ELEC 399

Final Project Report

Audio with Kinect

Course: Elec 399

Professor: Dr. Tao Lu

By:

Group Members:

Nathan Willson, Cameron Smith, Drew Harris, Brian Richter

URL: http://web.uvic.ca/~dbharris/elec399/

Supervisor: Dr. Peter Driessen

Due: August 3rd, 2012

Table of Contents

Chapter 1: Goals

The goal of the project was to research and investigate all the necessary steps before the actual implementation. Once a basic design and framework are in place the only remaining task is to develop and test the project, which would be accomplished in ELEC 499. A full description of the project is available in "Chapter 3: Detailed Project Description."

Goals specific to the project included:

- **Investigating methods for capturing an impulse response**
  - Locating academic resources
  - Comparing measurement methods and discussing advantages
- **Interfacing with the Kinect**
  - Using the Kinect API
  - Uploading test code
- **Audio capture with the Kinect**
  - Record and write 4 channel raw audio from the Kinect
- **Measuring the spatial properties of a room with the Kinect**
  - Find libraries and documentation to capture a room
- **Complete all project deliverables**: Progress reports, a final report, a presentation, and a website.

Chapter 2. Progress Overview

The purpose of this project is to determine the possibility of using the Microsoft Kinect to simulate the audio and visual response of a room using an array of cameras and an array of microphones. The actual implementation is not documented in this project, though the Conclusion/Future Works section offers insight into methods to do so.

Several methods and techniques for capturing an accurate impulse response of the room were investigated. The built-in Kinect microphone array is limited in effectiveness due to the closeness of the microphones and sensitivity to internal device noise. An appropriate use of microphone extension would be to place a microphone in four corners of a room and measure the impulse response – several different placement methods should be tested to determine effectiveness. Capturing the response is uniform regardless of placement method – play the impulse in the room to be captured, record the impulse response with four microphones and apply convolution reverb techniques to simulate a sound being played in the room.

Interfacing with the Kinect was completed successfully using the official Microsoft Kinect SDK. Several example games and programs were tested with the Kinect to demonstrate successful connectivity (via USB). Example code in C++ was compiled using the SDK and Microsoft Visual Studio to gather video and audio data from the Kinect. Programs written in both C# and C++ were run in Windows and had no issues connecting to the Kinect.

Using USB connectivity between a host computer and the Kinect, a 4-channel impulse response of a room was captured. Raw PCM wave file data was captured and written to the disk of the host computer and loaded with Matlab. Using Matlab's built-in convolution functions, an input signal was convolved with a recorded impulse response of a room. As the microphones used to record the response were the Kinect's built in microphone array (not extended), the impulse was lacking in clarity and the output (convolved) waveform did not hold the desired sonic qualities of the room. This demonstration did show that writing 4-channel audio to a host computer from a Kinect via USB is possible.

While writing raw audio was successful, none of the Kinect's built-in beam-forming algorithms or echo cancellation techniques were successful. Though these features are implemented in the Microsoft Kinect application-programming interface (API), they were not tested. This may be essential to acquiring an accurate impulse response of a room using the built in microphones, but it may not be necessary if external microphones are used.

Though a full implementation was not completed, all project deliverables, including progress reports, final presentation and project website were completed.

Chapter 3. Detailed Project Description

This projects objective is to design a system, which can capture a 3D acoustical impulse response of a room and spatial dimensions using the Microsoft Kinect. The Microsoft Kinect is multi-sensor input device equipped with a RGB camera, depth sensors and even a microphone array. Although the device was initially developed to extend the interaction between the user and the gaming console, engineers around the world have instead adapted the Kinect as an input device into one's computer. Developments of the Kinect's motion and camera capabilities have been extensively documented, however the audio features have remained relatively untapped. Currently, the Kinect is capable of performing complex echo cancellation techniques and sound source localization using beam forming algorithms with four microphones.

The device is limited by the size and placement of the microphone array, located just underneath the cameras. This constrains users from developing new audio capabilities. However, in principle an external connection could be made to each microphone input to enabling placement at various positions within a room.

This project is an investigation into extending the microphone processing of the Kinect, in particular for the purposes of measuring a 3D impulse response of a room. With four microphones one is able to measure the response at various positions within a room. This is useful as a single impulse response measurement would not account for wave effects such as low frequency modes and would likely assume diffuse field theory. By taking the impulse response at multiple positions, one can more accurately characterize the room response.

