J Multimodal User Interfaces (2008) 2: 43–52
DOI 10.1007/s12193-008-0005-1

ORIGINAL PAPER

Prototyping and evaluating glove-based multimodal interfaces

Julián García · José P. Molina · Diego Martínez · Arturo S. García · Pascual González · Jean Vanderdonckt

Received: 17 August 2007 / Accepted: 28 February 2008 / Published online: 28 May 2008

Abstract Despite the early demonstrations a couple of decades ago, multimodal interfaces still present several challenges. One of these challenges is the creation of such kind of interfaces, which is currently being tackled by the OpenInterface project. The other challenge is related to the lack of experience of evaluating multimodal interfaces, which raises the question of how to check usability in such interfaces. In this context, this paper presents a new contribution to the OpenInterface project by adding data glove support, which is illustrated by prototyping a viewer for large size images. This application is also presented as a case study for usability evaluation, trying to translate our experience of multiple evaluations conducted at our VR lab.

Keywords Multimodal interfaces · Data gloves · Usability evaluation

J. García · J.P. Molina (✉) · D. Martínez · A.S. García · P. González
LoUISE Group, I3A Research Institute, Universidad de Castilla-La Mancha, Albacete, Spain
e-mail: jpmolina@dsi.uclm.es

D. Martínez
e-mail: diegomp1982@dsi.uclm.es

A.S. García
e-mail: arturo@dsi.uclm.es

P. González
e-mail: pgonzalez@dsi.uclm.es

J.P. Molina · J. Vanderdonckt
BCHI Group, Université catholique de Louvain, Louvain-la-Neuve, Belgium

J.P. Molina
e-mail: molina@isys.ucl.ac.be

J. Vanderdonckt
e-mail: vanderdonckt@isys.ucl.ac.be

1 Introduction

Multimodal systems process two or more user input channels —such as speech, gaze, hand gestures, head and body movements, etc.—in a coordinated manner. Just like multimedia systems allow the user to receive information in several formats and through several channels, in a multimodal system this user can also send information to the system through several channels.

One of the earliest examples of this kind of systems is the "Put That There" demonstration system [1], which combined a simple speech recognizer and a tracking device to manipulate a 2D interface. Many other systems have emerged since then, inspired by the goal of supporting more transparent, flexible and powerful means of human-computer interaction in order to achieve interfaces which are easier to learn and to use and, thus, accessible for a wider range of users.

The development of multimodal systems has been encouraged by the development of novel input and output technologies, including new devices and improvements in recognition-based techniques. Data gloves are just an example of these novel devices, providing a flexible and natural way of interaction. These devices can be used together with recognition methods to achieve highly effective gesture recognition systems to develop new ways of interaction. An example of this could be the interface shown in the movie "Minority report" [2], which we reproduced using data gloves and a tracking system [3]—see Fig. 1.

However, one of the main drawbacks for the development of this kind of systems has been the difficulties in reusing previous efforts for new systems. This is where OpenInterface emerges as a powerful solution, bringing a new concept in the design of multimodal systems and allowing the reuse of multimodal techniques, signal processing techniques and

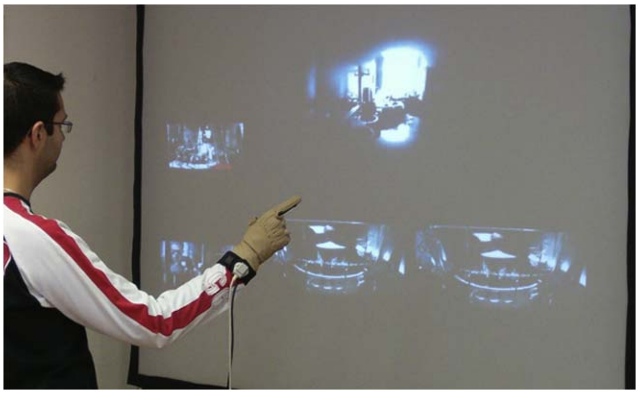


Fig. 1 A "Minority report"-like interface based on glove

recognition algorithms to facilitate the development of this kind of systems.

In this paper we first focus on the development of some reusable OpenInterface components that add support for two different data gloves. A case of study is then presented, which also serves us to focus on the evaluation of the multimodal interface, with special attention to its usability.

This paper is structured as follows. The next section gives an overview of the OpenInterface project. Following that explanation, the new OpenInterface modules and the data gloves they support are detailed. Then, the prototyping and evaluation of a sample application is addressed. Finally, conclusions are given together with proposals for future work.

2 OpenInterface

OpenInterface [4] is the name of an European organization aimed to expand, coordinate and integrate the efforts of several research labs in the field of multimodal interfaces. This organization is responsible for developing a software platform to allow a faster prototyping and development of multimodal interfaces based on the reutilization of components, following a user driven process and implementing the main concepts proposed in the ICARE model [5].

Thus, this platform provides not only a flexible standard for multimodal interaction, but also an extensive repository containing the previous experience of other developers by means of components encapsulating already tested multimodal techniques. Furthermore, the platform also makes it possible to easily add new components allowing the integration of new techniques.

