Tinmith-Metro: New Outdoor Techniques for Creating City Models with an Augmented Reality Wearable Computer

Wayne Piekarski and Bruce H. Thomas
Wearable Computer Laboratory
School of Computer and Information Science
University of South Australia
Mawson Lakes, SA, 5095, Australia
{wayne, thomas}@cs.unisa.edu.au

Abstract

This paper presents new techniques for capturing and viewing on site 3D graphical models for large outdoor objects. Using an augmented reality wearable computer, we have developed a software system, known as Tinmith-Metro. Tinmith-Metro allows users to control a 3D constructive solid geometry modeller for building graphical objects of large physical artefacts, for example buildings, in the physical world. The 3D modeller is driven by a new user interface known as Tinmith-Hand, which allows the user to control the modeller using a set of pinch gloves and hand tracking. These techniques allow user to supply their AR renderers with models that would previously have to be captured with manual, time-consuming, and/or expensive methods.

Keywords: augmented reality, 3D modelling, constructive solid geometry, mobile user interfaces

1. Introduction

This paper presents a new methodology for capturing city models and a software implementation of this methodology, known as Tinmith-Metro. This technique supports the modelling of relatively complex buildings and other large physical structures (rendered in Figure 1 and photographed in (3) of Figure 2), and the positioning of prefabricated graphical models of typical community infrastructure and city objects. The user performs these tasks interactively while walking around outdoors with the aid of an augmented reality (AR) wearable computer. The modeller uses constructive solid geometry (CSG) related techniques to allow the user to build up complex shapes from simpler primitives. The Tinmith-Metro system combines the CSG modeller with a user interface, known as Tinmith-Hand, which supports interaction techniques based on pinch gloves and vision based hand tracking.

This work was motivated while exploring AR methods and systems for use in an outdoor environment. Those investigations have required us to construct the necessary 3D models indoors to allow the AR systems to render appropriate immersive information. We believe it is possible to streamline this entry process by constructing these models interactively outdoors, making the process more accurate and efficient. Currently, the authors know of no previous work that allows interactive 3D outdoor AR modelling.

The remainder of this introduction discusses the aims of this research and an overview of the newly implemented technologies. The relevant previous work on which this system is based is then presented; followed by a discussion of currently available methods for outdoor model capture. Two application domains are examined with detailed examples, showing our techniques in action in real life situations to demonstrate their power and flexibility. As part of the example, Tinmith-Hand, the menu and glove user interface is discussed, showing the user interaction for model construction. The CSG modelling system is described, and how it is used to construct the higher level shapes. The Tinmith software and hardware architecture, which forms the base for Tinmith-Metro, is also explained. Finally, a number of other application domains are examined.

In this paper, we do not focus on the problems of tracking or registration, but concentrate on the techniques for users to build large outdoor 3D graphical models. We present solutions that work within the technology available today to implement our ideas.

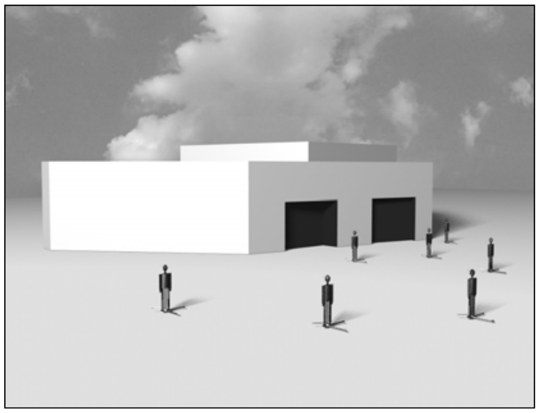
Figure 1 - Final rendered output of 3D school building model, captured using new techniques demonstrated in Tinmith-Metro

Goals

The main goal of this work is to produce new techniques for constructing large 3D graphical models of existing structures, without external information such as maps, images, or designs. Our aim is for an intuitive methodology; one that can be used to construct models in a way that seems natural. Using CSG operations, we leverage a person's common understanding of combining and carving solid shapes in the physical world, and their ability to apply this understanding to the construction of objects in a virtual environment. We believe the user's hands is the most appropriate vehicle to communicate with the computer. Finally, augmented reality allows a user to visually verify the quality of the models at the time of creation.

New technology implemented

The new methodologies for the construction of outdoor buildings are implemented into a complete city modelling demonstration system, Tinmith-Metro. The following are the major technologies:

CSG operations to create 3D objects – Using the CSG engine and Tinmith-Hand, the user can create large 3D objects (such as buildings) from simple primitives (such as infinite planes) while walking outdoors. By combining these operations, the user may produce more complicated shapes and visually verify the results interactively.

