# Integrating Grid Computing Technology into Multimedia Application

Rafid Al-Khannak[1], Abdulkareem A. Kadhim[2], Nada S. Ibrahim[3]

[1]*Computing Department, Buckingham New University, UK*

[2]*Networks Engineering Department, Al-Nahrain University, Iraq*

[3]*Information and Communications Engineering Department, Al-Nahrain University, Iraq*

***Abstract*— Systems now days are requiring huge database and massive power of computation. This cannot be gained using the available computation technology or computers. Execution time and large data processing are problems which are usually encountered in highly demanded applications such as multimedia processing. Grid computing technology offers a possible solution for computational intensive applications. Canny edge detection algorithm is known as the optimum algorithm for edge detection that performs edge detection in five different and computationally extensive stages, which consume large processing time. In this work, grid computing is used to perform Canny edge detection as an application of grid computing in multimedia processing. Univa grid engine is used the middleware for the proposed grid system. It is demonstrated here that the proposed grid based solution that integrates grid computing technology into Canny edge detection reduces the processing time while preserving the expected performance of Canny edge detection. The time reduction factor is about three times for the adopted grid system and may become better with careful selection of the communication networks technology.**

***Keywords*— Grid Computing, Canny Edge Detector, Univa grid engine.**

## I. INTRODUCTION

The existing individual computers cannot offer the computational power that is needed by large number of modern applications. Therefore, dealing with such challenges had raised the necessity of linking and utilizing number of distributed resources through the use of networks with high speed to build virtual supercomputers with powerful capability. Computational power could include data storage capacity, network performance and processor processing speed [1, 2]. Grid computing technology had been integrated in many applications that belong to various fields, such as power systems applications, biometrical applications, drug discovery, astronomy applications and many others [3,4].

Grid computing concept started as a project for linking supercomputers that are distributed geographically, but now it has developed far beyond of its essential intent [5]. The grid infrastructure can serve many applications such as data exploration, collaborative engineering and high-throughput computing, thus distributed computing was rising to a higher evolutionary level [6]. The goal was the creation of an illusion that is simple and yet robust self-managing computer which is virtual and out of a large heterogeneous systems collection that are connected and sharing dissimilar sets of resources [5,6]. The grid portal is a way in which end users can access the resources through. It is representing a method for accessing these grid resources in a regular manner. Grid portal is supporting the users with some facilities such as submission of jobs, management of these jobs, and monitoring them [7,8]. Grid computing systems cover two main types, which are the computational and the data grids. The computational grid is focusing on the optimization of the execution time of applications that are in need of a great computing processing cycle's number. Data grid is providing the solutions for large scale data management problems. There is also a third category which is represented by the service grid [9,10].

The integration of grid computing technology into multimedia applications is the focus of this project. Canny edge detector is a marketable tool for edge detection and has been applied on numerous applications in the image processing, computer vision, and multimedia area [11,12]. It is able for detecting the edges even in images that are contaminated by noise intensely. However, it is a time consuming algorithm [13]. Therefore, this work is focusing on using the grid computing technology to decrease the required processing time for Canny edge detector algorithm.

## II. UNIVA GRID MIDDLEWARE

Univa grid engine is a distributed resource management system which increasing the resources utilization, in addition to the productivity of end-user that increased through harnessing the available computing resources. Grid Engine software distributes workloads in an efficient manner among the pool of resource. This is done by choosing the most appropriate resources that are suitable for each work segment [14]. Univa grid engine is being used by research organizations and enterprises all over the world due to its leadership, global support and development of grid engine technology [14, 15]. All Univa grid engine available hosts can either be set up in a single cluster, or split up into multiple groups of hosts, where each group defines a cluster. These smaller sub-clusters are named cells. Each host can adopt one

or more roles, but each host should belong to only one cell [15].

In Univa Grid Engine there are three general categories of users, which are managers, (there is always one manager by default), which have comprehensive permission in Univa Grid Engine. The second category is Univa Grid Engine operators, who are permitted for modifying the state of particular objects, such as enabling or disabling a queue. The third category is containing all other users. These users have permission for only submitting jobs, modifying and deleting their own jobs, in addition, getting information related to the status of the cluster [16]. The operations that Univa grid engine do represented by: **Accepting jobs** which are usually being requests for granting resources most while Every job is contenting information about what this job should do, **Holding jobs** when the master host of the grid system keeps the jobs until the computational resources that are needed will be available, **Sending jobs** to the most suitable or chosen execution host once the computational resources are available and this particular execution host begins executing that job, **Managing running jobs** while the master host is responsible for managing the running jobs, so it collects reports from all execution hosts at a constant time interval, and finally **Logging the record of job execution when jobs are finished** since users are permitted to use the Accounting and Reporting Console (ARCo) for gathering instant data reports from the grid system and storing it for further analysis on database dedicated for storing reports, which could be an SQL database [17]

