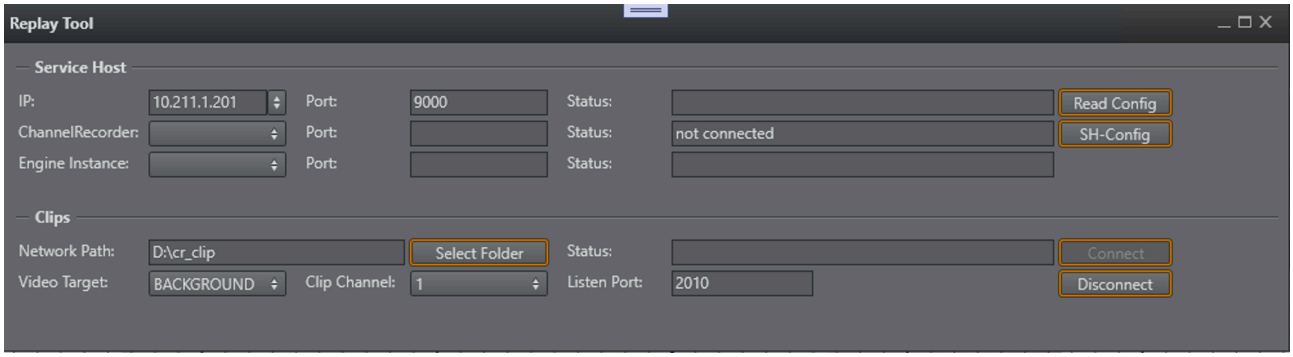
The connect window opens. The first step is to connect to the Service Host instance. Select the IP address of the Service Host and enter the configured port (default `9000`). After pressing **Connect**, a connection is established and all plug-ins installed on the Service Host are analyzed.

If the **Connect** button is not enabled, Studio Manager was not able to find a correct installed Engine on this IP address. Please check if *use trackinghub* is activated in the Engine configuration and that the correct IP address is provided.

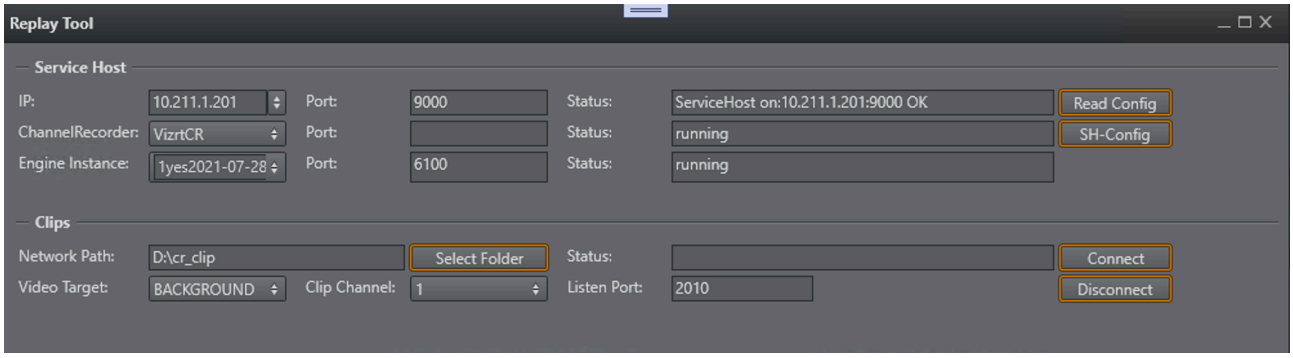
Service Host Section

Select the Channel Recorder plug-in to be used for recording from the C*hannelRecorder* drop down. The connection also reads the Engine instance out of the configuration. It is normal that the Viz Engine Instance reports *not connected* after this step. The connection is established after the final connect. It is important that an Engine instance was found in the Service Host configuration. It is also important that Viz Engine was started and the plug-in instance has been started after Viz Engine.

- **IP:** Shows the Service Hosts IP address.
- **ChannelRecorder:** Shows all channel recorder plug-ins installed on the Service Host.
- **Engine Instance:** Shows all Viz Engine instances running on the Service Host. It also checks whether Viz Engine is running.



Copyright © 2026 Vizrt



Clips Section

The next step is to connect a clip folder. If the Replay Tool runs on the same machine as the Engine, you can simply select the clip folder which was configured during Service Host and Channel Recorder installation. If the control runs on another Engine, it is necessary to map the clip folder from the Engine to the local machine, the Replay Tool is running.

To map a network drive, open Windows Explorer and right click on the *This PC* entry. Then select *Map Network drive* and browse to the folder you shared on the Engine. Please refer to the Windows manual on exact instructions for Mapping Network drives.

To find the correct mapped or local folder, click on *Select Folder*. A dialog opens where you can browse your computer. If the folder is still valid from the last session, you can now click **Connect**. If everything goes well, the configuration dialog is hidden and the Replay Tool opens the Replay tab.

The *Video Target* defines, if the clip and live video is played either in Foreground or Background. In case of a Virtual Studio with chroma keyer, select FOREGROUND. In case of Augmented Reality, use BACKGROUND.

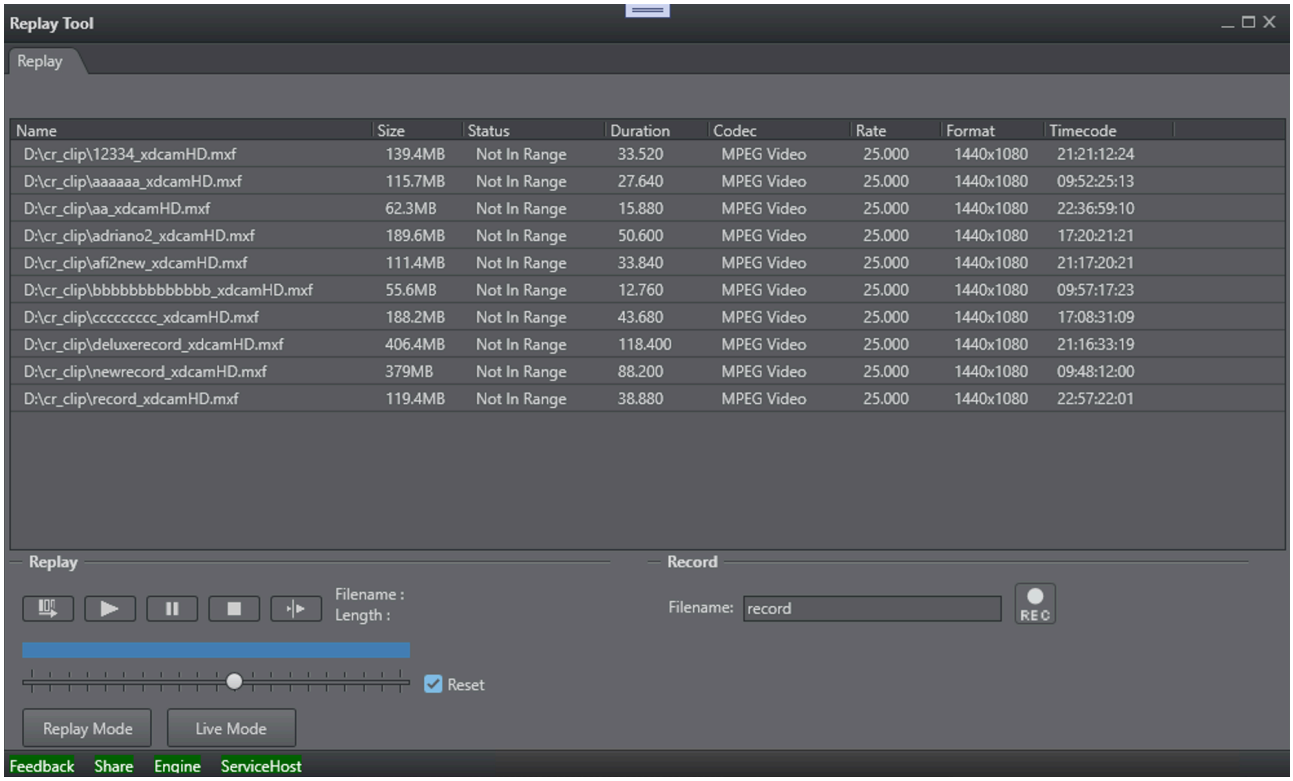
The Listen Port defines the feedback port the Replay Tool is registering its information channels. Usually, there is no need to change this value, the only exception is, when another application uses the same ports.

Press **Connect** to continue.

Replay Tool

It consists of two tabs. The Replay Tab and the Record Tab.

Copyright © 2026 Vizrt



Replay Group

Clip Section

In the clip section the tool observes the clip folder and continuously analyses changes. The analyzer is able to read timecode, format, length, duration and format info from the clips. In the case of multiple time codes in the clip, the first one is used. This time code information is used to display if tracking data for the clip is actually in memory.

Select a clip to continue.

Play Section



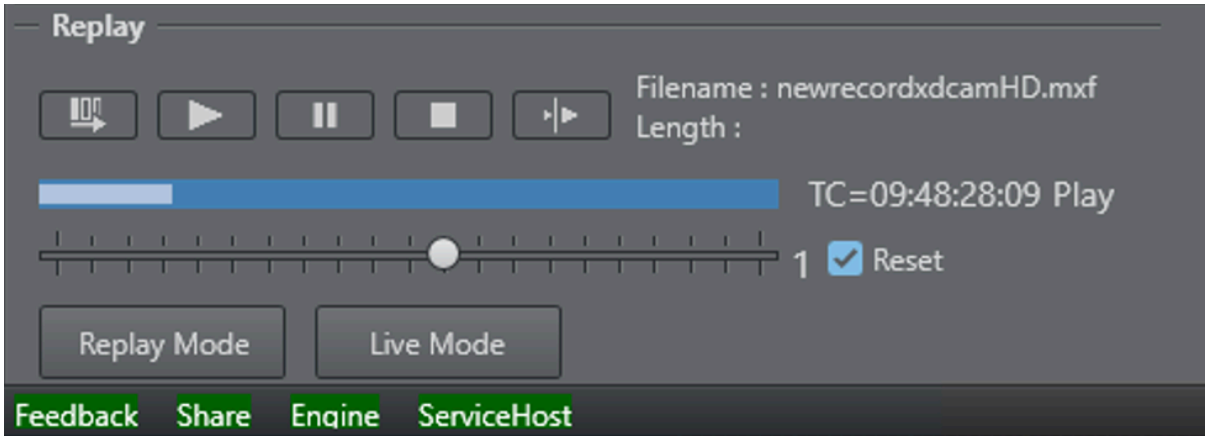
The Play section consists of five buttons to control the clip player of Viz Engine. The commands are sent directly by the Replay Tool. The IP and Command port of the Engine is read from the Service Host configuration.

- **Take:** Loads the selected clip from the clip section into Viz Engine. As soon as the clip has been loaded, its name is visible in the Feedback Section.
- **Play:** Starts the clip playback from beginning.

Copyright © 2026 Vizrt

- **Pause:** Stops replay at the current location. If you want to continue playing, press **Continue** instead of **Play**.
- **Stop:** Stops the clip and makes it disappear from screen.
- **Continue:** Resumes playback of the clip from the position pause was pressed.

Feedback Section



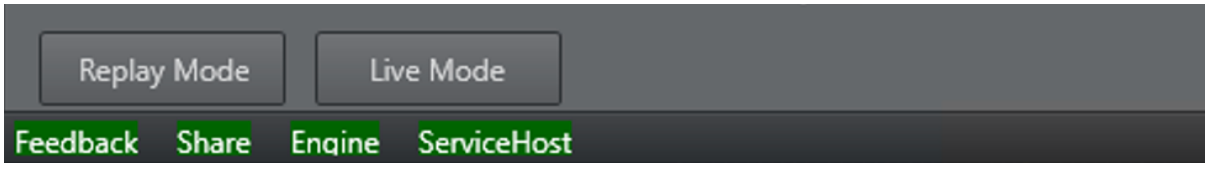
The Feedback Section reflects the clip channels feedback given from the Engine. During connect, the Replay Tool registers several feedback channels. One of them is the *clip loaded* feedback, which is triggered when a clip is ready for replay in the Engine. Therefore, *Clip Name* reflects the actual clip loaded in the Engine. If you try to load a clip with the **Take** button and the name does not appear in the *Clip Name* field, something went wrong or the clip is not playable. Please refer to the Viz Engine console for error messages.

The second feedback channel is the *Clip Position* channel. It is displayed in a progress bar and the position reflects the actual position which is played out in the Engine at this moment.

The third registered feedback channel is the actual timecode, which is played out by Viz Engine. The actual timecode is shown on the right side from the progress bar. This text field also shows the playback status of the clip.

On the bottom of the position progress bar the playback speed can be adjusted. By default the slider is set to 1 which is the default playback speed. You can drag the slider and adjust the speed from -10 up to 10. The reset button on the right side defines what happens if the user releases the slider. When **Reset** is selected, the speed returns to 1. If reset is not checked, the selected speed is kept.

Replay Live Section



The **Replay** button sets the selected clip channel as FOREGROUND or BACKGROUND depending on which source was selected in the connect dialog.

Copyright © 2026 Vizrt

The **Live** button does the same with the live input.

Record Group

The Record tab is used to start or stop the recording of a new clip. If not recording, a clipname can be entered in the filename field. The filename is then extended by the channel recorder plug-ins and used for the new clip. Pressing on the **REC** button starts recording.

This is indicated by red blinking of the record button. Pressing **REC** again stops recording.

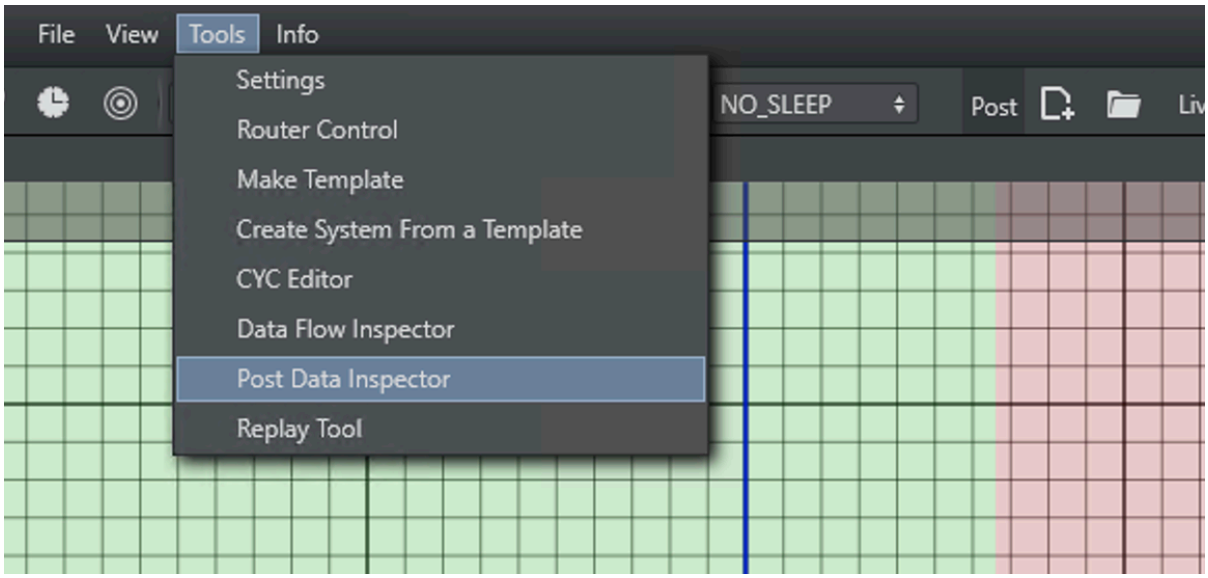




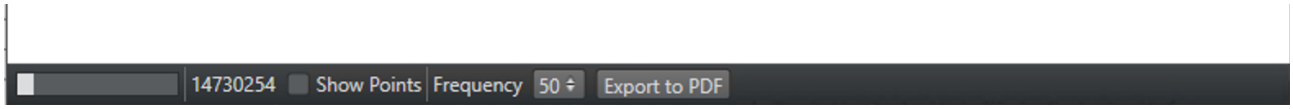
Copyright © 2026 Vizrt

6.6 Post Data Inspector

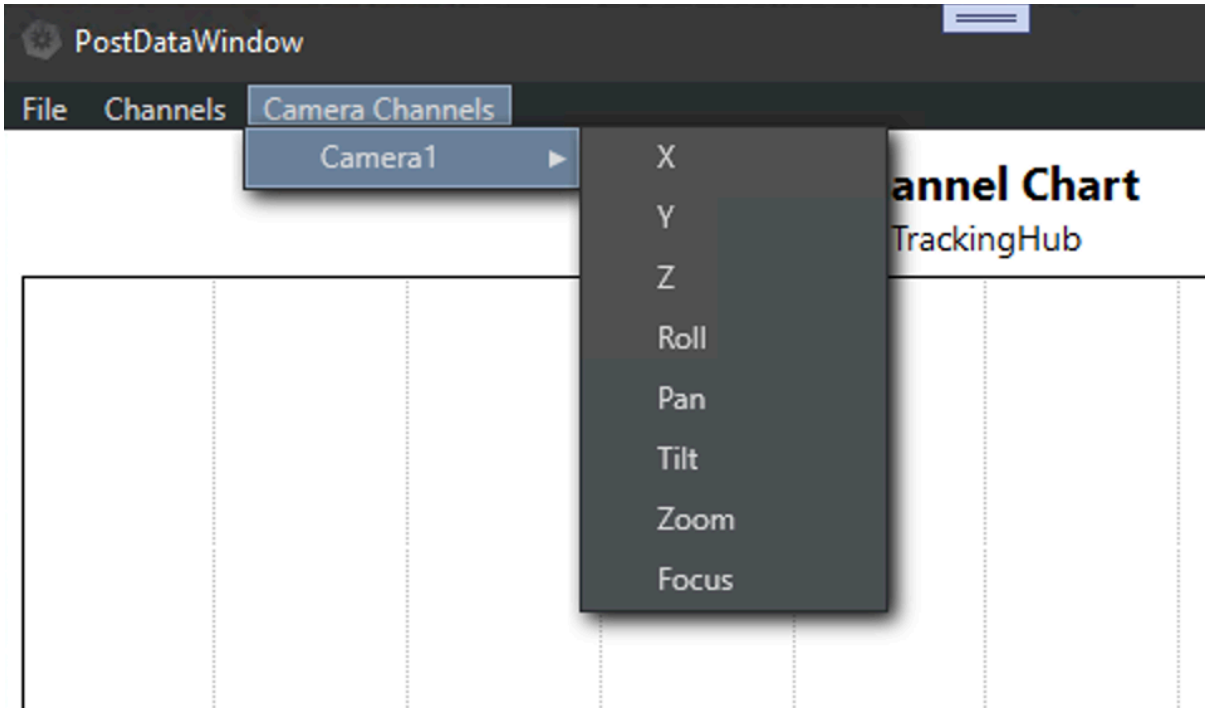
The post data inspector is a tool which can load recorded data, either from parameters or camera services, and display this data in a window.



After the window opens, a recorded file can be opened by selecting **File > Load**. A File dialog box opens and a post record session can be selected. If the recording is over several hours, loading can take quite some time. A progress bar in the status line shows the position of reading. After the file has been loaded, the Menus and Camera Channels are filled with the available parameters.

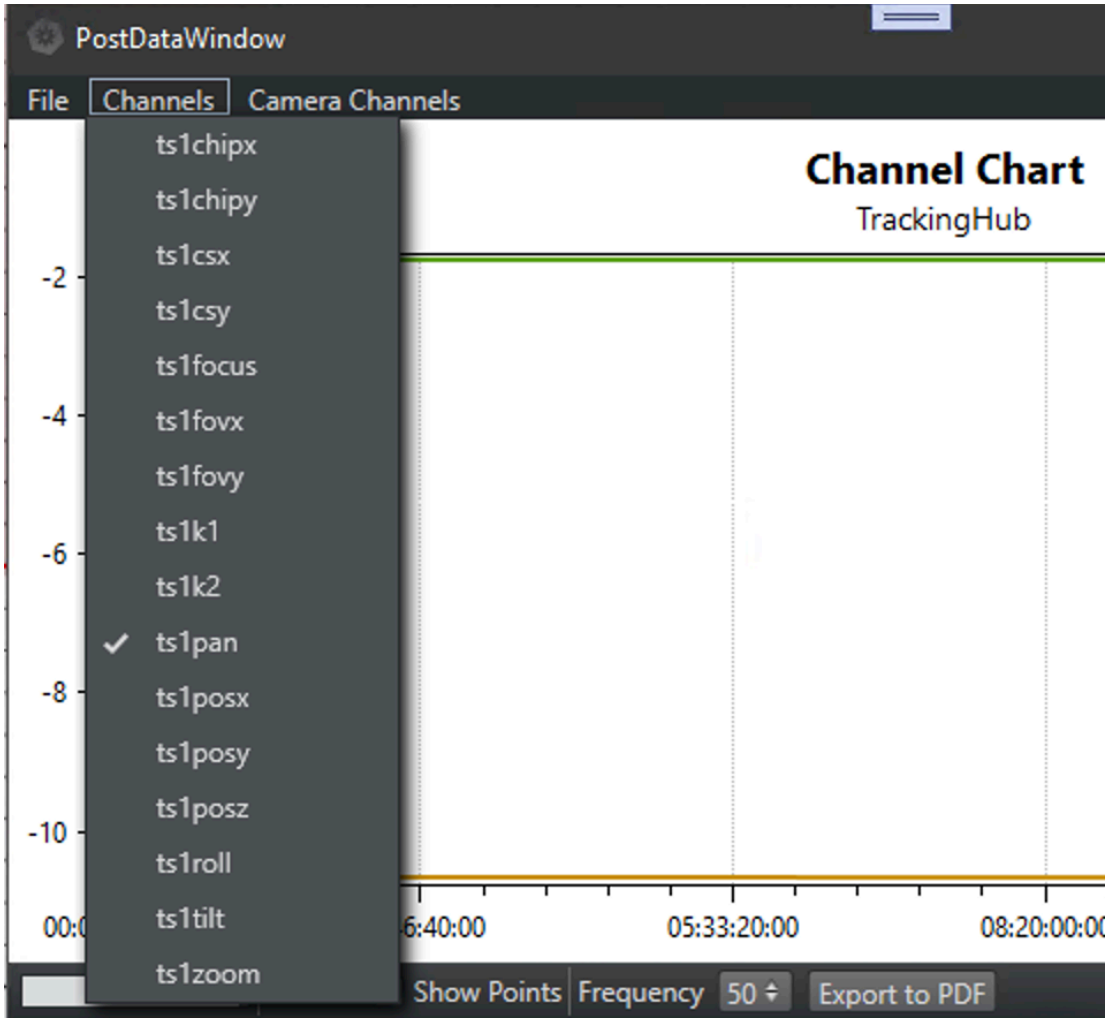


Copyright © 2026 Vizrt

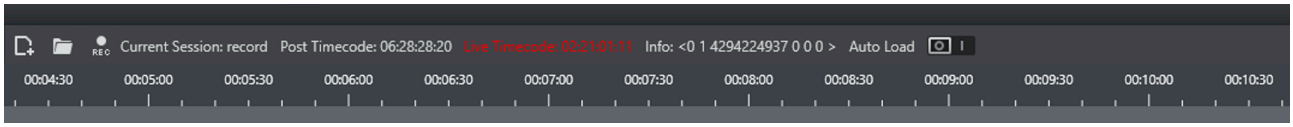


It is possible to multiple select parameters in the menus; however, the units of the parameters can be different (for example, it is not possible to compare distances (cm) with Euler angles).

Copyright © 2026 Vizrt

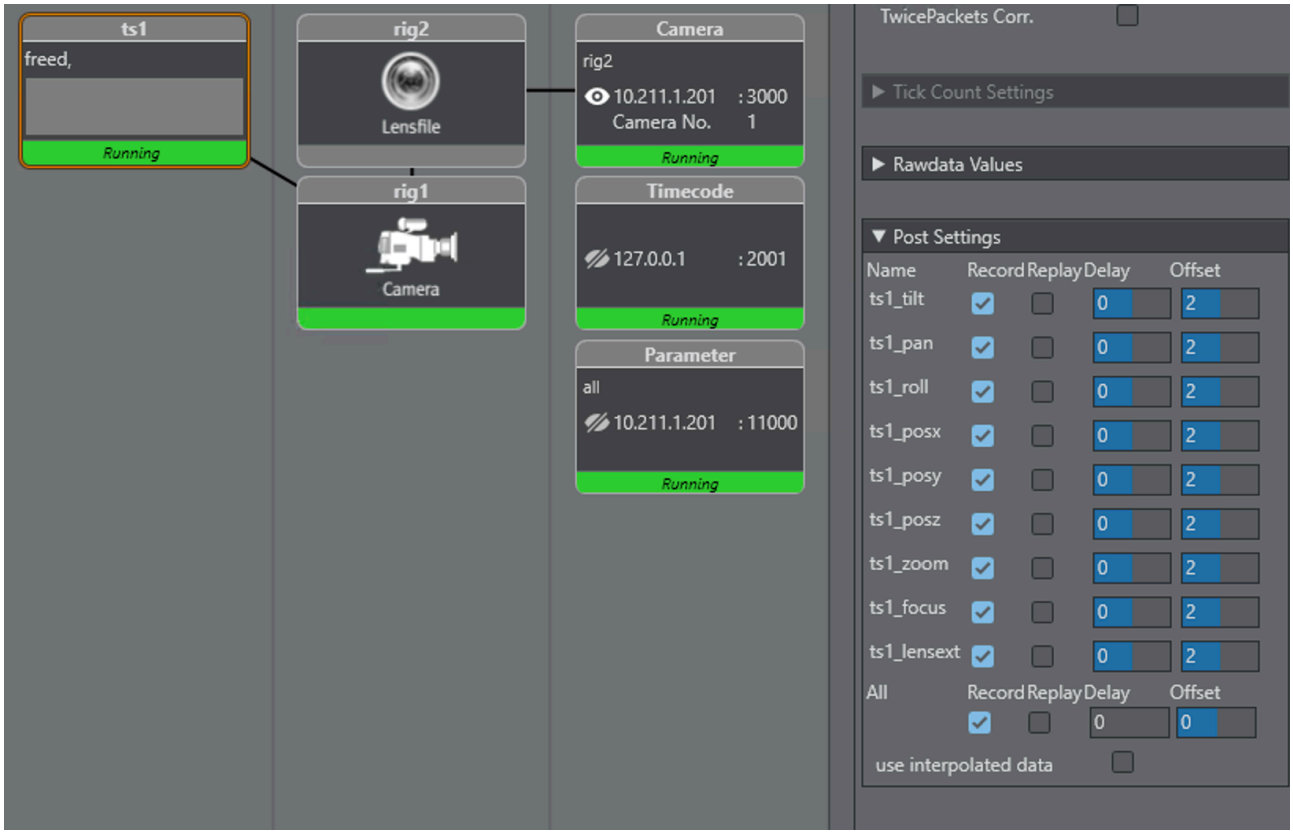


This tool can also be used to make long term recordings of the incoming tracking data. If there is no timecode present or no timecode reader card built into the machine, the Tracking Hub provides SimPost and SimLive timecode.

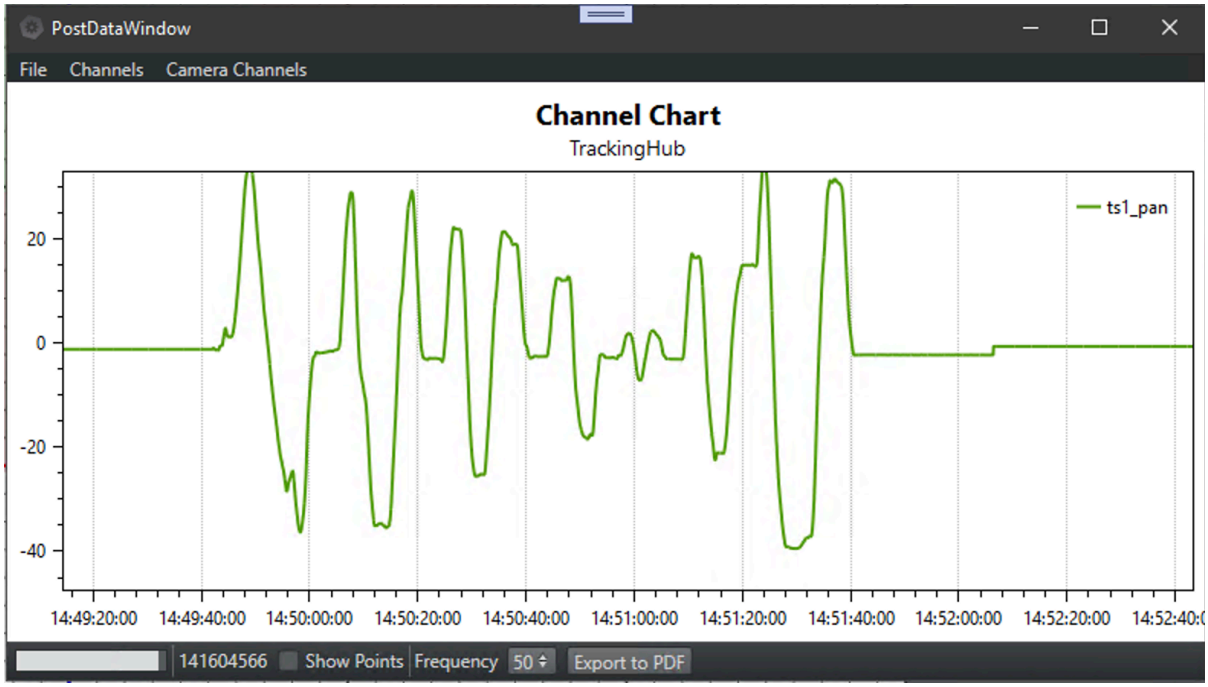


This simulated timecode start at startup time with a value of Zero. So if you need to know the time, the error happened, not the time of the Tracking Hub startup time. Select the tracking system you want to observe and activate the record settings for the incoming parameters.

Copyright © 2026 Vizrt



The record time is limited to 24:00 hours, but the recommended recording time should not exceed 12 hours. Otherwise, the PostDataWindow slows down.



Copyright © 2026 Vizrt

6.6.1 Controlling the Data-plot

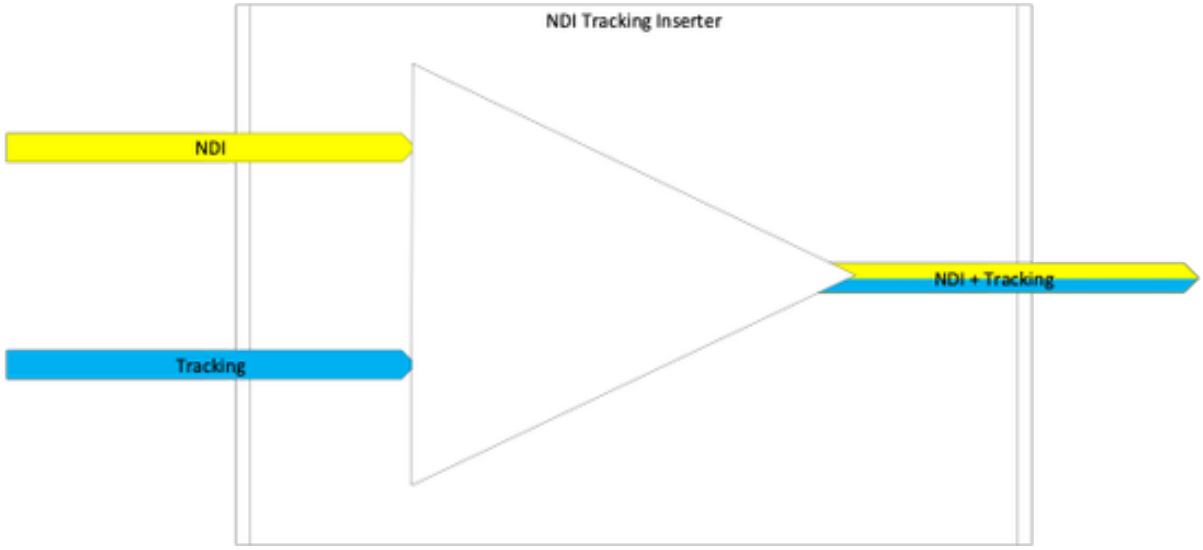
Action	Shortcut
Pan	Right mouse button, **ALT +** Left mouse button, **UP/DOWN/LEFT/RIGHT ARROW** keys, **CTRL + ARROW** keys for fine pan
Zoom	Mouse wheel, **CTRL +** Mouse wheel for fine zoom
Zoom in	Mouse extra button 1, **+** (Plus), **PAGE UP**, **CTRL + +** (Plus)/**PAGE UP** for fine zoom in
Zoom out	Mouse extra button 2, **-** (Minus), **PAGE DOWN**, **CTRL + -** (Minus)/**PAGE DOWN** for fine zoom out
Zoom by rectangle	**CTRL +** Right mouse button, Middle mouse button, **CTRL + ALT +** Left mouse button
Reset	**CTRL +** Right mouse button double-click, Middle mouse button double-click, **CTRL + ALT +** Left mouse button double-click
Reset axes	**A**, **HOME**
Copy bitmap	**CTRL + C**
Tracker	Left mouse button, **SHIFT +** Left mouse button for points only tracker, **CTRL +** Left mouse button for free tracker (show mouse coordinates basically),

Copyright © 2026 Vizrt

6.7 NDI Output Service

6.7.1 Embedding UDP Tracking into an NDI Stream

Tracking Hub supports embedding the rigs tracking data into an NDI stream.

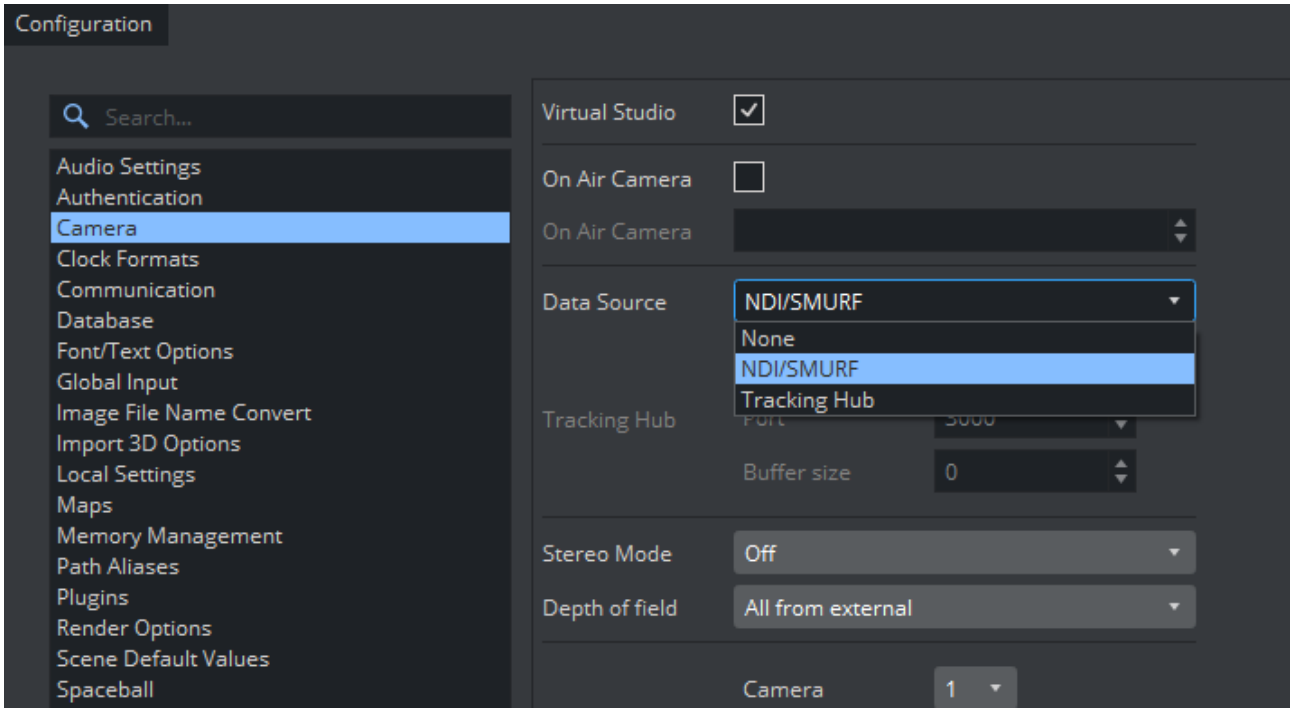


The NDI Service is free running and tries to synchronize the incoming packages from the camera to the incoming video frames from the NDI stream. It is quite important that the NDI camera is connected to a separate network without any router between the Tracking Hub and the PTZ Camera. It is also mandatory that the Tracking Hub is the only connected instance to the camera.

6.7.2 Viz Engine Configuration

Open the Viz Engine configuration, select the Camera section, and set **Data Source** to **NDI/SMURF**. See Tracked Cameras and Viz Engine for the full procedure. The Viz Engine must be restarted afterwards.

Copyright © 2026 Vizrt



6.7.3 Camera Configuration

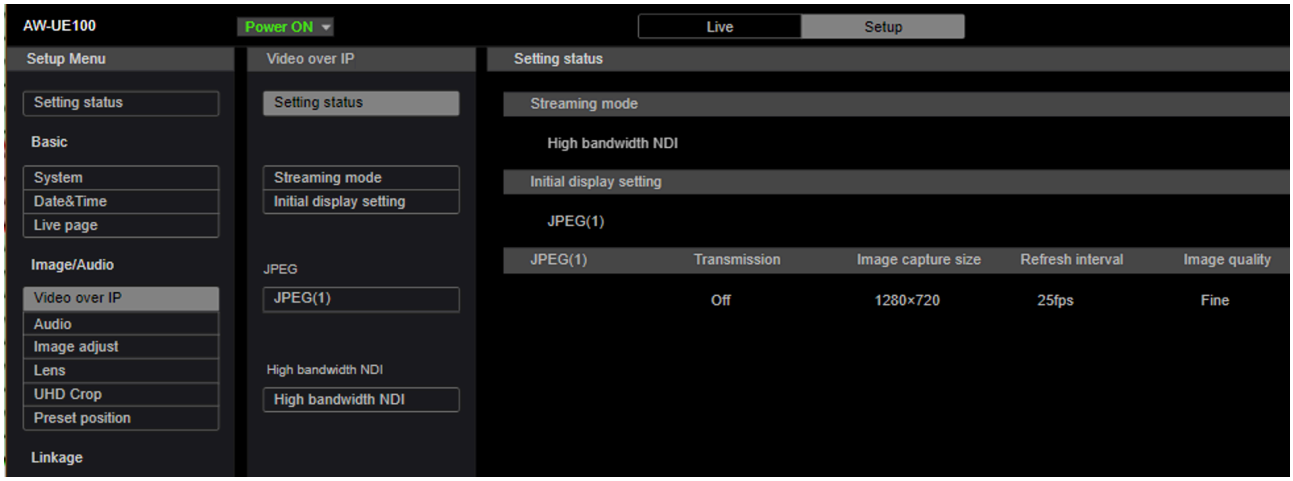
There are also some important settings which need to be done in the Camera Setup. As an example, we configured the Panasonic AW100 PTZ camera.



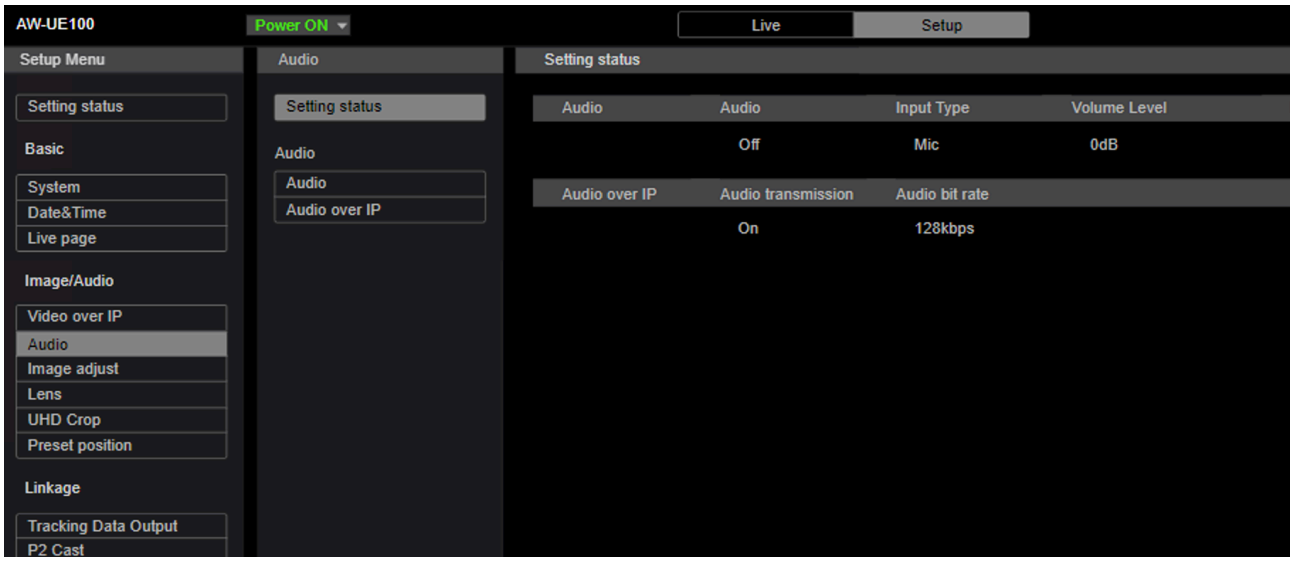
See Supported NDI Cameras for the cameras tested with NDI tracking data.

Copyright © 2026 Vizrt

Note: The minimum recommended firmware version is V01.33.

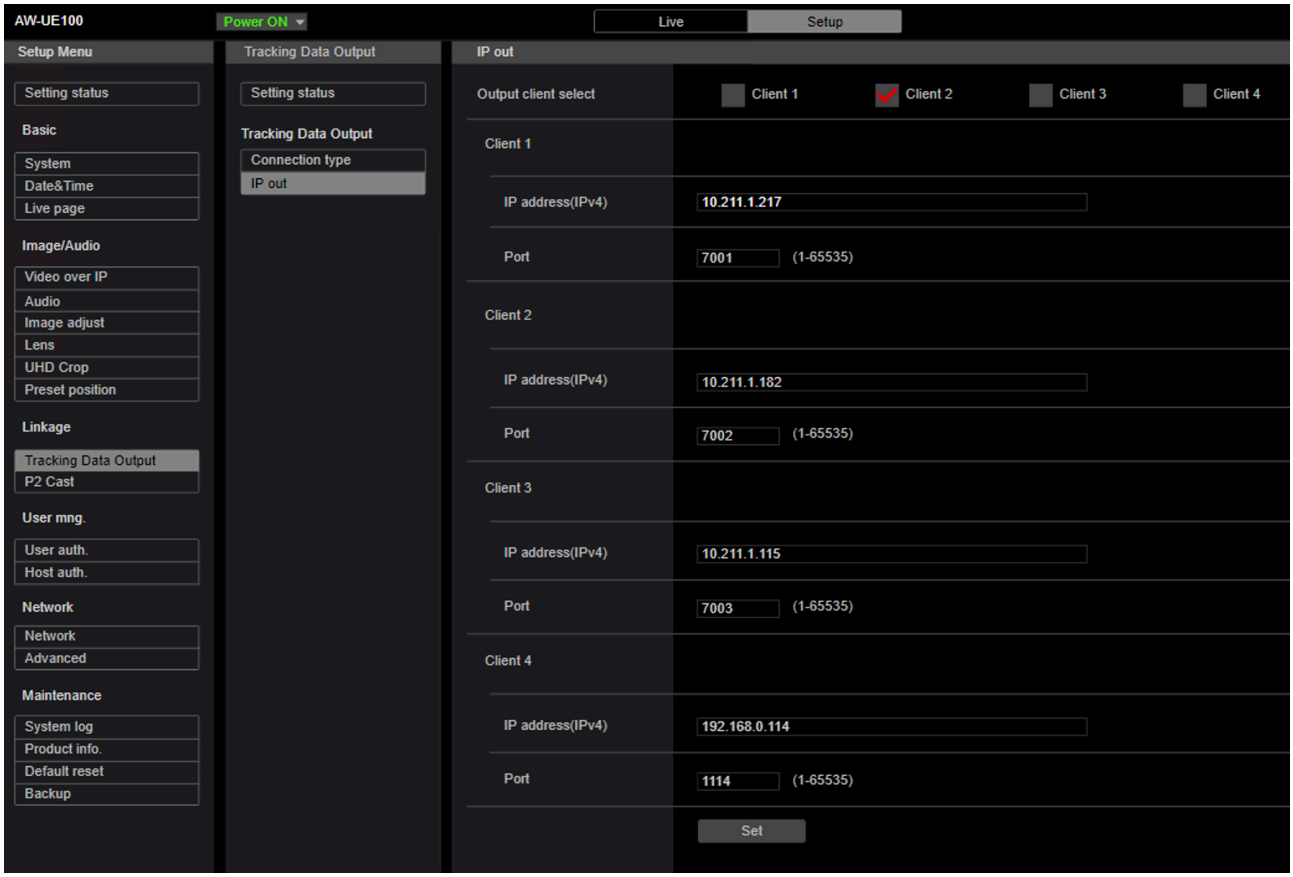


The streaming mode must be set to High Bandwidth. It is recommended to switch off JPEG streaming to the web GUI.



Audio is not supported in Tracking Hub. The reason is that NDI sender like the Panasonic AW100 attempts to synchronize video and audio by repeating or skipping frames when audio gets unsynchronized. This leads to delay changes in the tracking data. When a frame is missed, the virtual objects lag behind the real video, and when a field is repeated the virtual objects are in front of the video. It is best to deactivate audio in the camera settings.

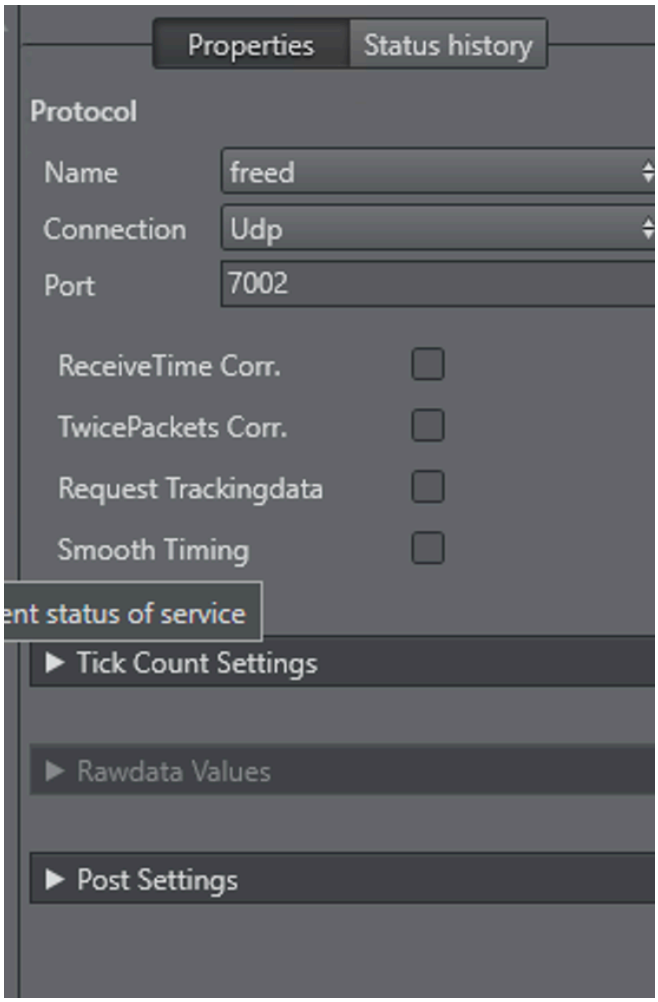
Copyright © 2026 Vizrt



There is also need to activate tracking data output of the camera. Activate one client and enter the IP address of the Tracking Hub. The camera then sends the UDP packets to the tracking driver of the Tracking Hub.

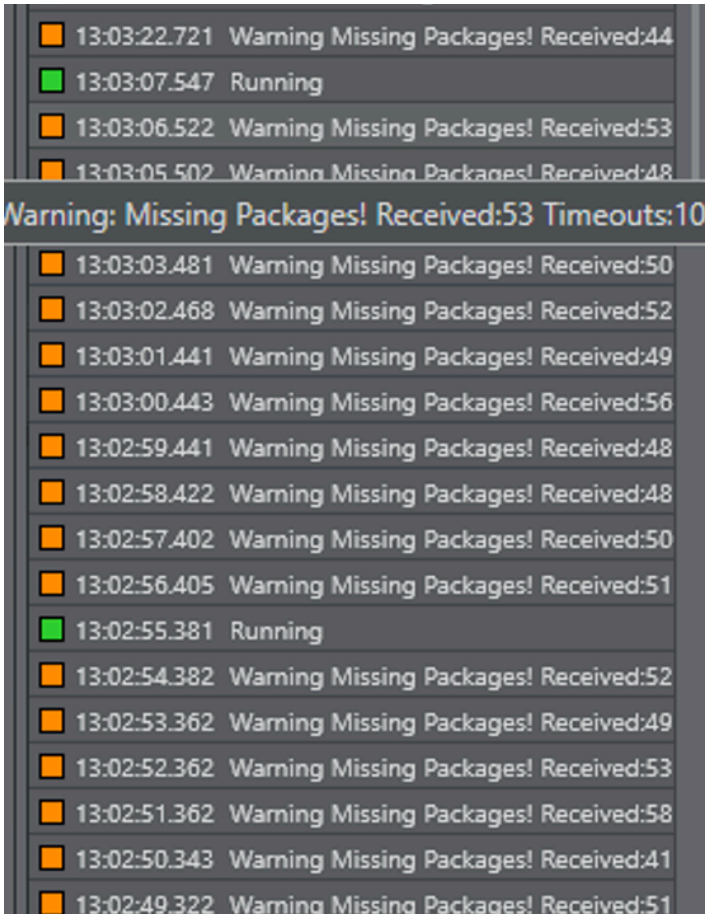
Copyright © 2026 Vizrt

6.7.4 Studio Manager Configuration



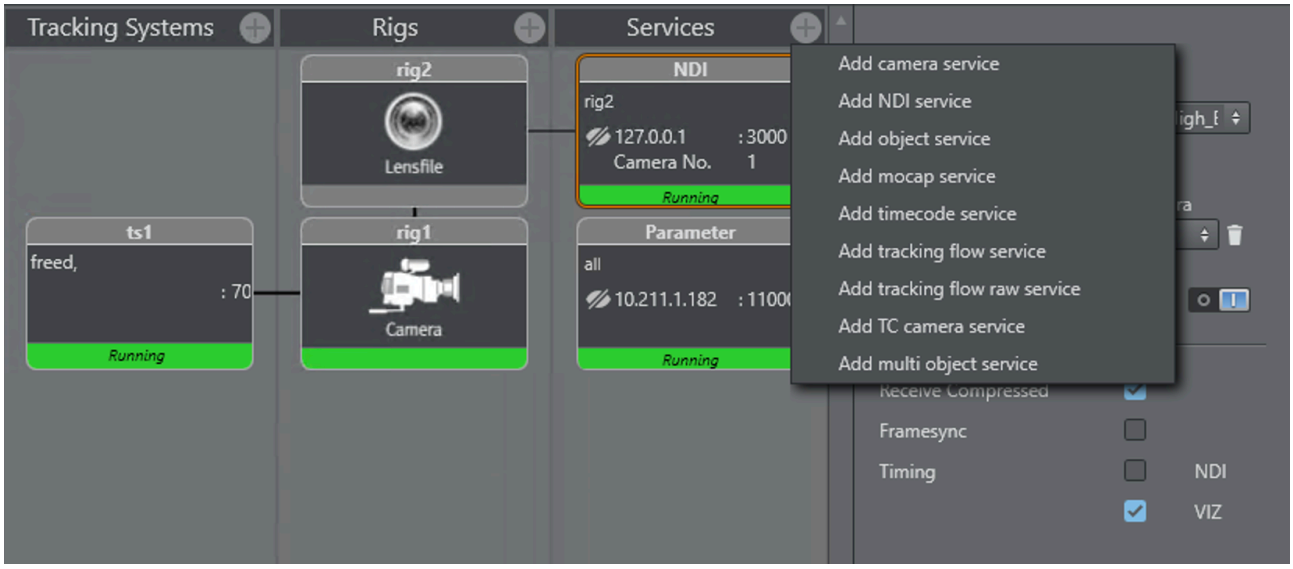
Add a new Tracking Driver and select *freed* as protocol. Enter the port number, which was selected in the setup section of the cameras user interface. In this example, it is `7002`. Then right click on the Tracking Driver and select connect. The tracker changes to green if all values were set correctly.

Copyright © 2026 Vizrt



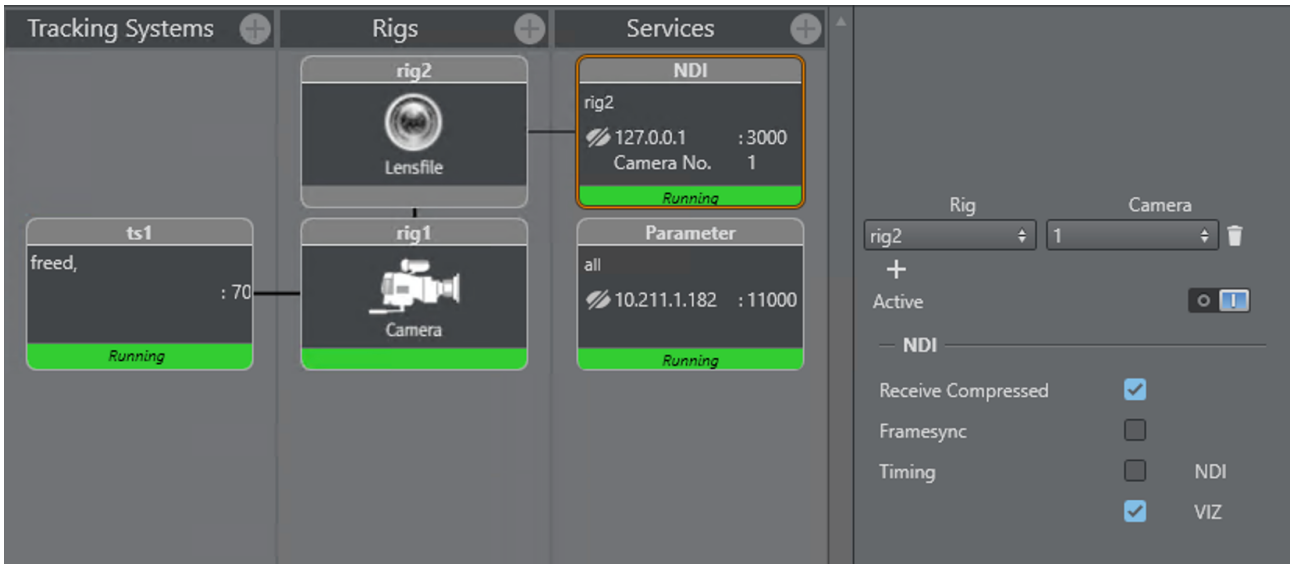
In case the tracker color goes from green to orange and the incoming tracking data is irregular or erratic, the camera needs a reboot. Power off the camera and turn it on again. If the camera data is not coming in correctly after this procedure, check the frame rate and video settings of the Tracking Hub and the camera. The next step is to set up the rig and lens file (please refer to the Configure Topology section). When the topology is set, a NDI service can be added.

Copyright © 2026 Vizrt



Click on the **+** Button in the service section of the Studio Manager and **Add NDI service**. The service properties a dropdown shows all available NDI streams. Select the stream of the NDI camera, then add a rig and select the camera number. When using more than one camera, every service should use a different camera number. When in Viz, the video should be switched from one camera to the other, the correct camera number must be sent to Viz Engine.

Like all other Services the NDI Service can be switched On and Off. The other options of the Service are:



- Click the **+** button in the service section of the Studio Manager and add an NDI service. In the service properties, a drop-down shows all available NDI streams. Select the stream of the NDI camera, then add a rig and select the camera number. When using more than one camera, every service should use a different camera number. When the video is to be switched from one camera to another in Viz, the correct camera number must be sent to Viz Engine.
  Like all other services, the NDI service can be switched on and off. The other options are:

Copyright © 2026 Vizrt

- **Receive Compressed:** Tracking Hub does not decode the video frames. It only extracts the video metadata and adds the tracking information into it. When deactivated, Tracking Hub needs much more CPU resources. This option is on by default.
- **Use AVsync:** Changes how the receiver handles the audio of the NDI stream. Off by default.
- **Timing:** Selects what the service paces its output on. The three settings are mutually exclusive:
  - **NDI:** Sends on every frame arriving from the NDI stream.
  - **NDI av:** Sends on an averaged frame time, which evens out irregular frame arrival. This is the default.
  - **VIZ:** Sends on the Tracking Hub synchronization instead of the NDI stream.

Receive Compressed is also available on the NDI tracking system, not only on the service.

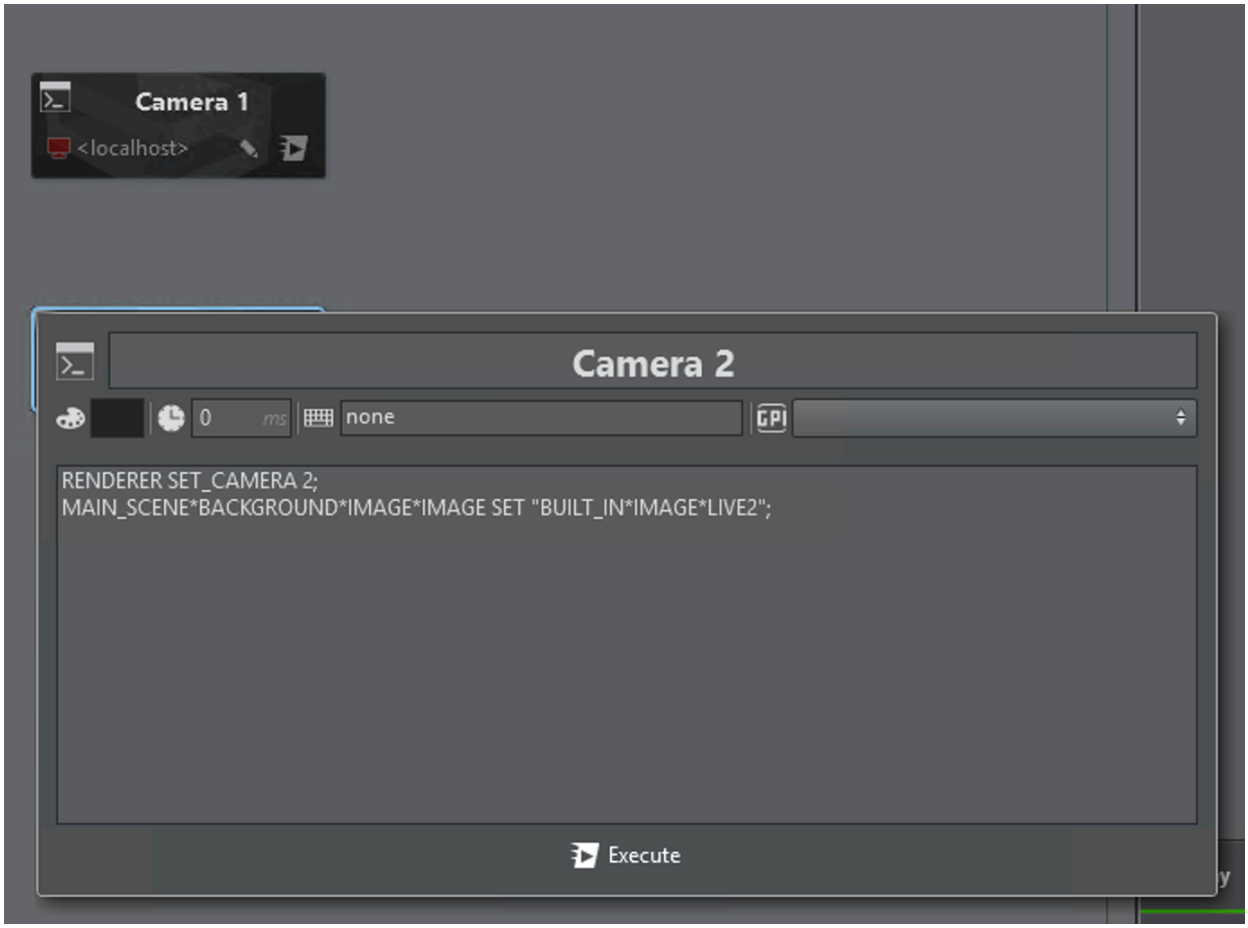
6.7.5 Viz Arc Camera Switching

The following command sequence from Viz Arc shows the correct commands to switch between camera 1 / LIVE1 and camera 2 / LIVE2. First, create two command buttons in Viz Arc and add the following commands to them:

```
RENDERER SET_CAMERA 1;
MAIN_SCENE*BACKGROUND*IMAGE*IMAGE SET "BUILT_IN*IMAGE*LIVE1";

RENDERER SET_CAMERA 2;
MAIN_SCENE*BACKGROUND*IMAGE*IMAGE SET "BUILT_IN*IMAGE*LIVE2";
```

Copyright © 2026 Vizrt



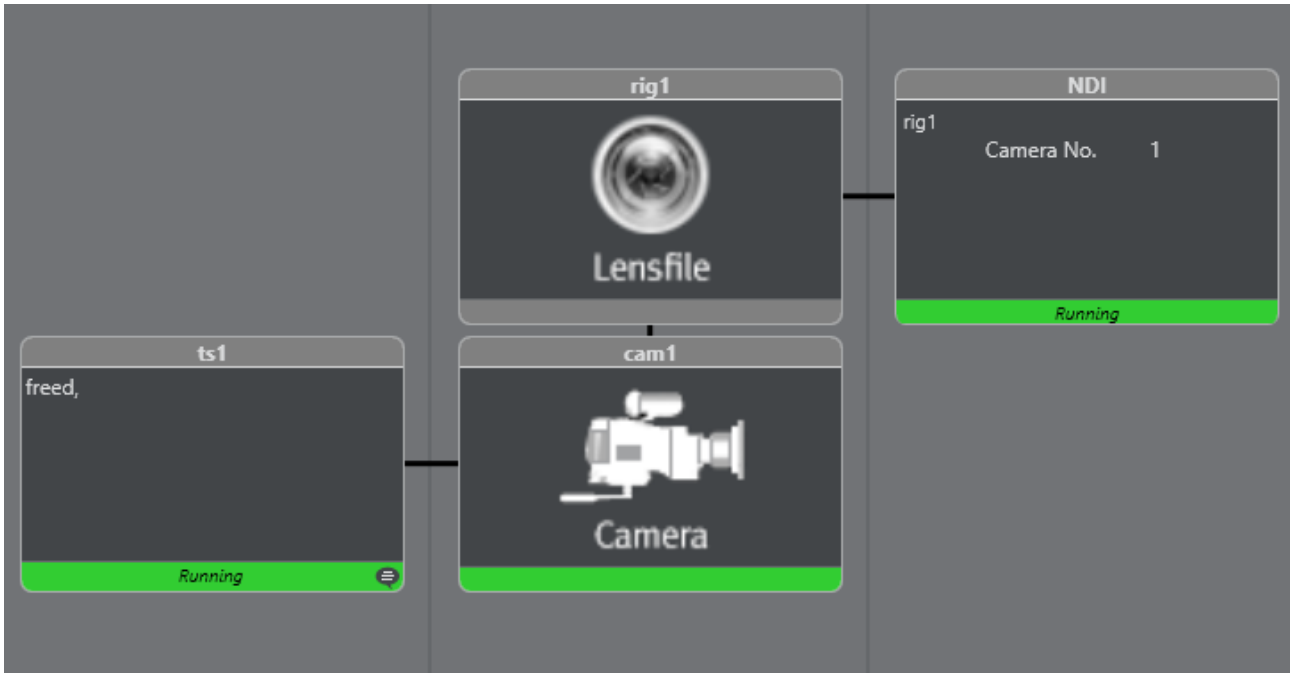
Copyright © 2026 Vizrt

6.8 Embedded NDI+ Tracking Data

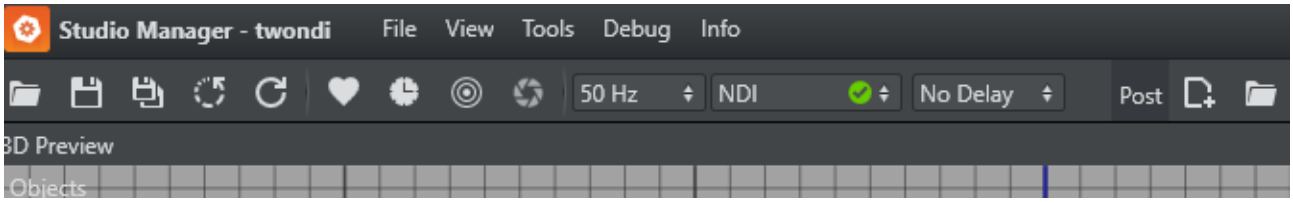
6.8.1 Introduction

When combining UDP and NDI data, the result depends on how the incoming NDI stream and UDP packages can be combined. The use of audio especially causes frame repeats and frame losses. In this case, a delay change can happen. The only way to overcome this problem is to embed tracking data into the video frames metadata by the camera itself. Tracking Hub is able to read the NDI+ included tracking data, adds additional offsets and lens information and wraps it into a new NDI output stream to be used in Viz Engine.

