PAMUKKALE ÜNİVERSİTESİ MÜHENDİSLİK FAKÜLTESİ
PAMUKKALE UNIVERSITY ENGINEERING COLLEGE
MÜHENDİSLİK BİLİMLERİ DERGİSİ
JOURNAL OF ENGINEERING SCIENCES
YIL : 2002
CİLT : 8
SAYI : 1
SAYFA : 63-69

JAVA PROGRAMLAMA DİLİNDE IŞIN İZLEME GERÇEKLEŞTİRİMİ

Aybars UĞUR, Mustafa TÜRKSEVER
Ege Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Bornova/İzmir

Geliş Tarihi : 17.04.2001

ÖZET

Bu makalede ilk olarak bilgisayar grafiklerinde gerçekçilik ve gerçekçiliği sağlayan bileşenler üzerinde durulmuştur. Bu kapsamda aydınlatma modellerine, yüzey kaplama yöntemlerine ve ışık kaynaklarına değinilmiştir. Ardından, üç boyutlu sanal bir ortamın iki boyutlu gerçekçi resmini oluşturma işlemi yani Işın İzleme (Ray Tracing) anlatılmıştır. Basit bir ışın izleme algoritması verilmiştir. Bu çalışma kapsamında, Java programlama dilinde gerçekleştirimi yapılan ve internet üzerinde işletilebilen "SahneIzle" yazılımı tanıtılmıştır. Son olarak, ışın izleme yazılımlarının İnternet üzerinde kullanımının önemi belirtilmiştir.

Anahtar Kelimeler : Işın izleme, Grafik programlama, Görsel Gerçekçilik, Java Programlama Dili, Web'de üç boyut

RAY TRACING IMPLEMENTATION IN JAVA PROGRAMMING LANGUAGE

ABSTRACT

In this paper realism in computer graphics and components providing realism are discussed at first. It is mentioned about illumination models, surface rendering methods and light sources for this aim. After that, ray tracing which is a technique for creating two dimensional image of a three-dimensional virtual environment is explained briefly. A simple ray tracing algorithm was given. "SahneIzle" which is a ray tracing program implemented in Java programming language which can be used on the internet is introduced. As a result, importance of network-centric ray tracing software is discussed.

Key Words : Ray Tracing, Graphics Programming, Visual Realism, Java Programming Language, 3D on the Web

1. GİRİŞ

Bilgisayar grafiklerinde yapılmakta olan çalışmaların önemli bir kısmı gerçekçiliği ve niteliği artırma üzerinedir. Gerçekçiliği sağlama için aydınlatma modelleri ve yüzey kaplama yöntemleri kullanılmaktadır.

İnternet üzerine üç boyutlu grafiksel içerik ekleme konusunda son günlerde yoğun olarak çalışmalar yapılmaktadır. Web3D şirketleri, üç boyutlu içerik oluşturmayı ve görüntülemeyi sağlayan yazılımlar geliştirmektedirler. Bu yazılımları alan şirketler, ürünlerinin 3B modellerini oluşturarak kendi web sayfalarına yerleştirmektedirler. İnternet kullanıcıları ise, "Netscape Navigator" ve "Internet Explorer" gibi standart web tarayıcıları ile bu sayfalara erişerek ürünleri (cep telefonu, otomobil, matkap gibi) üç boyutlu, etkileşimli ve canlandırmalı olarak inceleyebilmektedir.

Internet2 ve NGI (Next Generation Internet) gibi, ileri ağ teknolojilerini oluşturmaya yönelik projelerden elde edilecek başarılarla daha yüksek bant genişliği, servis kalitesi sağlanacak ve çok kullanıcılı, etkileşimli, gerçek zamanlı uygulamaların beklediği ortam oluşacaktır. Web3D yazılımlarında gerçekçiliği artırma konusunda da daha ciddi çalışmalar yapılacaktır.

