Visualización avanzada en globos 3D.

D.Gómez-Deck[(1)], M. de la Calle Alonso[(1)], O. Koehler[(1)], F. Pulido Galán [(1)]

[(1)] IGO SOFTWARE. C/Ceclavín 5 2º I 10004 Cáceres. dgd@igosoftware.es mdelacalle@igosoftware.es, fpulido@igosoftware.es

RESUMEN

Durante los últimos meses IGO Software ha estado desarrollando activamente el proyecto glob3, la librería euclid y junto a la Universidad de Extremadura la migración de la librería sextante a un entorno 3D. Todas estas tareas han concluido en la liberación de un Framework para el desarrollo de aplicaciones 3D denominado glob3 que seguirá desarrollándose al menos durante todo el 2011 y cuyo roadmap puede ser consultado en la página del proyecto http://www.glob3.org.
En este marco se han desarrollado varias formas avanzadas de visualización de información georeferenciada que pensamos pueden ser interesantes para la comunidad ya que no han sido desarrolladas anteriormente en software libre, concretamente presentaremos:

- *Visualización de nubes de puntos mediante streaming*
- *Visualización de modelos 3D*
- *Visualización de archivos multidimensionales en formatos netCDF (network Common Data Form)*
- *Visualizacion de archivos GIS vectoriales*
- *Gigaimágenes y fotos 360º*

Palabras clave: *3D, glob3, netCDF, Nasa World Wind, euclid, CSIRO, Oceanografía*

INTRODUCCIÓN

Glob3 es un Framework para el desarrollo de SIG en 3D. Está en plena implementación y esperamos tener una reléase a lo largo del primer cuatrimestre de 2011. Este proyecto está siendo desarrollado por IGO SOFTWARE[1] y financiado por el CDTI (Centro para el Desarrollo Tecnológico e Industrial) del Ministerio de Ciencia e Innovación[2]. Por ahora nuestro elemento diferenciador ha sido el diseño e implementación de nuevas formas de visualización de la información SIG y en nuevas formas de organización de estos datos para que sea posible verlos independiente del tamaño que tengan. Para poder hacer esto hemos tenido que desarrollar librerías que tratan las geometrías 3D de forma específica como ya mostramos en las Jornadas de SIG Libre de 2010.[3] y el FOSS4G de Barcelona[4]

GENERALIDADES DE GLOB3

Toda la información sobre el proyecto puede ser encontrada en http://www.glob3.org. El proyecto es completamente de fuentes abiertas y tiene una licencia muy libre, concretamente un BSD, está escrito es java por lo que es multiplataforma y además puede visualizarse via web mediante java Applets. Es un contenedor de información multiresolución y multipropósito y estamos desarrollándolo para que implemente todos los estándares OGC que vamos necesitando.

El propósito es tener un Framework que nos permita el desarrollo muy rápido de aplicaciones SIG 3D con todas las complicaciones que tiene el hecho de trabajar con 3D real.

La arquitectura del proyecto en la actualidad es la siguiente:

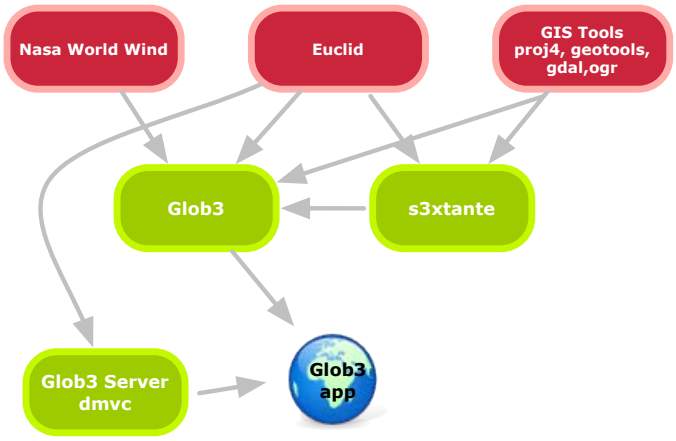

Figura 1: *Arquitectura del proyecto Glob3.*

Tenemos varias librerías distintas o subproyectos distintos que componen glob3 completo, en algunas aplicaciones no se usan todos.

