



Tracking Hub

Version 1.2



Tracking Hub





Copyright © 2018 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.

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

2018/12/12

Contents

1	Introduction.....	7
1.1	Document Structure.....	7
1.2	Customer Feedback and Suggestions.....	7
2	Overview.....	8
2.1	Introduction and Terminology.....	8
2.1.1	Key Technologies and Procedures.....	11
2.1.2	Software Components.....	12
2.1.3	Documentary.....	12
2.2	Tracking Hub.....	12
2.3	Tracking Hub Structural Design.....	13
2.3.1	Protocol.....	13
2.3.2	Parameters.....	13
2.3.3	Rig Objects.....	14
2.3.4	Basic Camera.....	26
2.3.5	Timing.....	30
2.3.6	Delay.....	30
2.3.7	Video Delay.....	31
2.3.8	Tracking Delay.....	32
2.4	Studio Manager.....	33
2.5	New WIBU based licensing system.....	33
2.5.1	Important Pre-installation Information.....	34
2.5.2	Key Features and Workflow of the New Licensing System.....	34
2.5.3	Notable Limitations and Known Issues.....	34
2.5.4	Basic Setup.....	34
2.5.5	License Information.....	36
2.6	Supported Protocols.....	37
2.7	WIBU License System in Tracking Hub.....	37
2.7.1	One License Consumed.....	38
2.7.2	Two Licenses Consumed.....	40
3	Installation and Startup.....	42
3.1	Important Checklist Before Installation.....	42
3.1.1	System Requirements.....	42
3.1.2	Synchronization.....	42
3.1.3	Minimum Hardware Configuration.....	43

3.1.4	Minimum Hardware Configuration for use with Post and Replay	43
3.1.5	Recommended Hardware and Software Configuration	43
3.1.6	Supported Hardware and Software	43
3.2	Viz Virtual Studio Folders	44
3.2.1	Installation Folders	44
3.2.2	Data Folders	44
3.3	Viz Virtual Studio Installation	44
3.3.1	To Install Tracking Hub and Studio Manager Using the Bundle Installer	45
3.3.2	To Repair or Remove Tracking Hub	47
3.4	Start the Viz Virtual Studio	47
3.4.1	Manual Configuration of the Network Adapter IP Addresses	48
3.4.2	Start the Studio Manager	49
4	Studio Manager GUI	50
4.1	Configuration Panel	50
4.2	Parameter Panel	51
4.3	Topology Panel	52
4.3.1	Tracking Systems	53
4.3.2	Rigs	54
4.3.3	Services	56
4.3.4	Topology Connection Lines	57
4.3.5	Topology Colors	58
4.3.6	Motion Capturing with Motion Analysis	58
4.4	Log Panel	59
4.5	Timing Analysis Window	61
4.6	Preview Panel	62
4.6.1	Preferences Menu	63
4.6.2	Configuration	63
4.6.3	View Menu	64
4.6.4	Navigation	64
4.6.5	HUD	65
4.6.6	Cyclotron Editor	65
4.6.7	CyC Editor Angle and Width	73
4.6.8	Examples	74
4.6.9	Shape Area	75
4.6.10	Cyc	75
4.6.11	CoCyc	75
4.6.12	Objects Menu	75

4.6.13	Selectable Display Modes	76
4.6.14	Camera Handles.....	76
4.7	Post System	78
4.7.1	Create a Post Session.....	79
4.7.2	Load a Post Session	79
4.7.3	Configure a Post Session.....	80
4.7.4	Selecting the Recording and Replay Sources.....	81
4.7.5	Camera Channels.....	82
4.7.6	Post Offset and Delay.....	83
4.7.7	Post Data Storage	83
5	Studio Manager Configuration	84
5.1	Configure the Studio.....	84
5.2	Configure Topology	84
5.2.1	Configure a Tracking System	85
5.2.2	Configure a Rig.....	86
5.2.3	Configure a Service.....	88
5.2.4	Set a Delay.....	89
5.2.5	Set an Offset.....	89
5.2.6	Analyze Time	90
5.2.7	Lens Range Calibration	93
5.2.8	Modify a Rig	95
5.3	Tracked Cameras and Viz Engine	98
5.4	Router Control.....	99
5.4.1	To Add a Router Control Preset.....	99
5.4.2	GPI IO Device Switching	102
5.5	Backup Configuration	102
5.6	Use of Templates	103
5.6.1	Using Existing Templates.....	104
5.6.2	Creating New Templates	105
6	Appendices.....	107
6.1	Words and Terminology.....	107
6.2	Description of the FreeD protocol	108
6.3	Motion Analysis Integration	109
6.3.1	The CORTEX System.....	109
6.3.2	CCD Calibration	109
6.3.3	Sync and FPS (Frames per Second).....	109
6.3.4	Timing.....	109

6.3.5	Network Connection	109
6.3.6	Cortex Precision Limit.....	110
6.3.7	Motion Capturing.....	110

1 Introduction

This User 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.

1.1 Document Structure

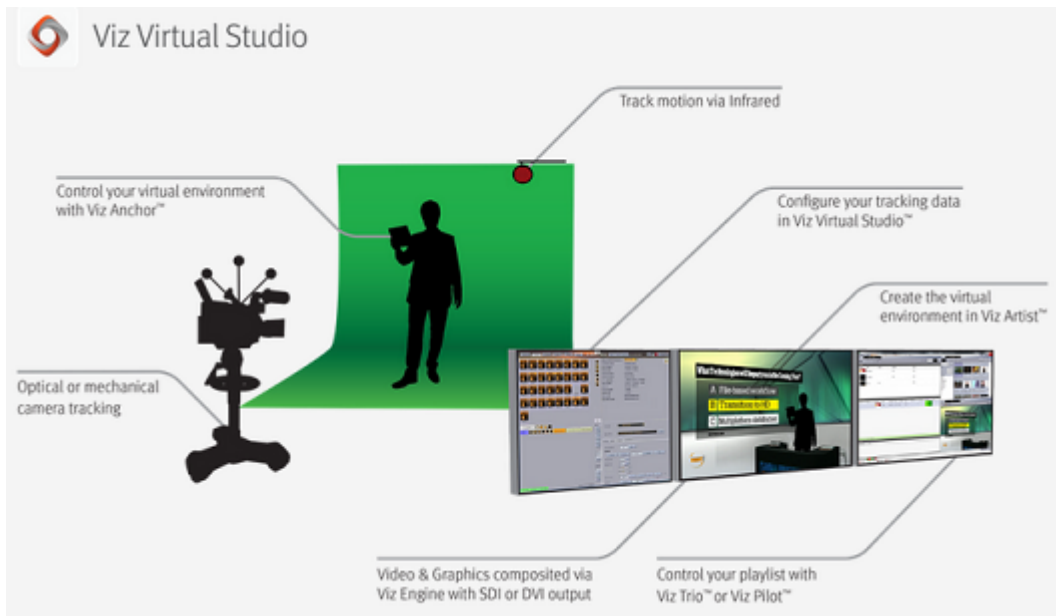
Title	Content
Introduction	This section gives details on related documentation and how to contact Vizrt for Customer support. Details are also given for Customers to provide feedback on their Vizrt product.
Overview	This section gives an overview of both the Tracking Hub and the Studio Manager.
Installation and Startup	This section describes the system requirements, planning for installation and steps to install and start Viz Virtual Studio.
Studio Manager GUI	This section describes the Studio Manager, skills needed to be familiar with the Studio Manager and how it operates.
Studio Manager Configuration	This section describes how to configure Studio Manager and its components.
Appendices	This section lists any additional information about the Viz Virtual Studio.

1.2 Customer Feedback And Suggestions

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

2 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 Cyclotron (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/Engine version 3.7.2 and 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](#)
- [Tracking Hub Structural Design](#)
- [Studio Manager](#)
- [New WIBU based licensing system](#)
- [Supported Protocols](#)
- [WIBU License System in Tracking Hub](#)

2.1 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.





2.1.1 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.

2.1.2 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.

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

2.1.3 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](#)
-

2.2 Tracking Hub

Tracking Hub is a console program for the 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:

- FreeD (see [Description of the FreeD protocol](#))
- RTHead
- XML Tracking: A special feature, a self-definable protocol for special solutions
- Motion Analysis
- mo-sys
- Trackman
- Vicon
- Libero
- CanonLens (Zoom and Focus driver for canon lenses)
- Spidercam
- Flair
- Thoma
- Stype A5
- Stype HF
- Camreginfo

- Technodolly
- Skycam
- Gsgeopoint
- Kuper
- FreeDa0
- spidercamfd

See Also

- [Tracking Hub Structural Design](#)
 - [Studio Manager](#)
-

2.3 Tracking Hub Structural Design

This section covers the following topics:

- [Protocol](#)
- [Parameters](#)
- [Rig Objects](#)
- [Basic Camera](#)
- [Timing](#)
- [Delay](#)
- [Video Delay](#)
- [Tracking Delay](#)

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

2.3.2 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.

2.3.3 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.



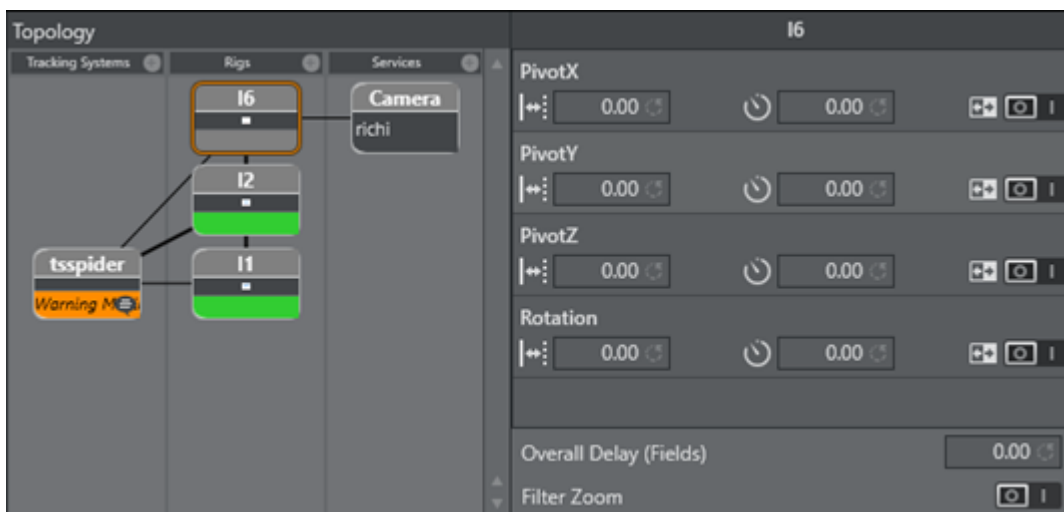
Currently, the following rig elements are covered:

- Pivot Rotation
- Location Pivot Rotation
- Camera Rig
- Position and Rotation
- Zoom and Focus
- Center Shift
- Mounting Offsets
- Extended Lens Parameters Rig
- Object Rig

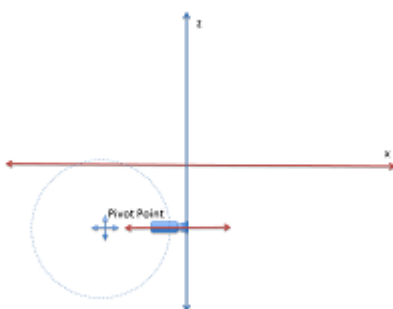
- Arm
- Translation
- CraneArm
- CraneHead
- Lensfile

Pivot Rotation

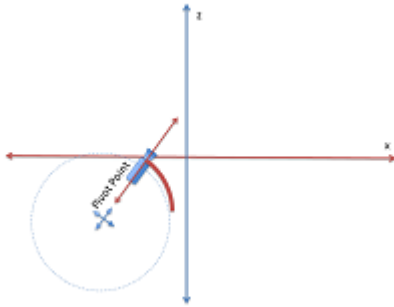
Rotates the coordinate system of a rig around a free definable point in space. The camera Rig is bound to the pivot point in a parent-child relationship.



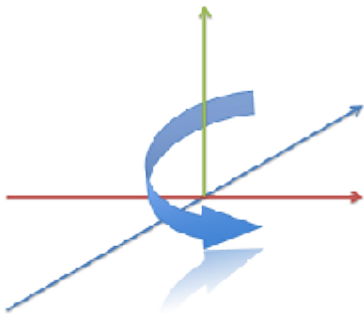
PivotX, PivotY and PivotZ define the location of where the rotation takes place. Rotation specifies the rotation in degrees around the Y Axis of the pivot point.



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.



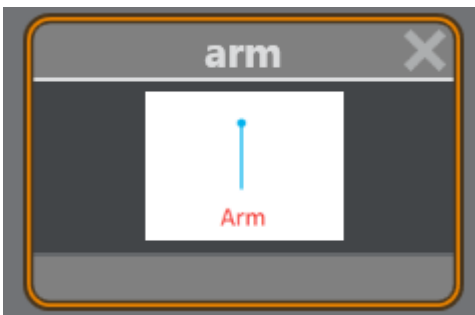
Location Pivot Rotation

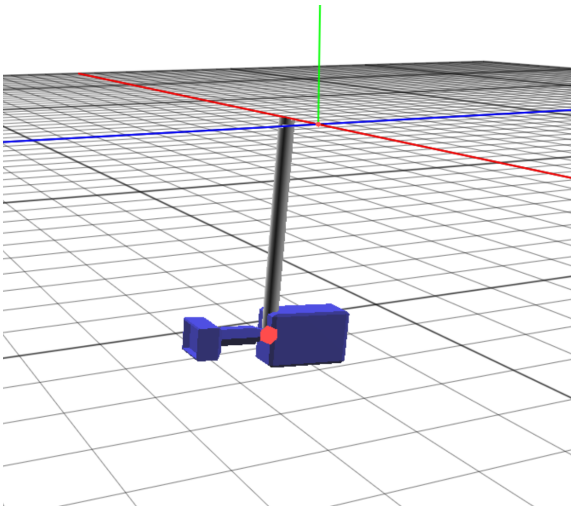


The location pivot rotation rig is similar to the Pivot Rotation. The difference is that this rig takes the pivot point coordinates from the child camera's position offset, which is attached to the camera position. This rig accepts only a rotation angle, as the position is taken from the child position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

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.



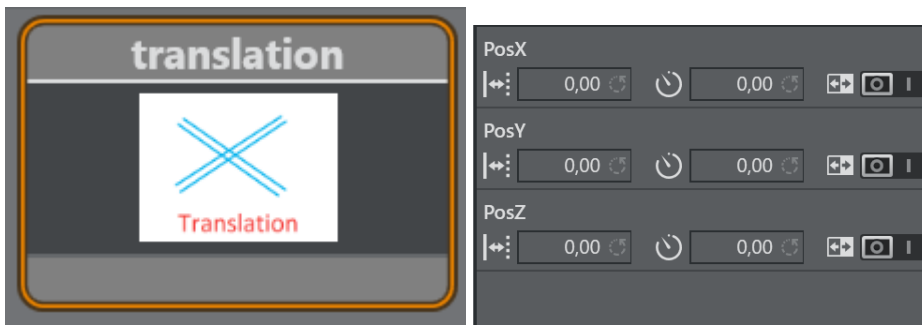


The arm length coordinates define the direction and length of the arm.



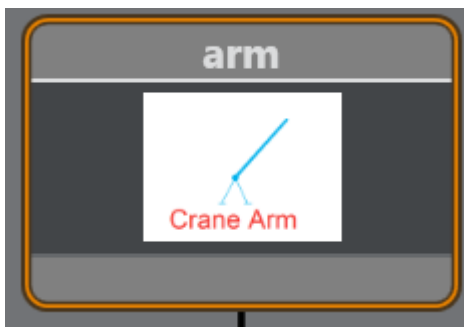
Translation

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.

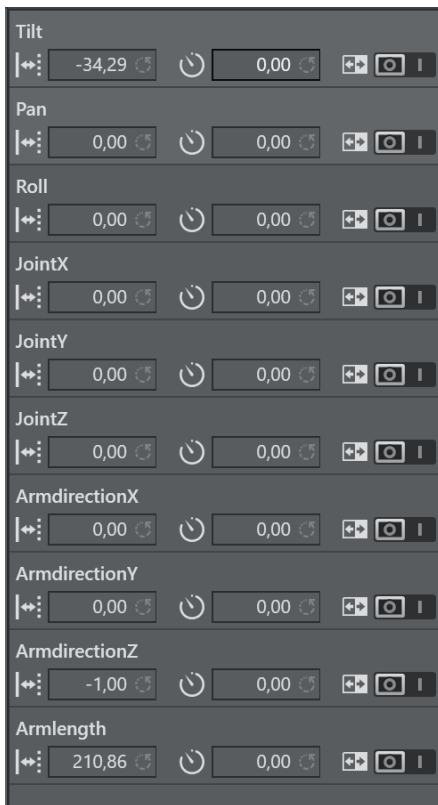


Crane Arm

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.



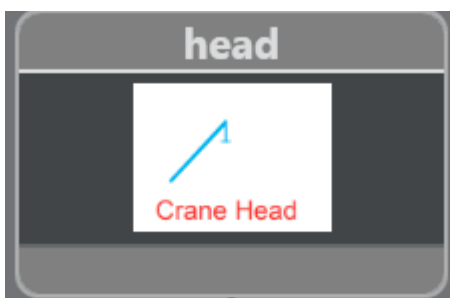
You can set **JointX**, **JointY** and **JointZ** values to adjust mounting offsets between the position giving element and the arm.

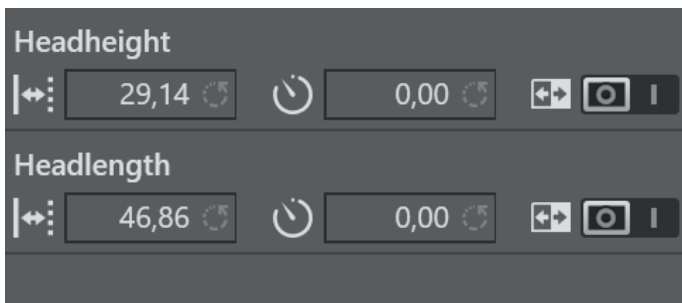
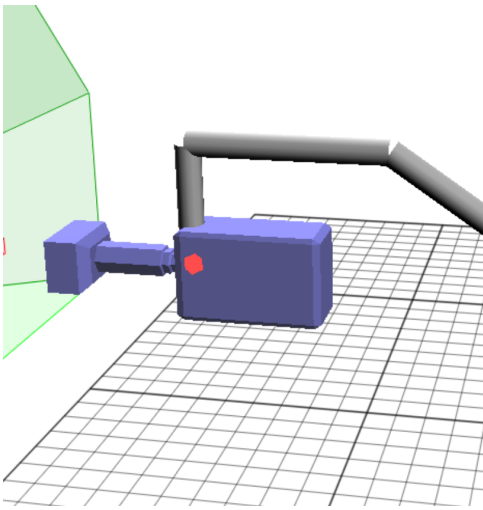


ArmdirectionX, **ArmdirectionY** and **ArmdirectionZ** are normalized to 1 and define the initial direction of the Arm. Usually, **ArmdirectionZ** is set to -1.

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.

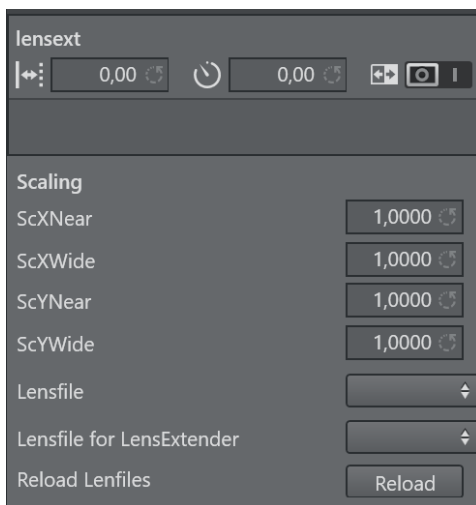




Lens File



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 **lensex** parameter. In some protocols, such as **FreeD**, the lens extender is tracked. The value of **lensex** 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 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.



CameraFD Rig

The CameraFD Rig is similar to the standard camera, but with special developments for the Spidercam fielddolly (FD) and a changed rotation order (X Z Y).

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.

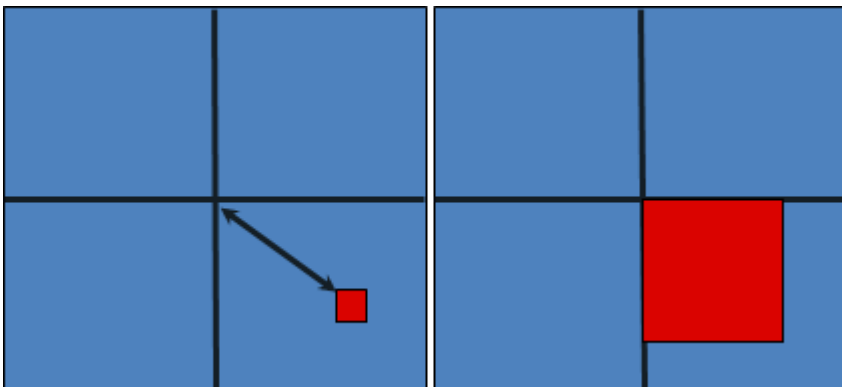
Zoom and Focus

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.

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.



In order 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.

⚠ 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.

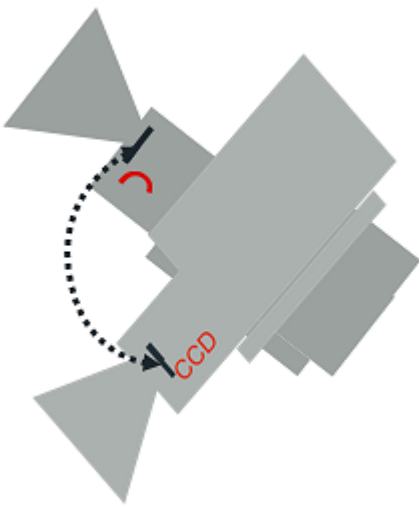
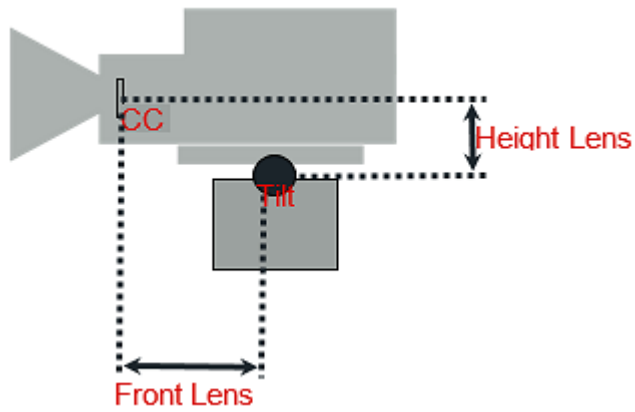
⚠ Note: Repeat the steps until the Center Shift Cross is on the corner or point at all times. After two or three iterations, this is usually the case.

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 sensor 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.



Extended Lens Parameters Rig

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

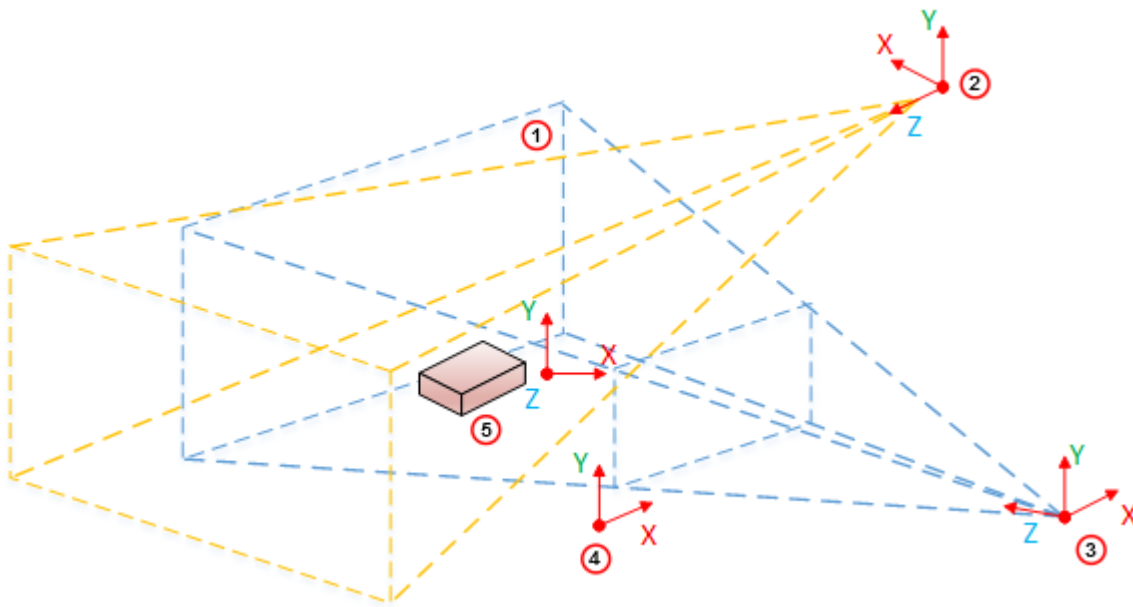
Object Rig



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.

