Proceedings of the IASTED International Conference
GRAPHICS AND VISUALIZATION IN ENGINEERING
January 3–5, 2007, Clearwater, Florida, USA
ISBN Hardcopy: 978-0-88986-625-6 / CD: 978-0-88986-627-0

MIND: A TILED DISPLAY VISUALIZATION SYSTEM AT CATE/FIU

Javier Delgado, Magno R. Guillen, Mouncef Lahlou, Malek Adjouadi, Armando Barreto, Naphtali Rishe
Department of Electrical & Computer Engineering
Florida International University, Miami, FL
javier.delgado@fiu.edu, magno.guillen@fiu.edu, mouncef.lahlou1@fiu.edu , malek.adjouadi@fiu.edu,
Barretoa@fiu.edu, rishen@cis.fiu.edu

ABSTRACT

This paper reports the experience of building a highly efficient system for cluster visualization. This implementation was based on the well known master/slave concept which states that, given a "master" node, which is a node chosen to interact with the user and able to process the graphical user interface, splits its display into sections and sends it to "slave" nodes for their individual rendering. Chromium was used to perform the parallel remote rendering operation. In this paper, the detailed aspects of the Visualization Cluster implementation, called MIND, the hardware and software utilized in the project execution and the foremost issues found during the implementation were discussed. A cluster with 16 nodes was employed, connected with a Gigabit LAN. 15 servers, each driving a 20" LCD, were used as slave nodes and one as Master node. Furthermore, the benchmark test results are presented to relate the network bandwidth and the visualization speed of a given data size. An elite design in this system is the development of a script to automate the creation and/or insertion of data into the display node's configuration files using a XML-schema, in so doing, any Linux machine can become a compute node used for the visualization display system.

KEY WORDS

Parallel visualization, Chromium, DMX, ROCKS, Java-interactive-Profiler

1. Introduction

Current advances in the technology used for electronic display, together with the advances in computing power and network device speed, have made it possible to create a very high-resolution display platform by combining displays of several computers. Likewise, several software packages have been written for simplifying the process of creating these "tiled displays" or "mural displays". However, tying together all of the necessary software still remains as a very difficult task. In 2000, an approach to solve the tile display visualization problem was presented. This solution consisted of a Multidisplay based on commodity computing hardware and an innovative algorithm based on "metabuffer" concept [1]. In 2002, a visualization cluster was built using commodity computers running Open Inventor and Chromium [2]. In 2004, Teravision, a network-enabled PowerPoint projector for distributing and displaying scientific visualizations was introduced [3]. This system was intended to transmit graphics streaming between single workstations and clusters. In 2005, a Scalable Adaptive Graphics Environment (SAGE) was designed based on paradigms that decouple rendering and display processes [4]. Some of these are limited architecture-wise and some require a fresh installation of a specific operating system on all participating display nodes. Our goal was to find a methodology to create a mural display on a running system, without interfering anything currently running, i.e. without a need to reboot the machine. Also, the method is to work on any POSIX-compatible operating system.

2. Implementation Overview

2.1 Hardware

The system comprises a battery of 16 servers; namely Dell PowerEdge 1850 with dual Xeon 2.6 GHZ processor, 2GB RAM, 72 GB SCSI HD, each with a 20" Dell 2001FP monitor.

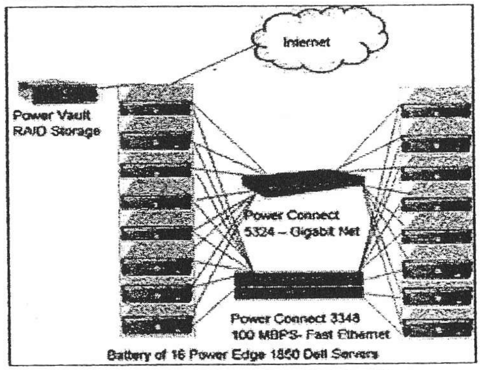


Figure 1: Network Connectivity for Mind Cluster

As a network storage, a Dell PowerVault 2205 (RAID Array) with 15 SCSI 72 GB HD is used to provide an excellent reliability for storage. As a backbone gigabit network for cluster communication, the PowerConnect 5324 was employed, also, a PowerConnect 3348 running at 100 MBPS is used as a secondary connection for administrative traffic.

Each server has two network interface cards, the first one was connected to the gigabit network and the second one connected to the 100 MBPS network. The network connectivity is shown in Figure 1, while Figure 2 shows a picture of the actual cluster.

