Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXVIII, págs. 1191-1201 (artículo completo)
Cristian García Bauza, Pablo Lotito, Lisandro Parente, Marcelo Vénere (Eds.)
Tandil, Argentina, 3-6 Noviembre 2009

TIRAS DE TRIÁNGULOS PARA MODELOS DE VISUALIZACIÓN MULTIRRESOLUCIÓN

María V. Cifuentes[a,b], **Lucas Lo Vercio**[a], **Javier Dottori**[a] **y Alejandro Clausse**[a]

[a]*PLADEMA, Universidad Nacional del Centro de la Provincia de Buenos Aires, Pinto 399, 7000 Tandil, Argentina, http://www.plademа.net*

[b]*CIC, Comisión de Investigaciones Científicas de la Provincia de Buenos Aires, calle ....., 1900 La Plata, Argentina, http://www.cic.gba.gov.ar*

Palabras Clave: Modelos Topográficos, Quadtrees, Triangulación, Tiras de Triángulos.

Resumen. Se formula un algoritmo para la construcción eficiente de modelos de visualización de escenarios topográficos. El método se basa en la generación de tiras de triángulos (*triangle strips*) sobre una grilla multirresolución construida a partir de un modelo digital de elevación en una representación matricial regular. La grilla resultante se define de manera que cada elemento acumule la misma cantidad de un dado escalar local (que se toma como guía para la visualización). Finalmente se estructura la información geométrica en un quadtree restringido que organiza a las diversas versiones simplificadas de la misma topografía con resolución variable. El algoritmo implementado recorre de izquierda a derecha las celdas de la grilla favoreciendo la formación de las tiras de triángulos.

1 INTRODUCCIÓN

En los sistemas de información geográfica (GIS), los simuladores de vuelo, los planificadores de misiones militares, entre otros, es necesario visualizar modelos de paisajes reales. Estos modelos son versiones simplificadas de modelos digitales de elevación (MDE) que alivian el tráfico de información entre la CPU y la GPU, garantizando tasas de interactividad apropiadas siguiendo algún criterio (Luebke y otros, 2003; Garland, 1999; Ribelles y otros, 2002). La primitiva básica usada en estos modelos es el triángulo. Sólo unos pocos modelos usan las primitivas basadas en tiras de triángulos. Las tiras de triángulos son estructuras más eficientes debido a que proporcionan la conectividad implícita en las actuales tarjetas de vídeo y, por lo tanto, el ahorro de ancho de banda entre la CPU y la tarjeta debido a que se envían menos de tres vértices por triángulo al hardware gráfico. No obstante, encontrar el conjunto de tiras óptimo es un problema NP-completo (Garey y otros, 1976), y por lo tanto requiere de cierta heurística.

Algunos autores usan las tiras de triángulos sólo en la etapa de rendering (Hoppe, 1997; El-Sana y otros, 1999), mientras que los trabajos recientes usan tales primitivas tanto en la estructura de datos como en el rendering (Belmonte y otros, 2004; Ramos y otros, 2004). Comparadas con las primitivas gráficas basadas en listas de triángulos, la utilización de tiras de triángulos aumenta la eficiencia de los modelos multiresolución. Hoppe (1997) propone un modelo multirresolución dependiente de la vista donde la información almacenada en una estructura jerárquica es recuperada desde un dado nivel de detalle sobre el que se construyen las tiras de triángulos al momento del renderizado. El-Sana y otros (1999) presenta un modelo poligonal multirresolución dependiente de la vista que se modela con tiras de triángulos usando el algoritmo STRIPE (Evans y otros, 1996). Ramos y otros (2004) presenta un algoritmo multirresolución dependiente de la vista cuya única estructura es un strip.

En (Cifuentes y otros, 2007) se propusieron algoritmos bottom-up para la simplificación automática de modelos topográficos digitales en O(n) mediante la construcción de una triangulación base que sirve de referencia a la visualización dinámica. Definido un indicador de curvatura local se zonifica la superficie en porciones rectangulares que acumulan una cota máxima de curvatura. Esta zonificación se organiza en un quadtree restringido cuyos nodos terminales se triangulan en una de cinco posibles situaciones que son identificadas rápidamente gracias a la localización directa de regiones vecinas. Luego, las regiones quadtree trianguladas se renderizan usando listas de triángulos. En general tales primitivas desmejoran las capacidades del hardware gráfico puesto que los vértices de triángulos adyacentes son enviados más de una vez a la placa gráfica.

