International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:4, No:3, 2010

3DARModeler: a 3D Modeling System in Augmented Reality Environment

Trien V. Do and Jong-Weon Lee

***Abstract*—**This paper describes a 3D modeling system in Augmented Reality environment, named 3DARModeler. It can be considered a simple version of 3D Studio Max with necessary functions for a modeling system such as creating objects, applying texture, adding animation, estimating real light sources and casting shadows. The 3DARModeler introduces convenient, and effective human-computer interaction to build 3D models by combining both the traditional input method (mouse/keyboard) and the tangible input method (markers). It has the ability to align a new virtual object with the existing parts of a model. The 3DARModeler targets non-technical users. As such, they do not need much knowledge of computer graphics and modeling techniques. All they have to do is select basic objects, customize their attributes, and put them together to build a 3D model in a simple and intuitive way as if they were doing in the real world. Using the hierarchical modeling technique, the users are able to group several basic objects to manage them as a unified, complex object. The system can also connect with other 3D systems by importing and exporting VRML/3Ds Max files. A module of speech recognition is included in the system to provide flexible user interfaces.

***Keywords*—**3D Modeling, Augmented Reality, Geometric Modeling, Virtual Reality

I. Introduction

WITH the rapid development of hardware technologies, computer graphics has become much more sophisticated and reached to a high level of reality. More than ever, 3D modeling, one of the radical issues of computer graphics, is getting much interest. Nowadays, 3D models are widely used in various areas such as manufacturing, games, animation, and film, etc. To meet needs related to 3D modeling, some 3D graphic systems have already been launched into the global market. The most well-known products are AutoCAD, 3D Studio Max, Maya, Lightwave 3D and Cinema4D. They are powerful tools enabling users to create 3D models visually. However, they still use 2D interaction techniques and lack the use of 3D display techniques. They often decompose 3D design operations into sequences of 2D interface operations. These operations are not direct and intuitive. Moreover, models created by them are not real 3D in some senses because they are totally isolated from the 3D physical world. The users do not have the feeling that they can touch or grasp those virtual models. To address this problem, researchers are focusing on pulling graphics out of conventional displays and integrating them into the physical world. This technology, called *augmented reality* (AR) [2], is blurring the boundary between the real and the computer-generated graphics by enhancing what humans see, hear, feel and smell.

As soon as AR became an active research area, a lot of effort has been put into the 3D Modeling area. Modeling in AR enables to create 3D models and embed them to the real world. The most interesting aspect is that it brings to the users the illusion of building models in the real world. Many intuitive user interfaces and interactive techniques have been introduced and applied to 3D modeling systems in AR. And the ultimate goal is to make the users more satisfied in creating models. The AR environment is not only a work space for creating model based on tracking techniques but also merges models with the physical world. This provides the vision of how 3D models "live" in the real world. In this paper, a modeling system, named 3DARModeler, is proposed to take those advantages of AR. It can be considered a simple version of 3D Studio Max but in the Augmented Reality Environment. However, we do not wish to develop such a powerful and sophisticated system as 3D Studio Max. The aim of our system is to provide a flexible interface that assists the users in creating 3D models quickly and easily. Targeting non-technical users with minimum requirement of knowledge of computer graphics, modeling techniques as well as hardware devices, we decide to develop a desktop marker-based system with a normal PC camera. Of course, a Head Mounted Display (HMD) can be used, but it is not necessary. The users just need to select basic objects, customize their attributes, and put them together to build a 3D model. Therefore the system would be simple and cheap to setup. The users will interact with the system through the combination of various input approaches: the tangible AR interaction (markers), the verbal interaction, and the traditional interaction (mouse and keyboard).

The 3DARModeler provides most of the necessary functions for a modeling system such as creating a 3D model, applying texture, adding animation, estimating real light sources and casting shadows. With the ability to align a new virtual object with the existed ones, it helps create models

This research is supported by the Foundation of UCN Projects, the MKE, and the 21C Frontier R&D Program in Korea as a result of subproject UCN 08B3-O1-20S.

Trien V. Do is with the Mixed Reality & Interaction Laboratory, Sejong University, Seoul, Korea (e-mail: vantriendo@hotmail.com).

Jong-Weon Lee is the Mixed Reality & Interaction Laboratory, Sejong University, Seoul, Korea (e-mail: jwlee@sejong.ac.kr).

faster and more precisely. Several objects can be grouped to a unified object by the hierarchical modeling technique. The system can also connect with other 3D systems by importing and exporting VRML/3Ds Max files.

In the rest of the paper, we summarize some previous work in Section 2; after discussing the modeling technique in Section 3, we give an overview of the 3DARModeler in Section 4, and discuss techniques used in developing the system in Section 5; experimental results are presented in Section 6; and Section 7 provides the conclusion for this study and discusses possibilities for future work.

II. Related Work

So far, much work has been done in AR (both indoor and outdoor) to provide more intuitive 3D modeling systems. Construct3D [3], [4] is an intuitive 3D construction tool in AR for 3D geometry education. It requires HMDs, pen and pad props for magnetic tracking, and a 4-camera infrared-optical tracking system. The users interact with the system through a Personal Interaction Panel (PIP), a two-handed 3D interaction tool composed of instrumented hand-help props - a pen and a pad equipped with position and orientation trackers. The users can select various geometries from the PIP, customize their attributes and perform necessary geometry operations. However, the system is expensive, and its setup is complicated, possibly explaining its restricted distribution to only a few technical rooms so far. Individuals could hardly possess one of their own.

