Estado del Arte: Computación Gráfica

Mateo Torres

Universidad Católica "Nuestra Señora de la Asunción"

Resumen Este documento es un estudio sobre el estado del arte de la computación gráfica. Se centra principalmente en las técnicas de renderizado en tiempo real realizando una comparación entre las mismas. Además describe el panorama que se tiene con respecto al futuro de la computación gráfica, sus limitaciones, los problemas que hay que superar y el enfoque que se esta utilizando para superarlos.

1. Introducción

El término "Computer Graphics" se ha usado en un sentido amplio para describir "Casi cualquier cosa en las computadoras que no sea texto o sonido" [1]. Típicamente, el término se refiere a diferentes conceptos:

- La representación y manipulación de datos de imagen hecho por computadoras.
- Las diferentes tecnologías utilizadas para crear y manipular imagenes.
- El sub-campo de la Ciencia de la Computación que estudia los métodos para sintetizar y manipular digitalmente contenido visual.

El desarrollo en Computer Graphics ha mutado a través de los años y esta jugando un papel muy importante inclusive en otras áreas como en la ingeniería industrial, ingeniería de productos, aeronáutica, automovilística, entre otros[2]. Hoy en día prácticamente en cada rubro se utilizan de alguna u otra forma Computer Graphics para resolver problemas[3].

2. Conceptos y técnicas básicas en Computer Graphics

Si bien existen miles de algoritmos y técnicas, hay algunas que siguen vigentes hasta hoy en día, en ésta sección se listan las que

siguen siendo objeto de investigación por parte de los especialistas en Computer Graphics en la última década.

2.1. Antialiasing

En el área del procesamiento digital de señales en general, se le llama antialiasing a los procesos que permiten minimizar el aliasing cuando se desea representar una señal de alta resolución en un sustrato de más baja resolución [4]. El aliasing es el efecto que causa que señales conínuas distintas se tornen indistinguibles cuando se muestrean digitalmente. En Computer Graphics se refiere al artefacto que hace que una pantalla ciertas curvas y líneas inclionadas presenten un efecto visual tipo sierra o escalón. Se produce por la dificultad de los gráficos que utilizan el método raster, que debe representar dichas curvas utilizando pequeños cuadrados (pixels) [5].

Esta técnica se utiliza en cualquier forma de representación gráfica de tipo raster, ya sean juegos, aplicaciones de escritorio, películas, efectos especiales, etc. Es útil porque ayuda a suavizar los elementos dentro de la representación discreta de los píxeles de las pantallas.

Las técnicas más recientes de antialiasing llevan características que involucran inteligencia artificial, como reconocimiento de patrones y de reflexiones de transparencia y texturas.

2.2. Shader

La tecnología shaders es cualquier unidad escrita en un lenguaje de sombreado que se puede compilar independientemente. Es una tecnología reciente y que ha experimentado una gran evolución destinada a proporcionar al programador una interacción con la GPU hasta ahora imposible. Los shaders son utilizados para realizar transformaciones y crear efectos especiales, como por ejemplo iluminación, fuego o niebla. Para su programación los shaders utilizan lenguajes específicos de alto nivel que permitan la independencia del hardware [6].

El shading es otra técnica que se usa en casi todos los ámbitos, las técnicas de shading en tiempo real se mejoran en eficiencia, y en non real-time se mejoran para lograr efectos mejores.

2.3. Shading

Se refiere al proceso de alterar el color de un objeto, superficie o polígono en una escena 3D, basada en el ángulo con respecto a las luces y la distancia del objeto a la misma, para crear un efecto fotorealista [7].

2.4. Texture Mapping

Es un método para añadir detalle, textura de superficia, o color a una imágen generada por computadora o un modelo en tres dimensiones.

Un mapa de textura es aplicado a la superficie del polígono. Este proceso es similar a aplicar papel a una caja blanca. Cada vértice del polígono es asignado a una coordenada de la textura. Luego muestras de la textura son interpoladas a través de las caras del poplígono para producir el efecto visual que parece tener más riqueza que la que podría lograrse con un número limitado de polígonos[8].

2.5. Bump Mapping

Es una técnica para simular bultos, hendiduras e imprefecciones en la superficie de un objeto. Se logra perturbando las normales de la superficie sin cambiar su geometría. Se utiliza un mapa de profundidades que se utiliza para la perturbación del vector normal (que seguirá siendo perpendicular a la superficie en la geometría del objeto) en el momento de calcular el efecto de la luz sobre el objeto.

Las limitaciones del bump mapping se hace evidente por lo general a la hora de generar las sombras, ya que para involucrar el trazo de las imperfecciones al cálculo de la sombra no es factible la perturbación de las normales de la frontera o borde del objeto con respecto al foco lumínico[9].

