Virtual Reality (2007) 11:23–43
DOI 10.1007/s10055-006-0055-1

ORIGINAL ARTICLE

An augmented reality interface for visualizing and interacting with virtual content

Fotis Liarokapis

Received: 15 December 2004 / Accepted: 19 October 2006 / Published online: 9 November 2006

Abstract In this paper, a novel AR interface is proposed that provides generic solutions to the tasks involved in augmenting simultaneously different types of virtual information and processing of tracking data for natural interaction. Participants within the system can experience a real-time mixture of 3D objects, static video, images, textual information and 3D sound with the real environment. The user-friendly AR interface can achieve maximum interaction using simple but effective forms of collaboration based on the combinations of human–computer interaction techniques. To prove the feasibility of the interface, the use of indoor AR techniques are employed to construct innovative applications and demonstrate examples from heritage to learning systems. Finally, an initial evaluation of the AR interface including some initial results is presented.

Keywords Augmented reality · Human–computer interaction · Tangible interfaces · Virtual heritage · Learning systems

1 Introduction

Augmented reality (AR) is an increasingly important and promising area of mixed reality (MR) and user interface design. In technical terms, it is not a single technology but a collection of different technologies that operate in conjunction, with the aim of enhancing the user's perception of the real world through virtual information (Azuma 1997). This sort of information is usually referred to as virtual, digital or synthetic information. The real world must be matched with the virtual in position and context in order to provide an understandable and meaningful view (Mahoney 1999). Participants can work individually or collectively, experiment with virtual information and interact with a mixed environment in a natural way (Klinker et al. 1997). In an ideal AR visualisation scenario, the virtual information must be mixed with the real world in real-time in such a way that the user can either understand or not, the difference (Vallino 1998). In case where virtual information looks alike the real environment, the AR visualisation is considered as the ultimate immersive system where participants cannot become more immersed in the real environment (RE).

The term AR usually refers to one of the following definitions (Milgram and Colquhoun 1999). A class of display systems that consist of a type of head mounted display (HMD) (Azuma 1997); those systems that utilize an equivalent of an HMD belong to the second class, encompassing both large screen and monitor-based displays (Milgram and Kishino 1994). A third classification refers to the cases that include any type of mixture of real and virtual environments. Overall, the majority of AR systems rely on electronic sensors or video input in order to gain knowledge of the environment (Haniff et al. 2000). All these variables make these systems more complex than systems that do not rely on sensors. Vision based systems on the other hand, often use markers as feature points so they can estimate the camera pose (position and orientation).

F. Liarokapis
Department of Informatics, University of Sussex,
Falmer, Brighton BN1 9QT, UK

F. Liarokapis (✉)
Department of Information Science, City University,
London EC1V 0HB, UK
e-mail: fotisl@soi.city.ac.uk; f.Liarokapis@sussex.ac.uk

In the upcoming years, AR systems will be able to include a complete set of augmentation applied and exploiting all people's senses (Azuma et al. 2001). However, although there are many examples of AR systems where users can interact with and manipulate virtual content and even create virtual content within some AR environments, one of their major constraints is the lack of ability to allow participants control multiple forms of virtual information in a number of different ways. To a great extend, this deficiency derives mainly from the lack of robustness of currently existing AR interface systems. At this stage, this can be dealt by using a user-friendly interface to allow users position audio–visual information anywhere inside the physical world. Since the pose can be easily estimated through an existing vision based tracking system such as the well-known ARToolKit (Kato et al. 2000a, b), the focus of this research is to provide effective solutions for interactive indoor AR environments.

Vision-based AR interface environments highly depend on four key elements. The first two relate to marker implementation and calibration techniques. The latter are interrelated with the construction of software user interfaces that will allow the effective visualisation and manipulation of the virtual information. The integration of such interfaces into AR systems can reduce the complexity of the human–computer interaction using implicit contextual input information (Rekimoto and Nagao 1995). Human computer interaction techniques can offer greater autonomy when compared with traditional windows style interfaces. Although some work has been performed into, the integration of such interfaces into AR systems (Feiner et al. 1993; Haller et al. 2002; MacIntyre et al. 2005) the design and implementation of an effective AR system that can deliver realistically audio–visual information in a user-friendly manner is a difficult task and an area of continuous research. However, it is very difficult even for technologists to create AR experiences to eliminate these barriers (MacIntyre et al. 2005) that prevent users to create new AR applications. To address the above issues, a prototype AR interface for assisting users that have some virtual reality experience to create fast and effective AR applications is proposed. The main novel contributions of this paper include the following:

- Simultaneous and realistic 3D audio–visual augmentation in real-time performance;
- Implementation and combination of five different ways for interacting with the virtual content;
- Design and implementation of a high-level user centred interface that provides accurate and reliable control of the AR scene;
- Two innovative applications: one for cultural heritage and one for higher education and
- Initial evaluation regarding the overall effectiveness of the system;

In the remainder of this paper, we describe our system starting with Sect. 2 that gives a historical overview of the AR interfaces. In Sect. 3, the architecture of the prototype AR interface is presented in detail. Section 4, presents various calibration approaches followed to calibrate our camera sub-system accurately. Section 5 presents realistic augmentation techniques that can be applied in real-time performance. Section 6 proposes five different ways of interacting with the AR scene while in Sect. 7 two application scenarios are presented. In Sect. 8, the results from an initial evaluation are presented whereas Sect. 9 summarises the key findings and the current status of research and suggests future work.

2 Historical overview of AR interfaces

One of the earliest applications involved an experimental AR system that supports a full X11 server on a see-through HMD. The display overlays a selected portion of the X bitmap, on the user's view of the world, creating an X-based AR. Three different types of windows were developed: surround-fixed windows, display-fixed windows and world-fixed windows. The performance of the system was in the range of 6–20 frames-per-second (FPS). A fast display server was developed supporting multiple overlaid bitmaps having the ability to index into a display a selected portion of a larger bitmap (Feiner et al. 1993). EMMIE (Butz et al. 1999) is another experimental hybrid user interface designed for a collaborative augmented environment that combines various different technologies and techniques such as virtual components (i.e. 3D widgets) and physical objects (tracked displays, input devices). The objects in the system can be moved among various types of displays, ranging from see-through HMDs to additional 2D and 3D displays. These vary from palm-sized to wall-sized depending on the nature of the task.

The MagicBook (Billinghurst et al. 2001) and the Tiles system (Poupyrev et al. 2002) are two of the most well known AR interfaces based on the ARToolKit. The Tiles system proposes a way of creating an AR workspace blending together virtual and physical objects. The interface combines the advantages (power and flexibility) of computing environments with the comfort and awareness of the traditional workplace

(Poupyrev et al. 2002). On the other hand, the MagicBook uses a real book to transfer users from reality to virtuality. Virtual objects are superimposed on the pages of the book and users can interact with the augmented scene (Billinghurst et al. 2001). Another example of an AR tangible interface is a tabletop system designed for virtual interior design (Kato et al. 2000a). One or multiple users can interact with the augmented scene, which consists of virtual furniture and manipulates the virtual objects.

MARE (Grasset and Gascuel 2002) is a collaborative system that mixes together AR techniques with human-computer interaction techniques, in order to provide a combination of natural metaphors of communication (voice, gesture, expression) with virtual information (simulation, animation, persistent data). The architecture of the system is based on OpenGL Performer and XML configuration files and it can be easily adapted to many application domains. Another interesting workspace is a wearable AR generic platform that supports true stereoscopic 3D graphics (Reitmayr and Schmalstieg 2001). The system supports six degrees-of-freedom (DOF) manipulations of virtual objects in the near field using a pen and a pad interface. Slay et al. (2001) developed an AR system that extends interactions from a traditional desktop interaction paradigm to a tangible AR paradigm. A range of issues related to the rapid assembly and deployment of adaptive visualisation systems was investigated. Three different techniques, for the task of switching the attributes of the virtual information in AR views, were presented.

Furthermore, the AMIRE project (Haller et al. 2002) aims at developing fast rapid prototyping through vision-based AR for users without detailed knowledge of the underlying base technologies of computer graphics and AR skills. AMIRE uses a component-oriented technology consisting of a reusable GEM collection, a visual authoring tool and object tracking system based on the ARToolKit library. Another system that allows users to create AR experiences is the designer's augmented reality toolkit (DART) (MacIntyre et al. 2005). The system is based on the Macromedia Director multimedia-programming environment to allow a user to visually create complex AR applications as well as providing support for the trackers, sensors and camera.

Although most of the above systems describe generic frameworks that allow for AR and/or MR applications, they have not focused on designing a high-level user-focused interface that can deliver audio–visual information. The DART system is the most similar to this approach but it is based on a commercial multimedia package and thus it is addressed to designers and not general purpose developers. However, this sometimes limits the capabilities of the generated applications because they will be limited to the specific package (i.e. Director). On the contrary, this work is targeting developers who want to develop AR applications and use higher level tools than currently exist (i.e. ARToolKit).

3 Architecture of the system

The scope of the AR interface is to provide all the necessary tools for developers to generate user-specific AR applications (see Sect. 7). They will select which sort of functionality is useful, and either use it as it is or extend it to fit the needs of the application. Based on previous prototypes (Liarokapis et al. 2004a, b) a tangible AR interface focused on superimposing five different types of virtual information and allowing users to interact using a combination of five different interaction techniques was designed and implemented. The system allows for the natural arrangement of virtual information anywhere inside the interior of a building or any other type of indoor environment. A diagrammatic overview of the operation of the system is presented in Fig. 1.