City-Planning System [8] is a 3D indoor modeling system based on AR with a tangible interface. The system enables the users to consider city plans more effectively and easily. Based on the concept called tangible user interface, Hirikazu Kato et al. proposes a new direct manipulation of virtual objects. Users hold a transparent cup upside down. Then, they can pick up, move or delete virtual objects. However the system mainly focuses on arranging the positions of 3D objects, no new object can be created.

iaTAR (Immersive Authoring of Tangible Augmented Reality Applications) [5] is a high level toolkit that enables rapid development of AR applications without any programming. The system uses Tangible AR interaction techniques and an authoring method called 'immersive authoring'. All the interactions with the system are done via markers and some tricks with hand. At first, users would be interested in such tangible interfaces, though over time they will grow weary of wasting time. Moreover, it might be difficult for the users to create 3D models which consist of several components because it does not support any automatic alignment features.

ARpm [6] is a front-end to the commercial system – 3D Studio Max. The system takes screen captures of the 3D Studio Max scene running in the background and overlays the virtual objects over the real camera image. It does not provide any interaction except sending the actual commands to 3D Studio Max.

Tinmith [7], [14] is a technology developed for 3D outdoor modeling in Augmented Reality by Wayne Piekarski and other researchers at University of South Australia. Using this technology they have written a number of applications. One of these is called "Street Furniture". The users can input virtual objects such as trees, tables, light posts, and other outdoor items. Using the markers on gloves, the users are able to manipulate these objects into expected positions, and then scale and rotate them. This application is useful for performing landscape gardening type tasks. It also allows designers to place virtual objects and see how they look in a given environment before purchasing the actual items.

To the best of our knowledge, there has been no system which addresses enough necessary functions of 3D modeling in AR. Most of the existed systems just demonstrate some specific functions or interactive techniques. Our main purpose is to develop a modeling system which can be widely used, not just in the laboratory. We implement the system keeping the user-oriented approach in mind. That is why we use a cheap camera and a fixed screen display instead of an HMD. We also combine the tangible interaction with the traditional input method such as mouse/keyboard and speech recognition to improve the usability of the system.

III. Modeling Technique

Most objects in the real world consist of several parts. Therefore, models created in a computer modeling system are often the combination of various parts. In order to meet this requirement, we define some primitive objects which are generally enough to create 3D models. They will be combined together in order to constitute more complex objects. However, these objects cause problems when they are transformed. An example object is a simple table consisting of a table-top and four legs. Fig. 1.a shows the side view with only two visible legs. Assume that the local origin of each of the five parts (the table-top and the four legs) is located at each cross. Now, the entire table needs rotating around the Z axis. If the parts are rotated individually, each part will rotate around its own local origin, resulting in Fig. 1.b. Even if each part is rotated in the same direction, the placement of each part relative to the others changes because the local origins are located at different points in the space. The similar problem happens when all the parts of the table are scaled.

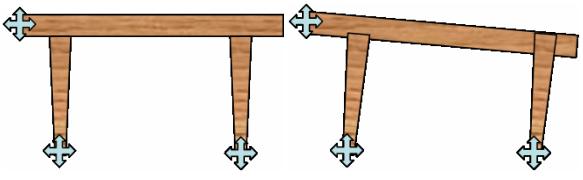


Fig. 1 The simple table (a) before, and (b) after rotated

Therefore, the entire table needs managing as a single, unified object to retain the original relationships among parts while rotating or scaling. In order for this to happen, all the parts of the table must be transformed around a common local

origin. A hierarchical structure should be used to manage a model. It is called *hierarchical modeling* [9]. Back to the case of the table, we define the whole table to be the top of the structure and the separate parts of the table to be the lower levels of the structure. Then, we can control the table as a unified object.

The hierarchical modeling technique is applied in the 3DARModeler. A model in the 3DARModeler may consist of several groups, and again each group may be the combination of some basic objects.

IV. System Overview

Being a desktop marker based AR system, the 3DARModeler uses ARToolkit version 2.71.3 [1] for augmented reality technique and OpenGL for graphic rendering. The system consists of a main cardboard and four operation markers:

- The main cardboard: This is the main work place where models are built. The cardboard contains multiple markers, so the users can freely interact with models without too much worry about the marker occlusion problem. When at least one marker is visible, the system can still compute the position of the marker set in the camera coordinate system and display models.
- The object marker: a new part of models is created on this marker and added to models on the main cardboard later. Depending on the position of this marker, the system is able to suggest positions where the users wish to put the new object. This makes the system smart and easy-to-use. The object marker is also used in light source estimation.
- The selecting marker: This marker is used to select any objects or groups of a model. All the editing operations are applied to these selected objects. It is also used to create the motion path for animation.
- The two editing markers: Depending on the selected operation, the increasing marker and the decreasing marker increase or decrease the value of objects' attributes. These markers change attributes of objects on both the object marker and the main cardboard.

Besides markers, with a toolbar, a menu system, and verbal command recognition feature, the 3DARModeler provides a flexible and convenient user interface.

In order to create a model, the users either attach a basic object or import a VRML/3DS object to the object marker. Then they can customize attributes of this object such as size, position, orientation, color, transparent level and texture. After the appropriate position is located, the object on the object marker can be placed to the main cardboard. Attributes of this newly added object are still customizable later. Next, the users can copy, move, or delete any parts of the model. To perform any editing operations, the users have to use the selecting marker to select the required objects. Furthermore, the users can group selected objects to manage them as a unified, complex object. Once again, any attributes of this complex object can be customized like any basic objects. Using the 3DARModeler, the users can create a model step by step as if they were dealing with objects in the real world.

