Remote VR Visualization Application VR-PBVR

User Manual

Ver. 2.2

Center for Computational Science & e-System, Japan Atomic Energy Agency

2023/11/29

Table of contents

Release history

version	update	contents
1.0	2023/03/15	First version
2.2	2023/11/29	Modified build and installation

1. Introduction

This manual describes how to install and use VR-PBVR, a particle-based remote VR visualization application developed at the Center for Computational Science and e-Systems (CCSE) of the Japan Atomic Energy Agency (JAEA). CCSE has developed a particle-based client-server remote visualization application PBVR (CS-PBVR), and VR-PBVR is an extension of CS-PBVR for VR visualization.

CS-PBVR is based on PBVR (Particle Based Volume Rnderring) and KVS (https://github.com/CCSEPBVR/KVS), a visualization library, and realizes high-speed remote visualization of large volume data on remote servers. CS-PBVR consists of a Particle Sampler running on the remote server and a Particle Renderer running on the user PC. The Particle Sampler generates particle data for visualization from the volume data on storage and transfers the particle data to the Particle Renderer connected via socket communication. The Particle Renderer generates a volume rendering image from the received particle data, and the viewpoint can be changed without re-generating particles.

VR-PBVR is an extension of CS-PBVR's Particle Renderer for the Oculus head-mounted display (HMD). VR-PBVR's Particle Renderer generates binocular images from particle data and sends them to the HMD. VR-PBVR is capable of changing viewpoints by head tracking, as well as transforming the coordinates of the visualization data by using the Oculus touch controller for VR. It is also possible to control the time step and bounding box display by GUI on the VR space. The visualization parameter editor GUI of CS-PBVR is available, which can edit transfer functions for multi-variable data.

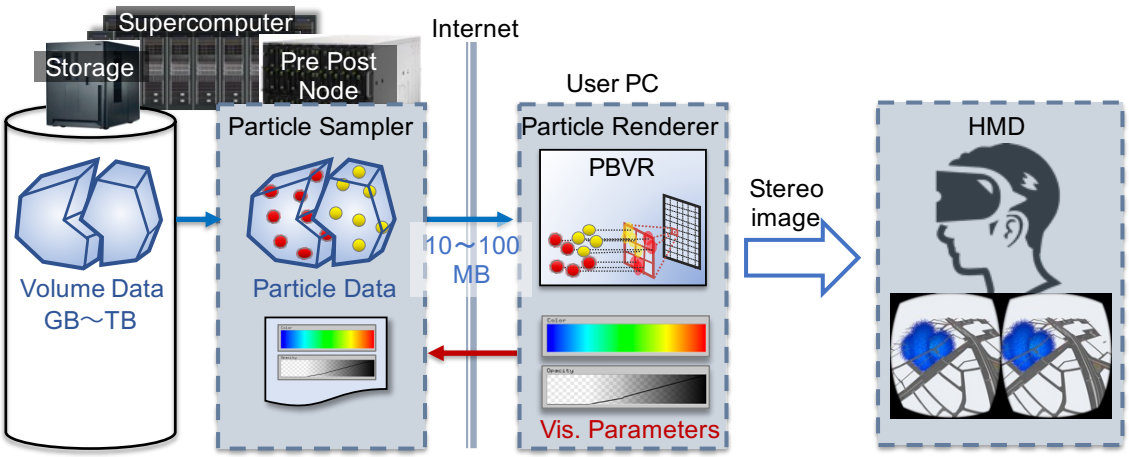

Figure 1.1-1Components of VR-PBVR

1.1. Environment

VR-PBVR has been tested in the following environments

	Specification
CPU	Intel i3-6100 / AMD Ryzen 3 1200 / AMC FX4350
GPU	NVIDIA GTX 1050Ti / AMD Radeon RX 470
メモリ	8GB RAM
HMD	Oculus Rift (CV1) / Oculus Rift S / Quest / Quest2
OS	Windows 10 (Home/Pro) / Windows 11 Pro
Compiler	Visual Studio 2017 / Visual Studio 2019
IDE	Qt Creator 5.9 / Qt Creator 6.5

2. Dependency Library

The following libraries are required to build and run VR-PBVR. Those of KVS2.9mod4meta and CGFormatExt4KVS are aavailable from following link.

github(https://github.com/CCSEPBVR/KVS)

The others are available from links described in ReadMe.txt of above github.

2.1. KVS2.9mod4meta

KVS2.9mod4meta is a KVS modified for VR-PBVR and provides basic visualization capabilities for HMD Meta.

2.2. CGFormatExt4KVS.zip

CGFormatExt4KVS is a library that converts polygon data in 3DS and FBX formats into polygon data in VR-KVS format.

2.3. assimp-5.0.0.zip

Assimp is a C++ library that provides the ability to load polygon data in 3DS format. It is statically linked at CGFormatExt4KVS build time, so it is not needed at runtime.

2.4. fbx20195_fbxsdk_vs****_win.exe

This is an installer for FBX SDK, a library provided by Autodesk that provides the ability to read polygon data in FBX format, which is required to build and run CGFormatExt4KVS and VR-PBVR. The version of Visual Studio is replaced as needed.

2.5. freeglut-MSVC-3.0.0-2.mp

This is a binary package of OpenGLUT, a GLUT for Windows, required to build and

run VR-KVS and VR-PBVR.

2.6. glew-2.1.0-win32

This is a GLEW binary package for Windows and is required to build and run VR-KVS and VR-PBVR.

2.7. ovr_sdk_win_1.30.0_public

Oculus SDK is required to build VR-PBVR.

2.8. cmake-3.16.3-win64-x64

Assimp build requires cmake 3.16 or higher.

2.9. imgui-1.79

Dear imgui is an open source C++ library that generates a 2D GUI on the HMD and is required to build VR-PBVR.

3. Setup

This chapter describes the pre-configuration and procedures for compiling VR-PBVR, as well as the pre-configuration of the filter and server programs required to run VR-PBVR. These operations need only be performed once for the first time.

In the following, the directory where the work starts is denoted as <BASE_DIR>.

3.1. Visual C++ Setting

To begin, download and install Visual C++ 2017/2019 (Community Edition or higher required) from the Microsoft website (https://visualstudio.microsoft.com/ja/vs/older-downloads/). Next, start Visual Studio Installer and select "Advanced ▼" -> "Change" (Figure 3.1-1).