Placement of 3D prefabricated objects – By loading pre-fabricated models into the system (such as cars, street lights, and trees), the user can position graphical objects in an outdoor environment interactively.

Tinmith-Hand menu system – Using a set of custom-built pinch gloves, the user may execute system commands by navigating special menus linked to finger press events on the gloves.

Tinmith-Hand finger tracking – Through the use of vision based tracking techniques, the system tracks the position and orientation of markers placed on the user's hands. This tracking forms the basis for one and two handed interactions.

Tinmith-Hand multiple input devices – The user has a choice of four input devices, one handed tracking, two handed tracking, head tracked eye cursor, and a traditional hand-held track ball. These input devices provide the user with the means to perform 3D manipulation of graphical objects through the use of image plane techniques [13].

Specialised visualisation techniques – In addition to the immersive AR display, the system supports the following external camera views: orbital view, 2D top down rotating map view, as well as other body relative camera angles.

Tinmith-evo5 augmented reality system – The Tinmith-evo5 augmented reality system is the foundation for all the higher-level domain specific applications. Tinmith-evo5 provides support for abstracting tracker devices, object communications, and a hierarchical 3D object renderer.

Earth coordinate systems – Tinmith-evo5 internally supports the use of multiple Earth based coordinate systems, and translations between them.

2. Related work

To place our work in the context of other researchers, a background of mobile outdoor augmented reality projects is presented. Related investigations on interactive model capturing systems are then described. Finally, a number of automated modelling techniques are discussed and deficiencies noted.

Previous mobile AR work

Currently, most AR (and even VR) systems focus on issues of improving information presentation. The user walks around the world (either indoors or outdoors), viewing the world with virtual objects superimposed. In an outdoor setting, a number of systems have been implemented which perform these tasks, such as the Columbia MARS system [5], the NRL BARS system [9], and previous UniSA Tinmith systems [12, 18]. Although most of these systems are also able to be connected and share information using wireless networks, a key point is these systems only allow the user to control how the information is presented. They are not designed to enter large quantities of new graphical information, apart from marking simple points.

Previous interactive capture systems

There are a number of techniques currently available to allow users to interactively construct models of objects. Two interesting systems for constructing simple models by means of tangible building blocks are described. In [15], cameras were used to capture the arrangement of 3D blocks, and then project textures onto the surface. The blocks could then be rearranged and the graphical models updated. In [1], plastic bricks with microcontrollers may be connected together, with the arrangement then captured by message passing and producing a 3D graphical model of the construction. The user may then tour the structure in a virtual environment.

In the Façade system [3], a collection of photos from various angles may be converted into an approximate 3D model, with the user guiding the system on making decisions about various geometric constructs.

MultiGen Smart Scene [11] is a commercially available system that allows users to construct objects in a virtual environment. The user interface is centred on two pinch gloves, supporting a number of two-handed interaction techniques. The system tends to focus on working on a direct ratio, rather than action at a distance techniques.

Other automated capture techniques

In the construction and surveying domain, a number of techniques are used to capture the geometry of outdoor objects [16]. In many cases, CAD drawings of buildings are already available, however for use in AR applications there may be a number of problems with these:

- Drawings may only be 2D floor plans, and immersive rendering requires full 3D models, preferably solid graphical models and not simple wire frames.
- Top down 2D floor plans do not always show 3D features like tunnels, pitched roofs, and door height. Extruding a 2D model cannot produce these features.
- During the construction or renovations of the building, the actual implementation of the building may vary without the plans being updated.
- The drawings may only be available on paper.

Based on these problems, there is a spectrum of possible information sources for AR systems, with two extreme end cases. The worst case is there is no useful information, and in the best case there is a complete 3D model in a proper data format. Our paper focuses on improving the methods for cases where there is little or no prior information about the object to be modelled. There are a number of methods to construct these models; two of these methods are as follows: directly measuring the building by tape measure, or the acquisition of the graphical model by range sensing.

A technologically simple method for constructing a 3D model building is to measure it with a tape measure, recording the dimensions and geometry of the building. This information can then be used to build the model using a desktop 3D CAD system. The model may be visually verified by loading it into an AR system, taken outdoors, and compared against physical building to check for accuracy. If some details have not been entered properly, the person must refine the model by moving between the AR system and the desktop until they are satisfied (which is how we have previously created our models).

