Design and Implementation of Interactive Visualisation Configuration using Interaction Paradigms in Virtual Reality Environment

Tomas Skripcak, Pavol Tanuska, Nils Schmeisser

Abstract*— this article is aimed on the specific task of interaction with an immerse visualisation application. The first part of the article provides basic introduction into interaction paradigms in 3D space. After that classification and description of standard interaction tasks are presented. We introduce our view of matter on relations between the 3D interaction and standard interaction techniques. The second part describes hardware and software components of our VR system. Furthermore an overview of the architecture and implementation details of system for interactive visualisation configuration is discussed. We describe design specifications of a 3D UI, which helps to make interaction less error prone for inexperienced users. A specific solution for performing numerical input is also provided. The main goal of the article is to describe how 3D user interface paradigms can be implemented in the VR system.

Index Terms: 3D interaction, virtual reality, visualisation.

I. INTRODUCTION

Visualisation is one of the techniques which is widely adopted by scientists. While tabular data representation is an artificial construct, visualisation enables us to get a (fuzzy) grip on the nature of data. Results are displayed in n-dimensional space, depending on the visualisation specification. In order to enable better exploration of the visualised data we often introduce some level of interaction. Classical approaches for HCI *(Human Computer Interaction)* so called WIMP *(Windows, Icons, Menus and Pointer)* becomes obsolete when we are not dealing with standard visualisation setup (desktop environment), but more immerse VR *(Virtual Reality)* environments like CAVE *(Cave Automatic Virtual Environment)* [16] or powerwall are applied. The ability to use more dimensions is substantial, however there are no established standards for the interaction in cyberspace. In most cases, HCI paradigms are implemented directly for the specific project needs.

Manuscript Received July 09, 2011.

Tomas Skripcak, Institute of Applied Informatics, Automation and Mathematics, Slovak University of Technology, Trnava, Slovakia, (e-mail: tomas.skripcak@stuba.sk), Department of Information Technology, Helmholtz-Zentrum Dresden-Rossendorf, Dresden, Germany, (e-mail: t.skripcak@hzdr.de).

Pavol Tanuska, Institute of Applied Informatics, Automation and Mathematics, Slovak University of Technology, Trnava, Slovakia, +421 918 646 061, (e-mail: pavol.tanuska@stuba.sk).

Nils Schmeisser, Department of Information Technology, Helmholtz-Zentrum Dresden-Rossendorf, Dresden, Germany, (e-mail: n.schmeisser@hzdr.de).

A. 3D Interaction

Bowman [1] defines 3D interaction as HCI where the user tasks are performed in 3D (spatial) context. It is important to notice that we can use 3D input devices as well as 2D input devices with appropriate mapping in 3D space. Bowman also introduced another two concepts which are directly connected with 3D interaction:

1) 3D UI (3D User Interface): is an ui which involves 3d interaction.
2) 3D interaction technique: is a method (comprehend hardware and software) which allows a user to accomplish a task in a spatial context.

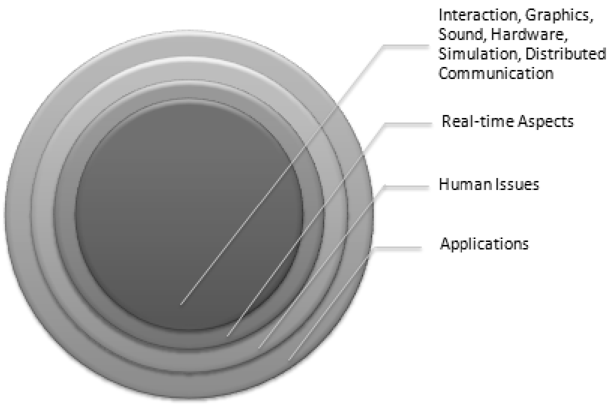
Interaction, Graphics, Sound, Hardware, Simulation, Distributed Communication
Real-time Aspects
Human Issues
Applications

Fig. 1. Interaction place in vr system [15].

According to [4] we can differentiate between 4 modes of interaction in a virtual reality application:

1) *Direct interaction*: this is a common mode of interaction where, in order to perform specific tasks in virtual environment, the user has to mimic the behaviour from real world.
2) *Physical interaction*: in this case, the input to the virtual environment is provided by handling some sort of input device.
3) *Virtual interaction*: is the one where interaction device is also a part of the virtual environment. The user interface could be considered as a virtual form of interaction. It is necessary to mention, that activation of the virtual device is usually done by direct or physical interaction.

4) *Agent interaction*: is a special type of interaction where we specify our task by choosing one of a predefined set of commands and express it to the so called agent. An agent, in the context of agent interaction, could be understood as a commuter entity, which is responsible for receiving and detecting commands and translating them into the specific set of actions.

The process of 3D interaction is closely related to real-time rendering and technical design of 3D input and output devices [18].