Este trabajo presenta una propuesta para la definición automática de tiras de triángulos en grillas multiresolución que modelan grandes terrenos. Esta representación permite la visualización de modelos digitales de elevación (MDE) a partir de nodos activos que constituyen las hojas de un quadtree restringido. La configuración de nodos resultante es isomorfa con la disposición de matriz sobre la cual la estructura de datos es visitada de norte a sur y de este a oeste a fin de construir las tiras de triángulos que compondrán la triangulación final del modelo. El costo computacional resultante es lineal. Finalmente, hemos comprobado que las tiras de triángulos muestran una mejora significativa en la velocidad de rendering en comparación con los esquemas basados en la representación de triángulos individuales.

En la sección 2 se describe el algoritmo de simplificación y el indicador de error usado para el remallado de MDE por sectores. La sección 3 describe el algoritmo de recuperación del nivel de detalle. La sección 4 detalla el algoritmo de generación de las tiras de triángulos. En la sección 5 se presentan aplicaciones sobre modelos digitales de elevación reales.

2 SIMPLIFICACIÓN Y RENDERING INTERACTIVO DE MDE

Considérese un MDE definido por una grilla regular de pixeles cuadrados que luego se triangulan. El proceso más eficiente para simplificar el modelo por uno de menor resolución basado en un dado criterio es generar una jerarquía de mallas de diferente complejidad a través de divisiones consecutivas de los triángulos. Esto conduce naturalmente a representaciones quadtree, lo cual provee un método simple y rápido de localizar cualquier nodo organizando cada nivel en arreglos separados. Un criterio usual para simplificar el MDE es reducir el número de pixeles en una región lejana al observador y en regiones donde la curvatura es pequeña (Cifuentes y otros, 2007). Un indicador compacto de este criterio es:

$$T \leq \frac{k \cos\theta}{d} \tag{1}$$

donde k mide la curvatura local, d es la distancia al observador y θ es el ángulo que forma la dirección del movimiento del observador y la dirección del vector que une la posición del observador con el centro de la región (ver figura 1).
Usando una estructura quadtree, el indicador acumulativo de cada nivel k es calculado antes de sumar la curvatura acumulada correspondiente a los 4 descendientes. De esta manera se optimiza el algoritmo de simplificación generando rápidamente un quadtree restringido, correspondiente a la malla poligonal de la superficie, logrando un acceso inmediato a la selección interactiva de hojas en un esquema dinámico dependiente de la vista.

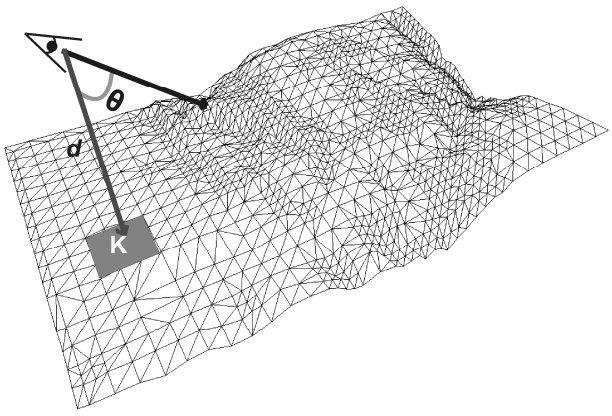


Figure 1: Esquematización del indicador compacto.

En las propuestas anteriores, las hojas del quadtree fueron trianguladas siguiendo templates de triángulos (Cifuentes y otros, 2009; Cifuentes y otros, 2007), representado el MDE mediante primitivas basadas en listas de triángulos. La desventaja de esta estructura de datos es que los vértices comunes a los triángulos adyacentes fueron enviados más de una vez al hardware gráfico. Para evitar este inconveniente, se propone usar tiras de triángulos, que son primitivas gráficas más eficientes y proveen conectividad implícita sobre las tarjetas de video

actuales. De esta manera, se optimiza el ancho de banda entre la CPU y la placa gráfica debido a que se envían menos de tres vértices por triángulo al pipeline gráfico. La desventaja de las tiras de triángulos es que en visualizaciones dinámicas la topología de la malla representada puede cambiar frame a frame, lo cual consume tiempo de CPU y espacio en memoria.

