2010年度
卒業論文

VR可視化のための
オープンソースソフトウェア開発

神戸大学工学部情報知能工学科

目野大輔

指導教員　陰山聡

2011年2月23日

VR可視化のためのオープンソースソフトウェア開発

目野大輔

要旨

スーパーコンピュータから出力されるデータは膨大であり、そのサイズは今後もさらに増大することが確実である。単なる数値の羅列に過ぎないその出力データから意味のある情報を引き出すためには、可視化のプロセスが必要不可欠である。スーパーコンピュータの発展に伴い、可視化技術も盛んに研究されてきた。中でも没入型のバーチャルリアリティ(VR)技術を応用した可視化手法は、人間の優れた空間認識能力を活かした解析が可能となるため、現在までに様々な研究がなされている。しかし、没入型VR装置はその開発から長い年月が経過しており、基本ライブラリや、それに基づく可視化ソフトウェアは古い設計に起因する様々な限界に直面している。そこで本研究では、新しいVR用ライブラリを利用した新しい可視化ソフトウェアの開発に着手した。

検討の結果、新しいVR用ライブラリとしてVR Jugglerを採用した。VR Jugglerは、VRシステムを抽象化するバーチャルプラットフォーム(VP)という革新的な特徴を備えている。VPには、VRシステムに依存しないアプリケーションを製作できるメリットがある。また、VR Jugglerは現在でも精力的に開発が続けられており、しかも無償である。そこでVR Jugglerが、新しい可視化ソフトウェアにふさわしいVR用ライブラリであると判断した。VR Jugglerに基づく新しい可視化ソフトウェアを現在設計中である。

目 次

1 序論 **1**

2 可視化とバーチャルリアリティ **3**
2.1 可視化とは 3
2.2 バーチャルリアリティとは 3

3 バーチャルリアリティシステム CAVE **4**
3.1 CAVE: Cave Automatic Virtual Environment 4
3.2 CAVElib 5
3.3 VFIVE: Vector Field Intaractive Visualization Environment . . . 5
3.4 pCAVE 6

4 VR Juggler とそれに基づく可視化ソフトウェアの開発 **10**
4.1 VR Juggler とは 10
4.2 VR Juggler の特徴 10
4.2.1 VR Juggler アーキテクチャ 11
4.2.2 バーチャルプラットフォーム 12
4.2.3 アプリケーションオブジェクト 13
4.2.4 使用可能な基本 API 13
4.2.5 マルチユーザー 14
4.3 VR Juggler プログラミング 14
4.4 VR Juggler における入力デバイスの取り扱い 16
4.5 VR Juggler に基づく可視化ソフトウェア 18
4.5.1 開発の目的 18
4.5.2 想定される問題点とその克服方法 18
可視化ソフトウェアの基本的な機能について 18
ユーザーインターフェース 19
4.5.3 実装すべき機能 19

5 まとめ **20**

6 謝辞 **21**

A VR Juggler のインストールとコンフィギュレーション **23**
A.1 VR Juggler のインストール 23
A.2 VR Juggler のコンフィギュレーション 24
A.3 入力デバイスの設定 28

1　序論

近年、スーパーコンピュータを用いた大規模な 3 次元シミュレーションが科学研究及び技術開発において中心的な役割を果たしつつある。スーパーコンピュータの発達はめざましく、それと比例してスーパーコンピュータから出力される 3 次元データも巨大なものとなっている。出力される 3 次元データは単なる数値の羅列であり、そのままではシミュレーション結果を理解することは到底不可能である。

そこで 3 次元数値データを 3 次元オブジェクトに変換し、画像として表示する 3 次元可視化技術が注目されている。人間には優れた空間認識能力が備わっており、その能力を活かすことで効率よくデータを解析することが可能になる。しかしながら、せっかく 3 次元オブジェクトを生成しても普通のディスプレイに表示したのでは 2 次元画像としてしか見ることができないため、人間の優れた空間認識能力を活かし切ることができない。また、人間の空間認識能力は視覚情報だけではなく、聴覚やその他複数の感覚器の協力のもとに成り立っているので、視覚情報しか与えられない立体視対応ディスプレイに表示したとしても、空間認識能力を十分に活かすことはできない。マウスやキーボードなどのユーザーインターフェースは 2 次元画面を操作するには有効なデバイスであるが、3 次元空間を操作するには貧弱である。

この問題はバーチャルリアリティ (VR) 技術を用いることで解決できる。VR では立体視するだけでなく、ヘッドトラッキング装置により視点の位置や視線の方向を測ることができ、それに応じた画像をリアルタイムに生成することによって、あたかも実際にそのデータの中に潜り込んでいるかのように解析することが可能となる。VR 空間との対話に使用するコントローラにもトラッキング装置が装備してあるため、直感的に 3 次元空間中の位置や方向を指し示すことが可能である。

VR 装置にはいくつか種類があるが、本研究では CAVE[1](Fig.1) と呼ばれる VR 装置を利用する。

Fig. 1: CAVE

CAVE はイリノイ大学シカゴ校で開発された VR 装置で、10 フィート四方のスクリーンを床と壁面に配置して小さな部屋を作り、それらに立体映像を投影することで VR 空間を構成するものである。CAVE 装置の詳細については 3 章で述べる。

スクリーンのサイズや配置の方法などは自由であり、多種多様な CAVE 装置が考案され使用されているが、その多くが CAVElib という基本 API を使用しているという点で共通している。CAVElib は VR 空間を構成する上で発生する複雑な処理を自動で行い、さらに VR 空間をより豊かにするためのヘッドトラッキング情報やコントローラの情報を提供するライブラリである。CAVElib は非常に優れたライブラリであるが、有償であることが CAVElib を使用した可視化ソフトウェアを広く普及させようとした場合に足かせとなってしまうことがある。また、設計から約 20 年が経過しており、最新グラフィクス技術や最新プログラミング技法との折り合いが悪くなりつつある。そのため、カリフォルニア大学デービス校の KECK CAVES[2] のように独自開発の API を使用している CAVE 装置もあるが、あまり広まってはいないのが現状である。CAVElib に代わる新しく、しかも無償かつライセンスの緩い VR 空間構築用ライブラリを使用した可視化ソフトウェアを開発できれば、あらゆる CAVE 装置への普及が期待できる。

本研究の目的は、最新かつ無償の VR 空間構築用ライブラリを使用した 3 次元可視化ソフトウェアを開発することである。開発したソフトウェアを最終的にオープンソース化することで、多くの研究者に活用してもらうことを目指す。新しい VR 空間構築用ライブラリはいくつか存在するが、第 4 章で述べるような様々な利点を鑑みて、VR Juggler[3] という VR 空間構築用ライブラリを採用することにした [4]。