## III. Proposed Model for Integrating Grid Computing With Multimedia Application

In spite the Canny edge detection algorithm's optimality, it consumes a very long processing time as compared to other detection methods for image. The processing time becomes huge when the algorithm operates on video record that contains hundreds or even thousands of image frames. Thus there is a need to reduce such processing time by invoking grid computing in the algorithm implementation. In this work, Canny edge detector algorithm is used to detect edges on a video record, which was used here as input material to the system. The used video is an Ultra HD (4K) quality record. The application was created using $C^{++}$ programming language with its useful libraries. The latters are those supporting multimedia operations on $C^{++}$ as introduced by openCV library [18]. The work model is focusing on utilizing grid computing system, which consists of number of PCs that are connected through the Ethernet by Cisco Catalyst-2960 series switch. This system is built using Univa Grid Engine (UGE) as grid middleware. Fig.1 illustrates the system model.

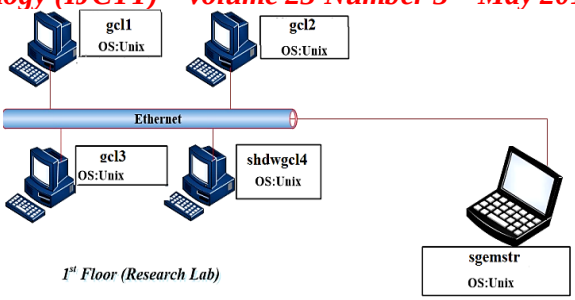


Fig. 1 Grid system design

The system design that had been chosen for the grid system is simple, since it includes five PCs, but it is used to state the grid concept and mechanism clearly. Table-1 shows the computational resources that are available in the system:

TABLE I
COMPUTATIONAL RESOURCES OF THE GRID SYSTEM

| Machine Name | Sgemstr | gcl1 | gcl2 | gcl3 | Shdwgcl4 |
|---|---|---|---|---|---|
| **Model** | Lenovo Thinkpad E530c | Dell OPTEPLE X 780 | Dell OPTEPLE X 780 | Dell OPTEPLE X 780 | Dell OPTEPLE X 780 |
| **OS** | Ubuntu Desktop 12.04 LTS | Ubuntu Desktop 12.04 LTS | Ubuntu Desktop 12.04 LTS | Ubuntu Desktop 12.04 LTS | Ubuntu Desktop 12.04 LTS |
| **Processor** | Intel® Core™ i5 2.06GHZ | Intel® Core™ 2Quad 2.66 GHZ | Intel® Core™ 2Quad 2.66 GHZ | Intel® Core™ 2Quad 2.66 GHZ | Intel® Core™ 2Quad 2.66 GHZ |
| **RAM** | 4GB | 4GB | 4GB | 4GB | 4GB |
| **HD** | 500GB | 250GB | 300GB | 300GB | 300GB |

The above table contains the hostnames that used for the machines. These names are the same ones that the grid system uses to identify the hosts (the machine that will act as grid master is Univa Grid Engine Master (sgemstr), whereas the remaining four machines are the Grid Clients and called here as gcl1, gcl2, gcl3 and shdwgcl4 (Shadow Grid Client). Sun Grid Engine Master is the grid controller and administrator, whereas the shadow grid client is the backup master client to compensate for master failure or problem.

## IV. Traditional Method Based Processing

The Canny edge detection algorithm application is implemented using single PC first. A short length video (2.45 min Ultra HD record with a rate of 25 frames per sec and MP4 format) has been used as an input for the Canny algorithm. The video original name was "spring 4k ultra hd.mp4" in Ref.19 and is called here as in.mp4 file. The

algorithm was running on a PC with Core™ 2Quad CPU Q9400, 4GB of RAM and 300GB HD. Fig.2 shows the application submission, while Fig.3 shows the output file that includes the last four frames and the processing time. The consumed time by one PC to complete the process is about 16 min., as shown in Fig.3, which is greater than the original time of the video record. A sample frame after edge detection and the corresponding frame of the original video are shown in Fig. 4.

