UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
VICERRECTORÍA DE FORMACIÓN
DIRECCIÓN DE FORMACIÓN POSTGRADUADA

Modelación y visualización de superficies de terrenos en tres dimensiones

Tesis presentada en opción al título de Máster en Informática Aplicada

Autor: **Lic. Yusnier Valle Martínez**

Tutor: **Dr. José Ortíz Rojas**

Co-tutor: **Dr. Emilio Escartín Sauleda**

Ciudad de la Habana, Julio de 2009

A mis padres, Martín y Odalys.

A mi esposa Yanet.

DECLARACIÓN JURADA DE AUTORÍA Y AGRADECIMIENTOS

Yo Yusnier Valle Martínez, con carné de identidad 79042401689, declaro que soy el autor principal del resultado que expongo en la presente memoria titulada "Modelación y visualización de superficies de terrenos en tres dimensiones", para optar por el título de Máster en Informática Aplicada.

El presente trabajo fue desarrollado individualmente en el transcurso de los años 2007-09.

En especial deseo agradecer al Dr. José Ortíz Rojas y al Dr. Emilio Escartín Sauleda, quienes fungieron como tutores de mi formación como máster. Además, deseo agradecer al Dr. Antonio Mesa Enríquez quien también contribuyó a mi crecimiento profesional y humano en general. A todos ellos, así como a otros colegas y amigos que no he mencionado por razones de espacio, mi más sincero agradecimiento.

Finalmente declaro que todo lo anteriormente expuesto se ajusta a la verdad, y asumo la responsabilidad moral y jurídica que se derive de este juramento profesional.

Y para que así conste, firmo la presente declaración jurada de autoría en Ciudad de la Habana a los 14 días del mes de Julio del año 2009.

_____

Firma del Maestrante

Resumen.

Los Modelos Digitales de Terrenos consisten en una representación digital de las propiedades de la topografía de una superficie. En particular, los Modelos Digitales de Elevaciones se encuentran entre los más conocidos, los cuales comúnmente se almacenan en forma de malla rectangular regular, malla irregular triangulada o curvas de nivel o de contorno. Por su importancia en una amplia gama de aplicaciones, el problema de visualizar este tipo de información de forma interactiva y eficiente constituye un reto en la actualidad para la comunidad científica.

En el presente documento se exponen nuevas técnicas de modelación y visualización interactiva de superficies de terrenos en entornos tridimensionales, con el propósito de modelar superficies que por su extensión puedan ser almacenadas en su totalidad en memoria RAM. Las técnicas clásicas en el tratamiento de este problema involucran tanto estructuras de datos espaciales como algoritmos para almacenar y manipular modelos poligonales construidos a partir de la información proporcionada por los datos; modelos estos que constituyen la base para la extracción de triangulaciones multiresolución -distintos niveles de detalles dependiendo de varios factores- con el objetivo de ser visualizadas con la calidad e inmediatez requeridas.

La diversidad de técnicas existentes en la literatura presenta varios elementos claves en común; tal es el caso de la eficiencia tanto en los algoritmos de creación de los modelos a partir de los datos y en el costo de almacenamiento de los mismos, como en la extracción de triangulaciones multiresolución interactivamente y en tiempo real. Las técnicas que proponen el criterio de Delaunay para construir triangulaciones sobre conjuntos de datos dispersos se encuentran entre las más aceptadas por la comunidad científica; así como las que utilizan estructuras de datos jerárquicas, como los Quadtree, para la representación de los modelos. La principal contribución del presente trabajo radica en la propuesta de una técnica que implementa un Quadtree haciendo uso de la curva de recorrido del espacio de Hilbert; con el objetivo de utilizar la menor cantidad de memoria posible en el almacenamiento de los modelos, y de incrementar la eficiencia en la visualización de las triangulaciones multiresolución extraídas a partir de estos últimos.

Índice general

Índice de figuras

Índice de tablas

Introducción

La visualización interactiva de superficies de terrenos ha sido uno de los temas más estudiados en los últimos años en la disciplina Gráficos por Computadora. El incremento constante de las capacidades de almacenamiento en los sistemas de cómputo, así como el perfeccionamiento de las técnicas de adquisición de datos de la superficie terrestre, ha permitido crear grandes volúmenes de información de superficies de terrenos que requieren de gran eficiencia en las técnicas a utilizar para su visualización en tiempo real en entornos tridimensionales. El desarrollo gradual de la tecnología ha posibilitado representar estos datos interactivamente mediante modelos poligonales con diversos niveles de resolución.

Un MDT[1] consiste en una representación digital de las propiedades de la topografía de una superficie. Dentro de estos, los MDE[2] se encuentran entre los más conocidos, los cuales comúnmente se almacenan en forma de malla rectangular regular, malla irregular triangulada o curvas de nivel o de contorno. Los modelos digitales de terrenos constituyen un elemento importante en una amplia gama de aplicaciones, desde Sistemas de Información Geográfica, video juegos, hasta simuladores de vuelo, entre otros. En la mayoría de los casos el terreno no constituye el elemento más importante en una simulación, de ahí la importancia de que su representación sea lo más eficiente posible.

Para lograr un mayor aprovechamiento de las características, tanto de hardware como de software, que presentan los actuales sistemas, se hace necesario que la complejidad de la escena que se está visualizando se reduzca tanto como sea posible. Por ejemplo, en una simulación de vuelo, tanto el terreno como los objetos en el mismo son visualizados con diferentes grados de resolución, dependiendo de la distancia a la que se encuentra el punto de observación. Para alcanzar este objetivo, se utilizan principalmente modelos constituidos por mallas de triángulos, y aunque la representación de este tipo de polígonos es ampliamente soportado por el hardware, y por tanto altamente eficiente su representación, la gran cantidad de datos pertenecientes a los modelos digitales de elevaciones es demasiado extensa para ser manejada en su totalidad, debido a su elevado costo

[1] Modelo Digital de Terrenos.
[2] Modelo Digital de Elevaciones.

de almacenamiento y manipulación.

A pesar de que la disponibilidad de memoria, las velocidades de cómputo y la eficiencia de los motores gráficos son cada vez mayores, en ocasiones no son suficientes dadas las dimensiones de la superficie a representar. El problema principal es el costo de almacenamiento del modelo poligonal, así como la implementación de algoritmos eficientes para su visualización y acceso espacial a la información. Por todas estas razones, resulta fundamental la simplificación de los modelos preservando en cierto grado la exactitud proporcionada por los datos.

Como consecuencia de estas deficiencias han surgido diversas teorías acerca de cómo generar los modelos con la mínima cantidad de polígonos necesaria. Todas ellas parecen converger en un punto: los llamados "modelos multiresolución" [Floriani y Magillo, 2002, Pajarola, 2002]. A los efectos del presente trabajo, solo son de interés las mallas rectangulares regulares representadas mediante arreglos bidimensionales, sobre las que se aplican técnicas de modelación muy eficientes basadas en estructuras de datos espaciales [de Berg y col., 2000, Samet, 1990a]. Posteriormente se extrae una triangulación con distintos niveles de detalles, dependiendo de factores como las características del relieve de las superficies, o la posición del observador, que es visualizada en tiempo real con cierto margen de error prefijado.

Antecedentes

El desarrollo del presente trabajo tiene su génesis en la solicitud, por parte del Instituto de Meteorología (INSMET), de confección de un software para mostrar información meteorológica en televisión (Show TV). Aunque el proyecto en su totalidad no se realizó, si se lograron alcanzar determinados resultados preliminares que avalaban la factibilidad de realización del mismo; tales son los casos del desarrollo de una herramienta de cálculo de isolíneas a partir de datos proporcionados por estaciones meteorológicas, y de otra de generación de videos de secuencias animadas de imágenes construidas a partir de datos extraídos de satélites meteorológicos. Paralelamente al desarrollo de estas dos herramientas, se comenzó el estudio del estado del arte en modelación y visualización de superficies de terrenos en tres dimensiones, como elemento clave para el proyecto que se prentendía realizar.

Los datos de superficies de terrenos revisten gran importancia no solo para Show TV, sino en general para las ciencias que integran el tratamiento, análisis, interpretación y almacenamiento de información geográfica, entre otras. Dada la inexistencia en nuestro equipo de trabajo, de una técnica que utilice una mínima cantidad de memoria para la

representación espacial de los modelos, a la vez que realice una visualización eficiente de los mismos, se plantea la necesidad de desarrollar un conjunto de estructuras de datos y algoritmos que permitan modelar y visualizar interactivamente MDEs, que por su extensión puedan ser representados completamente en memoria RAM.

El **objeto** sobre el que se enfoca el **estudio**, tanto desde el punto de vista teórico como práctico, con vistas a la solución del problema planteado consiste en la modelación y visualización interactiva de superficies en entornos tridimensionales, específicamente en las técnicas de modelación y visualización interactiva de mallas poligonales con múltiples niveles de resolución, lo cual constituye el **campo de acción** de la investigación.

Con el propósito de brindarle una solución efectiva al problema, se plantea como **objetivo general** proponer un conjunto de estructuras de datos y algoritmos que permitan modelar y visualizar superficies de terrenos de manera eficiente y en tiempo real, a partir del estudio de técnicas existentes en la literatura que se utilizan para modelar este tipo de información.

Hipótesis

Diversas son las estructuras de datos y algoritmos que se aplican en la resolución del problema descrito. En el marco del presente trabajo, se proponen nuevas variantes de solución partiendo de la siguiente hipótesis:

Si se realiza una implementación eficiente de estructuras para modelar espacialmente los datos proporcionados por modelos digitales de terrenos, y se minimiza el número de primitivas a enviar al sistema gráfico para llevar a cabo el rendering de triangulaciones extraídas de los modelos resultantes, entonces se obtendrá una técnica de modelación y visualización interactiva de terrenos en tres dimensiones, que utiliza una reducida cantidad de memoria para la representación espacial de los modelos, a la vez que acelera el proceso de visualización de las mallas poligonales representativas de las superficies.

Tareas de investigación

Como tareas de investigación se proponen las siguientes:

- Identificar las variantes de solución existentes y tendencias a seguir en la solución del problema planteado, a partir de un estudio profundo del estado del arte que precede la realización del presente trabajo.
- Diseñar una estructura de datos que utilice la menor cantidad de memoria posible para representar espacialmente los modelos digitales de terrenos.

- Diseñar algoritmos eficientes de búsqueda y acceso espacial sobre la estructura, factor que posteriormente influirá directamente en el proceso de rendering del modelo.
- Diseñar algoritmos de recorrido sobre la estructura con el objetivo de generar eficientemente mallas poligonales con múltiples niveles de resolución a partir de éstas.
- Diseñar e implementar una herramienta simple que permita realizar pruebas en tiempo real de las técnicas de modelación y visualización propuestas.

La contribución fundamental del presente trabajo consiste en la propuesta de un enfoque novedoso de implementación de un Quadtree, haciendo uso de la curva de recorrido del espacio de Hilbert, para la representación en memoria de modelos de elevaciones. Como resultado complementario se proporciona una herramienta sencilla de modelación y visualización interactiva en tiempo real de MDEs, demostrando la factibilidad de uso de las técnicas propuestas.

Estructura del documento

El presente documento se encuentra dividido en tres capítulos. En el primero de ellos, fundamentación teórica, se realiza un análisis de algunas de las técnicas más eficientes para representar modelos de terrenos descritas en la literatura, además de brindar elementos esenciales sobre el concepto de curvas de recorrido del espacio.

El segundo capítulo, describe los elementos fundamentales referentes a las técnicas de modelación propuestas. Comienza exponiendo un conjunto de elementos referentes a la construcción del modelo espacial de los datos, continúa abordando las particularidades de la simplicación geométrica que se realiza para obtener las vistas multiresolución de la superficie, y culmina con un análisis de las variantes de triangulación de los modelos para su visualización interactiva y en tiempo real.

En el tercer capítulo se presentan los resultados obtenidos durante el proceso de investigación y desarrollo llevado a cabo. Posteriormente, se plantean un conjunto de elementos que consideramos importantes para su futura incorporación a las técnicas planteadas en el documento con el objetivo de incrementar su eficiencia.

Además de un conjunto de conclusiones y recomendaciones, como anexos del documento, se proporcionan elementos básicos de Geometría del Espacio utilizados en la implementación de una herramienta de prueba de las técnicas propuestas; se abordan las características esenciales de la interfaz de software para aplicaciones gráficas `OpenGL`, y finalmente se muestran un conjunto de fotografías de corridas de la herramienta construida.

Capítulo 1

Fundamentación teórica

Con el objetivo de facilitar la comprensión del alcance de la investigación, en el presente capítulo se exponen conceptos fundamentales asociados al dominio del problema planteado. Además, se realiza un análisis detallado del estado del arte que precede a la realización de este trabajo y que contribuye a esclarecer su objeto de estudio. Como elemento adicional se exponen las principales características de curvas de recorrido del espacio utilizadas en el diseño de estructuras de datos espaciales.

1.1. Conceptos asociados al dominio del problema

El concepto de **superficie** puede definirse desde el punto de vista geográfico como la extensión de un determinado territorio. Aunque definiciones más rigurosas, desde el punto de vista matemático, pueden encontrarse en trabajos relacionados con Topología[1], la proporcionada por el matemático griego Euclides en su tratado Los Elementos resulta lo suficientemente intuitiva para su comprensión:

Definición 1.1.1 *Una superficie es aquello que sólo tiene longitud y anchura.*

A partir de la definición 1.1.1 deriva el término superficies cerradas[2] que, en tres dimensiones, intuitivamente no son más que superficies que encierran un volumen como es el caso de la esfera o la Botella de Klein[3]. Contrariamente, las **superficies abiertas** son aquellas que presentan fronteras, como es el caso del cilindro.

[1]Disciplina matemática que estudia las propiedades de los espacios topológicos y las funciones continuas.

[2]Tomado de: http://es.wikipedia.org/wiki/Superficie_(matematica)

[3]Disponible en: http://es.wikipedia.org/wiki/Botella_de_Klein

A los efectos de este trabajo, de la clase de superficies abiertas solo son de interés las porciones de superficies terrestres. Según [de Berg y col., 2000], una **superficie de terrenos** puede definirse como sigue:

DEFINICIÓN 1.1.2 *Se denomina superficie de terrenos al grafo de una función* $f : A \subset \mathbb{R}^2 \rightarrow \mathbb{R}$ *que asigna un valor de altura* $f(p)$ *a todo punto* p *en el dominio* A *del terreno.*

Si se denota por S la superficie de terrenos definida por el conjunto de puntos en el plano $P = \{p_1, p_2, ..., p_n\}$, entonces una de las formas de representar a S es mediante una triangulación de P (en lo adelante $T(P)$). Según [de Berg y col., 2000], una $T(P)$ se define como una subdivisión planar maximal cuyo conjunto de vértices es P. De todas las $T(P)$ se ha demostrado [de Berg y col., 2000] que la que mejor aproxima a S se conoce como **triangulación de Delaunay**, Figura 1.1.

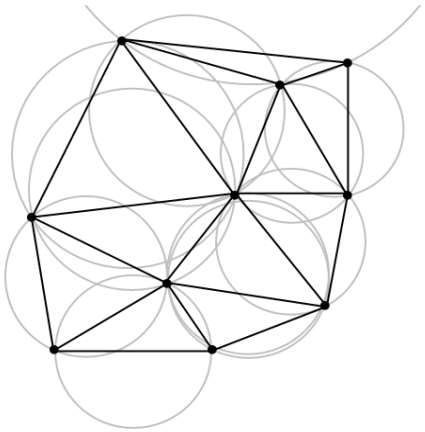


Figura 1.1: Triangulación de Delaunay.

Una triangulación $DT(P)$ es de Delaunay si y solo si la circunferencia circunscrita a cualquier triángulo de $DT(P)$ no contiene ningún otro punto de P, Figura 1.1. Un resultado establecido que garantiza la idoneidad de las $DT(P)$ para la representación de S, consiste en que cualquier $DT(P)$ maximiza el ángulo mínimo sobre todas las $T(P)$.

1.1.1. Rendering

El término **rendering** o rendereo normalmente se utiliza en la literatura para designar el proceso de creación de gráficos en entornos tridimensionales en computadoras. En la disciplina Gráficos por Computadora, una **tubería de rendering**[4] contempla un conjunto

[4]Tomado de: http://es.wikipedia.org/wiki/Tuberia_de_rendereo

de etapas (ej: transformación, iluminación, proyección, texturización, rasterización, etc.) en las que, a partir de una escena tridimensional de entrada, se produce como resultado una imagen raster.

