International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-3 Number-3 Issue-11 September-2013

Interactive 3D Remote Visualization of Medical Volume Data in Distributed Environment

Alok Pratap Singh[1], Piyush Kumar[2], Anupam Agrawal [3]

Abstract

Medical volume data is often large in size and computationally intensive. It takes lot of memory space for storage, retrieval purpose and also it needs high computational environment. The objective of this paper is to enable the user to work with volumetric data without having specialized software for visualization and without having large data on its local machine. To perform slicing on volume data Marching cube algorithm and for visualization purpose we are using Ray casting algorithm. For the further implementation of these techniques in distributed systems we are using socket programming along with VTK. This system has used four types of volume datasets like .raw, .stl, .ply, and .vtk available in public domain.

Keywords

Ray casting, Marching Cube, Volume Data, Distributed Visualization.

1. Introduction

All The medical Data generated by currently available 3D scanners and utilized in medical applications is very large in size. Due to bandwidth limitations and other networking constraints, generally it is difficult to provide the facility to interact with volume data.

It enables to find out useful visual information from huge complex volumetric data created in various engineering and scientific areas. In Fig. 1 human abdomen has been visualized. Volume visualization enables to find out useful visual information from huge complex volumetric data [12] created in various engineering and scientific areas [2].

Alok Pratap Singh, Dept. of Information Technology, IIIT Allahabad, Allahabad, India.
Piyush Kumar, Dept. of Information Technology, IIIT Allahabad, Allahabad, India.
Anupam Agrawal, Dept. of Information Technology, IIIT Allahabad, Allahabad, India.

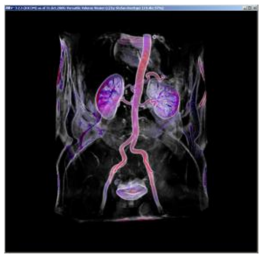

Fig. 1. Volume Data Visualization [1]

In many fields such as medical science, engineering, fluid dynamics, space and earth, volume data is so much of huge size. The size of the volume data ranges from hundred megabytes to many gigabytes. So it needs a lot of effort and care to work with such a massive data. Many algorithms for storing, manipulating and transmitting the volume data, have been developed to make it easy to work with volume data.

There are three ways a client can interact with volume data in distributed environment: In the first method the whole 3D volume data is transferred on client side. It can perform any operation on the downloaded medical volume data as per its needs. This method requires that client must have complex software installed on its machine. The serious problem associated with this approach is that a client has to download every dataset which it needs.

In the second method instead of sending every volume data on the client side only small data are sent to the client. If the client wants to perform some operations on large data, the request is sent to the server. The client specified operation is performed on the data which is located at the server and the result is sent back to the client. This approach is better than previous one. But in this approach also, the client needs to have specialized software to work with volume data.

According to third approach server has all the computing resources. All the processing is done at server side. We have chosen third technique to assist distributed volume rendering.

This paper will go through some related work of volume visualization in distributed system in next section. The third section describes the proposed methodology. A few interactive result snapshots and performance analysis results are summarized in fourth section. Finally we have concluded this entire work with the future scope in fifth section.

2. Related Work

After Sanchez et al. [5] suggested the method for compressing the medical data using optimized volume of interest coding. Frey et al. [3] proposed an approach i.e. considering load balancing for data redundancy in distributed volume rendering [18].
Engel et al. [11] suggested a method in which the capabilities of client and server were combined to visualize large medical volume data. Remote visualization has been done for the historical data [21]. Markon et al. [14] gave a method for visualization on floating images. These images can be controlled using gesture recognizing devices. Interaction with the volume data will be done with the help of gesture recognizer. Liu et al. [15] proposed a technique in which the whole data is partitioned into buckets. For partition the data into buckets, the octree method is used. Multimedia medical data in Distributed environment for health sytem have been developed [18]. Handling of large scale dataset [16] is difficult for rendering in distributed system. To implement platform independence JavaView [9] has been used in Polthier et al. [10]. Wei et al. [6] proposed a method to assist distributed volume rendering using VTK [13], C++ libraries and using applet and servlet for communication between client and server.