Recent technology has enabled the automated capture of buildings using portable laser scanners, synthetic aperture radar, and stereo imagery. However, these methods tend to generate very large quantities of polygons for even simple cube shapes, and so filtering on the results is required. All features in the object must be visible to the scanning device; otherwise, those obscured areas will be poorly defined. These methods may be ineffective in environments where it is not possible to obtain all the necessary views of the building, such as when it is covered by trees, clouds, or other objects.

3. Building construction example

Our first application domain for Tinmith-Metro is the modelling of large outdoor structures. An example of how to construct an object model is detailed to examine this modelling technique. This example models a school building, which is a round shape, with an air conditioning tower on the roof, and large windows on the side. The building is neither a box nor a cylinder, and so requires different primitives for modelling.

System start up – The user dons the wearable computer, HMD, and pinch gloves. The user then starts Tinmith-Metro and performs the calibration of the trackers.

Create perimeter walls – The overall top down outline of a building must first be specified. In the example building, there are 32 facets, but the user will only define the outline of the building with 10 planes for simplicity. The building is approximately a cylinder, but not similar enough to use the real cylinder primitive. To create the outline, the user creates infinite planes to mark each wall. Each plane is created by the following: 1) the user positions themselves to look down the edge of a building wall, at any convenient distance, 2) the user places the eye cursor along the wall edge, 3) the user selecting the menu option with the gloves to create a right facing wall, and 4) the right facing wall (an infinite plane) is added to the virtual world intersecting the eye cursor and perpendicular to the image plane. This wall cuts the entire infinite world in half, in the same way as the real wall does, with the left being inside the building, and the right being outside. By walking around the building and marking each plane, the user is carving away sections of the infinite space and defining the volume of the building. Eventually, when the user has completely circled the building, the perimeter of the volume is no longer infinite, and is now closed. The final result is a 2D bounded perimeter as shown from the top down view in (1) of Figure 2. This figure shows the very long planes that created the bounding volume.

Create floor and roof – To complete the first solid shape of the model, the 2D perimeter is constrained with a roof and floor. These are created by the user looking toward the centre of the building and creating a default floor at zero metres and a default roof at three metres.

Create solid object – The infinite planes are all separate

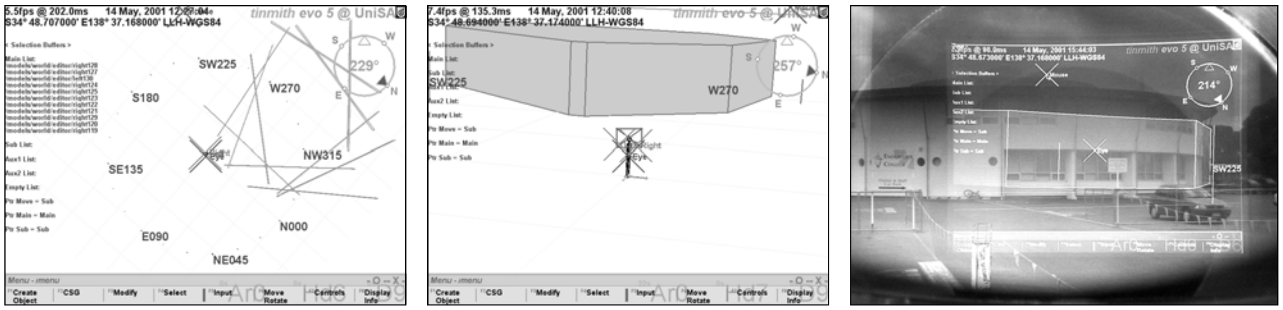

Figure 2 – Tinmith-Metro Demonstrating Various Stages of Construction of 3D School Model
(1) Top down view of infinite planes defining volume of building, (2) Solid round shaped building, (3) Immersive HMD shot of school, looking south

and do not currently form a proper solid object. To form a solid object, the CSG intersection operation is selected from the Tinmith-Hand menu with the pinch glove. Once the operation has been initiated, the renderer draws a preview of current model. The user interactively corrects the height of the roof by lifting or lowering it to match the correct height of the building; and this is verified by using the registration of the virtual object against the physical building. When the roof is in the correct position, the intersection operation is committed with another glove menu selection. The model of the building is now at the stage depicted in (2) and (3) of Figure 2.

Create air conditioner tower – The air conditioning tower may be added in a number of ways. One method uses a default cylinder primitive, scaled and moved into position. A second method is to use the infinite planes technique to create a completely new shape. In this example, the user will use the infinite planes technique and the same CSG intersect operation, but this time the tower roof is higher, with a reduced width.

