KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

YEREL AĞDAKİ KİŞİSEL BİLGİSAYARLARLA PARALEL IŞIN İZLEME

Bilgisayar Mühendisi Ömer ÇAKIR

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"Bilgisayar Yüksek Mühendisi"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 28.06.2004
Tezin Savunma Tarihi : 02.08.2004

Tez Danışmanı : Yrd. Doç. Dr. Cemal KÖSE
Jüri Üyesi : Doç. Dr. Rıfat YAZICI
Jüri Üyesi : Yrd. Doç. Dr. Ali GANGAL

Enstitü Müdürü : Prof. Dr. Yusuf AYVAZ

Trabzon 2004

ÖNSÖZ

Günümüzde bilgisayar grafikleri, bilgisayar mühendisliğinin popüler bir araştırma alanı olmuştur. Işın izleme gerçekçi 3-Boyutlu görüntü üretimi için yaygın olarak kullanılan bir yöntemdir. Işın izleme ile görüntü üretiminde hesaplama zamanının çok fazla olması karmaşık görüntülerin üretimi için paralel bilgisayar sistemlerinin kullanılmasını kaçınılmaz kılmıştır. Bu tezde, ışının yansıma ve kırılma özellikleri özyinelemeli bir algoritma ile modellenerek ışın izleme ile gerçekçi görüntüler üretilmiştir. Tek bilgisayarın hız açısından çok yetersiz kaldığı karmaşık görüntüler için yerel ağdaki bilgisayarlar paralel koşturulmuş, işlemci çiftliği (processor farm) modeli kullanılarak yük dengelemesi yapılmıştır.

Çalışmalarım boyunca bana değerli zamanını ayıran ve verdiği fikirler ile beni yönlendiren sayın Yrd. Doç. Dr. Cemal KÖSE 'ye teşekkür ederim. Ayrıca manevi olarak hep yanımda olan aileme ve özellikle Babama çok teşekkür ederim.

Ömer ÇAKIR
Trabzon, 2004

İÇİNDEKİLER

Sayfa No

ÖZET

Işın izleme yöntemi 3-Boyutlu görüntü üretiminde yaygın olarak kullanılan bir yöntemdir. Işın izleme, gerçek hayatta ışık kaynağından saçılan ışınların 3B nesnelerle etkileşimini fiziksel gerçeklere uygun olarak modelleyebildiğinden bu yöntem kullanılarak fotoğraf kalitesine yakın görüntü üretmek mümkündür. Işın izleme, kaliteli 3B görüntü üretimi için çok iyi bir yöntem olmasına karşın oldukça fazla hesaplama yapılmasını gerektiren bir yöntemdir. O nedenle tek bilgisayar kullanılınca üretimi uzun zaman alan bir görüntü paralel bilgisayarlar kullanarak çok daha kısa zamanda üretilebilir. Yeterli sayıda bilgisayar kullanılarak gerçek zamanlı görüntü üretimi de mümkündür.

Bu çalışmada, öncelikle ışın izleme ile 3B görüntü üretimi gerçeklenmiştir. Geliştirilen özyinelemeli algoritma ile yansıyan ve kırılan ışınlar da modellenmiştir. Gölgeleme yapılmış ve sert gölgeler yumuşatılmıştır. Üretilen görüntüdeki piksellerin renk değerleri Phong aydınlatma modeli ile belirlenmiştir. Doku kaplama yapılarak görüntünün kalitesi arttırılmıştır. Bump mapping ile doku kaplama yapılmıştır. Doku kaplamada karşılaşılan aliasing problemi supersampling yapılarak çözümlenmiştir. Işın izlemeyi hızlandırmak için arka yüz kaldırma ve çevreleyen hacim yöntemleri kullanılmıştır. Ayrıca etkileşimli ışın izleme yapılmıştır. Buna göre kullanıcı üretilen görüntü içerisinde tuşlarla ile geri ilerlemekte ve sağa sola dönebilmektedir.

Işın izlemede herhangi bir pikselin renginin hesaplanması diğerlerinden bağımsızdır. Bu özelliği ile ışın izleme paralel hesaplamaya yatkın bir yöntemdir. Yapılan çalışmada yerel ağ üzerindeki bilgisayarlar paralel çalıştırılmış üretilen görüntüler için speedup değerleri hesaplanmıştır. Heterojen bir ağ için yük dengelemesi yapmak paralel hesaplamanın etkinliği açısından gereklidir. Ayrıca üretilecek görüntüdeki herhangi bir piksel için gereken zaman diğerinden farklı olabilmektedir. O nedenle ağ homojen bile olsa yük dengelemesi yapmak gerekebilir. Bunun için paralel çalışmada işlemci çiftliği modeli gerçeklenmiştir. Buna göre ana bilgisayar üretilecek görüntüyü parçalara bölerek bir iş havuzu oluşturur. Ağ üzerindeki her bir bilgisayar da bu iş havuzundan bir iş alıp tamamlar ve tekrar havuzdan iş ister. Böylece dinamik iş tahsisi ve yük dengelemesi yapılmıştır.

Anahtar Kelimeler: Işın İzleme, Paralel Işın İzleme, İşlemci Çiftliği, Dinamik Yük Dengeleme.

SUMMARY

Parallel Ray Tracing on Networked Personal Computers

Ray tracing technique is commonly used to generate realistic images in computer graphics. Ray tracing is a realistic technique to describe the interaction between rays spreading from a light source and objects. Although ray tracing is a good technique for rendering a high quality three-dimensional image, it requires rather much processing time. Parallel ray tracing technique for general-purpose multi-computer system offers the potential for realistic rendering such complex scenes in a reasonable time. Therefore, real-time image rendering may be achieved when sufficient numbers of computers are used.

In this study, ray tracing technique is performed for high quality three-dimensional image rendering. Phong illumination model is employed to determine the intensity and color values at each pixel. The quality of the image is improved by performing texture mapping. Applying bump mapping technique may produce even more realistic textured three-dimensional images. Performing super-sampling solves aliasing encountered in the image and improves the quality of the final image. Then, back-face culling and bounding volume methods are employed to accelerate ray tracing. In addition, interactive ray tracing is achieved to allow a user to move inside the scene.

In ray tracing, calculating the color value of a pixel is independent of others' which makes parallelisation of ray tracing technique rather simple. Thus, personal computers on a LAN can be effectively used as a parallel ray tracing platform. The ray tracing has a computational complex problem and so involves using a dynamic load balancing mechanism to ensure the best performance on a heterogeneous parallel computing platform. Therefore, the processor farm model is used for dynamic load balancing, where a task pool is maintained on the master process. In this way each application process on the parallel system requests a task from the master process, performs it, and requests another untill the task pool is empty. Result shows that parallel ray tracing achieves almost linear speedup on the parallel system. As a result the parallel ray tracing technique uses the dynamic load balancing mechanism to produce high quality three-dimensional images in a reasonable time.

Keywords: Ray Tracing, Parallel Ray Tracing, Processor Farm, Dynamic Load Balancing.

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

SEMBOLLER DİZİNİ

b^c	Çevreleyen hacim dikdörtgen prizmanın merkezi
b^u , b^v , b^w	Çevreleyen hacim dikdörtgen prizma için normali edilmiş yön vektörleri
m_{dif}	yüzeyin diffuse rengi
m_{shi}	parlaklık parametresi (shininess parameter)
s_{dif}	Işık kaynağın diffuse bileşeni
S_e	Kürenin merkezinden *e*kvatora doğru olan vektör
S_n	Küre üzerindeki kesişim noktasının normali
S_p	Kürenin merkezinden kuzeye doğru olan vektör
3B	3 Boyutlu

1. GENEL BİLGİLER

1.1. Giriş

Günümüzde Bilgisayar Grafikleri, 3-Boyutlu tıbbî görüntüleme, gerçek hayattaki olayların simülasyonu (pilotlar için uçak simülasyonu, şoförler için araba simülasyonu), sinema filmi ve çizgi film yapımı, eğitim ve bilgisayar oyunları olmak üzere bir çok alanda kullanılmaktadır. Bilgisayar grafiklerinde temelde iki amaç vardır:

1. Gerçeğinden ayırt edilemeyecek kalitede görüntü üretimi.
2. Gerçek zamanlı 3B görüntü üretimi.

Bu iki amaçtan birincisi için kullanılan yöntemler çok zaman almakta; ikincisi için kullanılan yöntemlerle elde edilen görüntüler ise gerçeğine yakın kalitede olmamaktadır. O nedenle tek bilgisayarın yüksek kalitede görüntü üretimi için çok yavaş kaldığı durumda paralel bilgisayarlardan oluşan sistemler kullanmak kaçınılmaz olmaktadır.

Bu çalışmada yüksek kalitede 3B görüntü üretimi için yaygın olarak kullanılan Işın İzleme (Ray Tracing) yöntemi ile 3B görüntü üretimi gerçeklenmiştir. Işın izleme gerçek hayatta ışık kaynağından saçılan ışınların çarptığı nesnelerle olan etkileşimini fiziksel gerçeklere uygun olarak modelleyebilmektedir. Yansıyan ve kırılan ışınlar fiziksel gerçeklere uygun olarak modellenip yüksek kalitede 3B görüntüler üretilebilmektedir. Yalnız ışın izleme çok fazla hesaplama zamanı gerektiren bir yöntemdir. Tek bilgisayar kullanılarak üretimi çok fazla zaman alan görüntüler için paralel bilgisayarlar kullanmak gerekmektedir. Bu çalışmada hesaplama zamanı yüksek olan görüntüler yerel ağdaki bilgisayarlar paralel koşturularak üretilmiştir. Paralel hesaplamada karşılaşılan en öneli problemlerden biri olan yük dengeleme (load balancing), işlemci çiftliği (processor farm) modeli ile gerçeklenmiştir. Buna göre ana bilgisayar işi parçalara ayırıp bir iş havuzu (task pool) oluşturmaktadır. Yerel ağdaki her bir bilgisayar bu havuzdan iş istemekte ve işi tamamlayıp sonuçları yollayınca havuzdaki bütün işler tamamlanana dek yeni bir iş isteyip onunla ilgilenmektedir.

1.2. Işın İzleme

Üç boyutlu (3B) görüntü üretimi yöntemi olan Işın İzleme (Ray Tracing), ilk olarak 1968 yılında Appel trafından ortaya atılmıştır. Appel'in ışın izleme yöntemi 3B nesneler için yüzey boyama (surface shading) ve gölgelendirme (shadowing) yapabiliyordu. Whitted ve Kay'ın araştırmaları sonucu ışın izleme algoritması ışının, speküler yansıma (specular reflection) ve kırılma (refraction) özelliklerini de modelleyebilecek şekilde geliştirildi. Işın izleme, 3B görüntü üretimi teknikleri içinde gölge belirleme, speküler yansıma, kırılma ve katı cisim üretimi (volume rendering) işlemlerinde en iyi yöntem olarak bilinir [7].