2.3.4 Basic Camera

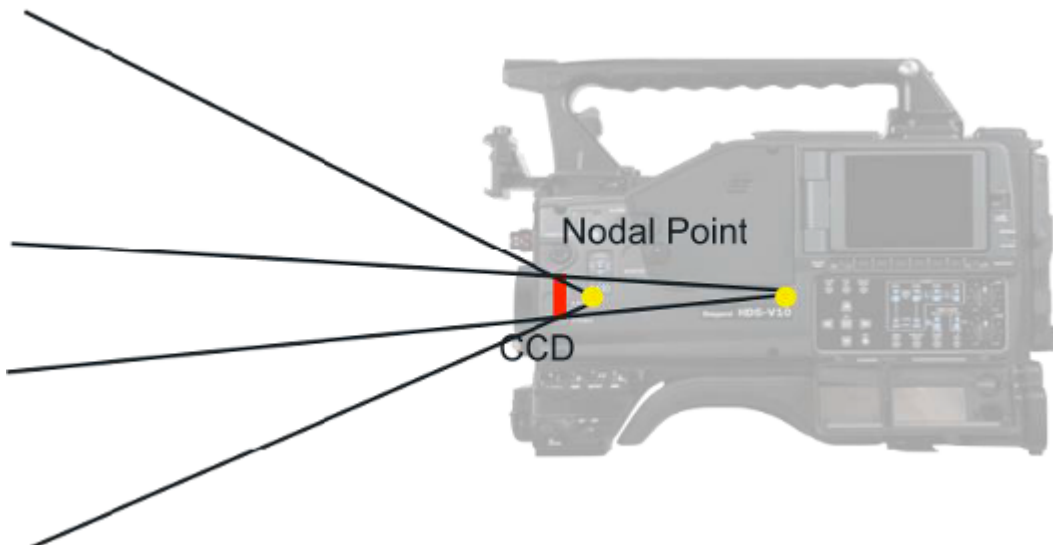
The Basic Camera rig object has to be attached at the end of every rig structure. The most basic and commonly used parameters required for exact tracking are stored in this rig object. This section shows how the Tracking Hub needs to place the idealized OpenGL point camera to create an approximation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



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.

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 converge 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.

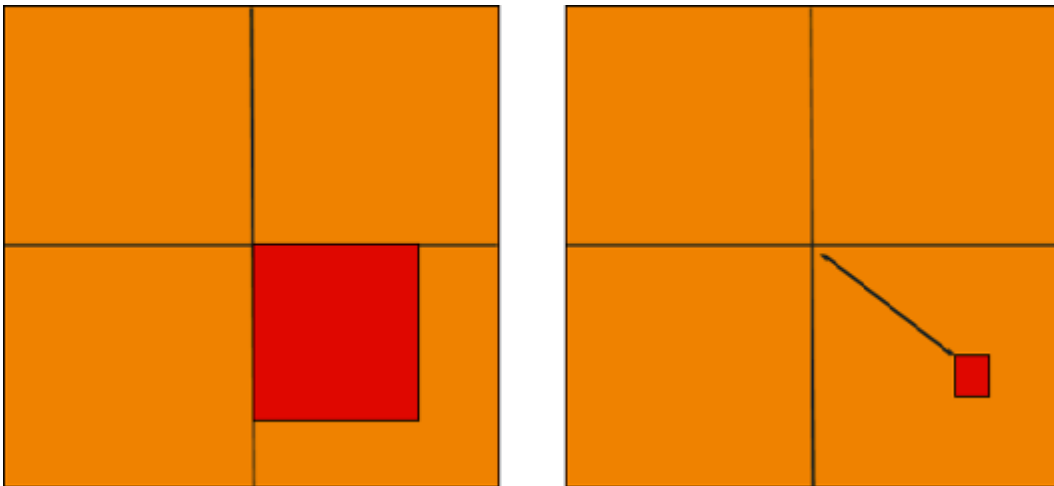
Zooming in and out, causes the light rays, going through the lens, to converge on a moving point behind the lens. In our Viz Virtual Studio software, we call this point the *Nodal Point*, and this point marks the position where we have to place our idealized OpenGL camera. The Nodal Point is defined 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.

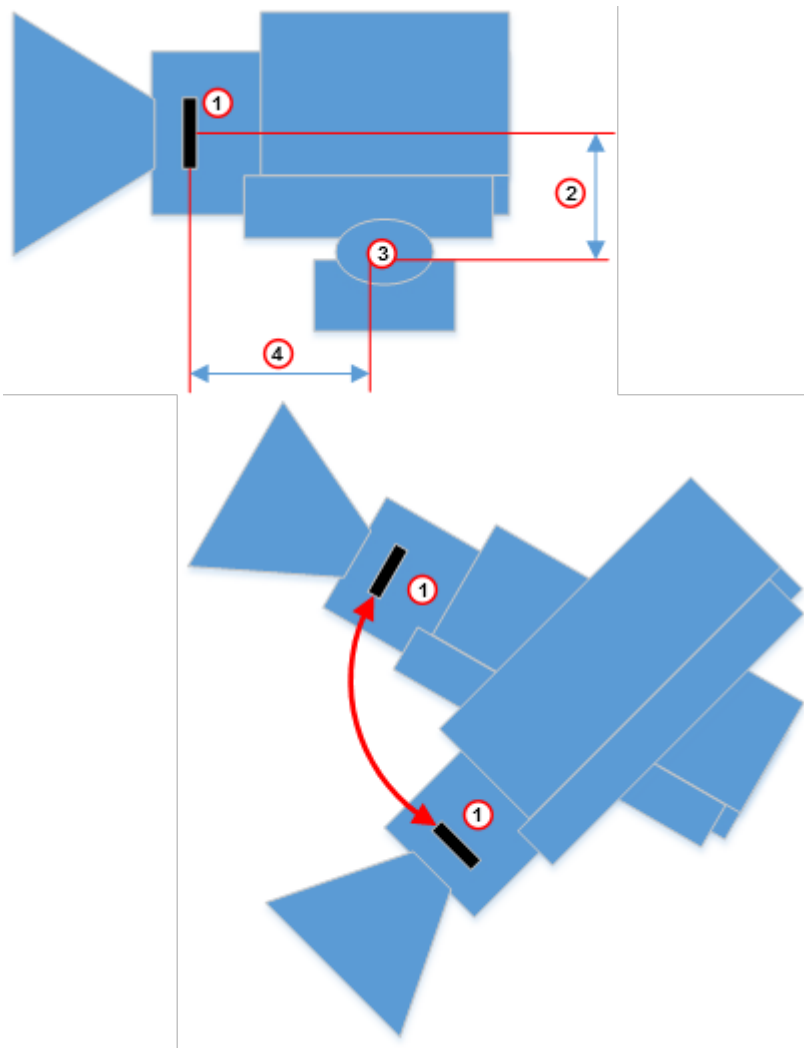
Center Shift

Whenever one object is mounted to another, the connection is not 100% straight. This is true for camera bodies and lenses as well. There is always an offset angle between the camera body and the lens. Therefore, the light does not come completely straight to the CCD. In the Viz Virtual Studio software this effect is called the Center Shift. Every lens shows its own center shift, which is much smaller than the mount error center shift and can be ignored most of the time. The most visible Center Shift is caused by the mount and becomes more and more visible/evident as the mount connection between the camera body and the lens becomes damaged.

⚠ Note: Currently Center Shift can be corrected in the lens file only.



Mounting Offsets



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

2.3.5 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.

It is 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.

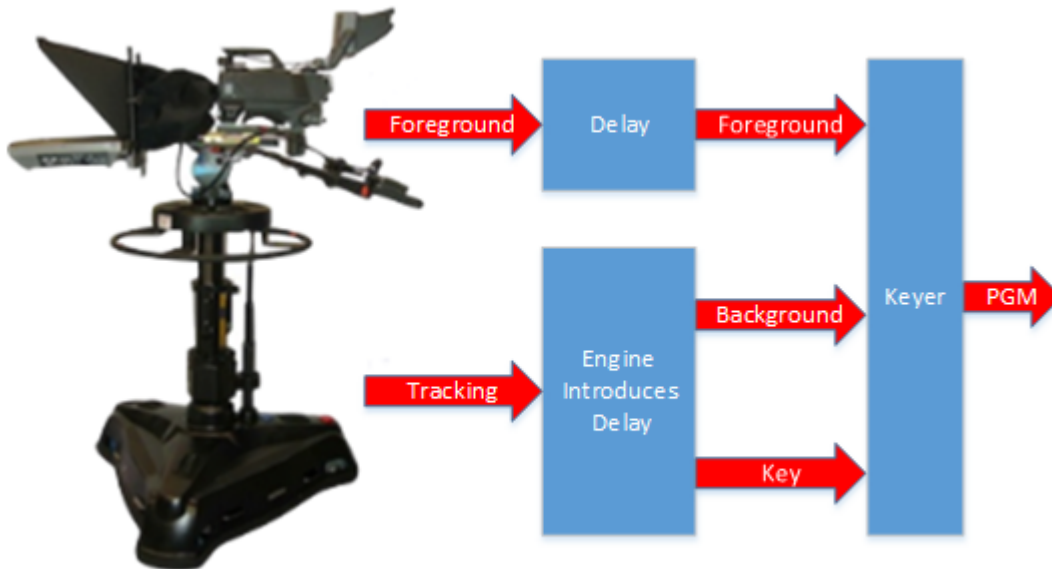
2.3.6 Delay

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

2.3.7 Video Delay



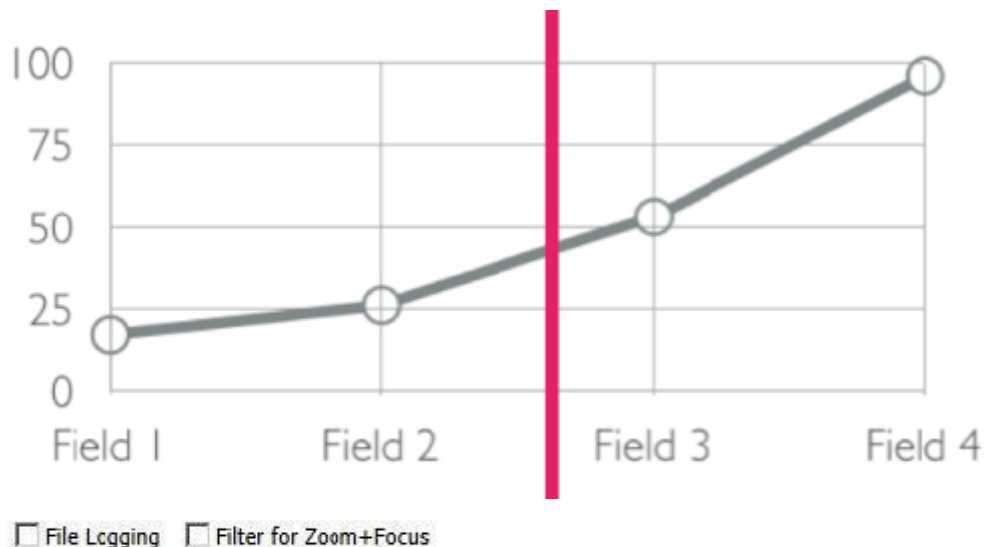
It is assumed that the camera and the tracking system are connected to the same sync signal. When the camera records an image, the tracking system acquires the camera's position, does some calculations and then sends this data to the Tracking Hub. One field later, Tracking Hub sends this tracking information to Viz Engine. Unfortunately, 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 Sapphire 3 have built in delay lines. An extra delay line must be installed when using older hardware.

2.3.8 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.

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?

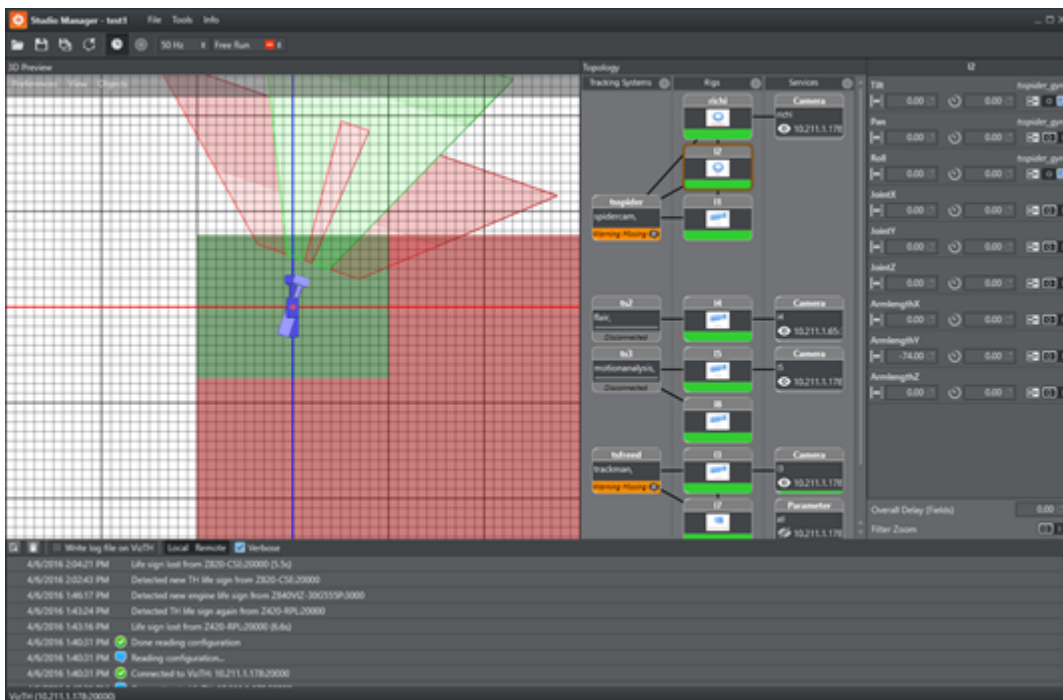
Note: When the delay differs 0.1 field from the video, it is visible to the eye!

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.

2.4 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](#)

2.5 New 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 old VALID/Sentinel/Hardlock Dongle licensing still works, as is.

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](#)
- [License Information](#)

2.5.1 Important Pre–installation Information

The WIBU licensing system requires the installation of the CodeMeter Runtime Software 6.60a (included in the Bundle installer).

When the license should be retrieved from a dedicated license server, it must be 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.

2.5.2 Key Features and Workflow of the New Licensing System

- VALID/Sentinel/Hardlock Dongle is still working without any changes. The license system to run on can be configured.
- 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).

2.5.3 Notable Limitations and Known Issues

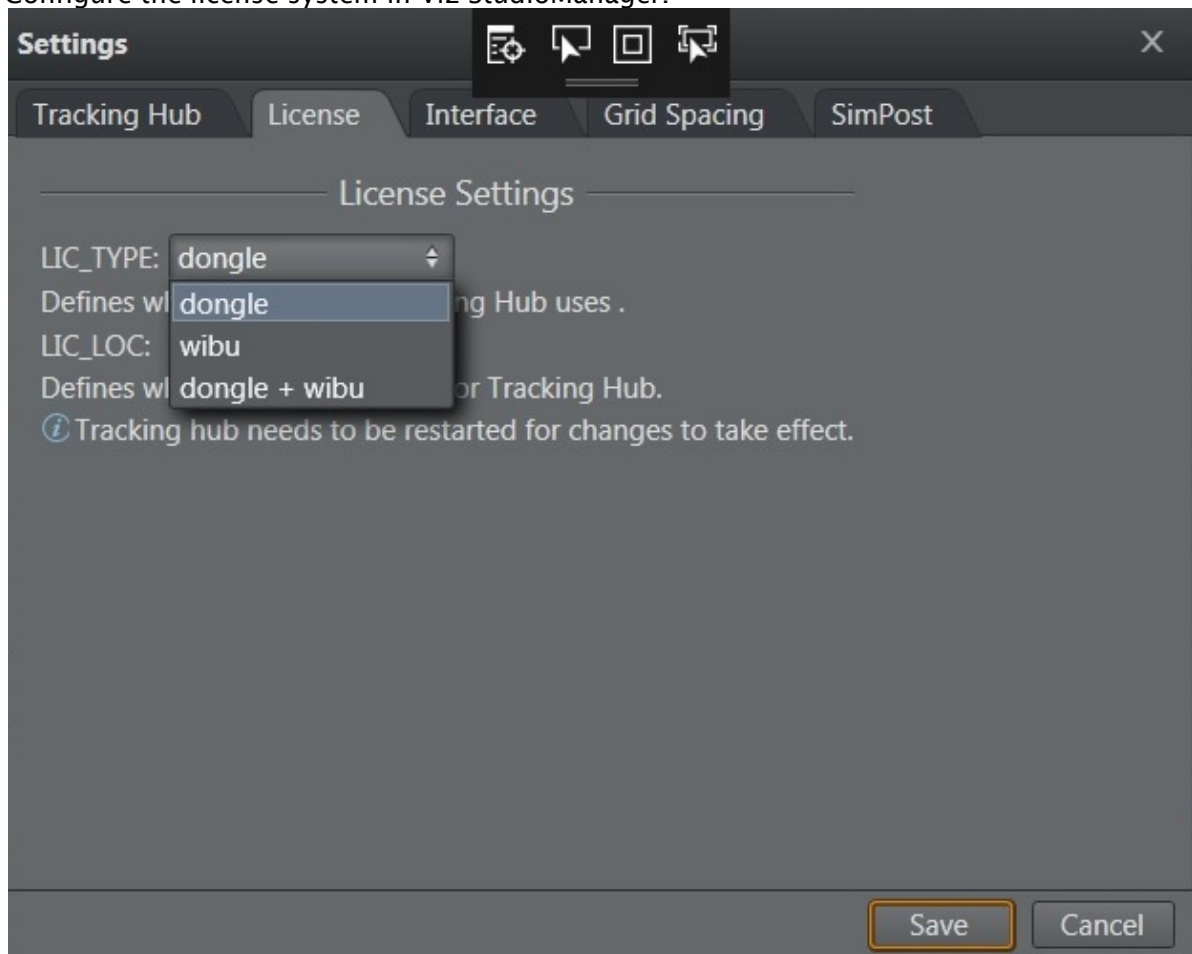
- Only one license system can be used (VALID/Sentinel/Hardlock or WIBU/CodeMeter).
- Uninstall any CodeMeter Runtime prior. And afterwards install the Virtual Studio and the newer CodeMeter Runtime.
- By default, the older Dongle licensing system is used when no configuration exists.
- Extension of a VALID/Sentinel/Hardlock license requires to restart Tracking Hub.

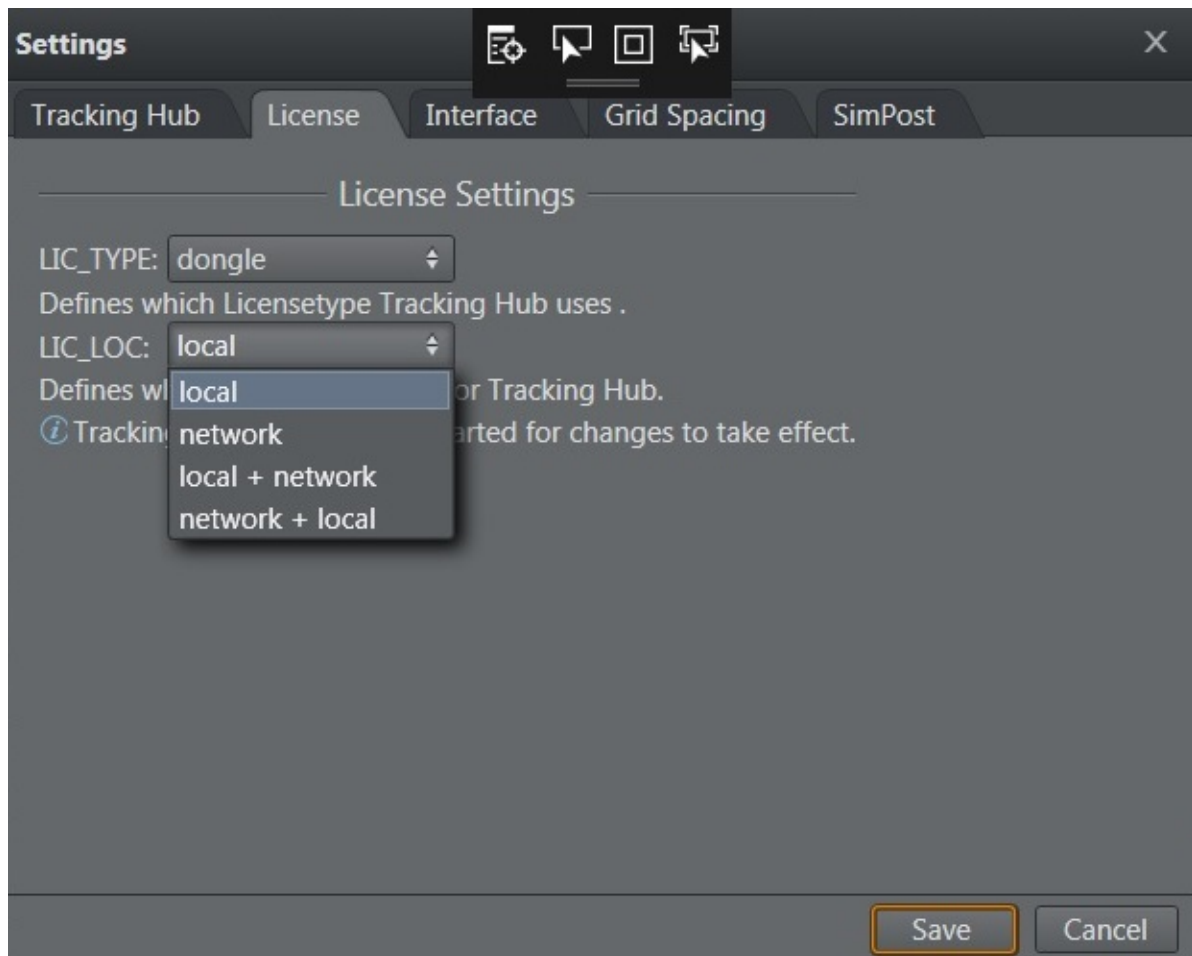
2.5.4 Basic Setup

These are the steps to set up Viz Virtual Studio licensing with WIBU or VALID/Sentinel/Hardlock:

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 VALID/Sentinel/Hardlock license system:
 - i. Attach the VALID/Sentinel/Hardlock dongle to a USB port.
 - ii. Configure to use legacy licensing. In the Studio Manager under **Tools Settings**, select **License**. Set LIC_TYPE to `dongle` and LIC_LOC to `local`.
 - b. When using a WIBU license server:
 - i. Open the CodeMeter WebAdmin and add the license server to the server search list.
 - ii. Configure Studio Manger to use the WIBU license system. In the Studio Manager under **Tools Settings**, select **License**. Set LIC_TYPE to `wibu` and LIC_LOC to `network` OR `network + local`.
 - c. 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_TYPE to `wibu` and LIC_LOC to `local`.
3. Click **Save** and restart.

2.5.5 License Information

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](#).
-

2.6 Supported Protocols

Currently, Tracking Hub has native support for the following protocols:

- FreeD (see [Description of the FreeD protocol](#))
- Shotoku
- RTHead
- NCam
- Xpecto
- XML Tracking: A special feature, a self-definable protocol for special solutions
- Motion Analysis (it is possible to work with two MA instances)
- mo-sys (with lens information)
- Trackman
- Vicon
- Libero
- Spidercam
- spidercamfd
- Spidercam-frameb
- Flair
- Thoma
- Stype A5
- Stype HF
- Camreginfo
- Technodolly
- Skycam
- Gsgeopoint
- Kuper
- FreeDa0
- Telemetrics
- hd-Skycam
- kromanov

2.7 WIBU License System In Tracking Hub

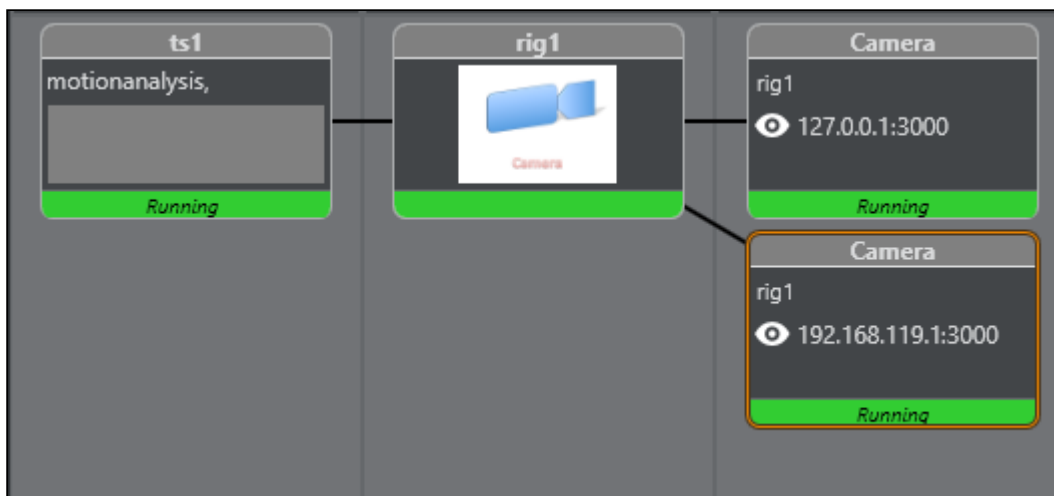
Tracking Hub supports the WIBU license system. This has the advantage that licenses can be held on a central server and licenses can be acquired by any machine on the company network. It is still possible to use WIBU Dongles on a Tracking Hub machine. The WIBU Driver and the WIBU control application Codemeter are installed with the new version of Tracking Hub.