V. System Implementation

In this section, we would like to discuss the following techniques regarding the implementation of the 3DARModeler.

A. User Interface

In order to provide the most convenient user interface, the 3DARModeler combines both the traditional input devices such as mouse/keyboard with a tangible interface and verbal commands. While the menu system (Fig. 2.b) provides all the system functions, the toolbar (Fig. 2.a) extracts some most frequently used functions from the menu system. Fig. 2.c shows two editing markers in the translation mode.

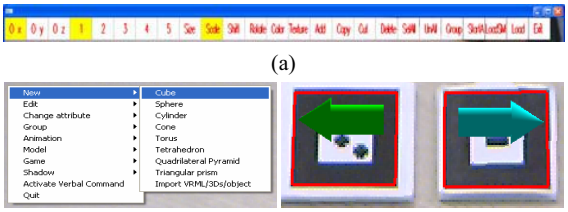


Fig. 2 User Interface

Because all the users are familiar with the mouse and keyboard, they should continue to use them to do simple tasks like selecting items from the toolbar or menu system and confirming an action. Though input devices like markers and hands are very intuitive, they are occasionally less effective. We can completely replace the menu system with a full graphics menu, and use the selecting marker to select functions. At first, the users would be interested in such a menu, though over time they will grow weary of wasting time for such simple tasks. Therefore, we use both kinds of input interfaces to offer the most effective and convenient system.

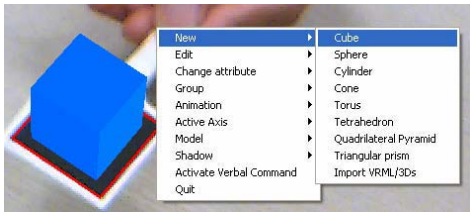
There is a problem with the selecting marker. If an object is too small or some objects are very close to each other, overlap each other, using the selecting marker may be not effective. In this case, the users can press the space bar on the keyboard to navigate sequentially object by object. The users also need to use some special keys to change the position of the virtual light sources until the real shadow and the virtual shadow become one.

Besides using those kinds of user interfaces, a speech recognition module is added to the system so that the users can use verbal commands to control the system instead of using mouse/keyboard. This feature is especially useful when the users wish to work in a large space. The Microsoft Speech API 5.1 integrated in Windows is used to implement this function. However, accuracy of speech recognition is not perfect. Two threads are used for speech recognition and command confirmation. This implementation is a little

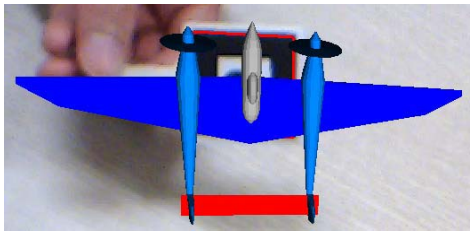
inconvenient for the users but it ensures that no misunderstanding occurs.

B. Binding Various Virtual Objects to a Marker at Runtime

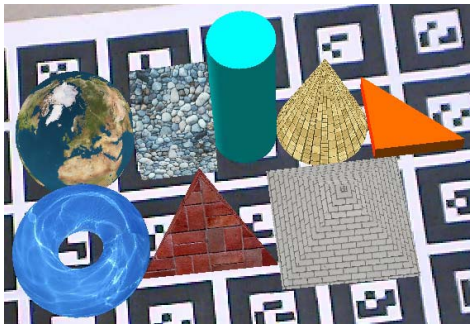
Normally, in marker based systems, each marker is fixed to a predefined virtual object. This means that the number of markers is proportional to the number of virtual objects whereas the users often work with one object at a time. Therefore, the system allows selecting any built-in primitive objects to attach to a marker at runtime (Fig. 3.a). The users can also browse VRML/3DSMax files on their local computers and attach them to the object marker (Fig. 3.b). With this feature, objects created by other systems can be easily reused.



(a)



(b)



(c)

Fig. 3 Attaching virtual objects to the object marker at runtime, (a) a built-in object, (b) an imported VRML object, (c) primitive objects

Currently, the 3DARModeler supports eight types of built-in primitive objects: spheres, cubes, cylinders, cones, triangular prisms, tori, tetrahedrons, and square pyramids (Fig. 3.c). However, with scaling operations on geometry, users can create other geometries such as ellipsoids, cuboids, planes, and lines. These primitive objects are enough to make most models.

By implementing this feature, we reduce the number of markers and make the system flexible and powerful. If the users do not like the built-in objects, they can use their own objects (VRML/3DSMax files) to create models effectively. The users can benefit from this as they do not need to remember which marker to use with a specific object.

C. Customizing Object's Attributes with Editing Markers

After created, an object should be customizable to constitute various models. Therefore, operations such as translating, rotating, scaling, sizing, coloring and texture mapping are provided. The required operation and active axis can be selected from the toolbar or the menu system (Fig. 4).

