




Evaluación del muro de visualización implementado con Chromium en el Cluster Peyote del LCI-ITVer

Visualization wall evaluation implemented with Chromium in Peyote Cluster of LCI-ITVer

Abelardo Rodriguez Leon1 , Guillermo E. Ovando Chaco2 , Marco A. Gomez Abascal3 , Juan C. PrinceAvelino4 

Unidad de Ingeniería Mecánica. Laboratorio de Computo Intensivo.
Instituto Tecnológico de Veracruz

Cal. M. A. de Quevedo 2779, Col. Formando Hogar, Veracruz, Ver. México. 91860
1arleon@itver.edu.mx, 2geoc@itver.edu.mx, 3marcoantonio48@hotmail.com, 4jcpa1000@gmail.com

PALABRAS CLAVE:

Muro de visualización, Evaluación de desempeño, Chromium

RESUMEN

La computación representa una valiosa herramienta para la investigación científica, ya que facilita el análisis de grandes volúmenes de información, así como la posibilidad de mostrar esta información de manera gráfica, permitiendo incluso la interacción con ella.

Una buena representación gráfica es capaz de transmitir conceptos complejos, lo que le facilita al investigador enfocarse de forma mas eficiente en los conceptos abstractos de su caso estudio.

En el ámbito científico, y especialmente en lo referente al estudio de fenómenos térmicos y de combustión (investigación que se realiza en el área de termofluidos de la Unidad de Ingeniería Mecánica del Instituto tecnológico de Veracruz) es común procesar complejos modelos matemáticos que generan grandes cantidades de información y requieren grandes cantidades de poder computo. Esto ha hecho una necesidad constante la mejora del hardware y software en Laboratorio de Computo Intensivo (LCI) del ITV.

El LCI cuenta con 2 clústeres Beowulf para computación de alto rendimiento (HPC High Performance Computing).

Dichos cluster (Agave y Nopal) tienen un poder de procesamiento computacional superior al que sería posible conseguir, con equipos individuales de la gama más alta disponible. Estos clústeres, además de resultar económicamente asequibles, permiten la adición de mas equipos ya que su naturaleza es modular; según la disponibilidad de equipos. La capacidad de computo de los clusters HPC se utiliza para la ejecución de procesos de simulación altamente demandantes.

Una mejora reciente del LCI se tiene en la representación de los resultados, a través de gráficos de computadora, lo cual requiere un procesamiento computacional no estrictamente vinculado con el cómputo de alto rendimiento. Este procesamiento gráfico requiere de un tipo diferente de clúster específicamente dedicado al procesamiento gráfico, por lo cual se ha creado un tercer cluster llamado Peyote que ayuda al control de un muro de visualización.

El trabajo presentado forma parte del macro proyecto “Visualización de un sistema de partículas 3D aplicable a la simulación de fenómenos en términos de fluidos y combustión”, con clave 5488.14-P y financiado por la DGEST-SEP. El trabajo específico de este artículo se refiere a la implementación y evaluación de un clúster Beowulf llamado Peyote que se usará para procesamiento gráfico en un muro de visualización (arreglo de monitores), para mostrar la simulación 3D de un sistema partículas y se desarrolla en el Cuerpo Academico ITVER-CA-3 “Análisis y Simulación en Termofluidos ” del Instituto Tecnológico de Veracruz.

KEYWORDS:

Visualization Wall, benchmark
performace , Chromium

ABSTRACT

The computing is a valuable tool for scientific research, because it facilitates the analysis of large volumes of information and the ability to display this information graphically, allowing even the interaction with it.

A good graphic representation is able to convey complex concepts, making it easier for the researcher focus more efficiently on the abstract concepts of your case study.

In science, and especially with regard to the study of thermal phenomena and combustion (research conducted in the area of thermal fluids Unit Mechanical Engineering from the Technological Institute of Veracruz) it is common to process complex mathematical models that generate large amounts of information and require large amounts of computing power. This has made a need constant the improve of the hardware and software, on compute-intensive Laboratory (LCI) of ITV.