3. Methodology

The technique we have used for visualization and working with 3D volumetric medical data is discussed below. In our approach we have divided the whole system into two parts-
1. Server
2. Client
Client has java installed on its system so that it could connect to the server using socket programming. Client - Server architecture has been followed.
Server has two modules, one module to handle client request such as accepting the connection of the client sending response in the form of images and receiving input from client in the form of events, generated at client side.

Server
The flow chart of the server is shown in Fig. 2.

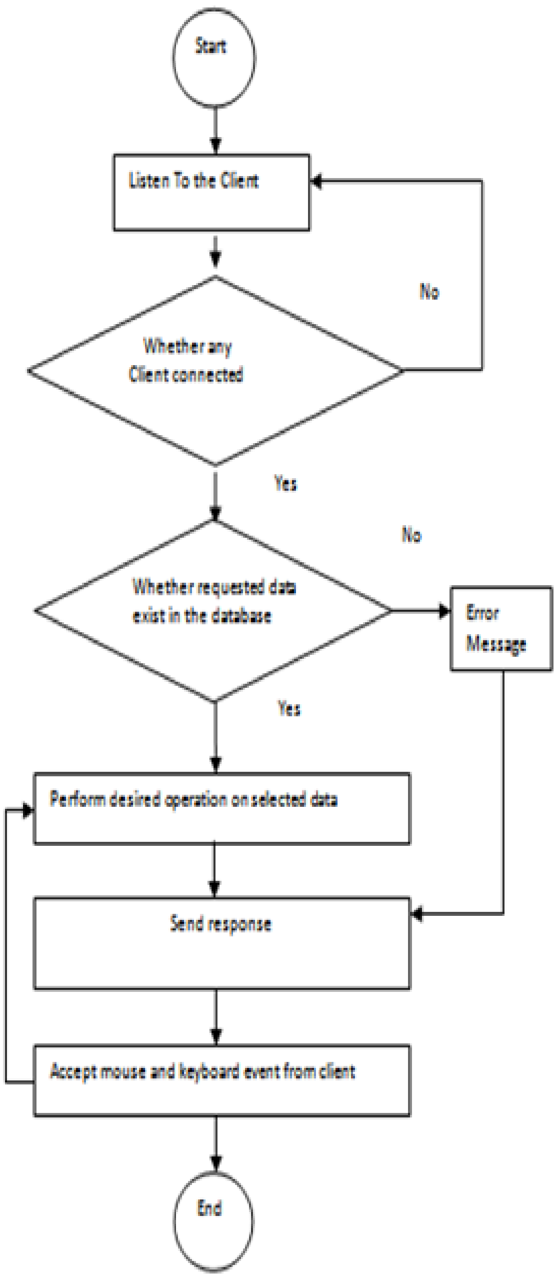
Start
Listen To the Client
Whether any Client connected
No
Yes
Whether requested data exist in the database
No
Error Message
Yes
Perform desired operation on selected data
Send response
Accept mouse and keyboard event from client
End

Fig. 2. Server Flow Chart

Client
Client Flow chart is shown in Fig. 3. The first step is to connect to the server for this client gives the server IP and the port number. After connecting to the server the client sends the parameters of the operation to be performed with the id of the volume data to the server. When client receive first response from server

in the form of image it work on the image and sends back the click event to the server. Server receives these events and sends back the series of resultant images.