3 ALGORITMO PARA RECUPERACIÓN DEL NIVEL DE DETALLE

El algoritmo de recuperación propuesto para recuperar una versión del modelo acorde al nivel de detalle demandado recorre la estructura de datos cuaternaria. Para ello proponemos integrar cada hoja del quadtree a una secuencia de polígonos, resultando un algoritmo de ensamble rápido de la tira de triángulos.

La figura 2 muestra la proyección vertical del conjunto de nodos activos (hojas) del quadtree. La configuración de nodos resultante es isomorfa con la distribución de la matriz sobre la cual la estructura de datos es visitada (N-S y E-O) para construir la tira de triángulos final del modelo. De esta manera, solamente las hojas del árbol son visitadas en cada remallado, por lo que el costo computacional resultante es lineal.

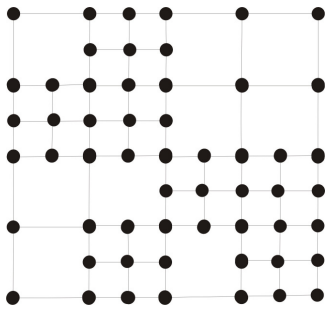


Figure 2: Proyección de los nodos activos (hojas) del quadtree.

La figura 3 muestra la secuencia de caminos para el conjunto dado en la figura 2. Los números en la figura guardan correspondencia con las líneas del pseudocódigo dadas en el algoritmo 1.

Hay situaciones que requieren especial atención. En particular, cuando un nodo adyacente ubicado al este del nodo terminal en proceso posee un nivel de subdivisión extra. En estos casos, la tira de triángulos en proceso incorpora a los hijos NO y NE del nodo adyacente ubicado al este. Mientras que los hijos SO y SE serán contenidos en una nueva tira. El primer recorrido transversal mostrado en la figura 3 ejemplifica la mencionada situación.

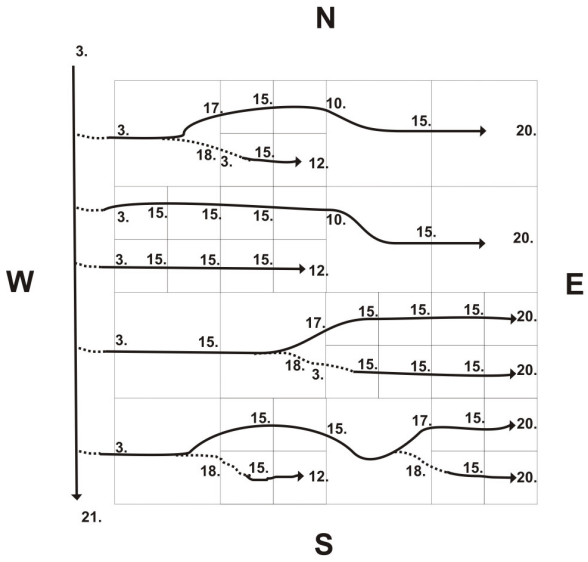


Figure 3: Secuencia del recorrido para la configuración de nodos activos de la figura 2.

Algoritmo 1: algoritmo de recorrido los nodos del quadtree mostrado en la figura 3

```
1.  Cargar en S los nodos terminales ubicados  más al oeste
2.   para  cada nodo en S hacer
3.          cadena.start( nodo)
4.          mientras  no es finDeCadena hacer
5.                  cadena.Work( nodo)
6                   nodoEste ← obtenerAdyEsteMismoNivel( nodo)
7.                  si  esValido(nodoEste) entonces
8.                          si  no existe AdyEsteMismoNivel entonces
9.                                  si    es hijoNorte( nodo) entonces
10                                          siguienteNodo ← obtenerPadre(nodoEste )
11                                  sino
12                                          finDeCadena ← TRUE
13                          sino
14                                  si   esHoja(nodoEste) entonces
15                                          siguienteNodo ← nodoEste
16                                  sino
17                                          siguienteNodo ← obtenerHijoNO(nodoEste)
18                                          agregar obtenerHijoSO(nodoEste) a S
19                  sino
20                          finDeCadena ← TRUE
21          cadena.end( );
```

.

4 GENERACIÓN DE LAS TIRAS

Dadas todas las posibles combinaciones, hay un conjunto de 16 posibles subdivisiones de una hoja del quadtree, representadas en la figura 4. Las flechas indican el orden en el cual los vértices son enviados a la placa gráfica, asegurando la vinculación correcta con las regiones anterior y siguiente. Este conjunto también asegura que la mayoría de los triángulos resultantes son acutángulos, lo cual beneficia la calidad de la visualización. En la figura 5 se muestra la generación de los strips de triángulos y la malla final para el ejemplo dado en la figura 3.

