Contenido

[**12/01**:Cesium 2](#_Toc410718652)

[Tiles 2](#_Toc410718653)

[Crear mesh con vértices + faces. 2](#_Toc410718654)

[Crear mesh con datos de QuantizedMeshTerrainData 3](#_Toc410718655)

[**19/01**: Tile obtenido representado en un mesh(malla). 3](#_Toc410718656)

[Mapbox <-> Cesium 4](#_Toc410718657)

[**20/01:** Añadir varios Tiles en la escena 6](#_Toc410718658)

[Merge tiles en un mesh 6](#_Toc410718659)

[Añadir textura 7](#_Toc410718660)

[Añadir GUI 12](#_Toc410718661)

[Ajustar Mapbox al Mesh 13](#_Toc410718662)

[Descargar imagen Mapbox 15](#_Toc410718663)

[**IMPORTANTE:** Rellenar los laterales. 15](#_Toc410718664)

[Unir textura Mapbox con Mesh obtenida con Cesium + Three.js 15](#_Toc410718665)

[**03/02 Más cosas** 16](#_Toc410718666)

[Uniendo el pfc 17](#_Toc410718667)

# **12/01**:Cesium

<http://cesiumjs.org/demos.html>

# Tiles

Obtener lat y long máximas y mínimas para establecer bien el tile correspondiente. Revisar si *omnivore.gpx* me da los valores máximos y mínimos.

Ajustar imagen mapbox como textura para el terreno con las elevaciones obtenidas de CesiumJS

Imagen de mapbox ponerla sobre el terreno a modo de prueba. Tiene que ser una imagen guardada en el servidor para que sea reconocida como textura.

Guardar una imagen en el pc, Ajax.

<https://ausdemmaschinenraum.wordpress.com/2012/12/06/how-to-save-a-file-from-a-url-with-javascript/>

blob

<http://msdn.microsoft.com/en-us/library/ie/hh779016%28v=vs.85%29.aspx>

<http://weworkweplay.com/play/saving-html5-canvas-as-image/>

**Problemas:** Para añadir una textura a un objeto el fichero debe existir en local(servidor), mapbox no deja importar los static map a ficheros, solución?

<https://www.mapbox.com/help/export-image-pdf/>

**solución**?: <http://stackoverflow.com/questions/6306935/php-copy-image-to-my-server-direct-from-url>

Three.js texture

<http://solutiondesign.com/webgl-and-three-js-texture-mapping/>

Cesium ver los índices con three.js, sino ver el quantized mesh.

<http://threejs.org/examples/obj/male02/Male02_dds.js>

# Crear mesh con vértices + faces.

Problemas:



El problema es que al añadir la geometría a la escena esta no puede leer una posición del array de vértices o faces ya que seguramente la posición exceda el tamaño de uno de los arrays.

**Solucionado**: no se estaban creando bien los puntos de la geometría (x,y,z) ni las caras. Cada vertice debe ser del tipo *Vector3* y cada cara debe ser del tipo *Face* ambos proporcionado por Three.js

# Crear mesh con datos de QuantizedMeshTerrainData

**Solucionado**: pero la salida no corresponde a lo esperado.

**Solucionado**: los datos anteriormente no estaban siendo pasados correctamente.

# **19/01**: Tile obtenido representado en un mesh(malla).

Se ha obtenido un mesh correspondiente al tile en esa coordenada. Según Wikipedia un tile es la parte [gráfica](http://es.wikipedia.org/wiki/Gr%C3%A1fica) de cada [videojuego](http://es.wikipedia.org/wiki/Videojuego) que puede ser utilizada para completar partes de un [fondo](http://es.wikipedia.org/wiki/Fondo) por medio de un [tileset](http://es.wikipedia.org/w/index.php?title=Tileset&action=edit&redlink=1) (conjunto de tiles). Cesium nos devuelve un tileset que es un *quantized-mesh,* un árbol cuaternario piramidal que contiene las alturas (heightmaps) según el TMS. (Tile map service layout and global geographic profile).

**19/01**:La idea es descargar varios tiles y crear sus respectivas mallas para obtener una mayor representación del terreno.

**19/01**: Luego de obtener varios tiles unirlos en un solo mesh con three.js

**19/01**: Añadir textura *Mapbox* al mesh con la ruta *gpx*.

~~¿Tamaño de cada tile proporcionado por cesium?~~

No es necesario en un principio ya que sabiendo el tile que seleccionamos sería suficiente con seleccionar el siguiente.

