CCSC Consortium for Computing Sciences in Colleges

Southeastern Region

28th Annual Southeastern Conference

Student Research Contest

Extended Abstracts

November 7 and 8, 2014
College of Charleston
Charleston, South Carolina

Table of Contents

Summarized Patient Information with Google Glass (SPIGLASS)

Chad Reynolds, Neea Rusch
Faculty Advisor: Dr. Michael Dowell
Georgia Regents University

Background

New wearable tech is being released nearly monthly to the consumer marketplace. These computing devices range from smart watches to virtual reality headsets. The marketing and advertising for these products target personal use as their main function. But the scope of these devices includes the workplace as well, with unique restrictions on I/O forcing users to step away from traditional methods of interacting. Along with the capacity to connect to wireless networks and tether to mobile phones, Google Glass is a versatile and portable device.

Project

The SPIGLASS project explored different types of applications that focus on the medical workplace domain. The different versions of the app provide doctors and medical staff with necessary patient data on the conveniently head-mounted display. Users can cycle between different summary screens of multiple patients. Navigation within the application is accomplished by utilizing the microphone through voice commands, speaking to bring up a context-dependent menu and then allowing the user to choose a menu item to display the next screen. As a backup option, the touchpad controls perform the same function. The voice commands allow the application to function entirely handsfree to minimize physical contact with the device in the hospital/medical environment where contamination must be avoided.

Development

Google Glass runs on the Android Operating System, with its own Glass SDK binding the OS to the hardware and providing access to features like voice control, accelerometer sensor data, camera controls, et cetera. Setting up a development environment and designing the SPIGLASS application proved troublesome initially as Glass documentation is sparse. Troubleshooting errors specific to the device required more trial and error than googling due to the low number of developers compared to older technologies. The voice control system that directs the application was not available at the beginning of the project; the feature was released in an update to the SDK during the project. These difficulties led to a significant amount of development time spent on researching and experimenting to better understand the options available with the device.

Results

The project results were several versions of the application that evolved as new features were discovered and the design was refined. Initially, the application featured only a touchpad interface with patients and their data displayed on two levels that had to be navigated in order. Recent iterations added restructuring of the screens for direct access to each information display and voice controls. The current state of the application has the example patient data loaded directly on to the device, but future versions would connect over a network and pull information from a server. That would allow other staff to keep the available information up to date. From that point features could be added for convenience, providing greater functionality to its users.

Screen Shots

GGC's First Fish: A Look into Tactile Books

By: Timothy Brooks, Jessica Bui, Alexander Gonzalez and Raquel Lawrence

Faculty Sponsors: Dr. Evelyn Brannock and Dr. Robert Lutz

3D printing has been around since the late 1980s. However, few have actually heard of it until the beginning of the 20th century. 3D printing is a form of printing that primarily produces an object through an additive process. This process creates a part by laying down successive layers of a substance and is driven from a model. The models are created using computer aided design (CAD) or via a 3D scanner. Multiple hardware technologies exist today to support 3D printing, such as inkjet derivatives that use a glued powder substrate, fused deposition modeling (FDM) which uses extruded thermoplastic filament, and emerging technologies such as electron beam melting. Today it is a booming industry with many benefits such as low cost for materials and high efficacy. Many companies take advantage of this technology to make quick parts for their machinery or to quickly print a simple logo design. [1][2] With its capabilities, 3D printing benefits can be realized in disparate fields from factory assembly to helping children with visual impairment. In this paper, we plan to emphasize the uses of 3D printing in tactile books and how it can help children with visual impairment gain the courage needed to interact with the world around them.

The purpose of tactile books is to raise illustrations and/or text that helps convey information to children with visual impairments. Tactile books give visually impaired children the needed information in order to experience the real world while experiencing the narratives. By feeling things (like a leaf or the bark of a tree instead of an outline of a tree) the child is given an idea of what a tree would be like. The physical interaction for most children helps them recognize an object and link certain feelings and shapes to actual real world objects. [3][4]

Publishing companies do have more advanced techniques and manufacturing equipment to better meet the needs of visually impaired children. The prices of these special books are prohibitive because of software, equipment maintenance, and labor. Therefore, parents can find it intimidating purchasing a book. Parents do not want to spend money on a book that could end up confusing their child rather than help them. For example, the book *Good Night Moon* normally costs $5 including shipping on Amazon. However, the same book in tactile form costs $30 and is held in extremely limited supply. This discourages parents and schools from purchasing tactile books. [3] Some parents make their own tactile books using cut-out flat shapes and textured material. However, this is very time consuming and many parents feel it is difficult to meet their child's developmental needs as well as maintain an engaging storyline. Schools are in the same situation. If students have tactile books in their library, often the book is in poor condition and the library does not allow the child to take the book home for them to explore on their own. [3][5]

