AUGMENTED REALITY ON GPS NAVIGATION (ARGPS)

Murugavel. KN[1]
[1](Asst. Professor, Department of Computer Science and Engineering,
Birla Institute of Technology-UAE Campus, CIIHE, Ras Al Khaimah, UAE)

Abstract

Augmented Reality is a new and upcoming field of technology aimed at providing information to the user via Optical methods. It consists of portable hardware and light software that enable the user to get extra information about his environment that would generally require a desktop computer or some form of a handheld device. Augmented Reality GPS Navigation is a conceptual project that is aimed at creating an Augmented Reality Interface and implementing a GPS based navigation application.

Keywords –Gult, Xact, Glew, Glee, Glaux, Ar.

I. INTRODUCTION

WHAT IS AUGUMENTED REALITY

Augmented reality (AR) is a term for a live direct or indirect view of a physical real-world environment whose elements are *augmented* by virtual computer-generated imagery. It is related to a more general concept called mediated reality in which a view of reality is modified (possibly even diminished rather than augmented) by a computer. As a result, the technology functions by enhancing one's current perception of reality. In the case of Augmented Reality, the augmentation is conventionally in real-time and in semantic context with environmental elements, such as sports scores on TV during a match. With the help of advanced AR technology (e.g. adding computer vision and object recognition) the information about the surrounding real world of the user becomes interactive and digitally usable. Artificial information about the environment and the objects in it can be stored and retrieved as an information layer on top of the real world view.

Augmented reality research explores the application of computer-generated imagery in live-video streams as a way to expand the real-world. Advanced research includes use of *Head Mounted Displays (HMD)*[2] and *Virtual Retinal Displays*[6] for visualization purposes, and construction of controlled environments containing *any number of sensors and actuators*.

II. USES OF AUGUMENTED REALITY

Augmented Reality is a group of technologies that provide the user with information about the environment via optical manipulation. It allows the user to have more accurate and detailed information about the object or scene

they are looking at. In a day where information exchange must be fast, accurate and easily available, augmented reality is crucial in delivering the said information in an easy, user friendly method without the need of a handheld device[8] or personal computer.

Potential uses of Augmented Reality include:

- **Advertising:** Marketers started to use AR to promote products via interactive AR applications. For example, at the 2008 LA Auto Show, Nissan unveiled the concept vehicle Cube and presented visitors with a brochure which, when held against a webcam, showed several versions of the vehicle. In August 2009, Best Buy ran a circular with an augmented reality code that allowed users with a webcam to interact with the product in 3D[7].

- **Support with complex tasks:** Complex tasks such as assembly, maintenance, and surgery can be simplified by inserting additional information into the field of view. For example, labels can be displayed on parts of a system to clarify operating instructions for a mechanic who is performing maintenance on the system.AR can include images of hidden objects, which can be particularly effective for medical diagnostics or surgery. Examples include a virtual X-ray view based on prior tomography or on real time images from ultrasound or open NMR devices[5]. A doctor could observe the fetus inside the mother's womb.

- **Navigation devices:** AR can augment the effectiveness of navigation devices for a variety of applications. For example, building navigation can be enhanced for the purpose of maintaining industrial plants. Outdoor navigation can be augmented for military operations or disaster management. Head-up displays or personal display glasses in automobiles can be used to provide navigation hints and traffic information. These types of displays can be useful for airplane pilots, too. Head-up displays are currently used in fighter jets as one of the first AR applications. These include full interactivity, including eye pointing.

- **Industrial Applications:** AR can be used to compare the data of digital mock-ups with physical mock-ups for efficiently finding discrepancies between the two sources. It can further be employed to safeguard digital data in combination with existing real prototypes, and thus save or minimize the building of real prototypes and improve the quality of the final product.

- **Military and emergency services:** AR can be applied to military and emergency services as wearable systems to provide information such as instructions, maps, enemy locations, and fire cells.