The LCI has 2 Beowulf clusters for high-performance computing (HPC High Performance Computing). Such cluster (Agave and Nopal) Have a higher computing processing power than the one would possible achieve, with individual equipment of the highest range available. These clusters, as well as being affordable, allow the addition of more equipment because of its modular nature; depending on the availability of equipment. The computing capacity of HPC clusters are used to run highly demanding simulation processes.

A recent improvement in the LCI is the representation of the results through computer graphics, which requires a computer processing not strictly linked to high performance computing. This graphics processing requires a different type of cluster specifically dedicated to graphics processing, so that it have been created a third cluster called Peyote that helps control a display wall.

The work presented is part of macro project "Visualization of a 3D particle system applicable to the simulation of phenomena in terms of fluids and combustion", with key-P 5488.14 and funded by the DGEST SEP. The specific task of this article refers to the implementation and evaluation of a Beowulf cluster called Peyote to be used for graphics processing in a visualization wall (arrangement monitors) to display the 3D simulation of a particle system that develops in the Academic body ITVER-CA-3 "Analysis and Simulation thermofluids" Veracruz Institute of Technology.

1 INTRODUCCIÓN

Los sistemas de visualización tienen dos objetivos principales. El primero es ayudar a manejar y a asimilar el conocimiento. El segundo objetivo es mostrar un amplio caudal de información (del orden de teras o peta-bytes). Lograr el primer objetivo, se puede llevar a cabo de varias maneras, por ejemplo usando complejos paradigmas que suelen ser metáforas del mundo real, aplicados a las interfaces gráficas en los sistemas. Un caso relevante en este sentido es el trabajo de [1], en el cual se desarrolló un ambiente de visualización colaborativo en 3D desplegable en grandes pantallas. Para lograr el segundo objetivo de mostrar amplios caudales de información se usan los muros de visualización. Estos muros representan una alternativa interesante que supera a muchos sistemas gráficos de alta resolución, como es el caso del proyecto Stallion de la Universidad de Texas [2] que ofrece una resolución de 328 megapíxeles (38400x8000), mucho más de los que proporciona el dispositivo de mayor resolución en la actualidad. Un muro de visualización utiliza un arreglo bidimensional de monitores (ver figura 1) los cuales en su conjunto dan una resolución superior a la de dispositivos convencionales, con menor costo por píxel. Sin embargo, tienen algunos inconvenientes entre los que se encuentra el ajuste entre ellos debido a los biselados de los monitores que dividen la imagen, lo que le quita algo de realismo a la proyección.

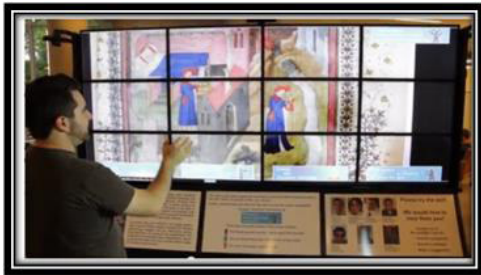


Figura 1. Ejemplo de un muro de visualización

Los sistemas de visualización, como los muros de visualización mejoran a medida que hay avances en las tarjetas gráficas (GPU) y la tecnología LCD.

2 DESARROLLO

Este trabajo consiste en la implementación y medición de la eficiencia (throughput) de un muro de visualización

implementado en el Laboratorio de Computo Intensivo del Instituto Tecnológico de Veracruz.

La implementación del muro pasó por el diseño, instalación y configuración del cluster de visualización Peyote. Para ello se usó el middleware Rocks versión 5.3 [3] que fue la última versión que incluyó el paquete (Roll) Viz. El Roll Viz [4] contiene entre otras aplicaciones a SAGE y CHROMIUM. Estas dos aplicaciones son necesarias para la configuración y puesta en marcha del muro. SAGE [5] es necesaria para la administración inicial y despliegue de demos de imágenes y vídeos los cuales vienen incorporados en la misma instalación. CHROMIUM [6] es necesario para dar soporte a animaciones desarrolladas en la librería gráfica estándar OpenGL o sus motores gráficos derivados. En OpenGL [7] se realizarán las simulaciones 3D finales del sistema de representación de partículas en 3D que se está desarrollando.