The example configuration above shows a typical NDI setup. The following section goes step by step how this can be set up.



6.8.2 Setting up the Studio



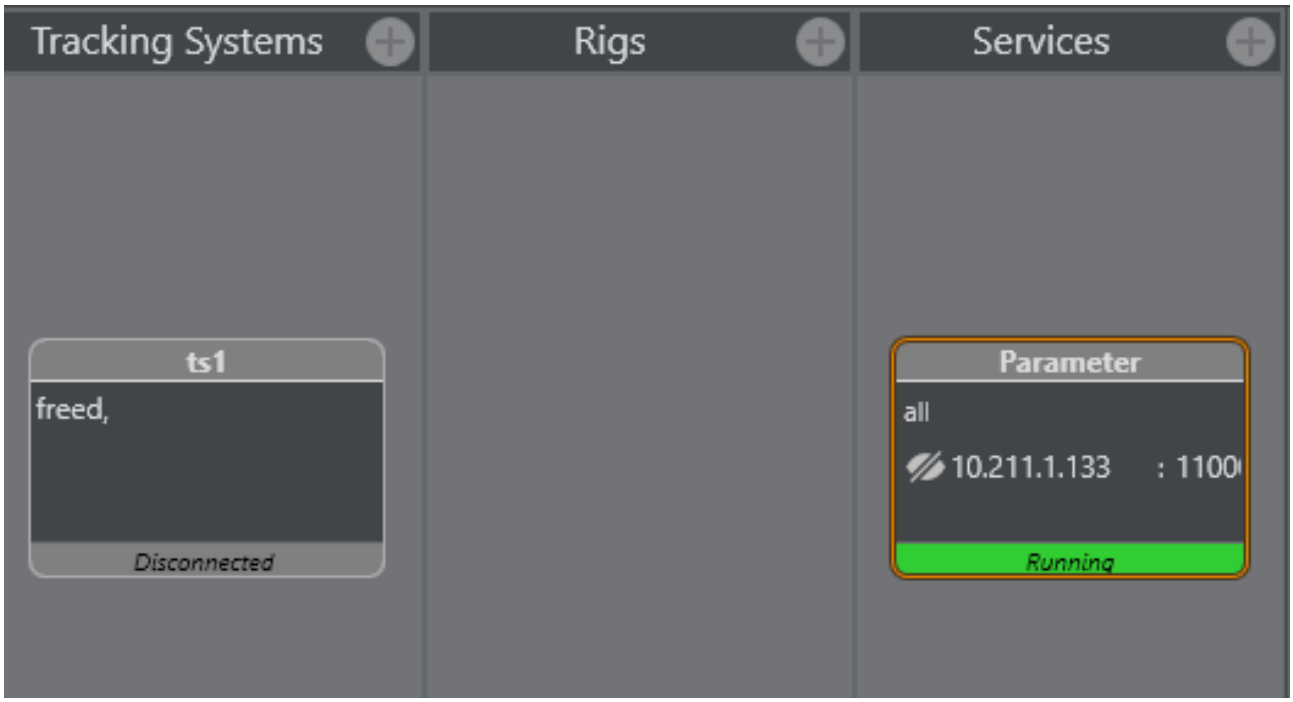
For NDI, the sync source of Tracking Hub can be set to *Free Run*. A new sync mode was introduced which is called *NDI* and which has the same functionality as *Free Run*. It is always shown with the green indicator. Every tracker and

Copyright © 2026 Vizrt

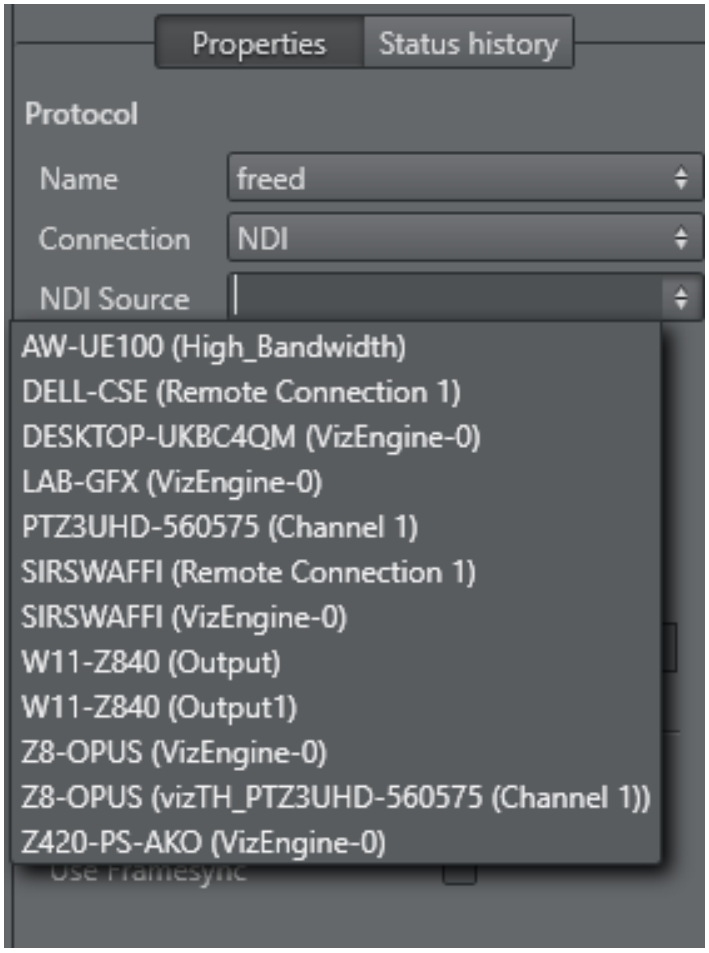
NDI service are organized into threads which synchronize to the incoming NDI stream. If there is a Plura card installed and Tracking Hub is set to *AV Card*, the NDI trackers and services are synchronized to the incoming NDI stream.

6.8.3 Setting up the Tracker

Caution: It is not recommended to have more than one active NDI tracker due to increased performance load on the system. Adding multiple NDI trackers can lead to insufficient GPU resources and an unstable tracking result. GPU recommendation for NDI embedded tracking are NVIDIA RTX 4000 cards or newer. NDI cameras and Tracking Hub should run on a dedicated network to prevent package loss.

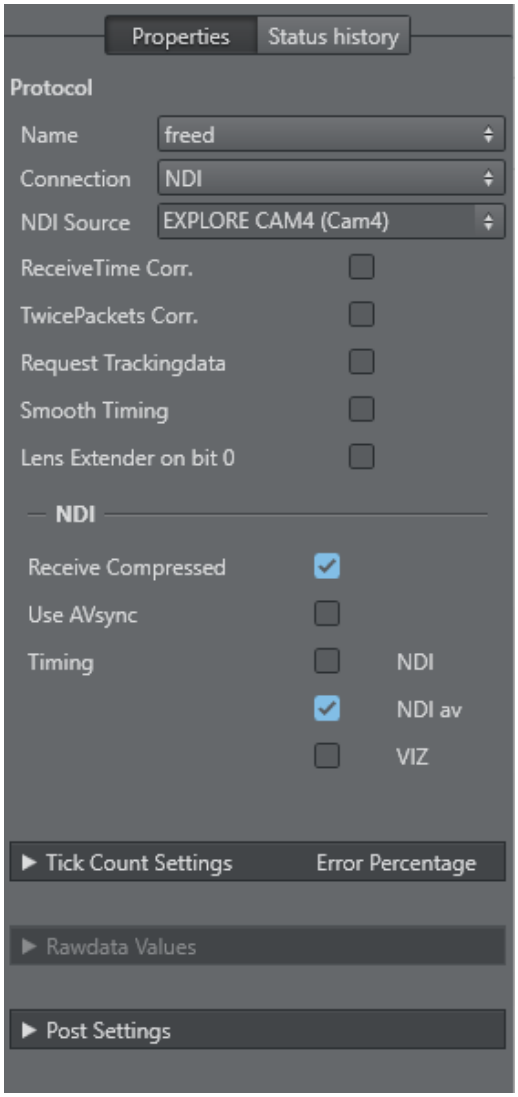


After adding a new tracking driver to the "Tracking Systems" panel a click on the tracker opens the property view.



When *NDI* is selected in **Connection**, a combo box appears where the NDI source can be selected.

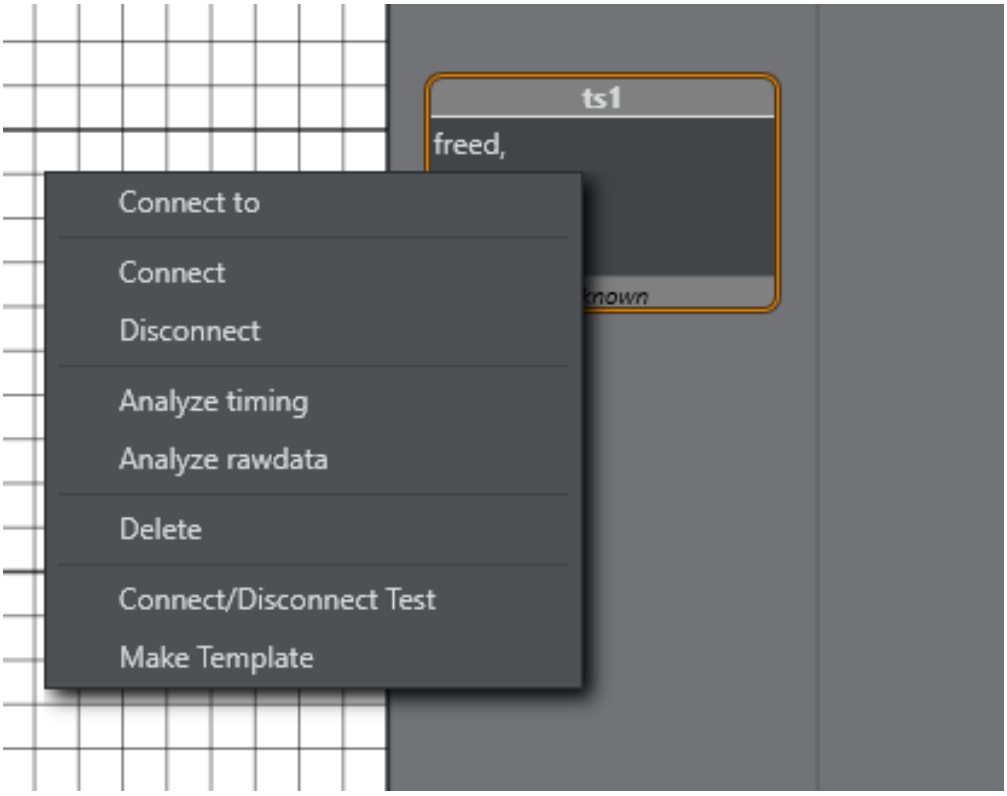
Copyright © 2026 Vizrt

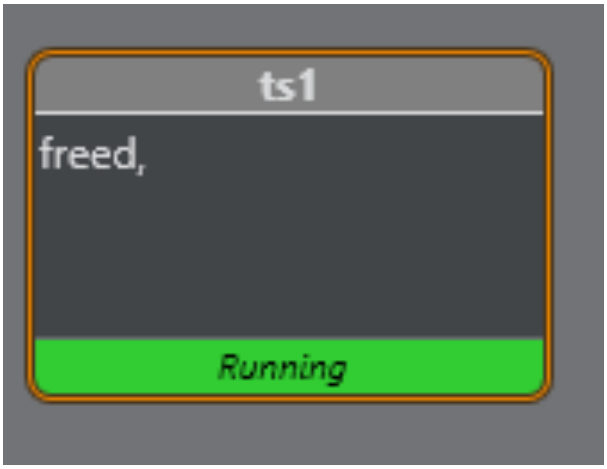


The following NDI options are available:

- **Receive Compressed:** The incoming NDI stream is not decoded by Tracking Hub. Only the metadata is read and analyzed, minimizing GPU and CPU usage. This option is enabled by default.
- **Use AVsync:** The NDI library tries to harmonize the timing of incoming and outgoing frames when activated. This is not needed for Tracking Hub to send a new frame as soon as one comes in, and not needed for the PTZ3 camera model, but it could be needed for future camera models. This option is not recommended and is switched off by default.
- **Timing:** Selects what the tracker paces its work on. The three settings are mutually exclusive:
  - **NDI:** Works on every frame arriving from the NDI stream.
  - **NDI av:** Works on an averaged frame time, which evens out irregular frame arrival. This is the default.
  - **VIZ:** Works on the Tracking Hub synchronization instead of the NDI stream.

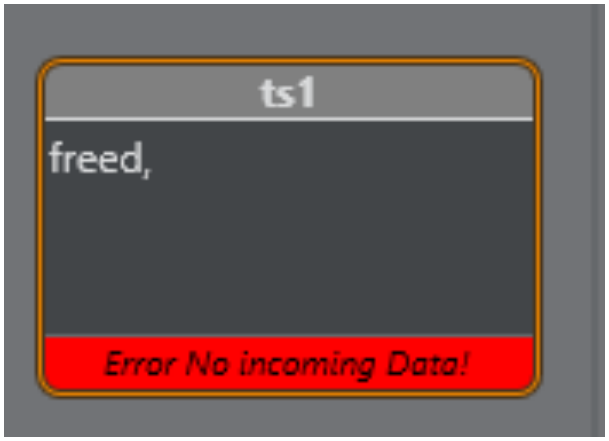
Copyright © 2026 Vizrt

- 



After connecting the tracking system the driver should become green and the status is running. Green means that Tracking Hub was able to find the tracking data in the metadata of the video and was able to decode it.

Copyright © 2026 Vizrt



If no data is present in the NDI stream, Tracking Hub shows *Error, No incoming Data!*.

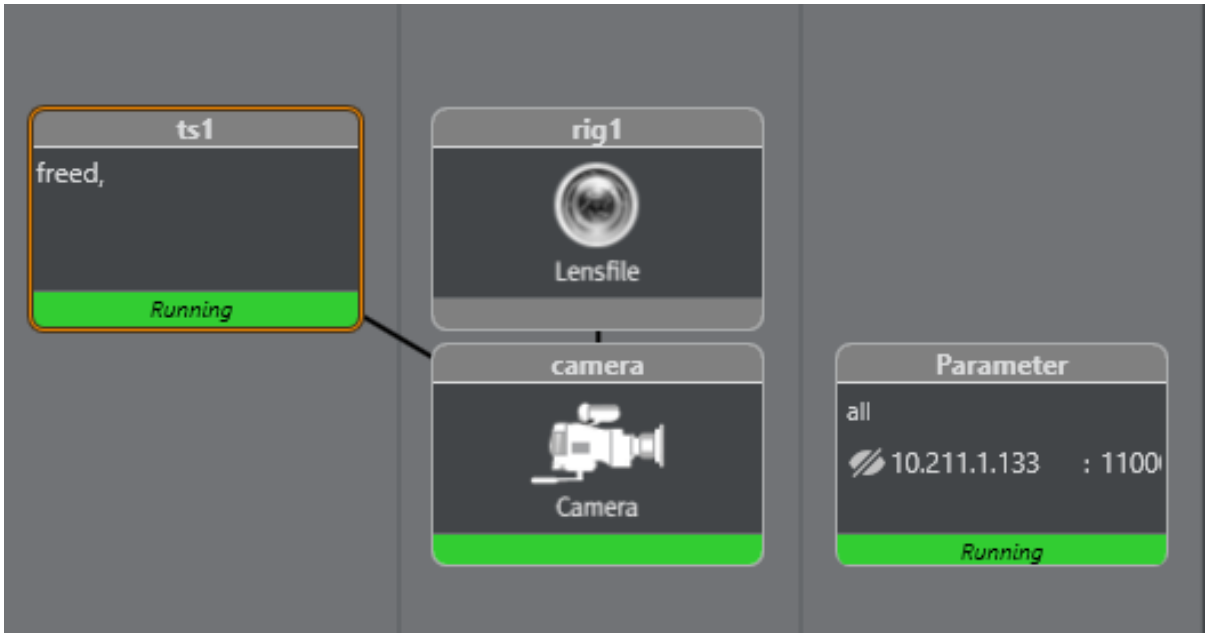
6.8.4 Setup Rig Topology

For setting up a rig topology in general, see Configure Topology. Make sure that the rig has the *Use LeastSquare* option activated when using NDI with embedded tracking data.

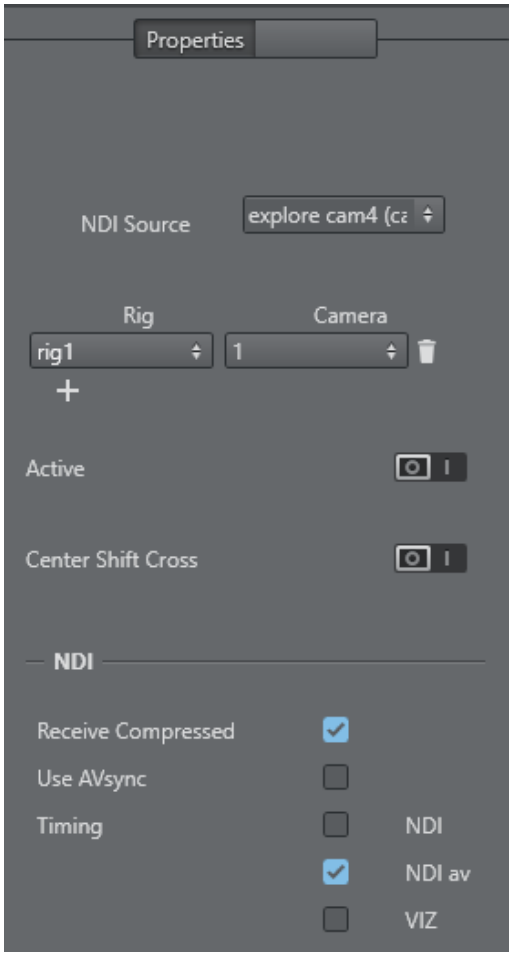


6.8.5 Setup NDI Service

After connecting the parameters to the Rig structure the setup looks like this:



After adding a NDI service and clicking on it the following properties panel appears:



Add a rig and camera number to the service. Please see: Configure Topology

IMPORTANT! The chosen NDI source in the NDI service has to match the source in the tracking system. If a different NDI source is connected, the tracking system automatically disconnects.

As soon as the rig is connected, the NDI source of the tracker and the service are connected. This is also true for the NDI options. Please see above in Setting up the Tracker for the meaning of the options.

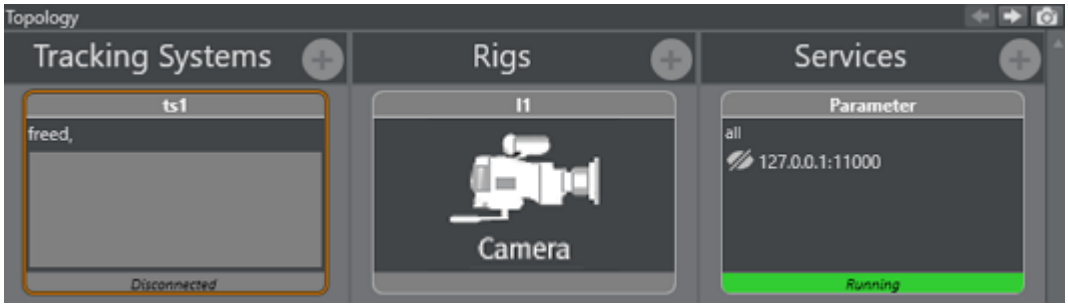
The NDI Queue size can be used to change the delay of the NDI signal. By default the size is set to `1`. This does not have any influence on the tracking delay. It delays the overall delay including the tracking data. The delay of the tracking data must be set in rigs overall delay and in the parameter delay settings.

Note: The cyc data is sent to Viz Engine only, when at least one camera service exists. In case of NDI, at least one camera service must exist for every Viz Engine.

Copyright © 2026 Vizrt

6.9 Configure Topology

This section details Topology configuration, which includes Tracking Systems, Rigs and Services.



At any time, view the properties of each configuration in the Configuration Panel.

The Topology configuration should be done in this order:

- Configure a Tracking System.
- Configure a Rig.
- Configure a Service.

6.9.1 Configure a Tracking System

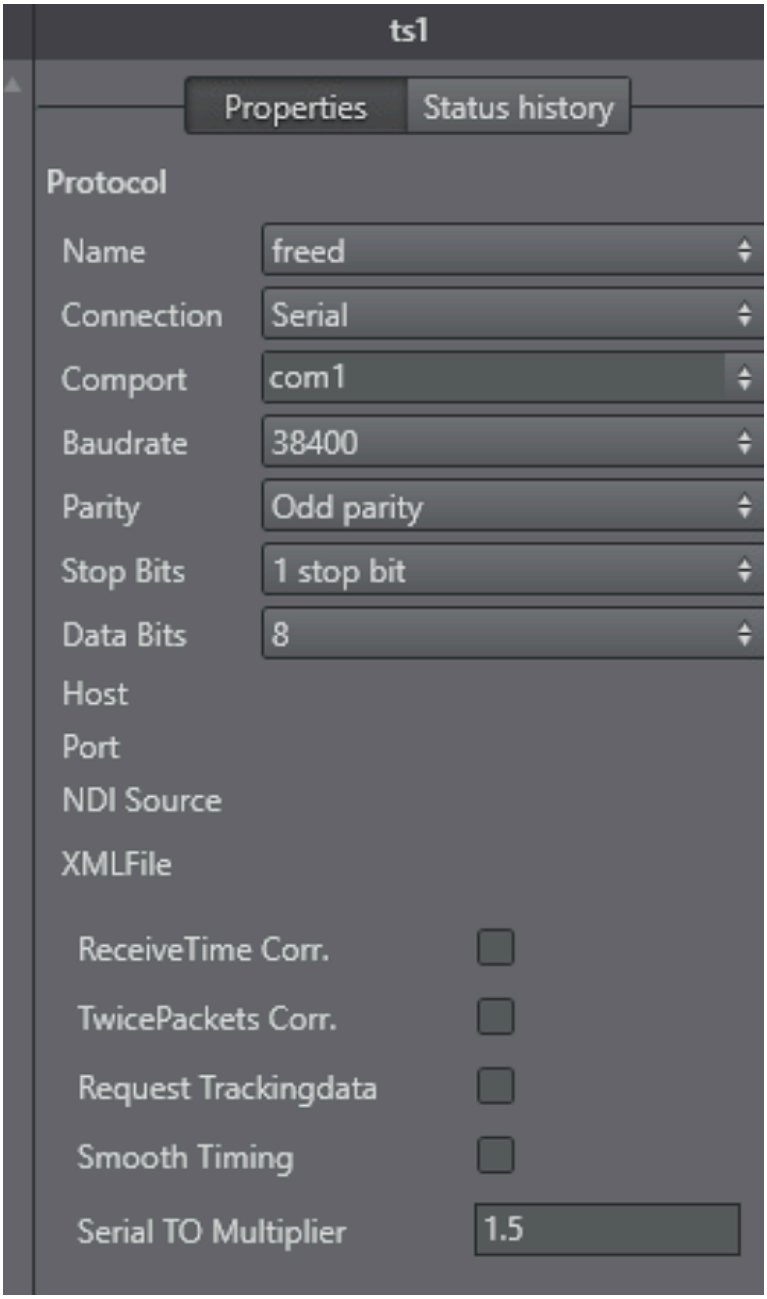
To Add and Configure a Tracking System

1. Click .
2. Type a **Name** for the configuration.
3. Select which protocol to use to receive data from the tracking system:

Note: If changing the protocol of an already configured tracking system, the default configuration parameters for the new protocol are read by the system.

4. Select a connection type. Depending on the tracking system select the connection type:
    - **Serial:** If serial is selected, also select a **Comport** (in the **Comport** box, all available comports are shown). If the default baud rate is not used, enter the correct one in the **baudrate** field. Also set the parity, stop bits and data bits according to the values provided by the manufacturer.

Copyright © 2026 Vizrt



The *Serial Timeout Multiplier* should only changed in values between `1.1` and `2.0` and needs to be adjusted only when receive problems occur.

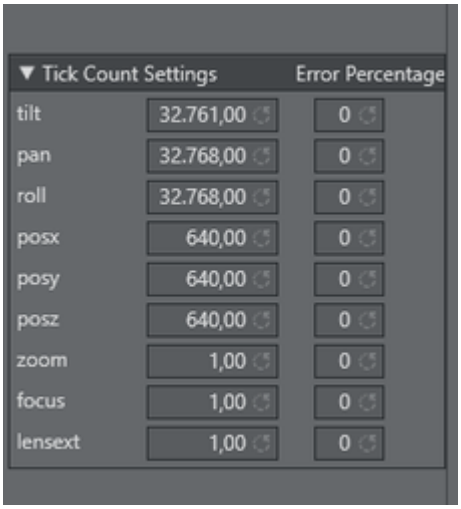
- **TCP/IP:** Type in the **Host** IP to use (it is possible to enter the port number as well).
- **UDP:** Type in a **Port** number to use.
- **NDI:** please see: Embedded NDI Tracking Data.

5. Options available for all tracking systems:
   - **ReceiveTime Corr:** Ignores the receive time of the package and sets it to the last sync time. This makes the interpolation smother, assuming that the tracking data sample rate is stable. Sometimes

Copyright © 2026 Vizrt

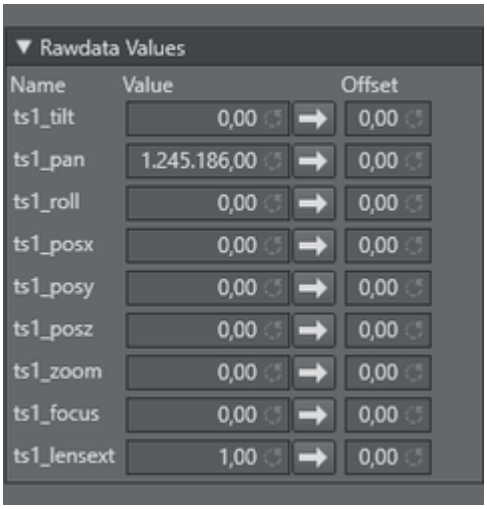
the tracking data transport can be unstable, for example, when serial data is converted to fiber and back to serial. In this case, the tracking packages are coming irregularly.

- **TwicePackets Corr:** Shifts double packets to the correct receive time. As a special case of the *ReceiveTime Corr.*, it can also happen that two tracking packets arrive at the same time. This is often the case when old USB converters are used.
- **Request Trackingdata:** Forces Tracking Hub to send the trigger to the tracking system. There are still some old tracking systems in the field which demand a trigger to send the data to Tracking Hub. Such systems are Thoma or old Vinten heads.
- **Smooth Timing:** Forces Tracking Hub to distribute the income time of all received packages evenly. Use when *Receive Time Corr.* and *TwicePackets Corr.* do not have the desired effect.
- **Serial TO Multiplier:** Determines the timeout when Tracking Hub reports missing tracking data is defined by the frequency of the studio sync (for example, for PAL the timeout is 20 milliseconds). When the packets are coming irregularly, Tracking Hub would continuously show an orange status. This timeout multiplier can improve the result by increasing this timeout interval.
- **Tick Count Settings:** Most encoder based systems do not send angles or position in centimeter to Tracking Hub. They send an encoder based tick count, which must then be converted to angles and positions. Usually, the divisor of the tick count is defined by the protocol. This default values are predefined in Tracking Hub. In rare cases it can happen that the tick count differs from the default values. When the tracker is disconnected this default values can be adjusted.



- **Rawdata Values:** The *Rawdata Values* show the actual incoming ticks per axis. It is also possible to add an offset to the tick count. This can be useful during lens calibration of when setting up a crane. This panel is only visible when the tracker is connected.

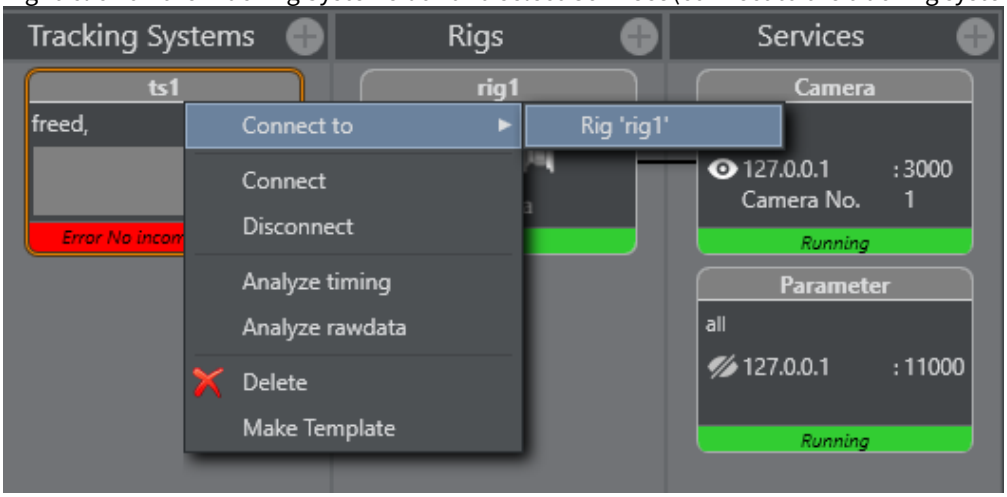
Copyright © 2026 Vizrt



- **Post Settings:** Please see: Post System.
- **Error Percentage:** Sometimes it can happen, especially when using old tracking systems, that an encoder get damaged. This usually results in one frame short random values sent to the Tracking Hub. The error percentage is the limit when Tracking Hub filters out unplausible values. The filter is activated when the actual value differs more than the error percentage from the last value.

Caution: This is an emergency solution. If such errors occur, the tracking system must be repaired as soon as possible.

6. Right-click on the Tracking Systems box and select **Connect** (connect to the tracking system).



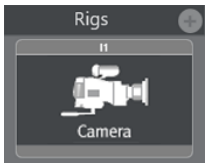
The Tracking Hub now connects to the selected Tracking System. Observe the colors on the bottom of the symbol:

- **Green:** Connection is done and data receiving in time.
- **Orange:** If the color becomes orange, after a few seconds, connection is done and data receiving, but not in time.
- **Red:** If the color becomes red, possible connection failure or data is not received.

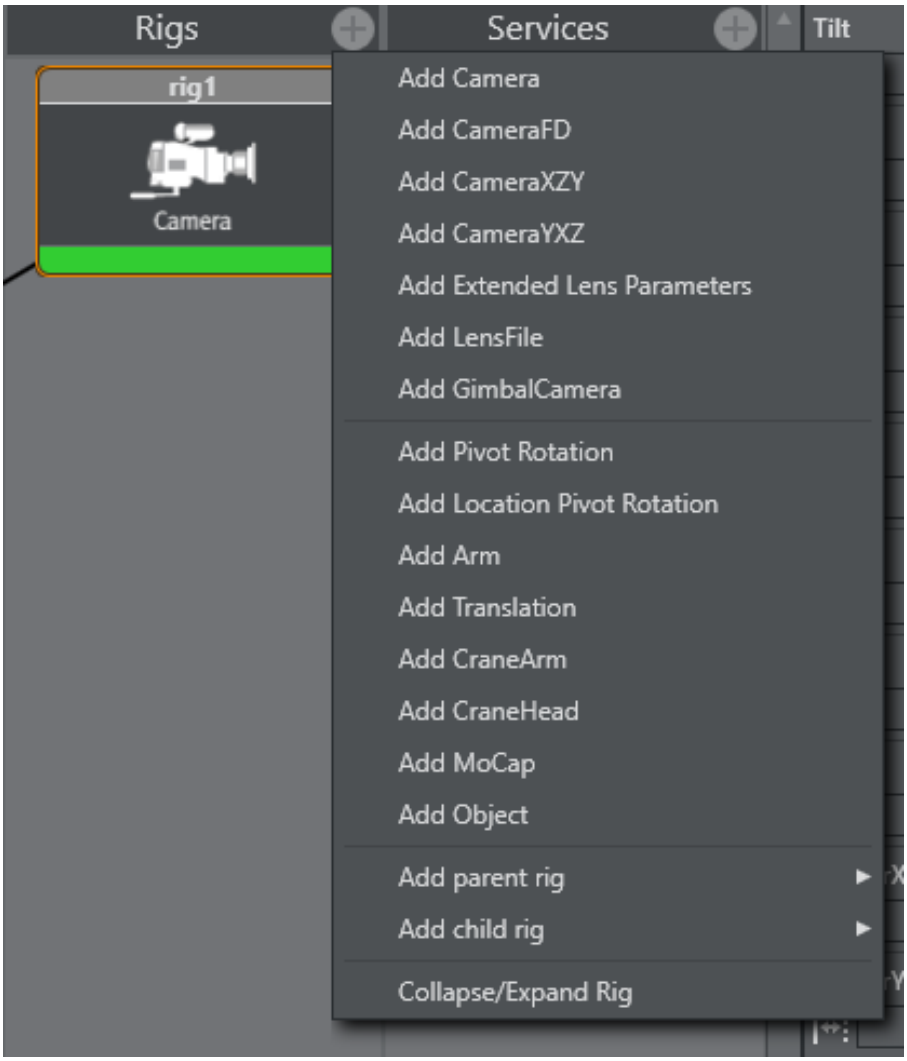
7. Go to Configure a Rig.

Copyright © 2026 Vizrt

6.9.2 Configure a Rig

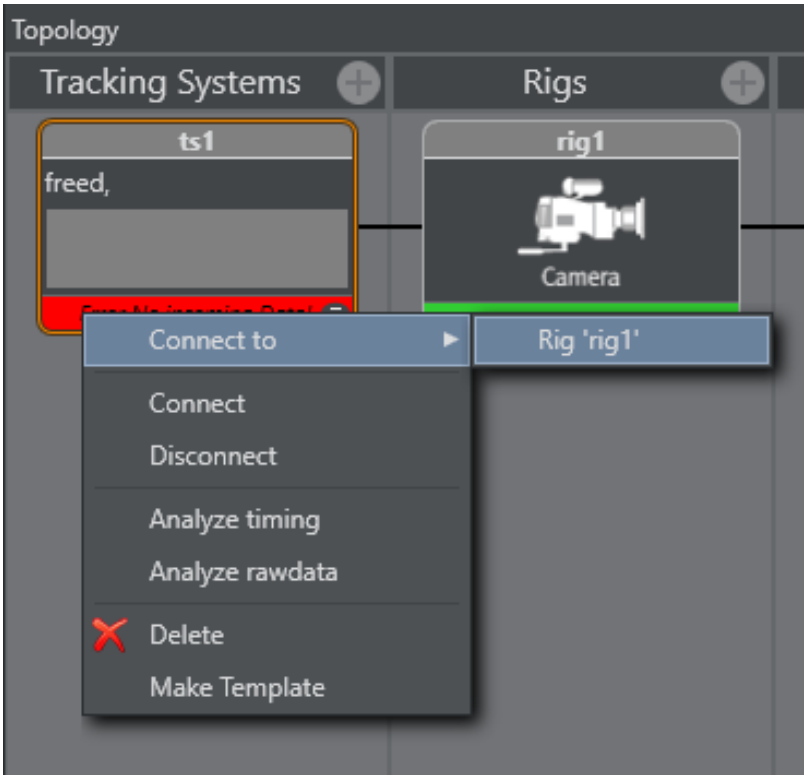


1. Click .
2. Select from the list:



   - Select the rig type to add. Which types are available, and what each one is for, is described under Rigs in Topology Panel.
3. Right-click the Tracking System and select **Connect to > Rig** (Lattice).

Copyright © 2026 Vizrt



4. Set or change tracking parameters as required. When finished Click **OK**.

Copyright © 2026 Vizrt

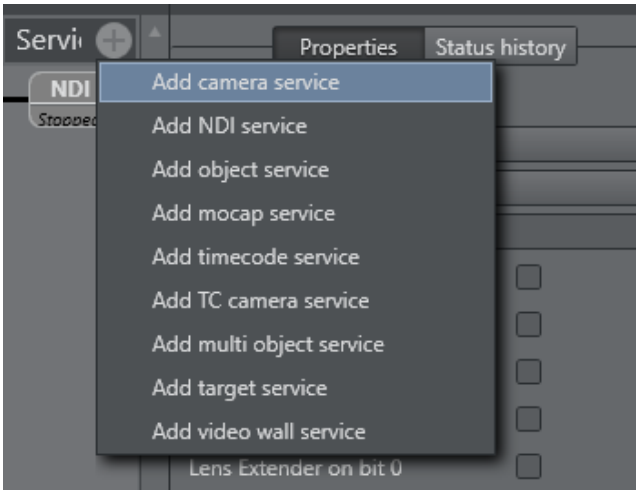


5. Go to Configure a Service.

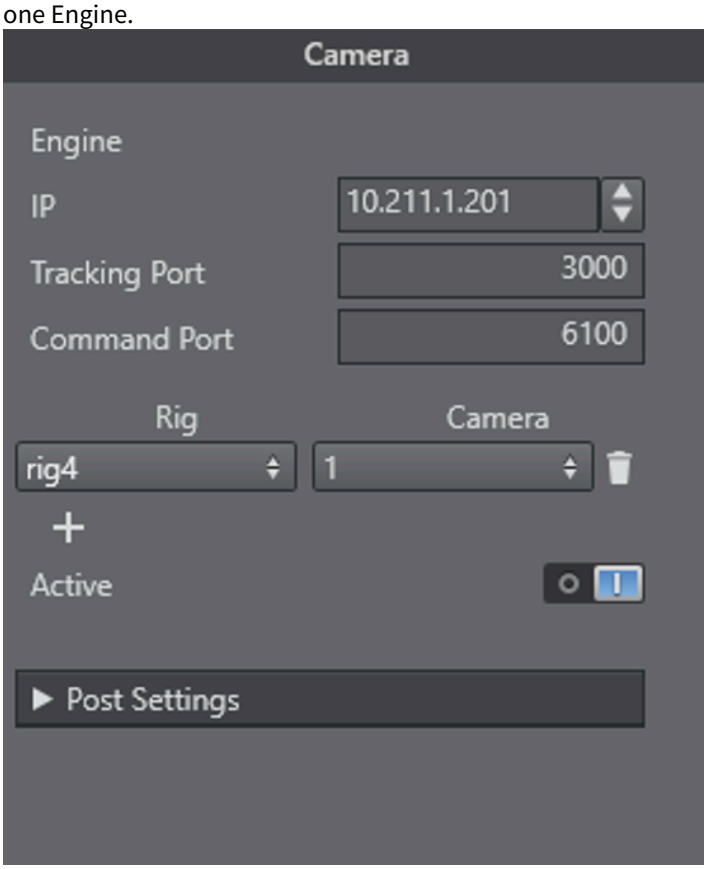
6.9.3 Configure a Service



1. Click .



2. Select the service to add. The available services are listed under Services in Topology Panel.
3. Click on the service icon in the Services panel.
4. Configure the service in the Properties panel. When the service is added, one rig and one camera field are available. Pressing the **+** button add a rig/camera combination. It is possible to send any number of rigs to one Engine.



a.
- **Rig:** Type the name of the rig whose data should be sent to an Engine.
- **IP:** IP address of the Viz Engine, which is to receive the tracking data.
- **Port:** Port number for this connection (same port as configured in the Viz Engine configuration).

Copyright © 2026 Vizrt

- **Cam.Number:** Same as Viz Engine is expecting.
- **Active:** When checked the service is active; unchecked, no data is sent to the Viz Engine.

5. Click **Start** in the Configure the Studio.
6. Click **Save** in the Configure the Studio.

6.9.4 Set a Delay

To match tracking data with the Engine rendering, set a delay (in fields). The delay can be set in fractions of fields and individual for every parameter.



A delay can be set for each parameter individually or overall for the whole camera rig node. The parameter delay is added to the the overall delay.

Copyright © 2026 Vizrt



6.9.5 Set an Offset

An offset (cm) can be added to a tracking parameter value, when passed to a rig, to correct positional rotation of a camera and, or an object

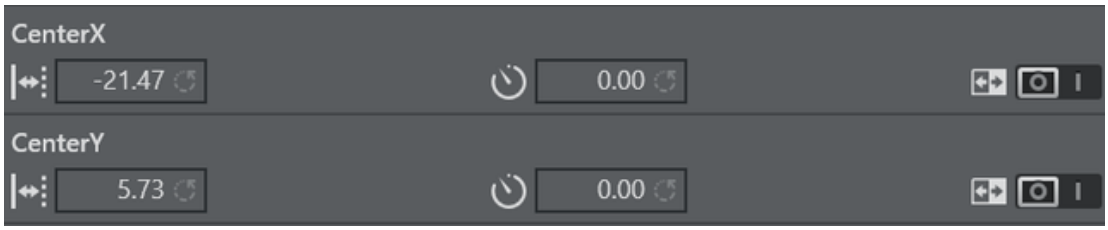


Values are set, if the field has lost its focus.

Copyright © 2026 Vizrt

6.9.6 Set Center Shifts

In case a drift of graphics can be seen while zooming the camera, the center shifts might need adjustment (image above shows example values).

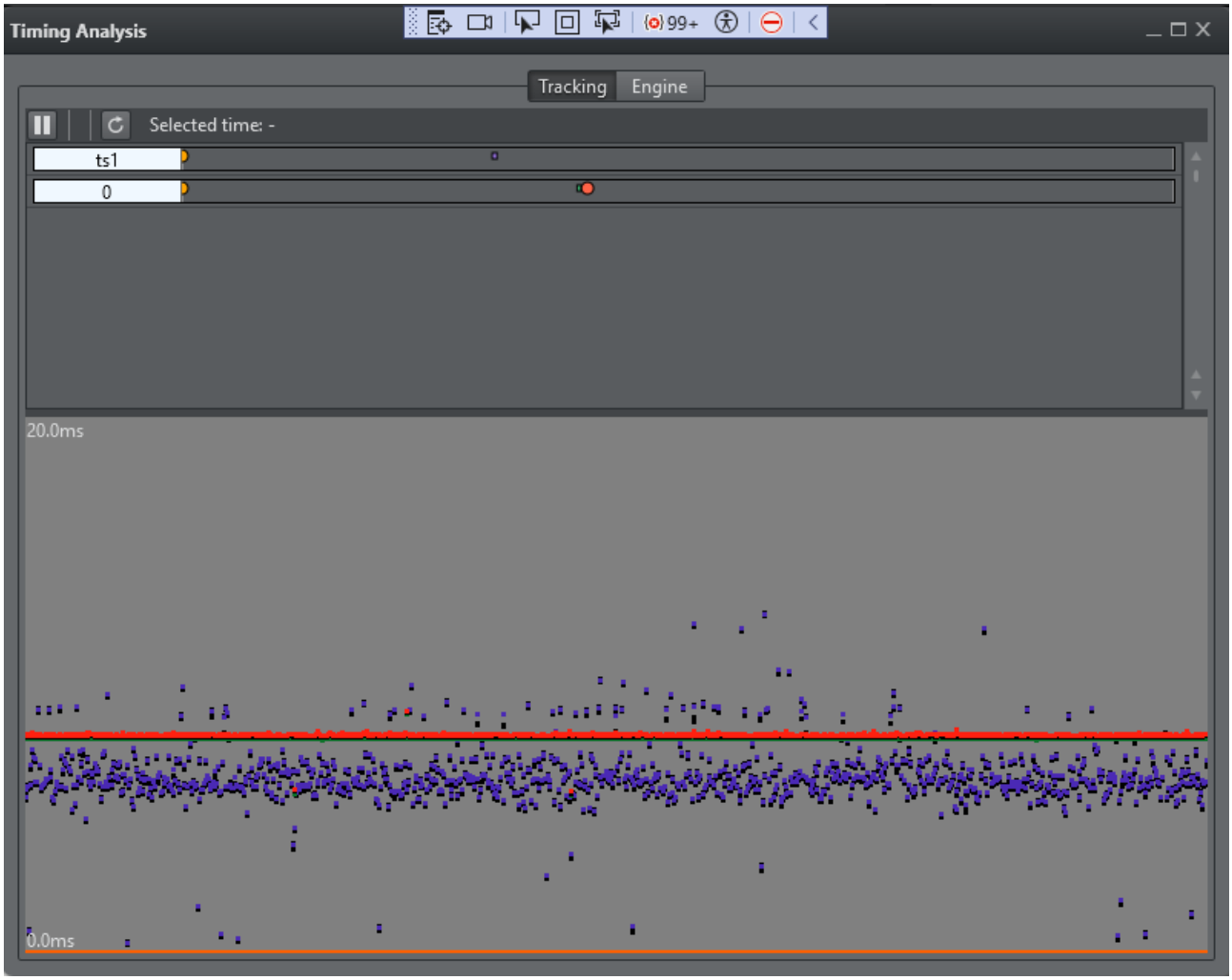


A guide on how to properly adjust center shifts can be found in chapter Tracking Hub Structural Design.

6.9.7 Analyze Time

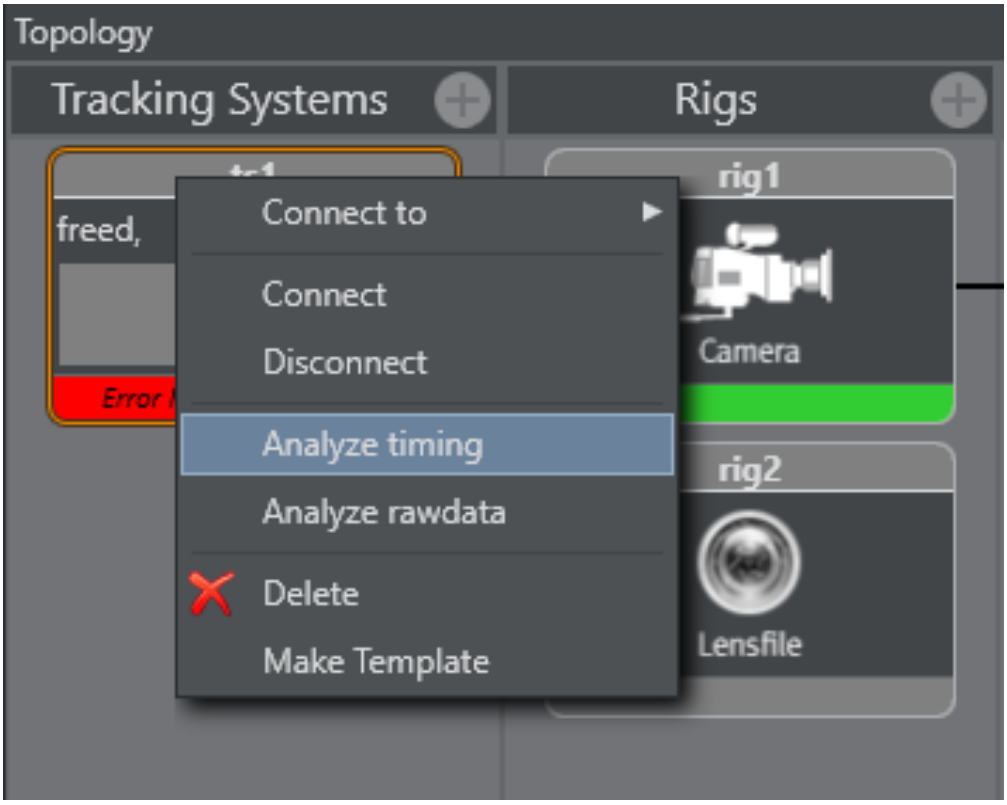
Analyze Time is a visual representation of the tracking.

Copyright © 2026 Vizrt



To Analyze Tracking System Time

1. Right-click on the Tracking System that is to be analyzed and select **Analyze Timing**.



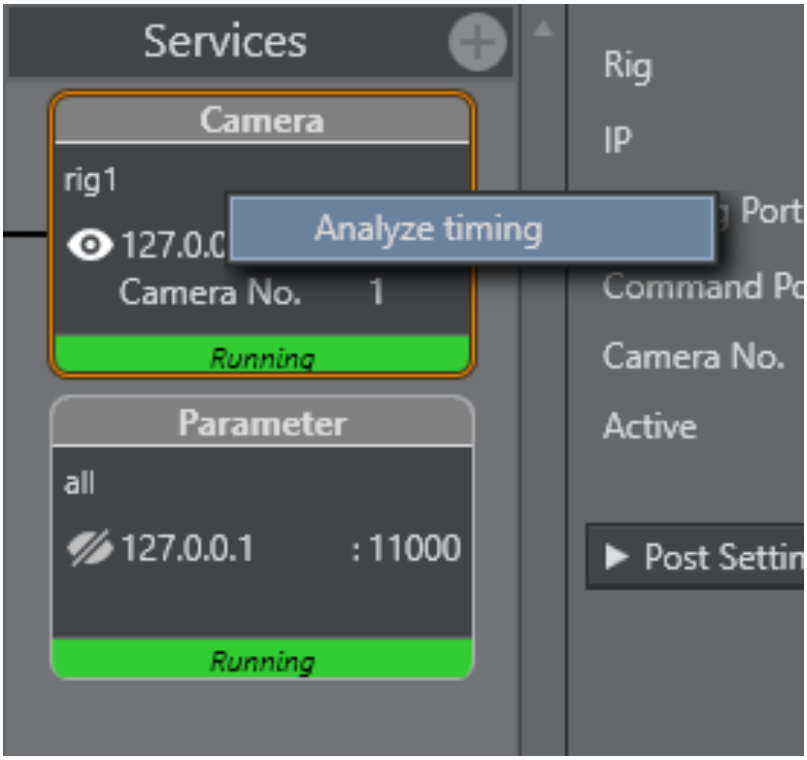
This adds a new element to the **Services** panel, called **TrackingTiming**.

2. Click on the **TrackingTiming** Service element. In the Properties panel, make sure that:
   - The IP of the local PC is inserted, and
   - The service is set to **Active**.

To Analyze a Service Time

1. Right-click on the Service System that is to be analyzed and select **Analyze Timing**.

Copyright © 2026 Vizrt



This adds a new element to the **Services** panel, called **TrackingTiming**.

2. Click on the **TrackingTiming** Service element. In the Properties panel, make sure that:
   - The IP of the local PC is inserted, and
   - The service is set to **Active**.

To View Timing Information

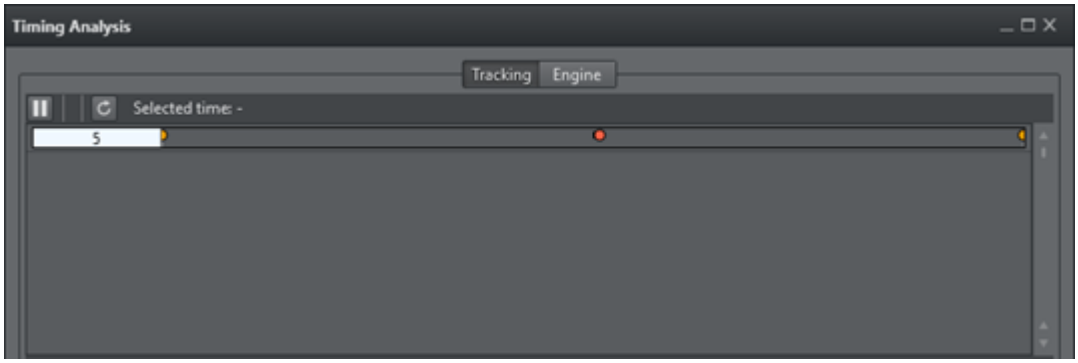
1. In the Configuration Panel, click the **View Timing Analysis** button . In a Timing panel with a correct data stream:

- **Purple line:** Shows the time when the Tracking Hub has inserted a data package from the tracking system into the data pool.
- **Red Dot:** Marks the moment when Tracking Hub has finished sending the tracking data to the Viz Engine.

Upper Section

In the upper section, the first line is the timing of live tracking data.

Copyright © 2026 Vizrt



These timestamps are shown:

- **Sync:**
  - **Start RX:** Start time, when data is received from the tracking system.
  - **End RX:** End time, when data stops from the tracking system.
  - **Start Ins:** Start time, when Tracking Hub inserts received data into the parameter pool.
  - **End Ins:** End time, when Tracking Hub has finalized inserting data.
- **Orange Dot:** Sync Timestamp
- **Blue:** The time between **Start RX** and **End RX**

> **Note:** The time between **Start RX** and **End RX** can be very short and thus almost invisible if network communication is in use.

- **Purple Line:** The time between **Start Ins** and **End Ins**
- **Line 2:** Shows when the live tracking data is processed and sent to the Viz Engine.
- **Start Calc:** The start time. When tracking data is calculated and prepared for a Viz Engine, this time is defined from send delay. After this delay has passed the Tracking Hub starts with this procedure for calculation and sending data.
- **End Calc:** The end time. When tracking data preparation is finished.
- **End Send:** Time when data has been send to the Viz Engine.
- **Green Line:** Time between **Start Calc** and **End Calc.**
- **Red Dot:** End Send Timestamp

Lower Section

The lower part shows the last ten second history of the timing. Three timestamps are shown.

Copyright © 2026 Vizrt



Note: Only three timestamps are shown to make the recognition of the timing behavior easier.

- **End Ins:** As time, when receiving is finished, shown in purple (1).
- **Start Calc:** When data processing starts, shown in green (2).
- **End Send:** When Tracking Hub work is finished for this field, shown in red (3).

To get the optimal timing, the **Start Calc** time must be later than the **End Ins** time, for every field. To achieve this, the send delay has to be adjusted correctly. The white line shows the delay. This line should be the absolute border between **End Ins** (purple line) and **Start Calc** (green line). To change the delay, use drag and drop to move the white line. If the delay can not be adjusted correctly, this can be for several reasons, for example, tracking data is received too late.

Note: This is not too bad, but keep in mind, that this result is in a tracking delay of one field.

This doesn't matter so much, because in normal production you need more the one field tracking delay.

Copyright © 2026 Vizrt

6.9.8 Lens Range Calibration

This section details how to calibrate the Lens Range.

1. First, collect the minimum and maximum Values of Zoom and Focus:
    a. In the Tracking System properties panel, click the **Calibration** button:



    b. Go to the broadcast camera and move Zoom and Focus to their upper and back to their lower limits, at least two times. Observe that the system receives the full set of data:

Copyright © 2026 Vizrt



To check if the calculation is correct, the corresponding Min/Max values, of the Lens encoder, are shown in the **Zoom** and **Focus** fields.

Copyright © 2026 Vizrt



Re-using Lens Ranges from Older Version Lens Files

The lens range in Tracking Hub is from `0` to `1`. For setups used with software preceding the Tracking Hub, such as Viz IO, the lens files may have used lens ranges specified slightly differently, such as from `0.01` to `0.99`. The Tracking Hub is capable of adapting to such a lens range. To achieve this, the `lenserange` parameter in the studio configuration file needs to be entered manually.

Warning: Editing the configuration manually can lead to software malfunction, and should only be done by qualified professionals.

6.9.9 Modify a Rig

If a Tracking System parameter does not equal a Viz Engine parameter, a Viz Engine parameter can be modified to adapt to the Tracking System parameter.

To modify a Rig

1. Right-click on a Tracking System. Select **Connect to** a rig.
2. The **Tracking parameter routing** window opens.

Copyright © 2026 Vizrt



- The **Tracking System** panel shows all the parameters that can be tracked.
- The **Rig** panel shows the Rig connections.

3. If a Tracking System parameter does not equal a Viz Engine parameter, a Viz Engine parameter can be modified to adapt to the Tracking System parameter. Click on a Viz Engine parameter and drag to its new position, as required. If a red cross shows, after a parameter has been modified, it means that the connection is still valid, but the connection is now a cross-over connection and not a straight one.

Copyright © 2026 Vizrt

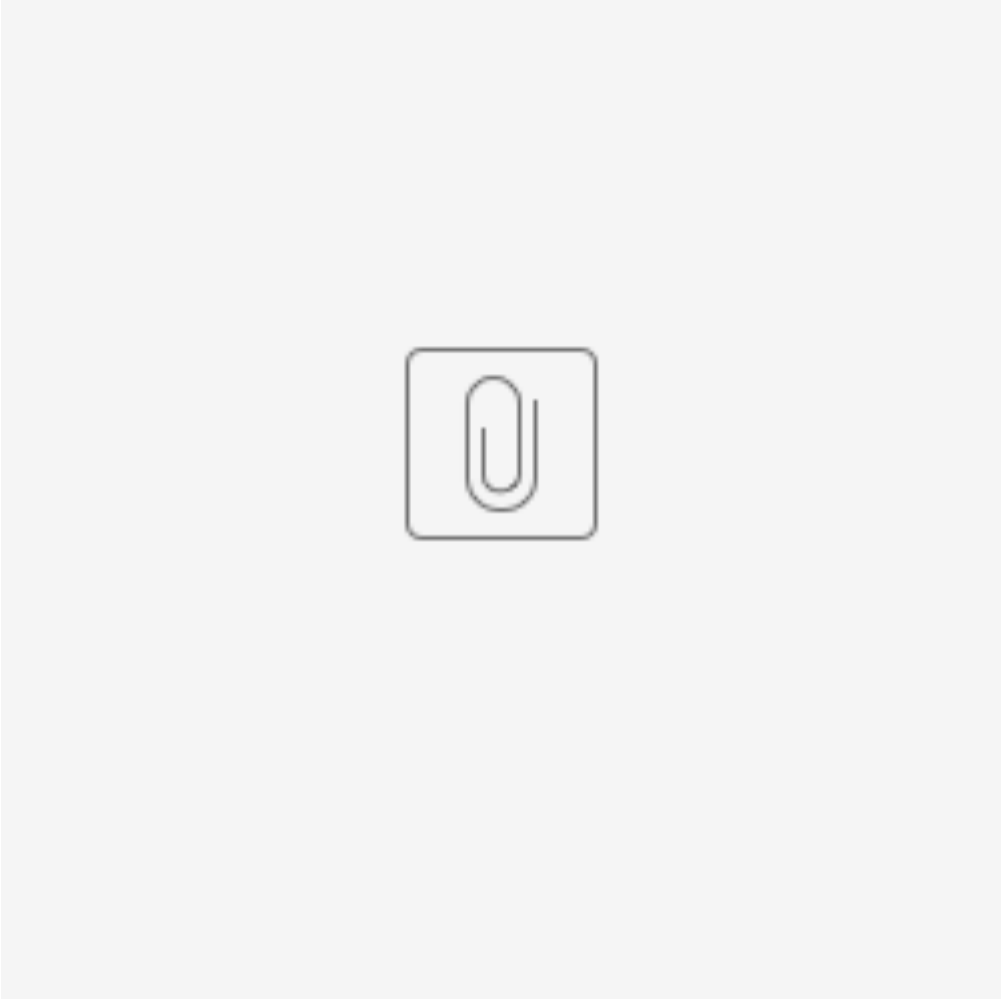


4. In the **Offset** column, define an offset to each parameter. The values are given in centimeters or degrees (value with dot). The values are stored when the field loses focus.
5. In the **Delay** column, enter how many fields sending of the parameter should be delayed.
6. Click on a box under **Connected** to connect a parameter through the rig.
7. Click **OK**.