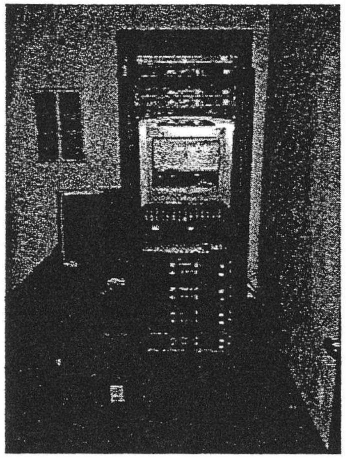


Figure 2: Mind Cluster at CATE/FIU

The 16 LCD were mounted on a structure of wood as shown in Figure 3. For our purposes, we considered there was no need to remove the edges of the LCD bezels. The 15 displays were organized in 3 rows and 5 columns inside the wooden structure.

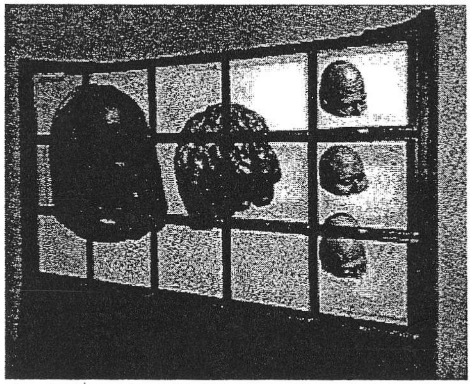


Figure 3: Final Appearance of the Tiled Mural Display

2.2 Software

Red Hat Enterprise Lin
system in all the servers.
distribution called Platfo
set of tools for the cluste
few words, Platform Ro
which deploys differe
manage, monitor and sc
Platform Rocks provid
software packages for
nodes. Figure 4 shows
Platform Rocks.

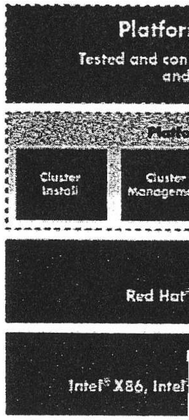


Figure 4: Module

For tile rendering we d
which is a system for in
workstations, along wi
Project (DMX) [8].
architecture to implemer

Image/J is an open sou
visualization, image pro
application was selecte
mural displayed [9].

To profile the code, t
Profiler (JiP) [10] was
loader and provides "re
It also factors out the
performance data, thus
that the program took t
resulting in very accura

3. Test Bed

As we described earlier
compute cluster using

system. This is similar to the systems developed by Brookhaven National Laboratory [2] and used the Viz Roll of Platform Rocks, which is an extension to the National Partnership for Advanced Computational Infrastructure (NPACI) Rocks operating system, except that the display wall was set up after the system was already installed and running, (the Viz roll requires a fresh OS installation.) Also, the concepts described here can be extended to any number of workstations that are connected through a Local Area Network, the system does not necessarily need to be a cluster, nor running any particular Linux distribution. A script was created to perform all of the operations. All that is needed is to have X windows and either the Gnome Display Manager (GDM) or the K Desktop Display Manager (KDM) and to be connected to the same LAN as the master node.

Although we utilized Rocks' "kickstart" scheme for preparing nodes, which made the process of preparing each "tile" of the mural display easier, we still had to find which files to modify and/or replace in order to get our display working. The process of doing this is outlined below.

4. Design Criteria

Several goals were defined before starting with the implementation of this mural-display system. First, user interaction with the display has to be transparent, i.e. starting and interacting with the display should not be different from interacting with a normal, single-headed workstation. Another goal was to make everything easy to setup, once all the necessary changes were done. That is, when each configuration file that needed to be modified (and how it needed to be modified) was known, applying these changes should be easy. We took this once step further and automated the creation and/or insertion of data into these files using the XML-schema used by Rocks for creating the "kickstart" files. We did mention that our setup could work with any system, not just Rocks. We used the Rocks installation mechanism (i.e. the "kickstart method) we are running Rocks. However, the same changes could also be carried out using system scripts, which would work on any system. The third goal, similar to the first one, was to allow simple starting and stopping of the display, i.e. doing so should be just as simple as doing it in a regular workstation.

5. Overview of Work

Several things needed to be done to arrive at a working mural display system. First, we chose a test node to work with. We had to determine which packages were to be modified and which were to be installed. Additionally, configuration files from several packages needed to be modified as part of the process. The first thing that needed to be done is determining exactly what packages were needed in each display node in order for the head node to be able to use it as part of the mural display. The X Windows system was an obvious requirement, without it we would be unable to display anything other than a command line or curses display. Once X Windows was installed, we realized that some of the fonts were missing, so several font packages were added to the list of packages. Then we noticed another problem: in order for the head node to access the display of other nodes, the other nodes would need to have a user already logged in, that allows others to access its display. Thus, we arrived at the conclusion that the most efficient way of doing this was to install a log in manager that allowed automatic, timed logins. For this, we chose the Gnome Display Manager. After checking the dependencies for this package we realized that we needed to install several other packages as well in order to use it. The final package that needed to be installed was Chromium, which is an application for displaying OpenGL graphics on a display wall. Although it is possible to do this with just XDMX [8], Chromium improves the speed of the graphics, since it just sends the instructions through the network, rather than the entire graphics information. Therefore, each display node's graphics processing unit renders the graphics on its own [5].

