Modelos para el Análisis de Pérdidas de AFR en Sistema Paralelo de Mejor Esfuerzo

Cristian F. Perez-Monte[1,2][0000−0003−4407−7811], Pablo Ignacio Ocampo[1][0000−0003−1031−3513], Enzo Raúl Crespillo[1][0000−0003−0167−6734], Gabriel Andrés Caballero[1][0000−0001−6388−6628], Mariana Lorena Cantón[1][0000−0002−5643−5375], María Fabiana Piccoli[2][0000−0002−3636−7360], Cristian Javier Luciano[3][0000−0001−6521−5182], and Silvio Rizzi[4][0000−0002−3804−2471]

[1] Laboratorio de Visualización Paralela (LVP) - Gridtics - Universidad Tecnológica Nacional Regional Mendoza, Ciudad de Mendoza M5502AJE, Argentina
cristian.perez,ocampo,ecrespillo,gcaballero,mcanton@gridtics.frm.utn.edu.ar
[2] LIDIC - Universidad Nacional de San Luis, Ciudad de San Luis D5700HHW, Argentina mpiccoli@unsl.edu.ar
[3] Departments of Bioengineering, Biomedical & Health Information Sciences, and Medical Education, University of Illinois at Chicago, Chicago, IL 60607, USA
clucia1@uic.edu
[4] Leadership Computing Facility, Argonne National Laboratory, Lemont, IL 60439, USA
srizzi@anl.gov

Resumen En este trabajo se presentan dos modelos de aproximación para la determinación de la relación de pérdidas de cuadros de la técnica de paralelización *Alternate Frame Rendering* en un sistema de renderización de mejor esfuerzo. El primero de los modelos considera que la velocidad de cuadros por nodo es constante en el tiempo, mientras que en el segundo, la velocidad de cuadros por nodo varía en el tiempo pero, la velocidad de cuadros promedio es constante entre todos los nodos. Finalmente, se presentan resultados de desempeño teórico simulados hasta 512 nodos y se concluye que, para ambos modelos, el sistema posee una buen desempeño para menos de 16 nodos.

Keywords: AFR · Volume Rendering · PD-Rend · Best Effort Computing.

1. Introducción y Trabajos Relacionados

Un modelo 3D es una representación de un objeto, el cual puede ser descripto mediante un volumen compuesto de voxels. La medicina moderna trabaja con modelos 3D, los cuales se obtienen a través de la tomografía computada (CT) o imágenes por resonancia magnética (MRI). Para la visualización por computadora de estos modelos se debe realizar un procesamiento gráfico denominado renderización. Existen aplicaciones tales como el entrenamiento médico y las

prácticas pre-quirúrgicas donde se requiere la renderización de volúmenes con extremo nivel de realismo. Por ello el proceso debe cumplir con dos características fundamentales: el foto-realismo y el cine-realismo [18]. Tanto el foto-realismo como el cine-realismo demandan gran capacidad de cómputo, no sólo para obtener imágenes quasi-reales (foto-realismo), sino también en tiempo real y con fluidez temporal (cine-realismo).

Las técnicas de computación paralela y distribuida permiten desarrollar un sistema de renderización de volumen escalable con un buen rendimiento, el cual produce resultados foto y cine realistas. De esta forma, los datos se pueden distribuir en píxeles, objetos 3D o cuadros temporales. La taxonomía de Molnar es aplicable a píxeles y objetos 3D [13], mientras que para cuadros completos se utiliza renderización de cuadros alternativos (AFR) [10].

