SIACG 2002 - 1st Ibero-American Symposium on Computer Graphics
1-5 July 2002, Guimarães - Portugal

MIXDesign, Tangible Mixed Reality for Architectural Design

José Miguel Salles Dias
Miguel.Dias@iscte.pt

Pedro Santos
Pedro.Santos@iscte.pt

Rafael Bastos
Rafael.Bastos@iscte.pt

Luis Monteiro
Luis.Monteiro@iscte.pt

Rui Silvestre
Rui.Silvestre@iscte.pt

Nancy Diniz
Nancy.Diniz@iscte.pt

ADETTI/ISCTE, Associação para o Desenvolvimento das Telecomunicações e Técnicas de Informática, Edifício ISCTE, 1600-082 Lisboa, Portugal, www.adetti.iscte.pt

Abstract

MIXDesign, provides a tangible Mixed-Reality system oriented towards tasks in Architectural Design, in several usage scenarios, such as Conceptual Design, Client Brief or even Architectural Design education. With MIXDesign, an architect can intuitively interact with a real scale model of the design, in normal working settings, where he can observe an enhanced version of the scale model, with 3D virtual objects registered to the real ones. The architect is then able to use intuitive tangible interfaces, such as a paddle, to choose menu options, select a 3D virtual object, transport a virtual object within the scale model surroundings and geometrically transform an object (by rotation or scaling). MIXDesign provides a platform for testing new design concepts while seamlessly transporting the Architect from Reality (RE) to Augmented Reality (AR) and then through Augmented Virtuality (AV), towards a full Virtual Environment (VE), and back, where he can perceive and judge both the virtual and the real models, interactively and in real-time.

Keywords

Mixed Reality, Augmented Reality, Tangible Interfaces, Architectural Design

1. INTRODUCTION

The adoption of 3D CAD tools in the processes of conceptual design and experimentation of urban and architectonic forms and spaces, is still not a commonly used methodology. Traditionally, the three dimensional virtual space has been almost only explored as a method of visualization and representation of the architectural project, and rarely used as a tool for conceiving and testing the architectural design. This is due, amongst other reasons, to the technical difficulty of working and interacting with these type of 3D CAD systems, and also to a certain "2D culture" in the traditional methodology of architectural production. Lately, this has been changing with the rapid evolution and spreading of the 3D design CAD systems as well as the adoption, under way, of 3D-based product model standards for this sector, such as the Industrial Foundations Classes, IFC V2.x [IAI96], now in the process of becaming an ISO standard. Creating a technological environment where people could experience and test an architectural project, as fully and as closely to the reality as possible, through an easy and intuitive way, is something very rewarding and revolutionary for architects and also for future users of the projected spaces. Through this interface, the architect would overlay the 3D-based virtual project onto a real architectural and/or urban context, and in real time. In the framework of this inter-relation of environments (where he could switch from the real environment, to an augmented reality environment and, subsquently, to a full virtual environment -> mixed reality interaction), he would also be able to use modelling and geometrical transformation tools, over 3D entities, allowing him to transform the project, in the real context and in real time. This would give him a very close perception of how the future project would correspond to the urban context and would enable the vivid experimentation of the space. This interface would also support multiple collaborators, where discussion meetings could take place amongst architects and clients, all together immersed on the different environments: real, augmented or virtual.

These are the chalanges of MIXDesign: to provide an environment where an architect can semalessly evolve through a real architectural space, into an augmented one, where virtual objects correctely superimposed onto the real ones, enhancing the perception of the reality and providing a framework for conceptualising design alternatives with an intuitive interface. Upon user request, this

environemt can be turned into a totaly virtual reality one, thus providing a truely mixed reality experience.

The MixDesign system lies in ArToolkit [Kato99] [Kato2001a], a C and Open GL-based publicly available library that uses accurate vision based tracking methods to determine the virtual cameras' viewpoint information through the detection of tracking fiducial markers. First the live video image is turned into a binary image based on a lighting threshold value. This image is then searched for square regions. ARToolKit finds all the squares in the binary image, many of which are not the tracking markers. For each square, the pattern inside the square is captured and matched against some pre-trained pattern templates. If there is a match, then ARToolKit has found one of the AR tracking fiducial markers. ARToolKit then uses the known square size and pattern orientation to calculate the position of the real video camera relative to the physical marker. In fact, upon detection of a specific marker, ArToolkit provides the programmer with a transformation matrix, that translates and re-orientes the local coordinate system associated to the marker, to the virtual camera coordinate system. Since the virtual and real camera coordinates are the same, the programmer can precisely overlay the image of a virtual object onto the real world, using OpenGL, resulting in a Augmented Reality effect. This virtual object can be exactly overlaid on top of the marker or in any position relative to a local coordinate system attached to it.

The paper is organised as follows: in section 2, we provide a background and state-of-the-art in the issues of Augmented Reality, Mixed Reality and Tangible Interfaces. In section 3, we present our system architecture, comprising a number of different modules. Section 4, covers the problems and developed solutions, concerning the smooth registration of virtual and real objects. Section 5 addresses the system component that manages tangible interfaces, a crucial aspect of MixDesign, since it deals with user interaction. Section 6 details the hardware and software platforms and explains our methodology and achieved results of system testing. Finally, in section 7, conclusions are drawn and future directions of research are given.

2. BACKBROUND: AUGMENTED REALITY, MIXED REALITY AND TANGIBLE INTERFACES

Augmented Reality (or AR) systems and technologies were introduced in 1992 by Caudel and Mizell [Caudell92], in the context of a pilot project, where they were used to simplify an industrial manufacturing process in a Boeing airplane factory. In general these systems provide the means for "intuitive information presentation, which enhances the perceiver's situational awareness and cognitive perception of the real world" [Behringer98]. This enhancement is achieved by placing virtual objects or information cues into the real world, which is made possible by performing "virtual camera calibration", that is, by computing the virtual camera parameters that mach the position and orientation of the observer of the real scene. With this technique, "virtual" objects can then be registered in relation to "real" objects, which means that these objects can be seen in the same position and orientation of other "real" objects of the scene, as perceived by the user. This is usually done using optical or video see-through head mounted displays and tracking devices, linked to, either standalone computers with 3D graphics capabilities, or mobile wearable computers. Video see-through AR is where virtual images are overlaid on live video of the real world. A possible alternative is optical see-through AR, where computer graphics are overlaid directly on a view of the real world. Optical see-through augmented reality has more complex camera calibration and registration requirements [Kato99].