No Tracking Parameters

If the **Tracking parameter** panel empty, there are no tracking parameters delivered from the tracking system.

Copyright © 2026 Vizrt



1. Check what the icon of the tracking system shows (disconnected or connected, color, etc.).
2. Check all connectors until the required connections show in the Tracking Parameters panel.
3. Go to To Modify a Rig.

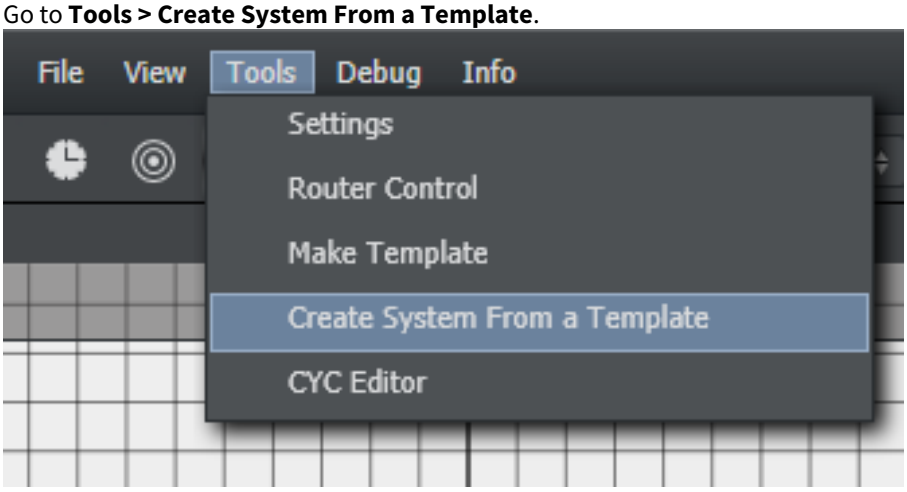
Copyright © 2026 Vizrt

6.10 Use of Templates

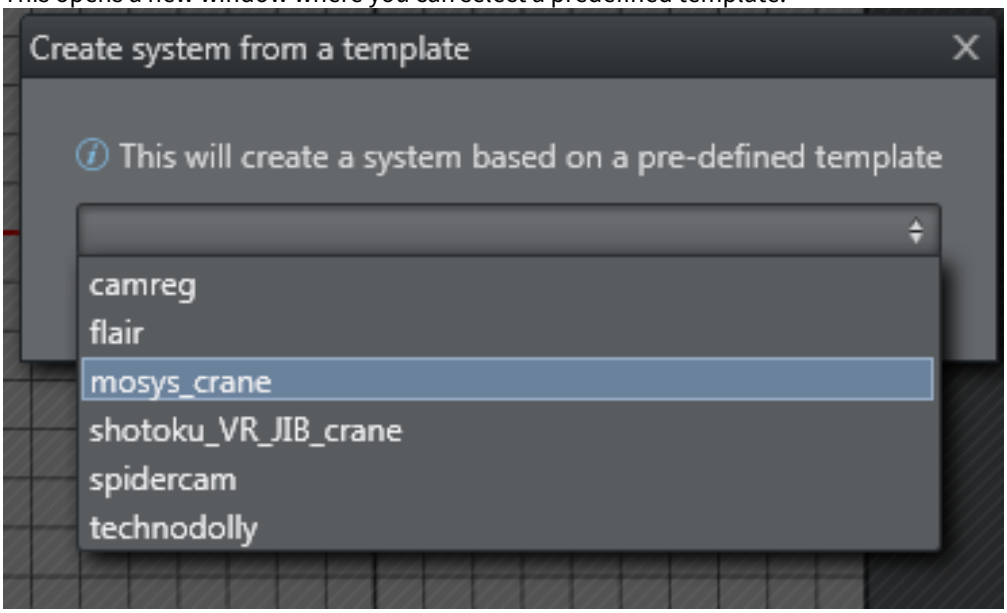
You can use templates for an easy and fast configuration of several studios. There are several pre-made templates available in `C:\ProgramData\vizrt\VizTH\Templates`. You can also create your own templates. However, you must copy them to that directory on a new installation.

6.10.1 Using Existing Templates

1. Go to **Tools > Create System From a Template**.

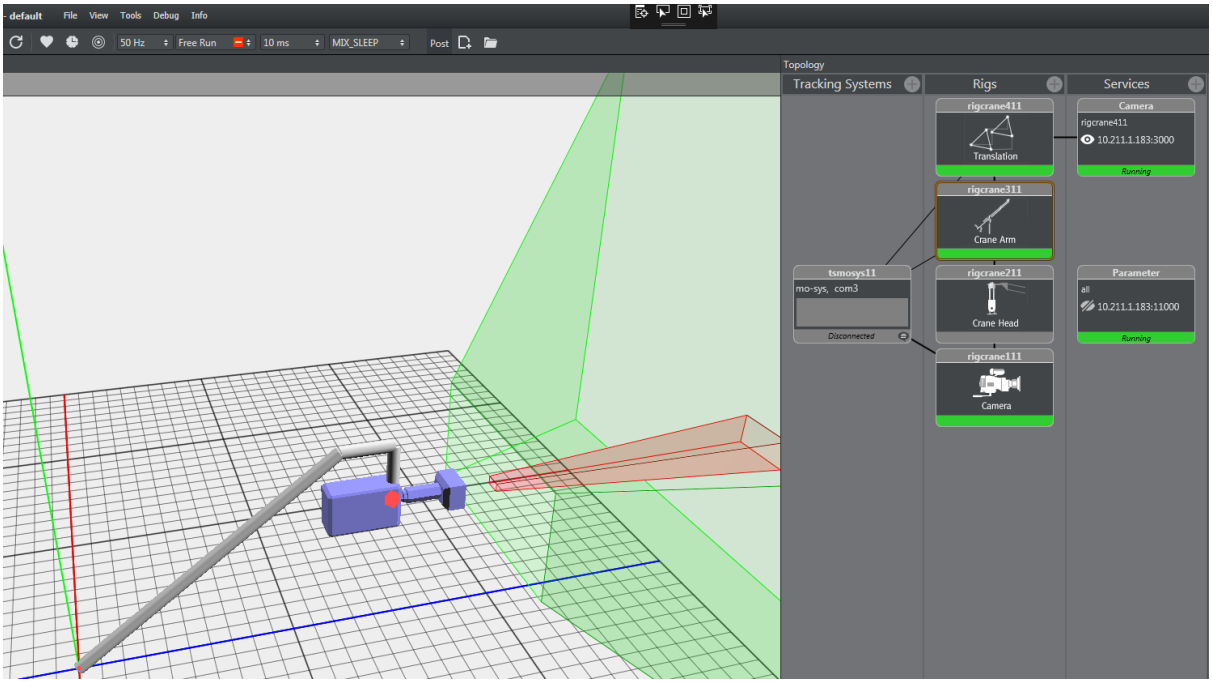


This opens a new window where you can select a predefined template.



2. After clicking **Create**, you get a complete system consisting of at least one Tracking System, one rig and one service. Depending on the template, it can include several Tracking Systems and rigs.

Copyright © 2026 Vizrt

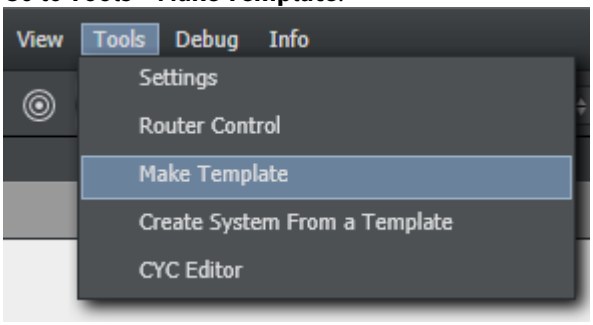


3. Configure the individual settings as applicable:
   - Interface of the tracking system.
   - Set the correct COM or UDP port.
   - Insert the correct IP address for the Viz Engine.

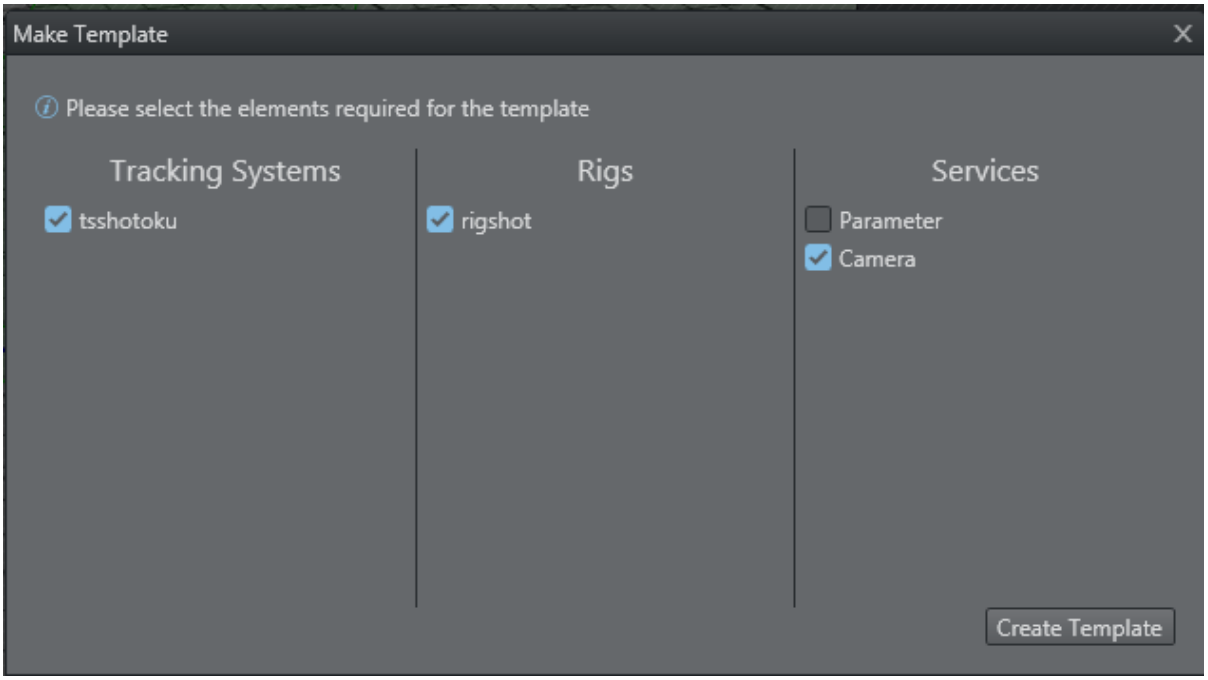
6.10.2 Creating New Templates

If you want to reuse a complete system, you can create your own template for it.

1. Go to **Tools > Make Template**.



This opens a new window where you can select the parts for your template:



Important: Do not select **Parameter** in the **Services** column!

2. When you have selected your items, click Create Template and enter a name for the template. Then click Save As. This saves a new template in the `C:\ProgramData\vizrt\VizTH\Templates` directory, in a file called `XML_TH_Temp_your_name.xml`.

Copyright © 2026 Vizrt

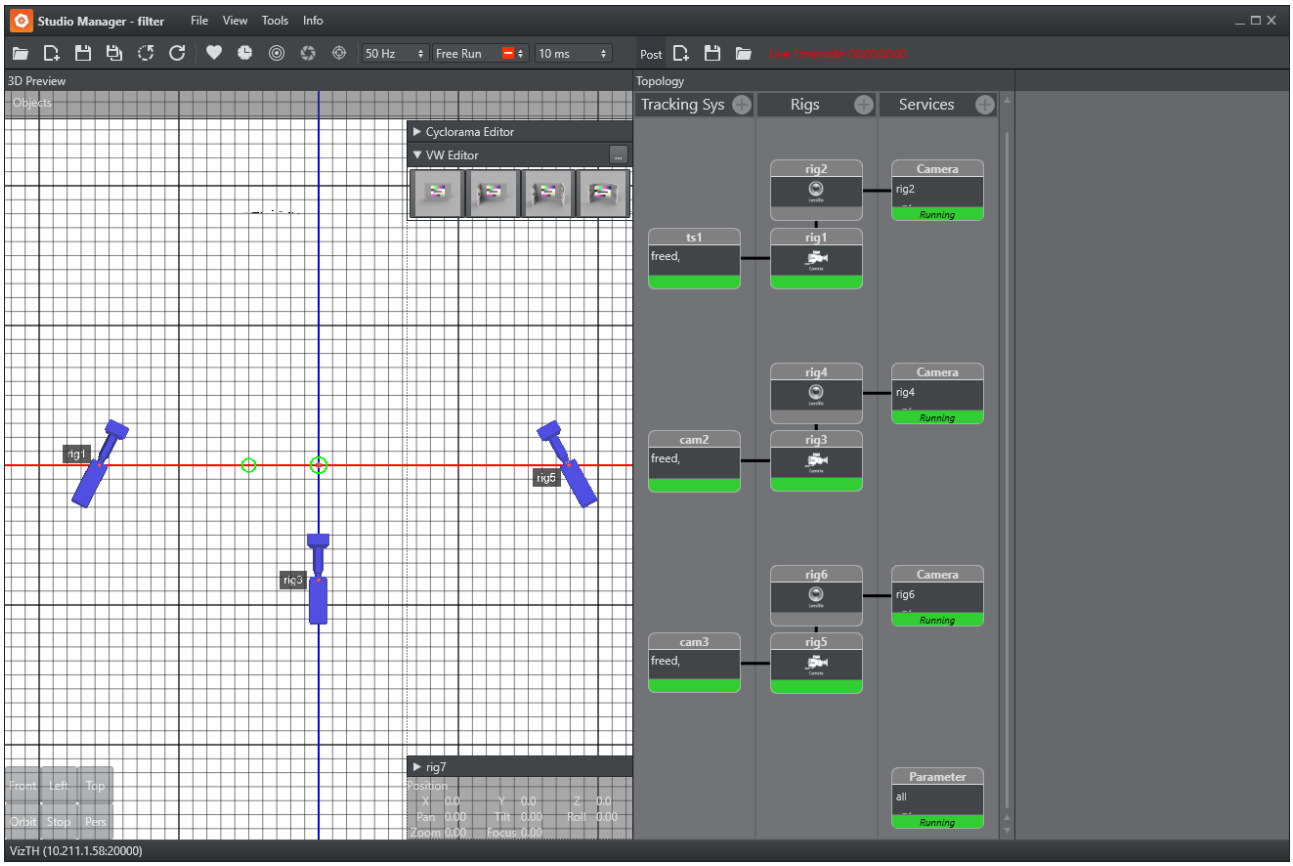
6.11 Studio Manager Camera Filter

Studio Manager normally displays the whole configuration of the Tracking Hub it connects to. Where one Tracking Hub serves several studios, an operator therefore sees — and can change — cameras and rigs that do not belong to them.

The **camera filter** restricts the display to a chosen list of rigs. It serves the purpose the `computers.txt` file had in VizIO. Each studio gets its own filter file, and Studio Manager is started with the name of the file.

Available from	Studio Manager 1.9.0
Jira	VIZVS-3492
Scope	Display filter only — see Limitations

6.11.1 Quick start



1. Create a file in `C:\\ProgramData\\vizrt\\StudioManager\\`, named after the studio, for example `studio1.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<CameraFilter Version="1.0">
  <Rigs>
    <Rig Name="rig1" />
    <Rig Name="rig5" />
  </Rigs>
</CameraFilter>
```

2. Start Studio Manager with the name of that file:

```
StudioManager.exe -camerafilter studio1
```

3. Studio Manager displays only `rig1`, `rig_crane` (camera 2) and `rig_handheld` (cameras 3 and 4), together with everything belonging to them. An orange banner across the top of the window states that the configuration is only shown in part.

Without the startup argument Studio Manager behaves exactly as before and displays everything.

6.11.2 Setting up a filter

Where the file goes

Filter files live next to Studio Manager's other configuration files:

```
C:\ProgramData\vizrt\StudioManager\
```

This is the folder that already holds `Settings.xml`, `ColorCorrection.xml` and `AISettings.xml`. A path is deliberately not accepted on the command line — one file per studio lives in this folder, and each operator gets a shortcut naming their own.

The startup argument

```
StudioManager.exe -camerafilter <name>
```

`<name>` is the file name, with or without the `.xml` extension. `-camerafilter studio1` and `-camerafilter studio1.xml` are equivalent.

The argument name is not case sensitive. These spellings are all accepted:

Copyright © 2026 Vizrt

Form	Example
Separate value	`-camerafilter studio1`
With `=`	`-camerafilter=studio1`
With `:` and slash	`/camerafilter:studio1`
Double dash	`--camera-filter=studio1`
Aliases	`-camera-filter`, `-visiblecameras`, `-visible-cameras`

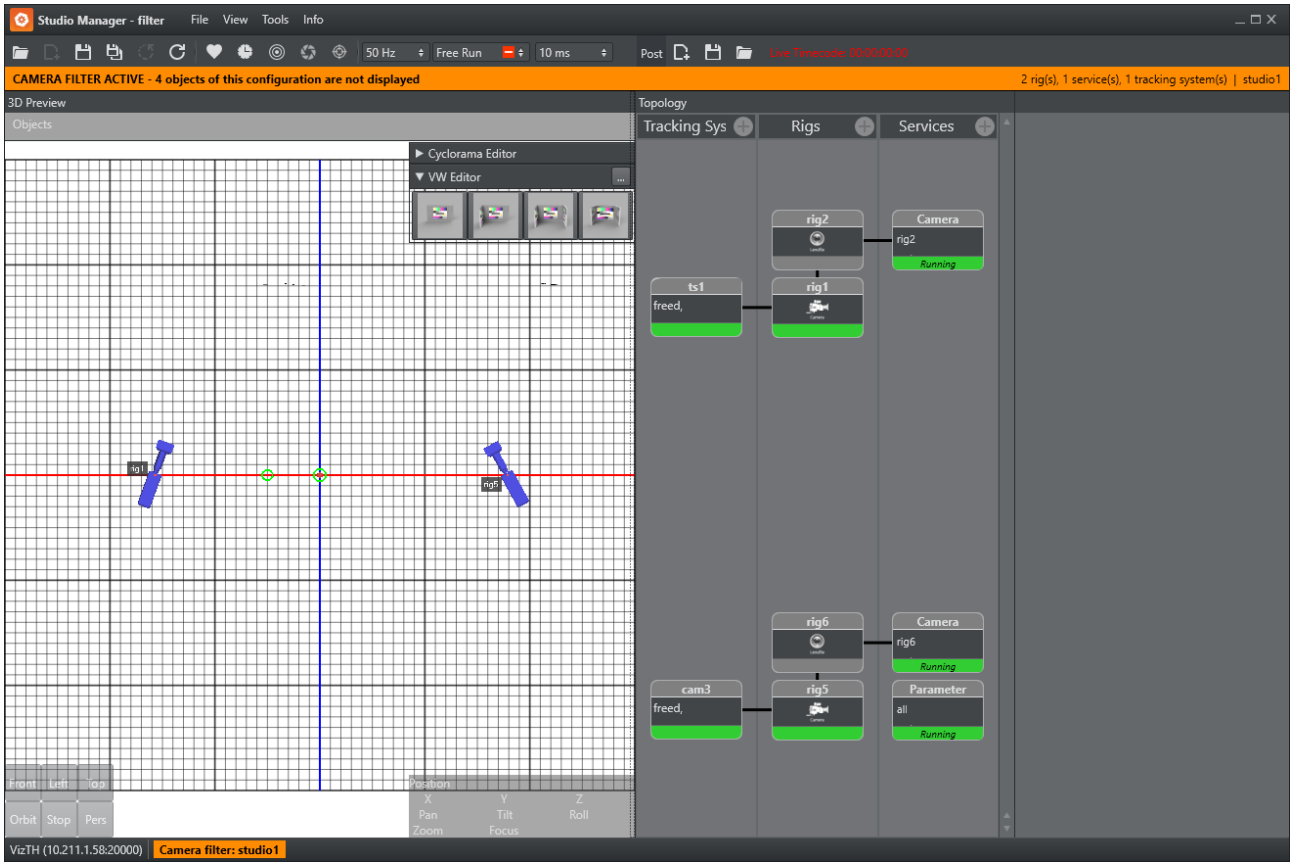
Rolling it out to operators

Give each operator a desktop shortcut to `StudioManager.exe` with their studio's filter appended to the target, for example:

```
"C:\Program Files\vizrt\Viz Virtual Studio\StudioManager\StudioManager.exe"
 -camerafilter studio1
```

The installer does not create filtered shortcuts and does not deploy filter files — both are a deployment step per site.

Copyright © 2026 Vizrt



6.11.3 File format

Elements

Element	Attribute	Meaning
`<CameraFilter>`	`Version`	Root element. Required.
`<Rigs>`		Optional grouping element.
`<Rig>`	`Name`	Name of the rig to display. Required.
	`Cameras`	Optional. Comma-separated camera numbers, **1-based, exactly as the UI shows them**. `*` or omitting the attribute means all cameras of that rig.

Copyright © 2026 Vizrt

Element	Attribute	Meaning
`<TrackingSystems>`		Optional grouping element.
`<TrackingSystem>`	`Name`	Optional. Keeps a tracking system visible even when no displayed rig is connected to it yet.

Full example

```
<?xml version="1.0" encoding="utf-8"?>
<!--Studio 1-->
<CameraFilter Version="1.0">

  <Rigs>
    <Rig Name="rig1" />                        <!--hard camera, all cameras-->
    <Rig Name="rig_crane"     Cameras="2" />   <!--crane, only camera 2-->
    <Rig Name="rig_handheld"  Cameras="3,4" /> <!--handheld, cameras 3 and 4-->
    <Rig Name="rig_steadycam" Cameras="*" />   <!--same as no Cameras attribute-->
  </Rigs>

  <TrackingSystems>
    <TrackingSystem Name="ts_stype" />
  </TrackingSystems>

</CameraFilter>
```

The parser is forgiving

These files are edited by hand on site, so:

- element and attribute names are matched **case insensitively** — `<rig name="...">` works as well;
- the `<Rigs>` and `<TrackingSystems>` grouping elements are a convenience, not a requirement — `<Rig>` and `<TrackingSystem>` elements are picked up from anywhere in the document;
- rig and tracking system names are compared case insensitively and surrounding spaces are trimmed;
- an entry that cannot be understood — a `<Rig>` without a `Name`, a `Cameras="abc"` — is written to the log **with its line number** and skipped. One bad entry does not stop the rest of the file from working.

You do not list the whole rig chain

Naming a rig automatically brings in the rigs it is chained to — its lens file, arm, and so on. A camera rig is presented in the UI as a whole chain of lattices, so listing the camera rig is enough.

Copyright © 2026 Vizrt

6.11.4 What is displayed and what is hidden

The filter is applied while the configuration is read from Tracking Hub, so hidden objects never reach any part of the UI: not the topology, not the 3D viewport, not the property panel, and not the rig drop-down lists.

Column	Rule
Rigs	Displayed if the file lists the rig, **or** if it is chained to a listed rig.
Services	A service bound to a rig (camera, TC camera, multi camera, videowall, object, multi object, MoCap, TrFlow, NDI) is displayed if one of the rigs it refers to is displayed and its camera number is allowed. Services **not** bound to a rig — timing, parameter, state, timecode — are infrastructure and are always displayed. A rig-bound service not yet assigned to any rig is also kept.
Tracking systems	Displayed if one of the displayed rigs is connected to it. A `<TrackingSystem>` entry forces one to stay visible even when nothing is connected to it yet.

Camera numbers restrict only the rigs the file lists **explicitly**, and only entries that actually carry a camera number. An object service on a rig listed as `Cameras="3"` is therefore not hidden by the `3`.

6.11.5 How you can tell a filter is active

Where	What you see	
Banner	An orange strip across the full width of the window, below the tool bars and above the three topology columns: *"CAMERA FILTER ACTIVE — 12 objects of this configuration are not displayed*", with the breakdown *"3 rig(s), 8 service(s), 1 tracking system(s) \	studio1 "* on the right.
Status bar	The same orange highlight at the bottom of the window, naming the filter.	
Tooltips	Hovering either one lists the filter file, the rigs it names, the per-category counts, and a reminder that the hidden objects are unchanged in Tracking Hub.	
Log panel	The startup arguments and the state of the filter are logged when the main window opens, including a warning if an argument was passed that Studio Manager did not recognise.	

Copyright © 2026 Vizrt

Before a Tracking Hub is connected nothing has been filtered yet, so the banner appears without counts. The counts are updated every time the configuration is read.

6.11.6 Commands disabled while a filter is active

Three commands act on the whole Tracking Hub, including the rigs the operator cannot see. Leaving them enabled would let a display filter destroy another studio's setup, so they are greyed out:

Command	Why
Reset Configuration	Wipes the entire Tracking Hub configuration.
New Configuration	Resets as well, then asks for a new name.
Quit Tracking Hub	Ends the Tracking Hub session for every connected client.

Hovering a greyed-out button explains the reason. Everything else stays available: Save, Save As, Synchronize Backup, and all editing of the displayed objects.

6.11.7 Limitations

This is a display filter, not access control. It hides objects from this Studio Manager instance. It does not prevent another instance — or another operator running without a filter — from changing the same objects. Enforcing that would require support on the Tracking Hub side.

The filter never changes the Tracking Hub configuration. Studio Manager sends incremental commands and lets Tracking Hub save its own configuration; it never uploads a whole configuration. Hiding an object therefore cannot delete it, and saving while filtered does not drop the hidden objects.

Objects you create stay visible until you restart. A rig created — or renamed — through Studio Manager while a filter is active remains displayed for the rest of the session, even though the file does not list it. Hiding something you have just created would be confusing. The filter file is never written to, so after a restart the display is exactly what the file says. Add the new rig to the file to keep it.

Rigs are drawn at their own slot positions. Rigs keep the slot index Tracking Hub assigns, so a filter that selects a few rigs out of many leaves empty rows between them in the topology.

6.11.8 Troubleshooting

Everything is still displayed

The filter is not active. Check the status bar at the bottom of the window — when a filter is in effect it names the filter in orange. If it does not:

- the startup argument is missing from the shortcut, or
- the argument is misspelled. An unrecognised argument is reported as a warning in the log panel, along with the full command line Studio Manager was started with.

Copyright © 2026 Vizrt

Studio Manager refuses to start

A filter was requested but cannot be used. The message names the reason and shows the expected file path and the file format. Studio Manager deliberately does **not** fall back to displaying everything, because that would silently defeat the purpose of the filter. Causes:

Message	Cause
does not exist	No such file in `C:\\ProgramData\\vizrt\\StudioManager\\`. The message shows the exact path it looked for.
is not valid XML	A syntax error, e.g. an unclosed tag. The XML parser's own description is included.
has no `<CameraFilter>` root element	The file is XML, but not a camera filter.
does not list a single rig	No usable `<Rig>` entry was found. Check the log for skipped entries.

"Camera filter does not match the configuration"

The loaded configuration contains rigs, but not one of the rigs the filter names, so Studio Manager would display nothing at all. The dialog lists the rigs the filter names and the file to correct. Usual causes: a typo in a rig name, or the wrong filter for the configuration that was loaded.

Rig names that the configuration does not contain are otherwise only a warning in the log, so one filter file can be used with several configurations.

A rig or camera I expected is missing

- Check the exact spelling of the rig name against the name shown on the tile in the **Rigs** column.
- If the rig is listed with a `Cameras` attribute, only those camera numbers are displayed. Camera numbers are 1-based, as shown in the UI.
- Look in the log panel — every hidden object and every skipped file entry is logged.

Copyright © 2026 Vizrt

6.12 Engine Color Correction

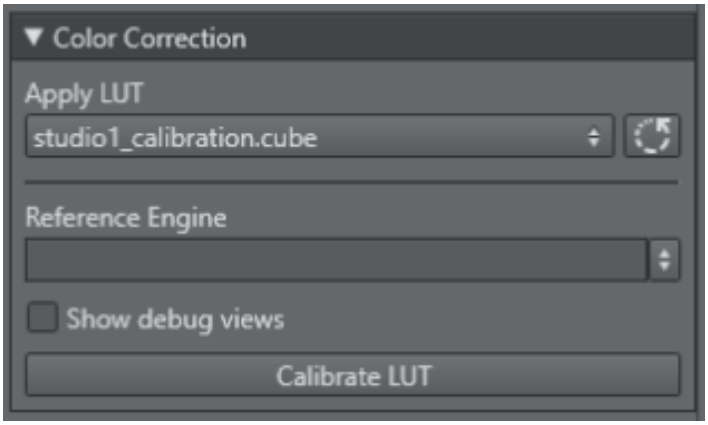
Dual-engine LED wall / camera color calibration. Studio Manager drives reference colors on the **wall (reference) engine**, reads them back through the camera on the **capture (AR) engine**, fits a 3D LUT (`.cube`) that describes the wall→camera color change, and (optionally) verifies and refines that LUT against the AR engine. The resulting LUT is stored in Tracking Hub (TH) and applied to the camera/AR service.



6.12.1 1. Overview

Term	Meaning
Wall / Reference engine	Displays the reference color patches full-screen.
Capture / AR engine	Reads the camera (LIVE input); later renders the AR test scene.
Forward LUT	`sent → captured` : the detected change between the wall and the wall as filmed.
Inverse LUT	`captured → sent` : the correcting mapping (reference/analysis).
Combined LUT	The final LUT (equals the forward LUT unless the second layer refined it). Saved for both scene color spaces: BT709 sRGB and BT709 Linear.
Second layer (AR refine)	Optional: builds an AR test scene, applies the LUT, and refines it.

Copyright © 2026 Vizrt

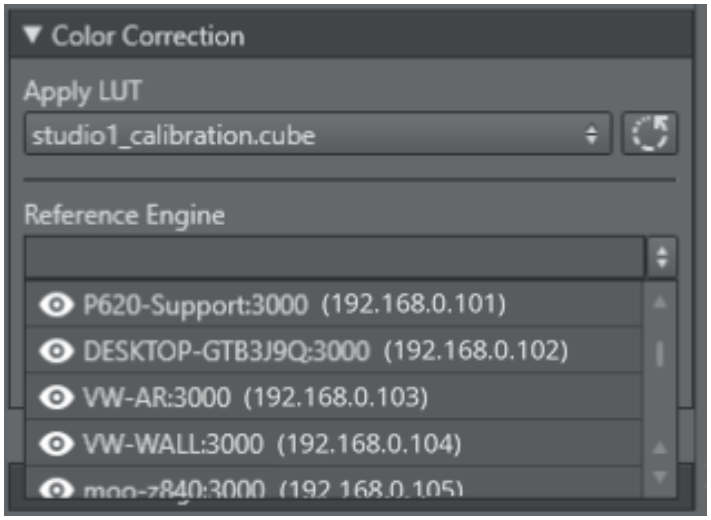


6.12.2 2. Prerequisites

- Both engines reachable on the network and **ON-AIR**.
- Tracking Hub connected to the camera/AR **service** (required to store/apply the LUT).
- The camera framing the LED wall; exposure/iris/white-balance set for the shot.
- Reference and camera engine on **different hosts** (wall vs capture).

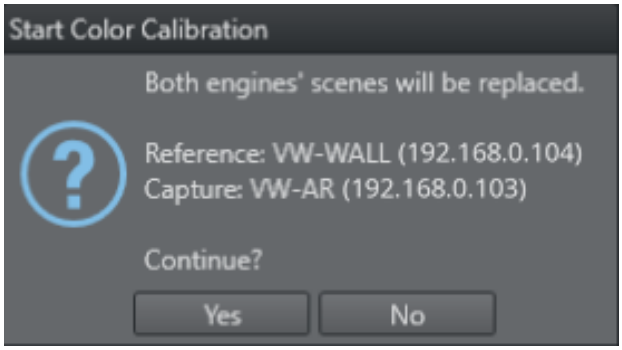
6.12.3 3. Running a calibration (operator workflow)

1. Open the **camera/AR service** and go to the **Properties → Color Correction** expander.
2. In **Reference Engine**, enter or pick the wall engine (the dropdown lists engines seen on the network). The service's own IP is the capture/AR engine.
3. (Optional) Tick **Show debug views** only if you want to step through each second-layer pass.
4. Click **Calibrate LUT** and confirm the dialog.



Copyright © 2026 Vizrt

5. Confirm the creation of test scenes.

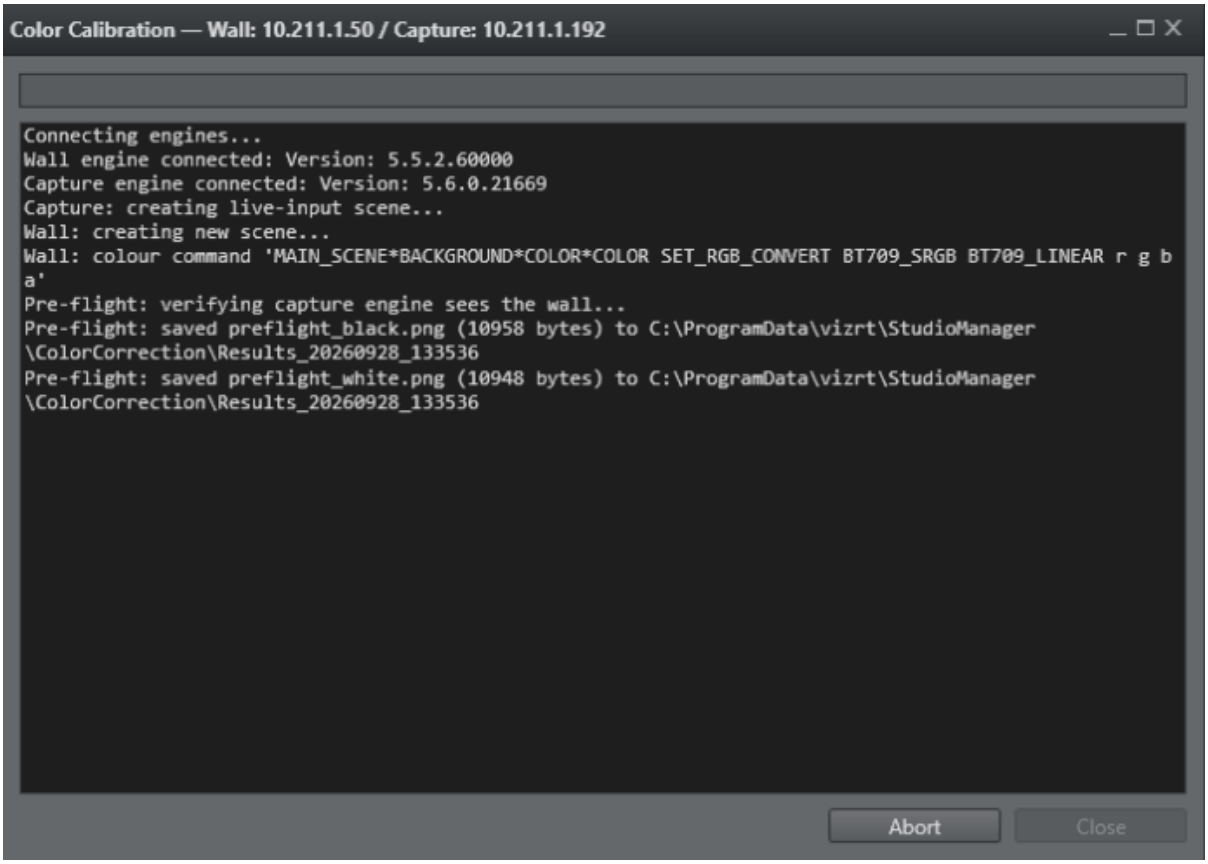


The process requires the creation of 2 test scenes on the assigned Engines. Currently opened scenes will be closed without saving.

The **calibration log window** opens and streams progress:

- Connects both engines, creates scenes.
- **Pre-flight camera check** (5 sample areas must see the wall).
- Captures the reference patch set (wall shows each color, camera reads it back): about 160 patches made of 24 ColorChecker-style colors, a 5×5×5 RGB cube and a 17-step grey ramp. Patches clipped by the camera are reported as a warning.
- Fits the LUTs (3D RBF, optionally on top of a per-channel tone curve through the grey ramp; smoothing and curve are chosen by leave-one-out ΔE2000) and writes the results folder.
- If the second layer is enabled, refines against the AR engine.

Copyright © 2026 Vizrt

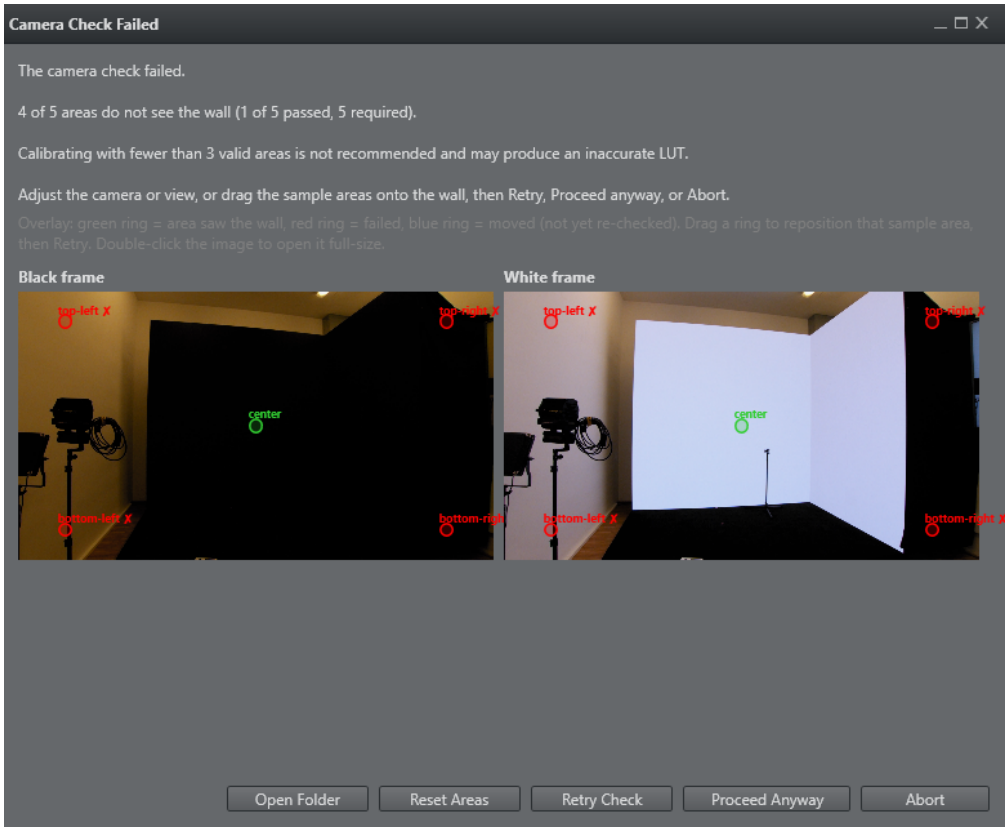


3.1 If the camera check fails

If fewer than the required number of sample areas see the wall, a dialog shows the captured black/white snapshots with **green (pass) / red (fail)** rings and offers:

- **Open Folder** — open the results folder with the check snapshots.
- **Reset Areas** — move the sample areas back to the default corners and centre.
- **Retry Check** — re-run the check after fixing framing/exposure or moving the areas.
- **Proceed Anyway** — continue despite the failure (not recommended below 3 valid areas).
- **Abort** — stop.

Copyright © 2026 Vizrt

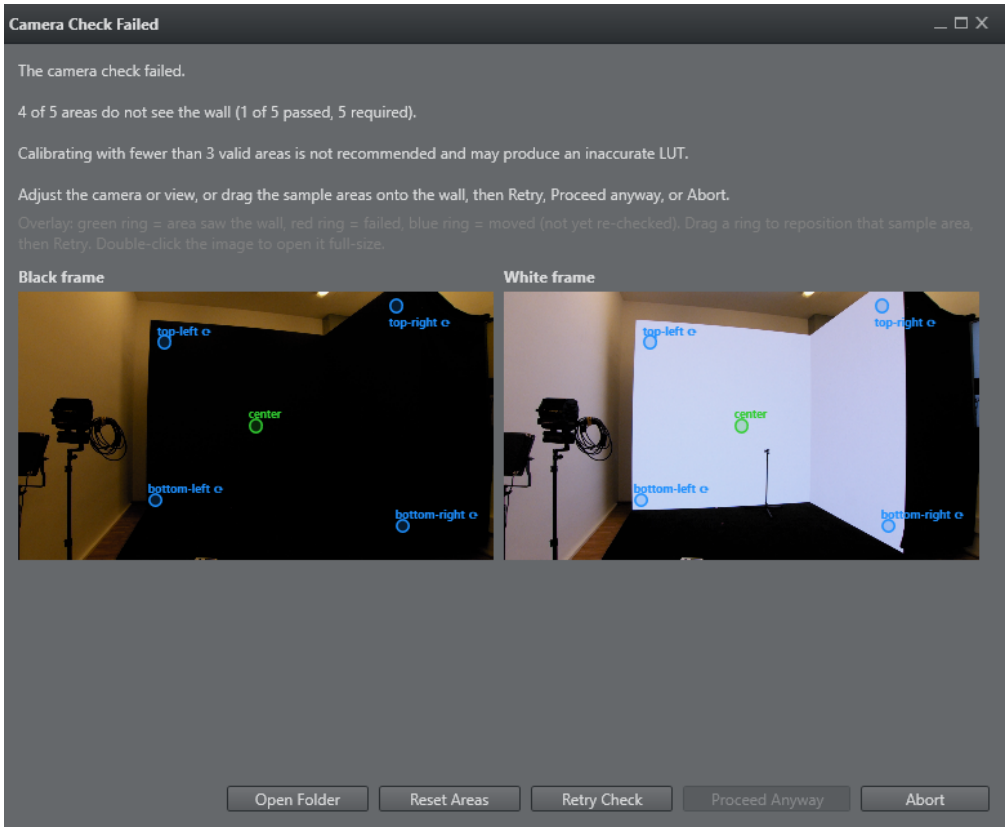


Moving the sample areas

If the wall does not cover the default positions, drag the sample areas onto the wall in the camera frame shown in the dialog, then use **Retry Check**. **Proceed Anyway** is disabled until the moved areas have been checked.

Moved areas are saved as the new default positions (`SamplePoints` in the configuration file); **Reset Areas** moves them back.

Copyright © 2026 Vizrt



Failure reasons

An area fails for one of these reasons (the log names the reason per area):

- **Black too bright** — the black level is above `CameraCheckAmbientMax`. Reduce room light or lower the exposure.
- **White too dark** — increase camera exposure/iris or the wall brightness.
- **Contrast too low** — the black→white delta is below `CameraCheckThreshold`.
- **White clipping, contrast still low** — the area is most likely not on the wall; move it.

Most failures come from the camera setup: check that iris, ISO/gain and white balance suit the production, then use **Retry Check**. If it still fails, abort, adjust `CameraCheckThreshold` / `CameraCheckAmbientMax` in the configuration file (see 7. Configuration file) and start the calibration again. `CameraCheckMinCircles` sets how many of the 5 areas must pass (default 5). **Proceed Anyway** skips the check for this run.

3.2 Saving the LUT

When the run completes, the **Save** panel appears. Enter a base name, pick the assignment in the dropdown and click **Save to TH**. The LUT is always written for both scene color spaces: <name>_bt709_srgb.cube (33^3) and <name>_bt709_linear.cube (49^3, more grid nodes in the shadows). Both files are saved to the results folder and uploaded to TH; the dropdown only decides which one is assigned to this service:

- **Assign BT709 sRGB** — for services whose scenes use the BT709 sRGB working color space.
- **Assign BT709 Linear** — for services whose scenes use BT709 Linear.

Copyright © 2026 Vizrt

- **Don’t assign** — upload both files only; the service keeps its current LUT.

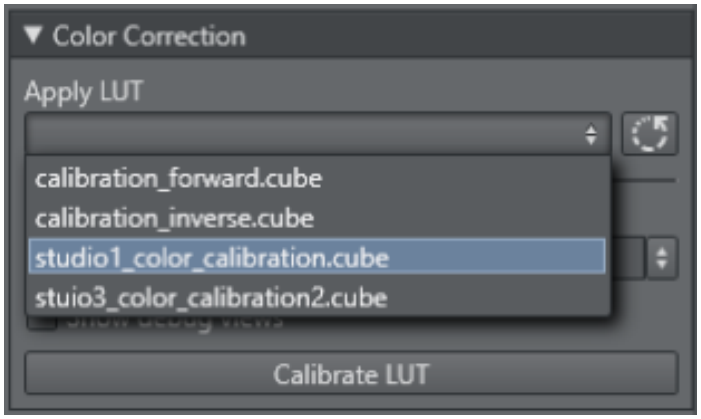
The dropdown remembers the last choice. The LUT that is not assigned stays on TH and can be picked later from the Color Correction LUT list (see 4. Applying / switching LUTs).



⚠️ *If you close the log window **without** saving, a warning appears ("The calibrated LUT has not been saved to TH. Close without saving?"). Choose **No** to go back and save.*

6.12.4 4. Applying / switching LUTs

The Color Correction panel lists the LUTs available on TH for the service. Selecting one applies it immediately (including the first pick). The list is re-read from TH after each save. Use **Clear** to remove the LUT from the engine.



6.12.5 5. The second layer (AR refine) — optional

When `EnableSecondLayer = true`, after the wall measurement the tool:

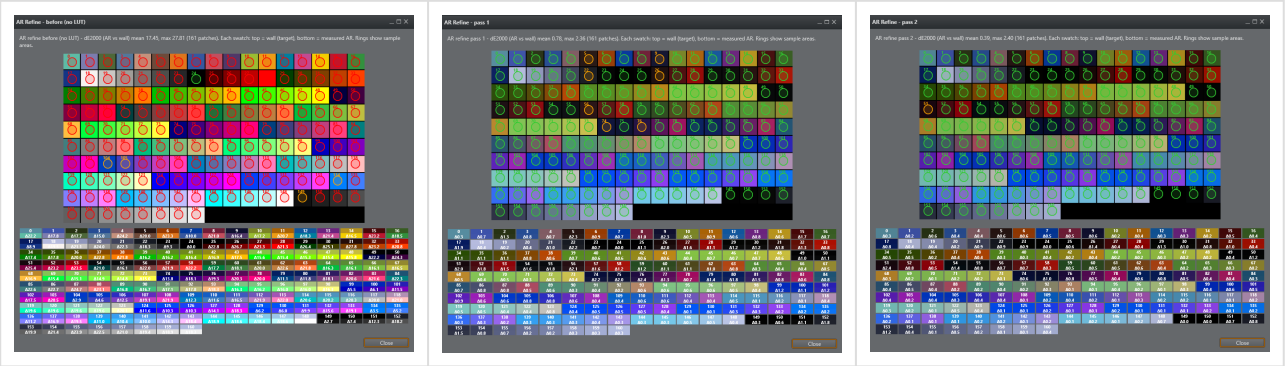
1. Builds a **matte scene** (full-screen red rectangle).
2. Builds an **AR test scene**: render sequence `MULTILAYER_XR`, links the matte scene, enables the LUT color correction, and lays out the measured reference colors as a grid of rectangles (the grid size follows the patch count). LUTs for this scene are sampled in `ArSceneColorSpace` (BT709 Linear).
3. Iterates: apply candidate LUT → measure the AR grid → diff against the recorded wall colors → refit → repeat until the worst-patch ΔE2000 is below `RefineThresholdDE2000`, the mean ΔE2000 stops improving, or `RefineMaxPasses` is reached. The pass with the lowest mean ΔE2000 is kept and applied to the AR engine for review at the end.

Copyright © 2026 Vizrt

If **Show debug views** is on, a dialog appears before the first pass (no LUT) and after every pass; close it to continue. With it off, the passes run automatically.

Each debug view shows:

- **Header** — the pass and the ΔE2000 between the AR output and the wall (mean and max over all patches).
- **Image** — the AR engine framebuffer with the rendered patch grid; the rings mark the sample areas.
- **Swatches** — per patch: wall target (top), measured AR color (bottom), patch number and ΔE2000.



The left view is the starting point: the AR test scene with an identity LUT, compared against the wall colors the camera recorded during the calibration. The middle view is the same scene after the first refine pass with the calibrated LUT applied: the mean error drops from 17.45 to 0.78, below the just-noticeable level of about 1. The right view shows pass 2, where the refined LUT halves the mean error again to 0.39; the worst patch stays at about 2.4, so the refinement continues until the mean stops improving or `RefineMaxPasses` is reached. The snapshots are saved in the results folder as `ar_pass0.png`, `ar_pass1.png`, …

6.12.6 6. Output files

Each run writes to `C:\\ProgramData\\vizrt\\StudioManager\\ColorCorrection\\Results_<yyyyMMdd_HHmmss>\\`:

File	Description
`calibration_forward.cube`	`sent → captured` — the first-pass / wall-vs-filmed change. Always saved.
`calibration_inverse.cube`	`captured → sent` — inverse/analysis.
`calibration_combined.cube`	The final LUT (= forward unless the second layer refined it), BT709 sRGB, 33³.
`<name>_bt709_srgb.cube`	Written by **Save to TH**: the final LUT for BT709 sRGB scenes (33³). Also uploaded to TH.

Copyright © 2026 Vizrt

File	Description
`<name>_bt709_linear.cube`	Written by **Save to TH**: the final LUT for BT709 Linear scenes (49³). Also uploaded to TH.
`calibration_review_ar.cube`	Second layer only: the final LUT sampled for the AR test scene, applied to the AR engine for review.
`calibration_identity.cube`	Second layer with **Show debug views** only: identity LUT for the before-refinement baseline.
`ar_pass*.png`	Second layer with **Show debug views** only: AR framebuffer snapshot per pass (`ar_pass0.png` = before refinement).
`measurements.csv`	Per-patch `Sent_RGB` and `Captured_RGB`.
`calibration_report.txt`	ΔE2000 statistics (avg / max / p95) for wall vs camera, LUT fit and leave-one-out; the chosen RBF smoothing and tone curve, worst patches and clipping warnings.
`preflight_black.png` / `preflight_white.png`	Camera check snapshots.
`snapshots\\snap_*.png`	Per-patch camera snapshots.

*Per-pass candidate LUTs (`calibration_refine_p*.cube`) are used during refinement and **deleted** at the end — they are not kept.*

6.12.7 7. Configuration file

Settings live in `C:\\ProgramData\\vizrt\\StudioManager\\ColorCorrection.xml`. The file is created
with defaults on first run; edit values and re-run (no rebuild needed). Unknown/missing elements fall back to the defaults below.

Copyright © 2026 Vizrt