2.1. INSTALACIÓN DEL MURO (HARDWARE-SOFTWARE)

El hardware empleado en el cluster Peyote se enlista a continuación.

- 1 Frontend con Core 2 Quad core con 4 GB
- 3 Nodos Core 2 Quad core con 4 GB
- 4 Nodos I5 Quad core con 4 GB Ram
- 10 Monitores CRT 14" de 1024x768 (7.5 Megapíxeles)
- 6 Nvidia GTX 650
- 1 Nvidia GT520
- 1 Nvidia GT220
- Red ethernet de 100 MB
- Red ethernet de 1 GB

Todos los nodos tienen solo una tarjeta gráfica. Cuatro nodos tienen 1 monitor y los tres restantes 2 monitores. Rocks configurará automáticamente en modo simple los nodos con un solo monitor y en modo meta los nodos con dos monitores.

En la figura 2 se puede observar el diagrama de conexión y una fotografía física de como se conectaron los equipos en el muro de visualización.

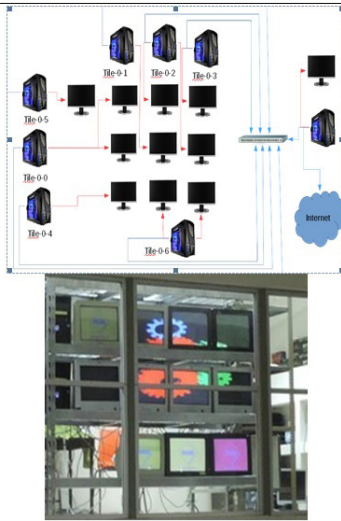


Figura 2. Diagrama de conexión del muro de visualización

Los pasos seguidos en la instalación se enumeran a continuación.

- Instalación del frontend.
- Actualización de los driver nvidia en el frontend
- Configuración del servidor X en el frontend
- Instalación de los tiles (nodos de visualización)
- Actualización de los driver nvidia en los tiles.

Posteriormente se procede a configurar el hardware para crear el muro de visualización.

2.2. CONFIGURACIÓN DEL MURO DE VISUALIZACIÓN

Los pasos desarrollados en la creación del muro de visualización son los siguientes:

Configuración de la disposición de los nodos en el muro de visualización.

Una vez agregados los tiles al clúster, es necesario indicar a Rocks la disposición que cada uno tendrá en el muro. Esta disposición se describe en el archivo `/etc/xml/layout.xml`.

Una versión gráfica de esta disposición se muestra en la Figura 3.

Tile-0-5:0.0	tile-0-0:0.0	tile-0-2:0.0
--	tile-0-0:0.1	tile-0-1:0.0
--	tile-0-4:0.0	tile-0-6:0.1

Figura 3. Esquema de posiciones de nodos dentro del muro

La notación usada por rocks para nombrar a los nodo tile se puede observar en la figura 4.

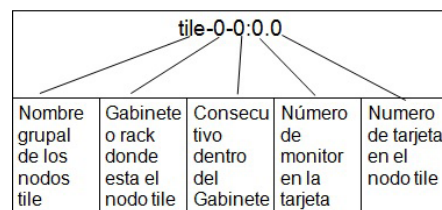


Figura 4. Nomenclatura de los nombres de los tiles en el muro

A continuación se realizaron los siguientes pasos:

- Se removió la configuración por defecto.
- Se cargó la nueva configuración de la distribución en los tiles, como se indicó en el archivo `/etc/xml/layout.xml`

• Se resincronizó el muro para que Rocks haga la actualización de los archivos de configuración del servidor X en los archivo `/etc/X11/xorg.conf` de cada uno de los nodos tile conectados al clúster.

Rocks cuenta con dos modos de utilización de GPUs para los nodos tile configurados en el clúster.