2.6. Mipmapping

Es una técnica de aceleración del renderizado que se basa en texturas prerenderizadas. Se cuenta con una textura compuesta por varias copias de la misma textura en distintos tamaños y que están prerenderizados. Cuando el objeto a representar se encuentra a una

profundidad suficiente, se utiliza una versión de menor calidad de la imágen, que aminorará las imperfecciones del renderizado, ocupando más memoria pero liberando carga de cómputo[10].

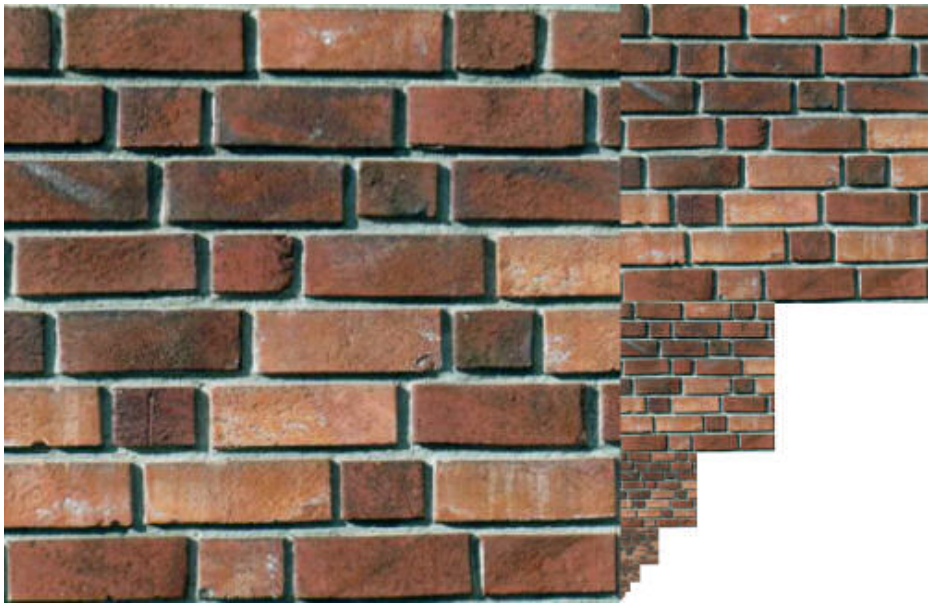

Figura 1. Ejemplo de una textura mipmap.

2.7. z-Buffering

Es el manejo de las coordenadas de profundidad de las imágenes en los gráficos en tres dimensiones, normalmente se calculan por hardware, y a veces por software[11].

Es una de las soluciones al problema de decidir qué elementos de una escende renderizada son visibles y cuales están ocultos.

2.8. Radiosity

Es un conjunto de técnicas para el cálculo de la iluminación global en una escena tridimensional. El transporte de la luz sólo se puede modelar de forma óptima considerando que cada fuente emite un número enorme de fotones, que rebotan al chocar contra una

superficie describiendo una cantidad de trayectorias imposibles de simular con el computador[12].

Se utiliza principalmente para lograr crear el efecto de rebote opaco realista en la iluminación de una escena 3D. En la figura 2 podemos observar la diferencia entre una escena renderizada utilizando la técnica y otra que no la utiliza.

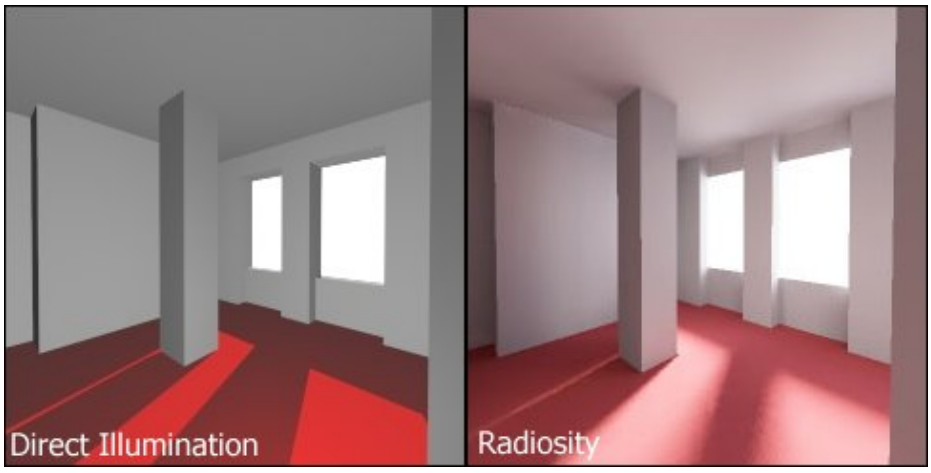

Figura 2. Comparación de la misma esceba sin y con radiosity.