This approach has the intention to bring deep benefits to the industry, but also to system developers, designers and final users experiencing the interfaces developed.

The key elements of this platform are explained in more details in the following sections.

2.1 Components

An OpenInterface system can be seen as a puzzle where components are the basic pieces. These components can contain any functionality, from an input mode, to processing algorithm or a user interface. The functionality contained in each of these components is always hidden to the rest of the components in the system. Components just publish interfaces that describe how other components can communicate with them, sending and receiving data through a mechanism called pins. Three types of pins are available:

- Sinks: pins through which components can receive data.
- Sources: pins through which components can send data.
- Callbacks: pins through which components can send events to other components.

The pins defined for each component grant access to the configuration and internal functionality of the component.

Meanwhile, each component can be internally implemented using a different language, having all their interfaces defined using a common XML based language called CIDL (Component Interface Description Language). This language allows components to be reused in any other OpenInterface program.

2.2 Components pipeline

OpenInterface applications are built by interconnecting different components in a way that describes the desired dataflow. The interconnection of a set of components (data sources, algorithms, input modalities, etc.) is what OpenInterface defines as the "components pipeline".

Pipelines are described using PDCL (Pipeline Description and Configuration Language), an XML language that allows the definition of the components involved in a given program, and the interconnections necessary to describe the appropriate pipeline.

2.3 OpenInterface runtime kernel

The OpenInterface runtime kernel is a program written in C++ that allows the communication among heterogeneous components, being local or distributed. Thus, it is responsible for parsing the PCDL files describing OpenInterface programs, initializing the components in each program and establishing the appropriate communications among them according to the description given in the PDCL file.

2.4 OpenInterface graphical editor

This tool is currently under development and will allow the easy design and modification of the Open Interface programs. This tool will allow building PCDL files by graphically interconnecting the appropriate components.

3 Hand gesture recognition

Since the creation of the first data glove model by T. Zimmerman and J. Lanier, data gloves have been considered, together with head-mounted displays, one of the most representative devices of Virtual Reality interfaces. A data glove can be used to create a direct correspondence between user's real hand and the virtual hand drawn in the scene. However, the common lack of tactile or force feedback prevents the user from feeling or picking the objects as they would do in real life. Data gloves can also be used as control devices with which the user can enter commands into the system using a language based on gestures. In this case, the main problem is that the user has to learn and master that language. Besides, there are less intrusive devices such as wands that can be useful for that same purpose.

Despite all these drawbacks, it is our opinion that data gloves are still a valid option to consider when creating new user interfaces, particularly multimodal ones. The first reason is that hands play a significant role in human communication, and so that they represent a valuable input channel in multimodal interfaces. The second reason is that data gloves are an obvious choice when hand information is needed, such as finger flexure or hand tilt.

However, no OpenInterface module supported this kind of device until now. Therefore, to tackle this lack and allow OpenInterface to be used for prototyping a wider range of multimodal user interfaces, new modules have been programmed supporting two different data gloves. The following paragraphs detail the design and implementation issues.

3.1 Anatomy of human hands

Our hands allow us to perform a wide range of gestures thanks to the high number of degrees of freedom (DOFs) they have. For each hand, the total amount is as high as 29 degrees of freedom.

From the wrist to the fingertips, there are 23, which result from summing the degrees of freedom of each finger joint. Thus, each finger comprises three joints that are, from the tip to the base, the distal interfalange (DIP, 1DOF), the proximal interfalange (PIP, 1DOF), and the metacarpofalangea (MCP, 2DOF). The thumb has also three joints, but they are known as thumb interfalange (Thumb IP, 1DOF), thumb metacarpofalange (Thumb MP, 1DOF) and trapeziometacarpal (3DOF). Besides, there are two more metacarpocarpal joints located between the carpal and the metacarpal of the 4th and 5th finger (ring and baby fingers, 1DOF each).

Those 23 degrees of freedom are then completed with 3 more DOFs due to wrist rotation around each axis (roll, pitch and yaw), and 3 more DOFs due to the actual position of the hand in the three-dimensional space, resulting in 29 degrees of freedom in total, as said before.

Developers of data gloves take into account those characteristics of human hands in their designs, although not every data glove is conceived to support all those degrees of freedom. Some models, such as the Cyberglove, which was used in our previous "Minority report"-like interface, do support a high number of them, having as much as 22 flexure and abduction sensors.

Others are simpler, such as the Dataglove 5 and the P5 models that are used in this present work, having one flexure sensor for each finger only, as it will be described next. Both models were assessed to suit our purposes in this work, but this decision did not imply a simpler hand model. Quite the opposite, the software developed for managing and recognizing postures takes into account the total amount of degrees of freedom of human hand, thus making it possible to use it with other kinds of data gloves, as it will also be explained later.

