Manipulation and presentation of object features in virtual reality environments through smart interaction device

Iliyan Nachev[1], Stoyan Maleshkov[2]

[1] Virtual Reality Laboratory, Technical University Sofia, Sofia 1797, Bulgaria

[2] Virtual Reality Laboratory, Technical University Sofia, Sofia 1797, Bulgaria

Abstract

The purpose of this paper is to present an approach for interacting remotely with virtual objects in virtual reality environment and assigning and displaying additional information about the object features through multimodal presentation techniques using visual, audio or tactile channels. The interaction is performed via client server application running on smart Android tablets and Windows-based virtual reality system. The software architecture and implementation details are discussed and evaluated. In comparison to similar techniques our solution has the advantage to be more user-friendly, providing additional information for the user and keeping minimal delay while performing the manipulation tasks.

Keywords: *Virtual Reality, Multimodal Presentation, Android, Smart Device Interface, Gesture Recognition, Engineering Analysis Results Validation.*

1. Introduction

Virtual reality (VR) is becoming a popular trend in the last decade not only in the consumer market, but in the market for professionals as well - engineering, architecture, etc. The added benefits are recognized by many companies and individuals (e.g. illumination design [2]), but there are still a lot of user interaction issues that need to be addressed. The whole process of interaction with objects in VR environments needs significant improvements and rethinking in order to provide ergonomic and intuitive experience for the end-user (more details about the different challenges can be found in [4], [5] and [11]). The solution provided in this paper relies on a smart tablet running Android OS that acts primarily as an input device for manipulating virtual objects in real time but in the same time can also output additional information. It connects wirelessly to a Windows-based PC running VR applications. The lack of cables gives the user the ability to control interaction with the objects in the VR environment from any place in the room without the need to use keyboard and mouse. The proposed approach is suitable for virtual reality systems where a wireless mouse or other specialized interface devices are not sufficient to implement advanced interaction techniques.

2. Representation of object features

The workflow for representation of object features in

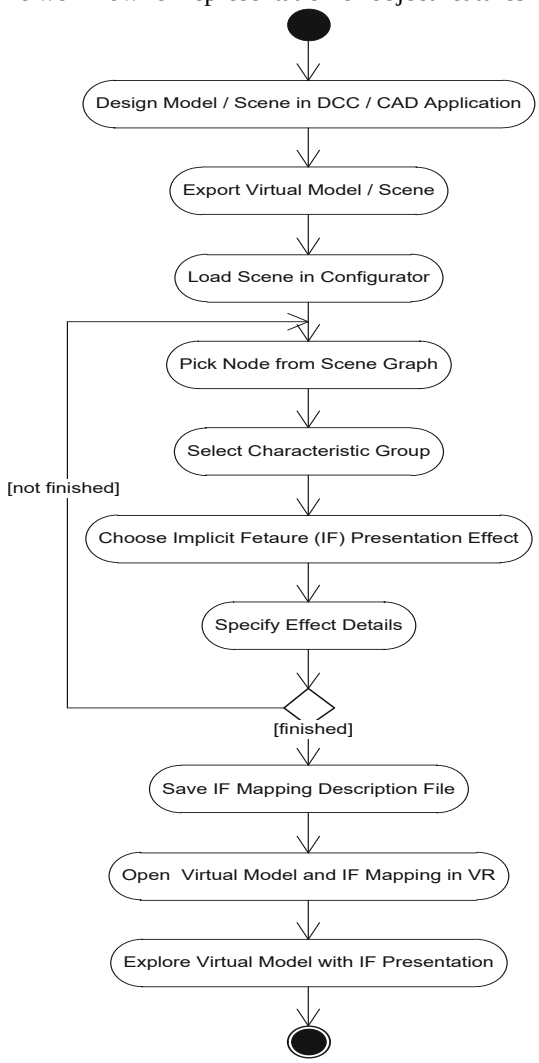

Fig. 1. Assigning and exploring features of virtual objects.