Combine two objects together – Currently, portions of the air conditioner tower exist inside the building object. The tower contains internal facets that are not visible and wastes graphics resources. Using the CSG union operation, the system can merge the tower and building objects into one object, removing the internal facets and simplifying the model, producing a single object.

Create windows – The next task for the user is to add coloured depressed windows into the building, which is done by carving into the building. First the user creates a box shaped object to use as a tool, whose profile matches that of the window, and has a depth of at least the window's depression. The box object tool may be created using a prefabricated model or constructed by the user. The user positions themselves to be able to see the window easily, and the CSG difference operation is selected. Using the gloves, the user pushes the box object tool into the building, in the same way that a cookie cutter is used to remove portions of dough. The box object tool is pushed into the wall until it is the same depth as the window depression. The facets cut into the building are coloured blue like the window, and are also indented into the shape, they are not just surface modifications.

Results

The user has just created a model of a building that is approximately round, along with an air conditioning machinery tower, and carved in windows, shown as a ray traced image in Figure 1. The user may continue to model the building to any level of complexity that they desire, depending on the requirements for the model.

The accuracy of the objects created with this system is largely dependent on the tracking hardware used, and the amounts of care taken by the user to accurately enter the information. For position tracking, a standard GPS with differential has accuracies ranging from 1 to 5 metres. For orientation tracking, the IS-300 has a resolution of 1° static and 3° dynamic accuracy, with models measured as close to the building as possible (without degrading the GPS) are the most desirable.

4. Street furniture example

A second application domain for Tinmith-Metro is to position models of typical community infrastructure, "street furniture", such as park benches, rubbish containers, and street lamps, located at our university campus. In this second example, the user operates a customised menu structure in Tinmith-Metro to position prefabricated models of smaller street furniture items. These models were created using NewTek LightWave and converted into the custom Tinmith file format.

Create grass area – The user creates a grass area by using the infinite planes technique to mark out a perimeter. The area the user is marking is between numerous campus buildings, and so the user approximates the area with several planes. A special "create grass" menu operation is selected, and then floor and roof are both created at default heights and intersected with the perimeter. The resulting object is a grass slab that is 5 cm thick. The purpose of the grass is to supply a background for the objects, and so accuracy or shape is not a concern.

Place down objects – The user moves around the area, standing near the real world objects. The user has previously modelled the required objects such as lamps, rubbish bins, benches, and trees on a desktop system, at the desired accuracy. By using the glove and menu, the object to place down is selected and then instantiated into the modelling environment.

Placement defaults and adjustment - By default, an object is placed one metre in front of the user, and oriented away from the image plane. This allows the user to immediately place an object at the correct orientation and position. The user may then manipulate the object if desired. Image plane techniques are inappropriate for the rotation of an object about the Z (heading) axis when in immersive view; therefore the top down map is used instead. If an object is not the correct size, it can be easily scaled to size using two handed manipulation techniques.

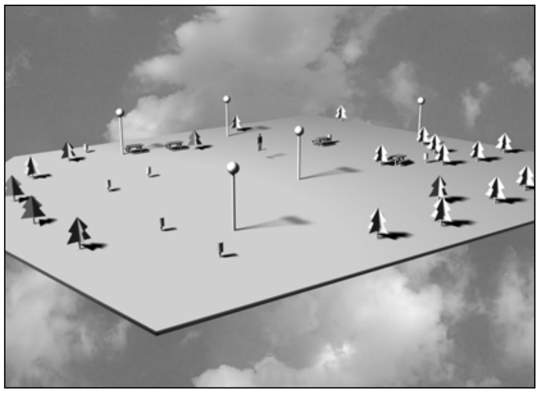

Figure 3 – Ray traced Tinmith-Metro captured street furniture model

Results

After the previous process is complete, the user has captured a model depicting the grass area, with various items of street furniture laid out on top. This model can be viewed immersively in the system, or displayed on a separate VR or desktop system indoors, as ray traced in Figure 3.

5. User interface

Traditional desktop input devices such as keyboards and mice are not suitable for working with augmented reality in a mobile outdoor environment. As a result, there is a need for new forms of user input devices and portable tracking devices to support these new input devices. As previously mentioned, the user interface Tinmith-Hand is based on gloves worn by the user, with vision based tracking, as an appropriate means to allow the user to manipulate 3D objects, while at the same time selecting menu options. Tinmith-Hand may be used for a wide variety of wearable AR applications, not just Tinmith-Metro.