According to a more broad definition by Didier Stricker [Didier99], in AR, "a user's view of the real world is augmented with additional information from a computer model. With mobile, wearable computers, users can access information without having to leave their work place. They can manipulate and examine real objects and simultaneously receive additional information about them or the task at hand". To synthesise, Azuma [Azuma95] argues that AR "1) combines real and virtual environments; 2) is interactive in real-time; 3) is registered in 3D". We can conclude that AR is a challenging multidisciplinary field and a particularly promising new user interface paradigm, integrating computer graphics, computer vision, positional and orientation tracking technologies, wearable

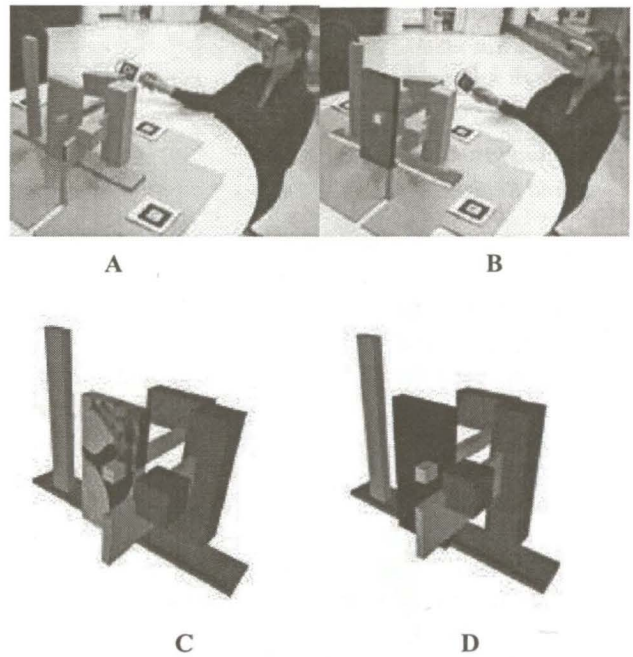


Fig. 1 The Mixed Reality paradigm:
A – Real Environment, RE
B – Augmented Reality, AR
C – Augmented Virtuality, AV
D – Virtual Environment, VE

computer architectures, psychology, artificial intelligence, ergonomics and others.

In view of this multidisciplinary and integration framework and associated complexity, Milgram and Herman [Milgran99], propose a new taxonomy that supplement the definition of Augmented Reality, by introducing the concepts of Augmented Virtuality (AV) and Mixed Reality (MR). They argue that Real Environments (RE) and Virtual Environments (VE) are, in fact, two poles of the Reality-Virtuality Continuum, being RE the left pole (see Fig. 1-A) and VE, the right pole (see Fig. 1-B). REs include sample data acquired and digitised from real scenarios, such as image, video, ultra-sound, X-ray, MRI-"Magnetic Resonance Imaging", CAT-"Computer-Aided Tomography, laser range and light intensity data and others, whereas VRs, refer to computer models of scenarios that

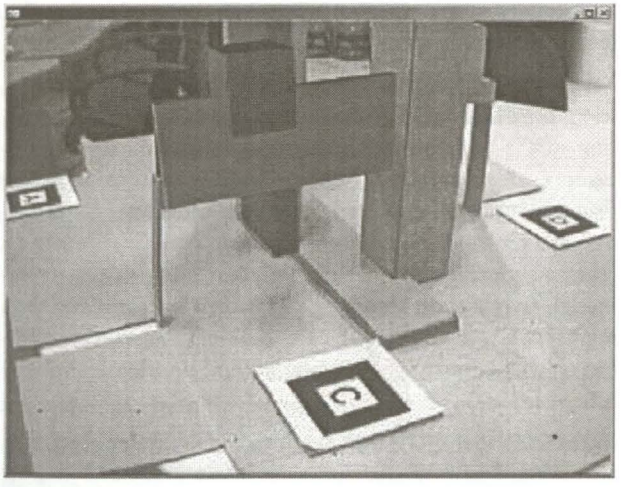


Fig. 2 Augmenting the real scale model with "registered" virtual models

can be rendered. If we browse this continuum from, say, the RE part towards the centre, we begin to enhance the RE with virtual objects registered to the real world (hence the concept of AR) and, if we go even further right towards the VE pole, past the centre, we immediately notice that the virtuality of the modelled environment, can be enhanced by sample data taken from real scenarios, such as, for example, digital images or videos, textured mapped onto 3D geometrical objects (and thus, we end up interacting with an Augmented Virtuality environment, see Fig. 1 C).

Mixed Reality is thus the overall framework that includes the continuum transitions from RE, to AR, passing through AV and towards VE, but excludes the end-points [Milgran99], perceived as limit conditions.

So far we've addressed the "visual augmentation" side of things, whereas the user interaction within a Mixed Environment, in the broad sense, deserves also special attention. Since the user is immersed in this type of environment by means of, for example, a video see-through head-mounted display, standard input devices such as keyboards or mice are useless, since they distract the user from the task at hand, thus creating a severe cognitive seam, within the interaction process. On the other hand, traditional input devices used in Virtual Environments, such as the data glove or a 3D mouse with 6 degrees of freedom, introduce undesirable complexity in the tracking technology, or add "strange" gadgets, to the user's workspace, with whom he's not daily accustomed. To face this problem, Kato [Kato2001b] proposes Tangible Interfaces, as a new approach for the design of Augmented Reality interactions. According to the author, Tangible Interfaces, "are those in which 1) each virtual object is registered to a (tangible) physical object; and 2) the user interacts with virtual objects by manipulating the corresponding tangible object". Tangible interfaces are described in the literature as intuitive, since physical object manipulations are mapped to virtual object operations. If the mapping is one-to-one, we classify the interface as a Space-Multiplexed one. A Time-Multiplexed interface is the one where a physical object may correspond to different functions, at different points in time.

MIXDesign provides a truly Tangible Time-Multiplexed Mixed-Reality system oriented towards tasks in Architectural Design, in several applications, such as Conceptual Design, Client-Brief [Dias2002], or even Architectural Design education, although it can also be applied to a number of other target sectors, such as automotive or aerospace design. With MIXDesign, an architect can intuitively interact with a real scale model of the design, in his usual working table, where he can observe an enhanced version of the scale model, with 3D virtual objects, registered to the real ones. The architect is then able to use intuitive tangible interfaces, such as a paddle, to choose menu options, select an object, transport an object within the scale model surroundings, geometrically transform an object (by rotation or scaling), thus testing new design options and seamlessly transport himself from AR to an AV or VE scenario, where he can judge the virtual model, interactively and in real-time.