- **Prospecting:** In the fields of hydrology, ecology, and geology, AR can be used to display an interactive analysis of terrain characteristics. Users could use, and collaboratively modify and analyze, interactive three-dimensional maps.

- **Art:** AR can be incorporated into artistic applications that allow artists to create art in real time over reality such as painting, drawing, modeling, etc. One such example of this phenomenon is called Eye-writer that was developed in 2009 by Zachary Lieberman and a group formed by members of Free Art and Technology (FAT), Open Frameworks and the Graffiti Research Lab to help a graffiti artist, who became paralyzed, draw again.

- **Architecture:** AR can be employed to simulate planned construction projects.

- **Sightseeing:** Models may be created to include labels or text related to the objects/places visited. With AR, users can rebuild ruins, buildings, or even landscapes as they previously existed.

- **Collaboration:** AR can help facilitate collaboration among distributed team members via conferences with real and virtual participants. The Hand of God is a good example of a collaboration system.

- **Entertainment and education:** AR can be used in the fields of entertainment and education to create virtual objects in museums and exhibitions, theme park attractions (such as Cadbury World), and games (such as ARQuake[9] and The Eye of Judgment).

- **Music:** Pop group Duran included interactive AR projections into their stage show during their 2000 *Pop Trash* concert tour. Sydney band Lost Valentines launched the world's first interactive AR music video on 16 October 2009, where users could print out 5 markers representing a pre-recorded performance from each band member which they could interact with live and in real-time via their computer webcam and record as their own unique music video clips to share via YouTube.

III. FUTURE POSSIBILITES

Expanding a PC screen into the real environment: program windows and icons appear as virtual devices in real space and are eye or gesture operated[4], by gazing or pointing. A single personal display (glasses) could concurrently simulate a hundred conventional PC screens or application windows all around a user.

Virtual devices of all kinds, e.g. replacement of traditional screens, control panels, and entirely new applications impossible in "real" hardware, like 3D objects interactively changing their shape and appearance based on the current task or need[3].

Enhanced media applications, like pseudo holographic virtual screens, virtual surround cinema, virtual 'holodecks' (allowing computer-generated imagery to interact with live entertainers and audience)

Replacement of cell phone and car navigator screens: eye-dialing, insertion of information directly into the environment, e.g. guiding lines directly on the road, as well as enhancements like "X-ray"-views[5].

Virtual plants, wallpapers, panoramic views, artwork, decorations, illumination etc., enhancing everyday life. For example, a virtual window could be displayed on a regular wall showing a live feed of a camera placed on the exterior of the building, thus allowing the user to effectually toggle a wall's transparency[6].

With AR systems getting into mass market, we may see virtual window dressings, posters, traffic signs, Christmas decorations, advertisement towers and more. These may be fully interactive even at a distance, by eye pointing for example.

Virtual gadgetry becomes possible. Any physical device currently produced to assist in data-oriented tasks (such as the clock, radio, PC, arrival/departure board at an airport, stock ticker, PDA, PMP, informational posters/fliers/billboards, in-car navigation systems, etc.) could be replaced by virtual devices that cost nothing to produce aside from the cost of writing the software. Examples might be a virtual wall clock, a to-do list for the day docked by your bed for you to look at first thing in the morning, etc.

Subscribable group-specific AR feeds. For example, a manager on a construction site could create and dock instructions including diagrams in specific locations on the site. The workers could refer to this feed of AR items as they work. Another example could be patrons at a public event subscribing to a feed of direction and information oriented AR items.

AR systems can help the visually impaired navigate in a much better manner (combined with a text-to-speech software).

Computer games which make use of position and environment information to place virtual objects, opponents, and weapons overlaid in the player's visual field.