Günümüzde 3B görüntü üretimi tekniklerinin görüntü üretirken yaptıkları iki ana iş vardır : görünmeyen yüzeylerin kaldırılması (hidden surface removal), boyama. Görünmeyen yüzeylerin kaldırılması, belli bir bakış noktasından bakan gözlemcinin manzaradaki 3B nesnelerin yüzeylerinden görünmeyenlerinin kaldırılarak görünenleri belirleme işlemidir. Boyama, 3B nesne üzerindeki herhangi bir noktanın rengini belirleme işlemidir. Renk belirleme ışıklandırmaya (lighting), gölgelere, geçirgenliğe, yansıtmaya, kırılma indisine ve yüzeyin dokusuna bağlı olarak belirlenir. Işın izleme yerel aydınlatmanın (local illumination) bu saydığımız bütün özellikleri için çözümler sunar.

1.2.1. İleri Yönde Işın İzleme

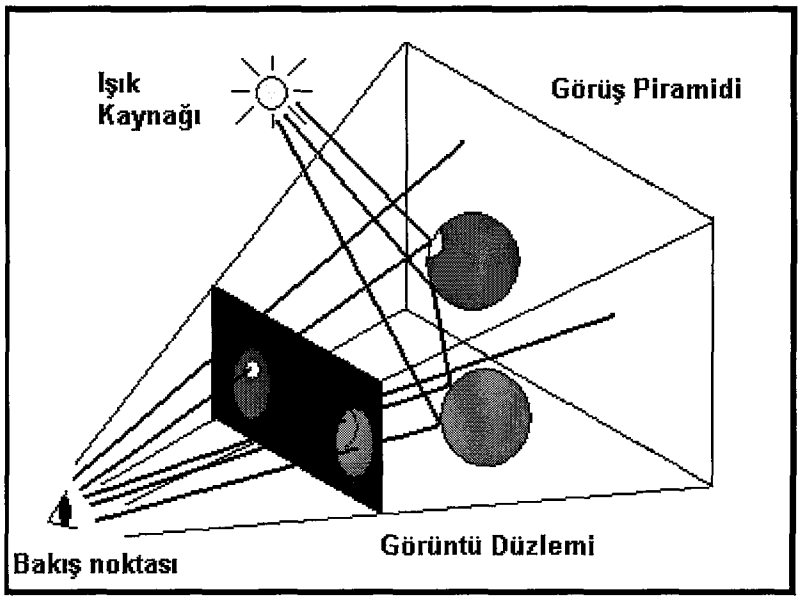
Işın izleme, gerçek hayatta etrafımızdaki nesneleri görmek için gerekli olan fiziksel esasları modellemeye çalışır. Standart ışın izleme algoritması fiziksel davranışlar için sayısal yaklaşımlar yaparak sınırlı hesaplama kaynaklarıyla gerçekçi boyama yapma esasına dayanır. Gerçek hayatta ışık, kaynağından fotonlar halinde yayılır. Işık kaynağından çıkan fotonlar önünü kesen bir yüzeye çarpana dek doğrusal olarak (ışınlar halinde) ilerler. Fotonlar ile yüzey arasındaki fiziksel etkileşim onların nasıl emildiğini, yansıdığını veya iletildiğini belirler. Gördüğümüz görüntüleri milyarlarca fotonun cisimlerden gözümüze ulaşması sonucu görürüz. Işık kaynağından yayılan ışınlar modellenirse bunun ışın izleme açısından karşılığı **İleri Yönde Işın İzleme (Forward Ray Tracing)**'dir. Bu yaklaşım ile bilgisayarda 3B görüntüler üretmek mümkündür yalnız bu durumda görüntü üretimi için geçen zamanın çok büyük bir kısmı gerçekte görünmeyen ışıkları modellemek için geçer. Çünkü gerçekte, ışık kaynağından saçılan ışınların çok azı

göze ulaşmaktadır. O nedenle çok fazla hesaplama zamanı gerektiren bu yöntem pek kullanılmaz.

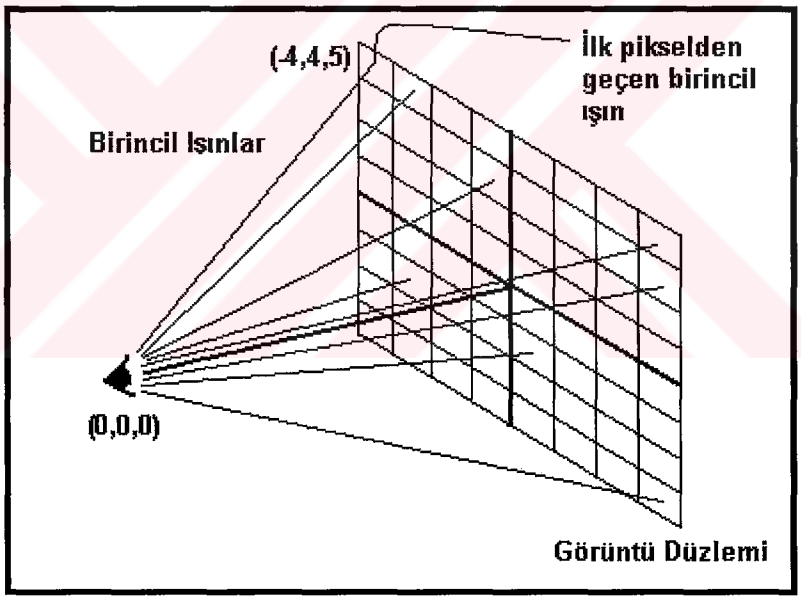
1.2.2. Geri Yönde Işın İzleme

Işınları gerçek hayattaki gibi ışık kaynağından bakış noktasına doğru yollamak yerine bakış noktasından ışık kaynağına doğru yollamak hesaplama açısından daha etkin bir yöntemdir. Görüntü üretimi için bu şekilde bir modelleme yapılırsa bunun ışın izleme açısından karşılığı **Geri Yönde Işın İzleme (Backward Ray Tracing)** olur. Geri yönde ışın izleme ile yalnızca gözlemciye gelen ışınlar için hesaplama yapılır. Geri yönde ışın izleme çok yaygın olarak kullanılan bir yöntemdir. O nedenle ışın izleme denince genelde geri yönde ışın izleme anlaşılır.

Geri yönde ışın izleme ile 3B görüntü üretimi Şekil 1.1.'de gösterilmiştir. Gözlemcinin belli bir bakış noktasından baktığı varsayılır. Bakış noktasının önünde görüntü düzlemi vardır. Görüntü düzlemi 3B görüntünün oluştuğu yerdir. Belli bir bakış noktasından yollanan ve görüntü düzleminin her bir pikselinden geçen ışınlar, cisimlerden yansıyarak ya da kırılarak ışık kaynağına ulaşıyorsa ilgili pikselin rengi ışık kaynağının yerine, cisimlerin rengine, yansıtma ve geçirgenlik özelliklerine göre belirlenir. Eğer ışın en son ışık kaynağına ulaşmamışsa ilgili piksel için bakış noktasından herhangi bir nesne görünmüyor demektir ve o piksel siyah renge boyanır. Genellikle cisimlerden yansıyan ya da kırılan ışınlar için belirli bir derinlik değeri tanımlanır. Yani belirli sayıda yansıyan ya da kırılan ışın, ışık kaynağına ulaşmamışsa görünmediği varsayılır. Üretilecek görüntünün kalitesini belirlemede bu değer önemlidir. Bu değer büyük seçilirse görüntü kalitesi yüksek olur. Yalnız bu durumda hesaplama zamanı oldukça artar. Nesnelerin görüntü düzleminde görüntülerinin oluşması için bakış noktasından başlayan bir pramidin içinde olmaları gerekir. Buna Görüş Piramidi (Viewing Frustum) denir.



Şekil 1.1. Geri yönde ışın izleme ile 3B görüntü üretimi



Şekil 1.2. Birincil ışınların üretimi

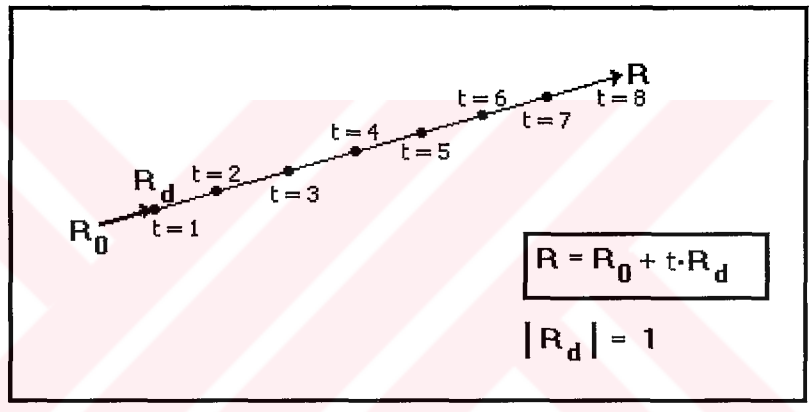
Bakış noktasından yollanan ve her bir pikselin merkezinden geçen ilk ışınlara Birincil Işınlar (Primary Rays) denir. Birincil ışınların bazıları Şekil 1.2.'de gösterilmiştir. Şekilde bakış noktası (0,0,0) 'dadır ve z-ekseni boyunca görüntü düzleminin merkeziyle aynı doğrultudadır. Görüntü düzleminin 8x8 çözünürlükte ve bakış noktasından 5 birim ileride olduğu varsayılmıştır. Işın izleme algoritması birincil ışınların üretilmesiyle başlar.

1.2.2.1. Işının Tanımı

Birincil ışınların nasıl üretildiğini anlatmadan önce ışının tanımını vermek faydalı olacaktır. Ardından Şekil 1.2.'de ilk pikselden geçen birincil ışının nasıl üretildiği anlatılacaktır.

Işın izleme yönteminin en temel bileşeni olan ışın, başlangıç noktası ve doğrultusu olan vektörel bir büyüklüktür. Başlangıç noktası $\mathbf{R_0}$'dan çıkan ve normalize edilmiş $\mathbf{R_d}$ doğrultusu boyunca ilerleyen **R** ışınını aşağıdaki gibi ifade etmek mümkündür [7]:

$$\mathbf{R} = \mathbf{R_0} + t\mathbf{R_d} \qquad ,t > 0 \tag{1.1}$$



Şekil 1.3. Işın ve R_d boyunca ilerlemesi

(1.1) ifadesinde **R,** ışının $\mathbf{R_0}$'dan t birim sonraki yerini gösterir. Yani t, **R** ile $\mathbf{R_0}$ arasındaki uzaklıktır. Farklı t değerleri için **R** ışınının $\mathbf{R_d}$ doğrultusu boyunca nasıl hareket ettiği Şekil 1.3.'te gösterilmiştir. t değerinin nasıl hesaplandığı Kesişim Testleri konusunda anlatılacaktır. t değerinin sayısal olarak tam anlamıyla **R** ile $\mathbf{R_0}$ arasındaki uzaklık olabilmesi için ışının doğrultusu olan $\mathbf{R_d}$'nin uzunluğunun 1 birim olması yani birim vektör olması gerekir. Çünkü ışın ifadesinde $\mathbf{R_0}$ 'a eklenen $t\mathbf{R_d}$ çarpımındaki t, **R** ile $\mathbf{R_0}$ arasındaki uzaklık olduğuna göre $\mathbf{R_d}$'nin boyu 1 olmalı ki $\mathbf{R_d}$ boyunca t birim kadar gidildiğinde tam olarak **R**'nin konumu elde edilsin. Bunun için de $\mathbf{R_d}$ normalize edilmelidir.

