Combining a Shared-Memory High Performance Computer and a Heterogeneous Cluster for the Simulation of Light Interaction with Human Skin

Aravind Krishnaswamy and Gladimir V.G. Baranoski
Natural Phenomena Simulation Group
School of Computer Science
University of Waterloo, Ontario, Canada

Abstract

When light interacts with human skin, there begins a complex and involved process as the light is reflected and propagated by cells, fibers and other microscopic materials. This interaction happens countless times each day and its accurate simulation is essential to biomedical and computer graphics applications. Simulating this interaction is computationally intensive, yet highly suitable to parallelization. This paper describes the use of both a shared-memory high performance computer and heterogeneous cluster to accelerate these simulations. With a description of the parallel software used, we present results to show the performance gains from using such a hybrid approach.

1. Introduction

The interaction of light with human skin is a complex process involving many levels of absorption and scattering within the various layers of the skin. The understanding and predictive simulation of this interaction is relevant to a variety of areas such as medicine, realistic image synthesis and cosmetology. Medical conditions such as erythema (redness of the skin) and hyperbilirubinemia (yellowish hue commonly known as jaundice) can be simulated by considering the visible region of the light spectrum (380nm to 780nm). In addition, biological processes such as melanogenesis (production of the pigment melanin) and carcinogenesis (development of cancerous cells) can be simulated by considering the ultraviolet region of the light spectrum (100nm to 380nm).

Recently, we have developed a model to address the issue of light interaction with human skin named BioSpec [7]. BioSpec is a biophysically-based spectral model based on standard Monte Carlo techniques. In order to reduce the variance of the images generated using this model, a large number of samples are required, which results in simulation times on the order of hours or even days on a modern PC. This issue was further compounded when animations need to be produced to investigate changes in skin appearance and medical conditions. These simulation sequences involve the generation of a large number of images using the BioSpec model. We resorted to using a shared-memory high performance computing system, henceforth referred to as SMHPC, to reduce the computation time of our simulations. However, in order to further accelerate our simulations we included a heterogeneous cluster. We also needed to develop a common mechanism by which we could evaluate the performance of both the cluster and the SMHPC. In this paper we describe how we implemented these simulations using this hybrid approach.

The approach presented in this paper can be easily applied to the simulation of various other organic materials such as plant leaves, stems, muscle tissues, ocular tissues, blood, hair and fur. These simulations have various uses in the areas of computer graphics, remote sensing and biomedicine.

We first outline the underlying biological aspects of human skin and its simulation. Our parallelization strategy is presented next. We then present the proposed metric for evaluating performance. Some results showing the performance gain from this strategy are presented. Finally, we close with a summary and directions for future work.

2. Light Interaction with Human Skin

2.1. Biological Aspects

Human skin tissue is made up of multiple layers and it is inhomogeneous [1, 11]. The stratum corneum is the outermost layer and is characterized with little absorption. Following the stratum corneum is the epidermis which propagates and absorbs light. The absorption comes primarily from the chromophore melanin. It is the presence of melanin that gives skin its different tones. Below the epidermis is the dermal layer which also propagates and absorbs light. The absorption in the dermis comes mainly from

the blood borne chromophore hemoglobin. It is hemoglobin that gives blood its reddish color. The hypodermis is a tissue below the dermis and is primarily reflective.

The scattering profile of human skin has two components: surface and subsurface scattering. Surface scattering follows Fresnel equations [10] and is affected by folds in the stratum corneum. Only 5-7% of light incident on the stratum corneum is reflected back; the remainder is transmitted to the other layers. The stratum corneum and epidermis are highly forward scattering media [3]. In the dermis, the spatial distribution of scattered light becomes diffuse. In addition, the presence of collagen fibers are responsible for Rayleigh scattering [6] which produces larger variations on the low end of the light spectrum.

2.2. Simulation

The BioSpec model considers the stratification of human skin into four main layers: stratum corneum, epidermis, papillary dermis, and reticular dermis. Light propagation within these layers is described in terms of ray optics, where light is assumed to be composed of non-interacting straight rays, each of them carrying a certain amount of energy[4]. Each ray of light has a wavelength of light (λ) associated with it. We follow the path of a ray of light as it travels through the medium until it is either absorbed or reflected back to the environment. The propagation of light through these layers is simulated as a random walk process [4]. The transition probabilities from one layer to the next are associated with the Fresnel coefficients, which are computed at each interface. The termination probabilities within a particular layer are associated with the free path length computed when a ray travels within a layer.

Within each of the layers of skin, there are two important biophysical processes that must be taken into account, namely scattering and absorption. When a ray enters a particular layer, it is first scattered. This scattering is accomplished by applying a warping function given in terms of the polar (α) and azimuthal (β) perturbation angles. In our simulation, the azimuthal perturbation angle for all layers is:

$$\beta_l = 2\pi\xi, \tag{1}$$

where:

- β_l = azimuthal perturbation angle for that particular layer,
- ξ = an uniformly distributed random number in the interval [0..1].

The manner in which the polar perturbation angle is computed varies from layer to layer. In the stratum corneum and epidermis we use a data driven look-up table approach to directly compute α [2]. Since scattering becomes quickly diffuse in the dermis, α is $\arccos(\sqrt{\xi})$. We also account for Rayleigh scattering in a similar manner [7].

Once a ray has been scattered in a layer, it is tested for absorption. If it is absorbed, the ray is terminated; otherwise, the ray is propagated to the next layer. The absorption test is based on Beer's law [12] and it consists of computing the ray free path length ($p(\lambda)$) using the following expression:

$$p(\lambda) = -\frac{1}{a_i(\lambda)} \ln(\xi) \cos\theta, \tag{2}$$

where:

- $a_i(\lambda)$ = total absorption coefficient of pigments of given layer i,
- θ = angle between the ray direction and the specimen's normal direction.

If $p(\lambda)$ is greater than the pigmented medium, the ray is propagated, otherwise it is absorbed. We continue this cycle of scattering and absorption within the layers of skin until either the ray leaves the skin or is absorbed.

3. Parallel Strategy

Due to the stochastic nature of the random walk process, the path that each ray of light takes through the skin layers is independent of another. Since the amount of data to process is relatively small, we employ a demand driven scheduling algorithm [8]; *i.e.*, every node has a full copy of all the data. Communications between the nodes of our cluster and the SMHPC was implemented by a custom lightweight packet oriented protocol which uses TCP/IP as the transport and network layers respectively. We chose to use a custom protocol to satisfy two requirements. We required a protocol to span several platforms including both Windows and Irix and one which was efficient enough for communication over the Internet and could support the sudden loss of a node. Our protocol uses a simple command system, where each packet contains a command or identifier and some chunk of data specific to the command or identifier. 'GetClientType', 'Disconnect', 'SceneFile', and 'NewCell' are just a few examples of some of the packet identifiers/commands. Figure 1 illustrates how the central server, cluster nodes and the SMHPC nodes are connected. There are three components in our algorithm: the central server, a cluster client node and the SMHPC client node. The appendix provides some broad pseudo code on the operations these components perform.

The final result for our simulation is an image (or a frame of an animation). For each pixel in the image, we use a computer graphics technique called ray tracing [9] to compute the color of that pixel from our simulation.

Our divide and conquer strategy subdivides the image into blocks of pixels. Each node is responsible for running the simulation algorithm on one block of pixels. Once

a node has completed its assigned block, it receives another block from the server. This process is continued until there are no more blocks remaining to process. Essentially, we can simultaneously process N blocks of pixels at any time. By only allowing nodes to operate on one block at a time, this scheduling algorithm implicitly adjusts for different performance characteristics of each node, giving us dynamic load balancing. The total number of blocks any particular node processes will vary depending on the speed at which that node runs the simulation.

We chose to represent the entire SMHPC as one super node. The block it receives from the central server are distributed among the individual processors of the SMHPC. Each processor performs S/p samples, where S is the total number of samples and, p is the number of CPUs in the SMHPC. Communications between the processors of the SMHPC system was accomplished using the Message Passing Interface (MPI) standard [5]. It is worth noting that we could have simply treated each processor on our SMHPC as a node and used just our custom protocol. However, our SMHPC and cluster were not physically colocated. Instead, they were connected via the Internet, which has motivated this hybrid scheme. The processor of rank 1 on the SMHPC communicates with the central server. It also combines and optimizes the processed blocks from all the processors before sending them to the server. This allows our scheme to be more efficient than treating every processor on the SMHPC as a cluster node.

The granularity and size of the blocks of pixels assigned to nodes is a user configurable parameter.

4. Evaluating Performance

In order to be able to determine if a particular node is faster than another and to measure performance, we require a metric to quantify a node's performance. Since we are dealing with heterogenous nodes with different processor and memory architectures, using a metric such as the clock frequency of the processor is inadequate. Furthermore, since elements such as cache sizes and memory speeds may not be fully represented by computing the number of floating point operations per second (flops), we chose to use the simulation software itself in evaluating the performance of a node. Our proposed metric, called performance rating (Pr) is specific to our application and has two essential qualities:

- a higher performance rating indicates better performance, and
- a linear increase in performance rating indicates a linear increase in performance.

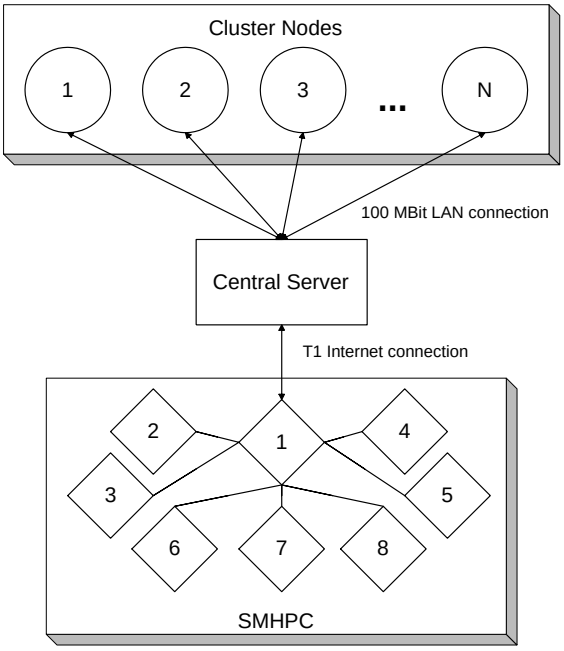

Figure 1. Diagram of how the nodes of the cluster are connected to the central server which is connected to the shared-memory high performance computer (SMHPC).

The basic performance rating is given by:

$$Pr = \frac{1}{t}, \tag{3}$$

where:

t = time (in minutes) to compute a test simulation.

However, performing a full test simulation can take a considerable amount of time. Instead, we only perform the computations on a certain subset of the final output pixels. We select our pixels by using a multi-stage N-rooks sampling method [13]. Our experiments indicate that selecting 10000 pixels for the sample yields results with less than 5% variance. Also, this number of pixels can be simulated in a matter of seconds on most PCs.

Efficiency is compared using this performance rating, rather than the number of the processors. This allows us to evaluate the performance of the cluster in an abstract manner and allows us to compare its performance to that of the SMHPC. We can also sum the individual performance ratings of the nodes to describe the computational power of all the nodes combined.

We use the traditional representation of speed-up, which is:

$$S_p = \frac{T_s}{T_p}, \tag{4}$$

where:

T_s = running time for fastest sequential algorithm on any one node,

T_p = running time for parallel algorithm.

The efficiency, however, must account for the difference in performance rating of one node versus the sum of all the nodes and is represented as:

$$E_p = \frac{S_p * Pr}{Pr_c}, \quad (5)$$

where:

Pr = performance rating of the node running sequential algorithm,

Pr_c = sum of performance rating of all nodes.

5. Results and Discussion

The SMHPC used in our simulations was an SGI Origin 3200 series computer with 8 MIPS R14000 processors. Each processor has a dedicated 1Gb of memory. Our cluster was made up of 3 dual processor Xeon PCs ranging from 2.4 to 2.6 GHz, two dual Opteron PCs and several Pentium III and Pentium IV PCs ranging from 800 MHz to 3.06 GHz. The HPC was running Irix 6.5, and most of the cluster nodes were running Windows XP with a few running various Linux distributions.

We performed two sets of simulations for one frame of data involving different numbers of samples: 128 and 4096. Figure 2 shows the speed-up gain as more nodes are added. Figure 3 shows how the efficiency of the system changes as more nodes are added. By examining the graph we notice that our parallel strategy works best when a larger number of samples is used. In fact, further experiments have shown that the speed-up of our parallel algorithm increases almost linearly as many new nodes are added as long as the number of samples is also increased. The reason for this is that, as the number of samples is increased, each node spends more time in computation relative to communication.

One of the goals during our research was the visual simulation of skin conditions, such as erythema, which is caused by increased blood presence in dermal tissues, triggered by some chemical, mechanical, electrical, thermal or luminous stimulation. As anyone who has experienced a severe sunburn or the first stages of frostbite is aware, this can cause skin to redden significantly. We wished to generate an animation several seconds long demonstrating the appearance of erythema in the skin surrounding a human ear (Figure 4). The use of the hybrid approach presented in this paper was crucial to our ability to generate these animations in a reasonable amount of time (approximately 8.5 hours for 120 frames with 4096 samples).

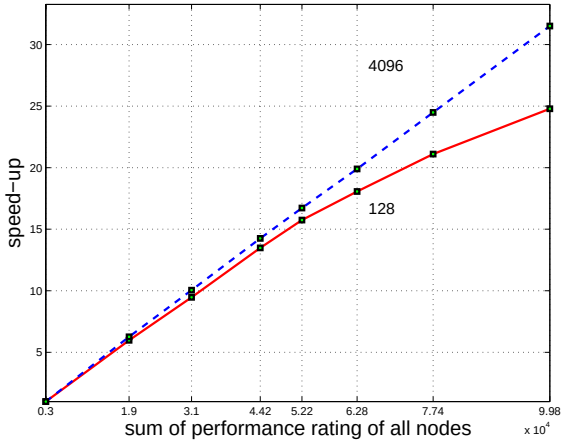

Figure 2. Comparison of speed-ups for two sets of experiments involving 128 and 4096 samples.

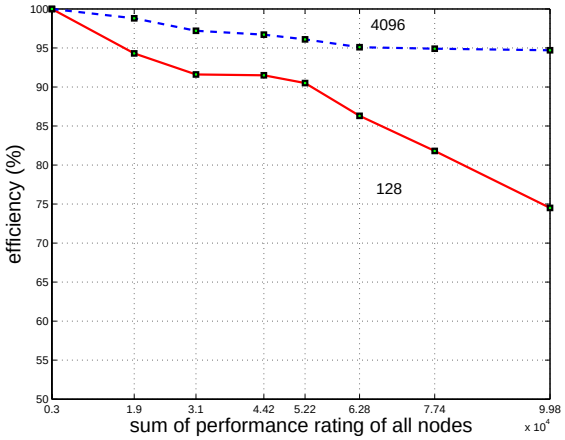

Figure 3. Comparison of efficiency for two sets of experiments involving 128 and 4096 samples.

6. Conclusion and Future Work

In this paper we have presented a parallel strategy combining a SMHPC and a heterogeneous cluster for the purpose of simulating light interaction with human skin. Our results show significant speed-up gains over a sequential algorithm. They also show that our parallel strategy performs better as the number of samples is increased.

The parallelization technique presented in this paper can be applied to the simulation of light interaction with other organic materials using similar simulation algorithms. Plant tissues (leaf and stem), muscle tissue, blood, hair, fur and ocular tissues are an example of the organic materials that

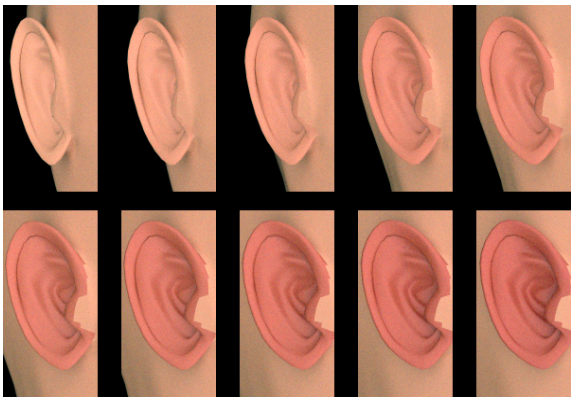

Figure 4. Frames of an animation showing the onset of erythema in the skin of a human ear as the head rotates.

can be simulated with applications in areas such as remote sensing, computer graphics and biomedicine.

Future efforts will involve the use of asynchronous communication, so that nodes can continue the simulation of another set of data while previously computed results are transmitted back to the server. We believe this will allow the entire system to be scaled more efficiently. In addition, we would like to explore the decomposition of data sets such that larger amounts of data can be efficiently processed by the SMHPC while smaller ones are processed in parallel by a cluster.

We would also like to create a repository to save the variety of generated data. This would allow future researchers access to comparison data for further experiments of this kind.

Acknowledgements

The authors would like to thank the reviewers for their insightful and helpful suggestions. In addition, we would like to thank Inscriber Technology Corporation for providing us with the cluster. The work presented in this paper was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC grant 238337) and the Canada Foundation for Innovation (CFI grant 33418).

References

[1] R. Anderson and J. Parrish. The optics of human skin. *Journal of Investigative Dermatology*, 77(1):13–19, 1981.

[2] G. Baranoski, A. Krishnaswamy, and B. Kimmel. Revisiting the foundations of subsurface scattering. Technical Report CS-2003-45, School of Computer Science, University of Waterloo, December 2003.

[3] W. Bruls and J. van der Leun. Forward scattering properties of human epidermal layers. *Photochem. Photobiol.*, 40:231–242, 1984.

[4] A. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann Publishers, Inc, San Francisco, 1995.

[5] W. Gropp, E. Lusk, and A. Skjellum. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. MIT Press, USA, 1999.

[6] S. Jacques. Origins of tissue optical properties in the uva visible and nir regions. *OSA TOPS on Advances in Optical Imaging and Photon Migration*, 2:364–369, 1996.

[7] A. Krishnaswamy and G. Baranoski. A biophysically-based spectral model of light interaction with human skin. *Computer Graphics Forum (Proceedings of the Annual Conference of the European Association for Computer Graphics - EUROGRAPHICS, Grenoble, France)*, 2004. To appear.

[8] E. Reinhard. *Scheduling and Data Management for Parallel Ray Tracing*. PhD thesis, University of Bristol, Bristol, United Kingdom, 1999.

[9] P. Shirley. *Realistic Ray Tracing*. A K Peters, USA, 2000.

[10] Y. Su, W. Wang, K. Xu, and C. Jiang. The optical properties of skin. In *Optics in Health Care and Biomedical Optics: Diagnostics and Treatment*, pages 299–304. SPIE, vol. 4916, 2002.

[11] V. Tuchin. Light scattering study of tissues. *Physics-Uspekhi*, 40(5), 1997.

[12] V. Tuchin. *Tissue Optics Light Scattering Methods and Instruments for Medical Diagnosis*. The International Society for Optical Engineering, Bellingham, WA, USA, 2000.

[13] C. Wang and K. Sung. Multi-stage n-rooks sampling method. *Journal of Graphics Tools*, 4(1):39–47, 1999.

Appendix: Pseudocode

In this appendix, we present broad pseudo code for the three components of our system.

The first component is the central server. The central server always waits for a connection. When a client connects, it performs the following operations:

```
Do handshaking
Get any completed blocks
If no more blocks
  Write completed image to disk
  Exit
End If
Ask client how many blocks it wants
Send blocks to client
Disconnect
```

The second component is the client running on the nodes of our cluster. The client, when launched, performs the following operations:

```
While TRUE
  Connect to server
  Do handshaking
  If we have a processed block
```

```
      Send processed block to server
    End If
    Ask for a block of pixels
    If no more blocks
      Exit
    End If
    Disconnect
    Process block
  End While
```

Finally, the third component is the client running on the SMHPC, and it performs the following operations:

```
While TRUE
  If rank == 0
    Connect to server
    Do handshaking
    If we have a processed block
      Send processed block to server
    End If
    Ask for a block of pixels
    If no more blocks
      MPI_Send exit code
      Exit
    End If
    Disconnect
  Else
    MPI_Recv status
    If status is no more blocks
      MPI_Finalize
      Exit
    End If
  End If
  MPI_Barrier
  Process of rank i performs (S/p) simulations
  If rank == 0
    MPI_Receive all completed blocks
    Combine completed samples
    Optimize completed block
  Else
    MPI_Send completed block
  End If
End While
```