En este caso particular, el enfoque de la problemática está aplicado a la utilización de la técnica de distribución AFR en un sistema de renderización de volúmenes en tiempo real y de mejor esfuerzo computacional denominado *PD-Rend*. Una característica clave en AFR es que procesa múltiples cuadros simultáneamente usando múltiples nodos GPU / CPU. Cada *Nodo Procesador* procesa la escena completa para diferentes cuadros temporales, seguido de una clasificación de cuadros en la secuencia correcta para la visualización. Aunque la latencia es inevitable entre la etapa de renderizado y la visualización final [14], AFR presenta una escalabilidad óptima y se usa comúnmente en entornos con múltiples GPU alojadas en una sola computadora [3]. Además, el método se puede implementar en la mayoría de las técnicas de iluminación volumétrica[7], como las técnicas basadas en la región local [11] , isosuperficies [1], rebanadas [20], funciones base [9], enrejados [19], iluminación espacial [6] y trazado de rayos [8]. AFR también está integrado en el framework Equalizer [4].

En este estudio se presentan dos modelos aproximados de análisis para la determinación de la relación de pérdidas de cuadros y para la obtención de la eficiencia en la utilización de *PD-Rend* en la resolución de la problemática antes expuesta. En publicaciones previas de los autores se ha trabajado principalmente con el primero de los modelos, como allí se detalla ampliamente la fundamentación teórica de la relación de pérdidas de cuadros global, en éste introducimos el estudio del segundo modelo y realizamos una comparación entre ambos.

El trabajo está organizado de la siguiente manera. En la sección 2 se presenta el sistema de renderización utilizado, en la sección 3 se proponen los modelos de análisis que son utilizados, en la sección 4 se presentan los resultados, y finalmente en la sección 5 se detallan los alcances de cada modelo de análisis, desempeño del sistema y trabajos futuros.

2. PD-Rend: Sistema de Renderización de mejor esfuerzo

El trabajo es analizado sobre *PD-Rend*, un sistema de renderización de volúmenes paralelo y heterogéneo, el cual está compuesto de m nodos procesadores que trabajan paralelamente para renderizar una escena. En la Figura 1 se presenta un esquema del sistema utilizado. Si bien el presente trabajo está

aplicado a su utilización con AFR, demostrado en [17], también es posible utilizar otras técnicas de paralelización tal como 2DR [16] y MFR[18].

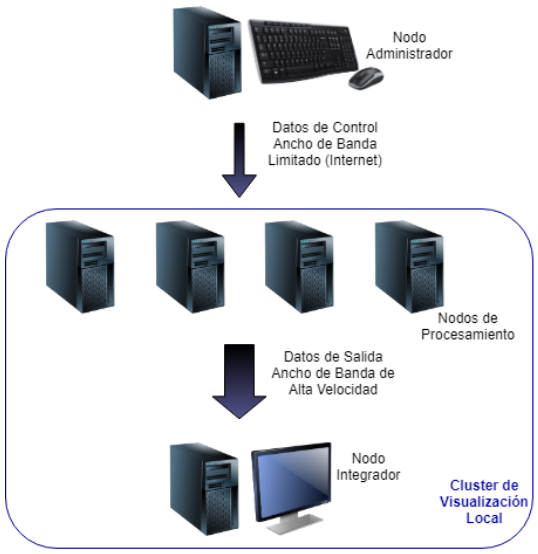

Figura 1. Sistema Pd-Rend.

Como parte de sus parámetros de entrada, el *Nodo Administrador* recopila la ubicación del espectador, la cámara y las luces en la escena virtual. Esta información se envía, junto a un número de secuencia de tarea, a los *Nodos Procesadores* por cada período de tiempo T_a. Mientras que cada *Nodo Procesador* está ocupado renderizando un cuadro, también escucha al *Nodo Administrador* y almacena las tareas recibidas. Una vez que cada *Nodo Procesador* termina de procesar un cuadro y envía los resultados junto al número de secuencia al *Nodo Integrador*, recupera la tarea recibida más reciente y descarta todas las demás tareas. Por lo tanto, la siguiente tarea puede comenzar inmediatamente sin tener que esperar un período de T_a para recibir otra tarea. El *Nodo Integrador* recibe todos los cuadros representados por los *Nodos Procesadores* y crea la animación final.