Spatial interaction is a task which can not be done easily. According to [15] there are problems which researchers and developers have to consider, when they are designing application with a spatial interaction:

1) *No constraints*: specifically when we design virtual world where everything is possible, the users usually suffers from confusion. Standard definition of confusion refers to state where the user is unclear about what is happening [3]. In another words user is loosing the control. We can provide two examples of this phenomenon. The first one occurs when the user of a VR system has to control all 6 DOF. This is quite difficult situation and only a small percentage of people are able to perform full 3D interaction. Second example of confusion is visible when we introduce not natural mapping. Let’s say that we are dealing with inner (intrinsic) dimension temperature. In cyberspace we can map it to inner property e.g. colour which is quite natural. But mapping to outer dimension (e.g. geospatial position) could cause a state of confusion.
2) *No standards*: research of 3D UI is still in progress. However the application of 3D UI is usually limited to special cases. That is the reason why there are no general accepted standards. This is slowly changing and we can see initiatives to develop standards for a natural user interface [8] [13].
3) *Task mapping*: it is difficult to find useable mappings for specific tasks (e.g. full 360 degree rotation, complex actions ...).
4) *Lack of tools*: no standard input devices (some of them are developed only for specific projects). Specialised tools for {3d ui} development are in early stages.
5) *Precision*: input devices suffer from static and dynamic errors. Calibration is necessary. For more information about the calibration process please refer to [7]
6) *Fatigue*: 3D UI are usually not designed with respect to resting points (and human body limits). It is not possible to handle the application 8 hours a day.
7) *Perception*: persons could have problems with perception. In the real world the basic principle of an action invoking a reaction in most cases provides immediate feedback on a user-triggered action. A 3D UI should behave the same way (via utilization of visual, audio, haptic, temperature or other feedback).

The most significant problem is that users are simply unable to do full 3D interaction without previous experience with the tracking device. Our opinion is that each single VR system should also provide a training environment. We can compare VR applications with games. What is similar to both is that controlling is done via a non standard set of commands. If we take a look at game industry, even a simple game is offering practical tutorial level, where the end user is guided how to use game UI and has the possibility to accustom on interaction paradigm.

The original idea behind NUI *(Natural User Interfaces)* is direct interaction with computer [11]. Naturalism in this context simple means that we are trying to find the most natural commands (motions, gestures ...), which are easy to discover in action and the user has a feeling of an immediate progress while learning how to handle the application. The problem, that users have to train mainly because they have to discover the limits (resolution, distances ...) of a tracking device remains. The history of computer interaction gives us a few examples how the users were already changing a way of working with computers:

1) Switching from the hardware keyboard to the virtual one (touchable screen in cell-phone and tablet devices).
2) Switching from the command line interface to the graphical one (learning how to use mouse effectively, now it seems to be natural to use mouse but lets have a look to beginner’s information technology class).

B. 3D Interaction Tasks

There are several well established and defined tasks which form the foundation of full 3D interaction: selection, manipulation, navigation, system control, symbolic input [1] [4] [15] [18] [19]. Here we are providing a brief description of tasks which were important for our interactive configuration application.

1) Selection

Selection is one of the fundamental tasks in the area of interaction with a virtual environment. The main objection of selection is to choose a target object from a set, in order to perform further interaction tasks. According to [15] we know local (select close object in the reach of the hand) and remote (select object behind the scope of hand reachable area) selection. Bellow is a list of possible selection techniques:

a) Pointing

Pointing is the most intuitive technique for selection which comes from a real world environment. In a virtual environment pointing is also known as *ray casting*. This task is quite easy to handle because the user has to control only 2 DOF *(degrees of freedom)*. There are two known variants of ray casting technique. First one is cone casting (flashlight/spotlight). The idea is that the selectable area is increasing with increasing distance from the source of flashlight. The second is known as indirect point (curved pointer) and it is trying to solve selection of objects behind the reachable scope. Introduction of a second hand into the process of pointing could also improve precision and user experience [1] [15].

b) Virtual hand

Virtual hand is favourite, fast and simple selection method. The selection process is performed by touching a virtual object with the virtual hand. Selection is computed in the real time by utilisation of collision detection.

The virtual hand is enclosed by a selectable bounding box and a collision detection algorithm is implemented for determining selected virtual objects. The main problem with this solution is the inability to select objects which are further away. Variation of virtual hand called go-go hand was proposed by [17]. In this case, a non 1:1 mapping of real world to virtual world is used. It allows us to reach behind the physical limits (arm extension). The virtual hand could also be used in combination with pointing, known as shish-kebab technique [19]. Pointing has to be performed in order to get initial set of objects (coarse selection), ordered according to depth, and than the virtual hand technique let us executes fine-grain selection.

c) Body related selection

More precious selection is done by utilisation of more parts of the user body. Line of sight selection was introduced by [6]. The idea is to use the relation between hand and eyes. According to this relation the group of selection techniques were developed (e.g. sticky finger selection, see [6]). Multiple volume selection is another example of body related selection where number of volumes of interests is attached to user body for better interpretation of user intentions. This process is described in more detail in [19].

Usually the haptic feedback is missing in all forms of selection (unless some sort of exoskeleton is used). Normally we replace haptic feedback with other forms of feedback (visual, audio) in order to make the selection task more users friendly.