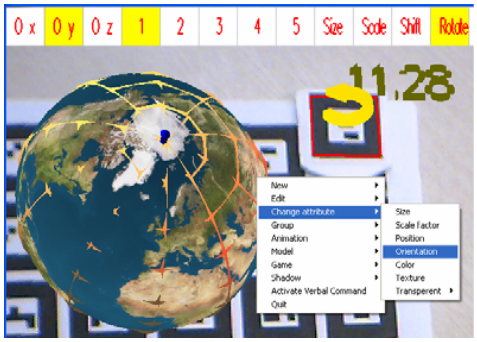
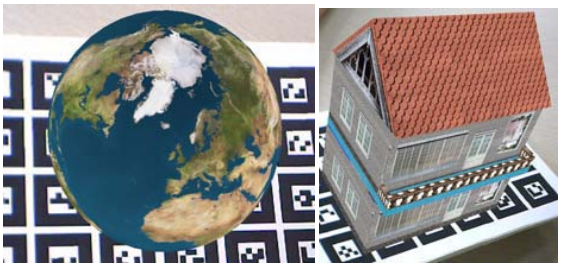


Fig. 4 Customizing Object's Attributes

Because with each operation the users should be able to increase or decrease the value of a certain attribute, two markers are used. Whenever these markers are visible to the camera, the operation is applied. The value of the attribute is displayed on the markers (Fig. 4). The users can intuitively observe the change of an attribute and terminate this progress by hiding the markers when the value reaches to an expected number. The speed of how fast the value should change can be determined by 5 levels (from 1 to 5 on the toolbar in the Fig. 4). Thanks to this, any refinement changes may be made to build precise models.

Texture mapping is a crucial technique in creating realistic 3D models. Without texture mapping the models would not be aesthetically pleasing. It not only increases the realistic level but also reduces the effort in modeling the real world. That is why special attention to this technique is paid in the 3DARModeler.

All the primitive geometries are redefined with the texture ability because by default OpenGL does not determine any method of texturing for those objects. So that, the users can browse common image formats such as .jpg, .bmp, .gif, .jpeg, .ico, .emf, .wmf, at any size and apply them to individual objects using the mipmaping technique. Fig. 5 shows two models of the earth and a house with textures. Customizing objects' attributes is an intuitive interaction technique of the 3DARModeler. The users can create a precise model intuitively as they can see the change of any attributes while modifying them.



(a) (b)

Fig. 5 The textured models of (a) The Earth and (b) a House

D. Selecting a Part of a Model

This is a necessary task to provide interactive capabilities. To control or customize attributes of a part of a model, that part must be selected first, like the way we often do in 2D space. However in 3D space, we cannot use a mouse to click on an object like in 2D space. Our solution is to use the selecting marker as a tangible interface for this purpose (Fig. 6). If the selecting marker points to an object for three seconds it will be selected and highlighted with its wireframe (see the earth with its wireframe in Fig. 4).

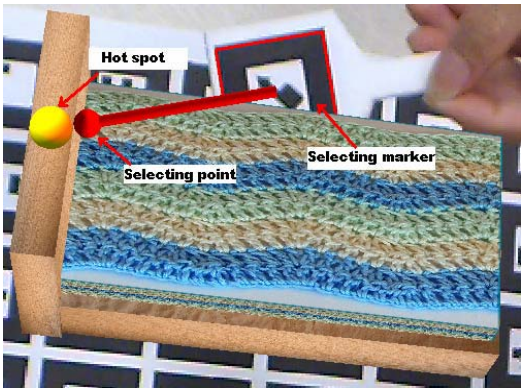


Fig. 6 Selecting a part of a model

The problem is how to determine which object is being pointed by the selecting marker. iaTAR [5] uses the bounding box to solve this problem. This solution is simple but it does not give much information about the pointed position within the object. In the 3DARModeler, this information is essential for the object alignment feature which is explained in the next section; therefore another method is implemented to address this task. For each kind of object some "hot spots" are defined. Each "hot spot" is a sphere with a certain radius. If the selecting point of the selecting marker is within one of those "hot spots" of an object, that object will be selected (Fig. 6).

The difficulty here is that each marker has its own coordinate system [1] (Fig. 7). We cannot directly use the coordinates of virtual objects attached to different markers to determine whether the selecting point is within the "hot spots", so we need to convert coordinates of every object to a unique coordinate system. In our cases, the camera coordinate system is used since it is concrete and often fixed. Of course, markers' or world coordinate systems are also possible. Another problem is that together with the object, the "hot spots" can be changed by translating, rotating, scaling, and resizing operations. Therefore, before converting the coordinates of those "hot spots", we need to recalculate their coordinates by multiplying them with the proper transformation matrices [10]. Once the coordinates of the "hot spots" are obtained in their marker coordinate systems, they can be presented in the 4x1 vector format for homogeneous transformation. Assume that a vector G presents the coordinates of a "hot spot" in a 4x1 vector.

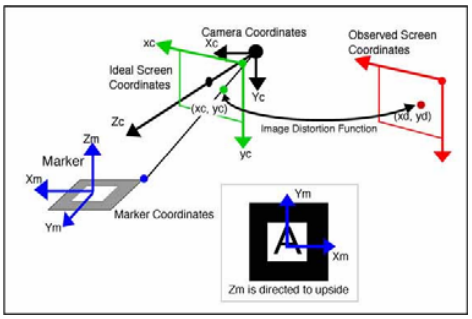


Fig. 7 Relationship among coordinate systems [1]

$$G^T = \begin{bmatrix} X_h & Y_h & Z_h & 1 \end{bmatrix}$$

When a trained marker is recognized by the camera, the ARToolkit library helps establish the transformation matrix between the marker and the camera coordinate systems. Say it is the 4-by-4 matrix T_{hc}.