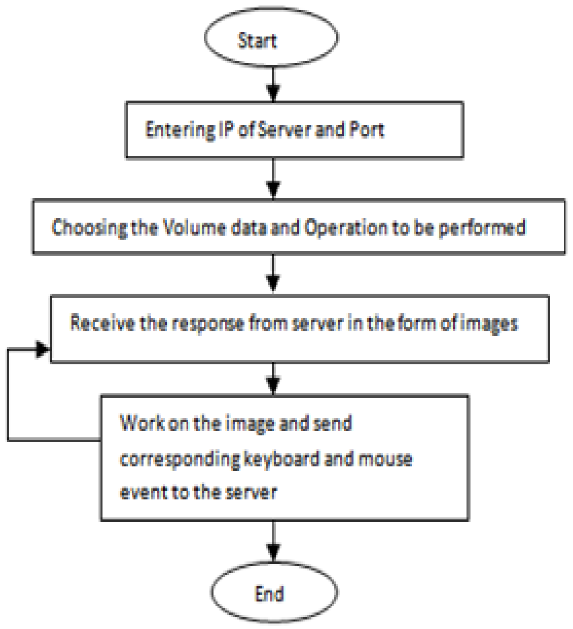
Start
Entering IP of Server and Port
Choosing the Volume data and Operation to be performed
Receive the response from server in the form of images
Work on the image and send corresponding keyboard and mouse event to the server
End

Fig. 3. Client Flow Chart

Communication

Client-server communication diagram is shown in Fig. 4. First server is started which waits for the client request. When a user runs the client code, server id and port of the server will be asked. IP address and Port No. can unequally identify any process in any network. Now the request will go to the server. Server initially sits idle and waits for the client. As soon as a client is connected to the server, the server determines the need of the client and accordingly chooses the medical data and operations to be performed on the data. The server socket is created and the graphics environment of the server is sent to the client using graphics environment class. Now the Screen Capture class is invoked.

Screen capture class actually captures the image that is drawn in VTK Render Window. There is a class in VTK libraries VTKCaptureScreen() [13], this class takes the snapshots of the VTK Render window and saves in some location. These snapshots will be sent to the client. So that user gets a series of images continuously and feels that actually he is working on data interactively.

At the server there is a delegate class, this class inherits the Scanner class. Scanner class actually recognizes the mouse and keyboard events those are happening at client side. For mouse up, mouse down and other mouse positions and keyboard events, there is a unique integer number generated at client side, which is recognized by the server so that server can understand which operation client wants to perform, such as rotating and panning of the objects, changing the skin transparency and color of human body. As soon as client fires any mouse event or any keyboard event on the image that it has received from the server, the event is passed to the server using delegate class and server recognizes the event that has been fired by client and acts accordingly.

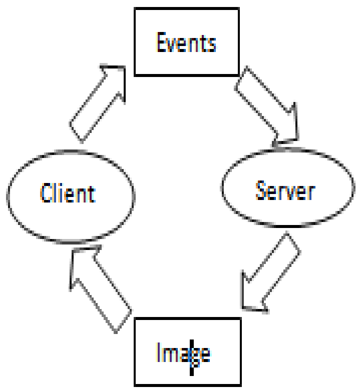

Fig. 4. Communication Diagram

VTK(Visualization Tool Kit)

VTK include C++ libraries, these libraries are used for volume data processing. VTK include libraries for performing various scalar tensor vector operations on volume data. VTK pipeline involves stages as shown in Fig. 5.

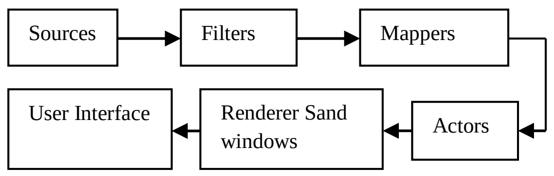

Fig. 5. Vtk Pipeline

Some of the libraries those are used in implementation of the proposed system are given below- vtkPiecewiseFunction
vtkColorTransferFunction

vtkVolumeProperty
vtkRayCastCompositeFunction
vtkRayCastMIPFunction
vtkBoxWidget
vtkConeSource
vtkCubeSource
vtkPlaneSource

Ray casting

In ray casting algorithm parallel rays are casted from the view point to the viewing volume. Attenuation which is caused by particle field is computed for every point which is in the path of the ray. The light scattered from the scene toward the viewing point is also calculated for each point along the ray. These values are combined together and single brightness is calculated for every ray.