Bu çalışma kapsamında, internet üzerinde gerçekçi görüntüler oluşturmaya yönelik olarak, Java programlama dilinde "SahneIzle" yazılımı geliştirilmiştir.

2. AYDINLATMA MODELLERİ VE YÜZEY KAPLAMA YÖNTEMLERİ

Ortamların gerçekçi görüntüleri, nesnelerin perspektif izdüşümleri üretilerek ve görünen yüzeylere doğal aydınlatma etkileri uygulanarak elde edilir. Bir aydınlatma modeli, ışıklandırma modeli (lighting model) veya gölgelendirme modeli (shading model) olarak da adlandırılır ve nesnelerin yüzeylerindeki noktaların görülen ışık yoğunluklarını hesaplamak için kullanılır. Yüzey kaplama algoritması, sahnedeki tüm yüzeylerin izdüşüm noktalarının ışık yoğunluğunu belirlemek için aydınlatma modelinin yoğunluk hesaplamalarını kullanır. Yüzey kaplama, görünen her yüzey noktasına aydınlatma modelinin uygulanması ile yapılabildiği gibi aydınlatma modeli hesaplamalarının küçük bir bölümünün yapılmasının ardından yüzeyler boyunca ara değerlerin hesaplanması (interpolasyon) ile de tamamlanabilir. Tarama hattı (scan-line) ve görüntü uzayı (image-space) algoritmaları genelde interpolasyon düzenlerini kullanırken ışın izleme (ray-tracing) algoritmaları, her görüntü yüzey noktası konumunda aydınlatma modelinden yararlanır. Yüzey kaplama prosedürleri, yüzey gölgelendirme yöntemleri olarak da adlandırılır. Karışıklığın önlenmesi açısından tek yüzey noktasındaki yoğunluğu hesaplamak için kullanılan modele “aydınlatma modeli” veya “ışıklandırma modeli” denmesi uygun olacaktır. Yüzey kaplamayı da sahnedeki tüm yüzey izdüşüm noktalarının yoğunluklarını elde etmekte kullanılan bir aydınlatma modeli yöntemi olarak görmek yararlı olacaktır (Hearn and Baker, 1997).

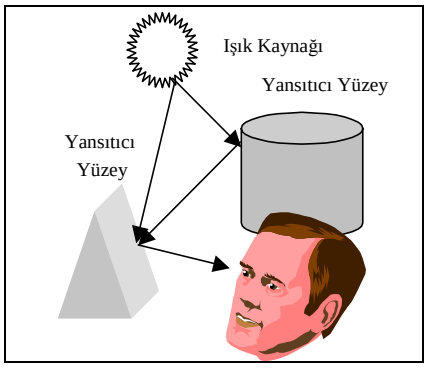
Bilgisayar grafiklerinde gerçekçiliği sağlamak için iki bileşene dikkat edilmesi gerekir : Nesnelerin grafiksel olarak doğru temsili ve ortamdaki aydınlatma etkilerinin (ışık yansımaları, saydamlık, yüzey dokuları yani “texture” ve gölgeleri) fiziksel tanımlamalarının iyi yapılması.

Bir nesneye bakıldığında görülen renkleri ve aydınlatma etkilerinin modellenmesi hem fizik hem de psikoloji kanunlarını içeren karmaşık bir işlemdir. Temelde aydınlatma etkileri, nesne yüzeylerinin elektromanyetik enerji etkileşimlerini dikkate alan modellerle tanımlanırlar. Işık göze ulaştığında ortamda gerçekte neyin görüldüğünü belirleyen algılama işlemleri başlatılır. Fiziksel aydınlatma modelleri, nesnenin türü, nesnenin ışık kaynaklarına ve diğer nesnelere göre yeri ve ortam için ayarlanan ışık kaynağı şartları gibi bir dizi faktörü içerirler. Nesneler, saydam olmayan modellerden oluşabildiği gibi, az veya çok saydam da olabilirler. Ayrıca parlak veya donuk yüzeyleri olabildiği gibi değişik yüzey dokularına da sahip olabilirler. Değişik tür, şekil, renk ve konumlardaki ışık kaynakları, ortamdaki aydınlatma etkilerini sağlamak için kullanılabilirler. Yüzeylerin optik özellikleri ile ilgili olarak verilen parametreleri, ortamdaki yüzeylerin göreceli konumlarını, ışık kaynaklarının renkleri ve konumlarını, görüntü düzleminin yeri ve yönünü dikkate alan aydınlatma modelleri, belirtilen görüntü yönünde tanımlanmış belirli bir yüzey noktasından izdüşümü alınan yoğunluk değerini hesaplarlar (Hearn and Baker, 1997).