We can calculate the coordinates *C* of a "hot spot" in the camera coordinate system as:

$$C^T = T_{hc} * G = \begin{bmatrix} X_c & Y_c & Z_c & 1 \end{bmatrix}$$

Similarly the coordinates of the selecting point can be converted to the camera coordinate system. The rest of the steps are simple. We simply use the distance formula in analytic geometry and select an *epsilon* to identify whether the selecting point is within any "hot spot". Assume that $S(x_s,y_s,z_s)$ is the coordinate of the selecting point and $G(x_g,y_g,z_g)$ is the coordinate of a "hot spot" in the camera coordinate system. The distance *d* between these two objects can be calculated with the formula:

$$d = \sqrt{(x_s - x_g)^2 + (y_s - y_g)^2 + (z_s - z_g)^2}$$

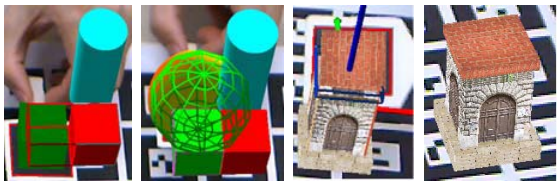
If $d < epsilon$ then that "hot spot" is being selected and we can identify the selected geometry.

The users can also select multiple objects or deselect an object with the same operation.

E. Aligning a New Component with the Existing Parts of a Model

Normally, all parts of a model must have some relative relationships with each others. This means a new component of the model should be added closely to an existing part of the model. Precisely sensing the position of objects in 3D space is

not easy as it is hard to feel the depth of objects in the real 3D space. So the 3DARModeler provides a convenient feature: position suggestion. Whenever the object on the object marker is close to any existing parts of the model, the system will give a suitable position suggestion. If the users decide to add the object to the suggested position, it is automatically aligned with the existing part. A suggested position depends on the attributes of the virtual object on the object marker and the close part of the model. And it should be correct even after the objects are translated, rotated, and scaled. This feature helps quickly create more precise models.



(a) (b) (c) (d)

Fig. 8 Suggesting suitable positions

To address this problem, the 3DARModeler first needs to be able to recognize the "expected part" of the model where the users would like to align the new component with, and then determines a suitable position to align it with the "expected part". Once again, the approach with "hot spots", which is described above, is employed. In this case, the new component plays the role of the selecting point. After a "hot spot" is determined to be close to the new component, the "expected part" will be identified. Based on the information of the "hot spot", the "expected part", and the new component, a procedure is implemented to align the new components with the "expected part". Fig. 8 illustrates this feature. In Fig. 8.a, the system suggests that the green cube should be put on the left side of the red cube at the position of the red wireframe, in Fig. 8.b it suggests that the orange sphere should be placed above the green cube, and the Fig. 8.c,d demonstrate how this feature works with transformed objects. This suggestion ability reduces significant users' effort and time in creating models.

F. Displaying Object' Attributes

To add a new part to a model, the users may need detailed information about a specified part of the model. So a billboard containing attributes of an object should be shown on the selecting marker when an object is selected. The problem is that the orientation of the selecting marker is not always easy for users to read information. The relative orientation of the selecting marker to the camera need calculating to make sure that this billboard is always in a suitable orientation. The function arGetAngle of the ARToolkit library helps do this by extracting Euler angles from a rotation matrix between any marker and the camera. Fig. 9 shows that the billboard displaying information of the brick block is very easy to read although the selecting marker is at an inconvenient orientation.

Fig. 9 The Information Billboard

G. Grouping Objects to a Unified Complex Object

Grouping is important in the hierarchy modeling technique. With this function the users can create complex objects, save them to a file and add them to any other models later. This function is implemented following the below steps:

1. Determine selected objects
2. Detach the selected objects from the free list
3. Add a new group and attach these objects to this group
4. Assign a new common origin for all the objects in the group

After that, the users can customize any attributes of a group like a primitive object. Of course, the users are able to ungroup a complex object.

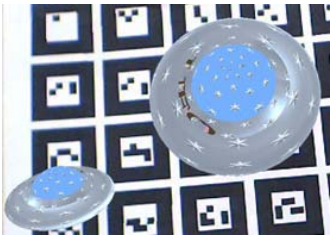


Fig. 10 Working with a group

In Fig. 10, a simple spaceship is a group of four basic objects: two cones and two spheres. They are scaled down around the Z axis and textures are applied. With copying operation the spaceship is duplicated. Then it is rotated, reduced, and put at another position on the main cardboard. Notice that the spaceship retains its form throughout each of the customizing operations.

H. Editing Operations (Copying/Cutting and Pasting)

Editing operations such as copying, moving, deleting, and pasting are useful for every system. Hirokazu Kato et al [8] implemented these kinds of operations using a cup interface. When the cup covers a virtual object on a table, the virtual object is locked in the cup. Then, all editing operations are applied to that virtual object. Although intuitive, this approach cannot apply to the 3DARModeler as the cup may cover several overlapped objects. Therefore, all the basic editing operations of the traditional system in 2D space are simulated in the 3DARModeler where the object marker plays the role of a clipboard. First, the users select a required object using the selecting marker, and then click on the copy /cut operation on the toolbar or menu system. The copied/cut object will be attached to the object marker and can be moved to any

position with the marker. Finally, this object can be placed back on the main cardboard with the adding operation. These operations are intuitive and simple.