1.2.2.2. İlk Birincil Işının Üretilmesi

İlk birincil ışının üretilmesi için bu ışının doğrultusunun belirlenmesi gerekir. Bunun için (1.1) ifadesi $\mathbf{R_d}$'ye göre düzenlenirse aşağıdaki ifadeyi elde edilir:

$$\mathbf{R_d} = (\mathbf{R} - \mathbf{R_0}) / t \qquad (1.2)$$

(1.2) ifadesinden de görüldüğü gibi ışının doğrultusunu bulmak için 3B uzayda iki noktaya ihtiyaç vardır. Bunlardan birincisi $\mathbf{R_0}$ = [0 0 0] olan başlangıç noktasıdır. İkinci nokta ise ilk pikselin koordinatlarıdır. Şekil 1.2. 'ten de görüldüğü gibi bu **R** = [-4 4 5] noktasıdır. t değeri de yaklaşık 7.55 'tir. Bu değerler (1.2) ifadesinde yerine koyulursa $\mathbf{R_d}$ aşağıdaki gibi hesaplanır:

$$\mathbf{R_d} = [\ -4/7.55 \quad 4/7.55 \quad 5/7.55\ ] = [\ -0.529 \quad 0.529 \quad 0.662\ ] \qquad (1.3)$$

(1.3) ifadesindeki sayısal değerlere dikkat edilirse (1.2) ifadesindeki ($\mathbf{R} - \mathbf{R_0}$)'ın t'ye bölünmesi aynı zamanda $\mathbf{R_d}$'nin de normalize edilmiş halini verir. Çünkü (1.3) ifadesinde $\mathbf{R_d}$'nin bileşenlerinin karelerinin toplamının karekökü yani $\mathbf{R_d}$'nin boyu :

$$| \mathbf{R_d} | = \sqrt{(-0.529)*(-0.529)+(0.529)*(0.529)+(0.662)*(0.662)} = 1 \text{ 'dir.}$$

$\mathbf{R_d}$'nin bulunmasıyla ilk birincil ışın üretilmiş olur. Işın ifadesindeki t değerinin hesaplanması için kesişim testlerinin yapılması gerekmektedir. Kesişim testi için ışının başlangıç noktası $\mathbf{R_0}$ ve doğrultusu $\mathbf{R_d}$'yi bilmek yeterlidir. Kesişim testleri yapılırken bu değerler kullanılarak eğer kesişim varsa t değeri hesaplanır, geri döndürülür. Hesaplanan bu t değeri yardımıyla ışının kesiştiği yüzey üzerindeki koordinatları hesaplanır. Eğer ışın bu yüzeyden yansıyarak veya kırılarak yoluna devam edecekse bu koordinatlar yansıyan veya kırılan ışın için yeni başlangıç noktası yani $\mathbf{R_0}$ olur. Yansıma veya kırılma doğrultuları hesaplanır ve ışın yeni başlangıç noktası ve doğrultusu boyunca yollanır. Üretilen bu yeni ışınlara **İkincil Işınlar** denir. İkincil ışınlar için de yine birincil ışınlar için yapılan işlemler tekrarlanır. Işınlar yansıyıp kırıldıkça üçüncül, dördüncül... n.'cil olmak üzere yeni ışınlar üretilir. Bu işlemler ışın ışık kaynağına ulaşıncaya kadar veya n derinlik değerine kadar özyinelemeli (recursive) olarak tekrarlanır.

Şekil 1.2.'de basitlik açısından görüntü düzleminin hem piksel mertebesinde çözünürlüğü 8x8 olduğu varsayıldı hem de eni ve boyu 8x8 birim alındı. Böylece piksel koordinatları tam sayı oldu. Eğer çözünürlük 640x480 olsaydı, 8x8 birimlik görüntü düzleminde herhangi bir (X,Y,5) noktasından geçen ışının piksel koordinatları (8*X/640 - 4, 4 – 8*Y/480, 5) olurdu. En basit ışın izleme uygulamalarında görüntü düzlemindeki her bir piksele yalnızca bir tane birincil ışın yollanır. Bu durumda her bir piksel için bir örnekleme yapıldığından üretilecek görüntüde bazı bozulmalar olabilmektedir. Özellikle doku kaplama yapılmışsa bu bozulmalar daha da artabilmektedir. Bozulmaların temel sebebi bir pikselden geçen ışının aynı anda nesnelerden iki veya daha çoğuyla birden kesişmesidir. Bu durumda gerçekte o pikselin rengi bu kesişim noktalarından hangisine göre belirlenecektir? İşte bu belirsiz durum görüntüde bazı hatalara neden olabilmektedir. Örnekleme hatasından kaynaklanan bu tür bozulmalara aliasing etkisi denir. Bu problemin çözümlenmesi için gelişmiş ışın izleme uygulamalarında bir piksele 1'den fazla ışın yollanır. Supersampling adı verilen yönteme göre bu ışınlar için hesaplanan renk değerlerinin ortalaması alınarak ilgili pikselin rengi belirlenir. Çeşitli supersampling teknikleri vardır. Doku Kaplama anlatılırken bu konuya daha fazla değinilecektir.

1.2.3. Görünmeyen Yüzeylerin Kaldırılması

Işın izleme yöntemi görünmeyen yüzeyleri kaldırırken diğer yöntemlerden farklı bir yol izler. Diğer yöntemler görünmeyen yüzeyleri kaldırırken poligonları esas alırlar. Nesneler çok sayıda poligonlara parçalanır. Genelde kullanılan poligon türü üçgenlerdir. Görünmeyen yüzeylerin kaldırılması poligonlar mertebesinde yapıldığından bu tür tekniklerle üretilen görüntülerde poligonların kenarları belli olmaktadır. Bu hatayı gidermek için poligonların piksellere sığacak kadar küçük seçilmesi gerekmektedir.

Işın izleme görünmeyen yüzeyleri kaldırmayı piksel mertebesinde gerçekler. Üstelik bütün pikseller birbirinden bağımsız olarak işlenir. Bakış noktasından yollanan ve her bir pikselden geçen ışınlar ile nesneleri oluşturan yüzeylerle kesişim testleri yapılır. Herhangi bir pikselden geçen ışın doğrultusu boyunca 1'den fazla kesişim söz konusu ise hesaplanan t uzaklık değerleri sıralanır. Nesnelerin geçirgenlik özellikleri de göz önüne alınarak en yakın yüzey bakış noktasından görünen yüzeydir. Diğer yüzeyler görünmeyen yüzeylerdir ve kaldırılır.

Işın izleme yöntemi görünmeyen yüzeyleri kaldırırken piksel mertebesinde hassas çalıştığından sanki yüzeyler piksel boyunda poligonlara bölünmüş gibi ele alınır. Böylece eğrisel yüzeylerde bile yüksel doğrulukta görüntüler elde edilir.

Görünmeyen yüzeylerin kaldırılmasıyla ilgili bir konu da Arka Yüz Kaldırma (Backface Culling)'dır. Herhangi bir bakış noktasından yollanan ışın doğrultusu boyunca kesişen yüzeylerin bazıları bakış noktasına bakmayabilir. Normalde bakış noktasından görünmesi imkansız olan yüzeyler için kesişim testi yapmak mantıksızdır. O nedenle kesişim testlerinin başında arka yüzler belirlenir ve kaldırılır. Böylece kesişim testleri hızlandırılır. Işın İzlemeyi Hızlandırma Teknikleri konusunda arka yüz kaldırma daha detaylı olarak anlatılacaktır.

1.3. Işın-Nesne Kesişim Testleri

Işın izlemenin en önemli aşaması ışın-nesne kesişim testleridir. Bakış noktasından belli bir doğru boyunca yollanan birincil ışının 3B nesnelerden hangileriyle kesiştiği belirlenmelidir. Kesişim testleri sonucu kesişim noktasının koordinatları ve kesişim noktasına uzaklık olan t değeri hesaplanır. Aynı doğrultu boyunca 1' den fazla kesişim varsa görünmeyen yüzeyleri kaldırmak için t değerleri kullanılır.

Işın izleme yöntemiyle görüntü üretiminde hesaplama süresinin çoğu kesişim testlerine harcanmaktadır. O nedenle kesişim testlerinin hızlı yapılması gerekmektedir. Test süresini azaltılmak için çeşitli algoritmalar geliştirilmiştir. Genellikle 3B nesneler üçgenlerden oluşturulduğu için ışın-üçgen kesişimi için birçok algoritma geliştirilmiştir. Bu bölümde de ağırlıklı olarak ışın-üçgen kesişim testleri anlatılacaktır.

1.3.1. Kesişim Testlerinde Kullanılan Lineer Cebir Temelleri

Işın izleme yönteminde vektörel bir büyüklük olan ışının davranışları matematiksel olarak incelendiğinden lineer cebirle ilgili bazı temel bilgilerin bilinmesi gerekmektedir. Bu bölümde anlatılan konular ışın izlemenin hemen hemen her aşamasında kullanılan temel bilgilerdir.

1.3.1.1. Skaler Çarpım

İki vektörün skaler çarpımı yapılırken karşılıklı olarak x, y ve z bileşenleri çarpılır ve bu çarpımlar toplanarak tek bir değer elde edilir. Skaler çarpım * sembolü ile gösterilecektir. R1 ve R2 vektörlerinin skaler çarpımı olan D 'nin nasıl hesaplandığı aşağıda verilmiştir:

$$D = \mathbf{R1} * \mathbf{R2} = R1_xR2_x + R1_yR2_y + R1_zR2_z \quad (1.4)$$

İki birim vektörün skaler çarpımı aralarındaki açının kosinüsünü verir. Bu gerçek difüz aydınlatmanın temelini oluşturmaktadır. Difüz aydınlatma anlatılırken bu konuda detaylı bilgi verilecektir.