3.2 Data gloves

Selecting the right data glove is not an easy task. Several models exist, each one offering different kinds and number of sensors that measure finger contact, finger flexure, finger abduction, hand pitch, hand roll, etc. The requirements of the application should determine which model is most appropriate, although there are other factors that must be taken into account, such as software support and, of course, price. As an example, [6] describes three different projects that required a different data glove in each case.

Fortunately, several data gloves are available at our lab: 5DT Glove 5, Essential Reality P5, Fakespace Pinch Glove system, Immersion Cyberglove and noDNA X-IST Datagloves. From them, only the first two are currently supported by the modules that were programmed to contribute to the OpenInterface project. Both were selected because they offer flexure sensors together with some orientation/position sensor, Linux support was available when the project started—by that time, the OpenInterface project run only on Linux, and they are not expensive. The particular specifications are given below.

3.2.1 Fifth dimension DataGlove 5

Fifth Dimension Technologies (5DT) offers several models of data gloves with different features. One of them is the DataGlove 5—see Fig. 2. It is made of lycra and has one loop of optic fibre in each finger. The end of each loop is connected to a LED, the emitted light is transmitted through the loop and then it is received by a photoreceptor. As the finger bends, the amount of light received at the end of the loop changes. The glove system processes these changes and sends the corresponding values to the computer through a serial port. Furthermore, this glove can also measure the pitch and roll of user's hand.

Fig. 2 5DT DataGlove 5

Fig. 3 Essential Reality P5 glove

There is another version with 16 sensors, but the documentation available for this model is not so clear. In any case, both DataGlove models have been recently discontinued, being substituted by the brand new Ultra series [7]. Since the SDK for Linux that can be downloaded the company's site and that has been used for the programming of our OpenInterface module is the same for all these models, it can be expected that the module will also support the new Ultra series.

3.2.2 Essential Reality P5 glove

The P5—Fig. 3—is a low cost data glove designed for leisure and first commercialized by Essential Reality. It has a bend sensor on each finger responsible of measuring its bending. Data is sent to the computer through an USB port. It also includes an optic system consisting of a set of LEDs placed on the back part of the glove. The light from these LEDs is received by the base unit where the data glove is plugged to. This system is used for measuring the position and orientation of the user's hand in its 6 DOFs.

Essential Reality was later acquired by Alliance Distributors, but the glove can still be purchased at Video Game Alliance [8] with an approximate retail cost of $50.

3.3 Methods for gesture recognition

Apart from the hardware, a software mechanism was necessary to recognize the gestures by using the data provided by the glove. To face this problem, different choices were analyzed. Some methods are more appropriate for recognizing postures, such as the Template Matching technique, while other methods, such as the Hidden Markov Models (HMMs), are better for recognizing gestures, that is, postures combined with motion [5].

However, it takes time to train a system to recognize particular gestures, and bearing in mind that the aim of OpenInterface is to allow quick prototyping user interfaces, it was decided that a simple method such as Template Matching would be enough for that purpose. Nevertheless, this decision added a new requirement: the development of an application that allows the user to record, save and retrieve hand posture templates.

4 Adding data glove support to OI

Once the data gloves to use were chosen and the hand model defined was completed, research moved on to the implementation phase to give actual support to the selected devices. The implementation phase was guided by two main goals:

- Producing components that could benefit from the hand model proposed.
- Producing components that could be reused by other designers even if they decided not to use the hand model proposed.

With these goals in mind, it was decided that data glove components had to stream data in the same format as the data glove vendors provide in their drivers, providing also additional facets that translate this native data to the XML representation specified by the model proposed. This way, designers that prefer not to use the XML format can still use the native one and adapt it to their needs, while other designers can benefit from using the new representation.

In order to test the benefits from the hand model it was also decided to implement a component containing a template matching algorithm, and then use it in some prototypes of the case study.

Other components were also implemented to build these prototypes, but this section just focuses on the three previously mentioned as they are the more relevant for the scope of this paper.

4.1 Glove5DTComponent

This component allows OpenInterface developers to use 5DT data gloves as an input device for their programs. The data about user's gestures can be retrieved from the data glove using two different ways of operation:

- The first way of operation is accessing data on demand. Using this way of operation, the program will send an event to the component sink each time a gesture must be read. When this event arrives, data is retrieved from the data glove and then sent back to the program. This way of operation might be slower than the second one, but, as it does not employ any threads, it is less resource consuming and can be useful for applications that only need to read these data at certain moments.
- The second way of operation is configuring the component so that it periodically sends the last data read from the data glove. This way of operation is well suited for highly interactive applications which use data gloves as its main input device—such as Virtual Reality applications or multimodal interfaces. On the other hand, this way of operation requires a thread to be run in order to periodically update data about the gesture performed by the user, which makes it more resource consuming.

Pins using the first way of operation have been tagged with the words pulled_event while the ones providing continuous data are tagged with the words pushed_events.