- **Euclid.** Es un Framework para manejar geometrías multidimensional y multiprecisión Es decir podemos trabajar con él indistintamente típicamente de las dimensiones en que tengamos los datos 2D – 3D y de su precisión Euclid se encarga de gestionar todo de manera que siempre los cálculos sean correctos.
  Se han desarrollado multitud de optimizaciones que nos permiten por ejemplo construir un octGrid con 127 millones de puntos con color estas optimizaciones se han hecho en dos sentidos:
  - Optimizar el uso de la memoria. Uso apropiado de las Clases de java. (Clases Buffer)
  - Aprovechamiento de la arquitectura de múltiples núcleos de los procesadores de ultima generación usando las posibilidades de programación multihilo de java.

  Un ejemplo de como funciona euclid son las clases de generalización de una bola n-dimensional, esta sería la clase n-Ball su especialización a 2D sería Disk y la especialización a 3D sería Ball

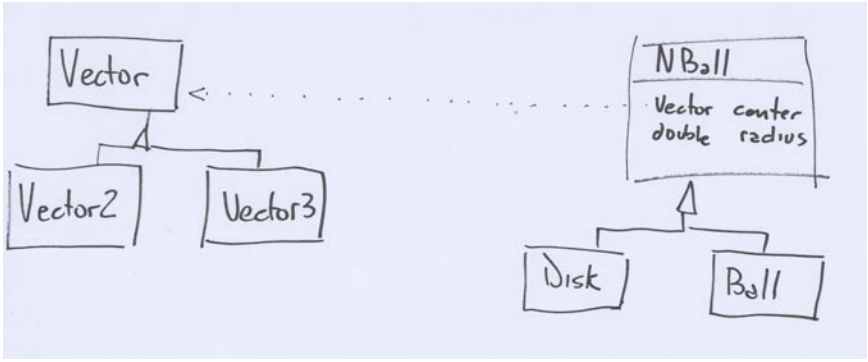

Figura 2: *Euclid*

- **Nasa World Wind.** Es la librería de la NASA que se encarga de dibujar el globo 3D. Es un proyecto que ya lleva varios años desarrollándose, muy bien programado y una base estupenda para trabajar con un SIG 3D [5]
- **Gis Tools.** Son todas las librerías SIG que usamos en conjunto para trabajar con datos SIG. Tenemos necesidades de manejar proyecciones (Proj4), leer archivos (Geotools) • leer raster (OGR) etc... Además hay que prestar especial atención a que todas las librerías que usemos sean compatibles con la licencia NOSA, lo cual excluye a cualquier desarrollo licenciado bajo GPL.
- **Glob3.** Es el proyecto core que amalgama el resto y nos provee de los servicios y métodos necesarios para construir las aplicaciones 3D
- **S3xtante.** Es la versión 3D de Sextante, todos los algoritmos que se van haciendo en 3D, ya sean nuevos o migraciones de otros anteriores están en este paquete. Además es un Binding a Sextante con todas sus funcionalidades, y permite la visualización de datos 2D sobre el globo 3D
- **Glob3 Server. DMVC**. Como para algunos de los set de datos que tenemos no hay servidores ni estándares, hemos desarrollado nuestro propio servidor y nuestra propia librería para usar el modelo vista controlador de manera remota. Un ejemplo es el servidor de streaming de nubes de puntos.

VISUALIZACIÓN DE NUBES DE PUNTOS

La visualización de enormes nubes de puntos es un problema complejo, hemos hecho varias trabajos y el último desarrollo es el de un Servidor de nubes de puntos mediante streaming, se encuentra en la actualidad en fase de pruebas pero el preproceso de los datos es el mismo que para mostrarlo en local, ¿Por qué hacer un servidor de nubes de puntos?

- Porque las nubes son muy grandes, muy complicadas de manejar y de esta manera quedan centralizadas en un servidor y los clientes sólo tienen que conectarse para recibir los puntos de manera optimizada.
- Desplazamos la tarea del preproceso al servidor, máquinas más potentes y preparadas para ello y los usuarios pueden visualizar nubes muy grandes desde máquinas normales.
- Las nubes son cada vez más grandes y hasta para visualizarlas en local es necesario seguir estrategias de renderizado 3D para mostrarlas.