1.3.1.2. Vektörel Çarpım

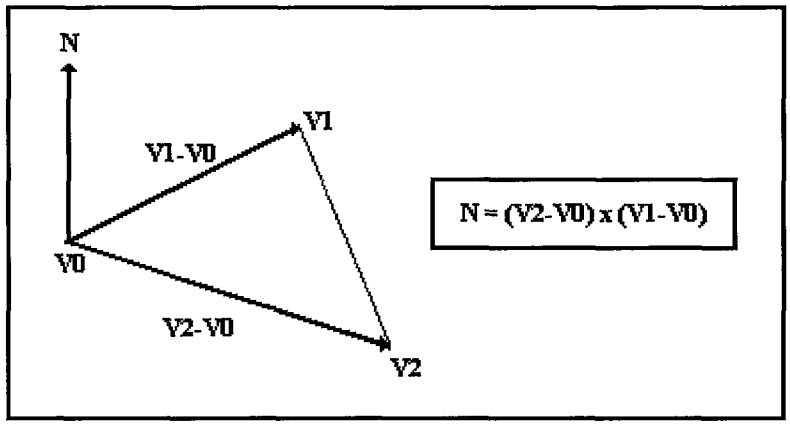
Vektörel çarpım 'x' sembolüyle gösterilirse **R1** ve **R2** vektörlerinin vektörel çarpımı olan **R** vektörü aşağıdaki gibi hesaplanır:

$$\mathbf{R} = \mathbf{R1} \times \mathbf{R2} = [\ R1_yR2_z - R1_zR2_y \quad R1_zR2_x - R1_xR2_z \quad R1_xR2_y - R1_yR2_x\ ] \quad (1.5)$$

Vektörel çarpım ışın izlemede çok kullanılır. Çünkü yüzey normali vektörel çarpımla hesaplanmaktadır. Yansıyan veya kırılan ışınların doğrultuları hesaplanırken yüzeyin normalini, yani yüzeye dik olan vektörü bilmek gerekir. Şekil 1.4.' te köşe noktaları $\mathbf{V_0}$, $\mathbf{V_1}$ ve $\mathbf{V_2}$ olan üçgenin yüzey normalinin nasıl hesaplandığı gösterilmiştir.

Difüz ve speküler aydınlatma yapılırken de yüzey normali kullanılır. Ayrıca arka yüz kaldırmada herhangi bir yüzeyin bakış noktasına göre arka yüz olup olmadığı belirlenirken de yüzey normali kullanılır.

Yüzey normali yukarıda saydığımız yerlerde kullanılabilmesi için ışının doğrultusu gibi birim vektör olmalıdır. O nedenle yüzey normali vektörel çarpımla hesaplandıktan sonra normalize edilmelidir.



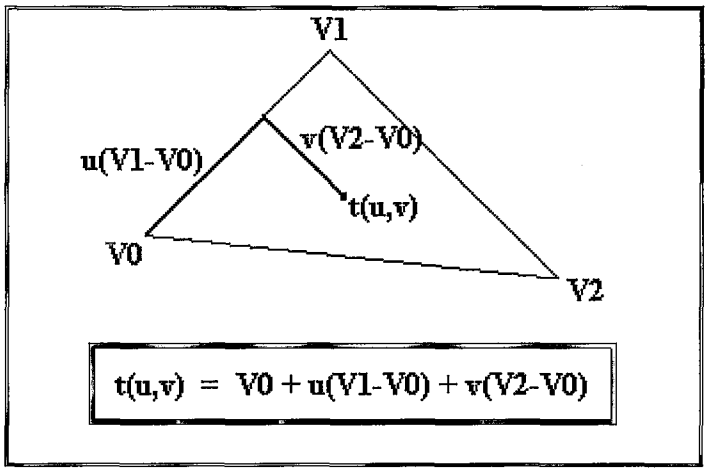
Şekil 1.4. Yüzey normalinin hesaplanışı

1.3.1.3. Barisentrik Koordinatlar (Barycentric Coordinates)

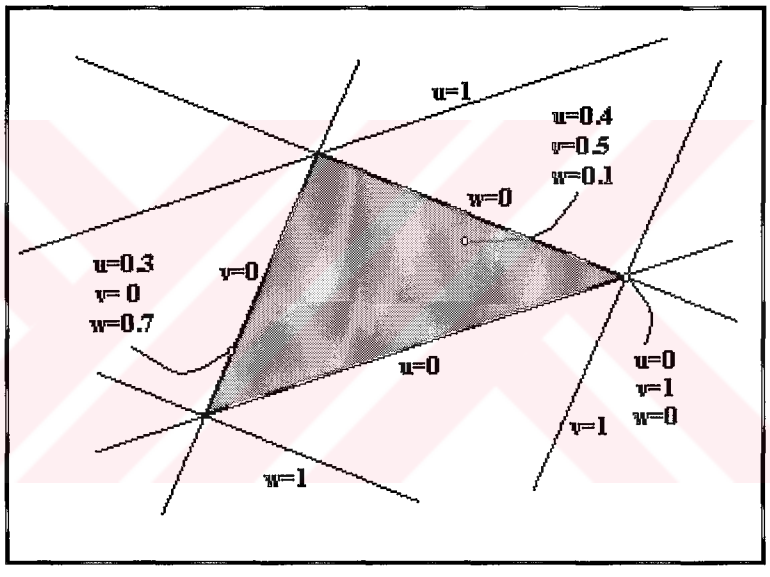
Üçgenin bir köşesinden başlayarak kenarları belli katsayılarla çarpılıp vektörel olarak bu köşeye eklendiğinde üçgen içinde istenilen noktaya gitmek mümkündür. Kenarları çarpmada kullanılan katsayılara barisentrik koordinatlar denir. Üçgen içindeki herhangi bir noktanın barisentrik koordinatlar cinsinden nasıl hesaplandığına dair vektörel ifade aşağıda verilmiştir [13]:

$$t(u,v) = \mathbf{V_0} + u(\mathbf{V_1}-\mathbf{V_0}) + v(\mathbf{V_2}-\mathbf{V_0}) \qquad (1.6)$$

Burada u ve v barisentrik koordinatlardır ve u ve v için $u \geq 0$, $v \geq 0$, $u+v \leq 1$ şartı sağlanmalıdır. (1.6) ifadesinden anlaşılacağı gibi üçgenin V_0 köşesinden başlanılarak $(V_1 - V_0)$ kenarı boyunca u kadar, $(V_2 - V_0)$ kenarı boyunca da v kadar gidildiğinde yani V_0 vektörüne önce $u(V_1 - V_0)$ sonra da $v(V_2 - V_0)$ vektörü eklendiğinde üçgen içinde istenilen herhangi bir noktaya gitmek mümkündür. Bu durum Şekil 1.5.'te vektörel olarak gösterilmiştir.



Şekil 1.5. Barisentrik koordinatlar



Şekil 1.6. Barisentrik koordinatların değişimi

(1.6) ifadesi yeniden düzenlenirse aşağıdaki hali alır:

$$t(u,v) = (1-u-v)\mathbf{V_0} + u\mathbf{V_1} + v\mathbf{V_2}$$

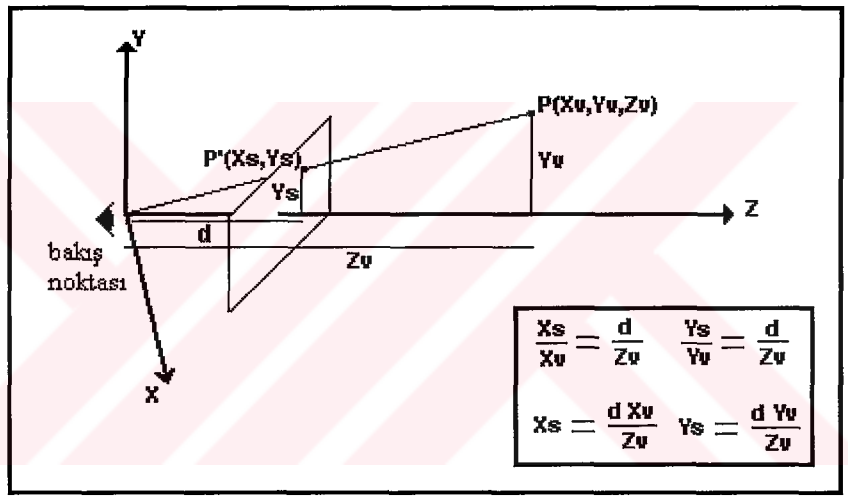
Bu ifadede $\mathbf{V_0}$ 'ın katsayısı olan (1-u-v), w ile gösterilirse üçüncü barisentrik koordinat da belirlenmiş olur. Barisentrik koordinatların üçgenin kenarları boyunca nasıl değiştiği Şekil 1.6.'da gösterilmiştir [13].

Barisentrik koordinatlar ışın-üçgen kesişim testi ve doku kaplamada kullanılmaktadır.

1.3.1.4. Perspektif İzdüşüm

3B nesnelerin 2B görüntü düzlemindeki koordinatlarının bulunması için perspektif izdüşümleri alınır.

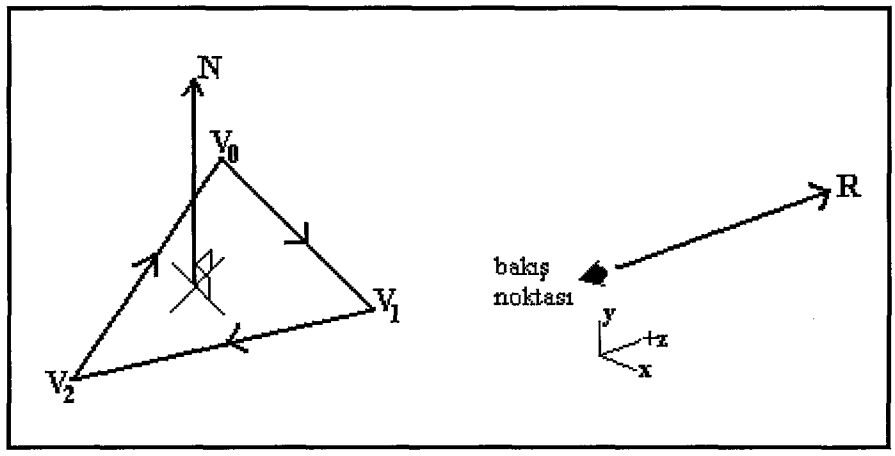
P(x_v, y_v, z_v) noktasının perspektif izdüşümü olan **P'**(x_s, y_s) hesaplanırken **P** noktasının z-bileşeninden faydalanılır. Benzer üçgenler özelliği kullanılarak y_s 'in nasıl hesaplanacağı Şekil 1.7.'de gösterilmiştir. x_s de benzer şekilde hesaplandığından ayrıca bir şekil çizmeye gerek duyulmamıştır. Şekilde bakış noktasının görüntü düzlemine uzaklığı d ile gösterilmiştir. Işın izlemede kesişim testleri yaparken perspektif izdüşümü kullanan yöntemler vardır.



Şekil 1.7. Perspektif izdüşüm

1.3.2. Işın-Poligon Kesişim Testleri

Üçgenlerden oluşturulan 3B nesneler için ışın-poligon kesişimi testi yapan algoritmaların çoğu öncelikle ışının, poligonun üzerinde bulunduğu yüzeyle kesişip kesişmediğini test eder. Eğer ışın yüzeyden geçiyorsa kesişim noktasının poligonun içinde olup olmadığı test edilir. O nedenle önce ışın-yüzey kesişim testi ve ardından da ışın-poligon kesişim testleri anlatılacaktır.