2.9. Renderización

Renderizado (render en inglés) es un término usado en jerga informática para referirse al proceso de generar una imagen desde un modelo. Este término técnico es utilizado por los animadores o productores audiovisuales y en programas de diseño en 3D.

La renderización se aplica en la computación gráfica, más comúnmente a la infografía. En infografía este proceso se desarrolla con el fin de imitar un espacio 3D formado por estructuras poligonales, comportamiento de luces, texturas, materiales (agua, madera, metal, plástico, tela, etcétera) y animación, simulando ambientes y estructuras físicas verosímiles. Una de las partes más importantes de los programas dedicados a la infografía son los motores de renderizado, los cuales son capaces de realizar técnicas complejas como radiosidad, raytra-

ce (trazador de rayos), canal alfa, reflexión, refracción o iluminación global [13].

2.10. Ray Tracing

Es un algoritmo para síntesis de imágenes tridimensionales.

Se determinan las superficies visibles en la escena que se quiere sintetizar trazando rayos desde el observador (cámara) hasta la escena a través del plano de la imagen. Se calculan las intersecciones del rayo con los diferentes objetos de la escena y aquella intersección que esté más cerca del observador determina cuál es el objeto visible.

El algoritmo de trazado de rayos extiende la idea de trazar los rayos para determinar las superficies visibles con un proceso de sombreado (cálculo de la intensidad del píxel) que tiene en cuenta efectos globales de iluminación como pueden ser reflexiones, refracciones o sombras arrojadas.[14].

2.11. Ray Casting

Técnica que comunmente se confunde con el Ray Tracing, pero difiere en el problema que tratan. Mientras el Ray Tracing trata de sintetizar el trazado del rayo para determinar los objetos visibles, lo hace normalmente de manera recursiva trazando rayos secundarios, mientras el Ray Casting no lo hace, lo cual lo vuelve más rápido [15].

Es preferible utilizar el Ray Casting cuando se cuentan con algoritmos eficientes o capacidad de software suficiente para utilizar técnicas que den mejores resultados que el Ray Tracing para calcular los efectos de la luz sobre los objetos.

3. Entornos de Programacióm Gráfica

3.1. Direct 3D

Es parte del API de DirectX. Es un software licenciado por Microsoft, que es ampliamente utilizado en sus consolas Xbox y Xbox 360, así como en sistemas que requieren alto performance en gráficos en PC (comunmente juegos 3D) [16].

Usos de Direct 3D Principalmente se usa en juegos para plataformas de Microsoft (sean las consolas o PC), destacandose principalmente en sus últimas versiones por ser un buen API para manejar partículas con fluidez y facilidad.

Se usa además en Windows para aplicaciones de escritorio y aplicaciones Metro (Windows 8).

Es en la actualidad el API más utilizado para juegos High-End en PC[16].

3.2. OpenGL

Es un API abierto y multiplataforma que provee de funciones para el dibujado de gráficos 2D y 3D.

Usos de OpenGL Se utiliza también para juegos, pero principalmente es utilizado en programas profesionales.

Es el más extendido en su uso para aplicaciones profesionales[17].

3.3. Direct3D vs. OpenGL

Si bien sólo se puede comparar plenamente hablando en términos de pc, podemos comparar algunas características independientes.

- **Portabilidad:** Al ser multiplataforma, OpenGL es más portable que Direct3D, aunque existen proyextos como Wine que tratan de portar Direct3D a sistemas Unix haciendo ingeniería inversa.
- **Facilidad de Uso:** Otra característica que favorece históricamente a OpenGL, implementado en C y siguiendo el modelo de una m'aquina de estados finitos. En contrapartida Direct3D en sus primeras versiones era complejo de utilizar, lo que llevo a

múltiples quejas por parte de los programadores, lo cual hizo evolucionar a Direct3D, que desde su version 9 se ha considerado como el API 3D más sencillo de utilizar [16].

- **Performance:** Se ha discutido mucho sobre el tema de la performance en la "guerra de APIs", pero la conclusión más objetiva lleva a considerar que los dos APIs estan pensados conceptualmente para distintos trabajos, en aplicaciones profesionales, OpenGL lleva la ventaja, mientras que Direct3D lo hace en juegos interactivos en tres dimensiones.

3.4. WebGL

No se puede dejar de mencionar a este nuevo participante en el mundo de las APIs para Computer Graphics, se trata de WebGL, una propuesta de WebGL Working Group para llevar el 3D a lo que se cree la nueva plataforma de cómputo, la web.

Implementada en JavaScript, es una versión de OpenGL. Se muestra en los navegadores en conjunto con el estandard que se encuentra en creaci'on actualmente: HTML5, en particular el tag *canvas* a través como una interfaz del DOM (*Document Object Model*).