2) Manipulation

During the manipulation task, the user is able to change position, orientation and scaling attributes of a selected entity in the virtual environment. Regarding to [1], *manipulation imitates, general target acquisition and position movements that we perform in the real world (a combination of reaching/grabbing, moving and orienting of objects)*. It also means that selection is one of manipulation subtasks (canonical task). We can differentiate between four types of manipulation:

a) Virtual hand

This technique follows virtual hand selection. Connection between a selected entity and virtual hand is done. Every transformation performed with virtual hand is also applied to the connected entity.

b) HOMER TECHNIQUE

HOMER is an acronym for *Hand-Centred Object Manipulation Extending Ray-Casting*. In this case pointing is used as a selection method and if the selection is successful, the hand is centred into the position of a virtual entity, which can be directly manipulated afterwards. For more information please refer to [2]

c) Scaled-world grab

This manipulation follows the line of sight selection method. In this case manipulation is done by virtual hand technique and in order to bring selected entity into user working area whole virtual world is scaled [1].

d) World-in-miniature

World in miniature is a special type of a manipulation method which was firstly introduced in [20]. It is based on displaying a secondary mineralised 3D map of virtual world. The user is manipulating that minimised version of virtual entities. All actions are transferred to the original virtual entities. This technique also serves the navigation task where user could be represented with his own avatar.

3) System Control

System control interactions contain methods and techniques used for communication with applications. This communication yields to changes in application state or could also be used to modify attributes of virtual entities and trigger special actions. Functionally system control is equivalent to WIMP from desktop environments but according to [18] it is not always a good idea to implement this kind of interaction for virtual reality system the same way. In the last years, there was a research dealing with the classification of 3D UI *(3D user interface)* specifically for the purpose of virtual environments (e.g. classification of 3D UI was one subtask of CONTIGRA [5] project). The results could serve as a starting point for the further implementation or improvements of ideas which already exist.

4) Symbolic Input

Symbolic input is one of the most difficult task for implementation in VR systems. In symbolic input interaction we usually want to provide the user a natural way for inserting textual or numeric information into the VR application. According to [1] most of the existing systems do not provide this option and if so they are usually using special hardware devices for this purpose.

Table I. Desktop and VR environment system control comparison.

DESKTOP ENVIRONMENT	VR ENVIRONMENT
WIMP user interface (triggered by ray casting selection)	3D user interface (virtual hand selection with embedded death zone)
keyboard shortcuts	gestures, speech recognition
textual input	virtual keyboard , gesture, speech recognition

In the Table I, we introduce our view of matter to relations between standard and spatial system control interaction.

II. VIRTUAL REALITY ENVIRONMENT SETUP

We try to build our system for interactive configuration and visualisation in a flexible manner.

Table II. Software used in VR system.

COMPONENT	SOFTWARE	DESCRIPTION
Visualisation	VTK [23]	Scientific visualisation
Visualisation	VIPER [21]	Extension to VTK
Configuration	DPY	Engine for VR applications
Configuration	3DS lib [10]	Loader of .3DS files
Configuration	libTrCalibr [12]	Calibration for tracking devices
Configuration	OpenNI [8]	Kinect [9] driver
Configuration	NITE [8]	Middleware for Kinect

From the software perspective our VR system consists of two different parts, experiment visualisation and visualisation configuration component. The visualisation part is implemented on top of VTK [23] *(Visualisation Toolkit - based on OpenGL)* which is used together with an extension called VIPER *Visual Interactive Plasma computer Experiments for Researchers* [21]. The main task in visualisation part is to map n-dimensional experimental data from dataset to m-dimensional visualisation. As an example in addition to first 3 dimensions used to construct set of 3D objects, we are using colour for indication difference in 4th dimension data. In addition glyphs (also called icons) are used for visual representation of additional dimensions. These small graphical entities will store additional dimension in their attributes (size, shape ...). Usage of glyphs in visualization is mainly based on human perception.

The visualisation configuration application is implemented with C++ on top of very ourselves made VR engine (rendering via *OpenGL*, providing generic access to tracking systems like Polhemus Patriot, Kinect, eMargin 7800, D5 cyberglowe).

Table **II** provides summary of software setup.

III. INTERACTIVE CONFIGURATION APPLICATION

The purpose of the interactive configuration application (for the future reference called *dashboard*) is the creation of scientific visualisation configuration in an immersive environment. The idea is focused on improvement of VR system usability by providing complete conceptual solution where the configuration as well as visualisation is done in a virtual environment and where there is no need for switching between desktop and VR setup in order to perform specific tasks. This should make the interaction process with the visualised experimental data more fluent and also enable a next level of the visual data analysis for scientists.