Modo	Descripción
Simple	Indica a Rocks que se utilizará un único monitor por cada GPU (un servidor X por cada nodo tile). 1 GPU por nodo tile / 1 Monitor por nodo tile / 1 Servidor X por nodo tile
Meta	Emplea una combinación de las tecnologías "TwinView" y "Xinerama" para unificar todas las GPUs de cada nodo, en un único monitor lógico. Se tiene un único servidor X por cada nodo tile. Esta configuración debe emplearse en caso de tener más de una tarjeta gráfica conectada a algún nodo, y uno o dos monitores conectados (s) a cada una de estas tarjetas. nGPUs por nodo tile / n Monitores por nodo tile / 1 Servidor X por nodo tile

CONFIGURACIÓN DE CHROMIUM

Chromium es la tecnología que permite la distribución del procesamiento de gráficos OpenGL en un clúster de computadoras. La arquitectura de Chromium está formada por 3 capas, que se pueden observar en la figura 5. Dichas capas cumplen distintas funciones, que se describen a continuación.

Aplicación OpenGL Ej. glxgears		Aplicación OpenGL Ej. fonttest	
Faker OpenGL			
Mothership			
Servidor Chromium			
Nodo 1	Nodo 2	Nodo 3	Nodo n
OpenGL	OpenGL	OpenGL	OpenGL

Figura 5. Arquitectura en capas de Chromium

•Faker: Esta capa es un proxy del subsistema OpenGL. Dicho proxy atrapa y direcciona el trabajo OpenGL de las aplicaciones (capa superior) hacia las capas inferiores de Chromium. Gracias al Faker, no es necesario realizar cambios en la programación de las aplicaciones OpenGL para poder aprovechar ésta tecnología.

•Mothership: La capa de mothership se encarga de identificar y configurar cada uno de los nodos de procesamiento que se utilizarán para la distribución del renderizado gráfico en las pantallas correspondientes del muro de visualización.

•Servidor chromium: Es la capa encargada de sincronizar los procesos de renderizado grafico en los diferentes GPUs, mediante mensajes entre los nodos involucrados en el procesamiento gráfico del clúster.

La distribución Rocks 5.3 rolled tacos, cuenta con la versión 1.9 de Chromium y requiere algunos ajustes en las variables de entorno antes de poder utilizar Chromium. Las siguientes variables deben ser modificadas (únicamente en el frontend).

PATH: La variable PATH proporciona la ruta de los ejecutables del roll Viz, ya que esta no se agrega por default cuando se instala este roll. Se debe definir en el archivo de configuración .bashrc (~/.bashrc), de la siguiente forma: export PATH=/opt/viz/bin:\$PATH, que es donde se encuentran los ejecutable de Chromium.

LD_LIBRARY_PATH: Esta variable contiene las rutas donde se buscarán las librerías precompiladas en el sistema. Se debe agregar en el archivo de configuración .bashrc, de la siguiente forma: export LD_LIBRARY_PATH=/opt/viz/lib:\$LD_LIBRARY_PATH.

CRMOTHERSHIP: Chromium utiliza esta variable de entorno para determinar el host en el que se ejecutará el proceso Mothership. Tambien se agrega al archivo de configuración .bashrc de la siguiente forma: export CRMOTHERSHIP=10.1.1.1 .

Una vez que Chromium se ha habilitado en el clúster, es posible ejecutar directamente cualquier aplicación OpenGL. Para esto, se debe configurar un puerto de ejecución para el mothership en el archivo de configuración default de Chromium, y abrir este puerto para TCP en el firewall del frontend.

Por ultimo se le debe indicar a Rocks que habilite chromium .

3 RESULTADOS

Para poder medir el rendimiento del muro instalado en el cluster Peyote, se emplearon las 3 métricas que se describen a continuación:

1.Carga de CPU.- Muestra el porcentaje de uso de los procesadores con los que cuenta el clúster. Para este caso de estudio se contó con un total de 28 cores. Dicha medición representa el poder computacional utilizado por el clúster al momento de ejecutar un programa OpenGL.

2.Carga de red local.- Como se comentó anteriormente, en el apartado de Instalación física del frontend, la comunicación entre los nodos y el frontend utilizan una conexión FastEthernet de 100Mbps. Para esta métrica se realizó la medición de la saturación de la red interna del clúster. Dicha medición se hizo en Bits/segundo.

3.Fotogramas por segundo.- Cada programa hecho con OpenGL, después de renderizar una escena, la envía como fotograma al dispositivo de despliegue. Esto genera un flujo continuo de fotogramas por segundo. El número de fotogramas determina la fluidez con la cual es representada una animación OpenGL, y se mide en FPS(frames per second).

