KONVEKS VE KONKAV CİSİMLERİN GÖRÜNTÜLENMESİ İÇİN KIRPMA ALGORİTMASI

Sema KOÇ, Ulus ÇEVİK
Elektrik ve Elektronik Mühendisliği Bölümü
Gaziantep Üniversitesi
27310 Gaziantep
E-mail : skoc@gantep.edu.tr
ulus@gantep.edu.tr

ABSTRACT

Constructive Solid Geometry is a powerful tool used in many computer graphics applications such as CAD, CAM, and flight simulators. In CSG, complex objects are constructed from simpler objects, CSG primitives, by usually applying union and difference operations in terms of Boolean representations.
In this paper, an efficient clipping algorithm, for rendering convex and concave CSG images, is presented. A surface operator is introduced to simplify the clipping volume expressions to reduce the rendering burden on the graphics processors.
Anahtar kelimeler: CSG, Boolean operatör, görüntü.

1. GİRİŞ

Karşılıklı etkileşimli sistemlerde bilgisayar grafiği kullanarak görüntü oluşturulurken, çizgi tabanlı modellemelerde görüntünün doğru algılanamaması nedeniyle katı modellemelere ihtiyaç duyulmaktadır. Katı cisimlerin modellemesinde kullanılan en popüler metotlardan birisi de CSG'dir.

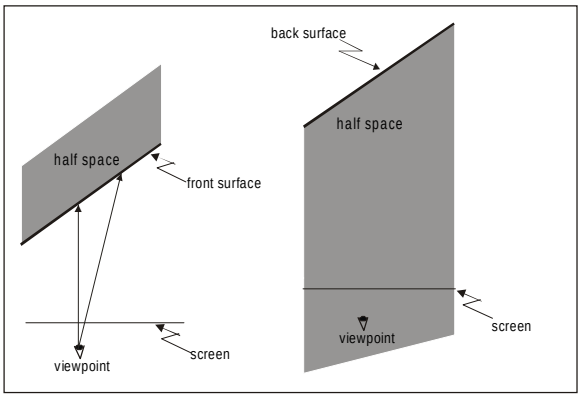
Daha önceki çalışmalarda, [1], [2], [3], [4], [5], cismin görüntüsünü doğru bir şekilde oluşturmak için tüm ön yüzeylerin arka yüzeylerden önce işlenmesi gerekmekteydi. [6]'da geliştirilen görüntü algoritması ise bu gereksinimi hem konveks hem de konkav cisimler için kaldırmaktadır. Ayrıca bu CSG modellerine, Boolean operatörlerini uygulayarak çok daha karmaşık yapıdaki görüntüleri elde etmek mümkündür.

2. GÖRÜNTÜ ALGORİTMASI

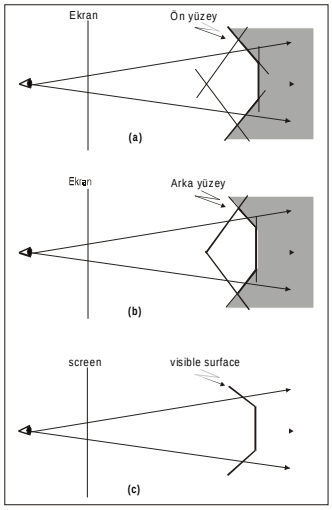
Bir cisimde ki bakış noktasından görülebilen düzlemler "ön yüzey", görünmeyenler ise "arka yüzey" olarak tanımlanır (Şekil 1). Düzlemlerin bu özelliklerini (ön, arka yüzey) kullanarak, yarı uzaylardan konveks ve konkav cisimlerin elde edilmesi mümkündür. Cismin yüzeyini oluşturan düzlemin görünen kısmı, diğer ön yüzeyler arasında bakış noktasına en uzak olan yüzey seçilerek elde edilir (Şekil 2.a). Cismin arka yüzeyi ise tüm arka yüzeyler arasında, bakış noktasına en yakın olan arka yüzeyin seçilmesi ile elde edilir (Şekil 2.b).

Son olarak, eğer cismin ön yüzeylerinin bakış noktasına olan uzaklığı arka yüzeylerinkinden küçükse, bu yüzey görünen kısım olarak tutulur, değilse arka yüzeyler tarafından kesilir (Şekil 2.c). Bu işlemler ekran üzerindeki her bir piksel üzerinde uygulanır.