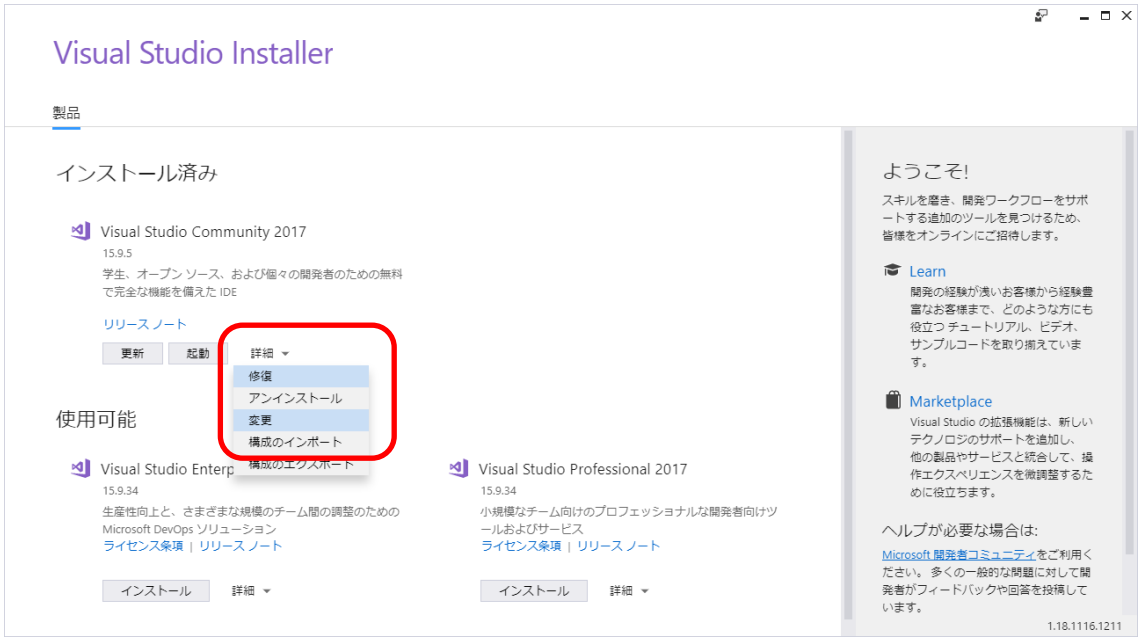

Figure 3.1-1 Advanced ▼ -> Change

On the component selection screen, install the following components (minor versions may be different).

(A) Select “Workload” tab

(1) Select “Desktop Development with C++” ()

(B) Select “Individual components” tab

(1) Windows 10 SDK (10.0.17763.0) (Figure 3.1-3)

(2) Windows 10 SDK for Desctop C++(10.0.16299.0)[x86 or x64] (Figure 3.1-3)

(3) Visual C++ tools for CMake(Figure 3.1-4)

(4) VC++ 2017/2019 (Figure 3.1-4)