Las principales API[5] de producción de gráficos -tanto en 2D como en 3D- en computadoras, como son los casos de `OpenGL` (ver Anexo B) y `DirectX`, ofrecen un conjunto de funcionalidades que permiten acelerar el proceso de rendering de las escenas. Específicamente para el caso de los triángulos existen varias [Angel, 2002, Shreiner, 2004, Ope, 1999] formas de llevar a cabo su visualización, resultando las más usadas:

- `TRIANGLE_STRIP (en lo adelante cadena de triángulos)`: consiste en representar la triangulación mediante una única lista de n vértices en la que cada trío consecutivo representa un triángulo, creándose $n-2$ en total.
- `TRIANGLE_FAN (en lo adelante abanico de triángulos)`: dada una lista de vértices, el triángulo n está definido por los vértices 1, $n+1$ y $n+2$.

1.1.2. Quadtrees

El término Quadtree es usado para describir una clase de estructuras de datos jerárquicas cuya propiedad común es que se basan en el principio de descomposición recursiva del espacio [Samet, 1990a, Samet, 1990b]. Básicamente pueden ser diferenciados por los siguientes criterios:

- El tipo de datos que representan: puntos (ej: Point-Quadtree), regiones (ej: Region-Quadtree), curvas, volúmenes, etc.
- El principio que guía el proceso de descomposición recursiva.
- La resolución (variable o no).

La descomposición suele ser en partes iguales -descomposición regular- en cada nivel, o guiada por los datos de entrada. De igual forma, la resolución puede prefijarse con antelación o estar determinada por las propiedades de los datos de la entrada. Según [de Berg y col., 2000], un Quadtree para un conjunto de puntos P dentro de un cuadrado σ se define como sigue:

Sea $\sigma = [x_\sigma : x'_\sigma] \times [y_\sigma : y'_\sigma]$.

[5]Interfaz de Programación de Aplicaciones (traducido de las siglas en inglés API: Application Program Interface).

- Si $card(P) \leq 1$, el Quadtree consiste en un nodo hoja que almacena a P y a σ.
- Si no, sean $\sigma_{NE}, \sigma_{NW}, \sigma_{SW}, \sigma_{SE}$ los cuatro cuadrantes de σ, y $x_{mid} = (x_\sigma + x'_\sigma)/2$, $y_{mid} = (y_\sigma + y'_\sigma)/2$. Se define:

$$\begin{aligned} P_{NE} &= \{p \in P : p_x > x_{mid} \quad y \quad p_y > y_{mid}\} \\ P_{NW} &= \{p \in P : p_x \leq x_{mid} \quad y \quad p_y > y_{mid}\} \\ P_{SW} &= \{p \in P : p_x \leq x_{mid} \quad y \quad p_y \leq y_{mid}\} \\ P_{SE} &= \{p \in P : p_x > x_{mid} \quad y \quad p_y \leq y_{mid}\} \end{aligned}$$

  El Quadtree ahora consiste en un nodo raíz v en el que σ es almacenado. Si se denota $\sigma(v)$ como el cuadrante almacenado en v, entonces:

  - El hijo NE es la raíz de un Quadtree para el conjunto P_{NE} dentro del cuadrado σ_{NE}.
  - El hijo NW es la raíz de un Quadtree para el conjunto P_{NW} dentro del cuadrado σ_{NW}.
  - El hijo SW es la raíz de un Quadtree para el conjunto P_{SW} dentro del cuadrado σ_{SW}.
  - El hijo SE es la raíz de un Quadtree para el conjunto P_{SE} dentro del cuadrado σ_{SE}.

La propia definición recursiva sugiere un algoritmo de construcción de la estructura: un cuadrante inicial σ se divide en cuatro cuadrantes -σ_{NE}, σ_{NW}, σ_{SW}, σ_{SE}- hijos, se particiona el conjunto adecuadamente, y de forma recursiva se construye el Quadtree correspondiente a cada descendiente. La división de un cuadrante concluye cuando su conjunto correspondiente contiene menos de dos puntos. Como resultado del proceso se obtiene una subdivisión jerárquica en la que cada cuadrante es dividido independientemente de sus vecinos, Figura 1.2 (a). Consecuentemente, una subdivisión Quadtree se denomina restringida si el nivel de división, por la arista que comparten, entre cualesquiera dos cuadrantes vecinos no excede la unidad, Figura 1.2 (b).

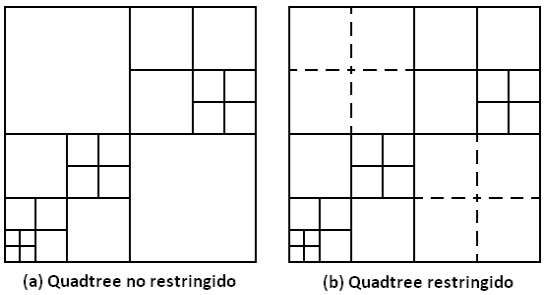


Figura 1.2: Vista de subdivisiones Quadtree restringidas y no restringidas.

1.2. Técnicas de modelación y visualización de terrenos

Cuando se trata el problema de realizar una visualización interactiva de superficies de terrenos en tres dimensiones, el consenso general en la comunidad científica es llevar a cabo la representación de las mismas mediante mallas poligonales. Generalmente se utilizan triángulos, debido a que tres puntos siempre determinan un único plano y por tanto su representación se hace más eficiente. Es por esta razón que para esta clase de polígonos se han desarrollado técnicas avanzadas que aceleran su rendering, convirtiéndose para muchos en la primitiva de dibujo más usada en gráficos de computadoras.

1.2.1. Redes Irregulares Trianguladas (TINs)

Diversas son las técnicas aplicadas en la representación de modelos digitales de terrenos. Los datos generalmente se presentan de dos formas: distribuidos irregularmente en el plano, o formando mallas rectangulares regulares. Aunque en el presente documento no se aborda la representación de terrenos a partir de conjuntos de puntos dispersos, Figura 1.3, si resulta importante destacar los avances más significativos obtenidos en tal sentido en los últimos años.

Las TINs [Cignoni y col., 1997, Cohen-Or y Levanoni, 1996, Guedes, 1997, Klein y Straßer, 1996, van Kreveld, 1997] generadas siguiendo el criterio de triangulación de Delaunay han sido ampliamente aplicadas en la modelación de este tipo de superficies. Entre los aportes más significativos referentes a la representación de TINs figura el expuesto por [van Kreveld, 1997], particularmente en la compresión y descompresión eficiente de una estructura de datos para su almacenamiento y transmisión a través de redes. Por otra parte, [Guedes, 1997] propone una optimización de una estructura piramidal, Figura 1.4, para representar una triangulación de Delaunay

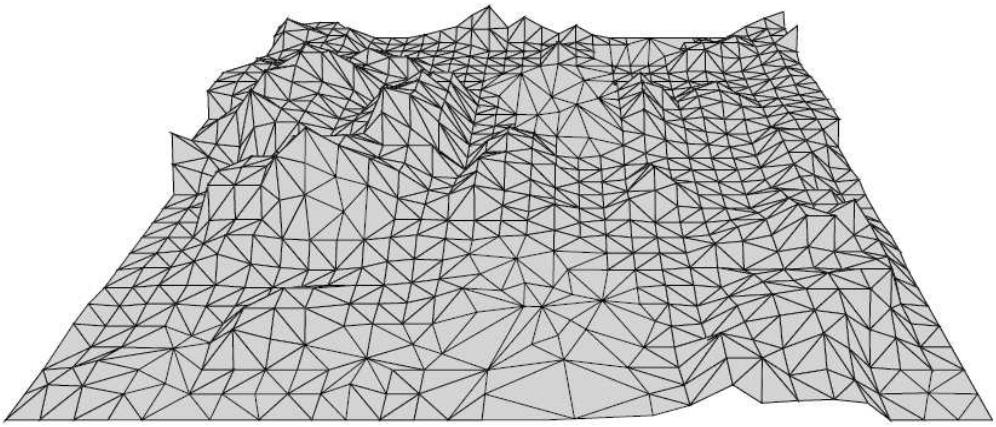


Figura 1.3: Vista en perspectiva de una superficie modelada con una TIN construida a partir de un conjunto de puntos dispersos.

jerárquicamente.

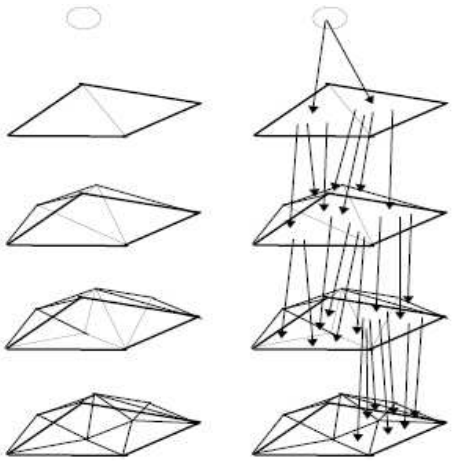


Figura 1.4: Representación piramidal de una triangulación de Delaunay.

El resultado fundamental de [Cignoni y col., 1997] consiste en una estructura de datos jerárquica y un conjunto de algoritmos que garantizan la extracción de triangulaciones irregulares aproximadas con precisión arbitraria, tanto constante como variable, sobre la superficie. La principal deficiencia de la representación de superficies de terrenos mediante TINs ha sido la complejidad en la obtención interactiva de triangulaciones con distintos niveles de detalle en tiempo real. No obstante, el nivel de precisión de las triangulaciones resultantes de la aplicación de este criterio en la modelación de superficies de terrenos, ha motivado la continua evolución de esta clase de técnicas. Aún cuando estructuras como las DCEL[7] han sido utilizadas ampliamente en la representación de TINs, la naturaleza

[7]Lista Doblemente Enlazada de Aristas (traducido de las siglas en inglés DCEL: Double Connected

un tanto rígida de éstas ha posibilitado que numerosas propuestas hagan uso de otras estructuras que, por su diseño jerárquico y facilidades de actualización, resultan más flexibles y adaptables a los requerimientos del problema en cuestión.

1.2.2. Quadtrees restringidos

Debido a la alta regularidad de los MDEs almacenados en forma de mallas rectangulares, los métodos de triangulación jerárquica de terrenos basados en Quadtrees se presentan entre los más eficientes. Este tipo de estructuras dividen recursivamente la superficie a modelar hasta lograr celdas de un tamaño dado. En el caso de las triangulaciones basadas en Quadtrees restringidos [Herzen y Barr, 1987, Lindstrom y col., 1995, Lindstrom y col., 1996, Lindstrom y Pascucci, 2002, Pajarola, 2002, Pajarola, 1998, Röttger y col., 1998], durante el proceso de división se debe tener en cuenta que para cada cuadrante, la diferencia entre el nivel de división del mismo y la de cada uno de sus vecinos, por la arista que comparten, no puede exceder la unidad, Figura 1.5.

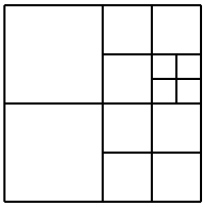


Figura 1.5: Divisiones realizadas por el Quadtree restringido.

Una vez terminada la fase de división, se procede a triangular cada celda de la siguiente forma: cada cuadrante se divide en ocho triángulos, dos triángulos por cada arista, a menos que una arista coincida con la de un vecino cuyo nivel de división es menor que el del cuadrante en cuestión [Herzen y Barr, 1987]. En este caso se forma un solo triángulo a lo largo de la misma, Figura 1.6.

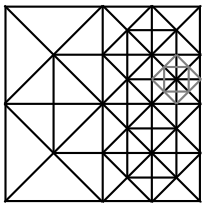
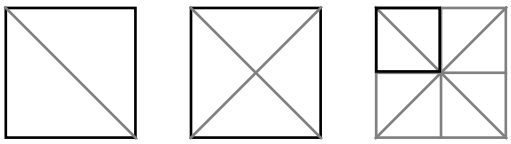


Figura 1.6: Vista de una triangulación sobre un Quadtree restringido.

La Figura 1.7 ilustra la forma en que se lleva a cabo el proceso de triangulación de los cuadrantes de la estructura. El cuadrante básico está constituido por cuatro puntos y dos

Edge List).

triángulos, Figura 1.7 a). El primer paso de refinamiento es llevado a cabo adicionando el punto medio del cuadrante. Este paso divide los dos triángulos iniciales en cuatro, Figura 1.7 b). Adicionando los puntos medios de cada arista del cuadrante, y dividiendo los triángulos consecuentemente, Figura 1.7 c), se completa el segundo paso del refinamiento.



a) b) c)

Figura 1.7: Proceso de triangulación de cuadrantes en un Quadtree.

Una segunda variante [Röttger y col., 1998], aunque mantiene la restricción de balance, plantea que las aristas compartidas por dos nodos del Quadtree que se encuentran en el mismo nivel de la jerarquía no necesitan ser divididas para garantizar una correcta triangulación, Figura 1.8.

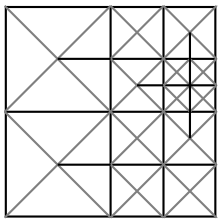


Figura 1.8: Variante de triangulación de un Quadtree restringido.

El principal aporte de los autores en [Röttger y col., 1998] consiste en lograr la representación del modelo mediante un Quadtree utilizando solo un arreglo bidimensional de ceros y unos, dependiendo de si un vértice determinado es seleccionado para formar parte de la triangulación multiresolución, Figura 1.9.

$$\begin{pmatrix} ? & ? & ? & ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & 0 & ? & 0 & ? \\ ? & ? & 0 & ? & ? & ? & 1 & ? & ? \\ ? & ? & ? & ? & ? & 0 & ? & 0 & ? \\ ? & ? & ? & ? & 1 & ? & ? & ? & ? \\ ? & 0 & ? & 0 & ? & 0 & ? & 1 & ? \\ ? & ? & 1 & ? & ? & ? & 1 & ? & ? \\ ? & 0 & ? & 0 & ? & 0 & ? & 1 & ? \\ ? & ? & ? & ? & ? & ? & ? & ? & ? \end{pmatrix}$$

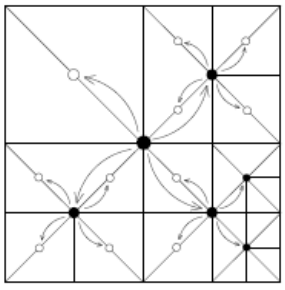


Figura 1.9: Matriz de ceros y unos y su correspondiente representación Quadtree.

Como aporte adicional se presenta una variante sencilla y eficiente para llevar a cabo el proceso de construcción de la malla de triángulos sobre la estructura, Figura 1.10. Aunque

la representación en memoria de los datos se lleva a cabo de manera muy eficiente, el proceso de construcción de la malla de triángulos puede ser acelerado aún más a partir del uso de cadenas de triángulos, en lugar de abanicos, como primitiva de rendering.

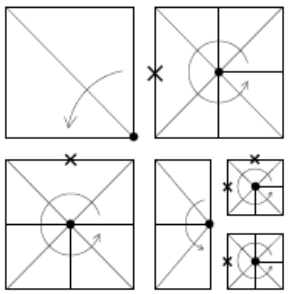


Figura 1.10: Generación de abanicos de triángulos por cada cuadrante de la estructura

Sin lugar a dudas entre los principales exponentes en lo que se refiere a triangulaciones basadas en Quadtrees restringidos se encuentran Lindstrom [Lindstrom y col., 1995, Lindstrom y col., 1996, Lindstrom y Pascucci, 2002] y Pajarola [Pajarola, 2002, Pajarola, 1998]. Ambos autores enfocan su atención en la representación eficiente de modelos construidos sobre conjuntos de datos arbitrariamente extensos que no pueden ser almacenados completamente en memoria, así como en la extracción eficiente de mallas triangulares con múltiples niveles de detalle y de manera progresiva para llevar a cabo su rendering haciendo uso de cadenas de triángulos. La principal contribución de Lindstrom radica en la introducción del concepto de dependencias entre vértices, Figura 1.11, como base para la extracción de las triangulaciones multiresolución sin la presencia de grietas en la superficie.

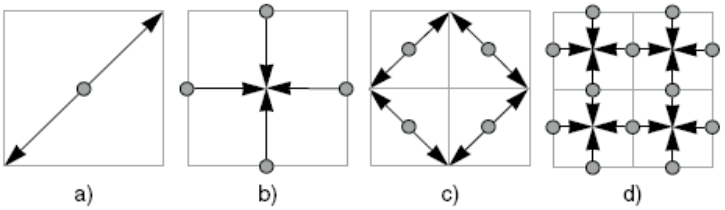


Figura 1.11: Esquema de dependencias entre vértices: a),b) nivel l y c),d) nivel $l+1$

El hecho de que el proceso de triangulación se lleve a cabo teniendo en cuenta las dimensiones de las regiones adyacentes, trae consigo que no se produzcan anomalías como las que se muestran en la Figura 1.12.