El *Nodo Administrador* difunde una lista detallada de tareas temporales a los *Nodos Procesadores*. Tan pronto como cada *Nodo Procesador* finaliza una tarea, selecciona otra tarea de la lista. Este es un enfoque de mejor esfuerzo, ya que no todas las tareas pueden ser procesadas por los *Nodos Procesadores* disponibles. Por lo tanto, si se incrementa el número de *Nodos Procesadores*, la resolución temporal de la animación también aumenta.

Otra característica importante de *PD-Rend* es la ausencia de barreras de sincronización entre los *Nodos Procesadores*. Cada *Nodo Procesador* hace su mejor esfuerzo para renderizar tantos cuadros como sea posible. Debido a este

enfoque de mejor esfuerzo, *PD-Rend* puede ejecutarse en entornos heterogéneos, combinando CPU y GPU de diferentes generaciones de hardware.

Esto permite crear clústeres de visualización utilizando el hardware disponible y lograr un rendimiento razonable.

La Figura 2 muestra una renderización de un conjunto de datos de tomografía computarizada médica con trazado de rayos de Monte Carlo. *PD-Rend* produce imágenes fotorrealistas de alta calidad utilizando el trazado de rayos de Monte Carlo con una versión modificada de Exposure Renderer [8]. Como se explicó anteriormente, cada *Nodo Procesador* aplica n iteraciones de Monte Carlo por cuadro renderizado [16] y envía los cuadros completos a través de la red.

Figura 2. Ejemplo de Imagen renderizada del Volumen Manix [21] con técnica de renderización basada en ray-tracing [8].

AFR permite crear animaciones suaves cuando la cámara o el espectador cambian sus posiciones. Además, si la cámara no se mueve, los *Nodos Procesadores* tienen la posibilidad de completar más iteraciones de Monte Carlo mediante la técnica MFR (Monte Carlo Frame Rendering)[18], lo que aumenta la relación señal-ruido determinada por $m * n$ iteraciones de Monte Carlo en el mismo cuadro de la escena. Más allá de AFR y MFR, *PD-Rend* también puede usar otras técnicas paralelas simultáneamente.

3. Modelos de Análisis para las pérdidas de procesamiento con AFR

Cada *Nodo Procesador* le toma un período de tiempo T_p para renderizar un cuadro. T_p está determinado por dos factores: el poder computacional y las características de la escena. En consecuencia, T_p generalmente varía de nodo a nodo y de cuadro a cuadro renderizado en el tiempo.

Cada *Nodo Procesador*, apenas se desocupa de una tarea anterior, realiza una tarea de las recibidas del *Nodo Administrador*. Al finalizarla, envía la tarea procesada (en este caso un cuadro renderizado) al *Nodo Integrador* junto a un número de secuencia de la tarea obtenido desde el *Nodo Administrador*. La comunicación entre el *Nodo Procesador* y el *Nodo Integrador* requiere de mucho ancho de banda y genera un tiempo adicional T_c. A diferencia de T_p , T_c es constante y depende de la resolución del cuadro y del ancho de banda de red. Por todo ello, el tiempo total de renderizado de cada *Nodo Procesador* de *PD-Rend* puede ser expresado de la siguiente manera:

$$T_t = T_p + T_c \,. \tag{1}$$

Sin embargo, este tiempo de transmisión se solapa con el período de tiempo de procesamiento del siguiente cuadro (el *Nodo Procesador* procesa un nuevo cuadro mientras está transmitiendo el anterior hacia el *Nodo Integrador*). Por lo tanto, T_t puede ser reformulada considerando únicamente T_p (T_c es despreciada), tal como indica la ecuación 2.