The camera capabilities of the Kinect also allow for spatial mapping of a 3D space. This is a feature that has been widely documented and is easily supported. By connecting the two technologies one would be able to correlate the impulse response to a 3D map of the room. This feature will be of secondary importance for the project.

Applications of this project include more accurate impulse response measurement capabilities, better reverberation modeling of rooms, and providing useful data for virtual reality techniques such as auralization. One could even consider implementing surround sound methods such as ambisonics, which at minimum requires four microphone channels. There is also currently a shift in the entertainment (video games and cinema) industry to incorporate 3D audio.

As there is no budget available for implementation, this project will be strictly theoretical. The group will investigate the necessary tools and methods that would be needed to make the device work. The project can be divided into three topics:

1. **External Microphones:** enabling the user to create more general-purpose microphone arrays.
2. **3D IR (Impulse Response) Measurement:** Techniques and audio processing.
3. **3D Camera:** Obtaining a spatial map of the room.

The deliverables will include a report on how to theoretically implement the system, a poster, and a web presentation. The goal is to have at minimum completed the 3D IR Measurement techniques, which inherently includes External Microphones, by the second progress report (June 26th, 2012). As was mentioned earlier, the 3D spatial mapping of a room is secondary to the main goal of the project.

Our group consists of four members: Brian Richter, Cameron Smith, Drew Harris, and Nathan Willson, and is supervised by Dr. Driessen.

Chapter 4. Workload Distribution and Achievements

Each member contributed equally to the progress reports, and final report. Research of the projected was split into two groups: An acoustics team, and a Kinect team. Each group was responsible for completing a set of tasks and goals. For the acoustics team the goal was to obtain sufficient information on how to obtain an acoustical impulse response measurement. For the Kinect team the main goal was to interface with the device and do some preliminary testing. Both teams recombined their findings for the final report.

Nathan Willson (V00682878)

Nathan is a fourth year electrical engineering student. His primary engineering interests are in the field of audio, in particular 3D audio and acoustics. Currently, he is completing school with a Digital Signal Processing specialization.

His skill set includes extensive MATLAB ability, competence with music and audio production software, and recording experience. Additionally, he also taken courses about acoustics and has worked with sound equipment on his coops. As Nathan is interested in 3D audio, he has knowledge regarding spatialization, localization of sound, beamforming, auralization, etc.

Nathan contributed to the project by researching the acoustics portion of the project. This included an investigation into methods of obtaining an impulse response.

Brian Richter (V00482743)

Brian is a fourth year computer engineering student with a specialization in software. His major interests include operating systems, database systems and the development of software and hardware as they relate to music production and distribution.

His software development experience varies drastically including C, C++, Java and assembly. His most recent invention was a homemade synthesizer constructed with a Tupperware container and breadboard named the ‘Snorlax’ (http://www.youtube.com/watch?v=hnOPUT1qnuE). He owns and has played with a Kinect in the past, which will help build a foundation to development.

Brians's primary work on this project has consisted of research into the Kinect interface and the system analysis and design methodology.

Cameron Smith (V0482018)

Cameron is a fourth year electrical engineering student taking the biomedical option with electronics and energy systems specializations. His interests include circuit design, electromagnetic theory and semiconductor physics.

Cameron's skill set includes extensive digital circuits experience, biological parametric measurement and power regulation. He has experience in a variety of programming languages including C/C++, Matlab, Python and VHDL and is currently learning System C.

Cameron's primary work on this project has consisted of research into the Kinect interface and the system analysis and design methodology. Cameron was a primary editor of the works produced.

Drew Harris (V00682489)

Drew Harris is currently completing a bachelor's degree in electrical engineering at the University of Victoria. Drew writes and produces electronic music and is involved in several independent projects using embedded computers and electronics to blend art, media and engineering together into unique installations. He is interested in using technology to further the fields of art and music and would like to learn more about implementing digital signal processing techniques to do so. He has been researching the use of commodity smartphone processing units to do complex signal processing operations in low-cost, low-power portable situations. Drew is interested in Linux and using open-source free software to find new ways to create and manipulate sound.

Drew has experience with hardware design and software development in a number of languages including Python, C++, Java and Bash. He has extensive experience recording and producing audio in many different audio environments.

Drew contributed to the project by researching the acoustics portion of the project. This included an investigation into methods of obtaining an impulse response.