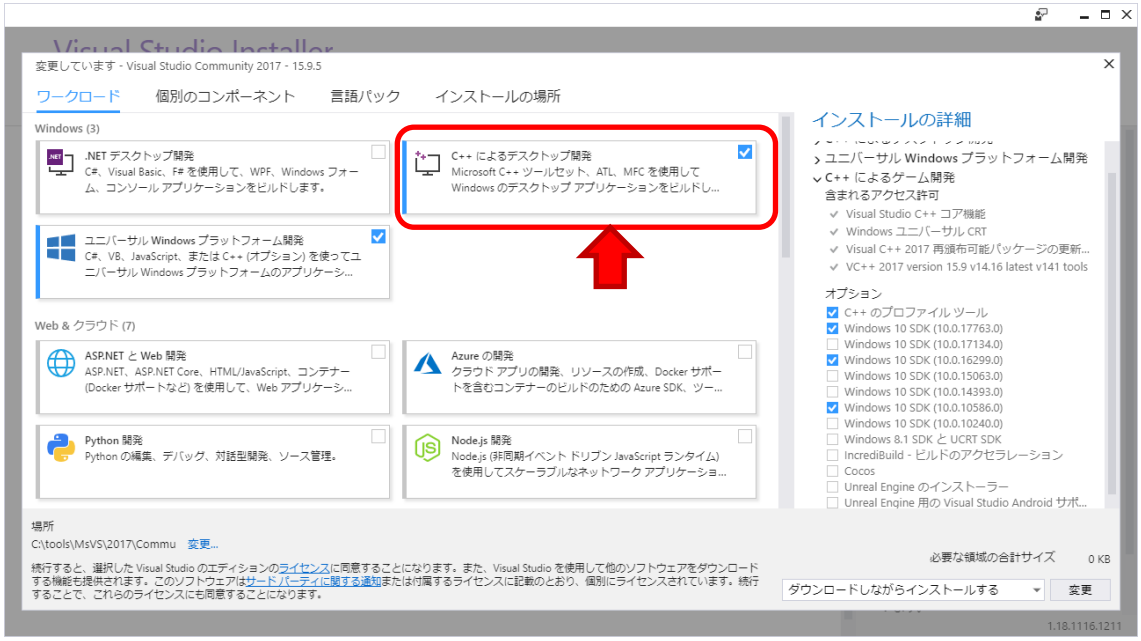

Figure 3.1-2 Desktop Development with C++

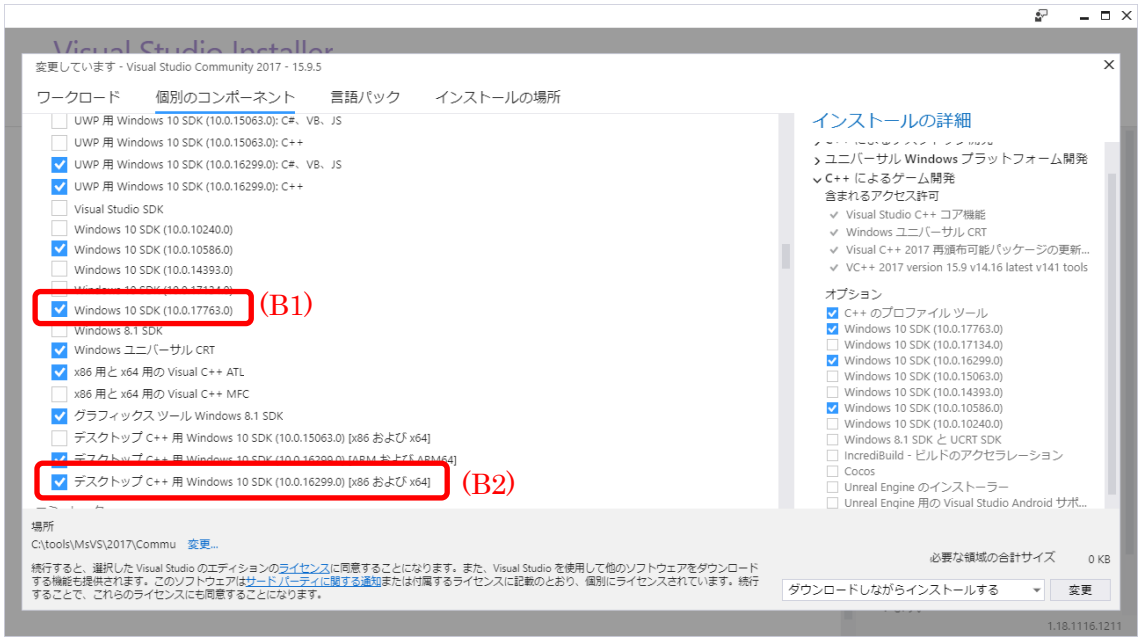

Figure 3.1-3 Individual components - Windows SDK

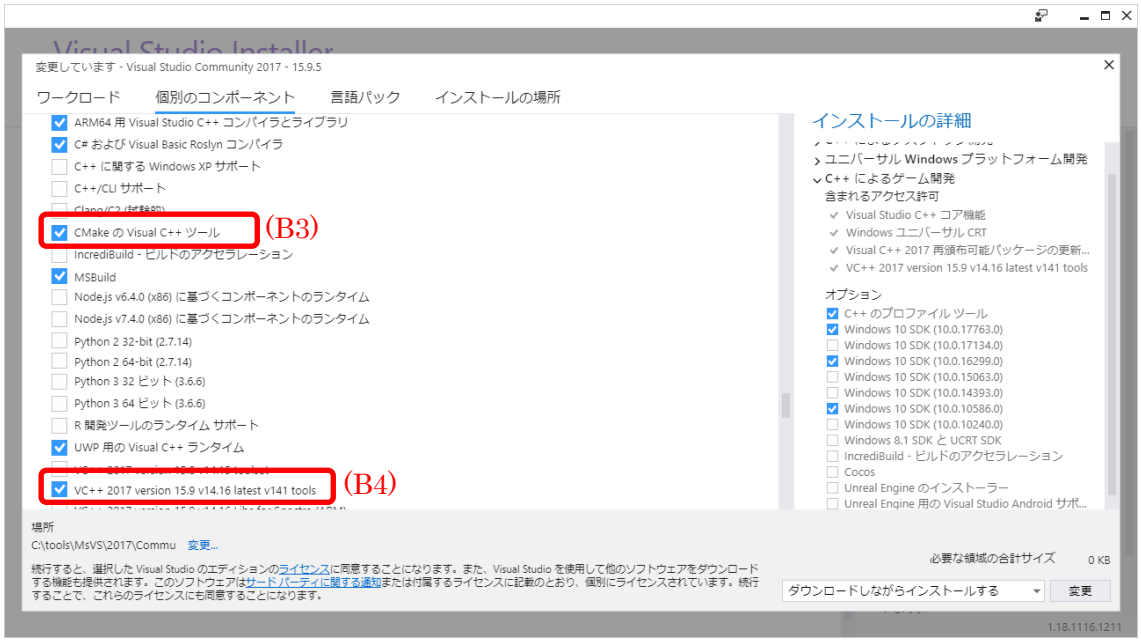

Figure 3.1-4 Visual C++ tools for CMake and VC++ tools version 15.9

After selecting components, click the "Change" button in the lower right corner to install.

3.2. Qt Creator Setting

Download and run the Qt Open Source installer (Qt Online Installer https://qt.io/download-open-source) from the official Qt website.

In the Select categories, uncheck "latest release" and "Preview" and leave only "LTS" checked (Figure 3.2-1).

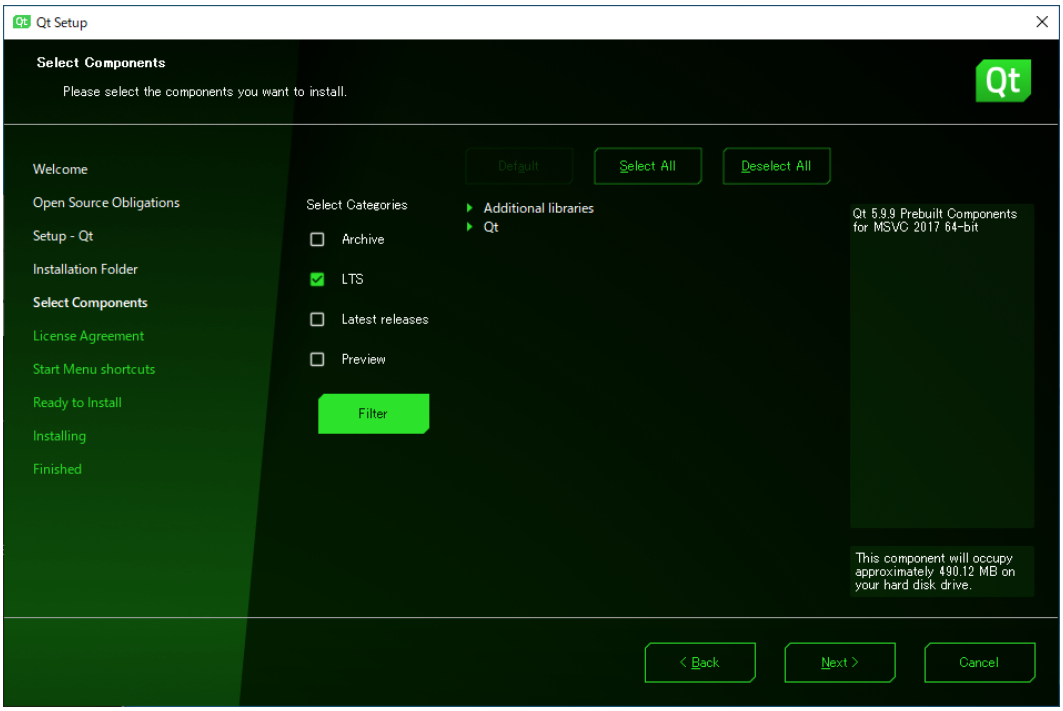

Figure 3.2-1 Qt installer, “Select category”

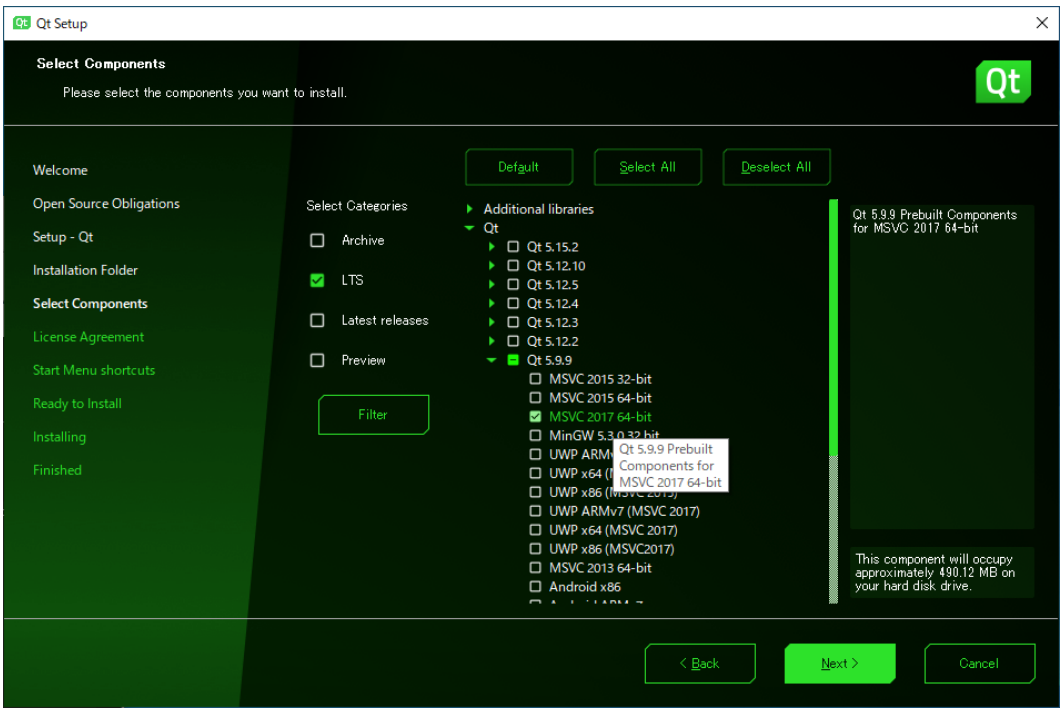

Figure 3.2-2 Qt installer, components

Then click "▸ Qt" and select a component from the table below (Figure 3.2-2).