In order for a tactile book to be useful to children it must follow a set of unwritten guidelines. For example, the printed book must be a size which the child can easily hold. Illustrations in the book must be simplistic so that it will be easy for a child to discover the picture in the book and relate it to the storyline. Stylized pictures must be avoided since all the minor details can be confusing to the touch for a child. Tactile illustrations should have a variety of textures and shape to keep the children engaged. When representing objects or the illustrations in tactile form, it is best to avoid all objects that are presented at angles or in a different perspective to avoid confusion. These guidelines can help publishers, parents, and schools print effective books for their children. Another important tip for those who want to create tactile books is that when creating tactile images do not present an image that is dependent on visual knowledge. For example, a child with visual impairment may not make sense of an outline of a tree and relate it to an actual tree since they do not have the visual knowledge of what a tree looks like. Instead a leaf would be a better representation of a tree since they can feel a leaf, its texture and shape. The silhouette of a tree bears no significance. The pictures in the books should also be spaced out properly to avoid clutter that could potentially confuse a child. Keeping the shapes and images distinct will help them to understand what the story is trying to convey. [4]

One way that has been proposed to cost effectively create tactile books is through the use of 3D printers. With the price of 3D printers going down it can become a more viable solution to parents, schools, and publishers. [3][5] However, this is not a complete answer yet. There is still a need for finding user-friendly software, general guidelines for creating tactile images, an online database of pre-designed and print ready books, and a large interest in tactile books.

In our research project we are exploring 3D printing of tactile books using a MakerBot Replicator 2X, a 3D printer which extrudes ABS filament to create our pages, and are utilizing (through our reading) the knowledge of those who are experts in the field of visual impairment to guide us to the correct representation for a page.

Our plan is to take all the information we have gathered and create a simple tactile page from the open source book Sam's First Fish [6]. We will also utilize a text-to-braille translator [7] in order to translate the text on the page to braille. With this we will combine the two images together as an image file for the page. Using 3D development software we will them create the model we will use to print. The entire process will be documented using video and still images. Still images will be presented, along with the test materials, and lessons learned from failures and successes.

We will demonstrate a way of utilizing this emerging technology as a tool in the hopes of helping visually impaired children. Ideally, a 3D printed page can provide a spark for children to explore the world around them, giving them the courage and confidence to experience what the world has to offer, as well as serve as a quality learning tool for parents and their teachers. Parents could easily buy a 3D printer and have it print out a page or two while they are at work and come home to quality tactile page of a book. Teachers could do the same as well; they could bring their materials and have the printer print it while they are teaching class.

[1] http://www.3dprinter.net/reference/what-is-3d-printing, last accessed 9/19/2014.

[2] http://individual.troweprice.com/staticFiles/Retail/Shared/PDFs/3D_Printing_Infographic_FINAL.pdf, last accessed 9/19/2014.

[3] Abigale Stangl, Jeeeun Kim, and Tom Yeh. 2014. *Technology to support emergent literacy skills in young children with visual impairments.* In Proceedings of the extended abstracts of the 32nd annual ACM Conference on Human factors in computing systems (CHI EA '14). ACM, New York, NY, USA, 1249-1254. DOI=10.1145/2559206.2581341, http://doi.acm.org.libproxy.ggc.edu/10.1145/2559206.2581341.

[4]Philippe Claudet. 2009. T*he Typhlo & Tactus Guide to Children's Book with tactile illustrations*, Les Doigts Qui, Revent.

[5] Abigale Stangl, Jeeeun Kim, and Tom Yeh. 2014. *3D printed tactile picture books for children with visual impairments: a design probe.* In Proceedings of the 2014 conference on Interaction design and children (IDC '14). ACM, New York, NY, USA, 321-324. DOI=10.1145/2593968.2610482 http://doi.acm.org.libproxy.ggc.edu/10.1145/2593968.2610482.

[6] https://archive.org/details/SamsFirstFish, last accessed 9/19/2014.

[7] http://www.brailletranslator.org/, last accessed 9/19/2014.

Forward Error Correction for Fast Streaming with Open-Source Components

John Reynolds
Faculty Advisor: James S. Plank
Collaborator: Diederick Huijbers (software development, Netherlands)
The University of Tennessee at Knoxville

Unreliable networks, implemented with wired or WiFi connections, cause a large amount of packet loss in a multitude of different networked applications. An example of such an application is video streaming. When video data packets are transmitted over lossy channels, the receiving user may not receive all of the packets that were sent. This data loss results in a bad streaming experience if it is not handled properly. The NACK (negative acknowledgement) protocol is used by many applications to request the retransmission of lost data; however, with FEC (Forward Error Correction) [5] the need for the NACK protocol and retransmission is greatly lessened (or rendered entirely unnecessary). In this project, we have built a video streaming environment to explore the effectiveness of FEC, based on erasure codes and implemented in software, across unreliable networks. We have focused on open-source components for video conversion, real-time networking, FEC coding, and adherence to standardized protocols for inter-component communication. In this paper, we briefly outline the software structure of our project, and demonstrate the effectiveness of FEC for video streaming.

