ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

A High Performance of 3D Object Reconstruction in a Cloud Environment

Mohamed-K Hussein Mohamed-H Mousa
Faculty of Computers and Informatics, Suez Canal University, Egypt

ABSTRACT: Reconstructing a 3D digital object from a set of points sampling real world objects has gained large attention from the computer graphics researchers. However, the 3D surface reconstruction of large model requires the processing of large-scale dataset which is indeed a very computationally expensive task. Therefore, a parallel algorithm must be adopted to improve the performance efficiency of the 3D surface reconstruction process. Even the parallel algorithms for the 3D surface reconstruction require high performance computing environment. The use of expensive high performance computing machines, such as supercomputers, is not practical for educational and research applications. Recently, Cloud Computing offers an outstanding potential of low cost high performance computing environment for such distributed large-scale applications. This paper presents a framework for high-performance execution of a parallel surface reconstruction of 3D objects on a private Cloud environment. First, a parallel algorithm for the 3D surface reconstruction is designed using the Master-Slave technique. The master computation node partitions the main dataset into a number subsets of data for the slaves to perform the parallel computations. The slave computation nodes simply performs the same computations on different datasets received from the master computation node. Finally, the slaves send the results back to the master node to finalize the computations. Experimental evaluation shows that the framework effectively enhances the time required to sketch the overall 3D reconstruction process.

Keywords: Graphics, Surface Reconstruction, High-performance, Cloud computing, Eucalyptus.

I. INTRODUCTION

Reconstructing 3D digital objects from 3D sample points of real world objects is used in the computer graphics research areas such as visualization, surface rendering, and shape similarities [1]. 3D surface reconstruction is used in a number of applications, including: computer animation, virtual reality, digital heritage collections and medical applications [2]. The surface reconstruction approaches can be divided into three categories [3]:

- Computational geometry category,
- Surface fitting category,
- Iso-surface extraction category.

In the first and the second categories, the surface is reconstructed explicitly using a polygonal mesh (first category) or a spline surface (second category). In the third category, the required surface is defined as the 0-level of an implicit function $f: \mathbb{R}^3 \rightarrow \mathbb{R}$. The main advantage of this category is that it enables the local reconstruction of a set of points to be performed in a linear time $O(n)$ where n is the cardinality of the local set of points [4]. In addition, the iso-surface extraction makes it possible to control the level of details of the resulting surface.

In general, the point based surfaces consist of surfaces represented by discrete point sets which are either directly obtained by 3D acquisition devices or converted from other surface representations. This kind of surfaces is well suited for multiresolution storage and transmission of complex objects. Unfortunately, efficient reconstruction of point-based surfaces requires reconstructing the surface approximating large-scale set of points. Therefore, this process requires a huge computational cost. Therefore, a parallel strategy is required to improve the processing time.

In this paper, a parallel 3D surface reconstruction algorithm is proposed. The proposed algorithm works in the implicit iso-surface extraction category. In fact, the proposed algorithm partitions the input sample points into a set of local patches that preserves the local neighbourhood properties of the surface. For each local patch a quadratic form is computed. These local quadratic patches are considered as the local implicit functions defining the surface. Next, these

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

local implicit functions are blended together to form the global implicit function. Finally, we get the required surface by extracting the 0-level iso-surface of the global implicit function. The step of computing the local quadratics, or the local implicit functions, has less inter-node communication. Therefore, it is more convenient for parallel computing. The parallelism is employed using the Master-Slave parallel model [5]. The master node partitions the set of points' dataset into a number of subsets where the slaves perform the computations on the data subsets in parallel.

The Cloud has become a promising high performance computing for scientific applications, such as scientific workflows, weather modelling and predictions [6, 7], as well as industrial applications, such as financial and medical applications [8, 9]. The Cloud computing offers cheap on-demand large-scale data centers and computing resources, such as CPUs and memory [10-12]. The Cloud has gained popularity by the ultimate advancement of the virtualization technology which allows the creation of multiple virtual machine (VM) instances on a single physical server [13].

