Internet of Things and Cloud Computing
2018; 6(2): 36-48
http://www.sciencepublishinggroup.com/j/iotcc
doi: 10.11648/j.iotcc.20180602.11
ISSN: 2376-7715 (Print); ISSN: 2376-7731 (Online)

Multilevel Performance Evaluation of Nvidia Grid VCA Using Iray and V-Ray Rendering Engines in 3DS Max Design

Kenneth Ritter III, Aaron Morgan, Charles Taylor, Terrence Chambers

College of Engineering, University of Louisiana at Lafayette, Lafayette, USA

Email address:
kar4499@louisiana.edu (K. Ritter III), aaronmorg@gmail.com (A. Morgan), cet1525@louisiana.edu (C. Taylor), tlc3715@louisiana.edu (T. Chambers)

To cite this article:
Kenneth Ritter III, Aaron Morgan, Charles Taylor, Terrence Chambers, Multilevel Performance Evaluation of Nvidia Grid VCA Using Iray and V-Ray Rendering Engines in 3DS Max Design. *Internet of Things and Cloud Computing*. Vol. 6, No. 2, 2018, pp. 36-48.
doi: 10.11648/j.iotcc.20180602.11

Received: March 22, 2018; **Accepted**: April 4, 2018; **Published**: May 5, 2018

Abstract: High performance computing is increasingly common in technological industries and there are many different solutions available on the market. Determining which computing solution is most cost-effective can be difficult. This study outlines the performance between a single-user, traditional high-performance workstation and a multi-user, virtualized workstation. Along with this direct performance comparison, the impacts of virtualization on rendering performance, GPUs, and the technological industry is evaluated in this study. Through the repeated rendering of two different Computer-Aided Design (CAD) models under varying test scenarios, a pool of data including render times and image quality is collected and analyzed. Two phenomena are observed and explained. One is a diminishing return in GPU power output that is observed after allocating four or more GPUs to a single rendering task. The second is a noticeable point of image-noise convergence during a render that could potentially be calculated and exploited to make rendering more time-efficient. These discoveries may impact the effectiveness of virtual GPU scalability and make time-consuming rendering more efficient for industry users. The NVIDIA GRID Visual Computing Appliance (VCA) is found to be cost effective for research laboratories that have several users with diverse needs.

Keywords: Rendering, Virtualization, Nvidia Grid Visual Computing Appliance, Graphics Processing Unit Computing, Iray, V-Ray

1. Introduction

Rendering an animation of a complex CAD model on a typical workstation can take extensive amounts of time and resources. In the past, a central processing unit (CPU) has handled rendering and the associated system tools. This was largely because graphics processing units (GPU) were not designed to perform as general-purpose computing platforms, but rather for specific low-complexity use cases. In recent years, though, with the efforts of NVIDIA and AMD, the GPU is evolving to work alongside, or even surpass, the CPU as a general computing platform. GPUs have increased in features and popularity, and their limitations are decreasing when compared to CPU engines. A rendering engine with two or three high-end GPUs can out number CPU cores by two orders of magnitude. This drastically increases rendering speed, which is the driving force for GPU integration into a rendering system.

Using NVIDIA GRID VCA remote GPU acceleration can greatly speed up the rendering process by simultaneously using eight GPUs without having a user-dedicated workstation. The NVIDIA GRID VCA is a network-attached appliance capable of hosting powerful virtual machines and streaming the display output to other devices on the network. This allows low-power devices, such as a laptop or tablet, to utilize the high-end hardware inside the VCA to complete resource-intensive processes far beyond the capabilities of the device's local hardware. The virtual workstations, shown in Figure 1, allow up to eight concurrent users but can be configured to allocate all available GPUs to a single user.

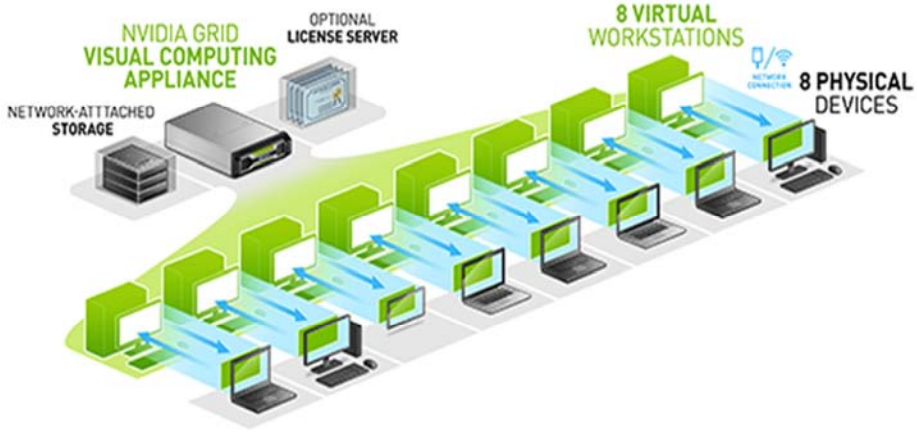


Figure 1. *Nvidia GRID VCA virtual workstations [1].*

Virtualized computing enables several types of computationally expensive work to easily be incorporated into the workflow without the need for specialized lab resources. With the Nvidia Grid VCA users can utilize high performance computing, such as computational fluid dynamics (CFD) simultaneously processing with 8 GPUs in the morning then have up to 8 virtual sessions using performance computing such as Computer Aided Design (CAD) model rendering that afternoon. This virtualized computing enables this type of work to easily be incorporated into the workflow without the need for specialized lab resources. This could be a space saving and cost-effective solution for certain applications. This paper identified the NVIDIA GRID VCA system as a potential solution for rendering animations of complex CAD models. An overview of the system is given along with a comparative performance analysis of using the VCA virtualized GPUs to high-end rendering workstations.

2. Literature Review

2.1. VCA Virtualization

The NVIDIA Kepler GRID K1 and K2 cards use architecture that allows hardware virtualization of the GPU, a technology termed vGPU. While hosting on-demand Virtual Machines (VMs) in a server environment is not a new idea, NVIDIA's development and implementation of the GRID vGPU technology is unique in its ability to allow multiple users to dynamically share available GPU resources. vGPU is, at its core, a hardware virtualization solution that removes the need for software-level adaptation to NVIDIA's shared GPU architecture. Additionally, NVIDIA has developed a memory management unit (MMU) responsible for allocating, mapping, and translating each VM's virtual address into a physical address, ensuring that each VM is given its own address space with no interference from other VMs [2]. This is an important feature that ensures scalability from a single GRID unit up to industry needs.

The hardware virtualization, in conjunction with the MMU, allows a system manager to dynamically allocate the available hardware among users based on their graphic demand. To increase energy efficiency schedulers can be enhanced so that GPU servers are put into low-power sleeping modes as long as their acceleration features are not required [3]. This is a large improvement in efficiency over previously used GPU Pass-through methods for remote graphics, where an entire GPU could be virtualized for a single user but not allocated dynamically. This type of GPU allocation causes a loss of effective resources if each user is not utilizing all of the resources allocated to their VM.

2.2. Cloud Computing

Traditionally, distributed and parallel computing technologies use off-line rendering farms to connect a cluster of computers together in a network. This would have a high capital cost, but with the ability to perform computationally expensive rendering jobs, alternate revenue streams can be created by finding companies looking to outsource. Cloud computing services such as Amazon EC2, offer dynamically scalable virtualized resources over the internet. These services are fundamentally changing the way IT services are invented, developed, deployed, scaled, updated, maintained and paid for [4–6]. Rendering-as-a-Service (RaaS) uses cloud computing technology to do rendering tasks in a pay-per-service model [7]. The animation files to be rendered are uploaded and sent to the RaaS service providers, and the rendering task is split into frames and rendered in the cloud resources using the virtual machines in their render farms [8].