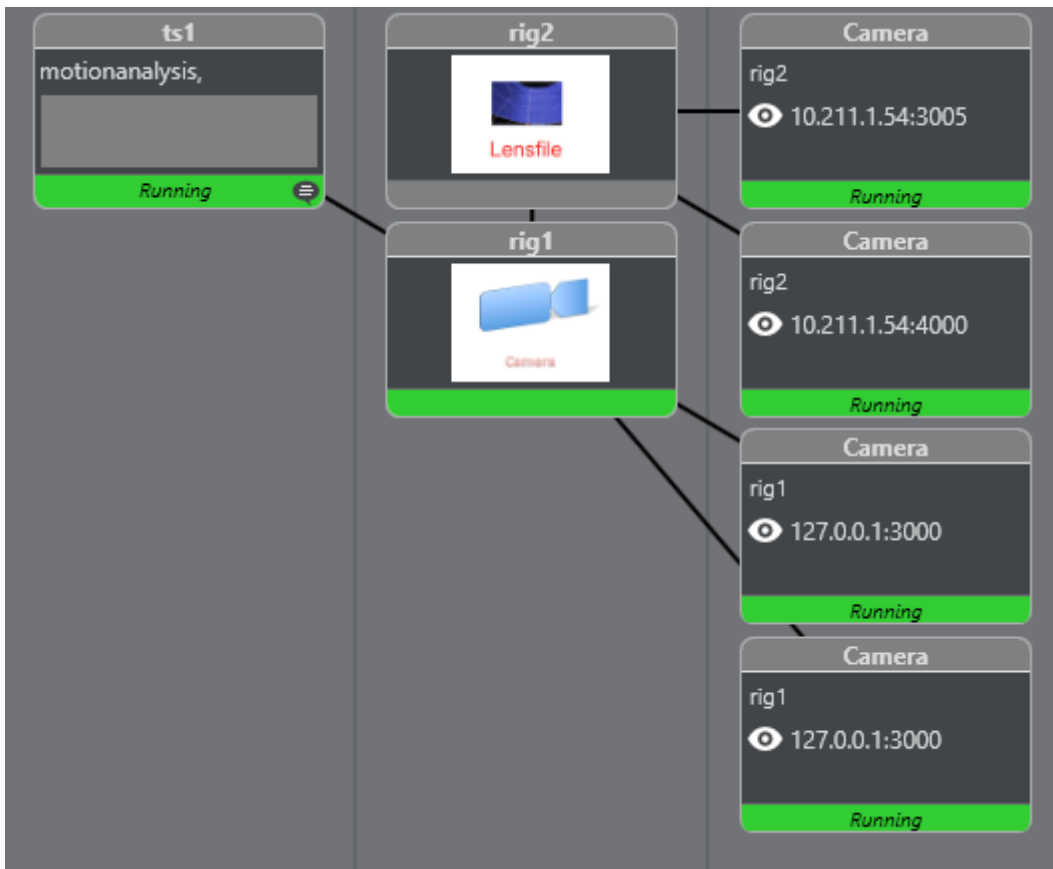
To run a Virtual Studio, two licenses are needed: The first one is called *Viz Virtual Studio*. Without this license, Tracking Hub starts up, but does not create any running services. This is the base license, used to run the Tracking Hub and the Studio Manager. The second license is for cameras. This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

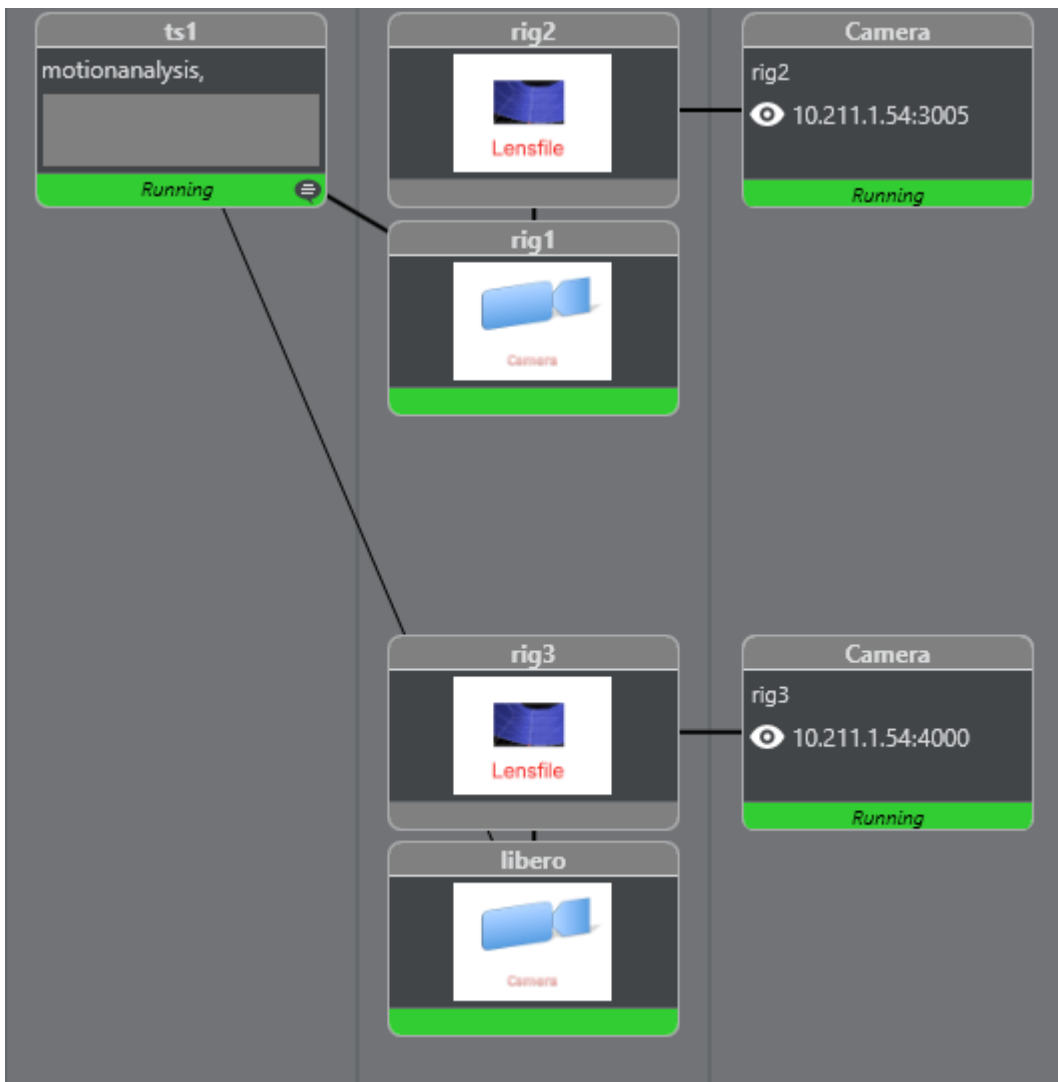
The *VS_CAMERA* license also limits the amount of rigs which can be connected to active Camera Services. It is still possible to create any amount of rig structures in the Studio Manager. Limited is the amount of rigs, which can be sent from Tracking Hub to the Engines in Virtual Studio.

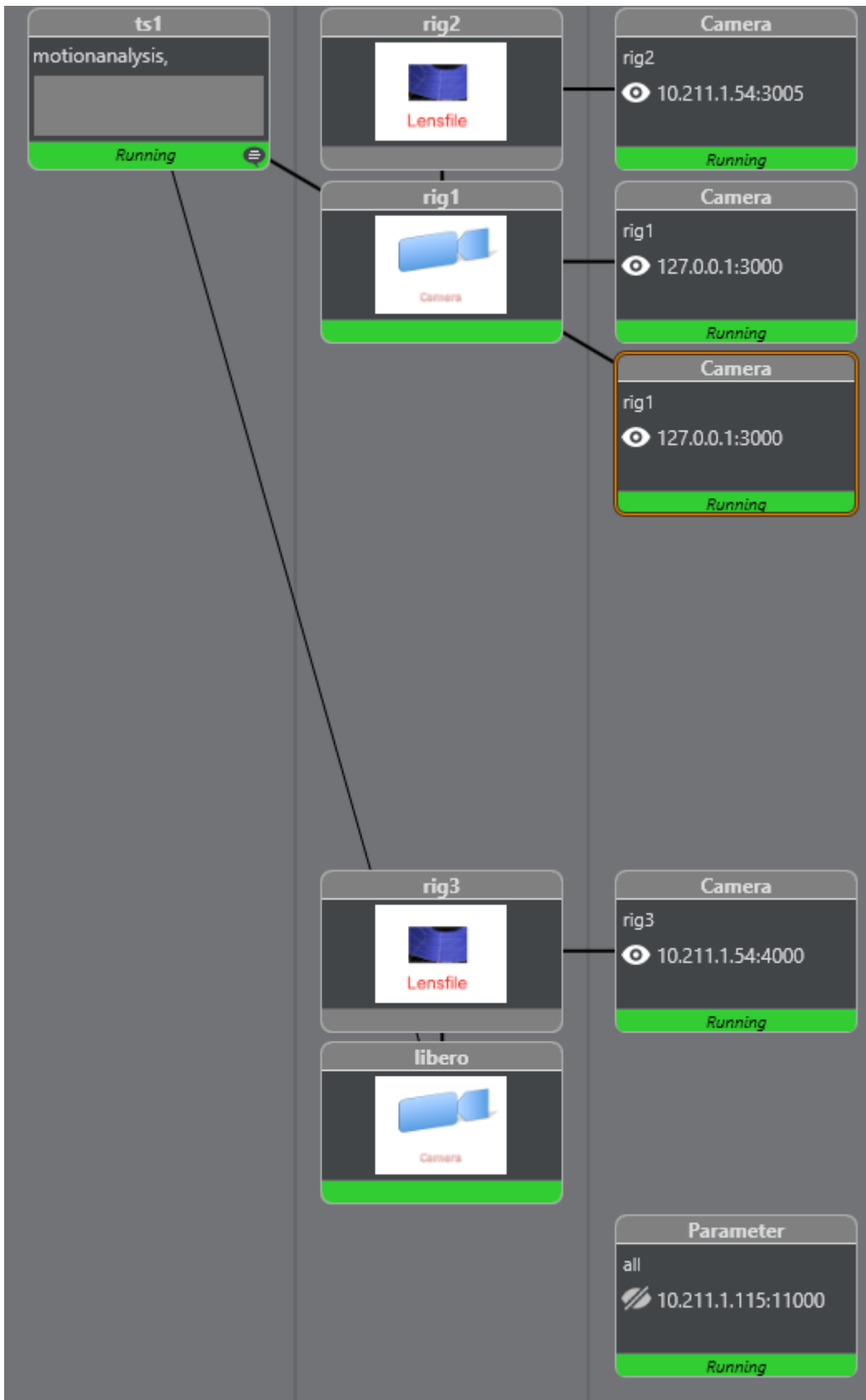
2.7.1 One License Consumed





2.7.2 Two Licenses Consumed





3 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 the Viz Virtual Studio](#)
-

3.1 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
 - Digital input 1&2 using SDI in
- 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.


3.1.1 System Requirements


This section details the system requirements.

3.1.2 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.

3.1.3 Minimum Hardware Configuration

- Minimum 8 GB Ram
- Dual-Core Intel CPU
- Windows 10 64-bit or Windows 7 (64-bit)
- Minimum 10 GB of free disk space
- OpenGL Hardware-accelerated Graphics Card

3.1.4 Minimum Hardware Configuration for use with Post and Replay


- Minimum 16 GB Ram
- Dual-Core Intel CPU
- Windows 10 64-bit or Windows 7 (64-bit)
- Minimum 30 GB of free disk space
- OpenGL Hardware-accelerated Graphics Card

3.1.5 Recommended Hardware and Software Configuration

- 32 GB Ram
- Quad-Core Intel CPU
- Windows 10 64-bit
- 30 GB of free disk space
- Nvidia P2000 Graphics Card

3.1.6 Supported Hardware and Software

- Windows 7 (64-bit)
- Windows 10 (64-bit)
- Viz Engine 3.9.0 (either 32-bit or 64-bit) or later
- Plura PCL-PCI and PCL-PCIe (L, LV, 3G) timecode reader cards (Driver 5.34)

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

- Studio Manager requires .NET framework 4.5.1 or higher.

See Also

- [Viz Virtual Studio Installation](#)
- [Start the Viz Virtual Studio](#)

3.2 Viz Virtual Studio Folders

3.2.1 Installation Folders

The default installations folders are:

Windows System	Viz Virtual Studio Platform	Installation Folder
Windows 7 32-bit	32-bit	C:\Program Files\vizrt\VizTHC:\Program Files\vizrt\Studio Manager
	64-bit	N/A
Windows 7 64-bit	32-bit	C:\Program Files (x86)\vizrt\VizTHC:\Program Files (x86)\vizrt\Studio Manager
	64-bit	C:\Program Files\vizrt\VizTHC:\Program Files (x86)\vizrt\Studio Manager

3.2.2 Data Folders

Files which are created or can be modified are located in %ProgramData%\vizrt\VizTH, and include:

- Config files.
- Crash dump files.

Temporary files are located at: %TMP%\vizrt\viz3, which usually resolves to C:\Users\\AppData\Local\Temp\vizrt\viz3. This folder is referenced as *<viz temp folder>* throughout this User Guide.

3.3 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-<VERSION>.exe	Bundle with Tracking Hub and Studio Manager
StudioManager.<VERSION>.msi	Studio Manager installer
TrackingHub<VERSION>.msi	Tracking Hub installer

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

Prerequisites for Tracking Hub are:

- Microsoft Visual Studio C++ 2012 Runtime
- Sentinel Runtime/WIBU Codemeter 6.60a

Prerequisites for Studio Manager are:

- Microsoft .Net 4.5.1 or higher.

Studio Manager asks to download Microsoft .Net 4.5.1 on startup if it is not already installed.

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

1. Log on to the computer as a Computer Administrator and start `TrackingHubBundle.<VERSION>.exe` to run the setup application.

- By default, the Tracking Hub and Studio Manager options are selected in the installer. If required, remove the options.



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

Note: The packages to install can be selected by check boxes, except the **Microsoft Visual C++ 2012 Redistributable** package, as it is a prerequisite. If a package is already installed (e.g. the Sentinel Runtime Dongle Driver), the check box is disabled.

After installation, the **Programs and Features** panel shows entries for each installed package and one entry for the Bundle:

Name	Publisher	I...	Size	Version
Sentinel Runtime	SafeNet Inc.	25.09.2015	15,0 MB	7.32.1.52786
Vizrt Tracking Hub 1.0.0.50612 32bit	Vizrt	25.09.2015	5,42 MB	1.0.0.50612
Vizrt Studio Manager 1.0.0.50612 32bit	Vizrt	25.09.2015	19,8 MB	1.0.0.50612
Vizrt Tracking Hub Bundle 1.0.0.50612 32bit	Vizrt	25.09.2015	82,8 MB	1.0.0.50612
Microsoft Visual C++ 2012 Redistributable (x86) - 11.0.61030	Microsoft Corporation	25.09.2015	17,3 MB	11.0.61030.0

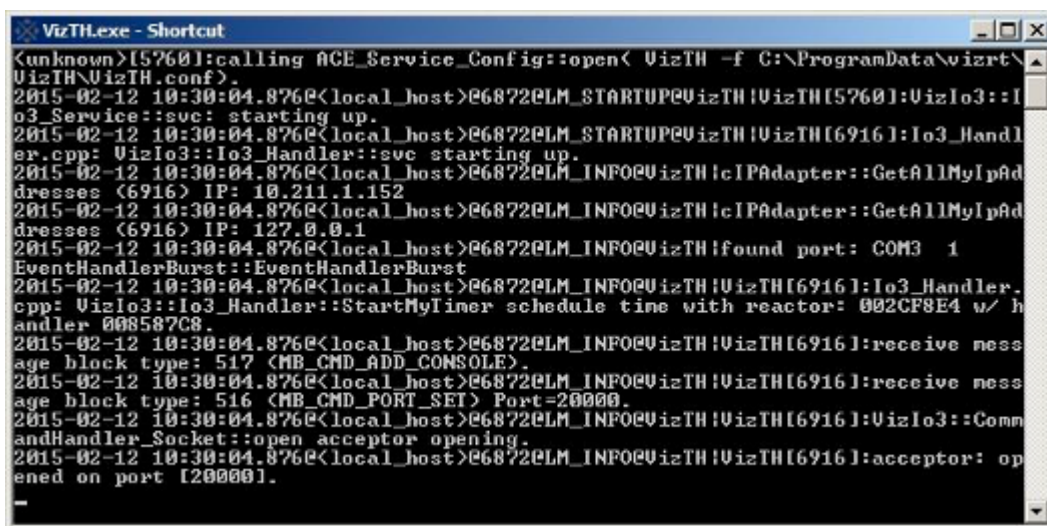
3.3.2 To Repair or Remove Tracking Hub

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 each individual package. Exceptions are the **Microsoft Visual C++ 2012 Redistributable** and the **Sentinel Runtime** packages, as they are intended to remain on the PC. You can remove these packages separately, if desired.
- After Repairing, you must restart the computer. After restarting, you can proceed to [Start the Viz Virtual Studio](#).

3.4 Start The Viz Virtual Studio

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



```

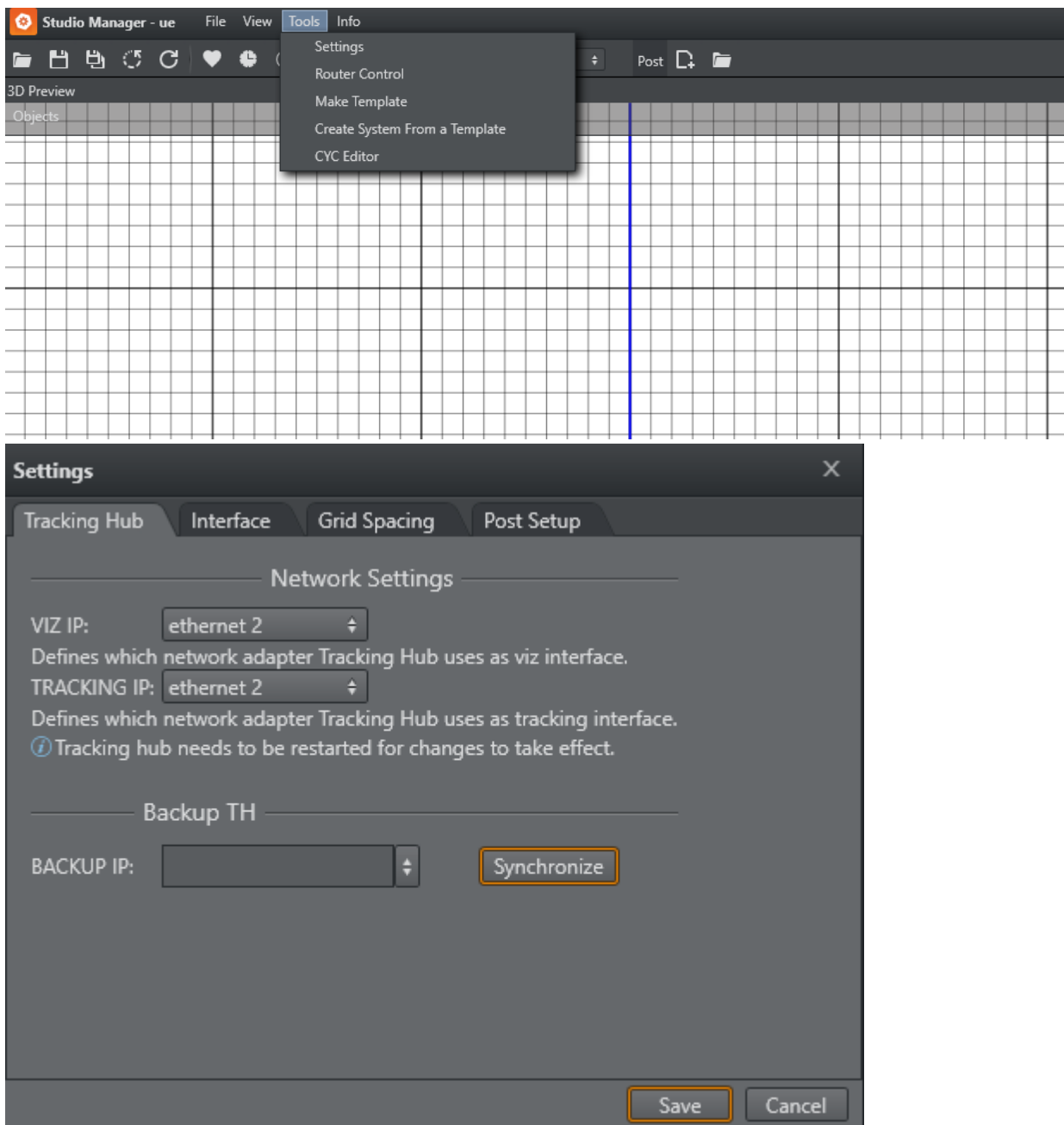
VizTH.exe - Shortcut
<unknown>[15760]:calling ACE_Service_Config::open< VizTH -f C:\ProgramData\vizrt\
VizTH\VizTH.conf>.
2015-02-12 10:30:04.876@<local_host>@6872@LM_STARTUP@VizTH[VizTH[15760]:VizIo3::I
o3_Service::svc: starting up.
2015-02-12 10:30:04.876@<local_host>@6872@LM_STARTUP@VizTH[VizTH[6916]:Io3_Handl
er.cpp: VizIo3::Io3_Handler::svc starting up.
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:Io3_Handler
addresses (6916) IP: 10.211.1.152
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:Io3_Handler
addresses (6916) IP: 127.0.0.1
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:Io3_Handler
EventHandlerBurst::EventHandlerBurst
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:Io3_Handler
.cpp: VizIo3::Io3_Handler::StartMyTimer schedule time with reactor: 002CF8E4 w/ h
andler 008587C8.
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:receive mess
age block type: 517 (MB_CMD_ADD_CONSOLE).
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:receive mess
age block type: 516 (MB_CMD_PORT_SET) Port=20000.
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:VizIo3::Comm
andHandler_Socket::open acceptor opening.
2015-02-12 10:30:04.876@<local_host>@6872@LM_INFO@VizTH[VizTH[6916]:acceptor: op
ened on port [20000].

```

Tracking Hub does not load correctly if the configuration file `VizTH.conf`, located at `C:\ProgramData\vizrt\VizTH`, is missing.

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. Therefore, 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 **VizTH Settings** from the **Tools** menu in Studio Manager (1), then select the desired network adapters (2).



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

3.4.1 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`.

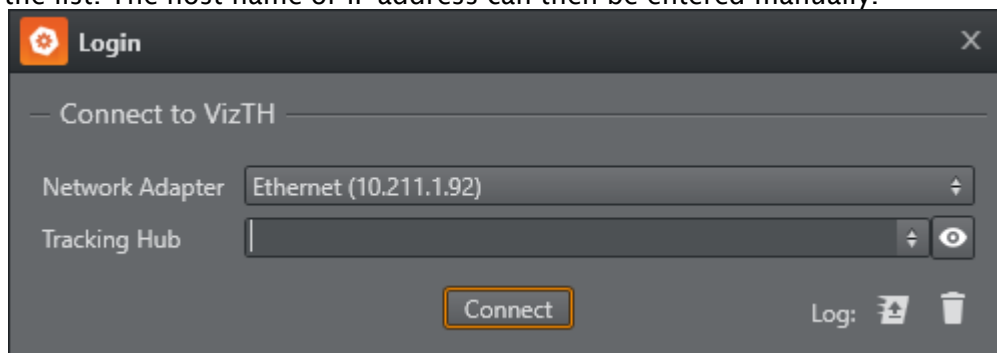
4. **VizAdapter** and **TrackingAdapter** can have the same setting.

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.

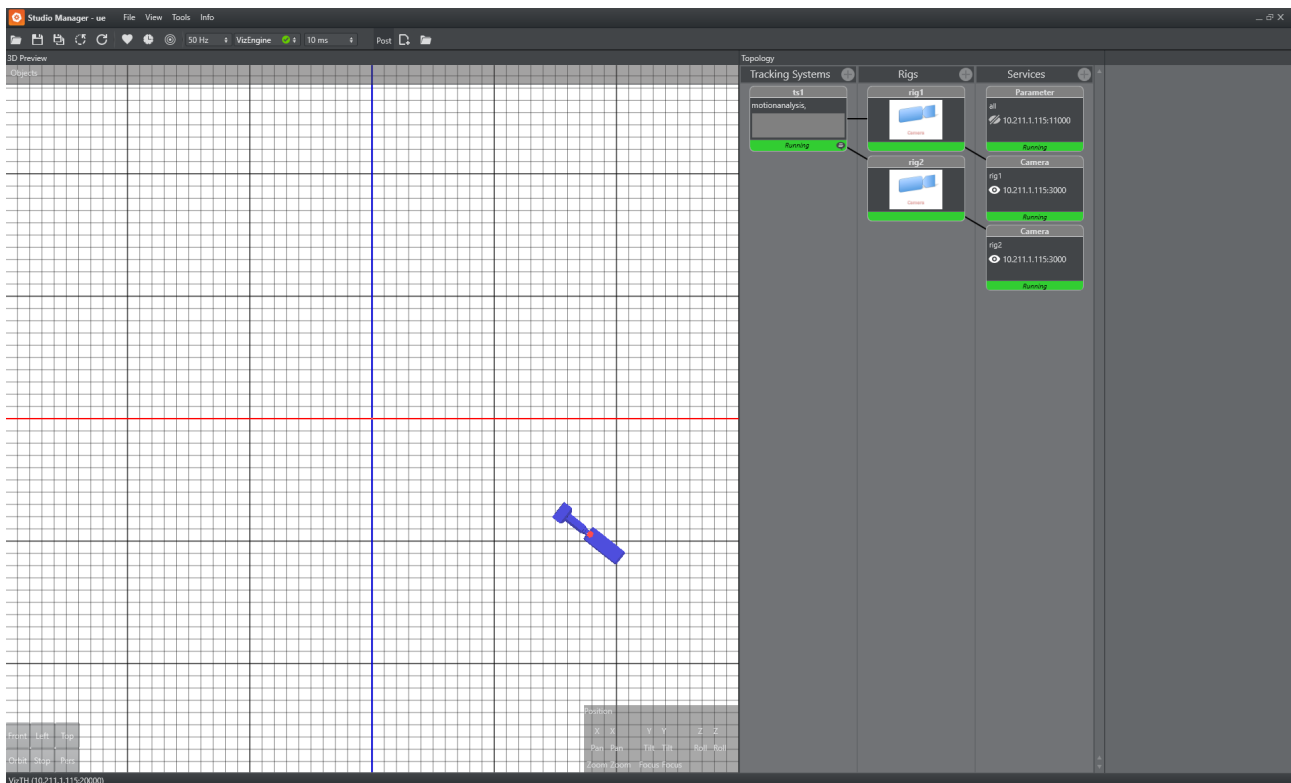
3.4.2 Start the Studio Manager

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.