In this paper, a Cloud-based architecture is used to significantly improve the performance efficiency of the proposed 3D parallel reconstruction algorithm. The architecture is successfully used for providing high performance execution of distributed scientific application, Coupled Model application [7], in a private Cloud environment. Originally, the framework is developed to achieve performance control of distributed scientific applications [14] in a dynamic Grid environment. In this paper, the Cloud-based architecture is adopted to provide high performance through supporting the Master-Slave model of the distributed application and through balancing the workloads on the slaves.

The presentation of the paper is organized as follows: Section II presents a detailed discussion of the related work on surface reconstruction, parallel algorithms for 3D surface reconstruction, and the used high performance execution environment for surface reconstruction in the previous literature. Section III presents the proposed parallel 3D surface reconstruction. Section IV describes the Cloud-based architecture for executing the parallel 3D surface reconstruction. Section V shows the experimental results and evaluation. Finally, the paper is concluded in Section VI.

II. RELATED WORK

In recent years, the surface reconstruction problem has gained much of interest due to its motivation in a large area of research in computer graphics [15]. As we mentioned above, the surface reconstruction techniques can be classified into three categories: (1) Computational geometry, (2) surface fitting and (3) iso-surface extraction.

Concerning the computational geometry category, the surface connectivity are established by the aid of the Delaunay triangulation or the Voronoi cells constructed over the input sample points [16, 17]. The advantage of the computational geometry techniques is that the time complexity is of order to the complexity of the input points. However, due to the use of the Delaunay triangulation, these techniques become inapplicable when the number of the input points becomes too large. In addition, these techniques become inaccurate when the density of the input samples is not sufficiently distributed over the surface.

In the surface fitting approaches, the surface is considered as a deformation of a base or a regular shape. The base shape can be considered as a growing balloon [18] or a set of point connected by springs [19]. Similar to the computational geometry approaches, the complexity of these techniques is of order to the complexity of the input points. However, these techniques have a restriction that the surface topology must be the same with the base shape.

The iso-surface extraction approach is based on defining an implicit function over the bounding box enclosed the surface and then define the required surface as the 0-level of the implicit function [3, 20, 21]. The advantage of the iso-surface extraction approach is that the time complexity is of order the required iso-surface resolution regardless of the complexity of the input points. In addition, the use of the implicit function during the reconstruction does not introduce any restriction about the topology of the input object. Moreover, the resulting surface is guaranteed to be a water-tight surface, i.e., a well-defined information about the object interior and exterior.

Due to the large computational cost of the 3D surface reconstruction algorithms, a number of research has been conducted to parallelize the surface reconstruction and to execute in a distributed environment. For example, Yau et al. [2] proposed a GPU-based surface reconstruction technique that parallelize the traditional region growing algorithm. Their algorithm starts by marking a few subset of points as independent seeds. Each seed is propagated along the surface

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Step 1: The master starts by receiving the dataset as a set of points.
Step 2: Apply noise removal algorithm on the dataset using point set technique.
Step 3: Partition the dataset in *n* subsets using kd-tree algorithm.
Step 4: Distribute the *n* data subsets to the *m* slaves
Step 5: For Each slave:
Receive the datasets, generates a quadratic form to fit the set of points for each subset.
Send the results back to the master.
Step 6: The master receives and merges the results.

Figure 1. The proposed 3D parallel surface reconstruction algorithm.

until the surface is covered by all the seeds. However, this algorithm does not provide a global connectivity of the input points. In addition, a post-processing step is required to remove the overlapping patches and filling holes. In a similar manner, Feng et al. [1] proposed a parallel algorithm based on generating contour lines of the sample points in parallel using the Master-Slave model. Further, a framework is used to run the parallel contour reconstruction algorithm on a Grid environment using Condor-G. However, this approach has a limitation that the input surface should be convertible to a layered dataset. This limitation restricts the domain of application to a limited number of domains, i.e., CT scans or MRI images. Our approach does not have this limitation and it can be applied to any set of points without any preformatting of the initial form.

