Tecnología modular orientada a shaders en la generación de grandes entornos.

TOVAR PEREZ, Carlota;

CITEF Centro de Investigación de Tecnologías Ferroviarias.

Universidad Politécnica de Madrid.

España

ctovar@etsii.upm.es ; citef-gjimena@etsii.upm.es ; jmcabanellas@etsii.upm.es ;

Resumen:

Este artículo presenta las últimas investigaciones y desarrollos en tecnología modular, la repetición optimizada de una misma geometría o módulo, para la generación de grandes entornos virtuales para los simuladores que desarrolla CITEF. La tendencia actual va por redirigir la máxima parte posible del cálculo gráfico a la GPU, descargando en la medida de lo posible la CPU, más lenta y menos apta para el cálculo gráfico. La tecnología modular, que se utiliza con éxito desde hace tiempo en los simuladores de entrenamiento en conducción, está especialmente indicada para la aplicación de operaciones en la GPU mediante la programación de shaders. De esta forma se consigue reducir sustancialmente el número de entidades geográficas del entorno y aumentar la diversidad y flexibilidad del mismo. Esto se consigue discretizando el entorno en una serie de módulos instanciables y aplicando los shaders de forma adecuada en base solamente a unos pocos parámetros. Son múltiples las ventajas que proporciona esta forma de generación, ahorro en tamaño del escenario, tiempos de carga y requerimientos de recursos, mayor flexibilidad y claridad del scene graph y una considerable mayor capacidad de actualización del entorno.

Palabras clave: shaders, modularidad, entornos virtuales.

Innovación, tecnología gráfica, proyectos.

Abstract:

This paper presents the latest research and developments in modular technology. That is the optimized repetition of the same geometry or module, for the generation of large virtual environments for simulators that develops CITEF. The current trend is on redirect the maximum possible share of graphical calculation to GPU, discharging to the extent possible the CPU, slower and less suitable for the graphical calculation. The modular technology, which has been successfully used for some time in the simulators for driving training, it is particularly suitable for the implementation of operations on the GPU through programming shaders. In this way a substantial reduction of the number of geographical entities in the environment and a increase of diversity and flexibility of the environment. This is achieved discretizing the environment in a series of instantiated modules and using shaders in an appropriate manner based only on a few parameters. There are many advantages provided by this form of generation, saving size of the scene, loading times and resource requirements, greater flexibility and clarity of the scene graph and a more substantial upgrade capacity of the environment.

Keywords shaders, modularity, virtual environments.

Innovation, graphic technology, research.

1. INTRODUCCIÓN

El empleo de simuladores virtuales de conducción como herramienta de aprendizaje y entrenamiento es una práctica totalmente afianzada y en contínua expansión, existiendo toda una gama de posibilidades que va desde los simuladores de alto ,medio y bajo coste.

Todo ello gracias a la vertiginosa evolución del hardware, que hace aumentar en la misma medida las exigencias de realismo de los escenarios de dichas representaciones virtuales.
Las elevadas capacidades de procesamiento de las actuales GPUs [1] han permitido descargar de tareas a la CPU, algo de vital importancia en las simulaciones de conducción terrestre y aumentar la carga geométrica a renderizar, convirtiendo el bus de comunicación CPU-GPU en el nuevo cuello de botella.
En respuesta a este cambio de prioridades en la optimización de un escenario, han comenzado a surgir una nueva generación de algoritmos de adecuación del nivel de detalle (*lods*) cuya primitiva de trabajo ha dejado de ser el triángulo [1] a [4]. En su lugar se emplean bloques de triángulos (*batchs*), optimizados en tiempo de precarga de manera que se garantice una óptima organización espacial de éstos (*stripping*) agilizando así el trasbase CPU-GPU.

Siguiendo esta última línea investigadora, la Tecnología Modular, metodología empleada para la construcción de entornos virtuales del CITEF, ha convertido al *módulo* en su primitiva base. Previa búsqueda de unos patrones repetitivos del entorno, tanto geométricos como funcionales, la Tecnología Modular discretiza el escenario en un número finito de módulos o porciones de entorno, que tras ser instanciados y sometidos a una serie de transformaciones geométricas mediante *shaders,* se ensamblan meticulosamente reproduciendo el escenario virtual.