Aunque las RQT[8]s han demostrado ser altamente eficientes en la visualización de

[8]Triangulación sobre Quadtrees Restringidos (traducido de las siglas en inglés RQT: Restricted Quadtree Triangulation).

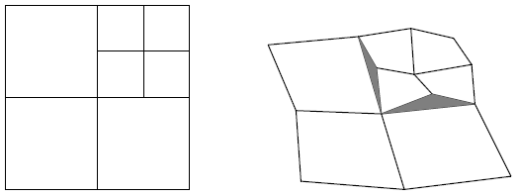


Figura 1.12: Grietas entre los cuadrantes representativos de una superficie.

superficies de terrenos, presentan una dificultad importante. La selección en la estructura de un cuadrante para ser triangulado provoca un efecto "cascada" en la selección de sus vecinos, dada la restricción impuesta en los niveles de división. Como consecuencia de esto pueden resultar seleccionados cuadrantes adicionales que no influyen en la calidad de visualización obtenida, traduciéndose en un mayor número de triángulos a visualizar, y por consiguiente en una disminución de la velocidad de rendering de la estructura.

1.2.3. Quadtrees no restringidos

Una alternativa a las RQTs son las denominadas triangulaciones basadas en Quadtrees no restringidos (NRQT[9])[Aguilera y col., 2006, Perez y col., 2004]. El proceso de división de cada cuadrante se realiza independientemente de sus vecinos, eliminando de esta forma la restricción de balance sobre la estructura, Figura 1.13.

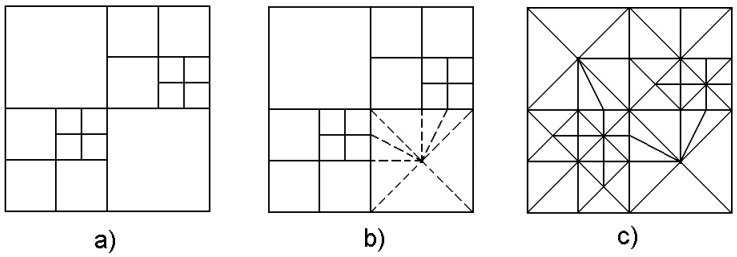


Figura 1.13: Triangulación sobre un Quadtree no restringido.

La Figura 1.13 a) muestra una vista de un Quadtree no restringido. Para obtener una NRQT, se adiciona el punto medio de cada cuadrante de la estructura y se trazan rectas hasta los puntos en la frontera del cuadrante en cuestión, Figura 1.13 b). Una posible triangulación sobre un Quadtree no restringido se muestra en la Figura 1.13 c).

La principal contribución de [Perez y col., 2004] consiste en una estructura basada en listas para almacenar una triangulación multiresolución, Figura 1.14. La estructura de

[9]Triangulación sobre Quadtrees No Restringidos (traducido de las siglas en inglés NRQT: Non-Restricted Quadtree Triangulation).

datos propuesta consta de dos conjuntos de listas que se mantienen ordenados de acuerdo al orden espacial de los vértices:

- El primer conjunto, denominado `ListsX`, contiene listas de vértices que comparten la misma coordenada y ordenadas por la coordenada x.
- El segundo conjunto, denominado `ListsY`, contiene listas de vértices que comparten la misma coordenada x ordenadas por la coordenada y

La forma que se propone para insertar los vértices de un Quadtree en las listas hace que el costo sea $O(1)$, en contraste con el $O(n)$ que requiere este tipo de operación en una lista ordenada arbitraria.

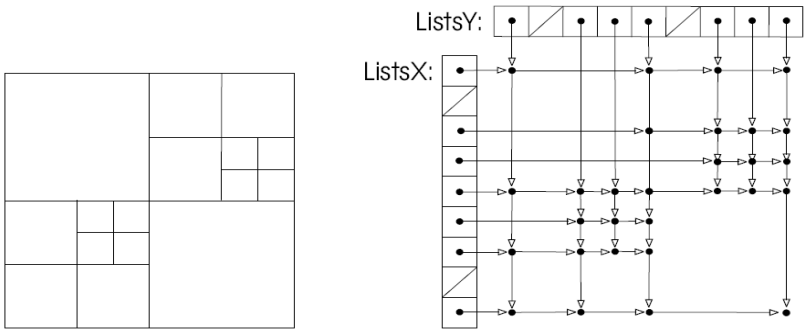


Figura 1.14: Quadtree no restringido y su correspondiente representación mediante listas de vértices.

Para llevar a cabo el proceso de triangulación de una estructura de forma eficiente, los cuadrantes son clasificados en dos categorías: aquellos que no poseen vecinos con mayor resolución y los que si los poseen. Para los primeros se generan cadenas de triángulos, y para los segundos abanicos, Figura 1.15.

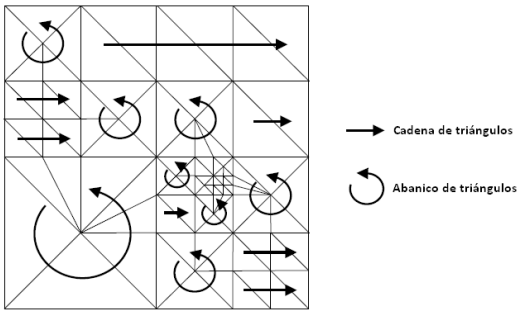


Figura 1.15: Generación de cadenas y abanicos de triángulos en una NRQT.

Dada la independencia en la selección de cuadrantes en el modelo para su posterior visualización, el número de triángulos en una NRQT es significativamente inferior al de una RQT equivalente.

1.2.4. Mallas Regulares R-Trianguladas

Otra de las técnicas de rendering de terrenos más utilizadas en este tipo de aplicaciones son las denominadas RTIN[10]s, donde cada triángulo resultante es rectángulo e isósceles. La principal contribución de [Evans y col., 2001] consiste en la propuesta de una estructura de datos muy eficiente para la representación del modelo en memoria, Figura 1.16. En una RTIN, cada triángulo es etiquetado recursivamente añadiendo un 0 o un 1 a la codificación de su ancestro, dependiendo de su posición como hijo izquierdo o derecho. La representación se realiza en forma de árbol binario de triángulos, para el cual se expone un algoritmo de búsqueda de vecinos muy eficiente.

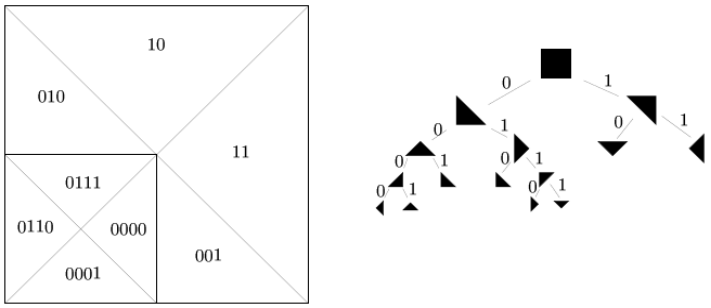
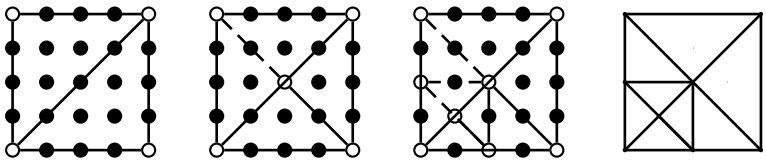


Figura 1.16: Esquema de una RTIN y su representación jerárquica en forma de árbol binario de triángulos.

Generalmente el proceso de triangulación comienza trazando una de las diagonales del cuadrante principal de la malla, Figura 1.17 a), obteniéndose como resultado dos triángulos. La división continúa, Figuras 1.17 b) y c), adicionando el punto medio, en la malla, de la hipotenusa de cada triángulo hasta alcanzar la resolución deseada. La Figura 1.17 d) muestra una posible triangulación resultante de aplicar el proceso a una malla rectangular regular.



a) b) c) d)

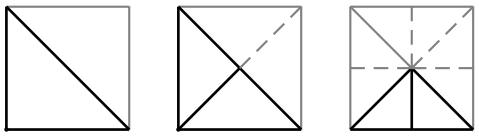
Figura 1.17: División Bitriangle.

Varios son los algoritmos propuestos en la literatura [Floriani y Magillo, 2002, Duchaineau y col., 1997, Evans y col., 2001, Floriani y Magillo, 2003] para generar

[10]Mallas Regulares R-Trianguladas (traducido de las siglas en inglés RTIN: Right Triangulated Irregular Networks).

RTINs de manera eficiente. La principal desventaja de esta técnica consiste en que la división de un triángulo genera, como en las RQTs, un efecto cascada de división en su vecino por la hipotenusa, Figuras 1.17 b) y c), contribuyendo de igual forma al aumento de polígonos a procesar por el sistema gráfico, y en consecuencia, disminuye la velocidad de rendering.

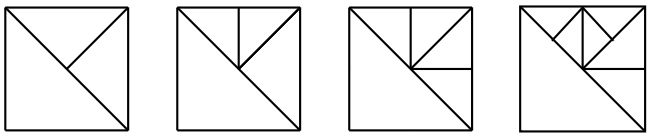
Una alternativa a las RTINs la constituye la NRBT[11] [Abásolo y col., 1999, Abásolo y col., 2000, Abásolo y Perales, 2003], que propone que la jerarquía de triángulos presente solo una etapa de la subdivisión recursiva de las que se realiza en el método anterior, Figura 1.18. El patrón básico consiste en un solo triángulo, como se muestra en la Figura 1.18 a), que es recursivamente subdividido adicionando el punto medio de su hipotenusa, Figura 1.18 b). La Figura 1.18 c) muestra como puede ser llevada a cabo la subdivisión de cada triángulo, obteniéndose resultados similares a los ilustrados en la Figura 1.17.



a) b) c)

Figura 1.18: Proceso de división de la superficie en una NRBT.

El proceso de subdivisión realizado de esta forma difiere del usado en el método anterior en el hecho de que, según el criterio seguido en este caso, cada triángulo puede ser subdividido independientemente de sus vecinos, eliminándose de esta forma el efecto cascada en la triangulación. En la Figura 1.19 se muestran posibles variantes de subdivisiones logradas con este esquema, que no son posibles obtener con las RQTs, ni con las RTINs.



a) b) c) d)

Figura 1.19: Variaciones logradas con el esquema “bitree”.

De la clase de técnicas que utilizan árboles binarios de triángulos para modelar los datos de superficies, ROAM[12] [Duchaineau y col., 1997] se encuentra entre las más acep-

[11] Triangulación sobre Bitrees no Restringidos (traducido de las siglas en inglés NRBT: Non-restricted Bitree Triangulation).

[12] Real-time Optimally Adapting Meshes.

tadas por la comunidad científica. De forma similar a [Evans y col., 2001], la jerarquía se construye realizando divisiones sucesivas partiendo de un triángulo ubicado como raíz de la estructura, Figura 1.20.

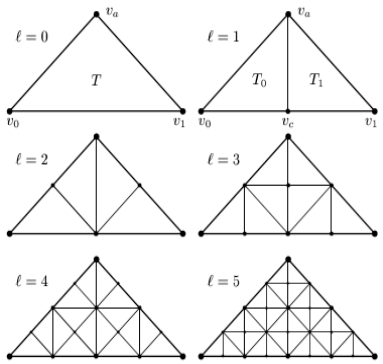


Figura 1.20: Esquema de creación de un árbol binario de triángulos.

Una vez modelada la superficie, las mallas a visualizar son representadas por dos colas de prioridad que permiten llevar a cabo operaciones de división y mezcla, Figura 1.21, sobre los triángulos. De esta forma se pueden obtener nuevas triangulaciones a partir de otras precedentes de forma sencilla y eficiente.

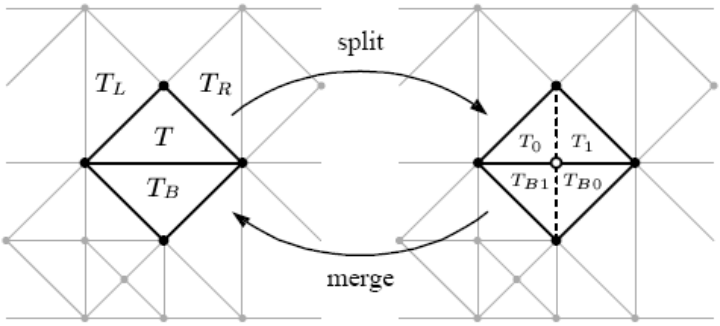


Figura 1.21: División (split) y mezcla (merge) de triángulos en una ROAM.

1.3. Combinación de técnicas

En secciones anteriores han sido analizadas varias técnicas que se caracterizan tanto por su regularidad, como por la irregularidad en las triangulaciones multiresolución resultantes de su aplicación. [Pajarola y col., 2002] realizan una propuesta de estructura de datos espacial, denominada QuadTIN, que combina las ventajas de los métodos de representación de superficies basados en TINs, con los modelos regulares construidos haciendo uso de Quadtrees, Figura 1.22.

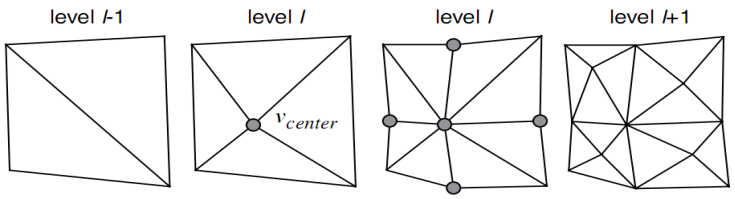


Figura 1.22: Representación por niveles de un QuadTIN.

La creación del QuadTIN está motivada por el hecho de que aún cuando se cuenta con conjuntos de datos en forma de mallas rectangulares regulares, la aplicación de algoritmos de análisis sobre éstos podría descartar una serie de puntos que, quizás por la naturaleza llana de la superficie, no brindan información de interés para el modelo. Por tal motivo, la "regularidad" de los conjuntos de datos pudiera ser sustituida por una versión irregular simplificada de los mismos.

Otras técnicas más recientes [Röttger y Ertl, 2001, Schneider y Westermann, 2006, Livny y col., 2009, Clasen y Hege, 2006] aprovechan las características del hardware, específicamente la Unidad de Procesamiento Gráfico (GPU[13]), para acelerar el proceso de rendering. En general la tendencia es a realizar en el GPU gran parte de los cálculos de las mallas a visualizar. Este tipo de variantes se encuentra en constante evolución dada la velocidad de desarrollo, tanto de hardware como de software, de los dispositivos dedicados al procesamiento gráfico. Debido a la gran facilidad de extensión de librerías como `OpenGL`, los fabricantes adicionan nuevas funcionalidades que optimizan el rendering de primitivas y contribuyen de esta forma a mejorar sustancialmente la calidad de visualización, pues mayor cantidad de información puede ser procesada para brindar mayor exactitud en los modelos.

La técnica BDAM[14] presentada por Cignoni y col. en [Cignoni y col., 2003b] pertenece a la clase que combina TINs con estructuras regulares. BDAM propone el uso de un Quadtree para la representación multinivel de texturas, y un par de árboles binarios para el manejo de pequeños conjuntos de triángulos. Estos conjuntos de triángulos son modelados mediante TINs que son construidas y optimizadas en tiempo de preprocesamiento, Figura 1.23. Otra importante contribución de BDAM consiste en su modelo de comunicación CPU[15]/GPU que impide que se haga un uso intensivo del primero de estos, aprovechando al máximo el potencial de cálculo de las GPU actuales. Si bien BDAM está diseñado para ser usado localmente en una PC, P-BDAM[16][Cignoni y col., 2003a]

[13]Unidad de Procesamiento Gráfico.
[14]Batched Dynamic Adaptive Meshes.
[15]Unidad Central de Procesamiento.
[16]Planet-sized Batched Dynamic Adaptive Meshes.

propone un novedoso algoritmo de simplificación de superficies que funcione de manera distribuida en una red estándar de computadoras. Esta técnica representa la evolución de BDAM hacia el trabajo con conjuntos de datos "esféricos" representativos de planetas. C-BDAM[17][Gobbetti y col., 2006] es el más reciente de esta serie de trabajos, y constituye una generalización de los dos anteriores en el sentido de que introduce conceptos que optimizan el desempeño de sus funcionalidades fundamentales.