III. PARALLEL SURFACE RECONSTRUCTION

The 3D surface reconstruction is a computationally expensive task. Therefore, in this section, a parallel 3D surface reconstruction based on the Master-Slave technique [5] will be proposed. In the Master-Slave parallel model, the master process divide the dataset into smaller set of datasets, and distributes the datasets onto the slaves. Each slave performs the required computation on the assigned dataset in parallel with other slaves, and sends the result back to the master process. Hence, the master process combines all the results, presents the result. Figure 1 shows the general step of the proposed parallel 3D surface reconstruction in the master-slave framework.

The proposed algorithm starts by a set of oriented points as an input to the master, Figure 2(a). The input points are processed to remove artificial artifacts like noises, low sampling over the surface, etc. In this step, we reduce the noise that exists in the point set and resample the point using the point set techniques [22, 23]. The input point are then partitioned using kd-tree to create local surface patches, Figure 2(b). The number of local patches depends on the shape of the model and the splitting condition as we will describe later in the following subsection. For each local patch, a quadratic form is generated to fit the set of points inside this patch, Figure 2(c). This step is parallelized over a set of slaves, the generation of the quadratic forms is computed over a set of slaves in order to perform the computations in parallel. The master controls the load balance of the computations between the slaves. Finally, the generated quadratic form are returned back to the master computer. These quadratic forms is blended together for final mesh extraction by the aid of the implicit surface framework, Figure 2(d). The following subsections will demonstrate the partitioning of the dataset, the parallel computation of each slave, and the combination of the results by the master.

A. Dataset partitioning (Spatial partition)

In this subsection, we show how to partition the set of points into a set of local surface patches using the kd-tree [20]. In fact, the spatial partitioning of geometric surfaces, which contain sharp geometric features, into a set of small and flat patches are well suited for the application of quadratic fitting. The partitioning process starts by initializing a bounding box surrounding all the surface, as shown in Figure 3. Therefore, we apply the splitting condition proposed by [24]. Each box of the kd-tree is divided into two sub-boxes if the contained points are not satisfying the following condition

$$\forall p \in B_k \rightarrow \overrightarrow{n_p} \cdot \overrightarrow{n_k} > \delta, \quad \delta \in [0 \quad 1]$$

where $\overrightarrow{n_p}$ is the direction of the normal to the surface at p and $\overrightarrow{n_k}$ is the average normal direction in B_k. The previous splitting condition ensures that there is no surface folding inside the box b_k.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

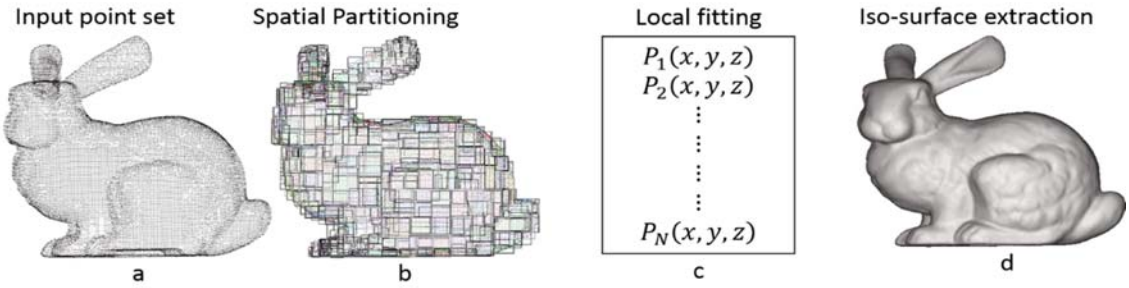

Figure 2. An example summarizing the general step of the proposed method. (a) The input model, 35k points. (b) The spatial partitioning of the set of points, 2335 partitions. (c) The set of the quadratic forms corresponding to the partitioning.