Se usaron dos Benchmark de referencias para tomar las medidas necesarias.

El primero fue un programa escrito en lenguaje C, que usa OpenGL para desplegar un rehilete tridimensional en una animación girando. Dicha animación fue desarrollada dentro del grupo de trabajo y por tanto se tenía a acceso código fuente. Se puede observar una muestra de este programa en la figura 6.A.

El segundo programa es un tester desarrollador por [8], sobre fractales en OpenGL. Se usó este programa porque el requerimiento de computo es muy alto y es lo que se necesita para poner a prueba el rendimiento del cluster. Se puede observar una muestra de este programa en la figura 6.b.

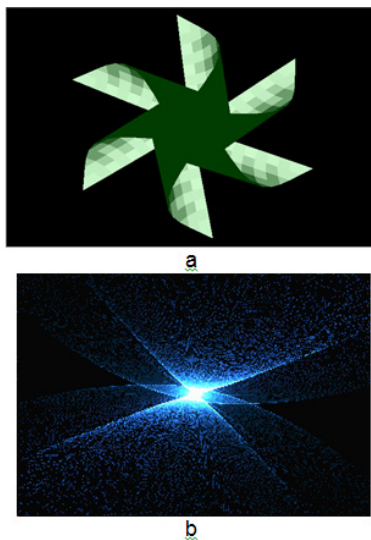


Figura 6. Muestra de los programas usados como benchmark

Ambos benchmark se ejecutaron en periodos de 10 minutos de forma continua. Al finalizar ese periodo de tiempo, el programa arrojaba el promedio FPS obtenidos en la ejecución. De cada ejecución se obtiene las métricas descritas previamente.

Se lanzaron varias configuraciones del muro de visualización, manteniendo siempre la relación de aspecto cuadrada. De esta forma el programa mantiene su proporción original evitando distorsiones significativas de la imagen.

Las pruebas se automatizaron mediante un script que lanzaba los programas y recogía datos. Se lanzaron 3 veces cada prueba con la finalidad de eliminar la variación debida a procesos en segundo plano durante

la ejecución y así obtener una medición mas confiable.

Se muestra en ejemplo en la figura 7 una imagen de la ejecución en el muro de visualización configurado para un arreglo de 3x3 monitores.



Figura 7. Ejecución benchmark rehilete en el muro

En la figura 8 se presentan los resultados obtenidos con el Benchmark rehilete usando la red de 100 mbps. Los FPS se obtuvieron directamente del propio programa de prueba, mientras que las otras 2 métricas fueron obtenidas utilizando la herramienta de monitoreo de clúster, GANGLIA, instalada en el frontend por Rocks.

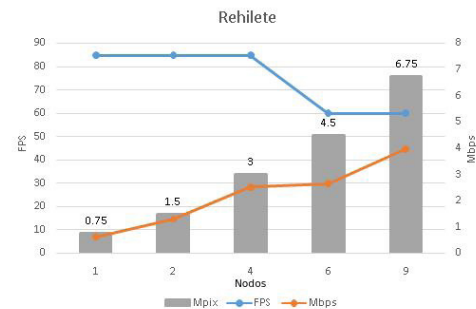


Figura 7. Resultados del Benchmark Rehilete con red de 100 mbps

En la figura 8 se muestran los resultados obtenidos por el benchmark fractales tambien con la red de 100 mbps.

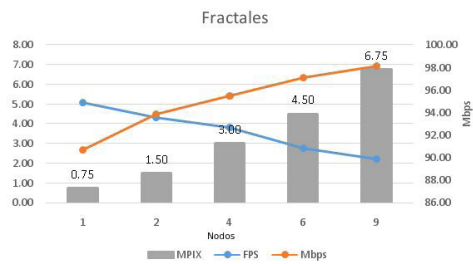


Figura 8. Resultados del Benchmark Fractal con red de 100 mbps

Los FPS miden throuput del sistema, es decir la cantidad de frames por unidad de tiempo (segundos).

Los Mbps son el ancho de banda de la red necesario para la ejecución concurrente del programa de prueba en todos los nodos de visualización.