Bilgisayar grafiklerindeki aydınlatma modelleri, yüzey ışık yoğunluklarını tanımlayan fizik kanunlarından türetilir. Yoğunluk hesaplamalarını en aza indirmek için birçok grafik paketi basitleştirilmiş fotometrik hesaplamalara dayalı deneysel modeller kullanır. Işıma (radiosity) algoritması gibi daha gerçekçi modeller ışık yoğunluklarını ortamdaki yüzeyler ve ışık kaynakları arasındaki ışıma enerjisinin yayılımını dikkate alarak hesaplarlar (Ashdown, 1994). İzleyen bölümlerde, ilk olarak grafik paketlerinde aydınlatmada yaygın olarak kullanılan ışık kaynaklarına değinilecektir. Sonra da, yüzey yoğunluklarının hesaplanmasında kullanılan doğru fakat daha zaman alıcı yöntemlerden ışın izleme algoritmaları anlatılacaktır.

3. IŞIK KAYNAKLARI

Saydam olmayan ve ışık saçmayan bir nesneye bakıldığında nesnenin yüzeylerinden yansıyan ışık görülür. Toplam yansıyan ışık, tüm ışık kaynaklarının ve ortamdaki diğer yansıtıcı yüzeylerden gelen yansımaların katılımının toplamına eşittir (Şekil 1).



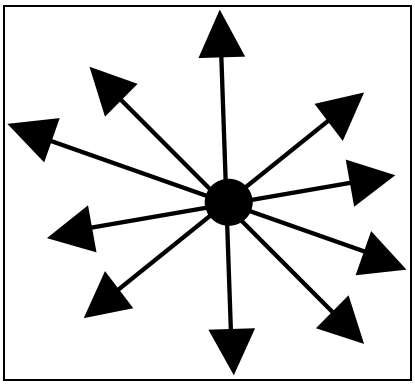
Şekil 1. Işık yansıması

Sonuçta, bir ışık kaynağından doğrudan ışık almayan bir yüzeyin yakınındaki nesneler de aydınlatılırsa görülebilir. Bazen ışık kaynakları, ışık yayan kaynaklar olarak, oda duvarları gibi yansıtıcı yüzeyler de ışığı yansıtan kaynaklar olarak adlandırılır. Işık kaynağı deyimi genelde güneş ve

lamba gibi ışıma enerjisini yayan nesneler için kullanılmaktadır (Hearn and Baker, 1997).

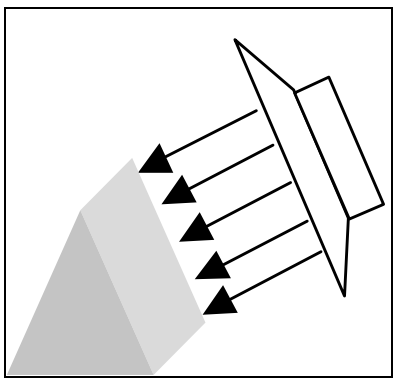
Işık yayan bir nesne genelde hem bir ışık kaynağı hem de ışık yansıtıcısı olabilir. Örnek olarak içinde lamba olan plastik bir küre hem ışık yayar hem de kürenin yüzeyinden ışık yansıtır. Küreden yayılan ışık, çevresindeki diğer nesneleri aydınlatabilir.