3. MIXDESIGN SYSTEM ARCHITECTURE

Assuming the availability at program start-up, of a virtual version of, say, a scale model, in a 3D file format (ISO VRML 97), MIXDesign is able to display the virtual model in the same position and orientation of the real one, as

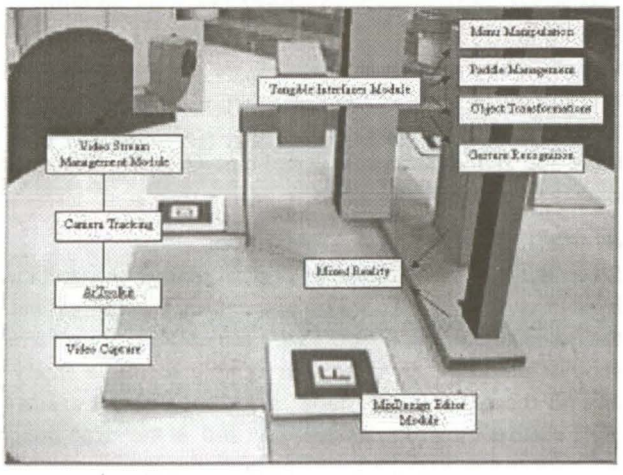


Fig. 3 MixDesign System Architecture

long as the video camera is "seeing" a fiducial marker (Fig 2).

By using video see-through glasses, the user is able to notice in real-time, the registration of the virtual and real models, which becomes the basis of the entire system. Once this is achieved, the architect can begin his work, interacting through tangible interfaces with the virtual objects registered on the scale model, importing new virtual objects from a library into this augmented workplace, or even adding new real objects. These imported objects can be primitive shapes like cubes, cones, spheres, parallelepipeds, etc., or complex VRML objects. Importing objects is achieved through a simple tangible panel that is made up of several markers, each one associated with a virtual object (Fig. 5). Animation is also supported as long as it's properly specified in the VRML description. For example, it's perfectly acceptable to have a full 3D model of a building with its elevators animating up and down across the floors, upon user interactive request, and with sliding doors opening and closing on the ground floor, as the viewer approaches them.

Object editing and modification are fully supported in MixDesign, through a tangible user interaction. Objects can be:

- selected and made visible/invisible;

- rotated, with three degrees of freedom;

- scaled, along any of the three main axes;

- and moved indistinctively to a particular position in augmented workplace.

The user is also able to switch back and forth, from augmented reality to full virtual reality and vice-versa, thus gaining a perception of its architectural design changes, in the framework of mixed reality.

To deploy this new design paradigm, a number of modules comprise the MixDesign system (Fig 3). The MixDesign **Editor** is responsible for associating 3D objects to markers, so that virtual shapes will be registered and superimposed over recognized 2D fiducial patterns. The **Video Stream Management** Module, includes a lower layer of **Video Capture**, which feeds into **ArToolkit**, allowing it to perform Virtual **Camera Tracking**. In order to interact with **Tangible Interfaces**, a dedicated system module was developed, which include **Menu Manipulation** (selection and activation), **Paddle Management** routines, **Object Transformations** (movement, scale and rotation) and modifications and real-time **Gesture Recognition**.

4. VIRTUAL CAMERA TRACKING

Since the system is based in fiducial marker recognition by means of computer vision techniques, it requires that the marker is in fact visible to the camera. So, the system should guarantee that the user is able to move freely around the scale model, look at it through arbitrary angles, explore it by any desired view and, at the same time, run properly as expected, i.e., maintaining accurate registration of virtual to real objects, as if at least one marker would always be visible to the video camera. However, this is not always true. As the user is moving around the scale model, the system will inevitably lose track of the marker, which will eventually be, at some point, blocked out of the camera's field of view, either because there's an obstacle occluding it or because it's actually out of the camera's view. As a result, as long as the system cannot recognize any visible markers, it will not be able to track down the exact video camera's position and display the registered virtual objects.

One of the simplest approaches to this problem is to have several markers laid out along the scale model. Ideally, there should be enough markers placed around the model, so that the video camera is always able to "see" one of them. We have chosen to use four different markers (with four letters of the alphabet printed on them) placed in the sides of the scale model.

. For each marker, the system defines a specific resultant transformation matrix, that corresponds to the composition of the transformation matrix from the marker to the camera reference frames, with an additional translation (no orientation, in our case), of the virtual model in relation to the marker reference frame. Once a specific marker is recognized, the system needs to draw the virtual scale model, overlaid on top of the real one. This means that marker "A" will display the same object that marker "F", but on different relative positions (and on identical absolute positions).

In the case that more than one marker is visible from the video camera attached to the user, we just need to find the nearest marker to camera, and use that one to register and display the virtual world superimposed onto the real environment. This is done by calculating the length of the vectors from each marker reference frame, to the camera viewpoint and determining the smallest one. The reason for using the nearest marker has to do with the inherent behaviour of the pattern recognition software layer [Kato2001b].

The system assumes that there is always pattern (marker) distortion relative to the camera position and orientation, and, thus believes that there are some unavoidable errors. The video camera unsteady position and motion and the slight variations of lighting conditions, means that the computed position and orientation of the virtual camera, relatively to the marker's reference frame, will never be exactly and precisely matching the real camera position and orientation, relatively to the real marker, even if the video camera is not moving. These slight variations mean that, as we take the computed position and orientation of the virtual camera relative to a marker's position, compose with the translation matrix (parameterised with the necessary distance offsets) and draw the virtual world a number of times per second, it will appear as if it was trembling. Because of these errors, the farthest away we draw the virtual scale model relatively to the marker, the more will it appear to tremble. These small errors are, in fact, consid-

ered as high-frequency noise. As such, the obvious solution is to apply a low-pass filter to the signal, cutting out all the undesired noise. This is done by keeping a predetermined number of computed transformation matrixes that transform the virtual camera frame into the visible marker's reference frame) and averaging them. As a result the augmented reality displayed by the marker, will appear more steady and still to the user.