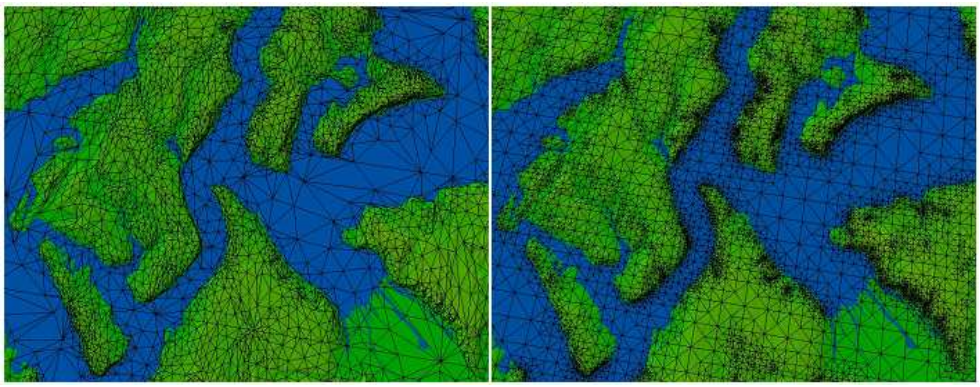


Figura 1.23: Vistas de una superficie modelada mediante BDAM (izquierda) y un modelo regular [Lindstrom y col., 1996](derecha).

1.4. Curvas de recorrido del espacio

Intuitivamente, una curva continua en 2, 3 o más dimensiones, puede ser vista como el camino descrito por el movimiento continuo de un punto. La Figura 1.24 muestra algunos ejemplos, conocidas como curvas de recorrido del espacio (SFC[18]). Dado que uno de los objetivos de este tipo de curvas es la preservación de la proximidad espacial, se utilizan frecuentemente en el diseño de algoritmos para la realización de búsquedas (ej. búsqueda de vecindades) en el campo de las estructuras de datos espaciales [Chen y Chang, 2005, Agranov y Gotsman, 1995].

Según [Asano y col., 1997], dada una malla rectangular regular M de orden $N \times N$, con $N = 2^n, n \geq 0$, se define SFC como una numeración de las celdas de la malla con valores desde $c+1$ hasta $c+N^2$, para algún $c \geq 0$. Una numeración P de M consiste en una correspondencia uno a uno de la forma:

$$P : N \times N \rightarrow \{1, \ldots, N^2\} \tag{1.1}$$

[17]Compressed Batched Dynamic Adaptive Meshes.
[18]Curva de recorrido del espacio (traducido de las siglas en inglés SFC: Space-filling Curve).

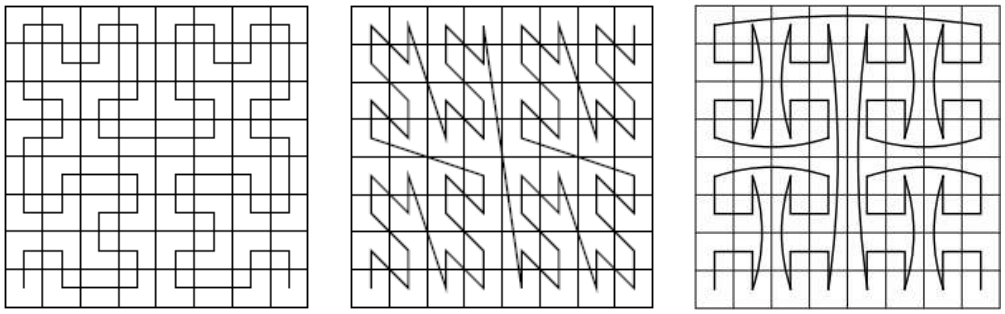


Figura 1.24: De izquierda a derecha: Curva de Hilbert, Orden Z y "Gray code".

Una SFC se denomina recursiva (RSFC[19]) si puede ser dividida en cuatro RSFCs cuadradas de igual tamaño [Asano y col., 1997]. Tal es el caso de las curvas de la Figura 1.24, que por sus características jerárquicas pueden ser generadas siguiendo reglas de permutación muy regulares. A partir de esta definición en el contexto de este trabajo se demuestra la utilidad de la curva de Hilbert en el diseño de estructuras de datos espaciales, específicamente en el caso de los Quadtree.

1.5. Conclusiones parciales

Tanto las técnicas de modelación y visualización de MDEs en ambientes tridimensionales basadas en TINs, como las concebidas para mallas rectangulares regulares, se caracterizan por su efectividad y eficiencia en los resultados que ofrecen. Las propuestas que combinan el uso de ambos criterios han permitido jerarquizar las TINs para mejorar los tiempos de respuesta y la calidad de las imágenes resultantes del proceso de rendering. La tabla 1.1 relaciona los trabajos que han marcado los hitos más significativos en materia de modelación y visualización de superficies de terrenos en tres dimensiones en los últimos años.

A pesar de la disponibilidad elevada de conjuntos de datos en forma de mallas rectangulares regulares, no siempre estos son adecuados para la representación de determinadas superficies. Si bien las propuestas de modelos regulares jerárquicos representados mediante Quadtrees o árboles binarios de triángulos han demostrado ser muy eficientes, la naturaleza restringida de muchos de éstos provoca que sean utilizados polígonos adicionales, para mantener la continuidad de las superficies, en el proceso de rendering de las triangulaciones multiresolución. La Figura 1.23 ilustra como con las TINs, en contraste con

[19] Curva de recorrido del espacio recursiva(traducido de las siglas en inglés RSFC: Recursive Space-filling Curve).

Tabla 1.1: Estudios sobre modelación multiresolución de superficies abiertas.

Autor	**Trabajo**
[Lindstrom y col., 1996]	Real-time Continuous Level of Details Rendering of Height Fields
[Hoppe, 1996]	Progressive meshes
[Duchaineau y col., 1997]	ROAMing terrain: Real-Time Optimally Adapting Meshes
[Röttger y col., 1998]	Real-Time Generation of Continuous Level of Details for Height Fields
[Pajarola, 1998]	Large Scale Terrain Visualization using the Restricted Quadtree Triangulation
[Floriani y col., 2000]	VARIANT: A System for Terrain Modeling at Variable Resolution
[Evans y col., 2001]	RTINs: Right-Triangulated Irregular Networks
[Lindstrom y Pascucci, 2002]	Terrain Simplification Simplified: A General Framework for View-Dependent Out-of-Core Visualisation
[Pajarola y col., 2002]	QuadTIN: Quadtree based Triangulated Irregular Networks
[Cignoni y col., 2003b]	BDAM - Batched Dynamic Adaptive Meshes
[Cignoni y col., 2003a]	Planet-sized Batched Dynamic Adaptive Meshes (P-BDAM)
[Schneider y Westermann, 2006]	GPU-friendly high-quality terrain rendering
[Gobbetti y col., 2006]	C-BDAM - Compressed Batched Dynamic Adaptive Meshes

los modelos regulares, la transición de una región llana a una más accidentada se lleva a cabo sin restricciones.

Capítulo 2

Modelación

La representación abstracta, conceptual, gráfica y/o matemática, entre otras, de fenómenos, sistemas o procesos constituye un elemento esencial e inseparable de toda actividad científica. A los efectos de este trabajo, el término modelación se refiere al proceso de representación espacial en memoria de un MDE haciendo uso de un Quadtree. Este tipo de estructura se aplica generalmente al análisis de imágenes, las cuales se descomponen en cuatro cuadrantes de un mismo tamaño, donde si estos no son uniformes, es decir, si están ocupados parcialmente por la imagen que se está analizando, se subdividen en cuatro cuadrantes, y así sucesivamente.

En el presente capítulo se explica detalladamente el proceso de construcción de los modelos poligonales propuestos para realizar la visualización interactiva de las superficies de terrenos. En primer lugar, se propone una variante de representación cuyo objetivo fundamental consiste en reducir al máximo la cantidad de primitivas a enviar al sistema gráfico para su visualización. Posteriormente, se presenta una técnica cuya prioridad la constituye la eficiencia de representación en memoria de los modelos, así como la velocidad de extracción de triangulaciones multiresolución a partir de los mismos.

2.1. Modelación de terrenos mediante Quadtrees

La construcción de un Quadtree para las técnicas que se proponen en el presente documento, se llevará a cabo a partir de una malla rectangular regular de medidas de altura de una superficie de terreno representada mediante un arreglo bidimensional, Figura 2.1. De igual forma que en el algoritmo descrito por [de Berg y col., 2000], los vértices extremos de la malla, comenzando por el vértice superior izquierdo, serán: $(0, 0)$, $(0, U)$, (U, U), $(U, 0)$, donde $U = 2^n$, para algún entero no negativo n. Como los índices se toman

a partir de 0 hasta U, la malla rectangular regular será de orden $(2^n + 1) \times (2^n + 1)$.

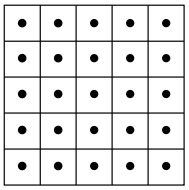


Figura 2.1: Malla rectangular regular.

El término "regular" de la malla viene dado por el hecho de que existen $j, k \in \mathbb{R}$, $j, k > 0$, tal que la distancia entre cualesquiera dos puntos consecutivos en una misma fila o en una misma columna son j y k respectivamente. De esta forma, el valor de las coordenadas (x, y) de cualquier punto vendrá dado por $(j * f + x_0, k * c + y_0)$, donde f y c son la fila y la columna respectivamente del punto en cuestión, y x_0, y_0 las coordenadas del punto extremo superior izquierdo de la malla.

El cuadrante base -o raíz- del Quadtree está determinado por los puntos en las posiciones $(0, 0)$, $(0, U)$, (U, U), $(U, 0)$. El proceso de división recursivo comienza tomando el punto medio $(M = \frac{U}{2} = 2^{n-1})$ de la malla y creando cuatro cuadrantes, Figura 2.2 a). Cada uno de ellos a su vez es subdividido en cuatro, Figura 2.2 b), hasta alcanzar la precisión deseada.

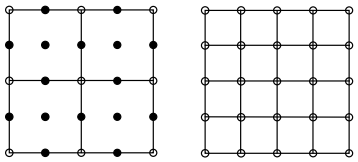


Figura 2.2: Proceso de partición: a) primer paso, b) segundo paso.

Si enumeramos los cuadrantes comenzando por el que se encuentra en el extremo superior izquierdo, y continuamos en el sentido de las manecillas del reloj, las coordenadas de cada uno de sus vértices serían las que se ilustran en la Figura 2.3.

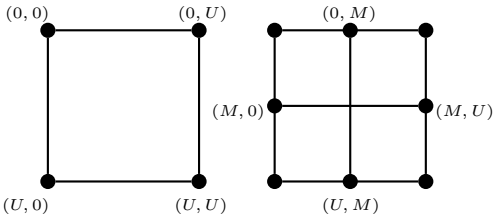


Figura 2.3: Coordenadas de los vértices de los cuadrantes.

2.2. Triangulación Quadtree-Bitree

En la implementación computacional clásica de un Quadtree [de Berg y col., 2000], el almacenamiento de la información se realiza mediante el uso de nodos y referencias. Durante el proceso recursivo de construcción para cada nodo se almacenan referencias a nodos hijos y padres. Posteriormente un algoritmo de búsqueda establece, para cada nodo, cuatro referencias a sus nodos vecinos. Una variante muy utilizada consiste en almacenar la estructura en un arreglo unidimensional donde -comenzando por la posición 0- para cada nodo en la posición k, sus descendientes se almacenan en las posiciones $4k+1$, $4k+2$, $4k+3$, $4k+4$. De esta forma se elimina la necesidad de crear referencias a hijos y padres dado que las posiciones de estos pueden ser obtenidas mediante simples operaciones matemáticas y de índices, aumentando de esta forma la eficiencia en la representación en memoria de los modelos.

Una vez que se ha concluido el proceso de división de la malla, se ha construido una estructura jerárquica semejante a la que se muestra en la Figura 2.4, donde los cuadrantes almacenados en cada nodo hoja tienen dimensiones $j \times k$.

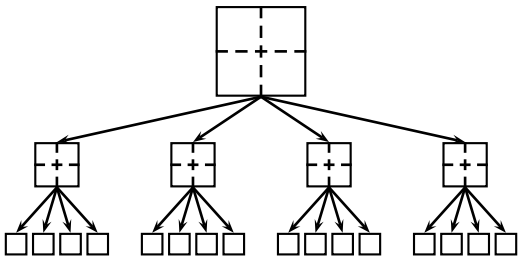


Figura 2.4: Esquema de un Quadtree completo.

Un Quadtree construido sobre una malla rectangular regular de orden $(2^n+1)\times(2^n+1)$ tiene $n+1$ niveles, tomando como nivel 0 el de la raíz. Los cuadrantes almacenados en los nodos hojas conforman una subdivisión a máxima resolución de la superficie (4^n cuadrantes en total). Una primera variante podría ser triangular y visualizar todos estos cuadrantes, obteniéndose una vista del modelo en su máximo nivel de detalle. Suponiendo que se dispone de datos de una superficie almacenados en un arreglo bidimensional de orden 513×513, una vez confeccionado el Quadtree tendría en su último nivel $4^9 = 262144$ nodos con sus respectivos cuadrantes, lo que representa una gran cantidad de polígonos para ser visualizados por un motor gráfico con la inmediatez requerida.

2.2.1. Simplificación geométrica

Diversos son los criterios a tener en cuenta para controlar la simplificación geométrica de los modelos, permitiendo la extracción de triangulaciones con diferentes niveles de resolución. Teniendo en cuenta la posición P del observador, se calculan las distancias entre P y la superficie a visualizar, o el ángulo entre la normal asociada a la superficie y la dirección de la vista. De esta forma regiones lejanas del observador, o con gran inclinación con respecto a la dirección de la vista, son representadas con menor resolución. Otros modelos, como el Bitree Geométrico-Texturado [Abásolo y Perales, 2003], basan la selección de puntos en los coeficientes resultantes de una transformada Wavelet luego de realizar un análisis multiresolución sobre la superficie original.

2.2.2. Selección de niveles de resolución

Varias de las propuestas expuestas en la literatura utilizan métricas basadas en términos de la distancia Euclidiana [Abásolo y col., 2000, Lindstrom y col., 1996, Röttger y col., 1998, Pajarola, 1998, Perez y col., 2004], donde el modelo simplificado se encuentra dentro de una distancia determinada con respecto al original. De esta forma un cuadrante c de un nodo puede sustituir los cuadrantes c_h de sus hijos si el error de representación de c con respecto a cada c_h se encuentra dentro de un márgen (umbral) predeterminado. Una vez calculado y almacenado el error ϵ de representación en cada cuadrante, una malla multiresolución puede extraerse del modelo tomando en cuenta a P y proyectando a ϵ en la pantalla mediante la ecuación:

$$\rho(\epsilon) = \lambda \frac{\epsilon}{d} \tag{2.1}$$

donde λ es una constante [Lindstrom y Pascucci, 2002] y d es la distancia entre P y la posición de un cuadrante determinado. La ecuación de proyección 2.1 se denomina isotrópica puesto que el error proyectado es el mismo en todas las direcciones alrededor de P, sin tener en cuenta la dirección de visualización. Este efecto se puede amortiguar aplicando distintos niveles de calidad de visualización dentro y fuera de la región del modelo que aparecerá en pantalla (view frustum), Figura 2.5.

2.2.3. Triangulación

Como se ha podido apreciar en la sección 1.3, tanto las RQTs como las NRQTs consisten en criterios de triangulación de las superficies modeladas mediante Quadtrees. Como primer resultado de este trabajo [Valle, 2007a, Valle, 2007b], con el objetivo de enviar al

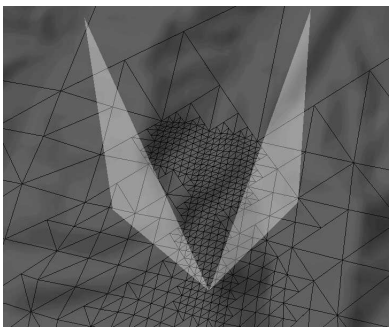


Figura 2.5: Vista de una triangulación multiresolución.

sistema gráfico la menor cantidad de primitivas para su visualización interactiva, se propone la variante de triangulación QBT[1] que combina las NRQTs con las NRBTs, Figura 2.6.

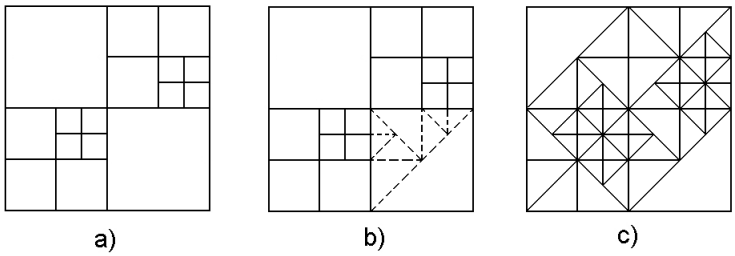


Figura 2.6: Triangulación "bitree" sobre un Quadtree no restringido.