En basit ışık yayan model nokta kaynaktır (point source). Böyle bir kaynağın ışınları Şekil 2'de gösterildiği gibi kaynağın bulunduğu yerde merkezden çıkan yolları izler.



Şekil 2. Nokta ışık kaynağından çıkan ışın yolları

Nokta ışık kaynağı modeli, ortamdaki nesnelerin büyüklükleri ile karşılaştırıldığında küçük boyutları olan kaynaklar için uygun bir yaklaşımdır. Güneş gibi ortamdan yeterince uzak kaynaklar, nokta kaynaklar şeklinde doğru olarak modellenebilir. Şekil 3'teki uzun floresan ışığı gibi yakındaki bir kaynağı, dağıtık ışık kaynağı (distributed light source) olarak modellemek daha uygun olur. Bu durumda aydınlatma etkilerine nokta kaynak ile gerçekçi olarak yaklaşım yapılamaz. Çünkü kaynağın alanı ortamdaki yüzeyler ile karşılaştırıldığında küçük değildir. Dağıtık kaynak için doğru bir model, kaynağın yüzeyi üzerindeki noktaların aydınlatma etkilerinin toplamını dikkate almalıdır.

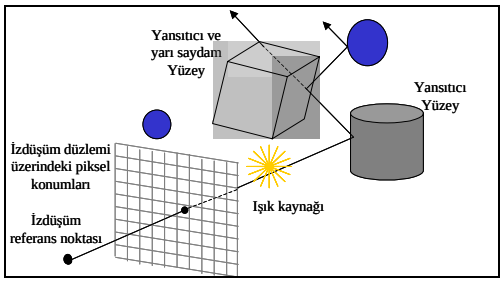


Şekil 3. Dağıtık ışık kaynağı ile aydınlatılmış bir nesne

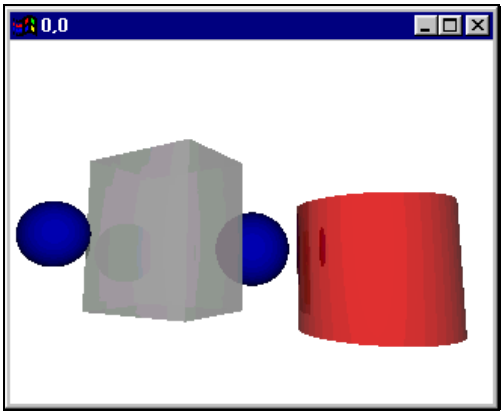
Işık saydam olmayan bir yüzeye geldiğinde, bir miktarı yansıtılırken bir miktarı emilir. Işığın yüzey tarafından yansıtılma miktarı, maddenin türüne bağlıdır. Parlak nesneler gelen ışığın çoğunu yansıtırken donuk yüzeyler gelen ışığın çoğunu emer. Aynı şekilde aydınlatılmış saydam bir yüzey, gelen ışığın birazını yansıtırken birazını da maddeden geçirecektir.

4. ÖZYİNELİ IŞIN İZLEME (RECURSİVE RAY TRACİNG)

Işın izleme (ray tracing), üç boyutlu sanal bir ortamın iki boyutlu resmini oluşturma işlemidir. Bu ortamda veya modelde yüzeyler ve ışık kaynakları bulunabilmektedir. Bakış noktası, modelin ekranda görüntülenecek olan kısmını belirler. Bakış noktasının önünde bir görüntü düzlemi seçilir. Modelin bakış noktasından görünen bölümü, görüntü düzleminde resmi biçimlendirir (Şekil 4). (Şekil 5). Işık kaynaklarının ışık ışınlarını yaydığı varsayılır. Bu ışınlar her yöne yayılabilmektedir. Bu ışınların bir bölümü yüzeylere çarpmakta ve sonra kısmen emilmekte, yansıtılmakta veya kırılmaktadır. Bazı ışık ışınları yansıtıldıktan veya kırıldıktan sonra yollarına devam etmektedirler. Bakış noktasından geçme veya geçmeme olasılıkları vardır (Reinhard et al., 1999).