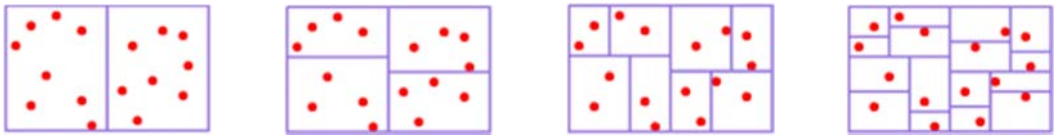

Figure 3. The general steps of the spatial partitioning using the kd-tree.

The size of the local surface patch $S_k$which is contained inside b_k depends on the value of the threshold δ. Small values of δ yield local patches with normal vectors closer to the average normal vector inside each patch. This yields small size local patches. Experimentally, we have found that choosing $\delta = 0.1$ yields a reasonable partitioning result. To reduce the calculation time, we restrict the number of local points inside each local patch, i.e. if the number of points inside the local patch is greater than a certain threshold, we repartition this patch. This additional condition is very helpful in the quadratic patch generation step. In fact, the generation of each local patch, as we will see later, is reduced into solving the eigenvalue problem of a system of linear equations. The size of this system of linear equations is equal to the number of points in each local patch. Therefore, minimizing the number of points in each local patch reducing the complexity of resulting system. Figure 4 shows an example of controlling the number of points in each local patch.

B. The parallel slaves computations (Local surface fitting)

This subsection shows how to fit a set of points into a quadratic surface patch [25]. This set of points is assumed to be clean, i.e. it has been manipulated such that the noisy points are removed. In addition, we assume that the considered set of points is sufficiently sampled over the surface. Therefore the resulting quadratic patch fitting the local points reflects the local behaviour of the actual surface.

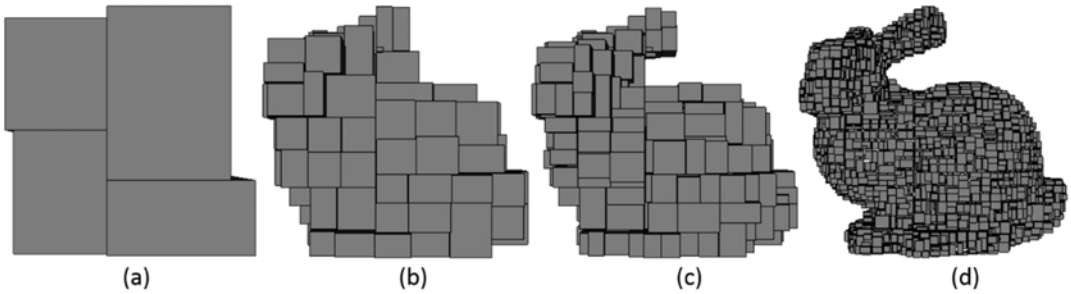

Figure 4. An example of partitioning the Stanford Bunny by restricting the number of points in each local patches. (a) 20k points ⇒ 4 partitions. (b) 10k points ⇒115 partitions (c) 1k points ⇒ 237 partitions. (d) 100 points ⇒ 2727 partitions.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

For each point $p = (p_x, p_y, p_z)$ in the local patch S_k, the objective here is to find a quadratic patch $f(p)$ such that:

$$f(p) = ap_x^2 + bp_y^2 + cp_z^2 + 2ep_xp_y + 2fp_yp_z + 2gp_xp_z + 2lp_x + 2mp_y + 2np_z + d = 0$$

Using the homogeneous coordinated, the previous equation can be represented as the following matrix form:

$$[p_x \quad p_y \quad p_z \quad 1]\begin{bmatrix} a & e & g & l \\ e & b & f & m \\ g & f & c & n \\ l & m & n & d \end{bmatrix}\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = 0$$