```
<ColorCorrection>
  <CaptureLiveSource>BUILT_IN*IMAGE*LIVE1</CaptureLiveSource>
  <CaptureWidth>1920</CaptureWidth>
  <CaptureHeight>1080</CaptureHeight>
  <SampleRadius>30</SampleRadius>
  <SettleMs>2000</SettleMs>
  <AutoSettle>true</AutoSettle>
  <SkipCameraCheck>false</SkipCameraCheck>
  <CameraCheckThreshold>0.3000</CameraCheckThreshold>
  <CameraCheckMinCircles>5</CameraCheckMinCircles>
  <CameraCheckAmbientMax>0.1500</CameraCheckAmbientMax>
  <LutSize>33</LutSize>
  <WallPort>6100</WallPort>
  <CapturePort>6100</CapturePort>
</ColorCorrection>
```

Capture / measurement

Element	Default	Description
`CaptureLiveSource`	`BUILT_IN*IMAGE*LIVE1`	Camera LIVE input source on the capture engine.
`CaptureWidth`	`1920`	Framebuffer capture width (px).
`CaptureHeight`	`1080`	Framebuffer capture height (px).
`SampleRadius`	`30`	Radius (px) of the sample disc per area/patch.
`SettleMs`	`2000`	Fixed wall→camera settle per patch (used when `AutoSettle` is false).
`AutoSettle`	`True`	Auto-measure the camera round-trip delay and use it as the settle.
`WallPort`	`6100`	Reference (wall) engine TCP port.
`CapturePort`	`6100`	Capture (AR) engine TCP port.

Copyright © 2026 Vizrt

Pre-flight camera check

Element	Default	Description
`SkipCameraCheck`	`False`	Skip the black/white "sees the wall" check entirely.
`CameraCheckThreshold`	`0.3000`	Minimum black→white luminance delta for an area to pass.
`CameraCheckMinCircles`	`5`	Number of the 5 sample areas that must pass.
`CameraCheckAmbientMax`	`0.1500`	Maximum black-level (ambient) allowed per area.
`SamplePoints`	(not set)	Positions of the 5 sample areas (normalized 0..1), written when the operator moves them in the camera check dialog. Not set = corners and centre.

LUT / color space

Element	Default	Description
`LutSize`	`33`	3D LUT grid size for BT709 sRGB LUTs (33³). BT709 Linear LUTs always use 49³, the largest that fits TH's 3 MB file limit.
`LutColorSpace`	`BT709_SRGB`	Last choice of the save panel's assign dropdown (`BT709_SRGB` , `BT709_LINEAR` or `NONE`); preselects it next time. Set by the UI.
`PatchCubeLevels`	`5`	RGB cube levels per axis in the patch set (5 = 125 cube patches, about 160 patches in total). Below 2 = the legacy 32-patch set.
`PatchGreySteps`	`17`	Steps of the neutral grey ramp in the patch set.
`EnableToneCurve`	`True`	Allow a per-channel tone curve through the grey patches under the 3D RBF (used only when leave-one-out prefers it).

Copyright © 2026 Vizrt

Element	Default	Description
`WallColorConvertMode`	`1`	Wall color-space conversion: **1** = sRGB→Linear (`SET_RGB_CONVERT BT709_SRGB BT709_LINEAR`), **2** = flipped (Linear→sRGB), **3** = none (plain `COLOR SET`).

Second layer (AR refine) — optional

Element	Default	Description
`EnableSecondLayer`	`False`	Enable the AR test-scene refine/verify. Off = layer-1-only (no AR scene workflow).
`RefineThresholdDE2000`	`1.0`	Converge when the worst AR-vs-wall ΔE2000 drops below this (about 1 = just noticeable). Replaces the old RGB-distance key `RefineThresholdDE` .
`RefineMaxPasses`	`5`	Maximum refine passes.
`ArApplySettleMs`	`1500`	Wait (ms) after applying a LUT to the AR engine before measuring.
`ArRenderSequence`	`MULTILAYER_XR`	Render sequence set on the AR test scene (`BACKGROUND*ALPHA SET`). Empty = leave scene default.
`ArSceneColorSpace`	`BT709_LINEAR`	Working color space of the second layer's AR test scene; candidate, identity and review LUTs are sampled in it.

Not in the config file *(supplied per run): the* ***Reference Engine*** *host (from the UI) and the* ***capture/AR*** *host (the service IP). The* ***Show debug views*** *checkbox is set per run as well. The AR scene extent (740 × 420) and the 24 ColorChecker base colors are fixed in code; the patch set and the AR grid follow* `PatchCubeLevels` / `PatchGreySteps` .

Copyright © 2026 Vizrt

6.12.8 8. Troubleshooting

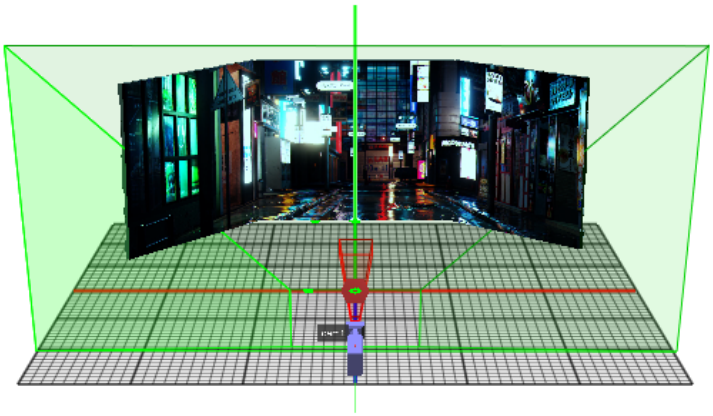
Symptom	Likely cause / action
Captured colors too bright / lifted blacks	Camera over-exposed or wrong color-space; compare `measurements.csv` shadow patches to a known-good run. Check camera iris/gain/white balance and `WallColorConvertMode`.
Camera check fails on some areas	Reduce room light, adjust exposure, or re-frame so the wall fills those areas; use **Retry Check**.
Wall colors render brighter than sent (e.g. 0.486 → 184 not 124)	Engine gamma-encoding the sent value — set `WallColorConvertMode = 1`.
LUT not applied after picking it	Fixed — the first dropdown pick now applies. The list is re-read from TH after each save.
Second-layer ΔE ~0 but AR doesn't match visually	Framebuffer measurement can't see the composite; verify on the actual AR output, not the engine framebuffer.

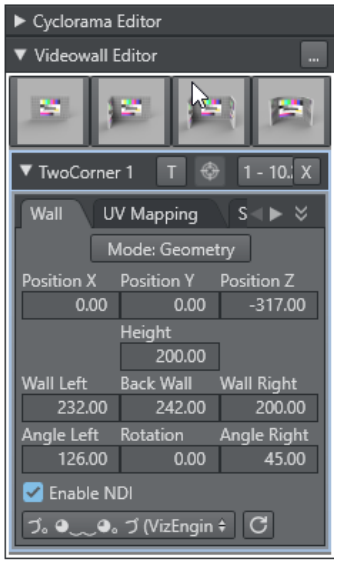
Copyright © 2026 Vizrt

6.13 Videowall Editor

The Videowall Editor (**VW Editor**) is the part of Studio Manager that describes the LED walls of a studio: where each wall stands, how big it is, what shape it has — and which Viz Engine renders it. From that description Studio Manager builds the wall geometry directly in the videowall engine's scene, so the engine's output lines up with the physical panels.

Everything in this chapter happens in the 3D studio view: a wall is an object in the same studio space as the cyclorama, the cameras and the tracking origin. All values are in **centimetres**, measured from the studio origin, with **Y pointing up**.





6.13.1 Setup

What you need

Viz Engine	One engine per videowall system, reachable from Studio Manager and **on air**.
Tracking Hub	A running Tracking Hub with a **videowall service** per engine.

GFX channel	A free GFX channel on that engine, used as the projection texture of the walls.

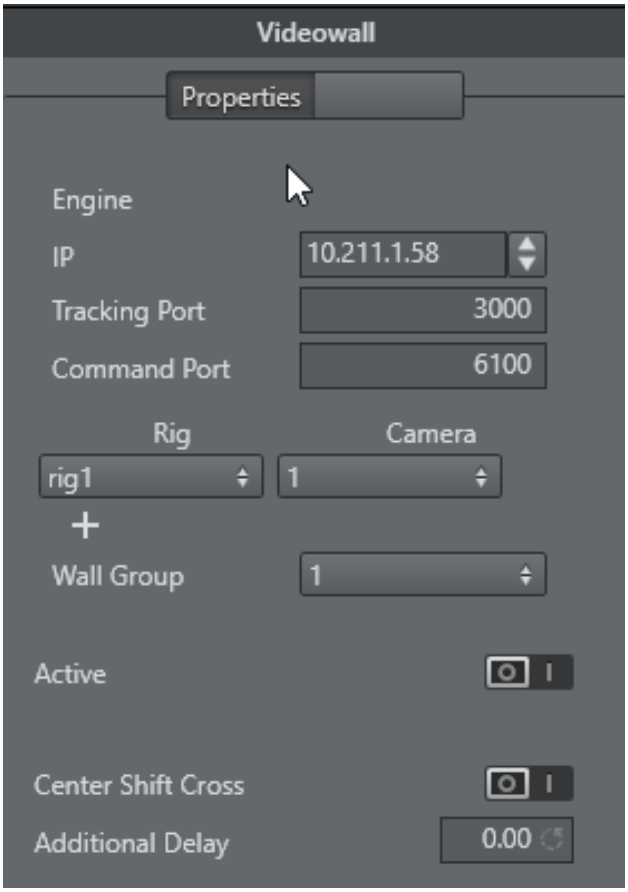
Studio Manager never writes into the engine's scene blindly: after every *Create Wall* it reads the scene tree back and reports which wall containers really arrived. If a step below is missing, the status line under the buttons says which one.

Adding the videowall service

The videowall service is the route from Studio Manager to the engine. Without it, walls can be drawn in the editor but nothing reaches an engine.

1. Open the **Topology** view.
2. Right-click in the service area and choose **Add video wall service**.
3. Select the new service and fill in its properties:
    - **IP** — address of the Viz Engine that drives the wall (default `127.0.0.1`).
    - **Port** — the engine's data port (default `3000`).
    - **Command port** — the engine's command port (default `6100`). Studio Manager uses this port directly to create the wall geometry and to read it back.
    - **Wall Group** — the videowall system this service feeds, see below.
4. Activate the service.

Copyright © 2026 Vizrt



Wall groups — more than one videowall engine

A **wall group** decides which engine a wall is sent to. There are four groups, numbered 1 to 4, and everything starts in group **1**.

- Every **wall** is in exactly one group (drop down in the wall's header in the editor).
- Every **videowall service** is in exactly one group (service properties).
- Tracking Hub sends a wall only to the services in the **same** group.

Groups are exclusive: there is no service that receives everything, and no wall that goes everywhere. A group may hold **one** videowall service — mirroring the same wall onto two engines is not supported. Both drop downs name a group after the service in it (for example *"2 — 10.0.0.31"*, or *"3 — no service"*), so you pick an engine rather than a bare number.

A configuration that never touches the groups has everything in group 1 and behaves exactly as it did before wall groups existed.

{info}
A wall in a group that **no service is in** is sent nowhere. The editor warns when you move a wall into such a group, and the settings panel shows the engine count next to the group selection.
{info}

Copyright © 2026 Vizrt

What Studio Manager builds in the engine

Create Wall replaces the scene on the videowall engine and rebuilds every wall of the selected group in it. Per wall it creates:

- a top-level container named after the wall (for example `Backwall1`, `OneCornerWall2`), carrying a **Virtual Window** whose projection texture is `GFX<n>` — the GFX channel chosen in the settings panel;
- one child container **Panel** (or `Panel1`, `Panel2`, … for multi-segment walls) with a **MultiPanel** geometry that carries the actual shape.

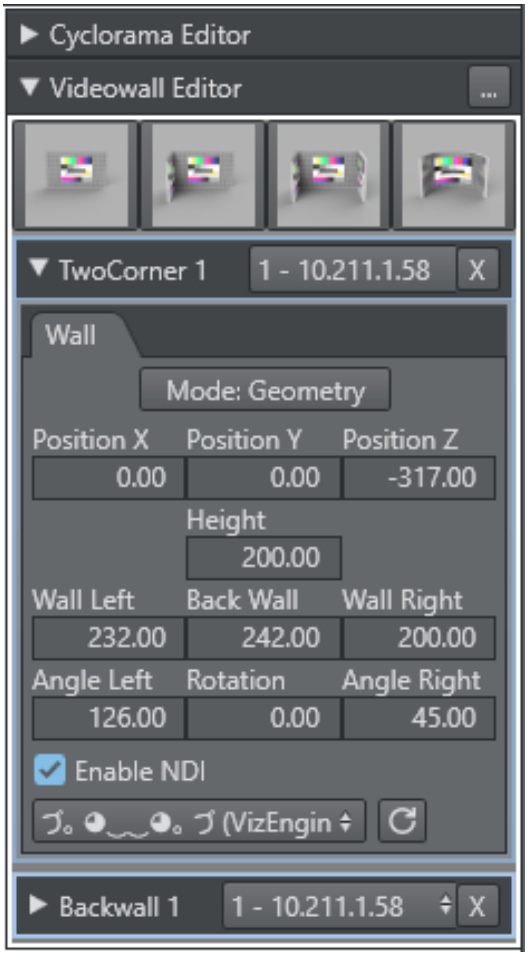
The wall is therefore a *window* into whatever is playing on that GFX channel. Playout, AR and the built-in test scene all arrive through that channel.

{note}
Changing the GFX channel takes effect on the **next** *Create Wall*, not immediately.
{note}

6.13.2 The editor panel

The VW Editor lives in the editor column on the right of the 3D view, below the Cyc Editor. Its header carries the panel title and a `...` button that opens the videowall settings — reachable whether the editor is expanded or collapsed.

Copyright © 2026 Vizrt



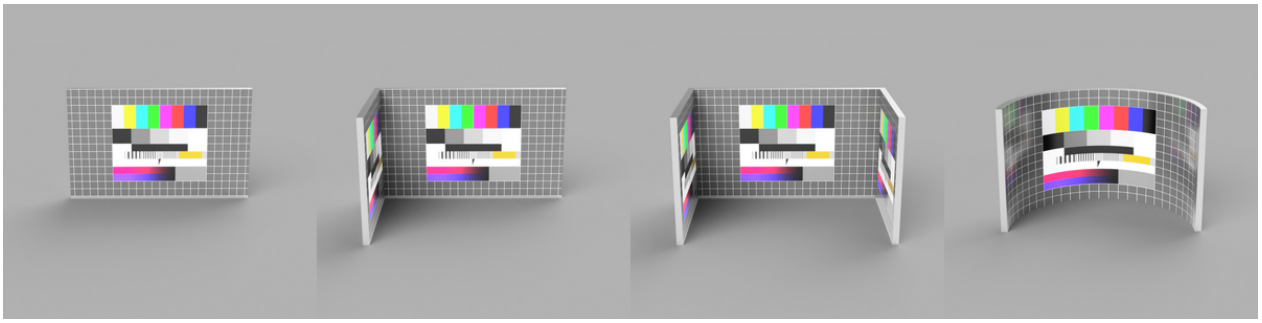
Adding a wall

The four buttons at the top of the panel add a wall of that type:

Button	Wall type	Panels created in the engine
▭	**Back Wall** — one flat panel	1
◣	**One Corner** — two panels meeting in a corner	2
◲	**Two Corner** — a back wall with a left and a right return	3
◡	**Curved** — a curved/arc wall	1 (tessellated into segments)

At most **5** walls can exist at the same time. Above **3** walls Studio Manager warns that render performance may suffer and lets you cancel; the wall is still allowed.

Copyright © 2026 Vizrt



The wall list

Each wall appears as a collapsible entry. Its header carries:

- the **wall name** (`Backwall 1` , `OneCorner 2` , …) — derived from the type and the wall id;
- the **wall group** drop down — changing it moves the wall to another videowall system: the wall is deleted on the old group's engine and created on the new one;
- the **remove** button (**X**) on the right, which asks for confirmation;
- in advanced mode, the per-wall **T** (send targets) and **detect** buttons — see Advanced mode.

Colour coding. Each wall entry is tinted: the **group picks the hue** (1 blue, 2 orange, 3 green, 4 purple) and the position within the group picks the shade, lightest wall first. The same tint is used for the wall in the 3D view and for its marker in the test room, so a wall can be traced from the list to the studio view to the engine. Moving a wall to another group visibly changes its family.

Editing and when the engine is updated

Every value in a wall's panel is applied to the 3D view immediately and forwarded to the engine as it changes, so dragging a slider moves the real wall live. Only the parameter that actually changed is sent.

The full wall definition — the thing that is **stored in the configuration** — is pushed once you let go of the control (and debounced for keyboard input), so a drag does not write hundreds of configuration updates. Walls survive a configuration reload with their ids and groups intact.

Nothing at all is sent while the videowall service is **stopped**; those updates are skipped, not queued. Start the service and use *Create Wall* to bring the engine back in line with the editor.

6.13.3 Wall geometries

Every wall type shares the same basic frame:

Field	Meaning
Position X / Y / Z	Anchor point of the wall in studio coordinates, in cm.

Copyright © 2026 Vizrt

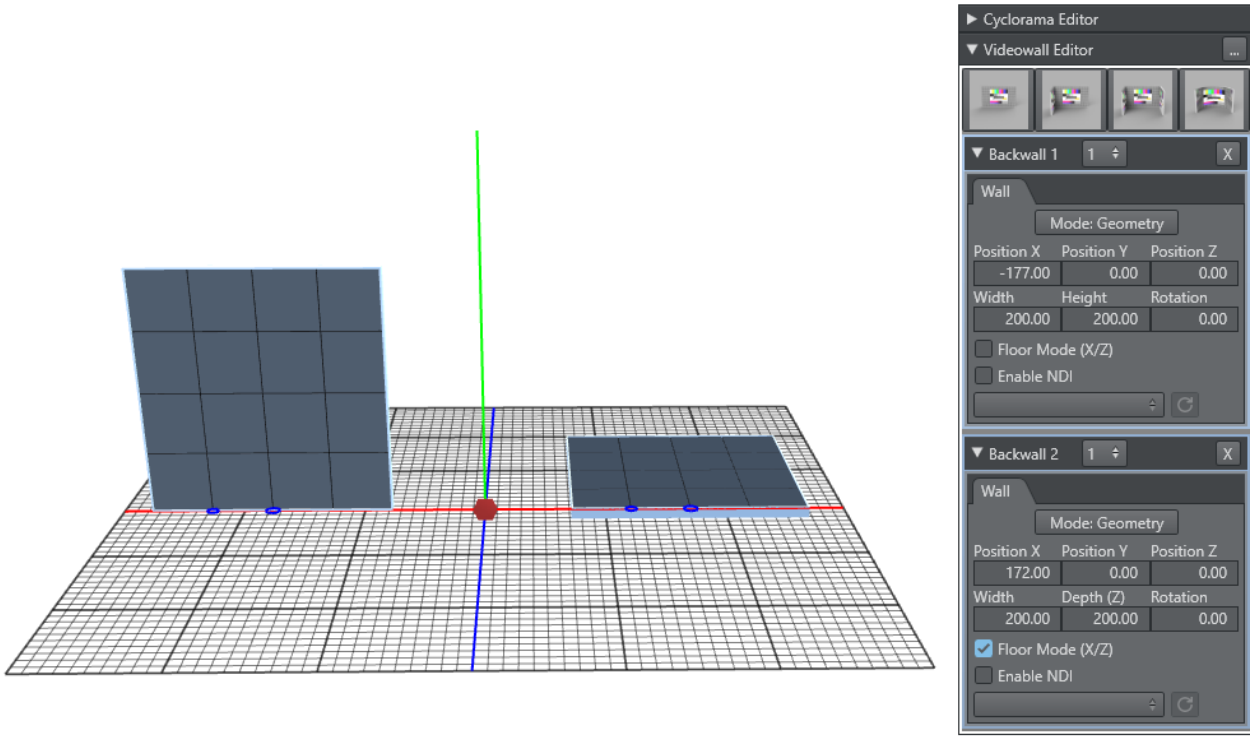
Field	Meaning
Rotation	Rotation around the vertical (Y) axis, in degrees.
Height	Height of the wall in cm (10 – 10000).

The type-specific fields are:

Back Wall

A single flat panel.

Field	Meaning
Width	Width of the panel in cm (10 – 10000).
Height	Height of the panel in cm.
Floor Mode (X/Z)	Lays the panel flat into the floor plane. The *Height* field then becomes **Depth (Z)**, because the second dimension now runs along Z instead of up. Use it for LED floors.

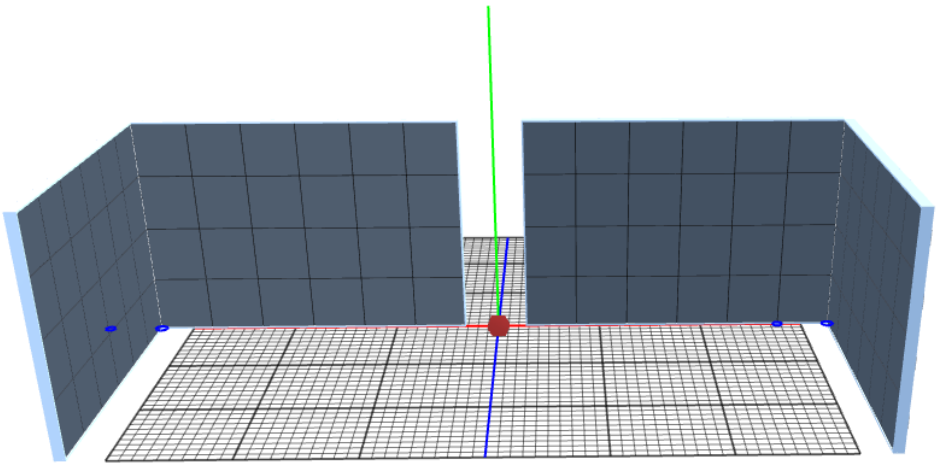


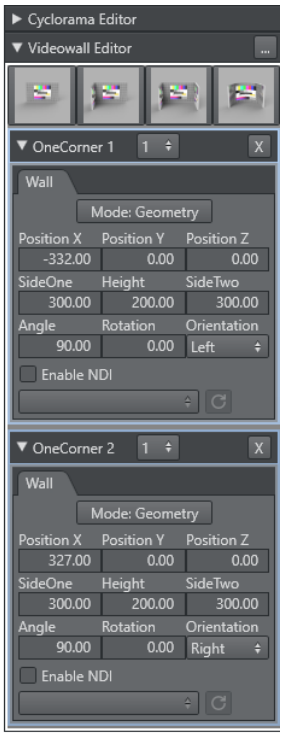
Copyright © 2026 Vizrt

One Corner

Two panels meeting in a corner — an L-shaped wall. The **position** is the corner (joint) itself.

Field	Meaning
SideOne	Length of the first leg in cm.
SideTwo	Length of the second leg in cm.
Height	Height of both legs.
Angle	Opening angle between the two legs, 0 – 180°. 90° is a square corner.
Orientation	**Left** or **Right** — which way the wall folds away from the corner.



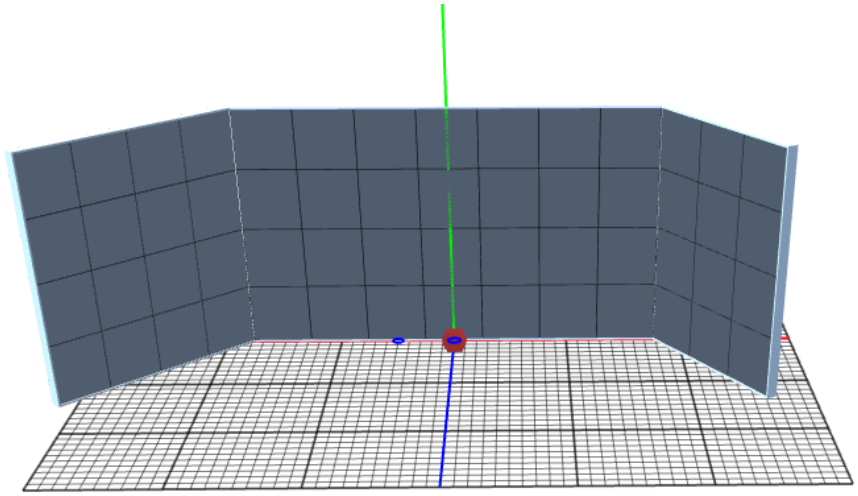


Two Corner

A back wall with a return on each side — a U-shaped wall of three panels.

Copyright © 2026 Vizrt

Field	Meaning
Back Wall	Width of the centre panel in cm.
Wall Left / Wall Right	Length of the left and right return in cm.
Height	Height of all three panels.
Angle Left / Angle Right	Opening angle of each return relative to the centre panel.





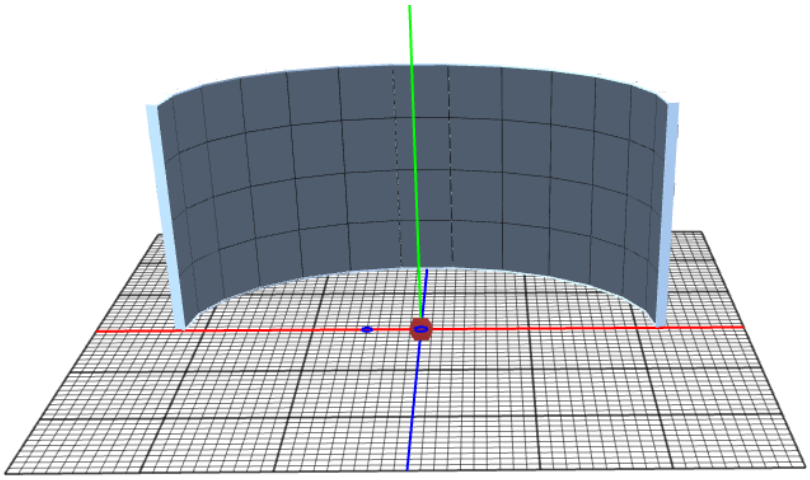
Curved

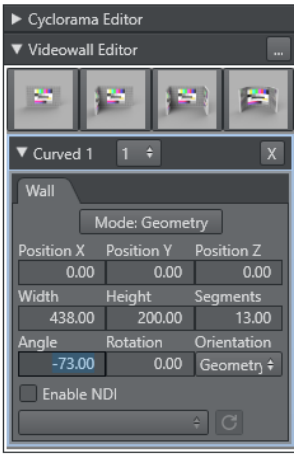
One curved panel, tessellated into straight segments.

Field	Meaning
Width	Chord width of the arc in cm.
Height	Height of the wall.
Segments	Number of straight segments the curve is built from, 1 – 100. More segments = smoother curve, more geometry.
Angle	Opening angle of the arc, −90° to +90°. 0 is flat; the sign decides which way it bows.

Copyright © 2026 Vizrt

Field	Meaning
Orientation	How the panel is anchored: **Geometry** centres the arc on the entered position, **Segment** anchors it on its left edge.





Geometry mode and corner mode

Every wall type can be described in two ways, switched with the **Mode** toggle button at the top of its *Wall* tab:

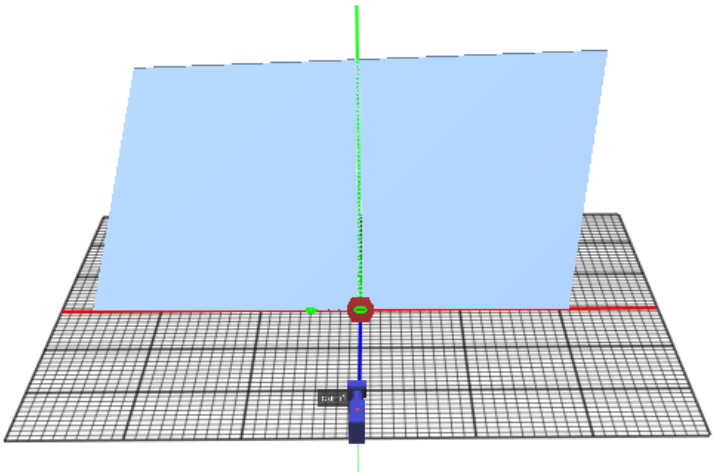
- **Mode: Geometry** — the wall is described by the fields above (position, rotation, width, height, angles). This is the normal way to build a wall from its dimensions.
- **Mode: Corners** — the wall is described by the **absolute world coordinates of its corner points**. Each corner gets an X / Y / Z box. Use this when the wall is not a clean rectangle, when it is tilted, or when the corners were measured in the real studio.

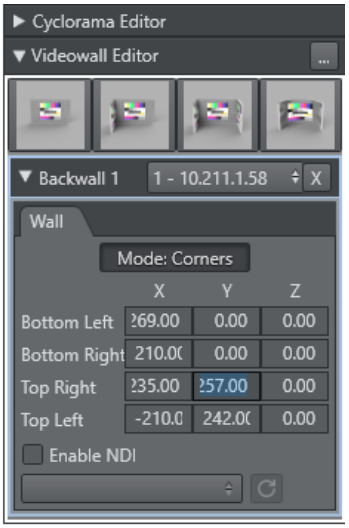
Switching to corner mode seeds the corners from the current geometry, so a wall built with the dimension fields starts corner mode from exactly the same shape and can be nudged from there.

Wall type	Corners in corner mode
Back Wall	4 (bottom-left, bottom-right, top-right, top-left)
One Corner	6 (far / joint / far, bottom and top)
Two Corner	8
Curved	6 (left, middle, right — bottom and top) plus a **Segments** field

Copyright © 2026 Vizrt

In corner mode the coordinates are sent to the engine as they are and the panel's own transform is reset, so the wall sits exactly where the numbers say.

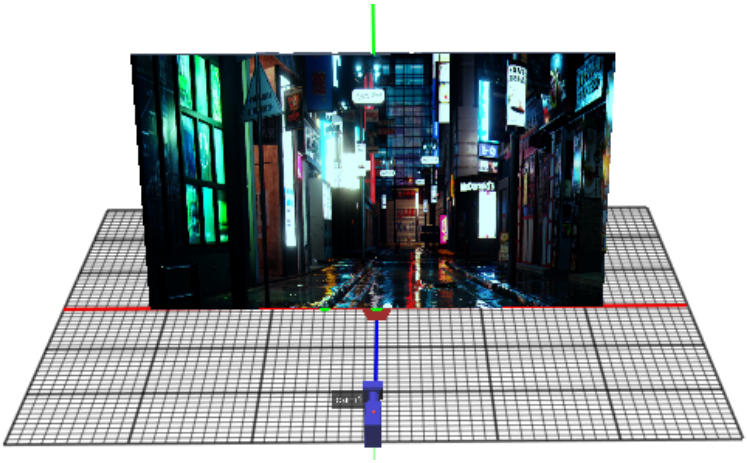


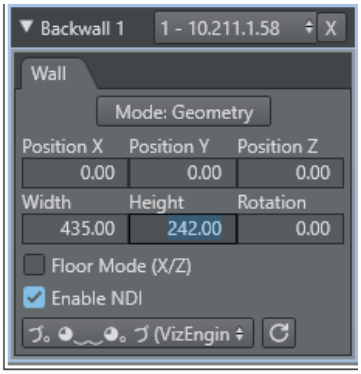


NDI preview

Each wall can show a live **NDI** stream on its surface in the 3D view, so the studio view shows what the wall is actually playing instead of a grey panel.

1. Tick **Enable NDI** on the wall's *Wall* tab.
2. Pick a source from the drop down; the ⟳ button re-scans the network for NDI sources.





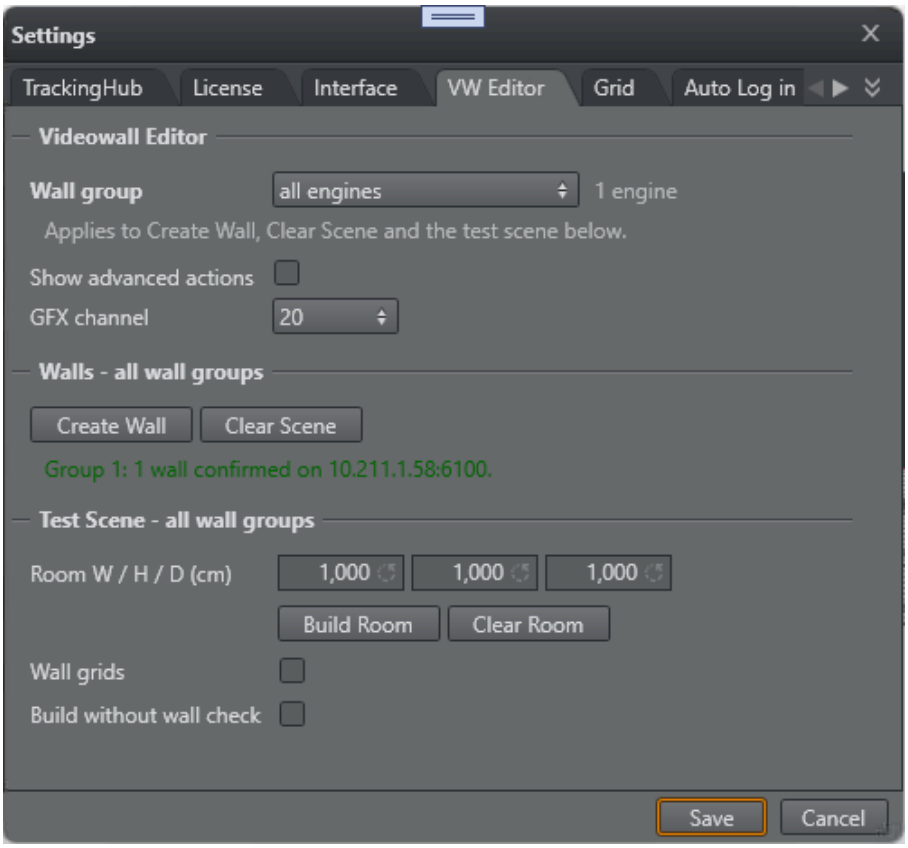
Copyright © 2026 Vizrt

This is a preview inside Studio Manager — it does not change what the engine renders. If one engine feeds several walls from a single output, see **Crop NDI preview to this area** under Advanced mode.

> **No NDI output in Videowall mode**
> When a Viz Engine output is configured to run in **Videowall mode**, the engine's NDI output is disabled. If you need an NDI feed of the wall, install NDI Tools on the videowall engine machine and enable its **Screen Capture** option.
> **Use this for setup only** and disable it again before going into production — screen capture costs performance on the videowall engine machine.

6.13.4 The settings panel

Opened with the `...` button in the VW Editor header, or from *Settings* → **VW Editor** tab. It holds everything that is not a property of a single wall.



Copyright © 2026 Vizrt

Videowall Editor

Setting	Meaning
Wall group	The videowall system every button on this tab acts on — *Create Wall*, *Clear Scene* and *Build Room* alike. The text next to it shows how many engines the selection resolves to, so an empty group is visible before you press anything. The extra **all engines** entry widens the actions to every videowall engine.
Show advanced actions	Reveals the advanced per-wall controls, see Advanced mode.
GFX channel	The GFX channel the wall panels project and the test room is loaded into. Walls pick up a change on the next *Create Wall*.

The headers of the two group boxes below name the active selection (*"Walls — wall group 2"*), so it is always clear what a button is about to touch.

Walls

Button	What it does
Create Wall	Replaces the scene on the selected group's engine and rebuilds every wall of that group in it, then reads the scene tree back and reports which wall containers arrived.
Clear Scene	Replaces the scene on the selected group's engines with an empty one. The walls stay in the editor and can be rebuilt with *Create Wall*. Asks for confirmation.

The status line underneath reports per engine and names the scope, for example *"Group 2: 3 walls confirmed on 10.0.0.32:6100."* A failure says why — engine not on air, no service in the group, containers missing.

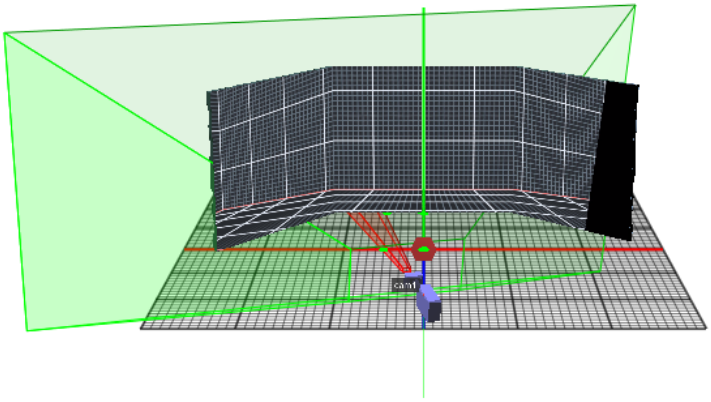
{warning}
Create Wall and *Clear Scene* **replace the whole scene** on that engine. Anything else built in it by hand is lost. Moving a single wall between groups deliberately does not replace a scene.
{warning}

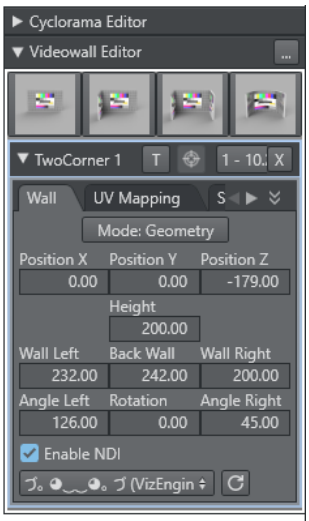
Test scene

The test scene is a grid room: a closed box around the studio origin, painted with a grid measured from that origin, plus one grid panel per wall marking where each videowall sits inside it. It is built as a scene of its own and loaded into the GFX channel the walls project — so you see it *through* the walls. It is the quickest way to check that a wall's position, size and orientation in the editor match the real panels.

Copyright © 2026 Vizrt

Control	Meaning
Room W / H / D (cm)	Extent of the room. Width and depth are centred on the studio origin, the floor sits on the studio floor (Y = 0). 50 – 10000 cm.
Build Room	Builds the room on the selected group's engines. Each engine's room carries markers only for the walls of that engine's group.
Clear Room	Closes the room on every engine it is loaded on, whatever the group selection — so changing the selection cannot leave a room stranded.
Wall grids	Shows each wall's position as a coloured grid panel inside the room, in that wall's editor colour. Applies to a loaded room straight away.
Build without wall check	Builds the room even when the wall containers cannot be found on the engine — for wall scenes created by hand. The room is only visible through a wall's virtual window, so without one it may show nothing.





6.13.5 Advanced mode

Settings → VW Editor → Show advanced actions reveals the controls that are not needed for a normal setup: the per-wall **T** and **detect** buttons in each wall header, and the **UV Mapping** and **Screen** tabs in each wall's panel.

Copyright © 2026 Vizrt



UV Mapping

Controls how the video content is mapped onto each panel of the wall — which part of the incoming picture lands where. One block per panel:

Field	Meaning
Left / Right	Horizontal UV range of this panel, −2 … 2.
Top / Bottom	Vertical UV range of this panel, −2 … 2.
Rotation	Rotation of the mapping on this panel.
Lock UV mapping (keep manual values)	Stops Studio Manager from recalculating the UVs from the geometry, so hand-tuned values survive a change to the wall's dimensions.

By default the UVs are derived from the geometry — a One Corner wall splits the picture between its two legs in proportion to their lengths. Edit them when the real content is laid out differently, and tick **Lock UV mapping** to keep the edit.

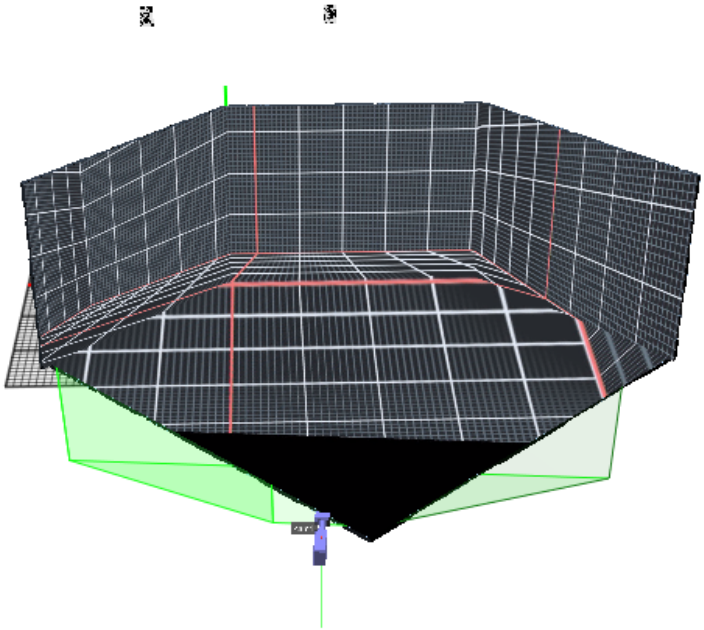
The curved wall has no UV Mapping tab; its mapping follows the arc.

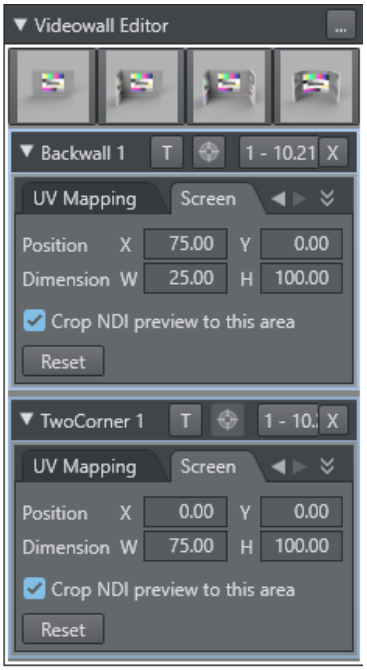
Copyright © 2026 Vizrt

Screen

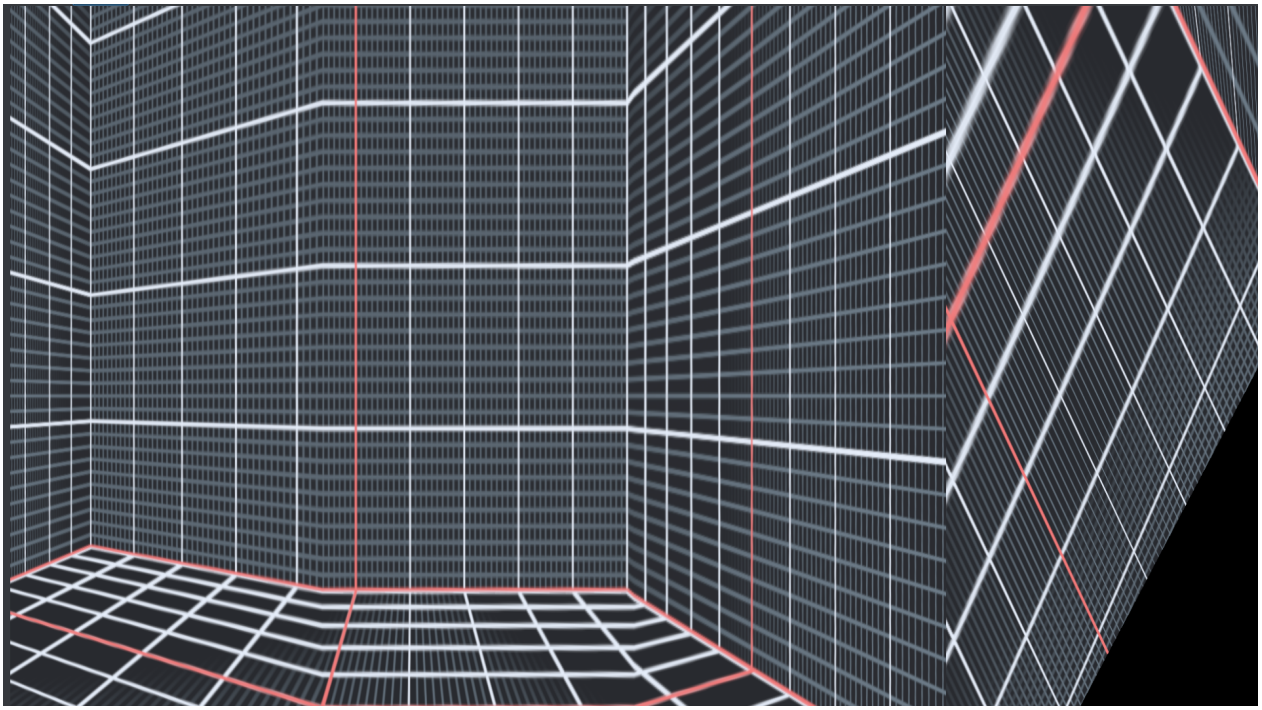
The **Virtual Window** screen attributes of the wall's top container — which part of the engine's output the wall shows.

Field	Meaning
Position X / Y	Offset of the screen area, −100 … 100.
Dimension W / H	Size of the screen area, 0 … 100.
Crop NDI preview to this area	Applies the same crop to the NDI preview inside Studio Manager, stretched across the wall's geometry. For an engine that renders several walls, where its video out carries all of them.
Reset	Restores the defaults.





Copyright © 2026 Vizrt



Target markers and corner detection

These two buttons support measuring a wall's real corners with the tracking system instead of typing them:

- **T (Send Targets)** — deploys the wall's target-marker scene to its engine. The wall then displays numbered markers on its corners, which the measurement/AI tool can detect from the camera image.
- **Detect** — opens the target-assignment dialog, which assigns the detected corner positions to this wall and fills its corner coordinates.

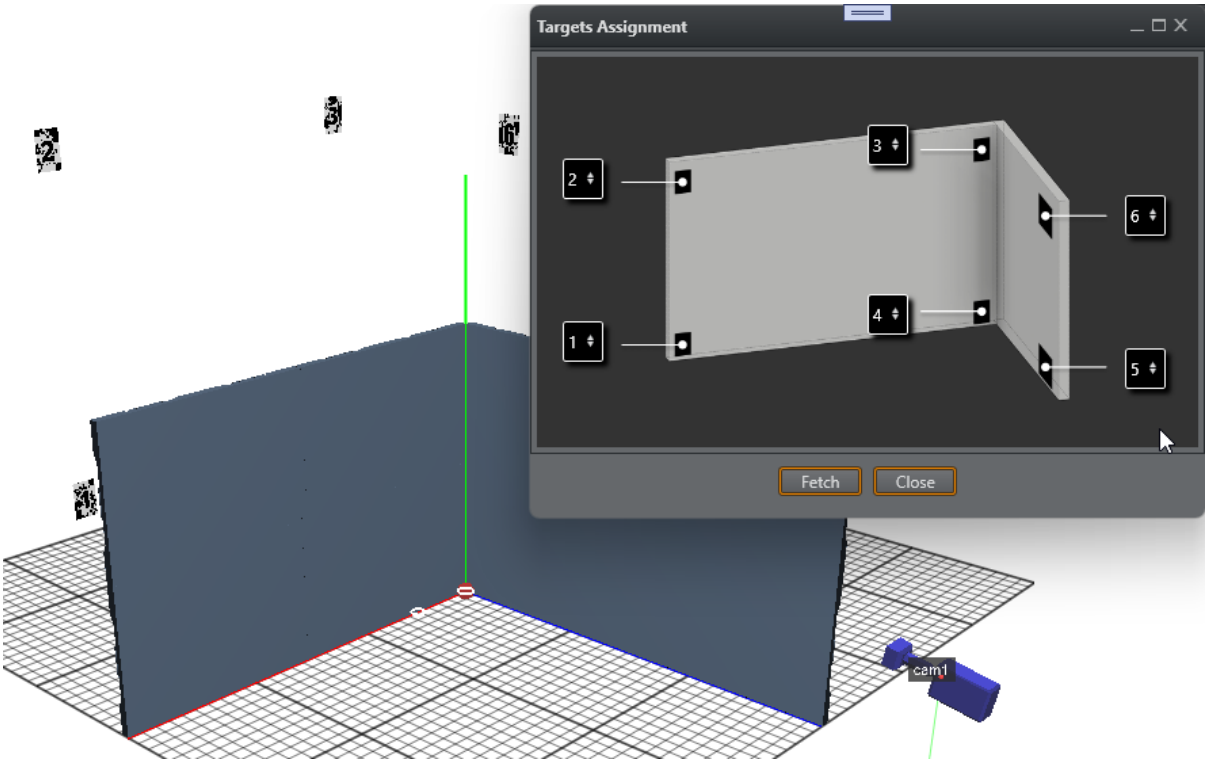
The detected corners are read from `targets.txt` in the AI watch folder configured in the settings (default `C:\\ProgramData\\vizrt\\StudioManager\\Calibrationresults`). Studio Manager watches
that folder, so the *Detect* buttons enable themselves as soon as enough targets are available. Each wall type needs its own minimum:

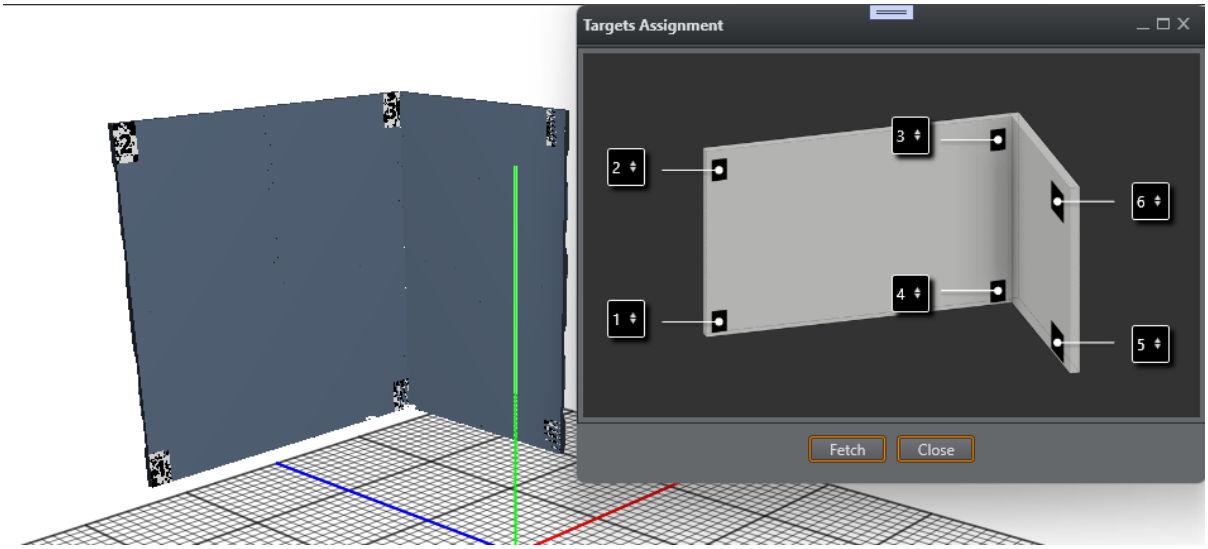
Wall type	Targets needed
Back Wall	4
One Corner	6
Two Corner	8

Copyright © 2026 Vizrt

Wall type	Targets needed
Curved	5

Until that many targets are present, the button stays disabled and its tooltip says how many were found. The result lands in the wall's **corner mode** coordinates, so it can be checked and corrected by hand afterwards.





Copyright © 2026 Vizrt

*The target-assignment dialog opened for a wall,
showing the detected targets and the wall's corner slots they are assigned to.*

6.13.6 Example setup

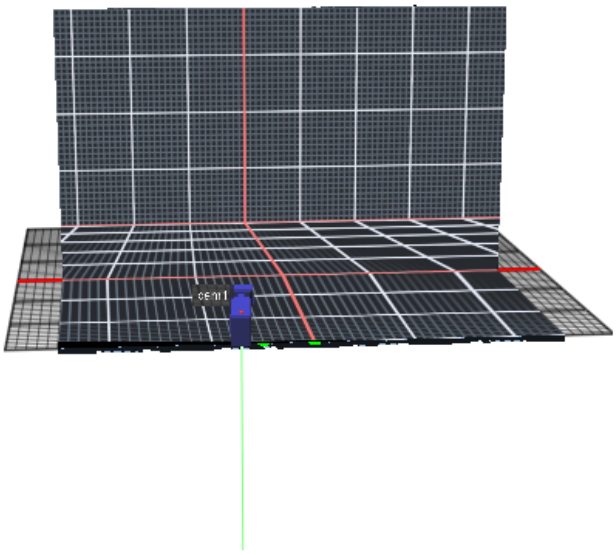
For a worked example — from an empty configuration to a wall on the engine — see **[Videowall Editor — Example Setup]**.

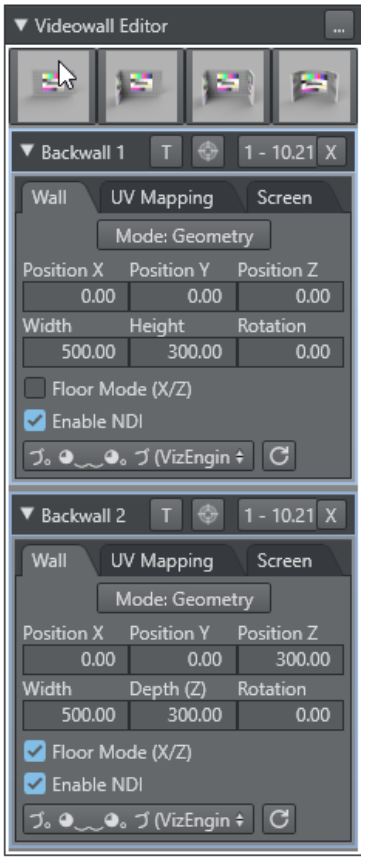
6.14 Videowall Editor — Example Setup

A complete walkthrough: from an empty configuration to two LED surfaces rendered by one Viz Engine, verified with the test room — and then extended to a second engine with wall groups.

The studio in this example:

- an **LED back wall**, 8 m wide and 3.5 m high, standing 5 m behind the studio origin;
- an **LED floor**, 8 m wide and 5 m deep, between the back wall and the origin;
- both driven by one Viz Engine at **192.168.0.10**;
- later, an **L-shaped side wall** added on a second engine at **192.168.0.20**.



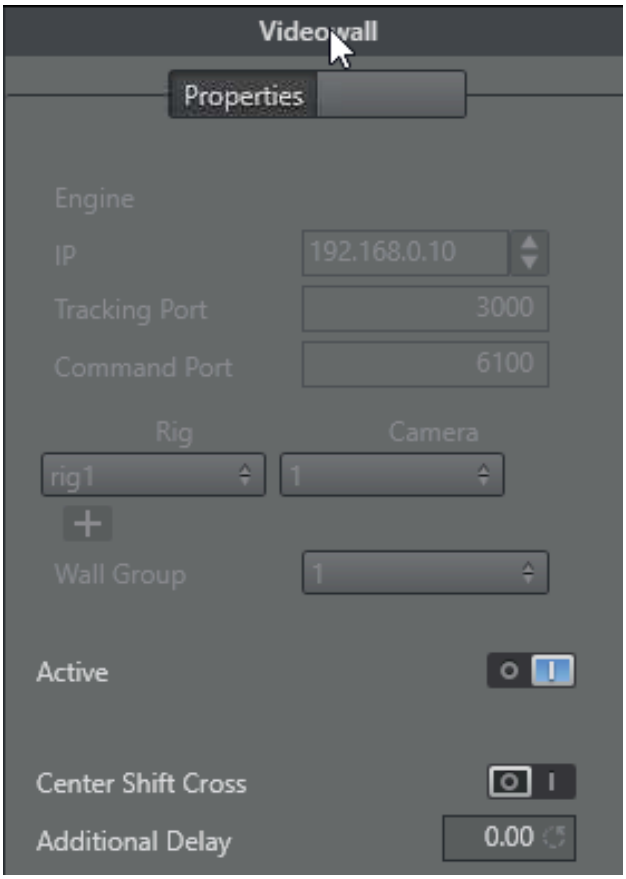


All values are in centimetres, measured from the studio origin. Where a coordinate sign does not match your studio, flip it — the 3D view shows immediately which way the wall moved.

Copyright © 2026 Vizrt

6.14.1 Step 1 — Add the videowall service

1. Open the **Topology** view.
2. Right-click in the service area → **Add video wall service**.
3. Select the new service and set:
    - **IP** — `192.168.0.10`
    - **Port** — `3000`
    - **Command port** — `6100`
    - **Wall Group** — `1`
4. Activate the service and make sure the engine is **on air**.



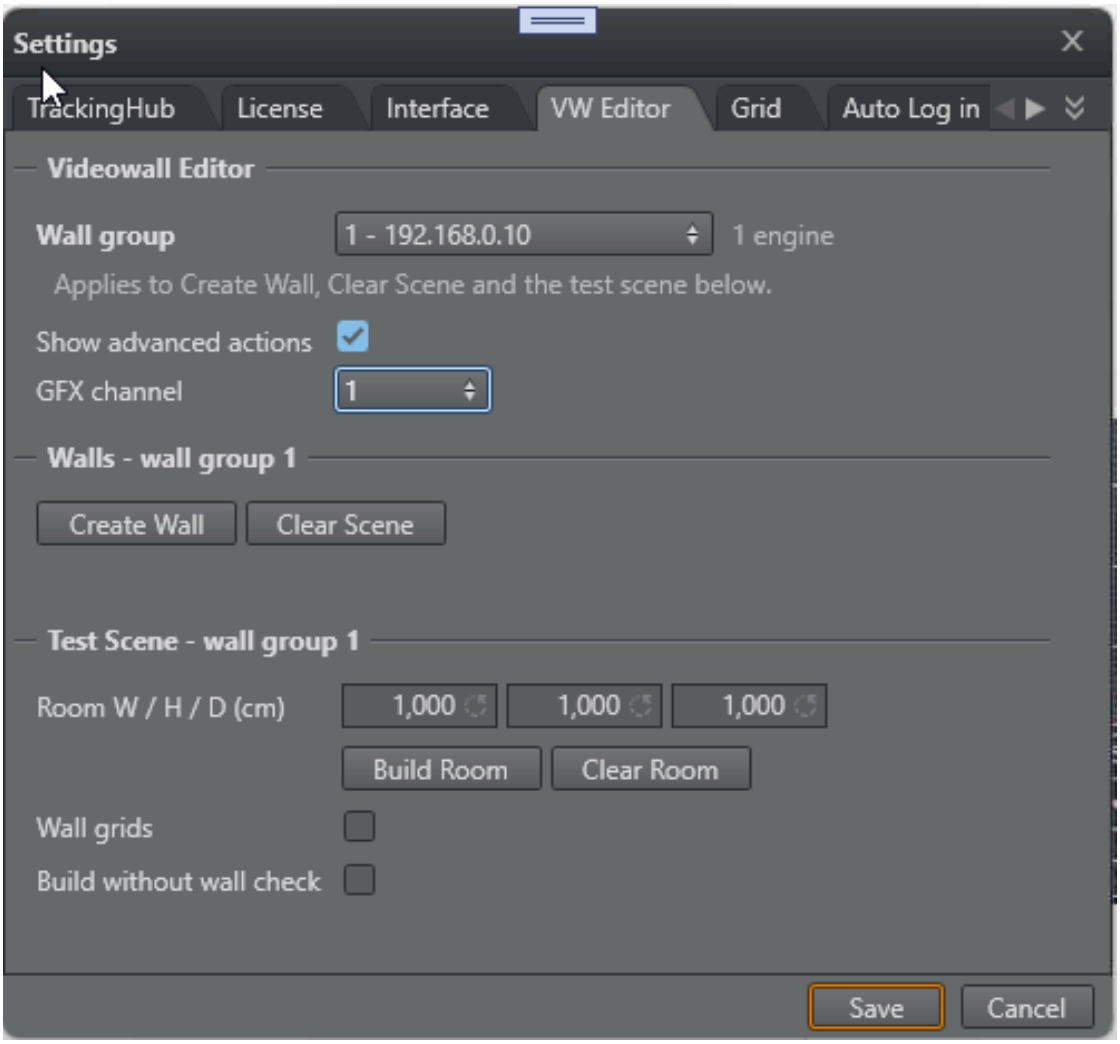
If the engine is not on air, the walls can still be created in the editor — but *Create Wall* will report *"engine at 10.0.0.31:6100 is not on air"* and nothing reaches the engine.

Copyright © 2026 Vizrt

6.14.2 Step 2 — Choose the GFX channel

1. Open the VW Editor settings: the `...` button in the VW Editor header, or *Settings* → **VW Editor**.
2. **Wall group**: `1 — 10.0.0.31`. The text beside it should say that the group resolves to one engine.
3. **GFX channel**: `1`.

This is the channel the walls will show. Whatever is played out on GFX1 on that engine is what the LED surfaces display.

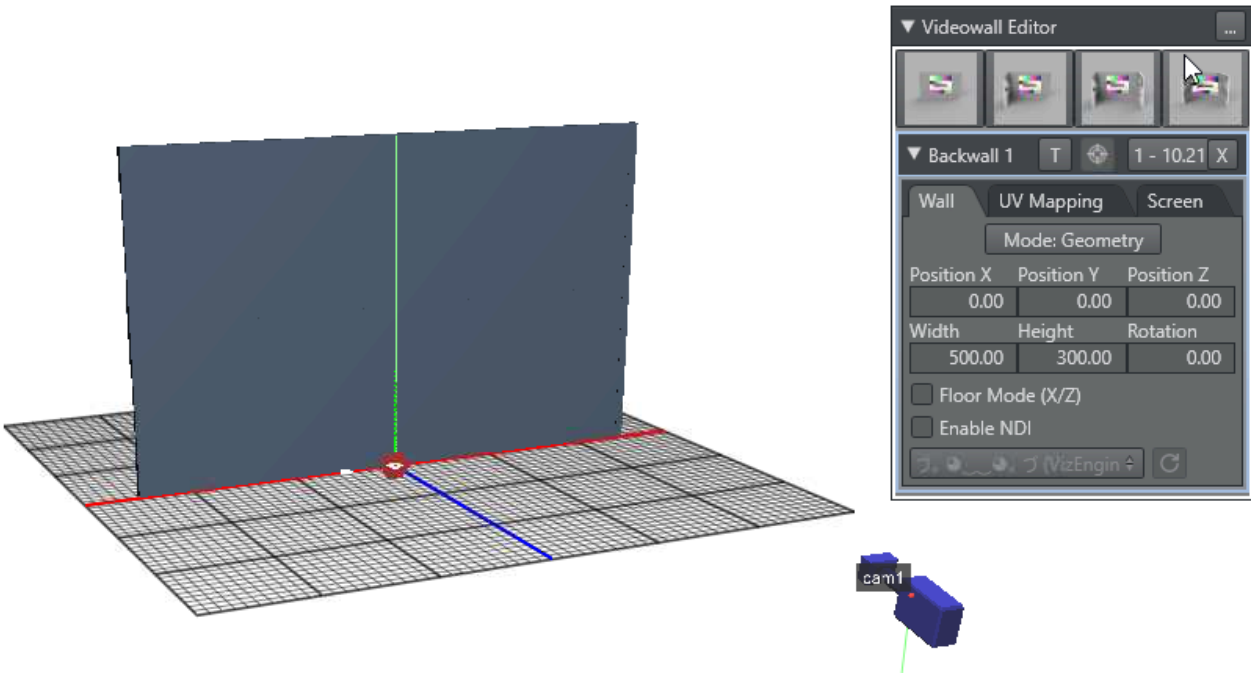


6.14.3 Step 3 — Add the back wall

1. In the VW Editor panel, click the **Back Wall** button. A wall entry *Backwall 1* appears, tinted blue (group 1).

Copyright © 2026 Vizrt

2. Check the 3D view: a flat panel standing 5 m behind the origin, facing the cameras. If it faces away, set **Rotation** to `180` ; if it stands in front of the origin, use `-500` for Position Z.



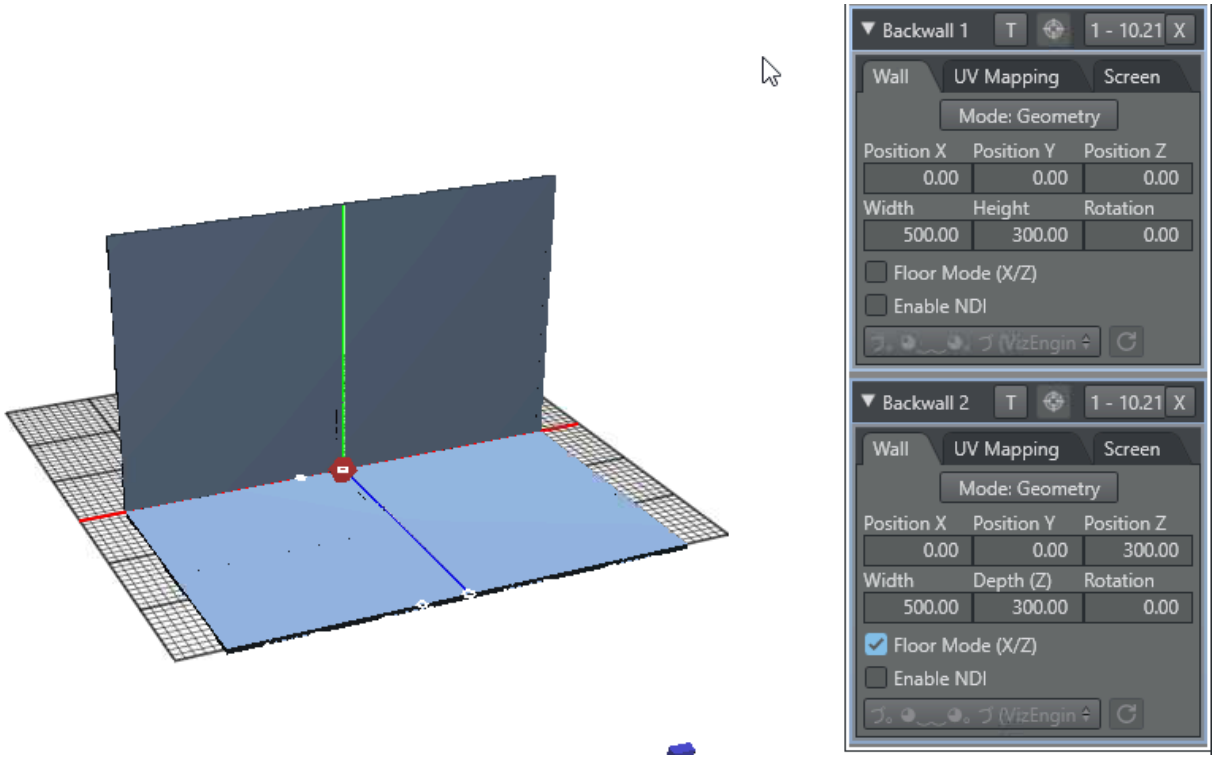
6.14.4 Step 4 — Add the LED floor

The floor is a second Back Wall, laid flat with **Floor Mode**.

1. Click the **Back Wall** button again — *Backwall 2* appears, in a second shade of blue.
2. On its **Wall** tab, tick **Floor Mode (X/Z)**. The *Height* field changes its label to **Depth (Z)**.

The floor now runs from the studio origin back to the foot of the back wall.

Copyright © 2026 Vizrt

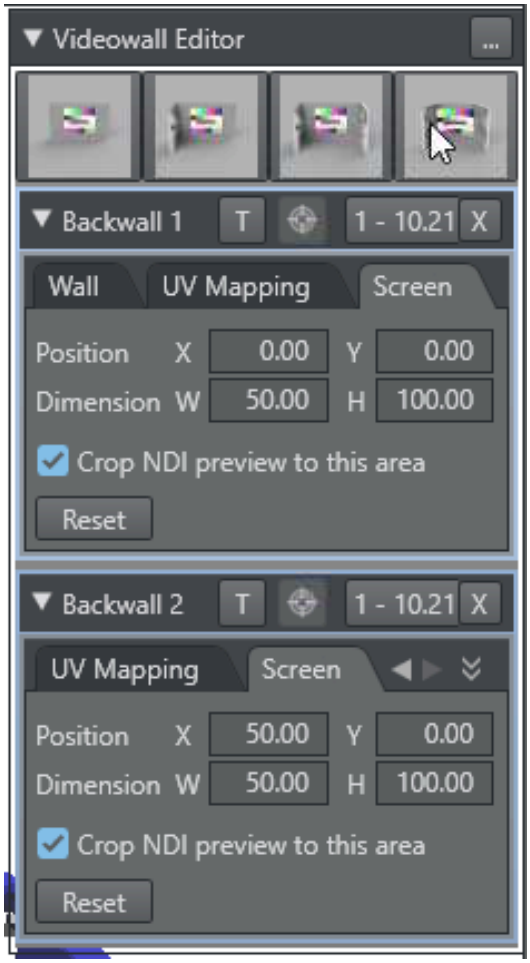


6.14.5 Step 5 — Define the Screen mapping

1. Click on the **Screen** panel for each wall item.
    a. (*Settings → VW Editor → Show advanced actions*)
2. Adjust it for a 50% - 50% split in the engine.
    a. Backwall1:
        i. Position `X = 0.00 / Y = 0.00`
        ii. Dimension `W = 50.00 / H = 100.00`
    b. Backwall2:
        i. 1. Position `X = 50.00 / Y = 0.00`
        2. Dimension `W = 50.00 / H = 100.00`

Activate the NDI Crop option for mapping to be applied on the NDI output as well

Copyright © 2026 Vizrt



6.14.6 Step 5 — Create the walls on the engine

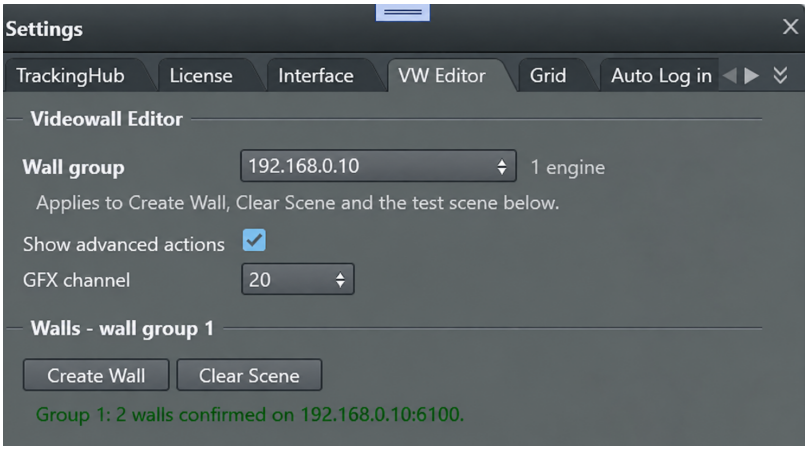
1. Back in the settings, check that **Wall group** is still `1 — 10.0.0.31`.
2. Press **Create Wall**.

Studio Manager replaces the scene on the engine, builds both walls in it and then reads the scene tree back. The status line should read something like:

```
Group 1: 2 walls confirmed on 10.0.0.31:6100.
```

If it does not, see Troubleshooting.

Copyright © 2026 Vizrt



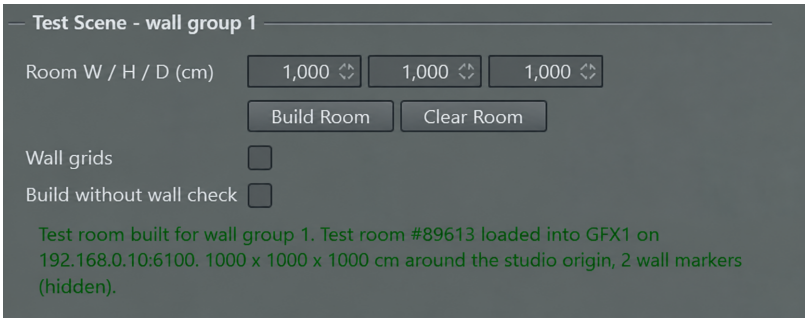
⚠ *Create Wall* replaces the whole scene on that engine. Anything built in it by hand is lost.

6.14.7 Step 6 — Verify with the test room

The test room is a grid box around the studio origin, shown *through* the walls. If the walls are placed correctly, the grid lines continue across them without a jump.

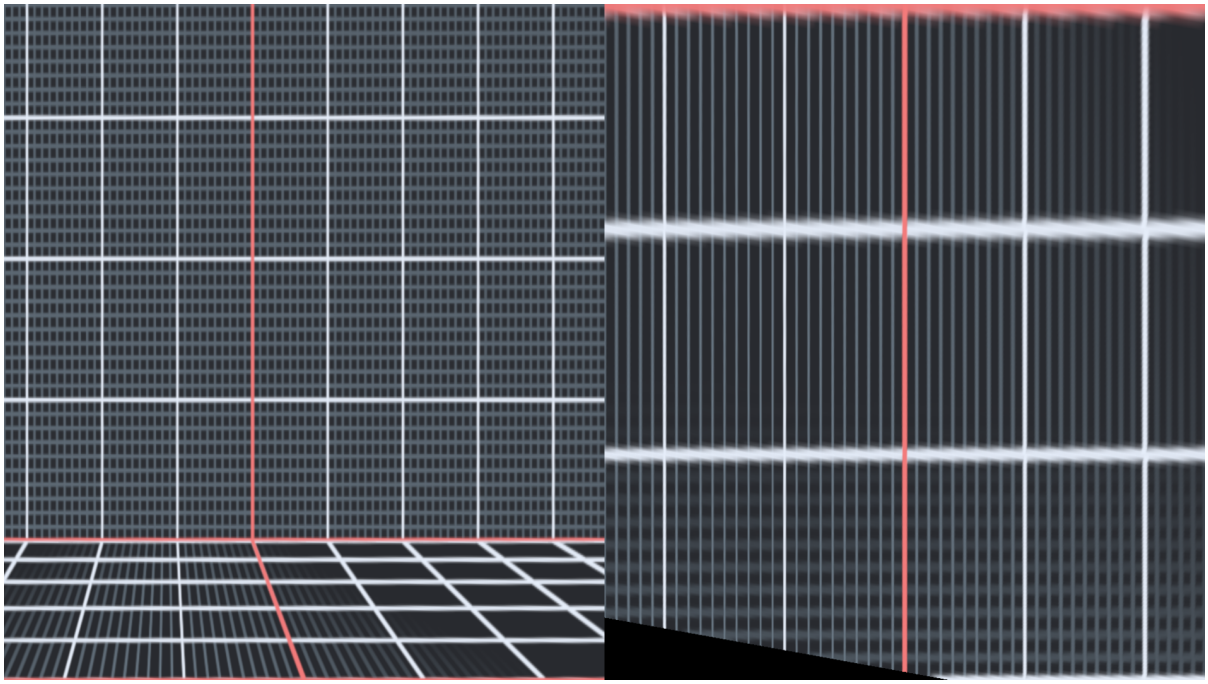
1. In the **Test scene** group box, set **Room W / H / D** to `1200 / 400 / 1200`.
2. Tick **Wall grids**.
3. Press **Build Room**.

On the engine output you should now see the grid room through both LED surfaces, with a coloured marker grid on each wall showing where Studio Manager thinks that wall is.



The Viz Engine output should no show the 2 panels:

Copyright © 2026 Vizrt



What to look for:

Observation	Meaning
Grid lines continue across the wall edge	Position and size are right.
Grid appears shifted or scaled on the wall	Check Position and Width/Height of that wall.
Grid tilts on the wall	Check Rotation, or switch to corner mode and enter the measured corners.
Wall shows nothing	The wall containers did not reach the engine, or the GFX channel differs from the one the room was loaded into.

Press **Clear Room** when you are done. It closes the room on every engine it is on.

6.14.8 Step 7 — Show real content

With the walls built, the engine's GFX1 channel is what the LED surfaces display. Point your playout or AR scene at GFX1 on that engine.

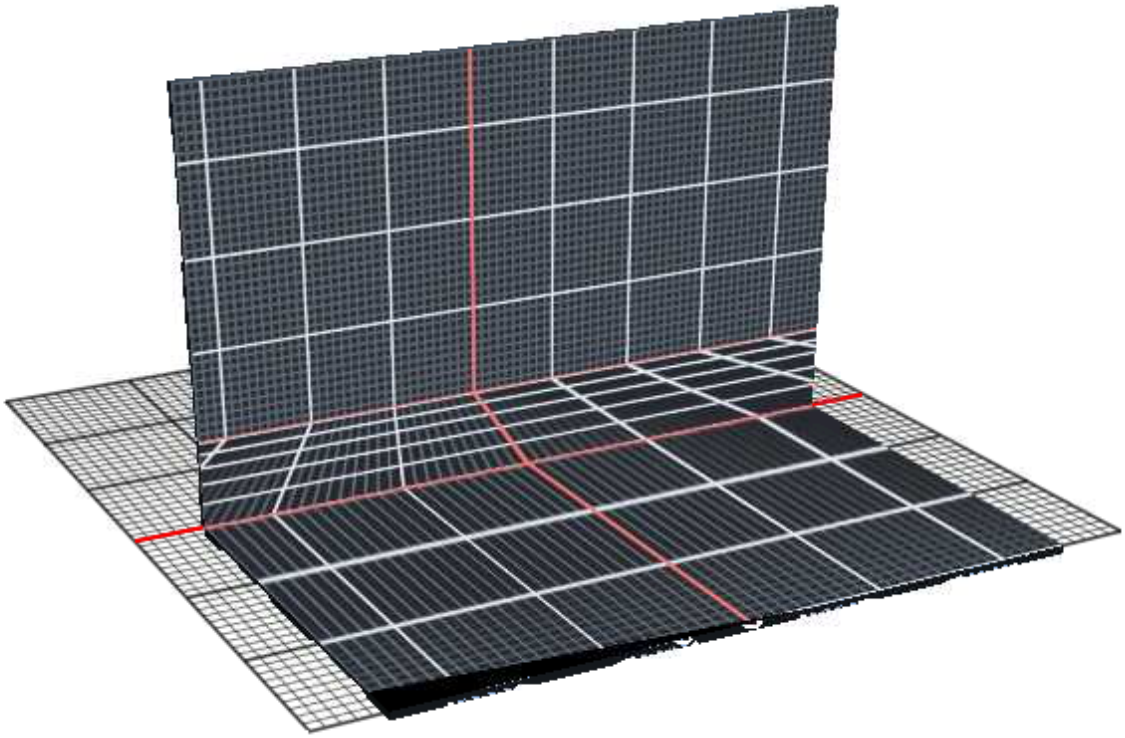
To see the same content inside Studio Manager, enable the preview per wall:

1. On the wall's **Wall** tab, tick **Enable NDI**.
2. Pick the engine's NDI output from the drop down (↻ re-scans).

If one engine output carries the picture for **both** surfaces side by side, switch on advanced mode (*Settings → VW Editor → Show advanced actions*) and use the **Screen** tab per wall to give each one

Copyright © 2026 Vizrt

its share of the picture, with **Crop NDI preview to this area** ticked so the Studio Manager preview matches.



6.14.9 Step 8 — Add a second engine with wall groups

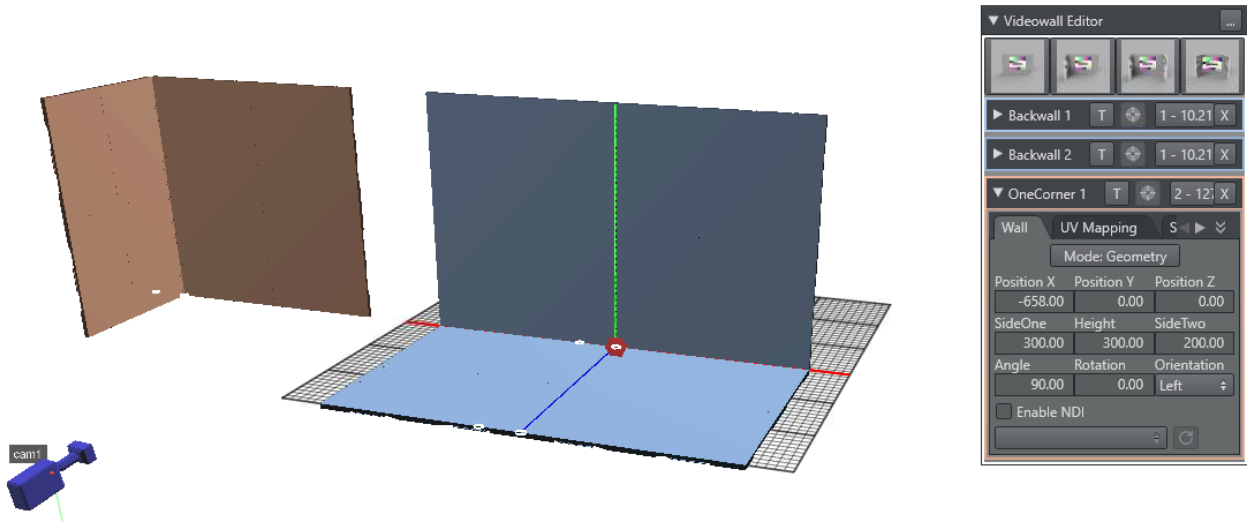
Now the studio gets an L-shaped side wall, and it is rendered by its own engine.

1. **Add the second service.** Topology → *Add video wall service*, IP `192.168.0.20`, ports `3000` / `6100`, **Wall Group** `2`. Activate it.
2. **Add the wall.** In the VW Editor, click the **One Corner** button. *OneCorner 3* appears.
3. **Move it to group 2.** In the wall's header, open the wall-group drop down and pick *"2 — 192.168.0.20"*. The entry changes colour from blue to orange — it is now in another videowall system. Studio Manager deletes it on the old engine and creates it on the new one.
4. **Build it.** In the settings, set **Wall group** to `2 — 192.168.0.20` and press **Create Wall**. Only that engine is touched; the walls on 192.168.0.10 are untouched.

