Paralelismo en Visualización Volumétrica

Anita Hung*
emsca!usb!hungleo@sun.com
Centro de Investigación y Soporte Tecnológico de
Petróleos de Venezuela, S.A,
Apartado 76343, Caracas Venezuela 1070

Alejandro Teruel
emsca!usb!teruel@sun.com
Departamento de Computación y Tecnología de la Información
Universidad Simón Bolívar
Apartado 89000, Caracas 1080-A, Venezuela

Resumen

Se estudia la paralelización de un conocido algoritmo de visualización volumétrica sobre una arquitectura MIMD mediante dos esquemas de descomposición de dominios: espacio del objeto y espacio de la imagen. La meta es evaluar el desempeño de cada uno utilizando un esquema de asignación estática de carga de trabajo. Las pruebas fueron realizadas en una plataforma distribuida de estaciones de trabajo SUN, donde la disponibilidad de recursos es exclusiva. Los resultados revelaron que el primer método aventaja considerablemente en efectividad al segundo e indican que en ambos esquemas, factores como el criterio de balance de carga de trabajo y operaciones de manejo de estructuras especiales provenientes de la paralelización, tienen mayor peso que el "overhead" de comunicación.

Keywords: Paralelismo, MIMD, Visualización Volumétrica

1 Introducción

El algoritmo de visualización implementado tiene como base el expuesto por M. Levoy en [10]. Dicho algoritmo utiliza una simplificación del método tradicional de trazado de rayos, en el sentido que obvía los cálculos recursivos debido a las bifurcaciones del rayo por reflexión

*Actualmente en IMG Consultores

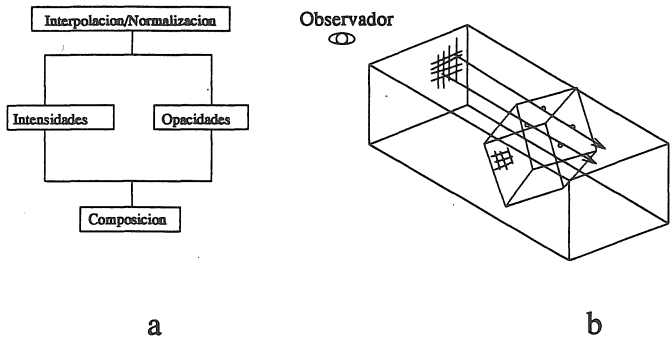

Fig. 1: Método secuencial.

y refracción. Los objetos constituyen los elementos (voxel) de una malla tri-dimensional regular.

Para aplicaciones en el campo de la geofísica, donde el tamaño de los volúmenes a visualizar están en el rango de varios gigabytes, el generar una imagen puede tomar horas, si no días. Frente a esta problemática, el procesamiento paralelo de alto nivel, constituye una solución inmediata. Las idea es distribuir los datos entre los diferentes procesadores procurando que el "overhead" proveniente de las comunicaciones sea mínima.

Algunos trabajos representativos en el área lo constituyen [11], [5], [8] y [2]. Una discusión más completa de los antecedentes se encuentra en [7].

2 Método Secuencial

La figura 1-a resume las principales tareas del método desarrollado. Previo a la visualización del volumen, se aplican una serie de preprocesamientos: a) Interpolación/Normalización de los datos mediante una sencilla interpolación lineal; b) Cálculo de Intensidades-a cada elemento volumétrico se le calcula el color mediante el modelo de Phong [6] c) Cálculo de Opacidades.

La generación de la imagen bi-dimensional se realiza en su totalidad en el módulo de composición. El cálculo del pixel (x_i, y_i) del plano de visualización comienza trazando un rayo, desde el observador, pasando por el pixel en cuestión hacia el volumen (figura 1-b).

Como optimización hemos colocado una condición en donde el rayo detiene su recorrido dentro del volumen cuando la opacidad acumulada sobrepase a uno. El valor final se obtiene de sucesivas composiciones de intensidades y colores de los voxeles que intersecta el rayo. Por otro lado, para reducir los problemas de "aliasing", a cada valor de intensidad y opacidad de dichos voxeles se le aplica un proceso de remuestreo, es decir, se promedia con los respectivos valores de los ocho vecinos más cercanos.