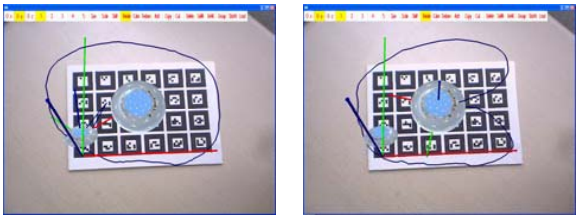
İ. *Animation*

Today, animation is often a part of a modeling system. It enables to build likely models with movements. Although at this stage, the 3DARModeler allows creating only simple animation, the users are able to create animation in a very intuitive and natural way. Animation can be added at two levels: object based and group based. An object can rotate around a certain axis of itself. In addition to rotating around its own axes like an object, a group is able to move along a certain motion path defined by the users. With these animation features, the users can simulate a running car, a clock and a lot more.

To enable a group to move along a motion path, the below steps must be done:

1. Determine the selected group
2. Draw the motion path in 3D space by moving the selecting marker
3. Move the group along the created motion path in each time step.

The speed, at which a group moves along its motion path, depends on how fast the users move their hand while drawing that motion path. It is simple and natural.



(a) (b)

Fig. 11 Groups and their motion paths for animation

Fig. 11.a, b show two models: a small spaceship and a big one created by the 3DARModeler. Each model has a separate motion path and it moves along its own path while rotating around its Z axis.

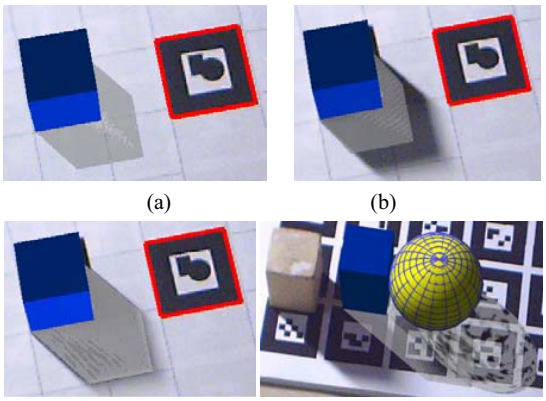
J. *Light Source Estimation and Shadow*

One of the main advantages of AR systems is that virtual objects are immersed in the physical world, co-existed with the real objects. An ideal system should make the users unable to distinguish between virtual and real objects. So the characteristic of the environment like shadow effects must be taken into account. Their shadows need to be logical and appropriate while standing with real objects [12]. To achieve this goal, real light sources are estimated then used to cast the virtual shadows.

Much effort has been put in the real light source estimation area [11], [13]. At this stage, the light sources are estimated semi-automatically by the following steps:

1. Create an exact model of a real object and a draw virtual shadow of the model (Fig. 12.a). Actually a simple model like a cube is enough.
2. Overlap the real object to the virtual model (Fig. 12.b)
3. Manually change the position of the virtual light source until the real shadow and the virtual shadow are collocated (Fig. 12.c)
4. Use the estimated light source to render models

This process can be applied for any numbers of light sources one by one. Currently, the 3DARModeler deals with two different light sources. Because the camera and light sources are often fixed, the users have to perform this task only once. The task is simple while the accuracy is high. The light source positions are estimated in the camera coordinate system, and then they are transformed to the marker coordinate system to render virtual models.

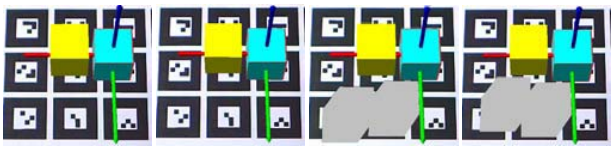


(a) (b)

(c) (d)

Fig. 12 Real light source estimation and shadow casting

As in Fig. 12.d, the real object is put next to some virtual objects and the shadows look fine on the main cardboard.



(a) (b) (c) (d)

Fig. 13 Advantage of shadow casting

The advantage of shadow casting in the 3DARModeler is that shadows provide a better understanding of the relationship between objects' positions in 3D space. In Fig. 13.a, two cubes have the same values of the y coordinate (the green axis) but the yellow cube is higher than the cyan one. In Fig. 13.b, they have the same values of the z coordinate (the green blue) but different values of the y coordinate. Without any shadows, the users may not know the relative relationship between the two cubes. By looking at Fig. 13.c, d users have better perception. This feature also assists the modeling progress when the users are adding new parts to models step by step as it provide users with a better sense of spatial relationships among parts.

K. File Input/Output, Import/Export

The system allows saving the current model or a part of the model to a file and loading it again later. Saving a part of the model is also a useful idea because the saved parts can be reused for several models. The users can create their own database containing complex objects in specific fields.

The users can also reuse models created by other systems like VRML/3Ds Max. This feature is especially useful. For example, if a user would like to arrange furniture in a room, they can import furniture from existing files and move them freely as if they were dealing with real ones. Notice that all of the imported objects are managed as basic objects, so their attributes can be customized as usual. In this case, the system can be used like a working space for other software.

A whole model or a part of a model can be exported to a VRML file, so they can be imported in other systems like 3D Studio Max. While implementing this function, the significant differences between the Marker's coordinate system and the VRML's coordinate system have to be taken into account. Another problem is that the primitive geometries of VRML cannot be used because the origins of object coordinate system of primitive geometries in VRML are different in OpenGL. Therefore, new procedures must be rewritten to generate such VRML's primitive geometries. Importing and exporting features provide the system with the ability to connect with any systems which support VRML and 3Ds Max files. It is an important feature for the system to be used widely as the users can reuse results and switch among different systems flexibly depending on specific needs.