Menu system and gloves

Since the modelling system supports such a wide variety of commands, it is not practical to make them all available at any given time. An eight item menu is placed on the bottom of the user's display to navigate the options and select the actions required. The items are logically mapped left to right, associated with the eight fingers of the user. The user makes pinching gestures between the finger and thumb to select the desired option. The touching of any finger to the palm of the hand returns the menu to its top level. Note that the tracking of the hands and head is not needed for this, as menu options can be selected at any time, and are always visible at the bottom of the display. All of the commands mentioned in the previous examples were executed via the glove and menu, and no keyboard or track ball (mouse) interaction was required.

Object selection and transformation

In order to manipulate objects, the user must be able to select objects and modify them. The system supports a number of cursors for pointing, one handed tracking, two handed tracking, centre of the display eye cursor, or a side worn hand-held track ball. Selection is activated when the user pinches the glove to trigger a menu option, the system then fires a perpendicular ray from the cursor on the display into the world, and intersects it with the first available facet. Using the menu, the user can move up the object hierarchy to select ancestor objects of the facet, if desired. It is not possible to select objects that are occluded by others using the immersive view. However, by switching to the top down map, (or any other external camera view) it is possible to select objects that would be otherwise hidden.

Selection may be a slow process especially when dealing with large quantities of individual objects, and so to cache these selections for later use, the system supports multiple selection buffers (ie, clip boards) and the ability to switch between them at any time. Objects in active selection buffers can then be manipulated.

Once selected, objects may be manipulated using image plane techniques. Image plane techniques treat the graphical objects as flat 2D objects, and map the 2D movement of the cursor into 3D transformations of the objects, while keeping the object attached to the cursor. Therefore, it is not possible to alter objects in directions the user is facing.

Tinmith-Hand may use any of the four cursors to perform image plane transformations on objects, although the two handed input gloves are the main method envisaged. Other previous work such as [2, 7, 19] also used two handed techniques to work in 3D environments. We have found rotation manipulation particularly well supported by two-handed interaction techniques. By using two hands, the user can specify the orientation to apply based on the angle the hands form, and scale an object based on the distance the hands are apart from each other. To move an object, it is slaved to sit underneath the cursor, and the user moves the cursor to the new location. It should be noted that during these operations, the actual locations of the hands is not important, just the projection of the cursor that is drawn on to the display.

6. CSG modelling system

The CSG engine that forms the core of the modelling system allows the construction of complex shapes using only simple input primitives such as infinite planes, boxes, and objects that can be described using an equation. The implementation is based on the same concepts as that used in ray tracers, such as POV-Ray [14].

Every primitive has the notion of an inside and outside region, defined by the direction of the surface normal(s). With this concept, it is possible to test if another object is partly or fully inside or outside a second object. The three fundamental CSG operations are shown in Figure 4.

Solid objects have a well defined 'inside', but infinite planes do not, as they extend on to infinity. As a result, the front and back faces are used to divide the world into two spaces, inside and outside. Hence, if 6 perpendicular planes are arranged correctly and intersected, they will form a new object that is a closed finite 3D box.

Using infinite planes, only a convex shape can be built with a single CSG operation – an object in which there are no holes or indentations. Buildings with indented windows, or T, L, and donut shapes cannot be modelled using a single CSG operation, since the planes would cancel each

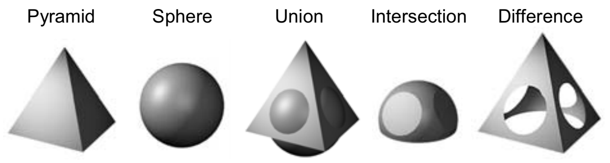

Figure 4 - CSG Primitive Operations

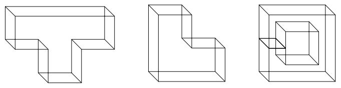
Figure 5 - Concave T, L, and Donut Shaped Buildings

other out. Figure 5 shows some examples of buildings that cannot be modelled using a single infinite plane CSG intersection.

However, by breaking the problem down into stages, a concave building may be created. The user first creates a bounding box for the entire object, and then creates a new box using the same techniques for the part of the building that is missing. Subtracting one part from the other produces a building that is concave shaped. The same techniques are applied to carve out doors, tunnels, and windows (as in our example).

As a result, using just infinite planes, the final example model shown in Figure 1 was created, demonstrating the true power of CSG operations. With the addition of other primitives like spheres, cones, and polygons, more complex shapes may also be constructed.