$$T_t \approx T_p \,. \tag{2}$$

Dado que T_t es diferente entre *Nodos Procesadores*, existe la posibilidad que el *Nodo Integrador* reciba cuadros fuera de secuencia desde los *Nodos Procesadores*. Para una correcta visualización, estos cuadros deben ser descartados. Si se quisiera evitar descartar dichos cuadros, una posible solución será almacenar los cuadros recibidos en un periodo de tiempo determinado y ordenarlos para visualizarlos en el orden correcto. Por desgracia, dicha solución incrementa la latencia del sistema [5] [2], lo cual es indeseable en aplicaciones tales como realidad virtual [12]. De esta forma para conseguir una mínima latencia, *PD-Rend* descarta todos los cuadros fuera de secuencia. Como resultado de esto, hay una pérdida de procesamiento. Determinar esta pérdida y la relación de pérdida con respecto al total de cómputo procesado es de vital importancia para estimar la eficiencia y speedup del sistema [15].

Dada la complejidad de obtención de una solución para la determinación de las pérdidas se han planteado dos modelos aproximados para estimarlas en un sistema real. Estos modelos se denominan *Régimen Estático* y *Régimen Dinámico* y se detallan a continuación.

3.1. Modelo de Régimen Estático

En el modelo estático consideramos la aproximación de que el período de renderizado de cada *Nodo Procesador* no varía en el tiempo. Cada *Nodo Procesador* tiene un período de renderización constante en el tiempo, diferente del resto y comprendido dentro de un rango de variación.

Considerando de que hay m *Nodos Procesadores*, es posible determinar las pérdidas teniendo en cuenta la aproximación de que cada *Nodo Procesador* tiene un período T_t diferente pero constante en el tiempo. De esta manera, es posible ordenarlos en forma creciente:

$$T_1 < T_2 < ... < T_{m-1} < T_m \,. \tag{3}$$

A partir de ello, se determinó en [17] que es posible obtener la relación de pérdidas de cuadros global R_t, para la aproximación de régimen estático, con la siguiente ecuación:

$$R_t = \frac{\sum_{x=1}^{x=m} (\frac{1}{T_x}(1 - \frac{\prod_{f=1}^{f=x-1} T_f}{T_x^{x-1}}))}{\sum_{x=1}^{x=m} \frac{1}{T_x}} = 1 - \frac{\sum_{x=1}^{x=m} (\frac{\prod_{f=1}^{f=x-1} T_f}{T_x^x})}{\sum_{x=1}^{x=m} \frac{1}{T_x}} \,. \tag{4}$$

El estudio de la determinación de distribución de T_x que hacen máxima a R_t es de especial importancia para la obtención de la eficiencia mínima del sistema.

Si bien posee una complejidad elevada del que aún no se han obtenido soluciones analíticas exactas, pudo ser resuelto mediante Monte Carlo mediante el muestreo aleatorio para la búsqueda de distribución de T_x en el rango entre T_1 y T_m que hace máximo a R_t. La utilización de Monte Carlo por aproximaciones sucesivas incrementó la eficiencia de la búsqueda de la distribución máxima [17]. La R_t media también pueden obtenerse por Monte Carlo promediando una cantidad considerable de muestras, utilizando el mismo rango anterior. Para la obtención de R_t máxima y R_t media se utilizó, para la caracterización de la distribución de los T_xs, un parámetro denominado índice de invariabilidad z [17]. El mismo está determinado por la ecuación 5 y define el rango de variación de los T_xs.

$$z = \frac{T_1}{T_m} \,. \tag{5}$$

3.2. Modelo de Régimen Dinámico

En el modelo dinámico se considera la aproximación de que el tiempo de renderizado T_x medio es igual entre *Nodos Procesadores*. Además el rango de variación en el tiempo es el mismo para todos los *Nodos Procesadores*, definido por z y comprendido entre T_1 y T_m. Cada *Nodo Procesador* varía, en el tiempo, su T_x instantánea independientemente del resto, pero todos ellos dentro de un rango de variación dado también por z.