Chapter 5. Project Discussion

5.1 Measurement of the Impulse Response

The main application of impulse response measurement is room simulation, computed by convolution reverb. Convolution reverb is the process of convolving an n-channel input signal with an m-channel impulse response. The impulse response is the recording of a signal spanning the audible spectrum played in the room to simulate. The idea is that this recorded impulse response will contain all of the sonic characteristics of a space, and convolving the impulse response with an input signal will create the illusion of the signal being played in the simulated space.

As mentioned above the IR requires an input signal into the system that covers the desired frequency range. In practice, three different methods are used. Each has its own advantages and disadvantages:

- White noise/Pink Noise
- Maximum Length Sequence signal
- Exponential sweep

5.1.1 White Noise

The white noise/pink noise method is the easiest to implement, but the IR response is poor and is virtually impossible to reproduce identically. There are also issues surrounding the phase response when using white/pink noise.

5.1.2 Maximum Length Sequence

Maximum Length Sequence signals, or MLS, are commonly used amongst acousticians. A MLS signal is pseudo random binary sequence. To the ear it might sound like random white noise, but it actually follows an algorithm. The usefulness of an MLS signal is because of the autocorrelation, which when applied to the MLS signal produces equivalently an impulse response. There is documentation and software online the uses an MLS sequence, but it is often not free. Alternatively, one could write a MATLAB script to accomplish the task. Theory about the subject can be found at [1] and [2].

5.1.3 Exponential Sweep

The final method is the exponential sweep. The theory is to have a sinusoid that exponentially increases with frequency. There is plenty of academic documentation available, [3] and [4]. Additionally, a group of UVic students previous implemented a similar project using exponential sweeps [9]. The method

has been argued throughout the academic community to offer the best response. In this project the measurement of IR's will mostly likely be conducted using the exponential sweep method due to the available resources.

5.1.4 Playback and Convolution Reverb

The Kinect has the capability to record a 3-channel (or 3-dimensional) impulse response with 3 different microphones. Convolving a 3-channel impulse response with a 1 or 2-channel input signal (such as a recording of a voice or an instrument) will yield a 3-channel output signal - simulating the room response in 3 dimensions. In this case, proper simulation would require a 3-channel audio output system, such as 3 speakers. An alternate way to simulate the room would be to use head-related transfer functions (HRTFs) to simulate three-dimensional sound, mapping each of the convolved three dimensions of audio into two channels of audio using the HRTFs. This would allow the output signal to be played back using over-ear headphones, or even speakers using cross-talk cancellation.

Convolution reverb is commonly used in audio production and post-production, for music creation, video production and video game sound production. Digital audio workstation software like Apple Logic and Steinberg Cubase have plugins that are able to convolve input signals with user-supplied impulse responses. Convolution reverb is a better choice to simulate a physical space than mathematical simulation as the output will be much more accurate. Impulse responses are often used to simulate unique spaces, such as churches, cathedrals, hallways and canyons; many pre-recorded responses are available on the Internet. The downside of convolution reverb is that it cannot be done in real-time - the convolution process requires converting the input signal into the frequency domain, multiplying it by the frequency-domain version of the transfer function (impulse response) and converting the output back into the time-domain. This requires having the entirety of the signal available at compute time (reading future values), something not possible to do in real-time.

5.2 Kinect Interface

5.2.1 Official SDK vs OpenKinect

Development of Kinect applications can be accomplished with two separate approaches. When Microsoft first released the Kinect they did not release an official software development kit, forcing the open source community to quickly hack the device and create an open source solution [5]. Recently an official SDK has been released, allowing developers to create applications with official support from Microsoft [6]. There are many important factors to consider when choosing a solution.

5.2.2 Development Environment

The official Microsoft SDK is only available for installation on Windows. This means that if development is to be accomplished on another operating system such as Linux or Mac OS X, the OpenKinect solution must be used [7]. It may be possible to run Windows within a virtual machine on one of these platforms but it is hard to find documentation of how to achieve this, and would involve complex mapping of usb device.

5.2.3 Programming Languages

The open source community has been quite busy creating libraries for a large variety of programming languages such as Python, C, C++, C#, Java, Lisp as well as many more. The Microsoft SDK is unfortunately only compatible with C++, C# and Visual Basic and by using Visual Studio.

5.2.4 Documentation

Microsoft has documented their SDK quite well and also provides a support forum for developers who need assistance. Since OpenKinect is an open source project the documentation for it is spread a bit thin. There are various ways to request assistance but it might be a little bit harder to find someone that has time to help. Also since it is cross platform there are many different things that can go wrong.