MPiX es la resolución de la imagen presentada en el muro dada en megapíxeles.

Después de haber obtenido los anteriores resultados, se realizaron pruebas usando el switch de 1 Gbps y se lanzaron de nuevo las pruebas anteriores, obteniéndose para el rehilete los resultados mostrados en la figura 9.

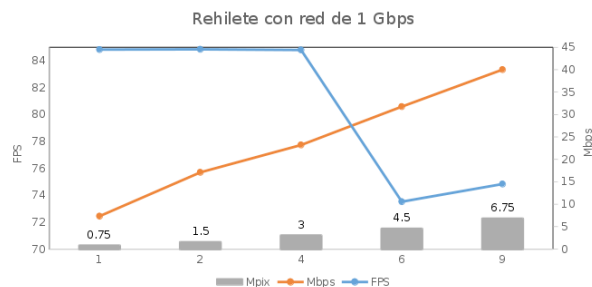


Figura 9. Resultados del Benchmark Rehilete con red de 1 Gbps

Como se puede observar, el throuputs es muy parecido al de la red con 100 Mbps, hasta 4 nodos, sin embargo en 6 y 9 si es mas alto. Por su parte el consumo de red es bastante mas alto que en el caso anterior.

Para la prueba con fractales en red de 1 Gbps se pueden observar los resultados en la figura 10.

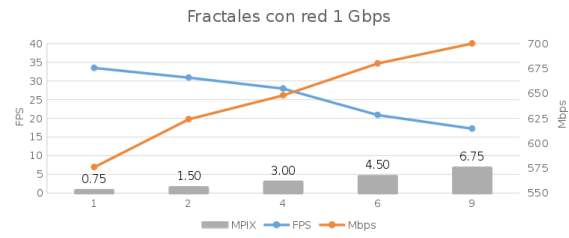


Figura 9. Resultados del Benchmark Rehilete con red de 1 Gbps

Como se puede observar, el throuputs es muy parecido al de la red con 100 Mbps, hasta 4 nodos, sin embargo en 6 y 9 si es mas alto. Por su parte el consumo de red es bastante mas alto que en el caso anterior.

Para la prueba con fractales en red de 1 Gbps se pueden observar los resultados en la figura 10.

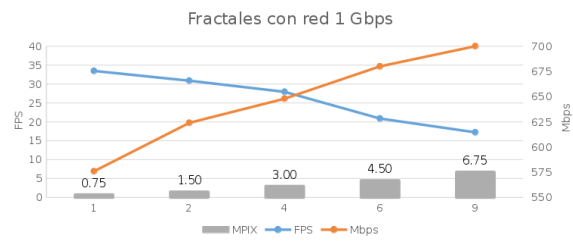


Figura 10. Resultados del Benchmark Fractal con red de 1 Gbps

En este caso, se puede observar una sustancial mejora en el throuput de casi 7 veces con respecto a 100 Mbps. El consumo del ancho de banda también sube sustancialmente, sin embargo como es mas amplio, la aplicación logra consumos hasta el 70% de la red, pero no la satura como en el caso anterior.

4. CONCLUSIONES Y TRABAJOS FUTUROS

Como se pueden observar en los resultados obtenidos para el benchmark rehilete con las pruebas en red de 100 Mbps, existe una correlación entre el ancho de banda de la red (entre 0.8 y 3.5 Mbps) y la resolución total de la imagen desplegada (0.75 a 6.75 Mpix). Lo anterior se explica debido a la forma en que trabaja chromium, ya que divide la imagen que se va a presentar entre todos los tiles del muro, enviando segmentos completos del renderizado a través de la red a cada uno. Lo anterior se traduce en incremento de tráfico conforme se incrementa la resolución del display en el muro. A pesar de tener la misma tendencia, se puede observar que a partir de 6 tiles el crecimiento del ancho de banda es menor que el crecimiento de los MPIX. Esto se explica porque la animación usada tiene principalmente monitores del centro y quedando los de la periferia vacíos o con poca carga, lo que reduce notablemente el tráfico hacia esos monitores.