GPU clusters are used where two or more computers with GPUs are networked together and share the computational load. This has significant advantages as the extremely parallel hardware architecture and high performance of floating point arithmetic and memory operations on GPU's, allow them to handle similar scientific and engineering workloads of high performance computing (HPC) clusters [9–12]. This leads to GPU incorporation as HPC accelerators that use the GPU and CPU combined to accelerate scientific, analytics, engineering, consumer, and enterprise applications. The total execution time of data-clustering saved by acceleration by GPU co-processing is significant even with

the costly data-transfer from GPU to CPU [13].

This NVIDIA GRID system could be used as a rendering farm on a 10 gigabit network, eliminating the need for complex rendering architectures. This allows GPU resources to be shared that eliminate performance-robbing CPU overhead and application issues with reliability and compatibility [2]. As shown in Figure 2, several types of devices can utilize virtual machines, and with optimal GPU sharing, each end user can effectively process the workload of their machines in a timely and effective manner.

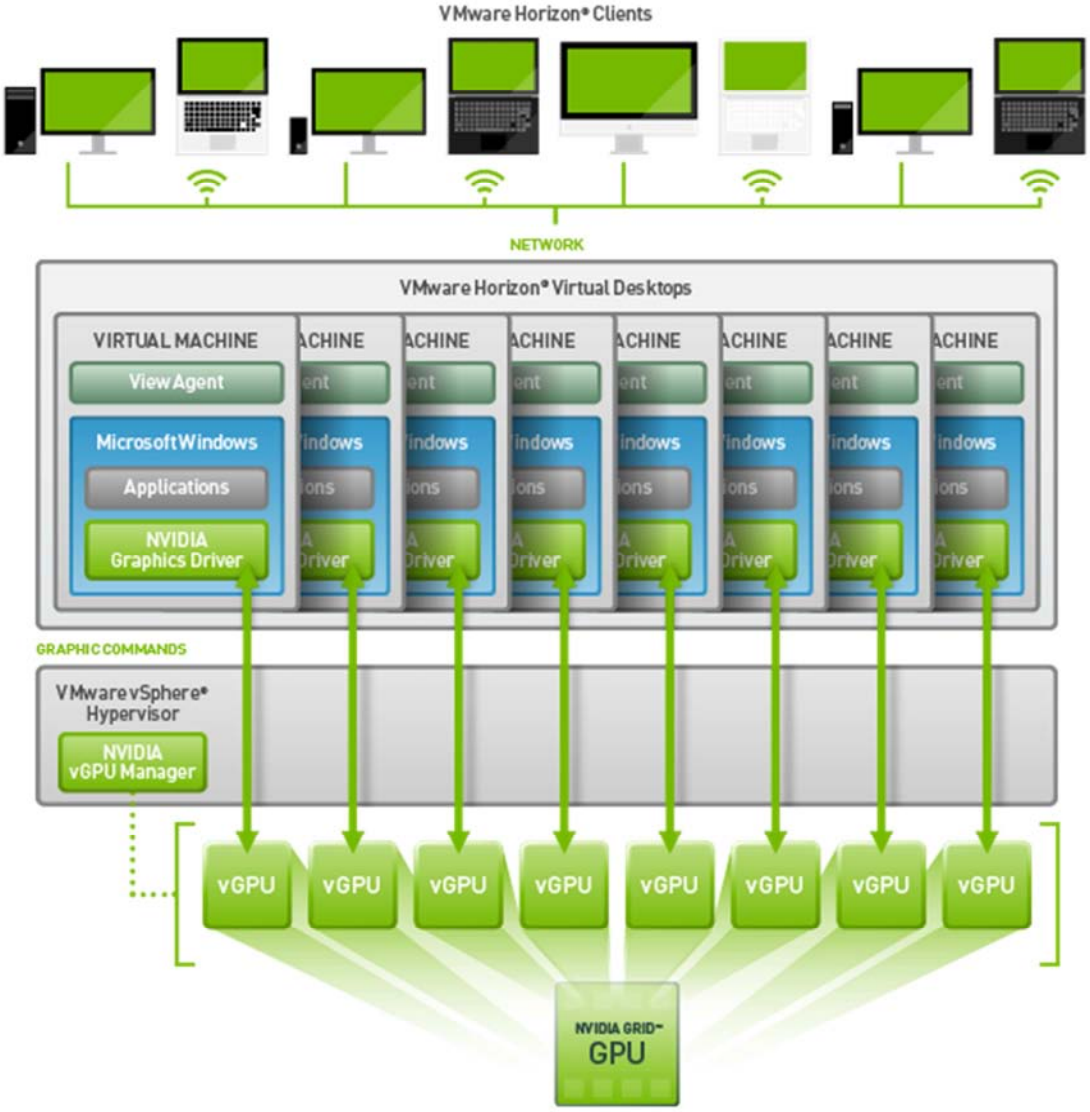


Figure 2. *All virtual machines share one physical GPU with GRID vGPU's hardware-based virtualization [2].*

This technology allows the virtual desktop screen to be pushed directly to the remoting protocol, giving access to all 3D-intensive applications to multiple clients as shown in Figure 2. Multiple users can now share a single GPU, giving rich graphics in virtualized environments and providing true PC performance and compatibility. When interacting with typical virtual machines, a patented low-latency remote display technology is utilized to reduce lag. This significantly improves the user experience when operating in the cloud [14].

Network-based computing is especially useful when using the GRID VCA for rendering due to the way the VCA operates. A user would first open the NVIDIA GRID client, which sends a request to the VCA for a virtual machine (VM). Each VM is allocated a set amount of CPU, GPU, and RAM that can be quickly changed through an administrative control panel, changing the maximum number of users based on total available resources. All VMs are managed, maintained, and reallocated through the use of a hypervisor, firmware specifically designed for this purpose. The VM display output is then delivered to the user via H.264 encoding, which is similar to streaming services such as Netflix and YouTube [14].

2.3. GPU Computing

The programmable parallel processors in GPUs exceed the computing power of multicore CPUs [15, 16]. GPUs are being utilized at an increasing rate in scientific computing applications and GPU rendering algorithms have emerged

[17, 18]. What GPU cores lack in versatility in handling operations such as loops and conditions, they make up for in core numbers. A GPU core can perform only 1 operation, 64-bit doubles or 32-64 bit pixels, but can have hundreds of cores per card. For algorithms where the processing of large blocks of visual data is done in parallel, the GPUs highly parallel structure makes it more effective than general-purpose CPUs. The GPU has the distinct advantage of simple linear hardware scaling. These advances in GPU technology have allowed the GPU to be promoted from a position of high-resource, low-complexity work dictated by the CPU to a platform capable of general purpose calculations and parallelizing calculation-intensive tasks.

The CPU and GPU have significantly different architectures as shown in Figure 3. The CPU is composed of just few powerful cores optimized for sequential serial processing with lots of cache memory, whereas the GPU has a massively parallel architecture consisting of hundreds of smaller less powerful cores that are more efficient cores designed for handling multiple tasks simultaneously.

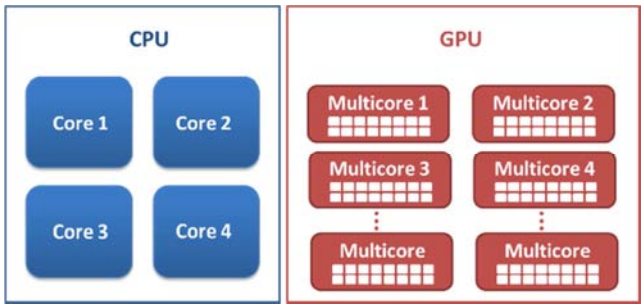


***Figure 3**. Architecture Comparison of a CPU and a GPU.*

This makes GPUs particularly good at parallel tasks giving them the ability to process thousands of threads, which can accelerate software by 100x over a CPU alone. Also, the GPU achieves this acceleration while giving off less heat and being more power- and cost-efficient than a CPU.

