Available online at www.sciencedirect.com
ScienceDirect
Computers & Graphics 30 (2006) 714–726
COMPUTERS & GRAPHICS
www.elsevier.com/locate/cag

Beyond traditional interaction in a mobile environment: New approach to 3D scene rendering

Zdenek Mikovec*, Ladislav Cmolik, Jiri Kopsa, Pavel Slavik

Czech Technical University in Prague, Karlovo Namesti 13, 121 35 Praha 2, Czech Republic

Abstract

In this paper the problems of user interaction in a mobile environment are investigated. Interaction in this environment imposes new requirements both on UI designers and the users. This paper deals in particular with problems of graphical interaction in a mobile environment. The interaction in a mobile environment requires new approaches that will result in a new type of rendering of graphical data. A rendering technique that enables rendering and annotation of objects in a 3D scene on mobile devices is presented. This technique is based on transformation of the 3D scene to a 2D vector graphic representation. The input 3D scene is given in the VRML format and the output 2D format is SVG. In the second part of this paper we are presenting a prototype of a voice user interface that allows users to interact with the graphical data in a mobile environment. The semantic description of the scene plays the key role in restriction of the language used for communication. The communication between the user and the mobile system is performed by means of natural language that is restricted according to the context the user is currently in. The implementation of the voice user interface is based on the existing VoiceXML platform.

Keywords: Mobile computing; User interfaces; Graphical interaction; Voice recognition; Ontology; XML; SVG; OWL

1. Introduction

In this paper we will deal with problems of user interaction in a mobile environment. Interaction in this environment imposes new requirements both on UI designers and the users. We will focus on interaction with application data that have graphical flavor (like blueprints, plans, etc.). The interaction with this type of data has been performed for decades in a static environment (e.g. PC, characterized by large screen, mouse, keyboard) for which some feasible interaction principles were developed. The interaction in a mobile environment requires new approaches that will result in a new type of rendering of application data.

These new requirements stem from several new issues typical for mobile computing: small screens, context dependency, new interaction devices like a stylus, dynamically changing environments, often task switching based on external events, etc. Such an environment will impose significant limits on interaction as the user's comfort when using mobile devices is much lower than in a static (e.g. PC) environment. If we focus only on the modification of the user interface we will very quickly encounter limits determined by the application data presented. To get beyond these limitations and reach much higher adaptability of the interaction we need to introduce new methods for application data rendering and interaction.

The rendering can be in a broader sense understood as a way to present application data to the user, by means of targeting different human sense organs. In our case we focus on two senses: seeing and hearing and thus we can divide the presentation modalities to visual and audio ones (see Fig. 1). The visual one will be based on traditional graphical rendering methods, which will be modified according to the situation in a mobile environment. The audio modality in our case will be text based, where by means of verbal communication it will be possible to get information about the structure of the scene. In this case

*Corresponding author. Tel.: +420 22435 7647; fax: +420 224 923 325.
E-mail addresses: xmikovec@fel.cvut.cz (Z. Mikovec), cmolikl@fel.cvut.cz (L. Cmolik), j.kopsa@fee.ctup.cz (J. Kopsa), slavik@fel.cvut.cz (P. Slavik).

0097-8493/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2006.07.022

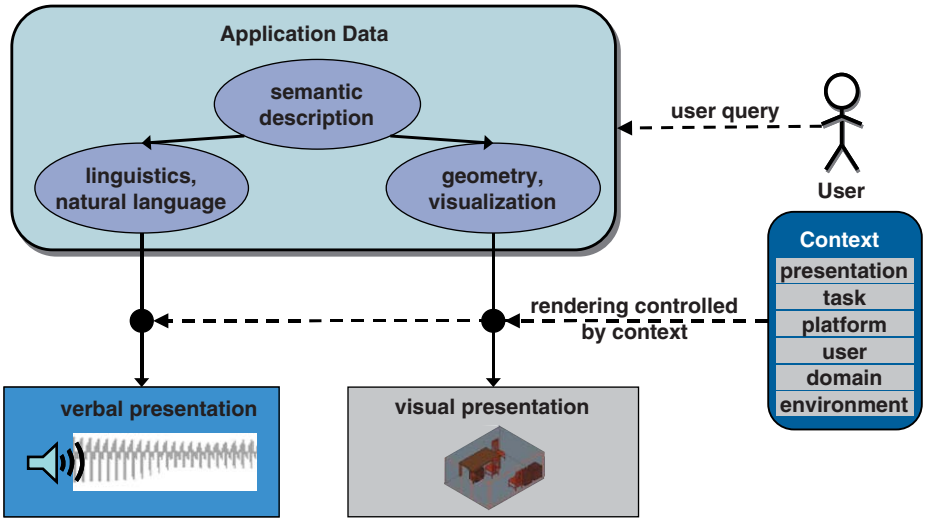

Fig. 1. Two types of rendering of graphical data.

the user formulates proper queries and the system answers in natural language.

The interaction with application data is in our case understood as a solution of navigation problems in 3D graphical data. We also assume simple manipulation with the data (e.g. creation of annotation to an already existing object).

For satisfactory adaptability of the rendering and interaction it is necessary to get information about the structure and the meaning of the data (3D scene in our case). To achieve this we introduce a semantic description of the application data, which is the starting point of both of our rendering methods (see Fig. 1).

Our goal is also to allow natural mixing of both types of rendering driven by the current user context (see Fig. 1). It is obvious that various aspects of context will influence the way application data are rendered and the corresponding interaction. The context in general imposes various constraints on user activity. These constraints stem both from limitations on the side of a device (like a small screen) or on the side of the user (some kind of impairment) or they can be the result of a specific situation the user is currently in (like being in a certain floor of a large building where the inspection is performed—see Section 2). All these constraints may in general influence the method of interaction with application data, thus influencing the particular way of data rendering (e.g. defining the user's viewpoint in the 3D scene) and the user interface by means of which we interact with graphical data. As the problem is rather general we will concentrate on specific issues that will show our approach to the solution of some aspects of this general problem. In our research we concentrated on development of new methods for scene rendering and interaction with such a scene.

In Section 3 of this paper we will discuss modification of the graphical rendering process (adjusted to the needs of a mobile computing environment) [1]. In Section 4 we will discuss a new approach to textual based rendering that could be an alternative to graphical rendering in a specific context the users might be in [2]. Under textual rendering we will understand the process where the users acquire the information about the structure of a 3D scene by means of textual based interaction (queries and answers are in textual form) in audio modality. Both approaches have been tested on a number of examples and the results were very encouraging.