1.1. La construcción de grandes entornos en simuladores de conducción terrestre en la actualidad

Se entiende por metodología constructiva de un entorno virtual al conjunto de procedimientos que permiten interpretar procesar, analizar, modelizar, optimizar y representar virtualmente una determinada información de partida.
A la hora de elegir la metodología más idónea para la generación de grandes entornos virtuales en simuladores de conducción terrestre, resulta decisiva la elección de las estructuras de datos que respondan a las restricciones impuestas por el entorno a representar, que minimicen el consumo de CPU y maximicen los recursos ofrecidos por las actuales GPUs.
En el caso de las simulaciones de conducción terrestre, el tipo de trayectorias a lo largo de las cuáles se va a circular será un factor decisivo en la elección del tipo de estructuras de datos más idóneo. Según esto se pueden distinguir dos tipos de entornos:

- ***Entornos con trayectorias sin prefijar***. Este es el caso de simulaciones de combate en entrenamiento militar**.** Puesto que las trayectorias no imponen ninguna restricción geométrica, las estructuras de datos son más flexibles (regulares o irregulares) y la variedad de algoritmos para la creación de modelos multirresolución es muy amplia. En este ámbito, las últimas tendencias apuntan, como se comentó en la introducción, a la liberación de carga a la CPU y agilización del trasbase CPU-GPU mediante la nueva primitiva de trabajo: el *batch* o loseta [6]**,** [7]**.** Los *batchs* o losetas se construyen en tiempo de precarga, optimizando al máximo la organización de triángulos en los mismos (*strips*).Gracias al *batch* la métrica de selección deja de evaluarse a nivel de vértice, realizándose a nivel de *batch*, liberando enormemente la carga de la CPU. Si bien esto supone una menor optimización en el número de polígonos a representar, la elevada capacidad de procesamiento de las actuales GPUS hace que esto no sea ningún problema.
- ***Entornos con trayectorias prefijadas***. Este es el caso de simulaciones de conducción ferroviaria y urbana, caso que se aborda en el presente artículo. El gran realismo exigido en las proximidades de las trayectorias puede conseguirse únicamente mediante la inserción geométrica de las trayectorias en el terreno, lo que tiene fuertes repercusiones en la selección de las estructuras de datos a emplear. La inserción de dicha geometría implica un reajuste y retriangulación de la malla del terreno en los alrededores de la misma [8], con el fin de adaptarse a la mayor resolución de la trayectoria. Esto supone un aumento notable en el número de cálculos a realizar, acrecentándose así el principal inconveniente de los algoritmos

de adecuación del nivel de detalle contínuo (*contínuos lods*): su elevado consumo de CPU. Los niveles de detalle discretos [9] a [12] se convierten así en la alternativa preferida, siendo sus principales inconvenientes:

- el elevado consumo de memoria para almacenar en tiempo de precarga la geometría asociada a los diferentes niveles de detalle,
- la escasa flexibilidad constructiva,
- el no aprovechamiento de las funcionalidades ofrecidas por las actuales GPUS y
- los elevados precálculos necesarios.

En respuesta a éstas últimas necesidades surge la Tecnología Modular. La Tecnología Modular emplea la traza de las trayectorias como punto de partida de sus desarrollos y a partir de ésta genera el entorno circundante mediante el ensamblado y repetición de un número finito de módulos. De esta manera se explota una de las herramientas base en la optimización de recursos: la instanciación. La determinación del número y diseño geométrico óptimo de los módulos así como la variedad de familias de éstos serán decisivos para conseguir el realismo y velocidad de refrescos deseados en estas simulaciones. Por otro lado, aprovecha el conocimiento a priori de estos caminos para generar niveles de detalle discretos pseudo-variantes con el punto de vista. Esto se consigue con una discretización del entorno transversalmente a la trayectoria.

Como resultado se obtiene un escenario con todo el realismo que las simulaciones de conducción requieren, con las velocidades de refresco deseadas, con una cómoda interfaz de comunicación con el módulo de simulación y motor gráfico y todo ello de una forma totalmente automática y con la posibilidad de introducir cualquier reestructuración del entorno fácilmente, algo indispensable en este tipo de representaciones virtuales. Y lo que es más, eliminando los principales problemas asociados hasta el momento a los algoritmos de adecuación del nivel de detalle discretos (*discret lods*):

- Gracias a la instanciación y los shaders tan solo es necesario almacenar en tiempo de precarga un módulo recto de cada familia, por lo que la memoria deja de ser un problema.
- El entorno se sintetiza en base a una serie de parámetros repetitivos, por lo que los cálculos se simplifican.
- Lós módulos son sometidos a transformaciones geométricas mediante *shaders* en tiempo de ejecución, lo que permite elevar el realismo de estos escenarios. Estas transformaciones serán definidas en tiempo de precarga, mediante una serie de parámetros incorporados junto al scene graph.
- La modularidad del entorno dota a éste de una gran flexibilidad constructiva, pudiendo generar toda una diversidad de escenarios virtuales con unos mínimos parámetros constructivos.