Some software developers such as Autodesk, creators of 3DS Max, have begun to see the potential applications of GPU computing in rendering. GPUs provide a responsive user workspace, interactive viewport, and GPU-accelerated frame rendering in applications such as Autodesk's 3DS Max Design. Rendering solution providers are also taking note of the strides made in GPU computing. This is shown by the development of GPU rendering solutions created to work alongside existing CPU rendering solutions such as NVIDIA's Iray and mentalray or CHAOSGROUP's V-Ray RT and V-Ray.

2.4. Iray

NVIDIA's GPU rendering solution, Iray, is an unbiased rendering engine designed to produce photorealistic renders using CUDA-enabled GPUs. The parallel computing architecture CUDA (Compute Unified Device Architecture) provides a C-like abstraction for executing on the GPU [19]. Iray uses global illumination to simulate the physical behavior of light, attempting to improve on previously developed rendering solutions. In order to more accurately model the interaction between light and materials, Iray uses deterministic global illumination in conjunction with bidirectional scattering and empirical distribution function shading frameworks. NVIDIA's proprietary global illumination algorithms allow Iray to render surfaces based on assigned material properties, freeing Iray from the need for complex renderer-specific shaders and settings. Iray was developed to work well in network-based computing environments and to scale well with increases in available hardware [20]. This is a key feature when dealing with cloud-based rendering using appliances like the VCA, which gives users access to powerful hardware exclusively through a local network.

2.5. V-Ray

V-Ray, created by Chaos Group, is a rendering plug-in that works with several software applications such as 3ds Max Design. Many advanced techniques used by V-Ray, such as path tracing and photon mapping, make it preferable to conventional renderers. V-Ray RT (Real-Time) is an interactive rendering engine that can utilize both CPU and GPU hardware acceleration to render models. The V-Ray RT engine can use GPUs exclusively using CUDA or OpenCL (Open Computing Language). OpenCL supports AMD processors where CUDA is exclusive to NVIDIA.

2.6. Image Quality Calculator

The amount of noise or lack thereof in an image is referred to as image quality when using the image quality calculator [21]. In order to calculate this value, the University of Manitoba's Physics and Astronomy Department created an Image Quality Calculator plugin to be used with ImageJ, a java-based image processing program. This program calculates the noise in an image by comparing each pixel with its four neighboring pixels, two horizontally adjacent and two vertically adjacent. The sum of the differences is used as the approximate noise found in that pixel set, and the average noise for the entire image is calculated. This value given is proportional to the amount of noise in an image. For this tool, a lower amount of noise indicates a higher quality image. It is important to note that there is no perfect score, as images containing different content will approach differing amounts of noise as their qualities improve. The Image Quality Calculator should only be used as a means of objectively comparing a set of images containing similar content.

2.7. Benchmarks

A benchmark is a standard, or set of standards, that may be used as a point of reference when quantifying performance or quality. In computer graphics, benchmarks are usually found as test packages containing scripts that run the system through a series of tasks designed to put stress on particular components or processes [22]. When comparing the

rendering performance of multiple workstations, strong figures of merit are render speed and amount of noise.

2.8. 3D Mark

3D Mark is a popular computer graphic benchmarking software that is used to measure, understand, and manage the performance capabilities of workstations and compare them with other computer hardware performance. 3D Mark is meant to measure raw, end-user gaming performance measured by framerate and graphic quality. This benchmarking software provides a normalized, consistent means of comparing workstations with differing hardware and software configurations.

The complete 3D Mark testing suite contains three smaller testing packages named Ice Storm, Cloud Gate, and Fire Strike. Each of these testing packages is designed to test specific facets of the graphics pipeline. Ice Storm and Cloud Gate are designed to test mobile devices and entry-level PCs, respectively. Fire Strike is designed to test high-end systems using two graphics tests, one physics test, and a combined GPU/CPU test [22]. A score is given from each test that is used for comparison that involves the graphics, physics, and combined total score of a system.

2.9. SPEC

Standard Performance Evaluation Corporation (SPEC) is a non-profit corporation formed to establish, maintain, and endorse a standardized set of relevant benchmarks that can be applied to the newest generation of high-performance computers [23]**.** SPEC has developed a performance evaluation benchmark specifically designed to measure the performance of a system running 3DS Max 2015 called SPECapc. This benchmark was created in collaboration with Autodesk, the creators of 3ds Max Design. The SPECapc benchmarking suite contains forty-eight tests involving modeling, interactive graphics, as well as general CPU and GPU performance.

The benchmark results for each test are given as time in seconds, and each test is run three times. The results are averaged and normalized using a Dell Precision 690 workstation with 2.0-GHz Intel Xeon 5130 processor, 4 x 4GB FB-DIMM DDR2 SDRAM (ECC) memory, NVIDIA Quadro Q600 graphics card, and Western Digital 500GB 7200 rpm hard drive. Having all test results normalized against a reference workstation allows for meaningful, confident comparisons between drastically differing workstations.

3. Materials

3.1. Rendering Workstation

Three rendering workstations are used for the comparative analysis of this study. Each workstation has different architecture that is designed to handle rendering of complex models. A brief explanation of each machine is given along with the specifications presented in Table *1*. SLI and Crossfire are technologies that link multiple GPUs together to produce a single graphic output. They are not used in any of the workstations for this study because this technology is not for rendering in 3ds Max Design.

Table 1. *Rendering workstations specifications.*

Name	VCA	FP8100	Q5000
Graphics Card			
Graphics Card	NVIDIA GRID K2	AMD FirePro W8100	NVIDIA Quadro K5000
Vendor	NVIDIA Corporation	Advanced Micro Devices Inc.	NVidia Corporation
# of cards	8	2	2
SLI / CrossFire	Off	Off	Off
Memory per card	4,096 MB	8,192 MB	4,096 MB
Core clock	797 MHz	300 MHz	324 MHz
Memory bus clock	1,249 MHz	150 MHz	162 MHz
Driver name	NVIDIA GRID K2	AMD FirePro W8100 (FireGL V)	NVIDIA Quadro K5000
Driver version	9.18.13.4052	15.201.2401.0	10.18.13.6175
Processor			
Processor	2 X Intel Xeon E5-2670	2 X Intel Xeon E5-2630 v3	Intel Core i7-4960X
Reported stock core clock	2,600 MHz	2,400 MHz	3,600 MHz
Maximum turbo core clock	2,600 MHz	3,200 MHz	3,700 MHz
Physical / logical processors	2-/-16	2-/-32	1-/-12
# of cores	16	16	6
Package	LGA2011	FCLGA2011-3	LGA2011
Manufacturing process	32 nm	22 nm	22 nm
Power	1,475 W	85 W	130 W
General			
Operating system	64-bit Win 7(6.1.7601)	64-bit Win 7 (6.1.7601)	64-bit Win 7 (6.1.7601)
Mother-board	4U rack mountable	Supermicro X10DAI	ASUS RAMPAGE IV BLACK EDITION
Memory	256 GB	64 GB	64 GB
Modules	Un-known	4 x 16 GB Micron DDR4 @ 1,866 MHz	8 X 8 GB G. Skill DDR3 @ 1,334 MHz
Hard drive model	139 GB XENSRC PVDISK SCSI Disk Device	1,425 GB Intel Raid 5 Volume SCSI Disk Device	512 GB Samsung SSD 840 PRO Series ATA Device

3.2. VCA

The NVIDIA GRID system has 20 physical CPU Cores (40 hyper-threaded), 2 TB of solid-state storage, and 256 GB of system memory. The network attached storage is used as primary storage for all workstations. As shown in

Table *2*, when configured for eight seats, each user workspace consists of a Quadro K5000-class GPU, 30 GB of system memory, and eight virtual Xeon processor cores. The GRID VCA is equipped with two Xeon E5 processors and four GRID K2 graphics cards, each containing two high-end GPUs.