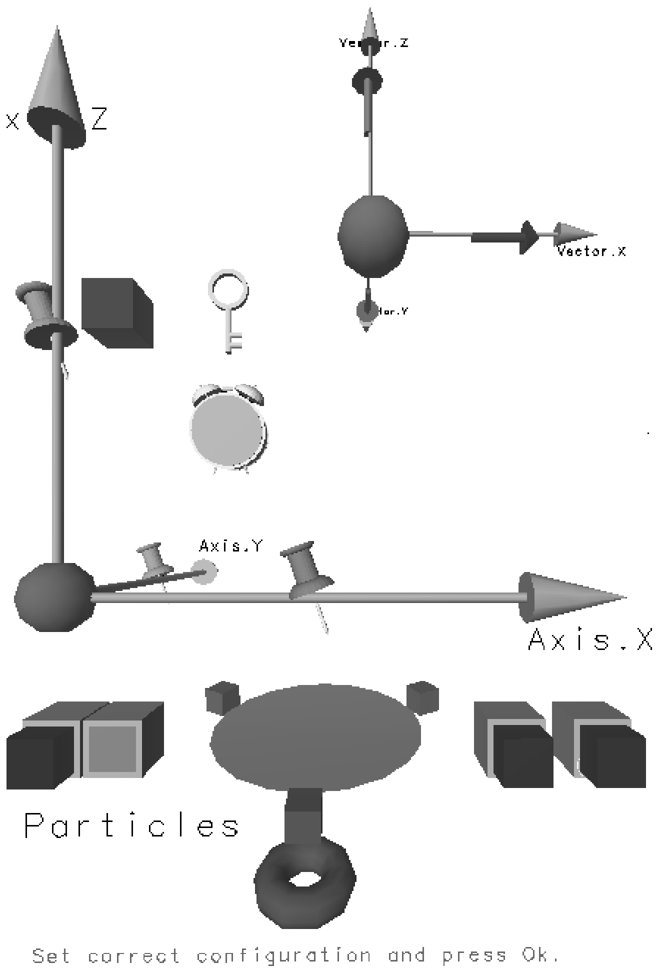

Fig. 2. Dashboard application.

A. Dashboard Design

From the architecture perspective we used a MVC *(Model-View-Controller)* design pattern. The controller class is the entry point to the application and it is directly mounted to our DPY virtual reality engine. All the input comes directly to the controller class and afterwards it is up to the controller which view will be displayed on the output or which action will be performed on the model *(Domain Model)* of the application. Tie view itself is responsible for rendering all the graphical elements. The actual state of application is encapsulated into a Domain Model (Fig. 3). It means that after user made all necessary configurations, Domain Model is taken and translated into valid visualisation configuration file. The common infrastructure of the dashboard application is divided into a bunch of *service* classes. Their major objective is to make specific tasks reusable in order to be executed from different MVC elements when the application will grow bigger. A basic overview of the architecture structure is shown in the Fig. 4. In order to perform a correct visualisation setup, the user has to follow a specific work-flow (Fig. 10). After that, the state of a Domain Model is translated into the visualisation configuration file and the user can initialise the experiment data visualisation part of the VR system.

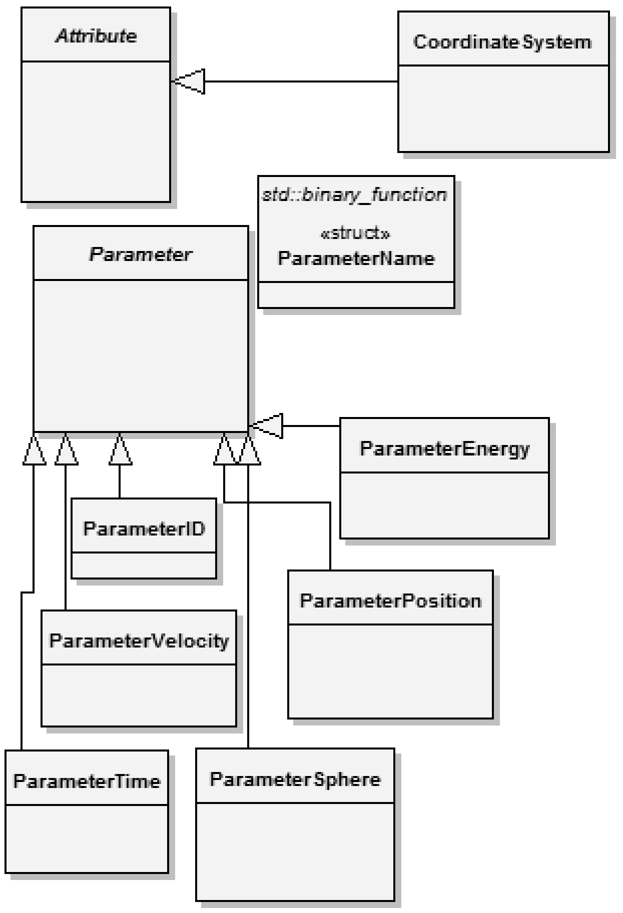

Fig. 3. Domain model of dashboard application.

B. Implementation of interaction tasks

In this section we describe our development of specific interaction tasks, which are needed for the dashboard configuration application.

C. Selection and Manipulation

For the purpose of implementation selection and manipulation tasks we followed the virtual hand paradigm described in the section Selection and Manipulation above. The main reason was that virtual hand provides the most natural mapping between a user action and an action performed in the virtual environment. In our case we restricted all UI objects to be located inside the virtual space addressable by our tracking system. The bounding box of a virtual hand has a spherical shape and provides enough precision for the selection task. In order to make even faster initial setup we introduced spatial entities into our Domain Model which have indirect impact on the created configuration. They trigger setup action on several other Domain Model entities (e.g. we have a position parameter and we want to assign position parameter to the virtual axis attribute with most common setting where position x is mapped to axis x, position y to axis y etc.).