In the simplest configuration, a laptop computer with a USB web-camera and a set of trained marker cards are employed. The most complex configuration performed for the purpose of this research included two cy-visor HMDs, four LCD monitors, an 18 in. iiyama touch screen and a 42 in. plasma screen (Sony PFM-42V1N). Depending on the capabilities of the

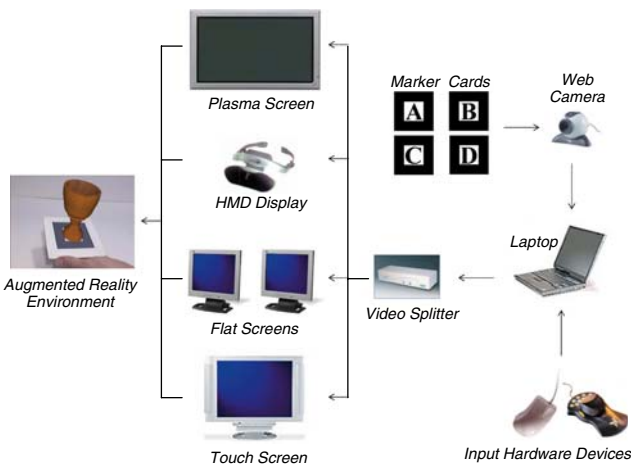

Fig. 1 Overview of operation of the system

splitter different configurations can be supported depending on the level of immersion and collaboration required. For example, for some applications (i.e. museum environments) the plasma screen could provide an idealistic cognitive environment for collaborative while the touch screen could be preferred as an effective means for user-centred interaction. All displays have been used to present the capabilities of the system in various demonstrations and other dissemination events and the plasma screen found to be the most appealing one. To further increase the level of interaction, a 3D mouse is integrated into the system allowing users to manipulate the virtual information in a natural way in six DOF (see Sect. 6.5).

Audio–visual augmentation techniques have been also been implemented (see Sect. 5) in order to achieve a realistic visualisation such as, matching virtual lighting to real lighting, texture mapping techniques, shading and clipping. To further improve the quality of the visualisation, planar shadows and reflections are generated in real-time so that the user can get a more realistic perception of the augmented information in respect to the real world. It is worth-mentioning that the software and hardware infrastructure of the prototype AR interface developed in this research is based on off-the-self hardware components and low-priced software resources. The hierarchy of the software architecture is presented in Fig. 2.

The blue boxes represent the off-line tools used and which form the basis of the implementation. The technologies in the orange boxes show the software components implemented for the creation of the AR interface. A brief overview of how each technology was used is presented in the following sections.

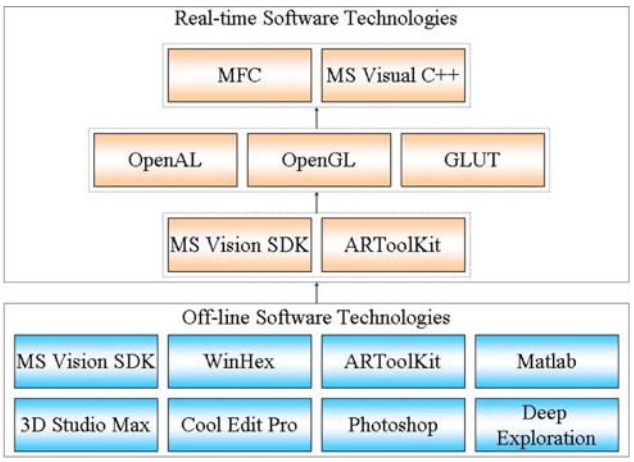

Fig. 2 Software technologies

3.1 Off-line technologies

The off-line software technologies include a number of commercial tools that must be used before the execution of the AR interface to prepare the content used in the augmentation (i.e. virtual information) as well as the AR environment. Specifically, the ARToolKit's tracking libraries were used for the calibration of the camera (see Sect. 4.2) as well as for the training of new markers designed for the needs of our research. Image processing (Adobe Photoshop) was appropriate for creating appropriate 2D images that were used as part of the visualisation process (see Sect. 5.2) and for generating textures for the 3D models.

To create professional-quality 3D models, 3ds max employed to digitise the models and export them into 3ds format. Next, deep exploration utilised to convert 3ds models into a number of formats including VRML and ASCII. CoolEdit Pro served as a useful off-line tool to record and processes all the necessary wave samples required for the augmentation. WinHex was helpful to analyse the robustness of the markers existing in the AR environment. Finally, the Calibration Toolbox for Matlab was used to improve the camera parameters calculated from ARToolKit (Sect. 4.2).

3.2 Real-time technologies

Real time software technologies consist of all the software libraries that have been integrated into a single application that comprise the AR interface. The Microsoft vision software development kit (SDK) was used as a basic platform to develop an interface between the video input (from and video or web cameras) and the rest of the AR application. Based on this, only ARToolKit's (Kato et al. 2000b) tracking library (AR32.lib) was integrated to calculate the camera pose in real-time. On top of the tracking library a high-level computer graphics rendering engine was implemented based on C++ that can perform mathematical operations between 3D vectors and matrices. Standard graphics functionalities like shading, lighting and colouring were based on the OpenGL API (Woo et al. 1999) while more advanced functions like shading and reflection were implemented in the rendering engine to provide a platform for the rapid development of AR applications (Sect. 7).

GLUT (OpenGL utility toolkit) (Angel 2003) was initially used to create a user-interface and to control the visualisation window of the AR interface. In addition, it was used for the textual augmentations (Sect. 5.4) because it provides sufficient support for

bitmap and stroke fonts. However, GLUT provides only a minimum set of functions for the user to control the visualisation and therefore a more advanced solution was implemented based on MFC (Microsoft foundation classes). The advantage of implementing a windows-based interface is that it allows users to familiarise quickly with the GUI (graphical user interface) as well as it provides menus and toolbars to implement any type of user interaction. Finally, OpenAL (open audio library) API was employed to generate audio in a simulated 3D space (Sect. 5.5) because it is similar to OpenGL coding style and it can be considered as an extension of it.

4 Tracking

A key objective of this research was to provide a robust platform for developing innovative AR interfaces. However, to achieve the best tracking (with commercial web-cameras) accurate calculation of the camera parameters is required. As mentioned before, ARToolKit's tracking library was preferred because it seems to provide accurate results with regards to the estimation of the location of the object especially at small distances and in cases where the camera is not moving fast. However, the major flaw of this approach is that all fiducials must be visible continuously. Also in un-calibrated environments, with poor lighting condition, tracking might not work at all. In this section, the results obtained from measuring ARToolKit's error and the algorithms used for calibrating the camera (calculating the camera parameters) are briefly analysed.

4.1 Measuring ARToolKit's error

ARToolKit was originally designed for small applications working on a limited range of operation, usually around one meter. In these applications the distance between the marker and the user is often small so most of the errors occurred are not easily detectable. But in wide area applications, its positioning accuracy is not very robust. In distances between 1 and 2.5 m the error in the x and y values increases proportionally with the distance from the marker (Malbezin et al. 2002). Because this research is focused on indoor environments, it is very important to work accurately in small distances ranging between 1 m and reasonably well for up to 3 m. For this reason an experimental measurement of the accuracy of ARToolKit's tracking libraries was performed in the laboratory environment. The aim of the experiment was to evaluate the error in distances ranging between 20 and 80 cm under normal lighting conditions. The experimental apparatus of this procedure is illustrated in Fig. 3.

The optimal area, which contains the least error, is the one that is perpendicular to the marker card. To allow placing the camera on specific points with high precision a grid is positioned on the ground (Fig. 3). Besides, a rigid path was designed so that the camera cannot loose its direction while moving backwards. For each point on the grid, numerous measurements of the location of the web camera in a local coordinate system were taken. The camera is setup in the shortest operating distance (20 cm) and after completing measurements on its position it moves backwards on a step of 1 cm. When the camera moves 60 cm (60 different positions) the program exits. For each position 20 measurements were taken and they were averaged. Figure 4 illustrates the results of this experiment (purple line) showing that the error is proportional to the distance. In very small distances the error in the detection of the marker is small while in larger distances the error becomes considerably bigger. It increases proportionally to the angle of rotation when the camera does not change position, but it is rotated around the Y-axis.

To verify that the best location is on the perpendicular axis of the camera and the marker, another set of measurements were recorded with the camera facing the marker at variable angle (yaw) having the other two (pitch, roll) stable. In this case, the camera was setup again in the same plane (ground plane) but the

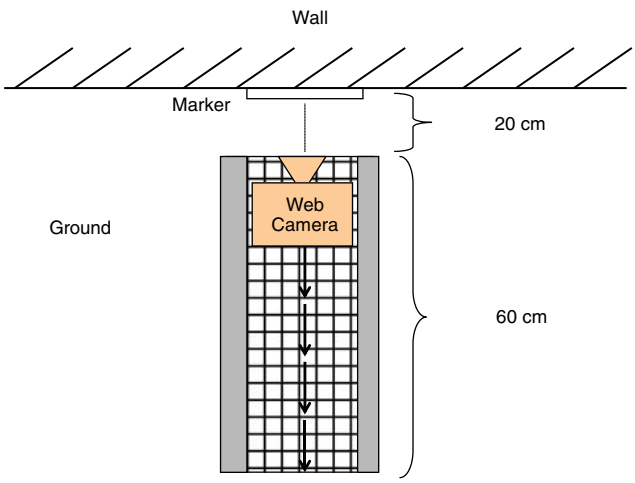

Fig. 3 Experimental setup for the measurement of ARToolKit's error

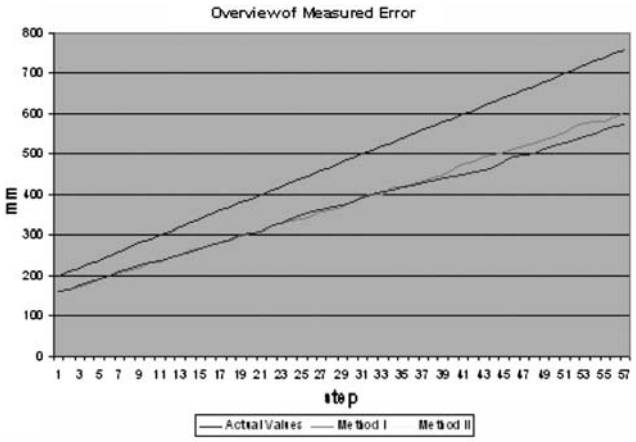

Fig. 4 Comparison of measured values

measurements were taken when the x and y values tended to zero values. It was measured that the angle in the initial position (20 cm from the wall, Fig. 3) is approximately 12° while for the final position the angle is approximately 4°. It is worth mentioning, that the second sets of measurements were not done automatically, so on each step the camera had to be manually adjusted to provide values as close as possible to values of x and y to zero. Figure 4 shows the results from the second experiment and illustrates that the measured error, when the camera is not pointing directly to the marker, is proportional to the distance. However, the difference is of minimal significance. This means that when the camera lies with a certain area and does not change its orientation the error is quite small. In contrast, if the camera changes direction the error increases considerably. Figure 4 illustrates differences in the errors produced from the experiments compared with the actual value (top dark line).

