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

Evaluación Optimizada de Arboles CSG de Sólidos Triangulados de Forma Libre

C.J. Ogayar[1], F.R. Feito[1], R.J. Segura[1] and M.L. Rivero[2]

[1]Dpto. de Informática - Universidad de Jaén - EPS Jaén - Jaén - España {cogayar,ffeito,rsegura}@ujaen.es
[2]Dpto. de Informática - Universidad de Jaén - EUP Linares - Jaén - España {mlina}@ujaen.es

Abstract

En este trabajo se presenta un algoritmo robusto para la evaluación de un árbol CSG completo de sólidos cuyas fronteras están delimitadas por mallas de triángulos. Este enfoque permite realizar la evaluación de un árbol CSG en una sola pasada, realizando las transformaciones geométricas y las operaciones booleanas asociadas a cada nodo en una sola expresión.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Constructive solid geometry (CSG),Boundary representations

1. Introducción

Las mallas de triángulos son utilizadas en multitud de aplicaciones como en CAD, Realidad Virtual y videojuegos para representar sólidos poliédricos. Este esquema de representación es un estándar en muchas áreas debido a su simplicidad. Además puede ser tratado de forma directa por el hardware gráfico programable actual. En [OFSR06] se presenta un método para la evaluación de operaciones booleanas sobre sólidos triangulados. En [HR05] se presenta un método para la visualización de operaciones booleanas entre sólidos de forma libre que incluye el tratamiento de un árbol CSG completo. En este artículo se presenta un método de evaluación de árboles de operaciones booleanas entre sólidos triangulados basado en [OFSR06] y con la misma filosofía que [HR05] aplicada a la evaluación en lugar de sólo a la visualización. El método extendido permite la evaluación de un árbol completo con optimizaciones que aportan mayor eficiencia que un enfoque clásico basado en la evaluación incremental del árbol CSG operación por operación. Además, parte del proceso de evaluación puede implementarse en GPU programable.

2. Trabajo previo

La clasificación entre dos sólidos A y B consiste en calcular tres conjuntos separados: $A-B$, $B-A$ y $A \cap B$ llamados regiones de subdivisión. Estas operaciones son indispensables para realizar operaciones booleanas con sólidos [Sha01]. Cuando se realiza la subdivisión, se selecciona la región resultado según la operación booleana utilizada. En este trabajo se utilizan sólidos triangulados existiendo varias técnicas para realizar la subdivisión con este esquema de representación. Algunos están basados en la utilización de líneas de sección [MT83] y otros en recorte de triángulos [PK89]. En este trabajo se utiliza el segundo enfoque. Para el tratamiento de varias operaciones booleanas como las que se presentan en la evaluación de un árbol CSG, existen varias técnicas. Para la visualización cabe citar [SFR04] y [HR05] que permiten la obtención de imágenes de un árbol CSG completo. En cuanto a la evaluación, tradicionalmente se emplean métodos que resuelven las operaciones booleanas una por una hasta completar la evaluación del árbol CSG completo.

3. Evaluación booleana

En esta sección se presenta el algoritmo utilizado para realizar la evaluación de una operación booleana entre dos sólidos triangulados basada en [OFSR06]. Este método será expandido posteriormente para la evaluación de un árbol CSG completo en un solo paso. La evaluación de una operación booleana entre dos sólidos se compone de tres etapas: la intersección entre los dos objetos, la clasificación de los triángulos resultantes y la selección de dichos triángulos para cada operación booleana.

3.1. Refinamiento

El primer paso consiste en realizar un refinamiento de cada una de las mallas que forman los objetos A y B, de manera que los triángulos resultantes en cada malla sean coincidentes o la intersección de su interior sea vacía. Esto nos asegurará que los triángulos obtenidos de cada uno de los sólidos de A y B, serán susceptibles de clasificarse enteramente como in, out, on respecto de B y A, respectivamente. En otras palabras, mediante el refinamiento de la malla, cualquier punto de cada triángulo resultante sirve para realizar el test de inclusión triángulo en sólido, reducido entonces a un test de punto en sólido.