2. SHADERS GEOMÉTRICOS PARA MÓDULOS

Un shader es un programa ejecutable en la propia GPU y puede actuar a diferentes niveles. Existen tres estándares actualmente de lenguaje de shader, HLSL (High Level Shading Language) propiedad de Microsoft para su empleo con Direct3D, GLSL (OpenGL Shading Language) que es el utilizado en este trabajo al estar desarrollado el motor gráfico sobre OSG basado en OpenGL y por último está CG propiedad del fabricante de aceleradores gráficos Nvidia. En [13] se tiene una referencia al GLSL del propio consorcio de OpenGL.

Los shader se dividen en dos tipos fundamentales: los vertex shaders que operan a nivel de vértice y datos adjuntos y que una vez compilados se ejecutan en el procesador de vértices y los fragment shaders que operan a nivel de píxel. Se están implantando ahora nuevos shaders geométricos, que serán capaces de introducir geometría en tiempo real, pero que todavía no son soportados por muchos aceleradores actuales.

La programación de shaders no está todavía generalizada pero ya hay un cierto bagaje en su uso en el desarrollo de video juegos [16] y en el desarrollo de técnicas avanzadas de renderizado o generación de imágenes [15].

En los desarrollos que realiza Citef para los motores gráficos se utilizan los shader para

obtener efectos de iluminación, perspectiva y manipulación del scene graph más avanzados de los que proporciona OSG.

En el caso de los shader que aquí se describen, estos van encaminados a la modificación geométrica del módulo a través de vertex shaders.

Con estos shader se reducen el número de geometrías que compone una familia modular y se amplían las posibilidades del mismo.

2.1. El módulo.

El módulo parte de un diseño básico constituido por una porción longitudinal recta de entorno. Este diseño básico se curva una serie de grados, constituyendo lo que se denomina un conjunto base de elementos modulares o módulos. De esta forma el conjunto base se caracteriza por la existencia de un número finito de módulos, de curvatura fija y de valores de curvatura discretos.

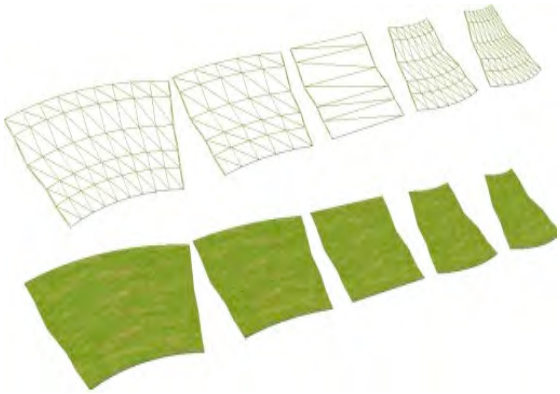

Figura 1 Conjunto base de módulos..