Forward Error Correction is an efficient solution to the packet loss problem. Data source symbols being sent across a network may be encoded using erasure codes to create redundant data. This redundant data may then be decoded on the receiving end to recover dropped packets. Many different erasure codes exist. The FEC component of this video streaming library has been implemented using Reed-Solomon based codes from an open-source erasure coding library, Jerasure [1]. Reed-Solomon codes fall under the category of erasure codes labeled as MDS (Maximum Distance Separable). MDS codes have a nice property: k source symbols can be encoded to create m repair symbols such that among the k+m total symbols, any m erasures are tolerable. This means that data recovery is possible for any combination of m packets lost as long as at least k packets are received. The type of packet(s) received (source or repair) does not matter.

The flow of data through the video streaming library we created is handled by many components. Video frames are captured using the “video_capture” library developed by Diederick Huijbers to discover hardware camera devices, activate them, and collect their video frame data output. These video frames are then converted to the appropriate YUV format by Google’s open-source YUV conversion & formatting library, libyuv. Once converted, the frames are encoded to VP8 video format using Google’s libvpx codec. The VP8 encoded video frame is broken into packet sized chunks and transmitted as payload data [4] in UDP/RTP packets [3]. If any of the k packets are dropped in transmission, the related VP8 frame cannot be decoded. The component dealing with packet loss before FEC was added to this project performed NACK-style retransmission. With FEC encoding enabled, the UDP/RTP/VP8 packets become the k source symbols. Jerasure is used to perform Reed-Solomon encoding. The encoding of source symbols produces m repair symbols and adheres to RFC 6865 [2]. The k source symbols and m repair symbols are sent to the receiver. If the receiver detects packet loss, the source and repair symbols received (assuming k symbols arrived) are decoded to retrieve the lost VP8 frame data. If no packets are lost and all k source symbols arrive, the Jerasure decoding step is skipped. Once all of the VP8 frame data is available, whether from received k source symbols or from Jerasure decoding, the data is merged back together from the packet sized chunks and decoded using libvpx to restore the original video format. The original YUV frame data is then bound to an OpenGL texture and rendered.

Real-time streaming data is typically sent in UDP (User Datagram Protocol) packets using the RTP (Real-Time Transport Protocol) packet format [3]. Because of the nature of UDP, packets are not guaranteed to arrive (unlike TCP). FEC renders NACK-style retransmission of lost VP8 frame data unnecessary in this project. Rather than having to request lost video data from the sender, which imposes the time constraints of re-transmission upon real-time streaming, we simply decode using Jerasure to reconstruct lost packets and rebuild the frame immediately without communication with the sender. FEC has allowed us to remove the NACK protocol for handling packet loss.

In real-time video streaming, the number of frames sent per second must match closely the number of frames received and rendered per second. We performance tested by sending and receiving a video feed on two separate wireless networks with 20% packet loss. The number of frames sent per second and the number of frames received and rendered per second both averaged around 20 fps. While there is plenty of room for optimization in the application, this figure is significant. This means that the receiving application is able to receive and decode frames at an average rate that is the equal to the average rate of frames being sent. The Jerasure decoding time to recover lost packets is negligible: 0.00008 seconds on average with an average frame size of 2573 bytes and an average of 4.6 RTP/VP8 packets per frame. The impact of applying FEC to video streaming is negligible concerning added computational complexity.

The development of this open-source video streaming library has provided a testbed. This testbed will allow us to explore the usage of other erasure codes, for example, Raptor codes [6], which have been designed to react to changing network conditions. We may also modify the k and m Reed-Solomon parameters based on statistics sent from the receiver to the sender periodically. Feedback can also be used to configure the coding parameters based on network conditions (reactively). We hope that our experience with video streaming will allow us to design a general purpose open source library for software-based FEC.

References:

[1] Plank, J. S., Simmerman, S., and Schuman, C. D. 2008. Jerasure: A library in C/C++ facilitating erasure coding for storage applications, Version 1.2. Tech. rep. CS-08-627, University of Tennessee.

[2] Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME V. Roca, M. Cunche, J. Lacan, A. Bouabdallah, K. Matsuzono [February 2013]
(Status: PROPOSED STANDARD) (Stream: IETF, Area: tsv, WG: fecframe)

[3] RTP: A Transport Protocol for Real-Time Applications H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson [July 2003] (Status: INTERNET STANDARD) (Stream: IETF, Area: rai, WG: avt)

[4] Westin, P., Lundin, H., Glover, M., Uberti, J., and F. Galligan, "RTP Payload Format for VP8 Video", draft-ietf-payload-vp8-11 (work in progress), February 2014.

[5] Rizzo, L. (1997). On the feasibility of software FEC. Univ. di Pisa, Italy, 1-16.
http://info.iet.unipi.it/~luigi/fec.html

[6] Shokrollahi, A. (2006). Raptor codes. Information Theory, IEEE Transactions on, 52(6), 2551-2567.