Table *2* shows the hardware, users, and amount of computing resource available per user. The NVIDIA GRID technology also has the ability to offload graphics processing from the CPU to the GPU in virtualized environments.

Table 2. *Grid hardware.*

Setting	Total # Users	Resource Allocation Per User
1 GPU	8 Users	1) One K5000 class GPU 2) 30 GB of system memory 3) 8 virtual processor cores
2 GPU	4 Users	1) Two K5000 class GPU 2) 60 GB of system memory 3) 8 virtual processor cores
4 GPU	2 Users	1) Four K5000 class GPU 2) 60 GB of system memory 3) 16 virtual processor cores
8 GPU	1 User	1) Eight K5000 class GPU 2) 60 GB of system memory 3) 32 virtual processor cores

Due to the nature of the VCA's virtual environment, a template of installed programs and settings are used to create the user's instance of the OS. Once the user session is released, the cloned environment is discarded, and all changes that were not saved outside of the VMs allocated storage are lost. This allowed any testing done on the VCA to be performed in a controlled manner using clean installations of 3DS Max 2015, 3D Mark, and SPECapc. To add to the consistency of the test results across many user instances, all program settings remained set to their default installation states, and the software versions remained consistent on each workstation.

3.3. FP8100

This workstation, referred to as the FP8100, has two FirePro W8100 graphic cards with 64 GB of system memory and dual Xeon 8-core processors. The AMD FirePro W8100 workstation graphics card comes with 8 GB of GDDR5 onboard memory and adds up to 4.2 TFLOPS of peak single-precision floating-point performance. This helps speed up time required for high performance rendering.

3.4. Q5000

This workstation, referred to as the Q5000, has two Quadro K5000 graphic cards, 64 GB of memory, and the Intel Core i7-4960X Extreme Edition processor. The Quadro K5000 cards are built on the NVIDIA Fermi architecture and are designed to be used across a broad range of design, animation, and video applications. The Quadro 5000 graphics card is centered on 3D performance, as it can process up to 950 million triangles per second.

3.5. 3D Mark

The 3D Mark benchmarking software was installed on each workstation in order to measure the performance capabilities of multiple workstations using various components and architectures. The Fire Strike Ultra testing package was allowed to run using the software default settings to objectively score each workstation. The full results for this benchmark were delivered through a web-based results summary page. [22]

3.6. SPEC

SPECapc is the benchmarking suite for 3DS Max 2015 offered by SPEC. This software was installed on each workstation, and the benchmark was performed using the default benchmark settings. The default settings used HD for Test Resolution and No AA for Anti-Aliasing Level.

4. Methods

4.1. Rendering Standard

As there was no software known to the authors at the time of this study to adequately test the VCA performance, three separate 3D models were created and set as standards for comparison amongst the rendering engines used in this study. A simple model of a ball bearing was used for quick iterative rendering, a more complex ball bearing model was used to quickly compare rendering engines, and a computationally expensive model of a piece of engineering equipment known as the Green Machine™ was used to show more contrast between the rendering engines. To render the models in 3ds Max Design, NVIDIA's Iray can be used on the VCA and the Q5000, while V-Ray can be used across all three workstations. Iray and V-Ray implement a method of rendering known as ray tracing. This method involves following the path of light as it encounters objects in the environment, then uses this information to generate an image. As the number of rays increase, the quality and sharpness of the final image increase [24]. Ray tracing is able to produce photorealistic rendered images but can require more computational resources than rasterization and scanline rendering.

4.2. Ball Bearing

The ball bearing object consists of nine arrays of spheres ascending around a large central sphere. This object is used to decrease contact resistance for oilfield tools in over 30-

degree declination wells. A reflective material was added to the spheres along with several light sources. The model was then duplicated, and this model consisted of over 120 spheres equating to 1.6 million polygons (or polys). This is done to create a computationally expensive scene to show strong contrast between different rendering architectures when comparing amount of noise, render times, and iterations performed. Figure 4 shows a rendered image of the ball bearing scene using Iray renderer in 3ds Max Design.

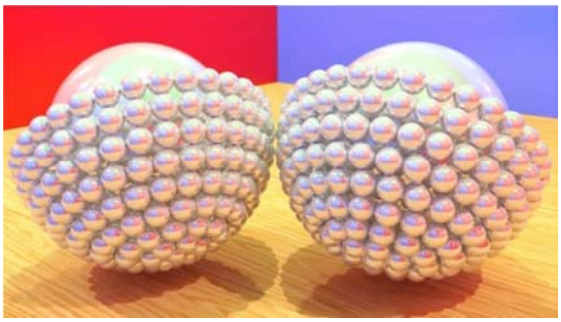


Figure 4. *Iray rendered ball bearing model with 120 reflective spheres and 1.6 million polys. Model credit: ADAMAX.*

4.3. Green Machine

The Green Machine™ is a generator manufactured by Electrotherm that produces power via the Organic Rankine Cycle (ORC), where heat is transferred to a fluid that is vaporized then expanded in a turbine to drive a generator, producing electricity. The Green Machine™ houses a turbine, a generator, and several heat exchangers for boiling, condensing, and pre-heating the working fluid. This system is utilized in the Concentrating Solar Power (CSP) plant at the Cleco Alternative Energy Center in Crowley, Louisiana [25]–[28]. Originally modeled in Solid Works, the 3ds Max Design imported Green Machine™ model, shown in Figure 5, consists of over 7 million polys.

Figure 5. *Internal Green Machine™ render with over 7 million polys.*

4.4. Mental Ray

In an effort to maintain consistency across multiple workstations, a standard number of iterations needed to be chosen for rendering tests. This number of iterations was found by rendering a model in 3DS Max 2015 using mental ray, a CPU-only rendering technology, then rendering the same model for the same amount of time using Iray while using only GPUs. The total number of iterations that the Iray renderer was able to perform in the allotted time was rounded and used as the iteration count for that scene across all workstations. This number of iterations proved to be sufficient to produce a final image of acceptable quality for comparison.

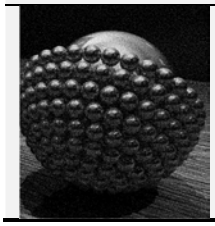
Using this chosen number of iterations and amount of time, the Ball Bearing model was rendered on the VCA using 2, 4, 6, and 8 GPUs and zero CPU cores. The Ball Bearing test scene was rendered using 8 GPUs both at a fixed time of 7 minutes and fixed iterations of 1300. This model was also rendered for 1300 iterations on the VCA using 8 GPUs and zero CPU cores. On the Quadro K5000 workstation, the model was rendered using 2 GPUs and zero CPU cores for both 7 minutes and 1300 iterations. The time required to produce each final render was recorded. The Image Quality Calculator was then used to determine the noise of each final render.

4.5. Rendering Metrics

When comparing the rendering performance of one workstation relative to another, there needs to be a consistency between final products that can be difficult to guarantee due to the differences in hardware and architecture. In order to confidently compare the final renders generated by different workstations, the metrics from which quality is derived needs to be quantified. The three metrics that were chosen to be analyzed using the Iray rendering engine were total iterations, render time, and amount of noise. The amount of noise in a render is calculated using an Image Quality Calculator software tool.