2　可視化とバーチャルリアリティ

この章では可視化とバーチャルリアリティの説明、及びこれらの 3 次元可視化における重要性について説明する。

2.1　可視化とは

可視化とは目で捉えることができないものを見えるようにしたり、そのままでは理解しづらいもの理解しやすくするプロセスのことである。3 次元シミュレーション結果を見える形に変換することによって現象をより深く理解することが可能となる。いずれも、人間の目と脳の能力を有効に活用するために行われる。スカラー場の可視化手法には等値面表示やボリュームレンダリングなどがある。ベクトル場の可視化手法にはパーティクルトレーサーなどがある。スカラー場に関しては多種多様な可視化手法が確立されているが、それと比較してベクトル場の可視化についてはまだ研究の余地が多く残されている。

2.2　バーチャルリアリティとは

バーチャルリアリティとは、人工現実環境を計算機によって作り出す技術である。バーチャルリアリティを構成するには、没入感、フィードバックおよび対話性などの要素が必要である。良く知られたヘッドマウントディスプレイ装置に代表されるように、視覚のみを利用したバーチャルリアリティが一般的である。その理由は人間が感覚器官から得る情報のうち、視覚の占める割合が大きいからであろう。最近では聴覚、臭覚、触覚などを利用したバーチャルリアリティの研究も盛んである。

本研究では、人間の空間認識能力を最大限引き出すために、単なる立体視ディスプレイではなくバーチャルリアリティ装置を用いる。特に、他のバーチャルリアリティ装置と比較して、没入性と対話性が優れている CAVE 型バーチャルリアリティ装置を用いる。

3　バーチャルリアリティシステム　CAVE

この章では本研究で利用したバーチャルリアリティ装置 CAVE について説明する。これに関連して、VR 空間構築用ライブラリである CAVElib と、CAVElib を使用した可視化ソフトウェアである VFIVE[5] についても説明する。

3.1　CAVE: Cave Automatic Virtual Environment

CAVE はイリノイ大学シカゴ校で開発されたバーチャルリアリティ装置である。オリジナルの CAVE 装置は 10 フィート四方のスクリーンを壁面に 3 枚、床面に 1 枚設置しそれらにステレオプロジェクタで立体映像を投影することで VR 空間を構成する。ヘッドトラッキング装置によってユーザーの位置や頭の位置と方向を検出することができ、その情報をもとに適切な画像が生成される。ワンドと呼ばれるコントローラを介して VR 空間と対話的にやりとりすることができる。典型的な CAVE 装置は Fig.1 に示した通りである。

現在までに様々な形状の CAVE が開発されており、立方体の面すべてがスクリーンになっている岐阜県情報技術研究所の CAVE「COSMOS」[6]、横に長くステージのような形状をした東海大学の CAVE「HoloStage」（HoloStage はクリスティデジタルシステムズ社の登録商標である）、1 面のスクリーンに対して 4 つのプロジェクタで投影し高解像度を実現する大阪大学サイバーメディアセンターの CAVE「HOPE」[7]、タイルドディスプレイと CAVE の融合ともいえる 16 面スクリーンのカリフォルニア大学サンディエゴ校の CAVE「StarCAVE」[8] などがある。なお、タイルドディスプレイとは液晶ディスプレイを大量に敷き詰めて、大画面及び高解像度を実現するディスプレイ装置のことである。比較的安価に導入できることから最近は導入例が増えているが、没入度に欠けるため本研究では採用しない。

CAVE を構成する計算機には 1 台の高性能グラフィクスワークステーション、または PC クラスタが使用される。スーパーコンピュータによる 3 次元シミュレーション結果を可視化する目的で CAVE を新たに構成する場合には、大容量共有メモリを搭載した 1 台の高性能グラフィクスワークステーションを用いるのが望ましい。なぜならば、基本的に PC クラスタ版バーチャルリアリティシステムには、PC クラスタを構成する各ノードに同じデータを配置しなければならないという制約が存在するからである。通例、各ノードに載せられるメモリ (主記憶) は大きなものではないため、スーパーコンピュータによるシミュレーション結果の 3 次元データが、各ノードのメモリに乗り切らない可能性があるからである。

3.2 CAVElib

CAVElib は VR 空間を構成する際に必要なヘッドトラッキングやそれに応じた射影計算などを処理する基本ライブラリである。CAVElib が自動でこれらの処理を行うので、ユーザーは VR 空間に何を表示させるかという問題に集中することができる。

コンピュータグラフィックスのライブラリは OpenGL を用いる。OpenGL はコンピュータグラフィックスの標準的なライブラリである。

CAVElib は優れたライブラリであるため多くの CAVE で基本 API として採用されているが、設計から約 20 年が経過している。また、高価なこともあって今後も標準 VR 用ライブラリの地位を維持できるかどうかは定かではない。

CAVElib は初期設定により特定のシステムに対してカスタムするために、新しいデバイスを追加しようとした場合、CAVElib のコードそのものを変更しなければならない。しかし、CAVElib は商用アプリケーションであるため、ユーザーは CAVElib のコードそのものを見ることも修正することもできない。これらは、CAVElib の弱点の 1 つと言える。

CAVE システムを設計から独自開発している場合には、その一環として独自に VR 空間構築用ライブラリを開発していることが多い。実際、東京大学インテリジェントモデリングラボラトリーの CAVE システム「CABIN」[9] や、カリフォルニア大学デービス校の CAVE システム「KECK CAVES」には、それぞれ OpenCABIN や Vrui という独自開発の VR 空間構築用ライブラリが採用されている。独自開発の CAVE 装置以外にも、CAVElib 以外の VR 空間構築用ライブラリを採用している例は少なからず存在する。

3.3 VFIVE: Vector Field Intaractive Visualization Environment

VFIVE は、本研究の指導教員である陰山教授が開発した CAVE 用可視化ソフトウェアである。スカラー場の可視化機能は当然のことながら、ベクトル場の可視化機能が豊富に実装されている。他の可視化ソフトウェアと比較して、より没入的かつ対話的に可視化することが可能であり、これが VFIVE の最大の特徴である。

VFIVE は CAVElib と OpenGL を基本 API として採用しているため、大きな変更を加えることなく多くの CAVE 上で動作させることができる。実際、VFIVE は核融合科学研究所、中央大学、大阪大学、海洋研究開発機構の CAVE 装置にインストールされ、活用されている。しかし、例えばあるシミュレーション研究者が近隣施設の CAVE を利用して VFIVE によるデータ解析を行いたいと考えた場

合、その CAVE に CAVElib が搭載されていなければ VFIVE を使用することができないという大きな問題を抱えている。

3.4 pCAVE

pCAVE は神戸大学に設置されている CAVE 装置である。標準的な仕様とは異なり、1 枚のスクリーンで CAVE を構成している。アクティブシャッター方式立体視を採用しており、その液晶シャッターメガネにはヘッドトラッキング装置が装着されている。トラッキング情報や姿勢情報を取得できるコントローラが接続されており、対話的に VR 空間とやりとりすることができる。