En cuanto al throughput, se observa que inicia cerca de 85 fps y se mantiene más o menos constante hasta 6 monitores, teniendo una caída a 60 fps en 6 y 9 monitores, donde se estabiliza. Esta caída se debe principalmente a la saturación de instancias (threads para atención a los gráficos) en el frontend. El frontend es una maquina con 4 cores de 64bits y si tiene en ejecución 4 o 6 instancias gráficas, se está en el límite de la capacidad máxima del procesador. Por ello en ambientes concurrentes es recomendable que se tengan entre 1 y 1.5 procesos intensivos (que consuman mas del 80% del procesador cada uno) por core. Con 6 nodos se tienen ya 1.6 y con 9 se tienen ya 2.25. Esta saturación de procesos en el frontend ocasiona un menor rendimiento del programa de prueba, que explica principalmente la reducción en FPS.

Los resultados en el benchmark fractal son ligeramente diferentes debido a que la aplicación tiene una alta demanda de poder de computo. Se ven un incremento sustancial en el tráfico de la red que inicia en 91 % de ancho total para 1 nodos y termina en 97% de la capacidad total de la red. Hay que recordar que el cluster tiene su red aislada de las demás redes, lo que quiere decir que el recurso esta totalmente dedicado al cluster, siendo efectiva la ocupación de la red. El tráfico tan alto se debe al alto nivel de detalle de las imágenes fractales que se replican en todo el espacio visual, lo que hace necesario transferir por la red una gran cantidad de información. Por otra parte el throughput se ve limitado por las exigencias de computo de la aplicación

comenzando en 5 fps para 1 nodo y bajando al rededor de 2 fps con 9 nodos. Esta caída se da principalmente por el freno que produce la saturación de la red local, mencionada inicialmente.

Los resultados con la red de 1 Gbps son muy similares para el benchmark de rehilete, ya que este no exige demasiados recursos de computo, pero en fractales si se nota una mejora bastante sustancial, debido a la amplitud del ancho de banda en la red de datos.

En resumen poder decir:

- Al incrementar la cantidad de nodos de visualización incrementa el tráfico en la red local.

- Al incrementar la resolución total del muro de visualización, se reduce el rendimiento de fotogramas por segundo

- Se encuentra una relación normalizada inversamente proporcional entre el rendimiento (fps) y el tráfico de red interna.

- En cuanto a la carga CPU en los nodos tile, no es significativa ya que no pasaba del 15% del uso total.

- La red es un factor significativo en el rendimiento. Con una red de baja capacidad el overhead que ocasiona hace caer significativamente el throuput, pero con una mejor red, el overhead disminuye y entonces el throuput mejora.

Con todos estos resultados se planean como trabajos futuros correr las mismas pruebas con otros renderizadores paralelos mas optimos en cuando a balance de carga como Equalizer y ademas usar Benchmarks mas costosos computacionalmente hablando. Se buscara incrementar tambien el numero de monitores a 16 para poder hacer pruebas de 4x4.

REFERENCIAS

1. Rob Jesús Alberto Verduzco Ramírez , Nicandro Farías Mendoza , Gilberto René Martínez Bonilla , Pedro Rocha Medrano , María Isabel Sáenz Rodríguez . "Infraestructuras de bajo costo para la visualización de datos a gran escala y resolución, una opción accesible para instituciones educativas ". Revista Iberoamericana de las Ciencias Computacionales e Informática . Vol. 3. No. 5. ISSN: 2007-9915 . Junio 2014.
2. [2]. TACC. Universidad de Texas. "Stallion user guide". Octubre 2011. Recuperado en Enero de 2015 de <https://www.tacc.utexas.edu/user-services/user-guides/stallion-user-guide> .
3. Universidad of California. "Base Roll: User Guide: Version 5.3", Edition Diciembre 2009. Rocks Cluster. Recuperado en Julio 2015 de <http://www.rocksclusters.org/roll-documentation/base/5.3/roll-base-usersguide.pdf>.
4. Universidad of California. "Viz Roll: User Guide: Version 5.3". Rocks Cluster. Edition Diciembre 2009. Recuperado en Julio 2015 de <http://www.rocksclusters.org/roll-documentation/viz/5.3/roll-viz-usersguide.pdf>.
5. Dana Plepys, Luc Renambot. "SAGETM (Scalable Adaptive Graphics Environment)". Electronic Visualization Laboratory, University of Illinois at Chicago . September 2013 . Recuperado en Julio 2015 de https://www.evl.uic.edu/documents/sagedocumentation_9-27-2013.pdf ..
6. Greg Humphreys, et all. "Chromium: a stream-processing framework for interactive rendering on clusters". ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2002. Volume 21 Issue 3, July 2002 Pages 693-702. Doi. 10.1145/566654.566639
7. Dave Shreiner, The Khronos OpenGL ARB Working Group, Bill Licea-Kane, Graham Seller. "OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.1". 8th Edition. Addison-Wesley Professional, Massachusetts, 2011/10/13. ISBN-13: 978-0321773036
8. Nick King. "OpenGL Programming". Recuperado en Junio 2015 de <http://opengl2.sourceforge.net/>..