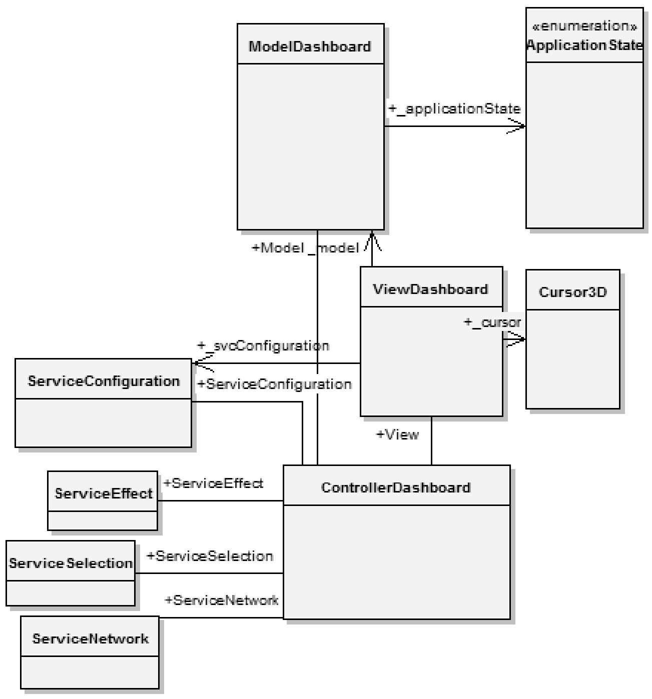

Fig. 4. Model-View-Controller architecture of dashboard.

D. System Commands and Symbolic Input

We implemented basic set of virtual 3D UI. First inspiration was taken from [5]. Following is the description of developed user interface:

Push Switch: has a form of 3D button. If the *IsButton* property is set to true, it serves the purpose of sending one time command which trigger execution of specific action (push and release). However it is possible to enable switch behaviour. In this case UI element holds state until some other element does not perform release command (push and stay pressed).

1) *Up and Down Switch*: is an extended form of 3D button with three inner states (Up, Down, None).
2) *Radio Button*: is a collection of Push Switch elements with enabled switch behaviour. It is used when we need to perform choice from small number of options.
3) *Dial Switch*: is an user control which let us perform selection from bigger number of options. The pressable knob of Dial Switch is rotated in order to choose right value.
4) *Ring Menu*: is sometimes known as a carousel panel. It is based on a revolving ring (conveyor belt) with visualized option elements. Typical usage is selection from bigger number of options.

The interaction with UI elements is done in two steps. First we need to focus an user control element (its inner state *IsFocused* property is changed to true), which activate user control for interaction via controlling some DOF (e.g. changing depth for pressing a switch, using circle gesture for rotation ...). We currently use virtual hand paradigm for focusing a correct UI element.

Another idea is introduction of tabbed UI. Elements will be inserted into some kind of layout controls. These controls will be responsible for positioning UI around virtual world and will also provide tabbing behaviour for their inner elements. The Focus will be stable and the user will move the focus only by performing tab gesture. According to this paradigm we can create easy navigatable UI hierarchy (graph), which does not require permanent interaction. The result will be increased comfort of the user while using the application, because his body can be most of the time in resting position.

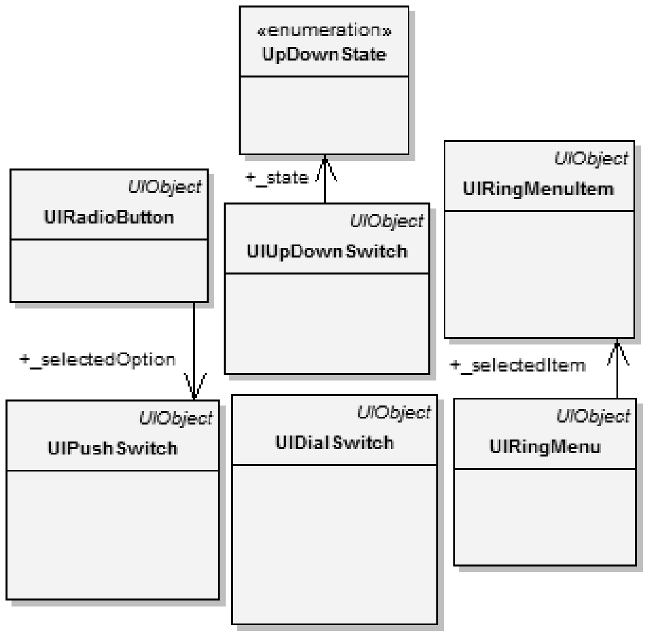

Fig. 5. Class model of implemented 3D UI.

Fig. 6 contains a set of graphical implementation for 3D UI used in our system. There is one more thing which should be noticed at this place. We think that it is a good idea to provide a quicker way of interaction with some graphical system control elements. Classical UI keyboard shortcuts fulfil this purpose. In a virtual environment we can try to emulate function of shortcuts with an agent based interaction focused on gesture and speech recognition. In the end both approaches (visual + command) supplement each other. By providing more options for executing specific actions we can enable more users to work with the application (e.g. disabled persons).

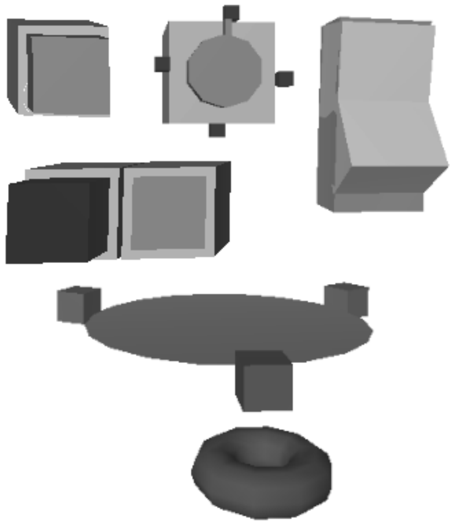

Fig. 6. Graphical representation for 3D UI.

In the dashboard application we have to find a way how to enable symbolic input interaction for the end user. Specifically we were dealing with a numerical symbolic input (where the range of input is known), which should be used to allow setup filters for visualisation parameters. Interaction task was discussed together with an input visual representation in order to find a quick and lightweight solution (regarding to display space). We come up with a concept which we call *"holy ring"*, figured in the Fig. 7. The input is visually represented as a ring and the portion of ring which is rendered depends on a specified input value. Gestural interaction was implemented for increasing (circular gesture in clockwise direction) and decreasing (circular gesture in counter clockwise direction) numeric input value. In the future we are also thinking about possibility of displaying a virtual keyboard for symbolic input interaction tasks.

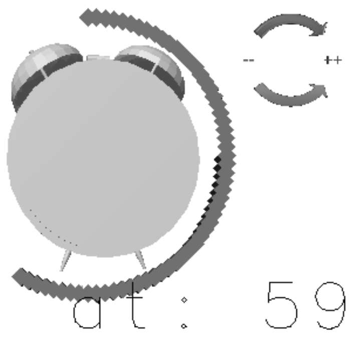

Fig. 7. Symbolic numeric input visualised in the form of a "holy ring".

E. Visualisation

The data visualisation a part of our system (Fig. 8) is dealing with the loading and presentation of data in 3D. As it was mentioned in section above, it uses VTK + VIPER extension. The visualisation part is using a configuration file created by dashboard application for the setup.

It also receives commands and tracking data permanently via UDP. The control process is outlined in the Fig. 9.

IV. FUTURE WORK

During the implementation we found out a few very important facts and we would like to do some further development according to them. First we need to employ more standardized components for tracking subsystem (VRPN *Virtual Reality Peripheral Network* [22] and OpenTracker [14]). An analysis of new natural interaction possibilities with 3D UI and gestures recognition has to be performed. The final application should be easy to handle (physical as well as mental). Finally usability has to be approved by scientists using the system in order to fulfil their (non IT) research tasks.

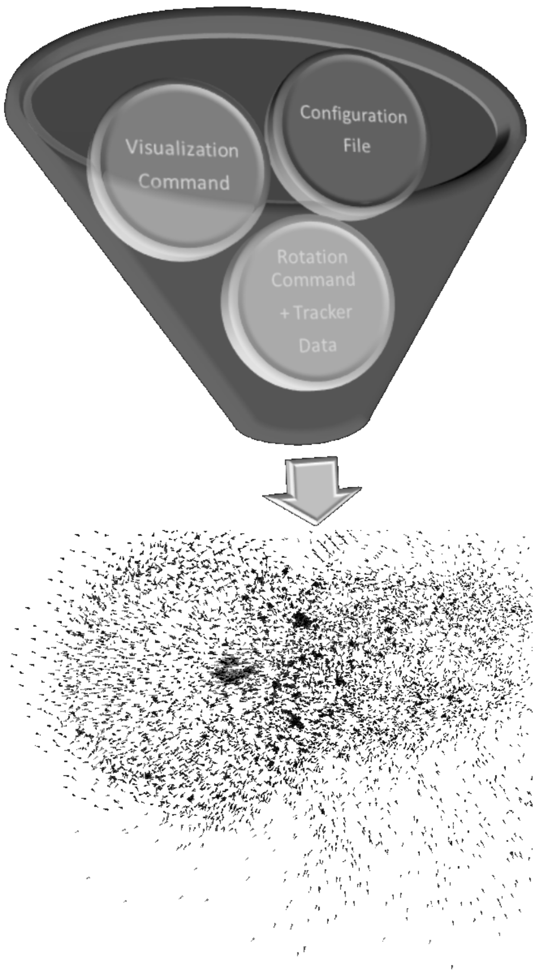

Fig. 8. Visualisation part retrieving all the necessary setting from dashboard.

V. CONCLUSION

This paper is focused on the problem of interaction and 3D UI in virtual reality applications. We are providing a basic insight into interaction paradigms available in 3D space. We provide and an example of 3D UI and discuss an idea of layout containers which creates focusable hierarchy structure of interaction elements. This form of interaction with user interface is easier to handle because it lower the number of DOF and stable focus does not require permanent interaction so the user could be in resting position most of the time. We also presented a symbolic input interaction focused mainly on numeric input. It is based on gestural recognition and visualised via a concept which we call *"holy ring"*. Chosen interaction tasks were implemented for the purpose of our VR system which is dealing with interactive configuration and visualisation of experimental data in order to provide scientists a better way for performing visual data analysis.

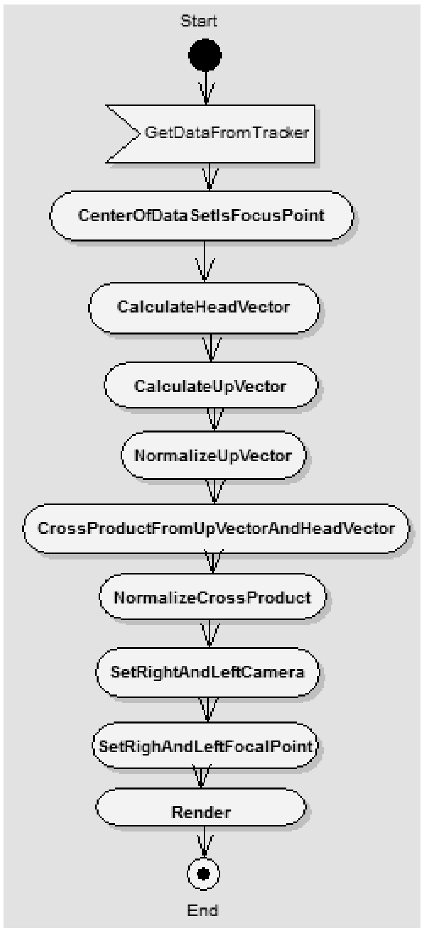

Fig. 9. Implementing passive stereoscopy in VTK.