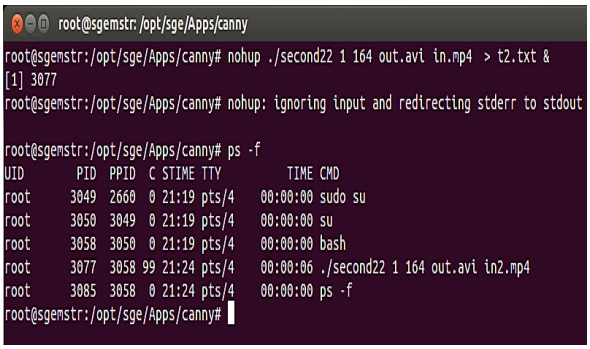


Fig. 2 Canny algorithm submission on single pc

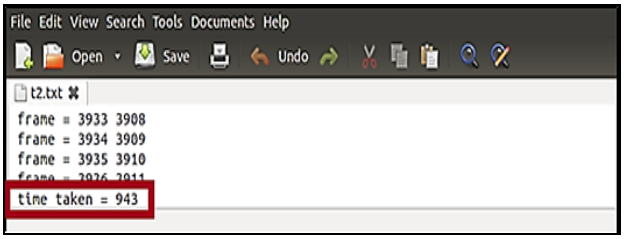


Fig. 3 Canny algorithm's consumed time for single pc

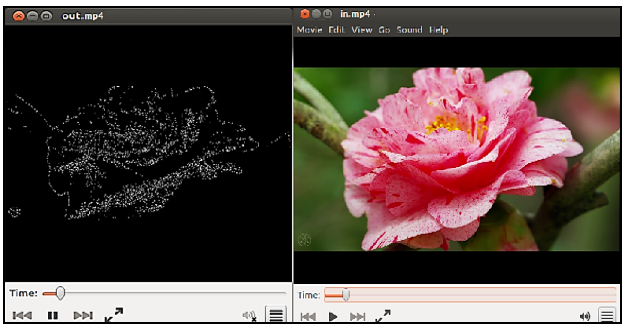


Fig. 4 Canny algorithm output and original frame for single pc

## V. GRID BASED PROCESSING

Canny algorithm is tested with the grid system using the previously mentioned input video which was located at the network file sharing (NFS) system and accessed by all grid hosts. The grid system produces two types of files which are out-files and error-files. The out-files include the last two frames along with the consumed time as shown in Fig.6 and 7.

The submission of the video record using Canny algorithm for the grid system is illustrated in Fig.5. As shown in Fig.5, the video record has been divided into 14 parts by a script that was written using shell programming language. These 14 parts were (in this case, each task to processes about 11 Sec.) distributed among the execution hosts by the grid scheduler as follows:

- Task 1,2,3 and 4 were running on the execution host named gcl3.
- Task 5,6,7 and 8 were running on the execution host named gcl1.
- Task 9,10,11 and 12 were running on the execution host named gcl2.
- Task 13 and 14 were running on the execution host named shdwgcl4.

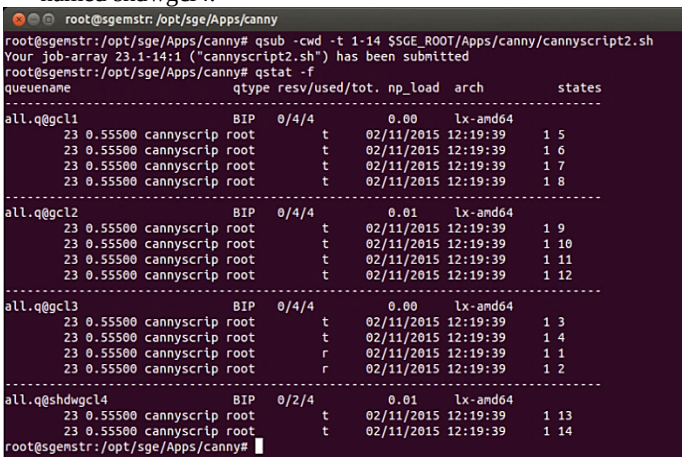


Fig. 5 Canny algorithm submission to grid system

The output files from the grid system that includes the last two frames, in addition to the processing time are shown by Figs.6 and 7.