```
Group 2: 1 wall confirmed on 192.168.0.20:6100.
```

6. **Verify.** Press **Build Room** with group 2 selected. The second engine's room carries markers only for the walls of group 2 — it never advertises the other system's walls.

Copyright © 2026 Vizrt



To work on both systems at once, select **all engines** in the **Wall group** drop down. *Create Wall* then rebuilds every group on its own engine in one go.

6.14.10 Troubleshooting

Status message	Cause	Fix
No TrackingHub connection.	Studio Manager is not connected to Tracking Hub.	Connect to Tracking Hub first.
No videowall service is in wall group N.	No service carries that group.	Set the **Wall Group** of a videowall service in the topology, or move the wall to a group that has one.
Videowall service is not running.	The service exists but is stopped.	Activate the service in the topology.
engine at …:6100 is not on air.	The engine is reachable but not on air.	Put the engine on air.
wall group resolves to 2 running engines (expected 1).	Two videowall services share one group — only possible in a hand-edited configuration.	Give each service its own group.
only 1 of 2 walls reached …; missing: Backwall2.	The engine did not create every wall container.	Check the engine log; run *Create Wall* again.

Copyright © 2026 Vizrt

Status message	Cause	Fix
No wall is in group N.	The selected group has no walls in the editor.	Select the right group, or move a wall into it.
Wall stays black, test room shows nothing	The GFX channel of the walls and of the content differ, or the wall containers do not exist.	Check **GFX channel** in the settings, then re-run *Create Wall* (the channel is only applied when the wall is built).

6.14.11 Checklist

- [ ] One videowall service per engine, active, with a unique wall group
- [ ] Engine on air
- [ ] GFX channel chosen, and content played out on that channel
- [ ] Every wall in the group of the engine that should render it
- [ ] *Create Wall* confirmed every wall on its engine
- [ ] Test room grid continuous across the wall edges
- [ ] Test room cleared again before going on air

Copyright © 2026 Vizrt

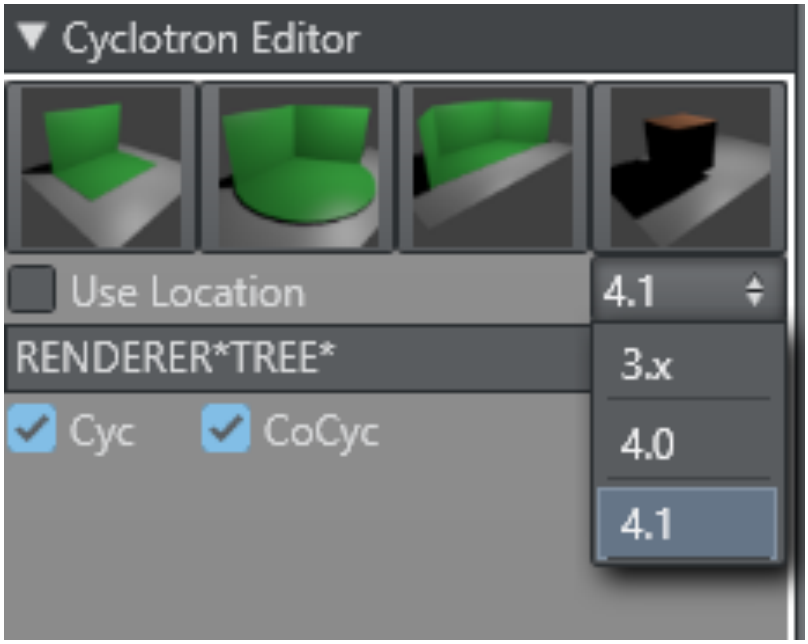
6.15 Cyclorama Editor

The CYC Editor enables creation of *Cyc* and *CoCyc masks* which are needed to mask out regions of the Studio not covered by the green/blue screen. The *CoCyc mask* is a geometry which is created in the CYC Editor and stored on the Tracking Hub. From there the CoCyc is sent to all connected Engines. Every change in the editor and its sub elements is immediately communicated to the Engines and to the Tracking Hub.

> **Note:** The cyc data is sent to Viz Engine only when at least one camera service exists. In case of NDI, at least one camera service must exist for every Viz Engine.

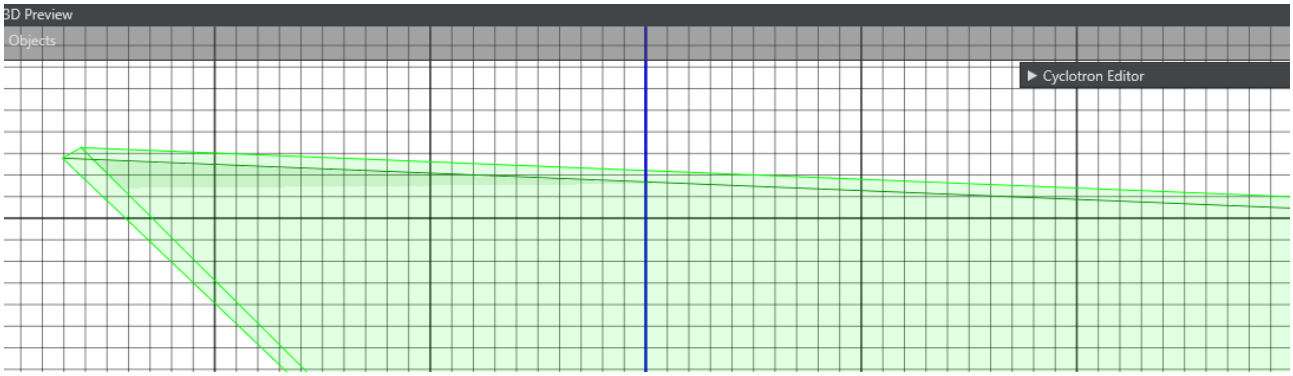
6.15.1 Workflows

A new Cyc workflow was introduced in Viz Engine version 4.0. The Cyc resides now in a special scene property, which resides in the Virtual Studio scene. Please refer to the Viz Engine Administrator Guide for further information. As the transportation of the Cyc information sent to the Engines is different now, there is a new dropdown in the Cyc Editor to select the used Viz Engine version.

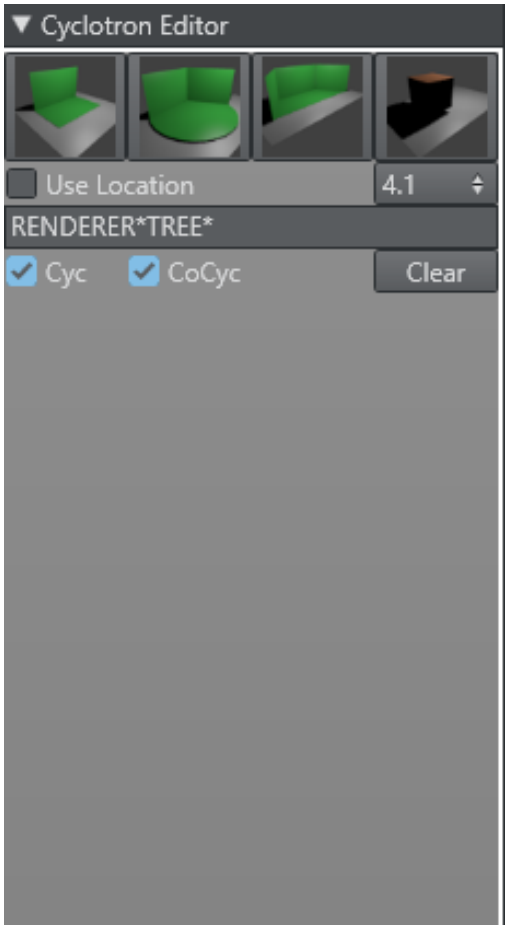


Please remember, that the Engine needs to be On Air to make the sent commands work.

Copyright © 2026 Vizrt



The new Cyc Editor is located in the top-right corner of the preview panel. It can be expanded by clicking the arrow on the left side.



There are three Cyclotron types are available: Back-Wall, One-Corner and Two-Corner Cyc.

Copyright © 2026 Vizrt

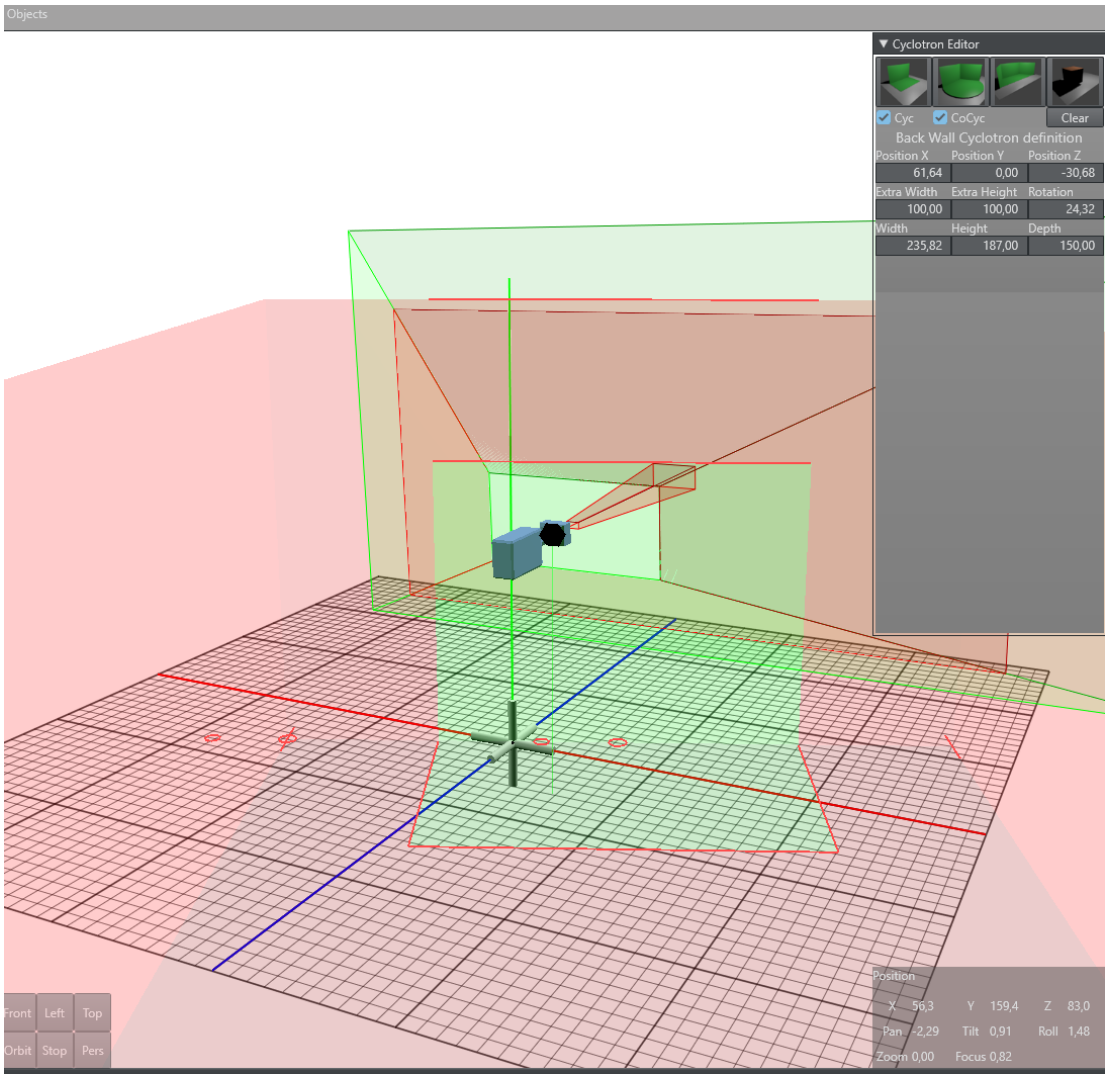
6.15.2 Common Cyc-Editor Elements



For every Cyc model, six common parameters can be defined. The position, rotation (Y-Axis), Extra Height and Extra Width. Extra height defines the distance from the top of the Cyc wall to the ceiling of the room. Extra width defines room size around the green wall and floor area.

Copyright © 2026 Vizrt

6.15.3 Back-Wall



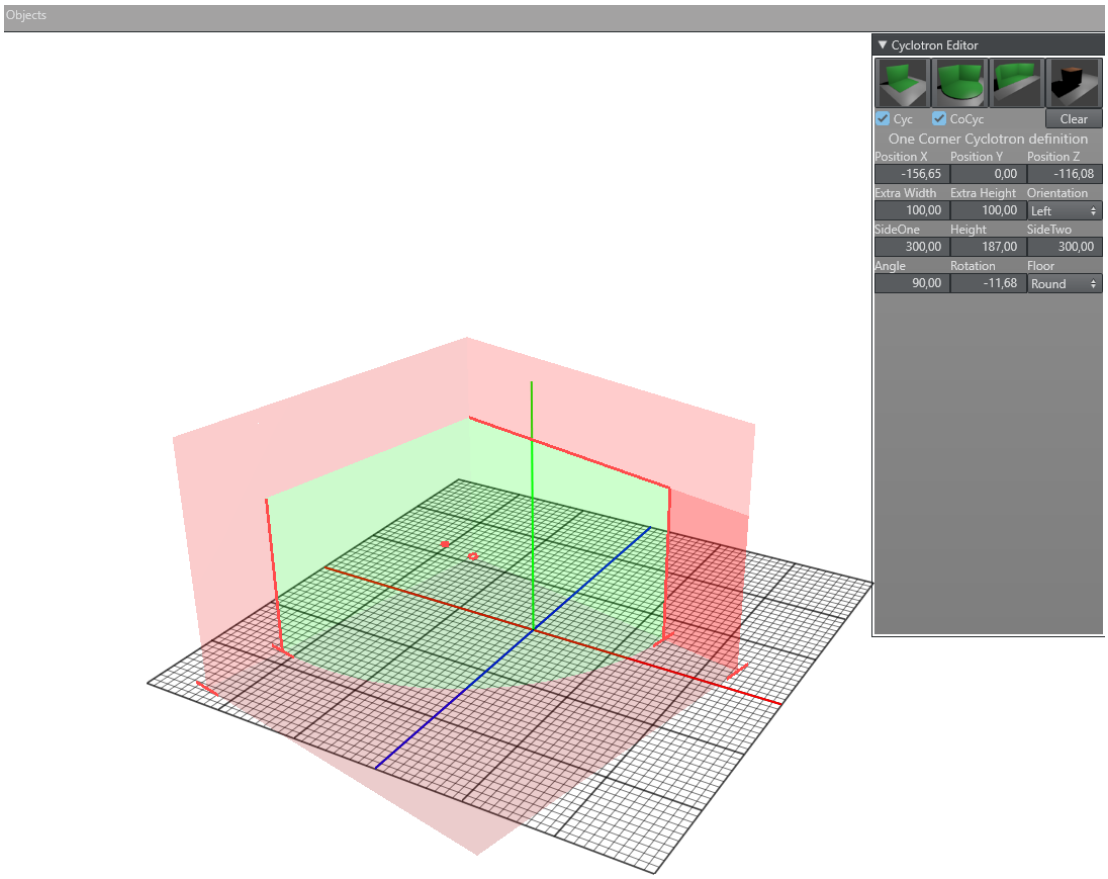
The back-wall Cyc has three additional parameters: These are the width of the green wall, height of the green wall and the depth of the ground area.

Copyright © 2026 Vizrt



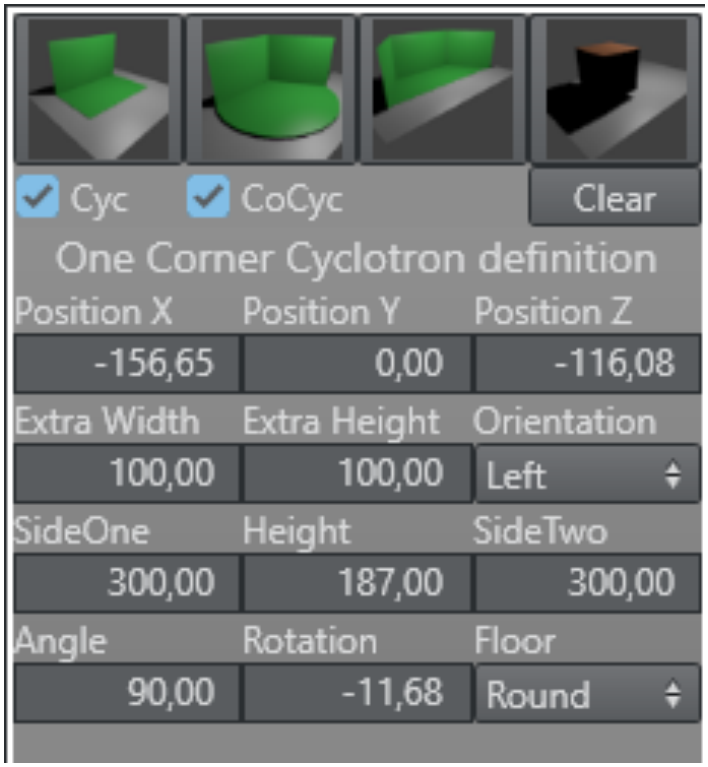
Copyright © 2026 Vizrt

6.15.4 One Corner



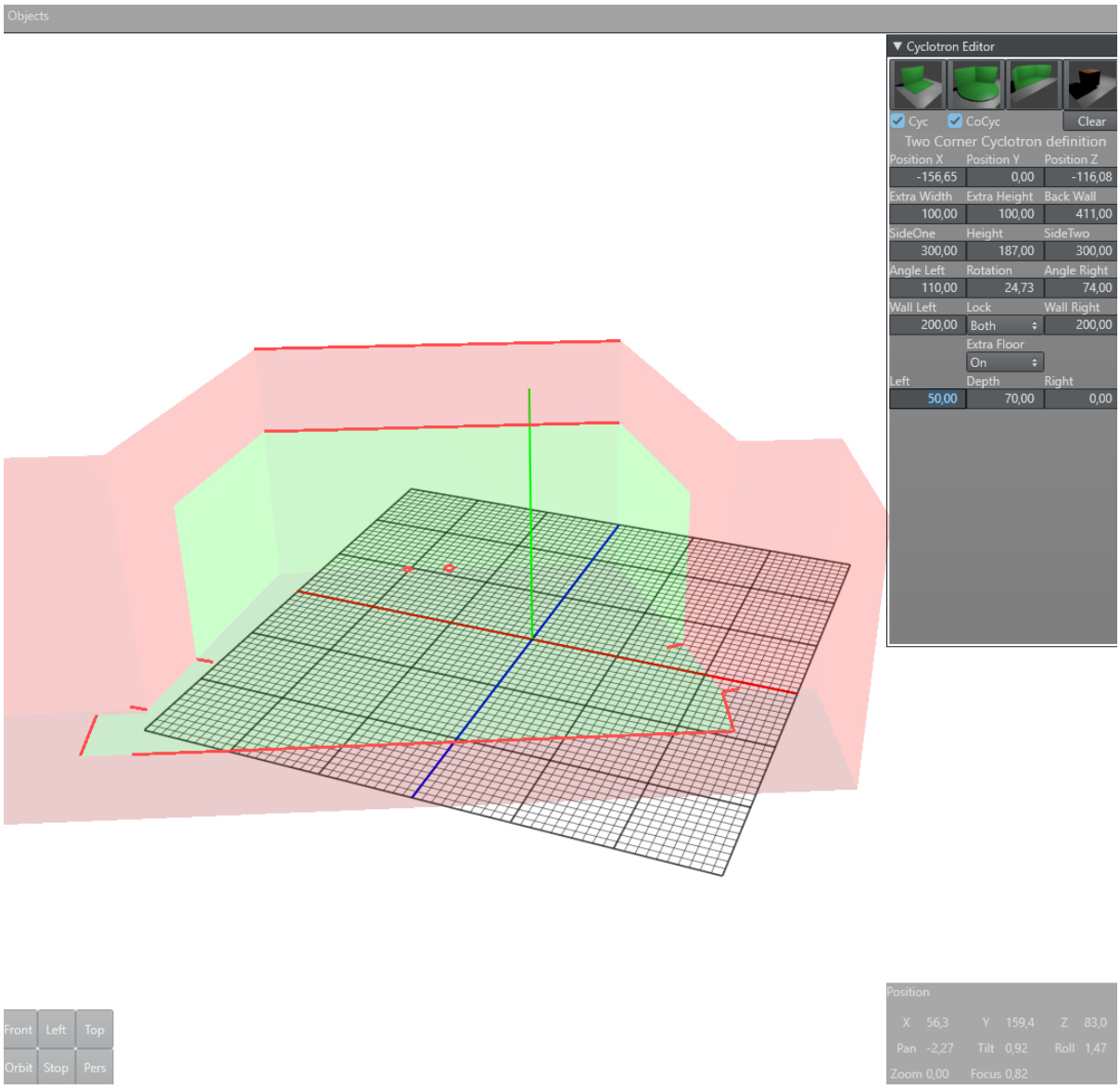
The One Corner Cyc is defined by two sides (side one and side two) and an opening angle. Additionally, the shape of the ground plane can be selected. The user can choose Round, Straight and Rectangular form.

Copyright © 2026 Vizrt



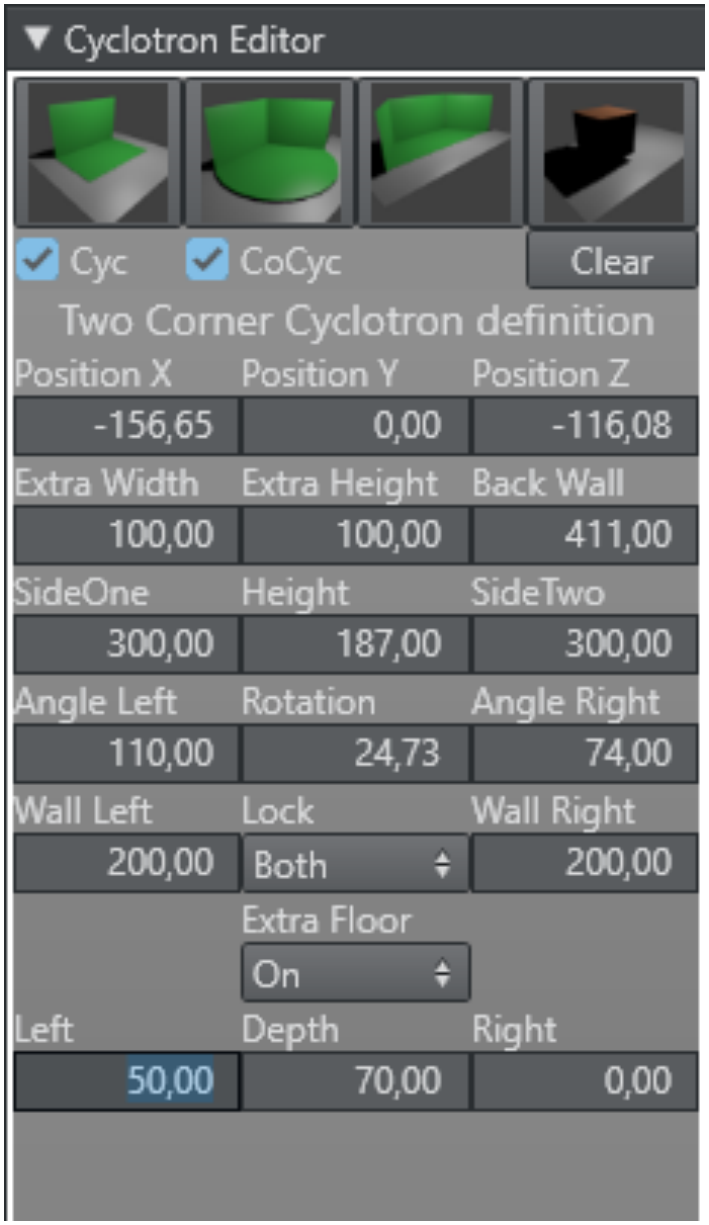
Copyright © 2026 Vizrt

6.15.5 Two Corner



The Two Corner Cyc is defined by the length of SideOne and SideTwo, two opening angles (AngleLeft and AngleRight). It is possible to define the green area of the sides shorter than the side itself. This can be done with the parameters Wall Left and Wall Right. It is also possible to define an extra floor area. This extra floor is added in front of the green walls and extends the shape of the floor.

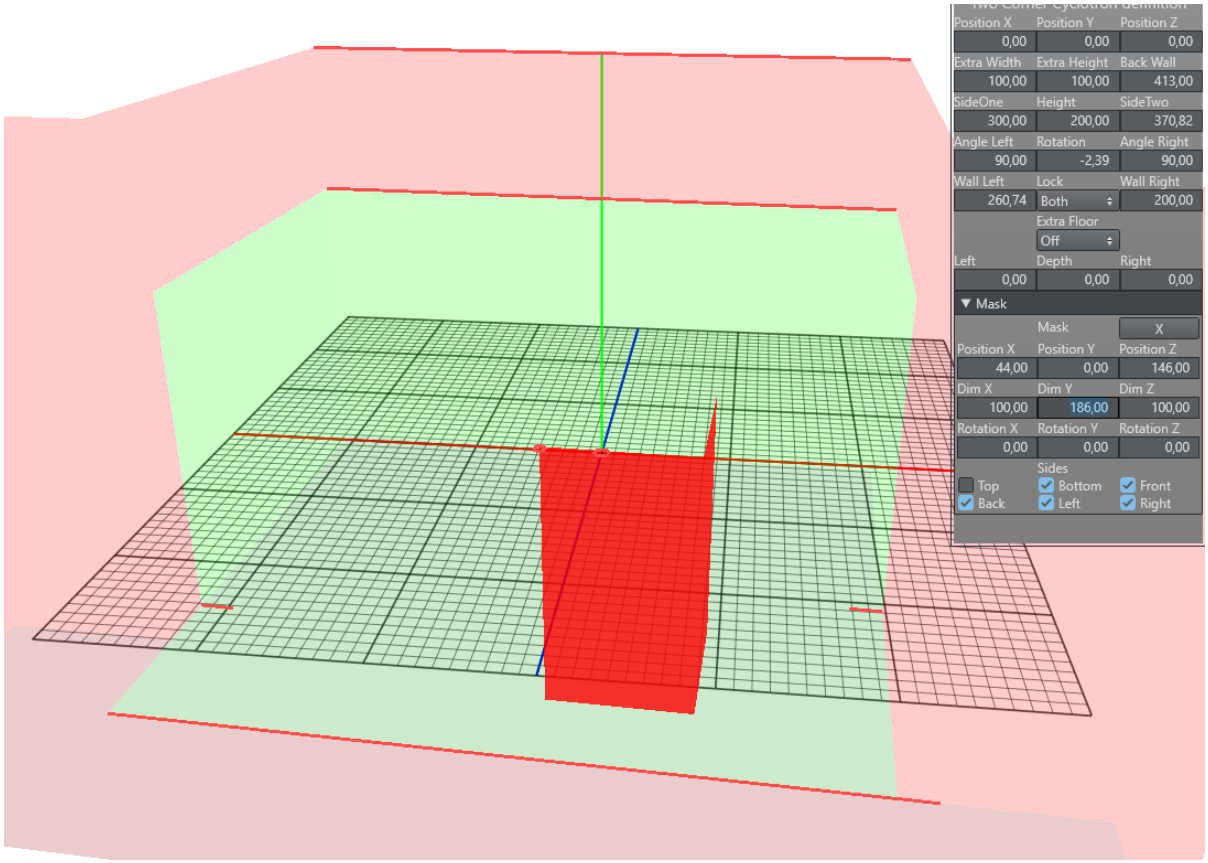
Copyright © 2026 Vizrt

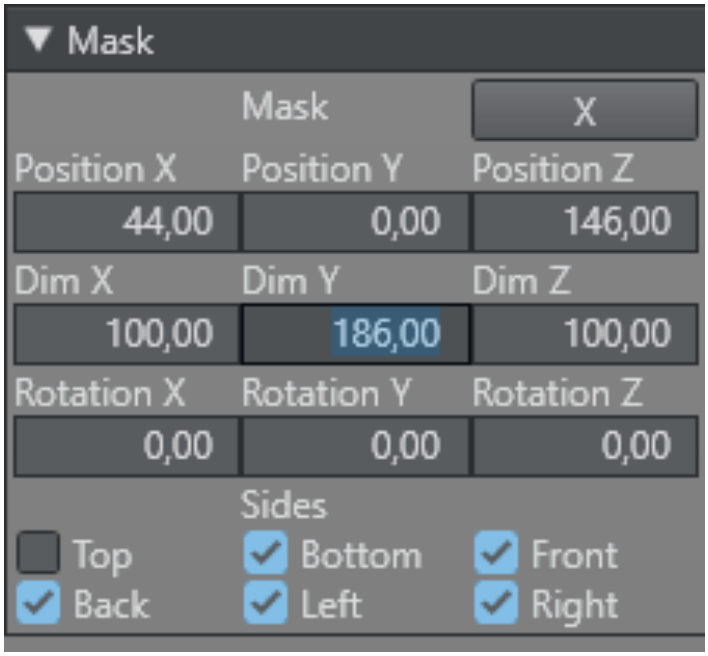


6.15.6 Masks

For any CYC model, any amount of Masks can be added. Masks are elements which are added to the CoCyc walls. For any mask element the position, scale and rotation can be defined in the Cyc Editor. It is also possible to select for every sides of the cube if it should be rendered or not. In the tracking hub, the amount of masks is not limited.

Copyright © 2026 Vizrt





Copyright © 2026 Vizrt

6.15.7 See Also

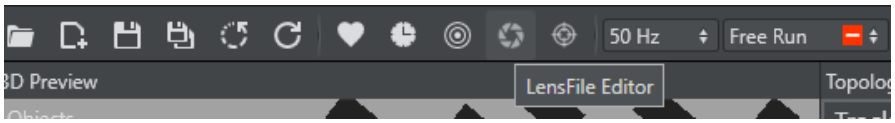
- Preview Panel
- Videowall Editor
- Configure the Studio

Copyright © 2026 Vizrt

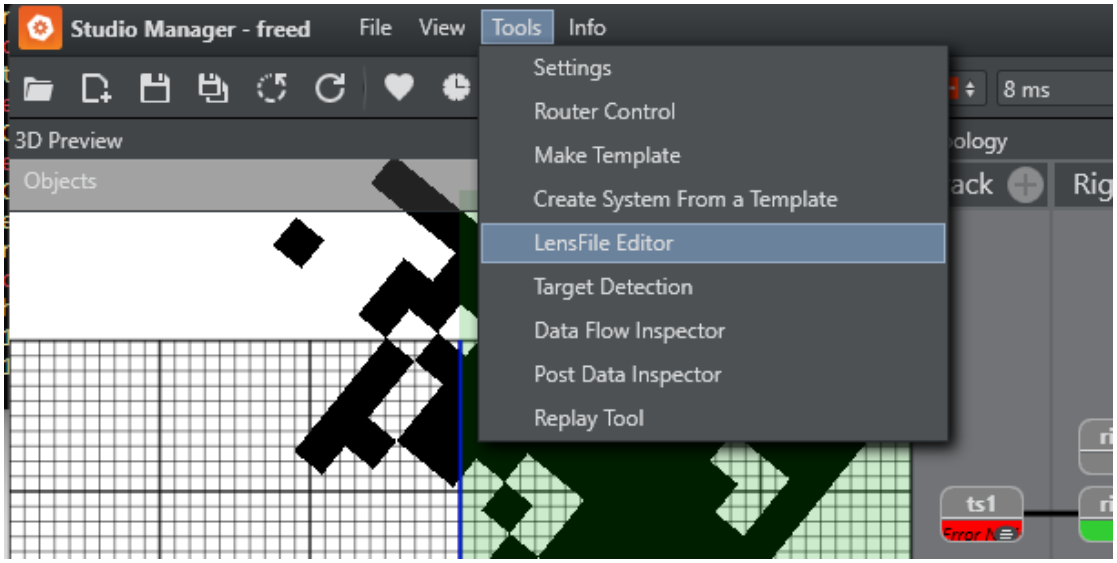
7 Lens File Editor

- New Lens File Format
- Lens File Editor
    - Camera Selection
    - Lens Area
        - File
        - Metadata
        - Settings
    - Group Portlet
    - Engine Control
    - Calibration Scene
    - Parameter Control and Views
    - 3D View
        - Navigation in the 3D View
        - Point Selection
            - Point Selection
    - Point Matrix
    - 2D View
        - Navigation in the 2D View
    - Shortcut System

The **Lens File Editor** is now part of the Studio Manager. Open by clicking the Lens icon in the toolbar:



Or selecting **LensFile Editor** in the tools menu:



7.1 New Lens File Format

As the Lens File Editor uses groups for related parameters, the format of the lens file had to be changed. The new format is human readable XML.

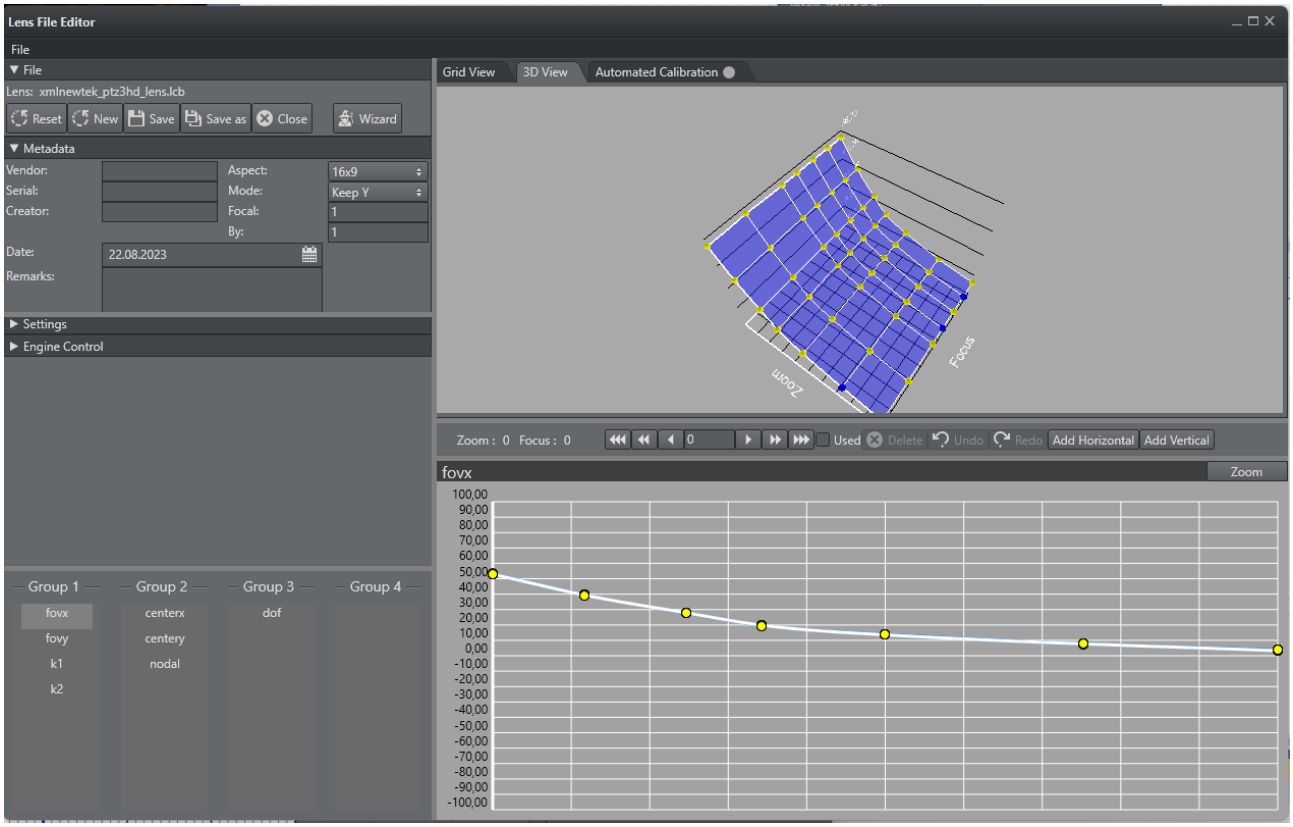
Conversion from the old file format, which was used by Viz Engine and older Tracking Hub versions, is done by Tracking Hub during load time. All old lens files found in `C:\ProgramData\vizrt\VizTH\Lensfiles` are automatically converted to the new file format. The converted file is given the prefix `xml_legacy_`, and spaces in the name are replaced. The original file is moved to the subfolder `C:\ProgramData\vizrt\VizTH\Lensfiles\old_lcb`.

In Tracking Hub, it is then possible to add a lens file rig, which interprets the new file and sends the data to Viz Engine.

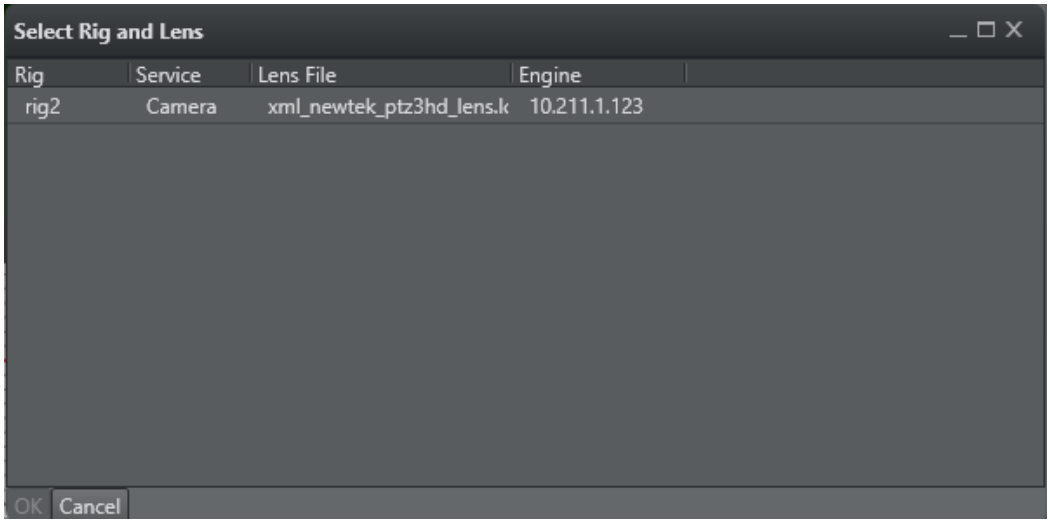
7.2 Lens File Editor

The main window consists of six regions or portals, and several modal windows.

Copyright © 2026 Vizrt



7.2.1 Camera Selection



After the connection to Tracking Hub is established, the Lens File Editor shows all of the lens files ready to be calibrated. The **Rig**, **Service** and **Engine** (ID of the camera service) columns, identify the Rig/Camera combination to be calibrated. The **Lens File** indicates the name of the file loaded in the rig structure.

Copyright © 2026 Vizrt

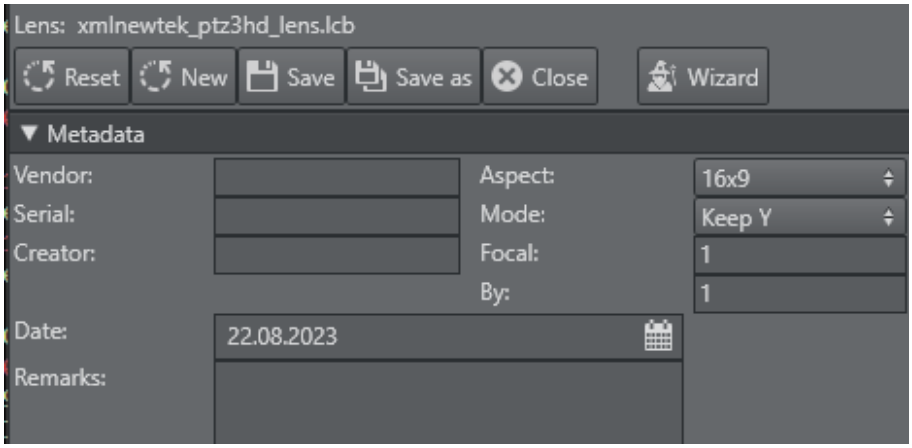
A lens file can be attached to more than one service. As the user selects a Rig/Camera combination, a copy of the lens file is loaded into the selected service. All other camera services are not affected.

When a combination is selected, it is locked in Tracking Hub and cannot be calibrated from another Studio Manager. The lock is released as soon as the editor window closes. It is therefore possible to calibrate different lenses at the same time, but not the same rig/camera combination.

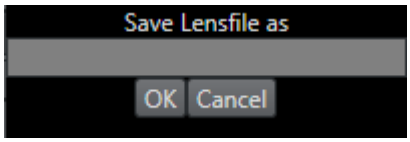
To start a calibration, select one of the table rows and press **Calibrate**. The window disappears and the lens data is loaded into the window.

7.2.2 Lens Area

File

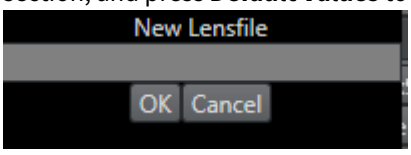


The **Save** button forces Tracking Hub to store the lens data into the *lensfiles/lensfile* folder of the configuration directory. When selecting a combination of Rig/Camera, the name of the lens file is added into the *File* text field. When pressing **Save**, the loaded lens file is overwritten. **Save As** opens a popup where you can enter a new lens file name:



After pressing **OK**, the actual lens are saved on Tracking Hub, and automatically loaded. You need to save the configuration in Studio Manager to make the changed lens file permanent in the configuration.

- **New:** Closes the lens file, and allows the user to start with a complete new calibration task. To generate the approximate field of view values, you need to enter the focal length and multiplication values into the info section, and press **Default Values** to calculate an approximation.

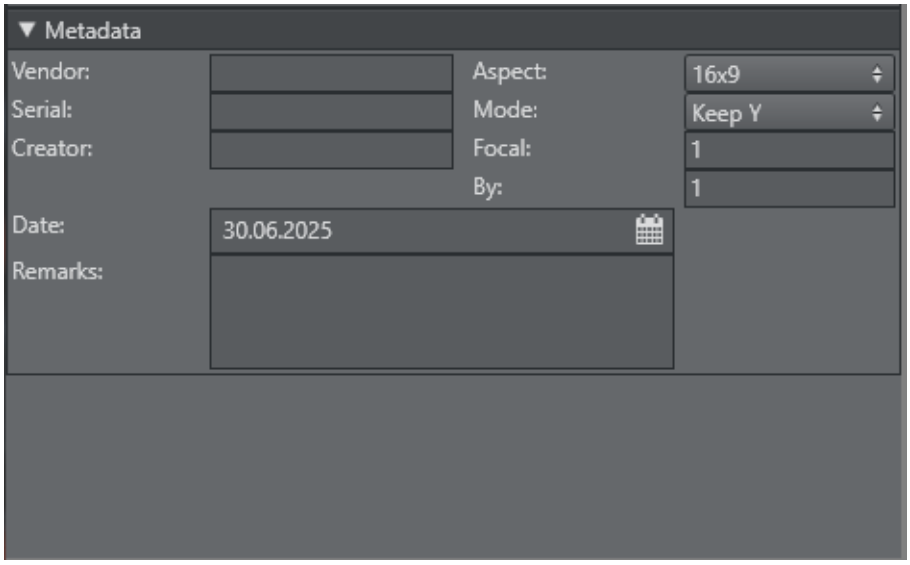


Copyright © 2026 Vizrt

- **Close Lens:** Releases the file lock from these lens, and closes the calibration process. The user is presented with a dialog to select a new rig.

Note: You need to press the **Reload** button in Studio Manager to select a new created lens file.

Metadata

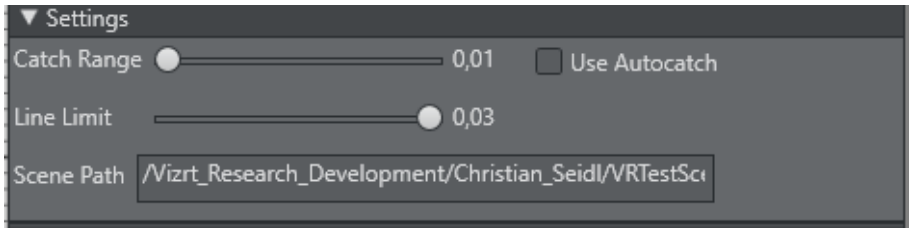


All fields should be filled with care, especially the *Focal Length* and *Focal Multiplier* fields. In case of new lens, the Lens File Editor calculates base FoV values out of this information. The dialog only accepts float numbers in the *Focal Length* and *Focal Multiplier* fields. The **Aspect** mode defines if the lens are used as 16x9 of 4x3.

Info: Tracking Hub calculates the field of view X value from field of view Y. During calibration, this value defines how Tracking Hub adjusts these values while dragging a point. Afterwards, this value is defined in the lens file rig of Tracking Hub.

Settings

In the settings section, you can adjust the catch range of the **Autocatch** feature (see Point Matrix), and the new **Line Limit**.



Copyright © 2026 Vizrt

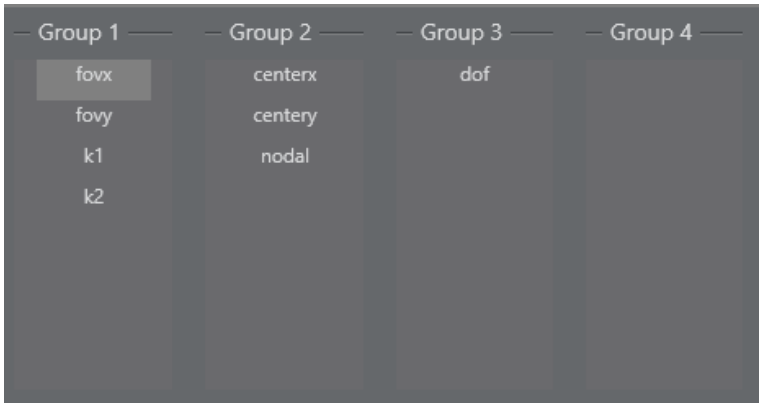
The line limit is used to prevent creating too many lines near one another, although in some cases, it is required to create more lines closer to each other. This can happen when calibrating big zoom lenses for sports.

The default value is `0.03`, meaning that a new line must be 3% distant to another line.

The **Scene Path** is the path to a calibration scene in Viz Engine. This calibration scene is not installed with Tracking Hub, and has to be added manually if needed. Its location path can be set in the UI field, but the best approach is to enter the UUID of the scene there.

This is the scene loaded when using the **Load Scene** button in the Engine Control section Calibration Scene.

7.2.3 Group Portlet



The Lens File Editor introduces the new concept of parameter groups. The old Lens File Editor only had the principle of measure points.

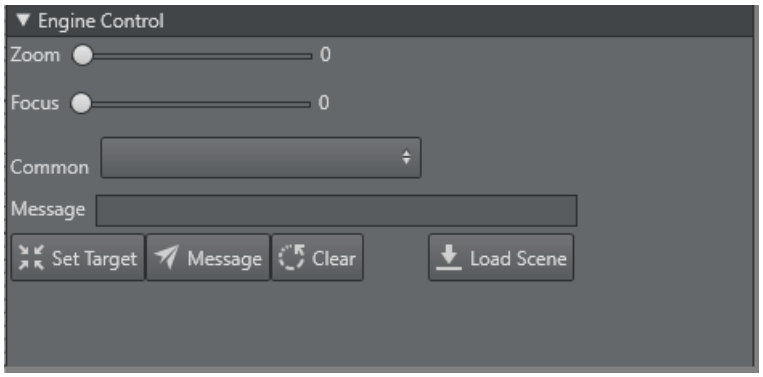
While *field of view* (FOV) and deformation parameters *k1* and *k2,* have to be adjusted for minor zoom and focus changes, *center shift* and *nodal point values*, do not show this sensitive behavior. For these parameters, only a few calibration points are needed, whereas for FOV at least 25 (5x5) or more measure points are necessary. These points are defined by the crossing of horizontal and vertical zoom, and focus lines that can be moved, deleted and inserted at any time during calibration.

By moving a parameter from a 7x7 group into a 2x2 group, only four parameters are mapped into the new group.

Tip: Always create the groups before you start calibration.

Copyright © 2026 Vizrt

7.2.4 Engine Control



The Viz Engine portlet works in combination with the calibration scene.

It is also possible to select one of the common messages in the **Common** drop box, or enter any other message in the **Message** text field. By clicking the **Send Message** button, the message is displayed on the Viz Engine screen, and can be cleared by clicking the **Clear** button.

The **Load Scene** button forces Viz Engine to load the scene the user entered in the **Calibration Scene** of the settings section.

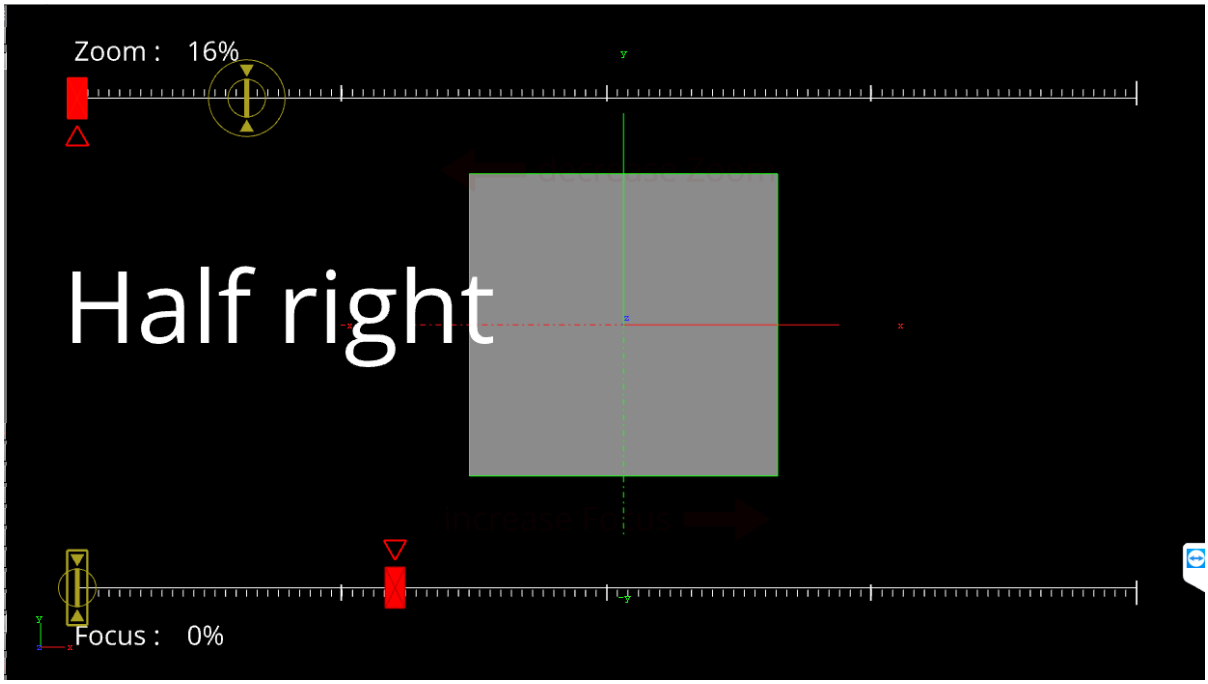
Important: The calibration scene is only for the Viz Engine Classic Render Pipeline. Please make sure your scene uses the Classic Render Pipeline scene in the Main layer.

7.2.5 Calibration Scene

Viz Tracking Hub comes with a default calibration scene (as viz Archive). This scene is designed to be used with the lens file editor, and allows interaction with the camera operator.

Info: The archive of this scene is located on the FTP Tracking Hub download folder.

Copyright © 2026 Vizrt

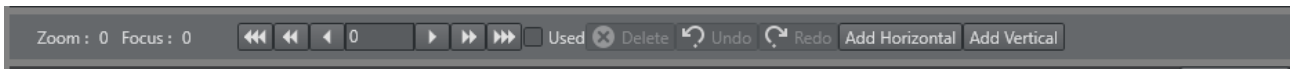


Info: The scene is loaded by default into the Front Layer of Viz Engine.

The scene must be imported into the database, and then loaded into the Front Layer of Viz Engine. In combination with the Engine Control portlet, the person running the calibration of the lens, can communicated with the camera operator.

This special scene shows the target points that need to be calibrated by changing zoom and focus values, and visually assist the operator. New target points can be sent to Viz Engine using the **Send** button. Whenever a new point is selected, whether in the 2D or 3D portlet, the new zoom and focus values are set in the **Zoom** and **Focus** fields of the Engine portlet.

7.2.6 Parameter Control and Views



The Parameter portlet and the 2D and 3D views work closely together. Whenever a point or line is selected in the views, the parameter control changes to the selected value, and reflects all possible input on the selected objects.

With the 2D View Controls, a user can select the diagram type of the 2D display. Type can define two values, Zoom and Focus:

- **Focus:** The X axis defines the focus range (`0` to `1`), while the Y axis shows the value of the calibrated points.
- **Zoom:** The X axis defines the zoom range (`0` to `1`), while the Y axis shows the value of the calibrated points.

Copyright © 2026 Vizrt

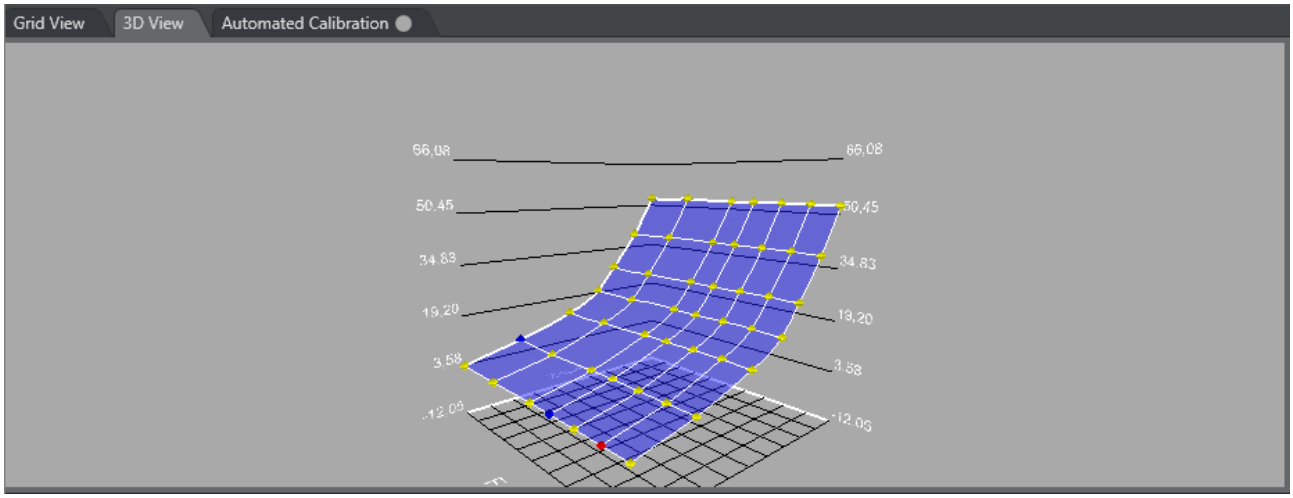
During calibration, the focus diagram is the preferred one.

In case of a selected point, the value of the point can be modified. The modify divisor defines the amount of change, when the user clicks on the up and down buttons of the value. The base of this value is defined as the zoom area in the 2D and 3D view. `10` means that the value is increased or decreased, the zoom is divided by 10, 100 or 1000. This is valid for line movement as well.

A selected line can be moved between its left and right, or upper and lower neighbor. The Modify Divisor again defines the step size of the movement.

As soon as a point has been modified, the point is displayed in yellow and its value are used in the spline interpolation. The yellow state can be cancelled by unchecking the **Used** checkbox, and it activates the blue state of the point, defining its value from the calibrated points around.

7.2.7 3D View



The 3D View shows the lens in a 3D Diagram. Focus is drawn in the X-Axis, and Zoom in the Z-Axis. The lens are drawn as a blue surface, and the measure points as spheres. Depending on the status of the measure points, different colors are used.

By clicking on the **Points** button, the display can be changed to a 2D view of the measure points.

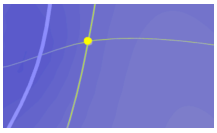
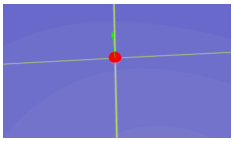
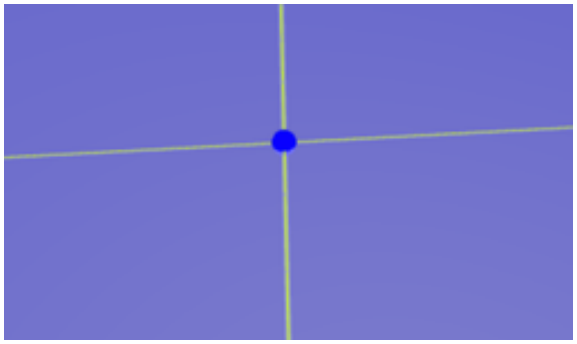
Navigation in the 3D View

Rotate the view by right clicking into the view, hold the right mouse button and drag the mouse. Move the view by clicking into the view, hold the right mouse button and press **SHIFT**.

- **Zoom:** Zoom in and out with the mouse wheel.

Point Selection

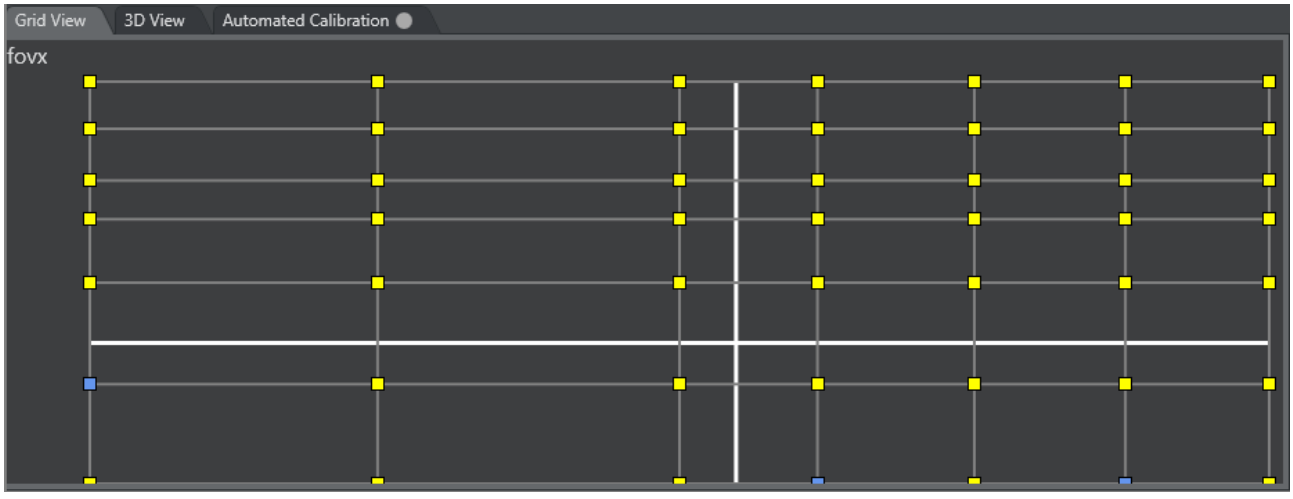
A point can have three statuses:

 A mandatory point is drawn as a yellow sphere.	 When the point is selected, its color changes to red.	 A point that is not mandatory and is untouched, is drawn in blue (such points are only calibrated if needed). If untouched, the point is placed on the surface of the spline. A selected, untouched point is drawn as light blue. As soon as the point is moved by the operator, it changes to touched.

Point Selection

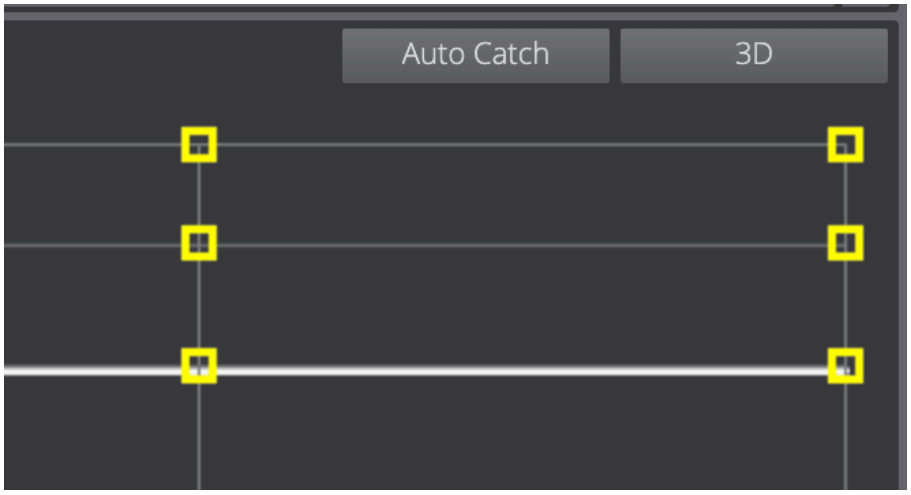
Clicking the point selects the point and changes the parameter control.

7.2.8 Point Matrix



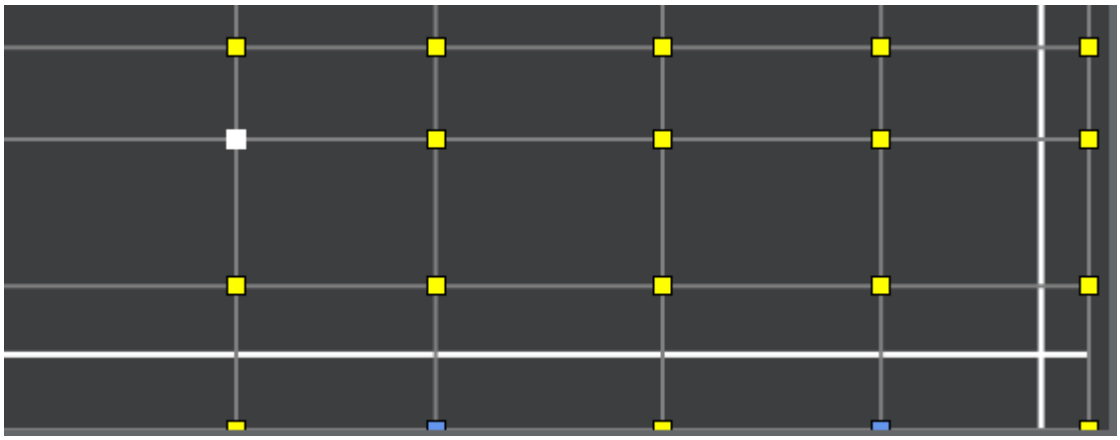
The point matrix displays all points and lines of a group. The X-axis represents the focus, and the Y-axis the zoom value of a point. Zoom and Focus are normalized to the width and height of the Portlet. Points can be selected by clicking on the rectangle, and lines can be selected by clicking on them.

Copyright © 2026 Vizrt



If **Auto Catch** is activated, the point under the cross hair is selected when the distance of zoom and focus of the point goes under a limit. If the cursor is out of range it is red, and as soon as it comes into the range of a point, it becomes white.

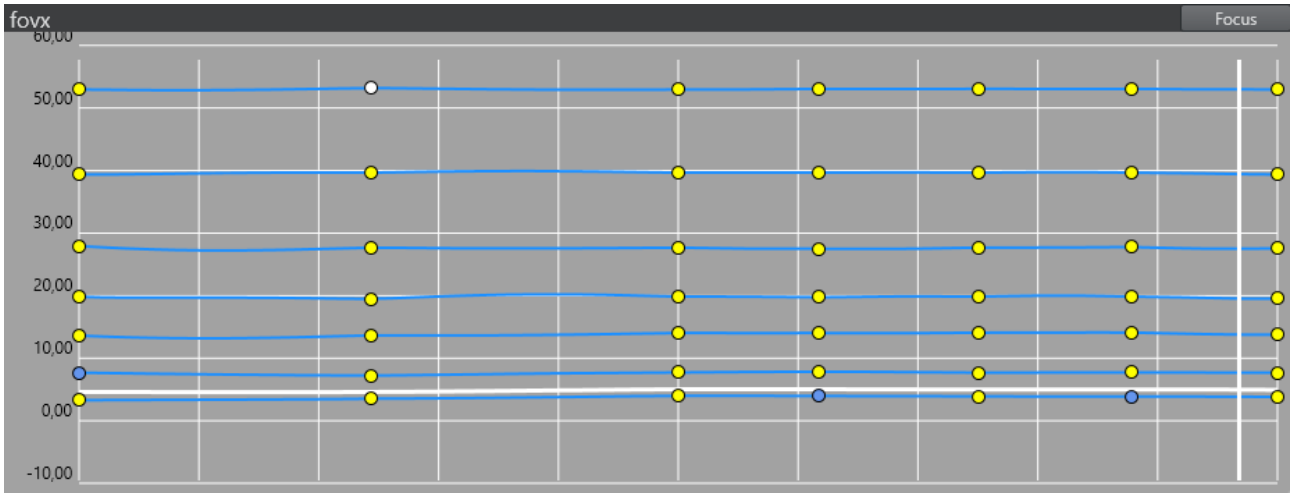
Used points are displayed as yellow rectangles, while unused points are in blue. If a point is selected, it is white.



7.2.9 2D View

The 2D View is the main view used for calibrating the lens points. Clicking into a point, selects the point in 2D View and 3D View, and dragging the point manipulates the value. Using the mouse wheel, you can zoom in and out (and increase the accuracy of the manipulation).

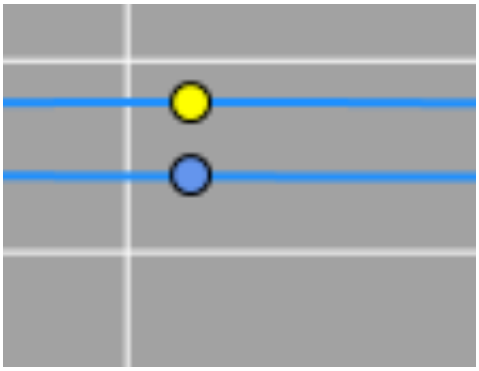
Copyright © 2026 Vizrt



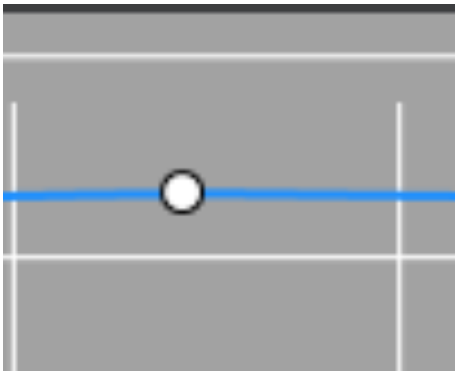
Navigation in the 2D View

- **Move:** Left click with the mouse in the 2D View, keep the left mouse button pressed and move the mouse. This moves the diagram.
- **Zoom:** Use the mouse wheel to zoom in and out.

Different States of Points



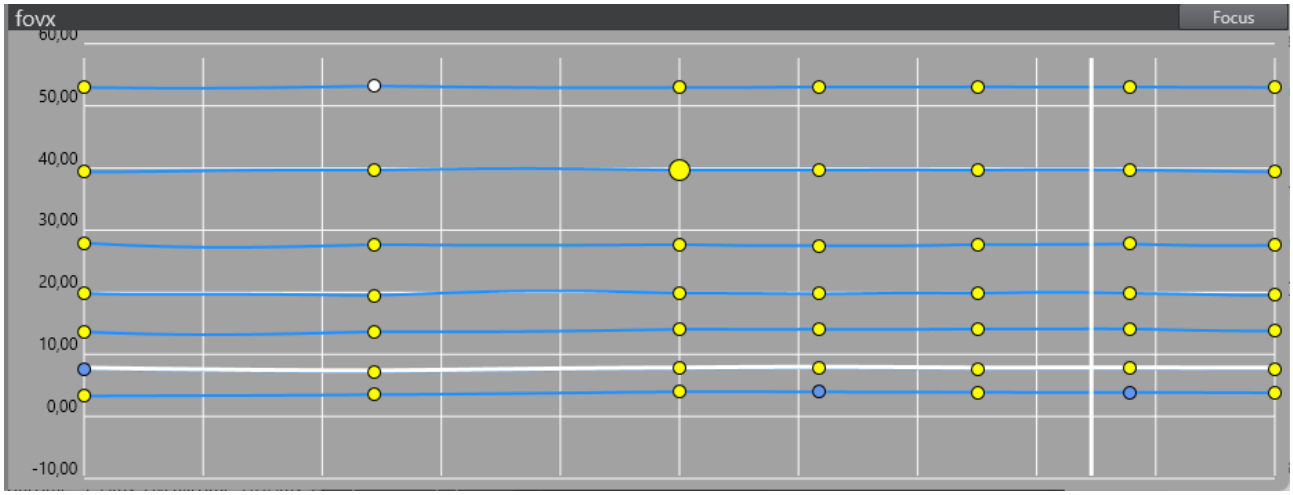
A point that is not mandatory and untouched (blue), and a touched point (yellow).

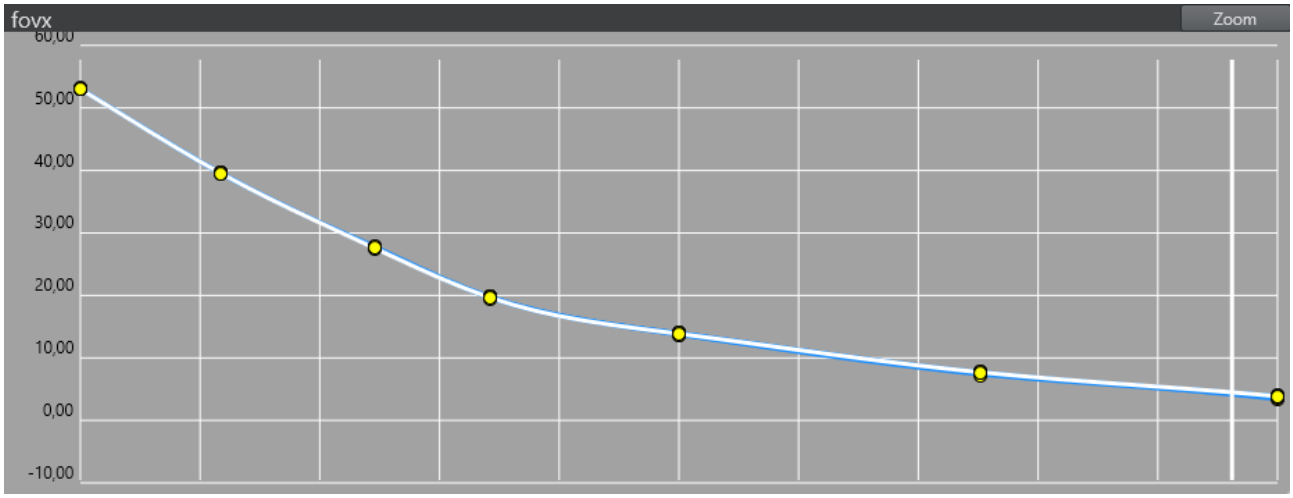


A selected point.

The 2D view can be switched between **Zoom** and **Focus** mode (upper right).

Copyright © 2026 Vizrt





7.2.10 Shortcut System

The lens file editor implements a shortcut system, enabling the user to operate the editor with keyboard shortcuts only.

Shortcut	Change Parameter
CTRL + 1	Field of View X
CTRL + 2	Field of View Y
CTRL + 3	K1
CTRL + 4	K2

Copyright © 2026 Vizrt

Shortcut	Change Parameter
CTRL + 5	Center X
CTRL + 6	Center Y
CTRL + 7	Nodal
CTRL + 8	Depth of field
Shortcut	Change Parameter
SHIFT + Up	Decreases the value of the selected point by 1/1000th of its actual value
SHIFT + Down	Increases the value of the selected point by 1/1000th of its actual value
CTRL + Up	Decreases the value of the selected point by 1/100th of its actual value
CTRL + Down	Increases the value of the selected point by 1/100th of its actual value
SHIFT + CTRL + Up	Decreases the value of the selected point by 1/10th of its actual value
ALT + Cursor Keys	Moves selected point for one row or column
CTRL + R	Rescales 2D view

Copyright © 2026 Vizrt

7.3 Automated Lens Calibration

The automated lens file calibration tool, allows the operator to create lens files without manually adjusting the lens parameters. It provides an easy solution for most of the known broadcast lens types.

Starting the lens calibration can be achieved by either using the general lens calibration settings in the Automated Lens Calibration Panel, or via the Lens Calibration Wizard.

7.3.1 Preparations

> **Note:**
> - A correct lens calibration can only be achieved if the tracking system is delivering correct positional data.
> - Automated calibration requires addition GPU resources. Refer to section Recommended Hardware Configuration.

The automated lens calibration expects the correct CCD position of the physical camera, in order to adjust the lens file, and its internal offsets properly.

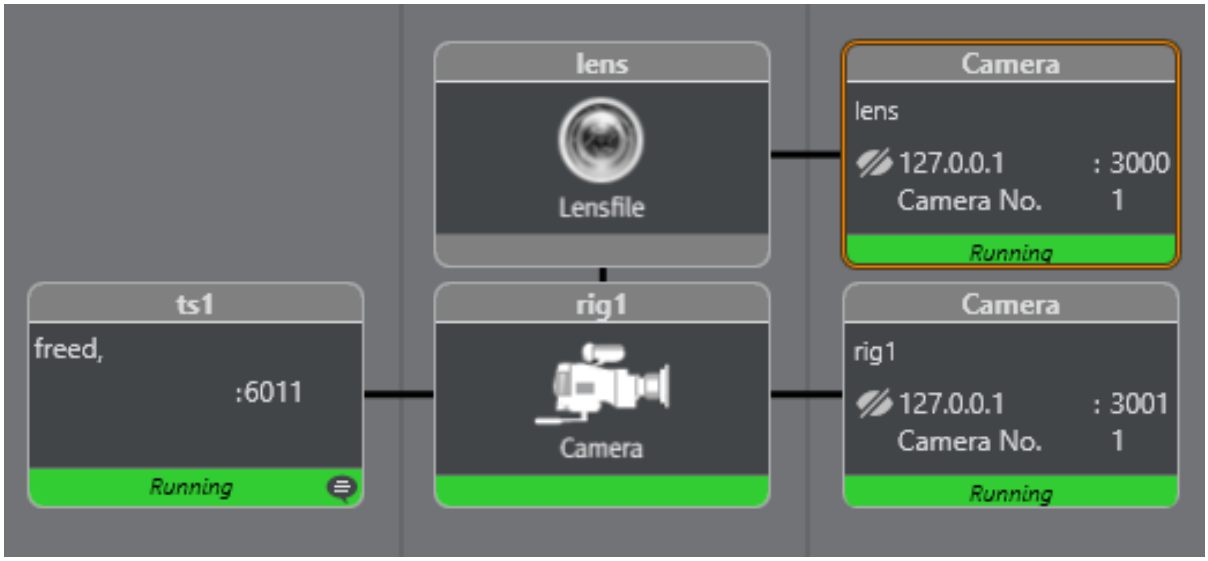
Make sure the following criteria is met:

- Tracking system and offsets in camera rig are set up to distribute CCD position.
  - Add Height Lens Shift, Right Lens Shift and Front Lens Shift (if needed).
  - Verify by measuring the CCD position, physically in the studio, if possible.
- Lens limits for Zoom and Focus are calibrated.
- No Center shifts are set within the camera rig.
  - Automated process is calibrated inside the lens file.
  - If a lens gets remounted on the camera body afterwards, the lens shifts need to be adjusted starting from `0.0`.

7.3.2 Start Process

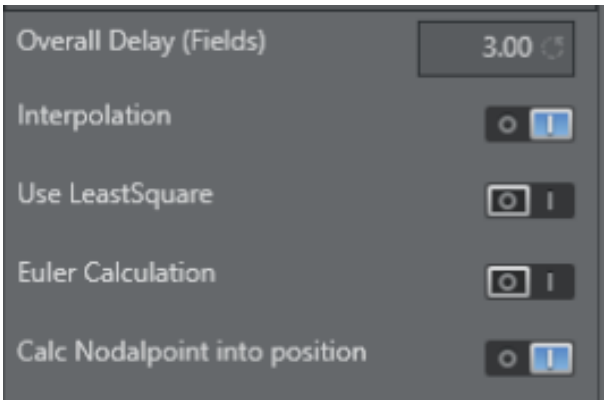
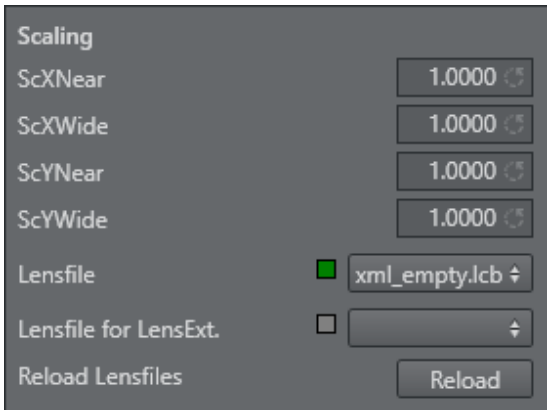
Starting with a default rig setup: A tracking system connected to a camera rig with a lens file node as a parent, containing the previously described needed offsets/adjustments to receive the cameras CCD position (see also Configure Topology).

Copyright © 2026 Vizrt



Two services are added, one to Viz Engine, and a second one to send the tracking data to the calibration tool.

The port for the second service needs to be set to `3001`. When using the Wizard, this service is created automatically.

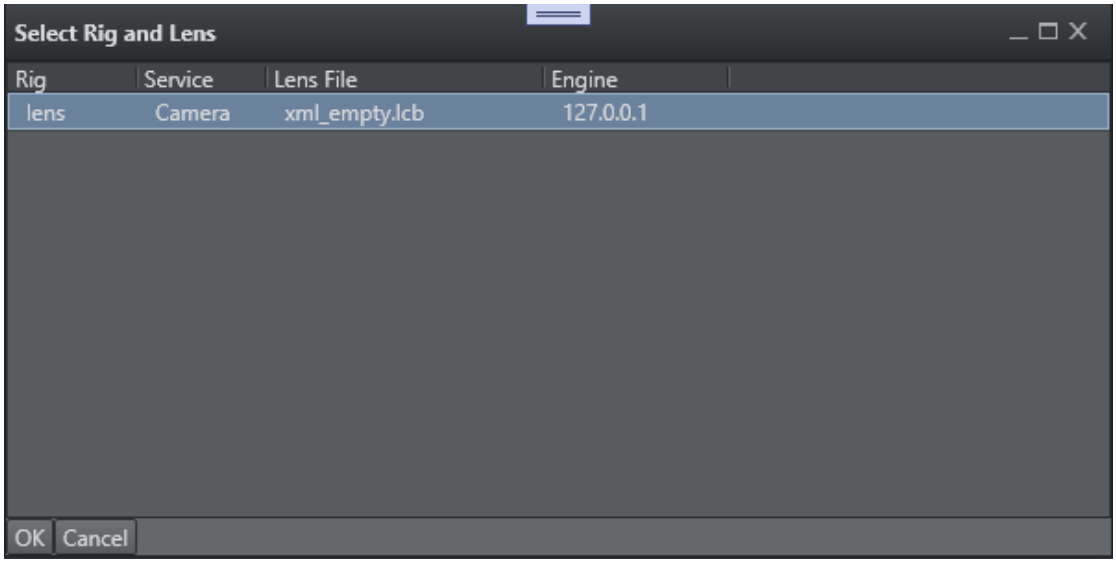
Camera Rig	Lens Rig
	 In this example, an empty lens file named *xml_empty,* is used. Any lens file can be used, while a new one can be created after opening the lens file editor as well.

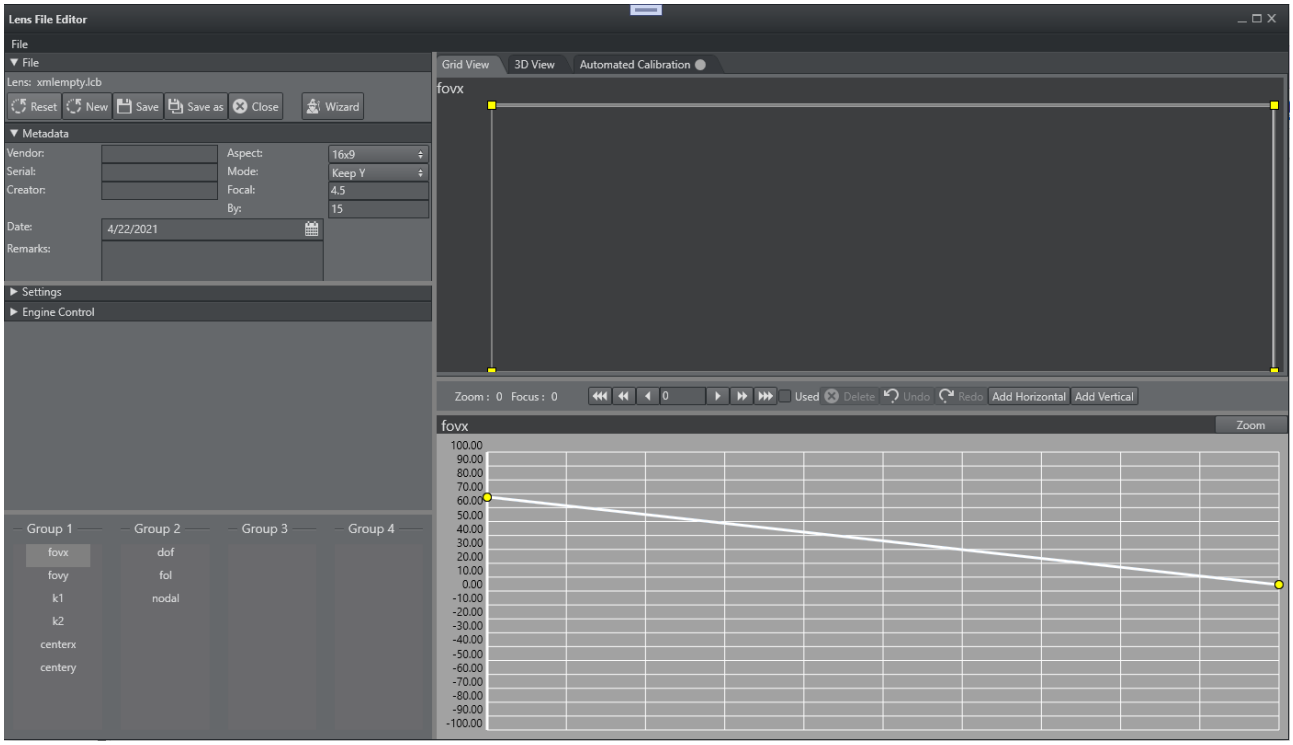
Note: Make sure the option **Calc Nodalpoint into position** is chosen, as this is mandatory for lens files calibrated with the automated calibration.

Open Lens File Editor

Choose the lens file assigned to the rig to be calibrated..

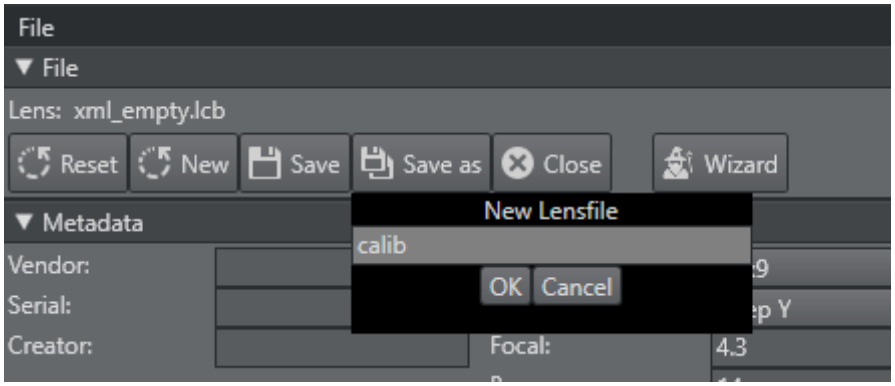
Copyright © 2026 Vizrt





With the lens file editor you can reset the current lens, or create a new one to start from scratch. Please note that a reset, results in overwriting the currently loaded lens file, once saved.

Copyright © 2026 Vizrt



Create a new file if necessary, and start the Wizard, or continue with the Automated Calibration Panel**.**

7.3.3 Prepare Targets

The hard plastic targets come in two fixed sizes (large and small), but any size can be used. PDFs for printing targets in A4 or A5 can be found on the FTP under /products/VizVirtualStudio/Latest Individual Installers/1.8.x.

Targets should be placed in front of the camera, no exact positioning, or any measurement is needed. At least two targets should be visible when the camera is zoomed out.

Copyright © 2026 Vizrt



A smaller third target should be added for zoom lenses, if the default A4 target does not match the cameras FOV, when fully zoomed in.

Copyright © 2026 Vizrt



7.3.4 Start Calibration

The lens calibration tool can be started in two different ways:

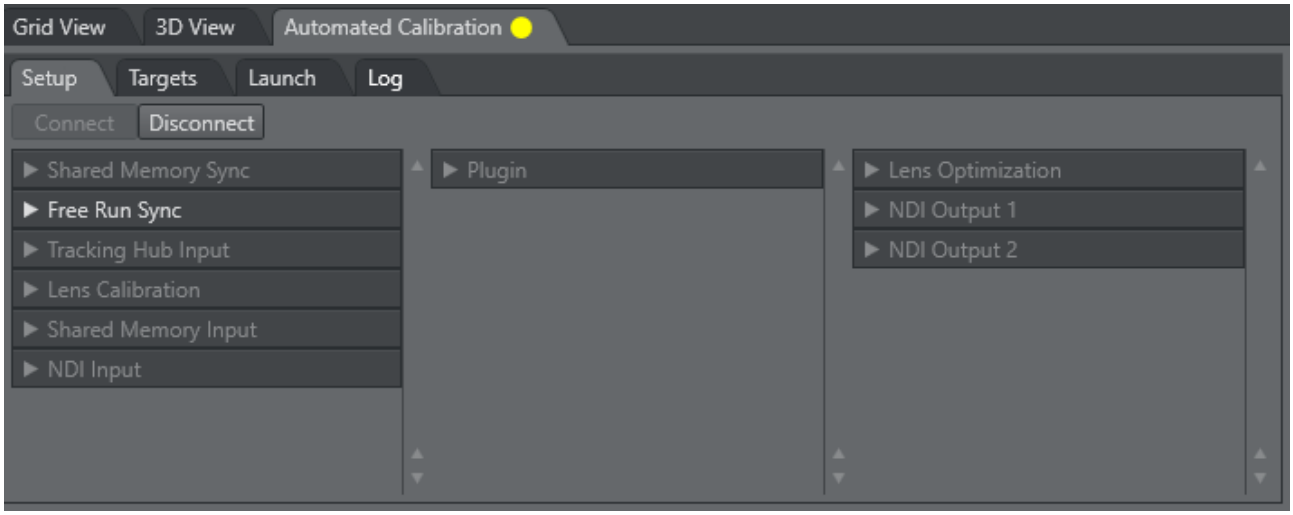
1. Directly out of the Lens Calibration Panel, after configuring the necessary parameters.
2. By using the Lens Calibration Wizard, and following the steps to achieve a proper lens file.

Copyright © 2026 Vizrt

7.3.5 Lens Calibration Panel

- Calibration Tab
- Setup the Automated Lens Calibration Tool
    - Continue From
    - Shared Memory Sync
    - Free Run Sync
    - Tracking Hub Input
    - Lens Calibration
    - Shared Memory Input
    - Plugin
    - Lens Optimization
    - NDI Output
- Targets
    - Adding Targets
- Launch

Calibration Tab



The calibration tool contains four tabs:

- **Setup:** Configures the tool for specific needs.
- **Targets:** Sets up the used targets, defining dimensions.
- **Launch:** Starts/stops the tool, and optionally adds lens points.
- **Log:** Prints the current status of the calibration tool.

Copyright © 2026 Vizrt

Setup the Automated Lens Calibration Tool

The setup tab provides all of the options to configure the tool for different use cases, such as incoming signals (video, sync, tracking, etc.), plugin settings, output, lens file, and NDI rendering.

Continue From

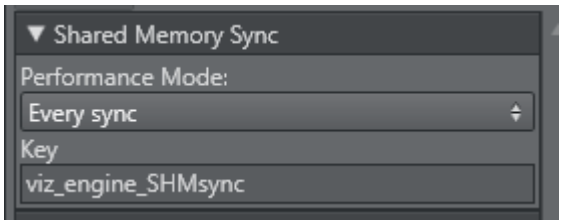
A running calibration can be stored at any point and picked up later, instead of starting a new lens from scratch. These controls sit next to Connect / Disconnect, and become available once the tool is connected.

- **Continue From:** Lists the stored calibrations. Selecting one loads the settings and targets it was made with, and the Start Calibration button on the Launch tab changes to Continue Calibration. Leave the field empty to calibrate a fresh lens.
- **New:** Clears the selection and restores the default settings and an empty target list. A confirmation is requested before anything is discarded.
- **Refresh:** Rescans the folder, to pick up calibrations stored while the panel was open.

Stored calibrations are kept in `C:\ProgramData\vizrt\StudioManager\Data`. Each one consists of a `.dat` file holding the calibration itself, and a `.cfg` file holding the tool settings and targets it was made with. A calibration is only offered in the list when both files are present.

Note: Continuing a calibration replaces the current settings and targets with the ones stored alongside it.

Shared Memory Sync

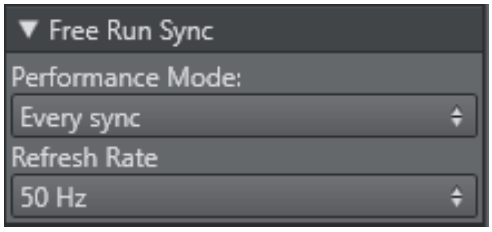


- **Performance Mode:** Defines the sync rate on how often an image gets received within the tool. The rate can be limited depending on the GPUs performance.
- **Key:** Sets the shared memory key to communicate with Viz Engine. *viz_engine_SHMsync* is used by default.

Free Run Sync

The sync source can be set to **Free Run** when no connection to a local Viz Engine sync source is present. This allows to calibrate any incoming NDI signal source without a Viz Engine Scene.

Copyright © 2026 Vizrt



- **Performance Mode:** Defines the sync rate on how often an image gets received within the tool. The rate can be limited depending on the GPUs performance.
- **Refresh Rate:** Sets the refresh rate of the used NDI camera / input signal.

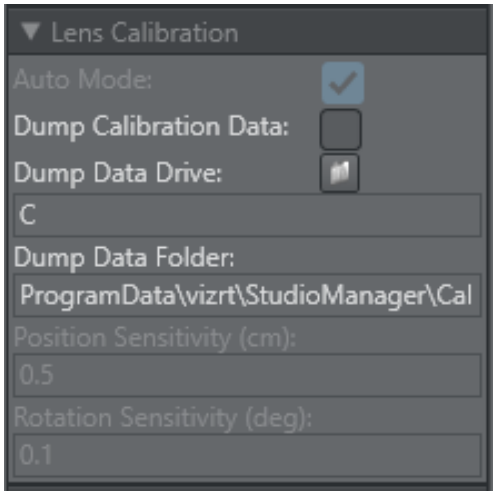
Tracking Hub Input



- **Image Height / Width:** Defines the aspect ratio of received TH camera image. Attributes should not be changed, as long as the standard 16:9 format is used.
- **Port:** Sets the UDP communication port where it receives the tracking information. Default is set to `3001`.
- **Render Scale:** Sets the scale value for deformation calibration. Default is set to 1.2.

Copyright © 2026 Vizrt

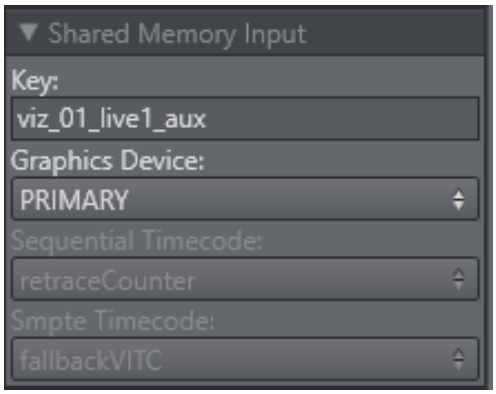
Lens Calibration



- **Auto Mode:** Fetches lens information automatically for a target seen in the defined area. Active by default (to be removed in a later version).
- **Dump Calibration Data:** Collects the images and tracking information for debugging (support cases).
- **Dump Data Drive / Dump Data Folder:** Defines location where dumps get saved.
- **Position / Rotation Sensitivity:** Threshold settings for Image detection. To be modified via config file only.

Shared Memory Input

Parameters to receive the camera input.

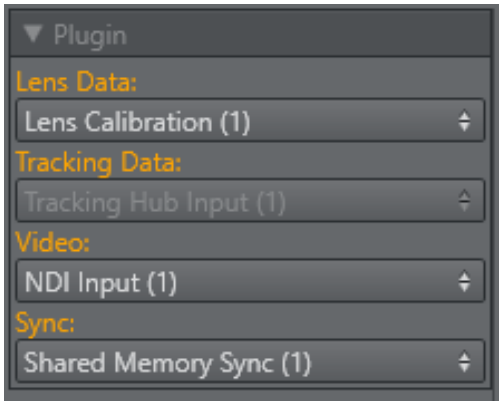


- **Key:** Sets the Shared Memory key of the received input. By default, Live Input 1 is used (*viz_01_live1_aux*).
- **Graphics Device:** Assigns the GPU to be used.

Plugin