WebGL Working Group está compuesto por los mayores vendedores de navegadores:

- Apple (*Safari*)
- Google (*Chrome*)
- Mozilla (*Firefox*)
- Opera (*Opera*)

4. Real-Time vs. Non Real-Time Rendering

Un problema crucial a la hora de hacer representaciones gráficas es la calidad de la imagen producida, y el problema no tiene una solución única pues depende crucialmente de la funcionalidad requerida para la representación.

Cuando hablamos de aplicaciones interactivas como juegos, CAD, aplicaciones de modelado en tres dimensiones y realidad aumentada, estamos hablando de render en tiempo real. Es decir, que el renderizado debe terminar en un período determinado de tiempo, mientras

más interactiva la aplicación, es más corto el tiempo, que se mide en FPS (*frames por segundo / cuadros por segundo*). La velocidad es crucial en un juego, para evitar que el usuario pierda la sensación de fluidez en el movimiento de los personajes.

Si en cambio disponemos de un tiempo relativamente ilimitado para lograr el render, por ejemplo en aplicaciones de creación de efectos cinematográficos o renderizados finales de modelos aarquitectónicos, las prioridades son otras y la calidad del render aumenta. La prioridad en este caso en la mayoría de los casos se centra en lograr imágenes realistas, en las que se puedan apreciar con mucha calidad los detalles de los objetos.

4.1. Real-Time

El problema principal es el trade-off entre capacidad de cómputo y calidad de la imagen generada[18]. Se busca con frecuencia aumentar la eficiencia de los algoritmos de renderizado, para que éstos puedan admitir técnicas como el antialiasing, raytracing, etc. de manera a generar con los mismos recursos imágenes de menor calidad sin perder muchos FPS.

En aplicaciones interactivas la cantidad de FPS deseada se encuentra cerca de 60, aunque hay aplicaciones que requieren menos (normalmente 30). También existen requisitos mayores para los juegos que soportan imágenes en tres dimensiones, donde el hardware debe poder producir una tasa de refresco de 120Hz (que es la carga de producir dos sets de imágenes en paralelo a 60 FPS)[19].

4.2. Non Real-Time

Si bien, luego de ver los problemas que acarrea el real-time rendering, es facil suponer que si contamos con "todo el tiempo del mundo" no tendremos problemas. Pero eso está muy alejado de la realidad, el primer motivo de esta equivocación es que en realidad no contamos con tiempo infinito, y el segundo motivo es que lograr una imagen de muy alta calidad involucra otro tipo de problemas no relacionados necesariamente con el tiempo.

En el mundo del non real-time rendering, se necesitan realizar cálculos de muy alta complejidad. Es por eso que surgen soluciones que hacen que un problema de render se vuelva tratable

computacionalmente[20]. Es por eso que los esfuerzos en el campo del non real-time rendering se enfocan principalmente en la paralelización y dsitribución de la carga computacional del renderizado[21].

5. Computer Graphics en la actualidad

Los gráficos por computadora no son un concepto nuevo, desde hace ya varios años es una herramienta consolidada para gran variedad de industrias. Puede ser categorizada de distintas maneras y hay múltiples técnicas para lograr los resultados deseados.

5.1. Photon mapping

El trazado fotónico es el proceso de emitir fotones desde las fuentes de luz y trazarlos a través de la escena. Esta será la técnica usada para construir el mapa fotónico. Vamos a ver cómo los fotones son generados en las fuentes de luz y cómo podemos seguir su recorrido en la escena de forma eficiente. Siendo esto la base para construir un buen mapa de fotones[22].

La técnica se desarrolla debido a la necesidad de un algoritmo que cuente con las ventajas de los algoritmos típicos de raytracing, pero que sea más eficiente y que se vea menos afectado por el ruido[23]. Los resultados de la nueva técnica se implementan con varios algoritmos, algunos más eficientes quu otros, y con diferentes niveles de realismo en el renderizado final.

Por sobre todo se utiliza el photon mapping para crear renders realistas en escenas que involucran una gran cantidad de objetos con diferentes niveles de transparencia que pueden afectar la luz reflejándola, refractándola y en donde se deben tener en cuenta aspectos como la porosidad del objeto que se está representando[22]. En la figura 3 podemos ver una escena renderizada utilizando photon mapping.