Şekil 1.8. Yüzey ifadesinin grafiksel gösterimi

1.3.2.1. Işın-Yüzey Kesişim Testi

(0,0,0) noktasından uzaklığı D olan $\mathbf{P_n}$ = [A B C] normaline sahip bir **P** yüzeyinin denklemi aşağıdaki gibi verilebilir:

$$Ax + By + Cz + D = 0 \qquad (1.7)$$

Şekil 1.8.'de yüzey ve onun normali gösterilmiştir. 3B uzayda bir yüzeyin tanımlanabilmesi için o yüzeyin üzerindeki en az 3 noktanın bilinmesi gerekir. Öyleyse bir üçgenin köşe noktaları biliniyorsa onun üzerinde oturduğu düzlemin denklemini belirlemek mümkündür. Köşelerinin koordinatları $\mathbf{V_0}$ = (1,2,4), $\mathbf{V_1}$ = (3,3,4), $\mathbf{V_2}$ = (4,1,4) olan üçgenin yüzey denkleminin nasıl hesaplandığı aşağıda gösterilmiştir:

Yüzey denkleminin hesaplanması (1.7)'de **P** yüzeyinin ifadesindeki A, B, C ve D katsayılarının bulunmasından ibarettir. Yüzey denklemindeki A, B ve C katsayıları yüzey normalinin x, y ve z 'sidir. Üçgenin köşe noktaları kullanılarak yüzey normalini hesaplamak mümkündür. Yüzey normali $\mathbf{P_n}$ = [A B C] ($\mathbf{V_1}$ - $\mathbf{V_0}$) ile ($\mathbf{V_2}$ – $\mathbf{V_0}$) 'ın vektörlerinin vektörel çarpımından $\mathbf{P_n}$ = [A B C] = [0 0 -5] olarak bulunur.

Üçgenin üç köşe noktası da yüzey denklemini sağlayacağından bunlardan V_0' ı kullanarak D katsayısı aşağıdaki gibi hesaplanabilir:

$$0*1 + 0*2 + (-5)*4 + D = 0$$

$$D = 20$$

Sonuç olarak düzlem ifadesi $-5z + 20 = 0$ ya da $z = 4$ olarak bulunur.

Işın ifadesinin $\mathbf{R} = \mathbf{R_0} + t\mathbf{R_d}$ olduğu biliniyor. Eğer ışın yüzeyle kesişiyorsa kesişim noktası için **R** ışınının x, y, z değerleri ile **P** yüzeyinin o noktadaki x, y, z değerleri aynı olmalıdır. Buradan şöyle yazılabilir:

$$A(X_0 + tX_d) + B(Y_0 + tY_d) + C(Z_0 + tZ_d) + D = 0 \qquad (1.8)$$

İfade t 'ye göre düzenlenirse başlangıç noktasıyla kesişim noktası arasındaki uzaklık değeri olan t için aşağıdaki ifade elde edilmiş olur:

$$t = -(AX_0 + BY_0 + CZ_0 + D) \;/\; (AX_d + BY_d + CZ_d)$$

t 'nin vektörel gösterimi şöyledir:

$$t = -(\mathbf{P_n} * \mathbf{R_0} + D) / (\mathbf{P_n} * \mathbf{R_d}) \qquad (1.9)$$

$\mathbf{P_n} * \mathbf{R_d} = 0$ ise ışın yüzeye paraleldir.

$t < 0$ ise kesişim noktası ışının başlangıç noktasından geridedir. Yani gerçekte görünen bir kesişim yoktur. $t > 0$ için kesişim vardır ve kesişim noktası $\mathbf{R_i}$ aşağıdaki gibi hesaplanır:

$$\mathbf{R_i} = [\, x_i \;\; y_i \;\; z_i \,] = [\, X_0 + tX_d \quad Y_0 + tY_d \quad Z_0 + tZ_d \,]$$

Işın-yüzey kesişimi için sayısal bir örnek aşağıda verilmiştir:

Kesişim testi yapılacak yüzey $\mathbf{P} = [\, 1\ 0\ 0\ -7]$ olsun. ($x = 7$ yüzeyi)
Işının başlangıç noktası $\mathbf{R_0} = [\, 2\ 3\ 4\,]$ olsun.
Işının doğrultusu $\mathbf{R_d} = [\, 0.577\ \ 0.577\ \ 0.577\ \,]$ olsun.

$$t = 5 \;/\; 0.577 = 8.66$$

t değeri 0 'dan büyük olduğu için kesişim vardır. Işının yüzey üzerindeki koordinatları aşağıdaki gibi hesaplanır :

$x_i = 2 + 0.577 * 8.66 = 7$

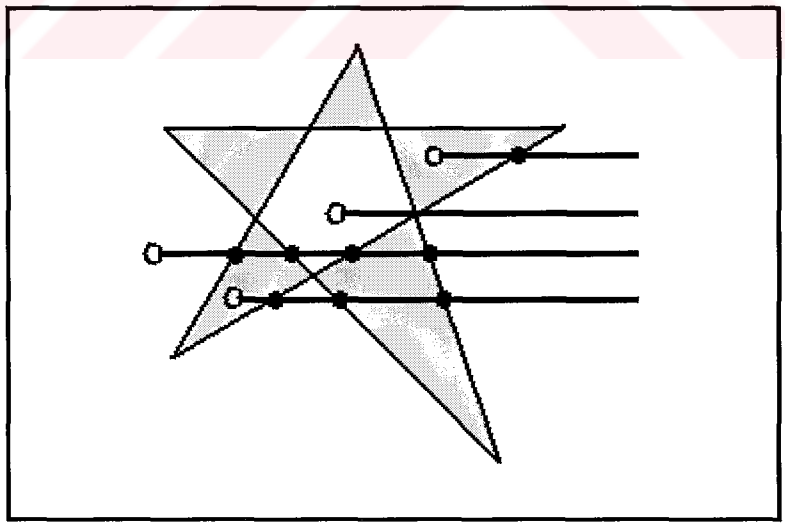
$y_i = 3 + 0.577 * 8.66 = 8$

$z_i = 4 + 0.577 * 8.66 = 9$

$\mathbf{R_i}$ = [7 8 9] olarak bulunur.

1.3.2.2. Jordan Curve Teoremi ile Işın-Poligon Kesişim Testi

Işın-poligon kesişim testinde yaygın olarak kullanılan yöntem Jordan Curve teoremidir. Bu yönteme göre öncelikle poligonun (burada üçgenin) görüntü düzlemine perspektif izdüşümü alınır. Işının görüntü düzlemini kestiği noktadan itibaren x veya y ekseni boyunca bir doğru çizilir. Bu doğru ile poligonun kenarları arasında kesişim testi yapılır. Kesişim sayısına göre ışının poligonun içinde olup olmadığı belirlenir. Eğer ışın poligonun içinde ise kesişim sayısı tektir. Eğer dışında ise kesişim sayısı çifttir. Şekil 1.9.'da Jordan Curve teoremine göre kesişim testinin nasıl yapıldığı gösterilmiştir [7].



Şekil 1.9. Jordan curve teoremi

Poligonun üzerinde bulunduğu yüzey üzerine perspektif izdüşümü alınmış olsun. Yeni eksenleri U ve V ile gösterelim. R ışının izdüşümü de alınmış olsun. Öncelikle 2B köşe noktalarının koordinatlarından izdüşümü alınmış R ışının koordinatları çıkarılır. Böylece R ışını yeni eksenlerin merkezine çekilir. Bundan sonra U veya V ekseni boyunca yollanan doğrular ile kenarlar arasında kesişim testleri yapılır. Algoritma aşağıdadır [7] :

Kesişimlerin sayısını tutan NC 'yi sıfırla

Birinci kenarın ilk köşe noktasının V bileşeni V_0 'ın değerine göre SH 'yi şöyle setle:

Eğer $V_0 < 0$ ise SH = -1

Eğer $V_0 >= 0$ ise SH = +1

Poligondaki (U_a,V_a) ile (U_b,V_b) noktaları arasındaki bütün kenarlar için aşağıdakileri tekrarla (burada kenar sayısı NV için a =0..(NV-1) b =(a+1) mod NV):

NSH 'ı V_b 'ye göre şöyle setle:

Eğer $V_b < 0$ ise NSH = -1

Eğer $V_b >= 0$ ise NSH = +1

Eğer SH NSH 'a eşit değilse:

Eğer $U_a>0$ ve $U_b>0$ ise kesişim var ve NC = NC + 1

Eğer $U_a>0$ veya $U_b>0$ ise :

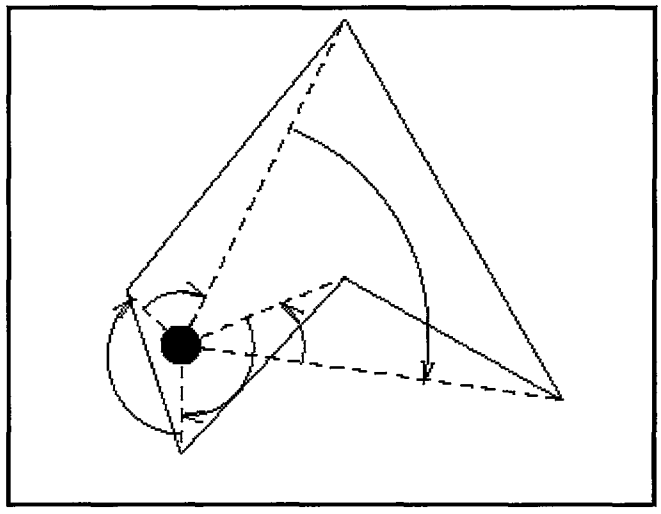
$U_a - V_a * (U_b - U_a) / (V_b - V_a) > 0$ ise NC = NC + 1

SH = NSH

Eğer NC tek ise ışın poligonun içinde, çift ise dışında.

1.3.2.3. Açı Hesabıyla Işın-Poligon Kesişim Testi

Açı hesabıyla ışın-poligon kesişim testinde önce ışının, poligonun üzerinde bulunduğu yüzeyle kesişim noktası bulunur. Bu kesişim noktasından poligonun köşelerine doğrular çizilerek alt üçgenler oluşturulur. Bu alt üçgenlerde kesişim noktasının olduğu köşelerdeki açılar hesaplanır ve toplanır. Toplam değer 0 veya 360 'ın katları ise kesişim noktası üçgenin içinde aksi halde dışındadır. Şekil 1.10.'da açı hesabıyla kesişim testinin nasıl yapıldığı gösterilmiştir.