Table.1 に pCAVE のスペックを示す。1 台のグラフィクスワークステーションに大容量共有メモリが搭載されているので、大規模データの可視化処理を高速に行うことが可能である。

Computer type	SGI Asterism adt08
CPU	AMD Opteron 2350 x2 (=8 cores)
Memory	64GB
GPU	NVIDIA Quadro FX 4600
OS	SUSE Enterprise 10.2
Projector	Christie Mirage S+4K
Controller	Intersense IS900
Tracker	Intersense IS900
Screen	3048mm x 2286mm

Table. 1: pCAVE specification

Fig.2 から Fig.7 に pCAVE の設備写真を示す。

Fig. 2: Screen of pCAVE

Fig. 3: Projector of pCAVE

Fig. 4: Controller of pCAVE

Fig. 5: Liquid crystal shutter glasses of pCAVE

Fig. 6: Graphics workstation of pCAVE

Fig. 7: Tracking workstation of pCAVE

4 VR Juggler とそれに基づく可視化ソフトウェアの開発

この章では、本研究で採用する VR 空間構築用ライブラリである VR Juggler について説明する。

4.1 VR Juggler とは

VR Juggler は CAVE を開発した Cruz-Neira 教授が中心となってアイオワ州立大学で新たに開発した VR 空間構築用ライブラリである。2001 年に発表され、現在も精力的にバージョンアップが行われており、LGPL ライセンスのもとで無償配布されている。オリジナルの CAVE に精通した人物が関わっていることが、このライブラリの特徴である。

バーチャルリアリティシステムのソリューションの一部として提供されることが多い CAVElib とは異なり、VR Juggler は基本的に既存のバーチャルリアリティシステムにユーザー自らがインストールを行う必要がある。実際、本研究を行うにあたり神戸大学の pCAVE システムに筆者自ら VR Juggler のインストールを行っている。

VR Juggler は世界中の 100 以上の CAVE で使用されているが、日本における導入事例はほとんど存在しない。そのため、VR Juggler に関する日本語での詳しい解説は存在せず、その普及の足かせとなっている。本研究を進める準備として筆者が行ったインストールに関する調査報告の結果を、付録に記載する。

4.2 VR Juggler の特徴

VR 空間構築用ライブラリの標準となるために必要な要件として、

1. 柔軟性と拡張性
2. クロスプラットフォームであること、及びクロス VR システムであること
3. オープンソースであること

などが挙げられる。VR Juggler は、これらの要件を満たす形で開発が行われている。

具体的には、

1. 複雑な VR システムの抽象化

2. ディスプレイやコンピュータ、インターフェースにおけるスケーラビリティ

3. 多様なハードウェアを定義するための拡張性

4. ハードウェアとソフトウェアを実行時に変更できる機能

5. 同時に複数のアプリケーションを実行出来る機能

を提供している。

CAVElib との違いは、

1. モノリシックカーネルアーキテクチャではなくマイクロカーネルアーキテクチャを採用していること

2. オブジェクト指向型であること

3. アプリケーションがライブラリから独立していること

4. オープンソースであること

が挙げられる。なお、モノリシックカーネルは、単層カーネルとも呼ばれており、システムに必要な機能をすべて取り込んだカーネルのことである。システムが肥大化しやすいのがその欠点である。また、マイクロカーネルは、システムに最低限必要な機能のみが実装してあるカーネルのことである。システムとしての機能は、カーネルの機能を用いたモジュール群が担当する。

簡単にまとめれば VR Juggler の最大の特徴は、VR システムを抽象化しているということである。VR Juggler は新しい標準的オープンソースソフトウェアを開発するために必要な API としての条件をあらゆる点で満たしており、本研究で採用するのにふさわしい API であると言える。以下のセクションでは VR Juggler の特徴について項目ごとに詳しく説明する。

4.2.1 VR Juggler アーキテクチャ

マイクロカーネルアーキテクチャを採用することでシステムがモジュール化され、拡張性や移植性が増している。CAVElib は CAVElib という大きな単体のシステムが稼動しているが、VR Juggler は機能ごとに独立したモジュール群が協調して動作している。具体的には以下のモジュール群が存在する。

- Gadgeteer : デバイスマネージャ。入力デバイスの制御を担当する。

- JCCL : コンフィギュレーションシステム。システムのコンフィグを担当する。

- VPR : OS に対する抽象化を担当する。
- Sonix： オーディオハードウェアやオーディオ API に対する抽象化を担当する。
- Tweek : Java による GUI プログラムと、C++アプリケーションとの通信を担当する。

これらのモジュールは Fig.8 に示すような階層構造をとっている。

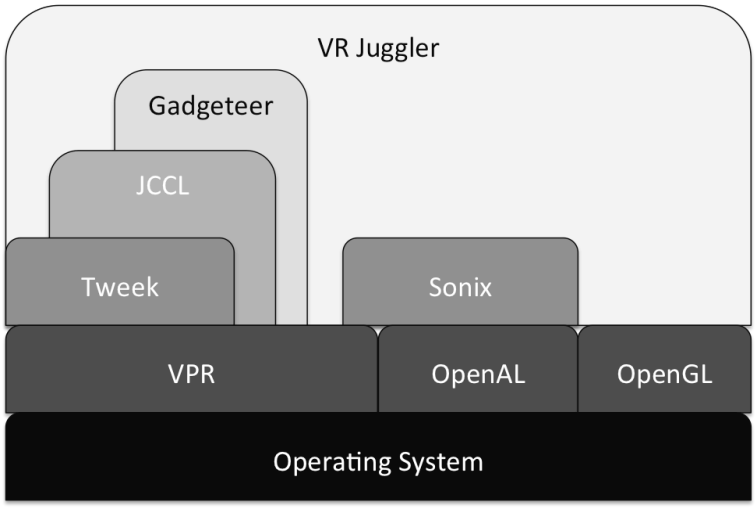

Fig. 8: Modules of VR Juggler

4.2.2 バーチャルプラットフォーム

VR Juggler ではバーチャルプラットフォーム (VP) という概念を採用することにより、クロスプラットフォームとクロス VR システムを実現している。バーチャルプラットフォームとは、ハードウェア固有の特殊な機能の隠蔽、つまり抽象化を施したものである。バーチャルプラットフォームによって、VR アプリケーションを開発する上で生じる高度なハードウェア等の知識が不要となる。つまりユーザーは VR Juggler を用いることで、ハードウェアやオペレーティングシステムに依存しないアプリケーションを作成することが可能になる。入出力デバイスが抽象化されることで、アプリケーションが特定の装置でしか動作しないということが起きないようにもなっている。VR Juggler が VR システムやオペレーティングシステム、アーキテクチャ、入力デバイスの違いをほぼ完全に吸収するため、ユーザーは VR 空間に何を表示させるかという問題に集中できるというのは大きなメリットである。