Durante el refinamiento de cada objeto, cada triángulo es intersectado con todos los triángulos del otro sólido, obteniendo dos versiones nuevas de los objetos. Las triangulaciones resultantes están adaptadas a la zona de intersección. Con cada intersección triángulo-triángulo se crean nuevos vértices en las estructuras B-Rep. Cada triángulo afectado por una intersección debe ser descompuesto. El cuello de botella de esta etapa es el test de intersección triángulo-triángulo, ya que debe ser efectuado para cada combinación de dos triángulos de cada malla. Para resolver este problema de rendimiento se utiliza un clasificador espacial que acelera las consultas sobre los triángulos de cada malla. En este trabajo se propone el octree. Con esta estructura se mantiene una jerarquía de nodos que contienen los identificadores de cada triángulo que los intersectan. Hay que resaltar que se utiliza el mismo octree para las dos mallas a la vez. Cada nodo mantiene los identificadores de los triángulos de las dos mallas que intersectan dicho nodo. Para la intersección triángulo-triángulo se propone el algoritmo de [Mol97], y para la subdivisión el algoritmo de recorte presentado en [RF04], que además resuelve eficientemente los casos de intersecciones entre triángulos coplanares.

3.2. Clasificación

Como resultado de la etapa anterior hay dos mallas refinadas, es decir, teseladas y adaptadas al perfil de intersección. Estas mallas cumplen la condición de que todos sus triángulos están completamente dentro, fuera o en la frontera de la otra malla. Lo anterior permite simplificar el test de inclusión de triángulo-en-sólido a punto-en-sólido, lo que proporciona una gran mejora en el rendimiento. El algoritmo de inclusión utilizado es el descrito en [OFSR06], que está basado en GPU. Lo importante del algoritmo de evaluación booleana propuesto es que el test punto-en-sólido consume la mayor parte de la carga computacional. Esto significa que su implementación en GPU implica que el método de evaluación se beneficia al máximo del uso del hardware gráfico.

Después de que el conjunto completo de triángulos provenientes de la fase anterior de subdivisión es clasificado respecto de los dos sólidos con estados *in*, *out*, *on*, pueden obtenerse nuevos sólidos como resultado de una operación booleana. El resultado serán nuevas representaciones B-Rep que surgen de seleccionar los triángulos que cumplan las condiciones de la operación booleana [SIAGC06]. Esta clasificación permite calcular tanto operaciones regularizadas como no regularizadas [RR99].

Cuando se calcula la intersección entre los dos sólidos involucrados en una evaluación booleana, hay triángulos de cada malla que no se dividen porque están totalmente dentro, fuera o en la frontera de la otra malla desde el principio. Estos triángulos están localizados en grupos que representan una parte de la frontera del sólido al que pertenecen. Si se calcula la información de conectividad entre los triángulos de la malla, pueden construirse los grupos de triángulos que están separados por las zonas de intersección y que cumplen la propiedad de estar totalmente dentro o fuera del otro sólido, ya que están compuestos de triángulos conectados que cumplen esta misma propiedad. De esta forma, cada grupo de triángulos conectados puede clasificarse respecto del otro sólido utilizando un sólo punto perteneciente a la superficie a la que representan, esto es, tomando cualquier punto de cualquiera de los triángulos del grupo. Ya que el test punto-en-sólido es el proceso que más tiempo de cálculo demanda de todo el algoritmo, esta optimización es fundamental para obtener un buen rendimiento, en especial con mallas de triángulos muy complejas.

3.3. Evaluación

Cuando todos los triángulos de cada malla están clasificados respecto a la otra (con estados in, out, on), ya pueden obtenerse nuevos sólidos mediante operaciones booleanas. Para ello, basta con seleccionar los triángulos que cumplan las condiciones necesarias según la operación. Consultar [OFSR06] para más detalles. El resultado final se obtiene copiando los triángulos que cumplen las condiciones de la operación booleana en la nueva malla (ver figura 1). Hay que destacar que cualquiera de las operaciones booleanas puede ser realizada de forma directa a partir de las mismas mallas refinadas. Por esta razón, el coste de realizar todas las operaciones booleanas simples que pueden aplicarse entre dos sólidos es casi el mismo que el de una sola de ellas. Esto puede ser de utilidad en algunas ocasiones, como en la evaluación optimizada de un árbol CSG.

