

Requirements Specification for Adaptation of 3D Application with Virtusphere

(Basic requirements for OS Windows based 3D applications to provide their compatibility with immersive virtual reality systems, in particular with Virtusphere (VS))

Virtual reality (VR) has become an important part of our modern world. Many kinds of simulation training, games, and researches cannot be done without immersion into virtual reality. As a rule such immersion requires a head mounted display (HMD) consisting of glasses (personal displays) and a 3DOF sensor. However, complete and realistic immersion requires orientation and position sensors for tracking movements of head and weapon (manipulator). These are Six Degrees Of Freedom Sensors (6DOF*). Besides devices are needed that allow the user to move in virtual space more naturally. Virtusphere is one of such devices.

The VS is a device for simulation training and games with full physical immersion into VR. Unlike most devices it provides the user with more natural interface and allows him to move in virtual space more naturally just like in ordinary life. In VS one can really physically move and run with any speed, in any direction and at any distance. For the developers of 3D applications this opens new possibilities – to make applications more natural and realistic for users. Simultaneously these VS possibilities give a benchmark which of the applications will gain from joint use with VS and which will not. For example, applications with racing cars, avia simulators or with a surgical operation have nothing to do with man's walk or run and consequently with VS. The best for VS are applications in which a hero walks in 3D space, for example, First Person Shooter games (FPS).

The VS is a complex consisting of 3 main blocks. Mechanical device to provide a user physical movement in virtual space, a system of sensors, force-feedback devices to provide communication of real and virtual worlds and a visualization system to generate virtual space and its display for the user. Mechanical device for user physical movement is a sphere capable to rotate in any direction on a wheel platform. When the user moves in any direction the sphere rotates in the opposite direction compensating user movement in real space. Sensors, virtual helmets and computers may vary in a wide range. The VS architecture is open and practically any block or its components may be easily changed depending on the device designation or user wish.

VS designation is easily changed by replacing an application by another one. Thereby the base part of VS mechanical and electronic components remains. In certain cases depending on specific character of the applications the types of manipulators and a part of sensors may change. Such versatility is provided by VRPN (Virtual Reality Peripheral Network) – a system of drivers connecting the VS electronic devices with the application. Its convenience is that the drivers of most peripherals used for VR are embedded into it. The developers and publishers who provided compatibility of their 3D applications with VRPN (VRPN support) at the same time make it compatible with most peripherals including VS. Manufacturers of peripherals for VR that embedded the drivers of their devices in VRPN provide a possibility of their use with many 3D applications that supported compatibility with VRPN. Users will give preference to applications and devices with VRPN support as to a considerable extent they get free hand of configuration choice and automatic compatibility when upgrading their peripherals.

Thus any application adapted with VRPN becomes compatible with VS as all peripheral devices operate via VRPN. For effective use of these devices and capabilities of the application a possibility of their setting should be provided. The settings may be both in the application itself and in separate setup and configuration files. The VS operator or user will be able to make all necessary settings basing on intuitive application interface or a manual. Qualitative settings permit to get all possible advantages of joint use of VS and an application and will provide user deep immersion into VR.

Joint testing is an important stage of VS and application alignment. At the initial stages usual peripherals – a mouse and a joystick – may be used for testing. Connecting them via VRPN one can see how VS will work with the application. At the subsequent stage it is advisable to carry out full-scale tests for final immersive system setup.

Main technical requirements under which 3D application becomes compatible with VS are described below.

Introduction to VS

Virtusphere consists of a big sphere (3m in diameter) placed on a special platform that allows the sphere to rotate around its center. The user walks with a head mounted display inside the sphere (like a squirrel in a cage). Sensors transfer information about user's head turns, direction and speed of movement to the computer. On the helmet display the user can see 3D virtual space, generated by the computer according to his/her movement. Unlike most devices VS has a natural interface and allows the user to move more naturally just like in ordinary life. In VS one can really physically move and run with any speed, in any direction and at any distance. The user can interact with objects of the virtual environment via a manipulator. The objects of the virtual space and avatars may obey program logic of the application or may be controlled by real people, for example, by a user of the other VS. An important peculiarity of VS and other immersive systems is that the data transmitted to the application from them are absolute, and movement direction is not tied to gaze direction.



VS set

The VS architecture is open and its components may be easily changed depending on the device designation or user wish. There is a minimum set of devices composing a base and most-common set. For specific tasks a complicated set that takes into account special requirements may be used.