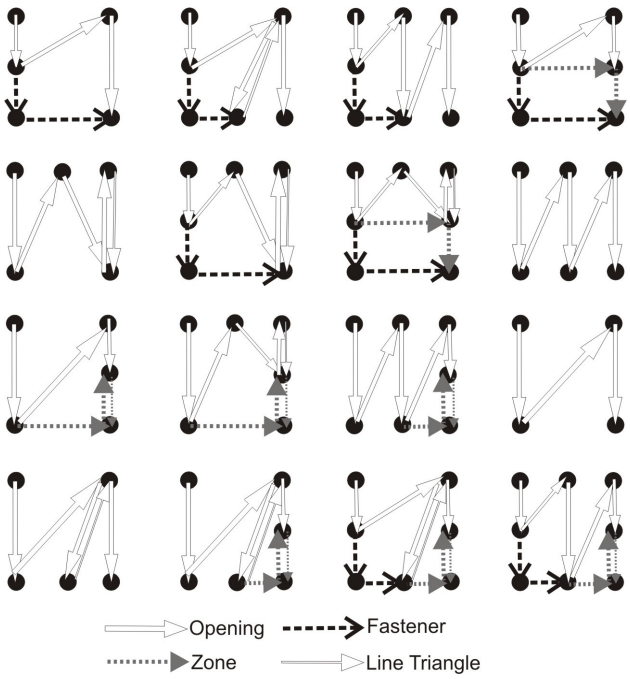


Figura 4: Conjunto de posibles strips.

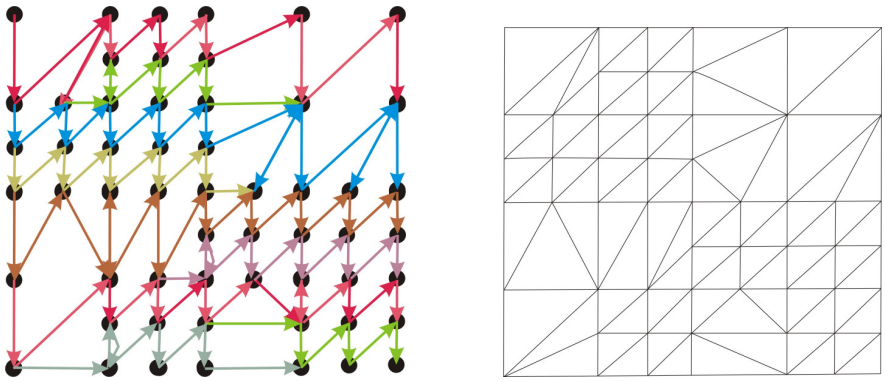
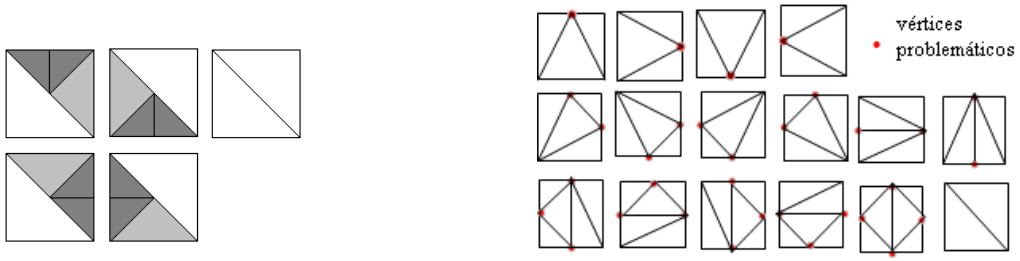


Figura 5: Generación de las tiras de triángulos y triangulación final para el ejemplo dado en la figura 3.

5 RESULTADOS Y EXPERIMENTOS

El algoritmo fue aplicado a la simplificación de una malla que representa al cañón del Colorado con 16.764.930 triángulos. La calidad de la malla es calculada mediante las diferencias cuadradas acumuladas entre las alturas del terreno original y la versión simplificada. La eficiencia de la simplificación es medida por el porcentaje adicional de triángulos requeridos para alcanzar el criterio aceptado relativo a la triangulación base. La aplicación visual fue implementada en C++ y OpenGL para la interface del usuario.