Para el caso de la obtención de una solución analítica del valor máximo de R_t para Régimen Dinámico se requiere un análisis de elevada complejidad, incluso mayor al Régimen Estático, lo cual excede el alcance de este trabajo y está en proceso de investigación. Igualmente, como en el caso de Régimen Estático es posible determinar una solución aproximada en base a aproximaciones con Monte Carlo. Sin embargo, debido al gran número de grados de libertad y el procesamiento necesario para determinar soluciones con un error aceptablemente pequeño, aún está en proceso de determinación. En el siguiente trabajo se determina una cota máxima, la cual es fundamentalmente superior al máximo de régimen dinámico y que, al usar un modelo de *Nodos Procesadores* homogéneos, puede determinarse fácilmente. Este caso hipotético representa la peor condición, para la cual arriban cuadros de todos los *Nodos Procesadores*, pero con una distribución tal que, al variar en el tiempo, permite que los cuadros de un solo *Nodo Procesador* sean renderizados y el resto descartados.

Así, la relación de pérdidas máxima para régimen dinámico será por ende igual o inferior a:

$$R_{t\,max-dyn} \leq 1 - \frac{1}{m} \,. \tag{6}$$

Si bien este caso puede resultar en pérdidas muy elevadas cuando el número de *Nodos Procesadores* m aumenta (produciendo un sistema muy ineficiente) la distribución temporal que determina esta condición es poco probable. De esta forma, en términos prácticos, lo realmente importante es la determinación de la R_t media para régimen dinámico. La variabilidad temporal por cada *Nodo*

Procesador determina gran cantidad de grados de libertad. La R_t media para régimen dinámico, a diferencia de régimen estático, no puede ser determinada fácilmente usando Monte Carlo dada la gran cantidad de grados de libertad. Por ello, la resolución se realiza con simulaciones matemáticas en tiempos de ejecución lo suficientemente largos.

4. Resultados

Dado que el análisis requiere de una cantidad de nodos muy superior a la disponible, se han realizado simulaciones matemáticas para la obtención de los resultados, con las consideraciones de cada caso anteriormente expuestas. Dichas simulaciones han sido validadas con los resultados de sistemas reales con pocos nodos.

En dichas simulaciones se ha estimado la relación de pérdida de cuadros descartados por fuera de secuencia con respecto a la cantidad de cuadros total generado por el sistema, considerando variaciones estocásticas en los tiempos de procesamiento. Para el caso de régimen estático se asignan períodos constantes en el tiempo por cada nodo y dentro del rango estipulado por z. Por el contrario, para régimen dinámico, se asignan períodos variables en el tiempo pero también acotados por el rango estipulado por z.

Los resultados de régimen estático para pérdidas de cuadros máximas y medias pueden observarse en la Figura 3 mientras que los resultados para régimen dinámico se muestran en la Figura 4.

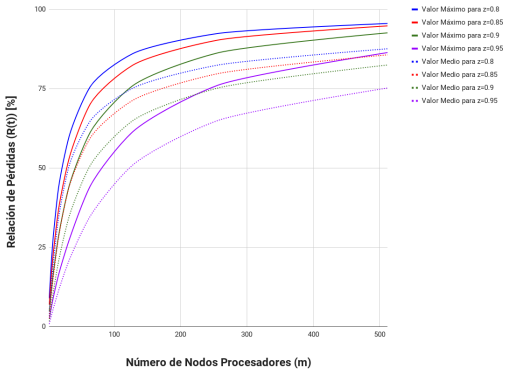

Figura 3. Relación de Pérdidas de cuadros Máxima y Media para Régimen Estático.

Como es posible observar en la Figura 5, el error en la determinación del valor medio de R_t mediante la aproximación de régimen dinámico y estático es pequeño, levemente superior en la estimación para régimen dinámico y en cualquier caso inferior al 3 %.