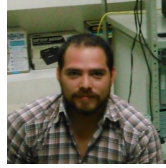
Acerca de los autores



Abelardo Rodríguez León es Ingeniero en Sistemas Computacionales por el Instituto Tecnológico de Veracruz (1990), Maestro en Ciencias Computacionales por la Universidad Veracruzana (1996) y Doctor en Ciencias Computacionales por la Universidad Politécnica de Valencia, España (2007). Actualmente es profesor investigador en el Departamento de Computación y Sistemas del Instituto Tecnológico de Veracruz. Sus áreas de interés incluyen: la computación de alto rendimiento, paralelismo y grid, además de estudios de modelos gráficos en 3D.



Guillermo Efrén Ovando Chacón, es Ingeniero Mecánico, por el Instituto Tecnológico de Tuxtla Gutiérrez, Chiapas, México, Maestro en Ciencias en Ingeniería Mecánica, del Instituto Tecnológico de Veracruz, México y Doctor en Ciencias de la Ingeniería, en la UNAM, México. Actualmente es Profesor de Ingeniería Mecánica (2006-presente) en el Departamento de Ingeniería Mecánica del Instituto Tecnológico de Veracruz. Es miembro del Sistema Nacional de Investigadores, México, 2007-2016 y Profesor con Perfil PRODEP (Secretaría de Educación Pública, México), 2006-2018. Es miembro de la Sociedad Mexicana de Física y de la Sociedad Mexicana de Ingeniería Mecánica (SOMIM), México. Sus áreas de interés incluyen: dinámica de fluidos computacional, mecánica de fluidos y transferencia de calor. Tiene más de 10 publicaciones indizadas.



Marcos Antonio Gomez Abascal es pasante de la carrera de Ingeniería en Sistemas Computacionales del Instituto Tecnológico de Veracruz (2015). Actualmente se desempeña como desarrollador independiente. Sus áreas de interés incluyen: diseño y desarrollo de aplicaciones web, diseño e implementación de sistemas de comunicaciones y programación en Java.



Juan Carlos Prince Avelino es Ingeniero Mecánico, por el Instituto Tecnológico de Veracruz, Veracruz, México (1985), Maestro en Ciencias en Ingeniería Mecánica, del Instituto Politécnico Nacional, México (1988) y Doctor en Ciencias de la Ingeniería, en la Universidad de Cambridge, UK (1995). Actualmente es Profesor de Ingeniería Mecánica (1995-presente) en el Departamento de Ingeniería Mecánica del Instituto Tecnológico de Veracruz. Ha sido Profesor Visitante en la Universidad de California, San Diego, en 2009-10 y en 2014-15. Es miembro del Sistema Nacional de Investigadores, México, 1997-2020 y Profesor con Perfil PRODEP (Secretaría de Educación Pública, México), 2006-2018. Es miembro del Combustion Institute y de la Sociedad Mexicana de Ingeniería Mecánica (SOMIM), México. Sus áreas de interés incluyen: dinámica de fluidos computacional, teoría de flamas, fenómenos de ignición y combustión catalítica. Tiene más de 30 publicaciones indizadas, incluyendo Editor de "Memorias del 8° Congreso Latinoamericano de Transferencia de Calor y Masa", 2001