- Qt > Qt 5.9.9 > MSVC 2017/2019 64-bit

- Qt > Qt 5.9.9 > Sources
- Qt > Developer and Designer Tools > Qt Creator 4.14.0
- Qt > Developer and Designer Tools > Qt Creator 4.13.2 CDB Debuger Support
- Qt > Developer and Designer Tools > Debugger Tools for Windows
- Qt > Developer and Designer Tools > CMake 3.18.3 64-bit
- Qt > Developer and Designer Tools > ninja 1.10.0

3.3. CMake

Decompress cmake-3.16.3-win64-x64.zip into the "cmake-3.16.3-win64-x64" folder and move it into the "<BASE_DIR>¥tools¥" folder.

3.4. KVS2.9mod4meta and Dependency

3.4.1. FreeGLUT

Decompress the "freeglult-MSVC-3.0.0-2-mp.zip" into the "freeglut" folder and move it into the "<BASE_DIR>¥lib¥" folder. Copy "freeglut.lib" in the "freeglut¥lib¥x64¥" folder to the "<BASE_DIR>¥lib¥" folder.

Set the user environment variable "KVS_GLUT_DIR" to the value "<BASE_DIR>¥lib¥freeglut".

3.4.2. GLEW

Decompress glew-2.1.0-win32.zip into the "glew-2.1.0" folder and move it to the "<BASE_DIR>¥lib¥" folder. Copy "glew32.lib" and "glew32s.lib" in the "glew-2.1.0¥lib¥Release¥x64¥" folder to the "<BASE_DIR>¥lib¥" folder.

Set the user environment variable "KVS_GLEW_DIR" to the value "<BASE_DIR>¥lib¥glew-2.1.0".

3.4.3. KVS2.9mod4meta

Set the user environment variable "KVS_DIR" to the value "<BASE_DIR>¥lib¥kvs" KVS will be installed in this folder.

Decompress the KVS2.9mod4meta.zip into the "KVS2.9mod4meta" folder and move it to the "<BASE_DIR>" folder. Launch the x64 Native Tools command prompt for Visual Studio (hereinafter referred to as "Tools command prompt"). The version of Visual Studio is replaced as needed.

On the Tools command prompt, navigate to the "<BASE_DIR>¥KVS2.9mod4meta" folder and execute the following command.

```
nmake
```

```
nmake install
```

Add the value "%KVS_DIR%¥bin " to the user environment variable "Path".

3.5. CGFormatExt4KVS and Dependency

3.5.1. Assimp

Decompress the "assimp-5.0.0.0.zip" as "assimp-5.0.0" folder and move it to the "<BASE_DIR>¥lib¥" folder.

At the Tools command prompt, move to the "<BASE_DIR>¥lib¥assimp-5.0.0.0" folder and execute the following command. The version of Visual Studio is replaced as needed.

```
> cd assimp-5.0.0

> SET CMAKE_BIN=<BASE_DIR>¥tools¥cmake-3.16.3-win64-x64¥bin¥cmake.e
  xe

> SET SOURCE_DIR=.

> SET GENERATOR=Visual Studio 15 2017

> SET BINARIES_DIR="./BINARIES/x64"

> SET CMAKE_GENERATOR=Visual Studio 15 2017

> SET CMAKE_GENERATOR_INSTANCE=C:¥Program Files (x86)¥Microsoft Visu
  al Studio¥2017¥Community

> %CMAKE_BIN% CMakeLists.txt -G "%GENERATOR%" -A x64 -D CMAKE_GENERA
  TOR_INSTANCE="%CMAKE_GENERATOR_INSTANCE%" -D CMAKE_GENERATOR="%CM
  AKE_GENERATOR%" -S %SOURCE_DIR% -B %BINARIES_DIR%

> %CMAKE_BIN% --build %BINARIES_DIR% --config debug

> %CMAKE_BIN% --build %BINARIES_DIR% --config release
```

Copy

<BASE_DIR>¥lib¥assimp-5.0.0¥BINARIES¥x64¥include¥assimp¥config.h

to

<BASE_DIR>¥lib¥assimp-5.0.0¥include¥assimp¥config.h

after building, the following library files are generated in “assimp-5.0.0¥BINARIES¥x64¥code¥Release / Debug¥”.

- Release
  - assimp-vc141-mt.lib
  - assimp-vc141-mt.dll
- Debug
  - assimp-vc141-mtd.lib
  - assimp-vc141-mtd.dll
  - assimp-vc141-mtd.pdb

Set following two environmental variables.

Variables	Values
ASSIMP_INC_DIR	<BASE_DIR>¥lib¥assimp-5.0.0¥include
ASSIMP_LIB_DIR	<BASE_DIR>¥lib¥assimp-5.0.0¥BINARIES¥x64¥code¥Release

3.5.2. Autodesk FBX SDK

Install the FBX SDK in <FBX_SDK_DIR> using Autodesk_FBX_Review_Win_64bit.exe. Then set the following two user environment variables. The version of Visual Studio is replaced as needed.

Variables	Values
FBX_SDK_INC_DIR	<FBX_SDK_DIR>¥include
FBX_SDK_LIB_DIR	<FBX_SDK_DIR>¥lib¥vs2017¥x64¥release

3.5.3. CGFormatExt4KVS

Rename "kvsmake_libs.vc.conf_template" in the "CGFormatExt4KVS" folder to "kvsmake_libs.vc.conf" and set the library path to "FBX_SDK_DIR" and "ASSIMP_DIR" respectively. This path may contain spaces, so it should be enclosed in " ".

If Assimp and FBX are unneseccary, edit variables defined in kvsmake_libs.vc.conf as following.

```
CGFORMATEXT4KVS_SUPPORT_FBXSDK = 0
CGFORMATEXT4KVS_SUPPORT_ASSIMP = 0
```

At the Tools command prompt, go to the "<BASE_DIR>¥lib¥cGFormatExt4KVS¥Lib" folder and execute the following command.

```
kvsmake lib
```

After the command execution, "LibCGFormatExt4KVS.lib" will be created in the "<BASE_DIR>¥Lib¥CGFormatExt4KVS¥Lib" folder.

Set the environment variable "CGFORMAT_EXT4KVS_SHADER_DIR" to the value

"<BASE_DIR>¥lib¥CGFormatExt4KVS¥Lib

3.6. VR-PBVR and Dependency

3.6.1. Oculus SDK

Create the folder "ovr_sdk_win_1.30.0_public" in the "<BASE_DIR>¥lib" folder. unzip the ovr_sdk_win_1.30.0_public.zip and copy all files and folders in it into the "<BASE_DIR>¥lib¥ Copy all the files and folders in the "<BASE_DIR>¥lib¥1.30.0_public¥" folder.

The version of Visual Studio is replaced as needed.

Then set the following two user environment variables

変数名	変数値
OCULUS_INC_DIR	<BASE_DIR>¥lib¥ovr_sdk_win_1.30.0_public¥LibOVR¥Include
OCULUS_LIB_DIR	<BASE_DIR>¥lib¥ovr_sdk_win_1.30.0_public¥LibOVR¥Lb¥Windows¥x64¥Release¥VS2017

3.6.2. Dear ImGui

Decompress imgui-1.79.zip into the "imgui-1.79" folder and move it to the "<BASE_DIR>¥lib" folder. imgui-1.79_cmakefiles.zip into the "imgui-1.79" folder and move it to the "<BASE_DIR>¥lib Move it to the "<BASE_DIR>¥imgui-1.79" folder.