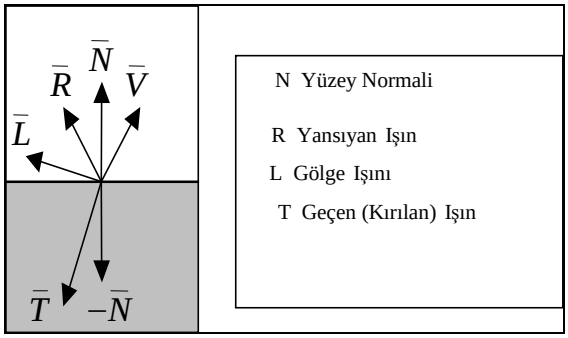
Şekil 4. Bir piksel konumu için, izdüşüm referans noktasından çıkarılan ışının izlenmesi [Birden çok yansıma ve iletim içermektedir]



Şekil 5. Şekil 4'teki sahnenin POV-RAY ışın izleme programında gerçekleştirilen ve bakış noktasına göre oluşturulan resmi

Bir resmi oluşturmak için işlem tersine çevrilir ve ışınlar bakış noktasından ortama doğru izlenir. Bu ışınlara birincil ışınlar adı verilmektedir. Böyle bir ışın bir yüzeyle kesiştiğinde yüzeyin gölgelendirmesi belirlenir ve görüntü düzlemindeki piksele atanır. Bu yöntem, ışın izlemenin en temel şeklidir. Resmin kalitesini artırmak veya işlem sayısını azaltmak için, aydınlatma etkilerini daha çok dikkate alacak şekilde iyileştirmeler yapılabilmektedir.

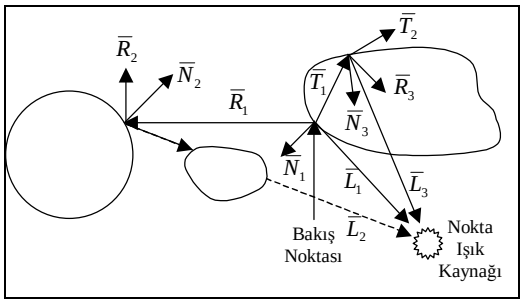
Temel ışın izleme algoritması, gözden çıkan ışın ile bir nesnenin kesişimine en yakın pikselin rengini belirlemekle sınırlıdır. Işın izleme algoritmaları, gölgeleri, yansımaları ve kırılmaları da dikkate almalıdır. Gölgeli bölgeleri bulabilmek için kesişim noktalarından tüm ışık kaynaklarına da ışınlar gönderilir. Işınlar arada herhangi bir nesneye rastladıklarında nesne bu noktasında gölgelidir ve algoritma bu ışık kaynağının katılımını dikkate almaz. Bilgisayar grafiklerinde ışın izlemeye ilişkin bir makale Arthur Appel tarafından 1968 yılında yayınlanmıştır. Whitted (1980), temel ışın izleme algoritmalarını, yönlü yansımaları ve kırılganlığı da dikkate alacak şekilde genişletmiştir. Whitted'ın algoritması, ikincil ışınları da dikkate almaktadır. Bu konuda ayrıntılı bilgi Foley et al., (1997)'de yer almaktadır. İkincil ışınlar, Şekil 6'da gösterilmektedir. .