Once all the packages were in perception, several configuration files needed to be modified in order for the applications to behave as necessary. The following requirements had to be met. First of all, the display node needed to start in graphical mode, i.e. in a graphical run level. Also, when the desktop is loaded, a user needs to be automatically logged in, and that user must allow the head node to access its display. Determining the exact configuration files which needed modification required some program testing and some stack tracing and verbose shell debugging. The exact configuration options required some further debugging.

Once the test node was working, a scheme for automating the whole process of installing packages and modifying configuration files needed to be devised. For our setup, this was made suitable with the use of Rocks "kickstart" XML schema. This allowed us to select which packages were needed, what changes needed to be made to existing files, and even allowed us to insert entire files by creating a specially-formatted XML file. In Rocks, this same XML file could then be included when creating the image that is generated for all of the nodes of the cluster.

As previously stated, these concepts can be extended to any system, not just a Rocks Linux distribution. Rocks' kickstart method was the obvious choice in our case, so we used it. However, to ensure that the proposed script would work on any system, it was implemented and tested on various heterogeneous systems. The complete list of files to be modified, as well as the script, can be found in `http://mind.eng.fiu.edu/info/`.

5.1 Performance Tests

This mural display will be used for visualization of high-resolution medical images. We run a few tests to see how the performance of the display compares to displaying on a single-head display system, and also to see the effect of different transmission time and network delays based on change of backbone network technology (100 MBPS vs. 1 GBPS). We were also interested in the scalability of such a configuration, i.e. what is the effect of doubling the resolution (and thus doubling the network traffic and the number of displays needed) on rendering speed.

The method used for running this benchmark was profiling the code used for opening images. The popular image manipulation suite for Java, Image/J, was used for opening up the images. To profile the code, the open source Java interactive Profiler (JiP) was selected out of many profilers available. These two applications will allows us to look at the source code corresponding to the profiler's generated output in order to see what each method does and determine if it should be included in determining the total time taken to render the image on the display. The following tasks were recorded:

- Opening a 784x548 image
- Opening a 1568x1096 image
- Opening a 2385x1137 image
- Opening a 1450x579 image
- Opening a 2352x1644 image
- Opening a 3100x1100 image
- Opening a 3277x1550 image
- Opening a 4500x1796 image

The resolutions were chosen in order to utilize a specific number of displays. Three different images were chosen; the first one, which was used for the first three resolutions, is a reconstructed brain image, and had approximately 1:1 aspect ratio. The second one was a brain image and the corresponding head next to it; this image was more rectangular in nature, with approximately 2:1 aspect ratio. The last image was used for the 3350x1337 image, it displays two views of a reconstructed head at two different angles. This was a 5:3 image, which took up the entire display wall (15 nodes). In the results section, the number of displays used at each resolution is tabulated. Both images were TIFF-format. It would not have been a good idea to use a heavily-compressed format such as JPEG since decoding the image would have contributed to the time it takes to open the image, even though the profiler theoretically does a good job at separating the time spent decoding versus the time to display. In order to account for caching of the file-open code, a random image was first opened. Then, each of the subsequent images was opened three times each and the performance metrics recorded, as seen in tables 1 and 2.

5.2 Detailed Outline of th

As mentioned earlier, ea
profiled three times. Rathe
three times in a row, three
were performed. The In
(with the profiler running),
opened, then Image/J was
results from the profiler s
repeated for the next image
the final image. After that
again from the first image,
the set of second round run

Profilers list statistical d
method involved during
although the profiler allows
methods, in our test, none
result, it was necessary to s
order to determine which m
the time-to-display results.
and thus uses many calls
methods. Furthermore, si
image spawned several thre
90 interactions, and gener
lines of profile data. There
done through the profiler's

When analyzing the profil
was done was filtering out
time, i.e. less than 5 millis
out to be the majority of the
was done to look for i
rendering. There were sev
useful, such as mouse ev
components, opening dial
methods and classes with
were picked out (fortunatel
good naming practices).

The source code for these
Several methods for decod
the files, and reading t
However, the most import
"`paint`" method of the
"ImageCanvas" class' docu
"canvas used to display i
corresponding `paint` meth
of the image on the canvas.
significant method since w
the time that it takes for t
screen, i.e. painted on the
occurs before that is all d
were curious about the spe
from the head node to the d
fast the head node could ge

6. Results

After executing the tests using the 100 MBPS backbone, the average results were recorded as shown in table 1.

Table1: Average Display Time for images using Fast Ethernet 100 MBPS Network Connection

Image Size Pixels	Nodes Involved	Average Display Time [ms]
784x548	1	1041.27
1450x579	2	2344.50
1568x1096	4	4810.77
2385x1137	6	7614.63
3100x1100	8	9550.10
2352x1644	9	10717.37
3277x1550	12	14133.67
4500x1796	15	22241.63

The cluster was reconnected using a 1 GB switch; again the test was performed as described in the previous section. The average results are published in table 2.

Table 2: Average Display Time for images using Gigabit Network Connection

Image Size Pixels	Nodes Involved	Average Display Time [ms]
784x548	1	235.20
1450x579	2	365.60
1568x1096	4	669.73
2385x1137	6	987.13
3100x1100	8	1195.87
2352x1644	9	1342.80
3277x1550	12	1669.30
4500x1796	15	2461.83

The results showed how the image size displayed is related to the numbers of nodes employed. As the image size increases, more nodes are used, and more time is required to display the image. Using a 1 GBPS backbone, the average display time highly improved. Figure 5 depicts the performance improvement achieved with a 1 gigabit.

Figure 5 depicts the differences in visualization time while using a fast Ethernet vs. a gigabit backbone.

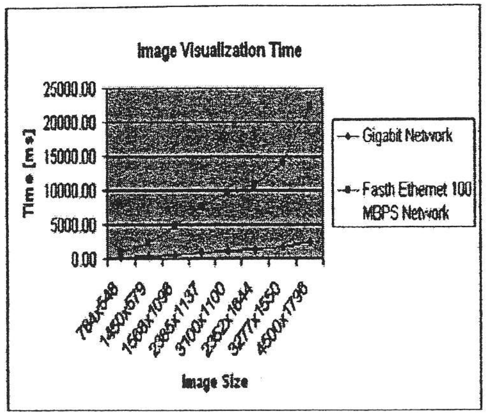


Figure 5: Image Visualization Time

The script was successfully tested in Red Hat Fedora, Gentoo, and Kubuntu, which are amongst the most popular distributions. The former two are using GDM and the latter two using KDM, proving to us that both work.

7. Conclusion

This paper described the implementation of a highly efficient platform for cluster visualization called MIND. The employments of hardware and software implementation were illustrated and the foremost implementation issues were discussed. MIND was successfully implemented at the Center of Advanced Technology and Education (CATE) at the Florida International University. This system will be used for several research projects such as 3D medical volume rendering, high definition video presentations, motion pictures computing, and web-based repository visual interactions.

Acknowledgments

The support of the National Science Foundation and the Office of Naval Research under grants EIA-9906600, HRD-0317692, CNS 042615, IIS-0308155, and N00014-99-1-0952 is greatly appreciated.

References

[1] W. Blanke, C. Bajaj, D. Fussel, and X. Xhang, The Metabuffer: A scalable Multiresolution Multidisplay 3-D Graphics System using Commodity Rendering Engines, *TR2000-16*, University of Texas at Austin, February 2000.

[2] S. Tomov, R. Bennet, M. McGuigan, A. Peskin, G. Smith, & J. Spiletic, Application of Interactive Parallel Visualization for Commodity Based Clusters using Visualization APIs, *Computers & Graphics, vol.* 28/2, April 2002, 273-278.

[3] R. Singh, B. Jeong, L. Renambot. A. Jhonson, J. Leigh, TeraVision: a Distributed, Scalable, High Resolution Graphics Streaming System, *proceedings of IEEE cluster 2004*, Sept 2004, San Diego, CA

[4] B. Jeong, L. Renanbot, R. Singh, A. Johnson & J. Leigh, High Performance Scalable Graphics Architecture for High Resolution Display, *Technical Paper*, Electronic Visualization Laboratory, University of Illinois at Chicago, May 2005.

[5] Platform Rocks, http://www.platform.com/Products/ Rocks [Accessed 2005 June 10]

[6] Humphreys G., Chromium Documentation, Version 1.8, http://chromium.sourceforge.net/doc/index.html [Accessed 2005 July 7]

[7] G. Humphryes, I. Buck, Y. Ng, R. Rank, S. Ahern, P. Kirchner, & J. T. Klosowski, Chromium: A Stream-Processing Framework for Interactive Rendering on Clusters, *Proceedings of* SIGGRAPH, 2002, 693-172.

[8] Distributed Multihead X Design - DMX, http://dmx.sourceforge.net/dmx.html [Accessed 2005 June 6]

[9] Image/J: Java Image
gov/ij/ [Accessed Jun

[10] Java interactive P
http://sourceforge.net/
6, 2006]

[11] Sanchez D., Adjou
"Registration of M
visualization of fiber
of the Sixth IASTEI
Imaging, and Image
Spain, Paper number

[12] M. Mourad, M. Adjo
Chapter 9: Registr
Maps to Magnetic
175-199, Book Tit
Research NOVA S
2006.

[13] M. Cabrerizo, M. A
An Application of
Analysis in Brain Fu
Book Title: **Progress**
NOVA Science, ISB

[14] M. Lahlou, M. Guil
Gaillard, "MEDIA
Repository site of fM
data for Childhood
Congress on Internet
Ontario, Canada, Oct

[15] Guillen, M. Adjouad
Web-Based Pediatri
Using OPNET", 4th I
Caribbean Confere
Technology, paper 1
Rico, June 21-23, 20

Graphics and Visualization in Engineering

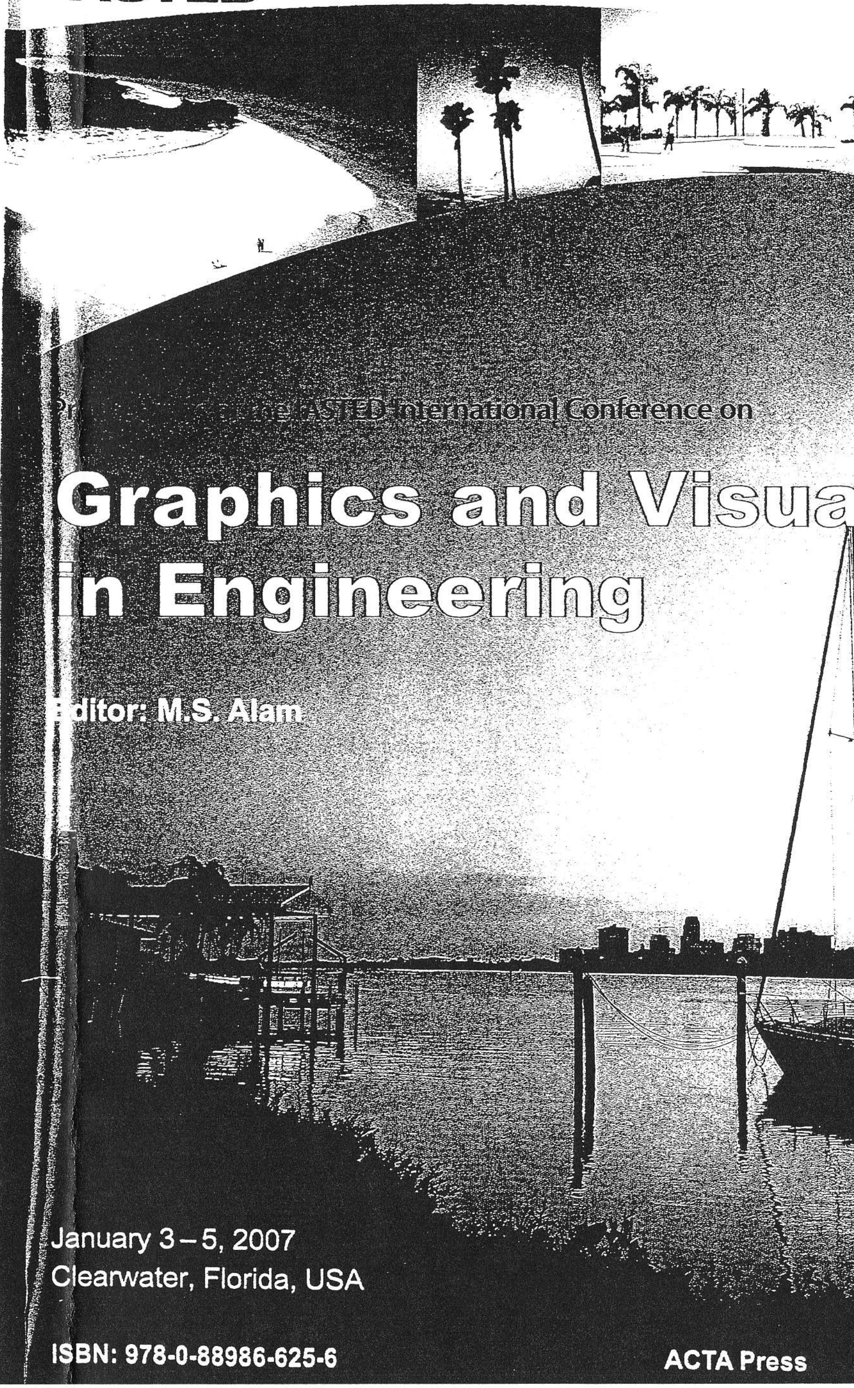
Editor: M.S. Alam

ISBN: 978-0-88986-625-6

562

IASTED

A Publication of the International Asso
of Science and Technology for Devel



Proceedings of the IASTED International Conference on

Graphics and Visua
in Engineering

Editor: M.S. Alam

January 3 – 5, 2007
Clearwater, Florida, USA

ISBN: 978-0-88986-625-6

ACTA Press

Proceedings of the IASTED International Conference on **Graphics and Visualization in Engineering**, held January 3 – 5, 2007, Clearwater, Florida, USA.

SPONSOR

The International Association of Science and Technology for Development (IASTED)
- Technical Committee on Graphics
- Technical Committee on Visualization

EDITOR

M.S. Alam – University of South Alabama, USA

KEYNOTE SPEAKER

T.K. Shih – Tamkang University, Taiwan

TUTORIAL PRESENTER

A. El-Saba – University of South Alabama, USA

INTERNATIONAL PROGRAM COMMITTEE

M.A.G. Abushagur – Rochester Institute of Technology, USA
M. Adjouadi – Florida International University, USA
E. Akleman – Texas A&M University, USA
J. Ben-Arie – University of Illinois at Chicago, USA
T. Calvert – Simon Fraser University, Canada
F. Cheng – University of Kentucky, USA
J. Cremer – University of Iowa, USA
P.K. Egbert – Brigham Young University, USA
R.F. Erbacher – Utah State University, USA
L. Giubbolini – WaveBand, USA
G. Grinstein – University of Massachusetts Lowell, USA
J. Hahn – George Washington University, USA
D. Hart – University of Alabama in Huntsville, USA
J.-Y. Hervé – University of Rhode Island, USA
S. Jaeger – University of Maryland, USA
L.J. Jia – Chinese University of Hong Kong, PRC
A. Kerren – University of Kaiserslautern, Germany
B. Kovalerchuk – Central Washington University, USA
F. Langbein – Cardiff University, UK
H. Levkowitz – University of Massachusetts Lowell, USA
B. McCane – University of Otago, New Zealand
T. Newman – University of Alabama in Huntsville, USA
V. Ostromoukhov – University of Montreal, Canada
E. Paquette – Superior School of Technology, Canada
V. Pascucci – Lawrence Livermore National Laboratory, USA
H. Qin – State University of New York at Stony Brook, USA
R.F. Riesenfeld – University of Utah, USA
S.A. Robila – Montclair State University, USA
Z.-C. Shih – National Chiao Tung University, Taiwan
P. Siebert – University of Glasgow, UK
H. Singh – Wayne State University, USA
P.L. Stanchev – Kettering University, USA
Y. Xiao – University of Akron, USA
X.D. Yang – University of Regina, Canada
S.G. Ziavras – New Jersey Institute of Technology, USA

ADDITIONAL REVIEWERS

R. Abdelfattah – Tunisia
A. Ben Hamza – Canada
L.-H. Chen – Taiwan
N. Cvejic – UK
P. de Carvalho – Portugal
T. D'Orazio – Italy
G. Georgiev – USA
P.M. Goebel – Austria
C. Grecos – UK
R. Jiang – Canada
J. Lee – USA
S.P. Kozaitis – USA
C.-C. Lu – USA
C. Popescu – Romania
J. Quinn – UK
I. Sengupta – India
J. Shen – USA
E. Soundararajan – USA
W. Tang – UK
J. Tavares – Portugal
B. Youcef – Algeria
L. Zeng – PRC

For each IASTED conference, the following review process is used to ensure the highest level of academic content. Each full manuscript submission is peer reviewed by a minimum of two separate reviewers on the International Program Committee/Additional Reviewers list. The review results are then compiled. If there are conflicting reviews, the paper is sent to a third reviewer.

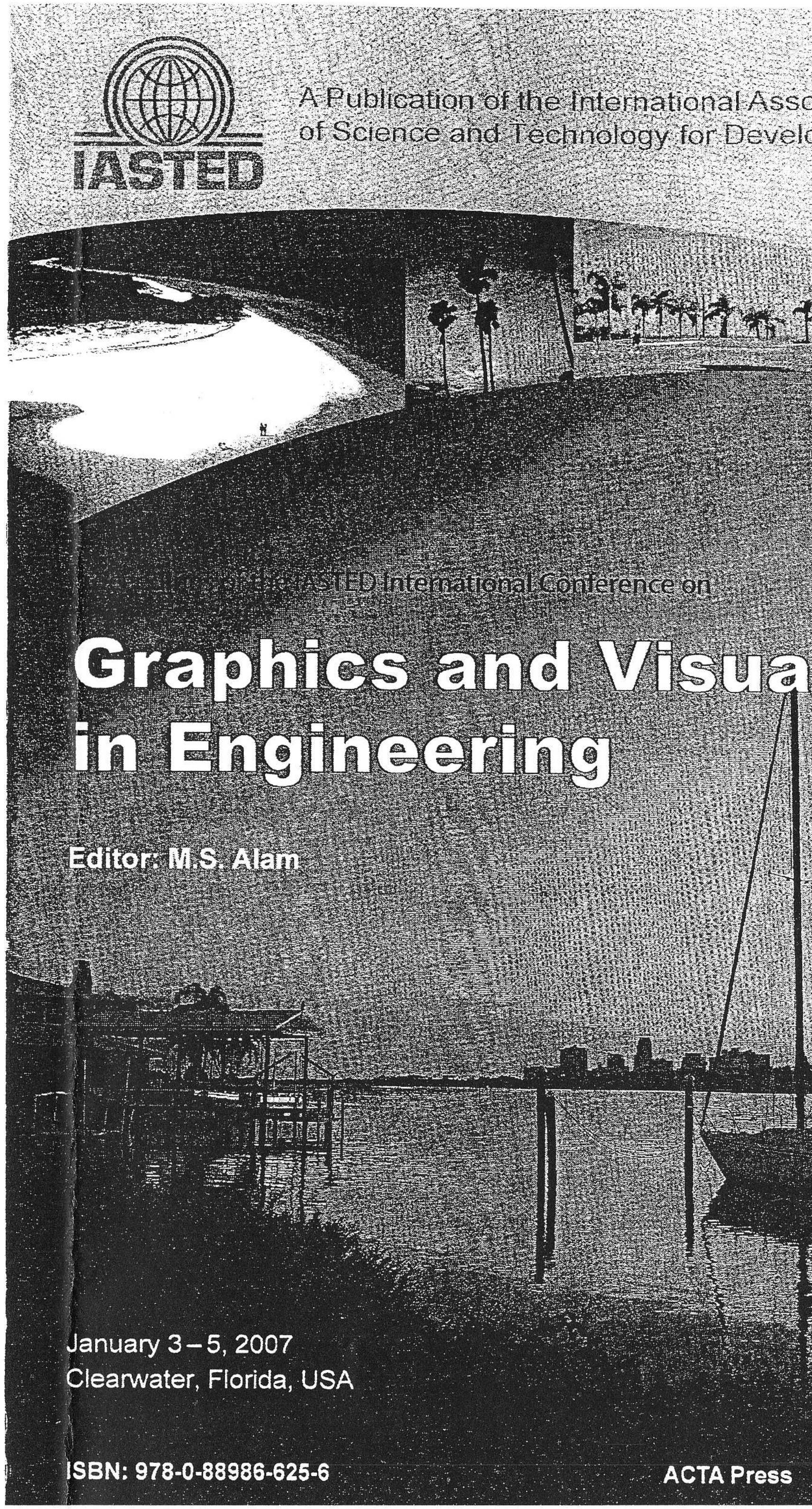
Photos © St. Petersburg/Clearwater Area CVB

ACTA Press
P.O. Box 5124
Anaheim, CA 92814–5124
USA

ACTA Press
B6, Suite #101, 2509 Dieppe Ave SW
Calgary, Alberta T3E 7J9
Canada

ACTA Press
P.O. Box 354
CH–8053 Zurich
Switzerland

Publication Code: 562



A Publication of the International Asso
of Science and Technology for Develo

[illegible] the IASTED International Conference on

Graphics and Visua
in Engineering

Editor: M.S. Alam

January 3 – 5, 2007
Clearwater, Florida, USA

ISBN: 978-0-88986-625-6

ACTA Press

$$\begin{aligned} \mathrm{comm}_{\mathrm{Maxwell}}(n) &= [A, \partial/\partial t\ A] / \sqrt{n} \\ &= 0 / \sqrt{n} \\ &= 0 \end{aligned} \tag{11}$$

We are interested in the limiting case

$$\lim_{n \to \infty} (\mathrm{comm}_{\mathrm{QED}}(n) - \mathrm{comm}_{\mathrm{Maxwell}}(n)) = 0 \tag{12}$$

For a system that has a very large number of photons, the quantum commutator behaves like the classical commutator, demonstrating that the limiting case of QED is Maxwell's equations. In most applications the number of photons actually is quite large and so the system behaves classically. But the quantum nature of the photon is always present, and is even evident in certain macroscopic systems (like the photoelectric effect), where Maxwell's equations cannot begin to explain the phenomenon.

How large is large for the number of photons? In the visible spectrum, red light has a wavelength λ of roughly

$$\lambda \sim 6 \times 10^{-7}\,\mathrm{m}$$

The energy $\mathcal{E}$ (in joules J) of a single "red" photon is

$$\mathcal{E} = h\frac{c}{\lambda} = 3 \times 10^{-19}\,J$$

Using a light source with power of 1 watt (1 J/sec), the number n of photons emitted per second is $1/\mathcal{E}$, or about 3×10^{18}. So even in a dimly lit scene, we expect a conventional (classical) renderer to produce accurate. That comes as no surprise; the point here is that we can quantify why classical illumination is good enough.

In order for the quantum field properties of photons in a rendered scene to make a difference, we must consider a situation where there is only a small number of photons. This can occur if the time interval for the light to be collected must be very small; or the light source is very dim; or the illuminated volume is very large so the photon density is low; or the rendered volume is a very small subset of the total space, containing only a few localized photons; or the wavelength of the light is very short but energetic (which means rendering a scene illuminated by gamma rays).

5. Conclusion

We summarized the essentials of quantum electrodynamics (QED) that are needed to relate it to classical electrodynamics. In brief, the photon states form a Fock space and are represented by linear combinations of kets and are acted on by a quantum field operator *A* defined via the least action together with a commutator relation. When the number of photons is large, the effect of the quantum commutator is negligible, and it asymptotically approaches the classical commutator for the vector potential A. It is in this sense that QED approaches classical electrodynamics as presented in Maxwell's equations.

Acknowledgments

This work was supported by NSF CCF #0430954 "Mathematical Foundations of Algorithms for Data Visualization." The authors gratefully acknowledge improvements suggested by the reviewers.

References

[1] Aleksandr I. Akhiezer and Vladimir B. Berestetskii, *Elements of quantum electrodynamics* (Chapter 3), translated from the Russian *Kvantovaya elektrodinamika* by the Israel Program for Scientific Translations, Jerusalem (London: Oldbourne Press, 1962).

[2] Barry G. Becker and Nelson L. Max, "Smooth transitions between bump rendering algorithms," *Proceedings of ACM SIGGRAPH 1993*, pp. 183-190, 1993.

[3] P. Beckmann and A. Spizzichino, *The scattering of electromagnetic waves from rough surfaces* (Pergamon Press, 1963).

[4] Jim Blinn, *Jim Blinn's corner: dirty pixels* (Morgan Kaufmann Publishers, Inc., 1998).

[5] Brian Cabral *et al.*, "Bidirectional reflection functions from surface bump maps," *Proceedings of ACM SIGGRAPH 1987*, pp. 273-281, 1987.

[6] Alexander L. Fetter and John D. Walecka, *Quantum theory of many-particle systems* (McGraw-Hill, 1971).

[7] Richard Feynman *et al.*, *The Feynman lectures on physics*, Volume 2: *the electromagnetic field* (Chapters 18-20) (Addison-Wesley, Inc., 1964).

[8] Andrew Glassner, *Principles of digital image synthesis* (Morgan Kaufmann, Inc., 1995).

[9] Jay S. Gondek *et al.*, "Wavelength dependent reflectance functions," *Proceedings of ACM SIGGRAPH 1994*, pp. 213-220, 1994.

[10] Michael Peskin, *An introduction to quantum field theory* (Harper Collins, 1995).

[11] David Griffiths, *Introduction to electrodynamics* (Chapter 10), 3rd Ed. (Prentice-Hall, Inc., 1999).

[12] Roy Hall, "Comparing spectral color computation methods," *IEEE computer graphics & applications*, 19(4), 1999, 36-46.

[13] John D. Jackson, *Classical electrodynamics* (John Wiley & Sons, 1975).

[14] Henrik W. Jensen and Niels J. Christensen, "Photon maps in bidirectional Monte Carlo ray tracing of complex objects," *Computers and graphics*, 1995, 19(2), 215-224

[15] James Kajiya, "Anisotropic Reflection Models," *Proceedings of ACM SIGGRAPH 1985*, pp. 15-21, 1985.

[16] James Kajiya, "The Rendering Equation," *Proceedings of ACM SIGGRAPH 1986*, pp. 143-150, 1986.

[17] Leonard Mandel and Emil Wolf, *Optical coherence and quantumoptics* (chapter 10) (Cambridge University Press, 1995).

[18] Franz Mandel and Graham Shaw, *Quantum field theory* (Wiley, 1993).

[19] James C. Maxwell, *A treatise on electricity and magnetism*, Vol. 1, unabridged 3rd ed. (Dover, 1991).

[20] Hans P. Moravec, "3-D graphics and the wave theory," *Proceedings of ACM SIGGRAPH 1981*, pp. 289-296, 1981.

[21] Mark S. Peercy, "Linear color representations for full spectral rendering," *Proceedings of ACM SIGGRAPH 1993*, pp. 191-198, 1993.

[22] P. Poulin and Alain Fournier, "A model for anisotropic reflection," *Proceedings of ACM SIGGRAPH 1990*, pp. 273-282, 1990.

[23] J. Ian Richards and Heekyung K. Youn, *Theory of distributions: a nontechnical introduction* (Cambridge University Press, 1990).

[24] L. H. Ryder, *Quantum field theory* (Cambridge University Press, 1996).

[25] David C. Tannenbaum *et al.*, "Polarization and birefringency considerations in rendering," *Proceedings of ACM SIGGRAPH 1994*, pp. 221-222, 1994.

[26] Paul Teller, *An interpretive introduction to quantum field theory* (Princeton University Press, 1995).

[27] Kenneth Torrance and Ephraim Sparrow, "Theory for off-specular reflection from rough surfaces," *Journal of the Optical Society of America*, 57, 1967, 1105-1114.

[28] Stephen H. Westin *et al.*, "Predicting reflectance functions from complex surfaces," *Proceedings of ACM SIGGRAPH 1992*, pp 255-264, 1992

[29] Lawrence B. Wolff and David J. Kurlander, "Ray tracing with polarization parameters," *IEEE computer graphics and applications*, 10(6), 1990, 44-55.

AUTHOR INDEX
GVE 2007