At the Tools command prompt, move to the "<BASE_DIR>¥lib¥imgui-1.79" folder and execute the following command. The version of Visual Studio is replaced as needed.

```
cmake -G "Visual Studio 15 2017" -A x64 .
cmake --build . --config release
copy Release¥libimgui.lib .
copy examples¥Release¥libimgui_impl_opengl3.lib .
```

After the command execution, "libimgui.lib" and "libimgui_impl_opengl3.lib" is created in the "<BASE_DIR>¥lib¥imgui-1.79" folder.

3.6.3. CS-IS-PBVR

Please clone the source code of CS-IS-PBVR from
github (https://github.com/CCSEPBVR/CS-IS-PBVR)
to <BASE_DIR>. In order to build source code with VR mode, edit config file in ¥CS-IS-PBVR¥QtClient¥qtpbvr.conf as following.

3.6.3.1. Project settint

Open the project file "<BASE_DIR>¥CS-IS-PBVR¥QtClient¥QtClient.pro" in

QtCreator. On the project settings screen, click "Desktop Qt 5.9.9 MSVC2017 64bit" > "Build" under "Buld & Run" on the left sidebar to display "Build Settings" (Figure 3.6-1).

The version of Visual Studio is replaced as needed.

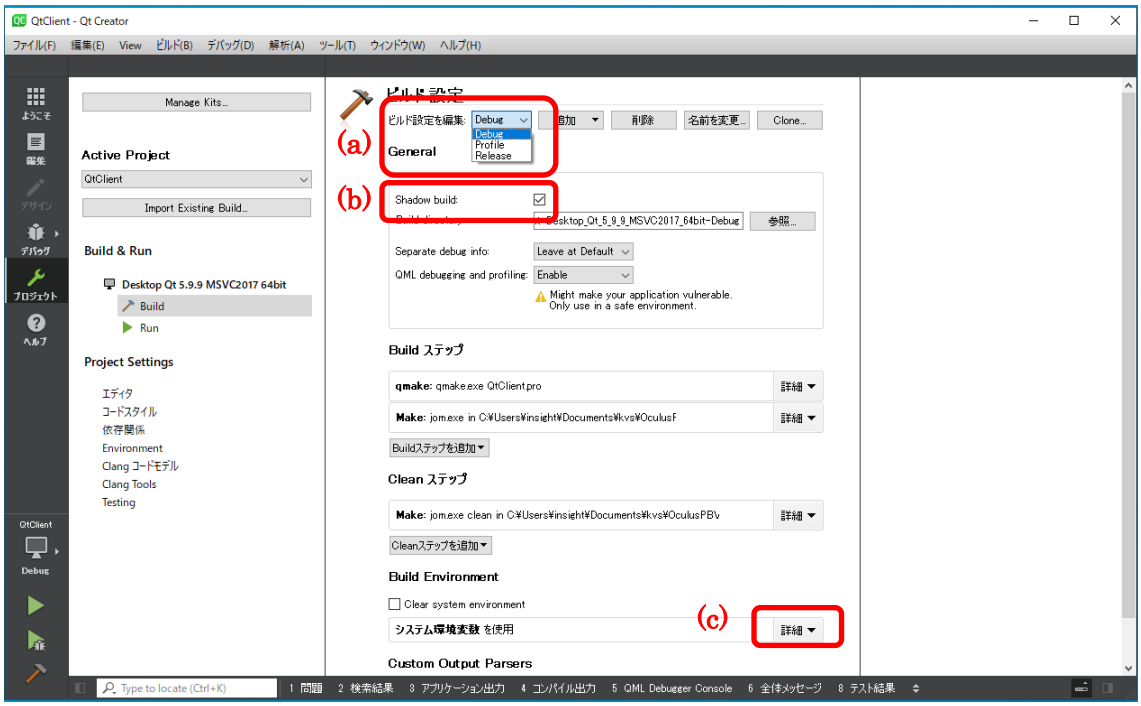

Figure 3.6-1 QtCreator build setting

(1) Build Setting

Select "Release" in the "Edit build configuration" (Figure 3.6-1 (a)).

(2) General > Shadow Build

Check "General > Shadow Build" (Figure 3.6-1 (b)).

(3) Build Environment

Click on "Details" (Figure 3.6-1 (c)) under "Build Environment" and add the following environment variables.

Variable	Value
GLEW_DIR	<BASE_DIR>¥lib¥glew-2.1.0
KVS_SOURCE	<BASE_DIR>¥KVS2.9mod4meta
IMGUI_DIR	<BASE_DIR>¥lib¥imgui-1.79

3.6.3.2. Configuration

The configuration of VR-PBVR can be set by the value of the DEFINES variable in the SETTINGS.pri file.

(1) Mirroring

VR-PBVR mirrors the HMD image to the GUI on the display; the default value of DEFINES is (a).

(a) DEFINES += MIRROR_IMAGE_BOTH_DISTORTED
For binocular images with distortion

(b) DEFINES += MIRROR_IMAGE_BOTH
For binocular images without distortion

(c) DEFINES += MIRROR_IMAGE_LEFT_ONLY
For left eye image

(d) DEFINES += MIRROR_IMAGE_RIGHT_ONLY
For right eye image

(2) Coordinate Transformation

The Touch controller grabs the visualization object and transforms the coordinates of the object with its movement. The grabbing motion is tied to the Touch controller's HandTrigger or IndexTrigger. The default value of DEFINES is (a).

(a) DEFINES += GRAB_USING_HAND_TRIGGER
Define a grabbing action on the middle finger button (HandTrigger).

(b) DEFINES += GRAB_USING_INDEX_TRIGGER
Define the action of grabbing on the index finger button (IndexTrigger).

(3) Debug Log output

Output processing logs of Scene, Screen, and TouchController classes to standard output.

(a) Scene class: “DEFINES += DEBUG_SCENE”

(b) Screen class : “DEFINES += DEBUG_SCREEN”

(c) TouchController class : “DEFINES += DEBUG_TOUCH_CONTROLLER”

3.6.3.3. Build

Menue bar > Build > Build project “QtClient”

3.7. Server Program

CS-PBVR consists of the filter program that divides large data into sub-regions for efficient parallel processing, the Particle Sampler that converts volume data into particles for visualization on the server, and the Particle Renderer that renders the volume on the user PC. VR-PBVR is a VR extension of the Particle Renderer, and the filter program and Particle Sampler are common to CS-PBVR. For details, please refer to CS-PBVR manual (https://ccse.jaea.go.jp/software/PBVR/) for more details.

4. Launching VR-PBVR

VR-PBVR uses a common server with CS-PBVR. This chapter describes how to start Particle Renderer of VR-PBVR. Please refer to the manual of CS-PBVR for how to use the server.