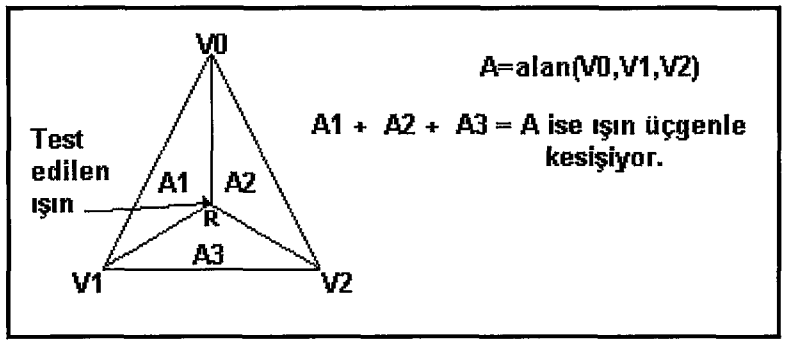
Şekil 1.10. Açı hesabıyla ışın-poligon kesişim testi

1.3.3. Işın-Üçgen Kesişim Testleri

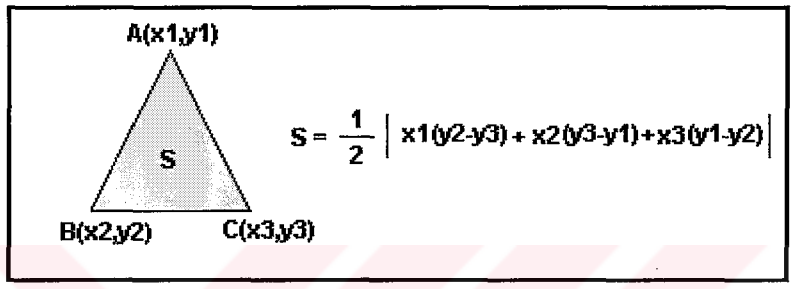
3B nesneler bilgisayarda modellenirken genellikle üçgenler kullanılır. O nedenle kesişim testi yapan algoritmaların çoğu üçgenler üzerine yoğunlaşmıştır. Üçgen de bir poligon olduğuna göre buraya kadar anlatılan ışın-poligon kesişim testlerini ışın-üçgen kesişim testi için de kullanmak mümkündür. Bu noktadan sonra anlatılacak yöntemler daha ziyade ışın-üçgen kesişimine özgü yöntemler olacaktır.

1.3.3.1 Alan Hesabıyla Işın-Üçgen Kesişim Testi

Işın-Üçgen kesişim testleri içinde en basit yöntem alan testidir. Bu yönteme göre önce ışın-yüzey kesişim testi yapılır. Ardından ışın ve yüzeyi oluşturan üçgenin köşe noktalarının görüntü düzlemine perspektif izdüşümleri alınır. Yüzeyle kesişen ışının üçgenin içinde olup olmadığının belirlenmesi için kesişim noktasıyla üçgenin köşelerinden ikişer tane alınarak 3 tane alt üçgen belirlenir ve bunların alanları hesaplanır. Hesaplanan bu alan değerleri toplanır. Toplam alan değeri kesişim testi yapılan üçgenin alanına eşit oluyorsa ışın üçgenin içindedir. Aksi halde dışındadır. Şekil 1.11.'de alan testinin nasıl yapıldığı gösterilmektedir.



Şekil 1.11. Alan testi



Şekil 1.12. Alan hesabı

Köşe noktaları A, B ve C olan üçgenin alanının nasıl hesaplandığı Şekil 1.12.'de gösterilmiştir. Formülden de anlaşılacağı gibi alan testinin yapılabilmesi için 3B'den 2B'ye dönüşüm yapmak gerekmektedir. Bunun için üçgenin köşe noktalarının ve kesişim noktasının perspektif izdüşümleri alınır.

1.3.3.2. Möller ve Trumbore 'un Hızlı Işın-Üçgen Kesişim Testi Yöntemi

Şimdiye kadar anlatılan kesişim yöntemlerinin hepsinin ortak bir noktası vardı. Yöntemlerin hepsinde kesişim testi yapmadan önce ışının ve üçgenin köşe noktalarının, üçgenin üzerinde olduğu yüzeyle kesişim noktası hesaplanıp köşe noktalarıyla bu kesişim noktasının xy eksenine (yz veya xz de olabilir) perspektif izdüşümü alınıyordu. Böylece problem 3B' den 2B 'ye indirgeniyordu. Kesişim testi olarak da sadece 2B noktanın yine 2B üçgenin içinde olup olmadığını belirlemek yeterliydi. Yalnız bütün bu işlemler için öncelikle yüzey denkleminin belirlenmesi bunun için de yüzey normalinin hesaplanması

gerekiyordu. Böylece ışının yüzey üzerindeki kesişim noktası bulunduktan sonra 3B 'den 2B 'ye dönüşüm için perspektif izdüşüm yapılıyordu. Kesişim testi öncesi yapılan bu işlemler algoritmaların yavaş çalışmasına neden olmaktadır.

Möller ve Trumbore tarafından geliştirilen yönteme göre yüzey normalinin hesaplanmasına gerek kalmamaktadır [14].

1.3.3.2.1. Kesişim Algoritması

O başlangıç noktasına sahip normalize edilmiş **D** doğrultusu boyunca giden **R** ışın ifadesi aşağıdaki gibidir:

$$\mathbf{R} = \mathbf{O} + t\mathbf{D} \tag{1.10}$$

Kesişim testi yapılacak üçgenin köşe noktaları V_0, V_1 ve V_2 olsun. Üçgen içindeki herhangi bir nokta u ve v parametrelerini kullanarak üçgenin köşe noktaları cinsinden ifade edilebilir.

Üçgen üzerindeki **t**(u,v) noktasının barisentrik koordinatlar cinsinden ifadesi aşağıdaki gibiydi :

$$\mathbf{t}(u,v) = (1-u-v)\mathbf{V_0} + u\mathbf{V_1} + v\mathbf{V_2} \tag{1.11}$$

R ışını ile üçgeni **t**(u,v) noktasında kesiyorsa kesişim noktası için **R** = **t**(u,v) olmalıdır. Buradan aşağıdaki ifade yazılabilir:

$$\mathbf{O} + t\mathbf{D} = (1-u-v)\mathbf{V_0} + u\mathbf{V_1} + v\mathbf{V_2} \tag{1.12}$$

Terimler yeniden düzenlenir aşağıdaki ifade elde edilir:

$$[\,-\mathbf{D} \quad \mathbf{V_1}-\mathbf{V_0} \quad \mathbf{V_2}-\mathbf{V_0}\,] \begin{bmatrix} t \\ u \\ v \end{bmatrix} = \mathbf{O} - \mathbf{V_0} \tag{1.13}$$

(1.13) ifadesindeki lineer denklem sistemi çözülürse barisentrik koordinatlar (u,v), ışının başlangıç noktasından kesişim noktasına olan uzaklığı olan t bulunur. Bu denklem geometrik olarak yorumlanırsa üçgenin orjine ötelenip ardından yz-ekseninde birim üçgene dönüştürüldüğü görülür. Bu durum Şekil 1.13.'te gösterilmiştir. (1.13) ifadesi için $\mathbf{M} = [\ -\mathbf{D}\ \ \mathbf{V_1}-\mathbf{V_0}\ \ \mathbf{V_2}-\mathbf{V_0}\ ]$ kısaltması yapılırsa çözümün $\mathbf{M}^{-1}$ ile denklemi çarpmak olduğu aşikardır.

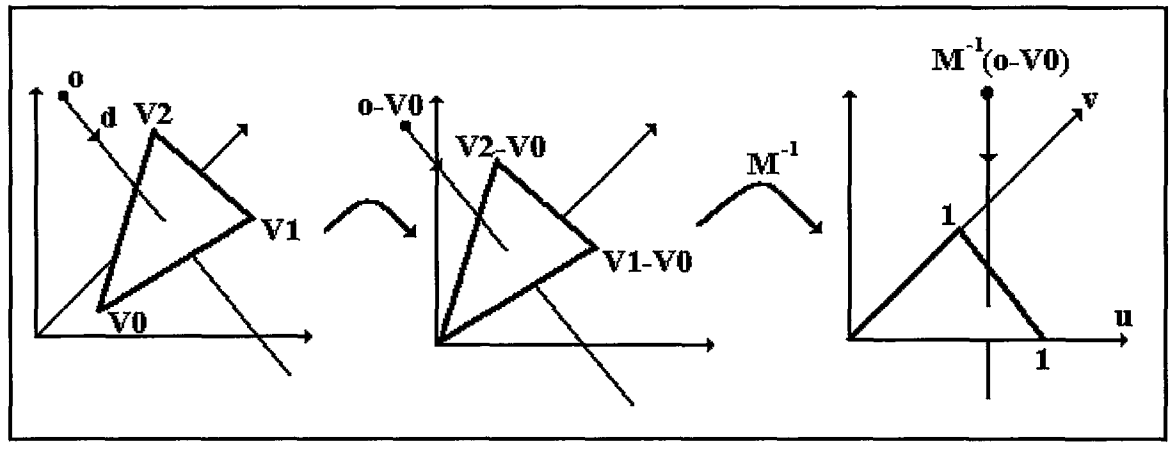
$\mathbf{E_1} = \mathbf{V_1}-\mathbf{V_0}$, $\mathbf{E_2} = \mathbf{V_2}-\mathbf{V_0}$ ve $\mathbf{S} = \mathbf{O} - \mathbf{V_0}$ kısaltmaları yapılıp denklem Cramer kuralına göre çözülürse aşağıdaki eşitlik elde edilir:

$$\begin{bmatrix} t \\ u \\ v \end{bmatrix} = \frac{1}{\det(-D, E1, E2)} \begin{bmatrix} \det(S, E1, E2) \\ \det(-D, S, E2) \\ \det(-D, E1, S) \end{bmatrix} \tag{1.14}$$

$\det(\mathbf{A},\mathbf{B},\mathbf{C}) = |\mathbf{A}\ \mathbf{B}\ \mathbf{C}| = -(\ \mathbf{A} \times \mathbf{C}\ )*\mathbf{B} = -(\ \mathbf{C} \times \mathbf{B})*\mathbf{A}$ olduğunu lineer cebirden biliniyor. (13) ifadesinde bunlar yerine yazılırsa aşağıdaki ifade elde edilir:

$$\begin{bmatrix} t \\ u \\ v \end{bmatrix} = \frac{1}{(D \times E2)*E1} \begin{bmatrix} (S \times E1)*E2 \\ (D \times E2)*S \\ (S \times E1)*D \end{bmatrix} = \frac{1}{P*E1} \begin{bmatrix} Q*E2 \\ P*S \\ Q*D \end{bmatrix} \tag{1.15}$$

(1.15) ifadesinde $\mathbf{P} = \mathbf{D} \times \mathbf{E_2}$ ve $\mathbf{Q} = \mathbf{S} \times \mathbf{E_1}$ kısaltması yapılmıştır. Bu kısaltmalar hesaplama zamanının azalmasında rol oynamaktadır.



Şekil 1.13. (1.13) ifadesinin geometrik gösterimi

Görüldüğü gibi bu yönteme göre kesişim testinde yüzey normali ve perspektif izdüşüm hesabına gerek yoktur. Tomas Möller ve Eric Haines tarafından yazılan ve ikinci baskısı 2002 yılında yapılan Real-Time Rendering kitabında bu yöntemin en hızlı ışın-üçgen kesişim testi yöntemi olduğu yazılmaktadır.