~~¿Coordenadas que abarca, dirección del tile, escala de los datos?~~

Antes de añadir un mesh a la escena le indico la posición o cualquier atributo que quiera modificar.

# Mapbox <-> Cesium

Relación entre el nivel del tile y el zoom *z* en los mapas de Mapbox.

Coordenadas para el ejemplo: ~~latitud~~-longitud 2.002 y ~~longitud~~ latitud 43.317







Imagen de San Sebastián con nivel de zoom 12 y mesh obtenida con las mismas coordenadas con nivel de detalle 12, se puede ver que no son niveles comparables.

Cesium posee un objecto Rectangle que contiene los valores en radianes de las latitudes y longitudes. Se pedirán las 4 esquinas para saber las coordenadas en esos puntos y así obtener el tamaño de nuestro tile en *real*.

*var positionLonLat = Cesium.Cartographic.fromDegrees(longitude, latitude);*

*positionTileXY = aCesiumTerrainProvider.tilingScheme.positionToTileXY(positionLonLat,12);*

*var rectangleTileXY = aCesiumTerrainProvider.tilingScheme.tileXYToRectangle(positionTileXY.x, positionTileXY.y, 12);*

Con esas instrucciones obtenemos el rectángulo con los datos.

Transformar de radianes a grados, cesium tiene una función pero al intentar ejecutarla igual que todas me da error.

Como es fácil la conversión creare una función propia .



Cada tile es de aproximadamente 4,9km por lado así que habrá que buscar una relación con esos datos.

<http://en.wikipedia.org/wiki/Decimal_degrees>

<http://cesiumjs.org/Cesium/Build/Documentation/Rectangle.html>

# **20/01:** Añadir varios Tiles en la escena

Se deberán añadir varios Tiles y unirlos para crear un terreno que concuerde con las coordenadas, se debe tener en cuenta la ruta que realizo la persona para que la cantidad de tiles abarque la ruta especificada.

**Problemas: ¿**posiciones de los tiles? ¿Cómo ponerlos bien? ¿Orden en la escena?



Tiles añadidos manualmente ahora hay que agregar tres más y crear una estructura con todos, es decir un solo mesh.

# Merge tiles en un mesh

**Problemas**

****

**Solucionado:** Para llamar a la función *merge()* se debe hacer desde una variable del tipo THREE.Geometry y los argumentos deben ser del tipo THREE.Mesh.geometry y THREE.mesh.matrix. Antes de llamar a la función *merge()* se debe actualizar la matriz del objeto THREE.Mesh con la función *updateMatrix()*.

<http://stackoverflow.com/questions/24353756/migrating-geometryutils-merge-to-geometry-merge>

Posible solución a error: <http://stackoverflow.com/questions/24422289/three-js-merging-geometries-and-mesh>



# Añadir textura

Posible solución a errores: <https://github.com/mrdoob/three.js/wiki/Updates>

**Problemas:** Al añadir texturas salta el siguiente error:

[.WebGLRenderingContext]GL ERROR :GL\_INVALID\_OPERATION : glDrawElements: attempt to access out of range vertices in attribute 2

Possible solución:

No funciona esta solución: <http://stackoverflow.com/questions/21200386/webgl-gl-error-gl-invalid-operation-gldrawelements-attempt-to-access-out-of>

No funciona esta solución: <http://stackoverflow.com/questions/16531759/three-js-map-material-causes-webgl-warning>

“unwrapping the uv's for each mesh i want to have a texture on.”

¿Solución? : <http://stackoverflow.com/questions/19875469/threejs-texture-causes-gl-error-gl-invalid-operation-gldrawelements>

Añadir textura o múltiples texturas **VERTEX** : <http://solutiondesign.com/webgl-and-three-js-texture-mapping/>

**SOLUCIÓN**: **UV Mapping**

****

La solución es entender uv mapping al ser geometrías construidas manualmente. geometry.faceVertexUvs[0]

Para que se pueda añadir una textura la cantidad de caras debe ser igual a geometry.faceVertexUvs[0], por lo tanto, están directamente relacionadas.

Solución:

* Unir cada vértice de cada geometría antes de crear el mesh que las combina a todas con la función *merge*.
* o al tener la mesh final lograr que se cumpla la condición.
* <http://threejs.org/examples/misc_uv_tests.html> MIRAR!