5. TANGIBLE INTERFACES

We have used a broader definition of tangible interfaces, in the sense that virtual objects are registered to real ones, such as a paddle or square pieces of cards with markers, as in [Kato2001b] or [Billinghurst2001], but they are also physical means of interaction between the user and the system that will trigger the functionalities of the platform, by the way of gesturing.

The MixDesign system requires real-time user interaction. In this framework, we have developed tangible interfaces as intuitive as possible and we have come up with a tool, similar to the ones found in [Kato2001b] or [Billinghurst2001], which are suitable for our tangible interfaces requirements. This tool (in several versions) is a specific marker with the shape of a paddle. As another visual aid, when it is recognized by the system, a virtual paddle will be displayed on top of it (Fig 4). The paddle is used in several interaction tasks, such as object picking, moving and dropping, scaling, rotation, menu manipulation, panel use, gesture recognition, and for all sorts of commands given to the system.

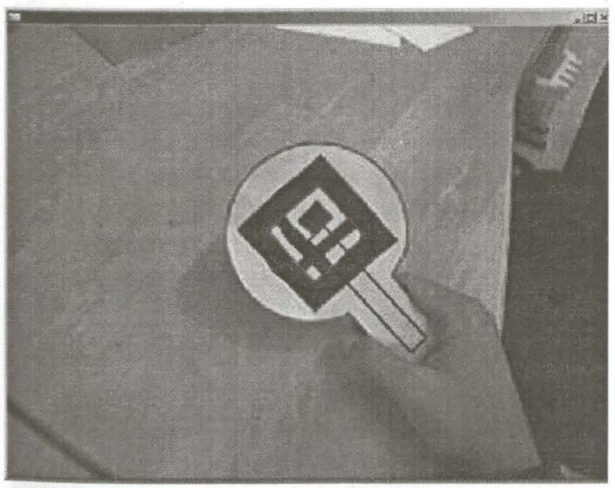


Fig 4 A virtual paddle registered on top of a real one.

5.1 Associating 3D Objects to Markers

The association of different 3D virtual objects to respective markers, for real-time registration, is performed by an auxiliary tool, which was developed, the "MixDesign Editor". In order to correctly use it, all markers should be ready for use. This can be achieved, by using the "Make Pattern" utility of the tool, which will produce a file with the necessary data, so that AR Toolkit will be able to later recognize the pattern inscribed in the marker. There are two types of virtual objects that can be associated with different patterns: Basic Geometric Primitives and VRML 97 Objects/Worlds (we have used the "Open VRML" library, that also supports object animation). Basic Primitives comprise different GLUT Primitives [Kilgard96], in Gouraud shaded or wireframe versions, such as: Cube, Utah Teapot, Dodecahedron, Octahedron, Tetrahedron, Icosahedron, Sphere, Cone and Torus. Within the MixDesign Editor, it's also possible to define and control different object properties, such as Colour, Translation, Rotation and Scale. Additionally, there are also specific properties for certain kind of objects, such as: radius for a Sphere, radius and height for a Cone, etc. After this stage of object properties specification, the user is able to associate the object with the corresponding marker. This procedure can be repeated for any desired object. The tool will build a configuration file that will be loaded and interpreted by the MixDesign platform.

5.2 Gesture Recognition

One of the basic tangible interfaces supported by MixDesign, is based in hand gesture recognition by computer vision means. Using a paddle (Fig. 4), the user is able to make certain spatial gestures that are known and can be recognised by the system, and that will then trigger certain programmed actions. Representations of these gestures are first loaded into the system at program start-up, so that MixDesign will be able to recognize them, while being created by the user, in run-time. We have used simple bitmap monochromatic images (black and white), as gesture representations, and have stored them in files. These images have the same spatial resolution than the system video capture stream. In black, we represent the background, while, in white, we code the actual interaction path that can be recognised. Some examples are shown in Fig 6. Once loaded, the gesture bitmaps are stored in matrixes filled with "zeroes" and "ones", corresponding these last values, to the white pixels of the original image. The system is able to verify if a gesture made by the user, is one that it can recognize. This proce-

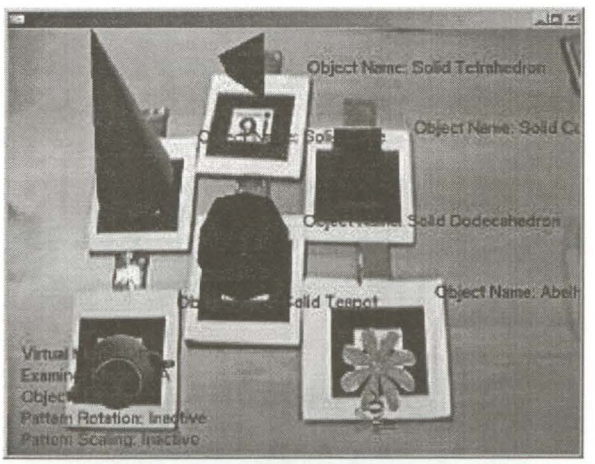
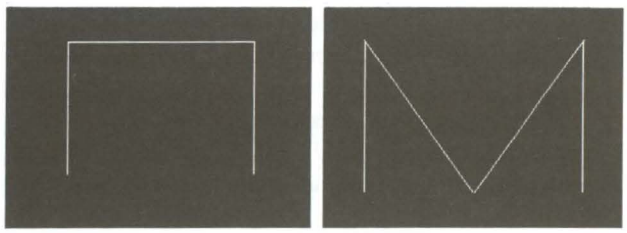


Fig. 5 A Panel with virtual objects associated and registered with real markers



A B

Fig. 6 Bitmap monochromatic images, as gesture representations

dure is as follows: initially, we store the paddle's position across a ß number of frames. This ß figure, should be large enough so that the user has enough time to create the complete gesture, and should be small enough so that user doesn't produce parts of the complete gesture during a too long period of time. The paddle's positions are stored in a matrix in memory (that has the same resolution than the stored bitmaps), for each pixel that the paddle moves through. So, if the centre of the paddle is in pixel $\mathbf{P}_1$ (213, 129), a value of "one", will be stored in our matrix at index 213,129. It should be noted that, as we store these positions, they should remain in the matrix only for ß frames. As the user moves the paddle, it's position in pixel $\mathbf{P}_1$ will be stored at frame f_1 but should be removed from the matrix in frame $f_1 + ß$. Similarly, the next paddle position stored at frame $f_2 = f_1 + 1$, corresponding to pixel $\mathbf{P}_2$, should be removed at $f_1 + ß + 1$. This means that we are continuously adding and removing paddle positions (pixels) to the gesture matrix. One of the simplest ways to implement this feature, is to store the value ß in the matrix's positions, instead of "one" that correspond to the white pixel of the original image. Then, on each frame, we loop through all the matrix's indexes and decrement the number stored in it (if bigger than zero). This way, each and every pixel will be contained in the gesture matrix only for ß frames. On each frame we will also analyse this matrix, and see if it matches one of the known recognizable gestures.