Virtual Beauty Competition: A Study in Facial Aesthetics

Hannah Sexton, Chad Howard, Joshua Gutowski, Daniel Hutton, Margaret Russell
University of North Carolina Asheville

Faculty Sponsor: Dr. Marietta E. Cameron, University of North Carolina Asheville

In the developing field of computational aesthetics, researchers examine and define how machines can judge beauty and creativity in a manner similar to humans. One endeavor is to investigate whether universal determinants of beauty exist and what those determinants may be. In particular, we seek to automate the judgments made in such determinations through theories stemming from psychology and mathematics, as well as through machine learning.

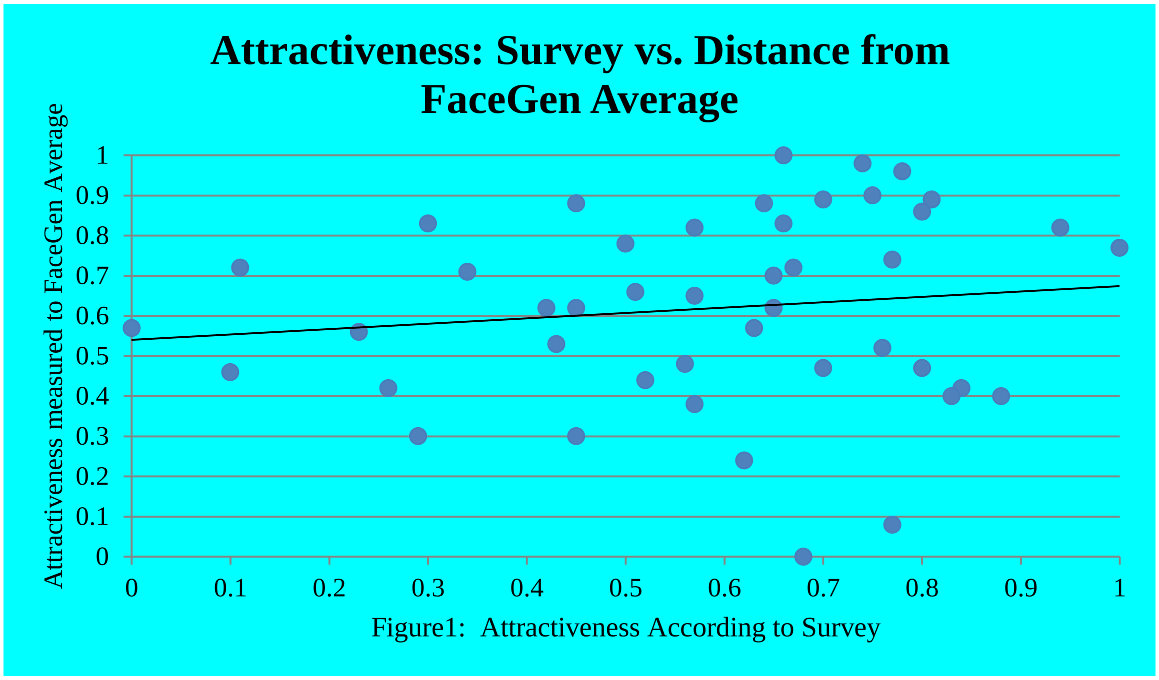
The universal indicators of beauty have been studied in numerous different fields and through different methods. One psychological theory of attractiveness is that humans find average faces to be the most attractive. This theory has dated back to the late 1800's, when Sir Francis Galton noticed a distinct improvement of attractiveness in composites of faces compared to the individual faces (3).

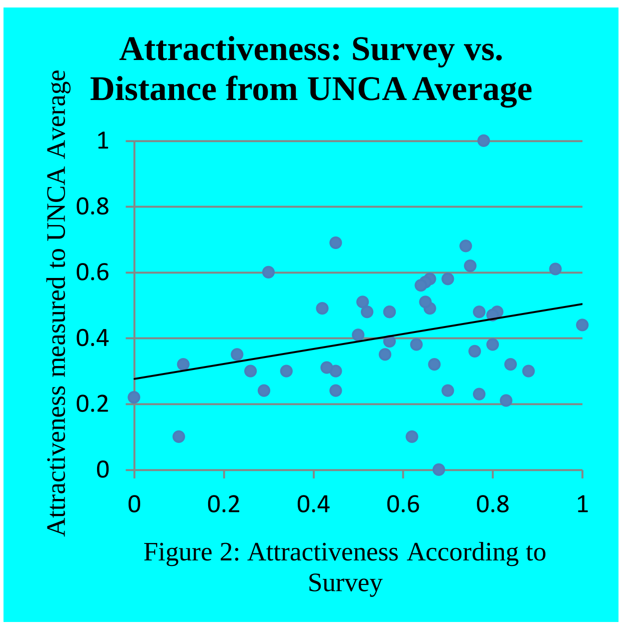
In this project we seek to take a three-dimensional facial model and create software with the ability to numerically measure attractiveness of the models. To create our facial models, we used the three-dimensional facial modeling software FaceGen (2). This software implements Blanz and Vetter's mathematical model of faces, which creates a linear combination of base shape and texture morphs which deviate from a base average face that is determined using a database of faces (1). This allows individual faces to be compared to the initial average face.