IV. DESIGN

GLUT simplifies the implementation of programs using OpenGL rendering. The GLUT application programming interface (API) requires very few routines to display a graphics scene rendered using OpenGL. The GLUT API (like the OpenGL API) is stateful. Most initial GLUT state is defined and the initial state is reasonable for simple programs. The GLUT routines also take relatively few parameters. No pointers are returned. The only pointers passed into GLUT are pointers to character strings (all strings passed to GLUT are copied, not referenced) and opaque font handles.

The GLUT API is (as much as reasonable) window system independent. For this reason, GLUT does not return *any* native window system handles, pointers, or other data structures. More subtle window system dependencies such as reliance on window system dependent fonts are avoided by GLUT; instead, GLUT supplies its own (limited) set of fonts. For programming ease, GLUT provides a simple menu sub-API. While the menu support is designed to be implemented as pop-up menus, GLUT gives window system leeway to support the menu functionality in another manner (pull-down menus for example).

Two of the most important pieces of GLUT state are the *current window* and *current menu*. Most window and menu routines affect the *current window* or *menu* respectively. Most callbacks implicitly set the *current window* and *menu* to the appropriate window or menu responsible for the callback. GLUT is designed so that a program with only a single window and/or menu will not need to keep track of any window or menu identifiers. This greatly simplifies very simple GLUT programs.

GLUT is designed for simple to moderately complex programs focused on OpenGL rendering. GLUT implements its own event loop. For this reason, mixing GLUT with other APIs that demand their own event handling structure may be difficult. The advantage of a built in event dispatch loop is simplicity.

GLUT contains routines for rendering fonts and geometric objects; however GLUT makes no claims on the OpenGL display list name space. For this reason, none of the GLUT rendering routines use OpenGL display lists. It is up to the GLUT programmer to compile the output from GLUT rendering routines into display lists if this is desired.

GLUT routines are logically organized into several sub-APIs according to their functionality.

The sub-APIs are

Initialization: Command line processing, window system initialization, and initial window creation state are controlled by these routines.

Beginning Event Processing: This routine enters GLUT's event processing loop. This routine never returns, and it continuously calls GLUT callbacks as necessary.

Window Management: These routines create and control windows.

Overlay Management: These routines establish and manage overlays for windows.

Menu Management: These routines create and control pop-up menus.

Callback Registration: These routines register callbacks to be called by the GLUT event processing loop.

Color Index Color-map Management: These routines allow the manipulation of color index color-maps for windows.

State Retrieval: These routines allow programs to retrieve state from GLUT.

Font Rendering: These routines allow rendering of stroke and bitmap fonts.

Geometric Shape Rendering: These routines allow the rendering of 3D geometric objects including spheres, cones, icosahedrons, and teapots[3].

V. ARTOOLKIT

ARToolKit is a C and C++ language software library that lets programmers easily develop Augmented Reality applications. Augmented Reality (AR) is the overlay of virtual computer graphics images on the real world, and has many potential applications in industrial and academic research.

One of the most difficult parts of developing an Augmented Reality application is precisely calculating the user's viewpoint in real time so that the virtual images are exactly aligned with real world objects. ARToolKit uses computer vision techniques to calculate the real camera position and orientation relative to marked cards, allowing the programmer to overlay virtual objects onto these cards. The fast, precise tracking provided by ARToolKit should enable the rapid development of many new and interesting AR applications[1].

ARToolKit is a software *ToolKit* like GLUT. It furnishes predefined functions that you need to call in a specific order for developing an AR program. But you can also use different parts of the ToolKit separately. ARToolKit[1] supports multiple platforms, while attempting to minimise library dependencies without sacrificing efficiency. ARToolKit uses OpenGL for the rendering part, GLUT for the windows/event handler aspect and hardware-dependent video library and standard API on each platform (e.g. win32 in Windows).

Figure.1 summarizes the relationship between Application, ARToolKit and dependent libraries.