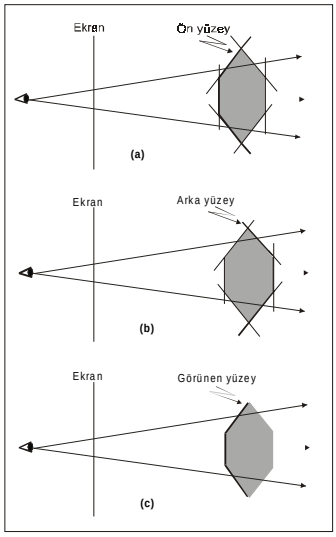
Konkav cisimleri oluştururken ise yukarıda anlatılan algoritmanın tersi uygulanır (Şekil 3.). Bu algoritmanın ayrıntılarına [6] da yer verilmiştir.



Şekil 1. Ön ve arka yüzeyler



Şekil 2. Konveks cisimlerin görüntü algoritması



Şekil 3. Konkav cisimlerin görüntü algoritması

2.1 Kırpma algoritmasının uygulanması

Daha önce de belirtildiği gibi konveks ve konkav cisimler daha karmaşık hacimleri oluşturmak için kullanılabilirler.

Bir konveks cismin diğer konveks bir cisimden çıkartılması iki şekilde gerçekleşebilir. İlk yöntem olarak hacim ifadesi aşağıdaki gibi açılabilir,

$$A.B.C.D.(\overline{E.F.G.H}) = A.B.C.D.\overline{E} + A.B.C.D.\overline{F} + A.B.C.D.\overline{G} + A.B.C.D.\overline{H}$$

Fakat bu yöntemin bazı sakıncaları bulunmaktadır. Görüntü listesindeki elemanların sayısı arttıkça, çarpma işlemlerinin sayısı büyük ölçüde artacağından, çıkarma işlemini yavaşlatacaktır.
Diğer bir yöntem ise sınırların açılımı yöntemidir. Matematiksel olarak bir cismin yüzeyini tanımlamak için bir sınır operatörü, @, kullanılabilir. @A, A objesinin yüzeyini göstermektedir. @ operatörünün iki faydalı özelliği bulunmaktadır:

$$@(A.B) = @(A).B + @(B).A,$$
$$@(A+B) = @(A).\overline{B} + (B).A.$$

Bu özellikleri yukarıdaki çarpma işlemine uygularsak;

$$A.B.C.D.(\overline{E.F.G.H}) = @(A.B.C.D).[\overline{E.F.G.H}] + @(\overline{E.F.G.H}).[A.B.C.D]$$

Bu durumda görüntü listesi düzlem sayısı yerine, ekrandaki cismin sayısına bağlı bulunmakta ve işlemin hızı düzlem sayısından bağımsız olmaktadır.

Boolean açılımının önemli bir özelliği de cismi oluşturan yüzeylerin herhangi bir sırayla işlenebilir olmasıdır. Bu sınırların açılımına dayalı algoritma kompleks ifadelerin açılımları için tekrarlanarak kullanılabilinir. Örneğin hacim V şu şekilde tanımlanmaktadır:

V=A.(B+(C.D),

A,B,C, ve D konveks veya konkav cisimler.
Sınır operatörünün yukarıda verilen iki özelliği kullanılarak hacim V'nin yüzey tanımlaması şu şekilde elde edilir.

$$@A.[B+(C.D)] = X1+X2+X3+X4$$
$$X1=(A)[B+(C.D)] \qquad X2=@(B).[A.(\overline{C}+\overline{D})]$$
$$X3=@(C)[A.\overline{B}.(D)] \qquad X4=(D).[A.\overline{B}.C]$$

Yukarıdaki ifadede @V, @(obje)[kırpan hacim)] şeklinde tanımlanmaktadır. Bu ifadelerin her birinde parantezin içerisinde ifade edilen konveks veya konkav cisimlerin yüzeyleri, köşeli parantezin içerisinde tanımlanan hacimle kesilmektedir.

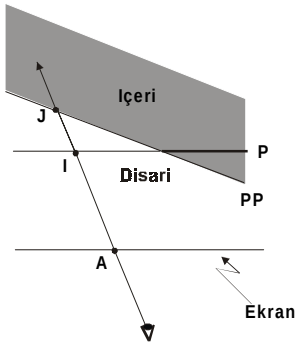
Kırpma algoritmasını oluşturmak için, kırpma ifadesi öncelikle postfix forma çevrilmelidir. Örneğin kırpma ifadesi (a+b).c.(d+e) ise postfix formda ab+c.de+., şeklinde ifade edilir. Daha sonra her bir kırpan düzlem için, her bir piksel üzerinde içinde/dışında testi uygulanır. Ve eğer bu piksel kırpan düzlem ile tanımlanan yarı uzayın dışında ise sonuç 0, içinde ise 1 olur, ve ilgili operatörle birlikte, (-, +) birleşme için, kesişme için (.), yığına itilir. En son kırpma işleminin sonucu, son kırpan düzlem işlendikten sonra yığındaki değerlerin her bir piksel için değerlendirilmesi ile elde edilir.