2. Use case

Our use case presented below is derived from the facility management (FM) application domain. In the FM domain the workers typically work on move. While walking through different buildings they perform various tasks like checking the inventory, finding hazardous materials, etc. In our particular use case the worker is called inspector. The inspector's task is to match the inventory in the building with the information stored in the inventory system (see Fig. 2). The inspector has at his/her disposal an inventory system on a mobile computer (like a PDA, or TabletPC). The inspector enters the office room and starts the inspection by asking the system verbally to list the inventory items in the investigated room (in this case an office). The system also answers verbally by listing the inventory numbers of all items stored in the system of the office room. The inspector finds some discrepancies (like wrong inventory number, missing item) and corrects them immediately (still in voice modality). This audio modality is very convenient for the inspector, because the inspector has free hands for manipulating the inventory items and is not limited when reading the inventory numbers. After this activity the inspector needs to do a deeper check of the types of inventory items (like the exact position, shape and material of the items). For this task the inspector switches to the visual modality, where the system offers a 2.5D view of the office room. The inspector can navigate in the visualized scene and finds that one container has a different

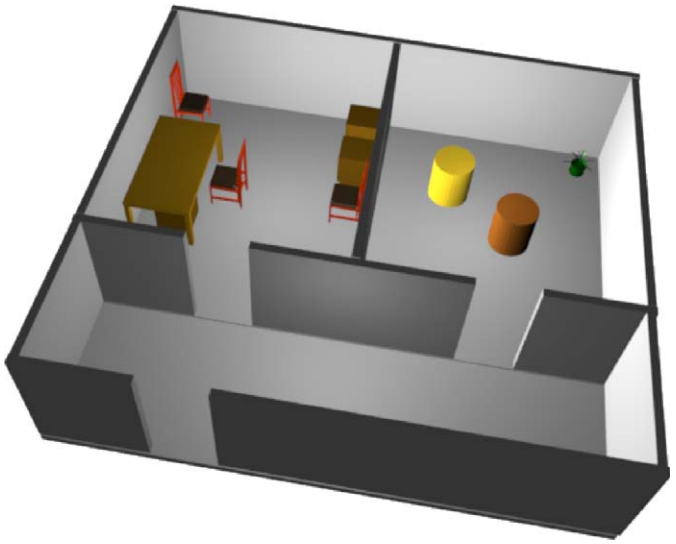

Fig. 2. A 3D plan of the building floor inspected by the inspector.

height (different number of drawers). The inspector makes a quick short annotation to the container in the scene noting this error. This annotation is done in audio modality.

As shown in our use case for efficient task solving the inspector needs to have specific tools at his/her disposal to investigate, annotate and interact with the application data in a mobile environment.

The tools must provide information in two modalities (visual—in a form of building plan; audio—in a form of structured textual description of the building and its inventory) to increase the efficiency of the inspection. The switching between these two modalities must be possible at any time.

On the presented use case we can see that the tools should have a more flexible user interface (multimodal one) and should be less demanding from the point of computational power (as mobile devices and wireless networks are much less powerful than standard PCs and LANs).

3. Graphical based rendering for mobile environment

The interaction with 3D scenes in a mobile environment is very difficult. The problem of navigation in the 3D scene is especially noticeable. In a static environment the navigation requires concurrent usage of special keys to switch between navigation modes (move, pan, rotate, fly, walk, etc.), in combination with a pointing device (mouse, joystick, etc.). Mobile devices do not allow us to perform such navigation comfortably. Moreover, common users are not trained to work in the 3D environment with so many degrees of freedom, which results in the loss of orientation in the 3D scene. As mentioned in [3] users prefer a 2D interaction environment or some kind of combination of the 2D and the 3D environment. The 2D interaction environment reduces the user interaction to two basic functions—zoom and pan, which is a much safer environment, where the user in general does not lose orientation and can navigate much faster.

There exist several approaches to the rendering of 3D scenes on mobile devices, which can be divided into three categories:

- Server-side methods. These methods [4] provide remote rendering of the 3D model as a reaction to user-interaction-requests from the mobile device and send rendered raster images back to the mobile device.
- Client-side methods. These methods [5] render the 3D scene completely on the mobile device.
- Hybrid-side methods. These methods [6,7] balance the workload between the server and the client to decrease network communication and improve performance on the client side.

All of the solutions mentioned above focus only on the rendering or on the load distribution between server and client side. None of the mentioned works focuses on the user navigation and interaction nor takes the navigation or interaction into account.

3.1. Our approach

In our solution we have started with the analysis of user needs. As mentioned before, the users prefer the safer 2D interaction environment or a combination of the 2D and 3D interaction environment. Therefore, we introduce a solution that allows the users to work in the 2D interaction environment while preserving their illusion of being in 3D.

Our approach is a hybrid-side method. The distribution of the workload between server and the client (mobile device) plays a very important role in our solution of the rendering problem in the mobile environment. Our approach is based on transformation of the boundary representation of a 3D scene by means of a projection to the 2D vector image (see Fig. 3). The transformation preserves the object oriented representation of the 3D scene—each 3D object can be wholly (all faces) identified in the 2D vector image. The resulting 2D vector image has 2.5D representation (each 2D object is located in its own projection plane and these objects are ordered by their distance from the camera). In other words, each face from the 3D scene has its representation in the 2D vector image (see Fig. 4). This object oriented approach allows the user to interact with the objects in the 2.5D representation.

The scalable vector graphics (SVG) format [8] was chosen to implement these features. SVG is an XML-based format developed for description of 2D vector graphics. It is object oriented and supports zooming, panning and interaction with the objects. Each SVG conformant viewer implements the painter's rendering model [8]. An object defined later in the SVG file is painted over objects defined before. This could be interpreted as the objects defined later in the SVG file are nearer to the viewer than objects

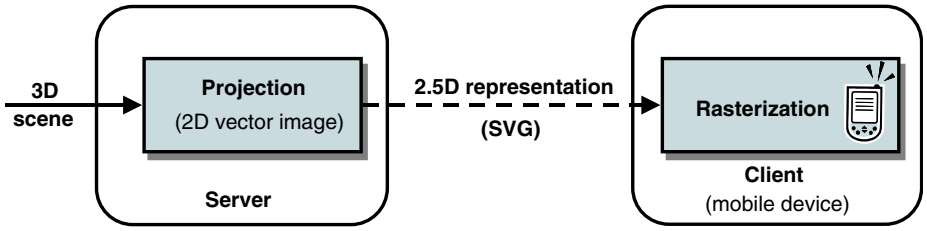

Fig. 3. Distribution of the rendering process of 3D scene.

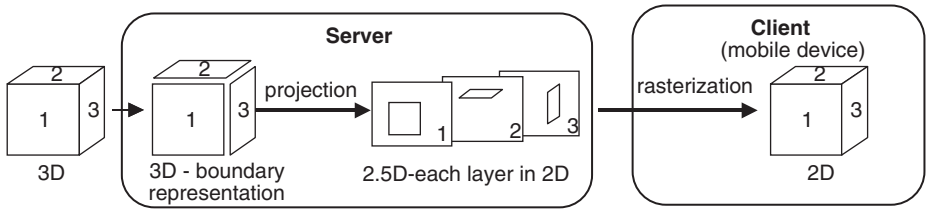

Fig. 4. Projection of a 3D object into the 2D vector image. Each face of the 3D object is projected into a separate layer in 2D vector image. This representation is in fact a 2.5D representation.

defined before. Thus, we can project each face in a 3D scene into a separate layer and use the SVG as the 2.5D representation format. Only the ordering of the faces is known, but the real distance of faces from the viewer is unknown.