4.2.3 アプリケーションオブジェクト

現在のほとんどの CAVE システムでは静的な設定で動作しているため、実行時に別の VR アプリケーションを起動したりアプリケーションを変更することができない。しかし、VR Juggler はアプリケーションオブジェクトという概念を導入することで、その実行時に設定やアプリケーションを変更することが可能となる。アプリケーションオブジェクトは、アプリケーションのプログラムを簡潔なものとし、アプリケーションのロバスト性を高める開発技法である。アプリケーションのロバスト性とは、いかにプログラムを修正せずに多様な VR システム上で動作するか、ということを意味する。Fig.9 に示すように、従来の VR 空間構築用ライブラリにおいては、アプリケーションがライブラリの機能を呼び出す形のプログラムであるが、VR Juggler ではカーネルがアプリケーションの関数を呼び出す形のプログラム構造になっている。これによって実行時にアプリケーションを切り替えたり、アプリケーションのパラメータを変更することが可能となる。つまりパラメータ調整のために何度もアプリケーションを立ち上げ直す必要がない。また、デモンストレーションを行う際に、スムーズにアプリケーションを切り替えることができる利点もある。

アプリケーションオブジェクト型プログラムと、そうではないプログラムとでは、アプリケーションのロバスト性に対する責任の所在が異なる。前者では呼び出した側つまりライブラリに責任があるが、後者では呼び出した側つまりアプリケーションに責任がある。つまり、アプリケーションオブジェクト型プログラムの方が、アプリケーション開発者のアプリケーションのロバスト性に対する責任が軽減される。

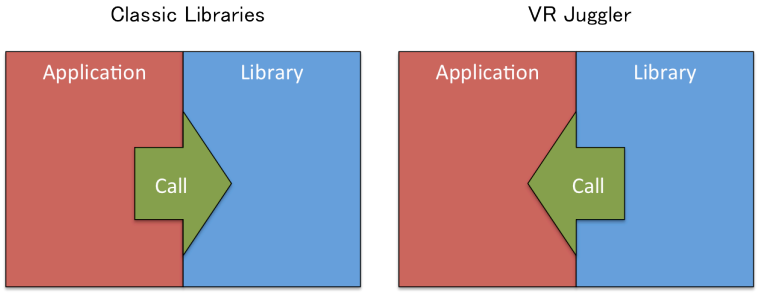

Fig. 9: Library model comparison

4.2.4 使用可能な基本 API

3D 空間を構成するグラフィクス API として OpenGL が採用されているが、グラフィクス API はカプセル化されているので、より簡単に 3D 空間を構成するた

めの上位 API である OpenSG や OpenSceneGraph を利用することも可能である。また、OpenAL による 3D サウンドの機能も搭載しており、スピーカーを設置している VR 装置で 3D サウンドによる情報の提示が可能である。OpenAL はクロスプラットフォームの 3D オーディオ API である。オーディオ API もカプセル化されてはいるが、対応しているオーディオ API の中で現在精力的に開発が続けられているのは OpenAL である。

4.2.5 マルチユーザー

VR Juggler ではヘッドトラッキングは、1 つだけでなく複数個設定することが可能である。ユーザーとディスプレイの関連を設定すれば、“スクリーンやプロジェクタごとに” という制限付きではあるが、複数のユーザーが同時に VR 空間を自由に動き回ることができる。

4.3 VR Juggler プログラミング

本研究で開発する汎用可視化ソフトウェアはオープンソースとして配布する予定である。つまりユーザーがソフトウェアを改造するということも十分想定される。一方、ソフトウェア開発、改良に不可欠な VR Juggler プログラミングの解説書は現在のところ日本語では存在しない。そこで、VR Juggler プログラミングについて以下で解説する。

VR Juggler プログラミングに必要な知識は、C++プログラミング、OpenGL などのグラフィクスプログラミングの知識、及び線形代数などある程度の数学の知識である。VR Juggler アプリケーションを作成するための煩雑な手間や作業は存在せず、グラフィクス API(OpenGL,OpenSG,OpenSceneGraph) 毎に用意されたクラスを継承し、用意されている関数に処理を記述するだけである。ここではグラフィクス API に OpenGL を採用する場合の解説を行う。OpenGL 用のクラスは主に、以下に示す関数を持っている。

- init : データの初期化、読み込み等の初めに一度だけ行う処理を記述する。
- apiInit : グラフィクス API に OpenGL を用いる場合は空白でよい。
- preFrame : 描画の前に呼ばれる。デバイスからの情報を更新を行う。
- intraFrame : 描画と同時に呼ばれる。積分などの計算を行う。その結果は次フレームに反映される。

- postFrame : 描画の後に呼ばれる。描画後のクリーンアップ、ネットワークで繋がっているマシンや計算プロセスとの同期を行う。
- bufferPreDraw : カラーバッファとデプスバッファのクリーンアップを行う。
- draw : 描画を行う。指定されない限り、OpenGL の命令はここに記述する。
- contextInit : ディスプレイリストの生成などの処理を記述する。

アプリケーションオブジェクト型の開発技法であるため、アプリケーションは main 関数を持たない。これらの関数はアプリケーションが起動した後、Fig.10 に示すサイクルで呼び出される。VR Juggler カーネルがこのサイクルを厳密に行うことを保証しているので、アプリケーションが何も処理を行っていない “安全” なタイミングが生まれる。その “安全” なタイミングの存在が、コンフィギュレーションを変更したりアプリケーションを切り替えたりできることの保証となっている。

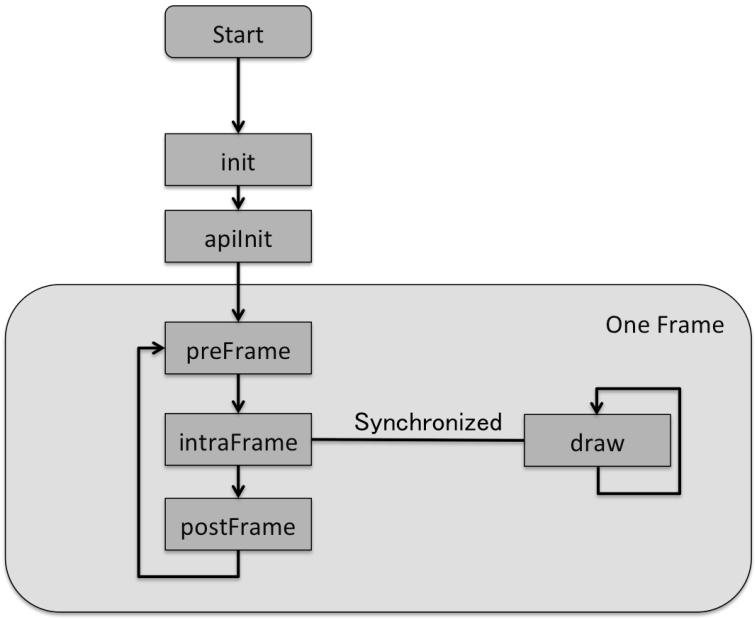

Fig. 10: VR Juggler kernel loop

CAVElib の関数と対応する関数は Table.2 のようになる。

VR Juggler	CAVElib
draw	CAVEDisplay
preFrame,intraFrame and postFrame	CAVEFrameFunction
init and contextInit	CAVEInitApplication