In Fig. 6 given below there is a volume of density D(x, y, z) which is penetrated by a ray R. The density function is given as

$D(x(t),y(t),z(t))=D(t)$

Whereas Illumination can be given as-

$I(t) = I(x(t), y(t), z(t)) \qquad (1)$

Illumination scattered will be given as-

$I(t)D(t)P(\cos\Theta)$

Where Θ is angle between R and L

If there are many light sources, illumination scattered will be calculated as follows-

$\sum I_n(t)D(t)P(\cos\Theta_n)$

Attenuation due to the density function along a ray is calculated using the formula given below –

$$\exp\left(-f\int_{t_1}^{t_2} D(s)\,ds\right)$$

Where f is a constant.

There are many algorithms for implementing ray casting method for the purpose of visualization of volume algorithms. Volume data is represented in the form of voxels. Voxel is the unit of the volume data. Each volume data is devided into number of voxels. In visualization pipeline voxels are shaded. The pipeline outputs the intensity C(X) for each voxel.

As shown in Fig. 6 every voxel in the volume data has two associated values-

C (X) shade calculated using local gradient α(X) opacity derived from CT (Computed Tomography). Now the next step is to combine the value of C(X) and α(X) and to produce two dimensional projections using these values. These values are combined to give the final intensity for each pixel. The transparency formula for every pixel along a ray can be calculated as-

$C_{out}=C_i(1-\alpha(X))+C(X)\alpha(X) \qquad (2)$

Where:

C_{out} is the colour for voxel X along a Ray.

C_{in} incoming intensity for voxel.

Interpolation between voxels will be done to find out the values which lie along a ray.

The intensity for a group of voxels intercepted by a ray is given as

$$C(R) = \sum_{k=0}^{k} \{ C(R,k)\alpha(R,k) \prod_{i=k+1}^{K}(1-\alpha(R,i))\} \qquad (3)$$

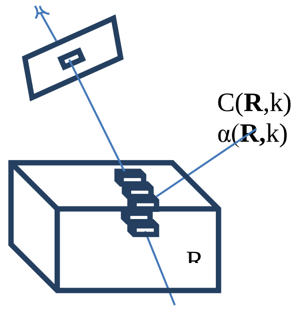

Fig. 6. Ray Casting through Voxels [4]

Where:

(R, k) is kth voxel

The illumination (I(t)), shading of local gradient (C_{out}), and calculating ray of local gradient (C(R)) are shown in equation (1), equation (2) and equation (3) respectively.

4. Result and Analysis

To analyze the performance of the proposed algorithm five different volumetric data sets have been used. The sizes of the data sets are 256*256*512 [19], 128*128*129 [8], 512*512*360 [7], 128*128*204 [8] and 128*128*64 [20].

In Fig. 7 given below, whole body of female has been shown. Three parts of female data those are head, abdomen and leg have been visualized at different clients. The visualization of human feet has been done in Fig.8.

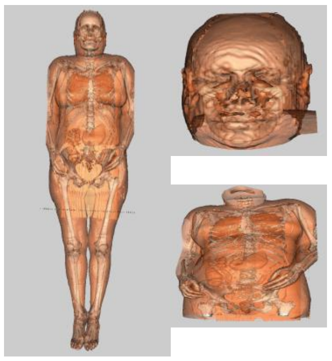

Fig. 7. Female Body and Partitioning of the Female Body into Three Parts

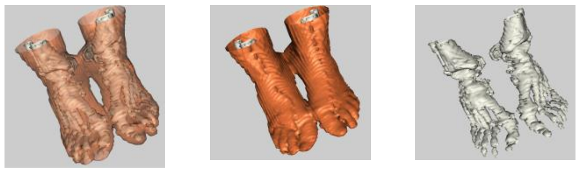

Fig. 8. Snapshot of Leg Data at Different Skin Levels

There are three images showing the human leg. In Fig. 8(a) skin transparency is fifty percent that is why both bone and skin are visible. In Fig. 8(b) skin transparency has been decreased to zero percent so bones are not visible. In Fig. 8(c) skin transparency has been increased to hundred percent that is why bones are clearly visible. So the skin level can be changed to visualize the internal part of the volume data.