VR-PBVR is started from the project file." Command line arguments can be entered from the "Run-Time Settings" screen opened from "Buld & Run" > "Desktop Qt 5.9.9 MSVC2017 64bit" > "Run". The version of Visual Studio is replaced as needed. VR-PBVR is run from the button under the left sidebar or from the menu bar. The following options are available as command line arguments

- `-vin (Specify the visualization data file name)`
- `-tf (Specify the transfer function file name)`
- `-lefty (Left-handed configuration)`
- `-cgmodel <FBX file or 3ds file> (Display FBX or 3ds files)`
- `Other options available, same as CS-PBVR.`

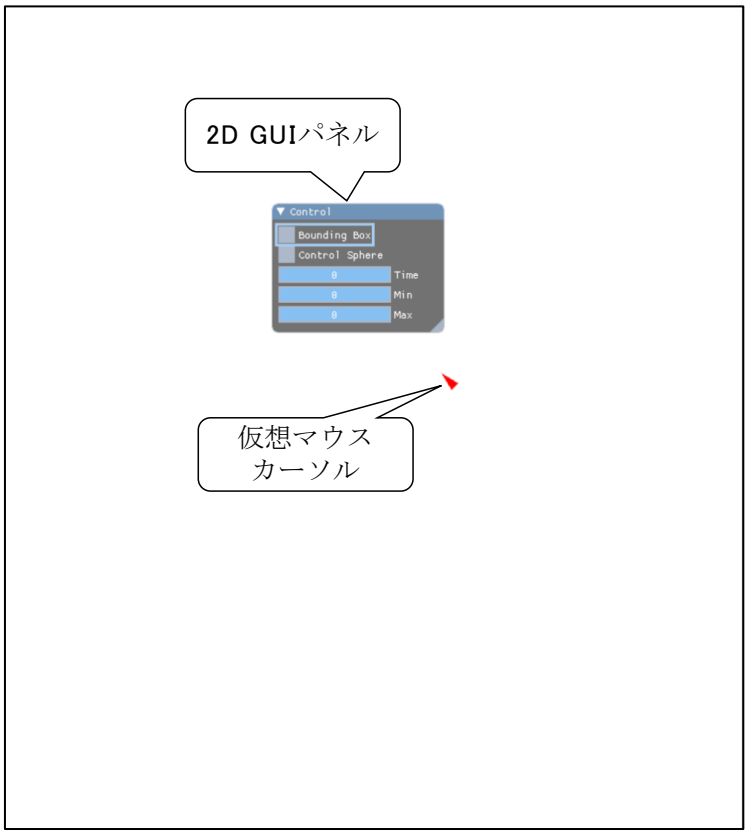

Figure 3.7-1 Control panel and mouse cursor immediately after startup

Immediately after execution, a control panel displayed in layover and a virtual mouse cursor for manipulating it are drawn in the 3D space (Figure 3.7-1).

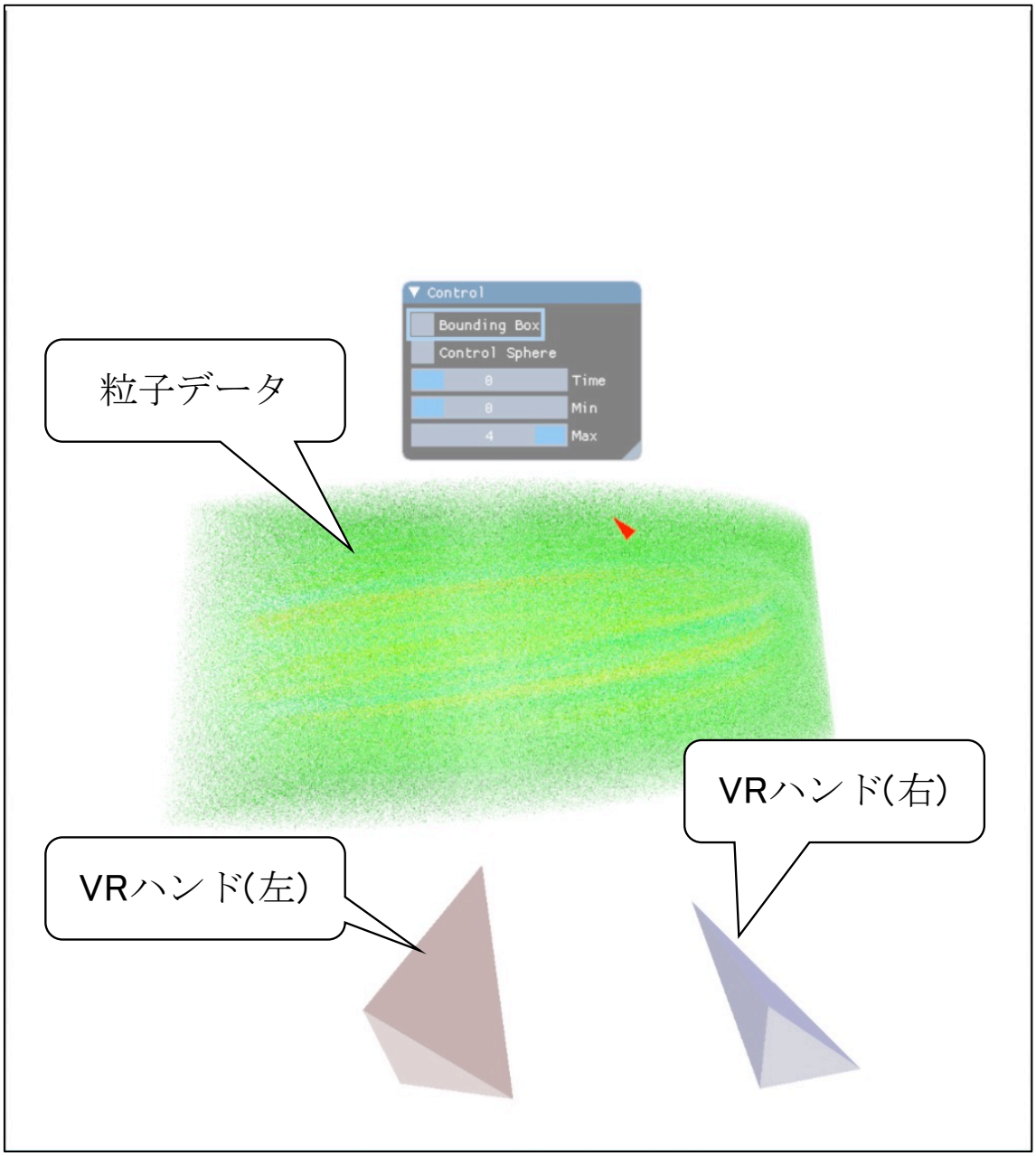

Figure 3.7-2 Visualization data and VR hand

The renderer receives particle data from the server and draws visualization data and VR hands (Figure 3.7-2).

5. Operation in VR space

5.1. Control Panel

The control panel can be operated with the red triangular mouse cursor. In right-handed mode, the mouse cursor is moved with the left stick of the Touch Controller and clicked with the A button (Figure 5.1-1). In left-handed mode, the mouse cursor is moved with the right stick and clicked with the X button (Figure 5.1-2).