5.2.5 Ease of Use

To setup the Kinect on Windows with the official SDK is very simple. The drivers will be installed effortlessly and developing can be achieved in minutes. The OpenKinect configuration is a lot more involved since their implementation has so many different variations and situations, resulting in a more complicated setup process. This process can become troublesome on some setups and oftentimes requires a lot of time spent debugging.

5.2.6 Resources

Both variations have great resources and libraries available to be used for both 3D depth data as well as audio streams. The OpenKinect variation may require more technical inspection and won't be as easy to access the data.

Considering all of these factors we are going to take the approach of using the official Microsoft SDK because of its ease of use and great documentation. This unfortunately will mean that we are going to be constricted to developing in C++ or C# and on Windows.

5.2.7 Microphones

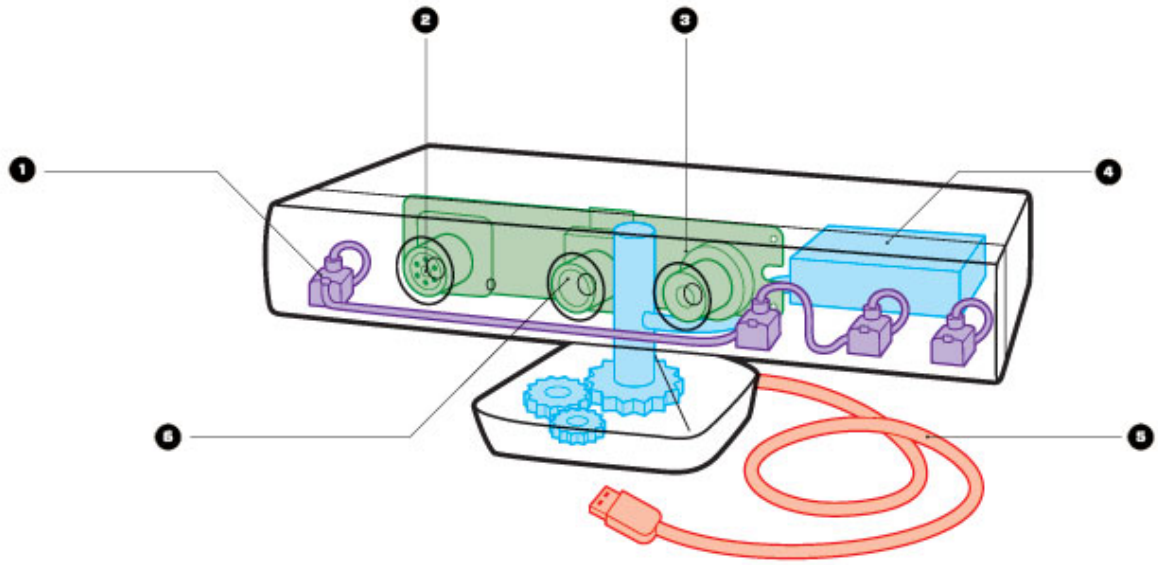

Figure 1: Kinect [5]

In the above picture the purple items labeled with a 1 are the microphones. These are not in the most optimal locations for recording 3D audio data. They are located extremely close to each other, resulting in a very small difference between the channels. This is especially evident in the 3 microphones on the right side of the image. They are all located within an area of around 10 centimeters. In theory these microphones can easily be extended by increasing the length of the wire that they are attached with outside of the shell of the Kinect.

5.2.8 Recording data using the Kinect SDK

The sample projects on the Kinect SDK website are quite informative. They were successful in describing how to program using the given Kinect API. To record from any of the inputs of the Kinect such as the RGB camera, depth camera, or microphone array there are file stream methods. Once these streams are read in real time from the Kinect they can then be written to the desired file type. We created a simple program to read the audio signals from the microphones. A sample of the code has been attached to the report. The audioStream.Read() method is where the signal from the microphones is being read from the Kinect into a data buffer. This data buffer is simultaneously being written to an output wav file.

```
private void RecordAudio()
{
    using (var source = new KinectAudioSource())
    {
        var recordingLength = (int) _amountOfTimeToRecord * 2 * 16000;
        var buffer = new byte[1024];
        source.SystemMode = SystemMode.OptibeamArrayOnly;
        using (var fileStream = new FileStream(_lastRecordedFileName, FileMode.Create))
        {
            WriteWavHeader(fileStream, recordingLength);

            //Start capturing audio
            using (var audioStream = source.Start())
            {
                //Simply copy the data from the stream down to the file
                int count, totalCount = 0;
                while ((count = audioStream.Read(buffer, 0, buffer.Length)) > 0 &&
                       totalCount < recordingLength)
                {
                    fileStream.Write(buffer, 0, count);
                    totalCount += count;
                }
            }
        }

        if (FinishedRecording != null)
            FinishedRecording(null, null);
    }
}
```