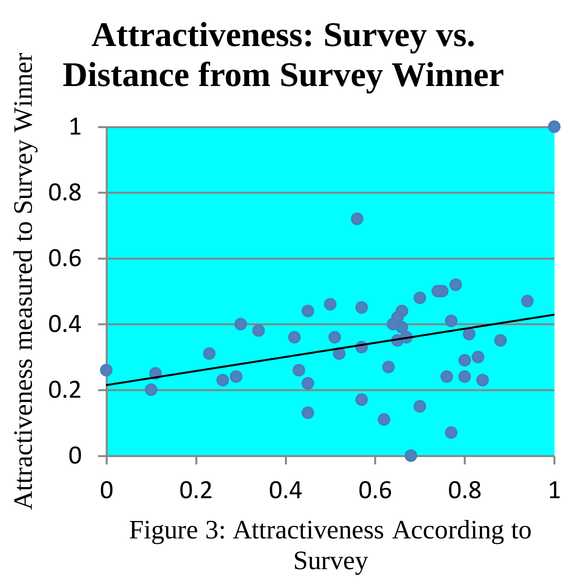
Using the FaceGen software development kit, we created 45 models from photos of 238 faculty and staff members. The 45 final models represented an averaged face of the faculty and staff for each of 44 different university departments and programs, with the final model developed from the averages of all faculty and staff. We then presented these models in a web application to the UNC Asheville student body and faculty in the form of a virtual beauty competition. The 45 models were split into three groups of fifteen. Each judge was given a group to evaluate based on several criteria including attractiveness. The competition was open for a week, during which time we received results from 249 volunteer judges.

In the initial analysis of the results, we looked solely at the attractiveness data given by each judge about individual face models. Acting under the hypothesis that an average face would be viewed most attractive, we considered the data under numerous pivots of the average face including the FaceGen average face, the UNC Asheville model's average face, and the winner of our survey. None of these pivots showed any correlation among the data (Figures 1-3).

We are continuing to analyze other data received from the competition, including information about the models perceived age and gender, as well as adjectives used to describe each model. We are also considering other means of measuring attractiveness which could provide correlation between our current data and future data obtained. We also plan to repeat the survey to verify and reinforce our existing data.







Reference List:

(1) V. Blanz and T. Vetter, “A morphable model for the synthesis of 3d faces”, in Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99, (New York, NY, USA), pp. 187-194, ACM Press/Addison-Wesley Publishing Co., 1999.

(2) “Facegen: The easy way to generate realistic faces," 1998. Company: Singular Inversions, URL: http://www.facegen.com.

(3) F. Galton, “Composite portraits," Journal of the Anthropological Institute, vol. 8, pp. 132-144, 1879. http://galton.org/composite.htm.

Predicting Global Memory Performance of Simple CUDA Algorithms

Randall Pittman
Faculty Advisor: Dr. Durell Bouchard
Computer Science Department, Roanoke College
September 22, 2014

With the advent of more powerful and versatile graphics cards, there is an increase in demand for high-performance programs that can harness their power. Towards this purpose, the CUDA library (Compute Unified Device Architecture) has been developed as an interface for C/C++ code to be executed on the GPU (Graphics Processing Unit). This paper develops a method to predict the memory access performance of CUDA programs written for the graphics card. By analyzing the C/C++ program, we can estimate the number of memory accesses that will be performed in the CUDA program. We can then measure the time required to access memory once, and thus predict the total time that will be spent accessing it. Software developers could then use these predictions to anticipate performance issues when developing CUDA programs.

In recent years, CUDA has proven itself to be an incredibly useful interface for programmers executing general purpose applications on the GPU, which originally was designed only for graphics processing. In order to process graphics, the GPU is inherently designed to be a massively parallel device, which is to say that it can execute hundreds or even thousands of tasks simultaneously. CUDA has simply made these tasks much more generalized, so that we are not limited to image processing.

However, in generalizing these tasks, CUDA hides certain performance issues from the programmer. One of these problems is slow memory access, specifically global memory, which is accessible to all the tasks that are currently running. It is necessary to use this memory when programming in CUDA because we need to store input and output data somewhere on the device. My research is concerned with estimating the total amount of time that will be spent accessing this global memory, so that potential performance problems can be anticipated.