VI. Experimental Results

A. Pilot Study

To get objective feedbacks from end-users, a pilot user study was seriously carried out. The 3DARModeler were installed on the PC, Intel® Core™2 Quad CPU @2.41 GHz, 1.00 GB of RAM, NVIDA GeForce 8600 GTS with a VIJE eye 1.3M camera. The system ran stably using around 30MB memory and 30% CPU. The main goal was to evaluate whether the 3DARModeler was easy-to-use, practical and to compare it with the 3D Studio Max in several aspects. The user study's scenario was divided into three phases:

1. A 10 minute demonstration video was played to explain how to use the system. The video showed how to create a simple animated spaceship like in Fig. 10 and some other functions which were not used in the process of creating the spaceship. The users might pause the video any time to pose questions.
2. The participants themselves worked with the system for 15 minutes with our support.
3. The participants were asked to re-create the spaceship as shown in the video (All the required textures were provided beforehand). The taken time was recorded.

After that the participants were asked to fill in a questionnaire. There were 15 compulsory questions for all participants: 10 about the 3DARModeler and 5 about the AR environment. Participants with experience of the 3D Studio Max needed to answer three other questions. Of course the participants were asked to give any other comments.

The user study ran for 10 days. We announced the study to students in our university on some notice boards. Each participant was required to work with us for an hour.

B. Results and Discussion

After 10 days, 37 participants (12 females and 25 males) took part in the user study. They were both undergraduates and postgraduates of various departments from 19 to 41 years old. 14 of 37 participants had some 3D Studio Max experience; others did not know any 3D graphics system. They all did not know or knew very little about AR before.

The results showed that after 17.5 minutes (on average), the participants could work smoothly with the system. To rate how easy-to-use the system was, five levels (1-Difficult, 2-A little difficult, 3-Fair, 4-Easy, 5-Very easy) were listed. All the operations were rated above 4.0 except 3.1 for the operation of selecting objects with the selecting marker. This is what we anticipated. Reasons are that the viewpoint and orientation of participants' eyes were a little different from the fixed camera (remember that we do not use an HMD); and the participants had not enough time to have a good spatial feeling. Participants rated the system was interesting (4.5/5.0) and practical (4.0/5.0). On average, it took the participants 18.5 minutes to model the spaceship which consists of two cones and two spheres. It was expected results as all participants were new to the system.

The participants were also asked to evaluate input metaphors (mouse/keyboard versus markers versus verbal command). 33 of 37 participants (90%) preferred the combination of markers and mouse/keyboard. Though intuitive and interesting, markers were a little less effective. Verbal commands encountered problems with pronunciation, command recognition and caused participants a little irritation with confirmation.

Answers for questions about the AR environment reveal that participants were extremely impressed by AR. They had the impression that virtual models were parts of the physical world and they felt they could grasp them. They enjoyed the idea of putting the virtual models at an appropriate position in the real world and observing how they "live" with physical objects. They all agreed that AR environment was very suitable for 3D modeling systems.

Fourteen participants with 3Ds Max experience expressed that the idea of developing a system like 3Ds Max in AR was practical and they expected that. They also gave comparisons between the 3DARModeler and the 3Ds Max in the following characteristics: F1-Intuitive, F2-Easy to use, F3-Suitable for creating simple models, F4-Suitable for creating complex models, F5- Suitable for quickly building a prototype, F6-Suitable for arrangement/planning tasks, F7-Suitable for beginners, non-professional users. The results are shown in the Fig. 13. Obviously, these results supported our goals in developing the 3DARModeler.

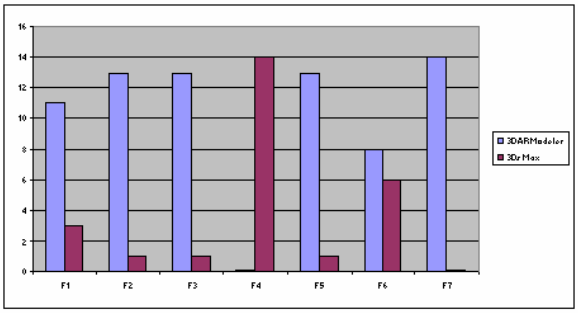
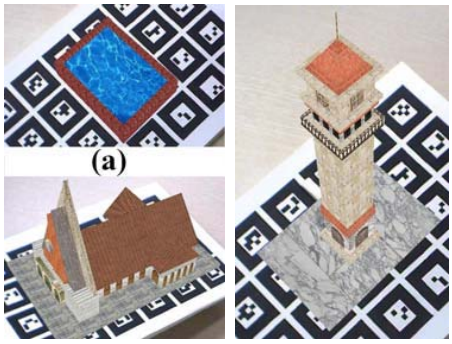


Fig. 14 3DARModeler versus 3Ds Max

All the participants expressed their interest in the proposed 3DARModeler. They especially appreciated the ability of component alignment feature as mentioned in the Section 5.E. However, they also gave some comments for the future work. In some cases they preferred directly entering values in a textbox for object's attributes than using the editing markers. Using hot keys is also a good idea. Some other features should be provided such as Boolean operations, blending a certain color of textures, etc.