VR environments (both explicit like color or material as well as implicit in the form of magnetic field strength or temperature distribution) is introduced in [1] and shown on Fig. 1. The process begins with the design phase, where the virtual model is build-up using CAD or other digital content creation software package. The completed model is exported in VR compatible file format (preferable VRML), followed by the mapping phase, where the one-to-one correspondence between virtual object feature and its presentation effect is accomplished [3]. For this purpose a specialized configuration application (*Configurator*) has been developed and implemented. The *Configurator* creates the so-called effects, which describe various ways for presenting virtual object features applying different sensorial stimuli: visual, audio and haptic. Each effect provides a set of parameters, which can be adjusted by the user [6]. After the competition of the mapping process the *Configurator* generates a feature mapping description file, which contains detailed description of the effects assigned to each of the virtual objects features.

3. Software architecture

The software solution proposed in this work consists of three main parts - client application (installed on the Android tablet), server application for input processing, and a specialized application called *Configurator* (both on the desktop side running under Windows). The Android application communicates with the input server using a simplified text-based protocol via Wi-Fi and receives data in XML format from the *Configurator*. Both applications: the server for the Windows side and the client for the Android side have similar architectures, presented on Fig. 2 and Fig. 3.

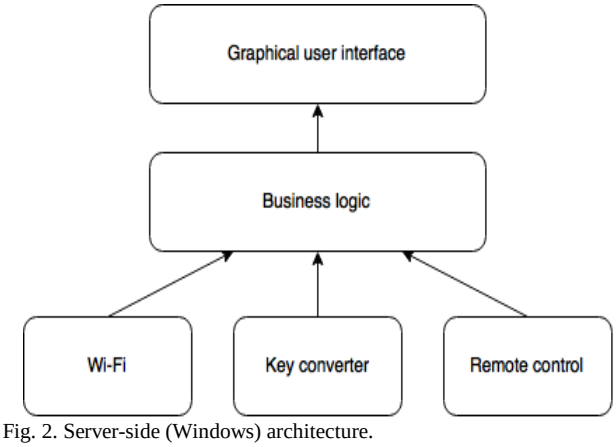

Fig. 2. Server-side (Windows) architecture.

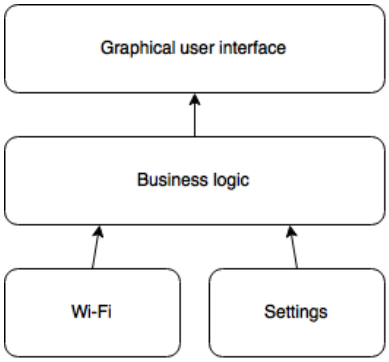

Fig. 3. Client-side (Android) architecture.

Fig. 4 shows the basic relationship between the three applications: client (smart device - Android), server and *Configurator* (PC - Windows) and how they interact with each other.

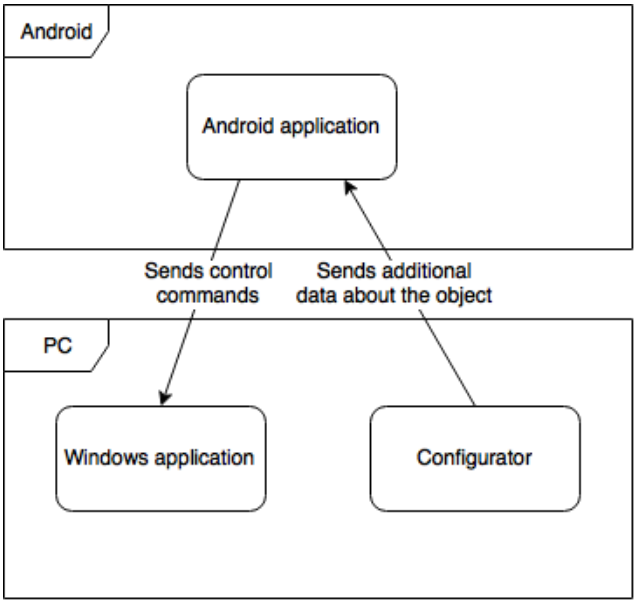

Fig. 4. Interactions between the applications.

3.1 Technical Approach

The client-server application for Android-based tablets was developed as supplementary tool for extending the functionality of the application *Configurator*, which is used to enhance the features of virtual objects with additional characteristics (e.g. temperature, stress, displacement, and magnetic field strength) obtained at selected points of interest via real-world measurements or through simulations [7], [8].

The interactions with the virtual objects executed by client-server application can be divided into two main types: control and feedback. The control type provides the ability to manipulate the objects in the virtual environment

performing transition, rotation, scaling or camera movement, while the second type corresponds to the ability to request and receive additional information related to selected point of a virtual object with subsequent representation of the received data via visual and/or tactile perception channel.

In order to implement and execute the first (control) type of interaction, an application for Windows-based desktop computers has been developed to proxy the commands received from the tablet application to the desktop operating system. The second (feedback) type of interaction is achieved via feedback interface that connects directly the Android application to the *Configurator* application.

3.2 Control Interactions

The Windows application establishes a wireless connection (via Wi-Fi) with the Android tablet application, in order to receive different commands, which have been transformed into corresponding mouse and keyboard actions. The tablet processes and recognizes gestures from the touch screen and generates the corresponding commands. UDP is used as transport layer protocol (OSI model) for the commands; on the OSI application layer, a text-based protocol was developed in order to achieve the required interactions (Table 1) [10].

Table 1: Protocol description

Event name	***Signature***	***Values***
Echo Reply	remoteVR:echo:1: remoteVR:reply:2:	Both are full events names without variables
Keyboard	key:code:state	**key**: event name **code**: key code according to Android's KeyEvent (e.g. 35) [14] **state**: 1 (up) or 2 (down)
Mouse keys	mousekey:mouse_event:	**mousekey**: event name **mouse_event**: -1 (left button click), -2 (left button double click), -4 (left button down), -8 (right button click) or -32 (middle button click)
Scroll	scroll:direction:	**scroll**: event name **direction**: 4 (up) or 8 (down)
Mouse movement	move:x:y:	**move**: event name **x**: the horizontal move (float value) **y**: the vertical move (float value)

The events defined in Table 1 cover the main user interactions that typically occur when working with mouse and keyboard. The events *Echo* and *Reply* are used for initial connection between the tablet and the desktop PC. The keyboard interaction is realized with one event for key down/up, while for the simulation of a mouse requires the usage of several events implementing: movement (x and y axis), buttons (clicks, double clicks; left, right and middle button) and scroll (up and down direction).

The interaction between the user and selected virtual reality object is realized by the following workflow:

1. User interacts with the tablet's touch screen. The tablet's OS generates touch input event.

2. Every touch screen event is processed and based on finger count the actual gesture is recognized. Table 2 presents in more details the mapping between recognized touch screen gesture and corresponding mouse action. If a user interaction leads to a gesture that cannot be recognized (e.g. movement with four fingers) – it is ignored and not processed any further.

Table 2: Gestures description

Mouse action	**Touch screen gesture**
Left mouse button click	Tap with one finger
Left mouse button double-click	Two fast taps with one finger
Right mouse button click	Tap with two fingers
Middle mouse button click	Tap with three fingers
Mouse movement	Move one finger
Scrolling	Move two fingers up/down
Selection	Tap and hold with one finger, then move

For keyboard input the Android's virtual keyboard is used (alongside with possible Ctrl, Alt and Shift keys modifiers).

3. Every matching gesture is transformed into a protocol package, that is sent via the wireless connection to the input processor application on the PC (using the standard Java networking APIs [12]).

4. The input processor reads the incoming packets and interprets them as mouse and keyboard interactions (using the standard Java Robot class [13]).

5. These commands are executed resulting in manipulation of the virtual object.

3.3 Feedback Interactions

When the Android application is connected to the *Configurator* application, *Configurator* sends back multimodal information [9] for each selected point via the wireless connection in XML format. The structure of the information, which is received by the Android application is shown on Figure 5.

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="AnalysisValuesPacket">
    <xs:complexType>
      <xs:sequence>
        <xs:element type="xs:string" name="packetType"/>
        <xs:element type="xs:string" name="destination"/>
        <xs:element name="data">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="AnalysisValue" minOccurs="1" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element type="xs:string" name="objectName" minOccurs="1" maxOccurs="1"/>
                    <xs:element type="xs:string" name="analysisName" minOccurs="1" maxOccurs="1"/>
                    <xs:element type="xs:float" name="analysisValue" minOccurs="1" maxOccurs="1"/>
                    <xs:element type="xs:short" name="colorRed" minOccurs="1" maxOccurs="1"/>
                    <xs:element type="xs:byte" name="colorGreen" minOccurs="1" maxOccurs="1"/>
                    <xs:element type="xs:byte" name="colorBlue" minOccurs="1" maxOccurs="1"/>
                    <xs:element type="xs:byte" name="lowRange" minOccurs="1" maxOccurs="1"/>
                    <xs:element type="xs:byte" name="highRange" minOccurs="1" maxOccurs="1"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Fig. 5 XML schema for properties description of selected point of a virtual object.

The XML schema above sets the definition that the main tag in the XML document should be called *AnalysisValuesPacket* and will contain meta information (packet type and destination) together with the actual data. Important for the tablet application are exactly the sub-tags under *AnalysisValue*. They represent one or more explicit or implicit characteristics of a given virtual object, which are described by giving name, value, presentation color (in RGB format), and also be constrained by specifying the corresponding maximal and minimal values.

```
<AnalysisValuesPacket xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <packetType>AnalysisValuesMsg</packetType>
  <destination>Client</destination>
  <data>
    <AnalysisValue>
      <objectName>Boiler</objectName>
      <analysisName>temp</analysisName>
      <analysisValue>390.25</analysisValue>
      <colorRed>255</colorRed>
      <colorGreen>0</colorGreen>
      <colorBlue>0</colorBlue>
      <lowRange>20</lowRange>
      <highRange>450</highRange>
    </AnalysisValue>
  </data>
</AnalysisValuesPacket>
```

Figure 6: Sample XML file which conforms to the XML schema rules

The sample XML file, shown on Figure 6, created following the rules defined with the XML schema presented on Figure 5, describes one implicit characteristics (temperature) of an object (called Boiler) with value (390.25), that is close to the maximum (450), colored in red. The XML document is parsed into a business object, which is used for subsequent visualization of the data on the tablet's screen. The parsing is implemented using the standard for Android XML pull parser [14].

The GUI of the Android client application (Figure 7) presents the information for each characteristic, which has been already processed by showing: name, current value, limits, and color code. In addition to this a tactile feedback (vibration) for the first characteristics has been implemented. That way more comprehensive multi-sensorial information about the position of the value relative to the given limits is provided to the end-user.

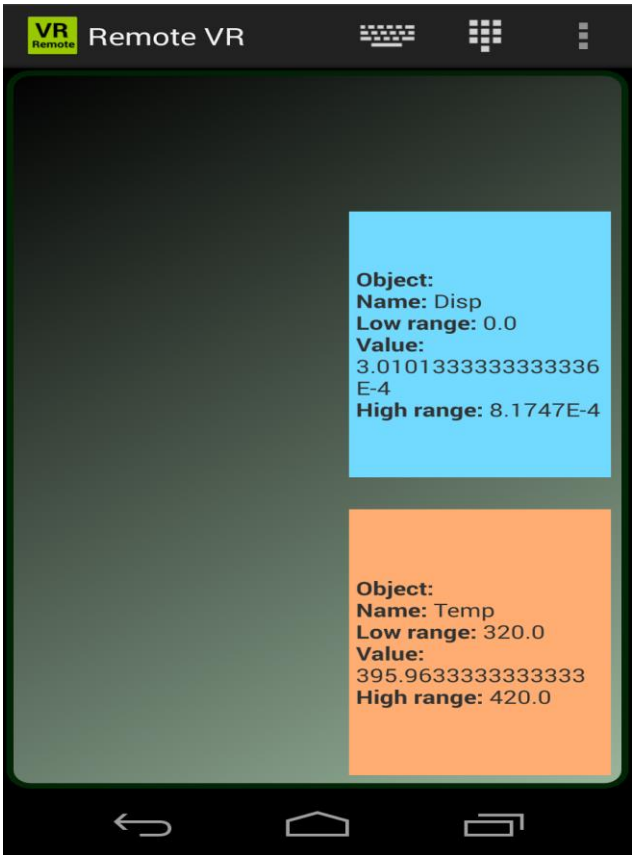

Fig. 7. Graphical User Interface (GUI) of the Android application that presents visually two properties.

The haptic output providing vibration is realized using the Vibrator API [14] according to the following algorithm:

- If the selected value is out of range (lower than the minimal value or higher than the maximal value), a vibration with very high frequency (vibrate 50 milliseconds, sleep 50 milliseconds) is triggered for two seconds;
- If the selected value is in the lowest 25% of the range, the device doesn't vibrate;
- If the range is between 25% and 50%, only a single one-time vibration with duration of 500 milliseconds is triggered;
- In the 50% to 75% range the device will vibrate for two seconds with normal frequency (vibrate 250 milliseconds, sleep 250 milliseconds);
- For the highest 25% of the range the triggered vibration is with high frequency (vibrate 100 milliseconds, sleep 100 milliseconds) and lasts two seconds.

Utilizing the frequency approach, the end-user can easily determine if a given value is within the expected range or may need additional observation and analysis. The tests and questionnaires we performed proved that the presented approach is evaluated better by the end-user when compared to an alternative one: to have a constant vibration frequency and distinguish different values based on the vibration duration (e.g. vibrate 3 seconds if out of range, 2 seconds if in the highest 25%, etc.).

4. Implementation as interface device and examples

4.1 Advantages

The solution proposed here exposes three major advantages for making the interactions in virtual reality environment easier and more intuitive:

First: the whole set of commands, which can be performed using mouse and keyboard, is delivered through a single wireless interface device, which is light enough to be held with one hand for a long period of time. With the provided portability, the immersion in the virtual environment is easier – there is no need to sit behind a desk in order to work with mouse and keyboard, the end user can stand directly in front of a large screen or virtual reality wall.

Second: the interface device itself receives additional information for the virtual object, what is being inspected, which is delivered through visual and tactile channels. This information gives clearer idea of the observed features at specific points of the virtual object, especially when different characteristics are combined. That way the perception of the data becomes easier and hypotheses about correlations between different properties could be explored effortless further in more details.

Third: the proposed solution is tightly integrated with the *Configurator* application, so once set up, it doesn't require additional calibration based on particular details of the virtual reality environment (e.g. type of screen) - it works simply as integrated smart device.

4.2 Experimental Studies

We have analyzed different virtual objects and presented their features using the proposed solution in the Virtual reality laboratory at Technical University of Sofia. Of particular interest to us was the implementation of the described approach for presentation of results from engineering analysis tasks e.g. the distribution of the temperature of the inner and outer surface of mechanical products like a heater. The approach for interaction with this object was performed on three different VR systems: the simplest one using a standard LCD monitor (2D) (Figure 8), a more advanced with a 3D monitor (Figure 9) and an immersive virtual reality system of the power wall type (Figure 10). Although the different screens allow for different levels of immersion, on all systems, the tablet application provided intuitive input alternative to the mouse and keyboard. The best results were obtained on the immersive virtual reality wall, where the advantages of the tablet application over the classic input methods became more obvious.