Figure 5.1-1 Right-handed mode (without "-lefy" option)

Figure 5.1-2 Left-handed mode (with "-lefy" option)

The control panel (Figure 5.1-3) provides the following functions

(1) Toggling the bounding box display On/Off (Figure 5.1 4)

(2) Toggle control sphere display On/Off (Figure 5.1 5)

(3) Time step setting

(a) Designation of display step

(b) Designation of minimum step

(c) Designation of maximum step

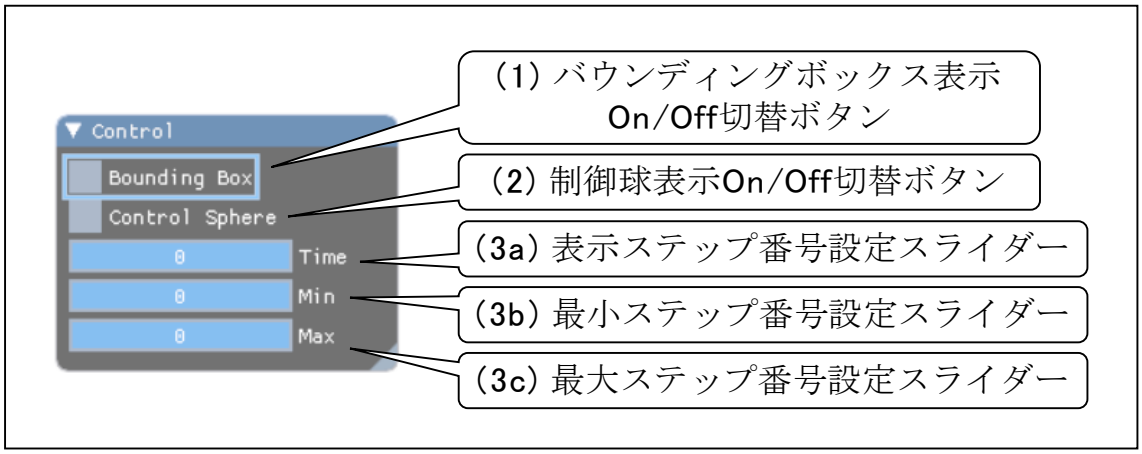

Figure 5.1-3 GUI panel operation items

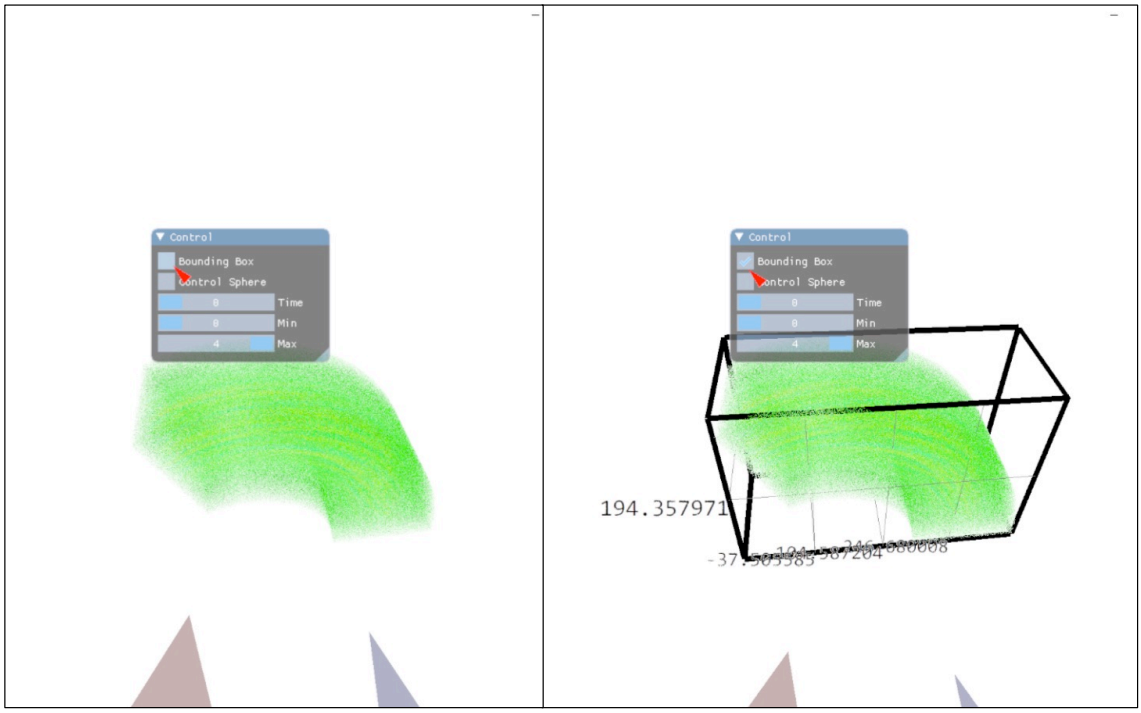

Figure 5.1-4 Bounding box display Off (left) and On (right)

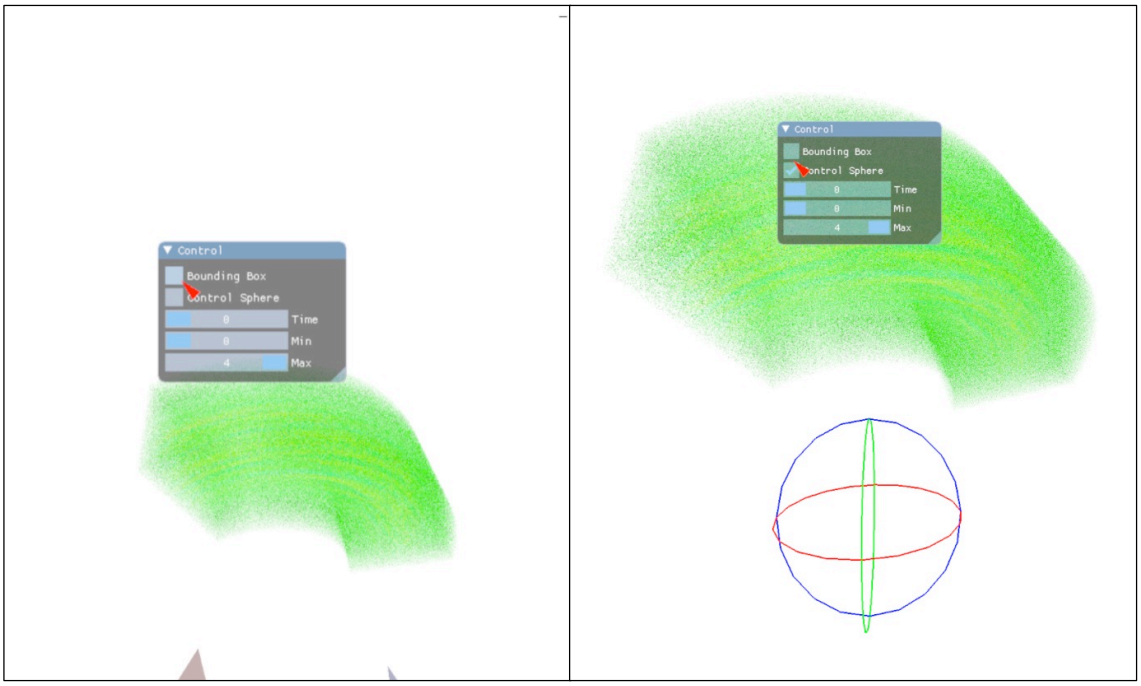

Figure 5.1-5 Control ball display Off (left) and display On (right)

For time steps, the top slider is the display step setting, the next is the lower limit of steps that can be displayed, and the bottom is the upper limit of steps that can be displayed. The time steps on the control panel and the GUI on the display are interlocked (Figure 5.1-6).