De igual forma que en las NRQTs, la selección en el Quadtree de los cuadrantes a visualizar se lleva a cabo con independencia de los vecinos, garantizando de esta forma la eliminación de la restricción de balance impuesta por las RQTs, Figura 2.6 a). Para obtener la QBT sencillamente se procede a triangular cada cuadrante seleccionado, Figura 2.6 b), según el criterio de triangulación utilizado por las NRBTs, Figura 1.18. La Figura 2.6 c) muestra una posible QBT obtenida sobre un Quadtree.

La formación de triángulos según el criterio propuesto por las NRBTs puede presentar un problema conocido como `Formación-T`[2]. Las API de creación de gráficos por computadora, como `OpenGL`, en la fase de rasterización utilizan técnicas de discretización para representar líneas. Es por ello que los puntos medios de las hipotenusas seleccionados en el proceso de creación de los triángulos, no necesariamente coinciden con los que determina la técnica de discretización que utilice el motor gráfico, por lo que pueden producirse discontinuidades en la superficie triangulada.

[1]Triangulación Quadtree-Bitree (traducido de las siglas en inglés QBT: Quadtree-Bitree Triangulation).

[2]Traducido del término en inglés T-junction.

2.3. Triangulación Hilbert-Quadtree

A pesar de la complejidad en la manipulación de las RSFCs, sección 1.4, su naturaleza jerárquica permite que sean ampliamente utilizadas en el diseño de estructuras de datos espaciales, Figura 2.7. En esta sección se demuestra que esta curva resulta muy útil en la modelación de superficies de terrenos y en la generación de cadenas de triángulos para su posterior visualización eficiente.

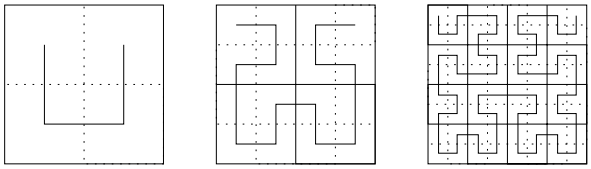


Figura 2.7: Niveles 1, 2 and 3 de la curva de Hilbert.

Para modelar un MDE mediante una curva de Hilbert [Valle, 2008, Valle, 2009], el Quadtree será implementado haciendo uso de operaciones de cálculo de índices y recursividad, como en [Pajarola, 1998], en lugar de referencias y nodos como en una implementación clásica. En la estructura de datos propuesta, en lo adelante `HQuadtree`, Figura 2.8, no resulta necesario almacenar información sobre nodos hijos, padres, ni vecinos.

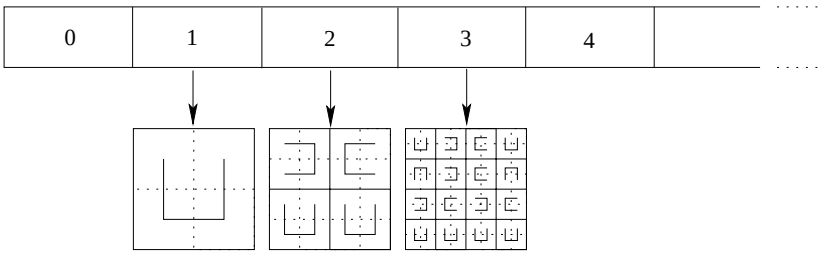


Figura 2.8: Primeros tres niveles en la estructura de datos.

El punto de partida, de igual forma que en el modelo anterior, es una malla rectangular regular de orden $(2^n+1)\times(2^n+1)$, $n \geq 1$, la cual es recursivamente dividida en cada paso del algoritmo. Cada nodo del Quadtree es representado como un segmento de la curva de Hilbert, en lo adelante "patrón de Hilbert", Figura 2.9.

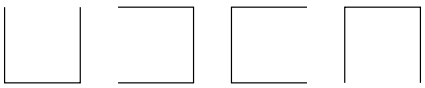


Figura 2.9: Cuatro patrones de Hilbert: `Abajo`, `Derecha`, `Izquierda`, `Arriba`.

2.3.1. Estructura de datos

Como se muestra en la Figura 2.8, el Quadtree es almacenado en un arreglo unidimensional que en cada posición almacena un arreglo bidimensional de patrones de Hilbert.

Cada índice en el arreglo unidimensional representa un nivel -desde 1 hasta la cantidad de niveles, la posición 0 permanece vacía- en el Quadtree, y cada arreglo bidimensional almacena la información correspondiente a los nodos en el correspondiente nivel. El algoritmo 1 muestra como el proceso de creación es llevado a cabo.

Algorithm 1 Creación del Quadtree

```
1: procedure CrearQuadtree
2:     niveles ← ⌊log2 n⌋
3:     quadtree ← matriz[niveles + 1]
4:     for i ← 1, niveles do
5:         quadtree[i] ← matriz[2^(i−1), 2^(i−1)]
6:     end for
7:     quadtree[1][0, 0] ← Datos(Hilbert.Abajo, Dir.Este)
8:     DivideAbajo(⌊n/2⌋, ⌊n/2⌋, ⌊n/2⌋, 1)
9: end procedure
```

El procedimiento CrearQuadtree comienza calculando el número de niveles con el objetivo de crear el arreglo unidimensional. Las líneas de la 4 a la 6 establecen el arreglo bidimensional correspondiente a cada nivel. Posteriormente el patrón de Hilbert inicial, Figura 2.7, es creado y almacenado en la matriz de orden (1×1) en el nivel 1, junto con la dirección a seguir por la curva de Hilbert para la posterior generación de la cadena de triángulos a visualizar (por ejemplo: la dirección del patrón de Hilbert noroeste en la Figura 2.7 en el nivel 2 es Sur). Finalmente, en la línea 8, se inicia el proceso de división recursiva realizando un llamado al algoritmo 2.

El procedimiento DivideAbajo recibe cuatro parámetros: las dimensiones del cuadrante que será creado en el próximo nivel, y la fila, columna y nivel del cuadrante padre. En este procedimiento, un patrón de Hilbert `Abajo` es divido en cuatro patrones que son almacenados en el próximo nivel. Antes de que se ejecuten las llamadas a los procedimientos en las líneas 10, 12 y 14, los valores de la fila y la columna en la malla y en la matriz correspondiente para los hijos suroeste, sureste y noroeste deben ser calculados. El cálculo es llevado a cabo de forma similar al código de las líneas 3 a la 6: $[f_{so} = f + \frac{dim}{2}, c_{so} = c_{no}]$, $[f_{se} = f_{so}, c_{se} = c + \frac{dim}{2}]$, y $[f_{ne} = f_{no}, c_{ne} = c_{se}]$. Los valores de f_m y c_m son actualizados para cada hijo. Los procedimientos DivideDerecha, DivideIzquierda y DivideArriba son muy similares a DivideAbajo.

Una ventaja importante de representar un Quadtree de esta forma es que cada patrón de Hilbert corresponde a un nuevo punto introducido en cada paso del algoritmo de división y representa cuatro cuadrantes, razón por la cual la información de los cuadrantes de máxima resolución no es almacenada explícitamente en la estructura. Ambos

Algorithm 2 Creación del Quadtree, proceso de división

```
1: procedure DivideAbajo(dim, f, c, nivel)
2:     if ⌊dim/2⌋ ≠ 0 then
3:         f_no ← f − dim/2                          ▷ Fila del hijo NO en la malla
4:         c_no ← c − dim/2                          ▷ Columna del hijo NO en la malla
5:         f_m ← ⌊f_no/dim⌋                          ▷ Fila del hijo NO en la matriz correspondiente
6:         c_m ← ⌊c_no/dim⌋                          ▷ Columna del hijo NO en la matriz correspondiente
7:         quadtree[nivel + 1][f_m, c_m] ← Datos(Hilbert.Derecha, Dir.Sur)
8:         DivideDerecha(dim/2, f_no, c_no, nivel + 1)
           ...
9:         quadtree[nivel + 1][f_m, c_m] ← Datos(Hilbert.Abajo, Dir.Este)
10:        DivideAbajo(dim/2, f_so, c_so, nivel + 1)
           ...
11:        quadtree[nivel + 1][f_m, c_m] ← Datos(Hilbert.Abajo, Dir.Norte)
12:        DivideAbajo(dim/2, f_se, c_se, nivel + 1)
           ...
13:        quadtree[nivel + 1][f_m, c_m] ← Datos(Hilbert.Izquierda, Padre.Dir)
14:        DivideIzquierda(dim/2, f_ne, c_ne, nivel + 1)
15:    end if
16: end procedure
```

atributos, el patrón de Hilbert y la dirección, pueden ser representados por valores del 0 al 3, necesitando solo 2 `bits` por atributo. Dada una posición de un patrón de Hilbert $h = [l_h][f, c]$ en la estructura, la correspondiente información del punto que representa en la malla regular de alturas puede ser calculada mediante simples operaciones matemáticas:

$$P_h = (f * 2^{l-l_h} + 2^{l-l_h-1}, c * 2^{l-l_h} + 2^{l-l_h-1}) \tag{2.2}$$

donde l es la cantidad de niveles de la estructura y l_h el nivel donde se encuentra el patrón. De forma similar pueden ser calculadas las coordenadas de los extremos de los cuatro cuadrantes representados por cada patrón en la estructura.

El procedimiento Padre en el algoritmo 3 retorna el patrón padre para un patrón de Hilbert en una posición válida $[nivel][f_m, c_m]$.

Algorithm 3 Patrón de Hilbert padre

```
1: procedure Padre(nivel, f_m, c_m)
2:     return quadtree[nivel − 1][⌊f_m/2⌋, ⌊c_m/2⌋]
3: end procedure
```

El procedimiento HijoSuroeste en el algoritmo 4 retorna el hijo suroeste para el patrón de Hilbert en la posición válida $[nivel][f_m, c_m]$. Los procedimientos HijoNoroeste,

HijoNoreste e HijoSureste son muy similares a HijoSuroeste.

Algorithm 4 Hijo suroeste

```
1: procedure HijoSuroeste(nivel, f_m, c_m)
2:     return quadtree[nivel + 1][f_m * 2 + 1, c_m * 2]
3: end procedure
```

El procedimiento VecinoNorte en el algoritmo 5 retorna el vecino norte para el patrón de Hilbert en la posición válida $[nivel][f_m, c_m]$. Los procedimientos VecinoSur, VecinoEste y VecinoOeste son muy similares a VecinoNorte.

Algorithm 5 Vecino Norte

```
1: procedure VecinoNorte(nivel, f_m, c_m)
2:     return quadtree[nivel][f_m − 1, c_m]
3: end procedure
```

Como se ilustra en los algoritmos expuestos, la búsqueda de padres, hijos y vecinos es llevada a cabo mediante el uso de simples operaciones matemáticas, por lo que no resulta necesario almacenar esa información en la estructura. Con respecto a los algoritmos fundamentales sobre este tipo de estructuras, dígase recorridos en profundidad y por niveles, localización, entre otros, la propuesta mantiene el orden de ejecución de los mismos, incluso mejora determinadas variantes que requieren de estructuras auxiliares en implementaciones clásicas de Quadtrees.

2.3.2. Generación de cadenas de triángulos

La velocidad a la que pueden ser visualizadas las superficies trianguladas resulta decisiva en la mayoría de las técnicas de visualización científica [Arkin y col., 1994]. Una variante de triangulación de cuadrantes de una superficie modelada mediante un Quadtree es la obtención de abanicos de triángulos [Röttger y col., 1998]. Si bien esta variante optimiza el proceso de visualización, su principal limitación viene dada por el hecho de que para cada cuadrante a visualizar se debe enviar al sistema gráfico la lista de vértices pertenecientes al abanico generado.

La obtención de cadenas de triángulos a partir de curvas de recorrido del espacio ha sido ampliamente usada para generar con rapidez y de manera eficiente triangulaciones a partir de estructuras de datos jerárquicas [Lindstrom y Pascucci, 2002, Pajarola, 1998, Velho y col., 1999]. La técnica propuesta en esta sección, que denominaremos Triangulación Hilbert-Quadtree (HQT), está basada en la curva de Hilbert para generar estas

cadenas, y la información que se necesita para su construcción se encuentra almacenada explícitamente en el modelo.

Para extraer una HQT[3] [Valle, 2008, Valle, 2009], en el algoritmo 1 cada vez que se crea un patrón de Hilbert, un valor de error en el espacio objeto similar a [Perez y col., 2004] es calculado -por ejemplo después de la línea 8- y almacenado para ser usado posteriormente en la extracción de la triangulación con múltiples niveles de resolución. Debido al hecho de que el procedimiento recursivo de creación de la estructura siempre comienza con un patrón de Hilbert `Abajo`, el inicio de la curva de recorrido del espacio siempre se encuentra en la región noroeste, mientras que el final se localiza en el área noreste. Siguiendo la curva, para cada nodo en la estructura se adicionan sus vértices a la cadena de triángulos siguiendo ciertas reglas, Figura 2.10.

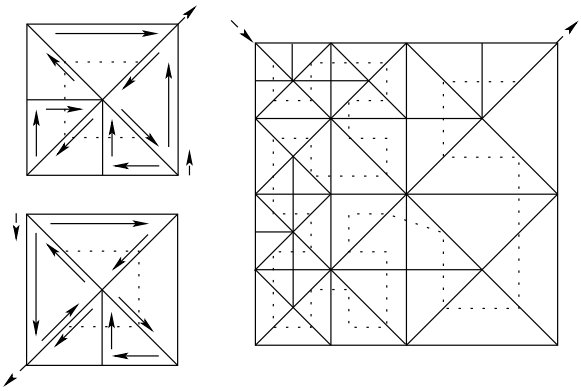


Figura 2.10: Izquierda: triangulación de los patrones de Hilbert orientados hacia la derecha y la izquierda. Derecha: superficie triangulada creada siguiendo una curva de Hilbert (líneas de puntos).

La triangulación de un patrón de Hilbert `Abajo` siempre comienza (termina) en el cuadrante superior-izquierdo (superior-derecho), y los patrones de Hilbert `Derecha` son triangulados comenzando en el cuadrante superior-izquierdo y terminando en el inferior-izquierdo. La triangulación de un patrón de Hilbert `Izquierda` siempre comienza (termina) en el cuadrante inferior-derecho (superior-derecho), mientras que los patrones de Hilbert `Arriba` son triangulados partiendo del cuadrante inferior-derecho y terminan en el inferior-izquierdo.

Los cuadrantes a ser visualizados son seleccionados de forma similar a [Röttger y col., 1998, Pajarola, 1998, Perez y col., 2004], garantizando que la diferencia de niveles entre cuadrantes adyacentes sea menor o igual a uno como en las RQTs. El

[3]Triangulación Hilbert-Quadtree (traducido de las siglas en inglés HQT: Hilbert-Quadtree Triangulation).

punto de partida de la curva de Hilbert es encontrado descendiendo recursivamente por la región noroeste del Quadtree hasta que un cuadrante indivisible es encontrado. Una vez encontrado el inicio, siguiendo las direcciones almacenadas en la estructura, cada cuadrante es triangulado en forma continua.

2.4. Conclusiones parciales

Durante la creación de una técnica de modelación y visualización de MDEs, se debe prestar especial interés al proceso de selección de estructuras de datos para la representación espacial de los mismos. En términos generales la estructura debe caracterizarse, entre otras propiedades, por:

- Brindar un acceso eficiente a la información espacial. Generalmente los algoritmos de localización de puntos o regiones tienen un costo $O(\log n)$, siendo n el total de nodos de la estructura. En este sentido el `HQuadtree` mantiene el costo de implementaciones anteriores.

- Permitir la localización de regiones vecinas a una dada en orden constante, o a lo sumo en $O(\log n)$[de Berg y col., 2000], garantizando de esta forma la navegación eficiente sobre superficies modeladas. A diferencia de sus similares, en la estructura propuesta la búsqueda de vecinos de igual tamaño de una región dada se realiza en $O(1)$, lo cual constituye una optimización importante en estructuras de datos espaciales.

- Requerir un mínimo de cantidad de memoria para su almacenamiento. Generalmente, en la implementación jerárquica de Quadtrees se almacenan apuntadores a nodos hijos, vecinos, entre otros, aumentando considerablemente su costo de almacenamiento. Como se ha podido apreciar, un `HQuadtree` solo necesita un `byte` por cada patrón puesto que las relaciones entre patrones hijos, vecinos y ancestros se manifiestan de manera implícita.

