

Collecting and Logging from OpenVR (CLOVR): User Manual

Primary Developer: **Esteban Segarra Martinez**

Supporting Project: **CCRI: Planning-C: Capturing and Logging Ecological Virtual Experiences and Reality (CLEVER)** [[LINK](#)]

Grant Number: **2232448**

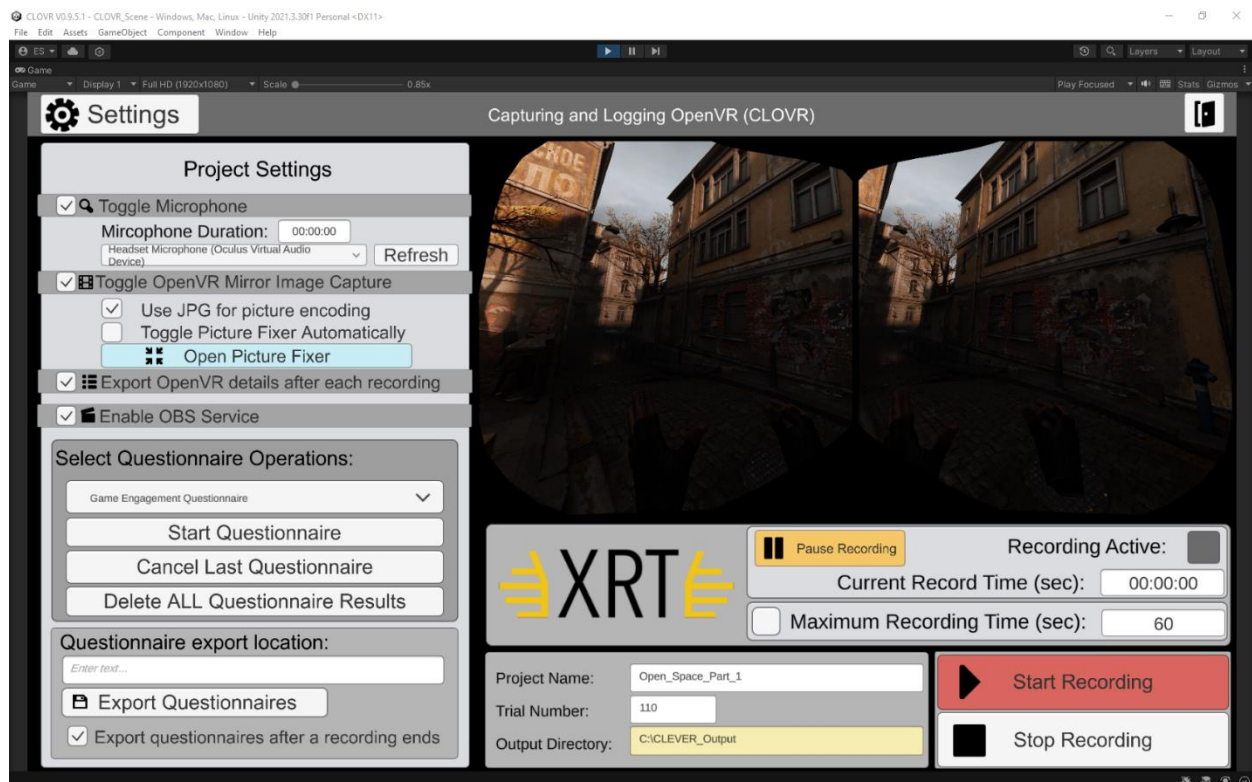


Figure 1: CLOVR Recording Half-Life Alyx

Table of Contents

Minimum requirements.....	3
Quickstart	3
Introduction.....	3
Installation	Error! Bookmark not defined.
Architecture.....	4
Questionnaires.....	4
Audio Recording.....	5
Image Recording	5
Open Broadcast Software	5
Architectural Diagram (High Level)	7
Basic Usage	7
Role of an Investigator	7
Role of a Participant	7
Example Usage:.....	8
Primary Controls	8
Settings	9
Manual Microphone usage	9
OpenVR mirror capture.....	10
Enabling OpenVR Detail Exporting	10
Enabling Open Broadcaster Software	10
Questionnaires.....	11
Parameters for saving data	11
Additional settings	11
Step by Step Example Trial.....	12
Closing CLOVR.....	12
Selecting the OBS VR View to collect	12
Recording OBS using Oculus Mirror View as target VR view.....	12
Recording OBS VR view for any other application	13
Data output format.....	13
Developer Notes	13
Preview VR mirror view	13
Multiple Threads for Recording	14
Persistent Settings	14
Example Datasets.....	15
Closing Notes	15

Minimum requirements

OS*: Windows 10 – 11.

Processor: Intel Core i5-4590/AMD FX 8350 equivalent or better

Memory: 16GB or higher

Graphics: Nvidia 1070 or better with 8GB to 10GB of VRAM.