Para ver las nubes de puntos primero debemos hacer algunas consideraciones tal y como saber que las nubes de puntos de LiDAR son distintas de las extraídas de los escáneres láser terrestre, en la siguiente tabla mostramos las diferencias:

Tabla 1: Comparativa de nubes de puntos LiDAR y de Láser escáner terrestre

LiDAR	Láser Escáner
Rango pequeño de Z	Grande
Se puede tratar como 2.5 D	3D
Cubre áreas muy grandes	Puntos mucho más concentrados

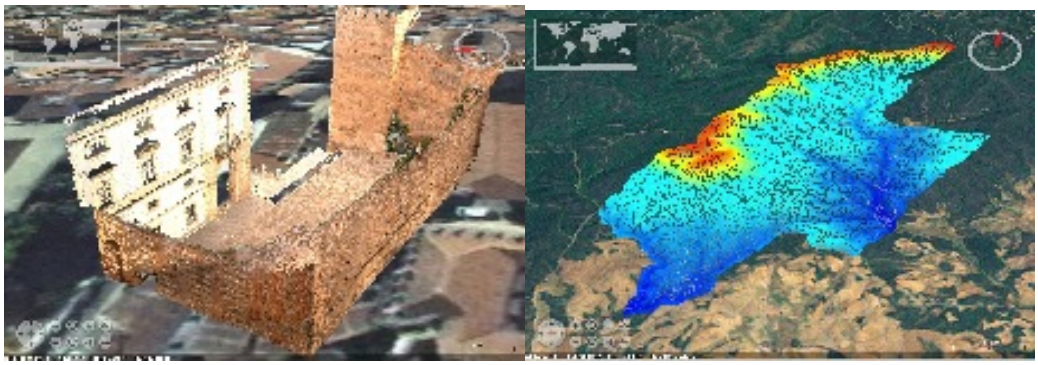

Figura 3: *Nube de Láser terrestre y nube de LiDAR*

El algoritmo usado para el preproceso básicamente tiene las siguientes partes:

- **Particionamiento.** Generación del Octree usando todos los puntos contenidos en la nube, se genera un octree adaptativo en función del tipo de nube
- **Organización de los datos.** Se ordenan los puntos mediante un KDTree creando una estructura de árbol en la cual se divide el espacio en una sola dimensión en cada nivel del árbol. Tras este ordenamiento el algoritmo de visualización solo tiene que:
    - Calcular cuantos puntos se necesitan para representar adecuadamente un tile.
    - Leerlos
- **Visualización.** El algoritmo de visualización utiliza técnicas de Level Of detail de manera que sólo muestra los puntos necesarios para que se vea la nube sin perder la forma de la misma.
- **Implementación del Servidor**. Gracias a la organización de los datos el proceso de streaming se simplifica. El servidor usa datos preprocesados en los procesos anteriores y utiliza el framework dMVC (Implementado en java con Netty) para recibir peticiones y enviar datos de manera asíncrona.

VISUALIZACIÓN DE ARCHIVOS MULTIDIMENSIONALES. NETCDF

El visor de datos multidimensionales permite una visualización en 3 dimensiones de sets de datos multidimensionales tanto escalares como vectoriales.
Existen diferentes políticas para colorear los gráficos (rampas de colores, escala de colores, etc). También puede hacer animaciones basadas en una dimensión temporal.
Se pueden hacer cortes en las dimensiones espaciales de los datos y, de esa forma, comparar diferentes ángulos del mismo set de datos.