In general, the points $pf \in S_k$ do not satisfying exactly the fitting equation. So that, a minimization step is required to find the coefficients $a, b, \dots, d$ such that the following summation is minimum

$$\sum_{p \in S_k} f^2(p)$$

To prevent the minimization process from selecting the trivial solution $a = b = \cdots = d = 0$, we add the following constraint:

$$a^2 + b^2 + c^2 + 2e^2 + 2f^2 + 2g^2 + 2l^2 + 2m^2 + 2n^2 + d^2 = 1$$

Using the least square minimization of quadratic patches [25], the minimization process is reduced to 10×10 real symmetric eigenproblem of the form $A = D^T D$, where D is a $N \times 10$ matrix of the form:

$$\begin{bmatrix} p_{x_i}^2 & p_{y_i}^2 & p_{z_i}^2 & p_{x_i}p_{y_i} & p_{y_i}p_{z_i} & p_{x_i}p_{z_i} & p_{x_i} & p_{y_i} & p_{z_i} & d_i \end{bmatrix}$$

Where $N = |S_i|$ and $i \in [1 \quad N]$. The matrix A has 10 real eigenvalues and the corresponding eigenvectors. The eigenvector corresponding to the smallest eigenvalue represents the coefficients $a, b, \dots, d$ that minimize the quadratic error of the fitting process.

C. *Merging the results by the master (Implicit surface generation and extraction)*

The generated local patches from the previous step represent closed quadratic patches. Moreover, the quadratic equations have uniform sign (positive or negative) for the values inside the quadratic patch and zero on the surface and the opposite sign outside the patch. Therefore, these quadratic patches are considered as implicit functions. Since the overall surface is the union of the local patches, the overall surface is the blending of these local implicit functions. Consider that the f_k is the quadratic function that represent the local surface S_k. The blending of the functions $f_k$are as follows:

$$f(p) = \sum_k \delta_k f_k(p)$$

where δ_k are defined as the following

$$\delta_k = \begin{cases} 1 & p \in B_k \\ 0 & otherwise \end{cases}$$

The value of $f(p) = 0$ for all the points p which lie on the surface S. We can extract a triangular mesh that sample the surface S by applying the marching cube technique [26] to the extract the iso-surface $f(p) = 0$.

The following section presents the Cloud-based architecture which is used to execute the proposed parallel 3D surface reconstruction in a Cloud Environment.

IV. CLOUD-BASED ARCHITECTURE

Figure 5 shows the overall architecture of the Cloud-based framework for executing the proposed parallel 3D surface reconstruction, presented in Section III, in a Cloud Environment. The architecture is successfully used for providing high performance execution of distributed scientific application, Coupled Model application [7]. Originally, the framework is

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

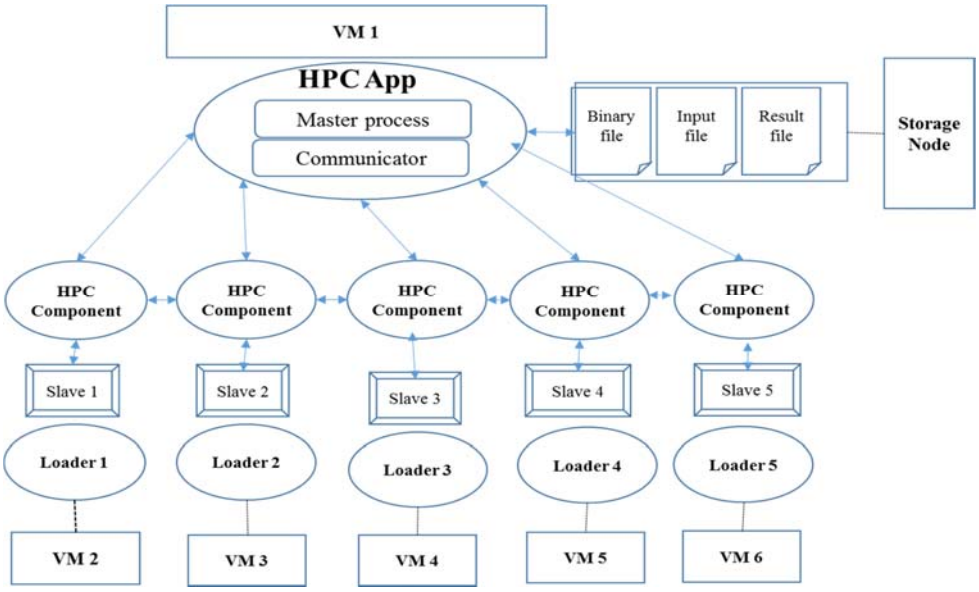

Figure 5. The proposed Cloud-based architecture.

developed to achieve performance control of distributed scientific applications [14] in a dynamic Grid environment. The Cloud-based framework consists of three main layers.