5.2. Voxel Rendering

Un *voxel* (volumetric pixel) es un elemento de volumen, representa un valor en una grilla regular en un espacio de tres dimensiones. Es enálogo a un pixel, que representa un valor en una imagen 2D en

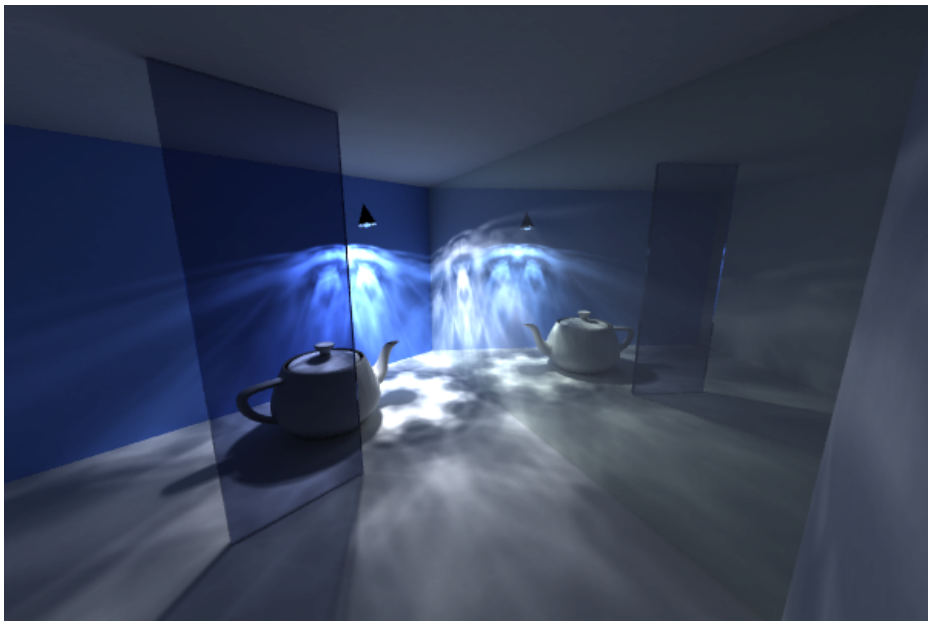

Figura 3. Ejemplo de una escena con transparencias y reflejos renderizada con photon mapping.

un mapa de bits (pixmap). [24] Como ocurre con los pixeles en dos dimensiones, los voxels no cuentan típicamente con la codificación de su posición, sin embargo, la posición de un voxel se encuentra dada basada en su posición relativa a otros voxels. En contraste, los puntos y polígonos típicamente requieren de la codificación de las coordenadas de sus vertices. Una consecuencia directa de esta diferencia es que los polígonos representan eficientemente estructuras simples en 3D con muchos espacios vacíos o rellenos de forma homogenea, mientras que los voxels son buenos representando espacios con muestreos regulares que no estan rellenos de forma homogenea.

Son frecuentemente utilizados en la visualización y analisis de datos médicos y científicos [25] además de su uso en engines de juegos.

Dada la cualidad de un voxel de poder ser modelada en una malla semi-compleja, esto da lugar a herramientas muy útiles para los seres humanos, que en la última década se han vuelto indispensables, como las imágenes médicas en tres dimensiones[26]. La capacidad de simular partículas con formas particulares además ha permitido crear imágenes detalladas de diseños que antes no podrían haber

sido renderizados con la misma eficiencia, ya que el cálculo de los polígonos sería una carga brutal para la memoria[27].

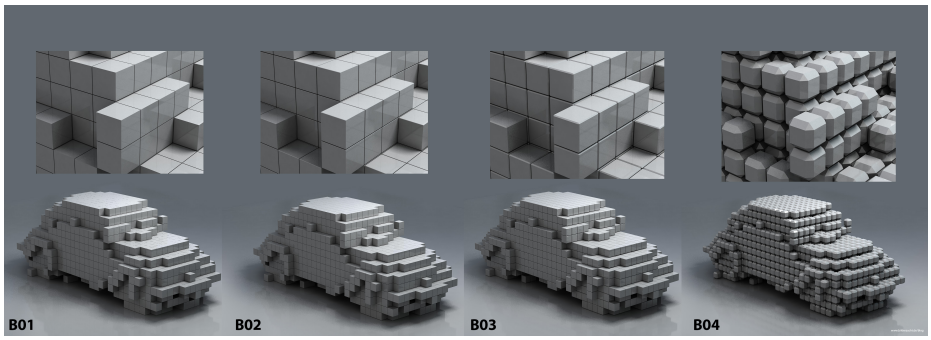

Figura 4. Ejemplo esquemático de render por voxels.

5.3. Point Cloud

la "Nube de Puntos" o *Point Cloud* es un set de vértices en un sistema de coordenadas tridimensional. Son creadas típicamente por escaneos en tres dimensiones, realizados por escáneres que miden en forma automática una gran cantidad de puntos en la superficie de un objeto [28].

Son utilizados típicamente para crear modelos CAD para manufacturación de piezas industriales, control de calidad de productos, visualizaciones en tres dimensiones, animaciones, renders y aplicaciones de *mass customization*.