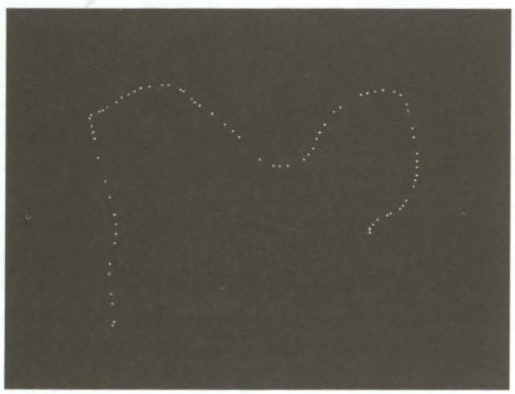
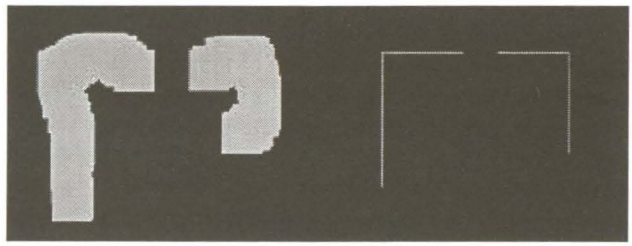


Fig 7. Path of a user gesture, while manipulating the paddle, as captured by the system

We will now make a gesture in front of the camera, along that predetermined number of frames, so that it can be recognised as a known path (Fig. 7). The image in Fig. 7, contains just the captured paddle's positions in a interval of time; obviously, the full sequence of images would also have the real environment as captured by the camera. As it can be seen, the capture bitmap comprises just some scattered points along the screen, that don't match the stored gesture image file. So, instead of just storing the paddle's position in the gesture matrix as a single pixel,



A B

Fig. 8 A larger portion of the user gesture (A) and the result of the logical multiplication with the coded gesture loaded from the library (B).

we must dilate it so that it covers a larger portion of the performed movement. This way, the gesture to be analysed will be smoother and more continuous (Fig 8 A). This is a suitable image for our purposes. Now, we multiply logically this matrix with the one loaded from the gesture library file, in order to see how much of the original movement did the user covered (Fig 8 B). If we subtract this image with the original one (Fig 6 A), pixel by pixel through all indexes, and count the pixels that are not zero (let's name it a), we will obtain the number of pixels that are not common on both images. As a consequence, the

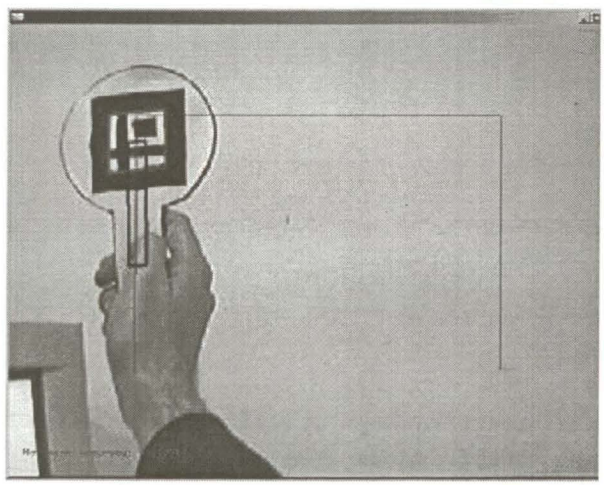


Fig. 9 Matching successfully a (library) coded gesture.

greater the value a, the more differences will exist between the movement the user performed, and the gesture he should have made. The smaller the a, the more will both images resemble each other. A perfect match will occur if a is zero. However, this is rather difficult for the user. Most of the times that he tries to make the gesture, there will be some differences. So, we will consider that a

gesture is recognized only if a is smaller than a certain confidence value. This value will be the error margin that the user is allowed to create. Typically, a suitable value for our system, after user experimentation, was 110 pixels of allowed differences.

We've also introduced an aid to the user. If he begins to perform a gesture that can be "expected", the partially recognised path will be drawn on the screen in real-time, so that he sees how he should continue the movement successfully. We also display, in the lower left of the screen, how good (in percentage) is the match between the movement of the user and the stored gesture. This is shown in Fig 9. After the gesture is recognized by the system any kind of action can be attached to it and performed. This introduces us a wide variety of possible operations. Literally hundreds of different movements can be programmed into the system, giving the user a highly flexible tangible interface.

5.3 Menu Manipulation

The **Gesture Recognition** feature enhances the system in such a way that we can now trigger, potentially, any number of events. A useful application of this feature, is the possibility to activate a menu where we can handle virtual object selection and visibility. After the specified gesture is identified, a menu containing a list with the names of every object (currently visible or not) in the virtual world will appear (Fig. 13). The "Exit" option is always available, allowing the user to leave Menu Mode and "concentrate" in the interactions in the Augmented Reality world. The interaction with the list of virtual objects menu, by use of a tangible interface, is straightforward: the user should lean the Paddle left, to select the object above, lean the Paddle right, to select the object below, and push the Paddle slightly up, to activate or deactivate the correspondent object visibility. If "Exit" is selected, pushing the Paddle slightly up, the menu will be disabled. Concerning visual awareness, the user is always informed about the object selected at the moment, and whether it is visible or not. The name in the list currently being dealt with is highlighted with a green box (Fig. 13). If the object is visible, the name will appear written in blue with a small orange dot behind it (Fig. 13); if it's not, the name will appear written in black (Fig. 13).

