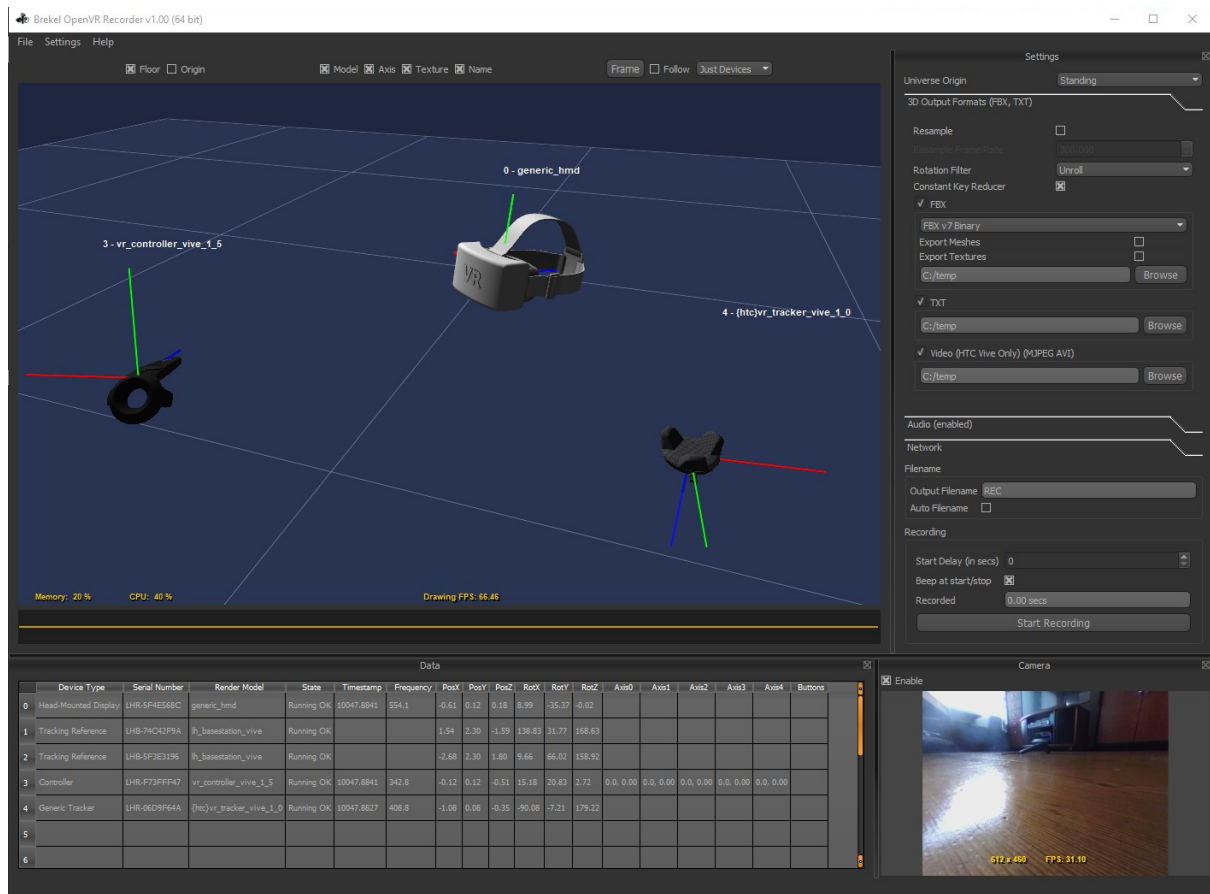


BREKEL

OPENVR

RECORDER

WHAT IS “BREKEL OPENVR RECORDER”



Brekel OpenVR recorder is designed to record tracking data from devices with drivers for OpenVR / SteamVR.

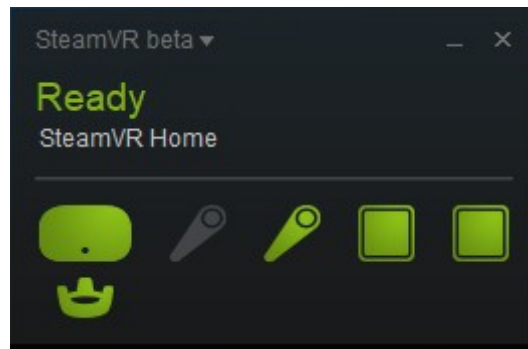
This currently includes controllers, HMDs and trackers from HTC Vive and Oculus Rift VR systems.

The accuracy and high framerates of these VR devices can be useful for use in 3D animation, VFX production, general research or record during play-testing of VR games/experiences.

It is written by Jasper Brekelmans, so in case you're wondering that's what "Brekel" refers to.

"Brekel" is pronounced as "Break-uhl".

REQUIREMENTS



On the hardware side of things you will need an HTC Vive, Oculus Rift or any other piece of hardware that is supported by OpenVR.

On the software side you'll need to install SteamVR, which can be downloaded for free from Steam:

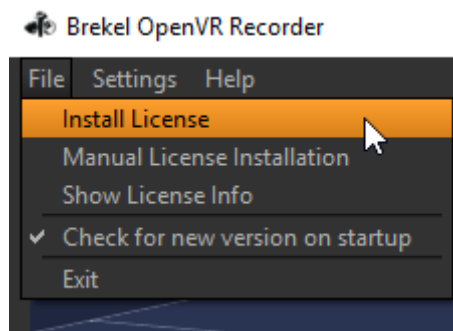
<https://steamcommunity.com/steamvr>

LICENSING

With your purchase you should have received a download link for the retail version of the software and an email with your license code. (Please allow up to 24 hours for processing your request and double check your spam filter)

The retail version will work immediately and with no restrictions even without a valid license, but will need activation within 5 days for continued operation.

You can activate your copy by using the “File > Install License” option from the menu.

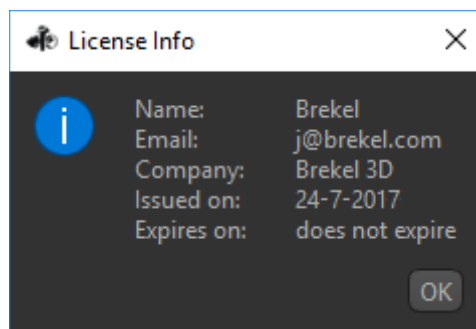


It will ask you to browse for your license file (which was attached to the mail).