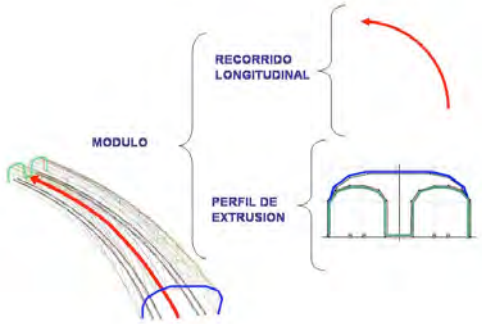

Figura 2 Componentes del módulo

El módulo está formado por tres elementos básicos: un recorrido longitudinal y un conjunto de perfiles transversales con sus correspondientes texturas. Estos elementos se encuentran asociados en estructuras denominadas solevados. La extrusión de los perfiles a lo largo del recorrido longitudinal genera la geometría tridimensional del módulo. Con el fin de hacer el texturado lo más versátil posible, cada perfil transversal se divide en tantos subperfiles, como texturas se deseen colocar. La asociación de cada subperfil, con su textura y recorrido longitudinal constituye un microsolevado. El conjunto de microsolevados transversales definen el módulo o solevado final. El conjunto base de módulos, junto a una adecuada algorítmica de posicionamiento, permite reproducir cualquier geometría sin agujeros ni solapamientos. El problema siguiente se plantea al pretender determinar los perfiles transversales de dichos módulos, para que puedan reproducir cualquier entorno. Surge así el concepto de familia de módulos, como el conjunto base de módulos, que comparten una misma definición de perfiles transversales. Estas familias verificarán unas relaciones de compatibilidad, que serán las que garanticen la ausencia de incoherencias o discontinuidades tras el correcto posicionamiento de los módulos

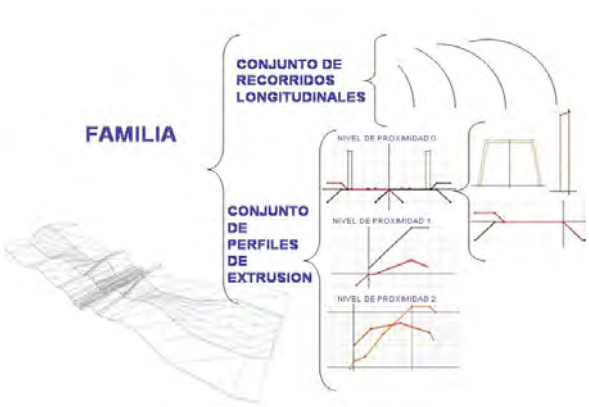

Figura 3 Componentes de una familia de módulos..

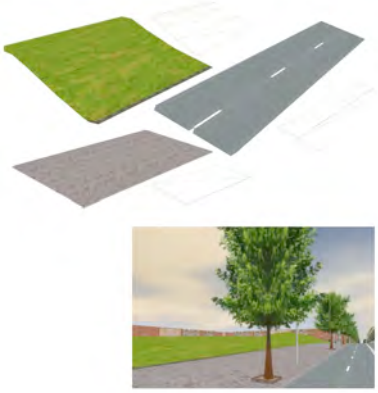

Figura 4 Ejemplo de empleo de módulos.

2.2. Shaders de transformación simple

La idea básica de la tecnología modular es desarrollar un módulo base y aplicándole diferentes deformaciones con distintos grados de intensidad se obtiene una familia modular adaptable a diferentes situaciones en el en entorno. El grado de intensidad se establece en un parámetro a la hora de crear un nuevo miembro de la familia modular. Para no tener una familia muy amplia de módulos el parámetro de la deformación ha de variar en incrementos discretos de cierto valor, p.e. 1º en la curvatura.

La idea de los shaders aquí propuestos es realizar estas deformaciones en tiempo de renderizado controladas por el motor gráfico al que se le suministra un valor instantáneo del parámetro de deformación. El parámetro puede entonces tomar un valor continuo cualquiera, lo que equivaldría a una familia modular deformada previamente de infinitos miembros.

Hay tres deformaciones básicas a realizar en un módulo. La primera es el curvado en planta. Como se puede ver en la

Figura **5** donde se tienen las vistas ortogonales en alambre y la perspectiva renderizada de un modulo que se ha curvado en la planta.

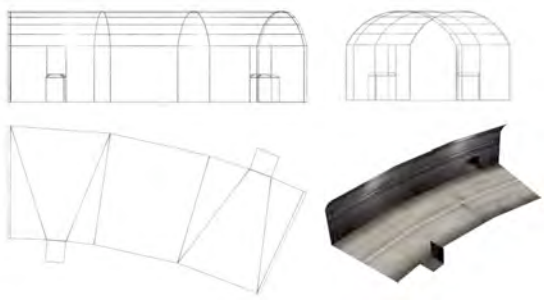

Figura 5 Curvado en planta.

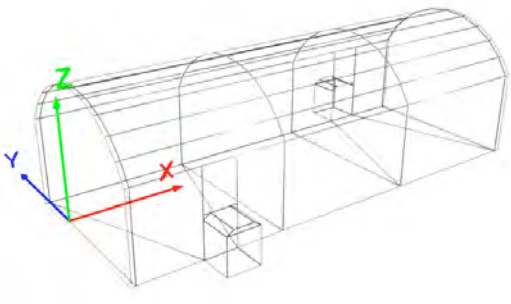

Figura 6 Sistema de referencia.