Furthermore, data can be retrieved using two different representations. 5DT data gloves measure data about each finger's flexion using a number between 0 and 4096—being 0 completely extended and 4096 completely bended. In order to maintain vendor's format, Glove5DTComponent can provide data in this native representation, but also in a normalized representation between 0 and 1. Pins using vendor's scale are tagged using Raw, while the ones using the scaled representation are tagged using the Scaled tag.

Finally, as 5DT Company offers several data glove models—there are currently models with 5, 7, 14 and 16 sensors—the component allows the control of almost all these models. The only exception is for the models with 16 sensors. For these models, the documentation available by the time this work was carried out was quite unclear, so support for this model could not be provided. Thus, for the rest of the models—5, 7 and 14 sensors models, the number of sensors is specified in the name of the pin—i.e. pulled_events5Raw would give us the last data read from a model of 5 sensors in its native scale.

4.2 Glove P5Component

This component allows OpenInterface developers to use P5 data gloves as an input device for their programs.

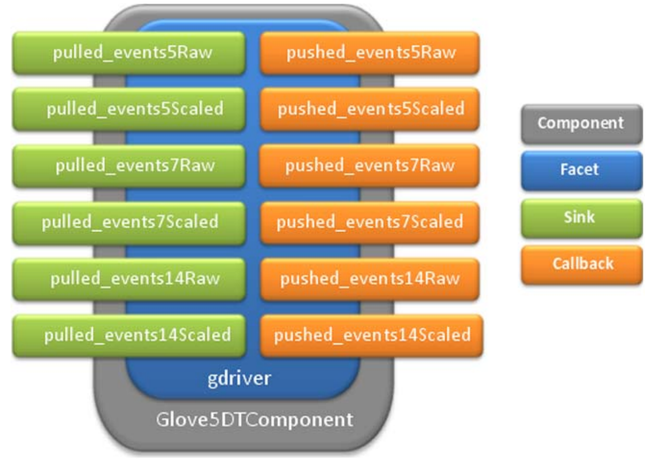


Fig. 4 Component that supports the 5DT glove series

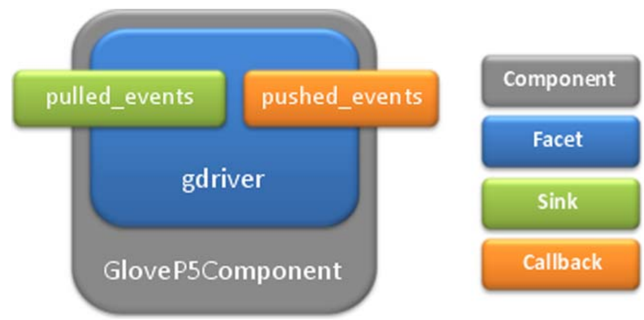


Fig. 5 Component that supports the 5DT glove series

Just like when using Glove5DTComponent, data can be retrieved under demand (using the pulled_event pins) or continuously (using the pushed_event pins). Given that in this case there is only one data glove model and not several as in the previous case, this component defines two pins.

4.3 GloveTemplateMatchingComponent

The GloveTemplateMatchingComponent encapsulates a template matching algorithm that, by using the general hand model proposed, can be used independently of the input data glove used. This algorithm is encapsulated by the facet rec_engine.

This facet uses a local database which stores gestures—defined in XML format—together with a string associated to each gesture. The recognition process it uses is based on a template matching approach, but with some small differences. Given that many data gloves do not have the high amount of degrees of freedom represented in our hand model, the algorithm does not always use them all. Instead of that, it parses the representation of each of the templates stored and only uses the degrees of freedom present at the template for the recognition process. This feature allows developers to have a database of gestures defined solely con-

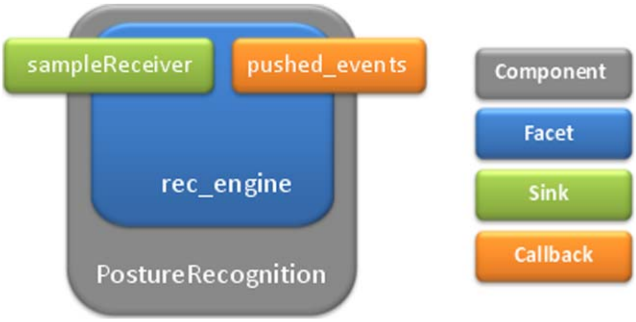


Fig. 6 Template matching component

taining those representative degrees of freedom for each gesture, and then taking only those into account when parsing the input data from the data glove.

The only drawback is that the data glove used must supply all the degrees of freedom provided by the template it is being compared to. This constraint follows the previous idea of having templates defined only by their most representative data. Given that every data in the template is considered as an important feature of the gesture it represents, all these data must be available to assure a right comparison.

This component also contains four additional facets—not shown in the figure—to translate the representation of each of the data gloves supported—5DT models with 5, 7 and 14 sensors and the P5 model—to the XML representation used by the algorithm. These additional facets are named Glove⟨Model⟩PostureRecLanguaje, to show that they depend on the data glove used, where ⟨Model⟩ can be 5DT5, 5DT7, 5DT14 or P5. These facets just define one sink, to receive the data in the format of the data glove supported, and one callback producing the XML representation of the input data received through their sinks.