It is important to note the relationship of these metrics relative to one another. By definition, two images generated using the same number of iterations in the same rendering engine will have an identical amount of noise. Using these metrics, there are a few ways to perform this type of testing and comparison. Due to the number of iterations, the time required to render, and amount of noise being intrinsically linked, three possible testing structures exist: hold the number of iterations constant for a given render then compare the time required to render; hold the allotted time constant then compare number of iterations performed; or choose some acceptable amount of noise and iteratively render until the chosen amount is achieved, then compare the time or number of iterations required to render. Table 3 shows logarithmic variations of iterations and resulting noise.

Table 3. *Render ball bearing at 1, 10, 100 and 1000 iterations and resulting noise.*

	Iterations	Noise
	1	117

	Iterations	Noise
	10	18.4
	100	14.3
	1000	11.6

The metrics specific to V-Ray rendering considered for comparison are render time, sampling level, and noise threshold. Similar to Iray, setting the maximum render time in V-Ray will specify the maximum time for refining the image. The sampling level of each rendered image is given by V-Ray in samples per pixel (SPP). This is comparable to the iterations metric used by Iray. By choosing to hold time constant, this value shows how many passes were performed on each render. When rendering using V-Ray, the noise within an image is compared to a noise threshold. As the noise falls below the noise threshold, the value of the threshold is lowered, and the image is processed further. This continues until the threshold reaches some specified value. If no value is specified, the default noise threshold is 0.005. This measure of noise behaves similarly to the noise of an image in that it converges to some value that is considered indicative of a sufficiently processed image.

Although Iray and V-Ray cannot be compared directly, as they use different material libraries, the variances between image qualities for set times can be shown. For this study, time was the metric chosen to hold constant. This decision was made due to the nature of the testing required to produce meaningful results. In order to perform a large number of renders on multiple workstations using many different models and settings, time was the biggest constraint. Holding time constant allows for rapid testing of many scenes and settings. As the image quality can be calculated on any image regardless of the rendering engine, this calculator can be used across all platforms. The amount of measurable noise decreases continuously over time, and this can be plotted by performing iterative renderings at fixed times. The curve profile produced with the image quality calculator can be used to analyze the noise amongst different models or rendering engines [29].

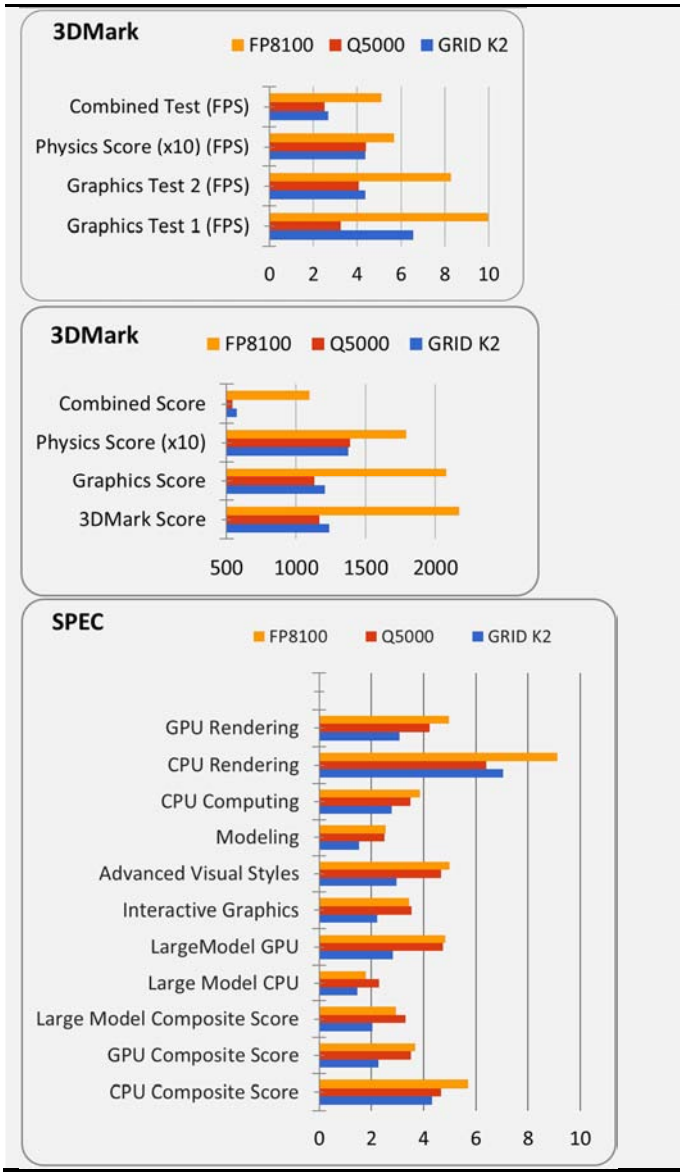
5. Results

5.1. 3D Mark

The 3D Mark benchmark test was performed on each workstation and a single GPU. Due to limitations with 3D Mark, the multi-GPU systems were not able to be tested in this study. The Fire Strike Extreme benchmark test has multi-GPU support only if the system is using SLI or CrossFire. Therefore, the 3D Mark results are only indicative of each single GPU performance.

As shown in Table 4, the GRID K2 outperformed the Quadro K5000 in all tested metrics, excluding the Physics Test Score and Physics Test FPS. A large discrepancy was seen between the GRID K2 and the Quadro K5000 average FPS during the first Graphics Test. The FirePro W8100 greatly outperformed both the GRID K2 and the Quadro K5000 in all tests and received a score twice as high in the Combined Score and FPS.

Table 4. *3D Mark results (top), and SPEC results (bottom).*

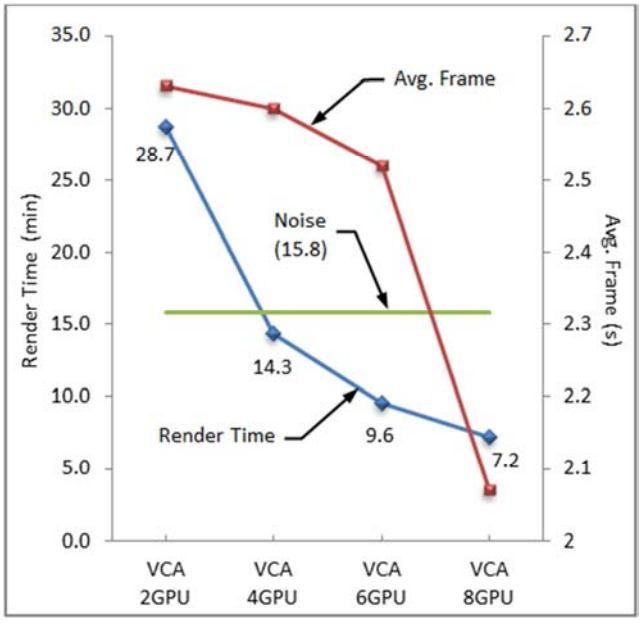


5.2. SPEC

The SPEC 3DS Max 2015 benchmark was able to recognize all 8 GRID K2s along with the dual GPUs in the FirePro W8100 and Quadro K5000 workstations. However, the software only utilized the resources of each single GPU. This leaves the SPEC results to also only be indicative of each single GPUs performance. As shown in Table 4, the FirePro W8100 outperformed the GRID K2 K5000 in all GPU tests by a significant margin, while only slightly outperforming the Quadro K5000.