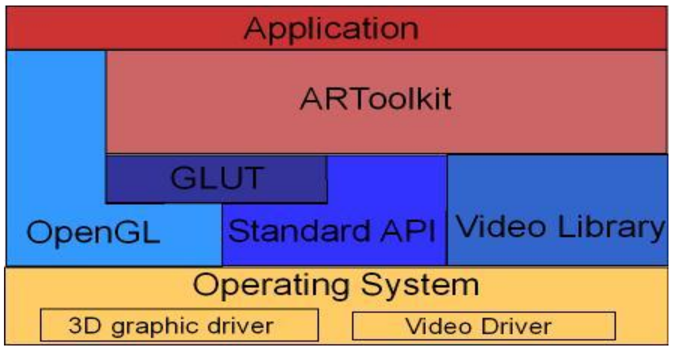


Figure 1: ARToolKit Architecture

VI. STRUCTURE

This section provides a better description of the main elements of ARToolKit[2].

The ARToolKit library consists of four modules:

- **AR module:** core module with marker tracking routines, calibration and parameter collection.

- **Video module:** A collection of video routines for capturing the video input frames. This is a wrapper around the standard platform SDK video capture routines.

- **Gsub module:** A collection of graphic routines based on the OpenGL and GLUT libraries.

- **Gsub_Lite module:** Replaces GSub with a more efficient collection of graphics routines, independent of any particular windowing toolkit.

The next figures show the hierarchical structure of ARToolKit and relation with dependencies libraries.

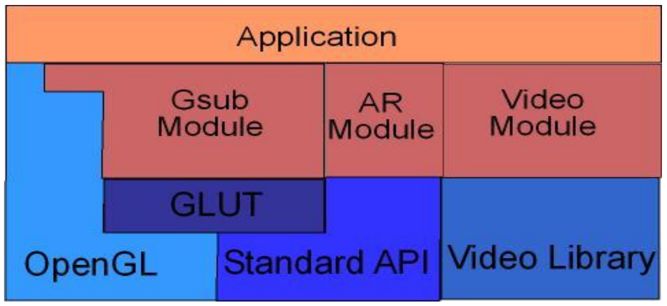


Figure 2: Hierarchical structure of ARToolKit using Gsub Module

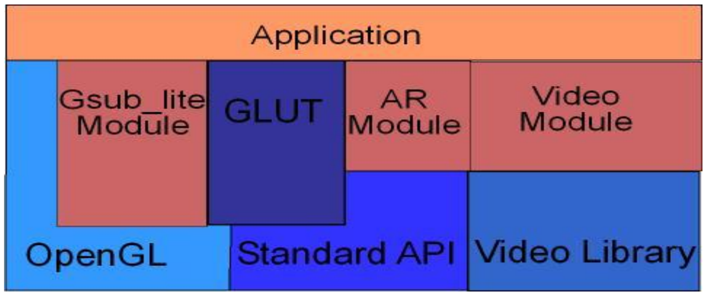


Figure 3: Hierarchical structure of ARToolKit using Gsub_Lite Module

The modules respect a global *pipeline metaphor* (video->tracking->display), so the user can easily replace any module with another (like gsub with Open Inventor renderer).

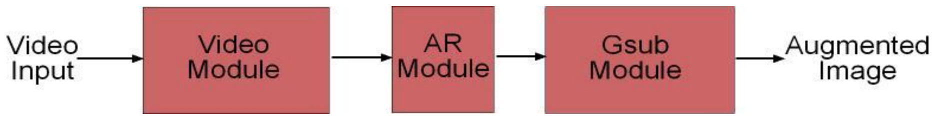


Figure 4: Main ARToolKit pipeline

VII. Data Types

ARToolKit manipulates a lot of different kinds of variable. Internally, it uses global variables that restrict re-entrant part of the code. Otherwise, standard multi-argument interface are used based on a data-flow approach.

ARToolKit uses different image formats between different modules. Figure 4 summarises all the different formats supported. Some formats are only available on certain platforms or with certain hardware.