Table. 2: The correspondence of VR Juggler function to CAVElib function

VR Juggler と CAVElib には設計思想に違いはあるが、ユーザーとしては特に違和感なくプログラミングすることが可能である。

4.4 VR Juggler における入力デバイスの取り扱い

VR Juggler はいくつかの入力デバイス (ワンドやコントローラ、トラッキング装置) に対応しているが、世に存在するすべての入力デバイスに対応しているわけではない。それを補うことができるデバイス処理ソフトウェアが VRPN(Virtual Reality Peripheral Network)[10] である。VRPN は多種多様な入力デバイスからの情報をある決まったフォーマットに統一して提供を行うソフトウェアである。同様の機能を持ったソフトウェアでは Trackd が有名だがこれは有償であり、しかも高価である。しかし VRPN は無償であり、本研究の目的であるオープンソフトソフトウェアの開発に最適である。なお、VR Juggler は Trackd にも対応しており、Trackd が既に導入済みの場合はこれを利用することも可能である。また、光学式トラッキングシステムを開発している VICON 社が自社の製品に VRPN を組み込んでいることからも信頼性を確認することができる。手軽に導入できるビデオゲーム用コントローラにも対応しており、任天堂の Wii リモコンや Microsoft 社の XBOX360 用コントローラ、及び XBOX360 用モーションセンサデバイス Kinect を、バーチャルリアリティ装置で利用することができる。また、名前が示す通り VR Juggler が動作しているコンピュータとは別のコンピュータに接続されている入力デバイスの情報を、ネットワーク越しに使用することも可能である。入力デバイスの情報を Fig.11 に示すような経路で VR Juggler に送信することで、その入力デバイスの情報をアプリケーションで用いることができる。

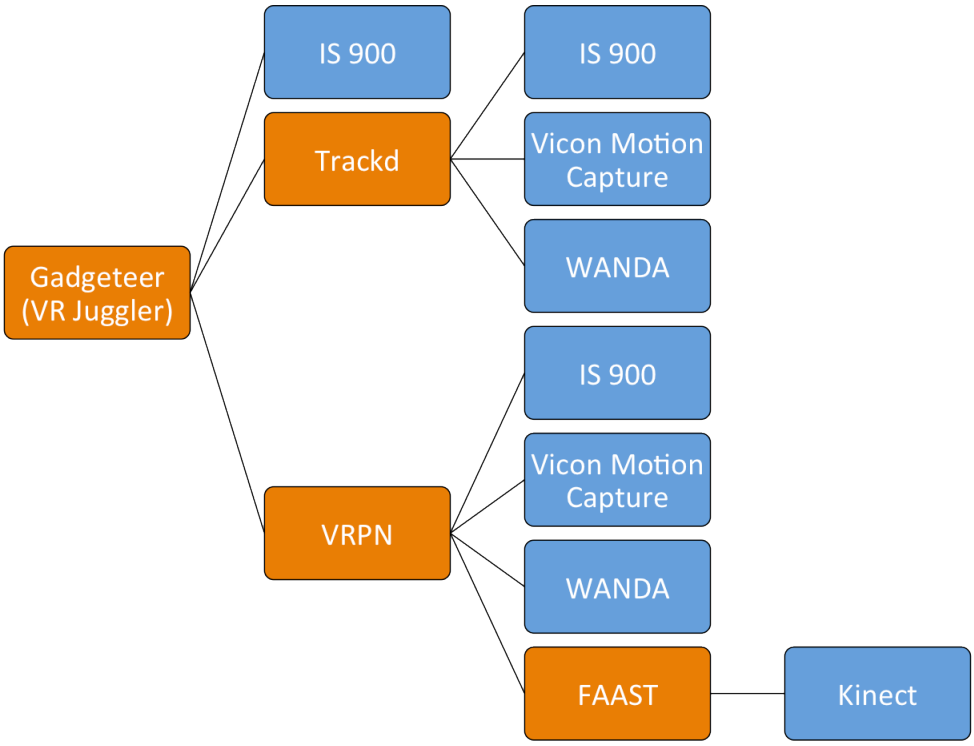

Fig. 11: Path of input device information. IS 900 は超音波式トラッキングシステム、Vicon Motion Capture は光学式トラッキングシステム、WANDA はトラックボールとボタンを搭載した VR 用コントローラ、FAAST は Kinect を VRPN で利用するためのツールキットである。

もし、Gadgeteer(VR Juggler) にも Trackd にも対応せず、さらに VRPN にも対応していないデバイスを使用したい場合は、Fig.12 に示すように、それぞれのデバイス処理ソフトウェアの API のいずれかを用いてドライバを自ら記述することで VR Juggler に対応させることが可能である。

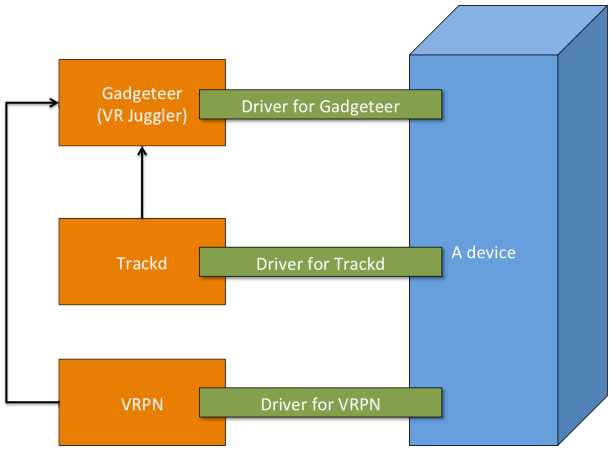

Fig. 12: Device driver

4.5 VR Juggler に基づく可視化ソフトウェア

VR Juggler を利用したバーチャルリアリティシステム向け 3 次元可視化ソフトウェアを現在開発中である。

4.5.1 開発の目的

現在利用されている VR 空間構築用ライブラリには様々な限界が来ている。特に、設計から約 20 年が経過しているという事実を見過ごすことは出来ない。また、従来のライブラリを使用した可視化ソフトウェアにも限界が来ている。それは古い VR 空間構築用ライブラリを利用していることや、古い VR 空間構築装置に対する設計に起因している。そこで、新しい VR 空間構築用ライブラリを利用した、新しい可視化ソフトウェアを開発する。

このソフトはオープンソースとして配布することを予定している。オープンソースとして公開しても、恥ずかしくないソフトウェアデザインが必要である。各種デザインパターン、開発技法を駆使しなければならないと考えている。