* As this application depends on SteamVR with a minimum version of 2.0; Windows 10 has become the minimum level of support for OS due to Valve dropping support for any windows version earlier to Windows 10.

Quickstart

1. Install OBS from a trusted source [\[LINK TO GITHUB\]](#)[\[WEBSITE\]](#)
 - a. Install this application first to be able to record videos.
2. Download and unpackage CLOVR in a known location.
3. **Start SteamVR.**
 - a. If on the Oculus platform, connect your Oculus, connect to Oculus Rift, and start SteamVR
4. Start CLOVR, record a test recording to confirm and setup/apply settings to OBS.
5. **If there is an OBS recording error, open OBS and confirm the following:**
 - a. **Profile set is: CLEVER_OpenVR_OBS_Capture**
 - b. **Scene Collection set is: Video-and-Microphone-VR-Capture**
 - c. **Check that your audio capture is working; Default audio capture does not always work in capturing audio.**
6. You should be ready to record any application!
 - a. Open a VR application and record sections of your interest

Special note: If running a questionnaire severely impacts runtime FPS and/or capture speed. Restart everything in the order of: SteamVR, Oculus Rift, and finally your PC. This is a SteamVR related issue, not a CLOVR related issue. Potentially this lag slow down may be fixed by Valve in the future.

Introduction

CLOVR is a free, open source project for collecting and recording pose and video data from a VR session. Its primary application is towards collecting live VR sessions from typical VR applications. These include test VR games from Unity, commercial/freeware VR games, or just from testing the Steam VR passthrough. Any application or VR system that can interface through SteamVR can be recorded using CLOVR, and with some special instructions, OpenXR exclusive applications.

Over the years, some solutions have been presented to record and store data, however none have combined the use of OBS or integrated questionnaires directly into the data collection process. Additionally, **we record the data in real-time and save the data at the same time to avoid data loss in the event of catastrophic application failure and to minimize I/O costs.**

This application can help researchers precisely record and replicate recording settings between different applications. This is because the recording settings and conditions are the same throughout different

applications as CLOVR does not interfere directly with an application and instead interfaces with OpenVR to gather data simultaneously to a OpenVR application. **CLOVR does not need to inject code or must be installed directly into a VR application.**

Add-ons or modifications are possible to be performed to CLOVR. While it is not coded to handle modifications like game modifications, developers can modify the application to add in specific functionality they would desire CLOVR to have.

Architecture

The architecture of CLOVR consists primarily of the Unity Engine application running as an independent application. One external library developed for CLOVR were used to specifically control starting and stopping OBS. CLOVR uses the OpenVR API that uses the SteamVR runtime to interface with VR equipment and collect pose and HMD image data.

Questionnaires

Questionnaires are constructed from XML files which contain several details used to form a questionnaire question. The following are the bare minimum requirements for a questionnaire survey:

- Questionnaire
 - Title
 - QuestionnaireName
 - Questions
 - <string>
 - Subquestions
 - <string>
 - Answers
 - <string>
 - Labels
 - <string>