La figura 5 muestra le representación de la iglesia "Il Duomo", ubicada en Milán, realizada en una nube de puntos que está siendo utilizada para proyectar los trabajos de restauraciones de la iglesia a terminarse en el 2015. El nivel de detalle del escaneo y la precisión que permite la resolución son la ventaja de esta técnica. Se consigue mapear objetos de grandes dimensiones con un nivel de detalle arbitrario.

El problema principal de ésta técnica es la cantidad de memoria requerida para mantener la información de la geometría espacial del objeto, por lo que su utilización se realiza típicamente utilizando softwares de conversión de Point Cloud Data a Mallas Ploigonales

(*Polygon Meshes*) que son fácilmente manejables por los softwares más extendidos en el mercado como Maya, Blender, 3ds Max, y otros.

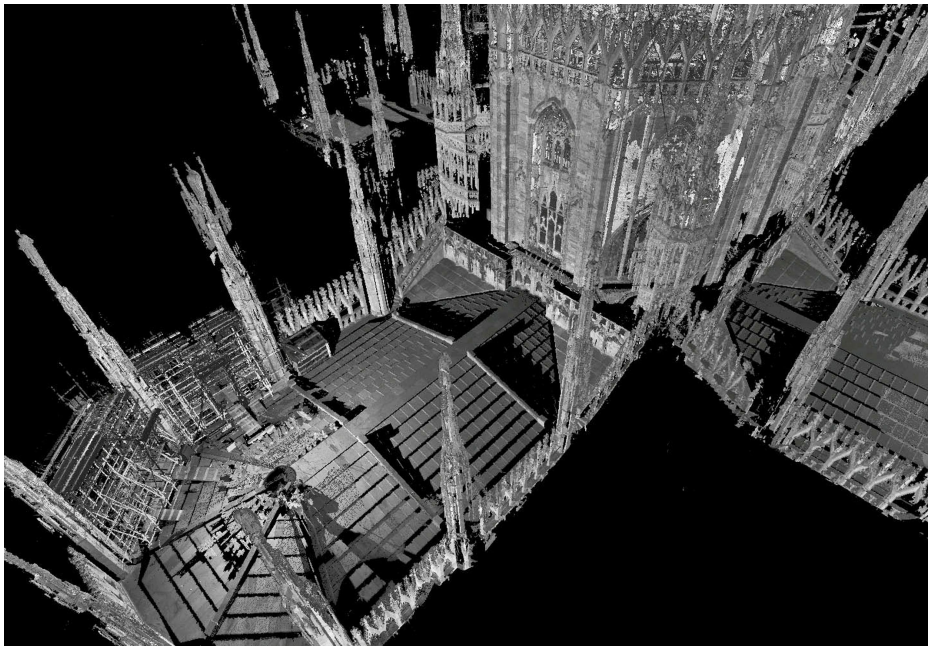

Figura 5. Iglesia "Il Duomo" (Milán) escaneada y retocada con técnicas Point Cloud (muestra a baja resolución).

5.4. Separable Subsurface Scattering

Teécnica de shading presentada en febrero del 2012 por el investigador en Real-Time Rendering Jorge Gimenez [29]. Se trata de una nueva técnica de renderizado de texturas heterogeneas con gran cantidad de imperfecciones.

Es utilizada por Activision Blizzard en el desarrollo de juegos *Next Gen*.

El proceso parte de un escaneo en resolución arbitraria (basado en Point Cloud), que luego se renderiza transformando la información en un conjunto de capas de texturas. Aplicando shaders y técnicas de antialiasing se logra el nivel de detalle de la figura 6.

Figura 6. Captura de una cabeza en movimiento utilizando la técnica de Separable Subsurface Scattering

5.5. Tessellation

El concepto de Tessellation (*Teselado*) es un método para partir polígonos en piezas más pequeñas [30]. Por ejemplo, si se toma un cuadrado y se corta por la diagonal, se ha "teselado" este cuadrado en dos triángulos. Por si mismo, el teselado hace poco para aumentar el realismo. Es decir, en una imagen no importa si un cuadrado está representado geométricamente como un cuadrado, dos triángulos o doscientos triángulos. El realismo extra se logra utilizando dichos triángulos para mostrar más información.

La forma más popular de dar uso al teselado es una técnica llamada Displacement Mapping (*Mapeo por Desplazamiento*) que consiste en una técnica que tiene como objetivo causar un efecto donde la posición geométrica actual de los puntos sobre la superficie texturizada son desplazados (a menudo a los largo de la normal de una superficie local) de acuerdo al valor que calcula la función de texturizado para cada punto de la superficie [31]. Es decir, se crea un mapa de alturas para cada sub-polígono del teselado.

