Senior Project Report

Cory Klein, William Lee, Lorenzo Swank, Michael Yang

***Abstract*—This project consists of glasses worn to create the perception of a three-dimensional view of a ray-traced scene in the user's mind. The glasses in this project utilize shutter stereoscopy methods. The glasses are made with liquid crystal displays (LCD). The project consists of three core components. The first is a program sends signals to the shutter LCDs and displays images from the raytracer on a monitor. The second part is an amplifier stage. The amplifiers increase the voltage of the synchronization signal. The third part is the LCD shutter glasses. The pair of glasses shutter each eye at a certain frequency to present a pseudo-3D image. Using this method, a user can see a 3D object from a 2D display.**

I. Motivation

There are several 3D glasses on the market already such as 3D Vision Glasses by NVIDIA and a similar product by 3Dstereo, Inc. NVIDIA's glasses work with a variety of DirectX games and 3D movies. Their product is by far the most popular product in the field. They present clear vision and depth perception effects very well. In order to use NVIDIA's glasses, users need to have a computer running Windows Vista or Windows 7. In addition, the glasses require an NVIDIA GeForce 8500 or higher graphics card.[1]

The other major product on the market now is 3D image viewing goggles from 3Dstereo, Inc. These 3D goggles are similar to what is presented in this paper. The 3D image viewing goggles support both cathode ray tube (CRT) and LCD monitors. The goggles work with a refresh rate of at least 70 Hz. 3Dstereo's solution supports Windows 95 or higher and supports most of the graphics cards in the market now.[2]

NVIDIA 3D vision requires high-end graphics cards and Windows Vista or Windows 7 as the operating system. The glasses themselves cost US$200. The user also needs to have a dual-core processor. The video cards supported are only available on desktop computers. Moreover, the glasses only work with a 120 Hz refresh rate LCD display. As the list goes on, costs of implementation continue to skyrocket. The cost is too much for a typical family. In other words, the cost and the computer requirement for using 3D vision goggles limits its market share. On the other hand, 3D image viewing goggles from 3Dstereo cost just over US$100. They can run on most Windows-based machines and most LCD displays. Most NVIDIA graphics cards and ATI graphics cards can work with these glasses. However, the price of this solution is still considered expensive for typical users.

One of the motivating factors for this project is to determine the minimum requirements in hardware, software, and cost to accomplish good-quality stereo vision with the computer.

II. Functionality

The idea which presents in this project to achieve 3D vision is stereoscopy method. Although there are several methods to present 3D vision such as autostereograms or anaglyphs, we choose stereoscopy because this method does not effect the original images but the other two methods would. Basically, the way stereoscopy works is by presenting correct side of image to the correct eye. The glasses block the right eye while the screen display the left eye image and vice-versa. In addition, the frequency of switching images have to be 60 Hz or more in order to present clear 3D vision.

III. Hardware Components

There are three separate hardware components that make up the system, excluding the computer displaying the image. These components are the USB Microcontroller (which is marketed and sold under the trade name "USB Bit Whacker"), the signal converting circuit, and the shutter glasses.

The USB Bit Whacker has the responsibility of transferring a signal from the software to the power circuit for signal modification. The USB Bit Whacker connects to the PC via USB, and pre-compiled drivers create a serial port in the OS with which to communicate to the USB Bit Whacker.

The USB Bit Whacker is actually a PIC18F2553 USB Microcontroller, soldered onto a board with some LEDs used for debuging, with fourteen pins in addition to ground, power, and RS232 RX/TX lines. The microcontroller is programmable via USB, but the USB Bit Whacker distributer, SparkFun Electronics, preloads the USB Bit Whacker with functional firmware for basic control of the output pins over a serial connection. On a FreeBSD based Unix system, the bit whacker appears as cu.usbmodem411 in the /dev directory.

This preloaded firmware proved to be sufficient for the needs of this project. The USB Bit Whacker provides two alternating square wave signals in sync with the image on the screen. At first the team was worried that the delay caused by serial communication and serial command decoding would prove too much, and the ability to create a phase change between switching the image on the screen and sending serial commands would be required. This fear proved to be unfounded. Once software-controlled commands were successfully sent and the square wave was detected on the oscilloscope, the speed limits of the communication were tested. We found that we can send over 800 commands per second without any commands being dropped, effectively generating an 800 Hz square wave. As the USB Bit Whacker would never be required to create a signal greater than 60Hz, any sort of phase shifting was unnecessary.