4. Evaluación de árboles CSG

El método presentado en la sección anterior permite la evaluación de una operación booleana entre dos sólidos triangulares. Para extenderlo a la evaluación de un árbol CSG basta con recorrer dicho árbol resolviendo las operaciones booleanas una a una hasta llegar al nodo raíz cuya evaluación dará como resultado la evaluación CSG final. Sin embargo, esta concatenación de operaciones booleanas lleva a una gran cantidad de estados intermedios que no son interesantes desde el punto de vista del resultado final, esto es,

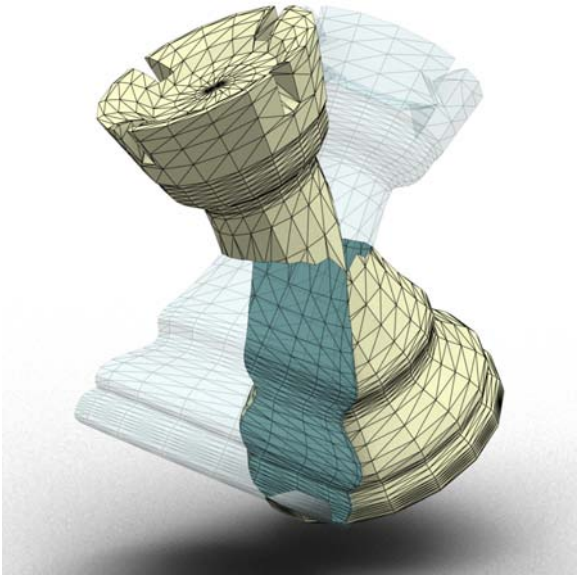

Figure 1: *Operación booleana Torre A - Torre B.*

son datos temporales. Muchos triángulos procesados, recortados y testados no estarán en el B-Rep final debido a operaciones como la intersección o la diferencia. Además, por la naturaleza del proceso, debe crearse una gran cantidad de estructuras de datos intermedias como los optimizadores espaciales para las intersecciones. El método extendido presentado en este trabajo se centra en las simplificaciones y optimizaciones que pueden realizarse en la evaluación de una expresión booleana como la que presenta un árbol CSG.

4.1. Conversión del árbol CSG

Hay una serie de inconvenientes que se presentan a la hora de evaluar el árbol CSG en un solo paso en lo que respecta al tratamiento de la topología de los sólidos. En primer lugar, cada uno de los sólidos se ve afectado por una matriz de transformación en cada uno de los nodos del árbol CSG. Además, como se verá más adelante, las operaciones booleanas de diferencia A-B y B-A complican en gran medida algunas optimizaciones espaciales que se aplican sobre las cajas englobantes de los sólidos en cada nodo del árbol.

Para simplificar el desempeño del algoritmo, las matrices de transformación deben llevarse a los nodos hoja, de forma que los triángulos puedas trocearse por posibles intersecciones entre sólidos en cualquier momento sin depender del nivel del árbol evaluado. Para ello, se realiza un cálculo de la matriz de transformación final de cada objeto que será la composición de las matrices asociadas a todos los nodos del árbol que afecten a dicho objeto. El objetivo es situar a cada objeto en la posición final en el sistema de coordenadas de referencia. Si hay resultados intermedios de operaciones booleanas en algunos nodos del árbol, la transformación geométrica de dicho nodo estará presente en la matriz de transformación final de cada objeto involucrado. Es decir, se aplica la propiedad distributiva sobre las operaciones booleanas y las transformaciones geométricas de cada objeto.