3. Click **Connect**.

4 Studio Manager GUI

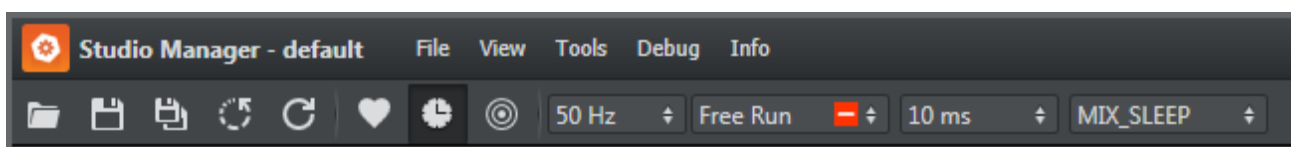


This section details each of the Studio Manager panels and their properties:

- [Configuration Panel](#)
- [Parameter Panel](#)
- [Topology Panel](#)
- [Log Panel](#)
- [Timing Analysis Window](#)
- [Preview Panel](#)
- [Post System](#)


4.1 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.

- **Save:** Saves the current Studio configuration.
- **Save As:** Saves the current Studio configuration with a new name.
- **Reset:** Resets the current Studio configuration. Note that the loaded configuration file is only changed if you click **Save** after resetting the configuration.
- **Refresh:** 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.
- **View Log:** Opens or closes the [Log Panel](#).
- **View Timing Analysis:** Opens the [Timing Analysis Window](#).
- **Frequency:** Selects the production frequency: 50 Hz, 60 Hz or 59.94 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.

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

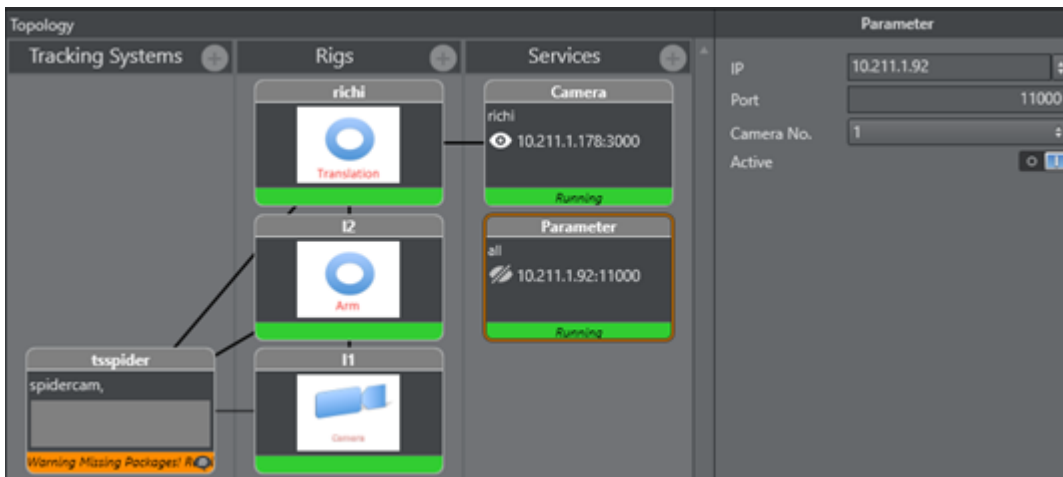
- **Send Delay:** Sets the time Tracking Hub waits until it sends tracking data to the Engine. This setting is especially important if your tracking delay is less than one field. With the help of the Timing Analysis, the transmission time can be set after the arrival of the tracking data. If your tracking delay is larger than one field, set the send delay value to **No Delay**. This reduces the CPU usage.
- **Busy Loop:** If it is necessary to set the **Send Delay**, use **Busy Loop** to select the way this happens. This setting depends mainly on your hardware.
 - **WITH SLEEP:** Lowest CPU load, less exact.
 - **MIX SLEEP:** Uses both WITH and NO SLEEP, almost exact.
 - **NO SLEEP:** Highest CPU load, exact.

See Also

- [Parameter Panel](#)
- [Topology Panel](#)
- [Log Panel](#)
- [Preview Panel](#)

4.2 Parameter Panel

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

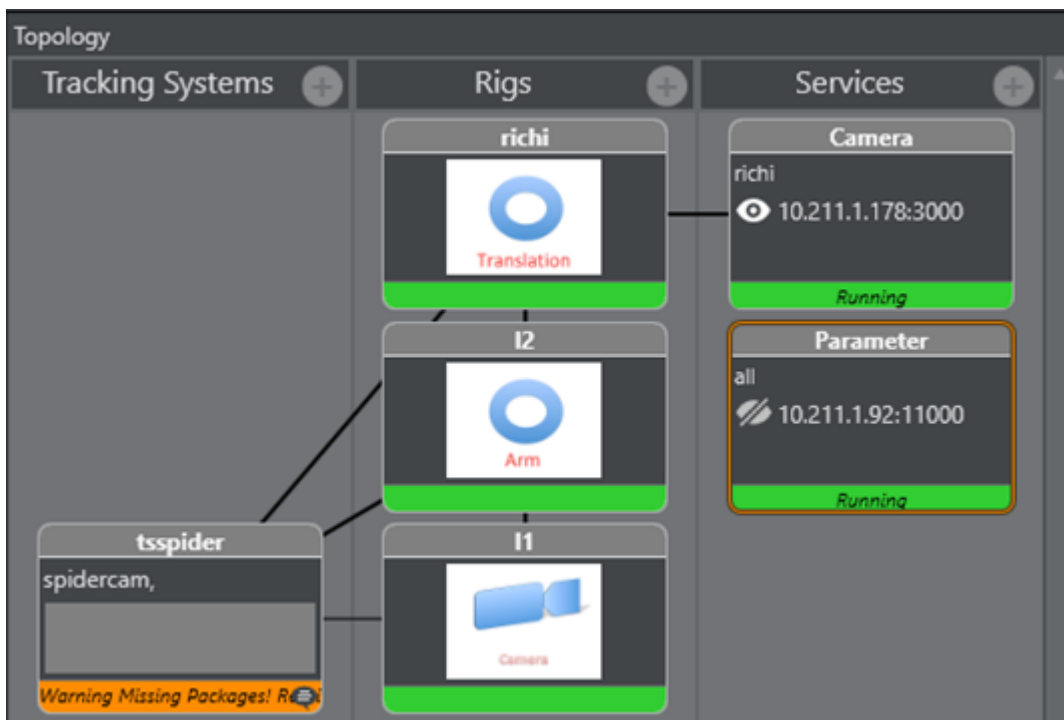


Click on a Tracking System, a Rig, or a Service, to view or change its parameters as required.

4.3 Topology Panel

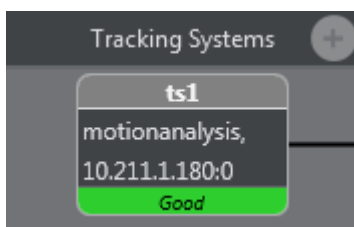
- [Tracking Systems](#)
- [Rigs](#)
- [Services](#)
- [Topology Connection Lines](#)
- [Topology Colors](#)
- [Motion Capturing with Motion Analysis](#)

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

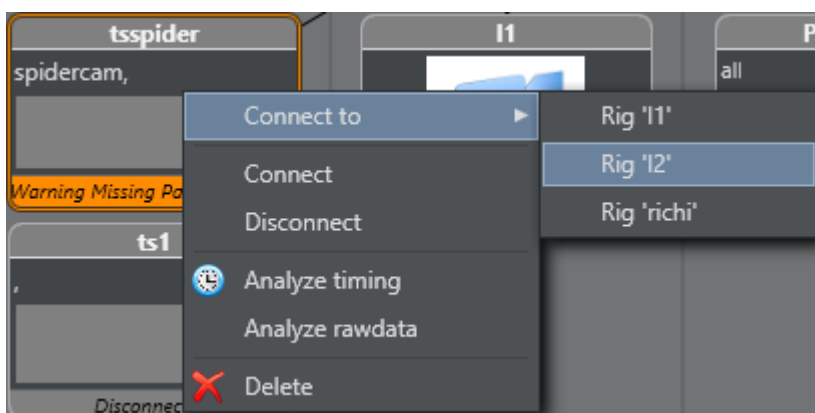


4.3.1 Tracking Systems

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.



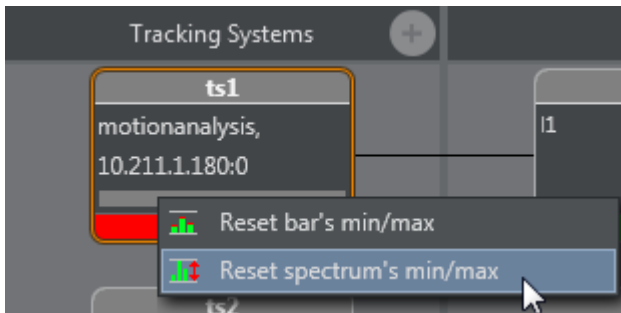
Right-click Menu:



- **Connect to:** Connects to a Rig.

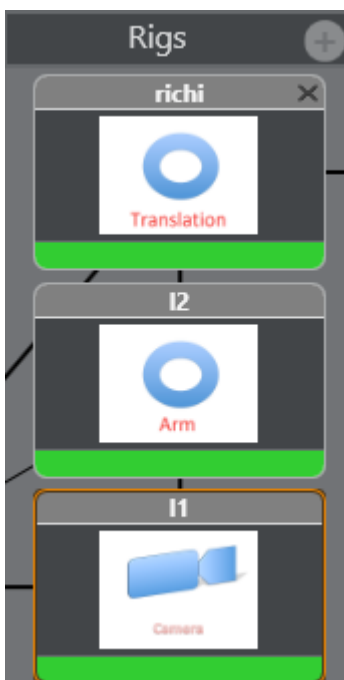
- **Connect/Disconnect:** Connects to or disconnects from the tracking system.
- **Analyze timing:** See Analyze Time.
- **Analyze rawdata:** Enables or disables the collection of raw data for analytical purposes.
- **Delete:** Removes the Tracking System.

Right-click on a Bar:



- **Reset bar's min/max:** Resets the minimum and maximum value of each bar.
- **Reset spectrum's min/max:** Resets the minimum and maximum value of the whole spectrum and each bar.

4.3.2 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:

- Translation
- Arm
- Location Pivot Rotation
- Pivot Rotation
- Camera Standard (rotation order YXZ Pan - Tilt - Roll)
- CameraXZY (rotation order YZX Pan - Roll - Tilt)
- CameraYXZ (rotation order ZXY Roll - Tilt - Pan)
- CameraFD (special design for Spidercam construction)
- Object
- Extended LensParameters
- Lensfile

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

- pos_x
- pos_y
- pos_z

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:



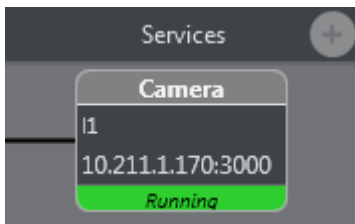
The Location Pivot Rotation rig is a rotation interface in the X-Z dimension, with center point (0/0). Pivot Rotation is a rotation rig with a definable center point. The rig provides the Tracking Hub with the following coordinates:

- pivotx
- pivoty
- pivotz

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

4.3.3 Services

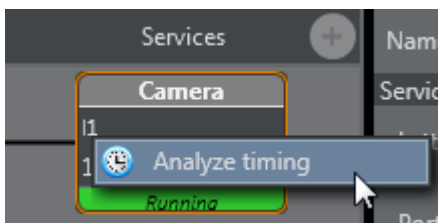
A Service is the element of a Viz Engine which is connected to the Tracking System through the Rig. Click on a Service icon to view its properties (see [Parameter Panel](#)). A line from the a Service represents a connection to a Rig (see Rigs).



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 the engines.

Right-click display object for additional menu:



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

4.3.4 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:

Tracking System "tsspider"		Rig "richi"				
Tracking parameter	Connected	Name	Inverted	Offset	Delay	
tsspider_pan	tsspider_posx x	PosX	<input type="checkbox"/>	0	0	
tsspider_tilt	tsspider_posz x	PosY	<input checked="" type="checkbox"/>	135	0	
tsspider_zoom	tsspider_posy x	PosZ	<input checked="" type="checkbox"/>	0	0	
tsspider_focus						
tsspider_gyrox						
tsspider_gyroz						
tsspider_gyroxy						
tsspider_posx						
tsspider_posz						
tsspider_posy						
tsspider_iris						

OK

4.3.5 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.

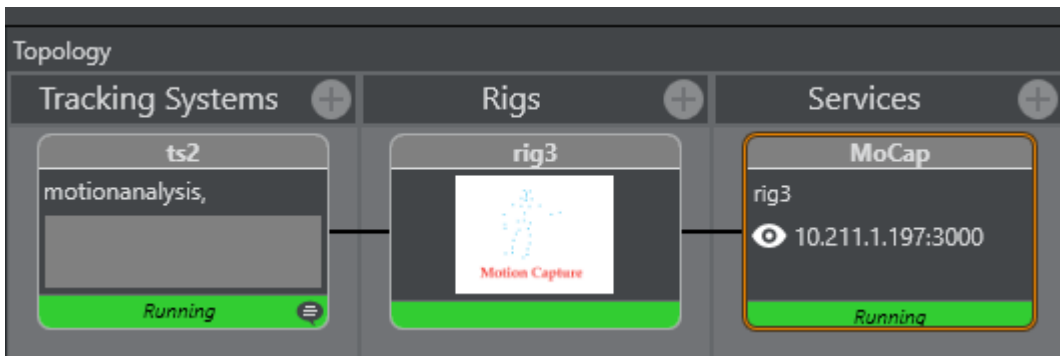
4.3.6 Motion Capturing with Motion Analysis

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.

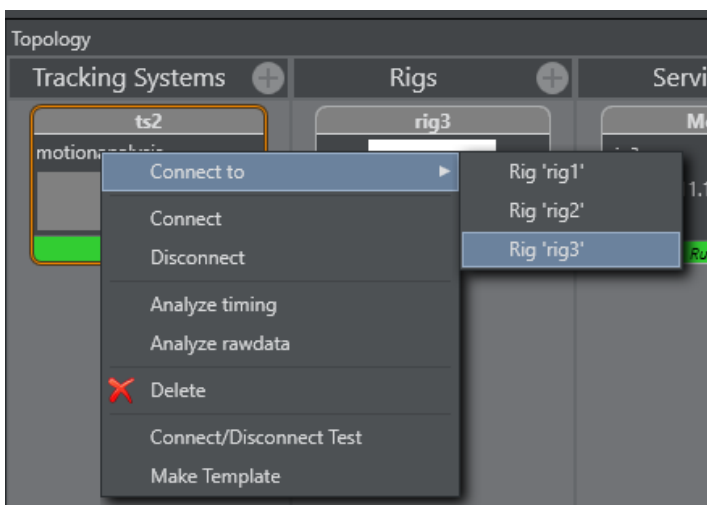
Selectable via the menu:

Rigs --> Add MoCap

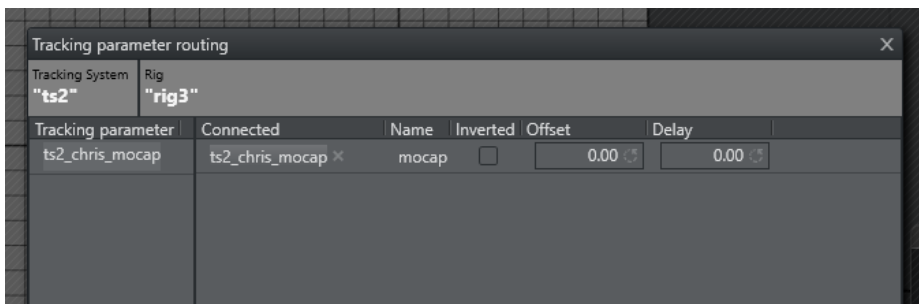
Services --> Add MoCap service



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



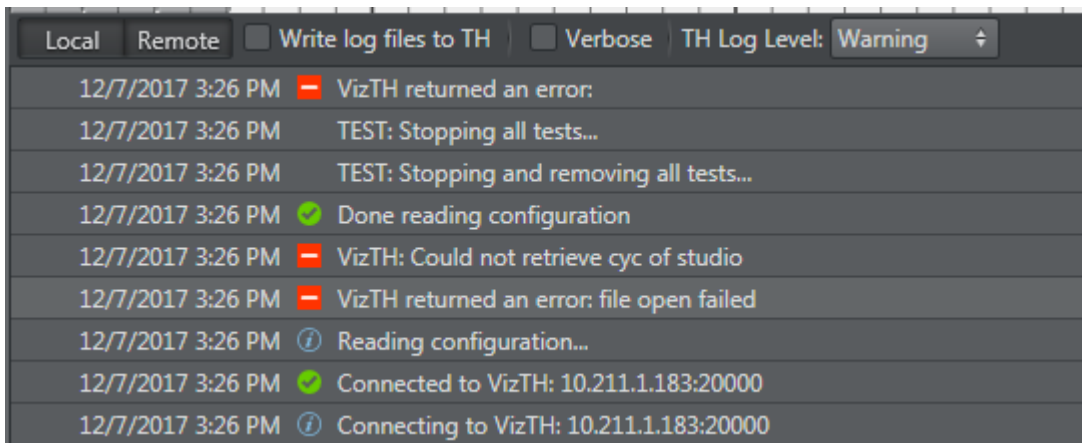
In this case, there is just one axis available.





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

4.4 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_rcv1.txt
 - VizTH_send1.txt
 - VizTH_timing1.txt
 Log-files are located in C:\ProgramData\vizrt\VizTH. If the log-file size reaches 1 MB, a new file is created with an incremented number.
- **Local Remote**: 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 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). In certain situations, these messages cannot be written to the console. In such cases, they are written to the log file only.

Messages are shown in the console and written to `VizTH.log`. From Level **Fatal** up to **Information**, the debugs are shown in the console and are written to the `VizTH.log`.

4.5 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).

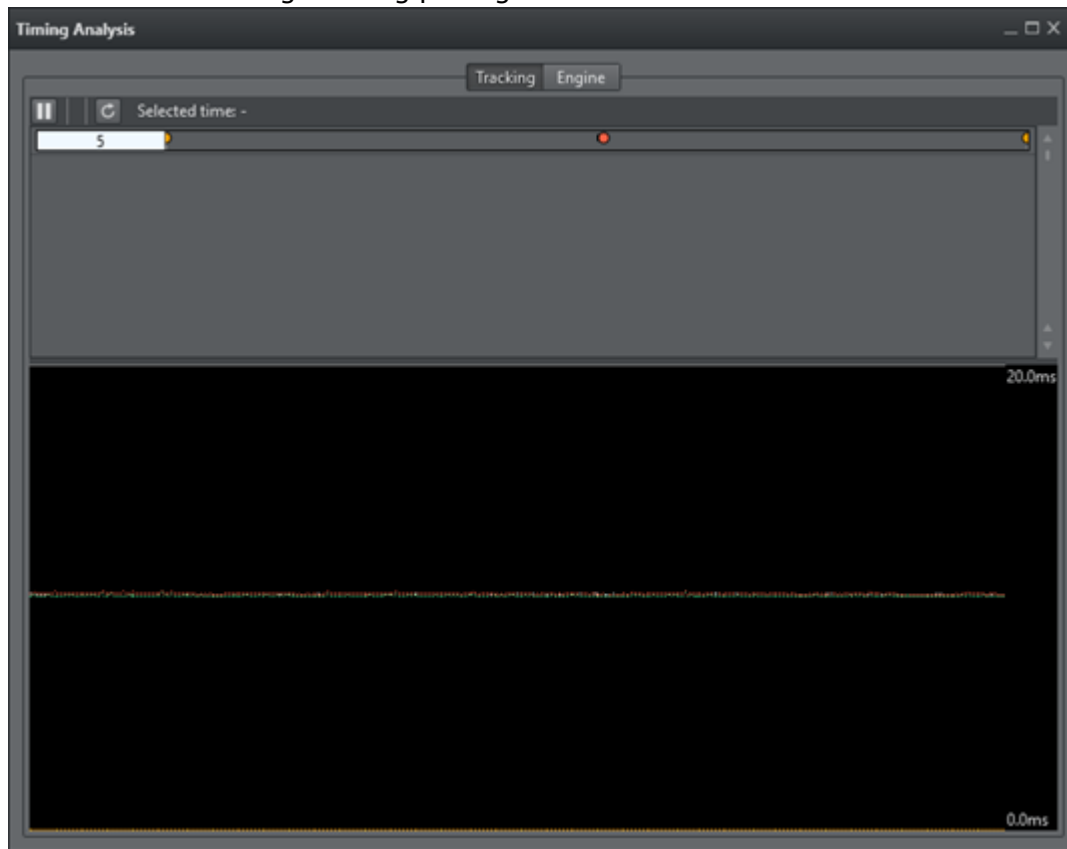
In the lower half of the panel, some of the events are plotted over time. The newest events are displayed at the right border and constantly move towards the left border as time passes. Colors correspond to those in the timing tracks in the top half of the panel.

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.

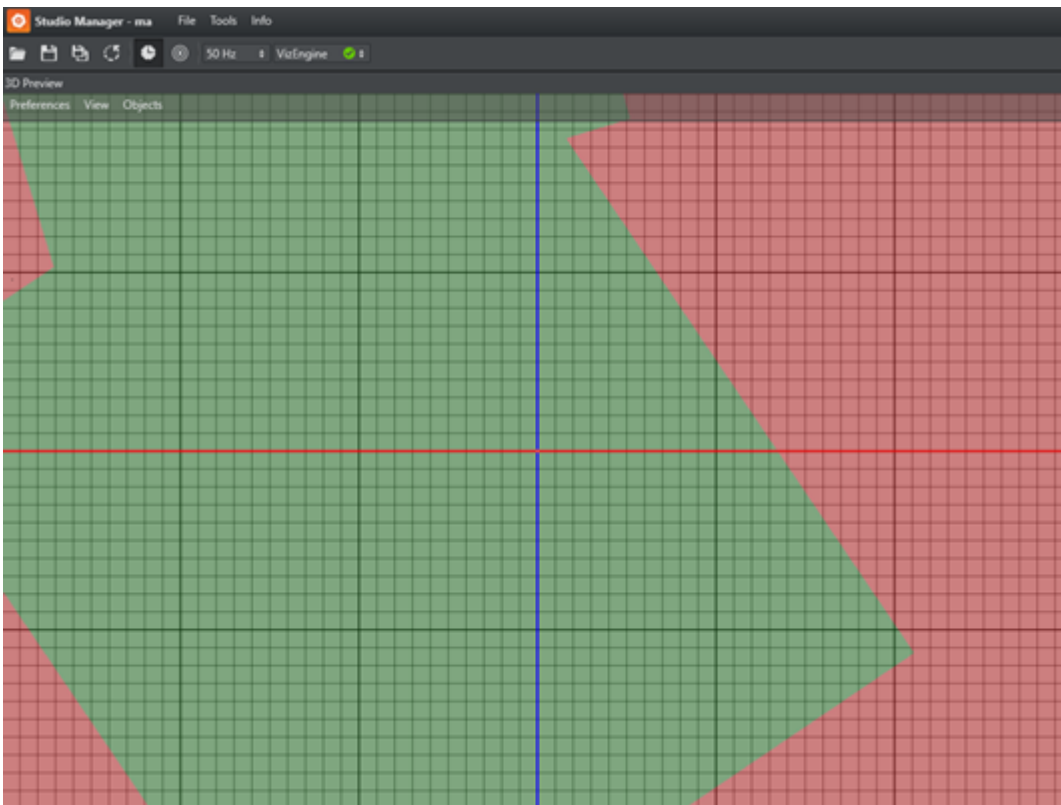


See Also

- [Configure the Studio](#)
- [Configuration Panel](#)
- [Parameter Panel](#)
- [Topology Panel](#)
- [Preview Panel](#)

4.6 Preview Panel

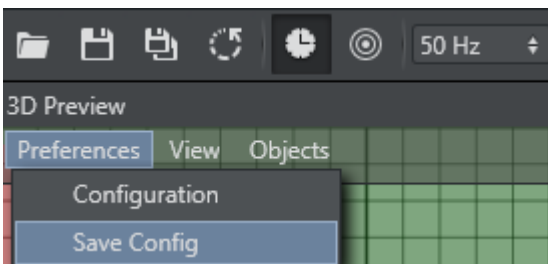
The 3D Preview panel gives a preview of the tracked cameras, which should be as close to the real studio as possible. When the Studio Manager starts up, a Parameter Service is established, which sends the positions of all tracked rigs to the preview panel:



The preview panel consists of the following parts:

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

4.6.1 Preferences Menu



- **Configuration:** Opens the Color Configuration window.
- **Save Config:** Saves changes made in the configuration.