5.4 Moving Objects

A typical tangible interface task is to be able to transport a virtual object, literally in one's hands, and then dump it somewhere else in the "world". Again, the paddle will be rather suitable for this purpose. To illustrate this interaction, it will now be useful to define two more markers, marker "A", an empty one, and marker "B", that contains a registered object. In order to perform one of the two possible actions (picking or dropping), the user will have to hold the Paddle, place it within a specific range of an arbitrary marker and then lean it left or right, whether the Paddle is standing on the right or the left of the marker, respectively. The action performed is also dependent on the type of marker, A or B. So, according to the type of user interaction, the type of marker, and assuming that the Paddle is positioned to the right (as opposite to left) of that marker, we can have the following combinations of

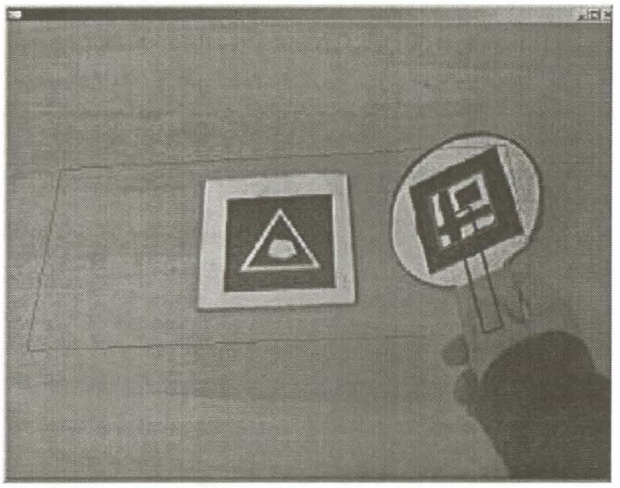


Fig. 10 A tangible interaction that will have no success: both the Paddle and the marker are empty of 3D objects

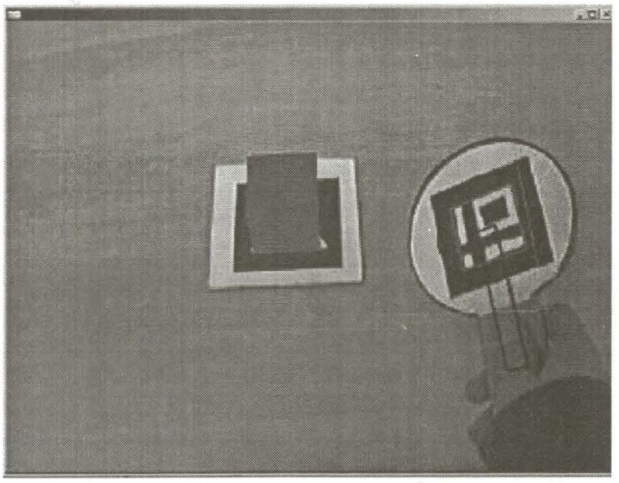


Fig. 11 A tangible inteeaction that will success: the Paddle is empty, but the marker has a registered 3D object.

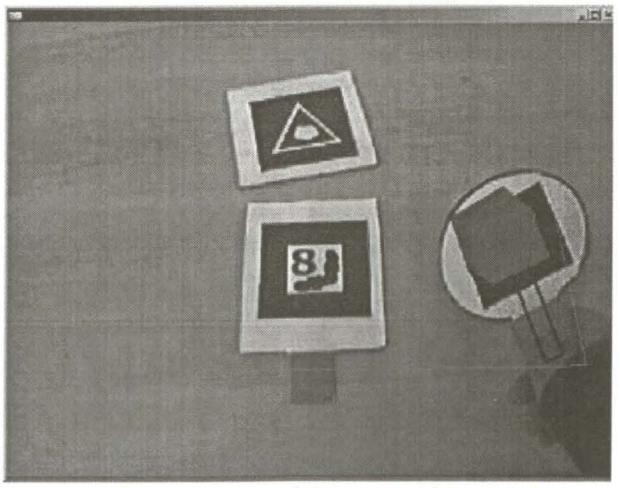


Fig. 12 Upon completion of the interaction, the object is transferred from the marker to the Paddle, in real time.

tangible user interactions:

- "Paddle + lean left + A" will perform a dropping action of the object, onto marker "B", if there's an object on the Paddle, and do nothing if the Paddle is empty;

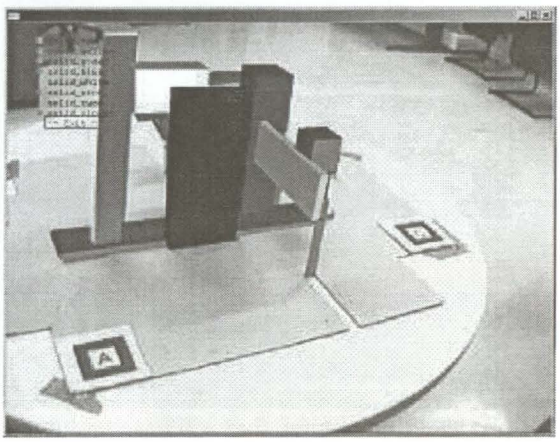


Fig. 13 Interacting with a menu list of objects, using a tangible interface

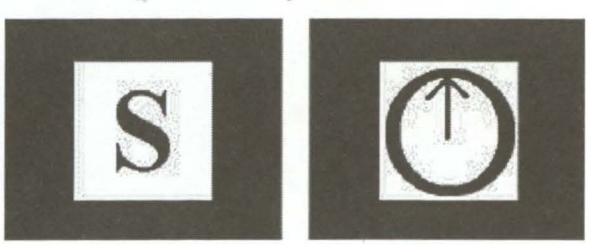
- "Paddle + lean left + B" will perform an object picking action if the Paddle is empty, and do nothing if the Paddle has an object.

These behaviours are summarized in Table 1.