5.3. Rendering Standard Tests

Allocating additional GPUs on the VCA resulted in a decrease in total time required to produce a render of sufficient quality. The ball bearing model, shown in Figure 4, was rendered with the fixed amount of 1300 iterations on the VCA using 2, 4, 6, and 8 GPU allocations. All renders were generated using zero allocated CPU cores. As seen in Figure 6 on the left, 2 GPUs performed 1300 iterations in 28.7 minutes, 4 GPUs in 14.3 minutes, 6 GPUs in about 9.6 minutes, and 8 GPUs in about 7.2 minutes. If the decrease in required rendering time is compared using the number of allocated GPUs, the near-linear scalability advertised by NVIDIA appears to hold true at this scale. However, when looking into the average frame time, shown in red, there is considerably more difference between 8 and 4 GPUs than between 4 and 2 GPUs. Two GPUs gave an average frame time of 2.63s or 1.32s per GPU, while 8 GPUs had 2.07s or 0.26s per GPU. Since the rendering time has near-linear scalability, the average frame time difference means there was much more deviation per rendered frame with 8 GPUs than with any other configuration. The noise is constant as it is directly proportional to the amount of iterations; therefore, fixed iterations is analogous to a fixed amount of noise.



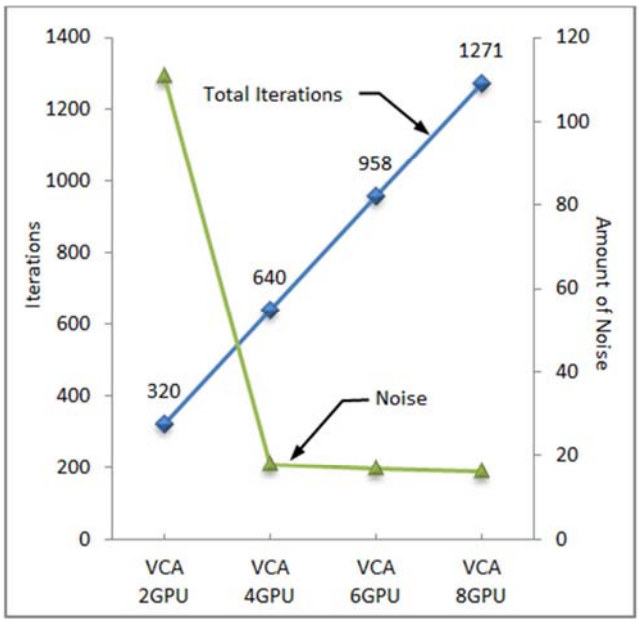


Figure 6. *Ball bearing render in Iray with fixed iterations (top) and fixed time (bottom) on VCA with various GPU setting and Q5000 workstations.*

Using a fixed time of seven minutes, the ball bearing model was rendered using the VCA at different GPU allocations and zero CPU cores. As shown in Figure 6 on the right, there is considerably more noise when using two GPUs as the noise threshold was not reached. The iterations showed a near linear decrease with less GPU resources with an average of 160 iterations per GPU regardless of how many GPUs are used. The number of iterations per minute performed by each GPU only decreases very slightly when the number of GPUs allocated is increased. When two GPUs are allocated, each GPU is capable of performing 22.86 iterations per minute, while when 8 GPUs are allocated, each GPU performed 22.70 iterations per minute or 0.7% less. Though very small, this decrease in performance seems to increase with the number of allocated GPUs and may require further investigation before claims can be made about the large-scale scalability of the VCA.

Rendering the ball bearing model using two Q5000 GPUs and allocating zero CPU cores required 33.5 minutes which is 17.47% more time than the VCA using two GRID K2 GPUs. With fixed time, the Q5000 performed 268 iterations or 134 per GPU with a noise level of 112.96. Therefore, the VCA performed 19.4% more iterations than the Q5000 with the same amount of GPUs. This would be a considerable difference when rendering an animation consisting of thousands of images.

In order to determine the time required for the VCA to produce a render of sufficient quality, the ball bearing model was rendered by incrementally increasing the render time until a satisfactory noise profile was created, as shown in Figure 7 on the left. This measure of image quality is the amount of noise in an image, and as shown, it converges to some value that is considered indicative of a sufficiently processed image. For the ball bearing model, this profile was created using a range between 1 and 20 minutes for Iray and 1 and 10 minutes for V-Ray. Investigating the changes in the

amount of noise for each increase in allocated time allows two platforms to be compared based on the time required to produce a render of sufficient quality. As shown in Figure 7on left, the VCA was able to greatly reduce the noise of the ball bearing model when allowed to render for 3 minutes with Iray and 2 minutes with V-Ray.

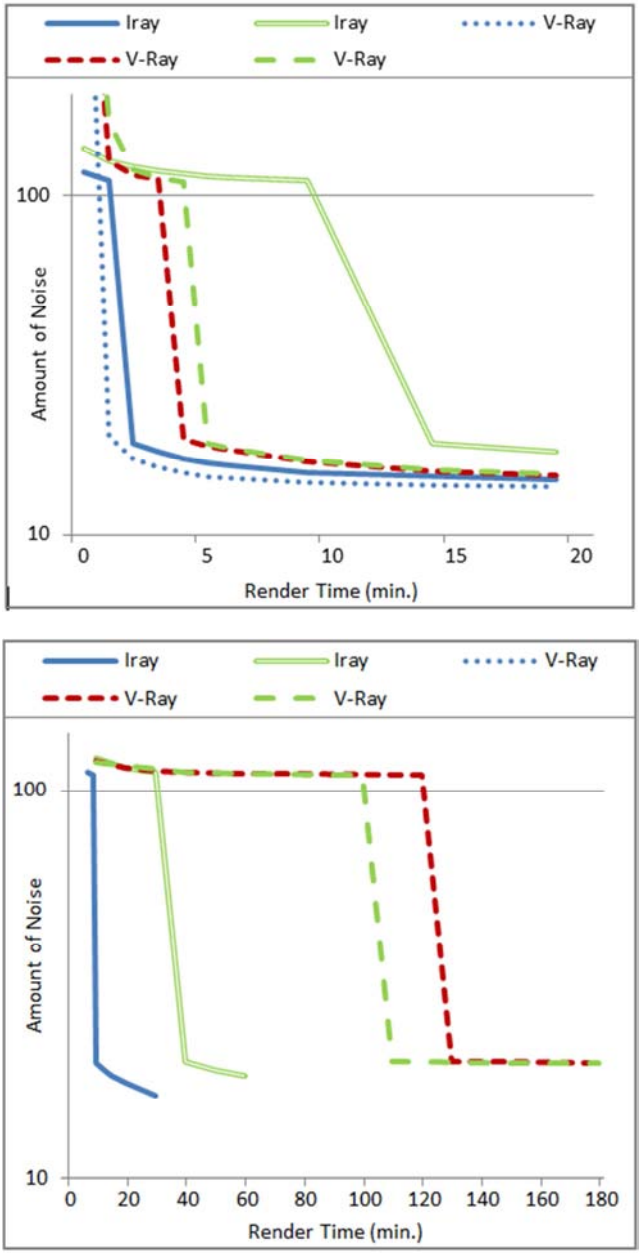


Figure 7. *Image Quality Calculator results for ball bearing model (top) and Green Machine™ model (bottom).*

The Q5000 rendered images were not able to achieve this decrease in noise unless allowed to render for longer than 10 minutes with Iray and 6 minutes with V-Ray. The FP8100 achieved the noise drop in 5 minutes, slightly less time than the Q5000. Identifying the point where a sharp decrease in noise is observed can be beneficial as the change in noise suffers from diminishing return after a certain point. Once this large drop in noise is achieved, subsequent iterations performed on the render will have less impact on the total possible quality as the curve asymptotes to an approximate noise amount. The level of noise to which the iterative rendering process converges can be considered an image of sufficient quality. Since this asymptotic noise value is indicative of sufficient image quality, reducing the amount of time spent rendering the image as it approaches this value can increase productivity and efficiency in industry.