Visualization of human head has been done. Fig. 9 (a), (b) and (c) shows the head data at 50, 0 and 100 percent skin transparency respectively.

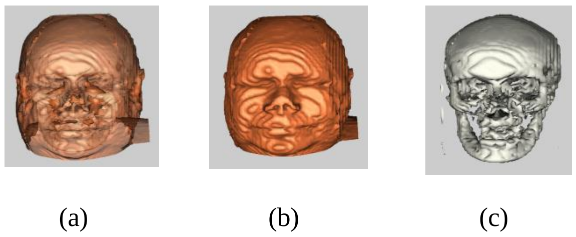

Fig. 9. Snapshot of Head Data at Different Skin Levels

Slicing is done on human head using a slicing plane. It enables user to see the internal structure of the head. For slicing purpose marching cube [4] method has been applied. Fig. 10(a) shows the head data and Fig. 10(b) shows slicing done using slicing plane.

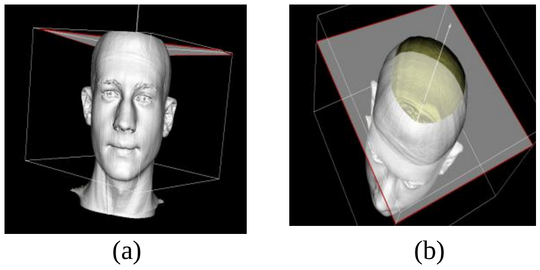

Fig. 10. Snapshot Of Head Data And Its Slicing

.Ply (Polygon File Format) data has also been visualized as depicted in Fig. 11. It shows human head visualized from two different viewpoints.

In our approach instead of sending of volumetric data, the rendered image is transferred to the client and only those images for which there is a change in the previous image are transferred. Subtraction of consecutive images is done to find out the location of the pixels for which the intensity has been changed. After finding out the location of the pixels for which there is a change in the intensity value, the location and corresponding intensity values are transferred to the client. Intensity values are updated at client side. As only those pixel locations and intensities are transferred for which there a change in intensity level, so it will reduce communication cost thus saving network bandwidth. So a thin client having less capability in terms of hardware and software can also work with volume data using the proposed algorithm The overall effectiveness of the system will depend upon the server processing power.

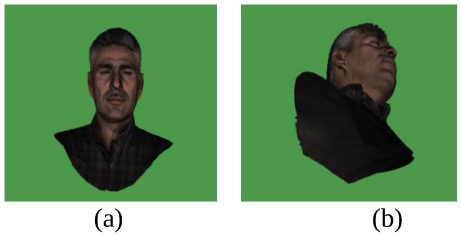

Fig. 11. Snapshots of .Ply Head data

Comparison between rendering time on standalone system and on distributed system using our approach is shown in table 1.

Table. 1. Input Output Performance of system with two different configurations (Standalone System /Distributed System)

Data Name	**Data Format**	**Data Size (MB)**	**RenderingTime (Seconds)**

			SS	DS
Female.raw [19]	.RAW	256*256*512	0.1	0.2
Body.stl [8]	.STL	128*128*129	0.5	0.65
Male.ply [7]	.PLY	512*512*360	2	2.2
Head.stl [8]	.STL	128*128*204	0.7	1
Mummy.vtk [20]	.VTK	128*128*64	0.2	.33

Here SS stands for Standalone System while DS stands for Distributed System. The Graph corresponding to above given table is shown in Fig. 12. In the graph X axis shows the sizes of the data sets in form of grids and Y axis shows the rendering time of different datasets in seconds on standalone system and on distributed system.
From the graph it is evident that if the capability of server is increased the capability of the clients also increases. So instead of having many thick clients our approach requires one thick server supporting many thin clients.