(b) (c)

Fig. 15 Some models created by users (a) a swimming pool, (b) a church (c) a tower

Some participants were so interested in our system that they asked for a copy of the 3DARModeler to install on their computer. Later they sent us some models. We were very surprised at their work. Fig. 14 shows some of them. All the models were created purely by the 3DARModeler without importing any VRML or 3Ds files. And in Fig. 15, we finally used some models to construct a villa containing 5 groups: a house, a tower, a swimming pool, a tennis court and a fence.

VII. Conclusion and Future Work

In this paper, a comprehensive approach and various techniques for developing a 3D modeling system in AR are described. The primary advantage of using augmented reality in this modeling system is that the users are able to create 3D models as if they were doing in the real world. Interesting and easy-to-use are the other advantages of the system. Setup tasks are simple too. Just run a setup file, print all the marker in the Pattern folder, then turn on a camera, the users can at once work with the AR environment to create 3D models without a lot of computer skills and training. Neither further libraries nor hardware devices are required. We hope that AR systems will be utilized more by a variety of users.

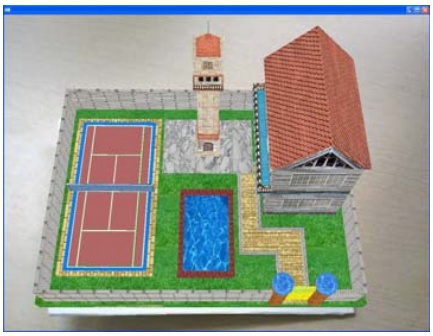


Fig. 16 The model of a villa

With regard to 3D modeling in Augmented Reality, we have made the following contributions:

- The 3DARModeler brings the conventional and natural way of 3D modeling to Augmented Reality. The users can create a complex model by assembling primitive geometries and grouping them to form a unified object.
- The 3DARModeler combines the tangible interaction approach with the traditional interaction and speech recognition to improve the usability of the system.
- The 3DARModeler provides a flexible method of adding a new part to a model. The users can use built-in objects or import their own objects (VRML/3DsMax files) to create a model. The system can realize the position where users intend to put a new part and give appropriate suggestions.
- The 3DARModeler allows users to customize objects' attributes in an intuitive and accurate way which is similar to how humans think and act in the real world.
- The 3DARModeler provides a new way of selecting, copying, moving, pasting and deleting 3D objects in augmented reality.
- The 3DARModeler introduces a simple but natural way of making animation in Augmented Reality.
- The 3DARModeler implements simple but effective method for real light source estimation.

With the 3DARModeler, the users are able to quickly design prototypes, models and conveniently express their ideas. The system can also be used in some scenarios like furniture arrangement, color and texture customization to meet user's specific needs.

The 3DARModeler may also be extended or modified to meet other purposes. An example idea is that it can be used to create books like the BlackMagic Book [15]. 3D contents should be created by the 3DARModeler or VRML/3D Studio Max then added to the main cardboard. Four button (First, Previos, Next, and Last) would be available for the users to navigate the books. Each book would be stored in a file and the main cardboard would display its contents

However, regarding a 3D modeling system, the

3DARModeler lacks a few features for it to become a commercial product as mentioned in Section 1. Convinced by the pilot user study, we plan to continue to improve the system. We believe that the idea of developing a version of 3D Studio Max in AR will be employed in the near future.

REFERENCES

[1] http://www.hitl.washington.edu/artoolkit/
[2] R. Azuma, "A survey of augmented reality", In Presence: Teleoperators and Virtual Environments 6, 4 (August 1997), 355 – 385.
[3] http://www.ims.tuwien.ac.at/research/construct3d/
[4] Hannes Kaufmann, "Construct3D: An Augmented Reality Application for Mathematics and Geometry Education", ACM Multimedia Conference 2002, DOI= http://doi.acm.org/10.1145/641140
[5] Gun A. Lee, Claudia Nelles, Mark Billinghurst, Gerard Jounghyun Kim, "Immersive Authoring of Tangible Augmented Reality Applications", Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2004)
[6] Peter Fiala, Nicoletta Adamo-Villani, "ARpm: an Augmented Reality Interface for Polygonal Modeling" ", Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2005)
[7] Wayne Piekarski and Bruce H. Thomas , "Developing Interactive Augmented Reality Modelling Applications", International workshop on software technology for Augmented Reality
[8] Hirokazu Kato, Keihachiro Tachibana, Masaaki Tanabe, Takeaki Nakajima, Yumiko Fukuda, "A City-Planning System based on Augmented Reality with a Tangible Interface", Proceedings of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2003)
[9] Michael O'Rourke, "Principles of Three-Dimensional Computer Animation: Modeling, Rendering, and Animating With 3d Computer Graphics", New York Norton Publisher, pp. 53-60.
[10] http://bishopw.loni.ucla.edu/AIR5/homogenous.html.
[11] Katrien Jacobs, Cameron Angus, Celine Loscos, Jean-Daniel Nahmias, Alex Reche, Anthony Steed, "Automatic generation of consistent shadows for Augmented Reality", Proceedings of Graphics Interface 2005
[12] Natsuki Sugano, Hirokazu Kato and Keihachiro Tachibana, "The Effects of Shadow Representation of Virtual Objects in Augmented Reality", Proceedings of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2003)
[13] Masayuki Kanbara and Naokazu Yokoya, "Real-time Estimation of Light Source Environment for Photorealistic Augmented Reality", Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)
[14] http://www.tinmith.net
[15] http://www.hitlabnz.org/wiki/Black_Magic_Book