DATE: August 31st, 2005
TO: Prof. Garry Hallock, Course Instructor
FROM: Divyanshu Vats, **UTEID:** vatsd, **EMAIL:** vats@ece.utexas.edu
SUBJECT: Project outline for senior design project to be taken in fall 2005 on designing and implementing a MATLAB toolbox for image-based rendering.

PROJECT OVERVIEW

Image-based rendering (IBR) has been a subject of investigation for several researchers. The first practical solution based on light field rendering was published in 1995. Before that techniques were either too complex or highly depended on getting geometric information of a scene. Recently industries are aiming at building 3D televisions which will enable viewers to watch games in a 3D way [1]. These technologies will highly depend on IBR techniques. Given the current popularity of IBR, it is important for researchers around the world to have a testing toolkit. The aim of this project is to build an introductory MATLAB toolbox for doing image-based rendering using the concept of light fields. During the process, some new techniques will also be discussed and implemented. Currently there is no such toolbox in any programming language. The sponsoring professor for this project will be Prof. Brian L. Evans.

DESCRIPTION OF PROJECT

The concept behind image-based rendering is to take images of a scene from various camera positions and be able to regenerate the scene from a virtual camera position. Two techniques have been popular so far, view dependent texture mapping (VDTM) [2] and light field rendering or lumigraphs [3, 4, 5]. Although VDTM requires less number of images, precise geometric information must be provided for decent reconstruction. This is mostly used for rendering objects having a well defined geometry. For lumigraphs almost no geometric information is required, but proper reconstruction is achieved only when a large number of input images are provided. Our toolbox will focus on building lumigraphs because extracting geometric information from a scene is a difficult problem.

Some work related to image-based rendering was done by me at Rice University during summer 2005 under the supervision of Prof. Richard Baraniuk. During that time a camera array was built to acquire images of an object and then using the images a partial lumigraph was created. Then by doing simple nearest neighbor interpolation followed by an averaging filter a reconstructed image was generated from some virtual camera position. The averaging filter was necessary because of the blocking artifacts in the final image. An example of a final reconstructed image from this system is shown in Figure 1. This image was rendered from about 400 images of a can from various camera positions. The camera was actually at a distance of about 60 cm and the image was reconstructed from a distance of 30 cm. The main input required for building the lumigraph are the camera parameters which include the camera position and some other internal parameters. These calculations were done by using a popular camera calibration toolbox published by Caltech [6]. The only downside is that the process requires us to keep a checkerboard in the background which restricts the number of views of the object we can take. In order to do calibration, we must click on four points on each image which can get tedious when we have 400 images. By doing some calculations, the number of images to click was reduced to about 40. The goal will be to further reduce the number of clicks required or to completely remove the need to click.

Figure 1 – Reconstructed image from current system

Another goal will be to make the reconstruction process faster so that it can be implemented in real-time. It took about 4 minutes to generate Figure 1 on an Intel® Pentium® 4 3.2GHz processor with 1GB of RAM. Improving the speed will require using some careful vectorized coding. In some cases it might be difficult to do so because of memory restrictions. The third goal will be to find an appropriate basis function which interpolates a pixel value using maximum amount of information available. The basis function should be such that it's dual can be easily implemented. This will ensure that for the already given camera positions, we don't get interpolated values. This will also make sure that the lumigraph information is maximized.

Another crucial problem to solve is compression. A lumigraph takes about 2GB of memory. This makes it difficult to transmit them commercially. There have been several techniques suggested for compression, which tells us about the amount of redundancy in lumigraphs. The goal will be to explore the possibilities of compression of lumigraphs and use one that easy to implement and at the same time efficient.

DESIGN CONTENT AND DELIVERABLE

For improving the camera calibration we will try to make use of the setup of the camera array system. The array is setup such that eight cameras point to a turntable and different views of the object are obtained by moving the turntable around. Currently in the first stage the camera array is calibrated for the relative camera positions and then images from one camera are calibrated. Using these all other camera positions are calculated. The goal will be to fix the amount of rotation on the turntable and calibrate these values before hand. This way the calibration process can be automated. Since the camera positions are already calibrated, we don't have to have the checkerboard in the background and thus we can get a full view of the object.

The main step in the reconstruction process that takes time is finding the point in the lumigraph that is at a minimum distance to some unoccupied point. This happens because the lumigraph is a 4D function and is very sparse depending on the number of images used. The goal will be to fasten this process through vectorized coding. If this fails to work because of memory restrictions, other ways of finding a nearest point will be explored.

For finding a suitable basis function with an easily implantable dual the approach will be start from scratch and see how different functions work. The first priority will be to make sure that the dual is easily implantable. The second thing to check is the quality of reconstructed images. The quality will be measured visibly or if possible through some mathematical formulation.

Compression of lumigraphs is a difficult topic and will not be studied in great detail due to lack of time. The main task will be to find a simple algorithm and implement it such that a lumigraph can be saved and transferred efficiently.

The final deliverable will be a fully functional MATLAB toolbox for doing image-based rendering. The toolbox will include functions for reading in images, calibrating them if necessary, building a user-defined lumigraph, reconstructing an image from a virtual camera position with a specific basis function, compressing, and decompressing lumigraphs. It will be fully documented and should be an excellent tool for learning about image-based rendering and doing further research in it.

TESTING

The initial testing will be done on rendered 3D images. This way we can get however many images and not worry about camera calibration. During the later stages, when most of the code will be written, testing will be done on actual images of objects. Some data sets were obtained from a camera array at Rice University during the summer 2005. Further Prof. Baraniuk has given me permission to use the camera array anytime during the semester. The testing will include seeing if the toolkit is user friendly, checking the quality of the reconstructed images, the speed of reconstruction, and the compression factor.

SCHEDULE

Aug 31 – Sept 12:	Vectorize code for making the current program faster. In addition collect literature on basis functions and lumigraph compression.
Sept 13 – Sept 24:	Find a good basis function whose dual is easy to implement.
Sept 24 – Oct 6:	Understand different compression algorithms and try to find one that is efficient and easy to implement.
Oct 6 – Oct 16:	Do the math for making the camera calibration faster.
Oct 16 – Oct 22:	Implement compression and decompression.
Oct 23 – Oct 31:	Complete writing of the toolbox
Nov 1 – Nov 12:	Test the toolbox on different set of images and look for errors.
Nov 13 – Nov 26:	Document the code and the toolbox
Nov 27 – Dec 9:	Publish code and write final report

Sponsoring Professor's Signature **Student Signature**

Prof. Brian L. Evans Divyanshu Vats

Date: _____ Date: _____

REFERENCES

[1] "Japan aims to create 3D television"
http://www.msnbc.msn.com/id/9007485/ (current 31 Aug, 2005)

[2] P. Debevec, C. Taylor, and J. Malik, “Modeling and Rendering Architecture from Photographs,” *SIGGRAPH 96*, pp 11-20.
[3] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen, “The Lumigraph,” *SIGGRAPH 96*, pp 43-54.
[4] M. Levoy and P. Hanrahan, “Light Field Rendering,” *SIGGRAPH 96*, pp 31-42.
[5] Chris Bueler, Michael Bosse, Leonard McMillan, Steven Gortler, Michael F. Cohen, “Unstructured Lumigraph Rendering,” *Proc. of the 28th annual conference on Computer Graphics and iterative techniques, 2001*, pp 425-432.
[6] “Camera Calibration Toolbox for Matlab” http://www.vision.caltech.edu/bouguetj/calib_doc/ (current 31 Aug, 2005)