4.4 Posture management tool

In order to facilitate the definition and management of the gesture templates used, a companion application was developed. This application allows the definition of gestures, its translation to XML format and its storage and previous load.

Gestures can be defined using one of the data gloves supported, but also—as shown in Fig. 7—manually using a set of sliders.

This application also includes a 3D representation of the hand based on the anatomy explained before. This hand represents the gesture which is being defined at each moment and it can be rotated to examine the gesture defined without modifying the information available about the pitch, yaw and roll of the gesture defined.

This tool also allows defining the threshold of each of the individual measures in the gesture for the template matching algorithm and previsualizing the XML representation generated.

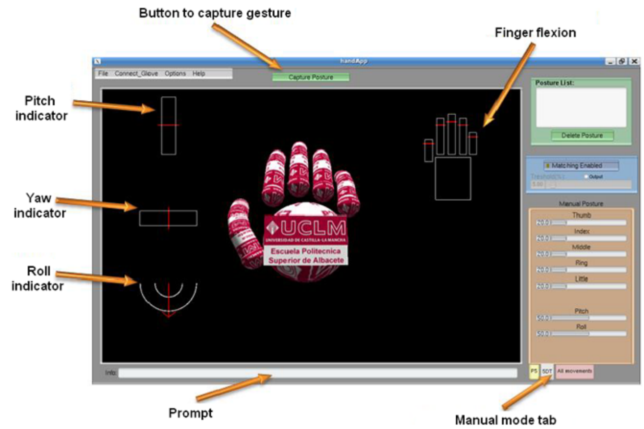


Fig. 7 Interface of the porture manager

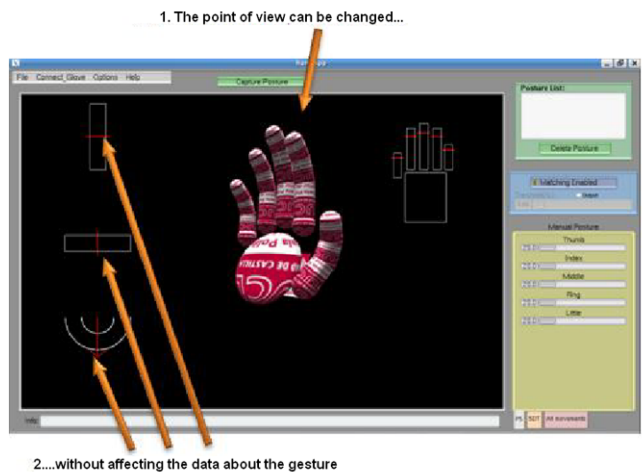


Fig. 8 Interface of the posture manager

This tool was developed in C++, using OpenGL for the rendering and FLTK [9] for the UI definition.

5 Case study

In order to put the new OpenInterface modules into practice, the prototyping of a simple application has also been addressed, exhibiting a multimodal interface. This case study is aimed to serve as a testbed for usability evaluations too.

The prototyped application is an image viewer that is intended to tackle the problem of exploring large size images, such as satellite ones—see Fig. 7. A similar application has also been developed by other people as part of the OpenInterface project, but this one has been built from scratch using the new data glove modules. Thus, using hand postures and other input channels, such as voice, the user can explore the images panning in four directions, and zooming in and out.

The application has been customized for using the two data gloves shown in this paper, with or without the template matching component, and together with other modali-

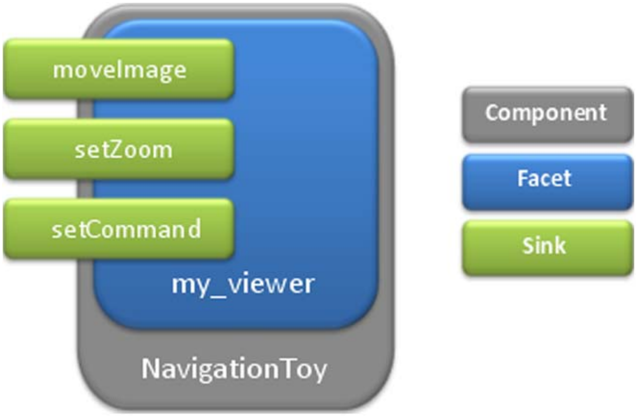


Fig. 9 Structure of the viewer component

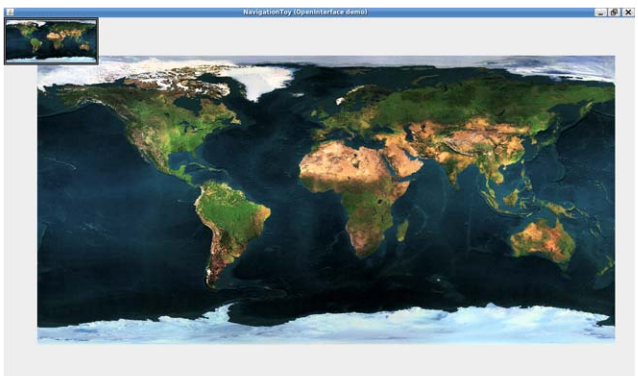


Fig. 10 Screenshot of the prototyped image viewer

ties, such as speech recognition. In this section we will discuss three different configurations tested and some usability issues which rose during their implementation and evaluation.