CSG Engine Implementation

The CSG engine operates using a set of user selected objects, which the system breaks down into facets. The objects are subdivided relative to each other, so that each polygon is used to cut every other polygon in half. This new complex mesh (which can be input^2 in size) is then processed, deleting facets that do not fit the set operation. The subdivision is then reversed to recover facets that were not altered, and the result is then updated on the display interactively so the user can view the results. This method is computationally expensive, although runs at interactive frame rates when creating building models. Given highly complex objects with thousands or more of facets arranged at many angles, the process can slow to the point of being unusable. Other techniques for solving CSG operations include using rendering hardware, ray tracers, or voxels, but there are a number of problems with each of these methods that make them unsuitable for our task.

7. Tinmith system

The Tinmith-Metro system, and the user interface Tinmith-Hand, were built on top of an architecture we have created for the development of augmented reality applications. The system, known as Tinmith, is a complete custom software system, as well as various off the shelf and custom built hardware components.

Hardware

The wearable computer system as shown in Figure 6 is based on a Gateway Solo P2-450 laptop (64 mb RAM, ATI Rage 3D) mounted on a hiking backpack. An Intersense IS-300 tracker is used for orientation sensing. Position information is gained from a Garmin 12XL GPS with DGPS, with an accuracy of 1-5 metres depending on conditions. The display is a Sony Glasstron PLM-700e monocular SVGA display. A large 12V battery powers the various trackers, as well as the small LCD television on the back for debugging, used for spectators to view. A SuperCam WonderEye USB video camera is used to provide images for the hand tracking system.

The laptop runs RedHat Linux 7.0 with kernel 2.4 as its operating system, including the standard GNU development environment. XFree86 v3.3.6 is used for graphics, as it does hardware accelerated OpenGL using Utah-GLX.

Pinch gloves

In order to control the menu system, we use a pair of custom designed pinch gloves (shown in (1) of Figure 6) as our main input device. These gloves are similar to a number of other products on the market, such as the FakeSpace PinchGlove [4]. The main difference with our gloves is that there are sensors on the palm as well as the thumb and fingers like the PinchGlove.

Our glove is a simple gardening glove, with special metallic tape attached which conducts electricity. This tape was attached to the finger tips, palm, and thumb, and wires were connected to a microcontroller. The microcontroller polls the fingers for new presses 30 times per second, and then sends events to the laptop via a serial cable. If a new event is received, the Tinmith-Hand user interface processes this new information to control the menus. Thousands of finger gesture combinations are possible with these gloves.

Tinmith evo5 software architecture

Although the architecture shares the same Tinmith name as previous systems (see [12, 18]), the current version, Tinmith evolution 5, is based on a completely new object oriented design, implemented in C++. The main goals were speed and efficiency, allowing us to write real world applications easily. Some brief details are presented in this paper.

The system is based around a concept of data flow, which is where information about the world (such as trackers and input devices) enters the system, is processed, and

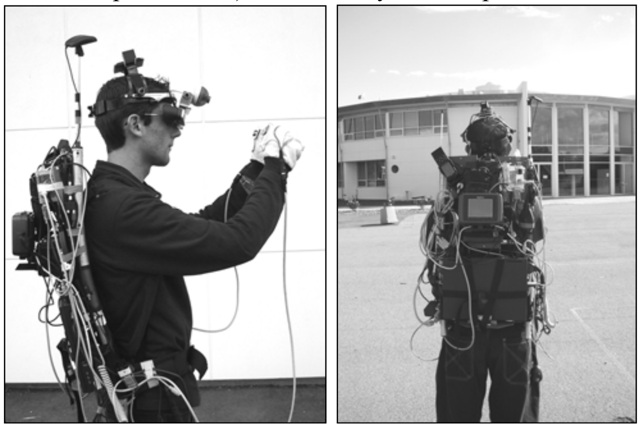
Figure 6 – (1) Tinmith wearable computer, with glove input devices
(2) Rear shot located near example school building, facing west

then the processed information is rendered to the HMD. Trackers and input devices are abstracted to specialised objects, which are then made available for other objects (known as listeners) to take values from. This is similar to an Observer/Observable pattern [6], and is implemented using a callback mechanism. When new tracker information enters the system, the values are propagated to all the listening objects. The display is then updated when the system becomes idle and all data values have been processed. Callbacks are performed using simple function calls, and no extra overheads are imposed from using threads, network transport, shared memory, or system calls. As a result, this ability to glue together objects makes writing the system simpler, while at the same time imposing little performance penalty on the CPU. However, network serialisation objects may be plugged in to allow the software to be distributed over multiple processes or machines (with an appropriate speed penalty).