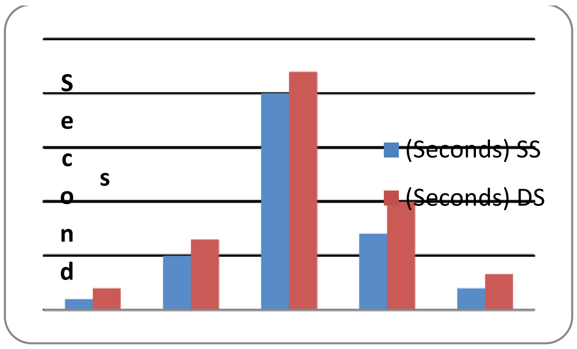

Fig. 12. Chart showing the rendering time Difference between standalone system and distributed system

5. Conclusion and future work

The developed system provides the facility to the user to visualize and perform slicing operation on 3D volume data stored at server side. The clients can visualize these volume datasets without having specialized software for visualization purpose. At a time four users can interact with the server. User need to have only java installed on its machine and address of the server. So a user who does not have much computation and storage capacity can also work on volumetric medical data. In future if intermediate result will be stored at client side then whenever client wants to visualize the results it will fetch the intermediate results from its database. Communication cost of retrieving same image will get reduced. Thus the efficiency of the system will increase.

References

[1] Visualization image: www.openterran.org last accessed on 17/03/2013.
[2] LHDL: http://www.livinghuman.org last accessed on 18/04/2013.
[3] Frey, S., and Ertl, T. Load Balancing Utilizing Data Redundancy in Distributed Volume Rendering. In *11th Eurographics Conference on Parallel Graphics and Visualization*(2011), 51-60.
[4] Alan, Watt., and Mark, Watt. Advanced Animation and Rendering Techniques, Addison-Wesley(1992).
[5] V. Sanchez, R. Abugharbieh, and P. Nasiopoulos, "3-D Scalable Medical Image Compression With Optimized Volume of Interest Coding", IEEE Transactions on Medical Imaging, vol.29 no.10, pp. 1808-1820, 2010.
[6] Hui Wei, Enjie Liu, and Gordon Clapworthy, "Interactive 3D Rendering to Assist the Processing of Distributed Medical Data", ACM IITM'10, Uttar Pradesh, India, pp. 28–30, 2010.
[7] Ply volumetric dataset download: http://www.cyberware.com/products/scanners/pxSamples.html, last accessed on 10/06/2013.
[8] Stl volumetric dataset download: http://www.eng.nus.edu.sg/LCEL/RP/u21/wwwroot/stl_library.htm#Download**,** last accessed on 20/06/2013.
[9] JavaView: http://www.javaview.de last accessed on 20/04/2013.
[10] K. Polthier, S. Khadem, E. Preuss, and U. Reitebuch, "Publication of Interactive Visualizations with JavaView", Springer Verlag Int'l Conf. on Multimedia Tools for Communicating Mathematics, pp. 1-24, 2002.
[11] K. Engel, P. Hastreiter,B. Tomandl, K. Eberhardt, and T. Ertl, "Combining local and remote visualization techniques for interactive volume rendering in medical applications", Proceedings on IEEE Int'l Conf. on Visualization, USA, pp. 449-452, 2000.
[12] M. Borkin, K. Gajos, A. Peters, D. Mitsouras, S. Melchionna, F. Rybicki, C. Feldman, and H. Pfister, "Evaluation of Artery Visualizations for Heart Disease Diagnosis", IEEE Transactions on Visualization and Computer Graphics, vol.17, no.12, pp. 2479-2488, 2011.
[13] VTK Introduction and download page: http://www.vtk.org/ last accessed on 04/05/2013.
[14] Sandor Markon, Hideaki. Miyake, Satoshi. Maekawa, Ahmet. Onat, and Zhao. Zhibo, "Improved interaction methods for medical

visualization with floating images", Proceeding of ICME International Conference on Complex Medical Engineering (CME), China, pp. 462-466, 2013.