It was necessary to develop an algorithm that sorts the faces in a 3D scene in such a way, that a proper image is obtained if the painter's rendering model is applied to them. Such an algorithm is based on the priority list approach introduced by Newell et al. [9]—also known as the painter's algorithm. In our approach the painter's algorithm generates an ordered list of 2D projections of 3D faces. Each individual 2D projection is in vector representation (SVG) and thus the whole SVG file is for the current view a 2.5D representation of the 3D scene.

The transformation of a 3D scene to 2.5D representation is performed on the server side by a modified rendering pipeline. The structure of our modified rendering pipeline is a standard one, except the back-faces are not culled and the solution of visibility is performed in the object space by the painter's algorithm. All parts of the algorithm are modified to be able to handle back-faces. The back-faces are preserved to allow various rendering modes. The 2.5D representation of the 3D scene is sent to the mobile device, where it is rendered. The rendering of 2D data is computationally less expensive than the rendering of 3D data and therefore the response of the mobile device on user interactions is faster.

3.2. Various rendering modes of projected data

The painter's algorithm preserves the information about all objects and also about their ordering with respect to the camera view. Investigation of the 3D scene in 2D space does not allow the user to access all the 3D information (like information about an object hidden behind another one). Our approach reduces this problem by benefiting from the 2.5D representation and by introducing special modes of object rendering.

The SVG used for 2.5D representation of the 3D scene is an XML-based format, therefore the CSS (Cascading Style Sheets) can be used to define various rendering modes of the 2D objects. Thanks to our modified rendering pipeline, which preserves back-faces, we are also able to introduce wire-frame and semitransparent rendering modes. To each SVG file a CSS representing the rendering modes is attached. Moreover, SVG enables interactive and selective change of a rendering mode of an object in the SVG file. This support of various rendering modes compensates in some way the movement of the avatar in the 3D scene. The virtual walkthrough is replaced by features allowing the user to investigate the structure of the 3D scene. We will demonstrate this feature on the earlier described use case (see Section 2).

3.3. Use case

Let us consider an inspector equipped with a mobile device (e.g. PDA) visiting an office (having a 3D model on the server). The task of the inspector is to revise the facility of the office. The inspector retrieves the information needed from the server through the wireless network. The inspector uses a 2.5D environment to investigate the office. The office was selected based on the current context of the inspector. In our use case the inspector requests the 3D data by uttering the following request: "Show me a 3D visualization of the office!" (see Section 4). The data delivered to the PDA are not in the original 3D form (that is not suitable for use in a mobile environment) but they have been transformed to 2.5D representation by means of the process described above. For each room four 2.5D images are created. Each image defines one view of the

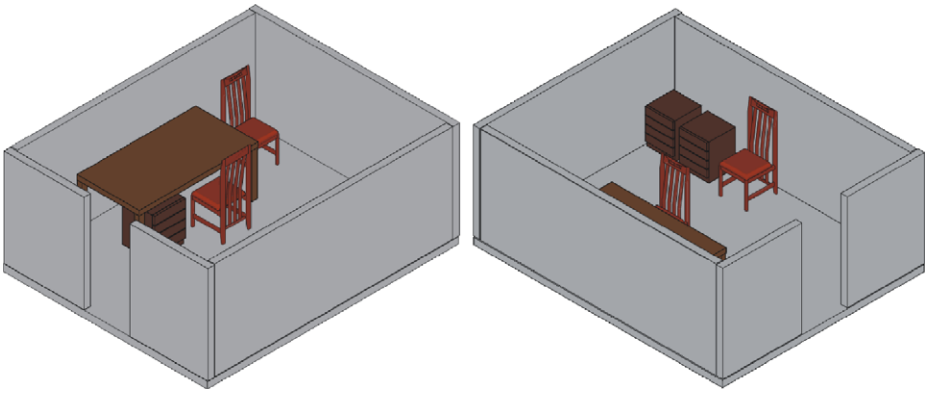

Fig. 5. Examples of initially rendered 3D scene (the office room from two different viewpoints).

room (see Fig. 5 for two of them) form different viewpoint. This allows the inspector to partially rotate the room by selecting the proper view.

The inspector can view the 2.5D representation in various rendering modes (the objects can be displayed in solid, semitransparent or wire-frame mode). Moreover, zooming and panning can be used without sending any request to the server. The inspector can also annotate the objects in 2.5D representation to record discrepancies found (in comparison with reality) in the data.

The inspector needs to examine the scene in detail. The 2.5D representation in Fig. 5 is not informative enough, because some objects can be hidden behind other objects. The inspector changes the rendering mode of the front walls to wire-frame mode to see what objects are hidden behind them (see Figs. 6 and 7).

By using the zoom function the inspector can get closer to the hidden objects and inspect them. While the objects are projected with parallel projection and described in 2D vector graphic the rendering quality during the zooming is preserved and from the inspector point of view it looks exactly the same as zoom in the original 3D scene.

Now the inspector has found out that there is a container with four drawers below the table, but in reality there is a container with three drawers. Therefore, the inspector wants to attach an annotation to the container. For annotation purposes s/he zooms the container in order to create an unambiguous relation between the annotation and the container (see Fig. 8).

Now the inspector can create a multimedia annotation, e.g. voice record and attach it to the container (see Section 4).

Notice that the whole process of scene exploration is done only on the client side (no requests to server are sent). The request is sent only in the phase of annotation creation or when the viewpoint is changed.

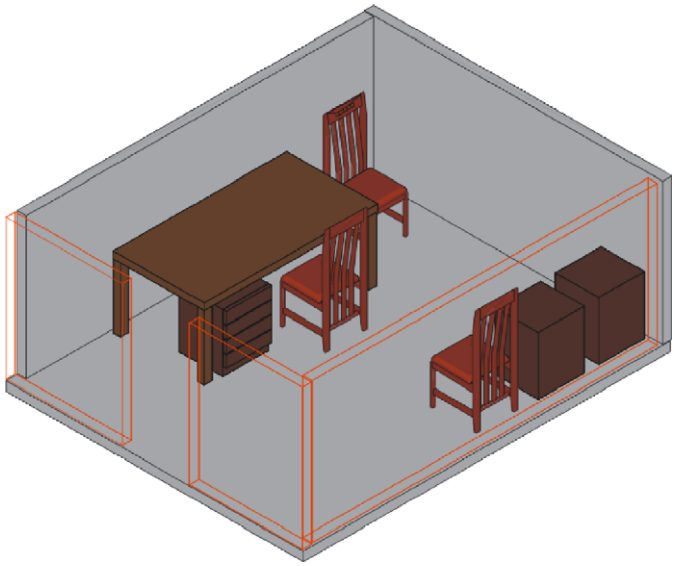

Fig. 6. The front walls rendered in wire-frame mode. The furniture hidden behind the walls could be identified.

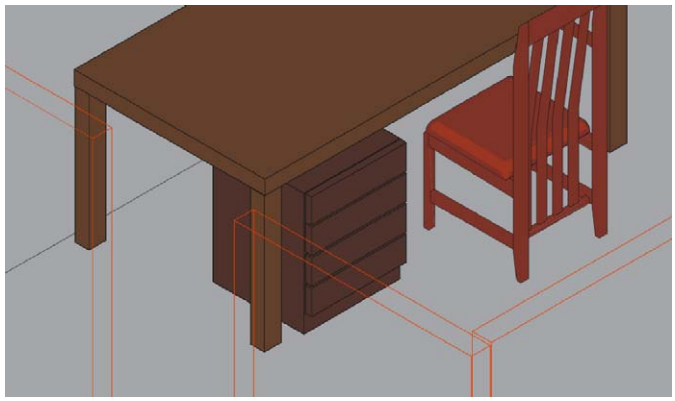

Fig. 7. Inspector performs zooming function to investigate the objects.

3.4. Testing of rendering method performance

This testing was focused on the performance of our application. The performance test was focused on solution of visibility where our modified painter's algorithm spends a significant amount of time from the whole time spent on the rendering. Our implementation [19] was tested on various 3D models of different complexity (number of

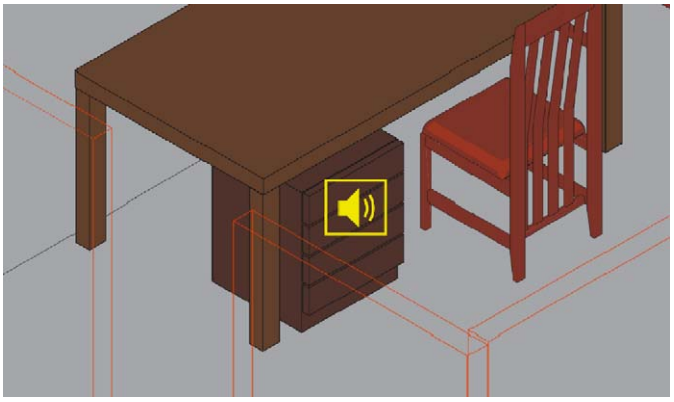

Fig. 8. The inspector has created an unambiguous relation between the container and the annotation.

model	# of surfaces	visibility time [ms]	file size [B]	compresed file size [B]	compression ratio
basic shapes	628	230	118 657	15 320	7.75
wheel	928	611	194 448	83 658	2.32
blender monkey	946	490	183 925	25 148	7.31
holes	1 058	581	212 150	26 310	8.06
torus knot	1 440	771	262 420	33 764	7.77
blender monkey 2	2 032	1 301	424 641	62 593	6.78
stanford bunny	2 915	2 593	536 186	67 053	8.00
pillar	5 260	6 590	953 910	130 415	7.31
cow	5 804	5 668	1 098 199	141 012	7.79
blender monkey 3	8 128	13 529	1 644 489	239 314	6.87

Fig. 9. Performance time of visibility computation and file size reduction after gzip compression.

surfaces). The time spent on the painter's algorithm and the size of the SVG file was measured. The data were measured on a PC equipped with Pentium 4 Mobile running at 1600 MHz and with 512 MB of RAM. The results are presented in the table in Fig. 9.

The size of the SVG file depends linearly on the number of faces. The computational complexity depends quadratically on the number of faces (see graph in Fig. 10). This corresponds with the $O(n)^2$ computational complexity of the painter's algorithm. The measurement showed that the performance of our implementation of the painter's algorithm is sufficient for scenes with a relatively small number of faces (up to 2000). In such a case the time of 2.5D representation generation requires about 1 s. The typical scene used in our use case contains up to 1000 faces. These scenes can be easily handled in a mobile environment.

We should also mention that the size of the SVG file is always larger than the size of the file describing the corresponding 3D scene in VRML format [10]. The SVG file contains much more detailed description of the scene, e.g. the color is stored in the SVG file for every face. In VRML usually one color is defined for the whole object. The size of the SVG file does not create any problems during data transmission as it was compressed in our experiments with compression ratio from 6.78:1 to 8.06:1.

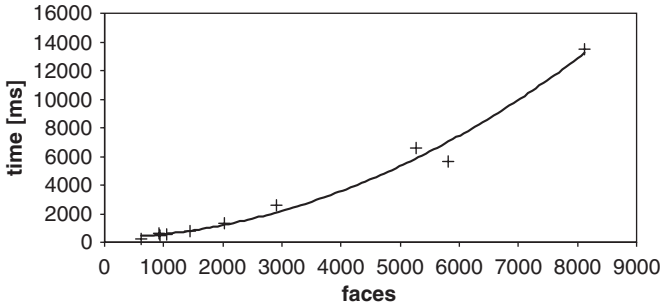

Fig. 10. Performance of our painter's algorithm in dependency on number of faces.

3.5. Usability testing

This test was focused on the approval of our assumption that our approach leads to simplification of the user interaction (by introducing a 2D environment for manipulation with the 3D scene) while the feelings of the user will be as close as possible to a true 3D scene exploration (by introducing the 2.5D representation and rendering modes).

We performed usability tests with six selected users on two different scenarios. All users were exploring the same scene, but half of them in a 3D environment and half in our 2D environment.

The first scenario was similar to the use case described in Section 3.3. The task was to explore the given scene and to check for all objects in the scene. The test results showed that the users had much less problems with navigation in our 2D environment. Moreover after a short time of using our system they forget that they are only in 2D space, which proved that our system induced an illusion of being in 3D space.

In the second scenario the users were exploring one street of a city (see Fig. 11). Their task was to perform a short walkthrough in the scene and again to check for all objects in the scene. The results are similar to results of the first scenario. The users exploring the scene in our 2D environment had again much less navigation problems than users exploring the scene in the 3D environment. A very interesting result of the second scenario is that users exploring the scene in our 2D environment used mostly the zoom function to move backward and forward and suppressed usage of the panning function.

4. Textual based rendering

In a mobile environment the users often get into a situation where they need eyes and hands free for performing their task (e.g. carrying a bag, taking samples of materials with gloves on their hands). At the same time they would like to interact with their mobile device to browse through some information or to make an electronic note. It is obvious that the common way of interaction with the mobile device (using stylus with touch screen and watching the display) becomes unusable in such a situation.

Fig. 11. Walkthrough in the 3D scene. The user is exploring the 3D scene in a 2D environment. Due to the perspective projection and vector nature of the graphics the zooming action is perceived by the user as walking on the street.

We have to switch to different interaction modality—to an audio one. Unfortunately, the existing solutions for audio interaction face serious usability issues when trying to introduce natural language interfaces. Especially in a mobile environment, where the end-devices have low computational power and lots of disturbing noises occur, the ability of such systems to understand the user queries decreases dramatically and the system becomes unusable.

The success rate of understanding user queries is often improved by restricting the language to a specific application domain [11,12]. In a broader sense contextual information is used in most of the voice applications to cope with the restriction process. The contextual information is obtained from various sources (sensors of user gestures, environment sensors) and used to restrict the language as well as to resolve the semantic ambiguity of the natural language as described in [13–15]. This approach does not provide the system with sufficiently detailed context information. There exist several approaches like [16] which try to build up more general multimodal interfaces for multiple applications. These approaches are based on the semantic description of the application domain (by means of ontology), which should be automatically connected with the generic multimodal interface.

In our case we need to interact with graphical data by means of audio modality. This leads to introduction of textual based rendering methods, which will allow verbal communication. To offer the user a comfortable natural voice interface, the communication language starts to be very complex too. Such a complex language cannot be satisfactorily handled by the general voice recognition system.

4.1. Our approach

The large size of the language needed to describe complex graphical data is the main issue. Our solution is based on typical use case in a mobile environment, where the user is moving from one place to another and solving problems related to those places. In our particular use case (see Section 2) the user is an inspector equipped with a mobile device inspecting the facility of the warehouse. The conversation context can be defined with respect to the inspector's context (environment, current task, etc.—see Fig. 1). This conversation context is used to reduce the size of the language to a subset, which will be sufficient for the expected conversation.

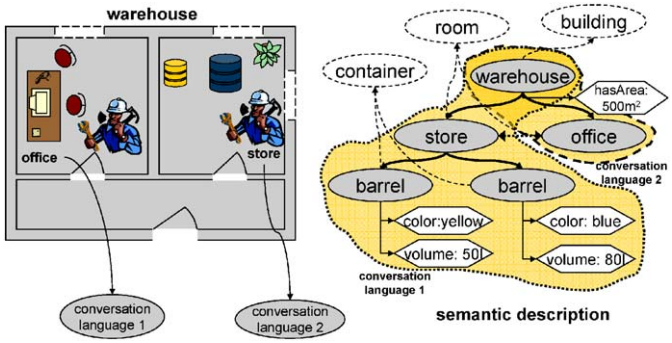

Fig. 12. Use case of construction site inspection: the inspector location restricts the conversation language. The language is derived from an appropriate part of the semantic description.

Fig. 12 explains the approach presented. When the workers are in the store of the warehouse, they will most probably ask about the barrels that are placed in that room or about objects that are related to them. Based on the workers location we generate restricted conversation language, which covers only the most probable discussion topics (see "conversation language 1 and 2" in Fig. 12). The probability that the workers will ask about the objects in the office is significantly lower in the given context. If the inspector asks about objects that are out of the current restricted conversation language, the system will not understand and remind the inspector about the current conversation context (e.g. location of the inspector).

The novelty of our approach is that the conversation language is defined by both contextual information (described in Fig. 1) and the semantic description of application data (see Fig. 12)—in our case the construction site plan—which is the subject of conversation. By introducing the semantic description we obtained much more detailed information about the conversation context. We are able to dynamically and seamlessly (from the user's point of view) restrict the conversation language according to the real conversation context. In every particular moment of the conversation only a relevant subset of the language is used (in other words: a specific context oriented language will be used). The union of these particular languages represents a general language large enough to

cover the general conversation about the given topic (in our case the union of all restricted languages covers all the rooms in the building and activities that could be performed there).

Besides navigation in the scene our approach also solves the problem of manipulation with the 3D scene. This manipulation includes both annotation and modification of the scene. In our system we understand that annotation is the possibility to create a multimedia content (e.g. voice record in WAV format, photo in JPEG). Modification might concern both geometric parameters and a semantic description (related to application domain) of the scene. For annotation there exists a set of commands the user has at his/her disposal.

The whole conversation with the system is led by user-initiative conversation; the user forms queries and commands and the system answers by providing requested information about the scene or performing the command.

In the following text we will describe our approach as a text-based communication between the user and the system. Formulation of queries and the system response will be handled on the textual level. In a mobile environment textual communication is not an adequate one, so the next step was the conversion of text-based communication into a voice-based one. This conversion was realized by means of the voice recognizer developed by IBM and an open-source voice synthesizer.

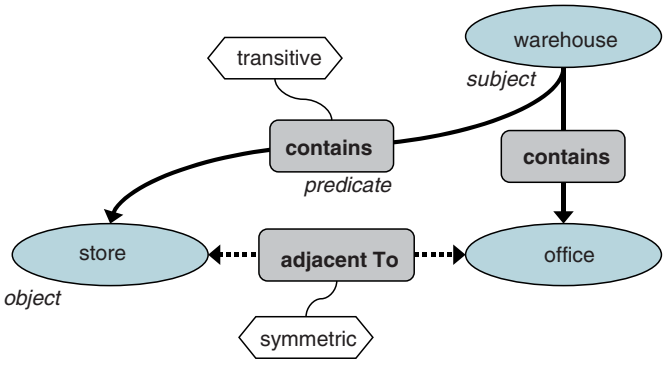

Fig. 13. Ontology structure—triplets: subject–predicate–object.

4.2. Semantic description—ontologies

Formally, the ontology used for semantic description consists of elements, their properties and relationships between them. The ontology is represented as an oriented graph with elements as nodes and relations as edges. An edge and its two nodes represent a fact in the form of a triplet "subject–predicate–object" (see Fig. 13). For example, the relation between the nodes "warehouse" and "store" represented by the predicate "contains" describes the following fact:

warehouse contains store

Element or relation properties may be of different types (property type) like transitive or symmetric. The transitivity of the relation (predicate) "contains" may result in the following fact (see Fig. 14):

warehouse contains barrel with inventory number 1

From this example we can see how the ontology is structured and interpreted.

The application data (3D scenes in our case) are semantically described in textual form—as an ontology (OWL [17]), which can be perceived as an oriented graph. The semantic description is created manually during the authoring process. In this case the textual based interaction is the solution to the problem given above. An ontology example (where combination of graphical data and its semantic description is stored) is presented in Fig. 14.