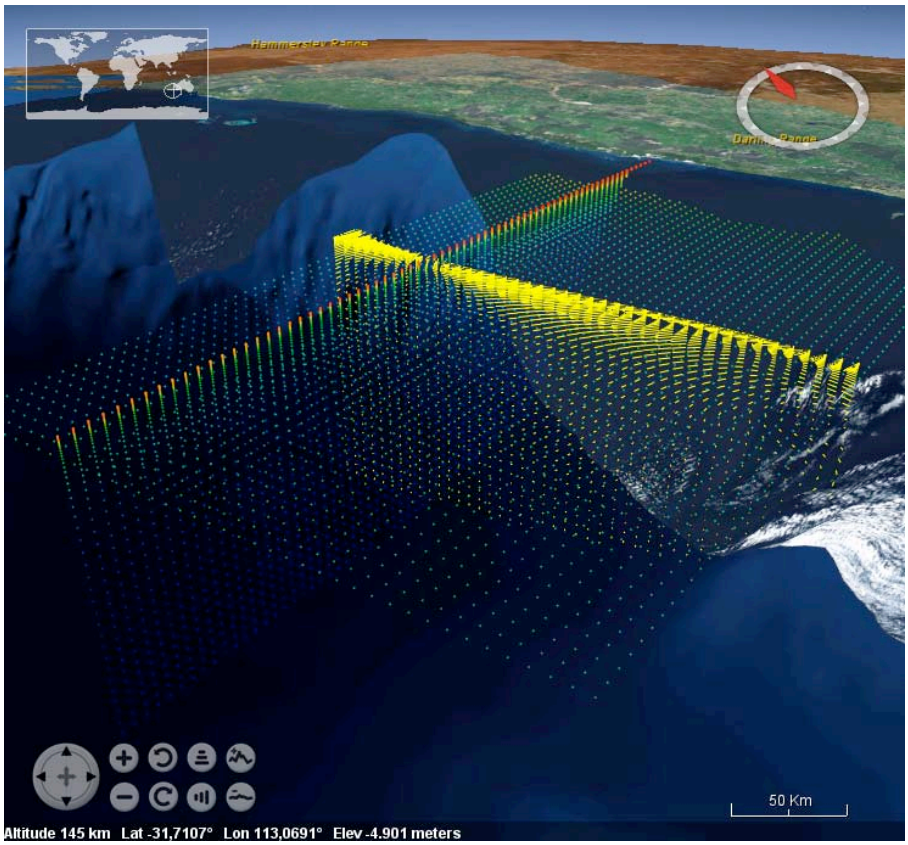

Figura 4: *Cortes del mismo set de datos NetCDF. Vectores en amarillo, corte horizontal con la salinidad del agua y vertical con la temperatura*

Actualmente, el visor de datos multidimensionales puede visualizar datos con formato NetCDF (utilizando la librería open-source NetCDF) y datos de bitmaps 3D (como los utilizados para tomografías computadas).