Con el objeto de escoger la metodología de paralelización más conveniente, se realizó un análisis detallado del comportamiento de la versión secuencial. El mismo concluyó que el mayor porcentaje de tiempo en una sola rutina era de alrededor de 50%. Según la llamada "Ley de Amdahl" [1], al aplicar a esta rutina por ejemplo 16 procesadores, no se obtendría un aceleramiento mayor de 2. Tomando en cuenta esta observación y las experiencias de paralelización en otros trabajos como [3] se decidió paralelizar la totalidad del método, lo cual implica distribuir el volumen entre los procesadores.

3 Descomposición del espacio del objeto (De0)

La partición del volumen se realiza seleccionando dos ejes del sistema de coordenadas del objeto, que determinan las directrices u orientación de los planos de corte del volumen. Esta partición origina subvolúmenes que serán posteriormente repartidos entre los procesadores. Para lograr balance de carga, autores como Kobayashi *et al.* y Goldsmith [9], [13], han adoptado un método de asignación de subvolúmenes llamado descomposición por dispersión (*scattering decomposition*) utilizando formatos F. La figura 2-a muestra un ejemplo con $F = 4$ y $P = 9$. En nuestro caso particular, hemos tomado los mismos lineamientos de repartición de carga con la salvedad que los formatos se generan colocando todos los procesadores en una fila, y sucesivas filas se crean haciendo un corrimiento circular hacia la izquierda de la fila inmediata anterior (Fig. 2-b).

Respecto a la arquitectura de paralelización, existen principalmente tres tipos de procesos: a) Generador de rayos (GR), b) Consumidores (C) y c) Colector de pixeles (CP). El nodo GR es el responsable de generar los rayos, intersectarlos con el volumen y enviarlos al nodo C respectivo. Los subvolumenes son distribuidos en los nodos C, quienes calculan

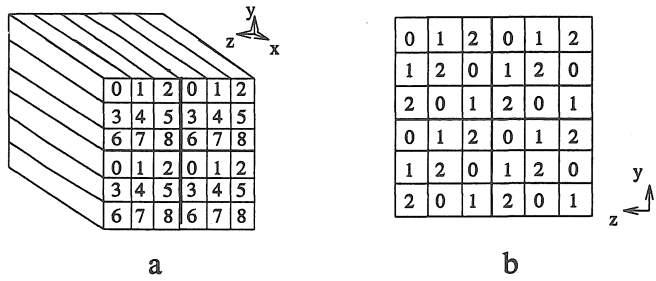

Fig. 2: Descomposición por dispersión.

las interpolaciones, intensidades y opacidades correspondientes. Al finalizar el recorrido, se transmite el resultado al nodo CP. Físicamente, los nodos GR y CP comparten un mismo procesador, mientras que los nodos C se distribuyen en diferentes procesadores. La finalización ocurre cuando el nodo CP recibe todos los resultados, con lo cual transmite la imagen al procesador maestro. El tipo de división adoptado puede obligar que diferentes secciones de un mismo rayo sean trabajados secuencialmente en diferentes procesadores, y por tanto es necesaria una considerable comunicación entre este tipo de nodos.

4 Descomposición del espacio de la imagen (DeI)

En este esquema, la partición está definida por planos de corte que intersectan tanto al volumen como al plano de visualización, y tienen la misma dirección que los proyectores[1]. Tomando la idea de las formatos mencionadas anteriormente, la división del dominio está determinada por el número de procesadores P que intenvendrán en el procesamiento y por el número F de formatos. La figura 3 muestra un ejemplo para $F = 2$ y $P = 3$. La carga de trabajo correspondiente a cada procesador constituye una o varias particiones cuya(s) área del plano de visualización y del volumen se encuentren limitados por el mismo par de planos de corte. Esto implica que cada rayo es procesado en su totalidad en el mismo procesador.