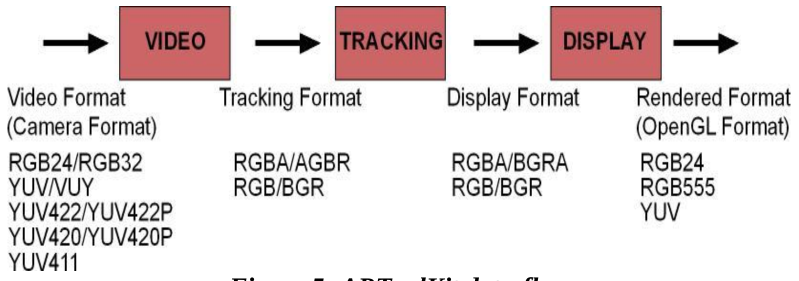
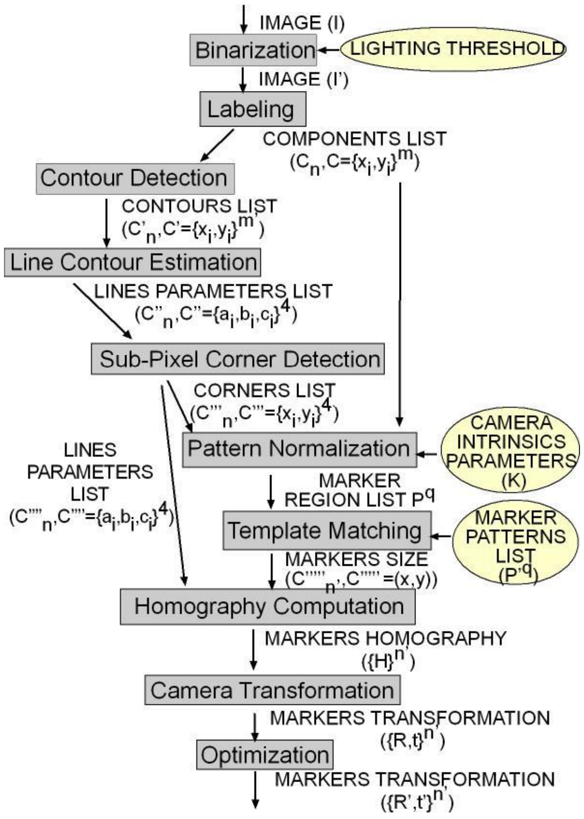
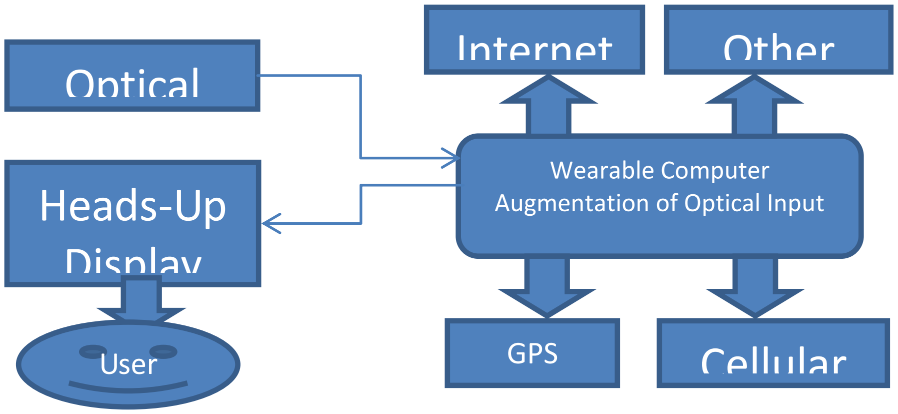


Figure 5: ARToolKit data-flow

VIII. Pattern Recognition Algorithm



IX. The Working Of The Application



The Video module consists of the DirectX webcam library which is implemented by ARToolkit. The application opens a video stream to the webcam and captures video frame-by-frame and also displays it on the HUD. Each frame gets searched by the application for a pattern. If a pattern is present, it gets recognized and the display function associated to the pattern is called.