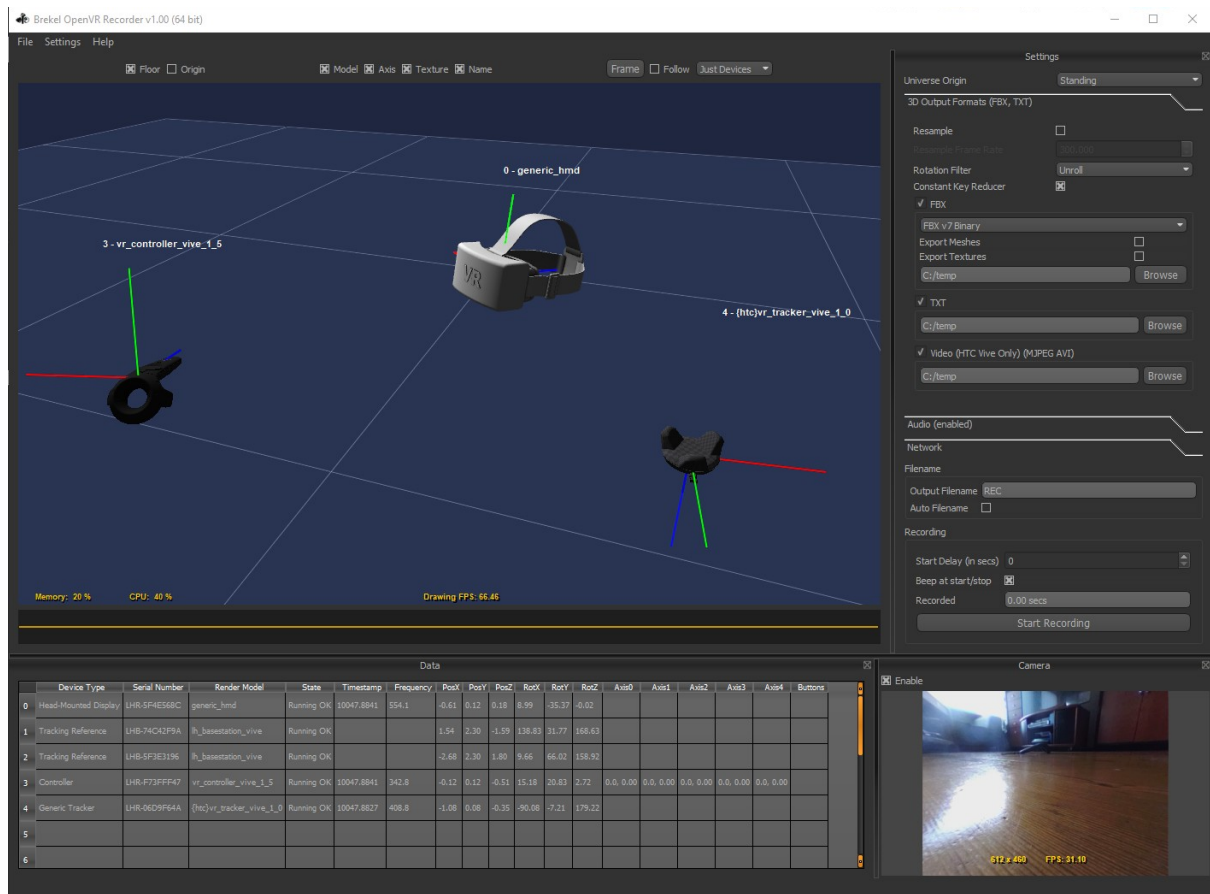
The license will automatically be copied to your “C:\ProgramData” folder and be active upon restart of the application.

If license installation fails (for example due to user permissions) you also manually extract the .zip file and copy the .key file manually to the “C:\ProgramData” folder.

To display the current license info use the “File > Show License Info” option from the menu to display a window like the following:



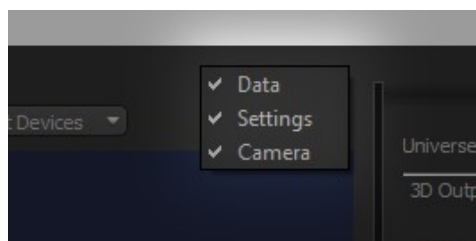
THE MAIN INTERFACE



The interface is divided in 4 main docks:

- Top Left: 3D Viewport
- Bottom Left: data spreadsheet
- Bottom Right: camera view
- Right: Settings

- These docks can be undocked and turned into a floating window by dragging from their title bar.
- Double clicking on the title bar of a floating window will dock it back into the main window.
- The overall GUI state will automatically be saved on exit and reloaded on start.
- Right Clicking on the title bar of the main window (next to Help) will popup a list that allows showing hidden windows again.

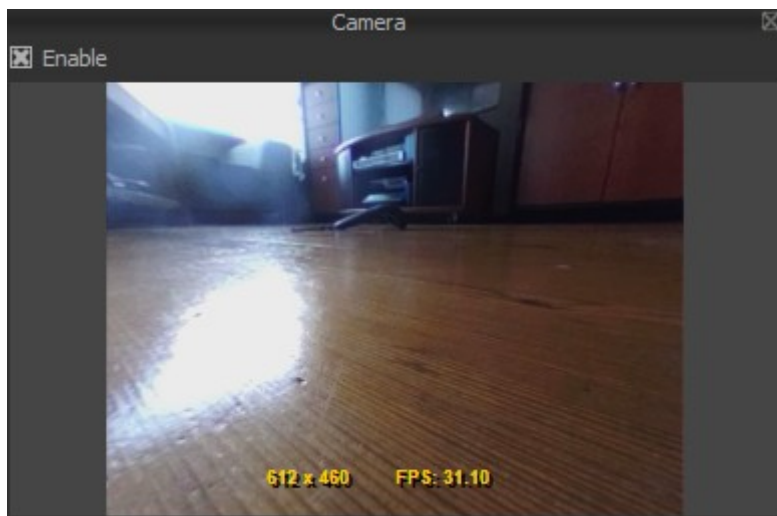


DATA SPREADSHEET

Data																		
	Device Type	Serial Number	Render Model	State	Timestamp	Frequency	PosX	PosY	PosZ	RotX	RotY	RotZ	Axis0	Axis1	Axis2	Axis3	Axis4	Buttons
0	Head-Mounted Display	LHR-5F4E568C	generic_hmd	Running OK	10047.8841	554.1	-0.61	0.12	0.18	8.99	-35.37	-0.02						
1	Tracking Reference	LHB-74C42F9A	lh_basestation_vive	Running OK			1.54	2.30	-1.59	138.83	31.77	168.63						
2	Tracking Reference	LHB-5F3E3196	lh_basestation_vive	Running OK			-2.68	2.30	1.80	9.66	66.02	158.92						
3	Controller	LHR-F73FFF47	vr_controller_vive_1_5	Running OK	10047.8841	342.8	-0.12	0.12	-0.51	15.18	20.83	2.72	0.0, 0.00	0.0, 0.00	0.0, 0.00	0.0, 0.00	0.0, 0.00	
4	Generic Tracker	LHR-06D9F64A	{htc}vr_tracker_vive_1_0	Running OK	10047.8827	408.8	-1.08	0.08	-0.35	-90.08	-7.21	179.22						
5																		
6																		

This window will display all the information regarding the detected and tracked devices.