REFERENCES

1. D. Bowman, "3D user interfaces : theory and practice." Boston: Addison-Wesley, 2005.
2. D. A. Bowman and L. F. Hodges, "An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments," p. 35--38, 1997.
3. Oxford Dictionaries Online, "Definition of confusion." [Online]. Available: http://oxforddictionaries.com/definition/confusion. [Accessed: 11-Aug-2011].
4. A. Craig and ScienceDirect (Online service), Developing virtual reality applications foundations of effective design. Burlington, MA. ;;Oxford :: Morgan Kaufmann,, 2009.
5. Chair of Multimedia Technology, "Component-oriented three-dimensional interactive graphical applications CONTIGRA." [Online]. Available: http://www.mmt.inf.tu-dresden.de/Forschung/Projekte/CONTIGRA/index_en.xhtml. [Accessed: 21-Jul-2011].
6. J. S. Pierce, A. S. Forsberg, M. J. Conway, S. Hong, R. C. Zeleznik, and M. R. Mine, "Image plane interaction techniques in 3D immersive environments," in Proceedings of the 1997 symposium on Interactive 3D graphics, New York, NY, USA, 1997, p. 39–ff.

7. T. Skripcak, P. Tanuska, and N. Schmeisser, “Interactive calibration and registration of electromagnetic tracking system for virtual reality,” in International Doctoral Seminar Proceedings [May 15 -17, 2011 in Smolenice Castle], Trnava, 2011, p. 365--375.
8. OpenNI, “Introducing OpenNI.” [Online]. Available: http://www.openni.org/. [Accessed: 20-Jul-2011].
9. Microsorft, “Kinect” [Online]. Available: http://www.xbox.com/en-US/kinect. [Accessed: 20-Jul-2011].
10. J. E. Kyprianidis, “Lib3ds - lib3ds is an overall software library for managing 3D-Studio Release 3 and 4 ‘.3DS’ files. - Google Project Hosting.”[Online]. Available: http://code.google.com/p/lib3ds/. [Accessed: 20-Jul-2011].
11. Wikipedia, “Natural user interface.” [Online]. Available: http://en.wikipedia.org/wiki/Natural_user_interface. [Accessed: 12-Aug-2011].
12. V. V. Kindratenko, NCSA libTrCalibr 2.00. National Center for Supercomputing Applications: , 2000.
13. Natural User Interface Group,“NUI Group.” [Online]. Available: http://nuigroup.com/go/. [Accessed: 20-Jul-2011].
14. Studierstube project, “OpenTracker.” [Online]. Available: http://studierstube.icg.tugraz.at/opentracker/. [Accessed: 21-Jul-2011].
15. Bowman et. al., “The Art and Science of 3D Interaction,” in Virtual Reality Conference, IEEE, Los Alamitos, CA, USA, 2000, p. 294.
16. C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart, “The CAVE: audio visual experience automatic virtual environment,” Communications of the ACM, vol. 35, pp. 64–72, Jun-1992.
17. I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa, “The go-go interaction technique: non-linear mapping for direct manipulation in VR,” in Proceedings of the 9th annual ACM symposium on User interface software and technology, New York, NY, USA, 1996, pp. 79–80.
18. E. Kruijff, “Unconventional 3D user interfaces for virtual environments.”
19. A. Olwal, S. Feiner, and L. Kjelldahl, “Unit—A Modular Framework for Interaction Technique Design, Development and Implementation.,” 2002.
20. R. Stoakley, M. J. Conway, and R. Pausch, “Virtual Reality on a WIM: Interactive Worlds in Miniature,” p. 265--272, 1995.
21. L. Zuhl, “Visualisierung von Laser-Plasma-Simulationen,” Diploma Thesis, Technishe Universitat Dresden, 2011.
22. R. M. Taylor, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser, “VRPN,” in Proceedings of the ACM symposium on Virtual reality software and technology - VRST ’01, Baniff, Alberta, Canada, 2001, p. 55.
23. Kitware, “VTK - The Visualization Toolkit.” [Online]. Available: http://www.vtk.org/. [Accessed: 20-Jul-2011].