Tomando un sistema de referencia, Figura **6**, donde ***X*** es longitudinal apuntando hacia delante, ***Z*** el eje vertical hacia arriba e ***Y*** el transversal formando un triedro a derechas, la transformación matemática para un módulo de longitud inicial ***L*** sería la expresada en la ecuación

$$x' = \frac{L}{2\,sen(\alpha/2)}\left(sen\left(\alpha\left(\frac{x}{L}-\frac{1}{2}\right)\right)+sen(\alpha/2)\right) \qquad y' = \frac{L}{2\,sen(\alpha/2)}$$

$$z' = z$$

(1), donde ***x', y', z'*** son las coordenadas transformadas de un vértice originalmente en ***x, y, z***. La transformación curvaría un ángulo y la cuerda de la deformada seguiría siendo ***L***.

$$x' = \frac{L}{2\,sen(\alpha/2)}\left(sen\left(\alpha\left(\frac{x}{L}-\frac{1}{2}\right)\right)+sen(\alpha/2)\right) \quad y' = \frac{L}{2\,sen(\alpha/2)}\left(\cos\left(\alpha\left(\frac{x}{L}-\frac{1}{2}\right)\right)-\cos(\alpha/2)\right) \quad (1)$$
$$z' = z$$

Para una mayor eficiencia en la programación de los shader los términos de funciones trigonométricas se substituyen por aproximaciones polinómicas de las mismas.

Las familias modulares convencionales y que se han empleado en la construcción de los entornos de simulación respondían únicamente a esta deformación junto con pequeñas variaciones de escala que permitían un ajuste perfecto entre los módulos.

Las siguientes deformaciones básicas corresponden al curvado en alzado, Figura 7, que permite la transición suave entre cambios de pendiente, y a la torsión o peraltado del módulo, Figura **8**.

Las ecuaciones que rigen el curvado en alzado son formalmente iguales a las de **(1)** pero intercambiando ***y*** con ***z***. La ecuación de la torsión es la de un giro alrededor del eje ***X*** proporcional al avance en el mismo $x' = x \quad y' = y\cdot\cos(\alpha(x/L)) + y\cdot sen(\alpha(x/L)) \quad z' = -y\cdot sen(\alpha(x$ **(2)**.

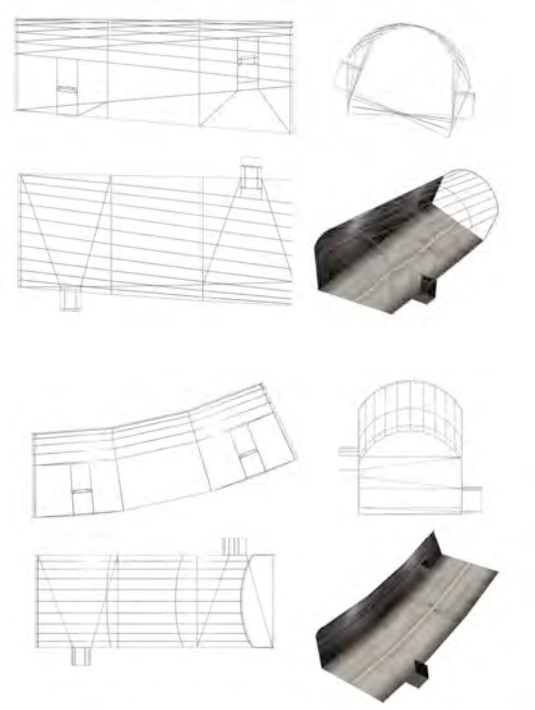

Figura 7 Curvado en alzado.