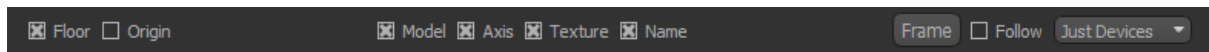
CAMERA



This window will show the video feed from the camera, if supported by your hardware. (HTC Vive only)

You can enable/disable the video stream here and furthermore specify in the Output Formats settings panel if you want to record the camera to a video file or not.

3D VIEWPORT



The 3D viewport visualizes the device data in 3D as it's coming from OpenVR.

Floor

Toggles drawing of the floor.

Origin

Draw the axis of the world at the origin (0,0,0)

Model

Draw meshes for each of the devices, note that these meshes will be automatically detected and provided by the OpenVR drivers for known devices.

Axis

Toggles drawing of axis for all devices, can give a better visual indication of rotation.

Textures

Toggles drawing of textures for the models note that these meshes will be automatically detected and provided by the OpenVR drivers for known devices.

Name

Toggle display of the device names in the 3D viewport.

Frame

Frame the 3D viewport so devices (see Framing Mode) are in view.

(CTRL+A or CTRL+F will have the same effect)

Follow

Keeps the 3D viewport centered so the devices (see Framing Mode) stay in view.

Framing Mode

All will use all devices when Framing or Following.

Just Devices will exclude tracking sources like Lighthouse base stations and tracking cameras.

3D VIEWPORT NAVIGATION

To orbit:

- Left mouse button, click & drag

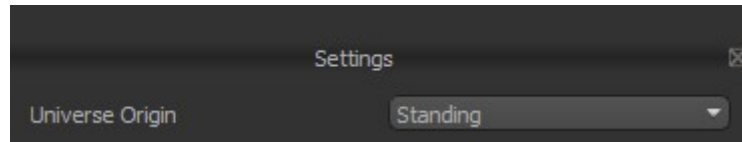
To dolly in/out:

- Middle mouse button, click & drag
- Shift + Left mouse button, click & drag
- Mouse wheel

To pan:

- Right mouse button, click & drag
- CTRL + Left mouse button, click & drag

SETTINGS



This section contains all the settings relevant to tracking and specifies how things are recorded to disk.

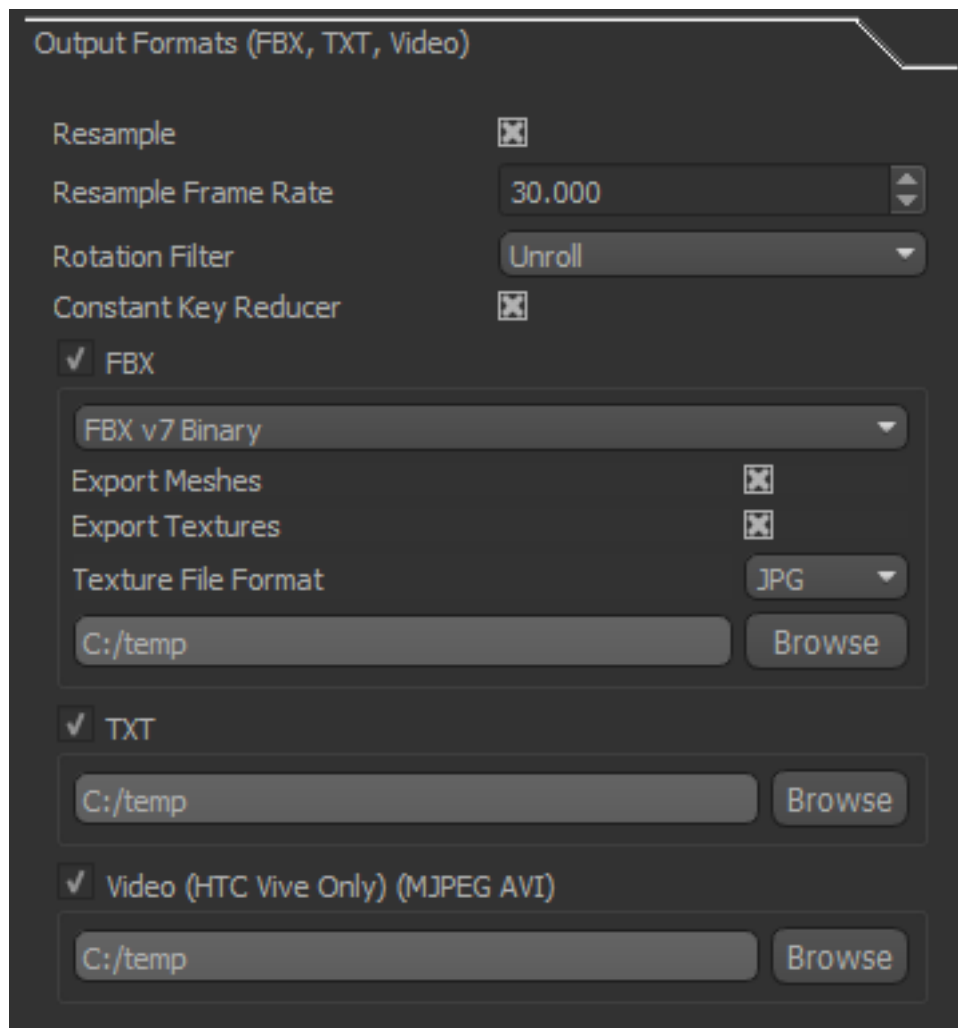
Universe Origin

Sets the origin of the OpenVR tracking space, note that this will use the calibration you've done in OpenVR Room Calibration.

Seated: set this when you've done your OpenVR Room Calibration in Seated Mode.

Standing: set this when you've done your OpenVR Room Calibration in Standing Mode.

Raw and Uncalibrated: for normal usage you generally don't need this, it will skip using the calibration data and use raw data.



Here you can choose which output format(s) to record to.

Resample

Toggle indicating if you want to resample the data to a custom target frame rate. When turned OFF the no resampling will occur.

Resample Frame Rate

Devices are usually tracked at very high framerates (300 or more frames per second) and internally be stored using timestamps. To ensure the highest amount of accuracy and fidelity.

You may want to resample this to a lower framerate (for example common are 24, 25, 50 or 60 fps) to comply with your needs

Rotation Filter

Filter to apply on rotation animation curves to prevent Euler gimble flips.

Constant Key Reducer

Filter to apply on all animation curves remove static values.

FBX

Exports to the FBX file format used by most 3D applications in the form of nulls/locators/groups objects.

This is the most versatile and accurate file format.

Note that buttons and analog axis data of controllers will be stored as custom attributes.

Everything will be parented to a reference object so you can easily translate/rotate/scale everything to your taste if needed.

FBX version

You can choose between the current v7 and older v6 and between human readable Ascii files and the smaller Binary files. In case of a problem experiment with this to see what your favorite 3D application supports.

Export Meshes

Export meshes for all the devices in the FBX file, for visualization purposes.

Meshes are parented to the nulls/locators/groups that hold the data.

Export Textures

This will export the textures used by the meshes as separate files in the same folder.

Note that if they already exist they won't be overwritten.

Texture File Format

Specifies in which file format to save the textures.

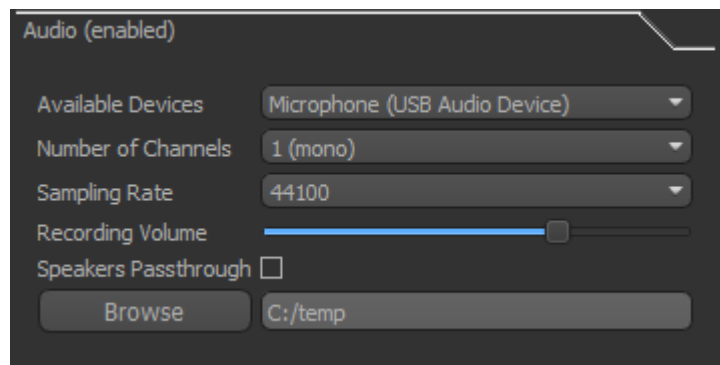
TXT

Exports to a text file which can for example be used to read the data into custom code.

VIDEO

If your headset has a built in video camera and you've enabled it (HTC Vive only at the moment) you can record this to an AVI video file with MJPEG encoding. Not that realtime video compression can take a significant amount of CPU power.

SETTINGS > AUDIO



Available Devices

List the available audio devices that can be used for recording and allows you to select one of them.

Number of Channels

Switches between Mono and Stereo recording.

Sampling Rate

Sample rate to record (in Hz).

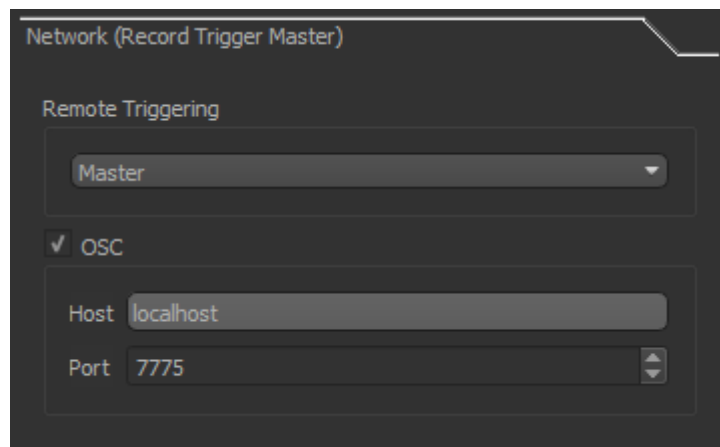
Recording Volume

Sets the volume used for audio recording

Speakers Passthrough

Playback the audio over the speakers while recording

Note that audio is always recorded in the WAV format.



The application has the ability to live stream the tracking data over a network port out in OSC format as well as communicate with other Brekel applications to synchronize recording start/stop/filenames.

Record Triggering

Allows synchronized recording across multiple Brekel applications.

One application can be in Primary mode, all others in Secondary or Ignore mode.

The Primary application will send a signal when recording is started and stopped so all applications start/stop at the same time and are using matching filenames.

Note that this works across multiple apps on the same machine and even across multiple machines on the same network.

Make sure your firewall isn't blocking port 8880-8890.

OSC

Enable/disable sending data out over UDP in OSC (Open Sound Control) format.

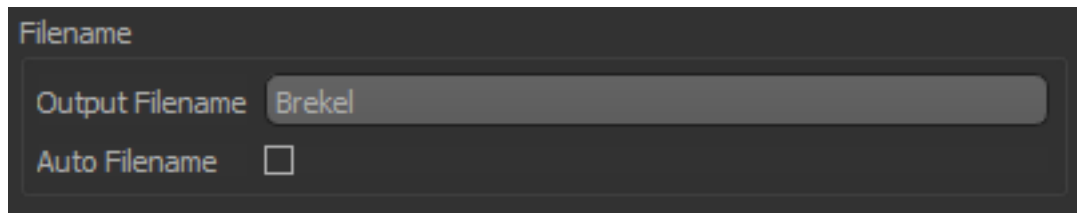
See chapter at the end of this manual for a description of the OSC messages that are sent.

Host

Hostname / IP of machine to send the OSC data to (localhost if receiving application is running on the same machine as the Brekel application).

Port

Network port over which to send the OSC data.



Filename

Output Filename Brekel

Auto Filename ☐

Output Filename

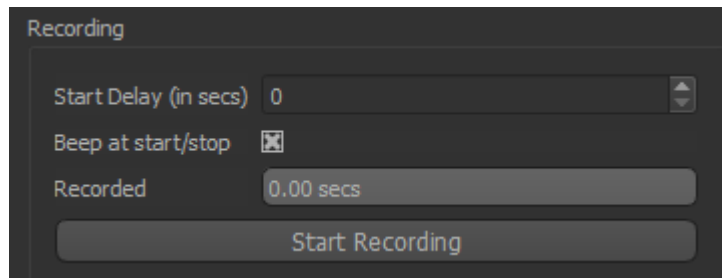
The name of the output file.

Note that the output folder(s) can be defined for each output format.

Auto Filename

When turned ON the filename will be automatically generated using the current date & time.

SETTINGS > RECORDING



Start Delay (in secs)

When bigger than 0 a countdown will start after starting a recording, giving the actor a few seconds to get into position before capture will start.

During countdown beeps will be played over the speakers every second, and the amount of remaining seconds will also be shown visually in the 3D window.

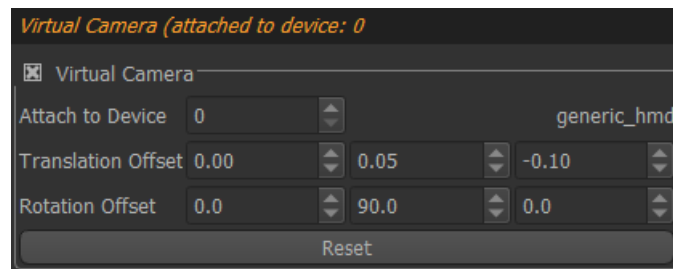
Beep at start/stop

Play a beeping sound when starting/stopping the recording.

Start/Stop Recording

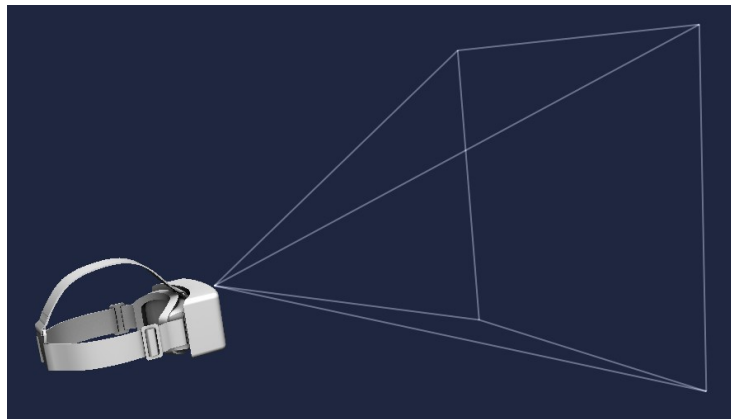
Toggle button for starting & stopping the recording.

SETTINGS > VIRTUAL CAMERAS



When enabled this will create a virtual camera attached to one of the existing devices with the applied offset.

It will be visualized in the 3D viewport and exported to the various output file formats.



Virtual Camera

Enable/disable virtual camera.

Attach to Device

Device id (from the table) to which to attach the virtual camera.

Translation Offset

Translation offset from the device the virtual camera is attached to.

Rotation Offset

Rotation offset from the device the virtual camera is attached to.

By default a camera (in FBX) is defined to look down the X axis so a rotational offset of 90 degrees on the Y axis is used by default to make it align with most HMD's and controllers.

SETTINGS > VIRTUAL MARKERS

Virtual Markers (3)

Delete All

Save Load

Default Name VirtualMarker

Start Virtual Marker Creation Mode

Name VirtualMarker_0

Export ☒ Delete This Marker

Position X/Y/Z -0.978 0.846 -0.299

Rotation X/Y/Z -134.27 50.01 109.01

Name VirtualMarker_1

Export ☒ Delete This Marker

Position X/Y/Z 0.098 0.316 -1.065

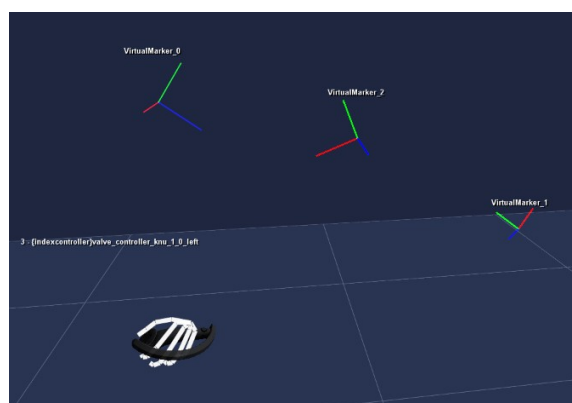
Rotation X/Y/Z 63.31 -39.62 9.01

Name VirtualMarker_2

Export ☒ Delete This Marker

Position X/Y/Z -0.572 0.758 -0.328

Rotation X/Y/Z 166.25 20.54 147.63



Virtual Markers can be created by moving a controller through the scene and pressing a button, they will leave markers in the scene that will be visualized in the 3D viewport and exported to the various file formats.

They can be a handy way to mark certain spots in your real world and have a reference in the 3D files.

They can be saved/loaded to a separate file and you have full control over their naming and position/rotation values (which were taken from the controller when it was created).

Delete All

Deletes all virtual markers from the list.

Save

Saves all virtual markers to a text file, which can later be loaded.

Load

Loads virtual markers from a text file that was previously saved.

Default Name

Default name that is used when creating a new marker (you can always change names for created markers).

Start/Stop Virtual Marker Creation Mode

Starts/Stops creation mode.

When creation mode is active move a controller through your space and hit the main fire button to create a new marker, it will appear in the list and will use the default name and position/rotation of the controller.

Name

The name of the virtual marker as it will appear in the 3D viewport and exported file, you can customize this.

Export

A toggle depicting if the marker will be exported to the output file formats or not.

Delete This Marker

Removes this marker from the list.

Position/Rotation

Transformation values for the marker, these were taken from the controller when it was created and can be manually altered if needed.

OPERATING THE APPLICATION FROM WITHIN VR

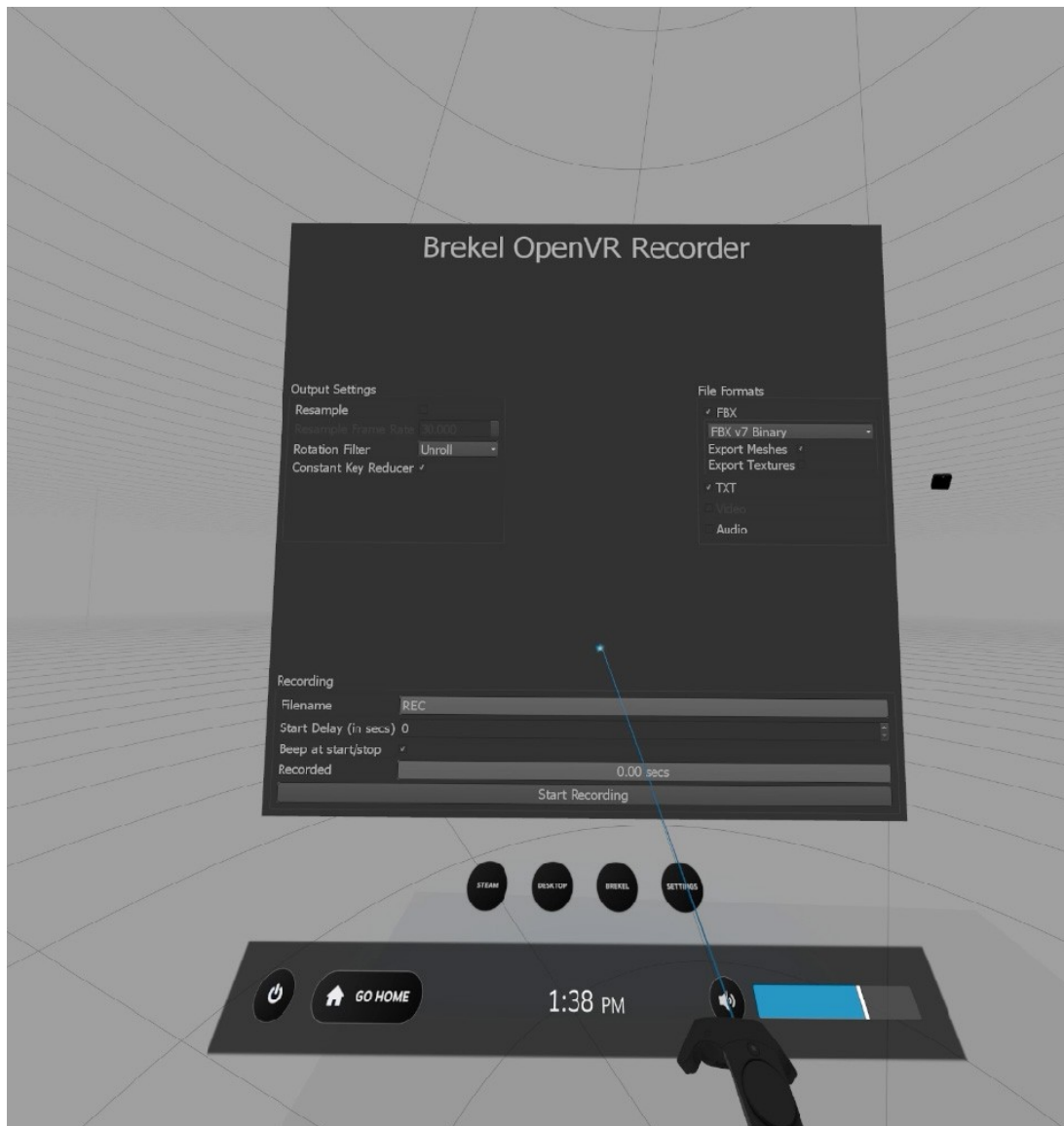
Besides controlling the application from the normal GUI on your PC you can also adjust some settings and start/stop recording while in VR.



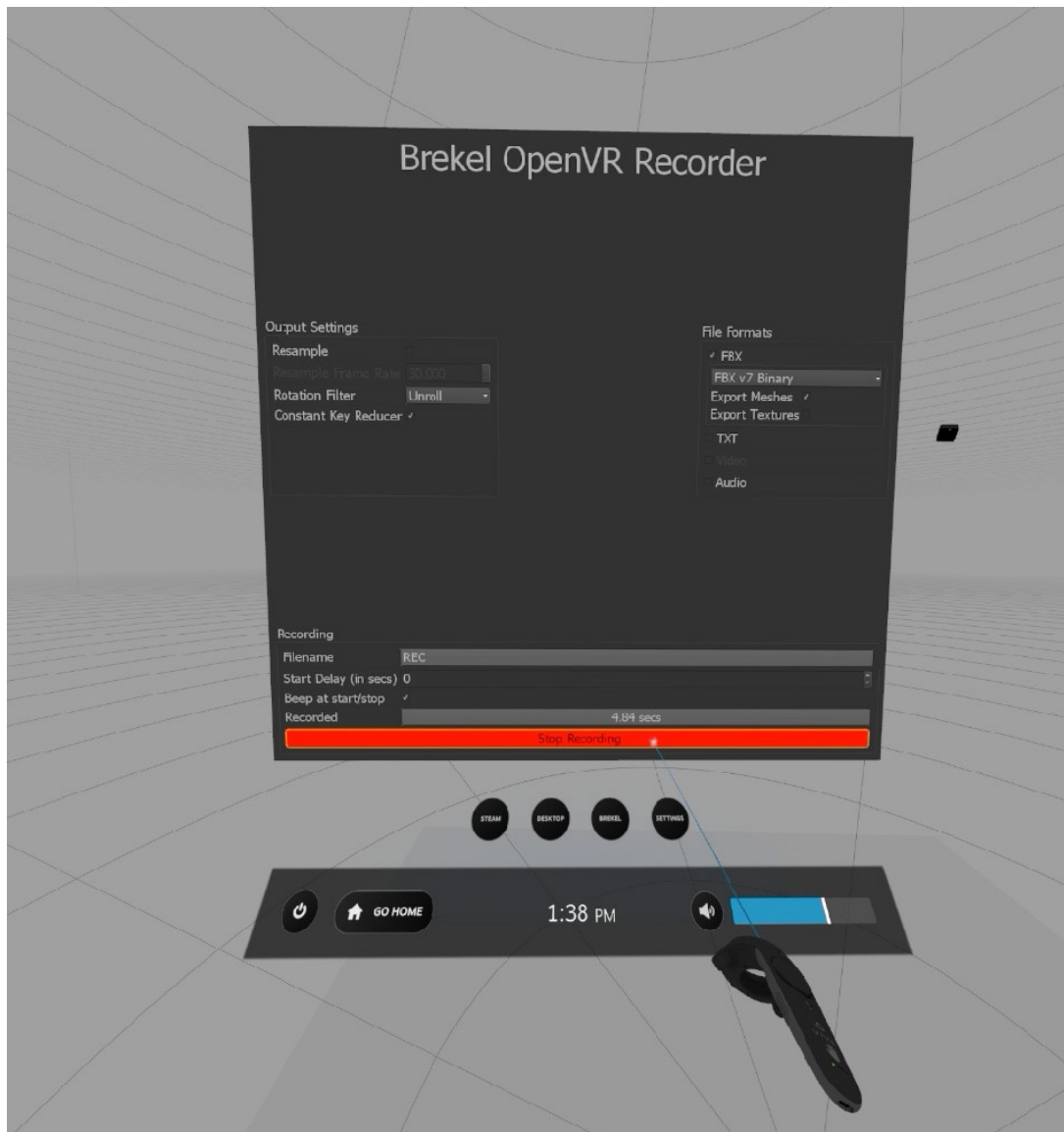
First bring up the standard Steam menu.

- On Rift touch controllers use the button with the 3 horizontal stripes on the Left controller.
- On Vive select the button just below the trackpad (icon with the 2 squares) on any controller.

Locate the “BREKEL” icon in the list of running applications and click on it.



This should bring up an interface with many familiar settings that mirror those in the base application. Note that only the basic settings for controlling the output format(s) are replicated here.



Near the bottom you can start/stop recording and check the recorded length.

Note that to change the filename (and other more advanced settings) you'll need the more elaborate controls of the desktop application.