Tangible Interaction Combination	Action
Paddle (empty) + lean left + A	Null
Paddle (empty) + lean left + B	Picking
Paddle (w/ object) + lean left + A	Dropping
Paddle (w/ object) + lean left + B	Null

Table 1

When trying to perform these actions, the user will have access to visual feedback about whether he is acting correctly or not, i.e., if the action is valid. When the Paddle approaches any marker, a box representing the valid area of interaction will appear. If both the Paddle and the marker are empty, or already have an item, the box will appear with a red colour (Fig. 10), meaning that it is impossible to drop or pick an object. On the other hand, a green box will be shown if any of the actions available are



A B

Fig. 14 Markers for Scaling (A) and Rotation (B) Paddles

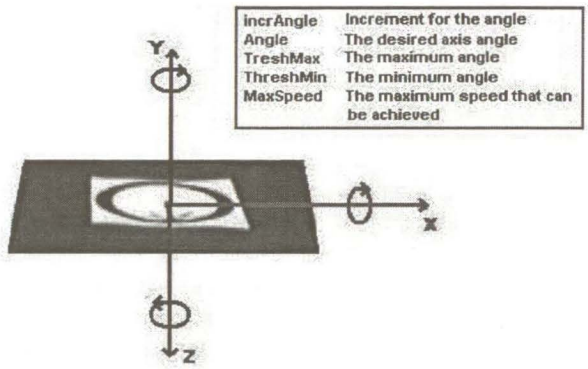
possible (Fig. 11). When completing a combination that will trigger a valid action, an animation will show the object sliding to or from the Paddle (Fig. 12).

5.5 Rotating and Scaling Objects

We have defined two different types of markers for the object rotation (Fig. 14 A) and scaling features (Fig. 14 B). When one or both of these markers are vis ible within the visual field of the video camera, the respective functions are activated, modifying the selected virtual object.

Rotating Objects

The tangible interface for object rotation is based on the position and orientation of the "Rotation Paddle". The paddle has a defined orientation that was set previously by the "Make Pattern" utility. This orientation is required because the camera interface needs to have a reference to




```
incrAngle = Angle < TreshMax && Angle > TreshMin ?
            (MaxSpeed/Abs(ThreshMax-ThreshMin))*(Angle + ThreshMax)
            :
            Angle > -ThreshMax && Angle < -ThreshMin ?
            (MaxSpeed/Abs(ThreshMax-ThreshMin))*(Angle - ThreshMax)
            :
            0
```

Fig. 15 Algorithm to compute the angle of rotation relativey to a principal axis

establish terms of comparison. The MixDesign platform, obtains from the lower layer AR Toolkit, the three angles of the "Rotation paddle" reference frame, relatively to the main axes (*X, Y, Z*), and if one of these angles is larger than an established limiting value, the object will start to rotate on the desired angle (Fig. 15). We have also developed a way to increase or decrease the rotation speed, which is determined by a linear dependence with the derivative of the angle: if the angle is increasing, the rotation speed will increase and, conversely, if the angle is decreasing, the rotation speed will decrease. It is only possible to rotate the object relatively to an axis at a time, and

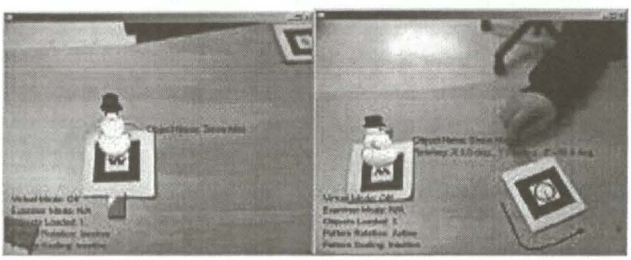


Fig. 16 Rotating a virtual object in Z axis

there are two ways of stopping the current rotation. One, is by placing the "Rotation paddle" in a position where all rotation angles are null, and the other, is by just hiding the paddle from the video camera field of view.

Scaling Objects

Similarly to the rotation case, we have developed a tangible interface for object scaling, which is based on the position and orientation of the "Scaling Paddle". With our scheme, it is possible to scale the virtual objects, along the three main coordinate axes. The "Scale Paddle" also needs a defined orientation to work conveniently. The operating method is quite similar to the one described for the "Rotation Paddle". If the angles retrieved by the "Mix Design Platform" are bigger than an established threshold, the object will be scaled (shrunk or grown) along the

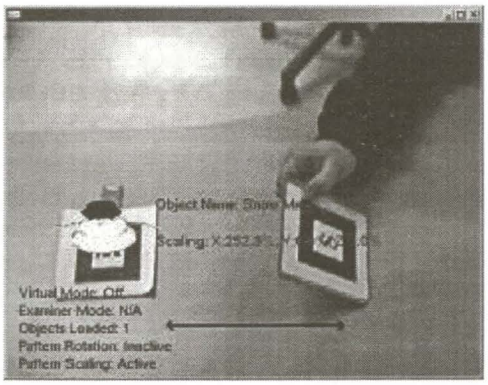


Fig. 17 Scaling a virtual object in the X axis

desired axis. To determine the scaling speed, we have used the same algorithm as described above for the Rotation functionality.

5.6 Virtual Reality Portal

To transport the user from Augmented Reality to the Augmented Virtuality or to a Virtual Environment, we have defined a special gesture to be produced with the paddle. If the "Mix Design Platform" captures this ges-

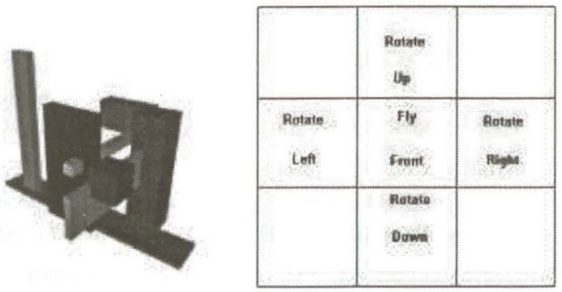


Fig. 18 Encoded gestures to control virtual camera navigation in VR

ture, all the "reality" will disappear from the current scene, and "virtuality" will pop-up (Fig. 18). The augmented reality will now become a virtual scenario, where only virtual objects and virtual backgrounds are visible. The user is able to interactively navigate through this virtual world using, now a "Navigation Paddle". To better illustrate the navigation method, we refer to the grid of Fig 18, where we encode the gestures required for the user to control the virtual camera, using a "fly through/navigation" metaphor. By positioning the Navigation Paddle in one of these five spatial areas, the system allows the user to rotate in two axes, centred in the viewpoint, and also to fly in his front direction. Again, if the user issues a pre-programmed gesture with the visible paddle, Augmented Reality will be again activated.