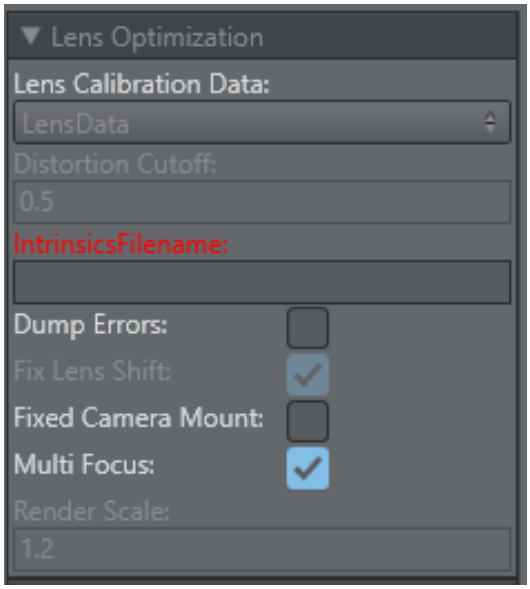
The settings mentioned above need to be assigned to the plug-in (defaults in screenshot).

Copyright © 2026 Vizrt



- **Lens Data:** Name of the node used to calibrate the lens (assigned automatically).
- **Tracking Data:** Sets the name of the node used for receiving the cameras tracking data (assigned automatically).
- **Video:** Sets the source of the incoming video frame. Options available are **Shared Memory Input** (via Viz Engine) and **NDI Input**.
- **Sync:** Assigns sync to be used for calibrating (assigned automatically). In case a direct NDI signal is used, **Free Run Sync** needs to be selected.

Lens Optimization



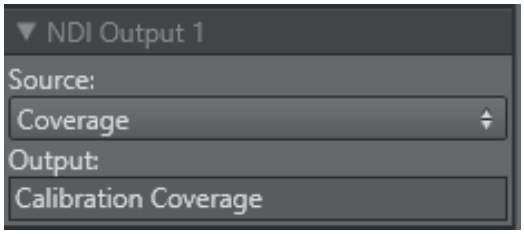
- **Distortion Cutoff:** Sets the percentage of distortion that does not get calculated into the lens file. Only to be set in the configuration file (not recommended).
- **Intrinsics Filename:** Sets the name for the target tracking meta information. This file can later be used to track targets with this lens file.
- **Fixed Camera Mount:** Ignores position tracking data, and the camera is considered static. Used for PTZ cameras.

Copyright © 2026 Vizrt

- **Multi Focus:** Tracks focus when disabled. Can be used for quick files with static focal length.

NDI Output

Shows the actual calibration tool as an NDI signal.



- **Source:** Type of output. **Coverage** shows the calibration tool, **Preview** marks the targets with rendered overlays for previewing the expected calibration result.
- **Output:** Name of the NDI stream in the network.

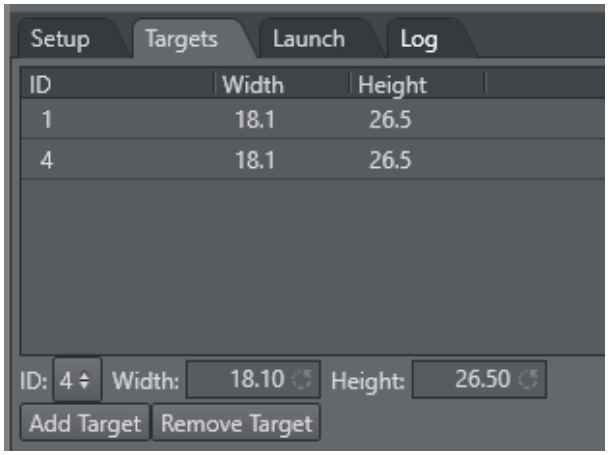
Targets

The Automated Lens Calibration tool uses special targets to detect the optics behavior.

Now four different targets have to be placed in front of the camera strategically. Two targets are usually enough for a good calibrating result, depending on the lens, adding more targets might be useful.

Adding Targets

If the shipped hard plastic targets are not available, the targets can be printed manually. The ID and the target dimensions have to be added in the **Targets** table. The target ID can be found on the printed target itself.



Dimensions of the targets must be entered in centimeters [cm].

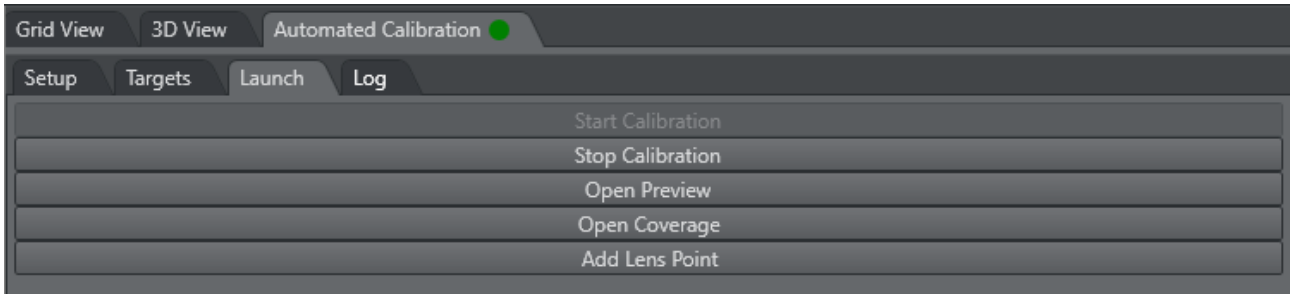
Copyright © 2026 Vizrt



Note: Width and height dimensions need to be measured within the target, not the sheet size.

Launch

Starts the actual calibration tool after all the required parameters are set.



- **Open Preview / Coverage:** Opens the NDI outputs in a dedicated window, locally.
- **Add Lens Point:** Adds an extra lens point at the current Zoom / Focus position to the raw lens file calibration.
- **Start / Continue Calibration:** Starts the calibration. The button reads Continue Calibration while a stored calibration is selected under Continue From, and the run then carries on from it.
- **Save Progress:** Stores the current state of a running calibration so it can be continued later. A dialog asks for the name; reusing an existing name overwrites it. The calibration keeps running. This does not write the lens file.

Copyright © 2026 Vizrt

Start the Lens Calibration Process.

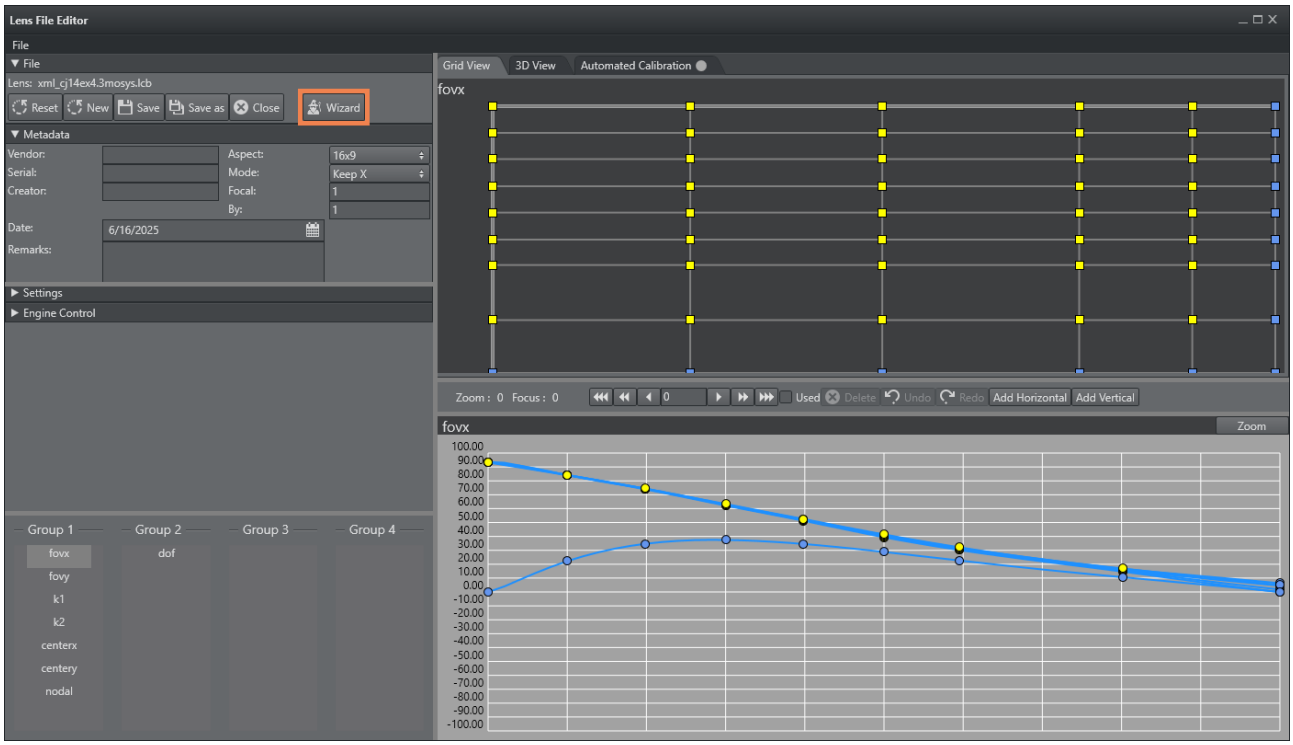
7.3.6 Lens Calibration Wizard

The Lens Calibration Wizard helps launch the Lens Calibration Tool quickly, and setting the required parameters.

> **Note:** The lens file calibration process requires a preconfigured, correctly set up tracking system. Inaccurate tracking, results in an inaccurate lens file.

Startup

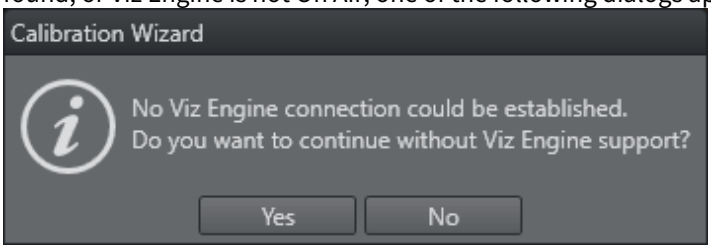
Start the wizard, by pressing the **Wizard** button in the Lens File Editor.

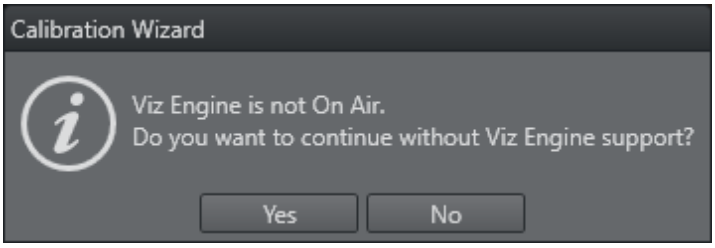


The start dialog of the Lens Calibration Wizard is shown:



1. **Launch:** Starts the Calibration Wizard process and sets up all necessary prerequisites. A check is performed if a connection to the configured Viz Engine camera service can be established. If no valid connection is found, or Viz Engine is not On Air, one of the following dialogs appears:





   In this case, the process can still be continued, but Viz Engine video inputs and live preview of targets are not available. If the usage without Viz Engine is not intended, press **No** and check that the rig setup is as described in Lens Calibration Process, and Viz Engine is On Air. Then press **Launch** again.

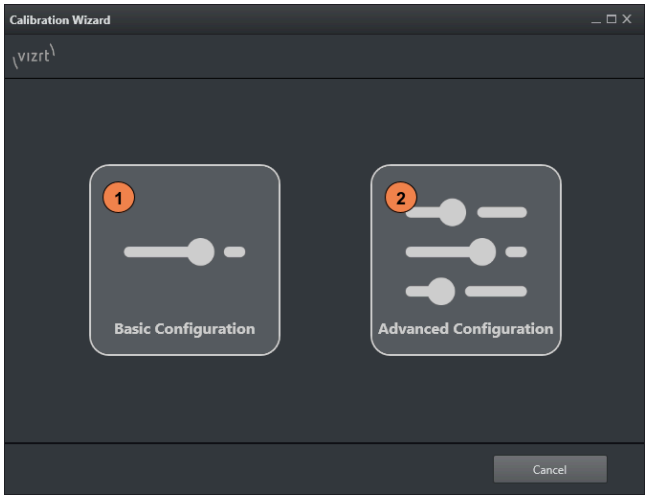
Warning: Starting the wizard process creates and sets a new scene on the configured Viz Engine. Make sure it is not in use before you proceed.

2. **Cancel:** Closes the Calibration Wizard dialog.

Copyright © 2026 Vizrt

Configuration Modes

Once the calibration wizard process has started, a configuration mode has to be selected:



1. **Basic Configuration:** Offers the most essential configuration options, and can be used for a quick setup of the calibration process. Refer to Basic Configuration for a detailed description of all options. If unsure, use this configuration mode.
2. **Continue Calibration:** Continues a calibration that was stored earlier with Save Progress, using the settings and targets it was made with. The wizard skips the configuration and target pages and goes straight to Calibration Controls.
   This mode is only shown when at least one stored calibration is available.

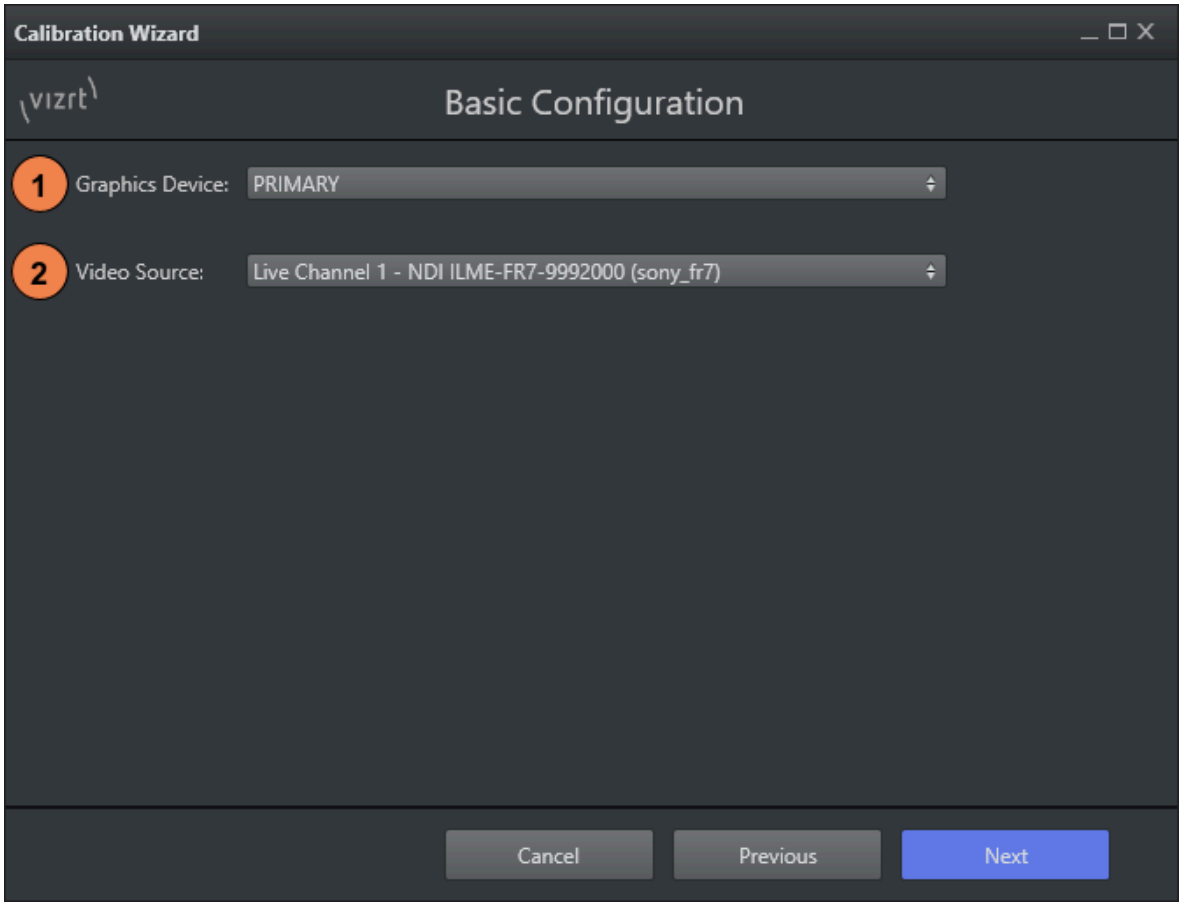
Note: This configuration mode is only available if a Viz Engine 5.4.0 or higher is configured and running on the same host machine as Studio Manager. Otherwise, you are not able to select it.



2. **Advanced Configuration:** Offers more configuration options, and controls the lens calibration process. Refer to Advanced Configuration for a detailed description of all options.

Copyright © 2026 Vizrt

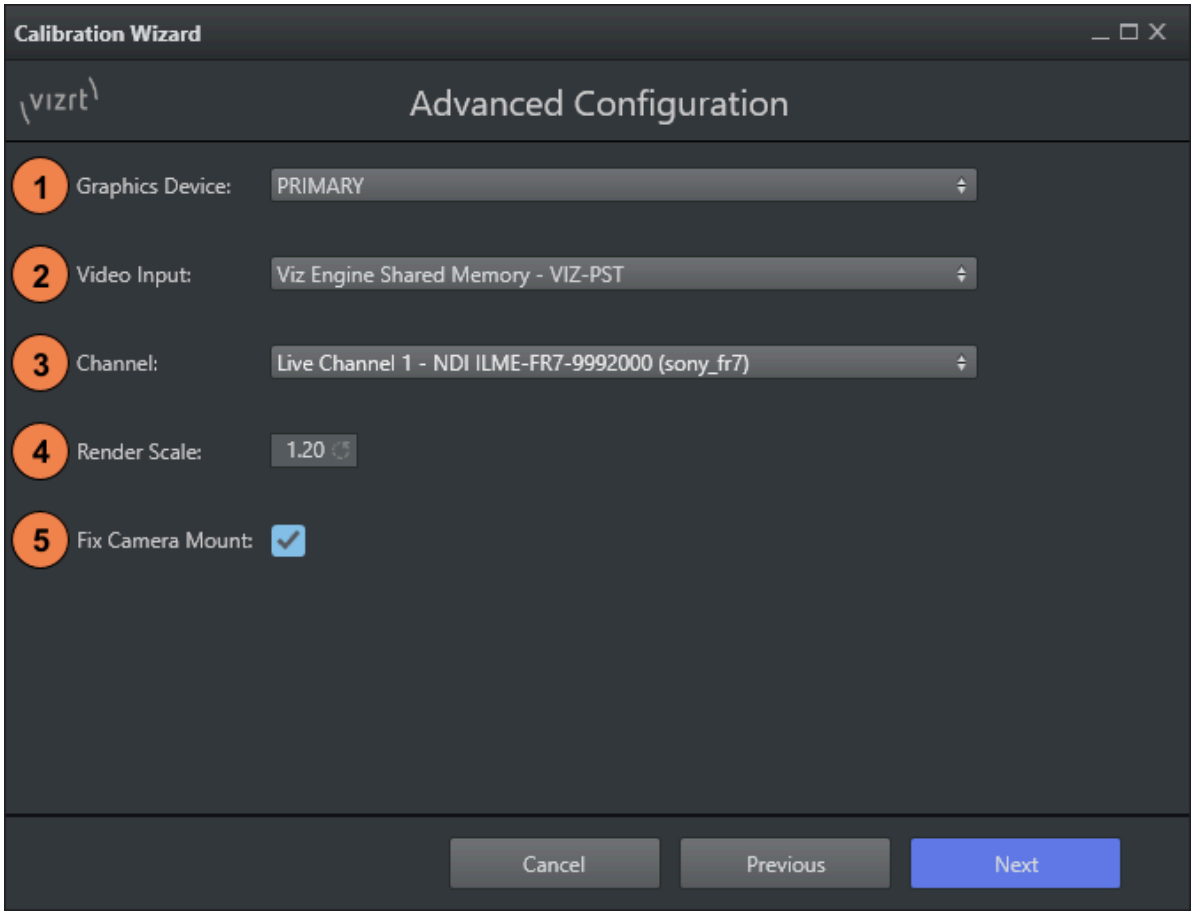
Basic Configuration



1. **Graphics Device:** Determines the graphics device (GPU) used for the lens calibration processing. To reduce the calibration time, select the most powerful device available. If unsure, leave this setting on **PRIMARY**.
2. **Video Source:** Determines the video input source for the calibration. The source can be any live channel configured on the local Viz Engine. Refer to the Viz Engine Administrator Guide for live channel configuration.

Copyright © 2026 Vizrt

Advanced Configuration



1. **Graphics Device:** Determines the graphics device (GPU) used for the lens calibration processing. To reduce the calibration time, select the most powerful device available. If unsure, leave this setting on **PRIMARY**.
2. **Video Input:** Determines the video input mode for the calibration. One of the following modes can be selected:
   a.) **Viz Engine Shared Memory:** Allows to select any live channel configured on the local Viz Engine, as video input. The used live channel is selected by the **Channel** control. This mode can only be used for local Viz Engine instances.

> **Note:** This mode requires Viz Engine to be running on the same host as Studio Manager to be available.

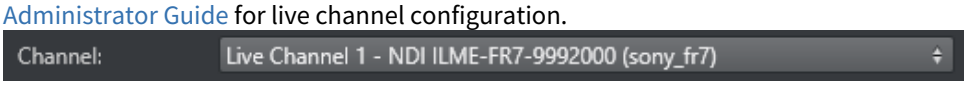
b.) **Viz Engine NDI Forward:** Allows to forward any live channel configured on Viz Engine as an NDI stream, and
use it as video input for the calibration. The used live channel is selected by the **Channel** control. This mode can be used for local and remote Viz Engine instances.

> **Note:** This mode requires Viz Engine 5.4.0 or higher with a valid **Parallel Output** license, and Tracking Hub 1.8.0 or higher to be available.

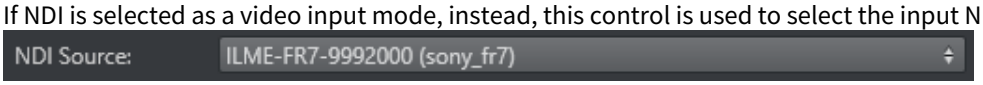
Copyright © 2026 Vizrt

c.) **NDI:** Allows to use any discovered NDI source on the network as video input. The used source is selected by the **NDI Source** control. This mode does not require any Viz Engine instance.

3. **Channel / NDI Source:** If Viz Engine Shared Memory of NDI Forward is selected as a video input mode, this control is used to select the live channel used as video input for the calibration. Refer to the Viz Engine Administrator Guide for live channel configuration.



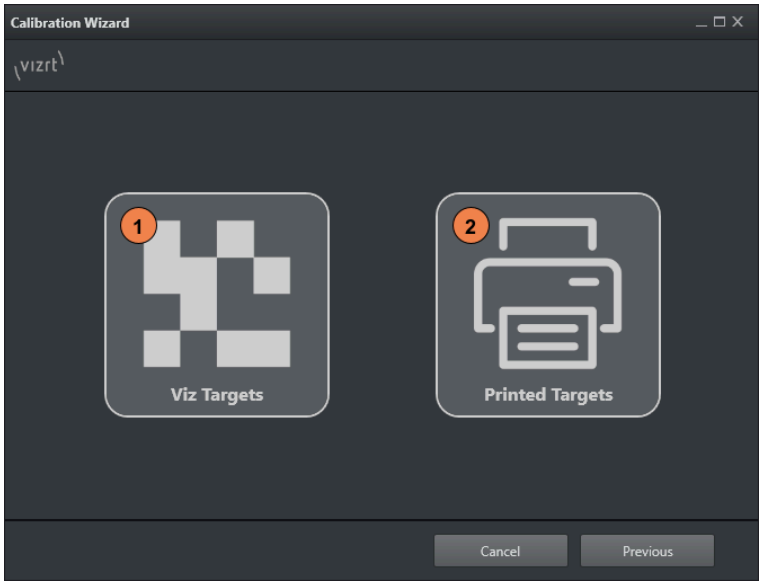
If NDI is selected as a video input mode, instead, this control is used to select the input NDI source.



4. **Render Scale:** Sets the scale value for deformation calibration. The default value for this setting is 1.2.
5. **Fix Camera Mount:** If this option is set, position tracking data is ignored, and the camera is considered static. Enable this setting if a PTZ camera is used.

Target Modes

After the configuration is finished, one of two target modes has to be selected.



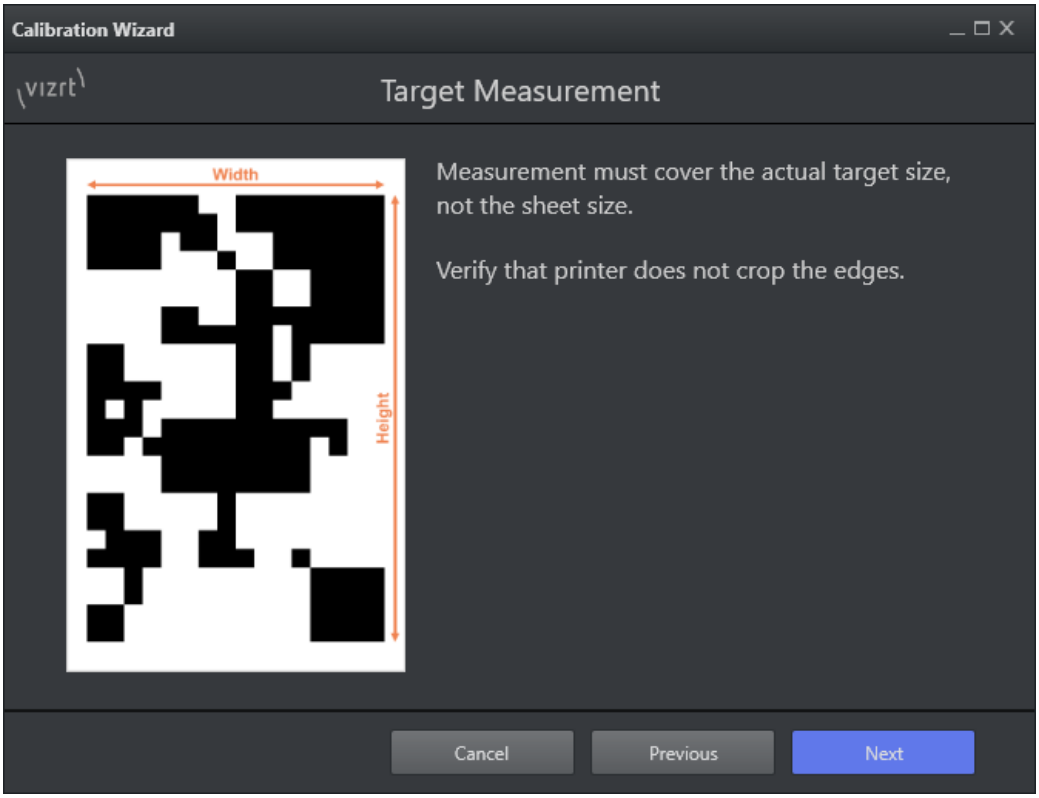
1. **Viz Targets:** Select this mode if targets provided by Vizrt are going to be used. In this case, continue to Target Selection.
2. **Printed Targets:** Select this mode if custom printed targets are to be used. In this case, continue with Printed Target Measurement onwards.

Printed Target Measurement

For custom printed targets, it is necessary to measure all targets used individually. PDFs for printing targets in A4 or A5 can be found on the FTP under /products/VizVirtualStudio/Latest Individual Installers/1.8.x.

- Verify that the printer did not crop any edges of the original template.
- Then measure the width and height of the calibration pattern as accurate as possible.
- The wizard then shows a visualization of the lengths to be measured.

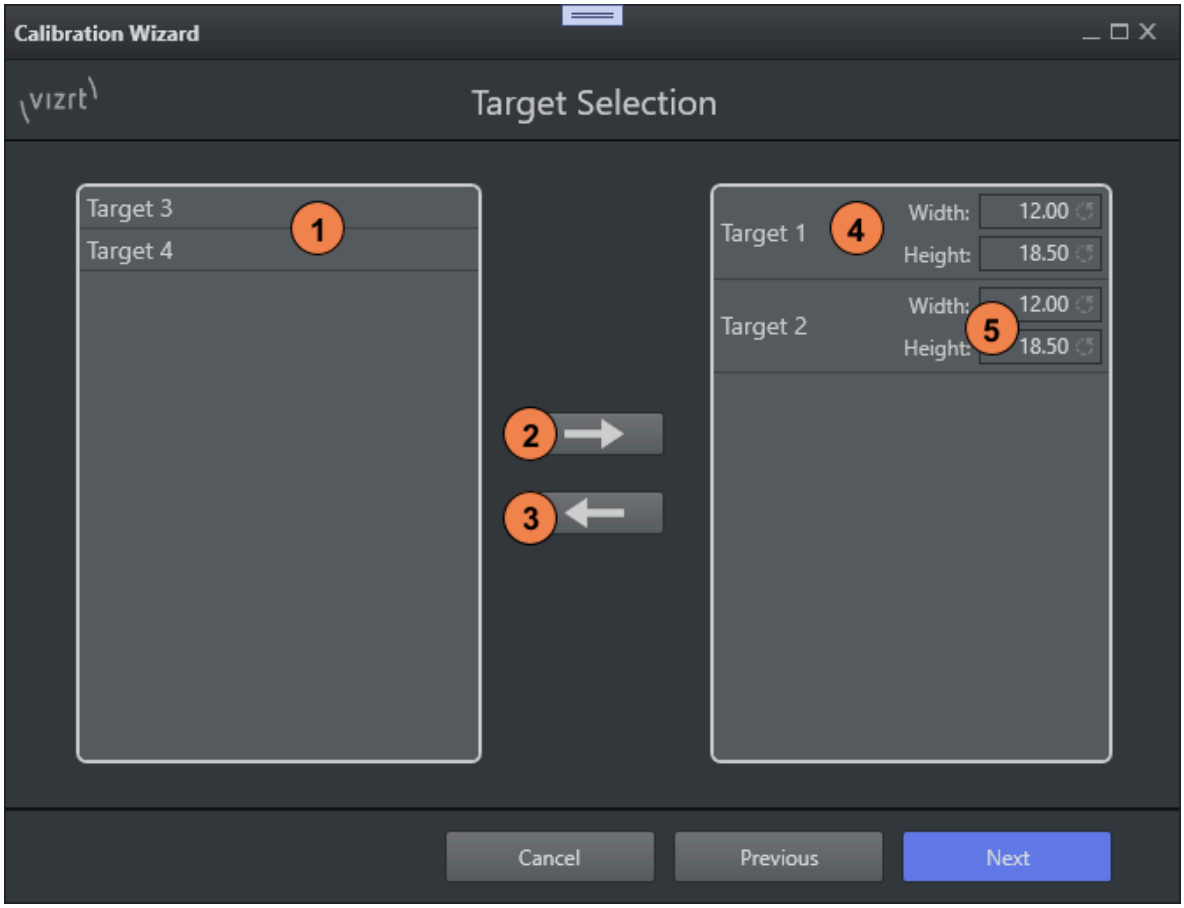
Copyright © 2026 Vizrt



Target Selection

Each target used during the calibration has to be selected. Additionally, the dimension of each target must be specified if custom printed targets are used.

Copyright © 2026 Vizrt



1. **Available Target List:** Displays all available targets for selection.
   Double click any target entry to select it for the calibration. Alternatively, select an entry and use the **Add Target** button. If **Viz Targets** mode was selected, each target has a **Small** and **Big** variant. Only one variant can be selected for each target and they replace each other.
2. **Add Target:** Adds the currently selected target entry to the calibration selection.
3. **Remove Target:** Removes the currently selected target entry from the calibration selection.
4. **Selected Target List:** Displays all targets selected for calibration.
   Double click any target entry to remove it from the selection. Alternatively, select an entry and click the **Remove Target** button.
5. **Target Width / Height:** Each selected target entry has a **Width** and **Height** control to define its dimension. Dimensions must be entered in centimeters [cm]. Width and height are predefined if **Viz Targets** mode was selected.

Target Alignment

After the targets for the calibration are selected, the wizard shows guidelines on how to position them in the studio. Read and confirm each guideline. The wizard only allows to proceed to the next page after all items have been checked. Alternatively, this section can also be skipped.

Copyright © 2026 Vizrt



Copyright © 2026 Vizrt



1. **Check List:** Target alignment guidelines that should be followed. Select an item to mark it as checked.
2. **Skip:** Skips the target alignment check, and goes directly to the Calibration Controls page.
3. **Next:** Disabled until all guideline items are checked.

Calibration Controls

Once all steps have been completed, the wizard moves to the final main page from which the calibration process can be started and controlled.

Copyright © 2026 Vizrt



1. **Start Calibration:** Starts the lens calibration process.
    a. **Continue From:** Selects the stored calibration to carry on from. Shown only when the wizard was started through Continue Calibration; selecting an entry loads the settings and targets it was made with, and Start Calibration then reads Continue Calibration.
2. **Stop Calibration:** Stops the lens calibration process and closes the wizard dialog.
3. **Open Coverage:** Opens the main calibration window. Only available after the calibration has started.
4. **Open Preview:** Opens the preview window showing the estimated target corner positions according to current calibration. Only available after the calibration has started.
5. **Add Lens Point:** Adds an additional calibration point for the current zoom and focus value. Only available after the calibration has started.
6. **Save Progress:** Stores the current state of the calibration so it can be continued later. A dialog asks for the name; reusing an existing name overwrites it. The calibration keeps running and the wizard stays open. This does not write the lens file.

Stored calibrations are kept in `C:\ProgramData\vizrt\StudioManager\Data`. Each one consists of a `.dat` file holding the calibration itself, and a `.cfg` file holding the settings and targets it was made with. A calibration can only be continued while both files are present.

For a detailed description on how to proceed after the calibration has started, refer to Lens Calibration Process.

Copyright © 2026 Vizrt

7.3.7 Lens Calibration Process

- Automated Lens Calibration
- Lens Coverage
- Lens Calibration Procedure
    - Blurry Targets
    - Adding Points
    - Lens Preview Window
    - Test Scene
    - Stop Calibration
- Editing Lens Points
    - Tool Window
    - Cursor Selection
    - Operations on Points
- Save Lens
    - Target Detection Metadata

Starting the lens calibration can be achieved by either using the general lens calibration settings in the Automate Lens Calibration tab, or via the Wizard.

Automated Lens Calibration

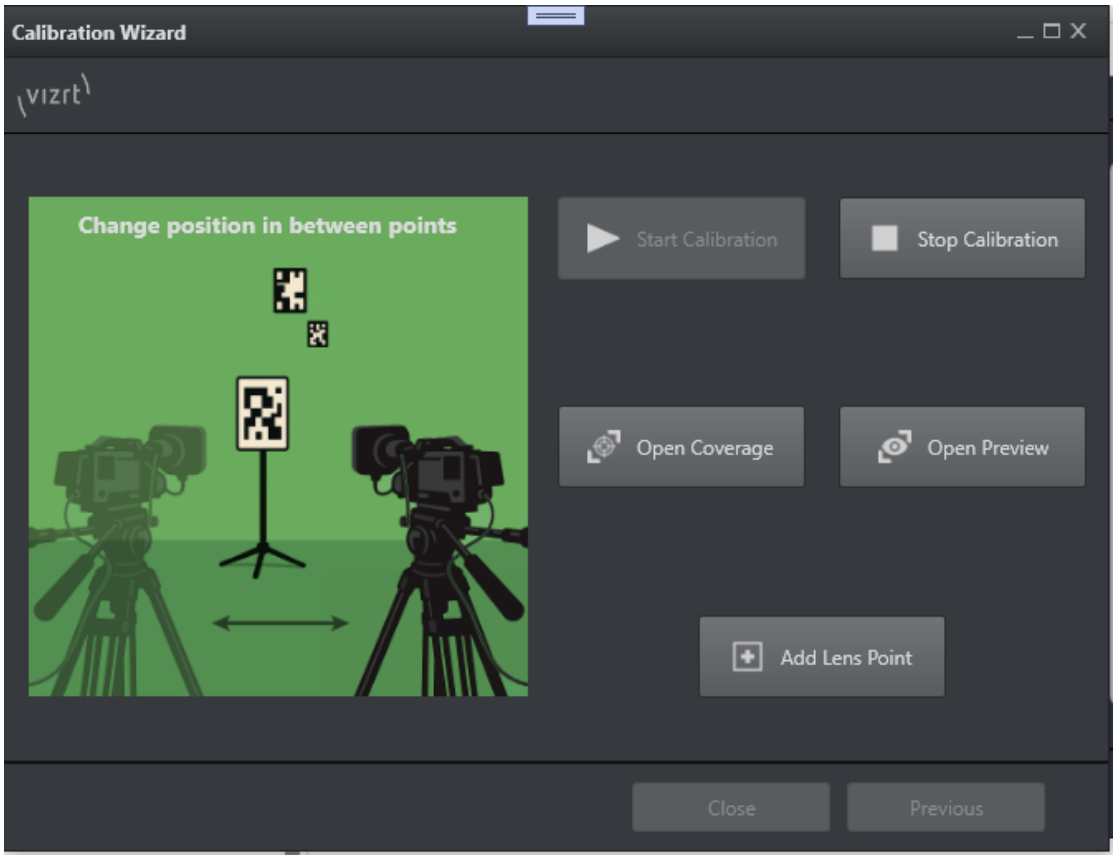
The automated lens calibration process uses image-based calibration techniques to automatically generate a lens file. It detects targets in the incoming images, and tries to compute the target positions and lens parameters that best explain the observed images. To fit an accurate lens model, it is important to capture a wide range of zoom/focus levels.

It is important that the images used by the process do not suffer from motion blur, and that the images and the tracking data are in sync. Therefore, the camera has to stay still for a few frames before the process captures an image to use it for the calibration process.

The coverage and the preview visualizations described in the next section, guide the user through the process of capturing suitable images, ensuring an adequate coverage of the zoom/focus space, as well as input images and tracking data of sufficient quality.

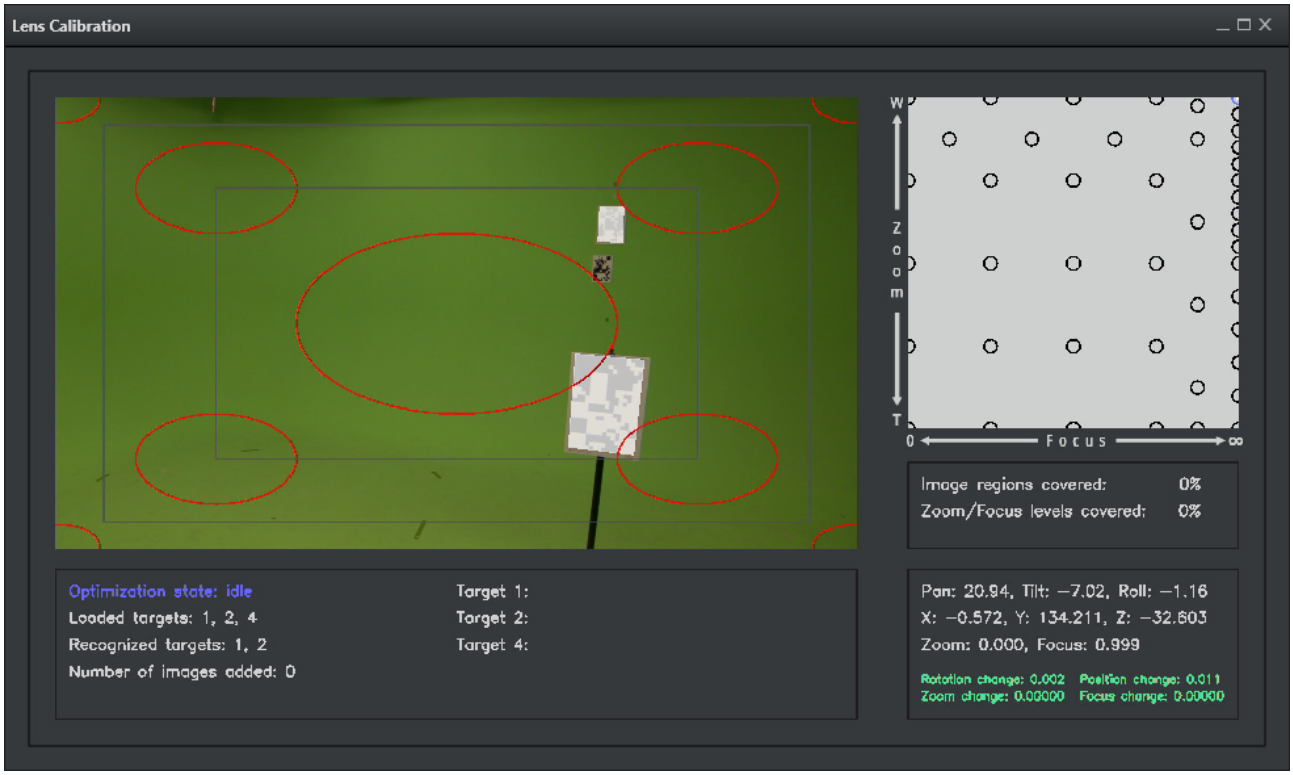
Copyright © 2026 Vizrt

Lens Coverage



The **Open Coverage** button launches an NDI window showing the calibration tool interface.

Copyright © 2026 Vizrt



The coverage panel used for the actual calibration is divided into five sections.

The video panel shows the camera frame, the detected targets rendered as white rectangles on top of the real targets, and nine elliptical shapes which represent the calibration hot zones for the targets to be detected. Once the camera is stable and a target is detected within a hotzone, the hotzone turns green, indicating that the process extracted a useful image covering this image area, with a target at the current zoom/focus level. The hot zones only become visible if the current camera zoom and focus are within one of several predefined areas as described below.

The graph on the right displays a heatmap of the calibrated lens related to zoom and focus. Red indicates an insufficient calibration, while green represents a fully calibrated area. The small circles are predefined zoom/focus areas, lens points, where the calibration can be done. While a rather dense distribution of zoom/focus points are defined by default, manual points can be added during the calibration if necessary (see Adding Points).

For each of the zoom/focus areas in the heatmap, the user should cover all of the hot zones displayed in the video panel.

The three other panels provide information about:

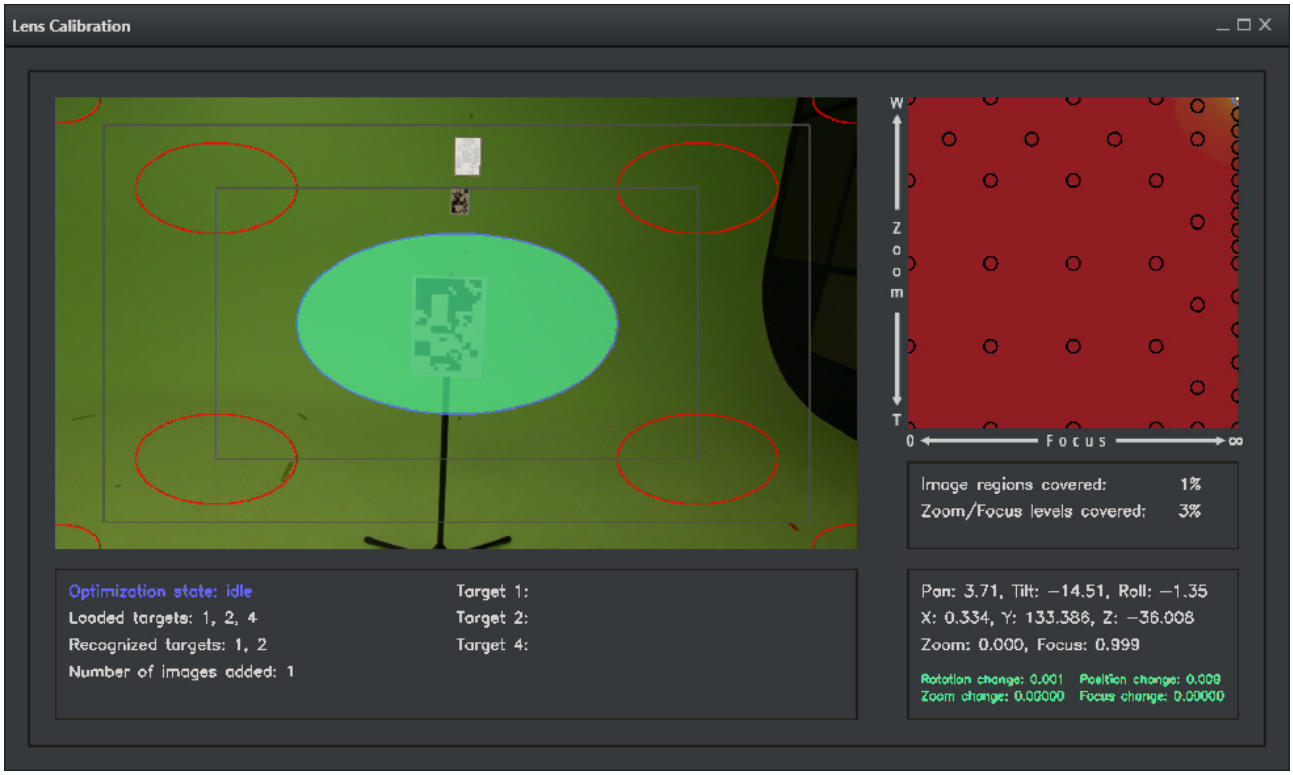
- Covered calibration status in %.
- Detected targets.
- Incoming tracking data.

Lens Calibration Procedure

Start with the first point at zoom 0% and focus infinite (top right). If the white indicator for Focus is on the right side, flip the Focus direction (invert) in the Camera Rig in Studio Manager.

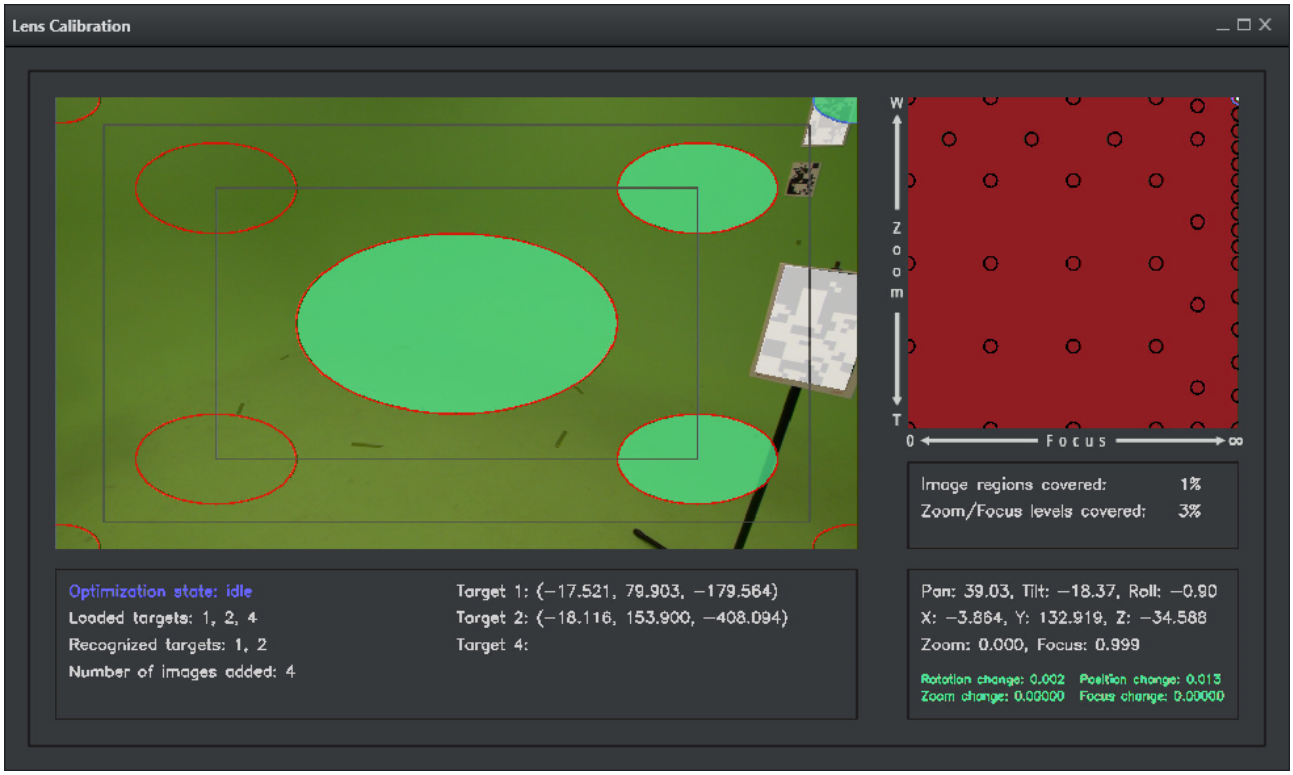
Copyright © 2026 Vizrt

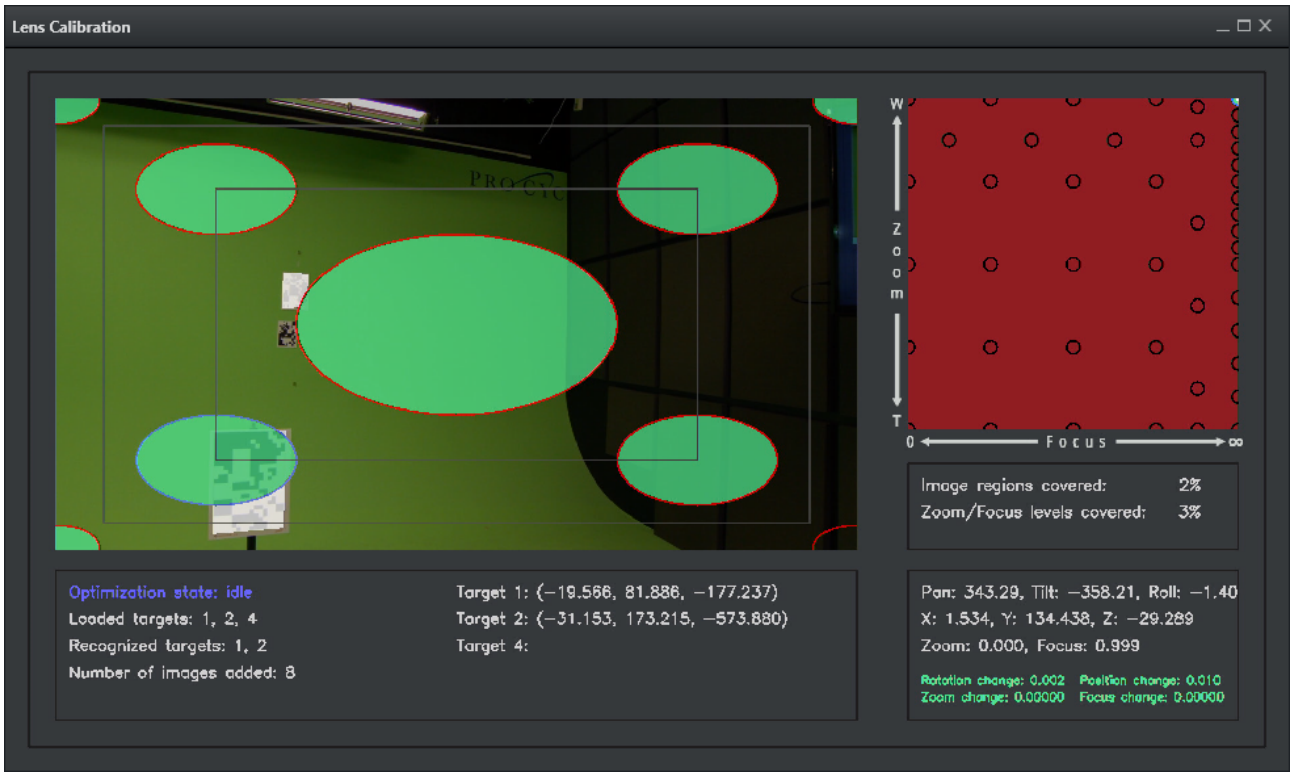
Start filling in circles/hot zones by panning/tilting the camera, so that the detected targets are within these areas. Try to vary the targets a bit between the different hot zones. As soon as a target is inside the hotzone, hold the camera still until the area is green.



Repeat these steps until all nine hot zones are green, if possible. There might be unreachable areas, depending on Zoom and Focus level. In this case, leave the hotzone empty, a small amount of missing areas per lens should not affect the whole calibration result.

Copyright © 2026 Vizrt



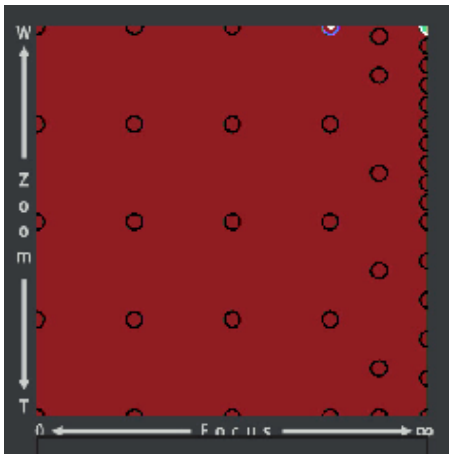


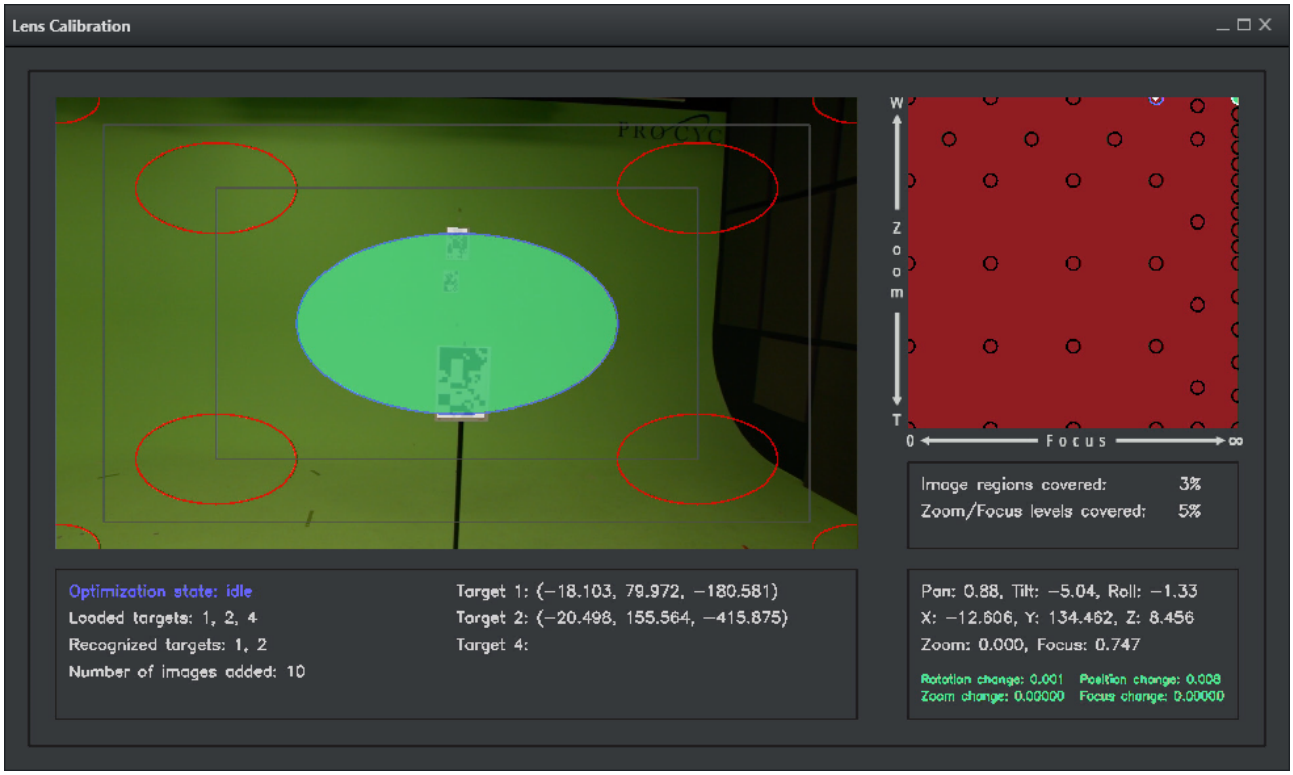
Copyright © 2026 Vizrt

Once all of the hot zones are covered, the current zoom/focus area is covered. If for some reason it is not possible to cover some of the hot zones, the process still works. However, for optimal results, the process should cover all of the hot zones for as many zoom/focus areas as possible.

For optimal results, change the camera position at least when moving from one zoom/focus area to the next one.

> **Note:** When calibrating a PTZ camera or a camera without tracking data for position, the camera cannot be moved during the calibration process. Position offsets need to be set up in the camera rig before starting the calibration.



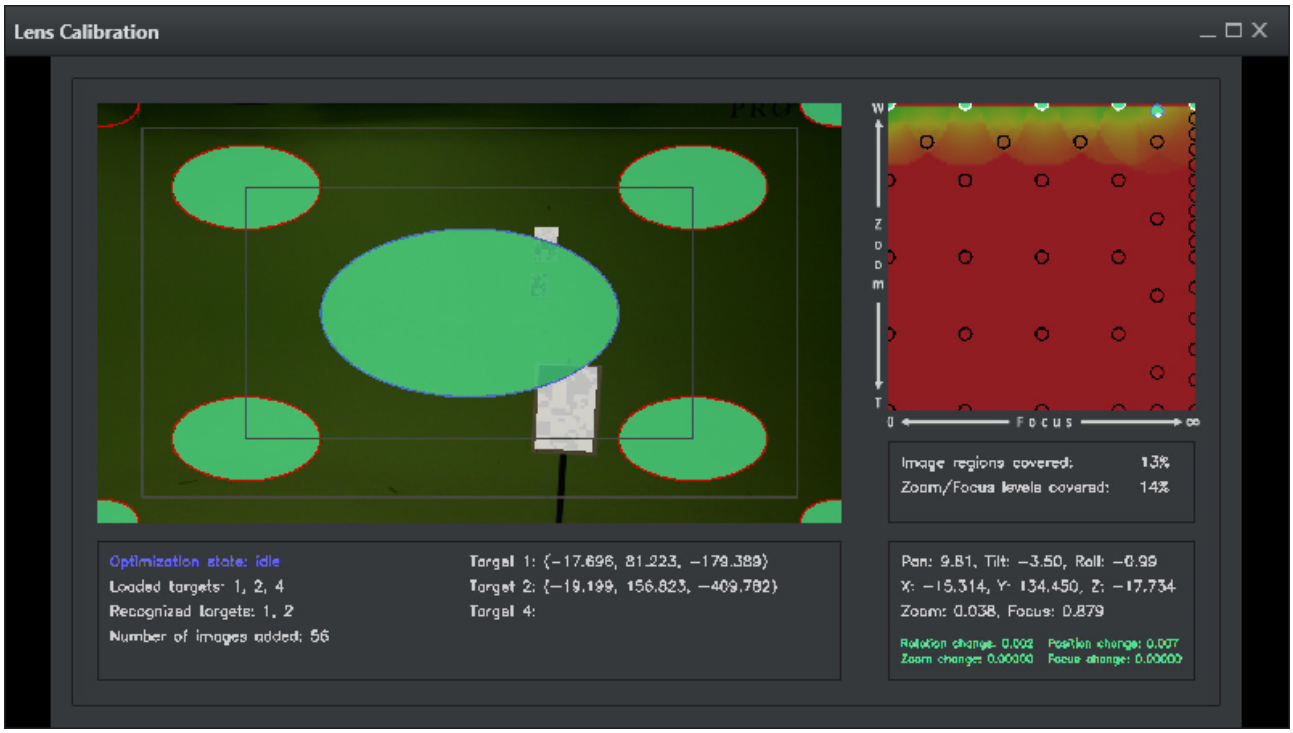


Copyright © 2026 Vizrt

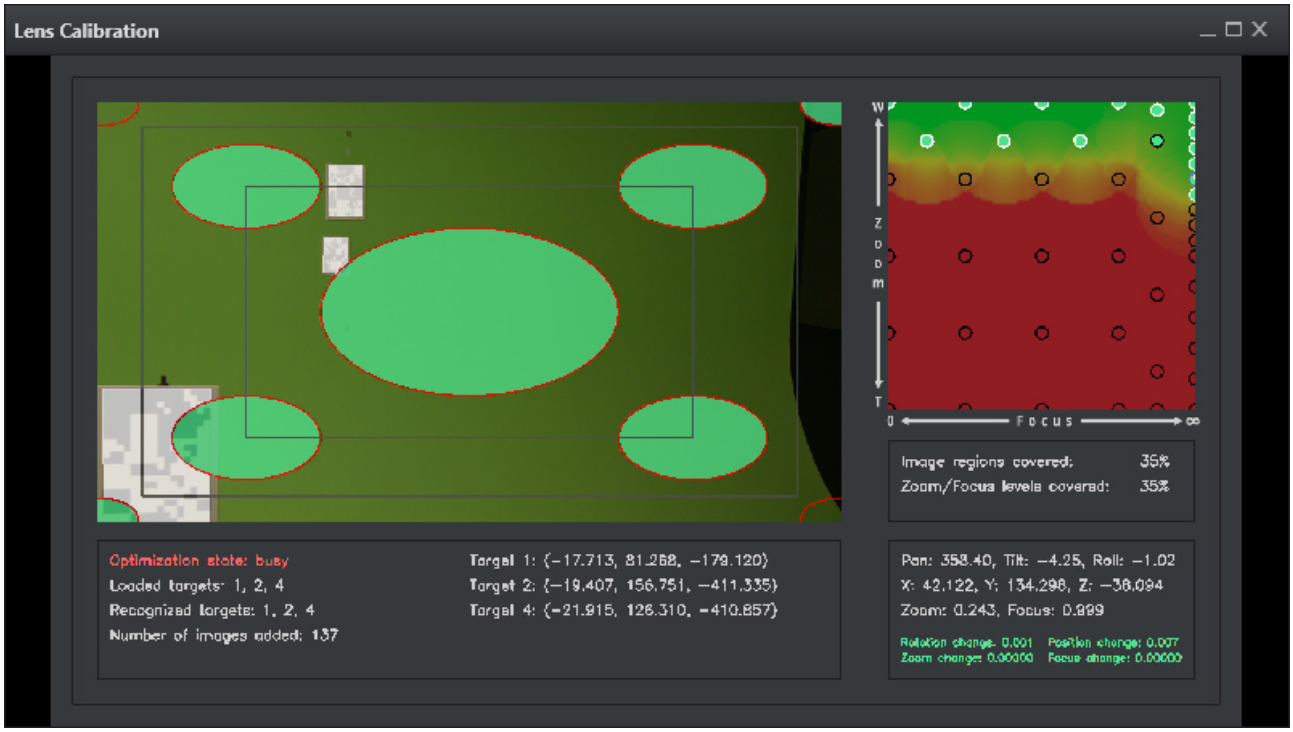
Repeat the steps to cover the hot zones for each zoom/focus area (or as many as possible).

The UI starts drawing green areas indicating a good coverage of this lens area.

> **Note:** This starts after a few points are made first.



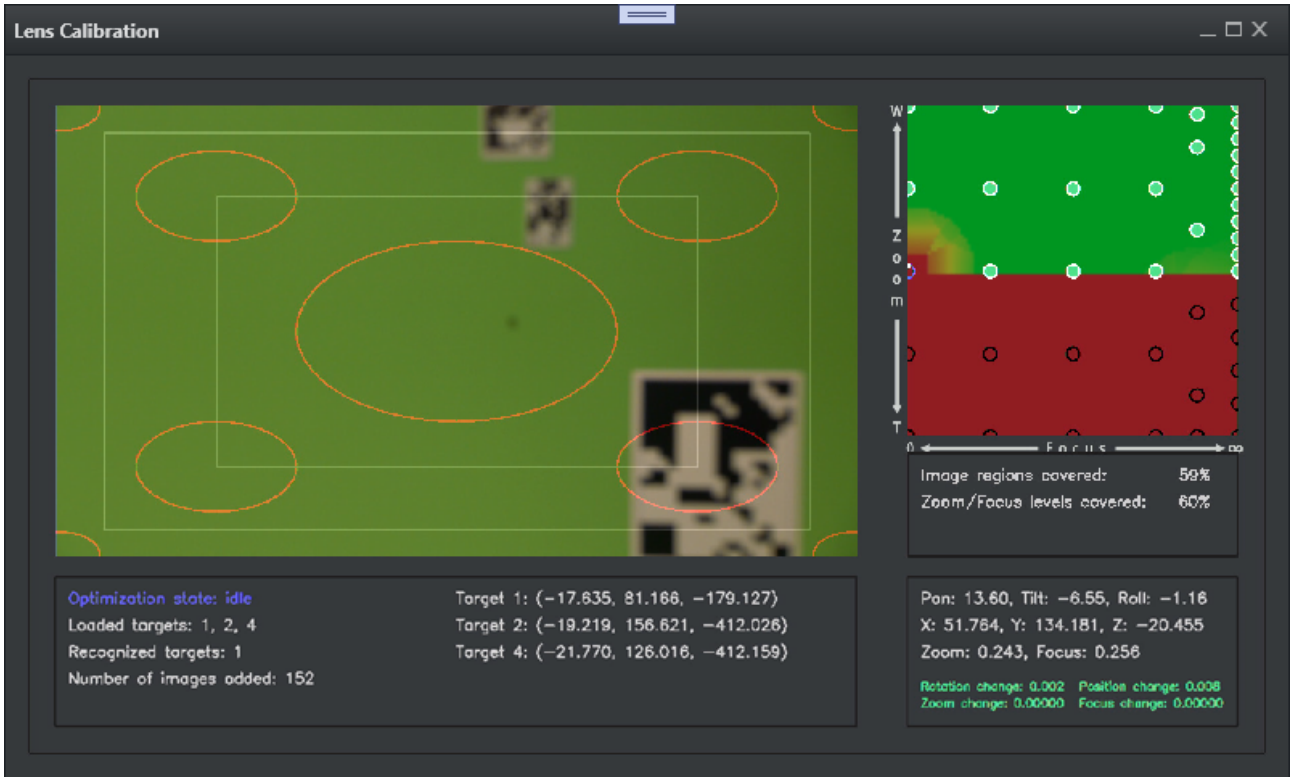
Copyright © 2026 Vizrt

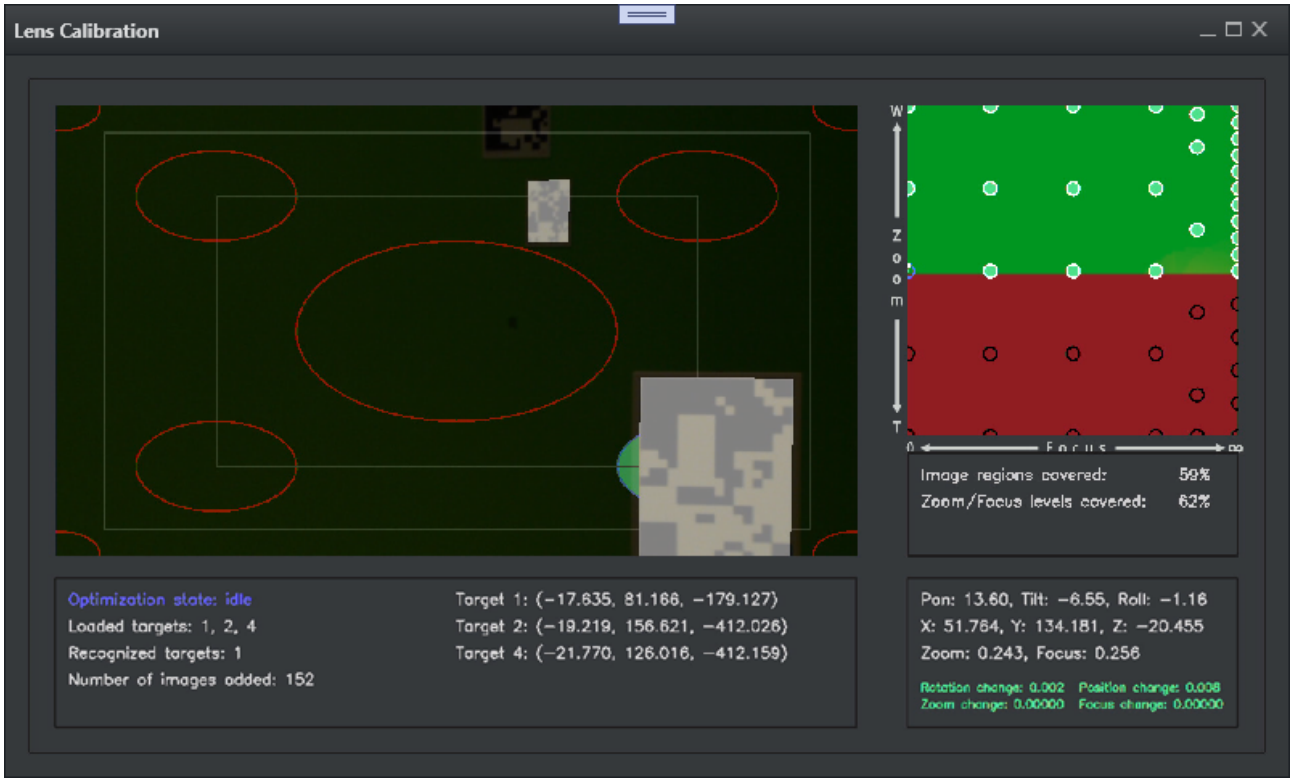


Blurry Targets

Change the iris and try to use lights on the targets to deal with blurry out of focus images (target detection stops working once the images are very blurry).

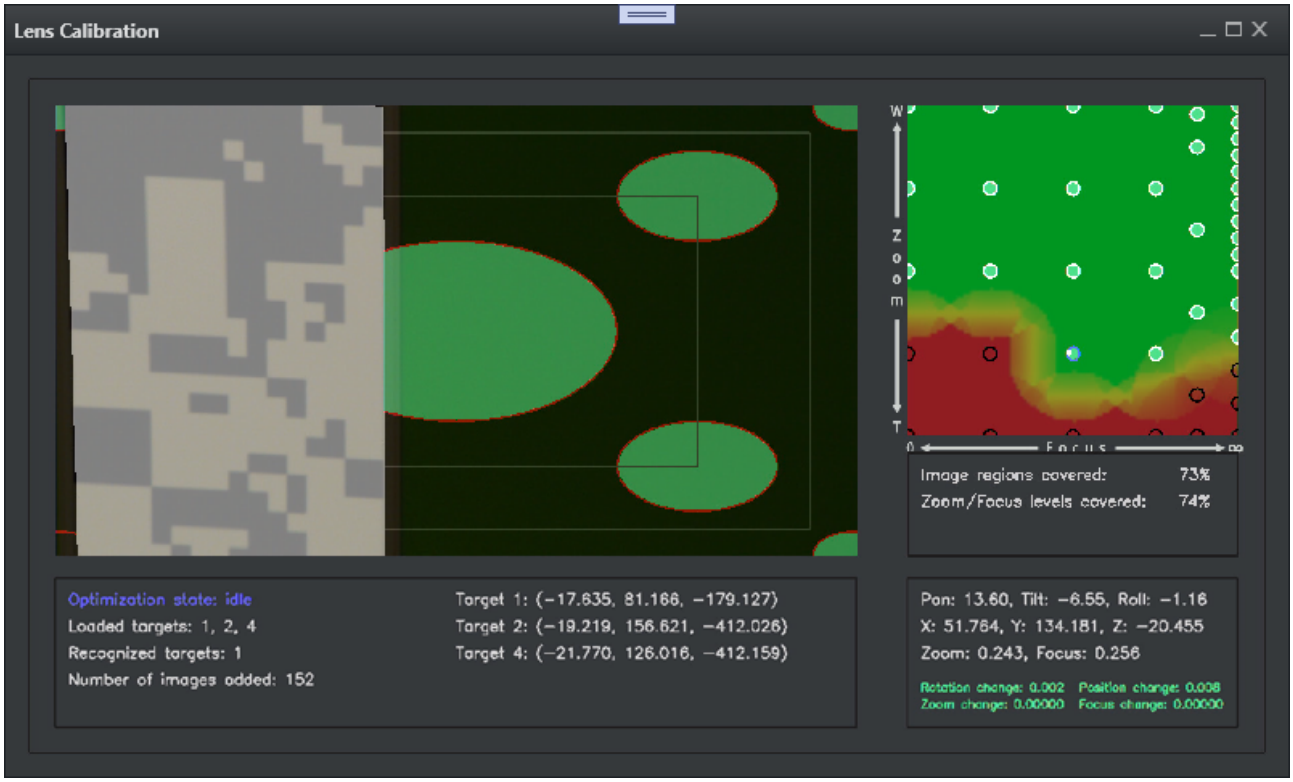
Copyright © 2026 Vizrt





Try to use all targets as long as possible, and try to cover as many zoom/focus areas as possible.

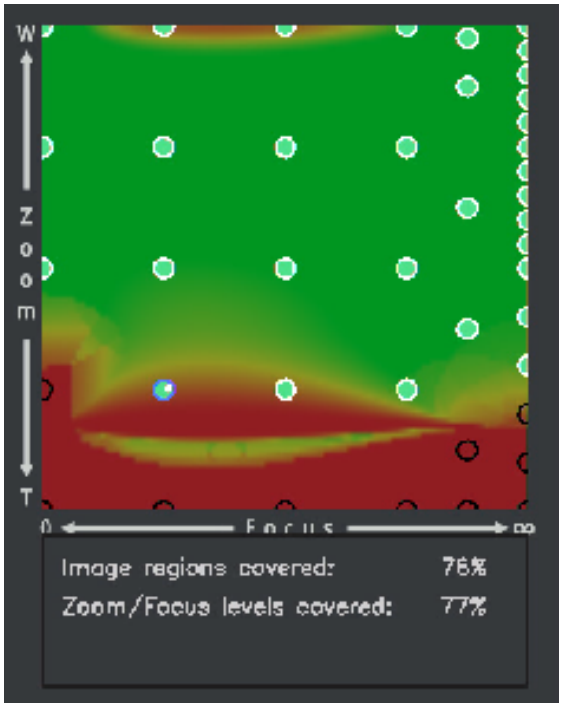
Copyright © 2026 Vizrt

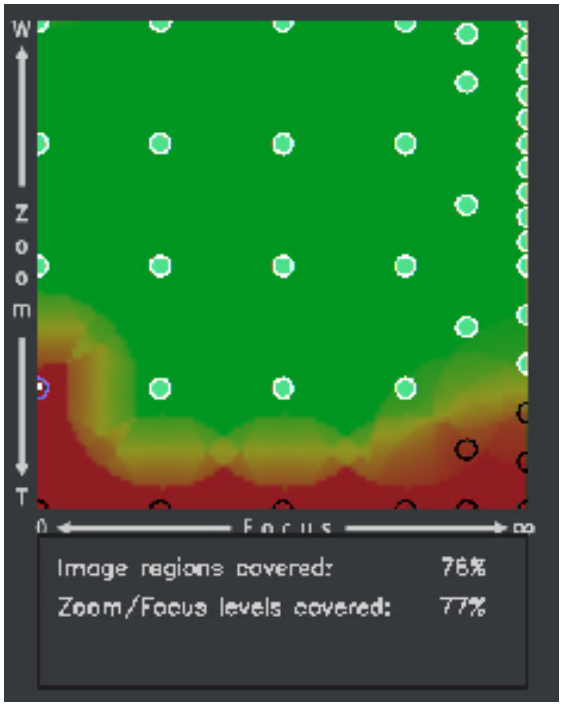


The calibration heatmap serves as a quality hint, and can be a bit off during calibration and optimization. Additional zoom/focus calibration areas may be added to improve the quality, see Adding Points.

The more points added, the longer calculation needs. The optimization state in the bottom left coverage panel indicates whether the lens calibration process is currently busy updating the lens calibration, or whether it is idle. If it is idle, this means that all of the images captured are processed, and the lens preview should be up to date.

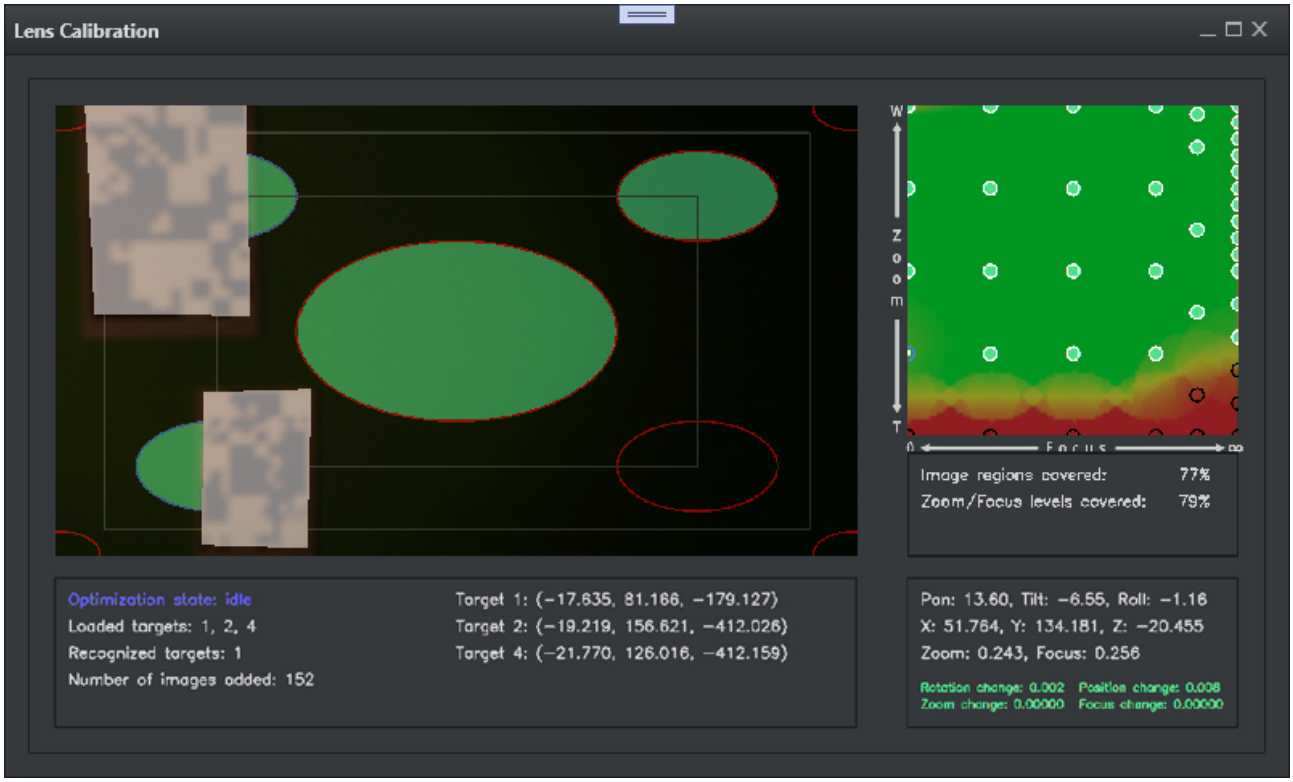
Copyright © 2026 Vizrt





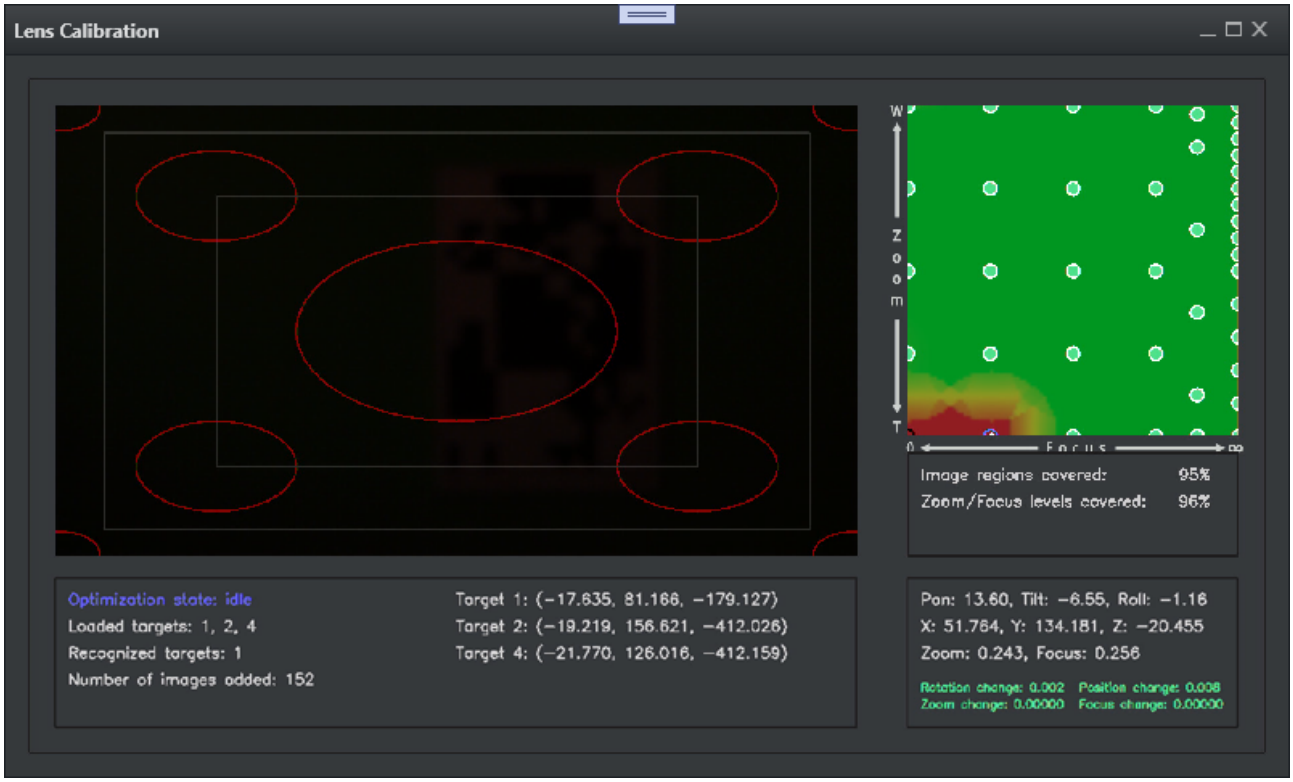
The example below shows how light can help to still be able to detect target in out of focus images (blurry areas).

Copyright © 2026 Vizrt



When the camera is zoomed in and the focus is close to zero (the bottom left of the zoom/focus coverage), it may become impossible for the calibration process to still reliably detect targets. This is expected, but the user should try to cover as many of those zoom/focus areas as possible.

Copyright © 2026 Vizrt

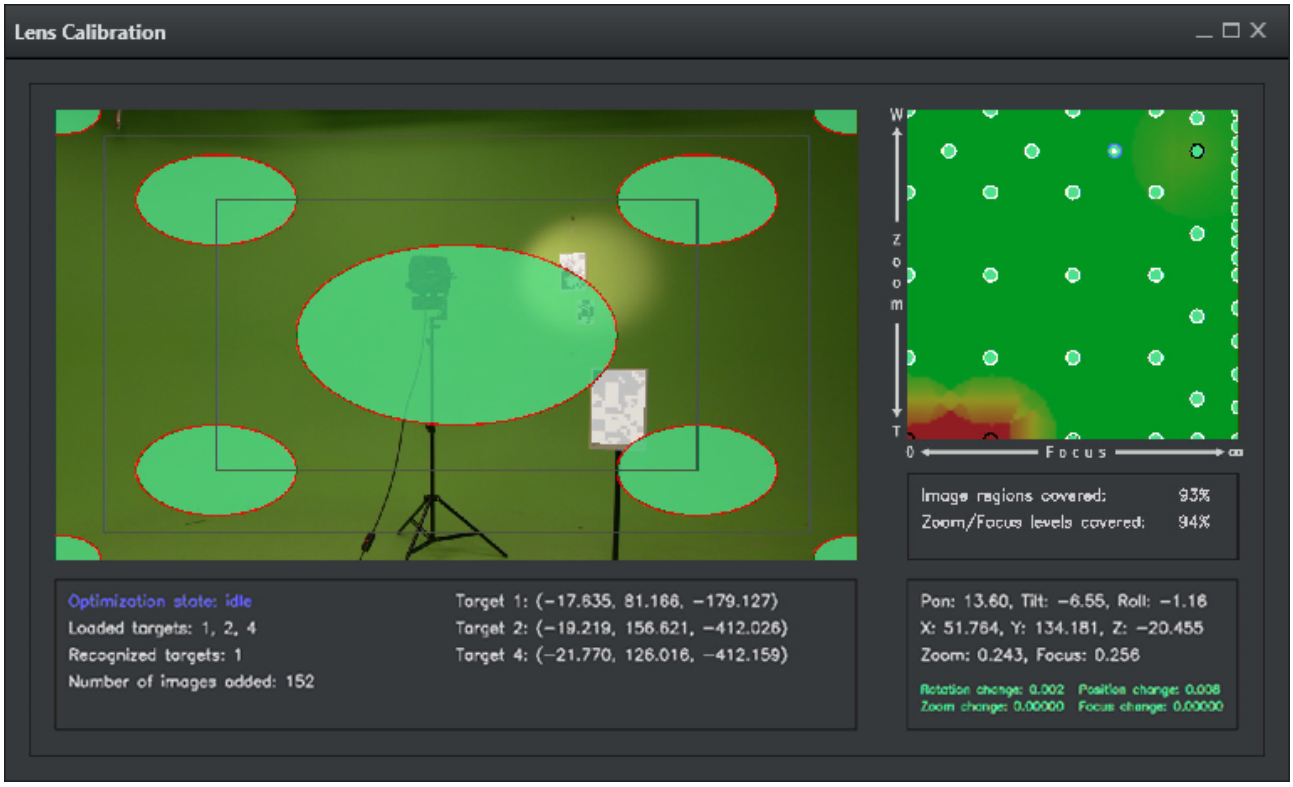


Adding Points

Once all of the zoom/focus areas are covered, the user may interact with the camera and check the quality of the lens parameters by moving the camera, changing the rotation, the zoom and the focus of the camera.

The quality of the lens can be verified in the lens preview window, and the test scene. If there are zoom/focus areas where the quality is not as desired, the user can add additional zoom/focus areas to the calibration process to improve the results. Ideally, the user should add a zoom/focus area where the quality of the current lens calibration is the worst.

Copyright © 2026 Vizrt

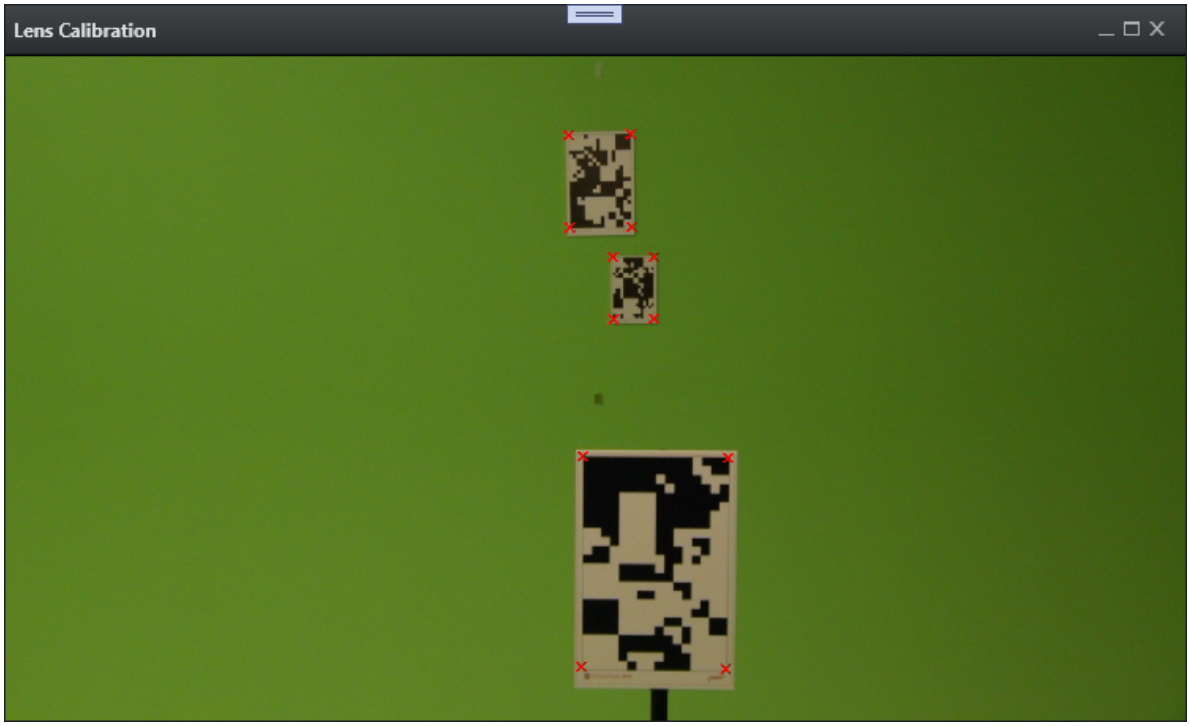


Lens Preview Window

The lens preview shows the current result of the calibration process. It uses the most recent optimized lens and target parameters, and is updated with the incoming video frames in real-time. This visualization can also be used to check the final quality of the calibrated lens. If there are areas of insufficient quality, you may add additional zoom/focus calibration areas (see Adding Points).

Note: To inspect the final quality, you must wait until all optimization processes have finished (Optimization state: idle).

Copyright © 2026 Vizrt

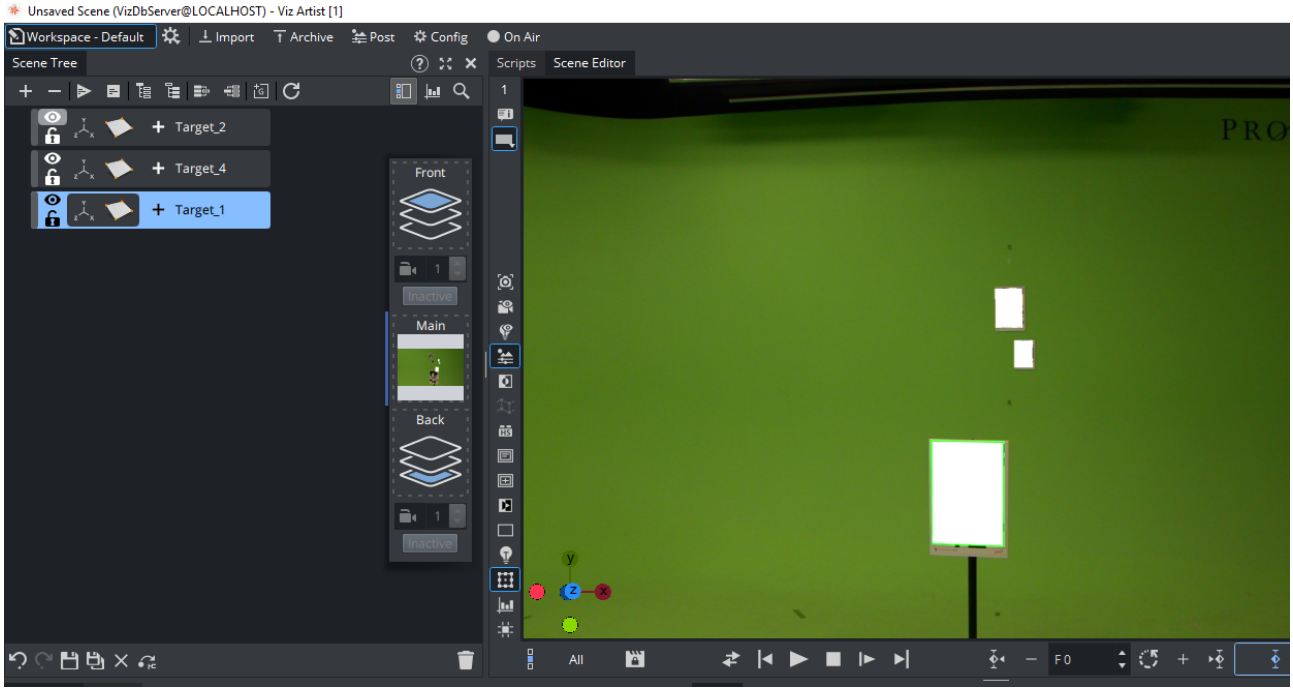


Test Scene

If the calibration was started through the wizard, a test scene is created automatically on start showing the rendered rectangles with the current calibration result.

The accuracy of the corners visualized in the lens preview window, should be the same as the accuracy of the rectangles rendered in the test scene. If the accuracy is not the same, it means there is an inconsistency between the results computed by the lens calibration process and the Viz Engine.

Copyright © 2026 Vizrt

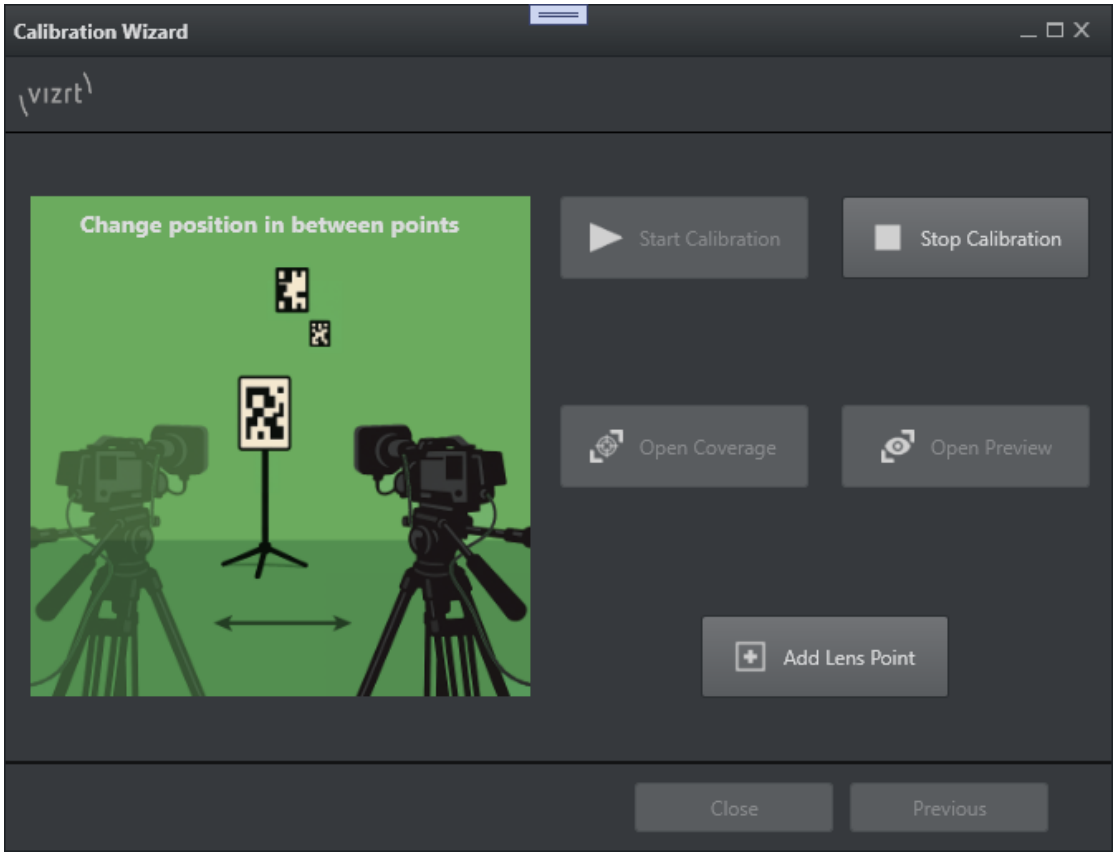


Note: The RectXYZ plug-in is distributed with Viz Engine 5.4.0. For previous Viz Engine versions, the plug-in must be copied manually to the Viz Engine plug-in folder *C:\Program Files\Vizrt\VizEngine\plugin*.
The plug-in can be found on the FTP under */products/VizVirtualStudio/*.

Stop Calibration

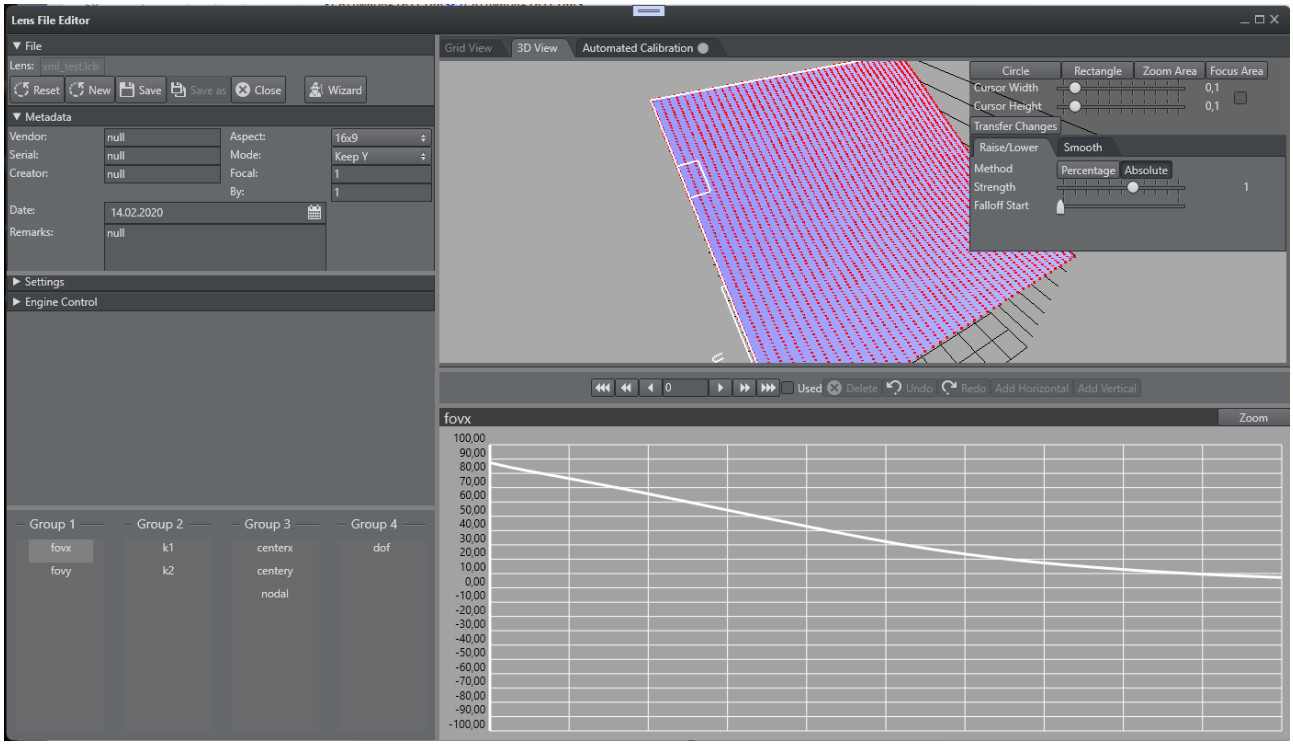
Once all possible points are recorded, finish the calibration by pressing **Stop Calibration**. The Wizard closes and the lens file is automatically loaded to the current lens rig.

Copyright © 2026 Vizrt



The file is displayed in the Lens File Editor.

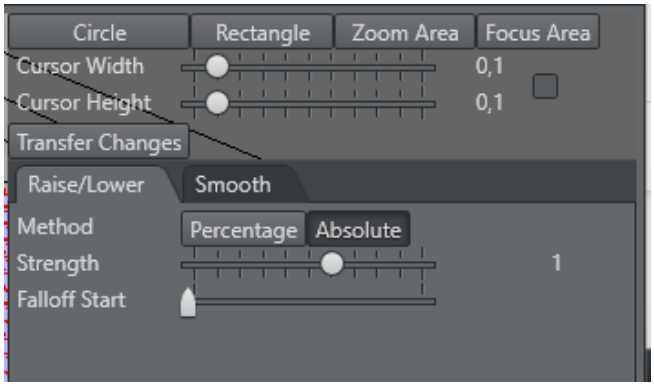
Copyright © 2026 Vizrt



Editing Lens Points

It is possible to adjust the calibration result by lifting, lowering or softening areas of calculated points. As soon as a generated parameter map is selected, the tool window appears in the top right area of the 3D view. Changes are applied by pressing **CTRL** + left mouse button onto the 3D lens mesh.

Tool Window



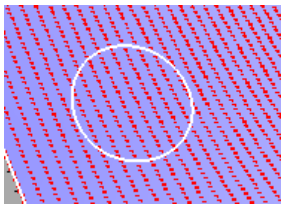
Cursor Selection

In the top area of the tool window the cursor type can be selected. It offers four types of cursors.

Copyright © 2026 Vizrt

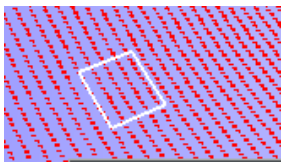
Circle

A circular or oval cursor. The size is defined by the *Cursor Width* and *Cursor Height* sliders. The size of the cursor can also modified by pressing **CTRL** and moving the mouse wheel.



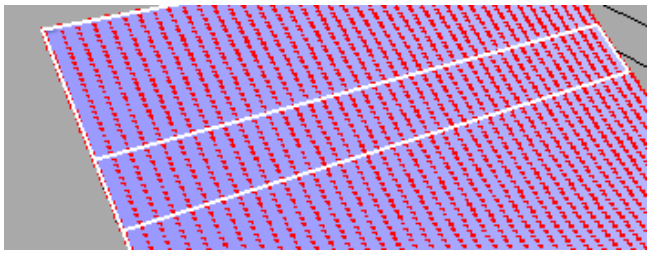
Rectangle

A rectangular or quadratic cursor. The size is defined by the *Cursor Width* and *Cursor Height* sliders. The size of the cursor can also modified by pressing **CTRL** and moving the mouse wheel.



Focus Area

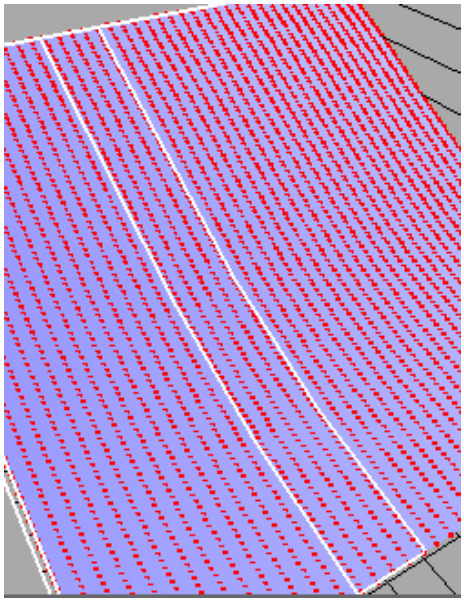
A cursor, which covers the complete focus area. Only *Cursor Height* is taken into account.



Zoom Area

A cursor, which covers the complete zoom area. Only *Cursor Width* is taken into account.

Copyright © 2026 Vizrt

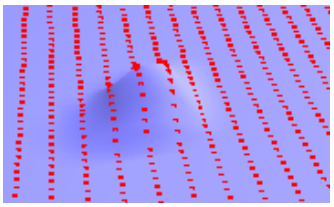


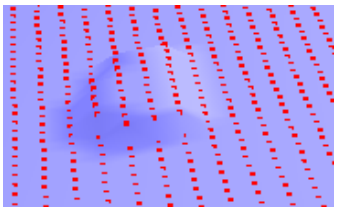
Operations on Points

Raise/Lower

Raises or lowers points in the cursor area. The strength falls off in direction of the cursor border.

- **Strength:** Defines how many of the points are raised or lowered. Points are lowered when negative.
- **Falloff Start:** Defines how far away from the border the falloff starts.
  - Falloff Start=0

    
  - Falloff Start= 1

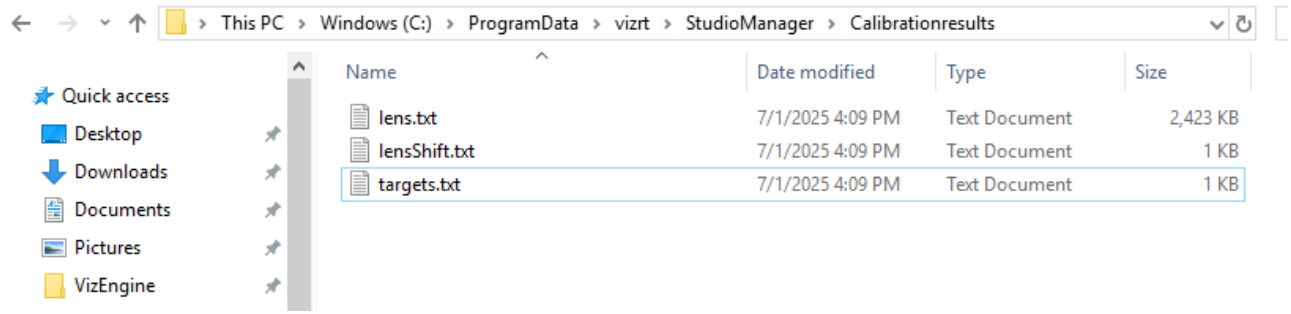
    
- **Method:** Changes the values:
  - **Percentage:** Changes by a percentage of its current value.
  - **Absolute:** Changes by a constant value.

Copyright © 2026 Vizrt

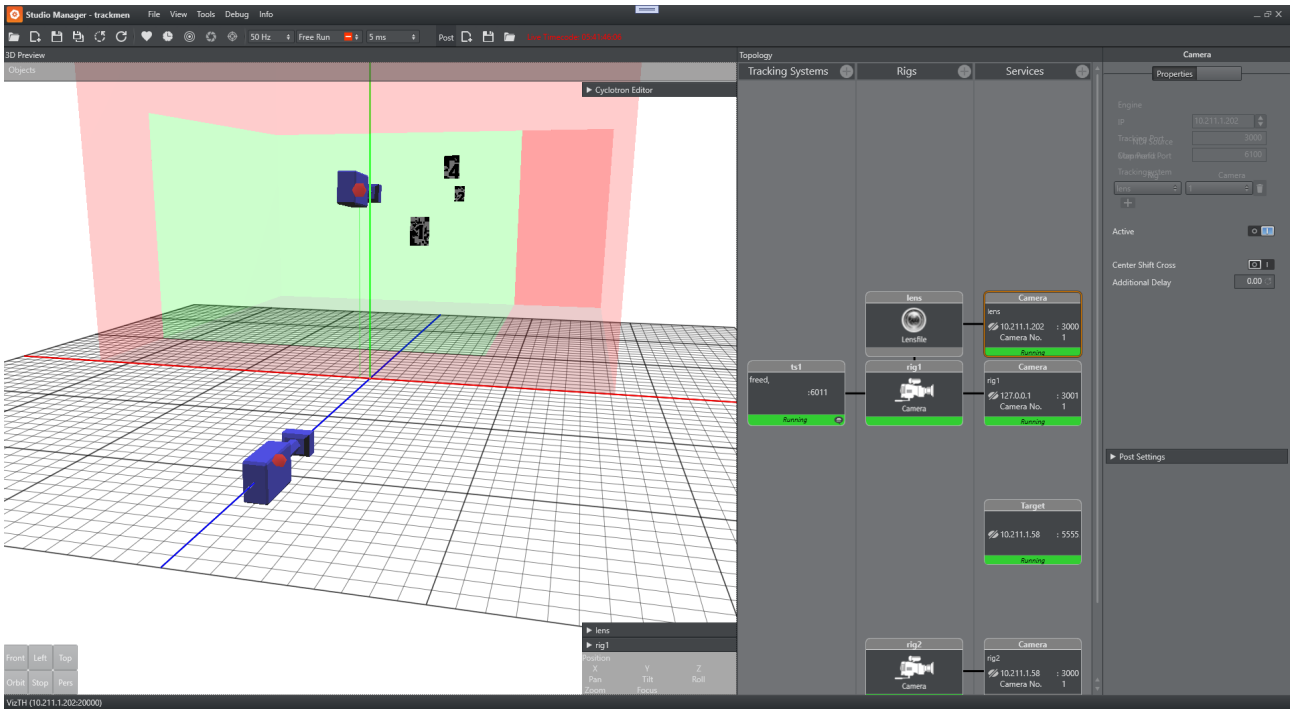
Save Lens

In order to permanently use the lens file, click **Save**.

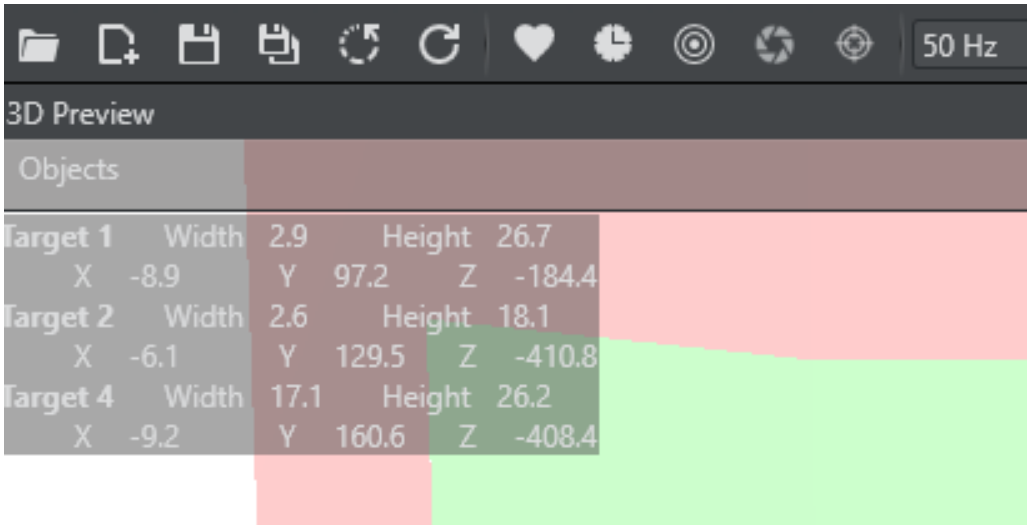
The raw lens files and other calibration results are temporarily stored in *C:\ProgramData\vizrt\StudioManager\Calibrationresults*.



The Studio Manager displays the used targets in the 3D preview panel:

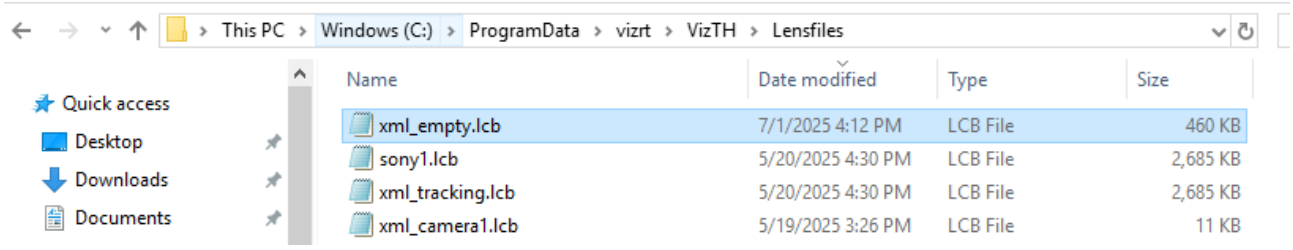


The HUD can be activated to show the detected Targets center coordinates.



Save the Studio Setup to finalize the calibration. Restart Tracking Hub to verify the lens file is loaded properly on restart.

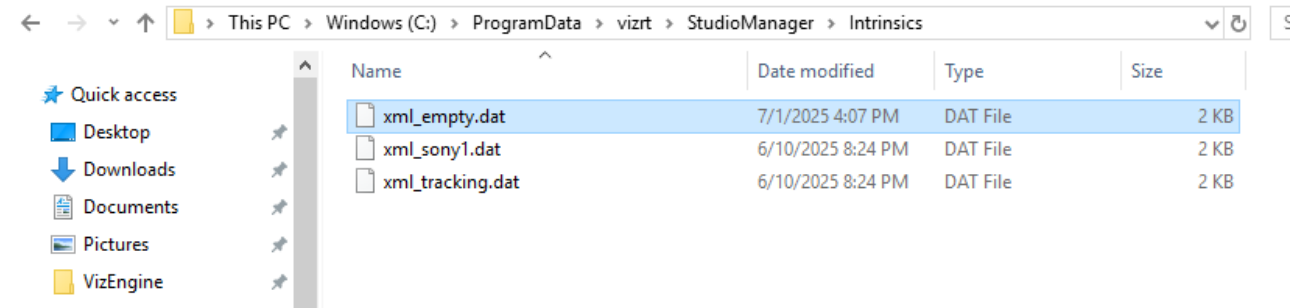
Lens files are automatically saved in the lens files folder (*C:\ProgramData\vizrt\VizTH\Lensfiles*).



Target Detection Metadata

During the automated lens file calibration process, an intrinsics file for each lens is stored in *C:\ProgramData\vizrt\StudioManager\Intrinsics*.

This file is required if the lens is used for target detection later on. Please note that the intrinsics file requires the same file name as the lens file. This needs to be considered in case lens files get copied or altered manually.



Copyright © 2026 Vizrt

8 Troubleshooting

This section lists some common pitfalls and how to solve them, as well as providing guidance on optimizing your Virtual Studio workflow.

- General
- Tracking
- Lenses
- Delay
- NDI
- Chroma Keying
- Remote Connection

8.1 General

- Depending on your tracking system, a zero point is required. This marks a special location in your studio defining the start- or reference point. The design of your set (Viz Scene) and the calibration relays on this location; therefore, clearly mark it.
- Make sure you use correct units during your set design. This makes it much more easier to match your set later on.
- The Studio Manager is mainly used to configure and setup your virtual set. During production this application is not needed and should be closed. Only Tracking Hub needs to be running.

8.1.1 Syncing

- **As a general rule:** Every single component in your setup needs to be synced correctly. This includes any router, camera, Viz Engine, keyer etc.

8.2 Tracking

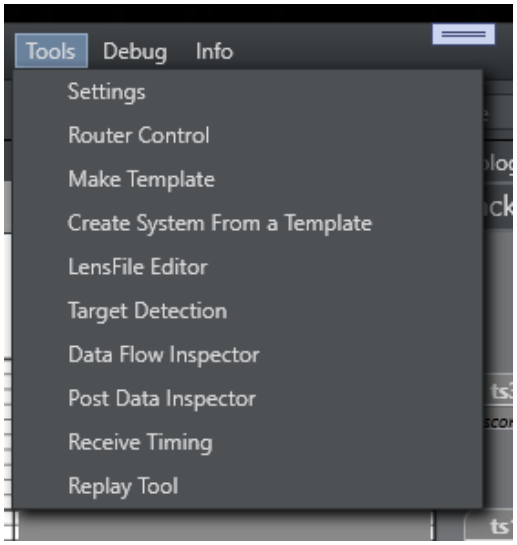
8.2.1 Tracking Networks

- It is recommended to use a dedicated network to transport your tracking data.
- Make sure no firewalls, load balancers or similar can have any negative impact on the transport of your tracking data.

8.2.2 Receive Timing tool

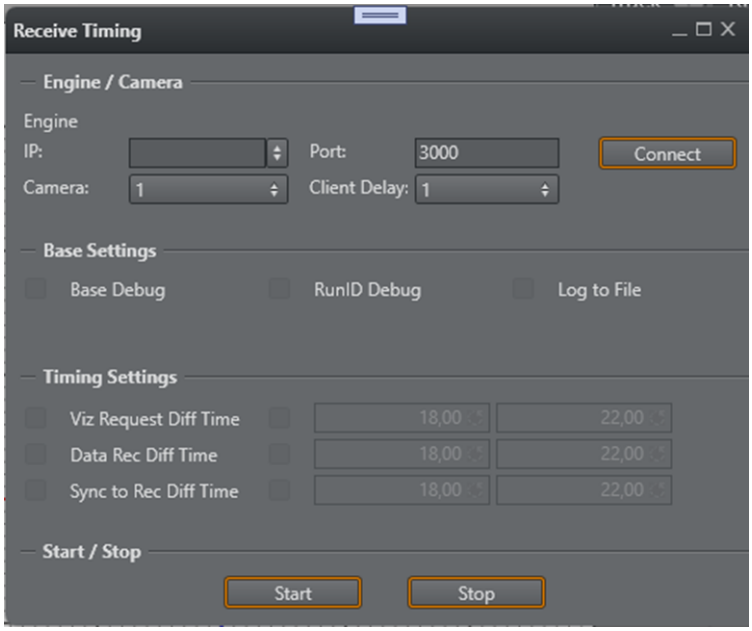
The receive timing tool is used to debug the timing of incoming packages on Viz Engine. It communicates using UDP packets with the *CVIODataExchgVizIo3.dll*, which is responsible for receiving tracking data on the Viz Engine side. It can be used, when there is the suspicion that the network is not able to transport the tracking data to Viz Engine.

Copyright © 2026 Vizrt



The tool can be activated by pressing **CTRL + R** or starting the tool from the **Tools > Receive Timing** menu.

The dialog is built up of four sections.



Engine / Camera

The user can select an engine in the dropdown box and enter the port where the engine receives the tracking data.

The user must select the camera and the client delay which is usually set to `1`. Pressing **Connect** starts the UDP sender.

Copyright © 2026 Vizrt

Base Settings

At least *Base Debug* must be selected to get an output on Viz Engine. As soon as *Base Debug* is enabled, the console outputs:

```
[2026-03-03 14:04:17.145] [I]: Debug mode: 256
```

That means, at least that buffer violations are logged.

```
[2026-03-03 14:11:47.792] [I]: Queue violated: 242939 0
[2026-03-03 14:11:47.793] [I]: 14:11:47.792 Deleting: 242939 0
```

A queue violation happens, for example, when you switch off the camera service on the studio manager. In normal operation, this is a serious problem and creates a jitter in the tracking output of Viz Engine.

```
[2026-03-03 14:28:21.517] [I]: 14:28:21.517 Cutting: 250861 1
[2026-03-03 14:28:30.537] [I]: Queue violated: 251311 0
```

It can also happen that the tracking hub runs too fast (for example, NTSC instead of PAL). In that case, tracking packages must be deleted, which also causes jitters, as packets get lost.

RunID Debug enables timecode debugging. To use this feature, Viz Engine and Tracking Hub machines must be synchronized.

Log to File forces the engine to write the results to a text file. Otherwise, the output is written to the console.

Timing Settings

Three significant values can be measured and displayed.

Viz Request Diff Time

This is the time which passes between two engine requests for tracking data. If this time is irregular, it can be and indicator that Viz Engine is not running real time or a ring buffer is active. The log line looks like this:

```
[2026-03-03 14:35:43.860] [I]: 14:35:43.860 CamVDIF 0: 272978 0.020197
```

The line starts with the actual time, followed by the string “CamVDIF” and the camera number starting with 0. After the camera number follows the running packet id and the difference time in seconds. For PAL, this time should always be around 20ms.

The following screenshot shows a sequence of difference times which mark an error.

```
[2026-03-03 14:43:38.329] [I]: 14:43:38.329 CamVDIF 0: 296699 0.006651
[2026-03-03 14:43:38.340] [I]: 14:43:38.340 CamVDIF 0: 296700 0.011075
```

Copyright © 2026 Vizrt

The renderer was requesting too much tracking packages. This can happen when loading scenes or switching from On Air to edit mode.

Data Rec Diff Time

This is the time which passes between two packages for the defined camera are received on the UDP port.

```
[2026-03-03 14:52:00.994] [I]: 14:52:00.994 CamRCTD 0: 321833 0.020575
```

The line starts with the actual time, followed by the string “CamRCTD” and the camera number starting with 0. After the camera number follows the running packet id and the difference time between two packages in seconds. For PAL, this time should always be around 20ms.

```
[2026-03-03 14:57:47.237] [I]: 14:57:47.237 CamRCTD 0: 339402 0.016395
[2026-03-03 14:57:47.254] [I]: 14:57:47.254 CamRCTD 0: 339403 0.016804
[2026-03-03 14:57:47.272] [I]: 14:57:47.272 CamRCTD 0: 339404 0.017585
[2026-03-03 14:57:47.289] [I]: 14:57:47.289 CamRCTD 0: 339405 0.017439
[2026-03-03 14:57:47.305] [I]: 14:57:47.305 CamRCTD 0: 339406 0.016286
[2026-03-03 14:57:47.322] [I]: 14:57:47.322 CamRCTD 0: 339407 0.017047
[2026-03-03 14:57:47.340] [I]: 14:57:47.340 CamRCTD 0: 339408 0.017245
[2026-03-03 14:57:47.350] [I]: 14:57:47.350 Cutting: 339408 1
```

The screenshot above shows as an example, a Tracking Hub running at 60hz and a Viz Engine running at 50hz. After six packets, the queue runs over and Viz Engine must cut a package.

Unstable difference timing indicates that there might be a problem with the transport of packages via network, or an unsynchronized Tracking Hub.

Sync to Rec Diff Time

This is the time between Viz Engine's request for camera data and the receive time of the next camera package via network.

```
[2026-03-03 15:07:51.558] [I]: 15:07:51.557 CamSYDF 0: 373833 0.004370
```

The line starts with the actual time, followed by the string “CamSYDF” and the camera number starting with 0. After the camera number follows the running packet id and the difference time between a sync and the receive time of the next package.

```
[2026-03-03 15:15:14.917] [I]: 15:15:14.917 CamSYDF 0: 396244 0.003046
[2026-03-03 15:15:14.934] [I]: 15:15:14.934 CamSYDF 0: 396245 0.000539
[2026-03-03 15:15:14.951] [I]: 15:15:14.951 CamSYDF 0: 396246 0.017909
[2026-03-03 15:15:14.968] [I]: 15:15:14.968 CamSYDF 0: 396247 0.014083
[2026-03-03 15:15:14.985] [I]: 15:15:14.985 CamSYDF 0: 396248 0.011004
[2026-03-03 15:15:15.002] [I]: 15:15:15.002 CamSYDF 0: 396249 0.008271
```

Copyright © 2026 Vizrt

The sequence above shows the difference time when Tracking Hub is running NTSC and Viz Engine is running PAL. This is an extreme example. Usually the drift is much slower and moves over a long period of time.

Usually the log is activated, switched off and switched on after one hour. The difference time should always be stable.

If this time is drifting, it is an indicator for an incorrectly synchronized Viz Engine or Tracking Hub. If this time is too close to 0 or the field border it can be adjusted with the send time in Tracking Hub.

Limits

All debug times can be limited to a specific range so that only spikes will be printed out. This makes it possible to debug and watch the behavior over a long period of time.

Start/Stop section

Press the **Start** button to send the selected options to Viz Engine and start debugging. Pressing **Stop** ends the debugging.

8.3 Lenses

Every Lens file is unique. It is highly recommended to calibrate each lens on the mounted final camera rig. Even if you have an existing lens file available, it might not be sufficient to use this one for another lens of the same type.

8.4 Delay

- To get a precise match between real live video and the Viz Engine rendering, the tracking data has to be delayed properly to react within the same field. This can be done for each parameter individually or, in most cases, is handled with the *Overall Delay* on the camera rig node.
- Adjusting the delay can be easily achieved by making quick changes in camera movement to see if the tracked graphic starts before the video signal or if its stops after it. Best practice here is to start with a high delay and decrease it slowly until both layers match perfectly while panning the camera left and right. Once this fits, check the other parameters. In some cases, zoom and focus need a different delay which is added or subtracted from the overall delay by adjusting the specific delay of these parameters separately.

8.5 NDI

8.5.1 Requirements and Limitations

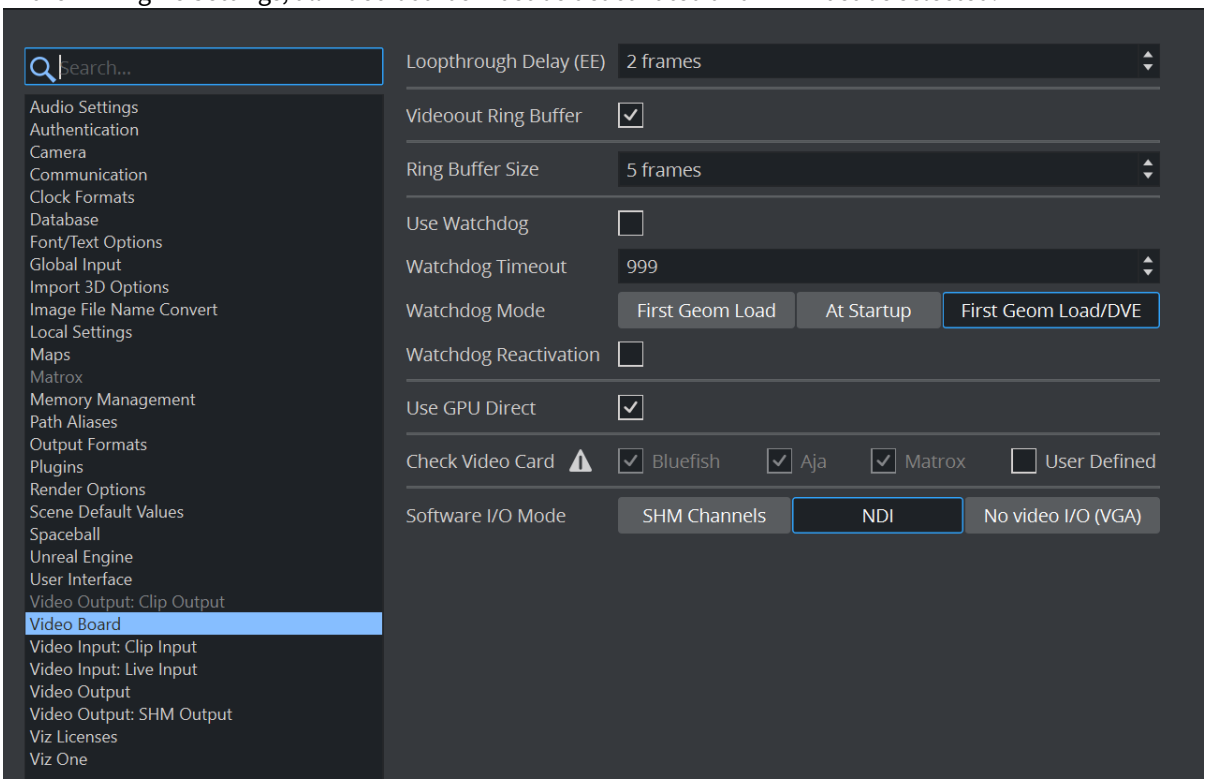
- It is not recommended to have more than one active NDI tracker, due to the increased load on the system. Adding multiple NDI trackers can lead to insufficient GPU resources and an unstable tracking result. GPU recommendation for NDI embedded tracking are NVIDIA RTX 4000 cards or newer. NDI cameras and Tracking Hub should run on a dedicated network to prevent package loss.

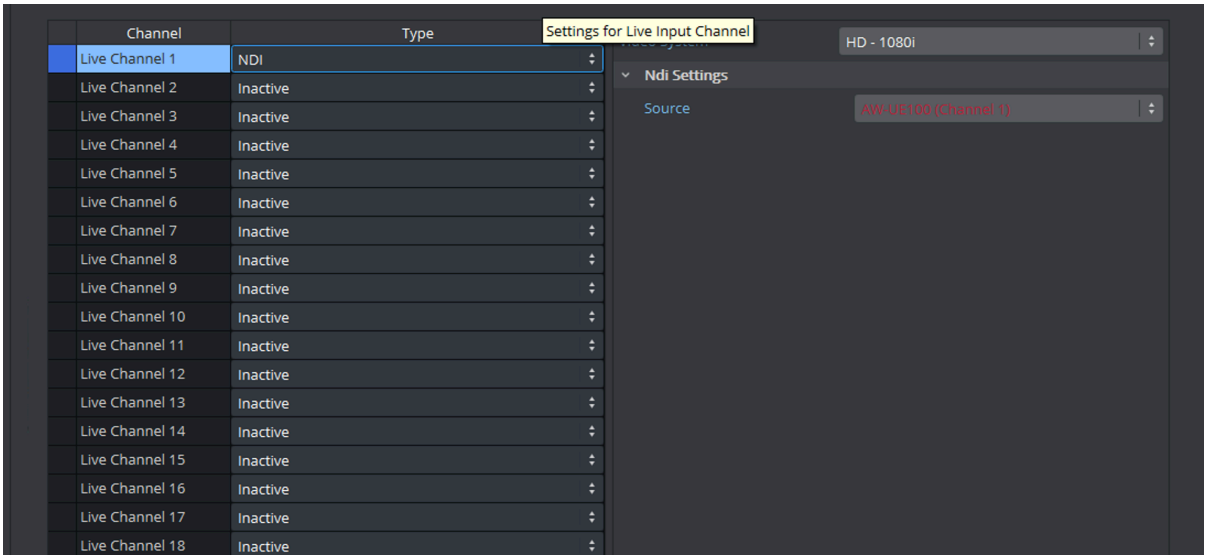
Copyright © 2026 Vizrt

- The network bandwidth needs to fit your needs. It is highly recommended to use a dedicated network for the NDI transport.
- Only tested and certified PTZ cameras are supported. See Supported NDI Cameras.
- Make sure no other applications connect to your NDI source. This also includes NDI Studio Monitor, for example.

To use NDI in your Virtual Studio, make sure the following conditions are fulfilled:

- In the Viz Engine Settings, all video boards must be deactivated and *NDI* must be selected.





Copyright © 2026 Vizrt

- As the Sync source settings are not available after switching to NDI. In the configuration file under the section **Genlock**, Genlock must be set to `2`.

8.6 Chroma Keying

Make sure your lighting conditions are the best ones you can achieve. Optimal chroma keying results depend strongly on the good quality of both cyc and homogeneous light.

8.7 Remote Connection

It is not recommended to use Windows Remote Desktop for remote connection to any Studio Manager Client as the Windows Basic Adaper can interfere with the GPU settings leading to Studio Manager no being able to start properly.

Copyright © 2026 Vizrt

9 Appendices

This section contains the following topics:

- Words and Terminology
- Supported Protocols
- Supported NDI Cameras
- Description of the FreeD Protocol
- Motion Analysis Integration
- TH and SM Ports

Copyright © 2026 Vizrt

9.1 Words and Terminology

Some commonly used words and terminology are described in this section:

Term	Definition
Virtual Studio	The real-time combination of people or other real objects and computer generated environments and objects in a seamless manner.
Viz Virtual Studio	A solution to create and manage virtual studios. Consists of the applications Tracking Hub and Studio Manager and is used in combination with Viz Engine and control applications such as Viz Trio, Viz Mosart or Viz Opus.
Tracking Hub	A control process (service) to receive, control and forward sensor input such as camera movements. A console program.
Studio Manager	A GUI program to setup and control Tracking Hub and the Virtual Studio environment.
Rig	Refers to the physical objects such as camera rigs, pedestals and more.
Lattice	In a virtual studio environment, lattice is often used synonymous with RIG.
Cyc	Comes from the word Cyclorama. It was formerly used in theaters to encircle or partially enclose the stage from a background. In a Virtual Studio, the Cyc defines the green (or blue) area, where virtual objects can appear behind real objects.
CoCyc	The CoCyc is the mask (geometry), which is used to hide real objects from the camera. The CoCyc is indirectly defined by the Cyc. Also known as the *trash matte*.
HUD	Heads Up Display.
Chroma Keyer	Chroma key compositing, or chroma keying, is a special effects / post-production technique for compositing (layering) two images or video streams together based on color hues (chroma range). The Viz Engine can generate key and fill and can, if required, also be used with an external Chroma Keyer.
Synchronization	Using a reference signal to synchronize a video signal. Typical Blackburst, Tri-Level or using the Viz Engine digital input signal as sync.

Copyright © 2026 Vizrt

9.2 Supported Protocols

Tracking Hub has native support for the following protocols. The name in brackets is the protocol identifier as it appears in the Studio Manager tracking system dropdown.

- FreeD (`freed`) — see Description of the FreeD Protocol
- FreeD A0 (`freeda0`)
- RTHead (`rthead`)
- XML Tracking (`xmltracking`) — a self-definable protocol for special solutions
- Motion Analysis (`motionanalysis`) — it is possible to work with two MA instances
- mo-sys (`mo-sys`) — with lens information
- Trackmen (`trackmen`)
- Trackmen Mocap (`trackmen-objtr`)
- Vicon (`vicon`)
- Libero (`libero`)
- Libero TC (`liberotc`)
- Spidercam (`spidercam`)
- Spidercam Fielddolly (`spidercamfd`)
- Spidercam FrameB (`spidercam-frameb`)
- Flair (`flair`)
- Thoma (`thoma`)
- Stype A5 (`stypea5`)
- Stype HF (`stypehf`)
- Camreginfo (`camreginfo`)
- Technodolly (`technodolly`)
- Skycam (`skycam`)
- HD Skycam (`hd_skycam`)
- GSS Geopoint (`gssgeopoint`)
- Kuper (`kuper`)
- Kronomav (`kronomav`)
- Telemetrics (`telemetrics`)
- Arraiy (`arraiy`) — only with a Vizrt lens file
- Eagle Eye (`eagleeye`)
- GeoPro (`geopro`)
- Cable Cam (`cablecam`)
- Cable Cam 2 (`cablecam2`)
- Sony (`sony`)
- Sony D0 (`sonyd0`)

Copyright © 2026 Vizrt

- Vips / Airpixel (`vipsairp`)

OpenTrackIO is implemented but not yet part of a released build. It will appear in this list once the feature is merged.

9.2.1 Protocols supplied by a plug-in

Besides the native protocols above, Tracking Hub loads protocol drivers as DLL plug-ins written with the plug-in API. NCam and Cincraft reach Tracking Hub this way, so they are not listed as native protocols.

9.2.2 Protocols and certified systems are not the same thing

Tracking Hub parses these protocols and sends the translated data to Viz Engine. A protocol listed here does not mean a tracking system using the same name is certified — see Certified Tracking Systems for the systems that are. Conversely, a certified system does not appear here under its product name: Shotoku and NCam are certified systems that speak one of the protocols above, and XPecto is the protocol implementation behind the `trackmen` entry.

Copyright © 2026 Vizrt

9.3 Supported NDI Cameras

The following cameras have been tested and certified by Vizrt:

- Panasonic AW-UE100KE
- Newtek PTZ3
- Newtek PTZ3 UHD

For augmented (no chroma key) using external FreeD tracking data:

- Panasonic AS-UE150W/K

For augmented or Virtual Studio using embedded FreeD tracking data:

- Newtek PTZ 3 UHD

Copyright © 2026 Vizrt

9.4 Description of the FreeD Protocol

Like most tracking formats the XML description can be used to send an arbitrary number of axes to Tracking Hub. It is possible to define a checksum and the default connection parameters, which is then set in Studio Manager, when the protocol is selected from the User.

Setting the connection parameters is optional. The definition starts with:

```
<?xml version="1.0" encoding="utf-8"?>
```

9.4.1 Start Element “vizth_xml_tracking”

A tracking definition must define this element with the attribute “title”:

```
<vizth_xml_tracking title="name">
```

In Studio Manager, the title attribute is used in the list of tracking systems. It should be unique.

9.4.2 Checksum Element

The checksum element defines the length of the packet and the method of the checksum calculation. The attribute “pkglen” is mandatory and must match the size of the complete package including header and checksum. It is given in bytes.

The attribute “checksum” is optional. If not given no checksum is calculated. If it is bigger than zero the following calculations are done:

- 1: 0x40 - sum of all bytes (same as Freed protocol)

```
Sum = 0x40u - Sum;
return ((Sum & 0xFF) == value of last byte of the package excluding checksum
byte
```

- 2: sum of all bytes

```
Sum= sum of all bytes
return ((Sum & 0xFF) == value of last byte of the package excluding checksum
byte
```

- 3: inverted sum of all bytes

```
Sum = (WORD)(~Sum);
return ((Sum & 0xFF) == value of last byte of the package excluding checksum
byte
```

Copyright © 2026 Vizrt

The checksum byte is always the last byte in the tracking package.

9.4.3 Extraction

The element “extraction” is mandatory and defines with the attribute “count” the amount of axes in the tracking package. Optional you can define separator, if you keep it empty, it is not used. If you define this separator, the tracking package departs first before the values are calculated. It also contains the “axis” elements which define the axis values, the type, the position in the package, the length, the byte order and the calculation method.

9.4.4 Axis Element

The “axis” elements are part of the “extraction” element. The amount of axes elements must match the attribute “count”.

Axis Attributes

- **name:** Defines the name of the axis and this also defines how the axis is displayed in Studio Manager. Every axis name must be unique for the actual extraction element.
- **start:** Defines the byte position in the package.
- **len:** Defines the length of the axis in bytes.
- **order:** Defines the byte order of the numeric value. This attribute can have the following values.
  - “bigendian”
  - “littleendian”
  - “ascii” If ascii is given the bytes are interpreted as string.
- **value:** Defines the numeric encoding of the axis. Valid values are:
  - “INT32”
  - “UINT32”
  - “FLOAT”
- **calc:** Defines the mathematical operation which is needed to calculate the axis value in Tracking Hub units. For angle values, Tracking Hub uses Euler angles. For distance or position centimeters are used. The value of this attribute is a string which starts with a mathematical operator, a white space and a number. Allowed operators are:
  - “+” Addition
  - “-“ Subtraction
  - “/” Division
  - “*” Multiplication
  - Any other character is ignored.
- Tickcount values and Offsets are handled outside this description, and can set on the standard places like for other protocols.

```
<?xml version="1.0" encoding="utf-8"?>
<vizth_xml_tracking title="xml_freed">
    <checksum pkglen="29" calculation="1" />
    <extraction separator="">
        <axis name="rotx" start="5" len="3" order="bigendian" value="INT32" calc="/
"></axis>
```

Copyright © 2026 Vizrt

```
        <axis name="roty" start="2" len="3" order="bigendian" value="INT32" calc="/
"></axis>
        <axis name="rotz" start="8" len="3" order="bigendian" value="INT32" calc="/
"></axis>
        <axis name="posx" start="11" len="3" order="bigendian" value="INT32" calc="/
"></axis>
        <axis name="posy" start="14" len="3" order="bigendian" value="INT32" calc="/
"></axis>
        <axis name="posz" start="17" len="3" order="bigendian" value="INT32" calc="/
"></axis>
        <axis name="zoom"  start="20" len="3" order="bigendian" value="INT32" calc="-
"></axis>
        <axis name="focus" start="23" len="3" order="bigendian" value="INT32" calc="-
"></axis>
        <axis name="iris" calc="!"></axis>
    </extraction>
</vizth_xml_tracking>
```

```
<?xml version="1.0" encoding="utf-8"?>
<vizth_xml_tracking title=”kuper”>
<checksum pkglen="0" calculation="0" />
<extraction  separator="," >
<axis name="tilt" field="1" order="ascii" value="FLOAT" calc="/" />
<axis name="pan"  field="2" order="ascii" value="FLOAT" calc="/" />
<axis name="zoom"  field="3" order="ascii" value="UINT32" calc="/" />
<axis name="focus" field="4" order="ascii" value="UINT32" calc="/" />
</extraction>
</vizth_xml_tracking>
```

Copyright © 2026 Vizrt

9.5 Motion Analysis Integration

Motion Analysis (MA) is handled like any other tracking system in Viz Virtual Studio. There is no hard limit to the number of objects which can be tracked by Tracking Hub. This section provides some information specific for the MA systems.

9.5.1 The CORTEX System

The setup and calibration of the MA Raptor camera system is done by trained Motion Analysis Corporation Engineers. The software to control the cameras and the tracking is called Cortex. Initially, Cortex was developed for Motion Capturing and not for camera tracking (camera tracking functionality was added later). Therefore, most of the camera settings are not immediately obvious. This is especially true when it comes to camera offset fine adjustment and CCD calibration.

9.5.2 CCD Calibration

Cortex calibrates the relative position of the CCD to the target base using an image based method. The camera observes a target, which is moved in several positions. From this data, the software is able to calculate the position and the field of view of the camera. Like all image based methods, this calculation is not 100% accurate and there is always need to fine-tune the pan, tilt and roll angles. The fine-tuning of position and angles is always done in the Cortex system and never in the offset page of the tracking driver.

9.5.3 Sync and FPS (Frames per Second)

Every part of a Virtual Studio must be in sync. This is true for Cortex and the Raptor cameras as well. You should check that MA are connected to the same sync as Visual Studio or if the signals times are moving independently.

9.5.4 Timing

Depending on the timing family of the sync signal the Cortex system must be set to the following settings:

- **50 Hz format:** Cortex runs at 150 FPS with three frame reduction.
- **59.94 Hz formats:** Cortex runs at 120 FPS with two frame reduction.

These settings guarantee the lowest latency of the cortex system.

9.5.5 Network Connection

The Cortex system communicates through a network with Tracking Hub. This communication is time critical. Every delayed package is worthless for the Virtual Set and results in a jitter.

IMPORTANT! A separate tracking network between the computer running Tracking Hub and the Cortex machine is mandatory.

The use of a managed switch is allowed, when it is possible to define a separate subnet for the tracking connection. If such a switch is not available, the use of a direct connection or a (basic) unmanaged switch for the connection is recommended.

Copyright © 2026 Vizrt

9.5.6 Cortex Precision Limit

All optical tracking systems show a jitter in position and angle. The jitter in the rotation angles is much more visible than position jitter. The Cortex precision limit for angles is below 1/100th of degree. Even though this is quite accurate, the jitter is still visible when zooming completely in on a telephoto lens. It is not recommended to use complete near shots with any optical tracking system. One possibility is to use a mechanical tracking head for those shots.

Zoom and Focus Encoder

The Motion Analysis system uses a Zoom and Focus encoder to allow Viz to calculate the actual Field of View of the Lens. The choice of the encoder is important for the perfect result.

The following external encoders are recommended when no internal digital encoders on the lens are available:

- MoSys
- Shotoku
- EncodaCam
- Internal digital lens encoders

9.5.7 Motion Capturing

Please see Topology Panel.

Copyright © 2026 Vizrt

9.6 TH and SM Ports

The following lists the ports used by Tracking Hub and Studio Manager. Please note that this may be different due to the tracking systems in use.

Port Number	Protocol	Used for
Tracking Hub		
`3000` (configurable)	UDP	Default port for sending tracking data to Viz Engine
`4568`	UDP	Backup Communication
`6100`	TCP	Viz Engine Communication
`7111`	UDP	For example, incoming FreeD protocol
`9090` (configurable)	TCP	Lensfile Editor Web Interface
`20000`	TCP	Command Port
Studio Manager		
`7000`	UDP	Viz Engine Communication
`11000` (configurable)	UDP	Parameter Service Port
Other Components		
`22352`	TCP, UDP	Codemeter
Other Ranges		
> `50465`		Used by Tracking Hub
> `50786`		Used by Studio Manager

Copyright © 2026 Vizrt