4.5.2 想定される問題点とその克服方法

可視化ソフトウェアの基本的な機能について 現状、無償で利用出来るバーチャルリアリティ向け可視化ソフトウェアはアプリケーションとしての基礎的な機能に乏しいものが多い。具体的にはアンドゥ、リドゥなどの機能や、可視化処理によって作成されたオブジェクトの保存機能等が用意されていない。その場で理解、

認識可能な程度の発見ならばともかく、一夜明けて重大な発見だと気がついたときにオブジェクトを復元するのは、従来のソフトウェアでは至難の業となっている。スクリーンショットを撮ることは可能だが対話性がなくなり、立体視ですらなくなるため、解析が困難なものになる可能性がある。従って、簡単なドローソフト程度の機能やヒストリー機能、オブジェクトの保存機能を実装する予定である。また、プレゼンテーションの際に自動で操作を行う機能や、その一時停止、巻き戻し、早送り等の補助機能も実装予定である。

ユーザーインターフェース　現在のCAVE装置では、ユーザーインターフェースとして、物理的なコントローラ (ワンド) を使うのが普通である。物理コントローラは高精度のトラッキングが可能であり、様々なボタンやトラックボール、ジョイスティックによってかなりの操作を可能にしている。しかし、我々は現実世界との間にコントローラを挟んではいないので、リアリティと没入感を高めるには、手には何も持たないことが望ましい。

幸いなことに、近年のモーションセンサ技術はCAVEシステムが開発された頃に比べて、大幅に進歩している。特にゲーム機に応用されるようになり、無償のライブラリが公開されるようになった今、さらに発達していくことは確実である。例えば、Microsoftの Kinectに関してはは OpenNIというライブラリが公開され、簡単にモーションセンシングすることが可能となった。しかし、CAVEでの導入例はほとんど無いのが現状である。積極的に最新のモーションセンサー技術を取り入れ、より対話性と没入性を高めたソフトウェアを開発する。

4.5.3　実装すべき機能

本研究で開発する可視化ソフトウェアに実装すべき機能を、簡単にまとめる。

- 豊富なスカラー場可視化手法
- 豊富なベクトル場可視化手法
- 同時に複数の場を可視化する機能
- 簡単なドローソフト程度の機能（アンドゥ、リドゥ、ヒストリー等）
- 可視化処理によって生成されたオブジェクトの保存機能
- プレゼンテーション向けの自動操作機能、及びそのオペレーション機能
- よりバーチャルリアリティらしい入力インターフェース

5　まとめ

新しい VR 空間構築用ライブラリである VR Juggler は革新的な特徴を備えている。VR Juggler はバーチャルリアリティシステムを抽象化することで、異なるバーチャルリアリティシステム上で動作させるために従来必要であった、アプリケーションの修正の手間を減らすことができるライブラリである。

一方、従来の VR 空間構築用ライブラリは設計からかなりの時間が経過しており、それを利用した従来の可視化ソフトウェアには、古いバーチャルリアリティ技術に合わせたソフトウェア設計による限界が訪れている。

本研究は、VR 可視化のための新しいソフトウェアを開発する目的で開始された。VR Juggler をベースとし、最新のユーザーインターフェースと可視化技術を組み合わせ、VR 可視化ソフトとしての業界標準となるような優れたソフトウェアを開発したいと考えている。

6 謝辞

1年間を通して熱心に指導していただいた、陰山聡教授と政田洋平助教に深く感謝いたします。海洋研究開発機構の川原慎太郎氏には最新可視化技術や、OpenGL技術について教えていただきました。同様に海洋研究開発機構の古市幹人氏には、CAVE に関するお話を聞かせていただきました。同研究室の山浦優気氏と吉崎健一氏には研究に関して様々なアドバイスをいただきました。

参考文献

[1] Carolina Cruz-Neira, Daniel J. Sandin, Thomas A. DeFanti, Robert V. Kenyon, John C.Hart, "The CAVE: audio visual experience automatic virtual environment", *Magazine of Communications of the ACM*, Vol.35, Issue 6, pp.64-72(1992)

[2] http://keckcaves.org/

[3] Allen Bierbaum, Christopher Just, Patrick Hartling, Kevin Meinert, Albert Baker, Carolina Cruz-Neira, "VR Juggler: a virtual platform for virtual reality application development", *Proceeding of IEEE Virtual Reality*, pp.89-96(2001)

[4] 目野 大輔, 陰山 聡, "VR Juggler による 3 次元可視化", スーパーコンピューティングニュース誌, Vol.13, No.1, pp.34-45(2011)

[5] 陰山 聡, 大野 暢亮, "バーチャルリアリティを用いた対話的3次元可視化ソフトウェアの開発とその応用", Journal of Plasma Fusion Research, Vol.84, No.11, pp.834-843(2008)

[6] Toshio Yamada, Michitaka Hirose, Yoshihiro Iida, "Development of Complete Immersive Display:COSMOS", *International Society on Virtual Systems and MultiMedia*, Vol.2, pp.522-527(1998)

[7] 清川 清, ミランダ・ミランダ・ミゲル, 野崎 一徳, 安福 健祐, 伊藤 一男, 岩田 恭典,"HOPE-高精細没入型周壁面ディスプレイの開発", 日本バーチャルリアリティ学会第 12 回大会論文集, (2007)

[8] Thomas A. DeFanti, Gregory Dawe, Daniel J. Sandin, Jurgen P. Schulze, Peter Otto, Javier Girado, Falko Kuester, Larry Smarr, Ramesh Rao, "The StarCAVE, a third-generation CAVE and virtual reality OptIPortal", *Journal of Future Generation Computer Systems*, Vol.25, Issue 2, pp.169-178(2009)

[9] 廣瀬通孝, 小木哲朗, 石綿昌平, 山田俊郎, “没入型多面ディスプレイ (CABIN) の開発 ”, 日本バーチャルリアリティ学会第 2 回大会論文集, pp.137-140(1997)

[10] http://www.cs.unc.edu/Research/vrpn/

A VR Juggler のインストールとコンフィギュレーション

VR Juggler とそのインストール方法についてスーパーコンピューティングニュース誌 [4] に発表した。その一部をこの付録にまとめる。

A.1 VR Juggler のインストール

VR Juggler の公式サイトからダウンロードできるのはバージョン 2.2 までである。今後は Google Code に新バージョンが掲載されることになる。最新バージョンは 3.0.0-1 であるが、ここでは Google Code からダウンロードした VR Juggler-3.0.0-0 を例にとって説明する。VR Juggler 本体をインストールする前に、依存ソフトウェア群 (Table.3) をインストールする必要がある。JDK と Doozer はマニュアルでは Optional と表記されているが、これらも入れておくほうが良い。JDK はコンフィグファイル記述用 GUI ツール vrjconfig のビルド、Doozer はサンプルプログラムや自作のプログラムのビルドに必要だからである。