1.3.3.2.2. Uygulama

Bu yöntem için özet bir yalancı dil kodu (pseudocode) aşağıda verilmiştir [13].

Işın_Üçgen_Kesişimi($\mathbf{o}, \mathbf{d}, \mathbf{v_0}, \mathbf{v_1}, \mathbf{v_2}$)

1: $\mathbf{e_1} = \mathbf{v_1} - \mathbf{v_0}$

2: $\mathbf{e_2} = \mathbf{v_2} - \mathbf{v_0}$

3: $\mathbf{p} = \mathbf{d} \times \mathbf{e_2}$

4: $\alpha = \mathbf{e_1} * \mathbf{p}$

5: if ($\alpha > -\varepsilon$ and $\alpha < \varepsilon$) return (REJECT,0,0,0);

6: $f = 1/\alpha$

7: $\mathbf{s} = \mathbf{o} - \mathbf{v_0}$

8: $u = f(\mathbf{s}*\mathbf{p})$

9: if (u<0.0 or u>1.0) return (REJECT,0,0,0);

10: $\mathbf{q} = \mathbf{s} \times \mathbf{e_1}$

11: $v = f(\mathbf{d}*\mathbf{q})$

12: if (v<0.0 or u+v>1.0) return (REJECT,0,0,0)

13: $t = f(\mathbf{e_2}*\mathbf{q})$

14: return (INTERSECT, u, v, t)

Eğer kesişim varsa kod (INTERSECT, u, v, t) yoksa (REJECT,0,0,0) döndürür. Görüldüğü gibi algoritma yalnızca kesişim olup olmadığını değil aynı zamanda kesişim noktasına uzaklık olan t değerini, barisentrik koordinatlar olan u, v 'yi bulmaktadır. Kodda 4. satırda α değeri hesaplanmaktadır. Bu değer $\mathbf{M} = [\ -\mathbf{D} \quad \mathbf{V_1}-\mathbf{V_0} \quad \mathbf{V_2}-\mathbf{V_0}\ ]$ matrisinin determinantıdır. Sonraki satırda determinantın sıfıra yakın olup olmadığı test edilmektedir. Belli bir ε değerinden küçükse sıfır olduğu varsayılmaktadır. Kayan noktalı (float) duyarlılıkta $\varepsilon = 10^{-5}$ alınırsa güzel sonuçlar elde edildiği görülmüştür. 9. satırda u değeri test edilmektedir. 0 'dan küçük ve 1' den u değeri için ışın üçgenle kesişmiyor demektir. Benzeri şekilde 12. satırda da önce $v<0$ sonra da $u+v>1$ testi yapılmaktadır. Testin ikinci kısmında v değil de u+v 'nin test edilme sebebi ilginçtir. 12. satıra gelindiğinde u' nun 0,1 aralığında olduğu anlaşılmıştır. Burada $v>1$ testi yapıp tekrar $u+v>1$ testi yerine yalnızca $u+v>1$ testi yapmak yeterlidir.

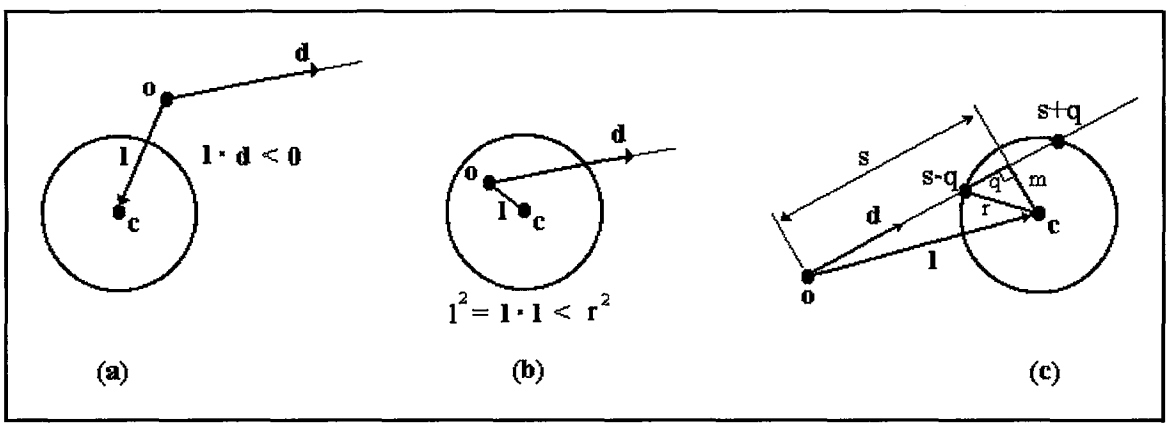
Koda dikkat edilirse gerekmedikçe bütün hesaplamalar geciktirilmiştir. Örneğin u değerinin olması gereken aralıkta olup olmadığı belirlenmeden v değeri hesaplanmamıştır.

1.3.4. Işın-Küre Kesişim Testi

Işın-küre kesişim testi, kesişim testleri içinde en basit olanıdır. Burada optimize edilmiş bir çözüm anlatılacaktır. Kesişim testi yaparken üç durum göz önüne alınmalıdır.

1. Kesişim noktası ışının başlangıç noktasının (**o**) gerisindedir. (Şekil 1.14.(a))
2. Kesişim noktası kürenin içindedir. (Şekil 1.14.(b))
3. Kesişim noktası ışının başlangıç noktasının (**o**) ilerisindedir. (Şekil 1.14.(c))

Bu durumlar Şekil 1.14 (a), (b) ve (c)' de gösterilmiştir [4]. Durumlardan yalnız 3. durumda gerçekten kesişim vardır ve ışın küreye teğet geçmiyorsa küreyi iki noktada keser.



Şekil 1.14. Işın-küre kesişim testi

Keşim testine 1. ve 2. durumun testiyle başlanır. Bunun için de ışının başlangıç noktasından kürenin merkezine olan $\mathbf{l} = \mathbf{c} - \mathbf{o}$ vektörü hesaplanır. Ayrıca ikinci durum için de **l** vektörünün uzunluğunun karesi olan $l^2 = \mathbf{l} * \mathbf{l}$ hesaplanır. (* skaler çarpımı gösterir) Eğer $l^2 < r^2$ ise ışının başlangıç noktası kürenin içindedir. Ardından **l**' nin ışının doğrultusu olan **d** üzerine izdüşümü şu şekilde hesaplanır: $s = \mathbf{l} * \mathbf{d}$. Eğer $l^2 > r^2$ ve $s < 0$ ise Şekil 1.14 (a)' daki durum söz konusudur. Yani ışının başlangıç noktası kürenin gerisindedir ve gerçekte bir kesişim yoktur. Diğer durumda pisagor teoremiyle kürenin merkezinin s' ye dik uzaklığı şu şekilde hesaplanır: $m^2 = l^2 - s^2$. Eğer $m^2 > r^2$ ise ışın kesinlikle küreyi kesmiyor demektir.

$m^2 <= r^2$ için kesişim vardır ve $q^2 = r^2 - m^2$ uzaklığı hesaplanır. $m^2 <= r^2$ olduğundan $q^2 >= 0$' dır. O nedenle kare körü hesaplanabilir. Sonuçta kesişim noktalarına uzaklıklar $t = s \pm q$ ile hesaplanır. Eğer ilk pozitif kesişim noktası isteniyorsa ışının başlangıç noktası kürenin dışında ise sadece $t_1 = s - q$; içinde ise de $t_2 = s + q$ hesaplanır. Gerçek kesişim noktasını bulmak için hesaplanan t değeri/değerleri ışın ifadesinde yerine konmalıdır.

Kesişim testi için yalancı dille yazılmış bir kod aşağıda verilmiştir [4]:

Işın_Küre_Kesişimi(**o**, **d**, **c**, r)

1: $\mathbf{l} = \mathbf{c} - \mathbf{o}$

2: $s = \mathbf{l} * \mathbf{d}$

3: $l^2 = \mathbf{l} * \mathbf{l}$

4: if ($s < 0$ and $l^2 > r^2$) return (REJECT,0,**0**)

5: $m^2 = l^2 - s^2$

6: if ($m^2 > r^2$) return (REJECT,0,**0**)

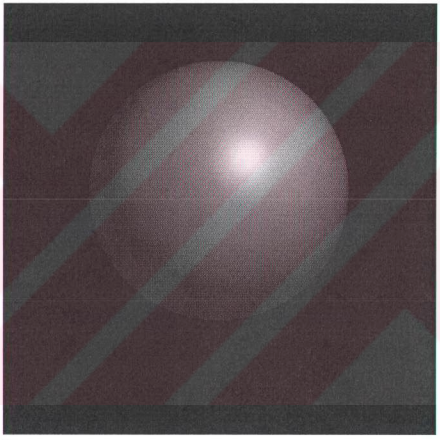
7: $q = sqrt(r^2 - m^2)$

8: if ($l^2 > r^2$) $t = s - q$

9: else $t = s + q$

10: return (INTERSECT, t, **o** + t**d**)

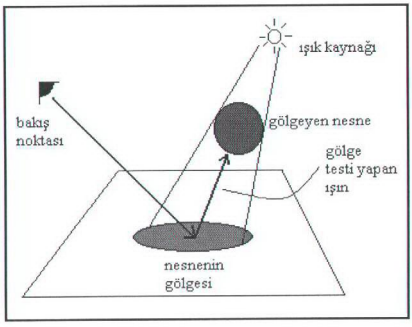
Şekil 1.15.'te ışın-küre kesişim testi sonucu ile üretilmiş bir görüntü vardır. Küreye ait piksellerin RGB (Red, Green, Blue) renk değerleri bulunurken ambient, diffuse ve specular aydınlatma bileşenleri kullanılmıştır. Aydınlatma modeli anlatılırken bu bileşenler hakkında detaylı bilgi verilecektir.



Şekil 1.15. Işın izleme ile üretilmiş bir küre

1.4. Gölgeler (Shadows)

Herhangi bir yüzey ile ışık kaynağı arasında kalan nesne bu yüzeye ışın gelmesini engelliyorsa bu yüzey o nesnenin gölgesinde kalıyor demektir. Yüzeyin gölgede olup olmadığını belirlemek için kesişim noktasından ışık kaynağına gölge test etme ışını yollanır. Yollanan ışın ışık kaynağına ulaşmadan başka bir nesne ile kesişiyorsa yüzey bu nesnenin gölgesinde kalıyor demektir. Işın izleme ile gölge testinin nasıl yapıldığı Şekil 1.16'da gösterilmiştir.