4.6.2 Configuration

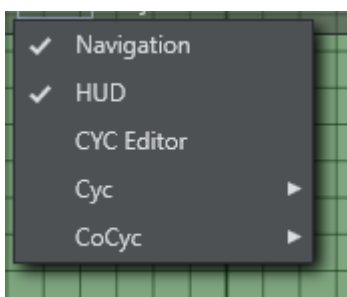
In this window, the user can adjust the render colors for the visible objects in the preview panel. Click in a drop-down list to open the color selection window. The RGB and Alpha values can be adjusted for all colors, by clicking the **Advanced** button.



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.

4.6.3 View Menu



4.6.4 Navigation

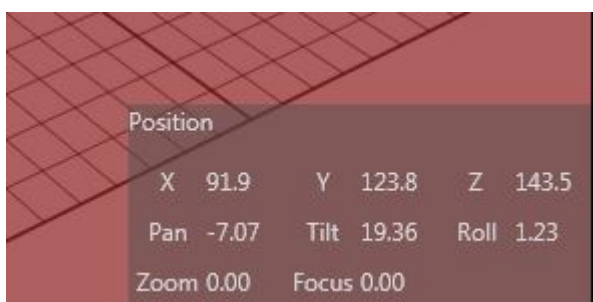
Activates or deactivates the navigation bar in the bottom left corner of the 3D Preview window:



- **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.
 - **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.
- **Move:** Adjusts the relative camera position when no middle mouse button is present. Adjust by pressing the **Move** button and dragging the mouse.
- **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.

4.6.5 HUD

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



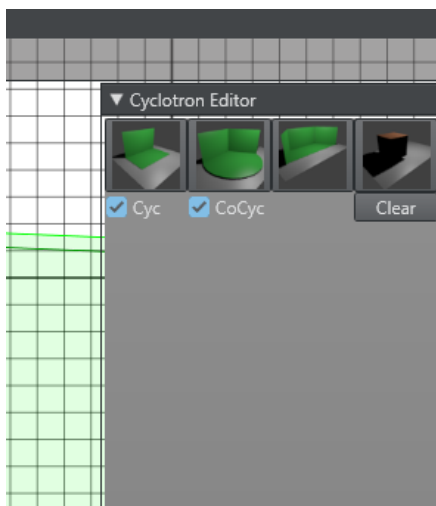
4.6.6 Cyclotron Editor

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.

A new Cyclotron Editor (Cyc Editor) was introduced in version 1.2. Many customers were used to the old VizIO Cyc editor and were satisfied with the functionality. This feature is back in Studio Manager, and at the moment, the two Cyc definition methods coexist together.



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.



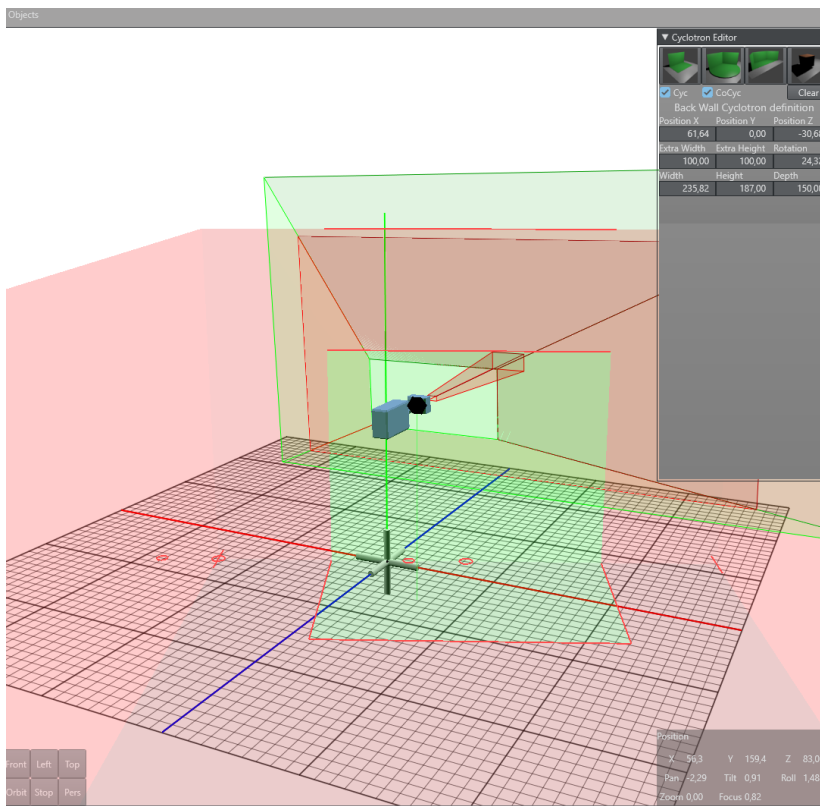
There are three Cyclotron types are available: Back-Wall, One-Corner and Two-Corner Cyc.

Common Cyc-Editor elements

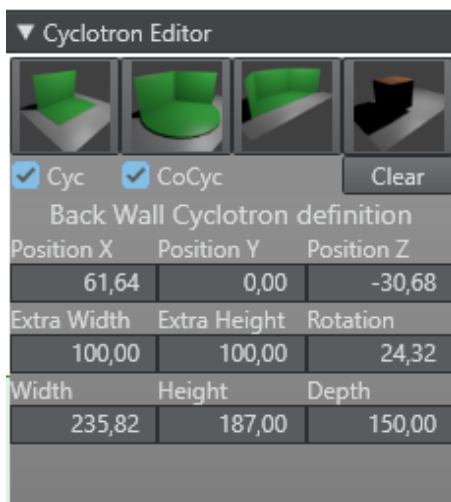
Position X	Position Y	Position Z
61,64	0,00	-30,68
Extra Width	Extra Height	Rotation
100,00	100,00	24,32

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.

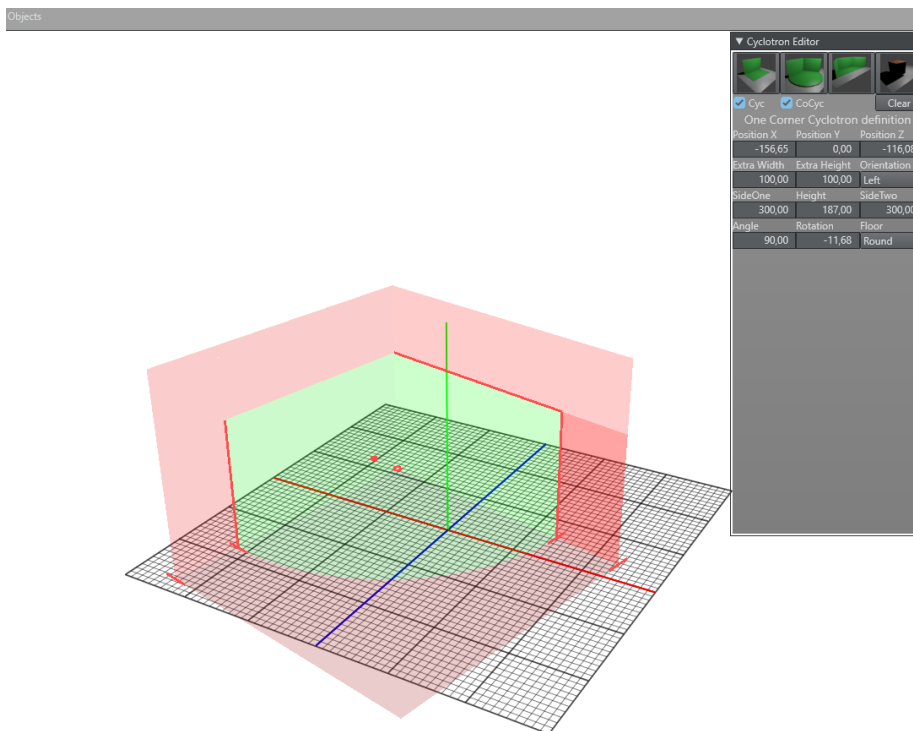
Back-Wall



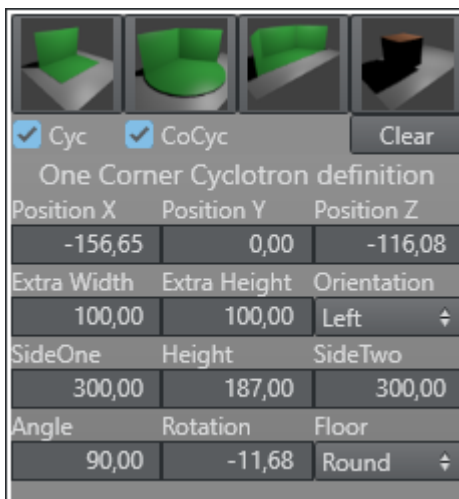
The back-wall Cyc has three additional parameters: These are the width of the green wall, height of the green wall and the depth of the ground area.



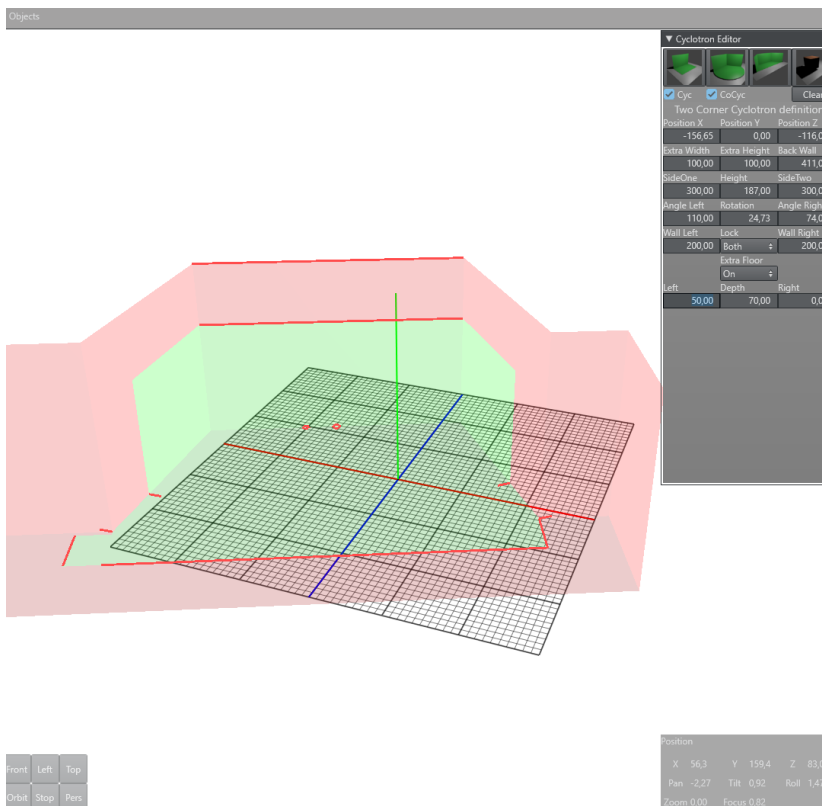
One Corner



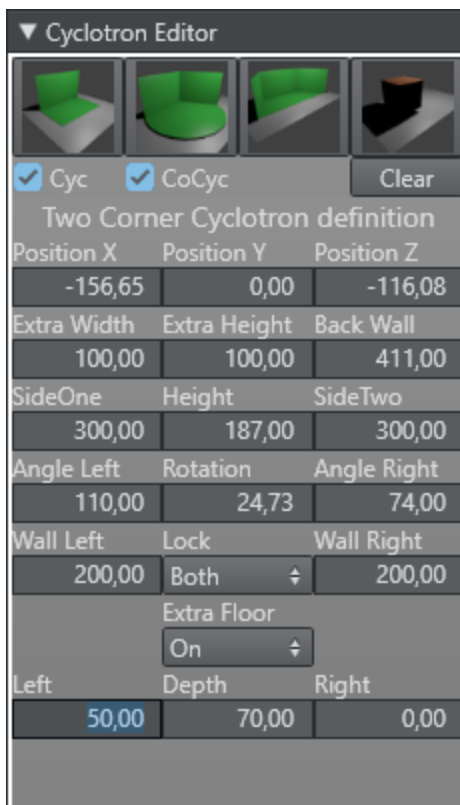
The One Corner Cyc is defined by two sides (side one and side two) and an opening angle. Additionally, the shape of the ground plane can be selected. The user can choose Round, Straight and Rectangular form.



Two Corner

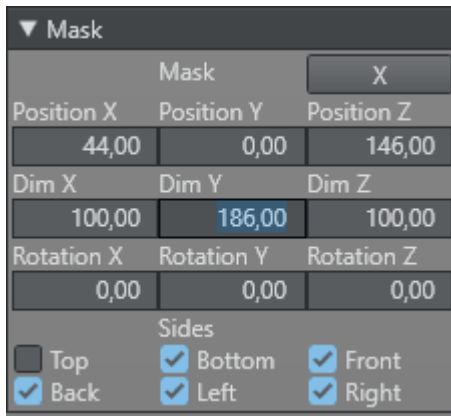
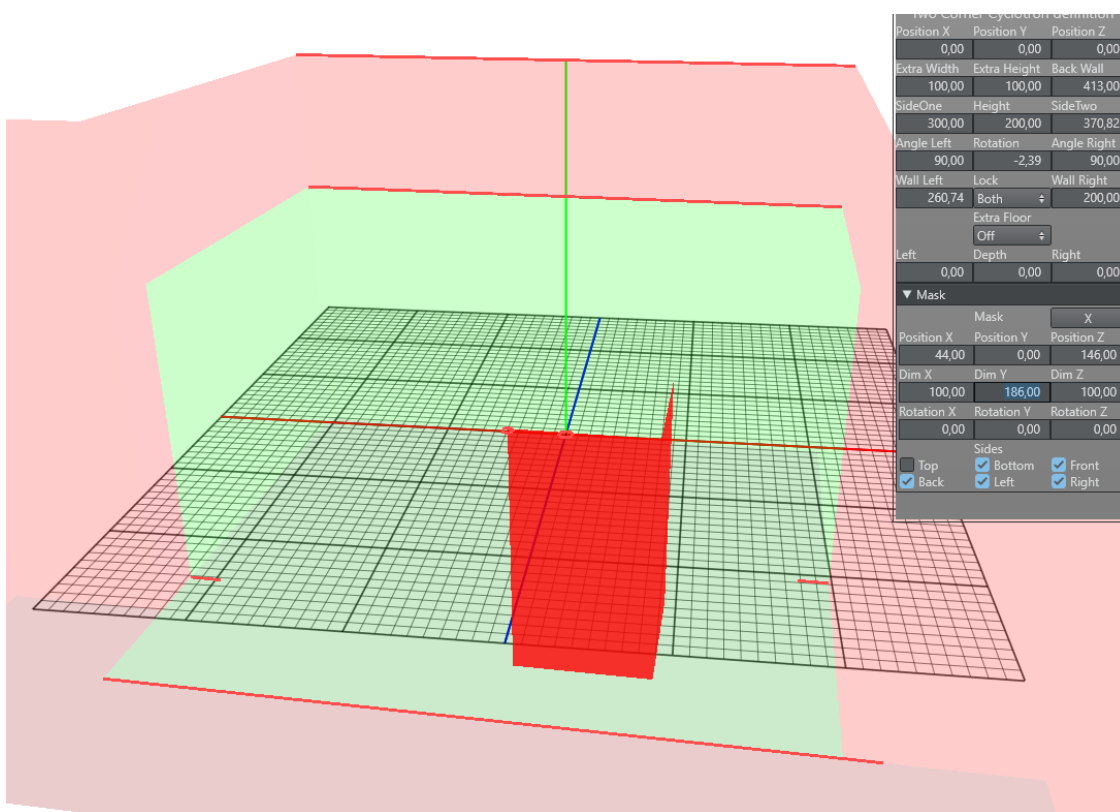


The Two Corner Cyc is defined by the length of SideOne and SideTwo, two opening angles (AngleLeft and AngleRight). It is possible to define the green area of the sides shorter than the side itself. This can be done with the parameters Wall Left and Wall Right. It is also possible to define an extra floor area. This extra floor is added in front of the green walls and extends the shape of the floor.



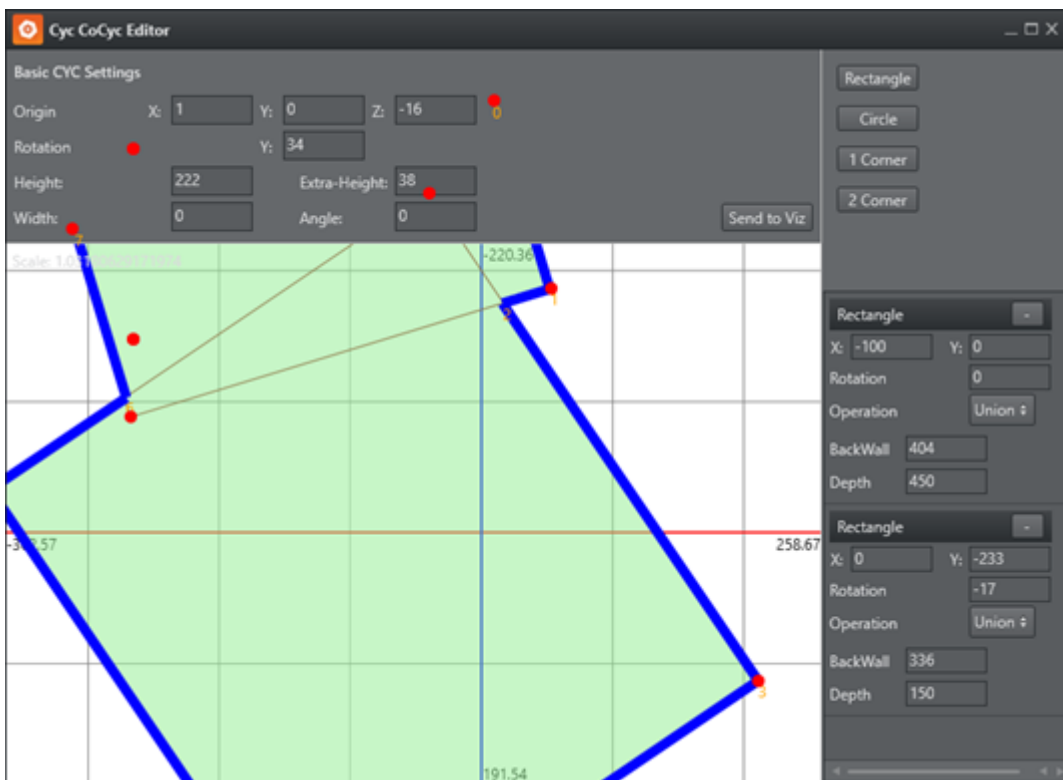
Masks

For any CYC model, any amount of Masks can be added. Masks are elements which will be 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.



The second option to define a Cyc/CoCyc is described below. Important to know is, that as soon as the user selects the old method, the new definition is deleted and vice versa.

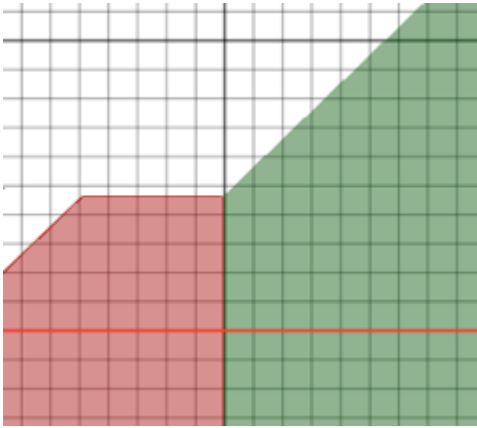
In Tracking Hub and the Studio Manager, a Cyc is defined by basic shapes, such as rectangles and circles, which can be added to a Shape List. All the shapes in the list are united by a Boolean operation which results in a segment list. The segment list is displayed in the segment area. From all these segments, a continuous subgroup can be selected in the editor to define if the segment is part of the green/blue wall or not. Every segment not selected (blue) is part of the floor. Every segment which is selected (green) is part of the Cyc wall. Based on this information, the Studio Manager is able to calculate a mask object which is then sent to Viz Engine.



The Cyc Editor consists of the following Areas

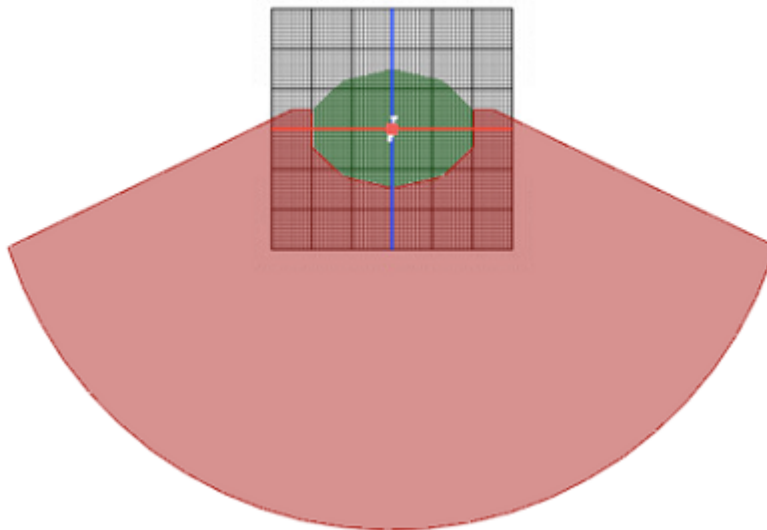
- **Origin:** Defines the X,Y,Z offset of the generated cyc with respect to the origin of the Viz Engine coordinate system.
- **Rotation:** Defines the X,Y,Z rotation offset of the generated cyc with respect to the Viz Engine coordinate system.
- **Height:** Defines the height of the green wall.
- **Extra Height:** Defines the height of the studio ceiling.
- **Angle:** See next section.
- **Width:** See next section.

4.6.7 CyC Editor Angle and Width

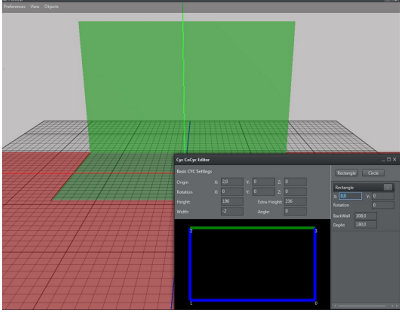
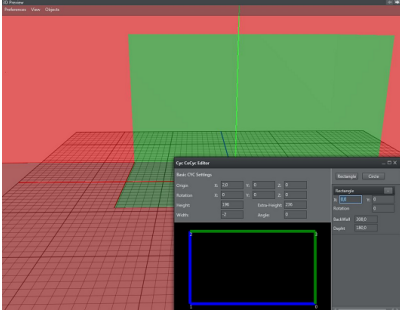
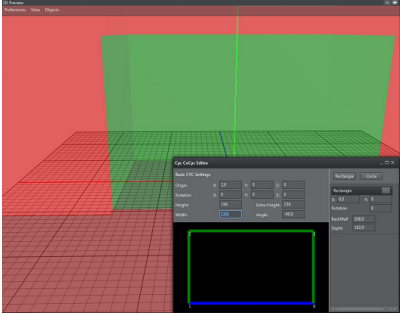
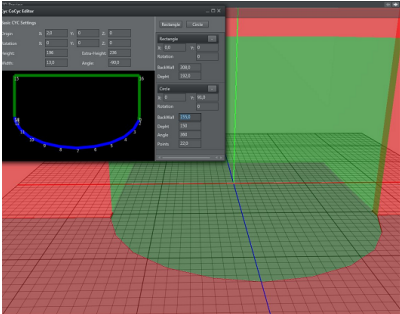


The CYC generation algorithm works as follows:

1. Find a continuous group of wall segments and mark the start and end segment.
2. Calculate the normal of the last segment which is not part of the wall group.
3. Calculate the direction of the first segment, which is a wall.
4. Extrude the CoCyc for Width cm in the calculated normal direction.
5. Continue the extrude in direction of the first wall segment plus Angle.
6. Do the same with the end segment of the continuous group.
7. Close the CoCyc with a big surrounding circular object.



4.6.8 Examples

Example	Description
	A simple back wall studio with one green wall.
	A corner studio with rectangular floor.
	A studio with three walls and rectangular floor.
	A studio with circular end floor.

4.6.9 Shape Area

In the shape area, basic shapes can be added to the shape list. Every list element shows a dash (-) in the Header line. Clicking on the dash (-) removes the shape from the list. Every shape can have its own origin and rotation about the Y axis. Additionally, every shape has its own parameters.

- **Rectangle**
 - **Back Wall:** Sets the width of the rectangle in centimeters.
 - **Depth:** Sets the depth of the rectangle in centimeters.
- **Circle**
 - **Back Wall:** Sets the width of the circle in centimeters.
 - **Depth:** Sets the depth of the circle in centimeters.
 - **Angle:** Defines the angle of the pie the shape creates, if smaller than 360.
 - **Points:** Sets the amount of segments the shape consists of.

4.6.10 Cyc

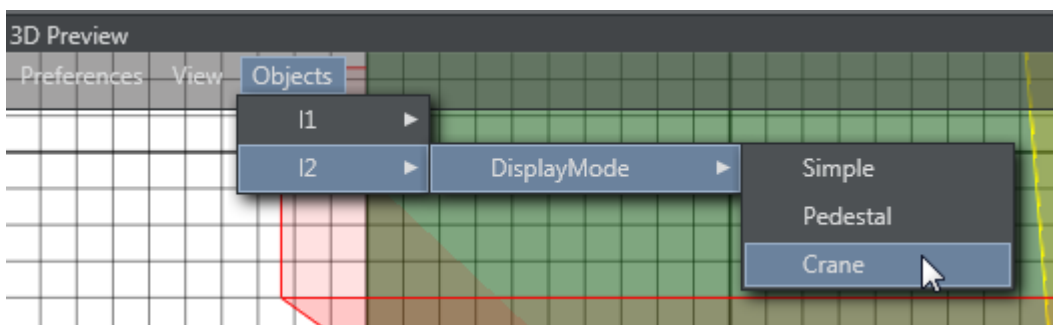
Defines what parts of the Cyc the preview panel renders. The selectable elements are Wall and Floor. All these settings are stored for local use in the user settings folder.