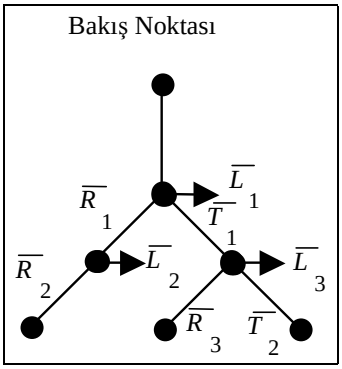


Şekil 6. Kesişim noktasından çıkan yansıma, kırılma ve gölge ışınları

- **Gölgelendirme Işınları (Aydınlatma Işınları) :** Bu ışınlar, ışığı, ışık kaynağından yüzeye doğrudan taşırlar.
- **Yansıyan Işınlar :** Bu ışınlar, yansıma yönünden gelen ışığın katılımını hesaplamada kullanılırlar.
- **Kırılan veya Geçen Işınlar :** Bu ışınlar, ışığı nesne boyunca taşırlar. Bir nesnenin geçirgenliği varsa, ışınlar bu ortama geçerken Snell kanununa göre kırılırlar.

Yansıyan ve geçen ışınlar birincil ışınların oluşturulmasına benzer şekilde oluşturulduklarından algoritmaya özyineleme getirirler (Şekil 7). Bitiş şartı olarak, özyinelemenin en yüksek düzeyini belirtmek üzere genelde bir eşik değeri verilir. Özyineli algoritmada kullanılan veri yapısı olan ışın ağacı ikili bir ağaç şeklindedir.





Şekil 7. Kesişim noktalarında diğer ışınlardan özyineli oluşturulan ışınlar ve kullanılan ışın ağacı

Işın ağaçlarının tutulması ve işlenmesi için gerekli olan yer ve zamanı en etkin hale getirmek için ilk teknikler 1989 yılında geliştirilmiştir.

Basit bir özyineli ışın izleme algoritması aşağıda verilmektedir (Foley et al., 1997) :

```
        Görüntü düzlemindeki pencerenin ve izdüşüm merkezinin seçilmesi
for (resimdeki her tarama hattı için) {
        for (tarama hattındaki her piksel için) {
                izdüşüm merkezinden çıkarak pikselden geçen ışını belirle
                piksel = RT_TRACE(ray,1);
        }
}
```

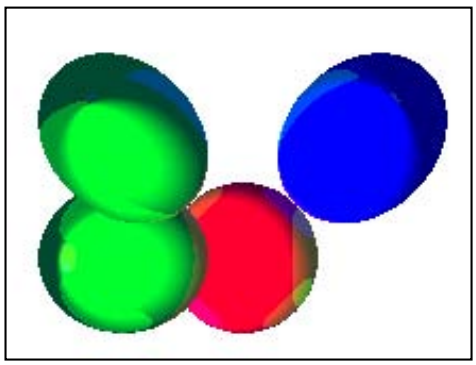
```
Işını nesnelerle kesiştir ve en yakındaki kesişimin rengini belirle
depth, ışın ağacındaki anlık derinliktir.
RT_color RT_trace(RT_ray ray, int depth)
{
        ışının bir nesne ile en yakın kesişimini belirle
        if(kesişen nesne varsa) {
                kesişimin yüzey normalini hesapla
                return RT_shade(en yakın nesne, ışın, kesişim, normal, derinlik);
        } else
        return zemin_rengi_değeri;
} /* RT_trace */
```

```
Nesne üzerindeki noktanın rengini belirle (gölgelendirme, yansıma ve kırılma ışınlarını
dikkate alarak)
RT_color RT_shade(
        RT_object object; /* Kesişen nesne */
        RT_ray ray; /* Gelen ışın */
        RT_point point; /* Kesişim noktası */
        RT_normal normal; /* Noktadaki yüzeyin normali */
        int depth; /* Işın ağacının derinliği */ )
{
        RT_color color; /* Işının rengi */
        RT_ray rRay, tRay, sRay, /* Yansıyan ışın, kırılan ışın ve gölge ışını */
        RT_color rColor, tColor; /* Yansıyan ve kırılan ışın renkleri */
        color = genel aydınlık değeri;
        for(her ışık) {
                sRay = noktadan ışığa giden ışın;
            if(normalin ve ışığa giden ışının noktasal çarpımı pozitifse) {
            Geçirgen olan veya olmayan yüzeyler tarafından tutulan ışık miktarını
            hesapla ve renge katmadan önce yayılan ve yönlü terimleri ölçeklendirmek
            için kullan; }
        }
        if(depth<maxDepth)
                if(nesne yansıtıcı ise) {
                rRay = noktadan yansıma yönüne ışın;
                rColor = RT_trace(rRay,depth+1);
                rColor'ı yönlü katsayı ile ölçeklendir ve renge ekle;
                }
        if(nesne geçirgense) {
                tRay = noktadan kırılma yönüne ışın;
            tColor = RT_trace(tRay,depth+1);
            tColor'ı geçirgenlik katsayısı ile ölçeklendir ve renge kat;
        }
        return color;
} /* RT_shade */
```