CppDom	DOM analyzer for C++
SCons	Software build tool
GMTL	General Math Template Library
Boost	C++ library
Flagpoll	Software dependencies solver
Python	Required devel
Doozer	Building tool
JDK	Java Development Kit

Table. 3: Software needed for VR Juggler installation

パスを通し忘れないようして、付属（または公式サイト上）のマニュアルに従えばインストールは完了する。重要なパスとその例を Table.4 に示す。

LD_LIBRARY_PATH	A path to Boost	/usr/local/lib
FLAGPOLL_PATH	A path to FLAGPOLL	/usr/local/lib/flagpoll :/usr/local/share/flagpoll
JDK_HOME	A path to JDK	/usr/java/jdk1.6.0_21
DZR_BASE_DIR	A path to Doozer	/usr/local/Doozer

Table. 4: Important paths for VR Juggler installation

我々の環境においては、インストール実行中にBoostのバージョンが不明と表示され、インストールが成功しなかった。この問題は、インストール前のconfigure時に –with-boost-includes オプションでBoostのインストールされている場所（/usr/local/include）を指定することで回避した。また、Boost.Filesystemが見つからないと表示されインストールに失敗する現象も発生した。これもインストール前のconfigure時に、—with-boostオプションでlibboostfilesystemのある場所（/usr/local）を指定することで回避することができた。インストールの手順を以下にまとめる：

1. 解凍してできたVR Jugglerフォルダの最上層で autogen.sh を実行する。
2. ビルド用のフォルダを生成する。mkdir build
3. フォルダ内に移動する。 cd build
4. コンフィグスクリプトを上記のオプションをつけて実行する。../configure.pl
5. ビルドする。gmake build
6. インストールする。gmake install

A.2 VR Jugglerのコンフィギュレーション

インストールが完了していればvrjconfigコマンドが使えるようになっている。実行するとJavaによるGUIツールが立ち上がる。コンフィグファイルの拡張子はjconfである。jconfファイルの中身はXML文書だが、vrjconfigを使わずに書き上げるのは不可能に近い。サンプルのjconfファイルは当環境では /usr/local/share/vrjuggler-3.0.0/data/configFiles にインストールされた。自作のjconfファイルもそこに配置する。マニュアルを読み、サンプルのjconfファイルを見比べた後、我々のCAVE環境のjconfファイルを作成する過程で重要であると思われた項目と説明、例を以下に挙げる。ここで説明していない設定項目は基本的にデフォルト値をそ

のまま使用している。まず、スクリーン等のハードウェア関係の基本的な設定項目について説明する。既にCAVElibを利用している場合には、CAVElib用のコンフィグファイル（.cavercファイル）を参考にすると良い。我々の場合、トラッカー原点に対するスクリーンの四隅の座標設定はCAVElibのコンフィグファイルの設定値を参考にした。今回我々が用いたCAVE環境は1台のグラフィックスワークステーションと１枚のスクリーンのみの構成なので、ディスプレイシステムに関してはTable.5のような設定を行った。複数のスクリーンをもつ通常のCAVE型VRシステムでは言うまでもなくこの部分をそれぞれの値に設定する必要がある。

Number of Pipes	Number of output displays	1
Pipe Identifire	Which display do you output ? “-1” means using environment variable	-1

Table. 5: Hardware (GPU) configuration. “Display System”

ウィンドウの設定に関してはTable.6にまとめた。VRアプリケーションを開発する上で不可欠なツールとして「シミュレータ」がある。これはアプリケーションプログラムを書いているときにそのテストのたびに毎回CAVE室内に表示・移動してテストする必要がないよう、モニタのウィンドウ上にCAVE室での表示をシミュレートして表示するものである。キーボードとマウスを使って、視点とコントローラ（ワンド）の移動を模擬する。CAVElibにはCAVE Simulatorというシミュレータソフトが付属されているが、VR Jugglerにも同様なシミュレータがある。Table.7にそのシミュレータに関係した設定項目をまとめた。CAVE型VR装置で最も重要な設定はスクリーンの設定である。VR表示を行う座標系におけるスクリーンの位置やステレオ表示の設定はここで行う。Table.8にそのスクリーンに関係する設定項目をまとめた。

origin	Place of window	X:0,Y:10
size	Size of window	Width:1400 , Height:1050
Pipe	Which GPU do you output ?	0
In Stereo	Config of stereo	true
Use border	Border of window	true
Hide Mouse Pointer	Mouse pointer inactive ?	true
Full Screen Window	Full screen or not	true
Always On Top	Always window on top or not	false
Use this Window?	Use this window?	true
Keyboard/Mouse Input Handler	Input source of keyboard and mouse	Keyboard/Mouse InputHandler0
Allow Mouse Locking	When you drag a mouse with any keys , the pointer is locked inside window or not.	true

Table. 6: Output window configuration. “Display Window”

origin	Simulator placement inside window	X:0,Y:0
size	Size of viewport. You should input percentages.	Width:1.0 Height:1.0
View	Left eye ,right eye or both eyes	Stereo
User	user	VRJuglerUser0
Use this Viewport	Do you use this viewport ?	false
Vertical Field of View	Vertical field of view	80.0

Table. 7: Simulator configuration. “Simulator Viewport”

origin	Surface placement inside window	X:0,Y:0
size	Size of viewport. You should input percentages.	Width:1.0 Height:1.0
View	Left eye ,right eye or both eyes	Stereo
lower_left_corner	Place of lower left corner of the screen	X:-1.524 Y:0.0 Z:0.0
lower_right_corner	Place of lower right corner of the screen	X:1.524 Y:0.0 Z:0.0
upper_right_corner	Place of upper right corner of the screen	X:1.524 Y:2.441 Z:0.0
upper_left_corner	Place of upper left corner of the screen	X:-1.524 Y:2.441 Z:0.0
User	user	VRJugglerUser0
Use this Viewport ?	Do you use this viewport ?	true
Is Tracked ?	The screen able to move with tracker ?	false
Tracker Proxy	A movable screen tracker	None
Auto Corner Update	Automatic corner update ?	No Update

Table. 8: Screen configuration. “Surface Viewport”

driver_path	Path to input drivers	/usr/loacl/lib/gadgeteer-2.0.0/drivers
driver	The name of the driver	Trackd_drv

Table. 9: Input devices configuration. “Input Manager”

VR Juggler では、インプットデバイスの管理を Gadgeteer と呼ばれるソフトウェアが行っている。標準的な CAVE 装置では、インプットデバイスからの入力信号や、トラッキングデータを Trackd と呼ばれるデーモンを介して受信するのが普通である。Trackd をそのまま使う場合には Table.9 のように設定すればよい。Trackd 以外のデバイスドライバを使うことも可能である。Gadgeteer は Virtual Reality Peripheral Network (VRPN) にも対応しているので、様々な物理デバイスをネットワーク経由で利用することも可能である。各種のトラッキング装置や、