The object oriented design includes an object store, which allows the storage of objects keyed by descriptive text labels and using a tree structure similar to a file system. The store also supports a special feature known as an object symbolic link, which allows the flow of data between objects to be redirected from different sources without the other objects being aware of these changes. Using this feature, we have implemented a patch board of tracking devices, allowing us to transparently change between them, combine devices, or use simulated devices without the application being made aware of this. This allows us to build the powerful user interaction techniques mentioned earlier.

The rendering system implemented in Tinmith is a full hierarchical modelling system, similar to SGI's Inventor [17], which contains a scene graph and transformation nodes, which can be directly linked to tracking devices. The CSG engine is tightly coupled into this system to allow complex interactive modelling operations. Each object in the scene graph is referenced in the object repository and contains its own unique reference path. The renderer reads a custom model format that is easily converted to primitive triangle lists or VRML files. As an added feature, we have included a 2 metre high human avatar, with 15 separate articulated parts, which is used to define the positions of the different tracking devices that are attached to the user. This allows us to easily resolve the coordinates of relative tracking devices such as the hands, and also to view the location of the user in the world from external camera views.

ARtoolkit image recognition

Tracking object movement outdoors cannot use standard virtual reality tracking systems. However, using fiducial markers on the hands, and the ARtoolkit system v2.33 [10], it is possible to implement a hand tracking solution that performs exceptionally well outdoors. The ARtoolkit reads video frames from a single USB camera, detects simple fiducial markers containing patterns, and resolves their position relative to the camera with a 6 DOF 4x4 transformation matrix. Frame rates of 5-10 fps (depending on camera compression) with 20% CPU utilisation is typical when using the system, so the performance is quite reasonable. The tracking works well outdoors, although the orientation jitters considerably (around 10°-20°), due to the low resolution of the cameras used. This is not a problem for our 2D cursors.

Two small 2 cm targets containing unique triangle patterns are placed onto the gloves' thumbs. The ARtoolkit libraries are built into a tracker class, passing on the position and orientation information to the scene graph, which calculates the real world coordinates based on the avatar's position and orientation.

Coordinate systems

Traditionally, navigation on a planet wide scale is done using latitude and longitude (LLH) spherical coordinates, and so GPS tracking uses this as its native coordinate system. However, on smaller scales, coordinate systems based in metres, such as UTM (Universal Transverse Mercator), are used by hikers and military personnel when using maps. In other specialised areas, ECEF coordinates (Earth centred XYZ) are used. As a result, tracking devices in the system support all of these coordinate systems [8], and can set and retrieve values with an automatic translation.

Since the renderer operates using metres, if the system operates far away enough from the UTM origin (which is typically millions of metres) then the centimetre level hand tracking and finely rendered objects distort. This is due to floating point resolution limitations, and to overcome this problem, the renderer uses a new coordinate system with a more local origin (based on UTM and within a few hundred kilometres), although this translation is completely transparent to the user and the model files generated.

8. Other application domains

Although the menus and prefabricated objects have been specially customised for two application domains supported by Tinmith-Metro, it should be remembered that this paper is presenting new AR methodologies. As a result, there is a wide range of application domains in areas such as the environmental, surveying, military, and architectural fields.

Although recording features such as city blocks, farm plots, and lakes is currently implemented, Tinmith-Metro does not easily support extruding 2D concave outlines into 3D solids, for the creation of features such as rivers. Currently, the user would have to construct a river as a series of convex shapes and join them together (since rivers tend to snake around, forming a concave shape). A better technique would be to use a technique we have devised known as 'the bread crumb trail', where the user walks along the perimeter of an object dropping markers at every turn, and then producing a solid shape from this outline. This technique only works if the user can be near the object; the infinite planes technique can work from any distance.

A second technique we would like to explore is using the ARtoolkit as the sole tracker, and deactivating the GPS. By placing a few fiducial markers on small objects that are below GPS resolution, we can rotate the small objects around, marking all the planes, perform a CSG operation and define 3D graphical objects as before.

Some limitations of the techniques are that large distant objects such as mountains would be difficult if not impossible to model. The system requires the user to be able to walk around the object (or in enough places to specify all the perimeter planes).

9. Conclusion

This paper has presented a new methodology for capturing outdoor city models and the placement of objects using wearable augmented reality systems. These new techniques have been implemented in a demonstration system known as Tinmith-Metro, based on the Tinmith evo5 software architecture. This demonstration system, which is executed on a wearable backpack computer with pinch gloves, has been used to construct example models of real world structures, as presented in this paper. A significant feature of this system is it allows the user to visually verify the object's accuracy at creation time, and allow others to view them on indoor VR or desktop systems in real-time, or at a later date.