4.6.11 CoCyc

Defines what parts of the CoCyc the preview panel renders.

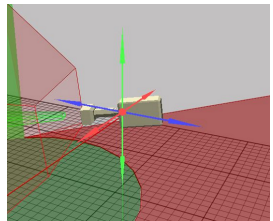
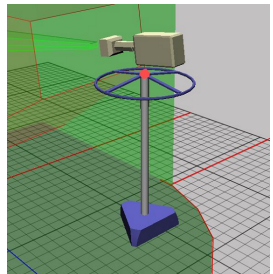
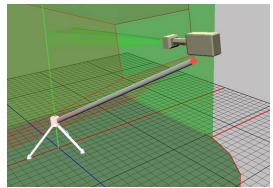
4.6.12 Objects Menu

Every rig present in Tracking Hub has its own display menu with it. These options are handled in the Object menu item. In the first release, Tracking Hub offered only one Rig: Simple Camera. In future releases, more rigs will be added to allow creating Cranes and other geometries.



- **I1 / I2 / I3:** Selects which Rig to use.

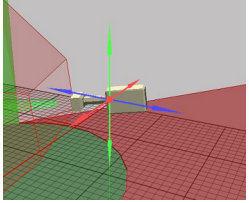
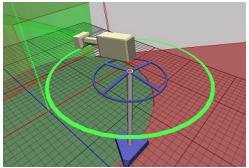
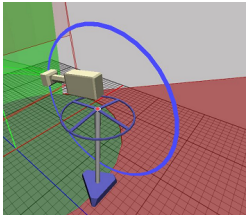
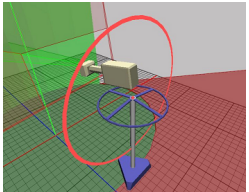
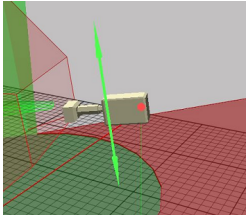
4.6.13 Selectable Display Modes

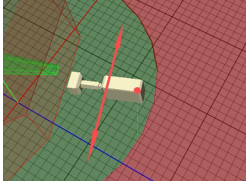
Mode	Description	Example
Simple	<p>The tracked point is displayed as selectable circle in the view.</p> <p>A rendered camera shows the offset settings.</p>	
Pedestal	<p>The rig is displayed as a tracked pedestal.</p> <p>The tracking point is always on top of the pan/tilt head.</p> <p>The rendered camera shows the offsets added to the rig.</p>	
Crane	<p>The rig is displayed as small jib.</p> <p>The origin of the base is the X/Z offset of the rig.</p> <p>The tracked point is a result of offset and height.</p>	
Object	<p>Object is a tracked object which is not a camera.</p> <p>E.g. Motion Analysis objects.</p> <p>No camera is rendered and only the tracked point is displayed.</p>	

4.6.14 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.

Rig Value	Description	Example
Position Offset	Every axis is color coded and only visible if modifiable. <ul style="list-style-type: none"> • Green: Y Axis • Red: X Axis • Blue: Z Axis 	
Pan Offset	Green: Modifies the pan offset of the camera.	
Tilt Offset	Blue: Modifies the tilt offset of the camera.	
Roll Offset	Red: Modifies the roll of the rig.	
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.	

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

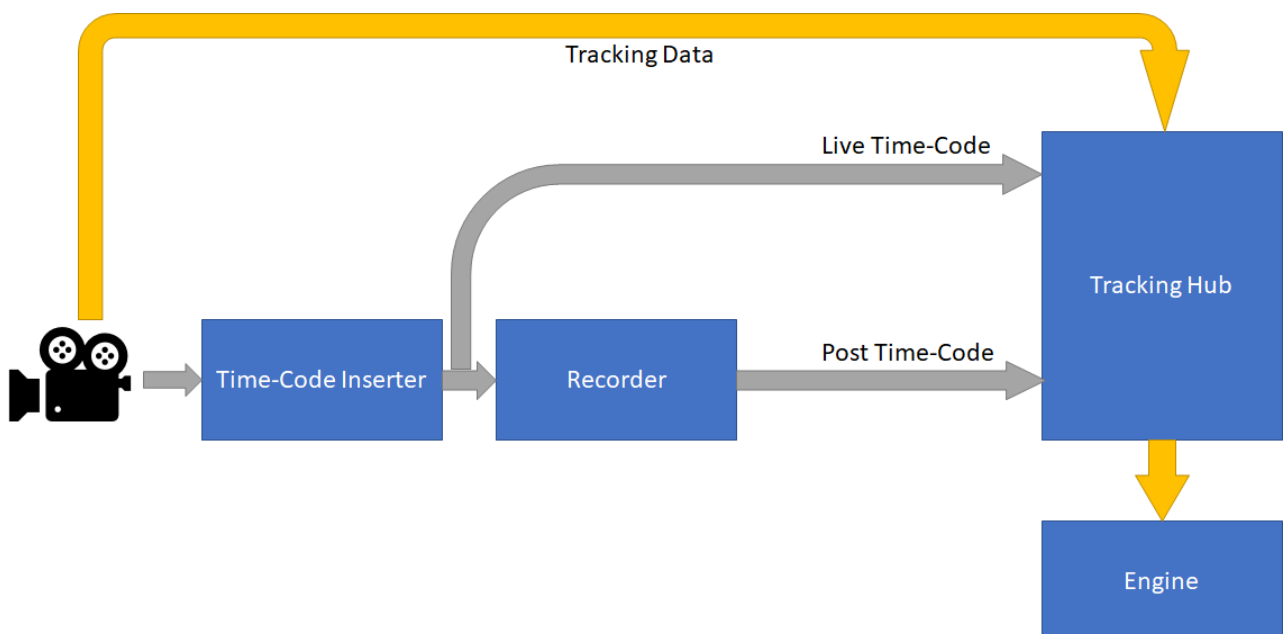
See Also

- [Configure the Studio](#)
- [Configuration Panel](#)
- [Parameter Panel](#)
- [Topology Panel](#)
- [Log Panel](#)

4.7 Post System

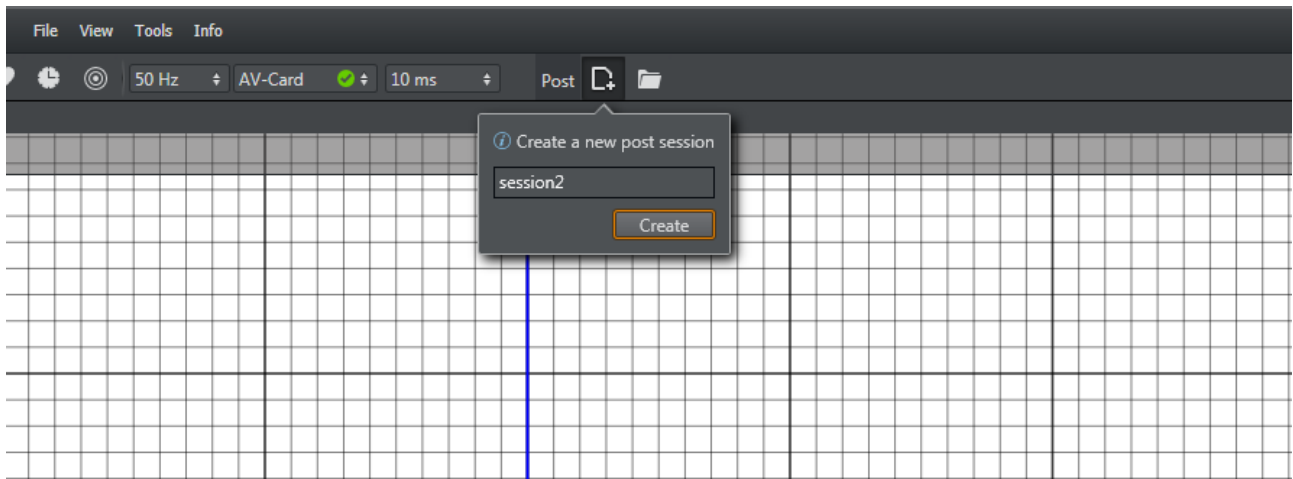
The Tracking Hub Post system lets you record tracking data and play the data back at a later time. To use this feature, every field in the video and every tracking data must be stamped with a timecode. The timecode is read by Plura timecode reader card (PCLPCIe 3G).

i 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 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.



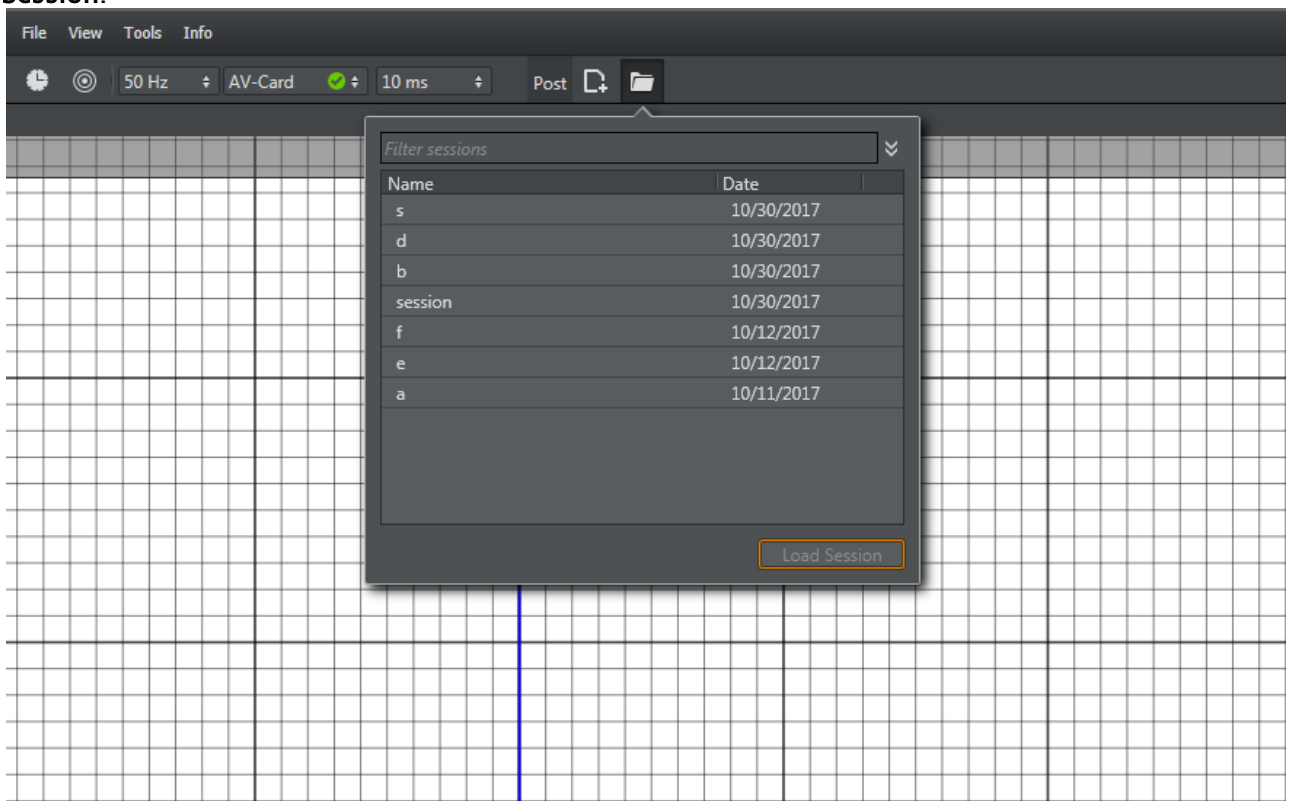
4.7.1 Create a Post Session

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



4.7.2 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**.

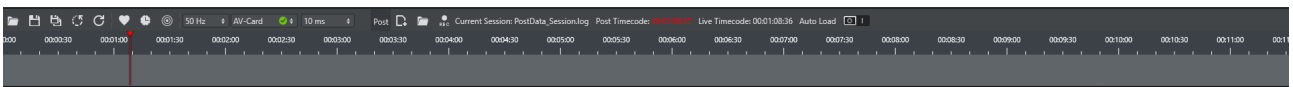


4.7.3 Configure a Post Session

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.

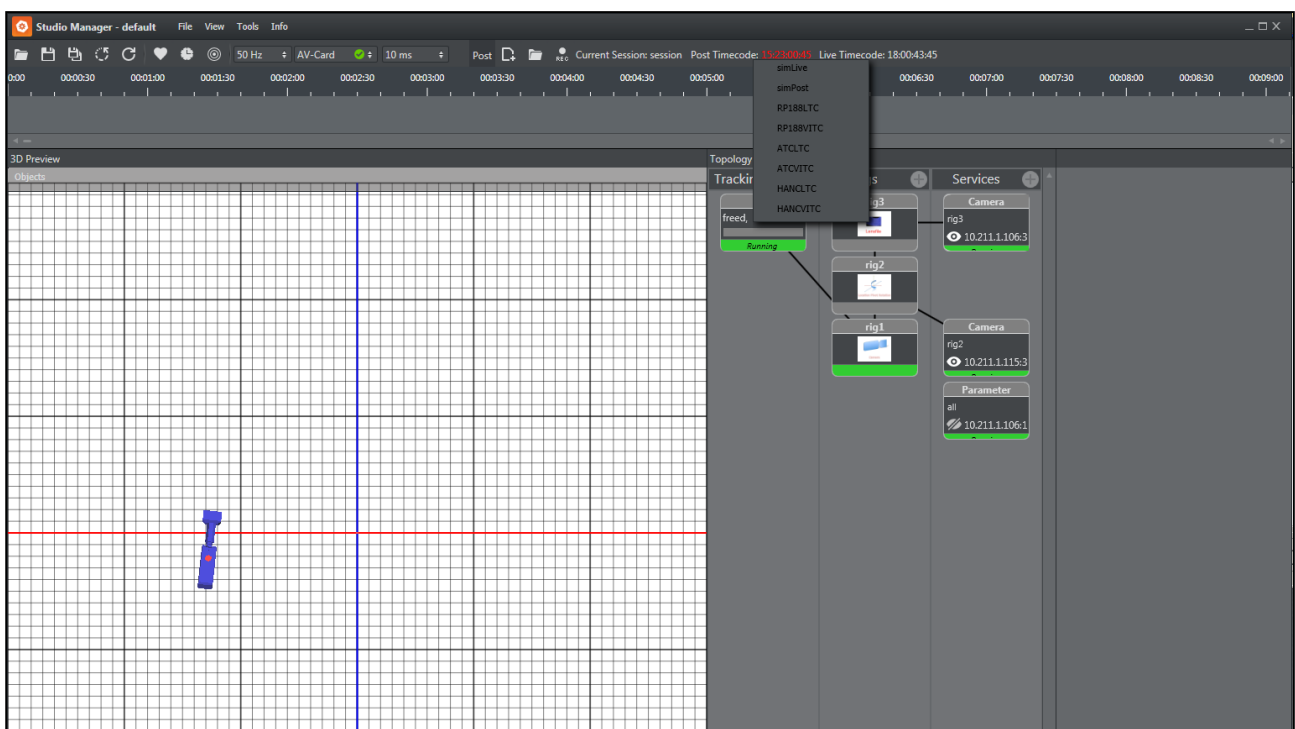
You can automatically load a saved session after Tracking Hub restarts. Activate the **Auto load** switch to enable this feature.



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.

⚠ The must be run in AV-Card mode to recognize the installed Plura timecode card. Otherwise, the timecode sources are not displayed.

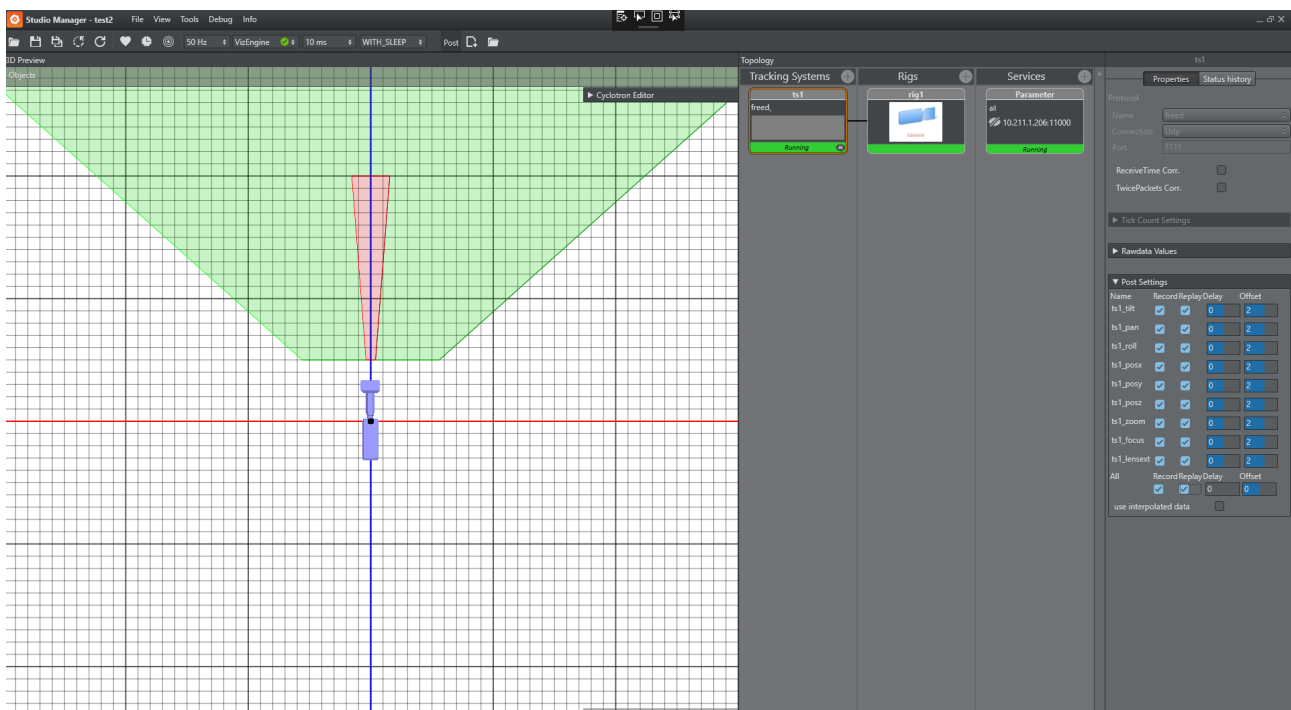
SimLive and **SimPost** are used to record tracking data without timecode reader card. This is for analysis purposes only and should never be used for production recording!



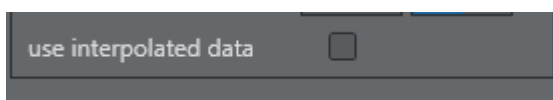
4.7.4 Selecting the Recording and Replay Sources

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.

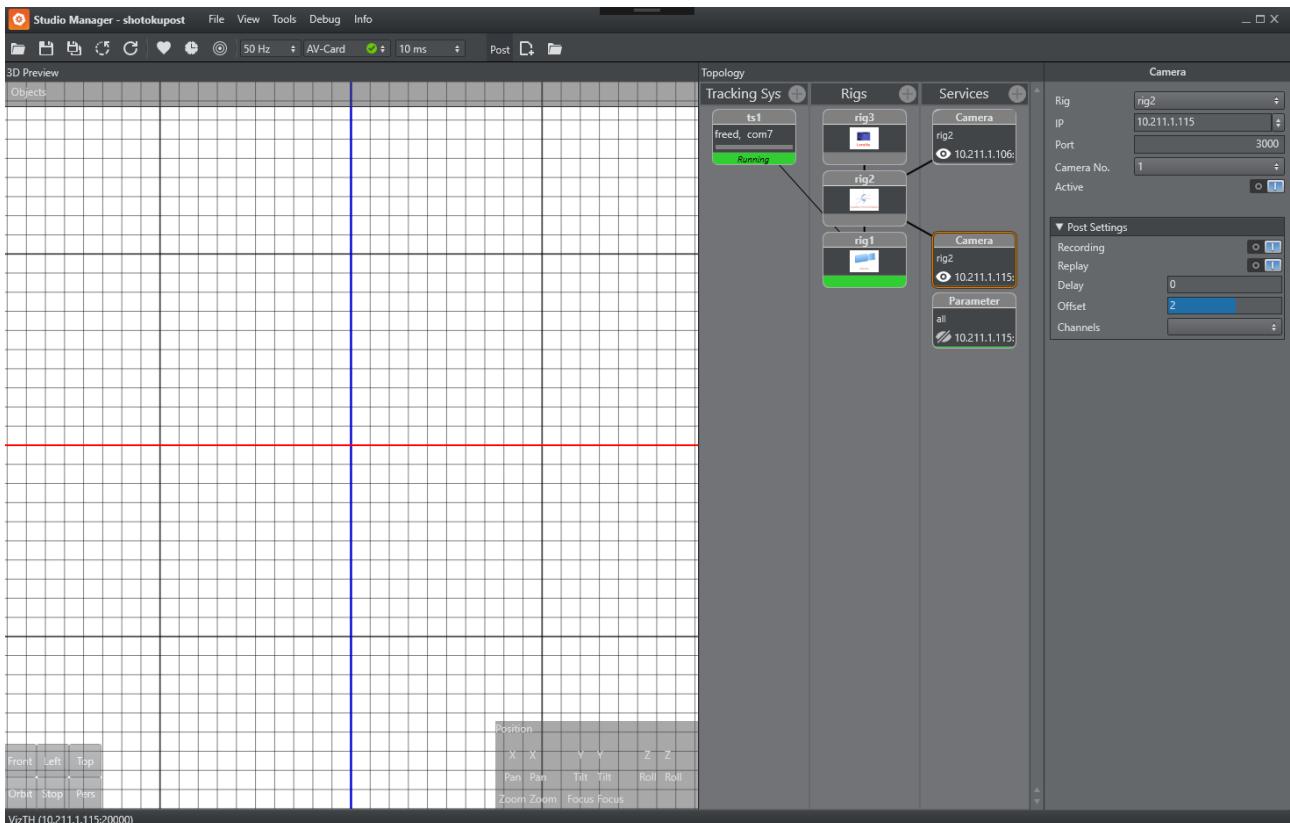
The following screenshot shows the parameter selection for a tracking system. For every parameter, recording and replay can be checked in a check box. An individual delay can be given for every parameter. Replay for a parameter is activated when replay is checked and the post timecode differs by more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

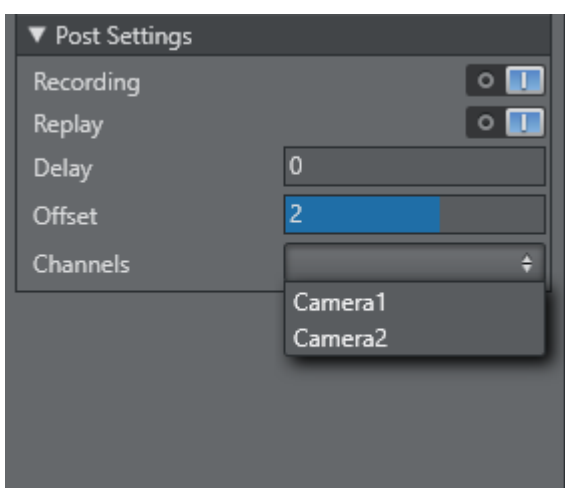


You can activate recording and replay of the camera matrix in the **Camera Service** settings. There is also an option for the camera delay.



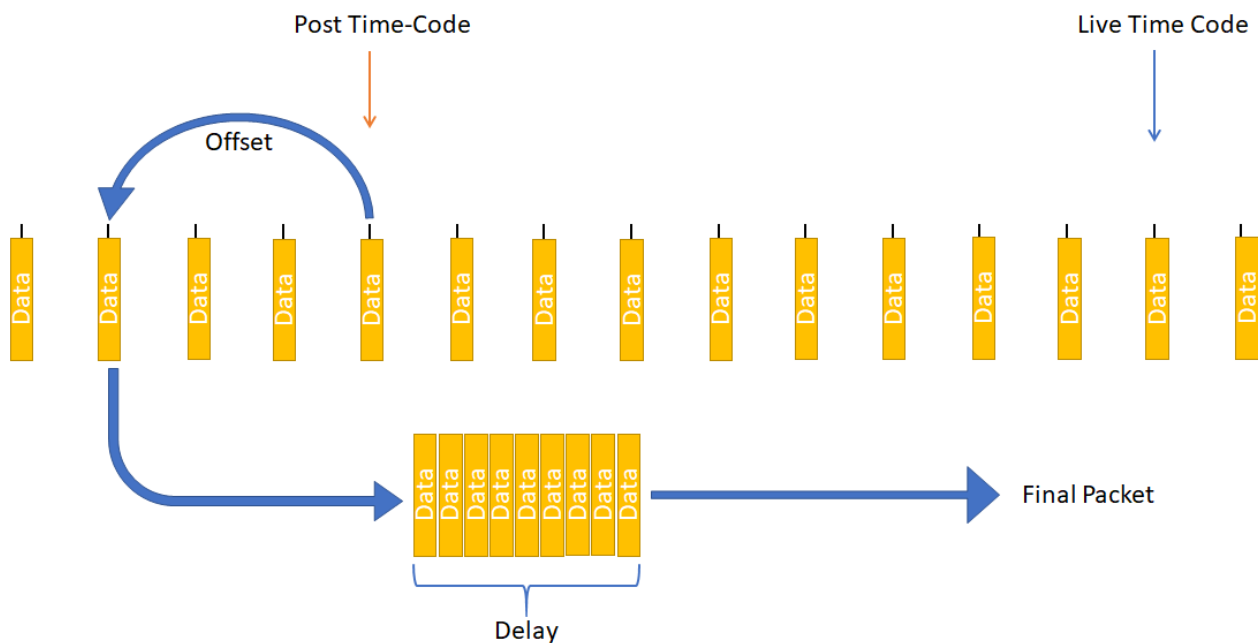
4.7.5 Camera Channels

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.