A. Simple VS set

- Head mounted display with head phones
- 3DOF sensor on the user's head to track its turns
- 3DOF sensor on the weapon (manipulator) to track its turns (optional)
- Inverted and oriented mouse as a sensor to track changes of coordinates of the user in the application. The turned over mouse is located underneath the VS, X-axis of the mouse is, for example, directed to the North.
- The buttons of the other mouse are located on the weapon (manipulator) and are used to control the application.
- It is possible to use VS alone as a local machine as well as in a network of several VSs for use with one application.
- Either a desktop computer as a local machine (with wireless devices on the user's side) or a portable notebook (with all devices connected to it) can be used.

B. Advanced VS set

- Head mounted display with head phones
- 6DOF sensor on the user's head to track its turns and movements
- 6DOF sensor on the weapon (manipulator) to track its turns and movements
- Inverted and oriented mouse as a sensor to track changes of the user coordinates in the application. It is placed underneath the VS, X-axis of the mouse, for example, is directed to North.

- The buttons of the other mouse, a joystick, a keyboard or other device are located on the weapon (manipulator) and are used to control the application.
- It is possible to use VS alone as a local machine as well as in a network of several VSs for use with one application.
- Either a desktop computer as a local machine (with wireless devices on the user's side) or a portable notebook (with all devices connected to it) can be used.
- For movement coordinates X and Y it is possible to connect data from different VRPN sensors (including a mouse)

C. Special VS set. Glance into the future.

- User movements additional sensors.
- Force-feedback devices (gloves and vests).
- Smell generators.
- Integration of VS with CAVE (in this case HMD is not needed).
- Image projection onto the VS matt surface using 5 projectors (in this case there is no HMD).
- Other hardware for VR.

VS drivers

The VS is a hardware and software complex comprising different devices composition of which may vary due to various reasons. To provide joint operation of these devices and applications the VS uses the VRPN universal driver system. This is an open source Virtual Reality Peripheral Network software. VRPN is a class library and ready servers developed for implementation of transparent network interface between applications and physical devices used in virtual reality systems. Its main purpose is allocation of a separate PC or host for each virtual reality system station of the control periphery (trackers, button, analog, tactile sensing, audio devices, etc.). VRPN provides connection of the application with all devices using an appropriate mechanism for each type of device. The application that uses VRPN need not be concerned about network topology and physical location of peripherals, no matter if a thousand kilometer away host or a local computer on which the application runs is used. VRPN-server and VRPN-client can also be realized both as separate applications and as a single application.

The application receives adapted data about all movements via VRPN from its virtual sensors. Virtual sensors receive data from real physical sensors. The application "does not know" anything about them and that is why replacement of a device by another one is trouble-free. The user of the application that supports VRPN does not have to change and compile the application codes when changing sensors. The user simply changes the name of the device to another one in the text configuration file. The drivers for all devices of the VS are embedded into VRPN. In the future efforts will be applied to make new devices for VR compatible with VRPN. This means that 3D application that supports VRPN becomes compatible with most devices for VR and VS.

Configuration of Application for VS

From the very beginning of work to make an application compatible with the VS one should bear in mind that users may have different VS set. From time to time users may change devices or change their designation. This means that a user will have to change the configuration and adapt the application for these devices. As a rule such a possibility is given in standard settings of the application and in advanced settings in separate text files. Through them a user must be able to turn on/off devices and functions, to change coefficients, signs, to shift axes, etc. This is connected with configuration peculiarities and designation of a separate immersive system and it is impossible to foresee it beforehand.

The use of VRPN in the system presupposes an obligatory availability of a configuration file `vrpn.cfg`, through which communication of VS devices with the application is provided. In this file the devices used in the system are selected and matches of the device names and ports are preset. Advanced device settings, their activation/deactivation, correcting parameters and other variables may be given in a separate file, for example, `VS_Application_name.cfg` (or something similar so that it would be easier for the user to find the file with VS settings). These two configurations should be enough to create immersive systems with different parameters that use capabilities of VS and the application as much as possible.