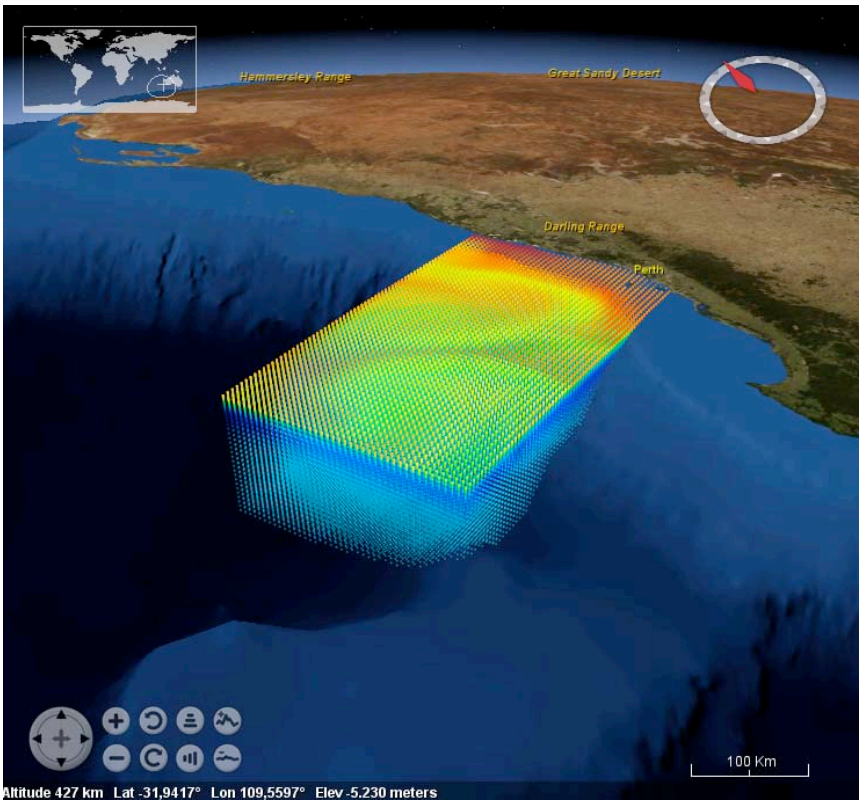

Figura 5: *Temperatura del Agua. NetCDF*

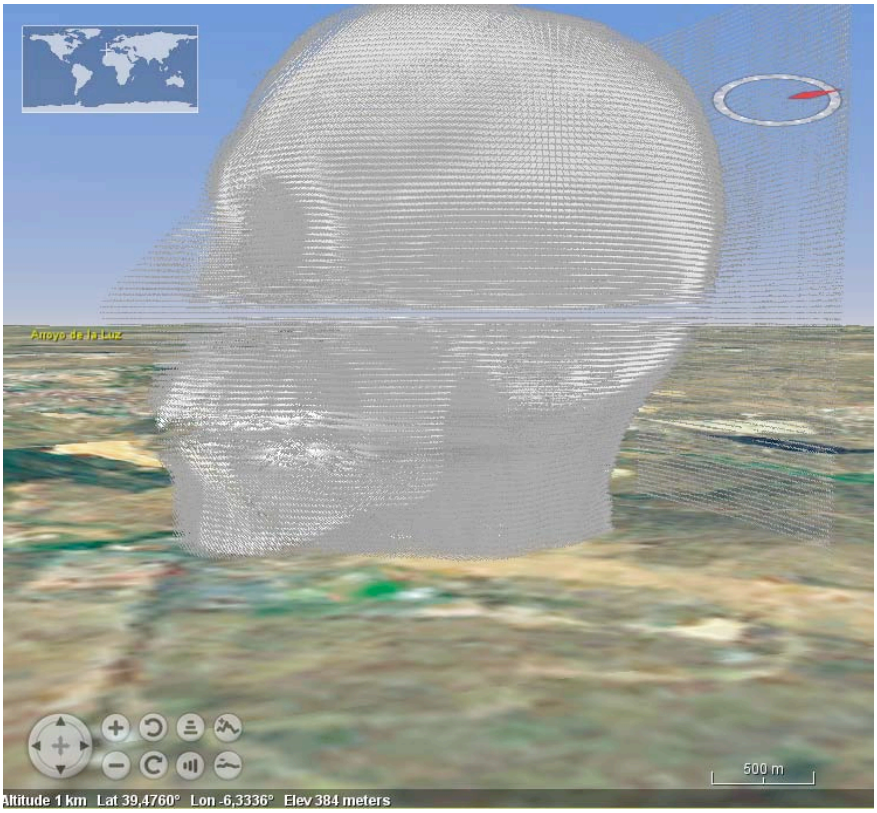

Figura 5: *Tomografía computada. Bitmap 3D*

VISUALIZACIÓN DE ARCHIVOS GIS VECTORIALES

La implementación del rendering (como el resto del glob3) se apoya en la librería geométrica multidimensional Euclid. Para la lectura de los datos se utiliza geotools, y un proceso convierte las geometrías de JTS (entregadas por geotools) en geometrías de Euclid.
Las geometrías se indexan utilizando un Quadtree.