El presente algoritmo fue comparado con otros algoritmos de simplificación basados en la triangulación de templates mostrada en la figura 6, donde la restricción de permitir un único nivel de diferencia entre regiones adyacentes es aceptada. Bajo tales implementaciones es necesario verificar el nivel de subdivisión de los nodos adyacentes en orden a decidir cuál de los templates del menú debería ser aplicado. Los triángulos grises de la figura 6a corresponden a divisiones forzadas requeridas para cumplir con la restricción, mientras que los vértices adicionales del centro del cuadrado aceleran el proceso de triangulación por desacoplar la división de cada región del resto de la malla. Nótese que tales vértices adicionales no introducen error en la representación debido al criterio de coplanaridad asumido en el modelo de simplificación que asegura que los vértices que definen a una región quadtree se distribuyen en un mismo plano. Otra propuesta de templates implementada es mostrada en la figura 6b (Cifuentes y otros, 2009).



(a) Template A (b) Template B

Figura 6: Templates de triángulos que coinciden con un simple nivel de restricción entre nodos adyacentes.

La figura 7 muestra la representación del cañón del Colorado durante la navegación interactiva del modelo usando los templates de triángulos presentados en la figura 6. Ambos métodos proveen soluciones eficientes para la solución del problema de los cracks y uniones T entre regiones multirresolución. En la figura 8, el remallado se construye mediante las tiras de triángulos propuesta y una vista fototexturada del terreno durante la navegación del modelo correspondiente al cañón del Colorado.

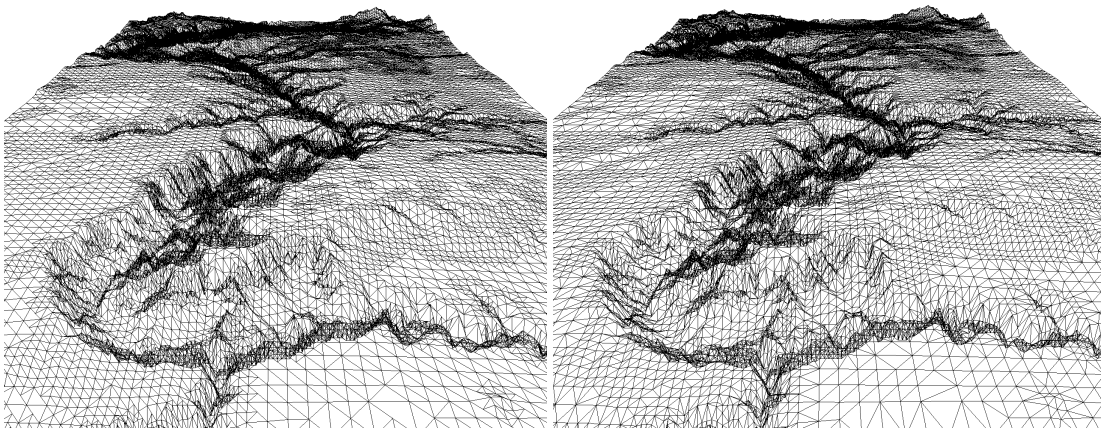


Figura 7: Remallado del Cañón del Colorado usando los templates de la fig 6a y 6b respectivamente.

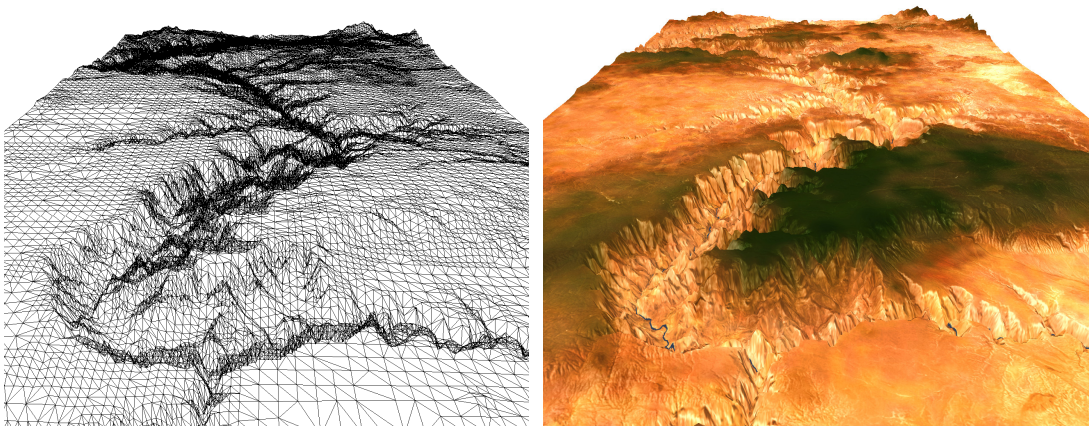


Figura8: Remallado del modelo del cañón del Colorado usando las tiras de triángulos.