Tanto las métricas de control de propagación de errores, como los criterios de triangulación de las regiones en las estructuras, deben garantizar la obtención de forma eficiente de superficies multiresolución para ser visualizadas con la calidad e inmediatez requeridas. En este sentido, con la QBT se obtiene una cantidad de triángulos inferior a las RQTs y las NRQTs para un mismo valor de error prefijado, lo cual constituye un factor que influye directamente en las velocidades de visualización de las superficies.

Capítulo 3

Análisis de Resultados

Técnicas de modelación como las tratadas en la sección 1.3, han demostrado ser muy eficientes y flexibles en la visualización interactiva de superficies de terrenos en tres dimensiones. En el presente capítulo se exponen un conjunto de resultados obtenidos a partir de pruebas realizadas a las técnicas que han sido propuestas en el trabajo. La Figura 3.1 muestra una vista de una QBT construida sobre una malla rectangular regular de orden 257×257, demostrando su factibilidad de uso en el tratamiento del problema planteado. En total fueron visualizados 131072 triángulos, correspondientes a su máxima resolución.

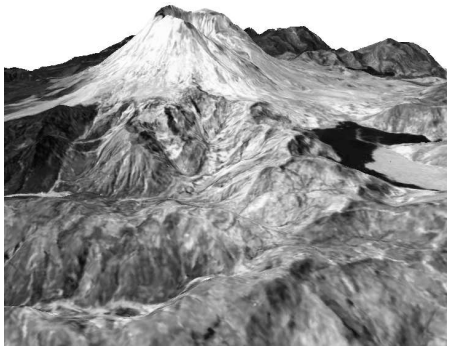


Figura 3.1: Vista del monte St. Helens, Washington, USA, con su correspondiente textura modelado mediante una QBT.

3.1. Herramienta de visualización

Para realizar las pruebas prácticas con las técnicas propuestas se desarrolló una herramienta simple de modelación y visualización de MDEs en tres dimensiones. Hasta el momento en que se escribe este documento, la herramienta cuenta con dos tipos fun-

damentales de proveedores: el proveedor de datos de altura y el proveedor de texturas, Figura 3.2.

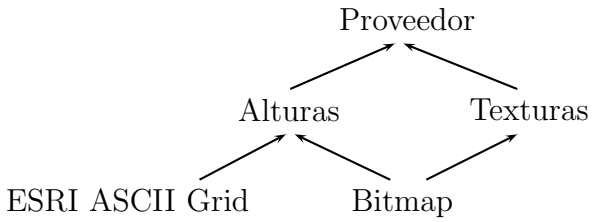


Figura 3.2: Esquema de proveedores de datos de la herramienta.

El proveedor de datos de altura es el encargado de leer de las distintas fuentes disponibles y generar un malla rectangular regular de orden $(2^n + 1) \times (2^n + 1)$, n entero no negativo. Como fuente de datos se usan ficheros ASCII Grid de ESRI[1] o imágenes en tonalidades de grises representativas de datos de altura, donde cada pixel representa un valor que se obtiene de su luminosidad. Como texturas se utilizan imágenes correspondientes a las que proporcionan los datos de relieve de las superficies.

3.2. Experimentación

Con el objetivo de mostrar la efectividad de los modelos propuestos para la representación y visualización eficiente de superficies de terrenos, algunas pantallas fueron tomadas de corridas de la aplicación en una simple Toshiba Satellite L20-273 con 512 MB de memoria RAM, un procesador Intel Celeron M360 a 1.40 GHz y una tarjeta gráfica ATI RADEON XPRESS 200M Series (0x5A62) de 64 MB. Para la visualización fue usada una métrica de error en el espacio objeto definida en términos de la distancia vertical entre puntos en cuadrantes en distintos niveles en la estructura, y datos de ejemplo pertenecientes a la regiones del Gran Cañón del Colorado y el área de Puget Sound[2] en Estados Unidos. Los datos de altura han sido ligeramente exagerados[3] con el propósito de destacar los cambios de elevación.

La tabla 3.1 ilustra una comparación realizada entre la técnica QBT propuesta en este documento, y las basadas en Quadtrees restringidos y no restringidos, de gran aceptación estas últimas por su flexibilidad y eficiencia. Puede observarse que para un mismo grado

[1]Empresa que desarrolla y comercializa software para Sistemas de Información Geográfica (ESRI: Enviromental Systems Research Institute).

[2]Disponible en http://www−static.cc.gatech.edu/projects/large_models

[3]Se refiere a un factor por el que se multiplican los datos de elevación originales para destacar o enmascarar sus características, según convenga.

de aproximación, las RQTs necesitan introducir mayor cantidad de polígonos que las NRQTs, y las QBT menor cantidad que estas últimas.

Tabla 3.1: Comparación de la cantidad de triángulos generados por la herramienta de visualización para mallas de diferentes tamaños y diferentes técnicas de modelación.

Error (%)	Cantidad de Triángulos					
	129×129			257×257		
	RQT	NRQT	QBT	RQT	NRQT	QBT
1	16558	14786	14431	72059	62755	60673
3	5737	4719	4493	21755	18242	17676
6	3260	2523	2398	9687	7559	7300
8	2123	1618	1526	6401	5096	4896
11	1370	1054	1004	4180	3186	3017
15	957	704	661	2286	1792	1725

En la gráfica de la Figura 3.3 puede observarse con mayor claridad, que las QBT resultan más eficientes que las RQTs y las NRQTs en cuanto a la cantidad de polígonos necesarios para visualizar una superficie de terreno con un grado de aproximación prefijado. Los datos fueron tomados de un modelo construido a partir de una malla rectangular regular de orden 257×257.

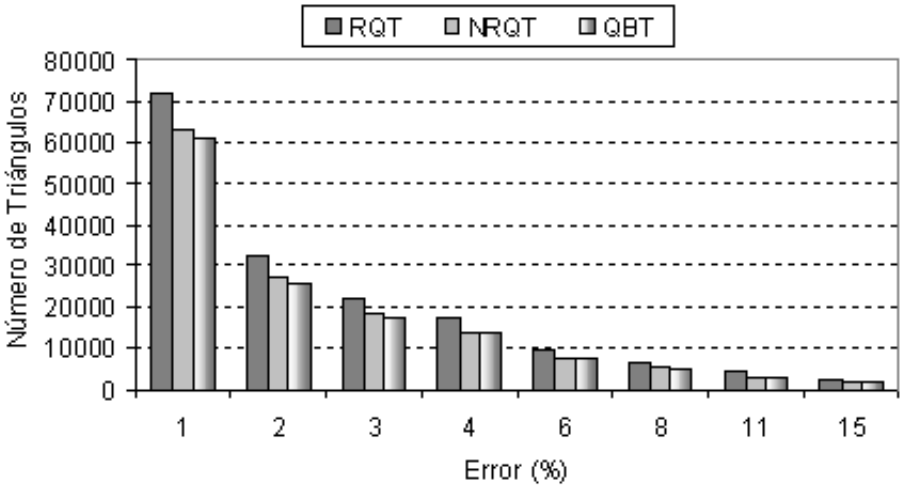


Figura 3.3: Comparación de la cantidad de triángulos que necesita cada técnica de modelación para alcanzar un grado de aproximación prefijado.

Aún con estos resultados obtenidos, el problema de la continuidad en la superficie planteado anteriormente y el alto grado de dificultad del algoritmo de generación de

la triangulación hacen que esta clase de técnicas no sea muy usada por la comunidad científica en el tratamiento del problema planteado.

La Figura 3.4 ilustra una triangulación multiresolución representativa de una reducida región del conjunto de datos original y su correspondiente curva de Hilbert. Puede observarse el carácter multiresolución de la curva en su recorrido generando la triangulación.

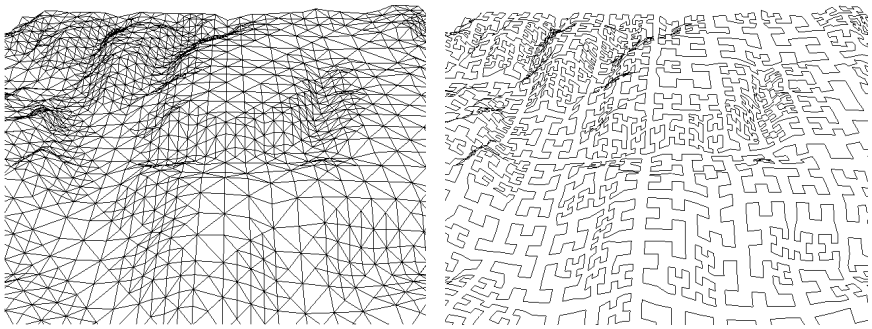


Figura 3.4: Fragmento de los datos originales del Gran Cañón: triangulación y su correspondiente curva de recorrido del espacio.

La Tabla 3.2 muestra la cantidad de puntos incluidos en la cadena de triángulos para un fragmento de tamaño 1025×1025 del conjunto de datos original del Gran Cañón con diferentes tolerancias de error. En lugar de $3N$ puntos necesarios para representar N triángulos, alrededor del 80 % son usados por la cadena de triángulos (TS).

Tabla 3.2: Puntos en la cadena de triángulos.

Error	0 metros	2 metros	5 metros	10 metros	20 metros
Triángulos	524288	503763	260529	138706	52229
$3N$ puntos	1572864	1511289	781587	416118	156687
Puntos en TS	1310720	1237622	633832	334273	125558

La Figura 3.5 ilustra paisajes ejemplo representados usando una HQT, mostrando superpuesta la triangulación y la correspondiente curva de Hilbert.

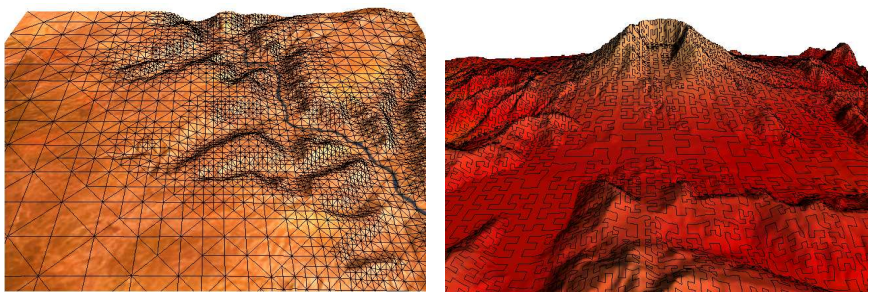


Figura 3.5: Ejemplos de paisajes: fragmentos de los datos originales de las regiones del Gran Cañón y Puget Sound.

En estos ejemplos ilustrativos los niveles de detalles claramente dependen de las características del relieve de las superficies. En un trabajo futuro debe considerarse la distancia del punto de vista del observador para la extracción de múltiples niveles de detalles, con el objetivo de reducir la cantidad de primitivas a representar en las visualizaciones.

De igual forma que varios trabajos anteriores, la principal contribución de esta investigación radica en la propuesta de una estructura de datos para incrementar la eficiencia en la representación en memoria del modelo digital de terreno. Sea M una matriz de orden $n \times n$, donde $n = 2^k + 1$, $k \geq 1$, que representa los datos de altura de una superficie de terrenos. Un Quadtree implementado mediante el uso de apuntadores y nodos construido sobre M tiene $k + 1$ niveles y

$$Q = \sum_{i=0}^{k} 2^i \tag{3.1}$$

nodos. Además de al menos 2 `bytes` por valor de altura en M, por cada nodo en el Quadtree se almacena información sobre vecinos, padre, e hijos. Por esta razón el costo mínimo aproximado de almacenamiento de un nodo en memoria viene dado por la ecuación:

$$S_n = s_p(d_q + v_q + f_q) + e_q \tag{3.2}$$

donde s_p es la cantidad de `bytes` que ocupa en memoria un apuntador, y d_q, v_q, f_q son el cantidad de descendientes, cantidad de vecinos y cantidad de padres ($f_q = 1$) respectivamente. El valor de e_q es real, por lo que se necesitan al menos 4 `bytes` para representarlo.

Debido a que en la estructura de datos propuesta para la HQT cada patrón de Hilbert representa 4 cuadrantes, los cuadrantes de mayor resolución no necesitan ser almacenados explícitamente. Por esta razón la cantidad de patrones en un `HQuadtree` es:

$$Q = \sum_{i=1}^{k} 2^{i-1} \tag{3.3}$$

Además de al menos 2 `bytes` por valor de altura en M, para cada patrón de Hilbert solo son necesarios 4 `bits` adicionales, 2 para identificar el tipo de patrón y 2 más para almacenar la dirección a seguir por la curva de recorrido del espacio. Sustituyendo los valores en las ecuaciones 3.1, 3.2 y 3.3 se concluye que la estructura de datos propuesta consume solamente el 6 % de la memoria necesaria para almacenar un Quadtree implementado con apuntadores y nodos.

Con respecto a la variante de utilizar un arreglo unidimensional para representar la estructura, la información de la localización de los nodos vecinos por cada cuadrante debe ser explícitamente almacenada, siendo de al menos 4 `bytes` por cada uno de los 4 vecinos. De lo contrario, en tiempo de ejecución localizar un vecino según los autores en [de Berg y col., 2000] es $O(\log n)$, por lo que el `HQuadtree` es superior también en ambos casos.

3.3. Conclusiones parciales

Debido a las limitaciones en la disponibilidad de memoria y velocidad de procesamiento que imponen los actuales sistemas, el diseño de técnicas de modelación y visualización de superficies de terrenos en ambientes tridimensionales debe caracterizarse por realizar un uso racional de estos recursos. El uso de recursividad y operaciones de cálculo de índices, en lugar de apuntadores y nodos, en la implementación de estructuras de soporte a los MDEs, garantiza que las relaciones jerárquicas en estas se manifiesten de forma implícita, factor que incide directamente en la disminución de los requerimientos de almacenamiento de los modelos.

Por otra parte, el orden de ejecución de los algoritmos diseñados para extraer triangulaciones multiresolución a partir de los modelos en memoria, debe ser proporcional a la cantidad de polígonos que las conforman. Satisfacer este criterio garantiza un adecuado aprovechamiento de las velocidades de cómputo de los sistemas, factor que contribuye positivamente en la eficiencia del proceso de rendering de las primitivas gráficas.

Conclusiones

- La representación de una superficie mediante un Quadtree, y la posterior extracción de la triangulación multiresolución a visualizar aplicando el criterio "bitree" a cada cuadrante de la estructura, reduce considerablemente la cantidad de polígonos a enviar al sistema gráfico para su procesamiento. Esta combinación también elimina la limitación impuesta por los Quadtrees restringidos sobre los niveles de división entre cuadrantes vecinos en la estructura, resultando en un aumento significativo de la eficiencia en la visualización.

- Aún con el problema de la `Formación-T`, la QBT ha demostrado ser altamente eficaz en el tratamiento del problema planteado cuando no se requiere de un alto grado de exactitud en las superficies modeladas.

- A partir del uso de la curva de recorrido del espacio de Hilbert en la implementación de un Quadtree, es posible reducir considerablemente la cantidad de memoria necesaria para modelar jerárquicamente una malla rectangular regular, a la vez que se acelera su proceso de visualización.

- Las relaciones entre cuadrantes hijos, padres y vecinos en un `HQuadtree` se manifiestan de manera implícita, y pueden ser calculadas eficientemente en tiempo de ejecución por lo que no es necesario almacenar explícitamente esta información en el modelo.

- Aún cuando los modelos teóricos garanticen que es suficiente almacenar un mínimo de información en memoria para manipular los MDEs, los mejores resultados se obtienen cuando se alcanza un equilibrio entre el costo de almacenamiento de las estructuras y el volumen de cálculos a realizar para su visualización en tiempo real.

Recomendaciones

- Incrementar la eficiencia del algoritmo de extracción de mallas poligonales con múltiples niveles de resolución en la QBT.
- Minimizar la cantidad de vértices que se envían al sistema gráfico para visualizar una HQT. Su algoritmo de generación de cadenas de triángulos emplea un 80 % de los $3N$ vértices necesarios para representar N triángulos, y aunque esto significa una optimización importante se considera que puede ser mejorado aún más.
- Implementar algoritmos de visibilidad jerárquica en superficies de terreno [Stewart, 1997], pues en un relieve accidentado se localizan regiones, incluso dentro del rango de visión del observador, que no son visibles porque son ocultadas por otras.

Publicaciones derivadas

Durante la investigación llevada a cabo en relación con la tesis presentada, se realizaron las siguientes publicaciones y presentaciones:

Modelo Multiresolución para Superficies de Terrenos en 3 Dimensiones
Yusnier Valle Martínez
III Congreso Internacional de Tecnologías, Contenidos Multimedia y Realidad Virtual, XII Convención y Expo Internacional Informática 2007.