La descomposición comienza seleccionando dos aristas del volumen, paralelas entre sí, quienes junto con el punto del observador determinan la orientación de los planos de corte

[1]Definición tomada de [6]

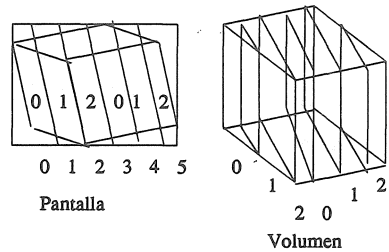

Fig. 3: Particiones de la pantalla y el volumen.

del volumen. Se ha convenido seleccionar aquellas que conforman la silueta de la proyección del volumen. A continuación se calculan $P * F - 1$ rectas paralelas equidistantes entre sí. De esta forma, los planos de corte son aquellos que contienen dichas rectas y que tienen la misma dirección que los proyectores de visualización. Respecto a la arquitectura de paralelización, ésta se simplifica notablemente, pues se traduce en un modelo de servidor y clientes [12].

5 Resultados Experimentales

El código se realizó en lenguaje C. La plataforma consta de nueve estaciones de trabajo SUN/IPX[2] conectadas por una red Ethernet. Con respecto a la comunicación entre los procesos, se utilizó la librería eMP [4]. Para analizar el desempeño se usaron el aceleramiento $A(N,P)$, la fracción serial empríricamente determinado f_c y el desbalance de carga $\delta_{N,P}$, medidos en términos del tiempo $T(N,P)$ empleado en el procesamiento de un problema de tamaño N aplicando P procesadores. Detalles de estas métricas se pueden encontrar en [7].

Las tablas 1, 2 y 3 resumen los resultados obtenidos para los modelos DeO y DeI sobre una red con disponibilidad exclusiva de recursos, en términos de dichas métricas. Los datos utilizados fueron los de un volumen inicial de tamaño $20x20x5$ interpolado a $128x128x128$ y pantalla de $256x256$ pixeles. El nivel de opacidad escogida fué altamente transparente para aumentar al máximo los requerimientos de procesamiento y comunicación.

Haciendo un análisis de dichas tablas, se observa que DeO supera en rendimiento a DeI, a medida que aumenta el número de procesadores. La explicación se debe a una combinación

[2]SUN/IPX es una marca registrada de Sun MicroSystems

	$P=2$	$P=3$	$P=5$	$P=9$
DeO	1.19	2.4652	4.7228	7.2757
DeI	1.91	2.5388	4.0929	5.6756

Tabla 1: Aceleramiento $A(N,P)$

	$P=2$	$P=3$	$P=5$	$P=9$
DeO	0.6807	0.1085	0.0147	0.0296
DeI	0.0471	0.0908	0.0554	0.0732

Tabla 2: Fracción serial f_c

de tres factores:a) Mejor balance de carga en la rutina de composición en DeO que en DeI; b) Menor área de cálculo redundante en DeO que en DeI y c) Estructuras menos costosas de manejar en DeO que en DeI.

Tal parece que el tiempo requerido en la comunicación de rayos en DeO, no juega un papel relevante en el rendimiento general. En este estudio, la arquitectura de paralelización de DeO contempla un complejo manejo de "buffers" y procesos satélites responsables de la transmisión y recepción de mensajes. Se hicieron pruebas con diferentes tamaños de mensajes y cotas superiores sobre el total de rayos procesándose simultáneamente en los nodos tipo C, obteniéndose que dichas variaciones pueden llegar a afectar el aceleramiento, el cual varía entre 5.5 y 7.4 cuando se tienen nueve procesadores. Por otro lado, el diseño de DeO mantiene al proceso GR ocioso cierto tiempo mientras los procesos se ocupan del preprocesamiento del volumen. Lo que quiere decir, que aún con esta desventaja se tiene que DeO tiene mejor rendimiento que DeI. El tiempo ocioso en GR explica también la baja aceleración cuando $C = 1$ en la tabla 1, lo que influye directamente en la fracción serial.

Otro punto interesante que se observa en la tabla 1 es la desaceleración que presentan ambos métodos a medida que aumenta el número de procesadores. En el caso de DeO, si el número de formatos $F > 1$, se obliga a que cada procesador mantenga varios subvolúmenes de los datos, lo cual genera más secciones del rayo lo que incide mayor tráfico en el sistema

	$P=2$	$P=3$	$P=5$	$P=9$
DeO	0.0000	1.2821	1.0526	3.7838
DeI	0.1002	11.332	12.179	17.777

Tabla 3: Desbalance de carga $\delta_{N,P}$

de comunicaciones. Por otro lado, en DeI, el desbalance de carga cobra mayor peso a medida que aumenta el número de procesadores. Obsérvese que el desbalance puede aumentar en casi un 50% al pasar de cinco a nueve procesadores.