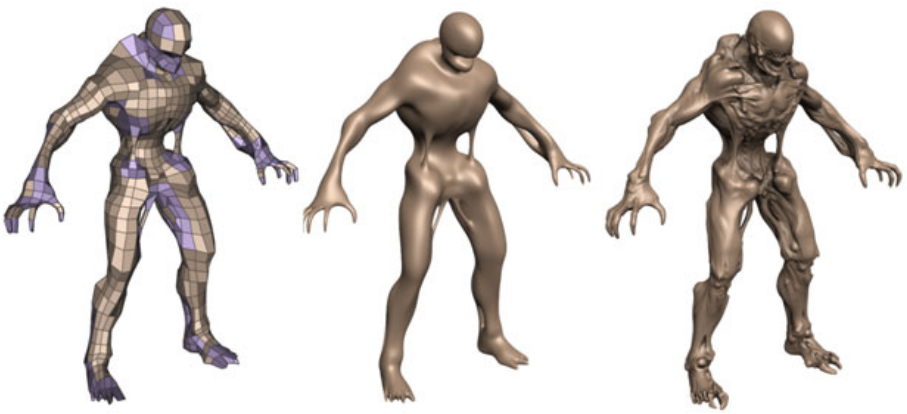

Figura 7. Luego de que una malla gruesa (izq.) pasa a través del teselado, se obtiene una malla suave (medio). Cuando se aplica Displacement Mapping se obtienen mallas con más realismo.

6. Estado del Arte

6.1. Unlimited Detail

Euclideon [32] presenta un avance su producto llamado Unlimited Detail, el cual genera mucha polémica y sorpresa, ya que la empresa afirma que pueden representar ilimitados átomos de tipo Point Cloud renderizando la geometría en tiempo real.

La afirmación es considerada de tan altas proporciones que han surgido críticas serias a la empresa, que se vio obligada a presentar una demostración en tiempo real del producto sin terminar, lo cual hicieron con un modelo de mas de 21,000,000,000,000 de átomos sin utilizar la GPU en una notebook gamer en el 2011.

Aún después de la demostración en tiempo real, las críticas continúan. Cuestionando por sobre todo la solución teórica de la limitación de la memoria ante las dimensiones de la descripción de la geometría. Hasta ahora, los engines basados en Point Cloud Data han realizado técnicas de reutilización de descripciones geométricas con el objetivo de ahorrar memoria, llegando a cantidades que son órdenes de magnitud menores a la demostración de Unlimited Detail. En la figura 8 se muestra una captura de pantalla de la presentación, que es una isla que de la forma del logotipo de la empresa con detalles hasta en los granos de arena.

Figura 8. Captura de pantalla de la demostración de Unlimited detail, la estatua cuenta con más información geométrica que varias escenas de un juego actual.

6.2. SMAA: Enhanced Subpixel Morphological Antialiasing

Técnica nueva de antialiasing en post-procesado. Ofrece soluciones prácticas a los problemas de los algoritmos comunes de antialiasing presentado por Crytek [33].

Algunos de los avances propios de éta técnica:

- Análisis de contraste local para mejor detección de bordes.
- Optimizaciones en líneas simples diagonales.
- Clasificación de patrones de precición acelerado.
- Reconstrucción de siluetas.

Es la primera técnica que combina el antialiasing morfológico (MLAA) con técnicas de multi y supersampling antialiasing (MSAA, SSAA). Lo que lleva a una técnica que obtiene la calidad visual del supersampling en una ejecución extremadamente veloz.

Figura 9. Captura de Un modelado hecho en Cry Engine Utilizando SMAA

7. Conclusión

Los avances en Computer Graphics en este momento estan generando sin duda alguna un cambio interesante en la historia de la representación gráfica. Las nuevas técnicas de rendering, shading y antialiasing hacen que los efectos sean cada vez más realistas, las animaciones más genuinas y la experiencia del usuario más completa.

En muchas industrias esto implica que el negocio va a crecer en muchos sentidos. El cine contará con herramientas para hacer mejores efectos especiales de manera más eficiente y económica. El mundo de los juegos y del renderizado en tiempo real va a dar un salto que hará que el level design y el diseño y animación de los personajes sea comparable con filmaciones tomadas en la vida real. En los juegos y películas, además se encuentran puntos intermedios debido a la calidad del render en tiempo real, lo que crea una experiencia mixta entre un juego y una película, las reacciones ante este tipo de interactividad ha sido muy positiva. Beneficiará también al mundo de la construcción, ya que los modelos de estructuras a resolución arbitraria permitirá la simulación de fenómenos físicos como tornados y terremotos afectando a una estructura.

Personalmente creo que nos acercamos a un punto de inflexión en el mundo de los gráficos por computadora. El hecho de que las imagenes puedan ser comparables a lo que una persona ve en la vida cotidiana abrirá las puertas a muchas experiencias nuevas con las que hasta ahora solo se podían soñar. Siendo una persona que disfruta mucho de las experiencias interactivas, espero con ánsias la siguiente generación de tecnologías relacionadas al mundo de los gráficos por computadora y estoy expectante con respecto a lo que podría lograrse más adelante.