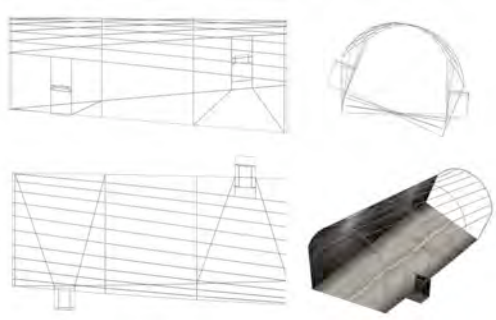

Figura 8 Torsión del módulo.

$$x' = x \quad y' = y\cdot\cos(\alpha(x/L)) + y\cdot sen(\alpha(x/L)) \quad z' = -y\cdot sen(\alpha(x/L)) + z\cdot\cos(\alpha(x/L)) \quad (2)$$

2.3. Shaders de transformación compuesta total y corregida.

A la hora de representar un entorno virtual para simuladores de conducción terrestre, las normativas constructivas del trazado exigen la combinación de las transformaciones simples vistas. Por otro lado, la elección del número y tipo de transformaciones que se han de aplicar está intrínsicamente ligado a aspectos perceptivos [17][18][19]. En el caso ferroviario y viario lo más eficiente resulta emplear un curvado en planta, que genere la familia de módulos, junto a un curvado en alzado, que suavice los módulos en casos de variación

brusca de pendientes. El peralte se sustituye por otros efectos visuales más sencillos de implementar.

Un aspecto a considerar es que cuando se combinan deformaciones estas no son conmutativas, el orden de aplicación conduce a resultados distintos. Lo que se aplica es una deformación, no siempre descomponible en combinaciones de las simples, en la que al inicio y final del módulo se establecen las condiciones angulares de forma que la sección transversal del módulo coincida plenamente con la de los módulos colindantes.

En la Figura 9 **Secuencia de varios módulos.** se muestra una secuencia de cuatro módulos que siguen una curva que parte de pendiente y peralte nulos y estos van creciendo. Las transiciones entre módulos son imperceptibles, sin holguras y adicionalmente los módulos se diseñan con unas pequeñas solapas que corrigen los pequeños defectos geométricos debido a las aproximaciones o a la poligonalización intrínseca de un entorno virtual.

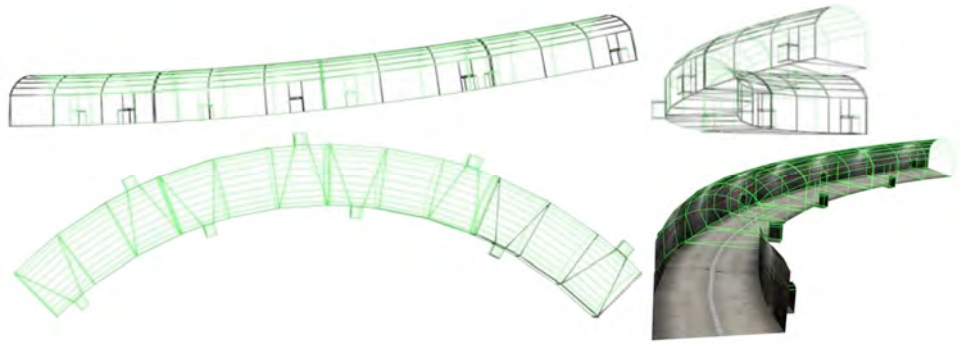

Figura 9 Secuencia de varios módulos.

3. APLICACIONES.

Todos los proyectos de entorno virtual desarrollados en Citef tienen una componente modular en mayor o menor grado, lo que facilita el desarrollo y la automatización. En las figuras Figura 10 **Túnel L3 Metro de Madrid.**aFigura **13 Paisaje tranviario.**se tienen distintas partes de escenarios que han sido generadas modularmente y en los que se aplica o puede aplicar shaders geométricos. Estos escenarios corresponden a los entornos virtuales de los simuladores de conducción que Citef [20] desarrolla para empresas como Metro de Madrid, CAF, Alstom o Invensys.

Figura 10 Túnel L3 Metro de Madrid.

Figura 11 Paisaje de calzada interurbana.

Figura 12 Paisaje genérico modular.

Figura 13 Paisaje tranviario.

4. CONCLUSIONES.

Que la tecnología modular tiene ventajas a la hora de la generación de entornos nos parece probado a la vista de las aplicaciones realizadas. Con la aplicación de los shaders las posibilidades de la modularidad crecen y con el incremento de la capacidad del hardware la calidad de los entornos virtuales crecerá de forma espectacular en un futuro próximo.

Los siguientes desarrollos van a ir por la aplicación de shaders geométricos que creen la geometría modular completa en tiempo de ejecución de forma que el entorno es introducido de forma paramétrica en base a perfiles y trayectorias y sólo se carga la geometría más singular y menos modular. De esta forma la generación en tiempo real sería adaptativa, a mayor capacidad de cálculo geométrico mayor complejidad y calidad del entorno.