Se analizó el rendimiento de DeO en una red no aislada, para ello se escogieron dos niveles de contención de la red a media y alta. Aunque son resultados muy particulares dependiente de las condiciones de la red, indican que DeO puede tener una degeneración de $A(V, F, C)$ de por lo menos un 50%, lo que puede indicar que el rendimiento de DeI puede aventajar a DeO bajo tales condiciones.

6 Conclusiones y Recomendaciones

Es manifiesta la necesidad de paralelizar métodos de visualización volumétrica debido la necesidad de obtener resultados en tiempos razonables. Se han presentado dos esquemas de paralelización para el método de visualización volumétrica basado en [10]. Resultados experimentales indican la factibilidad de tal paralelización.

Entre los dos esquemas de paralelización expuestos, DeO aventaja en rendimiento a DeI, indicando que el tiempo de comunicación no necesariamente puede llegar a dominar el tiempo de procesamiento global en una plataforma distribuida de estaciones de trabajo conectadas mediante una red Ethernet.

La paralelización de métodos de visualización volumétrica requiere poner especial cuidado en los "overhead" de paralelización como son: área de recálculo, facilidad de ubicar un voxel dentro del volumen, balance de carga, comunicación.

La aplicación de técnicas de paralelismo al área de visualización volumétrica ha sido poco explotada y queda mucho por hacer. Este primer prototipo de DeO requiere todavía una serie de afinamientos para lograr tiempos más aceptables de cómputo.

7 Agradecimientos

Deseamos expresar nuestro agradecimiento a las siguientes personas por su valiosa colaboración en el desarrollo de este trabajo: Benjamín Sagalovski, Arturo Puente, Antonio De

Lilla, Angel Sánchez, Lucio Nori, Jorge Manrique, Todd Elvins, José Ramón Jiménez, Jesús Ferrer, Rafael Montaño, Decanato de Postgrado de la Universidad Simón Bolívar y por último pero no menos importante, al Laboratorio de Computación USB.

Referencias

[1] G. M. Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In AFIPS Press, editor, *AFIPS Conference Proceedings*, pages [483–485], Reston, Va., 1967.

[2] J. Challinger. Parallel volume rendering on a shared-memory multiprocessor. Technical Report UCSC=CRL-91-23, University of California, Santa Cruz, March 1992.

[3] Z. Cvetanovic. The effects of problem partitioning, allocation and granularity on the performance of multiple processor systems. *IEEE Transactions on Computers*, C-36(4):[421–432], April 1987.

[4] A. DeLilla and R. Karia. Implementing eMP an easy message passing library. *Sun User Group Conference Proceedings*, December 1991.

[5] John Ellis, Gershon Kedem, Menon, and Herb Voelcker. Breaking barriers in solid modeling. *Mechanical Engineering*, 113(2):[28–34], February 1991.

[6] J. D. Foley, A. Van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice.* Addison Wesley, 1990.

[7] A. Hung. Composición volumétrica en paralelo. Master's thesis, Universidad Simón Bolívar, 1991.

[8] A. Kaufman and R. Bakalash. Memory and processor architecture for 3d voxel based imagery. 8(11):[10–23], November 1988.

[9] H. Kobayashi and H. Kubota and Y. Shigei S. Nishimura. Load balancing strategies for a ray parallel ray-tracing system based on constant subdivision. *The Visual Computer*, pages [197–209], April 1988.

[10] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, May 1988.

[11] Marc Levoy. Design for a real-time high-quality volume rendering workstation. In *Chapel Hill Workshop on Volume Visualization Conference Proceedings*, pages [85–92], 1989.

[12] David May and Roger Shepherd. Communicating process computers. Technical report, INMOS Ltd., Bristol, 1987. Inmos Technical Note 22.

[13] J. Salmon and J. Goldsmith. A ray tracer hypercube. *Communications of the ACM*, pages [1194–1286], 1988.