The Green Machine™ rendering standard model, shown in Figure 5, was rendered using the same methods as the ball bearing model, and the time required to achieve a noise indicative of a render of sufficient quality was compared. As shown in Figure 7 on the right, the VCA surpassed the noise threshold at 10 minutes with Iray where the Q5000 took over 30 minutes. The Q5000 took slightly over 100 minutes to converge with V-Ray where the FP8100 took over 120 minutes. Therefore, the FP8100 was slightly faster in the simpler ball bearing model but slightly slower with the more complex Green Machine™ model. Currently, the VCA is not able to render the Green Machine™ model in V-Ray RT due to unhandled exception errors; therefore, the results are not available.

As stated in the rendering metrics paragraph above, V-Ray offers different metrics when comparing logs from the rendered images. These are analyzed to further validate the results obtained from using the image quality calculator. The maximum number of paths per second represents the power of the hardware performing the render. This number stays relatively constant irrespective of the rendering time allocated. As to be expected, a rendering platform with more available resources will be able to calculate more paths in a given amount of time. For the ball bearing model, the VCA performed an average of 13.94 paths per second, ignoring the one-minute render. The FP8100 and Q5000 performed an average of 3.31 and 3.77 paths per second, respectively.

To analyze the render passes performed, the sampling level, which is similar to iterations used by Iray, is analyzed for the three rendering engines. Using a linear regression on the samples per pixel (SPP) and the render time from the ball bearing model, the VCA performed approximately 4.24 SPP each second while the FP8100 and Q5000 performed approximately 0.96 SPP each second. At the 1920x1080 resolution used for all renders, this equates to performing at 8.8E6, 1.99E6, and 1.98E6 samples per second from the VCA, FP8100, and Q5000 rendering engines, respectively. Therefore, the VCA performed 4.4 times as many samples per second as the FP8100 and Q5000, which were roughly the same.

When rendering using V-Ray, the noise within an image is compared to a noise threshold. As the noise falls below the noise threshold, the value of the threshold is lowered, and the image is processed further. This continues until the threshold reaches some specified value or the render is stopped due to reaching time or sampling maximums. If no value is specified, the default noise threshold is 0.005. A graph showing the final noise threshold compared to the amount of noise found from the image quality calculator for the ball bearing model is shown in Figure 8. As shown, both noise calculators exhibit similar curve profiles.

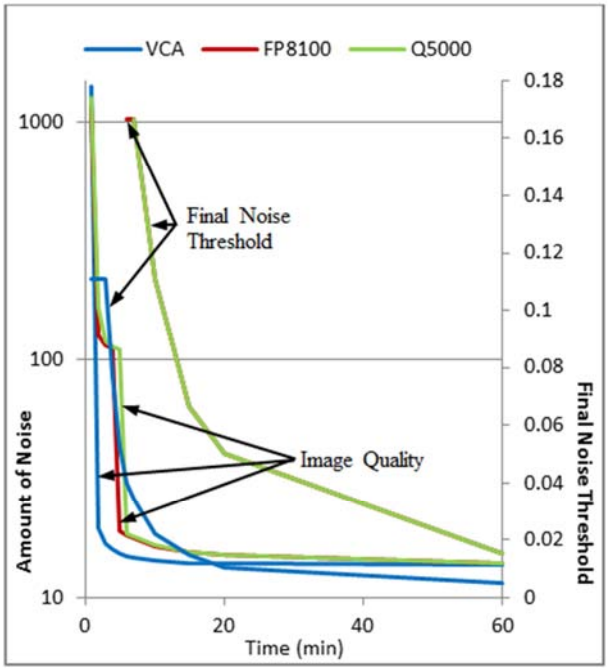


Figure 8. *Image quality (amount of noise) and final noise threshold using V-Ray for ball bearing model.*

The FP8100 and Q5000 took over 7 minutes before a notable decrease in the noise threshold was shown, where the VCA showed a decrease after 3 minutes. The noise threshold of 0.005 was reached for the VCA in 38 minutes and 46 seconds, and no more refining occurred for the image. The FP8100 and Q5000 did not manage to reach the noise threshold in the time provided.

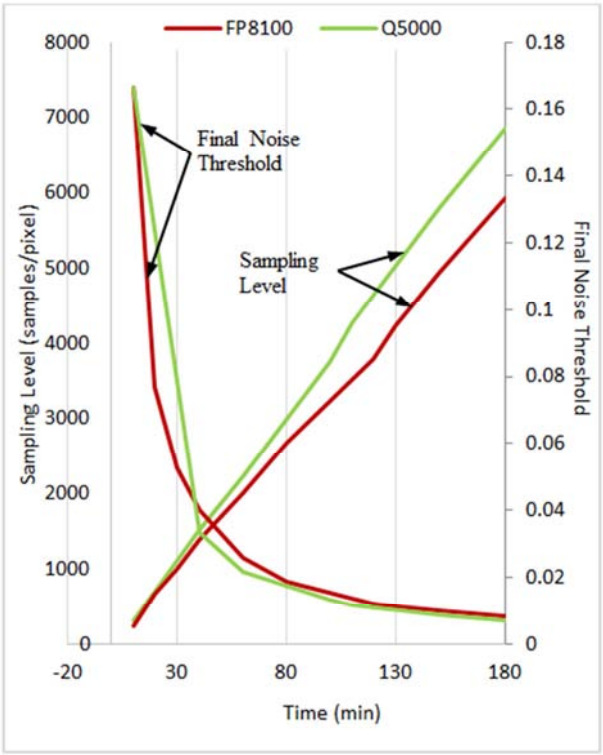


Figure 1. *Final noise threshold, and sampling level using V-Ray for Green Machine™ model.*

For the Green Machine™ model, the final noise threshold and sampling level is shown in Figure 9. The VCA results are not shown because, as stated before, the VCA is currently not able to render the Green Machine™ model in V-Ray RT due to unhandled exception errors.

Although a near identical final noise threshold was reached, the FP8100 took slightly longer to archive a satisfactory amount of noise from the image quality calculator as stated before. This was further confirmed with the higher sampling level rate achieved by the Q5000.

6. Discussion and Conclusions

To identify a solution for rendering animations of complex CAD models efficiently, this study performed an objective analysis of the unique, GPU-focused architecture present in the NVIDIA GRID VCA and evaluated its performance relative to other high-end workstations. To properly evaluate and compare the workstations, a method of testing needed to be used that would remain consistent across differing platforms. Benchmarking software was used as a way to score the different workstations based on many metrics. However, the two benchmarking suites used, SPEC and 3D Mark, only tested the capabilities of a single GPU regardless of configuration or architecture. Consequentially, the results of the benchmarking software could only be used as a direct GPU to GPU comparison and not as a comparison of the capabilities of the respective architectures, which was the focus of this study. For this reason, the testing method chosen consisted of creating computationally expensive models, then using the VCA and the two workstations to render images of these models. The metrics used for comparison were the time required to render an image of sufficient quality and the noise in an image, which was calculated using a third-party tool developed by the University of Manitoba's Physics and Astronomy Department. By analyzing the image quality results of the three workstations, the diminishing return in noise per unit of render time became evident. There existed a specific time during the rendering of a model where the rate of decrease in noise dropped dramatically and the total noise in an image began to converge to a value. By finding this time or noise value and not allowing the image to render far past this point, images were rendered more time-efficiently. Utilizing the GPU CUDA cores in the VCA can drastically increase rendering speeds without compromising quality, and the remote access from consumer laptop or tablet gives clear advantage. Using VCA resulted in 417% less time or 706% better quality than the Q5000 workstation with the Green Machine™ model in Iray. The largest performance advantage was the near-linear scaling that seemed to live up to NVIDIA's advertised scaling. The rendering performance of each individual GPU, measured in iterations per minute for Iray, only slightly decreased as the number of allocated GPUs increased. In the case of large-scale VCA stacking, this decrease in performance may need to be investigated as the decrease in performance grows with each subsequent GPU addition.