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <Questionnaire xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3   <title>Please take some time for this questionnaire.</title>
4   <questionnaireName>Template Questionnaire</questionnaireName>
5   <questions>
6     <string>Empty.</string>
7   </questions>
8   <subquestions>
9     <string>N/A</string>
10  </subquestions>
11  <answers>
12    <string>Empty</string>
13  </answers>
14  <labels>
15    <string>Empty</string>
16  </labels>
17 </Questionnaire>
```

Figure 2: Default template for a Questionnaire

Currently, subquestions are disabled from being displayed. The intent was to create a subcategory of questions that would have custom questions per question. However, this was restricted to only Likert answer questionnaires.

By default, CLOVR comes with

- Game engagement questionnaire (GEQ)
- Spatial presence experience scale (SPES)
- Simulator Sickness Questionnaire (SSQ)
- Preliminary Embodiment Short Questionnaire (PESQ)

Users can create and add new questionnaires by adding them to the default questionnaire folder or by telling CLOVR to look at a different folder to look for questionnaires.

Audio Recording

Users can record audio from an audio source on the computer. This can include the HMD microphone or installed computer microphones. Users can disable this feature if they do not feel like they require to record the microphone manually. Users have to set the interval of time they desired to record, such as 2 – 10 minutes. If recording was set by using the timer feature, the microphone will be updated to span the duration of the set trial.

Image Recording

This aspect handles the recording of the view as seen from the eyes of the head mounted display. This is useful data if for example, researchers wanted a full preview image of what the user was looking at while looking around in VR. This is offered in conjunction with OBS video capture as OBS can only capture what it can see through preview windows from SteamVR or Oculus Mirror View. The images captured will be captured at the maximum resolution size of the device.

A separate python tool is provided for advanced users to decompose the video into individual frames under the name *Python Video Decomposer*.

Open Broadcast Software

CLOVR has support for using Open Broadcast Software for recording and storing videos collected from the HMD. By default, CLOVR will attempt to install the necessary scenes and profiles to get OBS to operate correctly, however some installation settings may be needed to be tweaked by the user to ensure recording is working. These include the window capture, system audio, and microphone audio (if required).

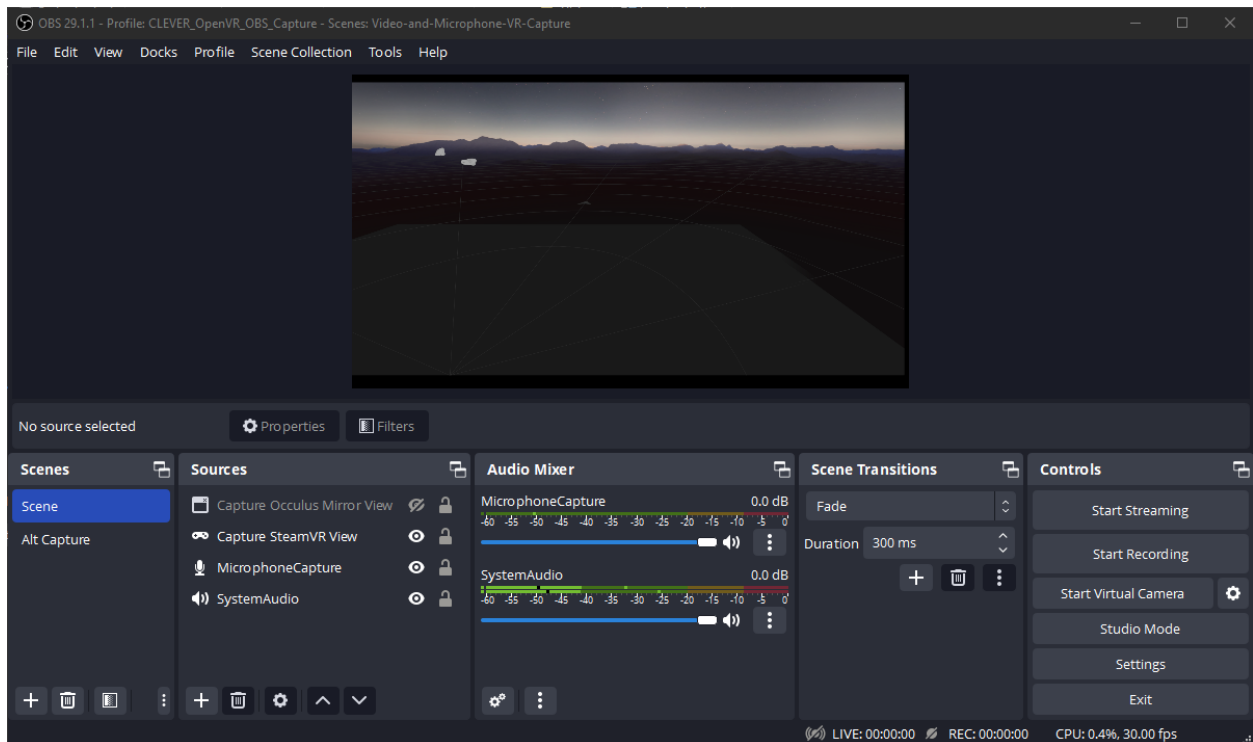


Figure 3: Typical OBS setup while using SteamVR VR view

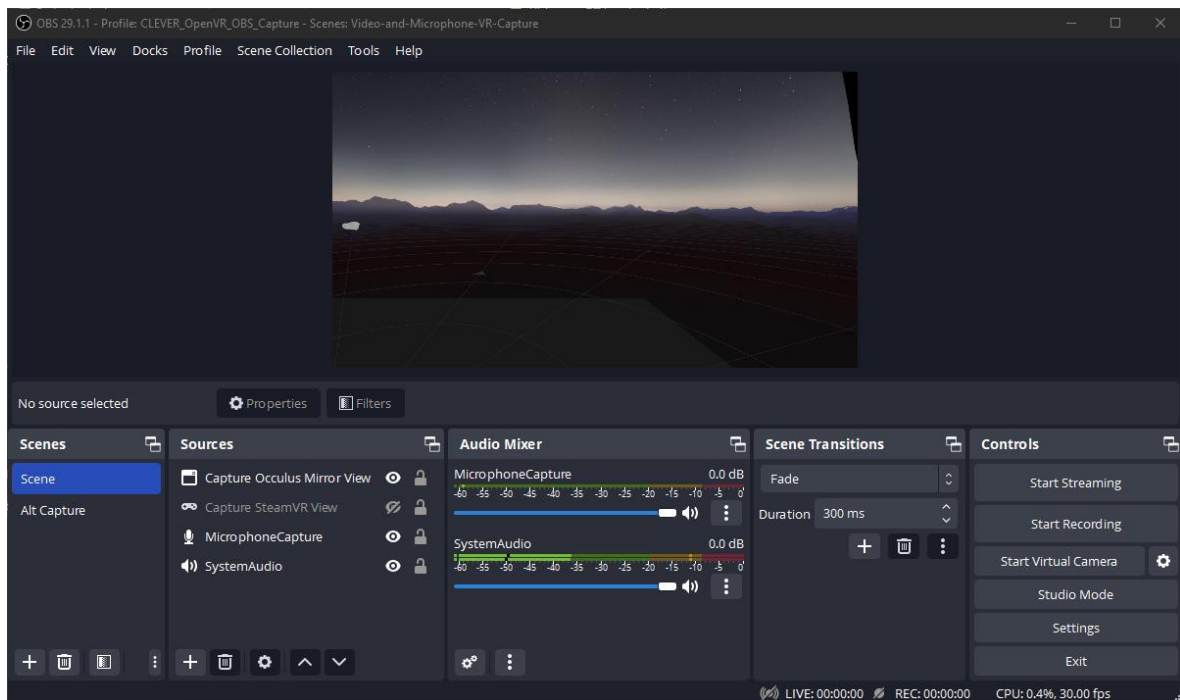


Figure 4: OBS setup to record from Oculus Mirror View.

Below is a diagram of components that compromise CLOVR. These represent a high level overview of the Unity scripting involved to put together the recording system.