In the final implementation, a single serial command was sent that toggled two separate alternating signals synchronously. This caused the signal for the left eye to switch ON at the exact same time as the right eye switched OFF. These two signals, along with the pin for ground were pulled off the board and used by the two stage amplifier. This amplifier contains two operational amplifiers which produce a negative unity gain amplifier in the first stage and a gain of negative five amplifier as the second stage. Because the output signal from USB is a square wave with the amplitude of 0v-5v, we need the amplifier with offset to achieve the required voltage difference which is amplitude of 3v-25v. The design of the amplifier is an inverting amplifier with some non-inverting input. The mix of inverting gain and non-inverting gain can amplify 5v to 25v and 0v to 3v in order to fulfill the requirement of LCD shutter glasses which turns the LCD all the way black or all the way transparent.

We choose Pi-cell LCDs because they are able to switch on and off at a faster rate. In general, LCDs are turned on and off by applying a voltage to them. When no voltage is applied, the liquid crystal molecules in the LCDs are aligned in a certain way to allow light pass through the LCDs. However, when enough voltage is applied, the alignment of liquid crystal molecules in the LCDs are changed so that lights can not pass through. The liquid crystal molecules in the Pi-cell LCDs are aligned in a special way so they are able to changed at a faster rate than other LEDs. The voltage we need to make them transparent and black are 3 and 25 volts respectively.

We built an amplifier in order to pump the voltage out from the USB to 25 volts. It is built by connecting multiple TC7660 charge pumps together. It works when we input 5 volts from the voltage supply. But when it is connected to the USB it does not pump the voltage up to 25 volts. That was because the USB does not provide enough current. To solve this problem, we will need to find a way to get more current.

IV. Software Components

There are three software components that are used. The first is the raytracer, the second is the front end for the ray tracer, and the third is the display program.

Raytracing is a method of generating computer graphics by which rays are sent from a camera through every pixel in the image into a scene. These rays bounce around from object to object. For each object that the ray intersects, the ray inherits some properties from that object. These properties determine what color is assigned to the pixel through which the ray entered the scene. If the ray intersects a light source, it will be brighter. If the ray intersects a region which does not have a path to a light source, it inherits only those ambient light characteristics of the scene, and is thus in shadow. If a ray intersects a perfect mirror, it does not inherit any color characteristics from that object, but merely bounces off of the object. This is done recursively.

The raytracer was written over a period of several weeks, with only minor refinements made in the later stages of the project. Scene files are from a course on raytracing taught by Peter Shirley and Steven Parker.

The front end for the ray tracer consists of mechanisms to set up dual cameras appropriately in a scene. First, a scene file is read in. An instance of the raytracer is instantiated for the scene file. A bounding volume hierarchy is constructed to serve as the first level acceleration structure for the raytracer. The primary camera is saved as a reference camera and popped off of the camera stack. Two cameras are then created with parallel target vectors, spaced an equal distance from the reference camera origin along the right-direction vector. These cameras are then pushed onto the raytracer camera stack.

The front end software allows the user to modify the distance between the cameras. Since there is not a standard unit of measurement in the ray tracing community, not every scene file will have the same scale. Each scene file may require the camera distance to be slightly different. Building for the common case means exposing this functionality in the easiest way possible. Consequently, this is one of the few features that is exposed as a graphical user interface element.

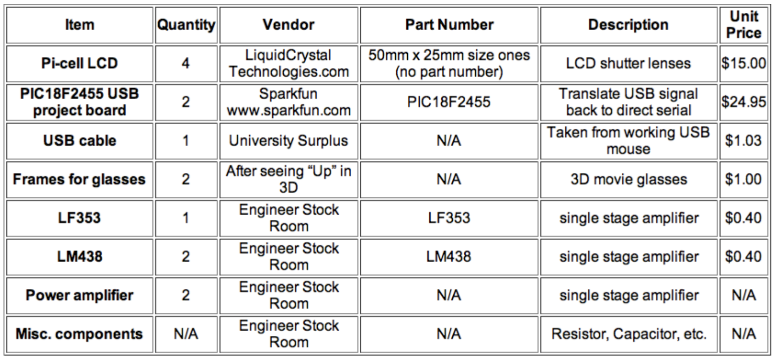
The third component is the display program. The display program initializes a serial/USB interface, loads images from disk, and displays images while sending synchronization signals to the serial port.