力覚デバイス、Wii リモコン等にも対応している。有名なデバイスに関してはほとんどドライバが同梱されている。VRPN を利用するには別途インストールする必要があり、VR Juggler の../configure 時に–with-vrpn オプションでインストール先を指定した後 VR Juggler をビルドする。最後にユーザ関係の主要な設定を Table.10 に示す。

Head Position	A proxy of Head position tracker	VJHead
User's eye separation	User's eye separation	0.069

Table. 10: User configuration. "VR Juggler User"

A.3 入力デバイスの設定

入力デバイスの設定は、以下の手続きで行う：

1. Input Manager を設定する。（ドライバの存在する場所の指定）
2. Device Driver を設定する。（使用するドライバの設定）
3. Proxy を設定する。（ボタンやスティックの設定）
4. Proxy Alias を設定する。（プログラムで使用するための名前を設定）

設定が完了するとこれらの参照関係が Device Driver を根とする木構造となっているはずである。アプリケーションから使用できるのは 4. で設定する Proxy Alias のみなので必ずここまで設定しなければならない。我々のシステムでは、入力デバイスとして、デジタルボタン 5 個、ジョイスティックが 1 つのコントローラ（ワンド）を使っている。ワンドの設定例を Fig.13 に示す。ジョイスティックの入力値は X 値と Y 値を独立に 2 つの入力として設定する必要がある。Fig.14 にトラッキングデバイスの設定を、Fig.15 にはキーボードデバイスの設定を示す。ここでは、エスケープキーでプログラムを終了するように設定している。このような設定をしないままフルスクリーンでプログラムを実行すると、プログラムを終了させる手段がなくなる。

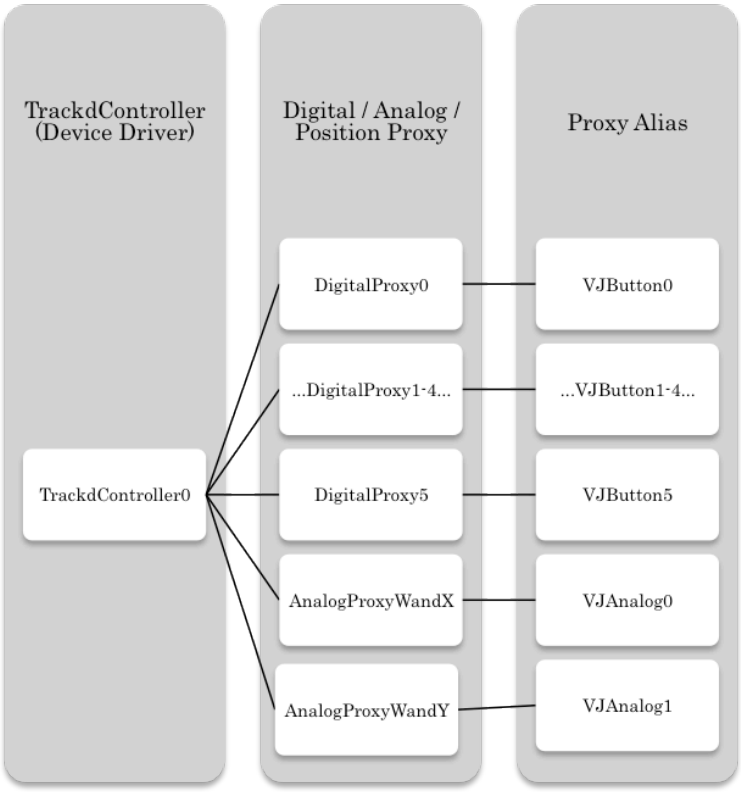

Fig. 13: Configuration of buttons and joysticks

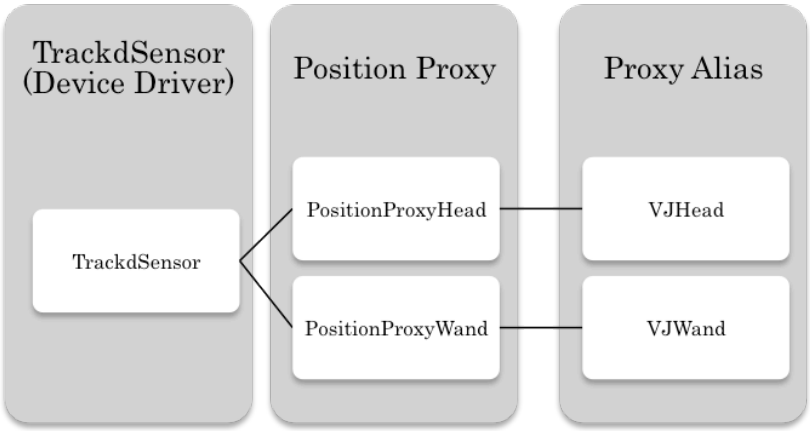

Fig. 14: Configuration of trakers

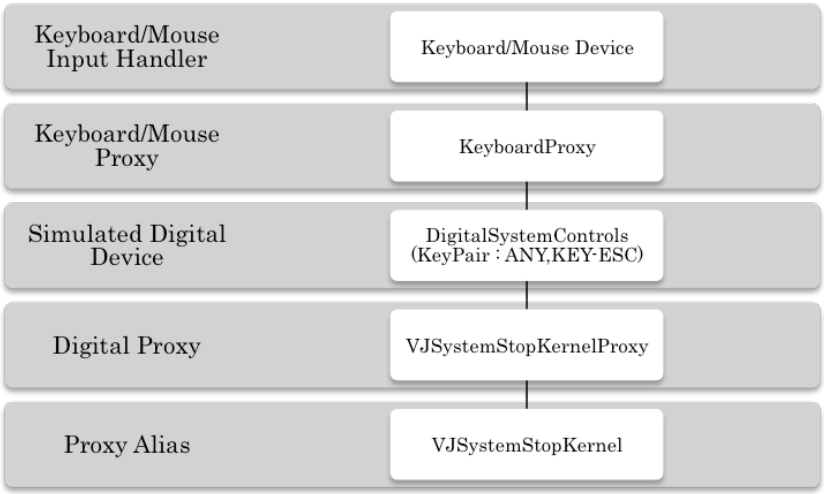

Fig. 15: An example of keyboard configuration; ESC key to exit.

以上の設定をこなせばそれぞれの環境固有のハードウェアを使用して VR Juggler が動作するはずである。サンプルプログラムもインストールと同時にビルドされているので、それを動かして確認することができる。もしビルドされていなければ、次に示すフォルダにおいて make コマンドを叩けば良い。サンプルプログラムはビルド時に作成したフォルダの中 (/PathToVRJ/vrjuggler-3.0.0-0-src/build/modules/vrjuggler/samples) にある。サンプルのソースはまた別の場所 (/PathToVRJ/vrjuggler-3.0.0-0-src/modules/vrjuggler/samples) にある。