5. JAVA'DA IŞIN İZLEME GERÇEKLEŞTİRİMİ

Bu çalışma kapsamında Java'da İnternet üzerinden uzaktan erişilerek de çalıştırılabilen basit bir ışın izleme programı yapılmıştır. Programın gerçekleştirimi sırasında Java'nın awt (abstract windowing toolkit) paketi ve iki boyutlu grafik özelliklerini destekleyen Java 2D API kullanılmıştır. Ayrıntılı bilgi Knudsen (1999) ve Deitel (1999)'de yer almaktadır. Java 3D API, ışın izleme yöntemlerini desteklemediği için kullanılmamıştır. Hazırlanan program bu açığı da kapatmaktadır. Şekil 8'de applet penceresi görülen "SahneIzle" adlı bu ışın izleme programında daire

ve kure sınıfları oluşturulmuştur. Bu sınıflardan türetilen nesneler istenildiği zaman istenildiği sayıda oluşturulabilmektedir. Sınıflara eklenen metotlar, değişik boyutlarda, konumlarda ve renklerde daire ve küreler oluşturulmasını sağlamaktadır. Şekil 9'da küre sınıfının tanımlarını içeren kaynak kod parçası görülmektedir :



Şekil 8. Geliştirilen "SahneIzle" ışın izleme programının applet penceresi

```
// Küre sınıfı
class RT_kure
{
  public double x, y, z, r;
  public Color renk;

  public RT_kure()
  {
    x = 0; y = 0; z =-100; r = 100; renk =
new Color(255,0,0);
  }

  public RT_kure(double c_x, double c_y,
double c_z, double radius)
  {
    x = c_x; y = c_y; z =c_z; r = radius;
renk = new Color(255,0,0);
  }

  public RT_kure(double c_x, double c_y,
double c_z, double radius,
  Color renkd)
  {
    x = c_x; y = c_y; z =c_z; r = radius;
renk = renkd;
  }
```

Şekil 9. Küre sınıfı

Nesnelerin birden fazla ışık kaynağı ile aydınlatılabilmesi sağlandıktan sonra, basit bir ışın izleme mekanizması kurularak ortamdaki tüm nesnelerin aydınlatılabilmesi gerçekleştirilebilmiştir. Bakış noktasından görüntü düzleminin her bir pikseline sıra ile ışınlar gönderilerek ışın ile kesişen nesneler belirlenmiştir. Önde olan nesne dikkate alınarak, ışık kaynaklarından gelen ışık miktarlarına göre ilgili pikselin yoğunluğu hesaplanmakta ve applet penceresine o renkte bir nokta konulmaktadır.