Referencias

1. of Computer Graphics., C.U.P.: What is computer graphics? (1998)
2. Sathyanarayana, K., Kumar, G.: Evolution of computer graphics and its impact on engineering product development. In: Computer Graphics, Imaging and Visualisation, 2008. CGIV '08. Fifth International Conference on. (2008) 32 –37
3. Potel, M.: A decade of applications [computer graphics]. Computer Graphics and Applications, IEEE **24**(6) (2004) 14 – 19
4. Wikipedia: (Antialiasing)
5. Olshausen, B.A.: Aliasing (2010)
6. Wikipedia: (Shader)
7. Whitted, T.: An improved illumination model for shaded display. (Graphics and Image Processing - Bell Laboratories)
8. Xu, H., Li, X.: Dynamic harmonic texture mapping using methods of fundamental solutions. In: Computer Science Education (ICCSE), 2011 6th International Conference on. (2011) 871 –875
9. Sun, J., Xu, L., Li, H., Wu, Q.: A practical bump mapping technique in scene simulation. In: Mechatronics and Automation, 2005 IEEE International Conference. Volume 1. (2005) 166 – 170 Vol. 1
10. Solomon, J., Horowitz, M.: Using texture mapping with mipmapping to render a vlsi layout. In: Design Automation Conference, 2001. Proceedings. (2001) 500 – 505
11. Chen, C.H., Lee, C.Y.: Two-level hierarchical z-buffer for 3d graphics hardware. In: Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on. Volume 2. (2002) II–253 – II–256 vol.2
12. Leblond, M., Rousselle, F., Renaud, C.: Hybridization techniques for fast radiosity solvers. In: Computer Graphics International, 2000. Proceedings. (2000) 269 –278
13. Wikipedia: (Renderización)
14. Miyazaki, D., Ikeuchi, K.: Inverse polarization raytracing: estimating surface shapes of transparent objects. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Volume 2. (2005) 910 – 917 vol. 2
15. Li, Z., Zhang, J.: Study on volume rendering of ct slices based on ray casting. In: Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on. Volume 7. (2010) 157 –160
16. Corporation, M.: (Directx graphics)
17. : (The industry's foundation for high performance graphics)
18. Zhao, L., DuanQing, X.: Real-time rendering of highly complex dynamic scenes based on parallel multi-core architectures. In: Information Management and Engineering, 2009. ICIME '09. International Conference on. (2009) 593 –597
19. Naemura, T., Tago, J., Harashima, H.: Real-time video-based modeling and rendering of 3d scenes. Computer Graphics and Applications, IEEE **22**(2) (2002) 66 –73
20. Patoli, M., Gkion, M., Al-Barakati, A., Zhang, W., Newbury, P., White, M.: An open source grid based render farm for blender 3d. In: Power Systems Conference and Exposition, 2009. PSCE '09. IEEE/PES. (2009) 1 –6
21. Ohlenburg, J., Frohlich, T., Broll, W.: Internal and external scene graphs: a new approach for flexible distributed render engines. In: Virtual Reality, 2005. Proceedings. VR 2005. IEEE. (2005) 293 –294

22. Spencer, B., Jones, M.: Hierarchical photon mapping. Visualization and Computer Graphics, IEEE Transactions on **15**(1) (2009) 49 –61
23. Steinhurst, J., Lastra, A.: Global importance sampling of glossy surfaces using the photon map. In: Interactive Ray Tracing 2006, IEEE Symposium on. (2006) 133 –138
24. Wikipedia: (Voxel)
25. of Computer Science., S.B.U.D.: (Fundamentals of voxelization)
26. Xie, H., Li, G., Ning, H., Menard, C., Coleman, C., Miller, R.: 3d voxel fusion of multi-modality medical images in a clinical treatment planning system. In: Computer-Based Medical Systems, 2004. CBMS 2004. Proceedings. 17th IEEE Symposium on. (2004) 48 – 53
27. Laine, S., Karras, T.: Efficient sparse voxel octrees. Visualization and Computer Graphics, IEEE Transactions on **17**(8) (2011) 1048 –1059
28. Wikipedia: (Point cloud)
29. Jimenez, J.: (The day has come)
30. Nvidia: (Directx 11 tessellation)
31. Wikipedia: (Displacement mapping)
32. Ecclideon: (Euclideon:geoverse)
33. Jimenez, J., Echevarria, J.I., Sousa, T., Gutierrez, D.: Smaa: Enhanced morphological antialiasing. Computer Graphics Forum (Proc. EUROGRAPHICS 2012) **31**(2) (2012)