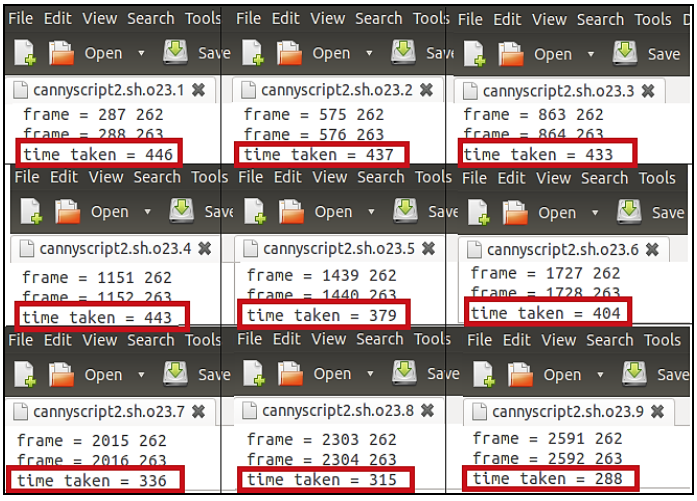


Fig. 6 Output files for case one using grid system (part-one).

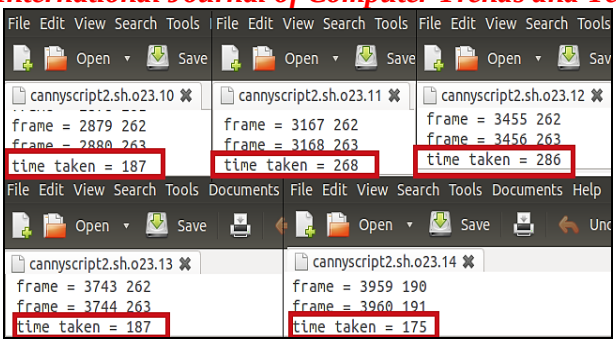


Fig. 7 Output files for case one using grid system (part-two).

The time consumed by the grid system to process the short video was about 5 min. Figs.8 and 9 show the output frame samples of edge detection output, these output samples are produced by task-1 and task-9, respectively.

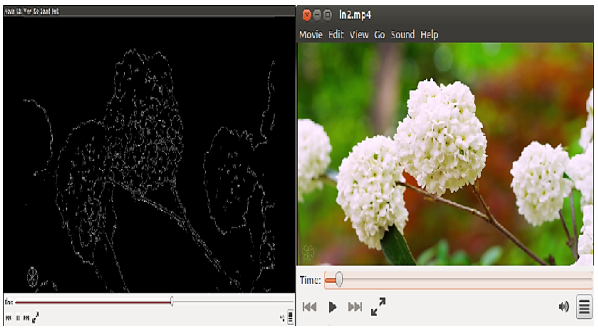


Fig. 8 Task-1 output frame sample of edge detection using grid system.

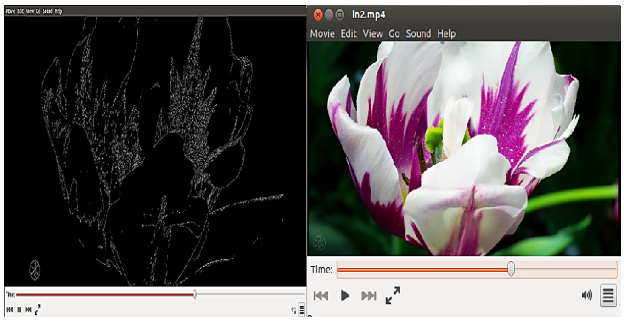


Fig. 9 Task-9 output frame sample of edge detection using grid system.

The network file system (NFS) includes the binaries that are necessary for the grid system to function, in addition, it containing all data that is needed to be accessed by grid hosts to complete a specific job. The network based accessing will be governed by the network file system read/write speed, rather than by the network's actual speed. This will affect the job processing. The reason here is that all job's operations will be performed within the NFS, (fig.10 shows the NFS read/write speed), and therefore, it will be limited to the NFS speed even if it is less than the general network's speed.

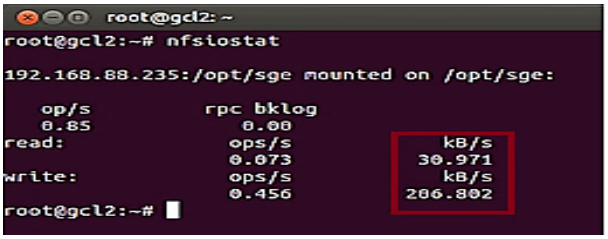


Fig. 10 NFS read/write speed

To examine the grid system actual performance without involving the network access effects, a hypothetical test is performed. All required data (the input video in this case) are stored locally on each grid host. Furthermore, the output will be on the execution hosts rather than in NFS system. This will minimize effects caused by the network on the system operation. The output files, the last two frames and the consumed time, are shown by Figs.11 and 12. According to these figures, the required time for Canny edge detection application to complete its process, using the grid system with local data availability, is about 2 min.