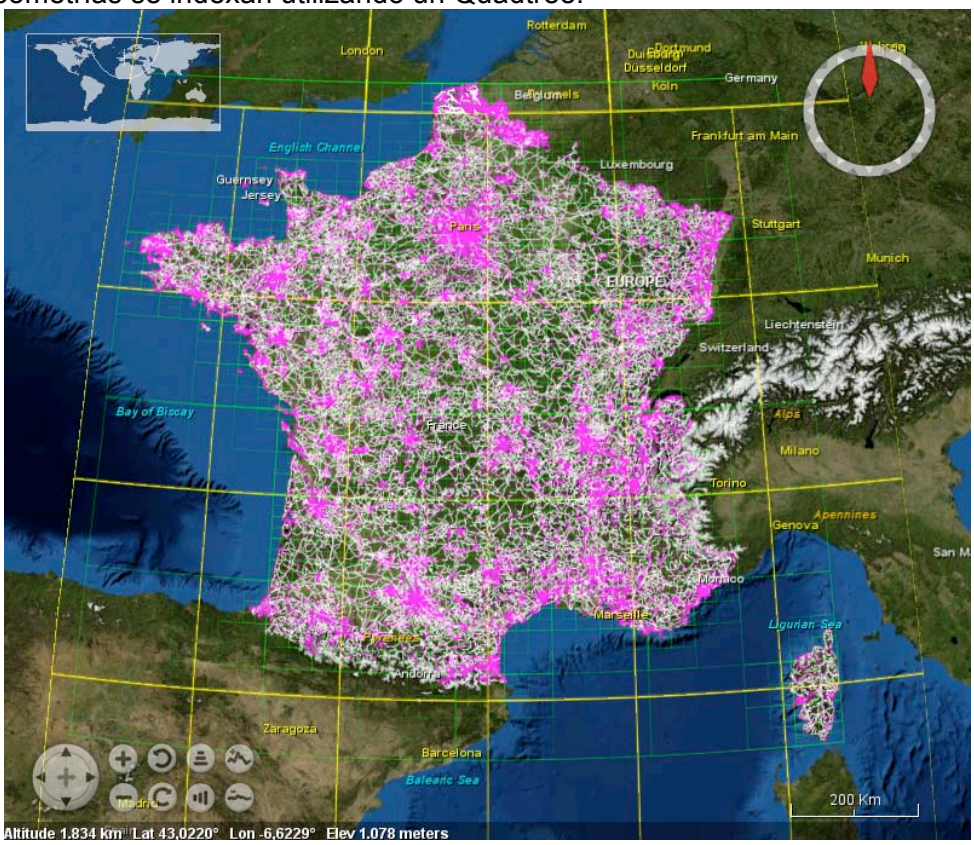

Figura 6: *Archivo vectorial de gran tamaño renderizado en glob3*

El Quadtree sirve a 2 propósitos.

- Selección eficiente de geometrías con queries espaciales. Se seleccionan las geometrías que corresponden a cada Tile (zona amarillas) de forma eficiente.
- Eliminación, en el proceso de rendering, de cuadrantes completos del Quadtree por su tamaño proyectado en pantalla.

Para aumentar la eficiencia del proceso de renderizado, además de las técnicas de LOD y uso inteligente del Quadtree arriba mencionadas, se utiliza un rendering multi-hilo que escala con los modernos procesadores multi-núcleo. Además, se utiliza tanto un cache en memoria como un cache en disco del los resultados del rendering.