Example Code: Recording audio

Chapter 6. Summary and Future Works

The objective of this project was to determine the feasibility of using the Microsoft Kinect sensor device to measure audio properties of a room using the Kinect's microphone array, potentially extended with more/better microphones. The team investigated the probable challenges involved in developing custom digital signal processing software for the Kinect and using the code with the device, including choosing the correct software development kit (SDK) and programming language to efficiently utilize the maximum capabilities of the device. An in-depth analysis was done on room simulation using impulse response measurements and convolution reverb – the advantages and disadvantages of using either the Kinect's built-in microphone array or an external microphone array were investigated. As an effective impulse response requires careful placement of microphones, it was decided that utilizing multiple external microphones would improve the quality of the room analysis.

As briefly discussed in the project description, applications of the project include: 3D reverberation modeling, this includes auralization and virtual reality simulation; the four audio channels of the Kinect meet the minimum channel requirement for ambisonics. Many companies are quickly moving into the field of 3D audio, especially in video games and the movie industry. There are also research opportunities to study the correlation between the spatial and audio response, which could yield insights for better room acoustics prediction modeling.

Future implementation of this project should follow the research detailed in this report. The best choice for writing Kinect specific software was determined to be the official Windows Kinect SDK due to its extensive support, documentation and ease to set up. The audio analysis code should be written in C++ due to the team's familiarity with the language and due to the support using the language with the official Kinect SDK. The Kinect hardware should be opened and extended using external microphones, as the internal microphones are not optimized for obtaining an accurate impulse response of a room. Though it was of secondary importance for research in this report, a final project utilizing the Microsoft Kinect for accurate three-dimensional room simulation should make use of the Kinect's visual sensors, such as its RGB camera and depth sensor (infrared light emitters and sensors). A device that is able to produce both an audio and visual response of a room would enable development of a room simulation toolkit, useful for those in the sound design industry and the construction industry, among others.

References

[1] “Maximum Length Sequence,” http://en.wikipedia.org/wiki/Maximum_length_sequence, July, 2012 [August 1st 2012]

[2] “MLS Theory Basics,” http://www.purebits.com/mlsteo.html, 2004 [August 1st 2012]

[3] Farina, A., “Advancements in impulse response measurements by sine sweeps”, *Presented at the 122ed AES Convention, Vienna, Austria,* May 2007.

[4] Farina, A., “Simultaneous measurement of impulse response and distortion with a swept sine technique”, *Presented at the 108th AES Convention, Paris, France,* 2000.

[5] “Open Kinect,” http://www.openkinect.org/, March 2012 [August 1st 2012]

[6] “Microsoft Kinect” http://www.microsoft.com/en-us/kinectforwindows/, 2012

[7] “Official Kinect SDK vs. Open-source alternatives“ http://stackoverflow.com/questions/7706448/official-kinect-sdk-vs-open-source-alternatives , 2011

[8] “Microsoft Kinect” http://www.wired.com/magazine/2011/06/mf_kinect/all/, June 2011

[9] M. Carson et al., Surround Sound Impulse Response, Victoria, 2009 July.

Appendix A. Textbook Review

Chapter 12 – Moving Into Implementation

Managing Programming

With the design phase completed, it is now up to the project manager to assign tasks to programmers. Ideally, fewer the tasks the better as to minimalize coordination problems within programming teams. However, big programming teams are often unavoidable for complicated tasks so it is important to improve coordination with regular meetings, ensuring regulations are being followed, etc.

Finally, it's a manager's responsibility to account for delays and adjustments. When assigning tasks to different programming teams it is important to allow some leeway and flexibility. Equal as important is to ensure delays and mishaps are monitors and identified immediately to avoid major back-ups.

Testing

A big part of any technical project is testing. Such tests will occur throughout the implementation of the product and need to be carefully planned out and examine all cases.