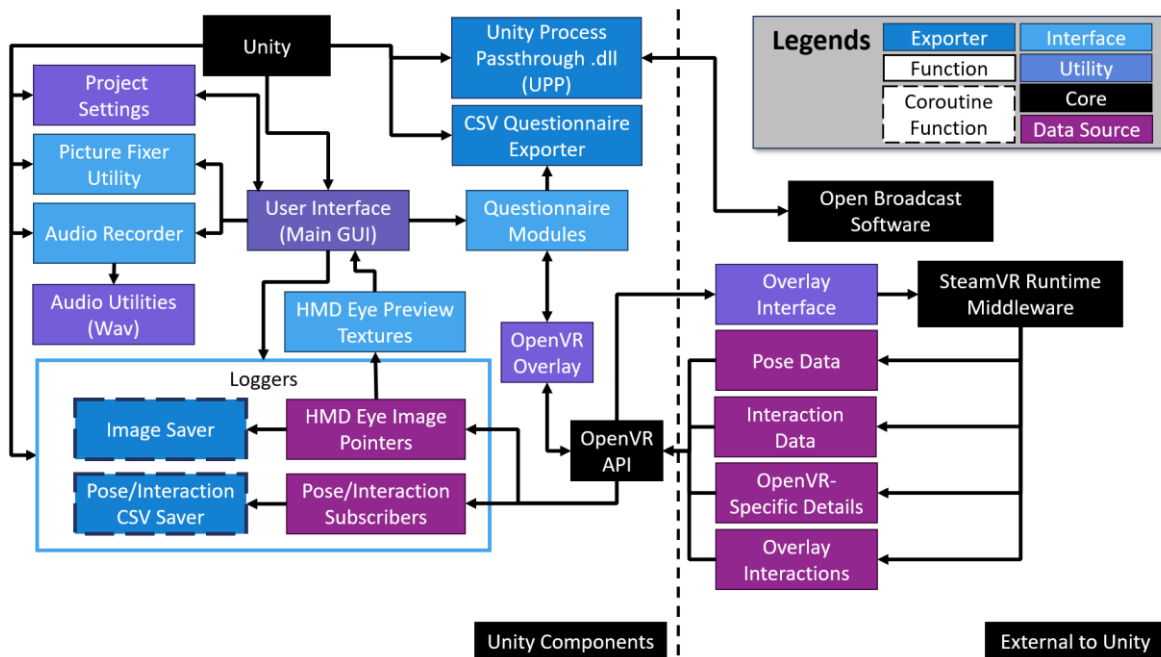


Figure 5: Architectural diagram

Basic Usage

The application is setup to be used as two role system defined as an investigator and a participant.

Role of an Investigator

The investigator will interact with CLOVR to set recording settings and record. The investigator can set the following settings:

- Maximum recording session time
- Enable/Disable recording video with OBS
- Enable/Disable recording OpenVR image frames
- Enable/Disable Microphone
- Setting output location for the trial recording
- Setting the name for the application being recorded
- Setting the unique identifier of the participant being recorded
- Questionnaire output

Role of a Participant

The role of the participant is to perform actions inside of the VR application. This can be any running VR application which can include interactions with the controller, movement with trackers, or poses with any currently connected device.

Special note: If a participant takes does not have the headset on their head or takes it off during runtime, the recording will pause until the headset is reseated.

Example Usage:

An investigator can record any application assuming OBS's settings are set and recording path is set. By default CLOVR will attempt to put a folder at **<local hard drive>/CLOVR_Output** if no output folder is provided.

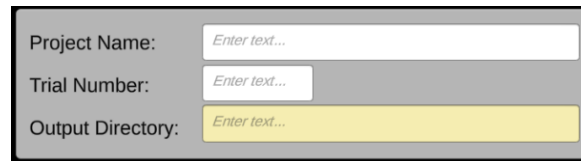
A form with three input fields. The first field is labeled 'Project Name:' and has a placeholder 'Enter text...'. The second field is labeled 'Trial Number:' and has a placeholder 'Enter text...'. The third field is labeled 'Output Directory:' and has a placeholder 'Enter text...'. The fields are arranged vertically in a light gray box with a black border.

Figure 6: Participant input fields and output directory.

Primary Controls

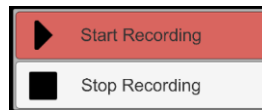


Figure 7: Start and stop buttons

The primary controls are similar to those of a standard editor, with a start, stop and pause button. Start triggers a couple of actions which include setting up the output directories, starting the OBS service, and capturing images from OpenVR.

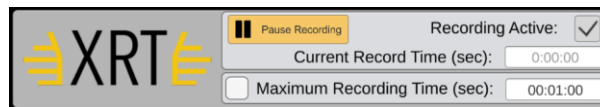


Figure 8: Additional settings bar and pause button.

Pressing the pause button will pause recording at any time but won't stop the OBS recording. Pause mode may be entered if the headset is removed from the user's head. A large pause warning will be in front of the VR preview windows.

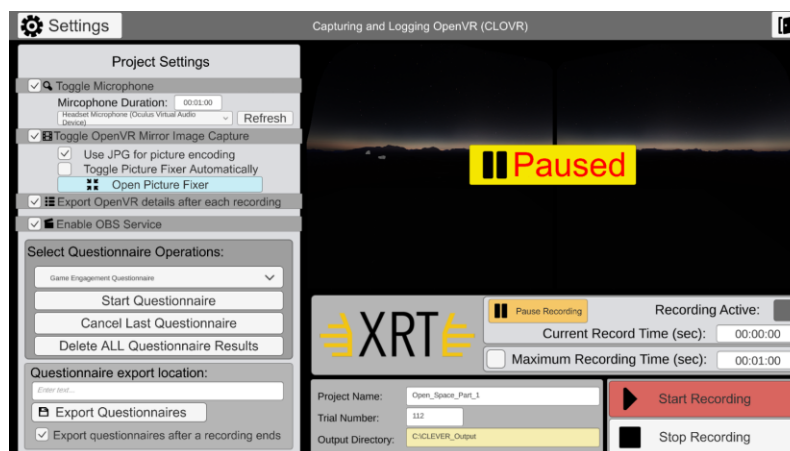


Figure 9: CLOVR while in pause mode

The **Maximum Recording Time** sets up bar sets up a maximum time for a trial, such that instructors can produce consistently the same resulting dataset each time. The checkbox next to this line enables or disables it.

A flashing red **Recording Active** checkbox indicates if CLOVR is actively attempting to record a session. It will continue to flash despite being paused but will not record.

If SteamVR was not enabled prior running CLOVR, a large banner will stop further use and will report to restart the program with SteamVR running.

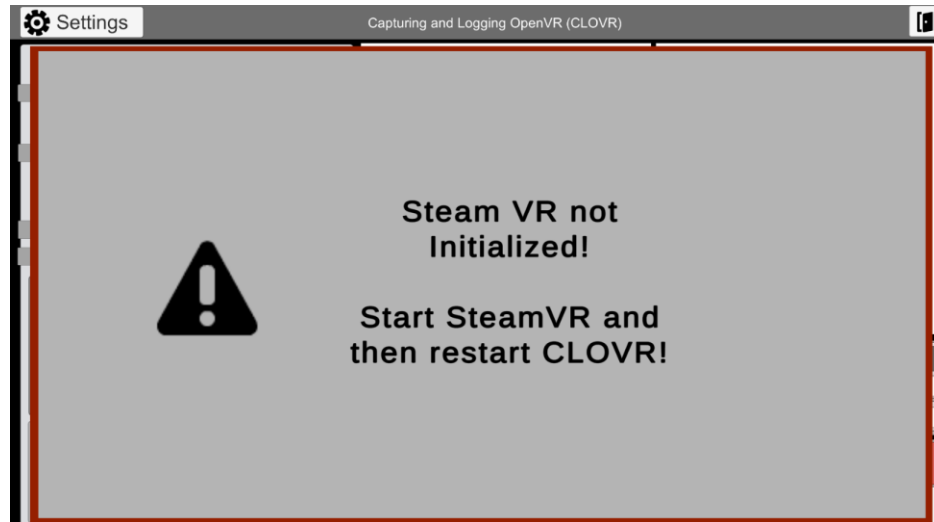


Figure 10: CLOVR Warning banner with SteamVR disabled

Settings

Investigators are then provided with a couple of choices as to how to setup the recording; which include the following:

- Toggling the manual microphone
- Toggling the OpenVR mirror capture
- Exporting OpenVR session details
- Enabling OBS service

Manual Microphone usage

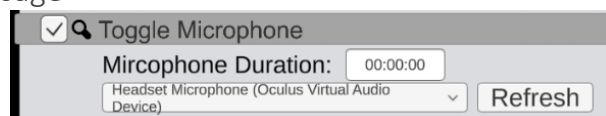


Figure 11 Manual microphone settings

The microphone can be setup in this tab of the interface. Users can set how long does the recording last and which port index microphone should be recorded. If recording sessions are time-limited, the microphone's maximum time is set to the maximum time recorded.

OpenVR mirror capture

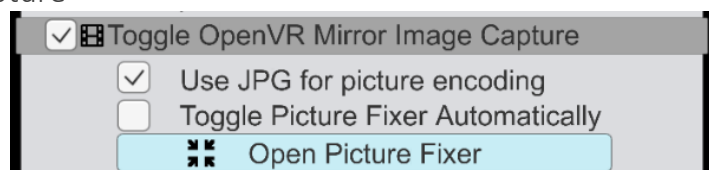


Figure 12: Toggle for OpenVR mirror capture

This section handles the capture of OpenVR images. OpenVR mirror images are images that are captured raw from the HMD with minor post processing involved. Since these images are sent to the HMD, they usually have some changes while being captured in Unity. These include being slightly darker and upside down while captured. CLOVR also provides the option of saving the images as .jpg (Default) or .png if needed. Images obtained from this technique obtain a vignette outline which hides parts of the image from being rendered and its outline is different depending on the HMD used. Currently there is no way of removing this from the final image.



Figure 13: Left and right Oculus HMD images

Fortunately there are some options to mitigate this problem. CLOVR has a built-in picture inverter for inverting images. This picture fixer can invert the images; however, it is currently a slow option which will be fixed in a future update. Users are encouraged to use [XnViewMp](#) to correct images in large quantities if needed.

Enabling OpenVR Detail Exporting



Figure 14: Exporting OpenVR room size and settings

This tab enables exporting the room size and applications running in OpenVR during the recording. Future updates may expand the amount of data available through this file.

A	B	C	D	E	F	G	H	I
Room Corner 0	Room Corner 1	Room Corner 2	Room Corner 3	Room Size X	Room Size Y	Number of Applications active	OpenVR Runtime Version	Amount of Recorded Sessions
1.702658;0;-1.459439	-1.702658;0;-1.459439	-1.702658;0;1.459439	1.702658;0;1.459439	3.405316	2.918878	31	2.2.3	0

Figure 15: Settings as seen through OpenVR exported to CSV

Enabling Open Broadcaster Software

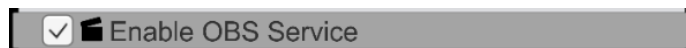


Figure 16: OBS service enabler

This checkmark enables the OBS service to be enabled every time when a recording is started. If disabled, the OBS service will be left disabled. By default, OBS will start **minimized** to avoid interfering with the instructor's