La figura 9 muestra el número de vértices y triángulos de los modelos simplificados con cada método desarrollado. Puede observarse que el método de las tiras de triángulos realiza remallados similares con un número de vértices mucho menor, reduciendo substancialmente el flujo de datos enviado a la tarjeta gráfica.

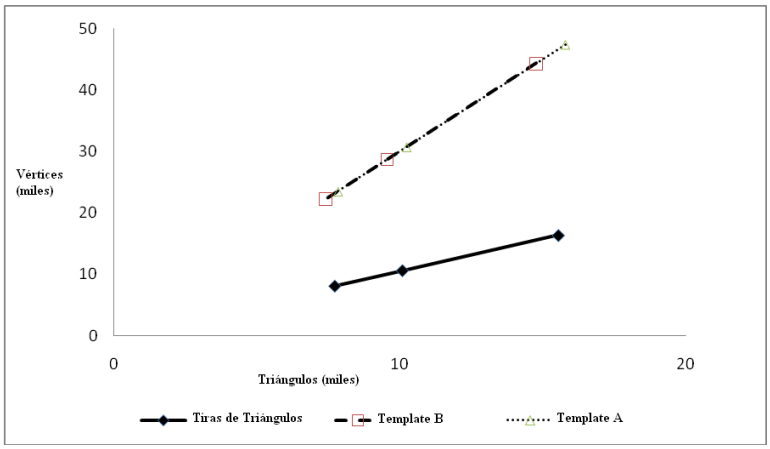


Figura 9: Triángulos vs Vértices enviados a la placa con los distintos templates.

La figura 10 muestra el tiempo de rendering requerido por cada método. Puede observarse que el tiempo de rendering empleado por el algoritmo de tiras de triángulos es cercano a la mitad del tiempo requerido por el uso de listas asociadas a los templates.

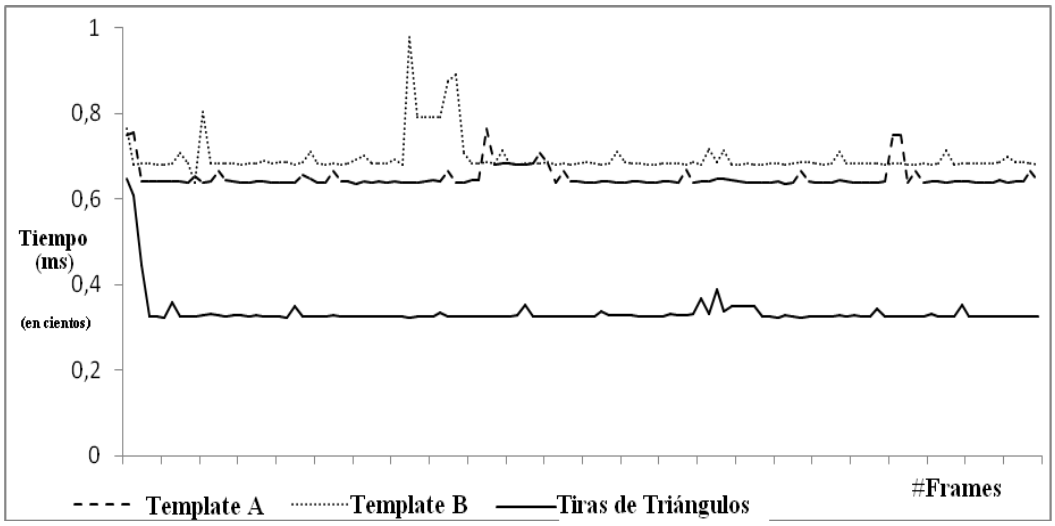


Figura 10: Tiempo empleado en el proceso de rendering para cada método estudiado.