5. REFERENCIAS

[1] Pharr, M., Fernando, R. GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation.Addison-Wesley.2005. ISBN-10: 0-321-33559-7.

[2] Pomeranz, A. Roam using surface triangle clusters (RUSTIC). Master's thesis, University of California at Davis, 2000.

[3] Levenberg, J. Fast view-dependent level-of-detail rendering using cached geometry.In Proceedings IEEE Visualization '02, pp 259–266. IEEE, Oct 2002.

[4] Cignoni, P., Ganovelli, F., Gobbetti, E.. BDAM – batched dynamic adaptive meshes for high performance terrain visualization. Computer Graphics Forum 22, 3.2003.

[5] Gobbetti,E., Marton,F., Cignoni,P.. C-BDAM - Compressed Batched Dynamic Adaptive Meshes for Terrain Rendering. Computer Graphics Forum, 25(3), September 2006. Proc. Eurographics 2006.

[6] Livny,Y., Kogan,Z., El-Sana,J. Seamless Patches for GPU-Based Terrain Rendering. WSCG 2007.

[7] Pajarola,R. Gobbetti, E. Survey on Semi-Regular Multiresolution Models for Interactive Terrain Rendering. The Visual Computer, 23(8).pp 583-605, 2007.

[8] Chew, L. P. Constrained Delaunay triangulations. In Proceedings of the Third Annual Symposium on Computational Geometry (Waterloo, Ontario, Canada, June 08 - 10, 1987). D. Soule, Ed. SCG '87. ACM Press, New York, NY, 215-222. 1987.

[9] Papelis, Y., Ahmad, O., Watson, G. Scenario Definition and Control for the National Advanced Driving Simulator. In Proceedings of the Driving Simulation Conference North America, Dearborn, Michigan. 2003.

[10] Wu, Q., Oza, A., and Mourant, R. R. Pedestrian Scenario Design and Performance Assessment in Driving Simulations. In Proceedings of the Driving Simulation Conference North America, Orlando, Florida. 2005.

[11]Suresh, P. and Mourant , R. R., A Tile Manager for Deploying Scenarios in Virtual Driving Environments. Proceedings of the Driving Simulation Conference North America, December, 2005.
[12]Govil, Vaibhav and Mourant, Ronald R. A Tile/Scenario Algorithm for Real-Time 3D Environments. Proceedings the 32nd International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 2005), Los Angeles, California.August 2005.
[13]C.TOVAR; J.M. CABANELLAS, , J. FÉLEZ *"Procedimientos geométricos para el ajuste de trayectorias ferroviarias en simuladores 3d de diseño modular"* Proceedings of ADM-INGEGRAF'2003 Junio 2003 Nápoles Italia
[14]John Kessenich, Dave Baldwin, Randi Rost *OpenGL Shading Language* [en linea]. Document Revision: 8 07-Sept-2006 Disponible en WWW http://www.opengl.org/documentation/glsl/
[15]ENGEL, Wolfgang. Shader X5 *Advanced rendering techniques*. 1ª Edición. Boston. (Charles River Media 2006) 692 Págs. ISBN 1-58450-499-4.
[16]DICKHEISER, Michael [Editor]. *Game programming 6*.1ª Edición. Boston. (Charles River Media 2006) 692 Págs. ISBN 1-58450-450-1.
[17] Yee, H., Pattanaik, S., Greenberg, D. Spatiotemporal sensitivity and Visual Attention for efficient rendering of dynamic Environments. In ACM Transactions on Computer Graphics, 2001, 20 (1), pp. 39–65.
[18]Cater, K., Chalmers, A., and Ward, G.. Detail to Attention: Exploiting Visual Tasks for Selective Rendering. In Proceedings of the Eurographics Symposium on Rendering, 270–280. 2003.
[19]Cater, K., Chalmers, A., Ledda, P. Selective quality rendering by exploiting human inattentional blindness: Looking but not seeing. In Symposium on Virtual Reality Software and Technology 2002, pp 1724. ACM, November 2002.
[20] Nortes, A., Zoido, C.,Maroto, J.,Cabanellas, J.M. Implementación de efectos medioambientales meidante GPUs. Proceedings of ADM-INGEGRAF'2006 Junio 2006. Sitges. España.