Figure 4 shows that the best results can be obtained when the camera is oriented to point at the centre of the marker. Even if the camera has a small offset, then the error increases linearly with the distance. Nevertheless, the tracking results are acceptable in the area of less than 1 m since the error is hardly noticed.

4.2 Camera calibration

This section describes the procedures used in order to calculate the intrinsic and extrinsic camera parameters. The purpose for this was to define an accurate camera model that can be effectively applied into indoor AR environments. Although there are a few camera calibration techniques available (Weng et al. 1992; Shi and Tomasi 1994) for calculating the intrinsic camera parameters, ARToolKit's calibration library (Kato et al. 2000b) was preferred since it works reasonable good in small distances and in cases where the camera is not moving fast. This method was originally applied to measure the camera model properties such as: the *center point* of the camera image; the *lens distortion*; and the camera's *focal length*. ARToolKit provides two software tools that can be used to calculate these camera properties, one to measure the lens distortion and the image center point, while the second to compute the focal length of the camera (Kato et al. 2000b). Based on this, the initial calibration was performed since it produces reasonable results for the calculation of the intrinsic camera parameters.

However, the greatest limitations of this vision solution include the tracking accuracy and the range of operation (Malbezin et al. 2002). To minimise some of the errors produced in the tracking of the markers, the extrinsic camera used had to be accurately estimated. The virtual objects will only appear when the tracking marks are in view. The size of the predefined patterns influences the effectiveness of the tracking algorithms. For instance, if the pattern is large then the pattern is detected further away. To calculate the extrinsic camera parameters the camera calibration toolbox (Camera Calibration Toolbox for Matlab 2003) was used which provides a user-friendly interface and it is very convenient when working with a large number of images. Another advantage over the previous method is that it provides very accurate results.

Before the camera calibration begins two steps need to be initially followed. In the first step the calibration rig must be generated while in the second all the calibration images must be collected. When done, the grid corners are easily extracted. The ToolKit offers an automatic mechanism for counting the number of squares in each grid and all calibration images used are searched and focal and distortion factors are automatically estimated. However, similarly to ARToolKit method, in most occasions the algorithm may not predict the right number of squares and thus provides a poor result. This can be clearer by observing the results of the calculation of the re-projection error. As it is clearly observed from Fig. 5a, the re-projection error is quite big compared to the scale. The reason behind this is because the extraction of the corners is not acceptable on some highly distorted images. However, the advantage of this technique is that it allows the user to improve the calibration. Specifically, the whole procedure can be repeated until the error is minimised up to a certain point.

After repeating the procedure for five times the error is reduced from a scale of five to a scale of one as illustrated in Fig. 5b. Furthermore, because ARToolKit accepts only binary data format for the calibration, a simple way to do this is to estimate the extrinsic

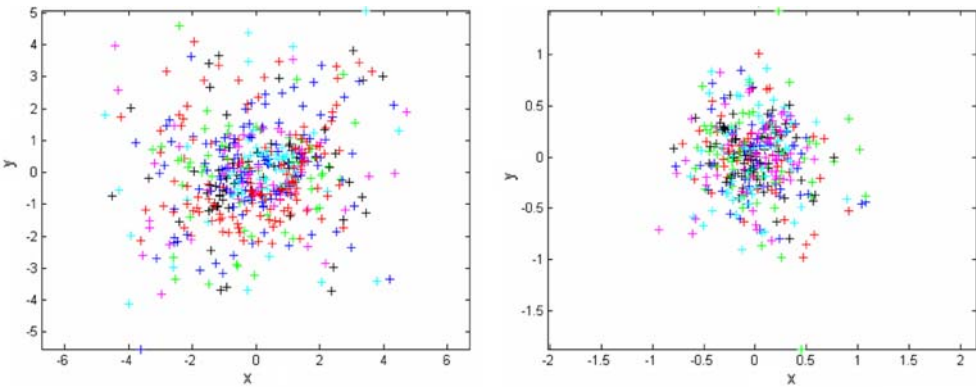

Fig. 5 Calculation of camera error **a** re-projection error **b** minimisation of error

parameters and then save the computed parameters in the data structure replacing the old values. The old data structure that holds the calculated camera parameters (ARParam struct) is shown below:

```
typedef struct {

int xsize, ysize;
double mat [3][4];
double dist_factor [4];
} ARParam;
```

In this structure the *xsize*, *ysize* and *dist_factor* have been experimentally replaced with the new values calculated from the above. Specifically, the camera parameters including the focal length (fc), the principal point (cc), the skew (s_k) and the distortion (k_c) have been computed and based on these values the intrinsic matrix can be defined as shown in Eq. (1):

$$\begin{pmatrix} \mathrm{fc}_0 & s_k\mathrm{fc}_0 & \mathrm{cc}_0 \\ 0 & \mathrm{fc}_1 & \mathrm{cc}_1 \\ 0 & 0 & 1 \end{pmatrix} \tag{1}$$

Since ARToolKit does not take into account the skew factor and makes use of the following matrix:

$$\begin{pmatrix} s_x f & 0 & x_0 \\ 0 & s_y f & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

To match the outputted camera matrix from Matlab and fit it into ARToolKit's matrix, the following matrix can be derived:

$$\begin{pmatrix} \mathrm{fc}_0 & 0 & \mathrm{cc}_0 \\ 0 & \mathrm{fc}_1 & \mathrm{cc}_1 \\ 0 & 0 & 1 \end{pmatrix} \tag{3}$$

After testing the new camera model a small improvement was succeeded in the distortion in the magnitude of 3–4%. As a further improvement, it was decided to add the skew parameters to the matrix, thus the skew parameter was used instead of zero in the matrix as shown below:

$$\begin{pmatrix} \mathrm{fc}_0 & s_k\mathrm{fc}_0 & \mathrm{cc}_0 \\ s_k & \mathrm{fc}_1 & \mathrm{cc}_1 \\ s_k & s_k & 1 \end{pmatrix} \tag{4}$$

Although the last modification provided us with a more correct camera model with an estimated improvement of about 1%, the effectiveness of the tracking system was not significantly improved. This is due to the fact that the optics used in the camera system (web camera) is really poor compared to professional video cameras. Other environmental issues that influence tracking include lighting conditions and range of operation.

5 Audio–visual augmentations

Each type of virtual media information is designed for specific purposes and as a result produces different outcomes. For instance, textual explanation can be utilized much more effectively than auditory description when communicating verbal information. On the other hand, pictures work better than text, for recalling or explaining diagrammatically a procedure. To describe a sequence of events video seems to be one of the most efficient techniques. In this section, the methodology used for the simultaneously multimedia visualisation of virtual information into an AR indoor environment is presented.

5.1 Object augmentation

An ideal AR system must be able to mix the virtual information with the real in a physical way. The participants should not realize the difference between the real and the augmented visualisation. The focus of this research is to present and implement methods of

realistically rendering 3D representations of real objects in an easy and interactive manner. The selection of the most appropriate 3D format is a crucial task in order to achieve a high level of realism in the system. In this research, both 3ds and VRML file formats have been used as shown in Table 1.

In any case, one of the first problems derived when displaying a 3D representation of a real model is the correct alignment on the required position. Virtual objects may appear to float on the marker and the user will be easily confused. This usually occurs because the 3D model is not registered correctly into the scene. For example, when a 3D object is transformed into the real scene it may appear below the origin as illustrated in the left image of Fig. 6.

To correct the problem of misalignment, Fig. 6a, a sorting algorithm for registering 3D objects precisely onto the top of the markers was implemented. To achieve a correct registration the virtual information need to be first sorted and then initialised to exactly the same level, as the marker is located in the Z-axis. An efficient way to align objects is by using a two-stage process. In the first part, the vertices of the object are sorted by the Z-axis. Upon completion, the vertices are translated to the minimum value, which is the origin of the marker cards, resulting in a proper object registration.

Next, to improve the realism of the AR scene a fast algorithm for planar shadows and reflections was implemented. The location of the shadow can be calculated by projecting all the vertices of the AR object to the direction of the light source. To generate augmented shadows an algorithm that creates a 4×4 projection matrix (P_s) in homogeneous coordinates must be calculated based only on the plane equation coefficients and the position of the light (Moller 1999). Say that L is the position of the point light source; P the position of a vertex of the AR object where the shadow is cast; and n the normal vector of the plane. The projection matrix of the shadow can be calculated by solving the system, which consists of the equation of the plane and a straight that passes from the plane point in the direction of the light source (see Eq. 5).

$$p_s = \begin{pmatrix} Lp \bullet Pc - Lp_0 \times Pc_0 & 0 - Lp_1 \times Pc_0 & 0 - Lp_2 \times Pc_0 & 0 - Lp_3 \times Pc_0 \\ 0 - Lp_0 \times Pc_1 & Lp \bullet Pc - Lp_1 \times Pc_1 & 0 - Lp_2 \times Pc_1 & 0 - Lp_3 \times Pc_1 \\ 0 - Lp_0 \times Pc_2 & 0 - Lp_1 \times Pc_2 & Lp \bullet Pc - Lp_2 \times Pc_2 & 0 - Lp_3 \times Pc_2 \\ 0 - Lp_0 \times Pc_3 & 0 - Lp_1 \times Pc_3 & 0 - Lp_2 \times Pc_3 & Lp \bullet Pc - Lp_3 \times Pc_3 \end{pmatrix} \quad (5)$$

where $L_p \cdot P_c$ is the dot product of plane and light position. The projection matrix has a number of advantages compared with other methods (i.e. fake shadows) but the most important is that it works fast and it is generic so that it can generate hard shadows in real-time for any type of objects independently of their complexity (Liarokapis 2005). An example screenshot that illustrates planar shadows is shown in Fig. 7.

The main disadvantage of this algorithm is that it renders the virtual information twice for each frame: once for the virtual object and another one for its shadow. Another obvious flaw is that it can cast shadows only into planar surfaces but with some modifications, it can be extended to be applied to specific cases such as curved surfaces (Liarokapis 2005).

To realistically model reflections in AR environments, many issues must be taken into account. Although in reality the light is scattered uniformly in all directions depending on the material of the object in this work, the effect of mirror reflections has been implemented. An example screenshot of a virtual object casting a shadow and a reflection on a virtual plane is illustrated in Fig. 8.