GUI. Quitting the application at any time will destroy the OBS instance and terminate the recording at that time.

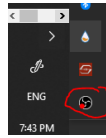


Figure 17: Minimized OBS service icon

Questionnaires

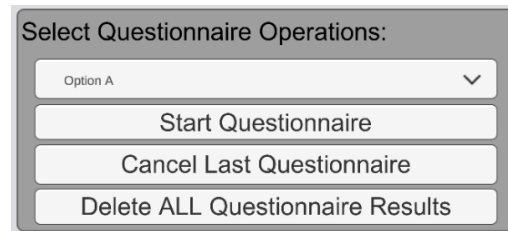


Figure 18: Questionnaire presenter and deleter

An investigator can also present questionnaires to the participant at any time, weather the application is recording or not. They can also delete the last taken questionnaire or all prior questionnaires.

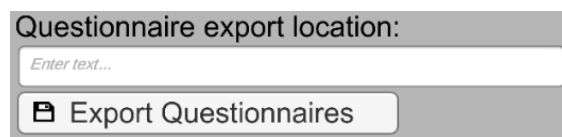


Figure 19: Questionnaire export location button and field

Questionnaires can be saved and exported to custom locations if manually saved. When set by the user, questionnaires can be automatically exported after a recording session ends.

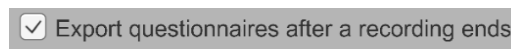


Figure 20: Questionnaire export after recording ends.

Parameters for saving data

Investigators can enable or disable recording of any of the key recording features of CLOVR for privacy or data savings.

Additional settings

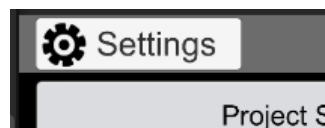


Figure 21: Additional settings button

Additional settings can be accessed by clicking on the cogwheel button on the top left of the application. These settings are more core to how CLOVR obtains its information such as the **OpenVR device bindings config files** and the **XML questionnaires**. Normally users would not want to change the bindings, however the option is there should researchers require changing the bindings.

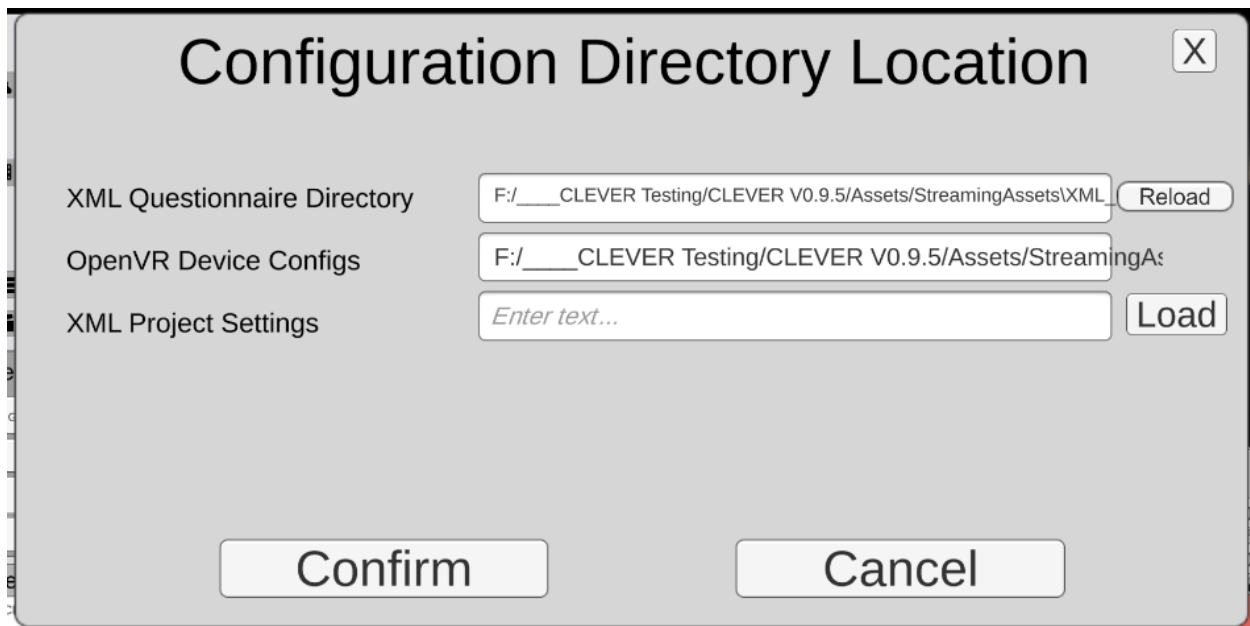


Figure 22: Additional settings window.

The XML project settings field is open for those who may desire to change the location where CLOVR stores its default settings in. These settings may be exported and recalled at a later time.

Step by Step Example Trial

1. Start SteamVR or Oculus Rift
 - a. If using SteamVR, Open VR view on the SteamVR application
2. Start CLOVR
3. Start target application to record
4. Start recording
 - a. Pass questionnaires to participant (optional)
 - b. Pause recording
5. Close/finish up CLOVR

Closing CLOVR

After closing CLOVR, CLOVR remembers the settings and options selected by the instructor. These options will be reloaded by the next time the program is started. This means that the saved options will remain persistent for the session. You can reset the options by deleting the [PersistentProjectSave.xml](#) file. CLOVR should reset the project settings to default.

Selecting the OBS VR View to collect

This may be necessary at times due to the way certain VR applications may choose to not preview VR view using the SteamVR mirror view. In these instances, users can choose to preview using either the preview provided by the open application, or in certain special circumstances, use Oculus VR Mirror View. Special instructions will be provided in the next section:

Recording OBS using Oculus Mirror View as target VR view