V. Problems Encountered

Establishing initial communications with the USB Bit Whacker were problematic. Most devices that communicate over an RS-232 interface require a complex system of handshakes. Although we could use a pre-existing serial communications program such as Screen in Unix, or HyperTerminal in Windows, to manually communicate with the device, getting our own program to connect to the device and perform the proper handshake was difficult. The board didn't seem to respond to any sort of handshake request that was tried. Detailed instructions concerning the preloaded firmware, and custom programming of the microcontroller were abundant, but documentation covering the actual RS-232 communication standards was sparse. We gathered from online forums that the communication speed was 9600 baud, but little else could be found. Eventually we found, by trial and error, that the bit whacker did not require any handshaking at all, which seems to be extremely uncommon for serial communication. At this point, however, communication with the USB Bit Whacker became simple.

The single greatest difficulty that plagued the project for the majority of development was our perceived inability to make the LCDs get dark enough in a short period of time. Complete redesigns of the circuit were made over and over until we switched displays. Switching from one LCD to another LCD exposed the root of the problem. Since both the LCD shutters and LCD on our notebook computers are polarized arbitrarily with respect to one another, it was impossible for us to get light from the source to be blocked completely by the receiving LCD shutter. Switching to a CRT monitor allowed us to sidestep the issue.

VI. Bill of Materials



Item	Quantity	Vendor	Part Number	Description	Unit Price
Pi-cell LCD	4	LiquidCrystal Technologies.com	50mm x 25mm size ones (no part number)	LCD shutter lenses	$15.00
PIC18F2455 USB project board	2	Sparkfun www.sparkfun.com	PIC18F2455	Translate USB signal back to direct serial	$24.95
USB cable	1	University Surplus	N/A	Taken from working USB mouse	$1.03
Frames for glasses	2	After seeing “Up” in 3D	N/A	3D movie glasses	$1.00
LF353	1	Engineer Stock Room	LF353	single stage amplifier	$0.40
LM438	2	Engineer Stock Room	LM438	single stage amplifier	$0.40
Power amplifier	2	Engineer Stock Room	N/A	single stage amplifier	N/A
Misc. components	N/A	Engineer Stock Room	N/A	Resistor, Capacitor, etc.	N/A

VII. Future Work

In the demonstration, prerendered frames were displayed. One of the first improvements to the software would be to allow real time scene rendering. This would allow the user to manipulate the image on the screen and see the changes in real time. Optimizations would be required, and the implementation would likely have to be scaled back to produce an image with a framerate sufficiently fast enough.

Although the tearing that was present during the demo did not prevent users from viewing the image in 3 dimensions, it did significantly reduce the quality of the image. This problem, however, seems to be rather difficult to solve. Double buffered rendering was used, which should eliminate tearing, but it didn't. Also, the tearing may have been aggravated by harmonic oscillations with the vertical trace on the CRT monitor. It is conceivable that either a hardware or software solution may be possible, with preference toward the software solution, which would allow the system to work for any reasonably recent piece of hardware.

One major problem of this project is that the 3D visualization can only be viewed when we display image on the CRT monitor. The issue happens because the light from LCD monitor is polarized and from CRT monitor is not. When we view the image through glasses to LCD monitor, the LCD glasses are not completely off when signal is high. This problem can be solved by adding a new layer of polarization filter to the glasses so that all the light can be shuttered.

The great majority of the code written for the project was not platform specific. Specific implementations were catered towards the Mac OS, but porting the system to Windows or Linux should be relatively pain free. This may also be beneficial in solving the tearing problem, as there were suspicions that the problem was related to the OS graphics calls or libraries. One platform that has great potential is the mobile platform. Developers were recently given access to hardware control on the popular iPhone. A small attachment could be produced that allowed a user to plug in 3d shutter glasses into their iPhone and play games or view graphical visualizations in 3d on a mobile platform.

VIII. Acknowledgements

Special thanks to Al Davis for his fearless leadership, undying support, and for looking like the King of Rohan from Lord of the Rings.

William would like to thank Reid Harrison explanation of the offset amplifier.

Lorenzo would like to thank his parents for giving him life, and William's parents, for letting him eat their food, sleep in their house, use their , and kidnap their child while writing this report.

References

[1] NVIDIA, 3D vision, www.nvidia.com
[2] 3Dstereo.com, inc., “3D image viewing”, www.3Dstereo.com