There is a significant amount of research on predictive models for CUDA programs. For example, [2] and [1] both develop models that are designed to predict CUDA performance. The major difference between other models and my research is that other research often uses actual CUDA code to make performance estimations. My research makes predictions without the actual CUDA code. I use the C/C++ code to estimate how much time the GPU would spend accessing memory if the program were reformatted to be run by CUDA.

This method uses the assumption that given a C/C++ program, the number of memory accesses will remain roughly the same when the program is rewritten for the CUDA interface. For example, in the all-pairs method of particle simulation, each particle needs to know the position of every other particle (see [3] for more information on this and other particle simulation techniques). This is true for both the C/C++ program and the CUDA program, because it is inherent to the particle simulation algorithm. Once we have the estimated number of accesses, we can predict the time spent accessing memory by multiplying this number by the average time for a single access. However, since a single memory access can potentially take little more than a nanosecond to complete, it can be difficult to measure this value precisely. To obtain this value, we can find a linear equation that precisely models memory access time given the number of accesses. In other words, we can test how long it takes to access memory 1000, 10000, and 100000 times, for example, then fit a linear equation to these data points, and then use this equation to get the time for a single access.

Access Type	Access Time (seconds)
Fast Read	1.041371×10^{-9}
Fast Write	7.287326×10^{-10}
Slow Read	4.229417×10^{-9}
Slow Write	4.376943×10^{-9}

Table 1: Time for global memory access

The CUDA tests for this research were performed using an Nvidia GeForce 8400 GS graphics card. The times measured for global memory access are listed in Table 1. Notice that there are 4 numbers in the table instead of 2 (for read and write). Normally there would be only a read and write time, but memory access time varies in CUDA depending on exactly how it is accessed. This is simply a peculiarity in the GPU hardware and the CUDA architecture. To account for this variability we can test for the best and worst case access times using the linear approximation technique as discussed above.

Using these times we can estimate the total memory access time for specific programs. For testing purposes I selected three different problems that are commonly used for performance testing, and that also had varying frequencies of memory access. I have already mentioned one of them: particle simulation. Again, this problem requires a large number of memory accesses because each particle needs to know the position of every other particle, and that position is stored in global memory. The second problem tested was matrix multiplication, which requires a moderate number of memory accesses in comparison to particle simulation. The last problem was generating an image of the Mandelbrot set. This problem essentially involves computing the value of a mathematical function hundreds of times for every pixel in the resulting image. This problem requires very few memory accesses, because it only needs to write a result for each pixel in the image. The final results of these tests indicated that our memory estimation model appears to be accurate.

In summary, this research developed a method to predict the total time that will be spent accessing memory in a CUDA program based upon the C/C++ program. This required knowing the time for a single global memory access, which was determined through linear approximation. We then tested this method on three different problems that each had varying frequencies of memory access. The results of those tests confirmed the accuracy of the model. Thus, with some improvement, this research could be used to make performance predictions that would influence the planning and development of a CUDA program.

References

[1] Sunpyo Hong and Hyesoon Kim. An analytical model for a gpu architecture with memory-level and thread-level parallelism awareness. In ACM SIGARCH Computer Architecture News, volume 37, pages 152{163. ACM, 2009.

[2] Kishore Kothapalli, Rishabh Mukherjee, M Suhail Rehman, Suryakant Patidar, PJ Narayanan, and Kannan Srinathan. A performance prediction model for the cuda gpgpu platform. In High Performance Computing (HiPC), 2009 International Conference on, pages 463{472. IEEE, 2009.

[3] Lars Nyland, Mark Harris, and Jan Prins. Fast n-body simulation with cuda. GPU gems, 3(1):677{696, 2007.

Measuring Engagement in Virtual Reality

Derek LaFever
Faculty Advisor: Dr. Durell Bouchard
Computer Science Department, Roanoke College

In this paper, we look at objective and subjective measures of engagement in a Virtual Reality (VR) environment. We have created suitable unexpected events that will occur in a virtual world, in this case a video game, in the hopes of obtaining a response, either through the postexperiment survey or physical motion, to use as a measure in determining how engaged the user is while in the virtual environment. In the context of this paper, immersion is defined as the sense of feeling a presence in the VR environment. Engagement however, is defined as a step towards being immersed, where it measures how accessible and interesting the VR program is to the user. Throughout this paper, we will discuss the events that will be presented to the user, the setup of the virtual world, peripherals used, and how data is collected.

Our project takes a close examination of what it means to be engaged in a video game. More so, we look at ways of properly measuring and determining engagement both in real time and through data analysis. In order to determine how engaged the user is, we take ideas and methodology from psychology, as well as many of our own measures that fit specifically to the game; all of which are based on critical analysis of engagement. What we came up with are stimuli called unexpected events. These events are aimed at testing attention of the user which plays a role in determining how engaged someone is during a task. Examples of these events include displaying images on the screen that are out of context from the game, slightly tilting the VR world, and creating out of context sounds in the physical test room. All of these events are testing attention, but we are looking for a different response for each of them. Ideally, we are attempting to obtain a physical response that we can capture for later data analysis.