La idea es la siguiente:

* Cuando una textura es añadida a un objeto se indica en el vector *geometry.faceVertexUvs[0]* cada coordenada que ocupará la textura, es decir, si suponemos que un plano está compuesto por dos triángulos se indica de la siguiente forma:  
    
  *vector1:{ (0,0), (0,1), (1,1)}* y *vector2:{(0,0),(1,0),(1,1)}* de este modo indicaremos que la textura debe pertenecer a todo el plano. (**Explicación más clara**).
* Se deben crear los *faceVertexUvs[0]* para que la textura pueda ser implementada en la geometría:
  + Se debe tener en cuenta la cantidad de caras ya que debe coincidir con el tamaño del vector *faceVertexUvs[0]*.
  + Cada cara se forma como un triángulo, por lo tanto debemos buscar la forma de unir todas las caras. Usaremos el total de cara para sacar una relación entre los puntos del triángulo, se debe tener en cuenta que los cuatro puntos de la geometría son (0,0), (1,0), (0,1), (1,1).



**Solución Problema:** <http://stackoverflow.com/questions/20774648/three-js-generate-uv-coordinate>**.** Usando el esto <http://paulyg.f2s.com/uv.htm> para crear el vector uvs – mapping.

**Problemas**





En la primera imagen creo un mesh con los datos de cesium y añado una textura, la relación del tamaño con la imagen es buena.

En la segunda imagen creo un mesh usando cuatro geometrías y añado una textura, la relación entre la textura y el objeto no es correcta porque se repite cuando debería ser una extendida.

**SOLUCIÓN:** cuando se hace un merge no se debe actualizar la matriz de cada mesh que se va a añadir ~~geometry.updateMatrix();~~

**Problema nuevo:** Al quitar *geometry.updateMatrix();* se superponen los mesh.



# Añadir GUI

Para la inclusión tuve que eliminar los estilos de los botones de Mapbox y eliminar el trackball control de la cámara.

# Ajustar Mapbox al Mesh

He encontrado la clase Rectangle en la cual dando las coordenadas Sur-oeste y Nor-este centrará el mapa en esas coordenadas, por lo tanto encajaría en el mesh creado a partir de las coordenadas, la idea:

Coordenada ruta gpx, obtener Tile con Cesium.

Obtener coordenadas sur-oeste y nor-este del Tile.

Mapbox: Clase rectangle y obtener textura para insertar en el mesh.



Con lo anterior obtengo una imagen del mapa con el rectángulo que coincide el mesh:

¿ Cómo guardar él rectángulo obtenido en la imagen anterior ?

<https://www.mapbox.com/mapbox.js/api/v1.6.1/l-map-class/>

<https://www.mapbox.com/mapbox.js/api/v0.6.6/#mapbox.layer>

<https://www.mapbox.com/developers/api/static/>

He realizado un ajuste manual de la zona a partir desde el punto central, es decir, la primera coordenada ofrecida:



No es exacto ya que se pierde un pedazo de arriba y otro de abajo, pero para efectos simple sirve.

**Problemas:** Esta imagen es válida para un punto, es decir, para un tile de Cesium. Si se cambia de tile en cesium debería obtenerse otra imagen (coordenadas, punto central, entre otros).

Mapbox opciones de color:

<https://www.mapbox.com/mapbox.js/api/v2.1.5/l-path/>

Rectangle svg, convertir a png.

<http://spin.atomicobject.com/2014/01/21/convert-svg-to-png/>

<http://stackoverflow.com/questions/11567668/svg-to-canvas-with-d3-js/23667012#23667012>

**Solución momentánea**

Se ha restringido la vista del mapa ajustado la vista a los límites nor-oeste y sur este. Además de ajustar el tamaño del bloque contender de la vista y especificar el Zoom al nivel que deseamos.

**IMPORTANTE:** Ver cuantos mesh debe ocupar la ruta, comprobar por coordenadas de la ruta GPS y los límites del tile que nos ofrece cesium.

*La idea sería crear todas las texturas para que el usuario luego pueda ir poniéndolas en el mapa.*

Controlar que en los dos html exista la variable de sesión con el nombre del fichero subido, si no existe que redirija a la página principal para subir fichero.