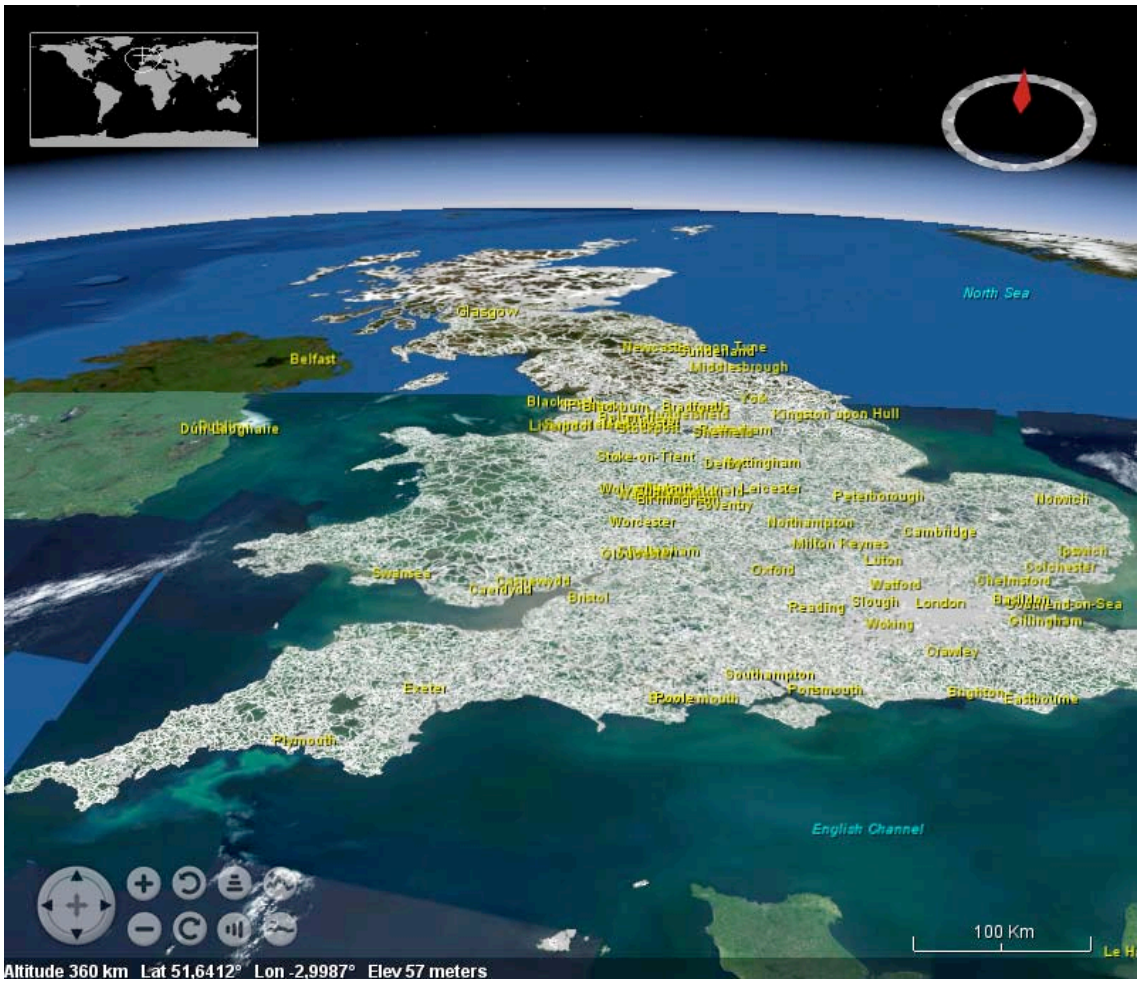

Figura 7: *Visualización de cerca de 2 millones de geometrías*

VISUALIZACIÓN GIGA-IMÁGENES Y FOTOS 360

La visualización de giga-imágenes (imágenes de cientos de megapixeles) o panorámicas es difícil debido al gran tamaño de estas imágenes, que suelen medir más de 100 MB. Para poder mostrarlas on-line (y en local también), las imágenes tienen que ser preprocesadas. Este proceso es totalmente automático y muy parecido para ambos tipos de imágenes:

- Primero se analiza la imágen y se determina cuantos niveles de zoom hay que crear. Estas informaciones se guardan en un fichero que se lee a la hora de dibujar.
- Se utiliza un proceso de tiling para crear los diferentes niveles de zoom. Cada nivel contiene toda la información de la imagen, pero a diferentes grados de resolución y repartidos entre diferentes cantidades de sub-imágenes. Mientras que el primer nivel solo consiste en dos sub-imágenes con una resolución altamente reducida, el último nivel típicamente contiene cientos de sub-imágenes con la resolución original de la imagen.

La presentación de los dos tipos de imágenes en el globo es diferente: Mientras las panorámicas se dibujan directamente dentro del contexto 3D, las giga-imágenes se muestran en una ventana aparte. No obstante, los algoritmos de dibujo son muy parecidos:

- Primero se carga el fichero de índice que contiene la información sobre niveles de zoom, ubicación de los ficheros, etc.
- El nivel de zoom que se muestra se determina según el espacio que ocupa el tile en pantalla . Cuando se acerca la cámara y el tamaño del tile en pantalla sobrepasa un determinado límite, éste está reemplazado por los tiles correspondientes del siguiente nivel de zoom, que tienen una resolución más alta.
- Para mejorar los tiempos de descarga on-line, se usa el mismo mecanismo de descarga asincrónica y cacheo de datos que se usa para los modelos 3D.