The game is an adaptation from the popular mobile app *Fruit Ninja*. The game plays as followed: The player is presented with a background where objects spawn from either the left, right, or bottom side of the screen. The goal is to use hand motions to 'cut' the objects and obtain the most points before the time limit. At random times throughout the game, the user will be presented with unexpected events, both internally in the virtual world and externally in the real world, in the attempt to invoke a response to give us a quantifiable measure of engagement. The game as a whole is meant to keep the user engaged and interest while keeping the learning curve minimum.

In order to obtain data on the user's physical reactions and create the virtual world, we need a couple of peripherals. The peripherals used in this project were the Microsoft Kinect and Oculus Rift. The Kinect was the main controller for the user as the Kinect tracked the user's hands for direct control in the game. Beyond this, the Kinect tracked all upper body movement (since the user is seated during the game) to record all physical reactions and movements for later data analysis. The Oculus Rift is a head mounted display (HMD) and serves as the VR that the user will use throughout the game. The HMD is used to overlay the user's vision with the virtual world. Because the HMD has an accelerometer, gyroscope, and magnetometer, we can track the user's head position and movement and therefore control the in game camera to the orientation of the user's head. This is useful for getting the user to feel immersed in the game so our measures of engagement can be more accurate.

To keep from repetitiveness, which may lead to boredom, we have created a dynamic difficulty adjustment system that actively monitors the user's performance and changes difficulty accordingly. Some effects from changing difficulty include more objects appearing on the screen to 'cut' at once or red objects that will subtract points if the user hits them. The goal is to keep the user focused on the game and feel engaged.

Because if the user is engaged, we can present these unexpected events and determine how immersed that are based on their response. Ultimately, we are trying to create an environment that emulates a video game where the user is immersed and engaged, so we can study the events that measures engagement the best to properly determine how engaged the user truly is.

Teaching CS concepts without Computers: Using CS Unplugged As A Recruitment Tool

Lauryn Burt, Lucy Copas, Kamilah Kiser

Georgia Gwinnett College

lburt1@ggc.edu, lcopas@ggc.edu, kkiser@ggc.edu

Mentor: Sonal Dekhane

Introduction

In this poster, we will provide an overview of Computer Science (CS) Unplugged activities that were used to engage students in computing. We will describe the modifications made so that they can be used by others. We will report upon the use of these materials in undergraduate classrooms at Georgia Gwinnett College. We will share participant feedback from the workshops, lessons learned, and ideas for future work.

CS Unplugged is a teaching method that allows students to understand how computers work - without ever booting up a machine. Through the use of cards, posters, and other interactive teaching methods, CS unplugged creates a teaching system that is suitable for both children and adults. Due to the easily customizable sets of exercises that are already available online, it can also be tailored to suit any demographic of students within only a few minutes. CS unplugged engages students with a hands-on approach and gives them the opportunity to learn about the systems that they use every day, in a literally out of the box method. Computer Science is often viewed as being “too hard” or “too abstract”. By using CS Unplugged to create a friendly and relatable teaching method, hopefully some of this stigma can be erased for some students and help them realize a personal interest in Computer Science.

Need/Goal

Many students (including some IT majors) find several concepts in Computer Science extremely daunting. By breaking down simple concepts such as binary and image compression and offering both a visual and tactile way of teaching these ideas to students, CS Unplugged not only helps make these otherwise sticky concepts easier to understand, but also allows for other types of learners (such as visual learners) a more relatable way to learn this material.

Current goals for this project include helping students learn how to: Understand binary numbers and how to convert from decimal notation to binary; a basic understanding of image compression and why it is important when transmitting images; and finally to offer understanding of what a programming language is, and how a program actually works. These simple concepts will help build a foundation for students to better understand the computers around them, and hopefully help spark an interest in computer science.

Methodology

In the past, we selected lessons from the CS Unplugged website on text compression, image compression, and error detection. We conducted 75 minute workshops in several introduction to computing classes on our campus. In the error detection activity, the presenter did a magic card trick to demonstrate how the computer detects errors. The presenter set up a grid of cards with some face up and some face down. Then he/she would ask a volunteer to flip over one of the cards while they were not looking. Then, the presenter would look at the cards and magically know exactly what card the volunteer flipped. Simple activities like these helped the students understand complex computer functions as well as helping them realize that computer science isn’t as scary as they thought.

This semester we have chosen three new activities from the CS Unplugged website to use in our workshop. Our activities will cover concepts such as binary numbers, programming languages, and image representation. The workshops will be a circuit of activities instead of one activity at a time like we have done in the past. The class will be split into groups and rotate to the different activities. In the binary numbers activity students will play a game to help them understand how to represent decimal numbers in binary. Programming languages activity will have the students pair up and act as a programmer and a computer. The programmer will have to instruct the computer to draw an image they are not allowed to see. The image representation activity will have the students shade in a grid to represent an image in a combination of 0s and 1s.