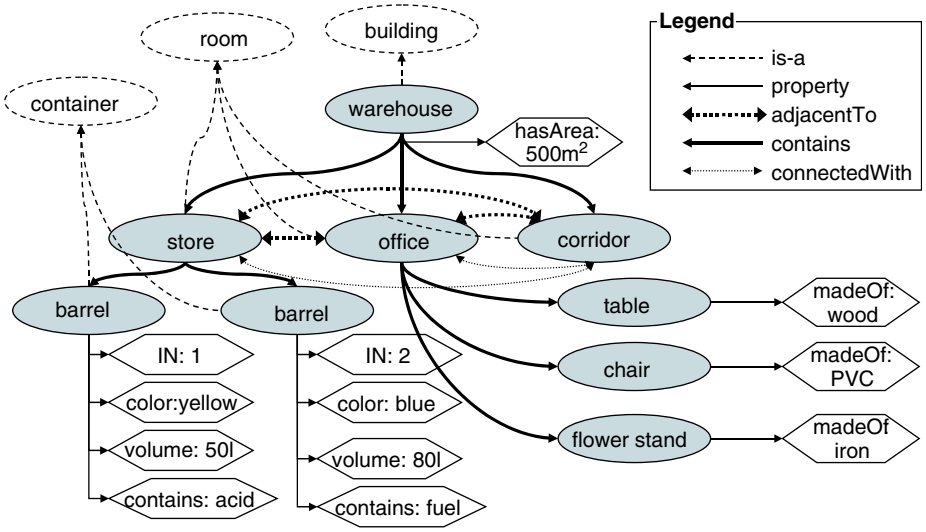

Fig. 14. Semantic description of application data (construction site shown in Fig. 12).

The semantic description consists of a domain and application specific description. The domain description defines abstract terms, classes of objects and their relations (marked by dashed lines in Fig. 14), while the application specific description specifies objects describing a particular construction site or facility. In Fig. 14 the nodes and edges marked with dashed line represent the domain description and the remaining nodes and edges represent the application description. For example, the room node has more general meaning than the store and office objects (the dashed edges represent links between abstract and concrete entities).

4.3. Conversation

The conversation from the user's point of view consists of formulating queries or commands. The commands are added to the conversation language based on the task model of the user (e.g. switching between graphical and audio modality; creating voice annotation for an existing object). As outlined above, the user can also formulate queries to ask about the facts (semantic description) stored in the semantic description (Fig. 14).

The key feature of our solution is that the contextual information is integrated with the semantic description. All necessary contextual information in sufficient detail is available to our system. Because of the linguistic markings made to the semantic description it is possible to access the application data in a natural language (in our case voice-based communication).

There are usually several forms of the same query (the use of synonyms). The query is specified with two key items—source set and target—and several other less important attributes.

The source set represents a node or set of nodes whose properties the users are interested in. The target specifies the properties, which the user is interested in. The two synonymic queries in the following example have the same meaning. Its source set contains one object—*store*—and the target is the property *contains*.

H *(Human)*: What is contained in the store?
OR
H: List the contents of the store!

The corresponding query is executed and the result of its execution is formed into an answer. The system would respond in the following way:

C *(Computer)*: It contains two containers.

For the construction of this answer the generalization stored in the semantic description is used. The abstract terms (like container) are defined as a general term for two barrels located in the store (see Fig. 14). This abstract term is used for constructing the answer given above. The plural form of the word container is defined by the linguistic marking. In the process of the query execution the conversation context is updated. Then the conversation language is modified based on the updated conversation context. The conversation context holds the state of the conversation. This conversation state is defined by the current user context, pointer to the objects in the ontology and conversation history—see Fig. 15. Except in the restricted language production, the conversation context is also used to resolve query ambiguities. For example, the identifier *store* used in previous user queries may be ambiguous since there may be a number of objects with the name "store". However, it can be resolved according to the current user location contained in the user context.

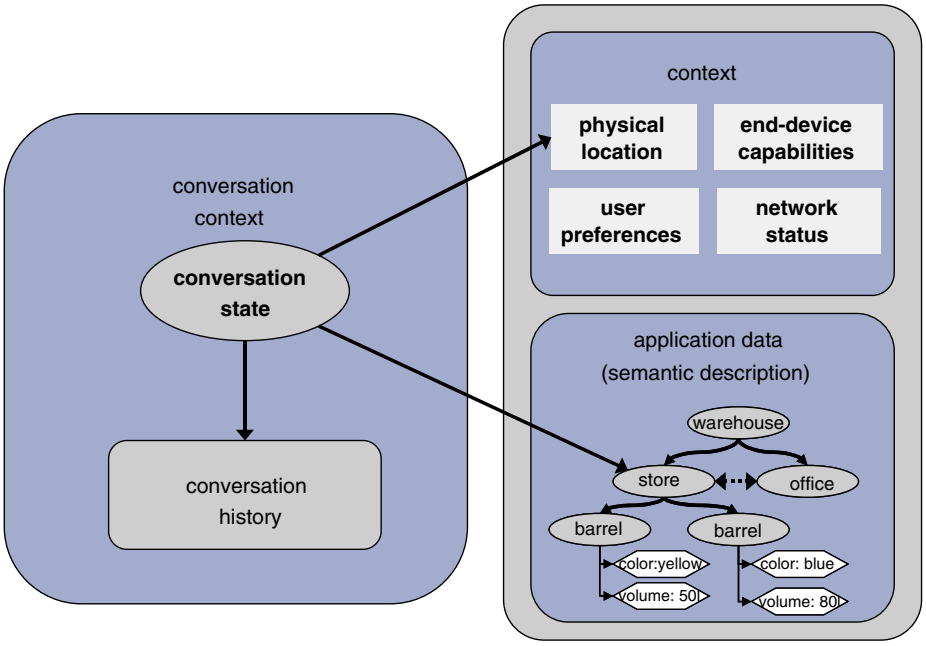

Fig. 15. Conversation context.

The conversation history is a queue that contains references to recently discussed objects of the ontology. It is used to resolve ambiguities of the natural language. For example, the conversation may continue in this way:

H: What is its area?
C: Its area is 500 m^2.

The pronoun *its* is resolved to object *store* by searching the conversation history. The notion of the conversation context is shown in Fig. 15.

The user can also create objects by forming special commands:

H: Add new sample 123 to barrel with inventory number one.
C: A sample 123 related to barrel with inventory number one was added!

This is aimed to fulfill the use case of taking samples—the inspector wears gloves and takes samples with tools. At the same time the inspector creates voice annotation describing the process of taking samples. The physical sample and the annotation are interlinked via the unique identification (in our case "`123`").

The range of allowed queries and commands is quite broad, e.g. the user may also specify a condition that must be met for all objects that are included in the answer:

H: List containers contained in the store and manufactured by Liquids Ltd.

For the switching between the two modalities (textual one and the graphical one) there is a set of commands. One form can be as follows:

H: Show me a 3D visualization of this room.

The detail specification of the whole language along with its semantics, query execution, and answers formatting processes can be found in [18].