6. HARDWARE AND SOFTWARE PLATFORMS AND SYSTEM TESTING

The hardware and software platform used in MIXDesign, are as follows:

Hardware:

- CPU: HP Pentium III 1GHZ, RAM: 256 Mbyte

- Graphics card: NVIDIA GFORCE2 32 Mbyte

- Video camera - 2 options were used: MicroCam wireless ultra miniature camera from Swann Security; and Web-Cam 5 from Creative Labs

- Video see-through head mounted display: Olympus Eye-track SMD 700, Multimedia Glasses, which support 800x600 pixel resolution

Software:

- MS Visual C++ 6.0 enterprise edition

- ArToolkit 2.5.2

- Video Input: Direct X 8.1

- Graphical output: Open GL

Indoor Tracking method used: Computer vision-based approach as provided by ArToolkit with sparsely placed fiducial markers.

The system was extensivley used by one of the authors (Nancy Diniz, an architect) and by an architect student, in normal working conditions. In the test example, a real scale model and a registered virtual version were available, within a mixed reality environment. The users were given an initial tutorial session of the system architecture and functionality, and where then asked to perform simple 3D design and visualisation tasks as in usual working settings. MixDesign was believed to have strong and weak features:

1. Strong Features: (1) "The system offers a high degree of experimentation of 3D architectural designs, in the context of an innovative technology";(2)"The system creates a cognitive dimension and a flavour of entertainment", when the user is "experiencing augmented reality and tangible interaction".
2. Weak Features: (1) On the issue of tangile interactivity, "the gesture that leads up to the menu avtivation", was perceived as "being dificult to handle and reconise"; (2) "The video image quality", was perceived as being "only satisfactory".

7. CONCLUSIONS AND FUTURE DIRECTIONS

This system presented in this paper opens a wide range of opportunities and new paradigms of study and conceptualisation in Architecture and Urban Design. Through the use of this technology, it will be easier to understand the main concepts of the 3D shape in study, in a more immersive, fulfilling and complete way. The authors believe that the possibility of overlaying (realistic) virtualities over realities, will act as a more rapid and efficient way to achieve design results, than the traditional "separated" use of real scale models and virtual reality models. Tangible interfaces, oriented for 3D modelling and editing tasks and rendering tools capable of real time performance, are believed to create a new dimension of user interaction for Architecural design. On the educational field, it will be interesting to create mixed reality platforms to study the work of some of the best architects. The authors believe that the system, wil also also introduce a new and exciting dimension to the Architectural learning experience. Adding other sensorial dimensions in the process, such as: touch feedback, 3D spatialised sound and odour, would certainly open new perspectives on the methodology of conceiving and experiencing Urban and Architectonical spaces. Several future directions can be envisaged for this kind of system, to enhance the new paradigm of tangible mixed reality interaction for architectural design, starting from establishing a complete methodology for user requirements capture, user trials and usability evaluation. Concerning the system architecture, improvements could be made in the mobility of the system, to increase the sense of mixed reality, that is, of seamless transition between real, augmented and virtual environments. In this respect, we plan to conceive an optimised and distributed software architecture, able to run on a network of wearable and base station computers. Hybrid indoor virtual camera tracking (with six degrees of freedom), could be a possible solution for the virtual-real registration problem, for example, enabling the user to walk freely within a certain indoor zone, while experiencing augmented or virtual worlds. In this new architecture, the wearable computer would equipped with:

- video-see through glasses, wireless video input
- high-performance 3D graphics board
- paddle (hand-held input device)
- audio input device
- enough battery power
- enough RAM and disc space
- mobile network interface
- tracking interface

The indoor virtual camera tracking solution to be used, could be based on a mixture of: (1) an array of ultrasound sensors positioned along the wokspace area; (2) computer vision approaches (ArToolkit), for short-range and long-range indoor tracking. The use of two cameras (stereo) and enhancing the filtering technique (using the Kalman filter), could also improve the accurate registration of virtual to real objects, by better control of the distortion errors.

8. ACKNOWLEDGEMENTS

The authors would like to thank Manuel Gamito for he´s valuable comments on Virtual Camera Tracking and Vitória Albuquerque for her willingness and dedication in carrying tasks related to the usability of the system.

9. REFERENCES

[Azuma95] Azuma, R. T., "A survey of augmented reality", *SIGGRAPH'95 Course Notes*, pp 1-38, 1995.

[Behringer99] Behringer, R., et al, editors, *Augmented Reality, Placing Artificial Objects in Real Scenes*, Proceedings of 1st International workshop on Augmented Reality - 1998, A. K. Peters ed., pp xi-xx, 1999.

[Billinghurst2001] Billinghurst, M., Kato, H., Poupyrev, I., "The MagicBook: A Transitional AR Interface", in *Augmented Reality: the Interface is Everywhere*, SIGGRAPH 2001 Course Notes 27, 2001.

[Caudell92] Caudell, T., Mizell, D., "Augmented Reality: An Application of Heads-up Display Technology to Manual Manufacturing Processes", Proc. Hawaaii International Conference on Systems Sciences, Maui, Hawaii, IEEE press, pp 659-669, January 1992.

[Dias2002] Dias, J. M. S., Gamito, M., "Automatic Design Verification in Building Construction", Specialty Conference on Fully Integrated and Automated Project Processes in Civil Engineering, USA, Blacksburg, Virginia, January 2002

[Didier99] Stricker Didier, Klinher, G., Reiners, "A Fast and Robust Line-Based Optical Tracker for Augmented Reality Applications", in *Augmented Reality - Placing Artificial Objects in Real Scenes*, Proceedings of 1st International workshop on Augmented Reality - 1998, A. K. Peters ed., pp 129-145, 1999

[IAI96] IAI, "End User Guide to Industry Foundation Classes, Enabling Interoperability in the AEC/FM Industry", International Alliance for Interoperability (IAI), 1996.

[Kato99] Kato, H., et al., "ARToolKit, PC version 2.11", Hiroshima City University, December 1999

[Kato2001a] Kato, H., "Developing Applications with AR-Toolkit", SIGGRAPH 2001 Course Notes 27, 2001.

[Kato2001b] Kato, H., Billinghurst, M., Poupyrev, I., "Tangible Augmented Reality", in *Augmented Reality: the Interface is Everywhere*, SIGGRAPH 2001 Course Notes 27, 2001.

[Kilgard96] Kilgard , M., "The OpenGL Utility Toolkit (GLUT) Programming Interface, API Version 3, *Silicon Graphics, Inc.*, November 13, 1996

[Milgran99] Milgran, P., Herman, C. J., "A Taxonomy of Real and Virtual World Display Integration", in *Mixed Reality, Merging Real and Virtual Environments*, Ohmshda & Springer-Verlag, pp 5-30, 1999