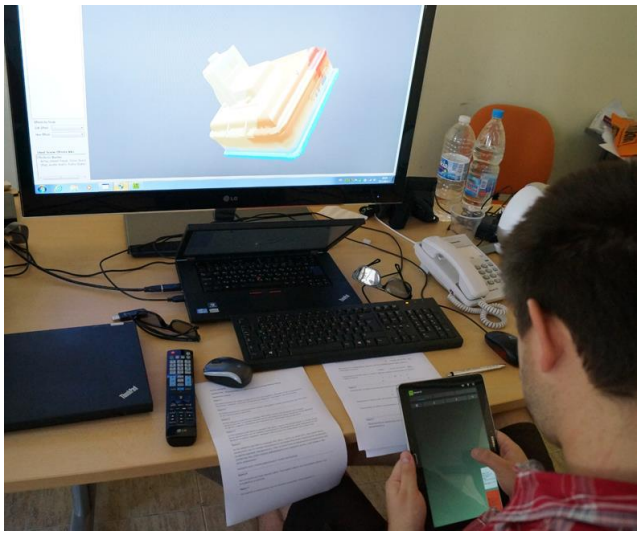

Fig. 8. Interaction in a VR system with regular 2D monitor

Fig. 9. Interaction in a VR system with 3D monitor

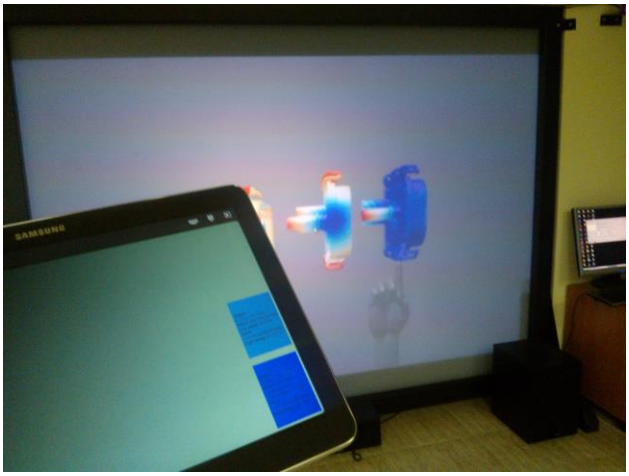

Fig. 10. Interaction in a VR system with immersive power wall

5. Future development

Although the current application bundle implements all planned functions, there are several areas that could be improved in a future version:

5.1 Initial setup

The current approach requires connection setup between the client application on the Android side and two desktop applications acting as servers. If the input processing server could be integrated as a plug-in in Configurator, the initial setup steps for the end user could be minimized which will lead to better use experience.

5.2 Configurable properties

The current implementation of the *Configurator* application assumes that the information for all object properties/characteristics received from Configurator should be visualized and in addition tactile feedback for the first property should be provided. When working with a variety of properties of a given object, the end-user may want to disable tactile feedback or assign the tactile feedback to a different property. In such situation information only for a subset of the received properties can be presented. In addition extended functionality can be exposed to the user giving him/her the choice to show and hide dynamically the presentation of those object properties, which are of a particular interest at any given moment.

5.3 Making use of the gyroscope/accelerometer sensors

Currently only the touchscreen of the tablet is used for user input. Since the mobile devices have also sensors registering different kind of movements (like gyroscope and accelerometer), it would be beneficial to explore the possibilities to utilize these sensors implementation of additional interaction tasks - e.g. navigation in virtual reality environment or performing specific manipulation of virtual object.

6. Conclusions

Our solution allows user-friendly manipulation of object features in virtual reality environment. The multimodal presentation of the object's properties enriches the end-user experience by distributing and delivering the information through different perception channels. That way the end-user can evaluate results from engineering analysis faster and can receive better understanding about the different properties of the virtual objects leading to increased sense of immersion.

Acknowledgments

The results presented in this paper are obtained in the framework of research activities funded by the Internal research grant of the Technical University of Sofia – 2015 and partly supported by LaSciSo (Large Scale Industrial Structural Optimization) project, funded under the FP7 PEOPLE Work Programme, Action IAPP (Grant 285782).

References

[1] A. Bachvarov, S. Maleshkov, D. Chotrov, Extending Configuration and Validation of Customized Products by

Implicit Features in Virtual Reality Environments, Proceedings of the 7th World Conference on Mass Customization, Personalization and Co-Creation (MCPC), Aalborg, Denmark 4-7.02.2014, Lecture Notes in Production Engineering, DOI: 10.1007/978-3-319-04271-8_17, Springer International Publishing Switzerland, 2014, pp. 189-199.