4.4. Toward the natural language

The main task when designing the natural language UI was to bridge the gap between the application data and the natural language. The application data described by the ontology does not contain information essential for the system to understand the user queries formulated in a natural language. Also without this information it is impossible to generate system answers to the user in a natural language as well. We have introduced linguistic patterns for generation of sentences from application data in a natural language (see Fig. 16). These linguistic patterns are special attributes added to the abstract level of the application data description (domain description—see Section 4.2). These attributes are then used for generation of the linguistic markings.

The linguistic markings play the fundamental role in the sentence generation process. These markings are generated by the creator of application data or automatically from the domain description linked with ontology elements. These linguistic markings control the process of creation of sentences in a natural language. During the control process the words are modified and placed in a proper place in the sentence in accordance with grammatical rules of a given natural language. The structure of ontology that allows us to analyze and generate sentences for a class of scenes is quite general.

4.5. System architecture

Text-based interaction performed by means of manual input is not suitable for a mobile environment. That is why the user input in a mobile environment should have the form of voice input. We have used the existing VoiceXML server platform for speech recognition and synthesis. The voice recognition part has been developed by IBM. It is configured to request VoiceXML documents from our system, which implements the interaction logic, conversation state management and application ontology data retrieval. The mobile devices are connected to our system with voice-over-IP clients. The architecture of the system implemented is shown in Fig. 17. Our system that we implemented fulfills both requirements for flexible communication between server and client and the easy conversation between the user and the system in a natural language.

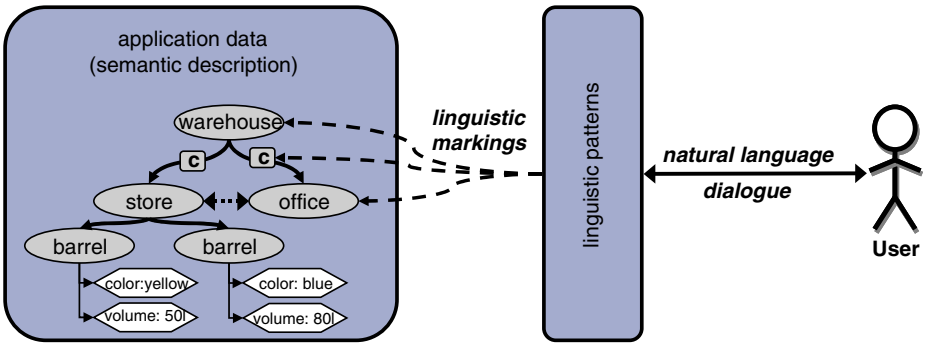

Fig. 16. Linguistic patterns used for transformation of application data into natural language.

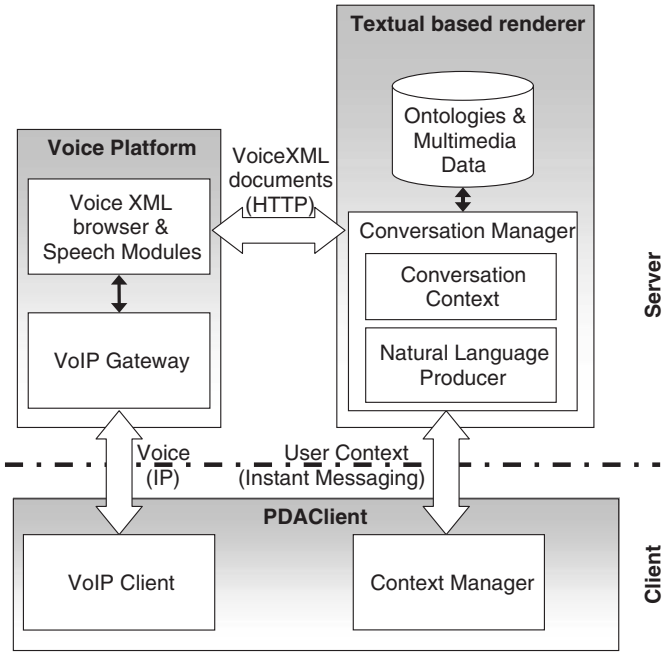

Fig. 17. System architecture.

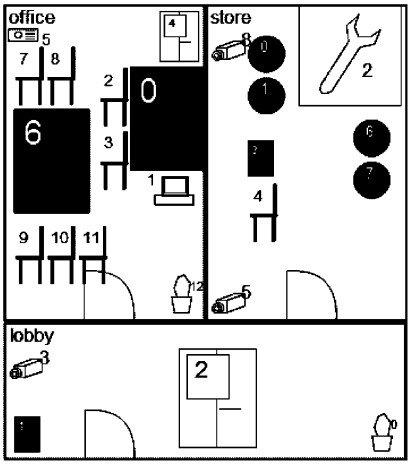

Fig. 18. The environment setup of the usability test.

4.6. Testing our approach

In this paper we have presented an ongoing work, but we are still in the process of development. We have implemented a prototype for testing purposes. The user acceptance of this communication method was tested by means of several usability tests. Our usability tests were focused on two aspects: first, to determine the size of the input language where the speech recognition system is reliable on the required level, second, to test our hypothesis that during the conversation we are able to dynamically reduce the input language (with respect to the current user context and detailed application ontology description) in such a way that the users will not be restricted in their queries (see Sections 4.1and 4.4) (Fig. 18).

The usability test was performed with 12 users. The user's task was to do an inspection in several rooms of a building. They had to check the objects in those rooms and make voice annotations where necessary. During the test we have determined the size of the input language where the speech recognition system was at least 90% successful in recognizing the user queries. The conversation context given by the user context, application ontology and conversation history was continuously changing and based on its current state the input language was dynamically generated to allow the natural language conversation with the user.

5. Conclusion

In this work new interaction methods for manipulation with 3D scenes suitable for a mobile environment have been presented. These methods are based on changing the rendering process of the application data to allow much higher adaptability of the whole interactive system to the user needs.

The first method is focused on modification of the graphical rendering process. The rendering process is distributed between the server and the mobile device. The 3D scene is projected to 2.5D representation on the server side and this 2.5D representation is delivered through the network to the mobile device. The users can interactively change rendering modes of objects in the 2.5D scene (e.g. solid, semitransparent or wire-frame mode). The users can also zoom and pan the 2.5D scene. Moreover, they can annotate the objects in the 2.5D scene. The possibility to choose various rendering modes of objects in the 2.5D scene allows the user to investigate the information normally available only in a 3D representation of the scene (e.g. objects hidden around the corner).

The second textual based method focuses on presentation of graphical data with voice user interface. The possibility to investigate the structure of a 3D scene by means of textual queries where voice-based communication was used was successfully proven. A crucial role plays ontologies for storing data including natural language attributes and the usage of contextual information to improve the speech recognition rate and to resolve natural language ambiguities.