The first layer is the high-performance control of the application (HPC App) which consists of two main components, namely communicator and the master process. The master process component starts the main task of the 3D parallel surface reconstruction, as shown in the algorithm described Figure 1. The communicator interacts with the HPC Components to send and receive the data and results between the master process and the slaves. The second layer is the high-performance of each distributed components (HPC Component). HPC component is a software wrapper for each slave. It handles the communication between the slave and the master process component. The third layer is consists of the loaders. Once the framework is deployed on the Cloud, each loader starts execution on a virtual machine. The loader are responsible for starting the slaves for execution by running the proper binary file of the slave for the corresponding virtual machine configuration. Hence, each slave becomes waiting to receive the dataset from the master process.

V. EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION

This section evaluates the feasibility and the performance of the proposed 3D parallel surface reconstruction algorithm, described in Section III, on a private cloud environment using the Cloud-based architecture, described in Section IV.

A private Cloud is set Using Eucalyptus which is an open source software which implements Infrastructure as a service (IaaS) Cloud [27]. The IaaS infrastructure allows the end user to flexibly execute distributed scientific applications over the allocated VMs. Eucalyptus is used as the Cloud management layer because of its compatibility with commercial cloud products such as Amazon EC2 and S3 [27]. This compatibility enables to run a scientific application on a private cloud using Eucalyptus and a public cloud using Amazon without modification in execution framework or the distributed application [7]. The experiments are conducted on a private Cloud consists of four physical machines, each machine has i7 core Intel 2.2 GHz processor and 32GB memory. The virtualization layer is based on Xen hypervisor version 4.3. The VMs are deployed using Eucalyptus version 4. Each node runs Ubuntu version 12 operating system. OpenMPI version 1.5 is used with Open-MX version 1.4 as an optimised architecture for message passing MPI. 1 Gigabit Ethernet network fabric is used for networking. Open-MX is used to improve the commination performance of the MPI [28]. On Each

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

compute node, a virtual machine is deployed and assigned 4 CPUs and 16 GB memory. As shown in TABLE 1, on each VM HPC Component is deployed, and HPC APP is deployed on a VM.

TABLE 1.
THE DEPLOYMENT OF THE ADAPTIVE FRAMEWORK ON THE CLOUD RESOURCES.

Compute Node Number	deployment
1	HPC APP (Initiate Master) HPC Component(Initiate Slave)
2	HPC Component(Initiate Slave)
3	HPC Component(Initiate Slave)
4	HPC Component(Initiate Slave)

Figure 6 shows the reconstruction of the same object at a different resolutions, at different level of details. Through controlling the grid dimension of the iso-surface. It is clear that as the resolution is increased, a better 3D image will be reconstructed.

Figure 7 shows the reconstructed execution times of the image of the Stanford Bunny in different parallelism using the proposed parallel algorithm, described in Section III, where the dataset of the points contains exactly 263,256 data points. The master will partition the point into 240 patch, and each slave will be assigned the same number of patches to compute in parallel. It is clear that as we increase the degree of parallelism, the execution performance of the reconstruction process is improved compared to the sequential algorithm where the number of slaves is equal to zero. The execution time corresponding to the number of slaves is obtained as the average of 10 executions for each number of slaves. Figure 8 shows the calculated speed up corresponding to the number of slaves. It shows that the execution performance is almost double the execution performance the sequential algorithm.

(a) (b) (c)

Figure 6. Reconstruction of the bunny rabbit at different resolutions. (a) The resolution of the grid is 24^3partitions. (b) The resolution of the grid is 64^3. (c) The resolution of the grid is 256^3.