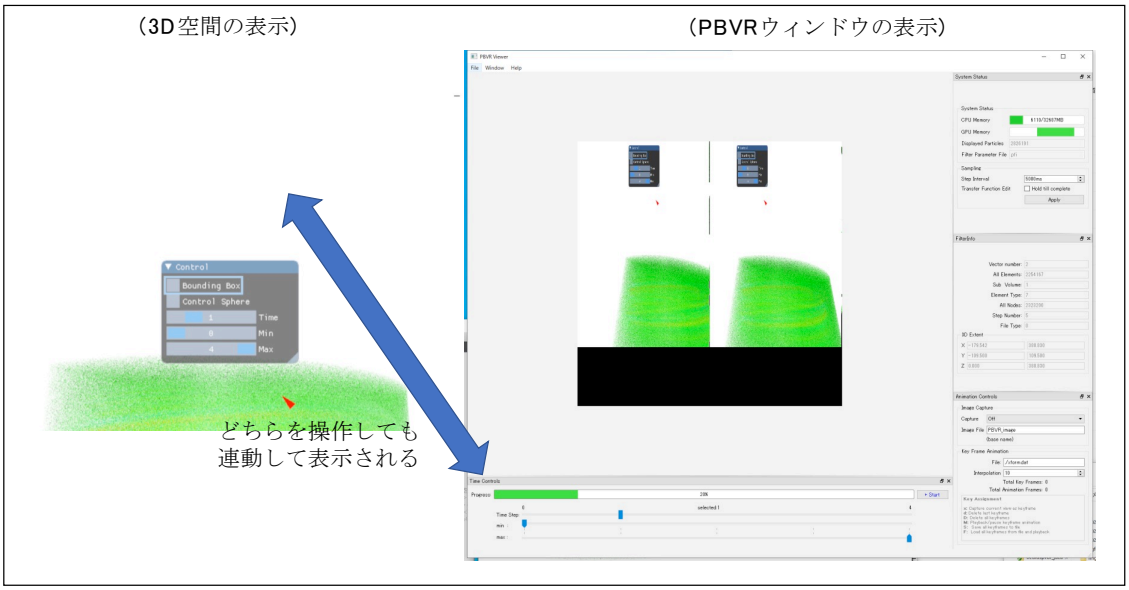

Figure 5.1-6 VR space and GUI on display

5.2. Coordinate Transformation

The object is grabbed by the trigger button on the Touch controller and the coordinates are transformed by the gesture. The trigger button is initially HandTrigger (Figure 5.2-1).

Figure 5.2-1 Trigger button position

Gestures that move both hands in the same direction cause the object to move in parallel (Figure 5.2-2).

Figure 5.2-2 Parallel translation

Gestures of releasing or moving both hands closer together cause the object to expand or contract (Figure 5.2-3).

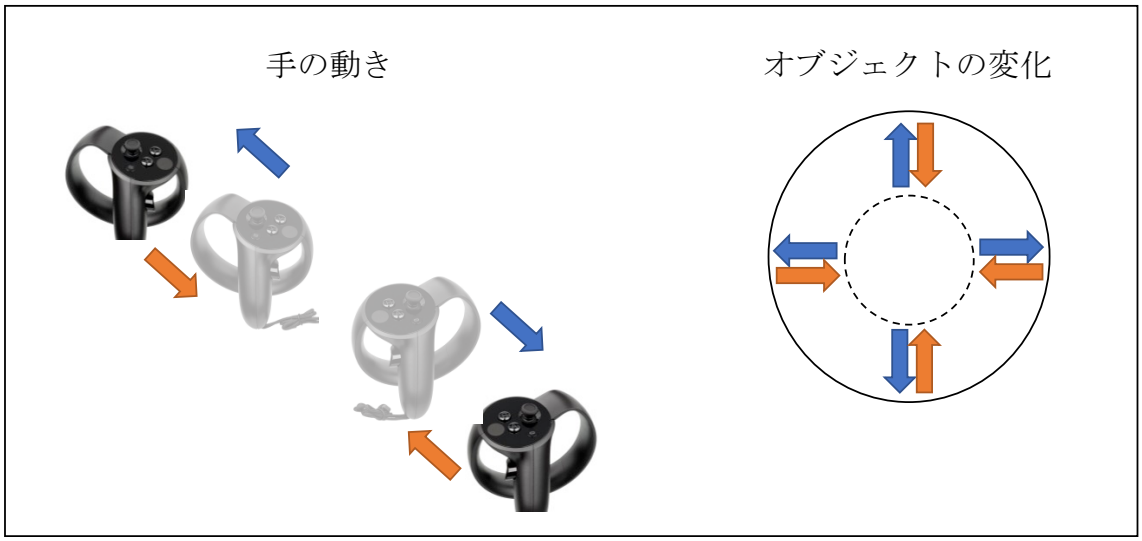

Figure 5.2-3 Scaling

The gesture of turning both hands rotates the object (Figure 5.2-4).

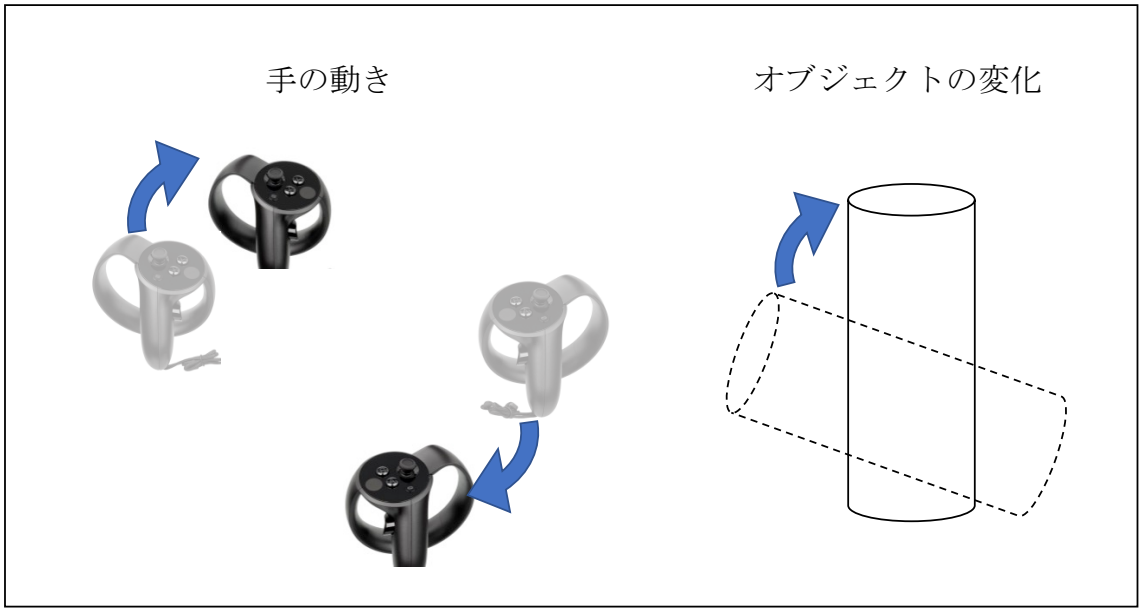

Figure 5.2-4 Rotation

6. OculusPBVR_JAEA の終了

The client exits by pressing the X button in the upper left corner of the main window on the display. The server terminates by typing Ctrl+C on the terminal.