Modelación y Triangulación de Superficies de Terrenos Mediante FTQuadtrees
Yusnier Valle Martínez
III Taller de Realidad Virtual, III Conferencia Científica de la Universidad de las Ciencias Informáticas (UCIENCIA 2007), Octubre de 2007.

Visualización de Superficies de Terrenos en Tres Dimensiones
Yusnier Valle Martínez
Proceedings of the 11th Iberoamerican Congress of Digital Graphics (SIGraDi 2007), México D.F. - México, 23-25 October 2007, pp. 257-260 [Online] http://cumincades.scix.net/cgi-bin/works/Show?sigradi2007_af45

An Efficient Restricted Quadtree Triangulation based on the Hilbert Space-filling Curve
Yusnier Valle Martínez
XVIII Congreso Español de Informática Gráfica (CEIG 2008), Eurographics Spanish Chapter, Barcelona, España, Septiembre de 2008, pp. 241-244.

An Efficient Quadtree Data Structure for Multiresolution Terrain Models
Yusnier Valle Martínez
Conferencia impartida en el *IV Taller de Entornos Virtuales*, IV Conferencia Científica

de la Universidad de las Ciencias Informáticas (UCIENCIA 2008), Octubre de 2008.

Implementación Eficiente de un Quadtree para la Modelación de Superficies de Terrenos
Yusnier Valle Martínez
VI Congreso Internacional de Geomática, XIII Convención y Feria Internacional Informática 2009.

Referencias bibliográficas

[Ope, 1999] (1999). *OpenGL Programming Guide: The Official Guide to Learning OpenGL*. Addison-Wesley Pub Co, 3rd edition.

[Abásolo y col., 1999] Abásolo, M. J., Blat, J., y de Giusti, A. (1999). A nonrestricted triangulation hierarchy for multiresolution terrain model. *Proceedings of the Spring Conference on Computer Graphics and its Applications, Budmerice, Eslovaquia*.

[Abásolo y col., 2000] Abásolo, M. J., de Giusti, A., y Blat, J. (2000). A hierarchical triangulation for multiresolution terrain models. *The Journal of Computer Science and Technology (JCS&T), Special Issue on Computer Science Research: State of the Art*, 1(3).

[Abásolo y Perales, 2003] Abásolo, M. J. y Perales, F. (2003). Wavelet analysis for a new multiresolution model for large-scale textured terrains. *The 11-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision,República Checa*.

[Agranov y Gotsman, 1995] Agranov, G. y Gotsman, C. (1995). Algorithms for rendering realistic terrain image sequences and their parallel implementation. *The Visual Computer*, 11(9):455–464.

[Aguilera y col., 2006] Aguilera, A., Feito, F., y Torres, J. (2006). Visualización de terrenos mediante niveles de detalle. In *XII Congreso Nacional de Tecnologías de la Información Geográfica*, pages 315–322, Granada, España.

[Angel, 2002] Angel, E. (2002). *Interactive Computer Graphics: A Top-Down Approach with OpenGL*. Pearson Addison Wesley, 3rd edition.

[Arkin y col., 1994] Arkin, E. M., Held, M., Mitchell, J. S. B., y Skiena, S. (1994). Hamiltonian triangulations for fast rendering. In *ESA '94: Proceedings of the Second Annual European Symposium on Algorithms*, pages 36–47, London, UK. Springer-Verlag.

[Asano y col., 1997] Asano, T., Ranjan, D., Roos, T., Welzl, E., y Widmayer, P. (1997). Space-filling curves and their use in the design of geometric data structures. *Theoretical Computer Science*, 181(1):3–15.

[Chen y Chang, 2005] Chen, H.-L. y Chang, Y.-I. (2005). Neighbor-finding based on space-filling curves. *Inf. Syst.*, 30(3):205–226.

[Cignoni y col., 2003a] Cignoni, P., Ganove lli, F., Gobbetti, E., Marton, F., Scopigno, R., y Ponchio, F. (2003a). Planet-sized batched dynamic adaptive meshes (p-bdam). In *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, page 20, Washington, DC, USA. IEEE Computer Society.

[Cignoni y col., 2003b] Cignoni, P., Ganovelli, F., Gobbetti, E., Marton, F., Ponchio, F., y Scopigno, R. (2003b). Bdam – batched dynamic adaptive meshes for high performance terrain visualization. *Computer Graphics Forum*, 22:505–514.

[Cignoni y col., 1997] Cignoni, P., Puppo, E., y Scopigno, R. (1997). Representation and visualization of terrain surfaces at variable resolution. *The Visual Computer*, 13(5):199–217.

[Clasen y Hege, 2006] Clasen, M. y Hege, H.-C. (2006). Terrain rendering using spherical clipmaps. pages 91–98, Lisbon, Portugal.

[Cohen-Or y Levanoni, 1996] Cohen-Or, D. y Levanoni, Y. (1996). Temporal continuity of levels of detail in delaunay triangulated terrain. In Yagel, R. y Nielson, G. M., editors, *IEEE Visualization '96*, pages 37–42.

[de Berg y col., 2000] de Berg, M., van Kreveld, M., Overmars, M., y Schwarzkopf, O. (2000). *Computational Geometry, Algorithms and Applications*. Springer, 2nd edition.

[Duchaineau y col., 1997] Duchaineau, M. A., Wolinsky, M., Sigeti, D. E., Miller, M. C., Aldrich, C., y Mineev-Weinstein, M. B. (1997). ROAMing terrain: real-time optimally adapting meshes. In *IEEE Visualization*, pages 81–88.

[Evans y col., 2001] Evans, W. S., Kirkpatrick, D. G., y Townsend, G. (2001). Right-triangulated irregular networks. *Algorithmica*, 30(2):264–286.

[Floriani y Magillo, 2002] Floriani, L. D. y Magillo, P. (2002). Regular and irregular multi-resolution terrain models: a comparison. In *GIS '02: Proceedings of the 10th ACM international symposium on Advances in geographic information systems*, pages 143–148, New York, NY, USA. ACM.

[Floriani y Magillo, 2003] Floriani, L. D. y Magillo, P. (2003). Triangle-based multiresolution models for height fields. In A. Cohen, J.-L. Merrien, L. S. e. N. P., editor, *Curve and Surface Fitting*, pages 97–106.

[Floriani y col., 2000] Floriani, L. D., Magillo, P., y Puppo, E. (2000). Variant: A system for terrain modeling at variable resolution. *Geoinformatica*, 4(3):287–315.

[Gobbetti y col., 2006] Gobbetti, E., Marton, F., Cignoni, P., Di Benedetto, M., y Ganove lli, F. (2006). C-bdam - compressed batched dynamic adaptive meshes for terrain rendering. *Computer Graphics Forum*, 25(3). To appear in Eurographics 2006 conference proceedings.

[Guedes, 1997] Guedes, L. (1997). Real-time terrain surface extraction at variable resolution. In *X Brazilian Symposium on Computer Graphics and Image Processing*, pages 87–94.

[Herzen y Barr, 1987] Herzen, B. V. y Barr, A. H. (1987). Accurate triangulations of deformed, intersecting surfaces. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 103–110, New York, NY, USA. ACM.

[Hoppe, 1996] Hoppe, H. (1996). Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, New York, NY, USA. ACM.

[Klein y Straßer, 1996] Klein, R. y Straßer, W. (1996). Generation of multiresolution models from cad-data for real time rendering. In Straßer, W., Klein, R., y Rau, R., editors, *Theory and Practice of Geometric Modeling*. Springer Verlag, Berlin, Heidelberg, New York.

[Lehmann, 1968] Lehmann, C. H. (1968). *Geometría Analítica*. Edición revolucionaria.

[Lindstrom y col., 1995] Lindstrom, P., Koller, D., Hodges, L. F., Ribarsky, W., Faust, N. L., y Turner, G. (1995). Level-of-detail management for real-time rendering of phototextured terrain. Gvu technical report; git-gvu-95-06.

[Lindstrom y col., 1996] Lindstrom, P., Koller, D., Ribarsky, W., Hodges, L., Faust, N., y Turner, G. (1996). Real-time continuous level of detail rendering of height fields. *Proceedings of SIGGRAPH'96*, pages 109–118.

[Lindstrom y Pascucci, 2002] Lindstrom, P. y Pascucci, V. (Julio 2002). Terrain simplification simplified: A general framework for view-dependent out-of-core visualization. *IEEE Transaction on Visualization and Computer Graphics*, 8(3):239–254.

[Livny y col., 2009] Livny, Y., Kogan, Z., y El-Sana, J. (2009). Seamless patches for gpu-based terrain rendering. *Vis. Comput.*, 25(3):197–208.

[Pajarola, 2002] Pajarola, R. (2002). Overview of quadtree based terrain triangulation and visualization. Technical report uci-ics tr 02-01.

[Pajarola y col., 2002] Pajarola, R., Antonijuan, M., y Lario, R. (2002). Quadtin: quadtree based triangulated irregular networks. In *VIS '02: Proceedings of the conference on Visualization '02*, pages 395–402, Washington, DC, USA. IEEE Computer Society.

[Pajarola, 1998] Pajarola, R. B. (1998). Large scale terrain visualization using the restricted quadtree triangulation. In Ebert, D., Hagen, H., y lly Rushmeier, H., editors, *IEEE Visualization '98*, pages 19–26.

[Perez y col., 2004] Perez, M., Olanda, R., y Fernandez, M. (2004). Visualization of large terrain using non-restricted quadtree triangulations. In Springer Berlin, H., editor, *Computational Science and Its Applications – ICCSA*, pages 671–681.

[Röttger y Ertl, 2001] Röttger, S. y Ertl, T. (2001). Hardware accelerated terrain rendering by adaptive slicing. In *VMV '01: Proceedings of the Vision Modeling and Visualization Conference 2001*, pages 159–168. Aka GmbH.

[Röttger y col., 1998] Röttger, S., Heidrich, W., y Slussallek, P. (1998). Real-time generation of continuous levels of detail for height fields. In *Proc. 6th Int. Conf. in Central Europe on Computer Graphics and Visualization*, pages 315–322.

[Sagarduy y Vilar, 1982] Sagarduy, M. S. y Vilar, R. R. (1982). *Geometría Analítica*. Pueblo y Educación.

[Samet, 1990a] Samet, H. (1990a). *Applications of spatial data structures: Computer graphics, image processing, and GIS*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[Samet, 1990b] Samet, H. (1990b). *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[Schneider y Westermann, 2006] Schneider, J. y Westermann, R. (2006). Gpu-friendly high-quality terrain rendering. *Journal of WSCG*, 14(1-3):49–56.

[Segal y Akeley, 2003] Segal, M. y Akeley, K. (2003). *The OpenGL Graphics System: A Specification (Version 1.5)*.

[Shreiner, 2004] Shreiner, D. (2004). *OpenGL 1.4 Reference Manual*. Addison-Wesley Professional.

[Stewart, 1997] Stewart, A. J. (Junio 1997). Hierarchical visibility in terrains. *Eurographics Rendering Workshop*, pages 217–228.

[Valle, 2007a] Valle, Y. (2007a). Modelo multiresolución para superficies de terrenos en 3 dimensiones. In *III Congreso Internacional de Tecnologías, Contenidos Multimedia y Realidad Virtual*, XII Convención y Expo Internacional, Informática 2007.

[Valle, 2007b] Valle, Y. (2007b). Visualización de superficies de terrenos en tres dimensiones. In *Proceedings of de 11th Iberoamerican Congress of Digital Graphics (SIGraDi 2007)*, pages 257–260, México D.F. - México.

[Valle, 2008] Valle, Y. (2008). An Efficient Restricted Quadtree Triangulation based on the Hilbert Space-filling Curve. pages 241–244, Barcelona, Spain. Eurographics Association.

[Valle, 2009] Valle, Y. (2009). Implementación eficiente de un quadtree para la modelación de superficies de terrenos. In *VI Congreso Internacional de Geomática*, XIII Convención y Feria Internacional, Informática 2009.

[van Kreveld, 1997] van Kreveld, M. J. (1997). Algorithms for triangulated terrains. In *Conference on Current Trends in Theory and Practice of Informatics*, pages 19–36.

[Velho y col., 1999] Velho, L., de Figueiredo, L. H., y Gomes, J. (1999). Hierarchical generalized triangle strips. *The Visual Computer*, 15(1):21–35.

Acrónimos

API Interfaz de Programación de Aplicaciones (traducido de las siglas en inglés API: Application Program Interface).

BDAM Batched Dynamic Adaptive Meshes.

C-BDAM Compressed Batched Dynamic Adaptive Meshes.

CPU Unidad Central de Procesamiento.

DCEL Lista Doblemente Enlazada de Aristas (traducido de las siglas en inglés DCEL: Double Connected Edge List).

ESRI Empresa que desarrolla y comercializa software para Sistemas de Información Geográfica (ESRI: Enviromental Systems Research Institute).

GPU Unidad de Procesamiento Gráfico.

HQT Triangulación Hilbert-Quadtree (traducido de las siglas en inglés HQT: Hilbert-Quadtree Triangulation).

MDE Modelo Digital de Elevaciones.

MDT Modelo Digital de Terrenos.

NRBT Triangulación sobre Bitrees no Restringidos (traducido de las siglas en inglés NRBT: Non-restricted Bitree Triangulation).

NRQT Triangulación sobre Quadtrees No Restringidos (traducido de las siglas en inglés NRQT: Non-Restricted Quadtree Triangulation).

P-BDAM Planet-sized Batched Dynamic Adaptive Meshes.

QBT Triangulación Quadtree-Bitree (traducido de las siglas en inglés QBT: Quadtree-Bitree Triangulation).

ROAM Real-time Optimally Adapting Meshes.

RQT Triangulación sobre Quadtrees Restringidos (traducido de las siglas en inglés RQT: Restricted Quadtree Triangulation).

RSFC Curva de recorrido del espacio recursiva(traducido de las siglas en inglés RSFC: Recursive Space-filling Curve).

RTIN Mallas Regulares R-Trianguladas (traducido de las siglas en inglés RTIN: Right Triangulated Irregular Networks).

SFC Curva de recorrido del espacio (traducido de las siglas en inglés SFC: Space-filling Curve).

TIN Redes Irregulares Trianguladas (traducido de las siglas en inglés TIN: Triangulated Irregular Network).

Anexos

Los Anexos a continuación brindan un conjunto de elementos básicos de geometría del espacio [Lehmann, 1968, Sagarduy y Vilar, 1982] necesarios en el proceso de concepción de los modelos, y en la implementación de una herramienta simple de prueba de las técnicas propuestas en el documento. Posteriormente se describen brevemente las principales características de `OpenGL`, el motor gráfico utilizado para el rendering en tres dimensiones de las primitivas en la herramienta desarrollada.

Finalmente se exponen un conjunto de fotografías tomadas de corridas de la herramienta a partir de conjuntos de datos extensos, apreciándose la calidad de los modelos multiresolución generados para áreas de bajo y alto grado de accidentalidad en el relieve.

Anexo A

Elementos básicos de geometría del espacio

Un sistema de ejes coordenados rectangulares en el espacio consta de tres ejes perpendiculares dos a dos (denominados ejes coordenados) con el mismo origen (Figura A.1).

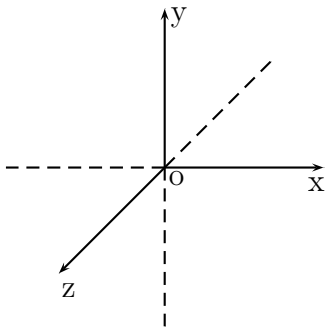


Figura A.1: Sistema de ejes coordenadas rectangulares

Cada par de ejes determina un plano que se denomina plano coordenado. Sea O el origen y OX, OY, OZ los ejes coordenados, entonces lo planos coordenados se denotan:

- XY el que determinan OX y OY.
- YZ el que determinan OY y OZ.
- ZX el que determinan OZ y OX.

El sistema se denota: XYZ, o bien: OXYZ.

A.1. El punto

Sea XYZ un sistema de coordenadas rectangulares en el espacio. Las coordenadas de un punto P en XYZ vienen dadas por el trio ordenado de números reales (x, y, z).

TEOREMA A.1.1 *La distancia d entre dos puntos $P_1(x_1, y_1, z_1)$ y $P_2(x_2, y_2, z_2)$ está dada por la fórmula:*

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{A.1}$$

A.2. El vector

DEFINICIÓN A.2.1 *Sea E el conjunto de todos los puntos del espacio. Se denomina vector a todo par ordenado de elementos de E.*

Si A y B son dos puntos del espacio, la notación (A,B) designa un vector. Se tiene además que (A,B) determina un segmento dirigido con origen en A y extremo en B, denotado por $\vec{AB}$. En caso que A y B coincidan, el vector que determinan se llamará vector nulo y se denotará $\vec{o}$.