Bu işlem sırasında, bir ışının belirli bir konumdaki küreyi kesip kesmediğini belirlemek ve kesiyorsa, hangi nokta veya noktalarda kestiğini bulmak önem taşımaktadır. Bu amaçla aşağıdaki formüller çıkarılarak programlanmıştır:

Işının uç noktaları (x_1, y_1, z_1) ve (x_2, y_2, z_2) ise ışın parametrik hale getirilir (Watt, 2000) :

$x = x_1 + (x_2-x_1)t = x_1 + it$
$y = y_1 + (y_2-y_1)t = y_1 + jt$
$z = z_1 + (z_2-z_1)t = z_1 + kt$
ve $0 \le t < 1$ olmak üzere,

(l, m, n) merkezli, r yarıçaplı küre :
$(x-l)^2+(y-m)^2+(z-n)^2 = r^2$

x, y, z yerine konduğunda :

$at^2 + bt + c = 0$
$a = i^2+j^2+k^2$
$b = 2i(x_1-l)+2j(y_1-m)+2k(z_1-n)$
$c = l^2+m^2+n^2+x_1^2+y_1^2+z_1^2+2(-lx_1-my_1-nz_1)-r^2$

Bu ikinci dereceden denklemin determinantı 0'dan küçükse, ışın küreyle kesişmemekte, 0'a eşitse tek noktada kesmekte, büyükse iki noktada kesmektedir. Gerçel kökler öndeki ve arkadaki kesişimleri verir. İlgili değerler yerine konularak kesişim noktaları hesaplanır.

6. SONUÇ

Ağ merkezli programlama dili olan Java üzerinde gerçekçi görüntüler oluşturulmasına yönelik olarak yapılacak çalışmaların önemi büyüktür. Bu günlerde çok popüler olan ve üzerinde çalışılan internet üzerine üç boyutlu grafiksel içerik ekleme konusu, ileride yerini gerçekçi görüntüler oluşturulmasına bırakacaktır. Bu alanda ışın izlemenin önemi büyüktür. Şu anda üç boyutlu grafiksel içeriklerden yararlanılan alanlar aşağıdaki gibidir :

- E-ticaret ve E-reklam
- Uzaktan Eğitim
- Kültür ve Sanat

Örnek olarak e-ticaret ve e-reklam alanlarında cep telefonu şirketleri yeni çıkardıkları ürünlerin üç boyutlu modellerini internet üzerine yerleştirerek, dünyanın her tarafından sayfalarına erişen müşterilerin cep telefonlarını inceleyebilmelerini sağlamaktadırlar. Telefonun rengini değiştirmekten, üç boyutlu olarak döndürmeye, düğmelerine sanal olarak basarak menülerinde gezinmeye ve daha yakından görüntülemeye kadar birçok hizmet vermektedirler. Örnekler çoğaltılabilir. Karmaşık ürünlerdeki gerçekçiliği artırmak, müşterilerin ürünleri daha net olarak inceleyebilmelerini sağlayacaktır. Üç boyutlu gerçekçi bir model, her

zaman bir fotoğraftan çok daha fazlasını ifade etmektedir.

7. KAYNAKLAR

Ashdown, I. 1994. "Radiosity : A Programmer's Perspective", John Wiley and Sons, Inc., USA.

Deitel, H. M., Deitel, P. J. 1999. "Java How to Program", Third Edition, Prentice-Hall.

Foley, J. D., Dam, A., Feiner, S. K. ve Hughes, J. F. 1997. "Computer Graphics Principles and Practice 2nd edition in C", Addison Wesley.

Hearn, D., Baker, M. P. 1997. "Computer Graphics C Version", Prentice-Hall.

Knudsen, J. 1999. "Java 2D Graphics", O'Reilly, USA.

Reinhard, E., Chalmers, E. and Jansen, F. W. 1999. *Hybrid Scheduling For Parallel Rendering Using Coherent Ray Tasks*, 1999. IEEE Parallel Visualization and Graphics Symposium, ACM SIGGRAPH,.

Watt, A. 2000. "3D Computer Graphics", Third Edition, Addison-Wesley.

Whitted, T. 1980. *"An Improved Illumination Model For Shaded Display"*, Communications of the ACM 23 (6), 343-349.