A unit test examines a module or program within the system, where test cases are specific the to program code itself. Integration tests examine how different modules work together. Test cases have to be more complex, using scenarios and data flow diagrams to monitor interactions. A system test examines the entire project as a whole, where testing cases examine the key goals of the system as well as its flaws. Finally, depending on the product, acceptance testing can be conducted to determine if the system is acceptable to a user.

Documentation

In order to use any product a set of guidelines and documentation needs to be available to the customer. The textbook describes three types of user documentation: Reference documents provide aid for specific functions (ex. An online help system); procedure manuals describe how to perform certain business tasks; and finally tutorials teach people how to use the system. Navigation features also need to be readily available to enable users to find select topics and make it easy to interface with the system.

Chapter 13 – Transition to the New System

Making the Transition to the New System

The transition to a new system is discussed in the textbook with Lewin's three-step model or organizational: Unfreeze, move, and refreeze. The model serves as a guide when replacing an existing system. Figure 13-1 of the breaks down each step as follows: Unfreeze, using analysis and design techniques; move, create a migration plan that includes technical conversion and change management; refreeze, support and maintenance.

The Migration Plan

The migration plan describes the transition portion from the old system to the new. This transition includes not only the technological preparations, but also the user integration. People can be resilient to change; so all cases must be account for. Careful planning will minimalize costs for the transition and allow for smooth integration. The migration plan should include risk analysis, cost benefit analysis, a business contingency plan, and a preparation program for those affected.

Post-implementation Activities

The support system provides the online and help-desk support for users. Depending on the product and system it may include: a support staff to handle basic questions and answer phone inquiries, level 1; a support staff to handle more challenging problems and even request developers to implement bug fixes, level 2. Bug fixes are implemented by a system maintenance staff who are constantly revising the software. A project team documents the changes for future references and to learn from original mistakes.

Chapter 14 – The Movement to Object

Chapter 14 discusses the shift of system analysis and design to object-oriented techniques. The ideas are not new, but simply expand on the ideas from traditional methods into a more organized manner.

The idea of object-oriented system is to capture sections of information into modules or classes, which can be arranged in a hierarchical structure. Objects can interact with each other, share information, and can split a much bigger problem into smaller components. There is even a unified modeling language (UML) that provides a

common vocabulary of object-based terms and techniques to use during analysis and design. The textbook discusses a set of diagrams and techniques:

Sequence Diagram

A sequence diagram illustrates the objects that participate in a use case and the messages that pass between them. It often shows all possible scenarios for a use case, but can be split up in to instance sequence diagrams (page 519).

Actors, or objects, participating in the sequence are placed across the top of the diagram. A focus of control is a placed underneath each actor/object and denotes the when an object is sending or receiving messages. A message is a line drawn between the points of focus and conveys information to trigger the next activity. "Sequence diagrams are helpful for understanding real-time specifications and for complex scenarios of a use case" (page 529 of text).

Behavioural State Machine Diagram

The point of the behavioural state machine diagram is to "show the different states that a single class passes through during its life in response to events, along with its responses and actions" (page 524). A state is a set of value or amalgamation of properties to describe a certain point in time. It represents a point in an object's life, which satisfies some condition or input. Transitions are indicated by arrows and describe the action or event that leads to the next state. A state machine diagram can be used to clarify the steps an object will take and predict its next stage.

ELEC/CENG 399 Design Project I
Final Project Report Evaluation Form

To be filled by students:
Project title: Audio with the Kinect
Group #:7
Group members: Brian Richter, Nathan Willson, Drew Harris, Cameron Smith
Supervisor(s): Dr. Peter Driessen

To be filled by the supervisor(s):		
Progress report distributed to the supervisors for grading: Friday, August 3, 2012		
Please complete the progress report grading by: **Friday, August 17, 2012**		
Please refer to the rubric for grading.		
Topics		Grade [%]
[5%]Chapter 1, Goals :		
[10%]Chapter 2, Progress Overview:		
[25%]Chapter 3, Detailed Project Description:		
[25%]Chapter 4, Workload Distribution and Achievements:		
[10%]Chapter 5, Project Discussion:		
[5%]Chapter 6, Summary and Future Works:		
Subtotal [80%]:		**0.0**
To be filled by the instructor:		
[20%]Appendix A: Textbook Review	[10%]Write the textbook review in a clear	
	[10%]Meet minimum page requirement (2 pages):	
	Subtotal [20%]:	**0.0**
Total [100%]:		**0.0**

Comments by the supervisor: