CEIG'08, Barcelona, Sept. 3–5 (2008)
L. Matey and J. C. Torres (Editors)

Visualización Esférica de Terreno 3D para Sistemas de Información Geográfica

María Ten, Jordi Torres, Jesús Zarzoso, Rafael Gaitán, Javier Lluch

Abstract

En este artículo, se presenta una librería multiplataforma desarrollada en C++ y Java sobre el grafo de escena de OpenSceneGraph que permite visualizar terrenos y planetas tridimensionales con multirresolución dependiente de la vista de forma eficiente y dinámica. La librería es capaz de visualizar datos de elevación o de textura (ortofotos o datos vectoriales rasterizados) tanto de archivos locales como de servidores remotos, sin necesidad de realizar un preproceso previo a la visualización. Nuestra finalidad es aportar un conjunto de herramientas software para la visualización y manipulación tridimensional de datos geoespaciales integrada en un Sistema de Información Geográfica.

1. Introducción

Un Sistema de Información Geográfica (SIG) es un conjunto de procedimientos (manuales o computerizados) utilizados para manipular datos referenciados geográficamente [Aro89]. En los últimos años, los SIG se han vuelto cada vez más populares debido a su gran capacidad de análisis siendo utilizados en numerosos ámbitos de aplicación como las finanzas, marketing, telecomunicaciones, medio ambiente, catastro, sanidad, transporte y turismo.

A pesar de que hoy en día existen varias herramientas y algoritmos para la representación de terrenos de forma realista son muy pocos los SIG que integran una vista tridimensional para visualización y/o edición de datos geográficos. Es cierto, que con la aparición de aplicaciones como Google Earth o Nasa World Wind que intentan acercar el mundo SIG al usuario doméstico, se ha ido popularizando la visualización tridimensional como una forma fácil e intuitiva de trabajar con datos espaciales.

El proyecto gvSIG [gvs03] es un cliente SIG gratuito, libre y multiplataforma desarrollado en JAVA por varias empresas e instituciones enmarcado dentro del Proyecto de Migración a Sistemas Abiertos de la Conselleria de Infraestructura y Transporte de la Comunidad Valenciana. Este cliente tiene implementados la mayoría de los servicios del *Open Geospatial Consortium* (OGC) [Ope06]: Web Map Service (WMS), Web Feature Service (WFS), Web Coverage Service (WCS), Servicio de Catálogo y Servicio de Nomenclator; y permite cargar datos de textura o elevación locales en la mayoría de formatos SIG actuales, así como datos alfanuméricos provenientes de bases de datos geoespaciales.

El proyecto **OSG Virtual Planets** (osgVP) nace con la finalidad de incorporar una vista tridimensional para gvSIG, para ello, se han desarrollado una serie de librerías y utilidades, que pueden ser utilizadas de forma conjunta o independiente, con el fin de trabajar con aplicaciones SIG en espacios tridimensionales.

Dentro de este proyecto se encuentra la librería **osgvp-planets** que se presenta a lo largo de este artículo. Esta librería permite visualizar terrenos o planetas de tamaño variable (tanto en su representación plana como esférica) de forma interactiva mediante técnicas de multirresolución dependientes de la vista. Los terrenos soportan múltiples capas de textura y de elevación, provenientes tanto de archivos locales como de servidores remotos.

Una de las principales ventajas que aporta **osgvp-planets** frente a otros sistemas es que no necesita realizar un preproceso de los datos para generar previamente la geometría antes de visualizarla, sino que es capaz de ir mostrando los datos conforme se van cargando en el sistema. Se trata de una librería diseñada específicamente para entornos SIG donde, a diferencia de los simuladores de conducción o los videojuegos, las texturas y los datos de elevación pueden cambiar en cualquier momento.

A continuación, se comentarán los antecedentes de la generación de terrenos, así como algunas aplicaciones capaces de realizar una visualización esférica de terrenos. Luego, se

estudiarán los requisitos que debe cumplir la librería para su integración en un SIG. Posteriormente, se tratarán los aspectos fundamentales que permiten a la librería realizar una gestión eficiente y dinámica de los datos geoespaciales proporcionados por el usuario. Finalmente, veremos algunos resultados, incluyendo la integración de la librería dentro de gvSIG, así como algunas conclusiones. Al final del artículo, se comentarán algunas de las líneas de trabajo que serán implementadas a lo largo de los próximos meses.

2. Antecedendes

Generar modelos tridimensionales de terrenos con un alto nivel de detalle en tiempo real no es una tarea sencilla, sino que requiere de muchas técnicas de aceleración gráfica, como la multirresolución dependiente del punto de vista, y del uso de algoritmos específicos para la generación de geometría. Muchos de estos algoritmos surgieron con la necesidad de renderizar terrenos en tiempo real a partir de una serie de matrices de datos de elevación (conocidos como *Heightfields*) para simuladores de vuelo o conducción.

Algoritmos como [DWS*97] se basan en la generación de una malla única a partir de un árbol binario de triángulos, de forma que cada triángulo de la malla es capaz de subdividirse o fusionarse para variar el nivel de detalle según el punto de vista del observador. Siguiendo esta filosofía existen algoritmos similares que varían la forma en la que se subdividen los triángulos de la malla como en [LKR*96], en [RHS98] y en [Paj98].

El incremento en el nivel de detalle de los conjuntos de datos de texturas y elevaciones disponibles para los terrenos en la actualidad hace que muchos de estos conjuntos de datos no quepan en memoria principal, sobre todo en el caso de los SIG donde el tamaño del terreno a renderizar puede abarcar toda la superficie de un planeta.

Esto hace necesario aplicar otro tipo de aproximaciones como en [Ulr02] donde en lugar de una única malla de elevaciones se aplica una estrategia de tipo *quadtree*: partiendo de una malla que abarca toda la superficie del terreno con baja resolución, se subdivide la malla en mallas mas pequeñas pero con mayor nivel de detalle. Este proceso recibe el nombre de *tiling* y cada una de las mallas generadas se llama *tile*. La geometría de los *tiles* se preprocesa de forma que en tiempo de ejecución se seleccionan aquellos que debemos renderizar según el punto de vista del observador. Podemos encontrar otras aproximaciones similares, algunas incluso con paginación para las texturas y datos de elevación, en [Boe00], [CGG*03a] y en [CGG*03b].

Existen aproximaciones como la presentada en [Fek90] donde a partir de un icosaedro se subdividen cada uno de los triángulos que lo forman para generar una aproximación al globo terráqueo. El problema de este tipo de algoritmos es que es difícil mantener la relación entre el nivel de subdivisión de la geometría y el nivel de detalle de las texturas o datos de elevación. Podemos encontrar más aproximaciones dentro del proyecto Virtual Terrain Project [Dis] que disponen de una vista esférica, así como aplicaciones comerciales como Google Earth, Nasa World Wind, ArcGlobe o UniView. Algunas de ellas incorporan una pequeña API o algún tipo de lenguaje de script para añadir funcionalidad de forma limitada.

El proyecto de código libre OpenSceneGraph [OB04](OSG) es un conjunto de herramientas de código abierto multiplataforma para el desarrollo de aplicaciones gráficas de alto rendimiento como simuladores de vuelo, juegos, realidad virtual y visualizaciones científicas. La filosofía de OpenSceneGraph es permitir a todo tipo de usuarios, comerciales y no comerciales, beneficiarse del uso de los grafos de escena. Basado en OpenGL y escrito en C++ estándar, provee de una API de alto nivel para trabajar con grafos de escena.

Dentro de este conjunto de herramientas se encuentra la librería osgTerrain. Esta librería permite generar terrenos multirresolución dependientes del punto de vista con paginación, a partir de un *Heightfield* con los datos de elevación y varías capas de texturas a las que podemos aplicar filtros y transparencias. La desventaja de esta librería reside en que las capas de datos deben estar disponibles antes de la generación de la geometría obligando a su reconstrucción si cambia alguna de las capas.

En el ámbito de OSG, se encuentra el proyecto Virtual Planet Builder [Osf07] implementado sobre la librería osgTerrain. Esta herramienta preprocesa grandes conjuntos de datos de elevación y textura generando bases de datos GIS con los terrenos paginados a disco, de forma que pueden ser utilizados posteriormente por una aplicación basada en el grafo de escena de OSG como simuladores de vuelo profesionales o aplicaciones del estilo Google Earth. Otro proyecto también basado en OSG que permite la visualización de datos geoespaciales tridimensionales es OSSIM Planets [oss96]

Una primera aproximación de la librería **osgvp-planets** basada en OSG está disponible en el artículo **geoviewer3d** [GTLS06] donde se presenta un prototipo de aplicación sencilla para la visualización tridimensional de terrenos con datos de elevación y múltiples capas de textura. Esta primera versión de la librería ya disponía de multirresolución dependiente de la vista y paginación.

3. Análisis y diseño

En esta sección se presentará la arquitectura de la librería **osgvp-planets** y se estudiarán cuales son los requisitos que debe cumplir para poder ser integrada dentro de un SIG.

Dado que la librería ha sido diseñada para su integración dentro de un SIG y en concreto dentro de gvSIG, existen una serie de requisitos básicos en cualquier desarrollo *software*

para mantener su usabilidad como son la eficiencia en el uso de los recursos de la máquina, la portabilidad a múltiples plataformas y la modularidad, para poder desarrollar nuevas funcionalidades de manera sencilla.

Además de los requisitos anteriores, **osgvplanets** debe implementar las especificaciones propias de una herramienta SIG:

- Generar terrenos tridimensionales dada su extensión (en coordenadas cartesianas o geográficas) y representarlos tanto en forma de globo terrestre como proyectados sobre un plano.
- Generar terrenos tridimensionales a partir de datos de elevación proporcionados por una o varias imágenes ráster de modelos digitales de terreno.
- Permitir aplicar múltiples capas de textura a partir de imágenes georreferenciadas en distintos formatos.
- Incorporar un visualizador específico para terrenos y planetas, con manipuladores especiales para permitir una navegación cómoda e intuitiva al usuario.

Debemos de tener en cuenta es que tradicionalmente los usuarios de GIS han trabajado siempre con vistas bidimensionales utilizando maquinas con tarjetas gráficas muy poco potentes. Desde el proyecto gvSIG se ha realizado mucho hincapié para que las librerías se puedan utilizar con casi cualquier tarjeta gráfica, impidiendo el uso de ciertos técnicas avanzadas que hubieran simplificado mucho algunas tareas.

Principalmente, los sistemas para la visualización de terrenos tridimensionales se basan en dos aproximaciones: la primera, consiste en generar una malla única de geometría capaz de subdividirse para mostrar diferentes niveles de detalle y la segunda se basa en una aproximación de tipo *quadtree*. Esta última será la estrategia escogida para la librería pues permite paginar la geometría, no sólo las texturas utilizadas para generarla.

3.1. Arquitectura

La librería **osgvp-planets** está desarrollada sobre el grafo de escena OSG, que aporta una gran cantidad de herramientas para renderizar sobre OpenGL de forma rápida y eficiente. Además, OSG es multiplataforma, lo que permite que nuestra librería escrita en C++ estándar también lo sea.

En la figura 1 se refleja la arquitectura modular del sistema. La librería escrita en C++ aporta una API sobre la cual se pueden desarrollar multitud de aplicaciones. Esta API permite:

- Crear un terreno o planeta indicando la extensión del mismo, los radios polar y ecuatorial, el sistema de coordenadas que utiliza y el tipo de representación con que deseamos visualizarlo (plana o esférica).
- Añadir, eliminar, reordenar, invalidar, activar y desactivar capas de textura o elevación, así como establecer rangos a partir de los cuales las capas empiezan o dejan de ser visibles (rangos de escala). Además las capas de textura pueden cambiar su opacidad y las de elevación permiten variar un factor de exageración vertical del terreno.
- Cambiar ciertos factores referentes a la generación de la geometría como son el uso de faldas o los factores de subdivisión de los terrenos.
- Seleccionar distintos manipuladores para la navegación.

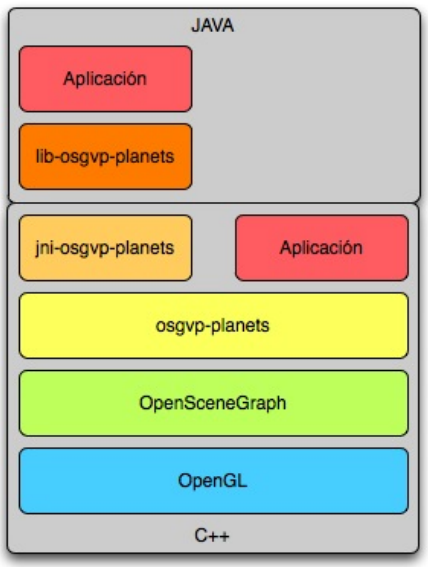


Figure 1: Arquitectura de la librería **osgvp-planets**.

El proyecto gvSIG como algunos otros SIG actuales que sólo poseen una vista bidimensional, está escrito en JAVA. Esto permite a las aplicaciones ser multiplataforma sin necesidad de realizar una compilación específica del producto para las diferentes plataformas existentes en la actualidad. La desventaja del lenguaje, es que al tratarse de un lenguaje interpretado no permite realizar cálculos complejos tan rápidamente como los lenguajes compilados. Este hecho, junto con la imposibilidad de encontrar grafos de escena eficientes sobre el lenguaje, hace su uso desaconsejable en el caso de los gráficos tridimensionales.

Por tanto, para poder integrarse con gvSIG, **osgvp-planets** incorpora una serie de *wrappers* que permiten acceder a la API desde aplicaciones escritas en JAVA. Esto se consigue gracias al uso de Java Native Interfaces (JNI), un lenguaje desarrollado específicamente para comunicar librerías nativas (en C++ o C) con el lenguaje JAVA.

4. Desarrollo

A lo largo de la siguiente sección se explicarán los distintos mecanismos que permiten realizar una gestión dinámica

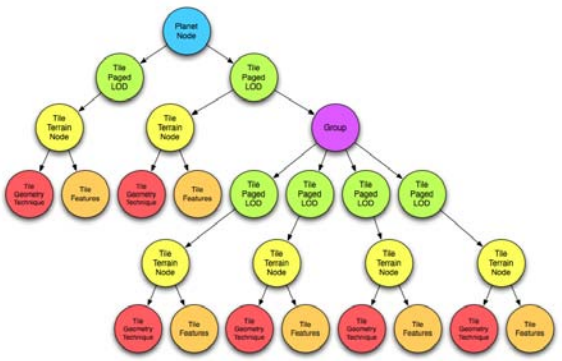


Figure 2: Grafo de escena generado durante la navegación por la librería **osgvp-planets**.

de las capas, así como las características más destacadas de la librería.

En la figura 2 se puede observar una muestra del grafo de escena generado por la librería. El nodo raíz, denominado **Planet Node** mantiene toda la información global del terreno: extensión, proyección, capas a visualizar, etc. Este nodo se compone de **Tile Terrain Nodes** que son pequeñas regiones regulares de terreno (*tiles*) que poseen su propia extensión y capas de textura y/o elevación. Estos *tiles*, a su vez, se encapsulan dentro de nodos multirresolución paginados, llamados **Tile Paged LOD** lo que les permite subdividirse en cuatro nuevos *tiles* siguiendo una estrategia de tipo *quadtree* dependiente de la distancia al observador y el tamaño de la ventana de visualización. Una explicación más extensa sobre el funcionamiento de los nodos **Tile Paged LOD** y el sistema de paginación puede encontrarse en **geoviewer3d** [GTLS06]. Por último, el nodo **Tile Geometry Technique** contiene la geometría correspondiente a la extensión del *tile* y el nodo **TileFeatures** está diseñado, para que en un futuro se puedan añadir elementos vectoriales dentro de los *tiles* y sean paginados junto con el terreno.

El nodo **Planet Node** mantiene un listado de estructuras de datos denominadas **PlanetLayerInfo** para la gestión de capas de textura y otro idéntico para las capas de elevación. Cada **PlanetLayerInfo** mantiene la información indispensable para la capa: el orden en el que debe visualizarse, si está habilitada o deshabilitada, los rangos máximos y mínimos de visibilidad, el nivel máximo de resolución que poseen los datos y la extensión que ocupa la capa en el sistema de coordenadas del planeta. Además las capas de textura disponen de la opacidad y las capas de elevación poseen un factor de exageración vertical.

Asimismo, cada nodo **Tile Terrain Node** tiene su propio listado de capas de textura y otro de elevación en el que sólo aparecen las capas cuya extensión intersecta con la extensión del terreno que representa el nodo. La estructura de datos **TileLayerInfo** empleada en dicho listado de capas, almacena la información que sólo afecta a los datos de ese *tile*: la textura que corresponde a esa capa y si la textura es valida o está propagada, en el caso de capas de textura y la matriz de alturas en el caso de capas de elevación. Para ambos tipos de capas siempre se almacena una referencia al **PlanetLayerInfo** que le corresponde.

4.1. Gestión de capas mediante eventos

Uno de los puntos más fuertes de **osgvp-planets** es que permite al usuario, a través de un aplicación SIG, la posibilidad de añadir o eliminar capas con datos de elevación o de textura en cualquier momento, sin necesidad de preprocesarlas y sin que la navegación se vea afectada, con un tiempo de respuesta relativamente corto.

Dado que muchas aplicaciones SIG no sólo trabajan con datos provenientes del disco local, sino que casi todas son capaces de trabajar con servidores remotos, el tiempo necesario para obtener las imágenes para texturas o elevaciones de la región que estamos visualizando puede llegar a extenderse demasiado. Ya que la navegación debe ser interactiva y fluida, no es conveniente esperar a disponer de todos los datos para renderizar el terreno.

La solución ofrecida por **osgvp-planets**, se basa en crear dos colas de eventos para interactuar con el planeta: una para los eventos de entrada y otra para los de salida. En lo sucesivo, se denominará evento a la llamada de cualquier método disponible en la API de la librería para la modificación del terreno que se está visualizando.

La creación de colas de eventos nos permite, interactuar en cualquier momento con la API de la librería sin que la navegación o el ciclo de renderizado se vea afectado. De esta forma cuando la interfaz de usuario de la aplicación llama a un método de la librería se genera un evento que es encolado en la lista de eventos de entrada del planeta. De forma que en la fase de actualización del grafo de escena (conocida como fase de *update*) estos eventos pueden ser procesados sin colapsar el ciclo de dibujado, manteniendo la consistencia en todo momento. Cuando la librería necesita solicitar datos a la aplicación SIG, procede de manera análoga, generando un evento en la cola de salida para que la aplicación lo procese y le envíe los datos necesarios con otro evento cuando sea posible.

Este sistema de eventos se vuelve realmente eficaz aplicado al manejo de capas de datos geoespaciales como podemos apreciar en la figura 3. La interfaz comunica a la librería que ha sido añadida una capa de datos mediante un evento, de forma que cuando el grafo de escena llega a su fase de actualización, el evento es procesado permitiendo que cada uno de los *tiles* generados por el planeta calculen si su extensión intersecta con la extensión de la capa añadida, y si es así, la capa se añade al listado de capas particular de cada *tile*. Los tiles generados posteriormente también calculan cuales son

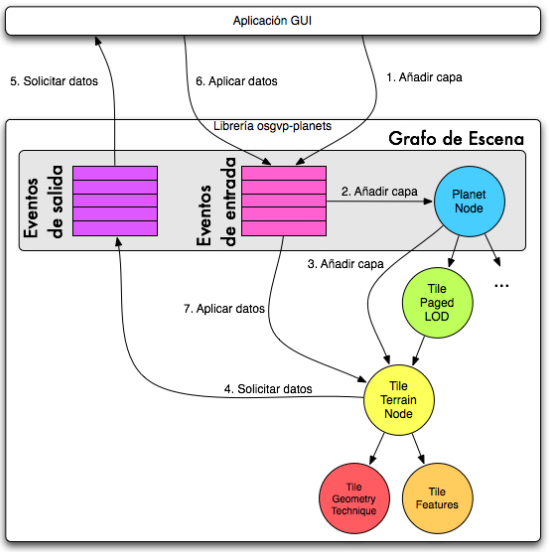


Figure 3: Esquema de funcionamiento del sistema de eventos para capas.

las capas que pertenecen a su región en el momento en que son creados.

De esta forma, los *tiles* son los que, al ser atravesados, comunican a la aplicación que no disponen datos para las capas de su listado mediante un evento de salida. Esto permite que no se detenga el ciclo de dibujado, pues la librería después de crear el evento sigue atravesando el resto de nodos del grafo de escena sin necesidad de esperarse, renderizando la geometría con los datos que disponía previamente.

La aplicación es la que debe recoger los eventos de salida de la librería y procesarlos en un hilo de ejecución aparte. Si la aplicación incorpora un sistema de *caché*, debería asegurarse de que los datos no están disponibles en la caché y solicitarlos al servidor remoto correspondiente o bien, si se trata de un archivo de disco, es la aplicación la que debe asegurarse que exista el fichero de disco. Si todo es correcto, deberá generar un evento en la cola de entrada del planeta para comunicar al *tile* que hizo la solicitud que los datos ya están preparados para ser procesados.

Todos los eventos, tanto de entrada como de salida del planeta se ordenan por prioridad. En el caso de eventos de entrada, la prioridad viene marcada por el tiempo en el que dichos eventos fueron generados, es decir, que son procesados siguiendo el mismo orden con el que se añadieron a la cola. Sin embargo, los eventos de salida corresponden en su mayoría a la solicitud de datos de textura o elevación que se van añadiendo a la cola de eventos de salida según se va atravesando el grafo de escena, de forma que si atravesamos varias veces un mismo *tile* no se generan nuevos eventos de salida para la misma capa, sino que se actualiza la prioridad del evento que está en la cola. Además, en el caso de de los eventos de salida, la prioridad se calcula a partir del tiempo en el que fue atravesado el *tile*, el nivel de subdivision y un factor dependiente del punto de vista, para asegurarnos de que siempre se solicitan primero aquellos datos que corresponden a la región del espacio que se está visualizando.

4.2. Propagación de texturas

Las capas de datos de textura, al igual que las de elevación, muchas veces proceden de servidores remotos de forma que no siempre están disponibles todos los datos necesarios al cargar la capa en la aplicación. Muchos sistemas SIG adoptan una solución progresiva para paliar los problemas que este hecho provoca. Dicha solución se basa en solicitar todos los datos necesarios para los hijos de un *tile* impidiendo que se pueda subdividir hasta que estos datos puedan ser cargados y procesados. El problema es que cuando se trabaja con terrenos de mucha resolución, es decir cuando están muy subdivididos, el usuario tiene que esperar demasiado tiempo hasta que puede visualizar los datos en la resolución correcta, pues debe solicitar los datos para todos los niveles anteriores.

En cambio, la solución implementada se basa en permitir a los *tiles* subdividirse siempre, dispongan o no de los datos, hasta alcanzar la resolución adecuada de forma que el sistema de prioridad de la cola de eventos hace que primero se soliciten los datos para los *tiles* de mayor resolución que se están visualizando en ese momento. Esto hace que el usuario pueda visualizar los datos en la escala adecuada mucho antes que con el sistema anterior, pero tiene el inconveniente de que no se disponen de algunos datos de forma que pueden aparecer secciones del terreno sin textura o sin elevación durante un breve periodo de tiempo como se observa en la figura 4(a). Para paliar el efecto producido por la falta de datos, se ha implementado un sistema de propagación de texturas de forma que no se visualicen *tiles* sin alguna de sus capas de textura.

Los grafos de escena permiten realizar propagación de texturas de forma natural, es decir, los nodos hijos heredan las texturas de los padres a no ser que dichos nodos cambien su propio *stateset*, pero la estructura del grafo generado por **osgvp-planets** impide este tipo de propagación. Esto se debe a que los datos de textura y elevación residen en el **Tile Terrain Node** en lugar de en los **Tile Paged LOD**, pues así se consigue que las texturas y las matrices de elevación se paginen junto a la geometría.

En lugar de una propagación natural, la librería utiliza un sistema de propagación forzado. Cuando un *tile* es atravesado en el bucle de dibujado, mira en su listado de capas si le faltan los datos para alguna de las capas y los solicita generando el evento correspondiente en la cola de salida. Posteriormente, el *tile* debe establecer si su nodo padre, es decir el terreno que a partir de subdividirse lo genero a él y a sus

(a) Capa raster sin propagación solicitando texturas.

(b) Capa raster con propagación solicitando texturas.

(c) Capa de textura con todas las texturas cargadas.

Figure 4: Ejemplo de capa de ráster aplicada como textura sobre el terreno.

hermanos, tiene datos de textura para las capas que le faltan. Si es así, el nodo hijo establecerá una referencia a la textura correspondiente del nodo padre (figura 4(b)) de forma que los cambios realizados en la textura original son reflejados en los nodos hijos. Cuando el nodo reciba la textura que le corresponde sustituirá la referencia a la textura del nodo padre por la nueva textura en el **TileLayerInfo** correspondiente (figura 4(c)).

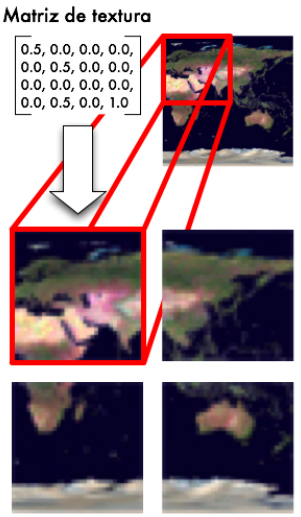


Figure 5: Ejemplo recorte y translación para la propagación de texturas.

Al utilizar una estrategia del tipo *quadtree*, cada nodo cuando se subdivide genera cuatro hijos. Si estos hijos no disponen aún de las texturas que les corresponden utilizarán las de sus padres, de forma que se verá la misma textura replicada cuatro veces. Esto se soluciona aplicando matrices de textura, de forma que cada hijo (y dependiendo de su posición) aplicará un escalado y una translación para ajustar la textura del padre a la región que le corresponde tal y como se puede apreciar en la figura 5. Sin embargo, los nodos que disponen ya de su textura (sin propagación) aplican una matriz identidad. De esta forma, se pueden propagar texturas entre padres e hijos con varios niveles de resolución de diferencia de forma muy sencilla. Cuando un hijo no dispone de textura y su padre dispone sólo de una textura propagada, bastará con hacer referencia a la textura propagada del padre y multiplicar su matriz de textura por la de su padre. Si el nodo padre obtuviera su textura antes que el hijo, su matriz de textura ahora sería una matriz identidad de forma que al multiplicarla por la matriz del nodo hijo, está se quedaría tal cual.

Este sistema presenta un problema: la propagación sólo puede realizarse de padres a hijos, nunca a la inversa. Dado que se establecen una serie de prioridades para solicitar los datos de las capas que obligan a pedir primero las texturas de los *tiles* de mayor nivel de subdivisión, es posibles que los hijos tengan sus texturas mucho antes que sus padres. Esto hace que al distanciarse el observador y volver a visualizar el nodo padre en lugar de los hijos, puedan verse *tiles* sin textura. Ajustando las prioridades para que primero se soliciten las texturas de los *tiles* iniciales y posteriormente las pertenecientes a los que aparecen en la visualización se soluciona este problema.

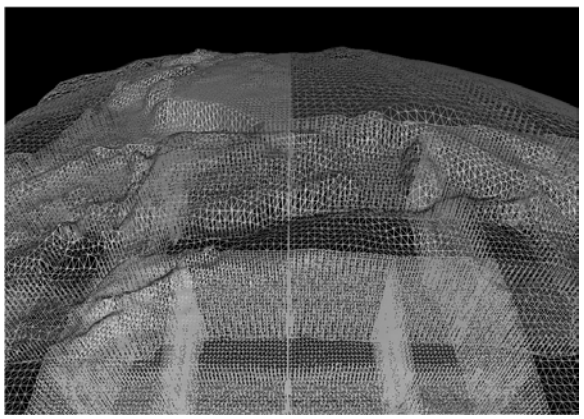


Figure 6: Terreno compuesto por tiles de diferente nivel de subdivisión.

Por supuesto, las texturas propagadas tienen una calidad más baja que la que corresponde según el nivel de subdivisión del *tile* y en principio puede parecer poco interesante implementar propagación respecto a esperar todos los datos para un nivel de subdivisión, pues no se mejora la calidad de la imagen hasta que se envía la textura definitiva. Pero en la práctica no resulta así, pues esta técnica tiene dos grandes ventajas. La primera, consiste en que permite obtener texturas de alta calidad en menor tiempo si se combina con un buen sistema de prioridades, pues sólo solicita los datos para el primer y último nivel de subdivisión mientras que el resto son propagados. La segunda, es que al trabajar con varias capas de texturas, se puede dar el caso de que dispongamos las datos para una pero no para las demás capas sobre todo al trabajar con servidores remotos, de manera que gracias a la propagación es posible visualizar diversas capas con distinto nivel de detalle cada una hasta que se dispongan de todos los datos.

4.3. Generación de geometría

En **osgvp-planets** el terreno no se representa a partir de una malla única, sino de pequeñas mallas de diferente resolución y tamaño generadas por los *tiles* tal y como se observa en la figura 6, de forma que pueden ser fácilmente paginadas. Cada vez que se crea un nuevo *tile*, se debe cubrir su extensión generando una malla regular del mismo tamaño. Este sistema tiene la desventaja de que se pueden producir agujeros en la superficie del terreno cuando dos *tiles* contiguos no están a la misma altura ya sea porque uno de los dos no tiene datos de elevación o los datos no tienen una calidad suficiente para formar una malla continua. Para solucionarlo, se genera una *falda* o tira de triángulos hacia el interior del terreno recorriendo el borde del *tile*.

Cuando se añade una capa de elevación al *tile* la geometría activa el flag *dirty* para que cuando se llegue a la fase de

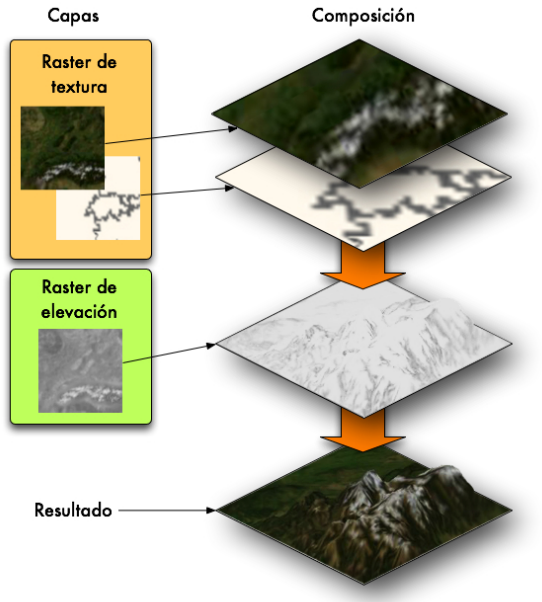


Figure 7: Ejemplo de construcción de un tile a partir de varias capas de textura y elevación.

update del grafo de escena, sea consciente de que la malla debe ser regenerada. Las capas de elevación no son más que imágenes raster en escalas de grises, de forma que la intensidad de cada píxel representa un nivel de altura distinto. En la malla la altura de cada vértice corresponde al valor de un píxel de la imagen raster multiplicado por un factor de exageración vertical establecido por el usuario. Si se varia este factor de exageración la geometría se debe volver a generar activando el flag de *dirty*.

La gestión de texturas se realiza de forma independiente a la generación de las mallas de geometría para que no sea necesario regenerarlas cada vez que se modifica el listado de capas de textura. Las texturas tienen su propio flag de *dirty* que permite modificar únicamente el *stateset* del *tile* cuando se añade, elimina, reordena o se cambia la opacidad de la capa. En la figura 7 podemos ver el resultado de superponer varias capas de textura sobre una capa de elevación en un *tile*.

En la vista esférica, el terreno se aproxima a un modelo matemático elipsoidal de forma que, dada la extensión de cada *tile* en radianes y los radios polar y ecuatorial del elipse, se pueden curvar las mallas de geometría y posicionarlas en el espacio para formar un planeta elíptico completamente cerrado. Durante la construcción de estas mallas curvadas, también se establecen los parámetros necesarios para realizar la *eliminación de caras traseras* a partir de la curvatura, la normal, el tamaño y la posición de los *tiles*.

El nodo **TileGeometryTechnique** establece la forma en la que se genera la geometría, pero es posible heredar de él los métodos necesarios para generar un nuevo nodo que construya mallas de elevación irregulares. De esta forma se podría trabajar con datos de elevación obtenidos a partir de *Triangulated Irregular Networks* (TINs), que son modelos digitales de terreno optimizados.

5. Resultados

A continuación se presentan algunos resultados obtenidos mediante la librería **osgvp-planets**. Actualmente la librería es capaz de generar terrenos y planetas de cualquier tamaño en vista proyectada, es decir, en plano (figura 8(a)) y en vista esférica (figura 8(b) y figura 8(c)). Los terrenos vienen definidos por el nombre, el tipo de vista (plana o esférica), el tipo del sistema de coordenadas (geocéntrico, geográfico o proyectado), el sistema de coordenadas, la extensión del terreno y los radios polar y ecuatorial del planeta al que pertenece el terreno.

Los planetas pueden contener una o varias capas, tanto de texturas como de elevación. La librería dispone de la API necesaria para añadir, eliminar y reordenar las capas. Además permite habilitar y deshabilitar la visualización de una o varias capas sin necesidad de volver a cargar los datos posteriormente. También se dispone de un método para invalidar una capa de textura de forma que los datos se vuelven a solicitar para todos los *tiles* que dispongan de dicha capa.

Las capas de texturas que se permiten cargar son imágenes raster georreferenciadas (o imágenes vectoriales rasterizadas) provenientes tanto de archivos locales como de servidores remotos. La opacidad de la capa puede cambiarse en tiempo real, para facilitar la vista de múltiples capas de textura. En la figura 9 podemos ver un ejemplo de funcionamiento de la transparencia.

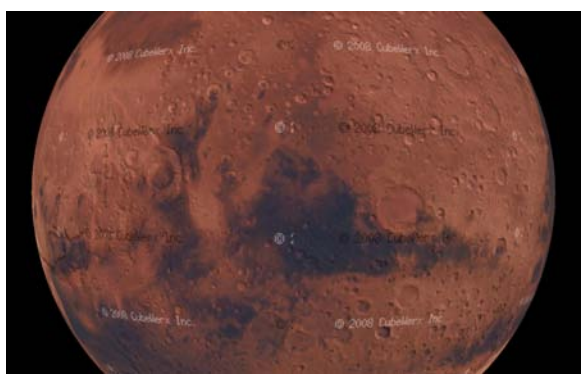
Las capas de elevación que soporta son también imágenes raster locales o de servidores remotos mediante las cuales se generan las mallas de elevación del terreno. Variando el factor de exageración vertical podemos magnificar el efecto de la capa de elevación sobre el terreno. En la figura 10 podemos ver un ejemplo de terreno con una capa de elevación procedente de un servidor remoto.

La librería es capaz de renderizar terrenos del tamaño de la tierra con elevaciones y múltiples capas de textura con un *frame-rate* bastante superior a los 30fps. Incorpora un visualizador 3D con manipuladores de navegación específicos para planetas y terrenos que se puede integrar dentro de cualquier aplicación SIG.

En la figura 11 podemos ver la aplicación gvSIG con una vista 2D junto a una vista 3D generada con **osgvp-planets**, con diversas capas de textura y elementos vectoriales rasterizados. La librería se integra perfectamente con la aplicación gvSIG permitiendo el uso de la mayoría de sus herramientas sobre visualización tridimensional del terreno como en la figura 12 donde podemos ver el uso del **localizador**, una herramienta que permite localizar ciudades, ríos, montañas, etc. a partir de sus nombres.

(a) Vista plana del puerto de Valencia usando datos de ficheros locales.

(b) Vista esférica de la Tierra usando datos del servidor WMS de la NASA.



(c) Vista esférica de la Marte usando datos del servidor WMS de CubeWerx.

Figure 8: Diferentes tipos de terreno generados por la librería **osgvp-planets**.

6. Conclusiones

La librería presentada se ofrece como un sistema de generación de terrenos para aplicaciones SIG en tres dimensio-

nes interactivo, dinámico y eficiente. A diferencia de otras librerías para generación de terrenos, **osgvp-planets** no necesita realizar un preproceso, para poder construir la geometría. Permite manejar múltiples capas de textura y cambiar su opacidad de forma interactiva, al igual que sucede con la exageración vertical del terreno. También puede trabajar con datos procedentes de archivos locales y de servidores remotos, sin que la navegación se vea afectada por el tiempo de descarga de las imágenes. Su sistema de cola de eventos priorizados junto con la propagación de textura, hacen que el usuario pueda visualizar datos de gran resolución mucho más rápido que en otras aplicaciones SIG.

Para mantener un *frame-rate* superior a los 30fps se han implementado técnicas de aceleración gráfica como *frustrum-culling*, *cluster-culling*, multirresolución dependiente de la vista o paginación (tanto para geometría como para texturas). Además, la utilización de flags *dirty* separados para texturas y geometría hacen que las mallas se regeneren únicamente cuando es indispensable.

7. Trabajos Futuros

Durante los próximos meses se va a implementar un sistema para propagación de datos de elevación similar al presentado para las texturas. Al mismo tiempo, se desea añadir un sistema de ecualización para las mallas con alturas, de forma que no sea necesario el uso de *faldas* para evitar los saltos en elevaciones de *tiles* contiguos.

También sería deseable implementar nuevos tipos de nodos **TileGeometryTechnique** que permitan el uso de TINs u otras geometrias optimizadas, así como la generación independiente de la malla geométrica de las regiones polares para evitar que se colapsen.

Respecto a las texturas, faltaría implementar la *multipasada* para permitir al usuario elegir entre ésta o *multitextura* para visualizar varias capas de textura sobre el terreno.

Por último, se ha planteado la posibilidad de portar la librería hacia plataformas móviles, como PDAs, para permitir el uso de aplicaciones SIG en trabajos de campo.

8. Agradecimientos

Este trabajo está siendo financiado por la *Conselleria d'Infraestructures i Transport* de la *Generalitat Valenciana* (Spain) y en parte por el proyecto TIN2005-08863-C03-01 del Ministerio de Educación y Ciencia.

References

[Aro89] ARONOFF S.: *Geographic Information Systems: A Management Perspective*. Ottawa, Canada: WDL Publications., 1989.

[Boe00] BOER W. D.: Fast terrain rendering using geometrical mipmapping, 2000.

[CGG*03a] CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., PONCHIO F., SCOPIGNO R.: Bdam - batched dynamic adaptive meshes for high performance terrain visualization. In *Computer Graphics Forum* (2003), vol. 22 (3), pp. 505–514.

[CGG*03b] CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., PONCHIO F., SCOPIGNO R.: Planet-sized batched dynamic adaptive meshes (p-bdam). In *IEEE Visualization, 2003* (2003), pp. 147–154.

[Dis] DISCOE B.: Virtual terrain project. http://www.vterrain.org/.

[DWS*97] DUCHAINEAU M. A., WOLINSKY M., SIGETI D. E., MILLER M. C., ALDRICH C., MINEEV-WEINSTEIN M. B.: ROAMing terrain: real-time optimally adapting meshes. In *IEEE Visualization* (1997), pp. 81–88.

[Fek90] FEKETE G.: Rendering and managing spherical data with sphere quadtrees. In *Proceedings of the First IEEE Conference on Volume* (1990), pp. 176–186.

[GTLS06] GAITÁN R., TEN M., LLUCH J., SEVILLA L. W.: Geoviewer3d: 3d geographical information viewing. In *Ibero-American Symposium on Computer Graphics (SIACG)* (2006), Brunet P., Correia N., Baranoski G., (Eds.).

[gvs03] gvsig, 2003. http://www.gvsig.gva.es/.

[LKR*96] LINDSTROM P., KOLLER D., RIBARSKY W., HODGES L., FAUST N., TURNER G.: Real-time continuous level of detail rendering of height fields. *Proceedings of SIGGRAPH'96* (1996), 109–118.

[OB04] OSFIELD R., BURNS D.: Openscenegraph, *www.openscenegraph.org*, 2004.

[Ope06] OPEN GEOSPATIAL CONSORTIUM: *OpenGIS® Web Map Service (WMS) Implementation Specification*, March 2006.

[Osf07] OSFIELD R.: Virtual planet builder, 2007. http://www.openscenegraph.org/projects/VirtualPlanetBuilder.

[oss96] Ossim planets, 1996. http://www.ossim.org/.

[Paj98] PAJAROLA R. B.: Large scale terrain visualization using the restricted quadtree triangulation. In *IEEE Visualization '98* (1998), Ebert D., Hagen H., Rushmeier H., (Eds.), pp. 19–26.

[RHS98] ROETTGER S., HEIDRICH W., SLUSSALLEK P.: Real–time generation of continuous levels of detail for height fields. In *Proc. 6th Int. Conf. in Central Europe on Computer Graphics and Visualization* (1998), pp. 315–322.

[Ulr02] ULRICH T.: Rendering massive terrains using chunked level of detail control. In *Course Notes of ACM SIGGRAPH 2002* (2002), vol. Course 35.

(a) Transparencia de la capa superior al 0 %.

(b) Transparencia de la capa superior al 50 %.

(c) Transparencia de la capa superior al 100 %.

Figure 9: Vista del puerto de Valencia variando la opacidad de dos capas de textura con imágenes del año 1980 y del año 1990 respectivamente.

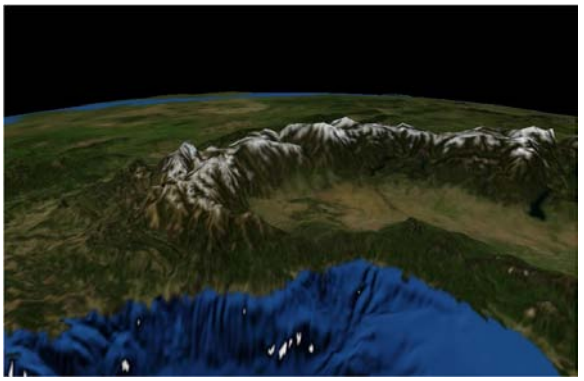


Figure 10: Vista de los Alpes con datos de textura y elevación del servidor WMS de la NASA.

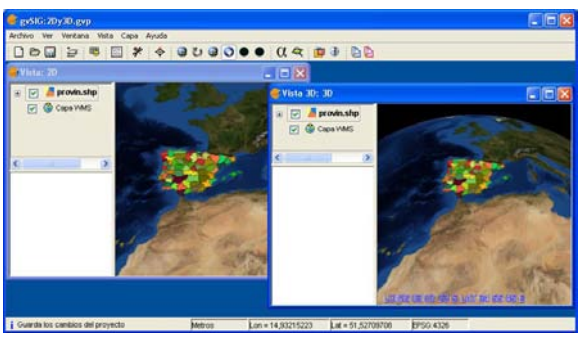


Figure 11: Comparación entre la vista 2D y la vista 3D incorporada con **osgvp-planets** en gvSIG.

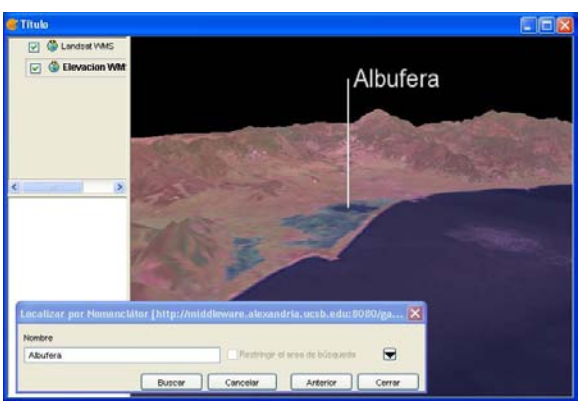


Figure 12: Señalización de la Albufera de Valencia mediante la herrmaienta **localizador** de gvSIG utilizando un terreno generado con **osgvp-planets**.