Table 1 Categorisation of 3D file formats

Advantages	Disadvantages
3ds	
Includes pre-vertex texture coordinates	3D can have 216 vertices maximum
Unknown parts can be skipped	Poor normal information
VRML	
Easy to read	Contains less information than 3ds
Standard for 3D internet presentations	Does not support advanced lighting and texturing
Contains animation and collision detection	

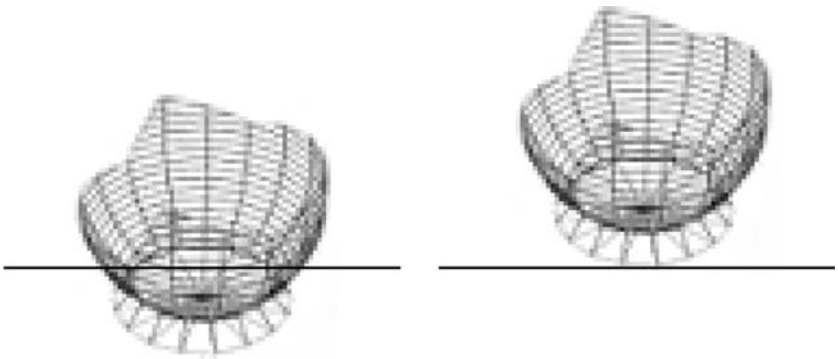

Fig. 6 Object augmentation **a** misalignment of object **b** correct registration

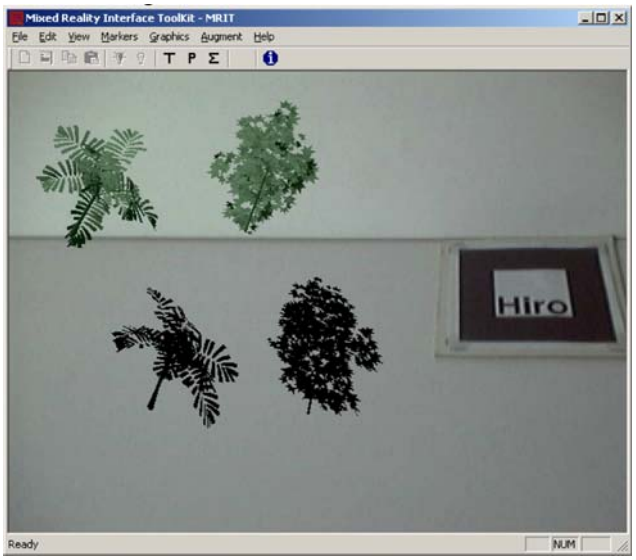

Fig. 7 Illustration of planar shadows

Fig. 8 Planar shadows and reflections

Based on the OpenGL's stencil buffer a reflection of the object is performed onto a user-defined virtual ground. The stencil buffer is initially set to sixteen bits in the pixel format function. Then, the buffer is emptied and finally the stencil test is enabled.

5.2 Image augmentation

Images are widely used as a means to increase realism and in the past, they have been used with success for educating purposes. The augmentation of images is a highly cost effective means to present simple 2D information in the real world. The use of their operation may be performed in a number of different ways depending on the learning scenario applied. The digital image augmentation can be either static or dynamic. Dynamic image augmentation is widely used for achieving video augmentation (see Sect. 5.3). With static augmentation, a single image only is rendered into the scene. Based on the theoretical framework provided by Smith (1994), images used for AR environments have been categorised into description, symbolic, iconic and functional as shown in Table 2.

The algorithm used is simple but very efficient and can be applied into two types of image formats (BMP and TGA). First, it loads an image file and checks if it is a valid image format. In the next step, textures are generated using data from the image file. Following this, the texture is created and the parameters are set based on the OpenGL API. Finally, the texture is bound to the target texture, which is a quadrilateral.

5.3 Video augmentation

The mode of operation within the video AR system is to read an AVI file, decompose it into $256 \times 256 \times 24$ bit images, mix it with the dynamic video (coming from the camera) and finally display it on the selected visualisation display (Liarokapis 2005). When

Table 2 Categorisation of images augmentation

Purpose	Usage
Description	
Most popular format	Explain a 3D real object
Describe the real world	Textual information have a useful meaning
Image itself can tell a self-explanatory story	
Symbolic	
Identify a basic principle or symbol	Concerns images that represent various types of well known symbols
Allow both simple and complex symbolism	
Interpretation can change over time	
Iconic	
Identify a case of a multinational meaningful icon that is not related to a specific language	Image contains different types of iconic representations that can illustrate something useful
For example the ''exit'' or ''danger'' sign	
Functional	
A single operation can be expressed	Functional images act as virtual buttons and a specific operation is assigned on each
Multiple operations can be also supported	

the video file is loaded, the program automatically counts the number of frames so that its size is known. Then all frames are decomposed into 2D images and each image is applied to a square quad, exactly in the same way as textures are wrapped to objects. It is worth mentioning here that because each animation has a specific length (in seconds) and its own frame rate, the time required for each frame is calculated.

Moreover, the augmented video starts automatically when two things occur: a marker is detected and user has loaded a particular file from the filling system using the interface menu. When the animation is completed it repeats itself until the user decides to stop it (by pressing a keyboard key or using the interface menu). To increase the feasibility of the system, if the camera is not in line of view with the marker card, then the video augmentation will continue playing until the video sequence is finished. This was designed on purpose to prevent cases where the user changes position or orientation rapidly and thus looses the perceived visualisation. The augmented animation can be controlled in a number of different means including the cease of the animation, resize the animation window or even manipulate the augmented video animation into six DOF (see Sect. 6).

Figure 9 shows four frames of a video sequence superimposed into a marker card, describing a complex concept in electronics (i.e. Moore Diagram). In terms of efficiency, the video augmentation results range between 20 and 35 FPS depending on the resolution of the videos. However, the drawback of this method is that when the animation is augmented the overall performance of the system is significantly reduced. In particular, the performance was experimentally measured to be reduced by approximately 20–50% the FPS of the system. For instance, if the performance of the system is in real-time (i.e. 25 FPS) then the AR video algorithm would drop the performance to 12–20 FPS. Another limitation of the proposed video augmentation is that it can currently decompose videos into only $256 \times 256 \times 24$ bit images.

5.4 Textual augmentation

Textual annotations are the simplest form of information that can be easily augmented in any type of AR environments. This can be either presented as a *label* or as a *description*. Label text has been used in the past (Klinker et al. 1997; Sinclair and Martinez 2001), to point out specific parts of a complex system using the minimum textual information. In this case, the most important aspect is to ensure that the augmented labels do not obscure each other and that the information is clearly presented to the user. Description text requires a much more demanding process because it needs to provide complete information about an object or about a virtual operation. The problems begin in cases where the magnitude of textual information needs to be augmented on a display is large.

In this research, label and descriptive textual information was performed by dynamically loading ASCII text files. Each file contained a very different level of information depending on the reasons for utilising it. For example, label text files were defined to specify the type of visualisation (i.e. image augmentation) or the name of an object. The main advantage of this method is that the textual information, which will be augmented on the real environment, is stored on a txt file. Text files are widely used and can be easily transferred

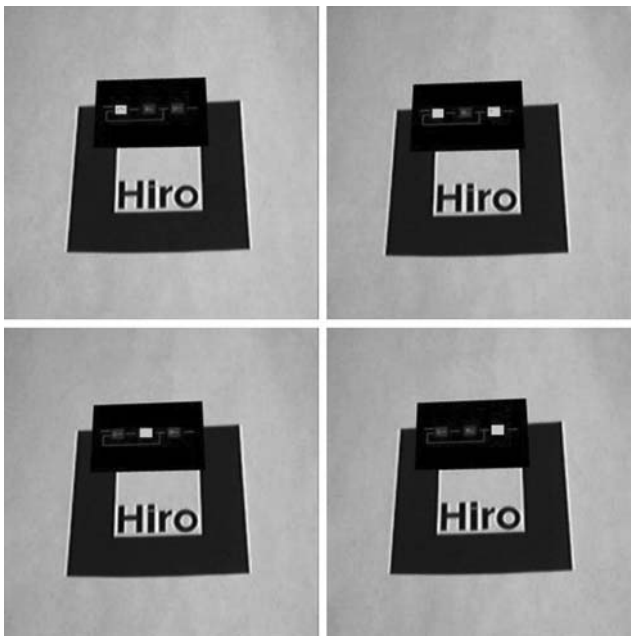

Fig. 9 Video augmentation

over all types of networks. Users of the system can position textual augmentations anywhere in the real environment using standard transformations. In addition, they can change their appearance in terms of colour, size and font type (Bitmap, Times Roman and Helvetica).

5.5 Audio augmentation

In the real world, audio is a process that is heard spatially and thus it is a very important aspect for any simulation scenario. The most important issue when designing 3D sound is to ''see'' the sound source (Yewdall 1999). However, most AR applications have not incorporated 3D sound component even if it can contribute to the sense of immersivity. The augmented sound methodology followed in this work, has some similarities with the ASR approach (Dobler et al. 2002) in the way virtual sounds are augmented in the real environment. Unlike this experimental approach, which is based on a Creative EAX API, the implemented 3D audio system is based on OpenAL, which has many similarities with OpenGL and was originally designed for generating 3D sounds around a listener. The recording of speech sounds was done in mono format, using a standard microphone and the mono samples were converted into stereo format. Furthermore, each sound source in the system has been specified to have the following three properties: *position*, *orientation* and *velocity*. The spatial audio system can handle multiple sound sources and mix them together. The user can move the sources in 3D space using the keyboard and menu interaction techniques illustrated in Sect. 6.