Al representar un terreno con diferente grado de resolución, la figura 11 grafica el tiempo empleado en cada frame usando las tiras de triángulos, en remallado, recuperación del nivel de detalle y renderizado. La tasa de frames por segundos se conserva en todo el proceso de remallado.

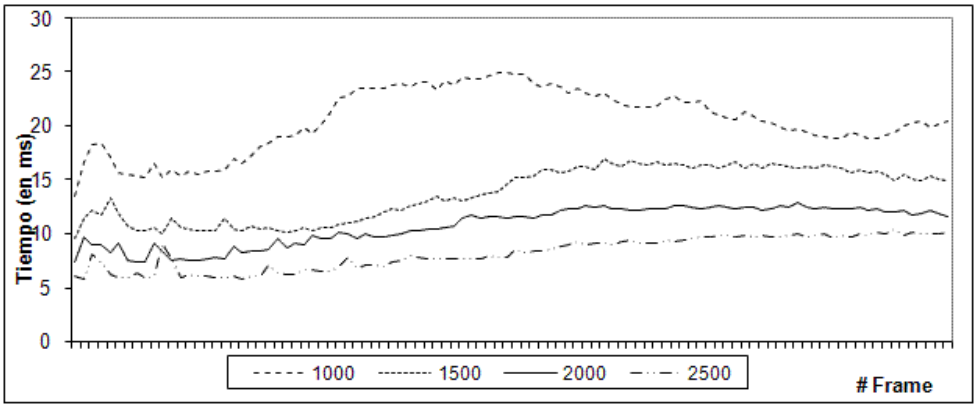


Figura 11: Tiras de triángulos sobre un terreno con distinta calidad.

6 CONCLUSIONES

Se presentó un método de simplificación de MDE mediante mallas multiresolución basadas en tiras de triángulos. La metodología propuesta y los algoritmos producen tasas de frames en tiempo real durante la navegación del terrenos mostrando vistas 3D fototexturadas de la superficie independiente de las dimensiones del terreno. El algoritmo de triangulación basado en tiras de triángulos realiza menos cálculos en CPU, resolviendo eficientemente el problema del ancho de banda y eliminando transformaciones y cálculos de iluminación redundantes en 3D. Además, los triángulos strips mejoran la performance en el rendering en comparación con el uso de listas de triángulos.

REFERENCIAS

Belmonte, O., Remolar, I., Ribelles J. , Chover M., Fernández, M. Efficiently Using Connectivity Information between Triangles in a Mesh for Real-Time Rendering. *Future Generation Computer Systems*, 20(8): 1263-1273, 2004.

Cifuentes, M., Vénere, M., Clausse, A. Interactive Resmeshing for Large Landscapes, *Latin American Applied Research*, ISSN: 0327-0793,en prensa, 2009.

Cifuentes, M., Dottori, J., Lo Vercio, L., Clausse, A. Visualización Interactiva Anisotrópica de Modelos Topográficos. *Mecánica Computacional V.26*, ISSN 1666-6070, p. 739-746, 2007.

El-Sana, J., Azanli, E., Varshney, A. Skip Strips: Maintaining Triangle Strips for View-dependent Rendering, *IEEE Visualisation* '99: 131-138, 1999.

Evans, F., Skiena, S., Varshney, A. Optimising triangle strips for fast rendering, *IEEE Visualisation* '96: 319-326, 1996.

Garey, M. R., Johnson, D. S., Tarjan, R. E. The planar hamiltonian circuit problem is NP-COMPLETE. *SIAM Journal of Computing* 5, 4:704–714, 1976.

Garland, M. Multiresolution modelling: survey & future opportunities, S*tate of the Art Reports of EUROGRAPHICS '99*: 111-131, 1999.

Hoppe, H. View-dependent refinement of progressive meshes, *Proceedings of SIGGRAPH '97*: 189-197, 1997.

Luebke, D., Reddy, M., Cohen, J.D., Varshney, A., Watson, B., Huebner, R. Level of detail for 3D graphics. *Morgan-Kaufmann*, 2003.

Ramos, F., Chover, M. LodStrips, Proceedings of Computational Science ICCS 2004, pp:

107-114, 2004.

Ribelles, J., López, A., Belmonte, O., Remolar, I., Chover, M. Multiresolution modeling of arbitrary polygonal surfaces: a characterization. *Computers & Graphics*, 26(3): 449-462, 2002.