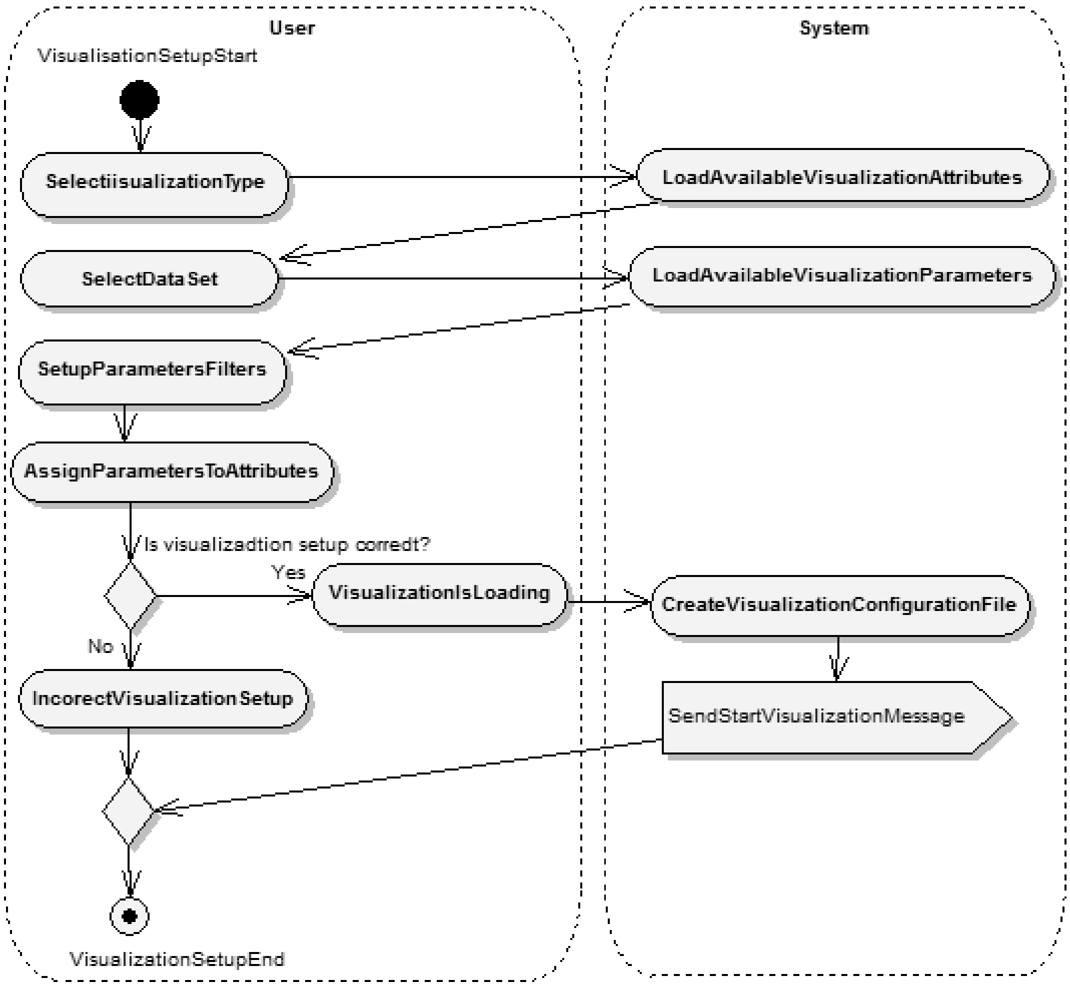
User
System
VisualisationSetupStart
SelectiisualizationType
LoadAvailableVisualizationAttributes
SelectData Set
LoadAvailableVisualizationParameters
SetupParametersFilters
AssignParametersToAttributes
Is visualizadtion setup corredt?
Yes
VisualizationIsLoading
CreateVisualizationConfigurationFile
No
IncorectVisualizationSetup
SendStartVisualizationMessage
VizualizationSetupEnd

Fig. 10. Visualisation configuration work-flow.

AUTHORS PROFILE

Tomas Skripcak received his Diploma degree, in the field of applied informatics and automation in industry, from Slovak University of Technology in Bratislava in 2010. During studies (2005-2011), he worked as a software developer in MMS Softec Ltd. (Trnava-Slovakia), where he was responsible for design, development, documentation and testing of information systems based on .NET technology. In certain time, he is a PhD student at Slovak University of Technology in the field of process automation and information. Since February 2011, he is situated in Germany at Helmholtz-Zentrum Dresden-Rossendorf. He has published 2 papers in international conferences and 1 journal article. His research area of interest includes software systems design and testing, natural user interface design and a novel way of human-computer interaction.

Pavol Tanuska (M ’09 IACSIT, M ’10 IEEE, M ’10 IAENG) received his PhD degree (2000) in area of Applied informatics and automation from Slovak University of Technology (STU), Faculty of Material Science and Technology Trnava, Slovakia. Since 2004 he is an Associate Professor in Institute of Applied Informatics, Automation and Mathematics at the Slovak University of Technology, Trnava, Slovakia. His research and development activities include the field of Information systems development, especially problems of verification and validation and Database systems. He has published over 120 scientific and technical papers in national and international conference proceedings and journals.

Nils Schmeisser received his Diploma degree in mathematics at the Technische Universitat Dresden. Before that he was attending a special school for mathematics and natural science in Riesa, Saxony. He spent one term at the Danish Institute of Fundamental Metrology in Lyngby, Denmark. His Diploma thesis focused on the processing of data acquired by scanning probe microscopy. Currently he is working at the Helmholtz-Zentrum Dresden-Rossendorf, IT department. He is intensively participating in many international projects (e.g. EuroMagNET II, DeNUF, TOPFLOW-PTS … – in order to mention a few) where he is dealing mainly with database programming tasks. His research fields of interests are high performance computing, visualisation/virtual reality and. database design and development.