Results

After the workshop, we asked the students to participate in a survey about their experience. The overall satisfaction of the workshop was a 4.2 on a scale of 1 (strongly disagree) to 5 (Strongly agree). Participating in these activities left 22 of the 30 participants feeling motivated to learn more about computer science. When students were asked if they learned anything new about technology, 27 of 30 felt they had. On a scale of 1 (strongly disagree) to 5 (Strongly agree) our workshop scored a 4.2 in the area of how engaging the activities were. 24 out of 30 students would recommend this workshop to others.

Conclusion

Positive feedback from students for the CS Unplugged activities has encouraged us to continue conducting such hands-on workshops to engage students in and educate them of computer science concepts. We will continue to engage and recruit students by modifying CS Unplugged activities to fit our campus needs and will also continue collecting feedback from workshop participants to further enhance our workshops.

Reference: *Computer Science Unplugged*. N.p., n.d. Web. 18 Sept. 2014. <http://csunplugged.org/>.

Computer Vision: Recursive Median Sample-based Blob Detection

Thomas Lux
Dr. Durell Bouchard
Roanoke College
September 22nd 2014

This paper proposes a new method for detecting convex patches of similar colors in images, otherwise referred to as blobs. The method is aimed at computational efficiency and consistency. The algorithm repeatedly samples an image to gather information including number of blobs, average blob size, blob colors, and locations.

The motivation for this new blob detection algorithm is a viable use for single camera autonomous navigation of space. Mobile robotic platforms are tasked with navigating space using a variety of sensors. Networked robotic systems that have offsite or collaborative processing gather large quantities of data and make use of it without spending excessive processor time and power. Robots that must do all computations on-board often experience limited times of operation because of the large amounts of processing they must perform to navigate. Therefore, a low powered system for navigation that consistently provides navigable data would benefit mobile robotic platforms. RGB Cameras are low powered and widely available, but most current algorithms in computer vision require unnecessarily large amounts of computation.

Performing the fewest possible computations sets new guidelines, this formula approaches images differently than most modern computer vision algorithms. It is common that operations which are performed on an image are performed on the entirety of the image. This has been done in the past to guarantee that all points in an image which meet given criterion are found. In navigation however, much of the visual plane becomes redundant. For a system that prioritizes speed, only the macroscopic components of the visual field need to be processed fully. That leaves small features to merely be attended in part, excluding all features that do not provide valuable data for the act of navigating.

The proposed method gathers a requested number of samples from an image and then computes the blob surrounding each sample point. The blob surrounding a sample point is identified using a recursive median search pattern. The recursive median search is performed by first choosing a search vector from the initial point and then spanning along both the positive and negative directions of the vector until a sufficiently different colored pixel is found. Once the edges of the potential blob are found along that vector, the median of those points is used recursively as the starting point for the next search vector until a desired number of searches has been performed. This process is repeated for every sample point in an image. For this formula to be most effective, a variety of search vectors should be included. While the aforementioned algorithm is performed, the colors, bounding edges, and the centroids of blobs in the image are found.

The advantages of this algorithm are seen in performance. The number of computations required to find the bounds of a blob in an image have a constant upper bound with an optimized vector search. This means that the dimensions of the image do not affect the performance of this algorithm. The number of computations is solely dependent on the number of samples gathered and the desired pixel precision for the edges of blobs.

The recursive median sample-based search algorithm has proved to be a valid means of blob detection in practice. The resultant blobs detected are large patches of similar color in images. In comparison to other blob detectors, which are more focused on finding blobs of specific shape and size, this algorithm finds blobs that are of a similar color. The benefits of using color are the lack of a need for pre-processing images and the effective use of multi-channel data present in RGB images. Through experimentation, color has proved very relevant in the detection of connected regions of images.

Computationally inexpensive and reliable navigation is the end goal of this blob detector. The important factors to extract over time are, in this case, depth and connectivity. Depth is important to know what space in the visual plane is navigable. Connectivity is important because effective classification of connected components can greatly reduce the number of computations required to navigate. The recursive median sample-based blob detection algorithm provides the approximate centers of blobs and the information necessary to group by color, size, and location. Extracting centers and bounding points of blobs across multiple image frames allows for relative depth to be extracted quickly. Comparing the motion and position of similarly colored blobs allows for grouping and classification.

The conclusion of this research is that a recursive median search is a computationally effective means of locating blobs relevant to navigation in images. Potential future work with this algorithm lies in three main areas. The method of sampling is uniform as of now, but could potentially be distributed over chosen portions of images to yield better blob detection rates. The number of vectors and recursive searches in use to best identify blobs of specific size and shape is still unclear. Lastly, automatically adjusting for variations in scene dynamic by changing the length of search vectors may yield a higher detection rate for blobs as more and less cluttered scenes are encountered.