To outsource computational work, options such as cloud computing or sending projects to render farms are currently available. Alternatively, a customized render farm or server can be constructed for use in-house, which would require dedicated IT personnel to set up and maintain the system as it is used. This need for IT personnel can be minimized when using the NVIDIA GRID VCA due to the nature of the system. By delivering the VM to each client in the form of a master template copy, in which no permanent changes can be made to the template from an end-user seat, the likelihood of technical malfunctions due to user error can be almost eliminated. Additionally, the NVIDIA GRID VCA has the added advantage of being delivered as a turnkey alternative to render farms. Traditional render farms can take a considerable amount of time to assemble, configure, and network properly while the NVIDIA GRID VCA can be unboxed and set up by a novice user in about an hour. If the desire is to streamline the production process by keeping the work in-house, the NVIDIA GRID VCA system can serve as a primary rendering platform. This provides a turnkey solution for companies with limited IT resources and will eliminate the need to have customized rendering farm or server built. A secondary study should be conducted in order to determine the change in performance, if any, between the NVIDIA GRID VCA and a traditional render farm of comparable specifications.

References

[1] D. Raj, "Solidworks Performance in a box: NVIDIA GRID VCA," *WordPress*, 2014. [Online]. Available: https://solidworksexpert.wordpress.com/2014/09/30/solidworks-performance-in-a-box-nvidia-grid-vca/. [Accessed: 30-May-2016]

[2] A. Herrera, "NVIDIA GRID vGPU: Delivering Scalable Graphics-Rich Virtual Desktops.," 2015.

[3] L. Adhianto, S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey, and N. R. Tallent, "Improving the user experience of the rCUDA remote GPU virtualization framework," *Concurr. Comput. Pract. Exp.*, vol. 22, no. 6, pp. 685–701, 2014.

[4] J. O. Oredo and J. Nijihia, "Challenges of cloud computing in business: Towards new organizational competencies," *Int. J. Bus. Soc. Sci.*, vol. 5, no. 3, pp. 150–161, 2014.

[5] C. N. Höfer and G. Karagiannis, "Taxonomy of Cloud Computing Services," *IEEE Globecom Work.*, pp. 1345–1350, 2010.

[6] H. Hussain, S. U. R. Malik, A. Hameed, S. U. Khan, G. Bickler, N. Min-Allah, M. B. Qureshi, L. Zhang, W. Yongji, N. Ghani, J. Kolodziej, A. Y. Zomaya, C. Z. Xu, P. Balaji, A. Vishnu, F. Pinel, J. E. Pecero, D. Kliazovich, P. Bouvry, H. Li, L. Wang, D. Chen, and A. Rayes, "A survey on resource allocation in high performance distributed computing systems," *Parallel Comput.*, vol. 39, no. 11, pp. 709–736, 2013.

[7] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," *5th Int. Jt. Conf. INC, IMS, IDC*, pp. 44–51, 2009.

[8] J. R. Annette, W. A. Banu, and P. S. Chandran, "Rendering-as-a-Service: Taxonomy and Comparison," *Procedia Comput. Sci.*, vol. 50, pp. 276–281, 2015.

[9] V. V. Kindratenko, J. J. Enos, and G. Shi, "GPU clusters for high- performance computing," in *IEEE International Conference on Cluster Computing and Workshops*, 2009, pp. 1–8.

[10] J. C. Phillips, J. E. Stone, and K. Schulten, "Adapting a message-driven parallel application to GPU-accelerated clusters," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, 2008.

[11] M. G. Pelletier, "Parallel Algorithm for GPU Processing; for use in High Speed Machine Vision Sensing of Cotton Lint Trash," *Sensors*, vol. 8, pp. 817–829, 2008.

[12] Y. Cao, H. Wang, and Z. Ai, "Distributed Multi-GPU Accelerated Hybrid Parallel Rendering for Massively Parallel Environment," *Proc. Int. Conf. Virtual Real. Vis.*, pp. 30–36, 2014.

[13] H. Takizawa and H. Kobayashi, "Hierarchical parallel processing of large scale data clustering on a PC cluster with GPU co-processing," *J. Supercomput.*, vol. 36, no. 3, pp. 219–234, 2006.

[14] NVIDIA Corporation, "NVIDIA GRID K1 and K2 Graphics-Accelerated Virtual Desktops and Applications," 2013.

[15] E. Lindholm, J. Nickolls, S. Oberman, and J. Montrym, "NVIDIA T ESLA : A Unified Graphics and Computing Architecture," *IEEE Comput. Soc.*, pp. 39–55, 2008.

[16] T. Li, V. Narayana, and T. El-Ghazawi, "Exploring Graphics Processing Unit (GPU) Resource Sharing Efficiency for High Performance Computing," *Computers*, vol. 2, no. 4, pp. 176–214, 2013.

[17] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU Computing," *Proc. IEEE*, vol. 96, no. 5, pp. 879–899, 2008.

[18] X. Li and Y. Wang, "Research of Real-Time Terrain Rendering on GPU," *Seventh Int. Symp. Comput. Intell. Des. Res.*, pp. 14–17, 2014.

[19] J. Mielikainen, B. Huang, H.-L. a. Huang, M. D. Goldberg, and a. Mehta, "Speeding Up the Computation of WRF Double-Moment 6-Class Microphysics Scheme with GPU," *J. Atmos. Ocean. Technol.*, vol. 30, no. 12, pp. 2896–2906, Dec. 2013.

[20] NVIDIA Advanced Rendering Center, "NVIDIA Iray Whitepaper," Berlin, 2012.

[21] J. West, "Image Quality Calculator," *University of Manitoba*, 2009. [Online]. Available: http://www.umanitoba.ca/faculties/science/astronomy/jwest/plugins.html. [Accessed: 02-Mar-2016]

[22] Futuremark Corporation, "3D Mark," 2013.

[23] C. Sandifer, "SPECapc for 3ds Max 2015™," *Standard Performance Evaluation Corporation*, 2014. [Online]. Available: http://spec.org/gwpg/apc.static/max2015info.html. [Accessed: 02-Mar-2016]

[24] B. S. G. Parker, H. Friedrich, D. Luebke, K. Morley, J. Bigler, J. Hoberock, D. Mcallister, A. Robison, A. Dietrich, G. Humphreys, M. Mcguire, and M. Stich, "GPU Ray Tracing," *Commun. ACM*, vol. 56, no. May, pp. 93–102, 2013.

[25] J. R. Raush, T. L. Chambers, B. Russo, and K. A. Ritter III, "Demonstration of Pilot Scale Large Aperture Parabolic Trough Organic Rankine Cycle Solar Thermal Power Plant in Louisiana," *J. Power Energy Eng.*, vol. 2013, no. December, pp. 29–39, 2013.

[26] T. Chambers, J. Raush, and G. Massiha, "Pilot solar thermal power plant station in southwest Louisiana," *Int. J. Appl. Power Eng.*, vol. 2, no. 1, 2013.

[27] T. Chambers, J. Raush, and B. Russo, "Installation and Operation of Parabolic Trough Organic Rankine Cycle Solar Thermal Power Plant in South Louisiana," *Energy Procedia*, vol. 49, pp. 1107–1116, 2014.

[28] K. A. Ritter III and T. L. Chambers, "Educational Gaming and Use for Explaining Alternative Energy Technologies," *Int. J. Innov. Educ. Res.*, vol. 2, pp. 30–42, 2014.

[29] J. Pizzini, "GPU Rendering vs. CPU Rendering – A method to compare render times with empirical benchmarks," *Boxxtech*, 2014. [Online]. Available: http://blog.boxxtech.com/2014/10/02/gpu-rendering-vs-cpu-rendering-a-method-to-compare-render-times-with-empirical-benchmarks/. [Accessed: 02-Mar-2016]