10. Acknowledgments

The authors are very grateful for the work of Arron and Spishek Piekarski, who both helped in the design and construction of the glove and HMD, and to Leonard Teo for the rendered graphical models. This work was supported in part by grants from the Division of ITEE and the Defence Science Technology Organisation (DSTO).

11. References

[1] Anderson, D., Frankel, J. L., Marks, J., Leigh, D., Ryall, K., Sullivan, E., and Yedida, J. Building Virtual Structures With Physical Blocks. In *12th Int'l Symposium on User Interface Software and Technology,* pp 71-72, Asheville, NC, Nov, 1999.

[2] Bowman, D. A. and Hodges, L. F. An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments. *In 1997 Symposium on Interactive 3D Graphics*, pp 35-38, Providence, RI, Apr 1997.

[3] Debevec, P. E., Taylor, C. J., and Malik, J. Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach. *In 23rd Annual Conference on Computer Graphics*, pp 11-20, New Orleans, LA, Aug, 1996.

[4] FakeSpace Labs. *Pinch Gloves.* 2001. URL - www.fakespacelabs.com/products/pinch.html

[5] Feiner, S., MacIntyre, B., and Hollerer, T. A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment. *In 1st Int'l Symposium on Wearable Computers*, pp 74-81, Cambridge, Ma, Oct 1997.

[6] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software.* Reading, Ma, Addison Wesley Publishing Company, 1995.

[7] Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F. A Survey of Design Issues in Spatial Input. In *7th Int'l Symposium on User Interface Software Technology,* pp 213-222, Marina del Rey, Ca, Nov 1994.

[8] Intergovernmental Committee On Surveying and Mapping. *Geocentric Datum of Australia - Technical Manual.* URL - http://www.anzlic.org.au/icsm/gdatm/index.html

[9] Julier, S., Lanzagorta, M., Baillot, Y., Rosenblum, L., Feiner, S., and Hollerer, T. Information Filtering for Mobile Augmented Reality. In *3rd IEEE and ACM International Symposium on Augmented Reality,* pp 1-10, Munich, Germany, Oct 2000.

[10] Kato, H. and Billinghurst, M. Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System. In *2nd IEEE and ACM International Workshop on Augmented Reality*, pp 85-94, San Francisco, Ca, Oct 1999.

[11] Multigen. *SmartScene.* 2001. URL - http://www.multigen.com

[12] Piekarski, W., Gunther, B., and Thomas, B. Integrating Virtual and Augmented Realities in an Outdoor Application. In *2nd IEEE and ACM International Workshop on Augmented Reality*, pp 45-54, San Francisco, Ca, Oct 1999.

[13] Pierce, J., Forsberg, A., Conway, M., Hong, S., Zeleznik, R., and Mine, M. *Image Plane Interaction Techniques in 3D Immersive Environments.* In *1997 Symposium on Interactive 3D Graphics,* pp 39-43, Providence, RI, Apr 1997.

[14] POV-Team. *The Persistence Of Vision Raytracer.* URL - http://www.povray.org

[15] Raskar, R. Welch, G., Chen, W. Table-Top Spatially-Augmented Reality: Bringing Physical Models to Life with Projected Imagery. In *2nd IEEE and ACM International Workshop on Augmented Reality*. pp 64-71, San Francisco, Ca, Oct 1999.

[16] Sester, M., Brenner, C., and Haala, N. 3-D Virtual Cities and 3D Geospatial Information Systems. In *IMAGE2000 Workshop*, Ipswich, Qld, 2000.

[17] Strauss, P. R. IRIS Inventor, A 3D Graphics Toolkit. In *8th Annual Conference on Object-oriented Programming Systems*, pp 192-200, Washington, DC, October, 1993.

[18] Thomas, B. H., Demczuk, V., Piekarski, W., Hepworth, D., and Gunther, D. A Wearable Computer System With Augmented Reality to Support Terrestrial Navigation. In *2nd Int'l Symposium on Wearable Computers*, pp 168-171, Pittsburg, Pa, Oct 1998.

[19] Zeleznik, R. C., Forsberg, A. S., and Strauss, P. S. *Two Pointer Input For 3D Interaction.* In *1997 Symposium on Interactive 3D Graphics,* pp 115-120, Providence, RI, Apr 1997.