Example of parameters included in the second configuration file:

1. Turning on/ off of VRPN in the Application
2. Address of VRPN data (IP or local machine)
3. Names of sensors of the user's head and weapon (manipulator) and their matches in the Application. Output of 3D sensor axes parameters with an option to change their order and inversion (e.g. +x,+y,+z), as soon as there is no uniformity with sensors manufacturers. A possibility to set parameter equivalence, for example, if the weapon (manipulator) turn is not tracked its data are set equal to the head data and in the result the weapon sight is always in the center of the display. Determination of parameter units variation.
4. Names of sensors of the user's head, weapon (manipulator) and body and their matches in the Application. Output of X,Y,Z linear axes movement parameters with an option of changing their sequence, inversion, substitution (SWAP) and reference point. It is advisable to introduce separate zooming facts for sensor sensitivity and movement speed. Data may come to the application from different sources, for example, data about the head movement come from magnetic sensor, data about user's body movement (the same as VS rotation) come from the mouse. A possibility of setting a constant parameter, for example, for Z axis if bending, squatting and jumps are not tracked. A possibility to set parameter equivalence, for example, if the weapon (manipulator) movement is not tracked its data are set equal to the head data. Determination of parameter units.
5. Some types of sensors require the function of their calibration and synchronization.
6. Buttons on the weapon (manipulator).
7. A possibility to turn on/off the application control buttons, configuration of buttons through VRPN (e.g. map output, GPS navigator, object capture, weapon change, and other functions of the Application). Your work to make the application compatible with the VS will be evaluated by grateful users yet at the stage of the immersive system setup. Intuitive application interface for configuration, suitable configuration manual will help the user to qualitatively setup the application and to get all possible benefits of using immersive system VS-Application.

Testing joint work of the application and VS

It is possible to test compatibility of the Application and VS on early stages of development, even without Virtusphere. Usual peripherals – a mouse and a joystick – may be used for testing. Connecting them via VRPN one can see how VS will work with the application. Instead of the 3DOF or 6DOF sensor a joystick and `vrpn_Tracker-AnalogyFly` can be used to feed data (e.g. to the HeadTracker).

Example of joystick configuration in `vrpn.cfg`

head tracker based on joystick positions

```
vrpn_Tracker_AnalogyFly    HeadTracker0 60.0    absolute
X      NULL      0      0.0    0.0 1.0 1.0
```

```

Y    NULL          1    0.0    0.0 1.0 1.0
Z    NULL          6    0.0    0.0 1.0 1.0
RX   *Joystick0    0    0.0    0.0 0.5 1.0
RY   *Joystick0    1    0.0    0.0 0.5 1.0
RZ   *Joystick0    2    0.0    0.0 -.5 1.0
RESET NULL  0

# walk tracker based on mouse positions
vrpn_Tracker_AnalogFly    WalkTracker0  60.0    differential
X    *Mouse0          0    0.5    0.0 25.0 1.0
Y    NULL            2    0.0    0.0 1.0 1.0
Z    *Mouse0          1    0.5    0.0 -25.0 1.0
RX   NULL            0    0.0    0.0 1.0 1.0
RY   NULL            1    0.0    0.0 1.0 1.0
RZ   NULL            2    0.0    0.0 1.0 1.0
RESET NULL  0

#modified mouse sensor for Virtusphere in VRPN
vrpn_MouseDelta_debug Mouse0

```

It is recommended to perform final testing with real sensors and head mounted displays. Only wide testing will show actual compatibility of the Application and Virtusphere.

Optimization of VS-Application immersive system

A distinction of the VS from majority of other immersive systems is high sensitivity of the VS to turns. Turns of the head and weapon (manipulator) in the VS occur like in ordinary life, i.e. practically in the real time mode. The image changing during the turn must be displayed for the user in the same mode. As soon as automobiles and aircraft are inert (they do not turn as quickly as the head or hand) there is no need in abrupt change of image in auto and avia simulators. Inert systems allow latency (latency or delay of image delivery to the user), it is natural and hardly visible there. In the VS latency is critical and should be minimized. Otherwise this may lead to simulator sickness (in most cases the reason of simulator sickness – nausea, pains – is that the image lags behind the turn of the user's head).

To avoid this problem all mechanical and electronic hardware in the VS is selected so that to minimize time loss of VS response to the user action. The application is one of the most important link in this system. Any application parameter may become critical including speed of image drawing, algorithm of sensors data recalculation and ways of configuration. Most modern 3D engines are optimized and are able to draw qualitative images in real time mode. Applications based on such engines are quite capable to provide adequate work with VS. We hope that together with interesting and fast application the user get an opportunity to set it up and optimize and also will get a convenient guide for its use with VS.

Good luck!

References:

http://www.virtusphere.com/Appl_compatib.html

<http://www.cs.unc.edu/Research/vrpn/>

* **Six degrees of freedom (6DoF)** refers to motion of a rigid body in three-dimensional space, namely the ability to move forward/backward, up/down, left/right (translation in three perpendicular axes) combined with rotation about three perpendicular axes (pitch, yaw, roll). As the movement along each of the three axes is independent of each other and independent of the rotation about any of these axes, the motion indeed has six degrees of freedom.