The GUI module is responsible for displaying all data to the HUD. It consists of OpenGL functions that enable creation of virtual objects as well as modification of the video window

The GPS device gets the co-ordinates and sends it to the application. Here, the co-ordinates are displayed on the screen as well as sent to the MAP application.

The Map application is an independent application utilizing C++ and Developers Image Library (DevIL). It gets the GPS co-ordinates and displays it on a map loaded from the hard disk.

X. LIST OF ABBREVIATIONS

- NMEA: National Marine Electronics Association.
- XACT: XInput and the Cross-platform Audio Creation Tool
- ARB: Architecture Review Board
- GLEW: OpenGL Extension Wrangler Library
- GLEE: OpenGL Easy Extension Library
- GLAux: OpenGL Auxiliary Library
- GLUT: OpenGL Utility Toolkit

XI. CONCLUSION

During the course of the project we have learned how to use an integrated developer environment (IDE) to create a standalone application. The OpenGL library, a major graphical library used in the mainstream market to create graphical applications for all sorts of platforms was successfully utilized to create the Heads up display interface as well as to display the video captured from the webcam. Pattern recognition was successfully implemented using ARToolkit's native library functions. The theory behind it was examined and understood. A new pattern was created and successfully recognized by the software. GPS was interfaced using a COM port, but due to closed source restrictions of certain libraries, we were unable to utilize it in the application. Source code using the library has been added to the application but it is defunctionate due to the library restrictions. The Map module has been successfully created and implemented as an independent application and utilizes the OpenGL and DevIL image libraries.

REFERENCES

Journal Papers:

[1] ARToolkit. http://www.hitl.washington.edu/research/ zhared_space

[2] Fuchs, Henry and Jeremy Ackerman. "Displays for Augmented Reality: Historical Remarks and Future Prospects." In *Proc. International Symposium on Mixed Reality*, pp.31-41, Yokohama, Japan, March 9-11, 1999.

[3] [36]T. Kanade, et. al, "Virtualized reality: digitizing a 3D time-varying event as is and in real time," *Proc. Int'l Symp. Mixed Reality (ISMR '99). Mixed Reality-Merging Real and Virtual Worlds*, Yokohama, Japan, 9-11 Mar. 1999, pp. 41-57.

[4] [10] Murray, Janet. *Hamlet on the Holodeck: The Future of Narrative in Cyberspace.* New York: Simon and Schuster, 1997.

[5] [53] N. Navab, A. Bani-Hashem, M. Mitschke. "Merging Visible and Invisible: Two Camera-Augmented Mobile C-arm (CAMC) Applications," *Proc. 2nd Int'l Workshop Augmented Reality. (IWAR '99).* San Francisco, 20-21 Oct. 1999, pp. 134-141.

[6] [62]H.L. Pryor, T.A. Furness, E. Viirre. "The Virtual Retinal Display: A New Display Technology Using Scanned Laser Light," *Proc. 42nd Human Factors Ergonomics Society.* Chicago, 5-9 Oct. 1998, pp. 1570-1574.

[7] [64]R. Raskar, G. Welch, W-C. Chen, "Table-top spatially-augmented realty: Bringing physical models to life with projected imagery," *Proc. 2nd Int'l Workshop Augmented Reality. (IWAR '99).* San Francisco, 20-21 Oct. 1999, pp. 64-71.

[8] J. Rekimoto, "NaviCam: A Magnifying Glass Approach to Augmented Reality," *Presence: Teleoperators and Virtual Environments*, vol. 6, no. 4, Aug. 1997, pp. 399-412

[9] [85]B. Thomas, et. al., "ARQuake: An Outdoor/Indoor Augmented Reality First Person Application," *Proc. 4th Int'l Symp. Wearable Computers. (ISWC 2000).* Atlanta, 16-17 Oct. 2000, pp. 139-146.