5.1 Case no. 1: 5DT and template matching

This prototype used template matching to select the command (scale or translation), and the 5DT data glove to actually perform these commands. As position could not be measured, information about user's hand pan and tilt was used to control the viewer. Users could move the image by bending their wrists up, down, left or right, or modify the zoom by bending it up and down.

The implementation of this application follows the structure shown in Fig. 13, using Glove5DTComponent to generate the input and posture5DTRecLanguaje to adapt data gathered from the data glove—through its pushed_events5scaled pin—to the XML representation required by rec_engine. For this configuration two additional, facets Glove5DT-TranslationLanguaje and Glove5DTZoomLanguaje were defined. The components were designed to use the information about the data glove position to control Translation and Zoom.

Fig. 11 Using the 5DT DataGlove 5 to explore an image

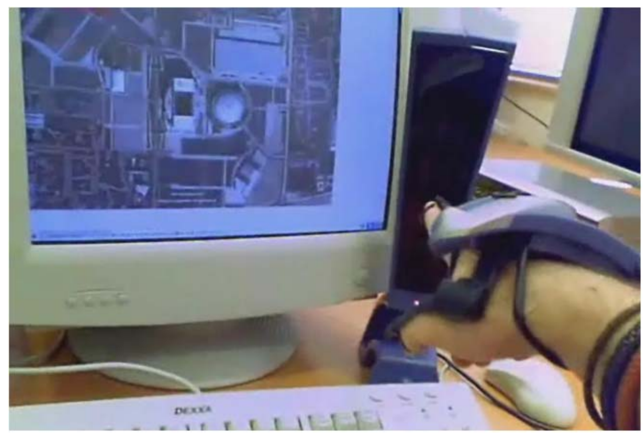


Fig. 12 Using the P5 to explore an image

5.2 Case no. 2: P5 and template matching

The second configuration also used the template matching algorithm for command selection, but this time together with the P5 data glove. Given that this data glove can measure the hand position, we used it to control the translation and scaling of the image displayed. Particularly, users could translate the image by moving their hands up, down, left and right. The zoom was increased if the hand approached the screen or decreased if it was moved backwards.

In this case, GloveP5Component was used, connecting its pushed_events callback pin to the rest of the elements in the application. To translate this data to the XML representation, required by rec_engine, the postureP5RecLanguaje was used. Two additional facets, GloveP5Translation- Languaje and GloveP5Zoom-Languaje, were implemented in the viewer, this time using pan and tilt to control zoom and translation. The posture_SelectCMDLanguaje facet could be reused from the previous configuration.

Fig. 13 Structure used for the first two configurations

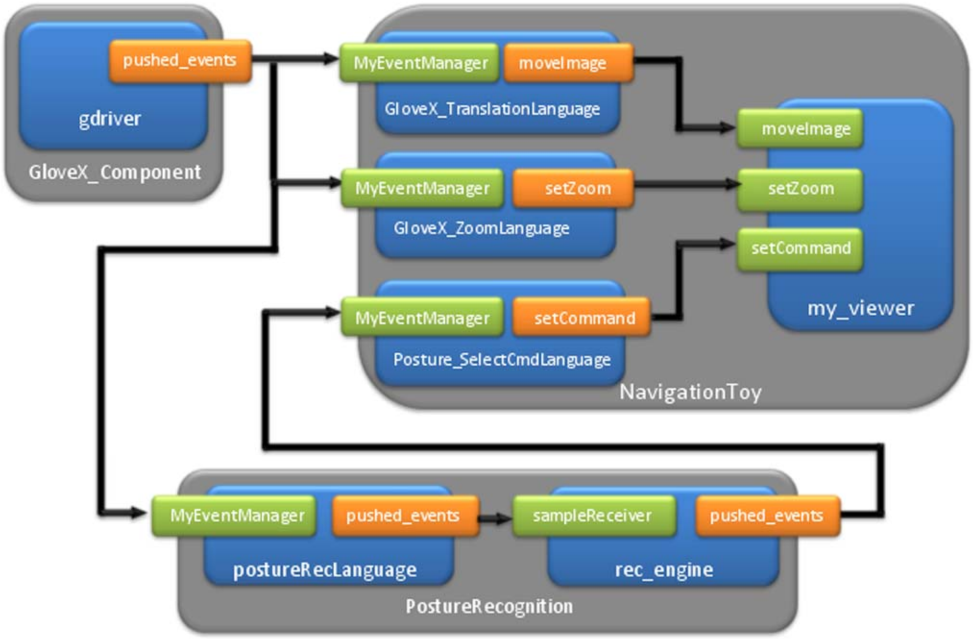
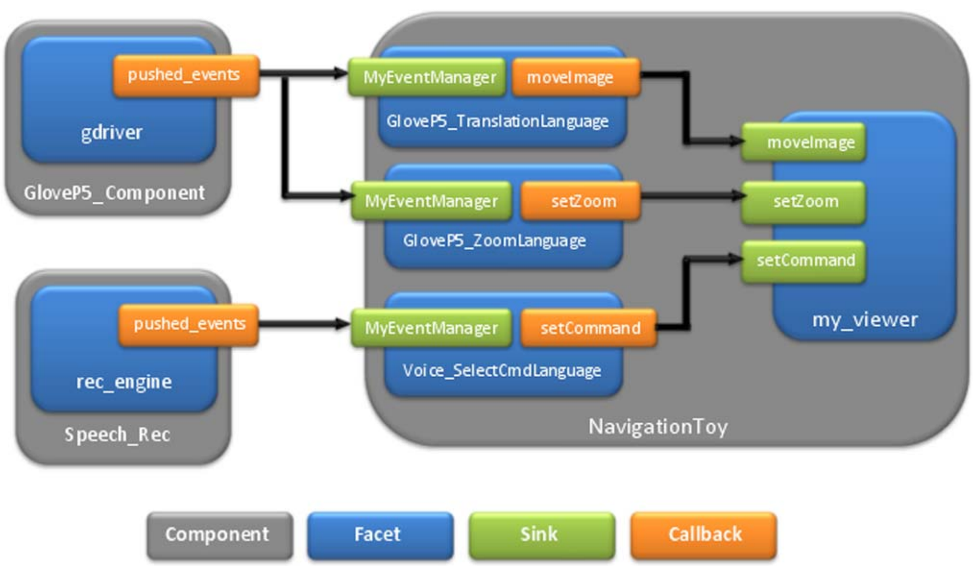


Fig. 14 Structure used for the third configuration



5.3 Case no. 3: P5 and speech recognition

The third case used speech recognition instead of template matching for the command selection, and the actions were actually performed using a P5 data glove in the same way than in the first configuration tested.

For the implementation of this application a new component was defined to provide speech recognition. In this component sphinx [10] was used to implement a speech recognition algorithm which sends through its only callback pin a string with the last word identified. Also, the posture_selectCmdLanguaje facet was included to translate the words recognized to the appropriate viewer command. The rest of the elements in the application could be reused from the ones used for case no. 2.

6 Evaluating multimodal interfaces

As stated before, the previous case study was chosen not only to test the implemented components, but also is aimed to serve as a testbed for usability evaluations.

Regarding the development methodology used, during the implementation phase, together with the typical human-computer interface mantras—such as keeping commands simple and straightforward—the guidelines proposed in

[11], were followed. For instance, we tried to avoid mismatching between the user action and the system outcome, such as coupling left-right hand movement with up-down image displacement. All in all, our opinion as developers may differ from other experts and, more important, from user's opinions.

For that reason, a usability evaluation was performed. As research in our lab is mainly focused on Virtual Reality, we chose a methodology proposed by Gabbard [12] and which is initially aimed for Virtual Environments as we consider it is well suited for evaluating this kind of interfaces.

The methodology is divided in four steps, but. It starts performing a user task analysis of the requirements of the system where the most scenarios and tasks, but also the dependencies between tasks, are identified. Next, the methodology suggests performing a heuristic evaluation based on a set of usability guidelines chosen for this purpose. The third step consists of a user-centred evaluation where the main scenarios identified in the first step are tested and where the opinions of the users are gathered and used to develop new prototypes. This step can be repeated several times before moving to the last phase.

The last phase consists of a summative comparative evaluation where users try all the implemented program versions choosing the one they find like the most appropriate.

The next section describes how this methodology was used in the evaluation of our prototypes.

6.1 Expert guidelines-based evaluation

After the implementation phase and according to the usability evaluation methodology used, our first evaluation of the image viewer was conducted by experts different to those that developed the application, taking into account ergonomic criteria and the design guidelines chosen in the previous phase [11]. In general, it was found that most of those guidelines were respected when designing the application, only a few of them were not satisfied. For instance, one guideline states that the interface should maximize the advantages of each modality to reduce user's memory load, as when coupling visual presentation with manual input for spatial information. In this case, the guideline is satisfied by our design as the image shown in the screen is translated in four directions following user's hand motion. However, another guideline states that the interface should adapt to the needs and abilities of different users, as well as different contexts of use. In this case, the guideline is not satisfied by our prototype, as the only available modalities are hand gestures, by means of the glove, and speech recognition, and no alternative modalities were taken into account, such as using the keyboard or the mouse to replace the glove if the user gets tired or feels uncomfortable with it.

6.2 User-centered evaluation

As the next step of the process after this heuristic evaluation, a user testing should be performed. Even though this steps could not be followed, this evaluation is planned to be conducted. In this case, we will rely on our previous experience evaluating Virtual Reality interfaces. In this particular field, we have carried out several evaluations on single-user Virtual Reality applications, such as [13], and also on Collaborative Virtual Environments (CVEs), such as [14, 15]. From both cases, we think that the experience gained through evaluating CVEs can be of more interest when translating it to the field of multimodal user interfaces, as the users can communicate with the system at the same time through different channels. In every experiment, we focus on both task performance—from direct observation—and user opinion—gathered through questionnaires, as we comment on the following paragraphs.

First, task performance can be quite significant when testing usability. For instance, in collaborative environments where several users work on the same task, the interface can be considered to be free of usability problems if the time completion equals the time taken to complete the task by only one user divided by the number of users that collaborate in the task. Then, this idea could be translated to multimodal interfaces where several channels are used to complete a task. However, this is not usually the case, as time completion is highly dependent on the task carried out and the objects of that task, as we found in our experiments.

Second, questionnaires are a good vehicle to gather user's opinions. Following our experiments, we have used standard questionnaires, such as QUIS [16], and also ad hoc ones. For instance, the QUIS questionnaire can be used to gather insight into user satisfaction in general, but we find that important questions related to non-conventional interaction is missed, such as fatigue derived from worn devices—i.e. the data glove—due to weight or motion needs. For that reason, we usually produce ad hoc questionnaires when planning those experiments. This experience can also be translated to the proposed evaluation of the prototyped application, and other multimodal interfaces in general.

In any case, as careful evaluation is needed for these interfaces, we find that both approaches, that is, evaluation with and without users, are equally important to test their usability, as recommended in [17].

7 Conclusions

In this paper, new OpenInterface components have been presented. These components add support for two data gloves, the well-knows Data Glove 5 from Fifth Dimension Technologies, and the cheap P5 from Essential Reality. Having support for this data gloves in the OpenInterface project allows us to prototype new multimodal interfaces that make use of hand postures based on finger flexure, hand orientation and hand position. The application shown as a case study is meant to be a proof of that.

However, the fact that we can create new multimodal interfaces based on hand interaction does not imply that these interfaces are free of usability problems, no matter how natural this kind of interaction can be. Bearing in mind that few usability guidelines exist for multimodal interaction, a careful evaluation is highly needed. In this sense, this position paper presents our experience in evaluating VR interfaces, in particular collaborative virtual environments, where several input channels from different users come into play, in a similar way that can be found in multimodal interfaces. We hope that experience to be interesting for other participants at the workshop and thus we are eager to share it with them.

Acknowledgements This work has been partially supported by the Ministerio of Educación y Ciencia of Spain (CICYT TIN2004-08000-C03-01), and by the Junta de Comunidades de Castilla-La Mancha (PAI06-0093).

References

1. Bolt RA (1980) Put-that-there: voice and gesture at the graphics interface. In: Proc of the 7th annual conference on computer graphics and interactive techniques, SIGGRAPH 1980, pp 262–270
2. Spielberg S (2002) Minority report. 20th Century Fox and Dreamworks Pictures
3. Molina JP, García AS, Martínez D, Manjavacas FJ, Blasco V, González P (2006) An interaction model for the TRES-D framework. In: Proc of 13th IEEE Mediterranean electrotechnical conference MELECON 2006
4. Openinteface project. http://www.oi-project.org/
5. Bouchet J, Nigay L, Ganille T (2004) ICARE software components for rapidly developing multimodal interfaces. In: Proc of ACM int conf on multimodal user interfaces ICMI 2004, pp 251–258
6. LaViola JJ (1999) A survey of hand posture and gesture recognition techniques and technology. Tech rep, Brown University
7. Fifth dimension technologies, data gloves. http://www.5dt.com/hardware.htm#lglove
8. Video game alliance, P5 glove. http://www.videogamealliance.com/VGA/video_game/P5.php
9. Fltk home page. http://www.fltk.org
10. Sphinx project. http://cmusphinx.sourceforge.net/html/cmusphinx.php
11. Reeves LM (2004) Guidelines for multimodal user interface design. Commun ACM 47:57–59
12. Gabbard JL, Hix D, Swan JE (1999) User-centered design and evaluation of virtual environments. IEEE Comput Graph Appl 19(6):51–59
13. García AS, Molina JP, González P (2005), Exemplar VE design guidance tool for selection and manipulation interaction techniques. In: Proc of 11th international conference on human-computer interaction, HCI international 2005
14. García AS, Martínez D, Molina JP, González P (2007) Collaborative virtual environments: you can't do it alone, can you? In: Proc of 12th international conference on human-computer interaction, HCII 2007
15. García AS, Steed A, Martínez D, Molina JP, González P (2007) An approach to the evaluation of ownership management techniques in collaborative virtual environments. In: Proc of 13th eurographics symposium on virtual environments, EGVE 2007
16. Shneiderman B (1998) Designing the user interface: strategies for effective human-computer interaction. Pearson Education
17. Whitefield J, Wilson F, Dowell J (1991) A framework for human factors evaluation. Behav Inf Technol 10:65–79