[15] Liu Danzhou, K.A. Hua, and K. Sugaya, "A Generic Framework for Internet-Based Interactive Applications of High-Resolution 3-D Medical Image Data", IEEE Transactions on Information Technology in Biomedicine, vol. 12, no. 5, pp. 618-626, 2008.

[16] S. Ubik, Z. Trávníček, P. Žejdl, and J. Halák, "Remote Access to 3D Models for Research, Engineering, and Art", IEEE Transactions on MultiMedia, vol. 19, no. 4, pp. 12-19, 2012.

[17] L. Constantinescu, Kim. Jinman, and D.D. Feng, "SparkMed: A Framework for Dynamic Integration of Multimedia Medical Data Into Distributed m-Health Systems", IEEE Transactions on Information Technology in Biomedicine, vol.16, no.1, pp. 40-52, 2012.

[18] R. Vasudevan, G. Kurillo, E. Lobaton, T. Bernardin, O. Kreylos, R. Bajcsy, and K. Nahrstedt, "High-Quality Visualization for Geographically Distributed 3-D Teleimmersive Applications", IEEE Transactions on Multimedia, vol. 13, no. 3, pp. 573-584, 2011.

[19] Female.raw volumetric dataset download : http: www9.informatik.uni-erlangen.de/External/vollib, last accessed on 25/06/2013.

[20] Mummy.vtk volumetric dataset download: http://www.cs.utah.edu/~ramanuja/sci_vis/prj4/REPORT.html last accessed on 28/06/2013.

[21] Verschuur J., "A Client/Server Framework for Interactive Remote 3D Visualization of Histological Data", Master thesis of Delft University of Technology, 2009.

Alok Pratap Singh, presently pursing his Master in Information Technology, specialization in Human-Computer Interaction from IIIT Allahabad, India. He received his B.Tech. degree in Computer Science and Engineering from NIEC, Lucknow, India in 2010. His major interest is in image Processing, Computer graphics, volume data visualization and distributed system.

Piyush Kumar, presently pursing his Ph.D. Degree and did his Master in Information Technology, specialization in Human-Computer Interaction from IIIT Allahabad, India. He received his B.Tech. degree in Computer Science and Engineering from KNIT, Sultanpur, India in 2009. His major interest is in Image Processing, Gesture Recognition, OCR, Data Glove, Virtual Reality and Augmented Reality and Large Volume Data Visualization.

Anupam Agrawal is presently working as Professor of Information Technology at Indian Institute of Information Technology Allahabad (IIIT-A). Before joining IIIT-A in the year 2000, he was working as scientist 'D' at DEAL, DRDO, Govt. of Indian, Dehradun. He received the M.Tech. degree in Computer Science and Engineering from the Indian Institute of Technology Madras, Chennai and the Ph.D. degree in Information Technology from the Indian Institute of Information Technology, Allahabad (in association with Indian Institute of Technology, Roorkee). He was a postdoctoral researcher at the Department of Computer Science & Technology, University of Bedfordshire (UK) during which he had contributed significantly in two major European projects. His research interests include Computer Vision, Image Processing, Visual Computing, Soft Computing and Human Computer Interaction. He has more than 75 publications related to these areas in various international journals and conference proceedings, and has co-authored on two books. He is on the review board for various international journals including IEEE, Springer, MDPI, Taylor & Francis and Elsevier. He is currently serving as Principal Investigator of an international (Indo-UK) Project. He is a member of ACM (USA), senior member of IEEE (USA) and a fellow of IETE (India). He is also serving as Chairman of the ACM Chapter at IIIT-A