The spatial sound algorithm first initialises all the necessary OpenAL variables (position, orientation and velocity) and then loads them into the appropriate buffers (format, length and frequency) for further processing. Next, sound sources and buffers are initialised and the sources are assigned to the buffers. The picth and gain are set to one and the sources are set into a continuous loop unless stopped by the user. Each time the camera detects the marker the transformation matrix is inverted to estimate the position of the camera. In the context of this research, the distance model experimentally applied used to simulate the distance followed the linear equation as illustrated below:

$$y = \alpha x + \beta \tag{6}$$

where α represents distance between the camera and the marker and β the offset position of the marker card. Although this cannot accurately represent the distribution of sound in 3D space it provides very good results. To provide more freedom to the listener the values of the linear function may change depending on the requirements of the visualisation. If the sound source is positioned in the origin then the above equation may be re-written as shown below:

$$Listener = \frac{camera\_position}{distance\_factor} \tag{7}$$

where *camera_position* refers to the inverse transformation of the camera and *distance_factor* to a constant number. To achieve a realistic simulation of the sound different values have been tried to simulate the distance_factor. However, this constant value may change off-line depending on the requirements of the visualisation. For example, some users may prefer to perceive the auditory information louder than others do. In addition, the system is capable of loading and mixing music sound files. This option can be extremely useful for simulating surround music audio. The sound files may be overlaid into the same marker or onto a different marker depending on the needs of the application.

6 Human–computer interactions

Human–computer interactions are one of the most important issues when designing a robust real-time system. They have to be performed in a natural way so that inexperienced participants familiarise quickly in the AR environment (Liarokapis et al. 2004a). The proposed

interface allows users tangible interaction with various types of multimedia information such as 3D models, images, textual information and 3D sound, using a number of interaction techniques. Interactions controlled by the user-computer can be distinguished into five different categories including physical manipulation, interface menu interaction, standard I/O, Touch Screen and SpaceMouse interaction as illustrated in Fig. 10.

Although, some types of interactions proposed in Fig. 10 are not novel (i.e. physical manipulation), the novelty comes in the way they are used by the participants. Participants can combine two or more types and experience a novel form of interaction with great flexibility. For example, the most significant combination of human–computer interactions is the use of intuitive methods like the physical manipulation with sophisticated devices such as the SpaceMouse. Users can hold in one hand a marker card with a virtual object superimposed and on the other hand use the SpaceMouse to perform graphics operations like virtual lighting. In the following sections, all the types of interactions are explained in detail.

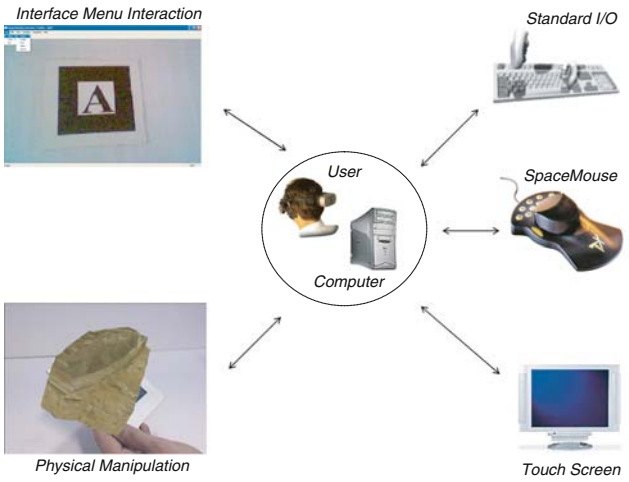

Fig. 10 Interactions within the system

6.1 Standard interactions

The first method is addressed to users with some computer experience and is based on standard interaction input devices like the keyboard and the mouse. For example, by pressing buttons (hot keys) the visual parameters of the virtual objects can be changed faster instead of using the menu dialogues. Some of the most characteristic are described in (Liarokapis et al. 2004a, b; Liarokapis 2005) and include the change of lighting conditions (ambient, diffuse, specular and shiness); the texturing information (standard and environmental); the switch from solid mode to wireframe mode; and others (see Sect. 6.2). Moreover, the keyboard is also employed for changing the position (translation), orientation (rotation) and scaling of the virtual information in six DOF. Initially, the above transformations were implemented based on the OpenGL functionality but soon it became obvious that OpenGL could not meet the requirements of this research because it provides only the minimum functionality to rotate an object around X, Y or Z-axis. However, in a tabletop AR environment this is constraining the user when rotating the virtual information as well as it restricts the use of simultaneous interactions. To tackle this problem a generic rotational matrix that takes as input three angles and rotates the object around an arbitrary axis is specified in Eq. 8:

$$\begin{bmatrix} \cos\cos\psi & -\cos\sin\psi & \sin \\ \sin\varphi\sin\cos\psi+\cos\varphi\sin\psi & -\sin\varphi\sin\sin\psi+\cos\varphi\cos\psi & -\sin\varphi\cos \\ -\cos\varphi\sin\cos\psi+\sin\varphi\sin\psi & \cos\varphi\sin\sin\psi+\sin\varphi\cos\psi & \cos\varphi\cos \end{bmatrix} \quad (8)$$

Based on the above rotational matrix, it became possible for users to rotate virtual information around an arbitral-defined axis. The above matrix was also implemented to the standard mouse providing a quick way to perform intuitive rotations. Although, it provides the means to perform a rotation around all three axes simultaneously if one interaction device is used, problems occur when more than one device is used (i.e. keyboard and mouse). An alternative way of performing transformations is by using quaternions. To specify multiple rotations, many intermediate control points are required where a quaternion interpolation depends only on the relation between the initial and final rotations. The easiest way to prove the link between a rotation matrix and a quaternion is by linking them in three dimensions. Say that $q = s + v \cdot I$ a unit

quaternion and defined Q, where $v = (u_x, u_y, u_z)^T$, it can be shown that there is a 3×3 matrix that represents a rotation matrix of the form (Eq. 9):

$$vv^T + (sI_{3\times3} + C_u)^2 = \begin{pmatrix} s^2 + u_x^2 - u_y^2 - u_z^2 & 2(u_x u_y - s u_z) & 2(u_x u_z + s u_y) \\ 2(u_x u_y + s u_z) & s^2 + u_x^2 + u_y^2 - u_z^2 & 2(u_y u_z - s u_x) \\ 2(u_x u_z - s u_y) & 2(u_y u_z + s u_x) & s^2 - u_x^2 - u_y^2 + u_z^2 \end{pmatrix} \quad (9)$$

To obtain a quaternion corresponding to a given rotation matrix we first define an arbitrary rotation matrix R and then the corresponding quaternion $q = s + u_{xi} + u_{yj} + u_{zk}$ to the rotation matrix. Using the above equation it is easy to solve the equation and derive the values for u_x, u_y and u_z, respectively. In OpenGL, rotations are specified as matrices since homogeneous matrices are the standard 3D representations. By combining the property of unit quaternion with the above rotation quaternion matrix we can deduce the following equation (Eq. 10):

$$vv^T + (sI_{3\times3} + C_u)^2 = \begin{pmatrix} 1 - 2(u_y^2 - u_z^2) & 2(u_x u_y - s u_z) & 2(u_x u_z + s u_y) \\ 2(u_x u_y + s u_z) & 1 - 2(u_x^2 + u_z^2) & 2(u_y u_z - s u_x) \\ 2(u_x u_z - s u_y) & 2(u_y u_z + s u_x) & 1 - 2(u_x^2 - u_y^2) \end{pmatrix} \quad (10)$$

Other functions that were integrated to the mouse include translations and scaling. On the other hand, using the mouse users can access the carefully designed GUI. This allows users to have full access to the superimposed virtual information. An example is presented in Fig. 11, where users can select the information that is going to be augmented on the real environment.

6.2 GUI Interactions

On the other hand, using the mouse or the Touch Screen users can access the functionality that has been carefully integrated into a novel GUI. The GUI consists of a menu, a toolbar, a status bar and a number of

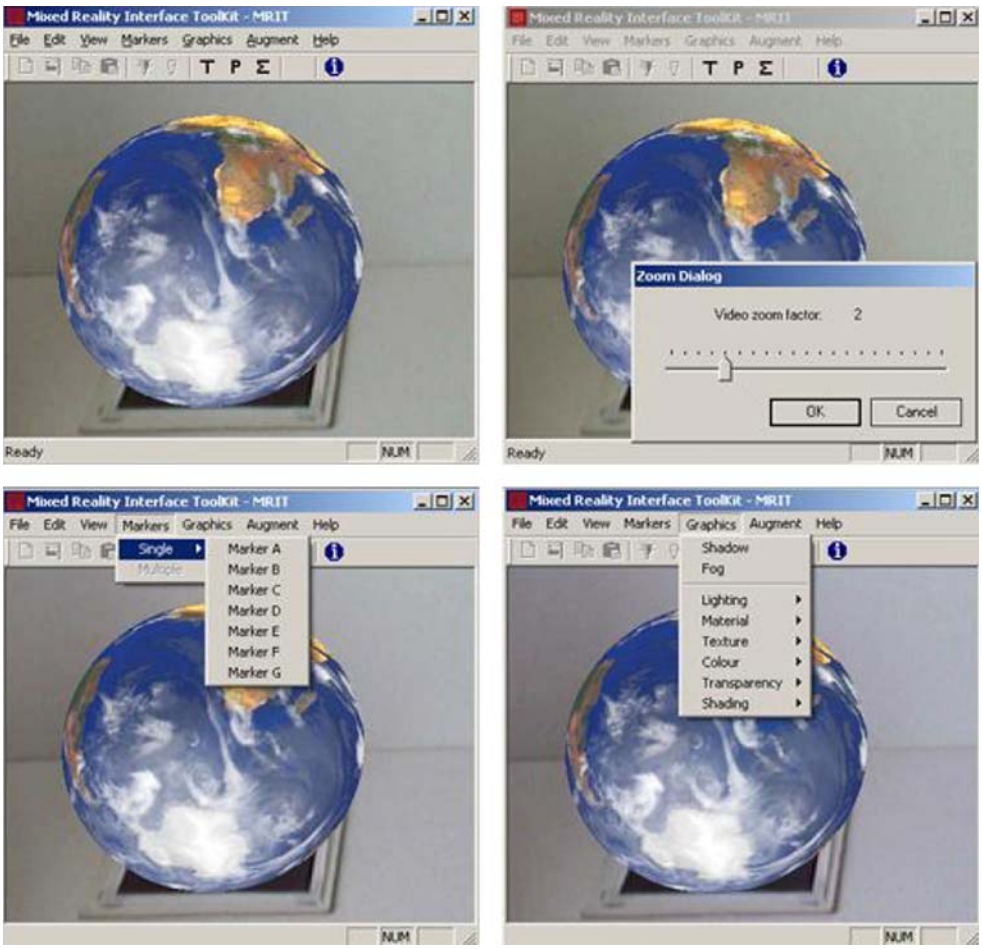

Fig. 11 GUI functionality

dialog boxes. This allows participants to have the same access to the augmented virtual information as if they were using standard interaction techniques. Four example screenshots that illustrate some of the functionalities of the GUI is presented in Fig. 11.