The hypothesis that the user's conversation language can be restricted accordingly to the conversation context determined by the application ontology and user context was proven. The dynamically generated input language of the voice user interface matched the user needs during the communication.

5.1. Future work

Our solution uses efficient information filtering based on the semantic description, which significantly increases the usability of the system in a mobile environment.

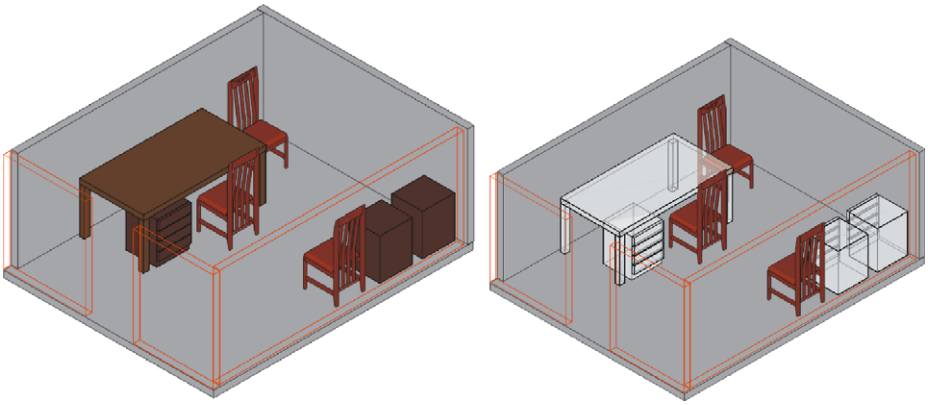

Fig. 19. Some of the furniture is described in semantic description as wooden chairs. The rendering mode of these chairs can be changed to highlight or suppress them.

One area of future work will investigate the possibility of utilization of the semantic description of the 3D scenes in the filtration process. The semantic description stored in the OWL [17] (Web Ontology Language) will be used for selective changes of the rendering modes of the objects in the 2D scene. The semantic description is usually application oriented—this means that the filtration process could be application driven. See an example in Fig. 19. This example shows the situation where only wooden chairs (application oriented information) are highlighted.

A second area of future work is finding a point with the optimal user experience with a compromise between the size of the input language and the recognition rate. One possible way is to introduce the behavior scenarios of the user in particular tasks (e.g. taking samples) and application domains (e.g. FM domain), which will help to predict the user actions and allow more precise restriction of the conversation language.

The goal is to develop a voice-based user interface, which will be usable in the real environment of a specific class of applications. For this purpose we plan to perform a second set of usability tests that would simulate the real work of a construction site inspector to find out whether input language being restricted in time really suits the needs of a real user.

Acknowledgements

The research was conducted within the framework of the MUMMY project (Mobile knowledge management—using multimedia-rich portals for context-aware information processing with pocket-sized computers in facility management and at construction site) and is funded by Information Society DG of European Commission (IST-2001-37365). See http://www.mummy-project.org/.

This research has been partially supported by MSMT under research program MSM 6840770014.

We appreciate very much the support provided by IBM research center Czech Republic for helping us to solve the problems of speech recognition and synthesis.

References

[1] Slavik P, Cmolik L, Mikovec Z. Object based manipulation with 3D scenes in mobile environment. Dagstuhl seminar proceedings 05181 mobile computing and ambient intelligence: the challenge of multimedia 〈http://drops.dagstuhl.de/opus/volltexte/2005/373/〉, 2005.

[2] Kopsa J, Mikovec Z, Slavik P. Ontology driven voice based interaction in mobile environment. In: Dagstuhl seminar proceedings 05181 mobile computing and ambient intelligence: the challenge of multimedia 〈http://drops.dagstuhl.de/opus/volltexte/2005/376/〉, 2005.

[3] Vainio T, Kotala O. Developing 3D information systems for mobile users: some usability issues. In: Proceedings of second NordiCHI'02 conference on human–computer interaction. New York, USA: ACM Press;2003. p. 231–4.

[4] Sanna A, Zunino C, Lamberti F. A distributed architecture for searching, retrieving and visualizing complex 3D models on personal digital assistants. International Journal of Human-Computer Studies 2004;60(5–6):701–16.

[5] Zunino C, Lamberti F, Sanna A. A 3D multiresolution rendering engine for PDA devices. In: Proceedings of seventh world multiconference on systemics, cybernetics and informatics (SCI03), vol. 5, 2003. p. 538–42 [ISBN9806560019].

[6] Hekmatzada D, Meseth J, Klein R. Non-photorealistic rendering of complex 3D models on mobile devices. In: Proceedings of eighth annual conference of the international association for mathematical geology, vol. 2, 2002. p. 93–8.

[7] Diepstraten J, Gorke M, Ertl T. Remote line rendering for mobile devices. In: CGI '04: Proceedings of the computer graphics international. Washington, USA: IEEE Computer Society; 2004. p. 454–61.

[8] Scalable vector graphics (SVG) 1.1 specification 〈http://www.w3.org/TR/2003/REC-SVG11-20030114/〉.

[9] Newell ME, Newell RG, Sancha TL. A solution to the hidden surface problem. In: ACM'72: Proceedings of the ACM annual conference. New York, USA: ACM Press; 1972. p. 443–50.

[10] Virtual reality modelling language (VRML) Specification 〈http://www.web3d.org〉.

[11] Ramakrishnan IV, Stent A, Yang G. HearSay: eabling audio browsing on hypertext content. ACM WWW2004, 2004. p. 80–9.

[12] Zue V. JUPITER: a telephone-based conversational interface for weather information. IEEE Transactions on Speech and Audio Processing 2000;8(1):100–12.

[13] Leong LH, Kobayashi S, Koshizuka N, Sakamura K. CASIS: a context-aware speech interface system. IUI '05: Proceedings of the 10th international conference on intelligent user interfaces. New York, USA: ACM Press; 2005. p. 231–8.

[14] Oviatt S. Advances in robust multimodal interface design. IEEE computer graphics and applications, vol. 23, no. 5. Los Alamitos, USA: IEEE Computer Society Press; 2003. p. 62–8.
[15] Pfleger N. Context based multimodal fusion. ICMI '04: Proceedings of the sixth international conference on multimodal interfaces. New York, USA: ACM Press; 2004. p. 265–72.
[16] Reithinger N, Alexandersson J, Becker T, Blocher A, Engel R, Löckelt M, et al. SmartKom—adaptive and flexible multimodal access to multiple applications. ACM ICMI'03. New York, USA: ACM Press; 2003. p. 101–8.
[17] Web ontology language (OWL) specification 〈http://www.w3.org/2004/OWL/〉.
[18] Kopsa J. Voice user interface for multimodal data. Master thesis, CTU in Prague, Prague, 2005.
[19] Cmolik L. Transformation of 3D scenes to 2D for mobile environment. Master thesis, CTU in Prague, Prague, 2005.