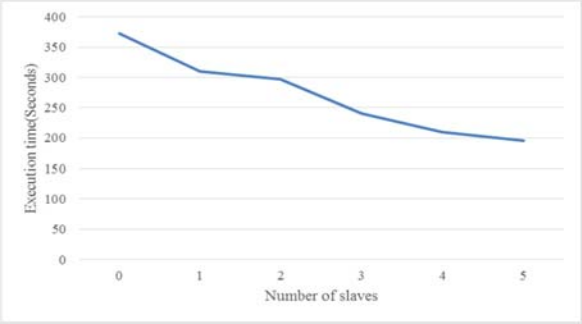

Figure 7. The execution times corresponding to the number of slaves.

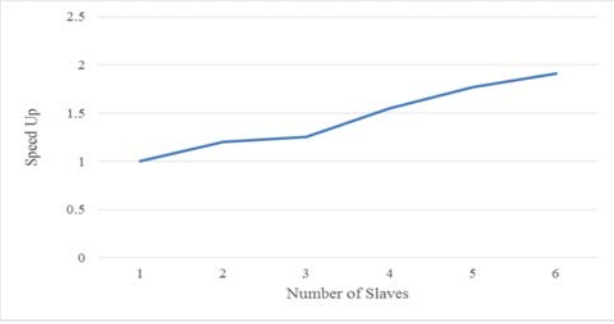

Figure 8. The speed up corresponding to the number of slaves.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

VI. CONCLUSION

This paper has proposed a novel parallel 3D surface reconstruction algorithm. The proposed algorithm partitions the input sample points into a set of local patches that preserves the local neighbourhood properties of the surface using the kd-tree algorithm. For each local patch a quadratic form is computed. These local quadratic patches are considered as the local implicit functions defining the surface. Finally, the local implicit functions are blended together to form the global implicit function. Hence, the required surface is computed by extracting the 0-level iso-surface of the global implicit function. The step of computing the local quadratics, or the local implicit functions, is parallelized using the Master-Slave parallel model. The master node partitions the set of points' dataset, using the kd-tree algorithm, into a number of subsets, local patches. Each slaves performs the computations of the quadratic forms on the assigned local patches in parallel.

Further, a Cloud-based architecture is used to significantly improve the performance efficiency of the proposed 3D parallel reconstruction algorithm. Experimental evaluations on a private Cloud managed by Eucalyptus have shown that the speed of the 3D surface reconstruction has improved significantly compared to the sequential algorithm. The future work will focus on providing adaptive performance management for the execution on a dynamic private Cloud where the computational nodes have different load conditions. Managing the adaptive performance for the Master-Slave model will be based on employing an intelligent model for assigning different workloads for the slaves according to the different load conditions on the computation nodes.