1. Navigate to [C:\Program Files\Oculus\Support\oculus-diagnostics\](#) and start the [OculusMirror.exe](#) application.

2. Upon starting the application, open OBS and look at the scenes.
3. Create a new [Window Capture](#) or [Game Capture](#)
4. Depending on your selection, set the target window to [\[OculusMirror.exe\]: Oculus Mirror](#)
5. Done, check the window preview on OBS to confirm it is working

Recording OBS VR view for any other application

1. Create a new [Window Capture](#) or [Game Capture](#)
2. Depending on your selection, set the target window to record as the window for the [VR application you are using](#)
3. Done, check the window preview on OBS to confirm it is working

Data output format

Datasets are outputted to a target destination set by the [Output Directory](#) setting on the UI. By default, if possible, CLOVR creates an output directory in [C:/CLOVR_Output](#) The following represents the output directory format, extensions represent the possible export type:

- Data
 - <Application Name> < Trial Number>
 - Microphone_recordings
 - .wav
 - Pictures
 - leftEye
 - .png
 - .jpg
 - rightEye
 - .png
 - .jpg
 - Poses
 - .csv
 - Questionnaires
 - .csv
 - Videos
 - .mkv or .mp4 (Depends on OBS settings)
 - Project Properties (.csv)

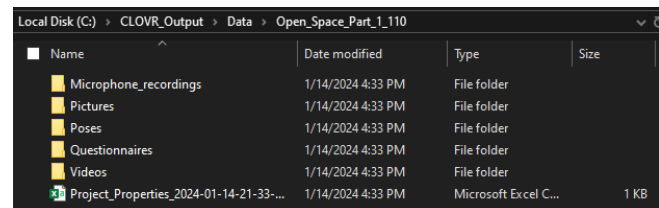


Figure 23: Directory output

Developer Notes

This section serves to give additional details about how CLOVR works that may be useful for users to understand.

Preview VR mirror view

From time to time, users may have to change the OBS recording setup in applications where SteamVR view was disabled. A disabled SteamVR mirror view does not necessarily mean pose recording is disabled. When mirror view is disabled, this means that it was disabled because the game engine does not support the mirror view or it was disabled to optimize rendering. This problem is more common in specialized VR engines or Unreal Engine.

Developers are welcome to see how OpenVR VR mirror view is captured by looking at the [OVRT_Manager.cs](#) and locating the [CollectTextureFromEye](#) function. Primarily CLOVR supports the function provided by OpenVR API under [OpenVR.Compositor.GetMirrorTextureD3D11](#). Users are warned that the function provided with a pointer to a texture in GPU and cannot be directly copied or accessed by normal means. The only way around this problem is to capture a snapshot of the texture using [AsyncGPUReadback](#) and handing the threading in a way such to avoid breaking the pose and image recording.

Multiple Threads for Recording

Data is stored as soon as it is received from the OpenVR API and stored locally until a buffer is filled. This buffer is used to then call a thread to save the data to disk. Image saving is also threaded to avoid impacting the performance of CLOVR's pose collection.

Persistent Settings

Settings are handled by the [LoggingManagerAPI.cs](#) script where CLOVR settings are saved in an XML file that handles the following attributes:

- xmlConfigFilesLocation
- openvrRuntimeLocation
- xmlQuestionnaireLocation
- pictureFileExtension
- obsProfileLocation
- obsSceneLocation
- obsExeLocation
- projectName
- recordingFrameInterval
- microphoneDurationSecs
- microphoneIndex
- bufferSize
- trialCount

Settings are set every time the application is successfully closed. Configuration of special locations such as where to obtain the OpenVR configuration bindings can be set in a special UI inside the application. An example XML file is provided to allow users to change this file with personalized settings.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <ProjectSettings xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3   <xmlConfigFilesLocation>F:/CLEVER Testing/CLOVR V0.9.5.1/Assets/StreamingAssets/PersistentProjectSave.xml</xmlConfigFilesLocation>
4   <projectOutputLocation>C:\CLEVER_Output</projectOutputLocation>
5   <openvrRuntimeLocation>F:/CLEVER Testing/CLEVER V0.9.5/Assets/StreamingAssets/Config/actions.json</openvrRuntimeLocation>
6   <xmlQuestionnaireLocation>F:/CLEVER Testing/CLEVER V0.9.5/Assets/StreamingAssets/XML_Questionnaires</xmlQuestionnaireLocation>
7   <pictureFileExtension>.jpg</pictureFileExtension>
8   <obsProfileLocation>C:\Users\Botan\AppData\Roaming\obs-studio\basic\profiles\CLEVER_OpenVR_OBS_Capture</obsProfileLocation>
9   <obsSceneLocation>C:\Users\Botan\AppData\Roaming\obs-studio\basic\scenes\VideoandMicrophoneVRCapture.json</obsSceneLocation>
10  <obsExeLocation>C:/Program Files/obs-studio/bin/64bit</obsExeLocation>
11  <projectName>Open_Space_Part_1</projectName>
12  <recordingFrameInterval>1</recordingFrameInterval>
13  <microphoneDurationSecs>60</microphoneDurationSecs>
14  <microphoneIndex>0</microphoneIndex>
15  <bufferSize>1000</bufferSize>
16  <trialCount>113</trialCount>
17 </ProjectSettings>
```

Figure 24: Example of a filled out Persistent Settings XML file.

Example Datasets

Example datasets can be downloaded at the following location Zenodo DOI location:

<https://doi.org/10.5281/zenodo.10499190>. The best recommendation is to create example datasets from your own VR applications and observing the results from the collected datasets. The datasets are intended to be used by machine learning algorithms such as those used for gesture recognition and user recognition for cases where the pose data may be used.

In situations where visual data is used, users can use the images for computer vision or simultaneous localization and mapping.

Closing Notes

Special thanks to Dr. Ryan McMahan for helping me string along a project of this scale. I also thank Ayesha Malik for helping with initial trialing and publication writing for the paper attached to this project.

For questions related to problems or concerns related to CLOVR, please post to the [Github repository](#) of the project.

For additional information, please contact me at Esteban.segarra@ucf.edu or Dr. McMahan at rpm@ucf.edu

Thank you and best regards for using our tool!