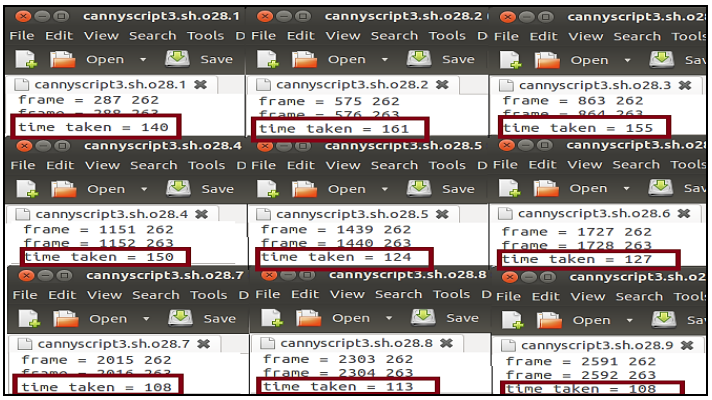


Fig. 11 Output files for the local data case using grid system (part-one).

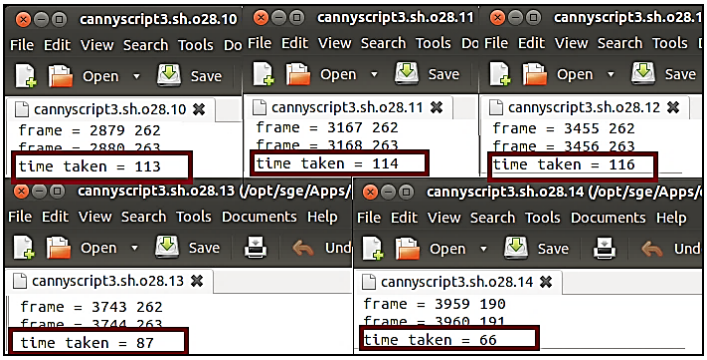


Fig. 12 Output files for the local data case using grid system (part-two).

## VI. Assessment of Results

Fig.13 shows the required processing time for Canny edge detection algorithm using both the traditional and grid based systems. It is clear that the grid based implementation gained useful time reduction as compared to traditional one. Further reduction in the processing time is also obtained when the effects of network are ignored and the data are made available at the hosts as expected.

According to chart (Fig.13), the grid based method was about three times faster than the traditional method using single PC for the case of shared data. In addition, it was seven times faster as compared to traditional method and when the required data are available locally. In this case the time consumed is less than the actual input video length making the grid approach an attractive one for the applications that needing fast real time processing. The only problem to be solved here is providing high speed network. It is worth to mention here that the PC used to get the traditional process time was having the ability to process about three frames per Sec., while in other hand, the PCs building the grid system (grid hosts are identical in hardware specifications) has ability to process about two frames per Sec each, which is considered to be low as compared to the CPU capability of the PC used for traditional method. This clarifies the idea that the grid system can improve the process time using PCs with relatively low hardware specifications.

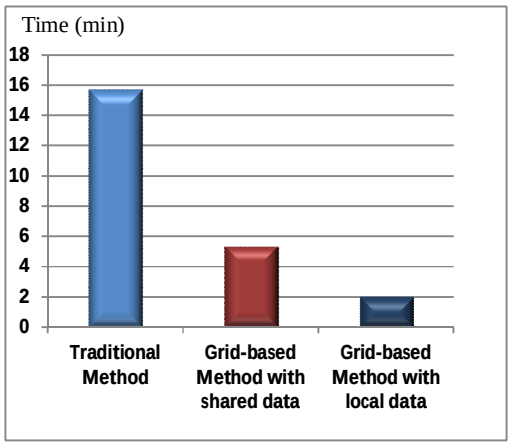


Fig. 13 Canny edge detection processing time comparison

## VII. Conclusions

Grid computing brings an innovative method to deal with applications that are highlighted with intensive processing operations. In this work, Canny edge detection based on grid system application is presented. The system used Univa grid middleware in a model involving 5 PCs. The application provided useful reduction in processing time as compared to traditional single PC based processing (about three times), yet having accurate performance. It is also found that the transmission network consumed considerable time while transferring data among the hosts. Thus more processing time saving can be achieved by careful selection of networking technology as well. The results show that the grid computing technology integration with multimedia application has a very promising performance that can be applied to similar computational extensive applications.