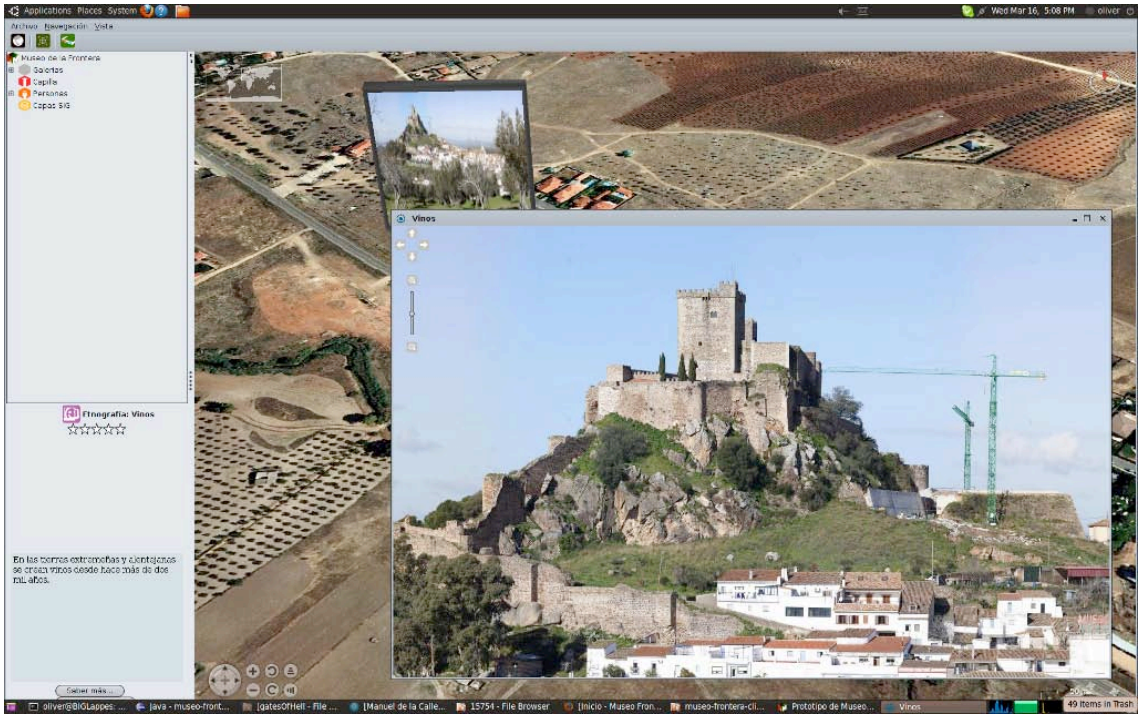

Figura 8: Cliente asincrónico de Giga Imágenes

VISUALIZACIÓN DE MODELOS 3D

La posibilidad de visualizar modelos 3D en el globo abre la puerta a una multitud de aplicaciones posibles, desde simples icónos tridimensionales que aumentan la experiencia visual, hasta mundos virtuales enteros.
Como formato de los modelos, elegimos .obj, porque casi todas las herramientas de modelado 3D tienen la capacidad de exportar a este formato. En el estado actual, se pueden visualizar modelos con una geometría muy compleja, con texturas y transparencias.
El algoritmo de visualización consiste en los siguientes pasos:

- **Parsear el fichero .obj**. Se leen las geometrías, materiales, etc. y se crean los objetos correspondientes. Los ficheros se leen de forma asincrónica tanto desde local o desde una URL, y se cachean en la máquina del cliente.
- **Traducción a OpenGL**. Los datos leídos se traducen a comandos OpenGL que se guardan en displaylists. Para minimizar el número de cambios de estado de OpenGL, hay diferentes display lists según las propiedades de los poligonos (si tienen textura, transparencia, etc). También se le asigna un bounding volume.

- **Dibujar.** Cuando aumenta el número de modelos 3D, puede bajar la performance. Por lo tanto hemos implementado algunas optimizaciones:
  - **Tamaño mínimo**: Antes de dibujar el modelo, se aproxima el número de pixeles que ocuparía en la pantalla (utilizando su bounding volume). Si este número está por debajo de un cierto límite, no se dibuja.
  - **Scene Graph**: También hemos implementado un sencillo scene graph. Además de las ventajas que ofrece en el manejo de los modelos, nos permite hacer un view frustum culling jerárquico. De esa manera se puede detectar muy rápidamente qué modelos/grupos de modelos están fuera de la vista y por lo tanto evitar dibujarlos.

Figura 9: Visualización de modelo 3D

CONCLUSIONES

IGO SOFTWARE continúa con el desarrollo de glob3 y esperamos pronto tener una versión usable para usuarios finales de todo el código que ya fue liberado en el mes de noviembre. En la actualidad todo este software está disponible para desarrolladores y se puede implementar cualquier aplicación como las mostrada. Queda por mejorar la documentación y los ejemplos. Además en estos momentos nuestra API se refactoriza continuamente lo que complica los desarrollos de terceras partes pero estamos encantados de dar soporte a todos los futuros desarrolladores.

Glob3 seguirá creciendo y creemos que es un buen software para desarrollar aplicaciones SIG 3D.

REFERENCIAS

- http://www.igosoftware.es
- http://www.cdti.es
- M. De la Calle, V. Olaya, D. Gómez-Deck. Desarrollo personalizado de aplicaciones SIG 3D. 4 Jornadas SIG libre Girona 2010

- D. Gómez-Deck, M. De la Calle, V. Olaya S3xtante. A 3D GIS Java Framework. Foss4g 2010. Barcelona
- http://worldwind.arc.nasa.gov/java/
- http://www.opengeospatial.org/standards/wms
- http://www.opengeospatial.org/standards/wfs