Para facilitar las optimizaciones espaciales utilizadas en este método de evaluación, y sin pérdida de generalidad, se convierte la expresión CSG a su forma normal [HR05]. Primero, se convierten todas las diferencias booleanas en intersecciones con el complemento del argumento de la derecha. Después se propaga el complemento hacia abajo en el árbol aplicando la ley de DeMorgan: $(A \bigcup B)' = (A' \bigcap B')$.

4.2. Evaluación del árbol CSG

El objetivo principal del algoritmo de evaluación es posponer la clasificación de los triángulos de todos los sólidos del CSG hasta el final, es decir, en el nodo hoja del árbol. De esta forma el algoritmo queda como sigue: (1) Como se menciona en el apartado anterior, se supone que las matrices de transformación en cada nodo hoja del CSG se han compuesto de forma que cada sólido asociado a dicho nodo incluye las transformaciones hasta el nodo raíz. Se supone también que el árbol CSG está expresado en forma normal, esto es, mediante uniones e intersecciones. (2) Se calculan las cajas envolventes de los sólidos en los nodos hoja y de las operaciones intermedias en los nodos intermedios del árbol. Se realizan las intersecciones de las fronteras entre sólidos en los lugares de intersección de cajas envolventes con ayuda de un optimizador espacial. (3) Utilizando la jerarquía de cajas envolventes, se clasifican los triángulos que intervienen en el resultado final de la expresión booleana. Esta clasificación se realiza siguiendo en orden la expresión booleana para cada objeto. A continuación se explica con más detenimiento cada paso del algoritmo.

El primer paso consiste en expresar el árbol CSG en forma normal, tal y como se muestra en el apartado anterior. Además, cada sólido de cada nodo hoja es transformado acumulando todas las transformaciones hasta el nodo raíz. Si dos instancias del mismo sólido tienen distintas matrices de transformación acumulada, serán tratadas como sólidos distintos y por tanto replicados en memoria. Esto es así ya que el proceso de recorte de triángulos debido a las operaciones booleanas genera nueva topología que será propia de cada instancia de cada sólido. En cada nodo del árbol se calcula una caja envolvente de la topología representada en ese nivel, es decir, de los triángulos de los sólidos que intervienen en la operación booleana. La creación de cajas envolventes se realiza de forma ascendente en el árbol, ya que es un proceso acumulativo, esto es, es la unión de las cajas envolventes de los niveles inferiores. Además en cada nodo se calcula la caja obtenida de realizar la operación booleana de las cajas envolventes de los nodos inferiores. Estas cajas, denominadas volúmenes de intersección, van a representar las zonas

del espacio donde se encontrarán los triángulos de los sólidos que intervienen en la operación booleana de ese nivel que son susceptibles de intersectarse. Esta optimización es útil ya que reduce en gran medida la cantidad de tests de intersección de triángulos a realizar. Ya que este método retrasa la subdivisión de la superficie de los sólidos hasta el momento de la evaluación del nodo hoja, todos los objetos participan en una intersección de superficies en el mismo nivel del árbol, lo que resulta en proceso altamente costoso sin ninguna optimización (todos con todos). Mediante el uso de volúmenes de intersección se asegura que los sólidos no relacionados en la expresión CSG original (no comparten nodos ni directa ni indirectamente) no serán intersectados.

En la etapa de intersección entre objetos se calculan las zonas de intersección entre sólidos que intervienen en una operación booleana. Este paso funciona igual que el algoritmo básico de evaluación de operaciones booleanas, con la salvedad de que se incorpora información extra en forma de volúmenes de intersección con el objetivo de seleccionar correctamente los triángulos de los objetos que intervienen en cada operación. Además de contar con la jerarquía de cajas envolventes asociadas a los nodos del árbol CSG, se utiliza un optimizador espacial para resolver eficientemente las intersecciones entre triángulos de sólidos afectados por una operación booleana. Este optimizador espacial debe tener un tiempo de construcción reducido, así como requerimientos de memoria ajustados en la medida de lo posible. Además debe permitir responder a la consulta sobre intersecciones triángulo-triángulo de forma efectiva y eficiente. En este trabajo se propone el uso de un octree por cada volumen de intersección del árbol. Esto es, para cada caja envolvente asociada a una zona de intersección entre sólidos.