# Descargar imagen Mapbox

**Error y solución**: pequeño error cuando envío el url de la textura ya que desaparece un símbolo ‘+’ pero es añadido con str\_replace en el lado del servidor.

Modificar los colores de la ruta del gpx para que no coincida con el estilo del mapa.

**Importante:** Se debería crear el objeto mesh o al menos obtener los datos para establecer la textura que se elegirá. Para ello necesitamos obtener el tile con cesium e indicarlo con Mapbox.

**Importante:** Recordar que la ruta debe estar en el mesh, comprobar que estén todos los puntos de las coordenadas. **Probar con la coordenada central**. Obtengo el mismo tile, hacer una función.

**Importante**: PFC funciona con lectura de datos de rutas con wikiloc por el formato aunque el formato es el mismo en todos los ficheros gpx, cambia la cabecera.

**Importante**: Control de errores si borramos el fichero.

**Cambios**: no cambia el mesh.

# **IMPORTANTE:** Rellenar los laterales.

Como creamos un mesh nuevo el cual no tiene laterales ni fondo, la idea es crear una función que rellene esos espacios, así se obtendrá una geometría para imprimir.

He clonado la geometría para poder añadir los mismos vértices a la geometría inicial, sino no se podría ahora hay que crear las caras.

# Unir textura Mapbox con Mesh obtenida con Cesium + Three.js

Unida sin problemas. Utilizar los vectoresUVS en las geometrías.

# **03/02 Más cosas**

**Problemas:** Las texturas (imágenes) dependen del tamaño del navegador.

## Sacar todos los mesh de la ruta.

A partir de una ruta obtener todas las mallas.

<http://html-color-codes.info/codigos-de-colores-hexadecimales/>

Si recorremos toda la ruta obteniendo todos los tiles podemos identificar las coordenadas límites para toda ruta (función *checkTile* y *moreMesh*), y fijamos la vista del mapa entre esas coordenadas *map.setMaxBound* ahora hay que obtener todos las mallas de la ruta y la textura de cada malla.

Obtener cada imagen para cada mesh, se puede hacer sin que el usuario vea, es decir, una página intermedia.

## Sacar todas las imágenes de los tiles (mapbox) de la ruta.

Sacar todas las imágenes de Mapbox correspondiente a la ruta, ¿JavaScript?

Se han dibujado rectángulos para identificar los tiles de cada coordenada, verificar con rutas más largas. Quizás haya problemas por el tema de las coordenadas de los tiles, de momento se que tiles son, por lo tanto, obtener los mesh correspondientes y la imagen para cada uno.

El elemento sobre el cual Mapbox dibuja el mapa es del tipo *svg*

## Relleno correcto mesh.

Relleno correcto del relieve de un mesh que obtenemos con cesium.

## Ajustar vista imagen -> Texture.

A modo solución simple he optado por copiar el fragmento de imagen con php, la idea sería hacerlo todo con JavaScript. El ajuste momentáneo se ha hecho jugando con los estilos pero para cada resolución es distinta así que hay que buscar otra manera de obtener el tile.

# Uniendo el pfc

Expresiones regulares: <http://webintenta.com/validacion-con-expresiones-regulares-y-javascript.html>

<http://www.w3schools.com/html/html5_webstorage.asp>

Variables de sesión, cuando se cierra el navegador se borran o al terminar de editar el modelado.

Validar y subir al servidor y/o almacenar en el lado del cliente.

Cargar ruta en el mapa MapBox, obtención datos AJAX.

Página de inicio, subir fichero al servidor y/o almacenar en el lado del cliente.

Crear mesh, puede ser uno o varios. Al crear varios y unirlos problemas con merge.

Crear textura de mesh con Mapbox. Debe contener la ruta y coincidir en tamaño.

Pasar un array php a JavaScript: <http://stackoverflow.com/questions/14480857/passing-array-through-ajax-from-php-to-javascript>

Iconos Mapbox: <https://www.mapbox.com/base/styling/icons/>

Cuando leemos un gpx y obtengo las coordenadas la primera es la latitud y la segunda es la longitud.

GUI en el navegador para acceder más fácil a los contenidos. <http://stemkoski.github.io/Three.js/GUI-Controller.html>

Los dos últimos diseños creados los borre porque dan error, debe ser por el enlace. En los dos primeros funciona correctamente~~.~~ **~~PROBLEMA:~~** ~~por la condición de control para mostrar los mapas.~~