[2] T. M. Buriol, S. Scheer, E. K. Saldanha, "Scientific Visualization and Virtual Reality Techniques Integrated to CAD/CAE Systems for Illumination Design", in International Conference on Information Technology in Construction, 2008

[3] Chotrov D., S. Maleshkov, Simultaneous Bidirectional Geometric Model Synchronization between CAD and VR Applications, Advances in Visual Computing - 9th International Symposium on Visual Computing ISVC'13, July 29-31, 2013, Rethymnon, Crete, Greece, G. Bebis et al. (Eds.) Lecture Notes in Computer Science series - LNCS 8034, Springer-Verlag, Part II, pp. 288-297

[4] S. Jayaram, J. Vance, R. Gadh, U. Jayaram, H. Srinivasan, "Assessment of VR Technology and its Applications to Engineering Problems", in Journal of Computing and Information Science in Engineering, Volume 1, Issue 1, 2001, pp. 72-83

[5] G. Kim, Designing Virtual Reality Systems: The Structured Approach. Springer, London, 2005.

[6] S. Maleshkov, A. Bachvarov, D. Chotrov, J. Ovtcharova, J. Katicic, "Using implicit features for enhancing the immersive object representation in multimodal virtual reality environments", in Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS), 2012 IEEE International Conference, 2012, pp. 91-96, DOI: 10.1109/VECIMS.2012.6273204.

[7] S. Maleshkov, D. Chotrov, "Affordable Virtual Reality System Architecture for Representation of Implicit Object Properties", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 2, July 2012, pp. 23-29.

[8] Maleshkov S., D. Chotrov, Post-processing of Engineering Analysis Results for Visualization in VR System, International Journal of Computer Science Issues, ISSN 1694-0784 Vol. 10, Issue 2, No 2, March 2013, pp. 258-263.

[9] S. Maleshkov, D. Chotrov, A. Bachvarov, K. Kamberov, Multimodal Presentation of FEM Analysis Results in Virtual Reality Environment, International Scientific Conference FIT`2012, ISBN 978-954-438-854-6, 18-20 October 2012, TU-Sofia, Bulgaria, pp. 301-306.

[10] Nachev I., S. Maleshkov, Android-based Control Interface Solution for Windows Applications, Proceedings in Advanced Research in Scientific Areas, The 1st Virtual International Conference ARSA 2012, Published by Thomson ISBN 978-80-554-0606-0, 3-7 December 2012, pp. 2073-2077.

[11] Pirhonen Antti, Hannakaisa Isomaki, Christopher Richard Roast, Pertti Saariluoma, Future Interaction Design, Springer, 2005.

[12] Official Java API net (1993 - 2014) http://docs.oracle.com/javase/7/docs/api/java/net/package-summary.html

[13] Sun, Oracle Java Robot API, 2000, http://docs.oracle.com/javase/7/docs/api/java/awt/Robot.html

[14] Google, Official Android API, 2015, Android XML Pull parser http://developer.android.com/reference/org/xmlpull/v1/package-summary.html; Vibrator API (2015) http://developer.android.com/reference/android/os/Vibrator.html; KeyEvent http://developer.android.com/reference/android/view/KeyEvent.html

Iliyan Nachev has received BSc. (2010) and MSc. (2012) degrees in computer systems and technologies from the Technical University of Sofia, Bulgaria. Currently he is PhD student, acting at the Virtual reality lab.

Stoyan Maleshkov has Eng. degree in system and control engineering (1975), master in applied mathematics (1977) and PhD in computer aided system design (1981), all received from the Technical University (TU) of Sofia, Bulgaria. Fulbright scholar (1989–1990) at the Interactive Modeling Research Lab, Louisiana State University, Baton Rouge, USA. Professor of computer aided engineering and computer graphics at the TU of Sofia. Department chair (2000-2004) and vice dean (2004-2008), both at the TU Sofia. Since 2008: Head of the Virtual reality lab, TU Sofia. Professor of computer graphics at the New Bulgarian University, Sofia, as a second job. IEEE member.