Una vez que se dispone de las estructuras de datos necesarias para realizar de forma eficiente consultas sobre intersecciones entre triángulos se procede a la intersección de los sólidos y a la subdivisión de triángulos. La última etapa es la clasificación de los triángulos resultantes según la expresión booleana. Para probar la inclusión de cada triángulo en los sólidos implicados se utiliza la jerarquía de cajas envolvente además de información de conectividad entre triángulos que permite agrupar zonas delimitadas por las líneas de intersección y que pueden ser clasificadas con un solo test de intersección [OFSR06]. Una vez están clasificados todos los triángulos de todos los sólidos, basta con seleccionar el subconjunto que cumpla con la concatenación de operaciones booleanas correspondiente con la rama del árbol que afecta a cada sólido. Esta operación se resuelve de izquierda a derecha respetando la precedencia de paréntesis. La selección de triángulos se realiza de acuerdo a los mismo criterios del algoritmo básico de evaluación booleana [OFSR06].

5. Conclusiones

En este artículo se presenta un método para la evaluación de árboles CSG que permite obtener la topología del resultado final sin tener que realizar las evaluaciones de las operaciones intermedias por completo, reduciendo en gran medida el costo computacional asociado a la evaluación operación por operación del árbol completo. Parte del proceso puede implementarse en la GPU programable actual. El método es robusto y se adapta muy bien al tratamiento de sólidos B-Rep en su forma más simple, por lo que casi cualquier formato de fichero 3D puede ser utilizado como entrada sin realizar complejas conversiones de datos.

6. Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio de Educación y Ciencia, la Junta de Andalucía y la Unión Europea mediante fondos FEDER con los proyectos TIN2007-67474-C03-03, P06-TIC-01043 y PO7-TIC-02773.

References

[HR05] HABLE, J., ROSSIGNAC, J. Blister: GPU-based rendering of Boolean combinations of free-form triangulated shapes. *ACM Transactions on Graphics, Proceedings of SIGGRAPH 2005*.

[MT83] MÄNTYLÄ, M., TAMMINE, M. Localized set operations for solid modelling. *Computer & Graphics*, 17 (3), 1983.

[Mol97] MÖLLER, T. A Fast Triangle - Triangle Intersection Test. *Journal of Graphics Tools*, vol. 2, pp 25-30. A.K. Peters, 1997.

[OFSR06] OGAYAR, C. J., FEITO, F. R., SEGURA, R.J., RIVERO, M. L. GPU-based Evaluation of Boolean Operations on Triangulated Solids. *Symposium Ibero Americano de Computación Gráfica, SIACG 2006*.

[PK89] PILZ, M., KAMEL, H.A. Creation and boundary evaluation of CSG models. *Engineer Computer*, 5, pp. 105-118, 1989.

[RF04] RIVERO, M., FEITO, F.R. Refinamiento de mallas triangulares. Aplicación para el cálculo de operaciones booleanas en 3D. *CEIG 2004*, Sevilla, 2004.

[RR99] ROSSIGNAC, J.R., REQUICHA, A.R. Solid Modeling. *In J. Webster, editor, Encyclopedia of Electrical and Electronics Engineering*, Webster, John Wiley & Sons, 1999.

[SFR04] SEGURA, R., FEITO, F.R., RUIZ DE MIRAS, J. Non-evaluated manipulation of complex CSG solids. *Journal of WSCG*, Vol. 12, No. 3, 2004.

[SFRTO*05] SEGURA, R., FEITO, F.R., RUIZ DE MIRAS, J., TORRES, J.C., OGAYAR, C. J. An Efficient Point Classification Algorithm for Triangle Meshes. *Journal of Graphics Tools*, 10(3), pp. 27-35, 2005.

[Sha01] SHAPIRO, V. Solid Modeling *in Handbook of Computer Aided Geometric Design, G. Farin, J. Hoschek, M.S. Kim*, Elsevier Science, 2001.