You can now simply exit the Steam menu and record what you want to record.

TOP MENU

File > Install License

Let's you browse to a license file and installs it to be used for the next time the application is run.

File > Manual License Installation

Open folder to which you can manually copy the license .key file.

File > Show License Info

Shows if a license is installed and who it belongs to.

File > Check for new version on startup

Checks for the availability of a new version upon startup, if a new version exists a summary of new/fixed features is shown and an option to take you to the download page.

Note that you will need a working internet connection for this.

File > Exit

Saves settings and exits the application (same as clicking the X in the top right corner)

Settings > Restore interface to default

Shows all hidden docking windows and places them back into their default positions.

Extend Right Docking Area To Bottom

When enabled the area where the Settings are docked extends all the way to the bottom.

When disabled the area where the Data & Camera widgets are docked extends all the way to the right.

Settings > Draw FPS

Toggle to draw the framerate in the bottom of the viewport.

Settings > Draw Memory/CPU usage

Toggle to draw the memory and CPU usage in the bottom of the viewport.

Settings > Warn for File Overwriting

Toggle to warn for overwriting of existing files.

Help > Downloads Page

Opens your default browser and takes you to the Brekel download page.

Help > Forum

Opens your default browser and takes you to the Brekel forums. (hosted on Google Groups)

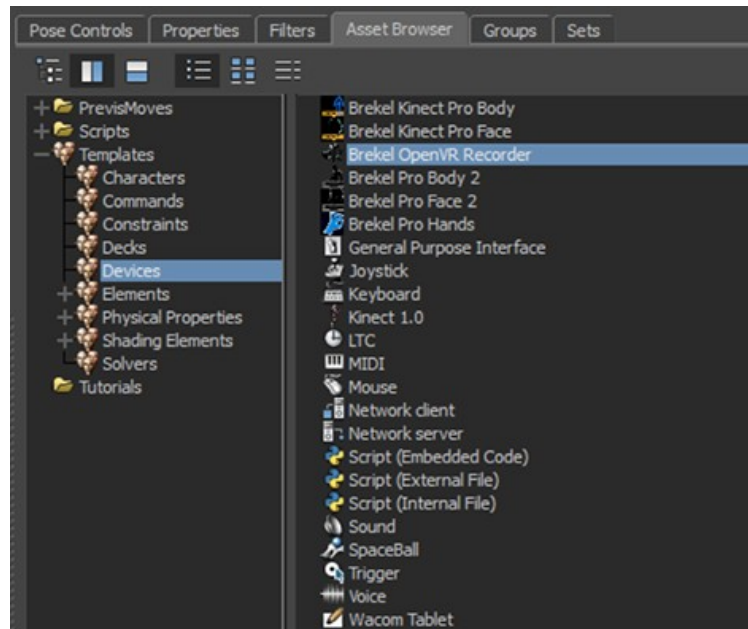
Help > About

Shows the about window with version information.

MOTIONBUILDER PLUGIN INSTALLATION

The installer should have automatically installed the plugins for the available versions of MotionBuilder.

And you should now have “Brekel OpenVR Recorder” listed under the “Devices” folder of the “Asset Browser”.



If it doesn't show up, you can find all the plugins for the various MotionBuilder versions (2009-2015 both 32- and 64 bit) in the “Brekel OpenVR Recorder” installation folder, usually in:

C:\Program Files\Brekel Pro Body 2 x64\MotionBuilder plugins

Simply copy the .dll file of your particular MotionBuilder version to the plugin folder.

Typically for a 32 bit version:

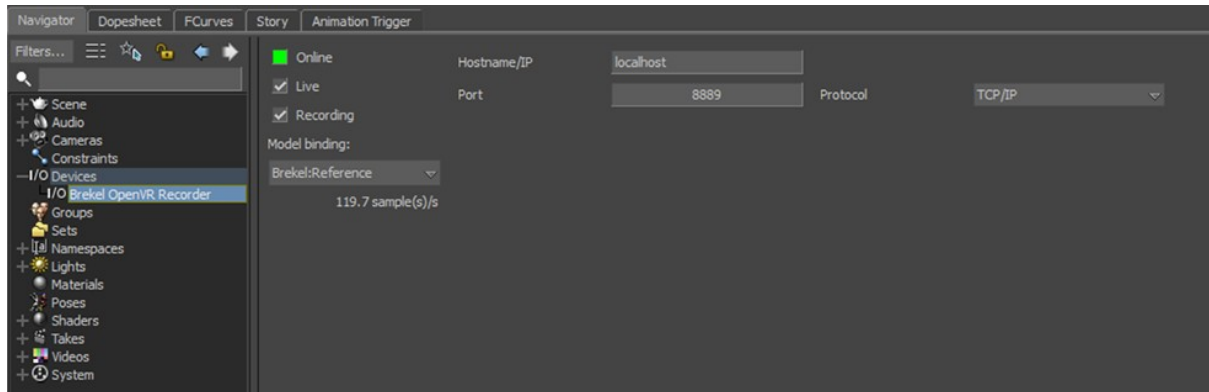
C:\Program Files (x86)\Autodesk\MotionBuilder 2013\bin\win32\plugins

For a 64 bit version:

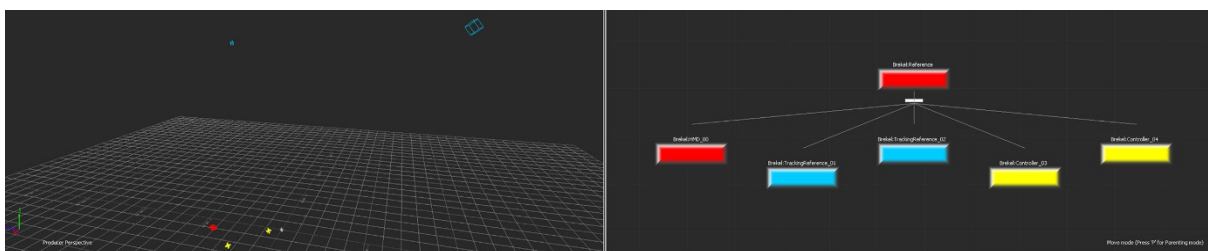
C:\Program Files\Autodesk\MotionBuilder 2013\bin\x64\plugins

USING THE MOTIONBUILDER PLUGIN

First make sure Brekel OpenVR Recorder is running, and it's set to stream out data in the Network settings.