Si $A(x_1, y_1, z_1)$ y $B(x_2, y_2, z_2)$ son dos puntos en el espacio, y OXYZ un sistema de coordenadas rectangulares, entonces las componentes de $\vec{AB}$ en OXYZ son:

$a = x_2 - x_1$, $b = y_2 - y_1$, $c = z_2 - z_1$

se denota $\vec{AB}$ como $\vec{AB} = (a, b, c)$.

DEFINICIÓN A.2.2 *Sean los vectores $\vec{AB} = (a_1, a_2, a_3)$ y $\vec{CD} = (b_1, b_2, b_3)$, se cumple que $\vec{AB} = \vec{CD}$ si y solo si:*

$a_1 = b_1, a_2 = b_2, a_3 = b_3$

DEFINICIÓN A.2.3 *Sean los vectores $\vec{AB} = (a_1, a_2, a_3)$ y $\vec{CD} = (b_1, b_2, b_3)$. Se denomina suma de los vectores $\vec{AB}$ y $\vec{CD}$, al vector $\vec{EF}$, cuyas componentes se obtienen sumando a las componentes del primer vector las del segundo:*

$\vec{EF} = (a_1 + b_1, a_2 + b_2, a_3 + b_3)$

DEFINICIÓN A.2.4 *Sean α un número real y $\vec{V} = (a_1, a_2, a_3)$ un vector. Se denomina producto de α y $\vec{V}$ al vector:*

$\alpha\vec{V} = (\alpha a_1, \alpha a_2, \alpha a_3)$

DEFINICIÓN A.2.5 *Sean* $\vec{V_1} = (a_1, a_2, ..., a_n)$ *y* $\vec{V_2} = (b_1, b_2, ..., b_n)$*, dos vectores de la misma cantidad de componentes. Se denomina producto escalar de* $\vec{V_1}$ *y* $\vec{V_2}$*, a la suma de los productos de las componentes del primer vector por las componentes respectivas del segundo:*

$$\vec{V_1} * \vec{V_1} = \sum_{i=1}^{n} a_i b_i \tag{A.2}$$

DEFINICIÓN A.2.6 *Sean* $\vec{V_1} = (a_1, b_1, c_1)$ *y* $\vec{V_2} = (a_2, b_2, c_2)$*, dos vectores cualesquiera de* R^3*, no paralelos y ninguno nulo. Se denomina producto vectorial de* $\vec{V_1}$ *y* $\vec{V_2}$*(en este orden) al vector:*

$$\vec{V_1} \times \vec{V_2} = \left(\begin{vmatrix} b_1 & c_1 \\ b_2 & c_2 \end{vmatrix}, \begin{vmatrix} c_1 & a_1 \\ c_2 & a_2 \end{vmatrix}, \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} \right) \tag{A.3}$$

El vector $\vec{V_1} \times \vec{V_2}$ *obtenido es ortogonal a* $\vec{V_1}$ *y a* $\vec{V_2}$*.*

DEFINICIÓN A.2.7 *Se denomina longitud o módulo del vector* $\vec{AB}$*, a la distancia (definida en A.1), entre sus puntos origen y extremo, y se denota* $|\vec{AB}|$*.*

A.3. El plano

De la geometría elemental se conoce que tres puntos no alineados en el espacio determinan un plano único π que los contiene. A continuación se define la ecuación cartesiana del mismo.

Sean $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ y $P_3(x_3, y_3, z_3)$ tres puntos no alineados en R^3. Se necesita además un vector normal a este plano [Sagarduy y Vilar, 1982]. Según la definición A.2.6, como P_1, P_2 y P_3 son no alineados, el vector $P_1\vec{P}_2 \times P_1\vec{P}_3$ es no nulo y ortogonal a $P_1\vec{P}_2$ y a $P_1\vec{P}_3$, luego es normal a π. Entonces la ecuación de π puede escribirse como sigue:

$$A(x - x_1) + B(y - y_1) + C(z - z_1) = 0$$

de donde:

$$Ax + By + Cz + D = 0, \qquad D = -Ax_1 - By_1 - Cz_1 \tag{A.4}$$

siendo A,B,C las componentes del vector $P_1\vec{P}_2 \times P_1\vec{P}_3$.

DEFINICIÓN A.3.1 *Sea π el plano de ecuación:*

$$A(x - x_0) + B(y - y_0) + C(z - z_0) = 0 \tag{A.5}$$

de vector normal $\vec{N} = (A, B, C)$, y que pasa por el punto $P_0(x_0, y_0, z_0)$, y sea $P_1(x_1, y_1, z_1)$ un punto no perteneciente a π. Se denomina distancia de P_1 a π, a la longitud del segmento de perpendicular trazada desde P_1 a π, y se calcula:

$$d(P_1, \pi) = \frac{|\vec{P_0P_1} * \vec{N}|}{|\vec{N}|} = \frac{A(x_1 - x_0) + B(y_1 - y_0) + C(z_1 - z_0)}{|\vec{N}|} \tag{A.6}$$

A.4. La recta

De la geometría elemental se conoce que dos puntos A y B distintos en el espacio determinan una recta única r que los contiene. A continuación se determina un sistema de ecuaciones paramétricas para r.

De la ecuación vectorial de la recta definida en [Sagarduy y Vilar, 1982], se tiene que:

$$\vec{P_0P} = \vec{V}t, \quad t \in R, \quad P_0(x_0, y_0, z_0) \quad y \quad \vec{V} = (a, b, c)$$

es una ecuación vectorial de r. Si $P(x, y, z)$, se tiene:

$$(x - x_0, y - y_0, z - z_0) = t(a, b, c)$$

que es equivalente al sistema:

$$\begin{aligned} x &= x_0 + at \\ y &= y_0 + bt \\ z &= z_0 + ct, \quad t \in R \end{aligned}$$

Recta determinada por dos puntos: Sean $P_0(x_0, y_0, z_0)$ y $P_1(x_1, y_1, z_1)$ dos puntos de R^3. un vector director [Sagarduy y Vilar, 1982] de r es $\vec{P_0P_1}$, por tanto, un sistema de ecuaciones para r será:

$$\begin{aligned} x &= x_0 + (x_1 - x_0)t \\ y &= y_0 + (y_1 - y_0)t \\ z &= z_0 + (z_1 - z_0)t, \quad t \in R \end{aligned}$$

TEOREMA A.4.1 *El ángulo θ formado por dos rectas cualesquiera dirigidas en el espacio, cuyos vectores directores tienen como componentes $[a_1, b_1, c_1]$ y $[a_2, b_2, c_2]$, respectivamente, está determinado por la relación*[1]*:*

$$\cos\theta = \pm\frac{a_1a_2 + b_1b_2 + c_1c_2}{\sqrt{a_1^2 + b_1^2 + c_1^2}\sqrt{a_2^2 + b_2^2 + c_2^2}} \tag{A.7}$$

[1]El doble signo de la ecuación indica que hay dos valores de θ, suplementarios entre sí [Lehmann, 1968].

Anexo B

OpenGL

`OpenGL` es una interfaz de software [Angel, 2002]. Consiste en un conjunto de distintos comandos, que se utilizan para especificar los objetos y operaciones necesarios para producir aplicaciones gráficas interactivas y portables tanto en dos (2D) como en tres (3D) dimensiones [Ope, 1999, Segal y Akeley, 2003, Shreiner, 2004]. Desde que fue introducida en 1992 por Silicon Graphics, se ha convertido en la Interfaz de programación de aplicaciones (API) gráficas en 2D y 3D más usada en la industria.

Entre las características más atrayentes de `OpenGL`, podemos citar:

- **Confiable y portable**: Todas las aplicaciones desarrolladas en `OpenGL`, producen resultados de visualización consistentes en cualquier plataforma que soporte su API.
- **Evolución**: Debido a su minucioso diseño, la API de `OpenGL` permite asimilar rápidamente las innovaciones que se producen en el hardware a través de su mecanismo de extensión. Esto permite a los desarrolladores de aplicaciones y proveedores de hardware, incorporar las nuevas características a sus productos.
- **Escalable**: Las aplicaciones que utilizan el API de `OpenGL`, se pueden ejecutar en una amplia gama de sistemas, que va desde las PC y estaciones de trabajo, hasta supercomputadoras.
- **Fácil de usar**: Posee una gran cantidad de primitivas muy eficientes, que encapsulan información sobre el hardware subyacente, liberando al desarrollador de tener que diseñar teniendo en cuenta la plataforma.
- **Amplia documentación**: El hecho de que numerosos libros que han sido publicados sobre `OpenGL` sean accesibles libremente, además de una gran cantidad de código de ejemplos, hace que la información sobre la misma sea fácil de obtener.

En los últimos años se han venido desarrollando varias librerías afines a `OpenGL`, entre las más conocidas podemos mencionar:

- `OpenGL Utility Library (GLU)`: Proporciona una serie de funciones que pueden ser utilizadas para llevar a cabo diversas tareas, tales como: actualizar matrices para obtener orientaciones y proyecciones del punto de vista adecuado, generar superficies, entre otras.
- `OpenGL Auxiliary Library (AUX)`: Cubren una gran variedad de funciones que pueden ser utilizadas a la hora de gestionar ventanas, eventos de entrada a la aplicación o en el dibujo de objetos complejos en 3D.
- `OpenGL Extension to the X Windows System (GLX)`: Utilizada para trabajar con el sistema de ventanas X Windows.

Anexo C

Modelos a partir de un HQuadtree

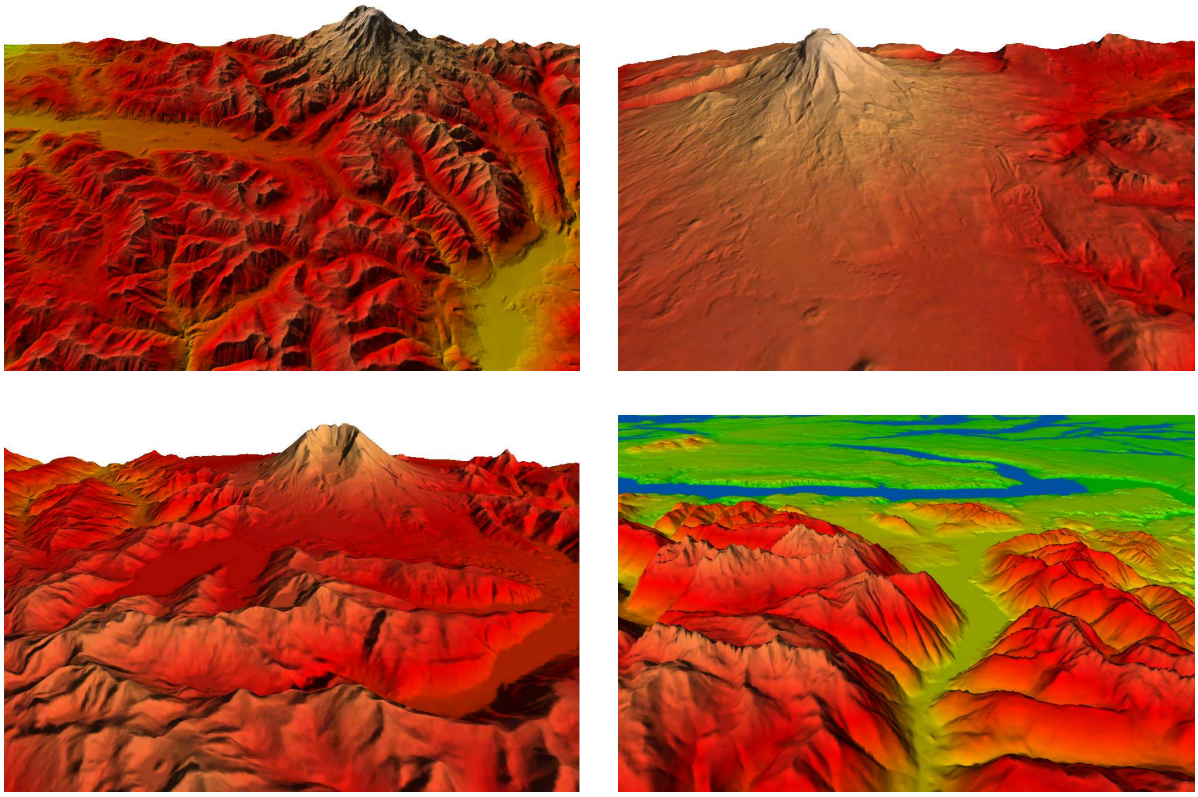


Figura C.1: Vistas del área de Puget Sound, Washington, USA. Los modelos son construidos sobre matrices de orden 2049×2049, procesándose más de 4 millones de puntos.

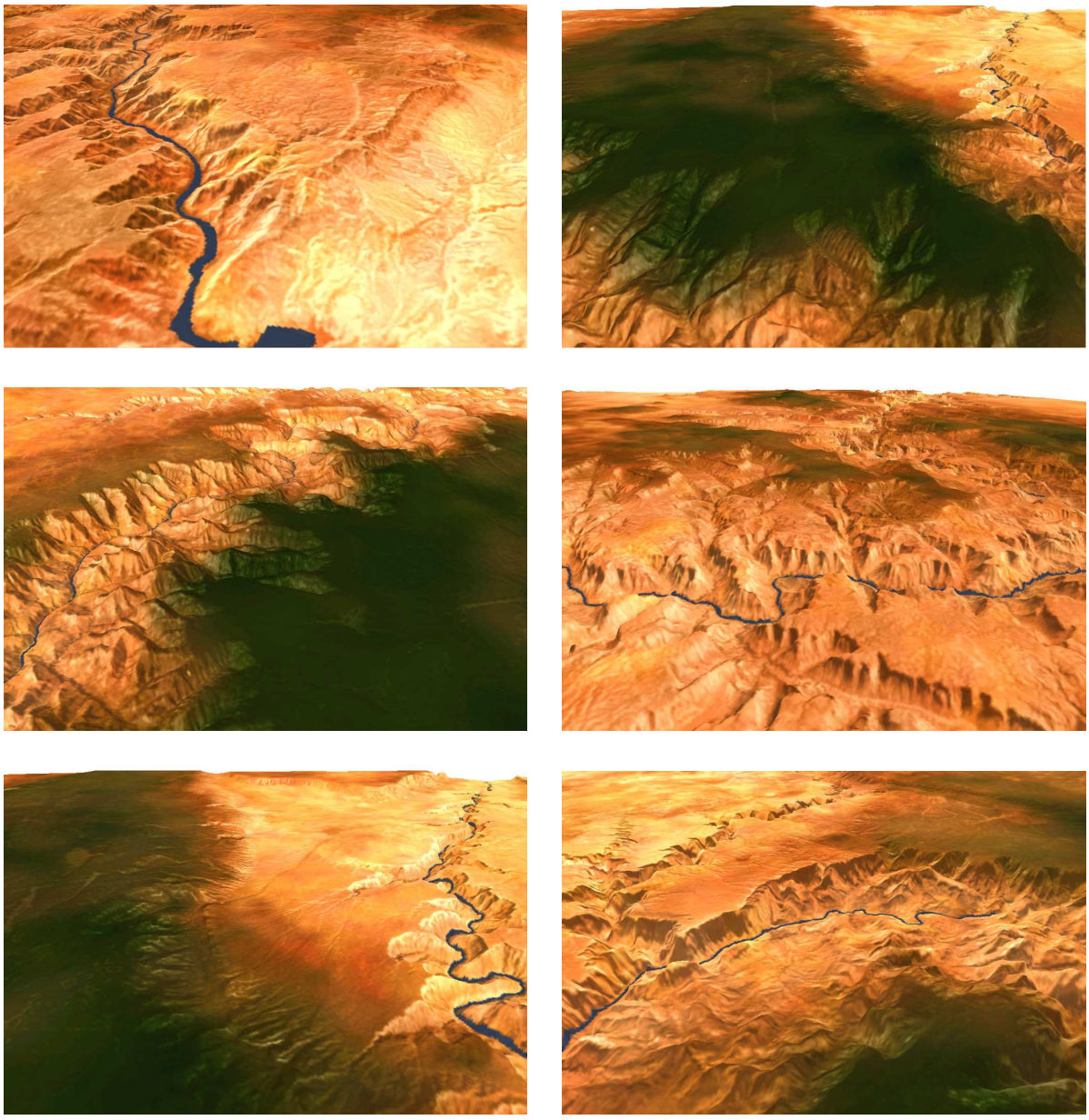


Figura C.2: Vistas del área del Gran Cañón, Arizona, USA. Los modelos son construidos sobre matrices de orden 2049×2049, procesándose más de 4 millones de puntos.