The greatest advantage of the proposed GUI is that it allows participants to perform complex operations very accurately. Specifically, sometimes it is of crucial importance to transform a virtual object in a specific location in the real environment. Using other methods it could take a great amount of time and effort (depending on the experience of the user) to achieve this and it will definitely not be very accurate. However, the GUI interaction techniques offer the solution to the problem using double point precision.

Next, the ''Edit'' category consists of three basic operations including video (start or stop), a zoom dialog box and a scale dialog box. The ''View'' category consists of two sets of operations. Firstly, a Toolbar and a Status bar, which is commonly found in windows based applications. It is worth-mentioning here that the GUI has been built on top of the windows API so that full compatibility with windows based operating systems is ensured. As far as this research is concerned this is the only true windows based interface that can superimpose five different types of multimedia content into the real environment.

The second set of operations consists of three functions called axis (to insert a Cartesian set of axis indicating the origin of the AR environment), debug (to threshold the live video sequence and thus check whether a marker is detectable) and clip (to clip the graphics geometry). The rest of the menu categories (graphics and augment) are used to control visualisation properties of the augmented information. Functions that have been implemented include shadows, fog, lighting, material, texturing, colouring, transparency and shading. Finally, the ''help'' category provides some information about the release version of the AR interface as well as the date and the author name.

6.3 Physical manipulation

Physical manipulations were specifically designed for users with no computer experience and refers to a physical manipulation of the marker cards (Kato et al. 2000a; Billinghurst et al. 2001). As illustrated in Fig. 12, users can manipulate freely the marker cards in six DOF to receive a different perception of the superimposed information.

Another benefit of natural interactions is that they can be used with the other types of interactions described in this section. This allows producing unique combinations (see Fig. 14) that can provide solutions for specific AR applications that require a high-level of interaction. In addition, apart from using the marker cards for just superimposing virtual information, they have been used to perform some basic operations such as: assign an object into a marker, de-assign an object from a marker, scale, rotate and translate. The advantage of this method is that users can use only physical objects (marker cards) to visualise and interact with the virtual information. However, the disadvantage is that when multiple markers are used the overall efficiency of the system is reduced. Specifically, the template matching algorithm used operates very

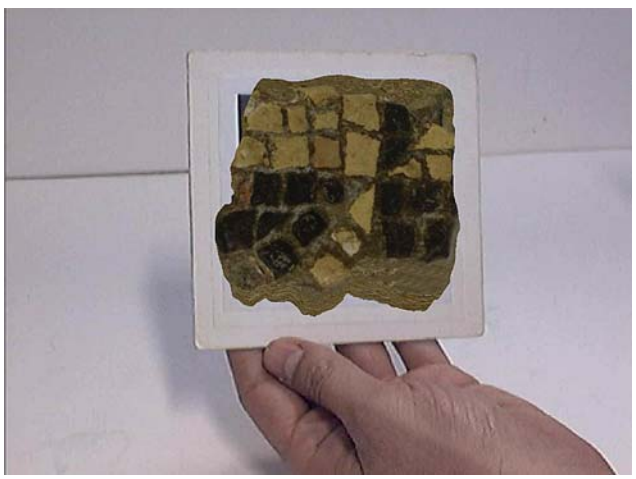

Fig. 12 Natural manipulation of virtual object

```
Read Event from hardware device
    if Translation button
        if Tx is positive then Translate forward in X axis
        if Tx is negative then Translate backward in X axis
        if Ty is positive then Translate forward in Y axis
        if Ty is negative then Translate backward in Y axis
        if Tz is positive then Translate forward in z axis
        if Tz is negative then Translate backward in Z axis
    if Rotation button
        if Rx is positive then Pitch right in X axis
        if Rx is negative then Pitch left in X axis
        if Ry is positive then Yaw right in Y axis
        if Ry is negative then Yaw left in Y axis
        if Rz is positive then Roll right in Z axis
        if Rz is negative then Roll left in Z axis
    if Scaling button
        if S is positive then Scale up by a predetermined factor
        if S is negative then Scale down by a predetermined factor
    if Lighting button
        Ambient Lighting is turned on/off
    if Ground button
        A virtual ground drawn to occlude the marker card is turned on/off
    if Clipping button
        An invisible plane that clips the AR scene is turned on/off
    if Reset button
        Initialise the transformations to default values
```

Fig. 13 Pseudo code for SpaceMouse

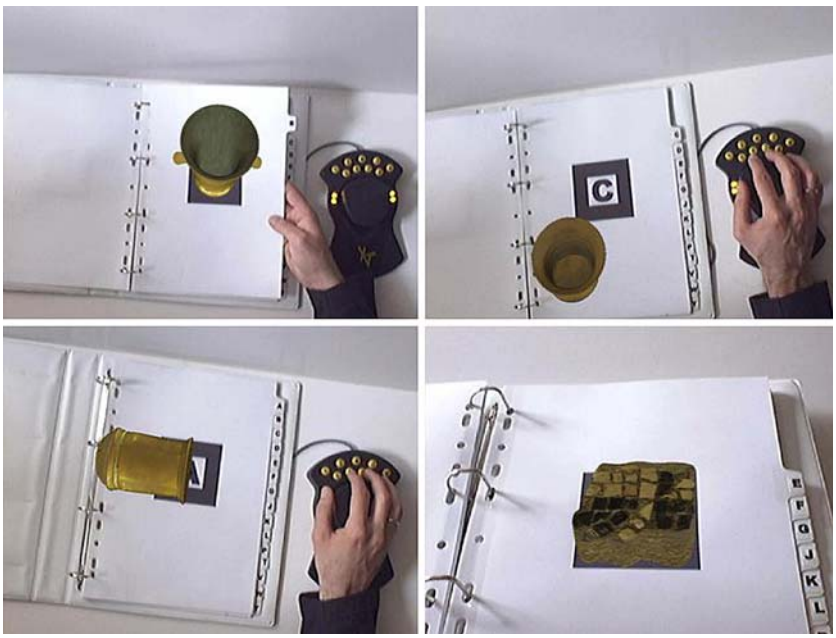

Fig. 14 SpaceMouse interactions

effectively in real-time performance with one marker but it starts to decrease drastically as more markers are added. The reason behind this is because for each marker the algorithm is aware; it creates four templates, one at each orientation. Each marker has to be compared to all known templates until the best match is detected. To calculate the number of comparisons performed by the algorithm the following equation is illustrated:

$$N_c = 4 \times N_m \times N_t \tag{11}$$

where N_c is the number of comparisons, N_m is the number of known markers and N_t corresponds to the number of known templates. If in the scene there are 10–20 markers and the application knows about 250 markers then the system performs around 10,000–20,000 comparisons. This makes any system to run much slower and makes the application operate in less than 25 FPS. Thus, to achieve a fast AR application in the final system it was preferred to use as few markers as possible having as limit ten markers.

6.4 Touch screen interactions

An alternative way of interacting with the virtual information is to make use of interaction devices such as Touch Screens. This is ideal for some application scenarios where, the use of other interaction devices is not possible. For example, in museum environments, Touch Screens are the most appropriate means of interacting with the virtual exhibitions. Besides, although it was easy to integrate the Touch Screen to the AR interface, many problems arose when users tried to interact with the GUI menu. The reason for this is because the menus in the GUI were too small and it was difficult for some users to select. To tackle the problem, large toolbar buttons and dialog boxes were designed and associated with appropriate functionality. The main advantage of using the Touch Screen is that it can serve both the visualisation and interaction all in one device. However, the major drawback is that the effectiveness of the interactions is dependent on the effectiveness of the GUI. If the GUI is not user-friendly, it will affect the usefulness of the Touch Screen interactions.

6.5 SpaceMouse interactions

Finally, users can manipulate virtual information using sophisticated VR sensor devices such as SpaceMouse (Liarokapis et al. 2004b) and InertiaCube. SpaceMouse allows the programmer to assign functionality to provide a customised nine button-menu interface. This method has the advantage manipulating virtual information in six DOF in a natural way using only one hand. A combination of C++ functions, SpaceMouse commands and OpenGL allowed the integration of the 3D mouse into the system. Important functionalities that have been implemented and assigned to the menu buttons include either standard graphics transformations for easier manipulation, or more advanced graphics operations (Fig. 13).

In Fig. 13, S represents the scaling operations, T_x, T_y and T_z represent the translations and R_x, R_y, and R_z the rotations. To perform one of the above operations

the user has to press one of the buttons (the translation button for example) and then use the bar to translate the object in 3D space. Depending on which direction force is applied, the object will move respectively. Furthermore, the ambient lighting, the clipping of superimposed geometry through an infinite plane and the augmentation of a virtual plane can be switched on and off using the remaining SpaceMouse buttons. Four example screenshots of a user interacting with 3D information using the SpaceMouse is illustrated in Fig. 14.

It illustrates how a user can adapt the MagicBook approach (Billinghurst et al. 2001) in conjunction with the SpaceMouse to visualise and interact with the virtual artefacts. On the top left image, the user is only visualizing the virtual artefact (marker B) while on the top right image, the user translates the artefact using the SpaceMouse. On the bottom left image, the user interacts (rotates) with another artefact (marker A) and on the bottom right image the user visualises another artefact (belonging on marker E). The most important limitation of this tangible interface is the use of a single marker for tracking by the computer vision based tracking system.

7 Application scenarios

To test the functionality of the proposed AR interface system two application scenarios have been designed. The first section (see Sect. 7.1) presents an educational application used to support and simplify teaching and learning techniques currently applied in the higher education sector. The second section (see Sect. 7.2) illustrates a museum application with the aim of facilitating access to museums and other cultural heritage galleries. In the following sections, each application is briefly analysed and the most important findings of the research are presented.