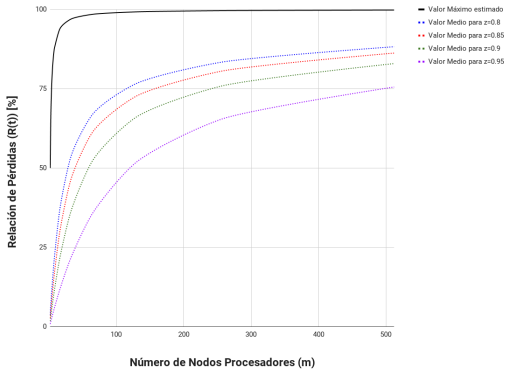

Figura 4. Relación de Pérdidas de cuadros Máxima y Media para Régimen Dinámico.

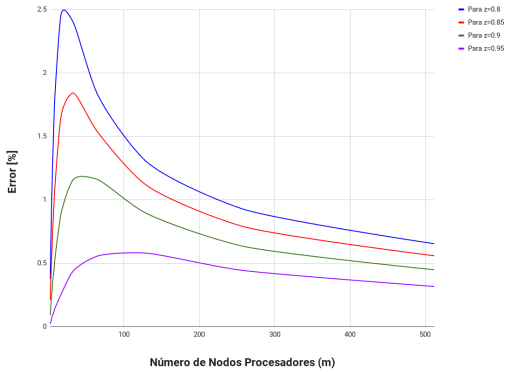

Figura 5. Diferencia entre la estimación de pérdida media de Régimen Dinámico y Estático.

5. Conclusiones y Trabajos Futuros

En este trabajo se utilizó el sistema *PD-Rend* aplicando la técnica de paralelización AFR a fin de observar el comportamiento del sistema para el modelo de régimen estático y dinámico. De los resultados obtenidos, se observa que, para ambos casos, R_t se encuentra limitada y permite la utilización del sistema para una cantidad de *Nodos Procesadores* m menor o igual a 16, permitiendo así una relación de pérdidas menor al 50 %. Si bien ambos modelos expuestos pueden utilizarse para la determinación de la eficiencia global del sistema, cada uno de ellos son apropiados para modos de funcionamiento específicos, ellos son:

- Modelo de Régimen Estático: Es apropiado para la determinación de la R_t debida a las diferencias de capacidad de cómputo entre los diferentes *Nodos Procesadores*. Esto resulta de importancia en un sistema que está planteado

para ser heterogéneo permitiendo el uso de *Nodos Procesadores* con diferente arquitectura y capacidad computacional.

- Modelo de Régimen Dinámico: Es apropiado para la determinación de la R_t ocasionada por las diferencias de tiempo de renderizado debidas a las características cambiantes de la escena, pero considerando la aproximación de que todos los *Nodos Procesadores* son homogéneos.

En un futuro se plantea la realización de un modelo global exacto mediante la utilización de los dos modelos aproximados de régimen estático y régimen dinámico para una determinación analítica exacta de la R_t.

Además, actualmente se están desarrollando métodos de compensación para reducir la R_t y, por ende, incrementar la eficiencia del sistema de forma de permitir a *PD-Rend* funcionar con una eficiencia aceptable para $m \geq 16$.

6. Agradecimientos

Este proyecto ha sido posible gracias al aporte de recursos humanos brindados por LVP-Gridtics perteneciente a Universidad Tecnológica Nacional Regional Mendoza (UTN-FRM), Universidad Nacional de San Luis (UNSL), University of Illinois at Chicago (UIC) y Argonne National Laboratory (ANL). Los resultados obtenidos del estudio de régimen estático han sido realizados en una tesis doctoral soportada económicamente por UTN y UNSL. Los trabajos actuales en régimen dinámico están siendo desarrollados parcialmente gracias al aporte económico de Conicet y UTN.