İçinde/dışında testi Şekil 4'de gösterilmiştir. Burada, öncelikle P düzlemi üzerindeki I noktasının kırpan düzlem PP tarafından tanımlanan yarı uzay içerisinde olup olmadığını belirlemek gerekmektedir. Sonuç kırpan düzleminin özelliğine bağlıdır. Eğer düzlem ön yüzey ise;

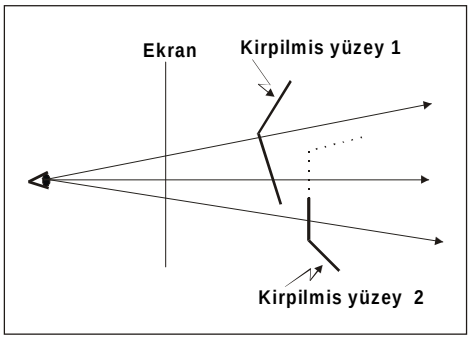
Eğer AI<AJ, I içinde
AI>AJ, I dışarıda
AI=AJ, sonuç kullanıcıya bağlı.

Eğer kırpan düzlem arka düzlem ise, ilk iki koşulun tersi alınır, sonuncu ise aynı kalır.

Görüntü algoritmasındaki son basamak, kırpılmış yüzeylerin birleşiminin görünen yüzeyini belirlemek. Görünen yüzey, her bir piksel için, bir bakış ışını boyunca, bakış noktasına en yakın kırpılmış yüzeyin seçilmesi ile elde edilir.



Şekil 4. İçinde/dışında testi



Şekil 5. Kırpılmış iki yüzey arasında görünen kısmın belirlenmesi

3.SONUÇ

Kırpma algoritmasının uygulandığı CSG görüntü programı C programlama dili ile yazılmış ve SUN işistasyonunda çalıştırılmıştır. Şekil 6.'da bu programla elde edilen bazı görüntülere yer verilmiştir. Burada görülebileceği gibi, oldukça karmaşık cisimlerin görüntülenebileceği açıktır.

Referanslar

[1] Fuchs, H., and et al. "Fast Spheres, Shadows, Textures, Transparencies, and Image Enhancements in Pixel-Planes." ACM.Vol. 19 (3). pp 111-120. 1985.

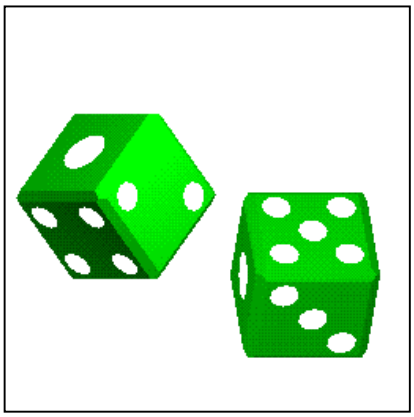
[2] Çevik, U. Design of an FPGA Based Parallel Architecture Processor Displaying CSG Volumes and Surfaces. PhD. Thesis. University of Sussex, Brighton. June 1996.

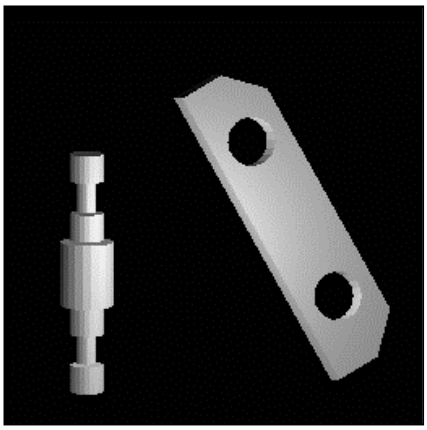
[3] Epstein D, Jansen F, Rossignac J. Z-Buffer rendering from CSG: The trickle algorithm. IBM Research report RC 15182. November 1989.

[4] Stewart N, Leach G, John S. A Z-Buffer rendering algorithm for convex objects. Proceedings of the 8-th International Conference in Central Europe on Computer Graphics, Visualisation and Interactive Digital Media' 2000-WSCG 2000, Volume II, pp. 369-372.

[5] Su C J, Lin F H, Ye L. A new collision detection method for CSG-represented objects in virtual manufacturing. Computers in Industry, 40 (1), pp. 1-13. September 1999.

[6] Koç, S. Development of an Algorithm for the Elimination of Surface Sorting in the Stage of Hidden Surface Removal in Displaying Constructive Solid Geometry (CSG) volumes and surfaces, June 1999.





Şekil 6. Görüntü algoritması ile elde edilen bazı görüntüler