7.1 Educational application

Most educational AR applications operate in indoor environments (Begault 1994; Fuhrmann and Schmalstieg 1999) and the scenarios proposed in this section are focused on enhancing the teaching and learning process for higher education institutions like colleges and universities. With this purpose in mind, AR educational scenarios have been designed to assist teachers to transfer knowledge to the students in other ways than traditionally has been the case (Liarokapis 2005). The aim is to provide a rewarding learning experience that is otherwise difficult or impossible to obtain by offering the ability to achieve better user interaction (with teaching material and complex tools) while the provision of an interactive augmented presentation provides students a high degree of flexibility and understanding of the teaching material. All scenarios are specifically engaged with the improvement of learning and teaching techniques in the fields of engineering and informatics at the University of Sussex.

Based on the functionality of the AR interface described in the above sections, a lecture was prepared introducing students on how computers work. This application has in practice some similarities with the experimental application proposed by Fernandes and Miranda (2003). However, the higher education application offers a very powerful user interface that allows audio–visual augmentation as well as simultaneous interactions. From a visualisation point of view, the system displays the data in a single window and the lecturer can describe basic IT principles with the use of AR technology in a number of different ways. In Fig. 15, a PowerPoint slide presentation that describes the characteristics of a computer system as well as relative textual information is augmented onto the appropriate marker card.

Learners can zoom into the diagram in two ways. Firstly, by using the predefined functionality (scale and translate) existing in the keyboard, the menu and the SpaceMouse interfaces. Alternatively, learners can either move the marker card intuitively closer to the camera and vice versa. In both ways, potential users can clearly observe and understand the theoretical

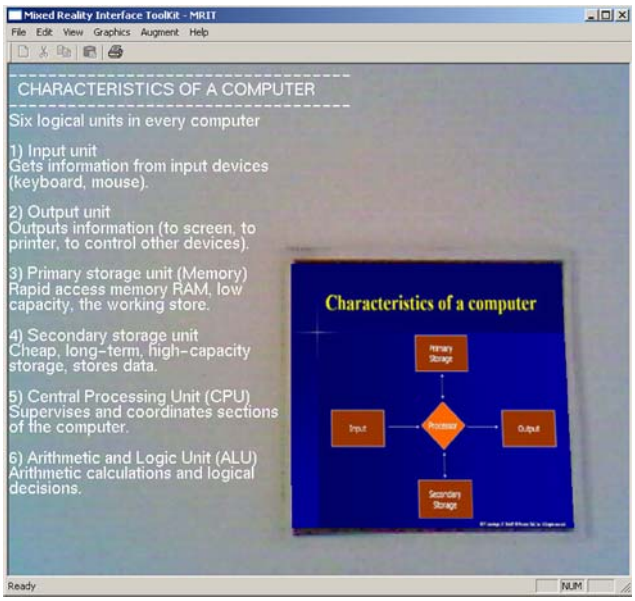

Fig. 15 Teaching IT using AR

operation of a computer. The textual information describes the diagram in detail providing a more complete learning presentation simultaneously. Learners can now get simultaneously appropriate audio–visual information that helps them to acquire a deeper knowledge about the characteristics of a computer. In the same way, to increase the level of understanding of the teaching material presented to the students, 3D information can be presented to deepen the level of knowledge transfer. Along these lines, learners can have a more rounded idea of what are the main characteristics of a computer, what are the main parts and how they look like in reality.

The main advantage of the educational application over the traditional teaching methods is that learners can actually ''see'' and ''listen'' the virtual information superimposed in the real world (Liarokapis et al. 2002). Students can naturally manipulate the virtual information using standard or sophisticated VR devices and they can repeat a specific part of the augmentation as many times as they want. Another benefit of the system is that it does not require students to have any previous experience to operate it. Finally, even AR has been experimentally applied for teaching engineering and information technology (IT) courses it has been designed in such a way that it can be easily adapted and applied very easily to other educational courses.

7.2 Cultural heritage application

The concept of virtual exhibitions in museums has been around for many years and researchers have designed and developed several applications (Liarokapis et al. 2004a, b; Hall et al. 2001; Gatermann 2000). In addition, a number of museums hold innumerable archives or collections of artefacts, which they cannot exhibit in a low cost and efficient way. Museums simply do not have the space to exhibit all the artefacts in an educational and learning manner. Augmented Representations of Cultural Objects (ARCO) was an EU-funded research project (completed in September 2004) in order to analyse and provide innovative but simple to use technical solutions for virtual cultural object creation and visualisation. In short, ARCO provides museums with a set of tools that allow them to digitize, manage and present artefacts in virtual exhibitions. To evaluate the usability of the system properly ARCO collaborated with Victoria and Albert Museum and the Sussex Archaeological Society.

The work illustrated in the previous sections has been applied in ARCO to explore the potential of AR in a museum environment by mixing virtual information in an environment comprised of real objects. The success of an AR exhibition is highly related to the level of realism achieved. In general, there are a few AR applications that do not require a high level of realism, but within the cultural heritage field realistic visualisation is an important issue (Liarokapis et al. 2004a). The scenarios illustrate how virtual museum visitors can visualise archaeological information such as virtual artefacts or even whole virtual museum galleries providing an educational narration for the preservation of cultural heritage. An example screenshot of four different virtual galleries from Victoria and Albert Museum are illustrated in Fig. 16.

In theory, this technique can be extended to as many markers as long as the camera can detect them within the field-of-view. The major drawback of this method is that the frame rate drops analogous to the number of markers used (as illustrated in Sect. 6.3) but the overall effectiveness in all galleries was between 25–30 FPS. Furthermore, the realism of the system highly depends on the 3D modelling procedure and for this reason the 3D models used in this scenario are the very high resolution models. The visualisation of an expedition to large groups of people can be considered as a collaborative activity. By looking at and interacting with the artefact visualisation visitors can communicate with each other by expressing their thoughts about any aspects that relate to the history of the artefact. This results in an exchange of opinions amongst the visitors in an implicit and explicit way. By zooming into the artefact more contained arguments can be made about the nature of the material used for its construction. On the other hand, in a perspective view more verbal communication is possible. By using the configuration setting of the collaboration in the AR interface, visitors can use HMDs and obtain a completely immersed view.

8 Preliminary evaluation

The knowledge gained from reviewing the literature and the experimental results, enabled an initial dissemination of the prototype AR interface. Even if this work is still on an experimental status, the results can be taken into consideration to improve the effectiveness of the presented system as well as to design future high-level AR interfaces. An expert-based evaluation approach was followed that argues that formal laboratory user studies can effectively evaluate visualisation when a small sample of expert users is used (Tory and Möller 2005). In terms of evaluating the system, some initial empirical research was conducted based on a two stage human-centred questionnaire. The first part

Fig. 16 Virtual museum gallery visualisation

is generic but aims at evaluating the usability of the system in the learning process while the second part is more technical and refers to the effectiveness of the visualisation and interaction techniques of the interface. An educator would design the questionnaire in a different way taking primarily into consideration educational aspects whereas in this case, the purpose was to obtain a number of useful conclusions regarding the technicalities and practicalities of the system. This pilot study was disseminated to a five research staff from Sussex University that had experience in working with VR applications. Four were men and one was woman. Subjects were between the ages of 24 and 28 and the average time of the evaluation was 30 min.

8.1 General questions

The feedback received following the completion of the evaluation process varied but in general lines was encouraging. As far as the first part of the questionnaire is concerned, all the users thought that the system has the potential to be used as a learning tool in the future although it currently lacks from interoperability issues. Specifically, they argued that the application scenarios were really interesting and exciting but for teaching purposes more comprehensive learning scenarios have to be implemented. The findings from this study are summarised in Table 3.

Table 3 General questions about the system

General questions	Mean (max = 5)	SD (yEr±)
Rate quality of performance?	4	0.7071
Rate the overall usage?	3.6	0.5477
Can aid the education process?	3.2	0.4472
Create new AR applications?	4.6	0.5477

Results illustrate that 92% of the users believe that the system has the potential to be used as a basic platform to create AR scenarios and applications whereas 80% rate the quality of the system good. On the other hand, 72% of the users liked the overall usage of the system and 64% feel that educational applications could benefit from this technology. Moreover, two users mentioned that the system would be much more useful if a multimedia database system with a content management system is used to increase interoperability issues. Another one stated that a print function would help to capture and store into images the different views of the AR environment.

8.2 Technical questions

Regarding the second part, all users agreed that the system is very easy to use and that the visualisation process is more than satisfactory. Surprisingly, most of the users preferred the HMD-based visualisation versus the monitor-based visualisation (Fig. 17).

Figure 17 shows the user-response in comparing the monitor-based AR (mean = 6.8, SD = 2.77489, SE = 1.24097) versus video see through HMD-based AR (mean = 8.2, SD = 2.48998, SE = 1.11355). Similar studies have shown the exact opposite result but in this study all users were computer literature and all had

previously used VR prototypes that make use of HMDs. Moreover, many difficulties were observed when participants tried to move around with the camera mounted on the HMD because they could not keep it in line with the sight of view. Also because the resolution of the HMD is limited to 800 × 600 and the quality of the overlaid graphics into the optics system is not very good, two participants felt nausea and motion sickness after a 10 min usage. However, even if these problems seem to restrict the use of HMDs, participants appreciated the level of immersion provided and thus preferred it. As far as the interaction techniques are concerned, the natural interaction techniques based on the marker cards were found to be very effective and intuitive to use compared to the other interaction techniques. Figure 18 illustrates a comparison based on the user-response between the most important interaction techniques implemented.

The I/O interaction techniques got the second highest score (mean = 6.2, SD = 2.16795, SE = 0.96954) since they are the standard way for interacting with computers and the end-users feel more familiar with. Surprisingly, the SpaceMouse interactions (mean = 5.4, SD = 2.50998, SE = 1.1225) received the most variable responses. Some participants argued that it is extremely useful to manipulate the virtual information using only one hand but others recorded that a lot of time is required to fully familiarise with the device and even then it is not as easy to use other means such as the I/O devices and the marker cards.

Moreover, the GUI interactions (mean = 4.4, SD = 2.19089, SE = 0.9798) got the worst score from all other types of interaction. One of the end-users argued that it is difficult to understand how to alter the orientation of the virtual objects since it was specified as yaw, pitch and roll. Other users stated that it takes the most time to perform a single rotation compared to the rest of the methods. For example, the keyboard keys replicate the functionality and as soon as the user becomes familiar with the "shortcuts" it is much faster. On the contrary, the marker cards interaction (mean = 7.8, SD = 2.58844, SE = 1.15758) received the most positive feedback of all other types of interaction and although it was pretty much expected, an initial comparison between different techniques has been made. All participants agreed that it very easy and intuitive to manipulate the virtual information in 3D space using any type of physical interface but they also proposed to use in the future a physical interface that consists of a handle. Overall, the preliminary evaluation was a profitable experience to complete the first cycle of this research but more user-studies need to be performed in the future.