Referencias

1. Banks, D., Beason, K.: Decoupling illumination from isosurface generation using 4d light transport. IEEE Trans. Vis. Comput. Graph. **15**(6), 1595–1602 (2009)
2. Blade, R., Padgett, M.: Virtual environments standards and terminology. Handbook of virtual environments pp. 15–27 (2002)
3. Diard, F., Young, W., Johnson, P.: Sequencing of displayed images for alternate frame rendering in a multi-processor graphics system (Jun 9 2009), uS Patent 7,545,380
4. Graphics, E.: Equalizer. http://www.equalizergraphics.com/ (2016), eyescale GmbH
5. Hale, K., Stanney, K.: Handbook of virtual environments: design, implementation, and applications. CRC Press (2014)
6. Jönsson, D., Kronander, J., Ropinski, T., Ynnerman, A.: Historygrams: Enabling Interactive Global Illumination in Direct Volume Rendering using Photon Mapping. IEEE Transactions on Visualization and Computer Graphics (TVCG) **18**(12), 2364–2371 (2012)
7. Jonsson, D., Sundan, E., Ynnerman, A., Ropinski, T.: A survey of volumetric illumination techniques for interactive volume rendering. Computer Graphics Forum **33**(1), 27–51 (2014). https://doi.org/10.1111/cgf.12252, https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12252

8. Kroes, T., Post, F., Botha, C.: Exposure render: An interactive photo-realistic volume rendering framework. PLoS ONE **7**, e38586 (07 2012)
9. Kronander, J., Jonsson, D., Low, J., Ljung, P., Ynnerman, A., Unger, J.: Efficient visibility encoding for dynamic illumination in direct volume rendering. Visualization and Computer Graphics, IEEE Transactions on **18**(3), 447–462 (March 2012). https://doi.org/10.1109/TVCG.2011.35
10. Laksono, I., Aleksic, M.: Multiple device frame synchronization method and apparatus (jun 2004), uS Patent 6,747,654
11. Levoy, M.: Display of surfaces from volume data. IEEE Comput. Graph. Appl. **8**(3), 29–37 (May 1988). https://doi.org/10.1109/38.511, http://dx.doi.org/10.1109/38.511
12. Meehan, M., Razzaque, S., Whitton, M., Jr., F.B.: Effect of latency on presence in stressful virtual environments. In: Proceedings of the IEEE Virtual Reality 2003. pp. 141–148. VR '03, IEEE Computer Society, Washington, DC, USA (2003)
13. Molnar, S., Cox, M., Ellsworth, D., Fuchs, H.: A sorting classification of parallel rendering. IEEE Computer Graphics and Applications **14**, 23–32 (1994)
14. Monfort, J.R., Grossman, M.: Scaling of 3d game engine workloads on modern multi-gpu systems. In: Proceedings of the Conference on High Performance Graphics 2009. pp. 37–46. HPG '09, ACM, New York, NY, USA (2009)
15. Pacheco, P.: An Introduction to Parallel Programming. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edn. (2011)
16. Perez-Monte, C.F., Piccoli, F., Luciano, C., Rizzi, S., Bianchini, G., Scutari, P.C.: Estimation of volume rendering efficiency with {GPU} in a parallel distributed environment. Procedia Computer Science **18**(0), 1402 – 1411 (2013), 2013 International Conference on Computational Science
17. Perez-Monte, C.F., Perez, M.D., Rizzi, S., Piccoli, F., Luciano, C.: Modelling frame losses in a parallel alternate frame rendering system with a computational best-effort scheme. Computers & Graphics **60**, 76–82 (2016)
18. Perez-Monte, C.F., Farias, F., Rizzi, S., Luciano, C., Piccoli, M.F.: Speedup in parallel rendering system with computational best-effort scheme. IEEE Latin America Transactions **16**(3), 987–995 (2018)
19. Schlegel, P., Makhinya, M., Pajarola, R.: Extinction-based shading and illumination in GPU volume ray-casting. IEEE Transactions on Visualization and Computer Graphics **17**(12), 1795–1802 (December 2011)
20. Soltészová, V., Patel, D., Bruckner, S., Viola, I.: A multidirectional occlusion shading model for direct volume rendering. Computer Graphics Forum **29**(3), 883–891 (Jun 2010)
21. Viewer, O.: Osirix imaging software (last rev may 2016) (apr 2016), http://www.osirix-viewer.com/datasets/