## References


[1] R. Al-Khannak and B. Bitzer, "*Grid Computing as an Innovative Solution for Power System's Reliability and Redundancy*", International Conference on Clean Electrical Power , Capri, June 2009.

[2] R. Al-Khannak and B. Bitzer, "*Grid Computing for Power and Automation Systems Implementations*", IEEE Universities Power Engineering Conference UPEC'06, Newcastle-upon-Tyne, U.K., Sept. 2006.

[3] R. Al-Khannak and L. Ye, "*Integrating Grid Computing Technology for Developing Power Systems Reliability and Efficiency*", 12th WSEAS International Conference on SYSTEMS, Heraklion, Greece, July 2008.

[4] R.Al-Khannak and B. Bitzer, "*Load Balancing for Distributed and Integrated Power Systems Using Grid Computing* ", IEEE International Conference on Clean Electrical Power, Capri , May 2007.

[5] M. Baker , R. Buyya and D. Laforenza , " Grids and Grid Technologies for Wide-Area Distributed Computing", Grid Computing and Distributed Systems Laboratory, Department of Computer Science and Software Engineering, University of Melbourne, Melbourne, Australia ,2002.

[6] L. Ferreira , V. Berstis , J. Armstrong ,M. Kendzierski , A. Neukoetter , M. Takagi , R. Bing-Wo , A. Amir , R. Murakawa , O. Hernandez , J. Magowan and N. Bieberstein, "Introduction to Grid Computing with Globus ", IBM Redbooks, 2nd Edition, IBM Corp., International Technical Support Organization, USA , Sept. 2003 .

[7] S. Pardeshi, C. Patil and S. Dhumale, "Grid Computing Architecture and Benefits", International Journal of Scientific and Research Publications, Volume-3, Issue 8, August 2013.

[8] P. Rani, "Middleware and Toolkits in Grid Computing", International Journal of Innovative Technology and Exploring Engineering (IJITEE) Volume-2, Issue-4, March 2013.

[9] C.H Lai and F. Magoules, "Fundamentals of Grid Computing Theory, Algorithms and Technologies", Chapman & Hall/CRC, Taylor and Francis Group LLC, London UK, 2010.

[10] L. Ferreira, F. Lucchese , T. Yasuda , C. Y. Lee , C. Alexandre , E. Minetto and A. Mungioli , " Grid Computing in Research and Education", IBM Redbook, 1st Edition , International Technical Support Organization , IBM Corp. , April 2005.

[11] X. He, J. Li, D. Wei, W. Jia, and Q. Wu, "*Canny Edge Detection on a Virtual Hexagonal Image Structure*", Pervasive Computing (JCPC), IEEE International Conference, Tamsui, Taipei, Dec. 2009.

[12] G.T. Shrivakshan, "*A Comparison of Various Edge Detection Techniques Used in Image Processing*", IJCSI International Journal of Computer Science Issues, Volume-9, Issue-5, Sept. 2012.

[13] C. Gentsos and N. Vassiliadis, "*Real-Time Canny Edge Detection Parallel Implementation for FPGAs*", Electronics, Circuits, and Systems (ICECS), IEEE International Conference, Athens, Dec. 2010.

[14] Grid Engine, "*Univa Products Grid Engine Software for Workload Scheduling and Management*", Retrieved Oct. 2014. [Online] Available http://www.univa.com/products/grid-engine

[15] Univa Engineering, "*Grid Engine Installation Guide*", Grid Engine Documentation, Univa Corporation, Version: 8.2.0, August 2014.

[16] Univa Engineering, "Grid Engine Users' Guide", Grid Engine Documentation, Univa Corporation, Version: 1.00, March 2014.

[17] U. Shankar, "Oracle Grid Engine User Guide", Oracle Company, Release 6.2 Update 7, August 2011.

[18] A. Huamán , B. Gábor , W. Kienzle , E. Christiansen, A. Pavlenko , B. Demiröz, M. Cosenza , V. Glumov , A. Myagkov, E. Feicho and A. Smorkalov, "*The OpenCV Tutorials*", Release 2.4.9.0, OpenCV.org, June 2014.

[19] SPRING 4K (ULTRA HD), mp4 video sample, downloaded on March 2014. [Online] Available: http://share2.earthlinktele.com/sharefiles.aspx?file=730831559