4.7.6 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.



4.7.7 Post Data Storage

The post data is written to file immediately after being received. Tracking Hub contains about four hours of data in memory.

- 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"
```

5 Studio Manager Configuration

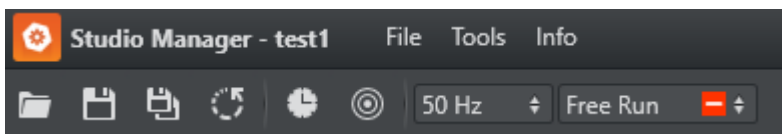
This section details how to configure the Studio Manager. Make sure to configure the Studio first, then the Topology.

This section contains the following topics:

- [Configure the Studio](#)
 - [Configure Topology](#)
 - [Tracked Cameras and Viz Engine](#)
 - [Router Control](#)
 - [Backup Configuration](#)
 - [Use of Templates](#)
-

5.1 Configure The Studio

This section details how to configure Virtual Studio.



1. Click **Open** to select a previously saved studio configuration, or create a new configuration by clicking **Save As**.
 2. Set the **Frequency** frame rate. Available options are:
 - 50 Hz
 - 60 Hz
 - 59.94 Hz
 3. Set the **Synchronization** base:
 - **Freerun**: 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 Plura PCL-PCI or PCL-PCIe sync cards.
 - **Viz Engine**: Synchronizes with a Viz Engine running on the same computer.
 4. Go to [Configure Topology](#).
-

5.2 Configure Topology

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



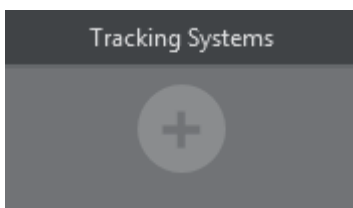
At any time, view the properties of each configuration in the [Configuration Panel](#).


The Topology configuration should be done in this order:

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

5.2.1 Configure a Tracking System

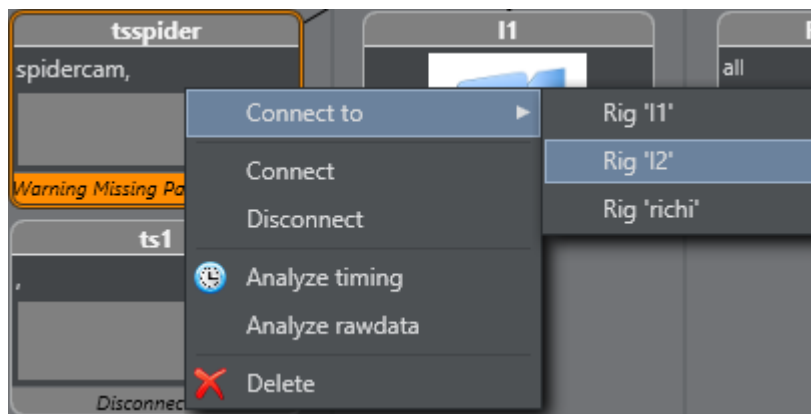
To Add and Configure a Tracking System



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

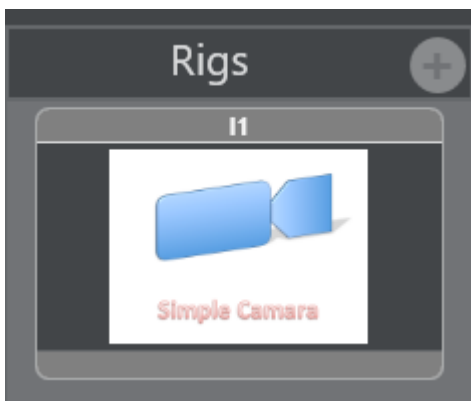


The Tracking Hub now connects to the selected Tracking System. Observe the colors on the bottom of the symbol:

- **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.

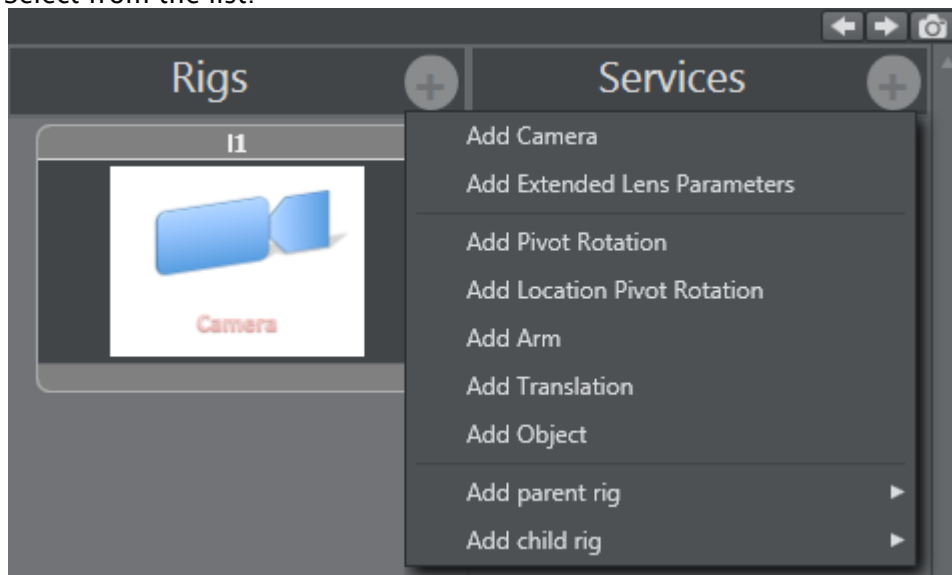
6. Go to Configure a Rig.

5.2.2 Configure a Rig



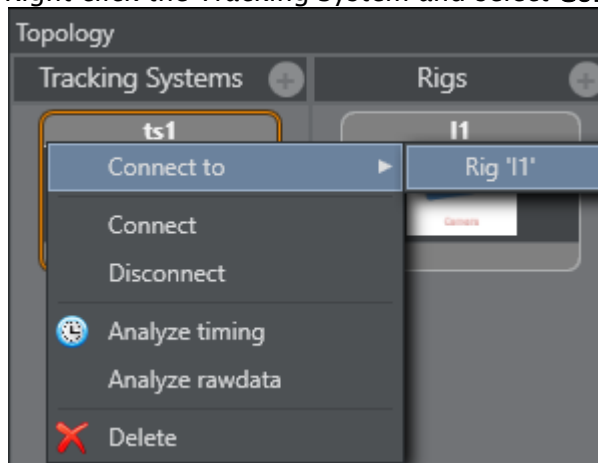
1. Click .

2. Select from the list:

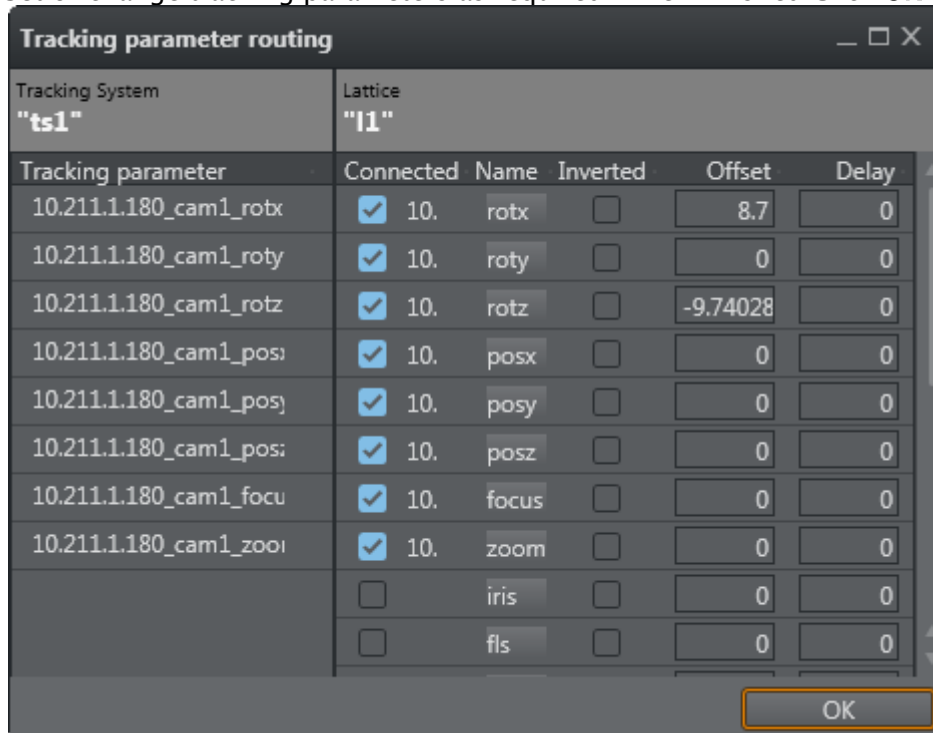


- Add simple camera:

3. Right-click the Tracking System and select **Connect to > Rig** (Lattice).

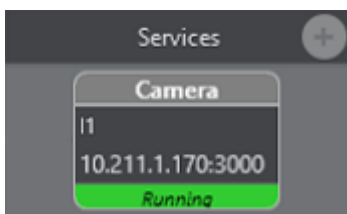


4. Set or change tracking parameters as required. When finished Click **OK**.

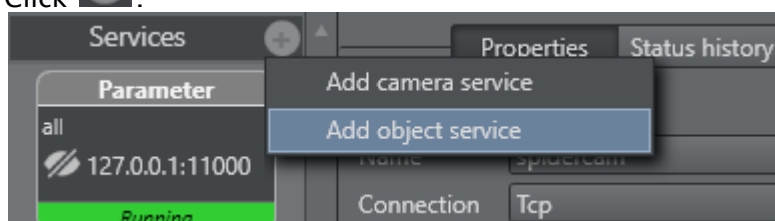


5. Go to Configure a Service.

5.2.3 Configure a Service

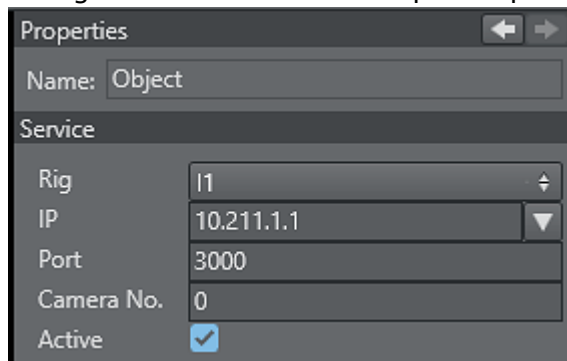


1. Click .



2. Select a Service:
- **Add camera service**
 - **Add object service**
- Click on the service icon in the Services panel.

- Configure the service in the Properties panel.



- **Rig:** Type the name of the rig whose data should send 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).
 - **Cam.Number:** Same as Viz Engine is expecting.
 - **Mode:** Not in use (default = 0).
 - **Active:** When checked the service is active, unchecked no data is sent to the Viz Engine.
- Click **Start** in the [Configure the Studio](#).
 - Click **Save** in the [Configure the Studio](#).

5.2.4 Set a Delay

Set a delay (ms) before a tracking parameter is sent to the rig. Insert a delay, in frames, for tracking delay adjustment.

The screenshot shows the 'Tracking parameter routing' window. It displays a table with tracking parameters and their routing settings.

Tracking System	Lattice	Tracking parameter	Connected	Name	Inverted	Offset	Delay
"ts1"	"l1"	10.211.1.180_cam1_rotx	<input checked="" type="checkbox"/>	10. rotx	<input type="checkbox"/>	8.7	0
		10.211.1.180_cam1_roty	<input checked="" type="checkbox"/>	10. roty	<input type="checkbox"/>	0	0

Insert this value in the first box. If you keep the second box empty, the value is set for all axes. If you choose an axis in the second box, the value is set for this axis only. Press the button after inserting.

5.2.5 Set an Offset

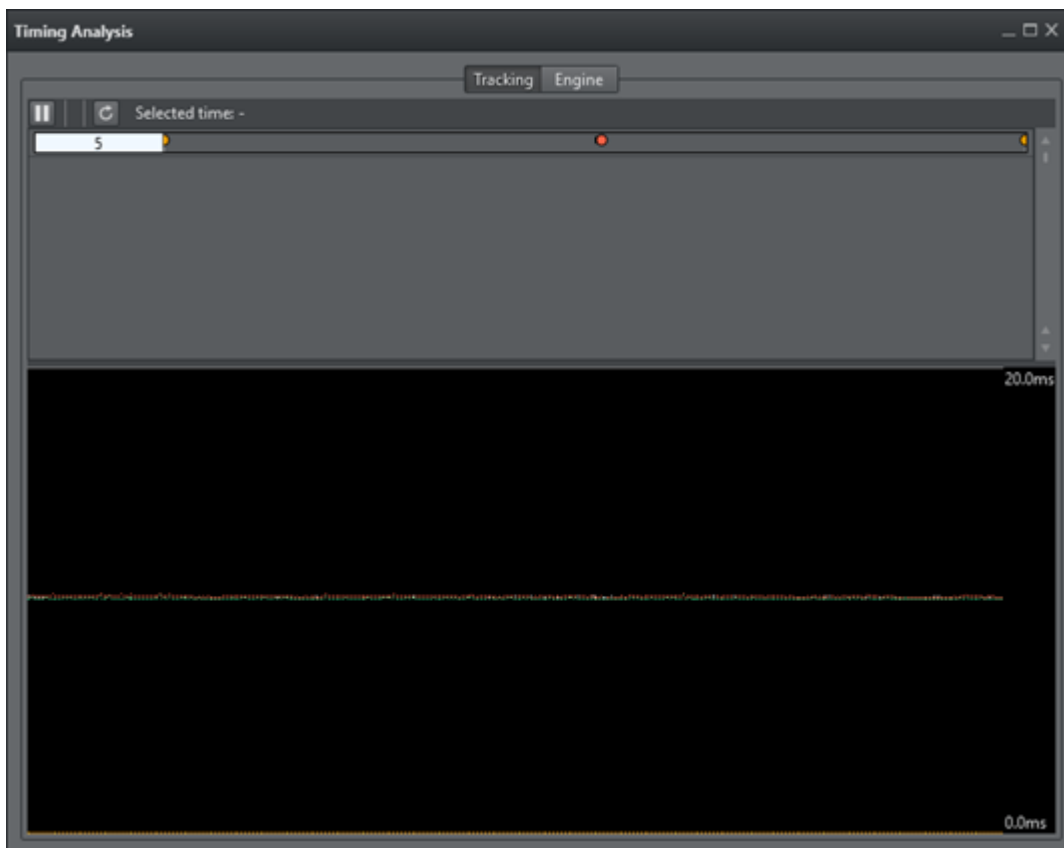
An offset (cm) can be added to a tracking parameter value, when passed to a rig, to correct positional rotation of a camera and, or an object

Tracking parameter routing						
Tracking System	Lattice					
"ts1"	"11"					
Tracking parameter	Connected	Name	Inverted	Offset	Delay	
10.211.1.180_cam1_rotx	<input checked="" type="checkbox"/>	10. rotx	<input type="checkbox"/>	8.7	0	
10.211.1.180_cam1_rotx	<input checked="" type="checkbox"/>	10. rotx	<input type="checkbox"/>	0	0	

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

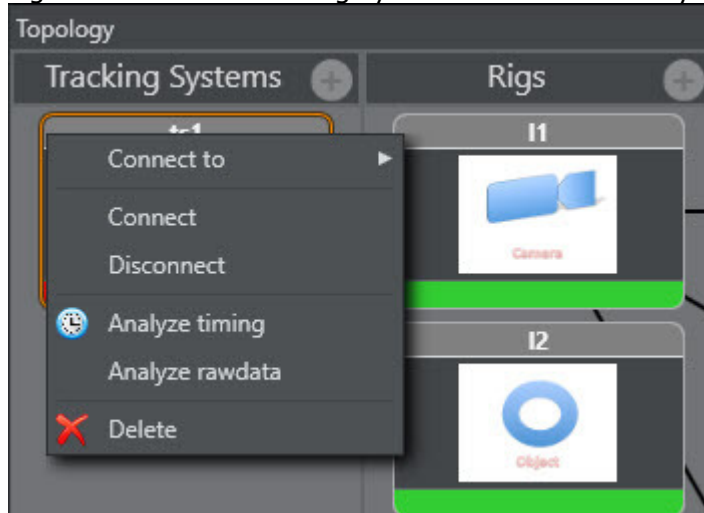
5.2.6 Analyze Time

Analyze Time is a visual representation of the tracking.



To Analyze Tracking System Time

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

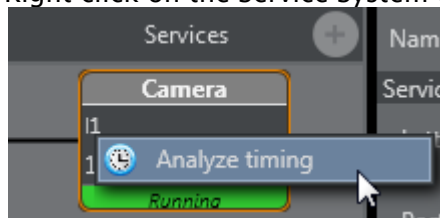


This adds a new element to the **Services** panel, called **TrackingTiming**.

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**.



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**.

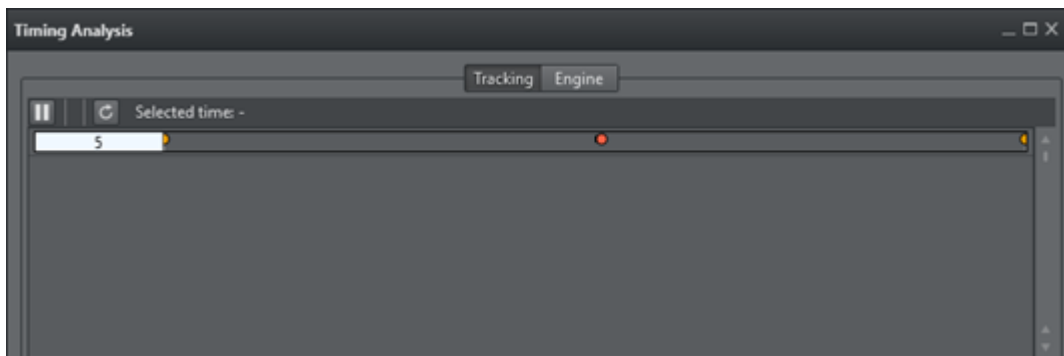
To View Timing Information

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.
 - **Green line:** Shows the time when the communication part of the Tracking Hub starts to read from the data pool.

- **Red Dot:** Indicates data sent to a Viz Engine has stopped.

Upper Section

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



These timestamps are shown:

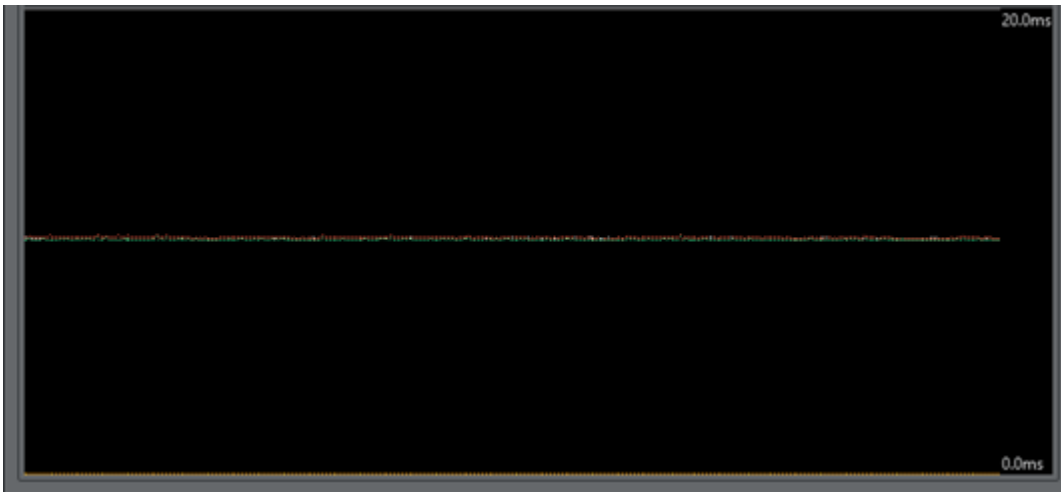
- **Sync:**
 - **Start RX:** Start time, when data is received from the tracking system.
 - **End RX:** End time, when data stops from the tracking system.
 - **Start Ins:** Start time, when Tracking Hub inserts received data into the parameter pool.
 - **End Ins:** End time, when Tracking Hub has finalized 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.



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.

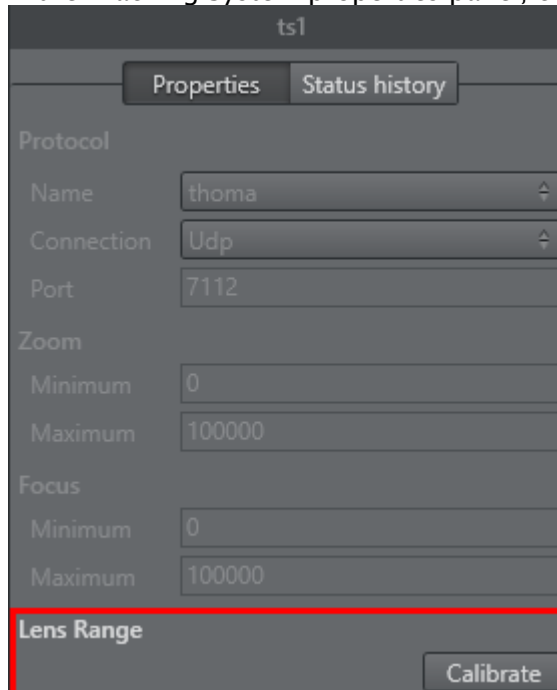
This doesn't matter so much, because in normal production you need more the one field tracking delay.

5.2.7 Lens Range Calibration

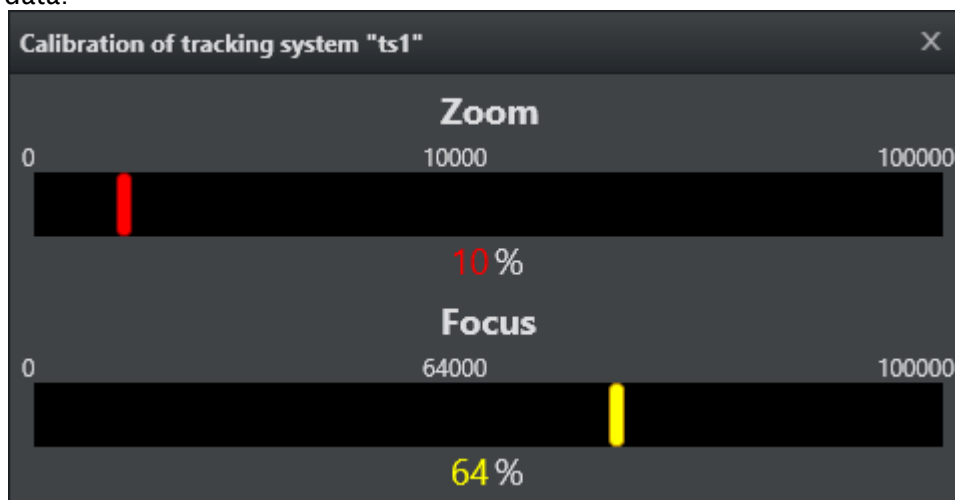
This section details how to calibrate the Lens Range.

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:



To check if the calculation is correct, the corresponding Min/Max values, of the Lens

encoder, are shown in the **Zoom** and **Focus** fields.

The screenshot shows the configuration window for a Tracking System (ts1). The 'Properties' tab is active. The configuration includes:

- Protocol:** Name: thoma, Connection: Udp, Port: 7112
- Zoom:** Minimum: 0, Maximum: 100000
- Focus:** Minimum: 0, Maximum: 100000
- Lens Range:** (Section header)
- Calibrate:** (Button)

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 `lensrange` 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.

5.2.8 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.

Tracking System	Rig				
"ts1"	"R13"				
Tracking parameter	Connected	Name	Inverted	Offset	Delay
ts1_tilt	ts1_tilt x	Tilt	<input type="checkbox"/>	0	0
ts1_pan	ts1_pan x	Pan	<input type="checkbox"/>	0	0
ts1_zoom	ts1_zoom x	Roll	<input checked="" type="checkbox"/>	0	0
ts1_focus	ts1_focus x	PosX	<input checked="" type="checkbox"/>	0	0
ts1_spare5	x	PosY	<input type="checkbox"/>	0	0
ts1_spare6	x	PosZ	<input type="checkbox"/>	0	0
ts1_spare7	x	Zoom	<input type="checkbox"/>	0	0
ts1_spare8	x	Focus	<input type="checkbox"/>	0	0
	x	CenterX	<input type="checkbox"/>	0	0
	x	CenterY	<input type="checkbox"/>	0	0
	x	FrontLensShift	<input type="checkbox"/>	0	0
	x	HeightLensShift	<input type="checkbox"/>	0	0
	x	RightLensShift	<input type="checkbox"/>	0	0

OK

- 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

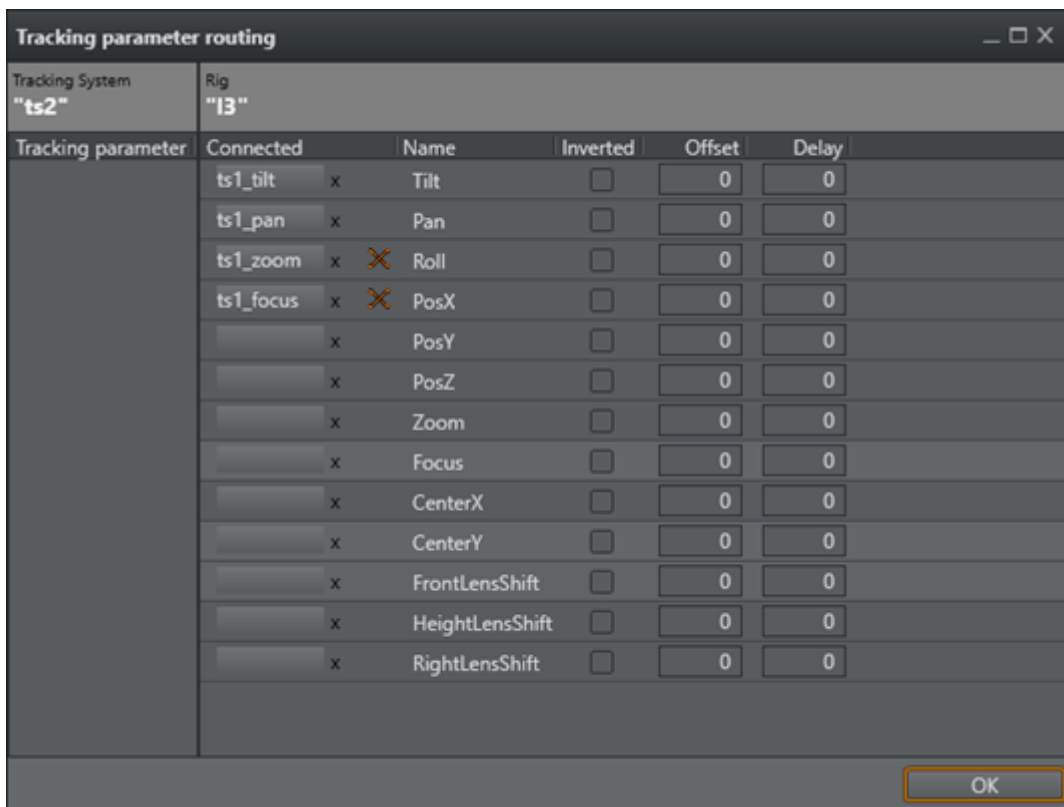
cross-over connection and not a straight one.

Tracking System "ts1"	Rig "I3"				
Tracking parameter	Connected	Name	Inverted	Offset	Delay
ts1_tilt	ts1_tilt x	Tilt	<input type="checkbox"/>	0	0
ts1_pan	ts1_pan x	Pan	<input type="checkbox"/>	0	0
ts1_zoom	ts1_zoom x	Roll	<input type="checkbox"/>	0	0
ts1_focus	ts1_focus x	PosX	<input type="checkbox"/>	0	0
ts1_spare5	ts1_spare5	PosY	<input type="checkbox"/>	0	0
ts1_spare6	x	PosZ	<input type="checkbox"/>	0	0
ts1_spare7	x	Zoom	<input type="checkbox"/>	0	0
ts1_spare8	x	Focus	<input type="checkbox"/>	0	0
	x	CenterX	<input type="checkbox"/>	0	0
	x	CenterY	<input type="checkbox"/>	0	0
	x	FrontLensShift	<input type="checkbox"/>	0	0
	x	HeightLensShift	<input type="checkbox"/>	0	0
	x	RightLensShift	<input type="checkbox"/>	0	0

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.



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](#).

5.3 Tracked Cameras And Viz Engine

Viz Engine needs to be correctly configured to receive tracking data from the Tracking Hub. For correct configuration, the Viz Engine configuration file must be edited manually following these instructions:

Warning: Create a backup copy of the configuration file before changing it.

1. Locate the Viz Engine configuration file. This is normally C:\ProgramData\vizrt\viz3\VIZ-MYHOSTNAME-0-0.cfg where MYHOSTNAME is the hostname of the PC the Engine is running on.
2. Open the Engine configuration file in a text editor like Notepad, the configuration file is a text-file and must be saved after editing as a text-file.
3. Locate the **Camera** section in the configuration file and make edits as follows:
 - The flag trackinghub_port = 3000 must be set to a valid and accessible port.
 - The flag use_trackinghub = 1 must be set to 1.

When this is done, the Engine no longer waits for tracking data from the discontinued Viz IO. Instead, the Tracking Hub port is opened and the Engine sends live signs to the Studio Manager and appears in all dialogs where an Engine can be selected.

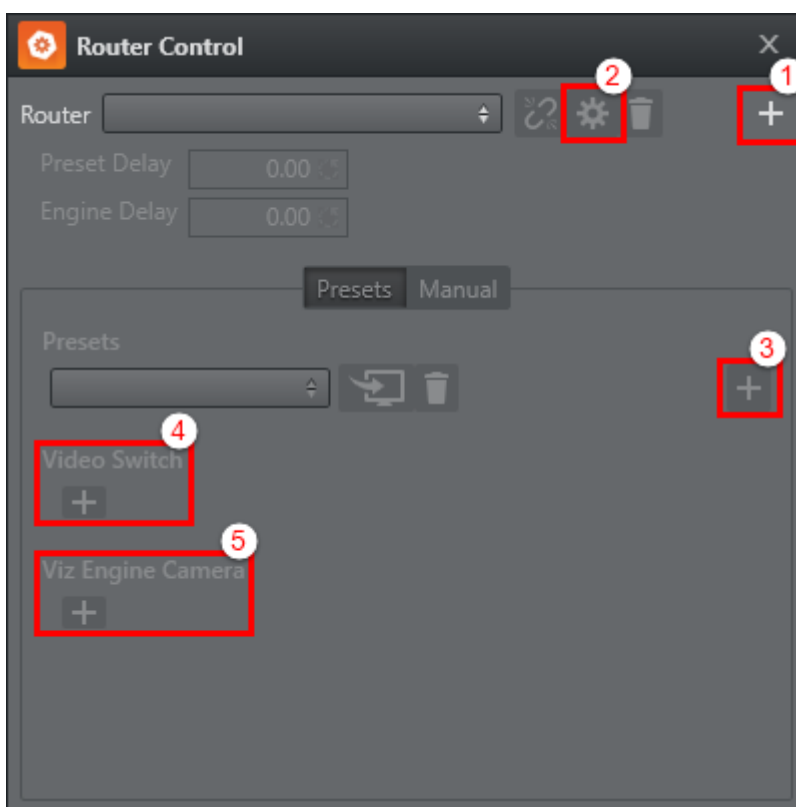
If more than one network adapter is present on the Engine, the user can bind the live signs and tracing data to a specific network interface. The flag `trackinghub_adapter = <IPADDRESS>` (for example: `10.211.1.65`) must be configured to the adapter IP that should be used. If the flag is empty, the first adapter is used.

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

5.4 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.

5.4.1 To Add a Router Control Preset

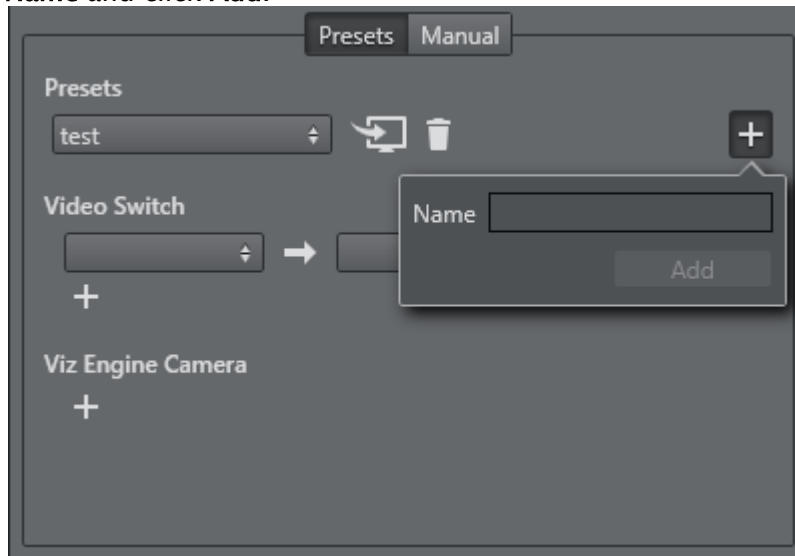


1. Open the **Tools** menu and select **Router Control**.
2. In the dialogue window that opens, select a **Model** and enter a **Name** for the router, then click **Add**. The selected model defines the communication protocol which is used, and is not vendor dependent. The available protocols are:

- Black Magic (serial)
 - Nevia MRP (network)
 - Nevia NCB (serial)
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).

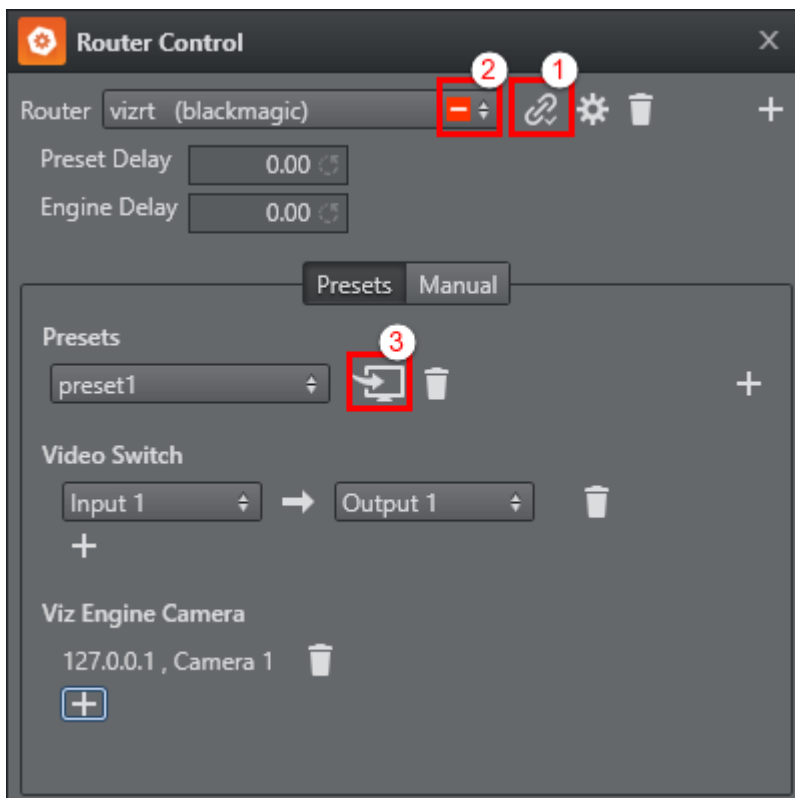
- **Connection:** Select the connection protocol. Available options are **Serial**, **UDP** and **TCP**. For serial connections, the **Com Port** needs to be set in all cases. The other parameters may be left to their **default** settings, or specified based on the requirements of the specific hardware connected. Please refer to the vendor documentation for connection details.
 - **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.
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**.



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:

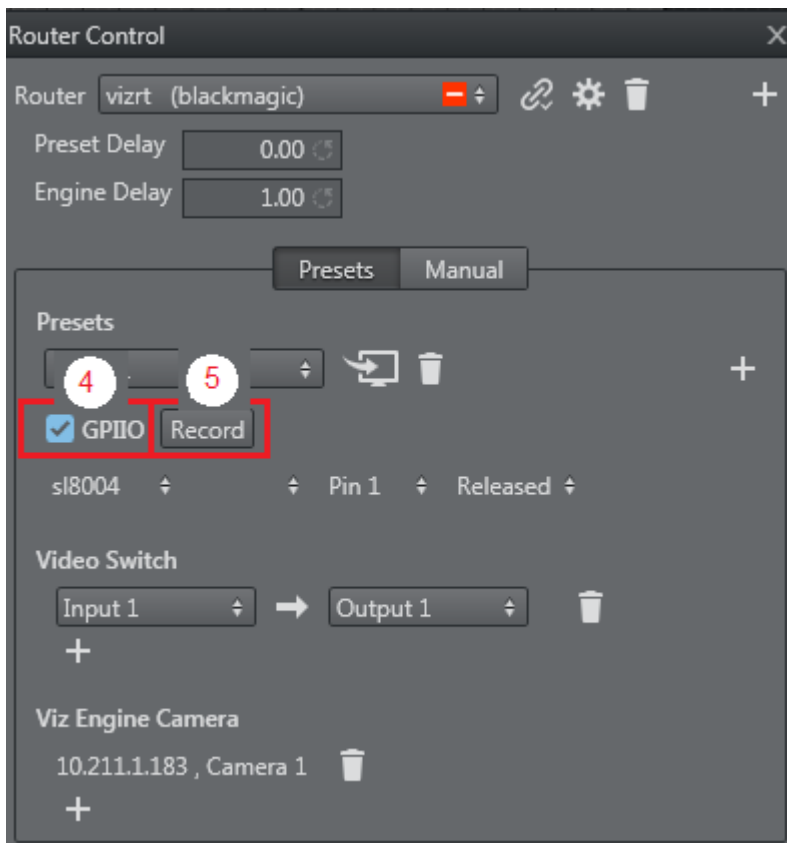


Click the **Connect** button (1) to connect to the configured router. If there is a connection problem,

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).

5.4.2 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.



To configure the device, you have two options:

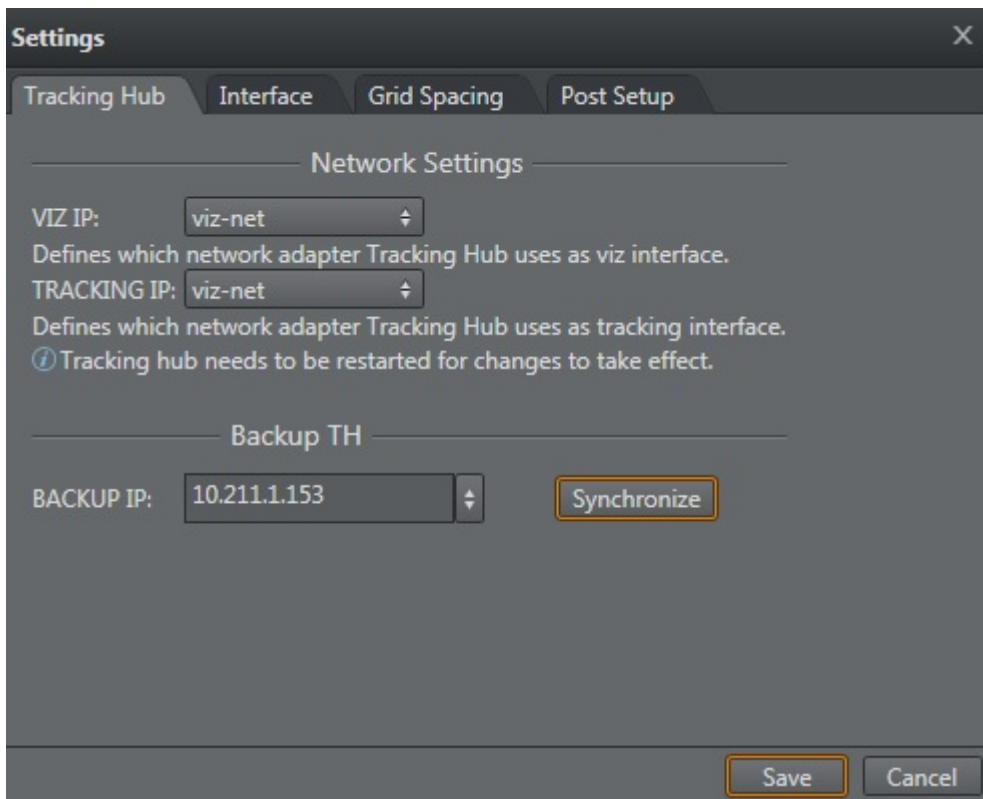
- **Manual:** Select the device, the port (not available for all devices), and the pin. In addition, you can select if the switching should react on pressing or releasing the button.
- **Automatic:** Click **Record** (5) and then select the button on the GPI IO device. The button is recorded and connected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

5.5 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 the 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:

TH1 has IP: 10.211.1.153
Backup TH: 10.211.1.183

TH2 has IP: 10.211.1.183
Backup TH: 10.211.1.153

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

Important!

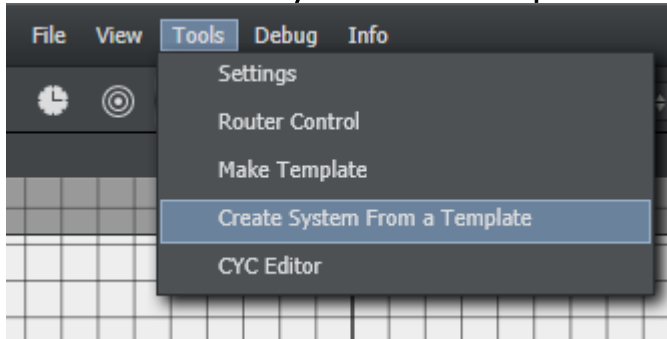
- The `sendDelay` value must be set manually on each Tracking Hub, to different values (e.g., 8 and 12 ms). This value is not synchronized.
- When using serial interfaces, they might be different on these PCs. Always inspect the selected COM ports after synchronization, and, if necessary, modify them.

5.6 Use Of Templates

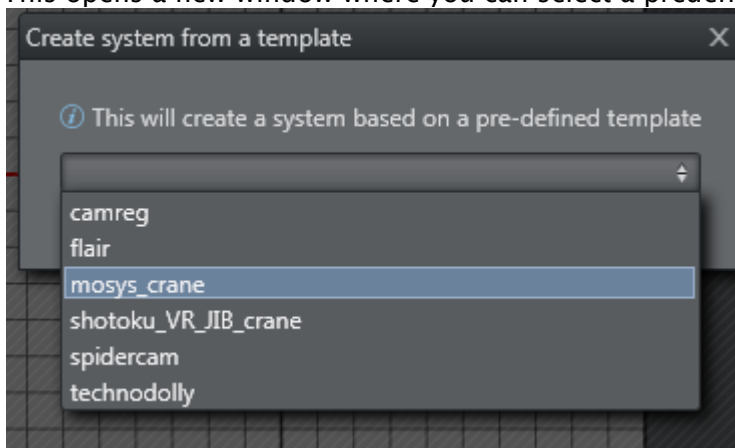
You can use templates for an easy and fast configuration of several studios. There are several pre-made templates available in `C:\ProgramData\vizrt\VizTH\`. You can also create your own templates. However, you must copy them to the Tracking Hub directory on a new installation.

5.6.1 Using Existing Templates

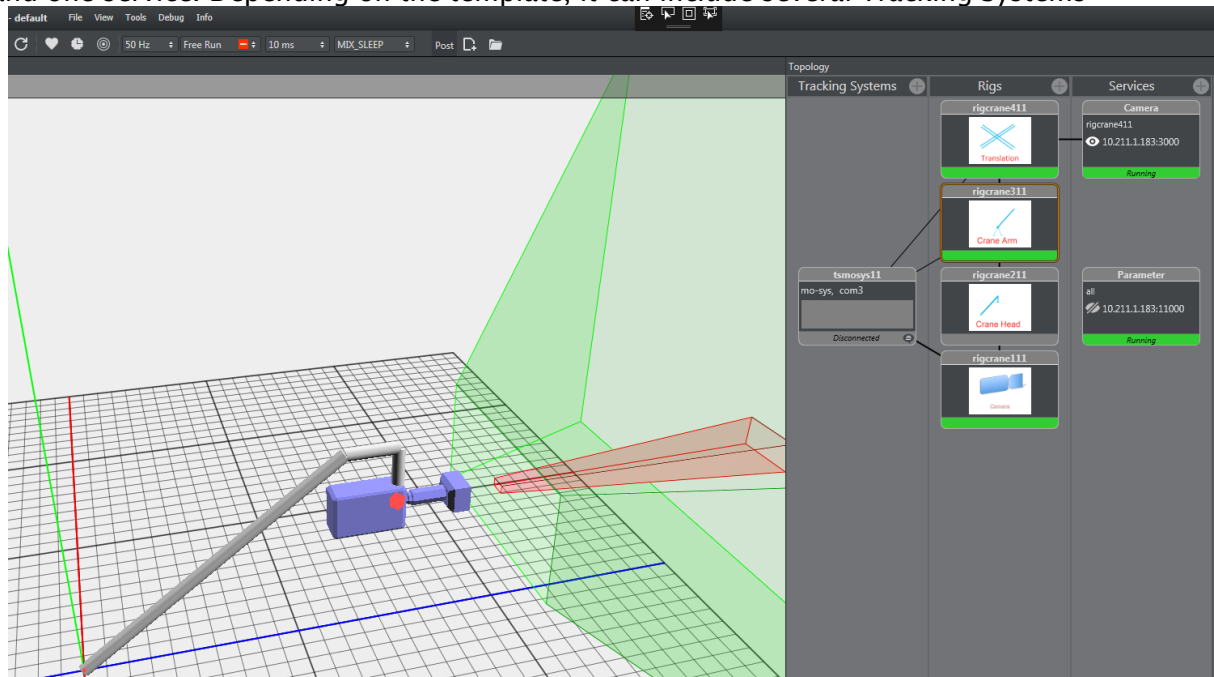
1. Go to **Tools > Create System From a Template**.



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



- 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



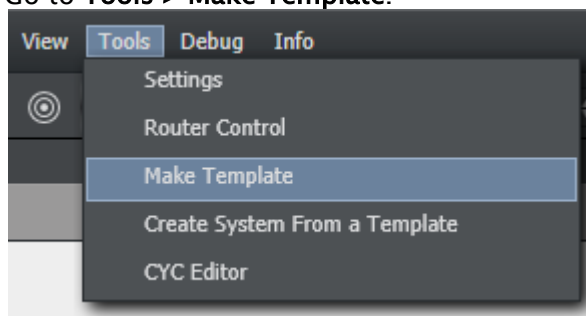
and rigs.

- 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.

5.6.2 Creating New Templates

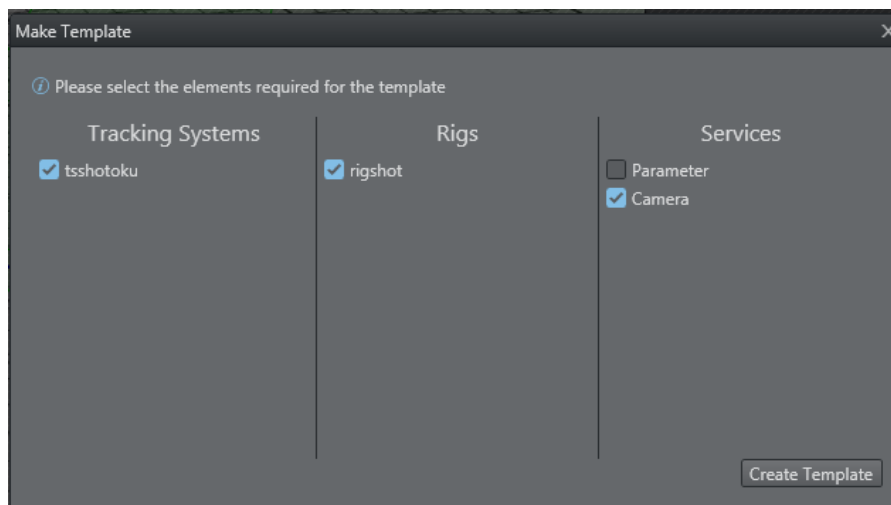
If you want to reuse a complete system, you can create your own template for it.

- 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.

6 Appendices

This section contains the following topics:

- [Words and Terminology](#)
- [Description of the FreeD protocol](#)
- [Motion Analysis Integration](#)

6.1 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 <i>trash matte</i> .
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.

6.2 Description Of The FreeD Protocol

The following shows a complete description of the FreeD protocol.

With the use of the XML Protocol Description, it is much easier and faster to react to new tracking protocols of new manufacturers:

```
<?xml version="1.0" encoding="utf-8"?>
<viz_xml_tracking title="xml_freed">
  <interface type="serial"
    baud ="38400"
    size="8"
    parity="odd"
    stop="1">
  </interface>
  <!--option for type "udp" "tcp" ?
  <!--option for parity "none", "odd", "even", "mark", "space" ?
  <!--option for stop 1, 2 ? <!--option for interface Type "udp" or "tcp" <ip>10.10.
  10.10</ip> <port>6000</port> ?
  <!--option for checksum calculation 1,2,3,4,... we'll use predefined functions> ?
  <!--option for value UINT32 INT32 UINT16 INT16 ?
  <!--option for order bigendian littleendian ?
  <!--option for calc + - * / and one value ?
  <!--do not use hex values, 0x80000 = 524288 -->
  <checksum pkglen="29"
    calculation="1">
  </checksum>
  <extraction count="9">
    <axis name="rotx" start="5" len="3" order="bigendian" value="INT32" calc="/
  32768"></axis>
    <axis name="roty" start="2" len="3" order="bigendian" value="INT32" calc="/
  32768"></axis>
    <axis name="rotz" start="8" len="3" order="bigendian" value="INT32" calc="/
  32768"></axis>
    <axis name="posx" start="11" len="3" order="bigendian" value="INT32" calc="/
  640"></axis>
    <axis name="posy" start="14" len="3" order="bigendian" value="INT32" calc="/
  640"></axis>
    <axis name="posz" start="17" len="3" order="bigendian" value="INT32" calc="/
  640"></axis>
    <axis name="zoom" start="20" len="3" order="bigendian" value="INT32" calc="-
  524288"></axis>

    <axis name="focus" start="23" len="3" order="bigendian" value="INT32" calc="-
  524288"></axis>

    <!--example for not define axis ?
    <axis name="iris" calc="!"></axis>
  </extraction>
</viz_xml_tracking>
```

6.3 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.

6.3.1 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.

6.3.2 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.

6.3.3 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.

6.3.4 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.

6.3.5 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.

6.3.6 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

6.3.7 Motion Capturing

Please see [Topology Panel](#).