- Drag a Brekel OpenVR Recorder device from the Devices folder in the Asset Browser into your scene
 - In the GUI for the device make sure the Hostname/IP points to the machine running the Brekel OpenVR Recorder application
(Note: if both applications are running on the same machine you can simply use the default Hostname/IP of "localhost" or "127.0.0.1")
 - Make sure Port and Protocol match in both Brekel OpenVR Recorder and MotionBuilder
 - Make sure your security software (firewall) isn't block this port/protocol
-
- Toggle the "Online" button to turn the device on
 - If it doesn't turn green go back to the steps above
 - Toggle the "Live" button to start receiving data
 - Under "Model binding:" hit the "None" option and then "Create", this will create a model hierarchy in your scene containing the skeletons



You should now objects in your scene and 3D viewport representing the OpenVR devices.

Hostname/IP

Should point to the machine running Brekel OpenVR Recorder.

If both MotionBuilder and OpenVR Recorder run on the same machine you can also leave this at the default of "localhost" or "127.0.0.1"

Port

Should be the same port as used in the Brekel OpenVR Recorder application. (default 8889)

Protocol

Should be the same protocol (TCP/IP or UDP) as used in the Brekel OpenVR Recorder application.

To record a sequence:

- Make sure “Online” on the device is toggled ON
- Make sure “Live” on the device is toggled ON
- Make sure “Recording” on the device is toggled ON
- Make sure the “Record” button on the timeline transport controls is toggled ON
- Hit the Play button, and Stop button once the action is over

To play back a recording:

- Make sure “Live” on the device is toggled OFF
- Hit Play on the timeline transport controls or scrub through the timeline

OSC (OPEN SOUND CONTROL) MESSAGES

The following messages are sent depending on the type of devices that are present:

There are devices without buttons:

When “Euler Angles” are selected in the GUI:

Address pattern: / HMD

Or / TrackingReference

Or / DisplayRedirect

int: ID of the device

string: serial number of the device

float: timestamp

float: position X

float: position Y

float: position Z

float: rotation X (euler angle in degrees)

float: rotation Y (euler angle in degrees)

float: rotation Z (euler angle in degrees)

When “Quaternions” are selected in the GUI:

Address pattern: / HMD

Or / TrackingReference

Or / DisplayRedirect

int: ID of the device

float: timestamp

float: position X

float: position Y

float: position Z

float: rotation X

float: rotation Y

float: rotation Z

float: rotation W

And there are devices with buttons:

When “Euler Angles” are selected in the GUI:

Address pattern: / Controller

Or / GenericTracker

int: ID of the device

string: serial number of the device

float: timestamp

float: position X

float: position Y

float: position Z

float: rotation X (euler angle in degrees)

float: rotation Y (euler angle in degrees)

float: rotation Z (euler angle in degrees)

float: state of button 1 (0.0 – 1.0)

float: state of button 2 (0.0 – 1.0)

.. .. .

float: state of button 14 (0.0 – 1.0)

float: Axis 1 X

float: Axis 1 Y

.. .. .

float: Axis 5 X

float: Axis 5 Y

Note that there are a total of 14 buttons and 5 axis send for each device, not all buttons/axis may contain actual data.

When “Quaternions” are selected in the GUI:

Address pattern: / Controller

Or / GenericTracker

int: ID of the device

string: serial number of the device

float: timestamp

float: position X

float: position Y

float: position Z

float: rotation X

float: rotation Y

float: rotation Z

float: rotation W

float: state of button 1 (0.0 – 1.0)

float: state of button 2 (0.0 – 1.0)

.. .. .

float: state of button 14 (0.0 – 1.0)

float: Axis 1 X

float: Axis 1 Y

.. .. .

float: Axis 5 X

float: Axis 5 Y

Note that there are a total of 14 buttons and 5 axis send for each device, not all buttons/axis may contain actual data.

And hands:

When “Euler Angles” are selected in the GUI:

Address pattern: / Hand_L

Or / Hand_R

float: timestamp

float: wrist position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: thumb knuckle 0 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: thumb knuckle 1 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: thumb knuckle 2 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: thumb knuckle 3 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: index knuckle 0 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: index knuckle 1 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: index knuckle 2 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: index knuckle 3 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: index knuckle 4 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: middle knuckle 0 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: middle knuckle 1 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: middle knuckle 2 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: middle knuckle 3 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: middle knuckle 4 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: ring knuckle 0 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: ring knuckle 1 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: ring knuckle 2 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: ring knuckle 3 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: ring knuckle 4 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: pinky knuckle 0 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: pinky knuckle 1 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: pinky knuckle 2 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: pinky knuckle 3 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

float: pinky knuckle 4 position X,Y,Z, rotation X,Y,Z (euler angle in degrees)

When “Quaternions” are selected in the GUI:

Address pattern: / Hand_L

Or / Hand_R

float: timestamp

float: wrist position X,Y,Z, rotation X,Y,Z,W

float: thumb knuckle 0 position X,Y,Z, rotation X,Y,Z,W

float: thumb knuckle 1 position X,Y,Z, rotation X,Y,Z,W

float: thumb knuckle 2 position X,Y,Z, rotation X,Y,Z,W

float: thumb knuckle 3 position X,Y,Z, rotation X,Y,Z,W

float: index knuckle 0 position X,Y,Z, rotation X,Y,Z,W

float: index knuckle 1 position X,Y,Z, rotation X,Y,Z,W

float: index knuckle 2 position X,Y,Z, rotation X,Y,Z,W

float: index knuckle 3 position X,Y,Z, rotation X,Y,Z,W

float: index knuckle 4 position X,Y,Z, rotation X,Y,Z,W

float: middle knuckle 0 position X,Y,Z, rotation X,Y,Z,W

float: middle knuckle 1 position X,Y,Z, rotation X,Y,Z,W

float: middle knuckle 2 position X,Y,Z, rotation X,Y,Z,W

float: middle knuckle 3 position X,Y,Z, rotation X,Y,Z,W

float: middle knuckle 4 position X,Y,Z, rotation X,Y,Z,W

float: ring knuckle 0 position X,Y,Z, rotation X,Y,Z,W

float: ring knuckle 1 position X,Y,Z, rotation X,Y,Z,W

float: ring knuckle 2 position X,Y,Z, rotation X,Y,Z,W

float: ring knuckle 3 position X,Y,Z, rotation X,Y,Z,W

float: ring knuckle 4 position X,Y,Z, rotation X,Y,Z,W

float: pinky knuckle 0 position X,Y,Z, rotation X,Y,Z,W

float: pinky knuckle 1 position X,Y,Z, rotation X,Y,Z,W

float: pinky knuckle 2 position X,Y,Z, rotation X,Y,Z,W

float: pinky knuckle 3 position X,Y,Z, rotation X,Y,Z,W

float: pinky knuckle 4 position X,Y,Z, rotation X,Y,Z,W