REFERENCES

[1] J. Feng, L. Kong, and X. Wang, "Surface Reconstruction Technology from Dense Scattered Points Based on Grid," in *High Performance Computing and Applications*. vol. 5938, W. Zhang, Z. Chen, C. Douglas, and W. Tong, Eds., ed: Springer Berlin Heidelberg, pp. 146-152, 2010.
[2] H.-T. Yau, T.-J. Yang, and H.-Z. Jian, "A region-growing algorithm using parallel computing for surface reconstruction from unorganized points," *Adv. Eng. Softw.*, vol. 59, pp. 29-37, 2013.
[3] M. Kazhdan, "Reconstruction of solid models from oriented point sets," presented at the Proceedings of the third Eurographics symposium on Geometry processing, Vienna, Austria, 2005.
[4] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel, "Multi-level partition of unity implicits," *ACM Trans. Graph.*, vol. 22, pp. 463-470, 2003.
[5] V. Kumar, A. Grama, A. Gupta, and G. Karypis, *Introduction to parallel computing: design and analysis of algorithms*: Benjamin-Cummings Publishing Co., Inc., 1994.
[6] M.-K. HUSSEIN and M.-H. MOUSA, "High-performance Execution of Scientific Multi-Physics Coupled Applications in a Private Cloud," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, pp. 11-16, 2014.
[7] M.-K. HUSSEIN, "Towards an Adaptive High-performance Execution of Scientific Applications in a Dynamic Cloud Environment," *International Journal of P2P Network Trends and Technology*, vol. 15, pp. 1-6, 2015.
[8] L. Griebel, H.-U. Prokosch, F. Köpcke, D. Toddenroth, J. Christoph, I. Leb, *et al.*, "A scoping review of cloud computing in healthcare," *BMC Medical Informatics and Decision Making*, vol. 15, pp. 1-16, 2015.
[9] M.-K. HUSSEIN, "Towards an Adaptive QoS of Cloud-based Web Services," *International Journal of Engineering and Innovative Technology (IJEIT)*, vol. 4, pp. 27-32, 2014.
[10] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Gener. Comput. Syst.*, vol. 25, pp. 599-616, 2009.
[11] J. Ekanayake and G. Fox, "High Performance Parallel Computing with Clouds and Cloud Technologies," in *Cloud Computing*. vol. 34, D. Avresky, M. Diaz, A. Bode, B. Ciciani, and E. Dekel, Eds., ed: Springer Berlin Heidelberg, pp. 20-38, 2010.
[12] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, pp. 50-58, 2010.
[13] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, *et al.*, "Xen and the art of virtualization," *SIGOPS Oper. Syst. Rev.*, vol. 37, pp. 164-177, 2003.
[14] M. Hussein, K. Mayes, M. Lujan, and J. Gurd, "Adaptive performance control for distributed scientific coupled models," presented at the Proceedings of the 21st annual international conference on Supercomputing, Seattle, Washington, 2007.
[15] S. Lim and H. Haron, "Surface reconstruction techniques: a review," *Artificial Intelligence Review*, vol. 42, pp. 59-78, 2014.
[16] N. Amenta, S. Choi, and R. K. Kolluri, "The power crust," presented at the Proceedings of the sixth ACM symposium on Solid modeling and applications, Ann Arbor, Michigan, USA, 2001.
[17] T. K. Dey and S. Goswami, "Tight cocone: a water-tight surface reconstructor," presented at the Proceedings of the eighth ACM symposium on Solid modeling and applications, Seattle, Washington, USA, 2003.
[18] Y. Chen and G. Medioni, "Description of complex objects from multiple range images using an inflating balloon model," *Computer Vision and Image Understanding*, vol. 61, pp. 325-334, 1995.
[19] D. Terzopoulos and M. Vasilescu, "Sampling and reconstruction with adaptive meshes," in *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pp. 70-75, 1991.
[20] M.-H. Mousa, R. Chaine, S. Akkouche, and E. Galin, "Toward an efficient triangle-based spherical harmonics representation of 3D objects," *Comput. Aided Geom. Des.*, vol. 25, pp. 561-575, 2008.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

[21] M. H. MOUSA and M. K. HUSSEIN, "Incremental Fourier transform of triangular closed 2-manifolds," *International Journal of the Physical Sciences*, vol. 7, pp. 5248-5255, 2012.
[22] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Point set surfaces," presented at the Proceedings of the conference on Visualization '01, San Diego, California, 2001.
[23] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and Rendering Point Set Surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, pp. 3-15, 2003.
[24] T. Boubekeur, P. Reuter, and C. Schlick, "Visualization of point-based surfaces with locally reconstructed subdivision surfaces," in *Shape Modeling and Applications, 2005 International Conference*, pp. 23-32, 2005.
[25] M. H. Mousa, "Matching 3D objects using principle curvatures descriptors," in *Communications, Computers and Signal Processing (PacRim), 2011 IEEE Pacific Rim Conference on*, pp. 447-452, 2011.
[26] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *SIGGRAPH Comput. Graph.*, vol. 21, pp. 163-169, 1987.
[27] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, *et al.*, "The Eucalyptus Open-Source Cloud-Computing System," presented at the Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009.
[28] B. Goglin, "High-performance message-passing over generic Ethernet hardware with Open-MX," *Parallel Comput.*, vol. 37, pp. 85-100, 2011.