9 Conclusions and future work

In this paper the design and implementation of effective AR interfaces for indoor-environments was presented and analysed. The proposed framework can be used as a generic tool to create high-level AR applications. The final visualisation can be performed either on a variety of display technologies ranging from monitor based to video see-through display technologies. A series of visualisation and interaction techniques were investigated in order to create the illusion that the virtual information coexists with the real world. In addition, two innovative AR case studies have been implemented: one for higher education purposes (university environments) and the second for archaeological and cultural heritage purposes (museum environments). Finally, an initial evaluation was performed to obtain useful critique concerning the overall technicalities and practicalities of the system.

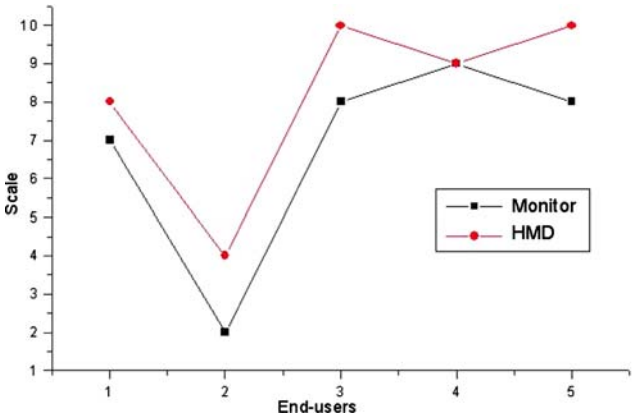

Fig. 17 Monitor versus HMD user response

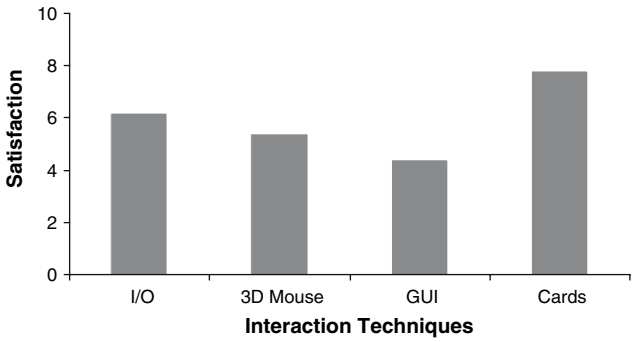

Fig. 18 Interaction techniques user response

The main advantages of the AR architecture are the low cost and the multimedia augmentation in real-time. The structure of the architectures is based on the philosophy that the most appropriate tool/device must be used for the task ones seeking to achieve. This, however, does not imply that the best tool/device is the most expensive one. The two different experimental setups successfully tested for this research clearly demonstrate this. One cost effective setup has been constructed comprising of off-the-self hardware and a second one based on state of the art expensive hardware components (i.e. Spacemouse, Touch Screen).

Although the system is designed for indoor environments it can be easily extended to operate in outdoor environments. The current status of the research is focused in various mobile devices such as personal digital assistants (PDAs) and third-generation (3G) phones as well as positioning technologies (such as GPS). This will create a robust mobile AR environment that will be integrated with the rest of the interface framework to provide prototype applications for outdoor environments.

Acknowledgments Part of this research work was funded by the EU IST Framework V programme, Key Action III- Multimedia Content and Tools, Augmented Representation of Cultural Objects (ARCO) project IST-2000-28366.

References

Angel E (2003) Interactive computer graphics: a top-down approach using OpenGL,3rd edn. Addison–Wesley, Reading, pp 17–18, 69, 107, 322–349, 472

Azuma R (1997) A survey of augmented reality. Teleoper Virtual Environ 6(4):355–385

Azuma R, Baillot Y et al (2001) Recent advances in augmented reality. IEEE Comput Graph November/December 21(6):34–47

Begault DR (1994) 3D Sound for virtual reality and multimedia, Academic, New York, 1, 17–18

Billinghurst M, Kato H, Poupyrev I (2001) The magicbook: a traditional AR interface. Comput Graph 25:745–753

Butz A, Höllerer T et al (1999) Enveloping users and computers in a collaborative 3D augmented reality. In: Proceedings of the 2nd IEEE and ACM international workshop on augmented reality '99. San Francisco, October 20–21

Camera calibration toolbox for Matlab, available at: [http://www.vision.caltech.edu/bouguetj/calib_doc/], Accessed at 14/01/2003

Dobler D, Haller M, Stampfl P (2002) ASR—augmented sound reality, ACM SIGGRAPH 2002 conference abstracts and applications, San Antonio, p 148

Feiner S, MacIntyre B et al (1993) Windows on the world: 2D Windows for 3D augmented reality. In: Proceedings of the ACM symposium on user interface software and technology, Atlanta, November 3–5, Association for Computing Machinery, pp 145–155

Fernandes B, Miranda JC (2003) Learning how computer works with augmented reality. In: Proceedings of the 2nd international conference on multimedia and information and communication technologies in education, Badajoz, December 3–6

Fuhrmann A, Schmalstieg D (1999) Concept and implementation of a collaborative workspace for augmented reality, GRAPHICS '99, 18(3)

Gatermann H (2000) From VRML to augmented reality via panorama-integration and EAI-Java, in constructing the digital space. In: Proceeding of the SiGraDi, September, 254–256

Grasset R, Gascuel J-D (2002) MARE: multiuser augmented reality environment on table setup. ACM SIGGRAPH conference abstracts and applications

Hall T, Ciolfi L et al (2001) The visitor as virtual archaeologist: using mixed reality technology to enhance education and social interaction in the museum. In: Spencer S (ed) Proceedings of the virtual reality, archaeology, and cultural heritage (VAST 2001), New York, ACM SIGGRAPH, Glyfada, Nr Athens, November, pp 91–96

Haller M, Hartmann W et al (2002) Combining ARToolKit with scene graph libraries. In: Proceedings of the first IEEE international augmented reality toolkit workshop, Darmstadt, Germany, 29 September

Haniff D, Baber C, Edmondson W (2000) Categorizing augmented reality systems. J Three Dimens Images 14(4):105–109

Kato H, Billinghurst M, et al (2000a) Virtual object manipulation on a table-top AR environment. In: Proceedings of the international symposium on augmented reality 2000, Munich,5–6 Oct, pp111–119

Kato H, Billinghurst M, Poupyrev I (2000b) ARToolkit user manual, version 2.33, Human Interface Lab, University of Washington

Klinker G, Ahlers KH et al (1997) Confluence of computer vision and interactive graphics for augmented reality, PRESENCE: teleoperations and virtual environments. special issue on augmented reality, August 6(4):433–451

Liarokapis F, White M, Lister PF (2004a) Augmented reality interface toolkit. In: Proceedings of the international symposium on augmented and virtual reality, London, pp 761–767

Liarokapis F, Sylaiou S, et al (2004b) An interactive visualisation interface for virtual museum. In: Proceedings of the 5th international symposium on virtual reality, Archaeology-Cultural Heritage, pp 47–56

Liarokapis F (2005) Augmented reality interfaces—architectures for visualising and interacting with virtual information. PhD thesis. University of Sussex, Falmer

Liarokapis, Petridis P, Lister PF, White M (2002) Multimedia augmented reality interface for E-learning (MARIE). World TransEng Technol Educ 1(2):173–176

MacIntyre B, Gandy M, Dow S, Bolter JD (2005) DART: a toolkit for rapid design exploration of augmented reality experiences. ACM Trans Graph (TOG), 24(3):932

Mahoney D (1999b) Better than real, computer graphics world, pp 32–40

Malbezin P, Piekarski W and Thomas B (2002) Measuring ARToolKit accuracy in long distance tracking experiments. In: Proceedings of the 1st international augmented reality toolkit workshop, Germany, Darmstadt, September 29

Milgram P, Colquhoun H (1999) A Taxonomy of real and virtual world display integration, mixed reality merging real and virtual worlds. Ohta Y, Tamura H (eds) Ohmsha Ltd, Chapter 1, pp 5–30

Milgram P, Kishino F (1994) A taxonomy of mixed reality visual displays, IEICE Trans Inf Syst E77-D(12):1321–1329
Moller T (1999) Real-time rendering. AK Peters Ltd, Natick, 23–38, 171
Poupyrev I, Tan D et al (2002) Developing a generic augmented reality interface. Computer 35(3):44–50
Reitmayr G, Schmalstieg D (2001) A wearable 3D augmented reality workspace. In: Proceedings of the 5th international symposium on wearable computers, October 8–9
Rekimoto J, Nagao K (1995) The world through the computer: computer augmented interaction with real world environments. In: Myers BA (ed) Proceedings of UIST '95. ACM, Pennsylvania, pp 29–36
Shi J, Tomasi C (1994) Good features to track, IEEE conference on computer vision and pattern recognition, Seattle, June, pp 593–600
Sinclair P, Martinez K (2001) Adaptive hypermedia in augmented reality. In: Proceedings of the third workshop on adaptive hypertext and hypermedia at the twelfth ACM conference on hypertext and hypermedia, Denmark, August 2001, pp217–219
Slay H, Phillips M et al (2001) Interaction modes for augmented reality visualization, Australian symposium on information visualization, Sydney, December
Smith GC (1994) The art of interaction. In: MacDonald L, Vince J (eds) Interacting with virtual environments. Wiley, New York,pp 79–94
Tory M, Möller T (2005) Evaluating visualizations: do expert reviews work? IEEE Comput Graph Appl 25(5):8–11
Vallino J (1998) Interactive augmented reality. PhD thesis, Department of Computer Science, University of Rochester, pp 1–25
Weng J, Cohen P, Herniou M (1992) Camera calibration with distortion models and accuracy evaluation, IEEE transactions on pattern analysis and machine intelligence, 14(10)
Woo M, Neider J, Davis T (1999) OpenGL programming guide: the official guide to learning OpenGL, Version 1.2, Addison–Wesley, Reading
Yewdall D (1999) Practical art of motion picture sound. Focal Press, Boston