Şekil 1.16. Işın izleme ile gölge testi

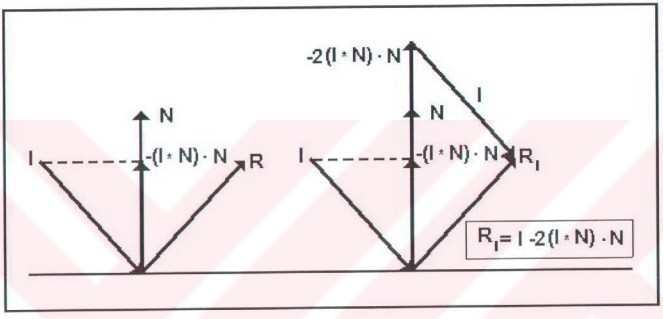
Gölge testi için yollanan ışın herhangi bir nesne ile kesiştiğinde bu nesnenin o yüzeyi gölgede bırakabilmesi için nesnenin ışık kaynağı ile yüzey arasında olması gerekir. Yani nesnenin yüzeye olan uzaklığının ışık kaynağının yüzeye olan uzaklığından küçük olması gerekir.

Gölge testi için yollanın ışın klasik ışın izleme uygulamalarında her bir ışık kaynağına birer tanedir. Bu durumda gölgelerin kenarlarında kırılmalar olabilmektedir. Gölgelerin kenarlarını yumuşatmak için gelişmiş ışın izleme uygulamalarında yüzeyden ışık kaynağına 1'den fazla ışın yollanır [19].

1.5. Aynasal Yansıma

Işın izlemenin belki de en önemli özelliği yansıyan ve kırılan ışınların mükemmel modellemesi ve böylece fiziksel gerçeklere uygun görüntüler üretilebilmesidir. Işın, yansıtma özelliğine sahip yüzeyden yansıyıp başka bir yüzeye çarptığında bu yüzeyin görüntüsü birinci yüzeyin üzerinde görülür. Işın izleme ile bunu modellemek oldukça kolaydır. Bunun için yansıyan ışının doğrultusunu bulmak yeterlidir. Nasıl ışın izlemenin başlangıcında yollanan birincil ışınlarla nesneler arasında kesişim testleri yapılıyorsa, yansıyan ışınlar için de aynı işlemler yapılacaktır. Farklı olan noktalar yansıyan ışınının başlangıç noktası ve doğrultusudur. Yeni başlangıç noktası ışının yansıtıcı özelliğe sahip yüzeyle kesiştiği noktadır. Yeni doğrultu ise yansıma doğrultusudur. Aynasal yansıtıcılar için yansıma doğrultusunun nasıl hesaplandığı Şekil 1.17'de gösterilmiştir.

Şekilde 1.17'de gelen ışının doğrultusu **I**, yüzey normali **N** kullanılarak aynasal yansıyan ışının doğrultusu $\mathbf{R}_r$ 'nın nasıl hesaplandığı gösterilmiştir. Burada **I** ve **N** 'nin normalize edilmiş olduğu varsayılmıştır. **–(I*N)N** ifadesi gelen ışın **I**' nın tersi olan **–I**'nın yüzey normaliyle skaler çarpımının tekrar yüzey normaliyle çarpımını yani **-I** ışının yüzey normali üzerine izdüşümünü verir. Ardından ikinci şekilden de görüldüğü gibi bu ifadenin 2 katı alınıp **I** ışınıyla toplandığında yansıyan ışın olan $\mathbf{R}_r$ nın, **I** ışını ve yüzey normali cinsinden ifadesi elde edilmiş olur.

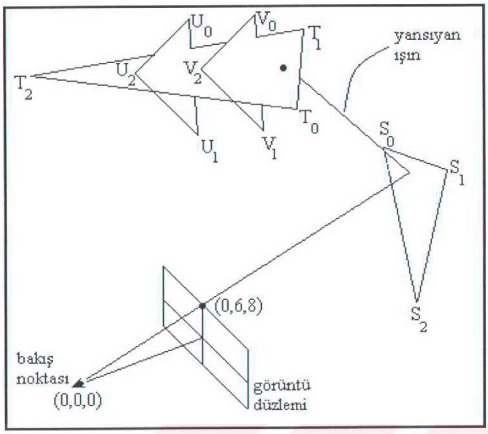


Şekil 1.17. Yansıyan ışının doğrultusu R_r 'nın hesaplanması

Işın izlemenin yansıma sonucu görünen nesneleri nasıl belirlediğinin daha iyi anlaşılması için aşağıda örnek bir soru ve çözümü verilmiştir:

Şekil 1.18.'den de görüldüğü gibi bakış noktası olan (0,0,0) 'dan çıkan ve görüntü düzleminde (0,6,8) noktasından geçen **R** ışını S üçgeninden aynasal olarak yansıyarak T, U ve V üçgenleriyle kesişmektedir. Böylece kesişim noktası için yansıma ile S üçgeni üzerinde T, U veya V üçgenlerinden birinin görüntüsü görülmektedir. Işın izleme yöntemiyle görünmeyen yüzeyleri kaldırarak bu üçgenin hangi üçgen olduğunu bulunuz. Işın ile üçgenlerin üzerinde bulunduğu yüzey arasındaki kesişim noktaları aynı zamanda üçgenlerin içindedir. Üçgenlerin köşe noktalarının koordinatları aşağıda verilmiştir.

$S_0 = (-20, 40, 40)$	$T_0 = (40, 51, 20)$	$U_0 = (-40, 80, -10)$	$V_0 =(-40, 80, 10)$
$S_1 = (20, 40, 40)$	$T_1 = (-40, 51, 20)$	$U_1 = (0, 30, -10)$	$V_1 =(0, 30, 10)$
$S_2 = (0, 0, 40)$	$T_2 = (0, 51,-60)$	$U_2 = (40, 80, -10)$	$V_2 =(40, 80, 10)$



Şekil 1.18. Yansıma örneği

Örnek soruda ışın ile üçgenlerin üzerinde bulunduğu yüzey arasındaki kesişim noktaları aynı zamanda üçgenlerin içinde olduğu söylendiği için ışın-üçgen kesişim noktasının koordinatlarının hesaplarken yalnızca ışın-üçgen üzerindeki yüzey kesişim testi yapmak yeterlidir.

Sorunun çözümü için yapılması gereken işlemler şunlardır:

1. Birincil ışının doğrultusunu bul.
2. Yansıtıcı üçgen üzerindeki kesişim noktasının koordinatlarını hesapla.
3. Yansıma doğrultusunu bul.
4. Başlangıç noktası ve doğrultusu hesaplanmış yansıyan ışının T, U ve V üçgenlerine uzaklıkları olan t_T, t_U ve t_V 'yi hesapla.
5. Hesaplanan t değerlerini sırala. En küçük t değerine sahip üçgen yansımayla görünen üçgendir.

1. Işının doğrultusu $\mathbf{R_d}$ için $\mathbf{R_d} = (\mathbf{R} - \mathbf{R_0}) / t$ olduğu bilinmektedir. Burada $\mathbf{R} = (0, 6, 8)$, $\mathbf{R_0} = (0, 0, 0)$ ve $t = 10$ 'dur. Bu değerler $\mathbf{R_d}$ 'nin ifadesinde yerine konursa $\mathbf{R_d} = (0, 0.6, 0.8)$ olarak bulunur.

2. Işın-yüzey kesişim testi yapmak için öncelikle yüzey denkleminin belirlenmesi gerekir. Bilindiği yüzey denkleminin A, B ve C katsayıları aynı zamanda

yüzey normalinin x, y ve z bileşenleridir. O halde yansıtıcı üçgen S 'nin köşe noktalarının koordinatları kullanılarak yüzey normali aşağıdaki gibi hesaplanabilir:

$(S_1 - S_0) = (40, 0, 0)$

$(S_2 - S_0) = (20, -40, 0)$

$P_S = (S_1 - S_0) \times (S_2 - S_0) = (0, 0, -1600)$

Yüzey normali ve üçgenin köşelerinden biri kullanılarak $D_S = 64000$ olarak bulunur.

$t = -(P_S * R_0 + D) / (P_S * R_d)$ olduğu bilinmektedir. Buradan $t_S = 50$ olarak bulunur.

Hesaplanan t değeri kullanılarak kesişim noktasının koordinatları aşağıdaki hesaplanır:

$K_S = (0,0,0) + 50(0, 0.6, 0.8) = (0, 30, 40)$.

3. Aynasal yansıma için yansıma doğrultusu aşağıdaki gibi hesaplanıyordu:

$R_y = R_d - 2(N * R_d)N$. Buradan

$R_y = (0, 0.6, 0.8) - 2(-0.8)(0, 0, -1) = (0, 0.6, -0.8)$ olarak bulunur.

4. Yansıyan ışının başlangıç noktası 2. adımda hesaplanan kesişim noktası, doğrultusu da 3. adımda hesaplanan doğrultudur. Bu bilgiler ve T. U ve V üçgenlerinin köşe noktalarının koordinatları kullanılarak t_T, t_U ve t_V değerleri aşağıdaki gibi hesaplanır.

$(T_1 - T_0) = (-80, 0, 0)$

$(T_2 - T_0) = (-40, 0, -80)$

$P_T = (T_1 - T_0) \times (T_2 - T_0) = (0, -6400, 0)$ ve $D_T = 326400$.

$(U_1 - U_0) = (40, -50, 0)$

$(U_2 - U_0) = (80, 0, 0)$

$P_U = (U_1 - U_0) \times (U_2 - U_0) = (0, 0, 4000)$ ve $D_U = 40000$.

$(\mathbf{V_1 - V_0}) = (40, -50, 0)$

$(\mathbf{V_2 - V_0}) = (80, 0, 0)$

$\mathbf{P_V} = (\mathbf{V_1 - V_0}) \times (\mathbf{V_2 - V_0}) = (0, 0, 4000)$ ve $D_U = -40000$.

$t_T = -(-6400*30 + 326400) / (-3840) = 35$

$t_U = -(160000+40000) / (-3200) = 62.5$

$t_V = -(160000-40000) / (-3200) = 37.5$

5. t değerleri sıralanırsa aşağıdaki sonuç elde edilir:

$35 < 37.5 < 62.5$ yani $t_T < t_V < t_U$. Bu sonuçtan anlaşılıyor ki bakış noktası olan (0,0,0) 'dan çıkan ve görüntü düzleminde (0,6,8) noktasından geçen **R** ışını S üçgeninden aynasal olarak yansıyınca ilk olarak T üçgeniyle kesişmekte yani yansıma noktasına T üçgeninin görüntüsü düşmektedir.

Şekil 1.19.'te bu örnekteki üçgenler kullanılarak elde dilen görüntü verilmektedir. Şekilde T üçgeni sarı, U mavi ve V de kahve rengidir. Dikkat edilirse yukarıda yapılan örnek için elde edilen görüntüdeki pikselin rengini sarı olduğu görülmektedir.

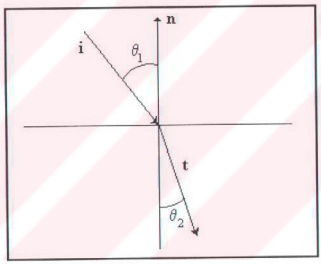
Şekil 1.19. Yansıma örneği için üretilen görüntü

1.6. Geçirgenlik ve Kırılma (Transparency & Refraction)

Yansımada olduğu gibi Geçirgenlik (Transparency) ve Kırılmada (Refraction) da ışın izleme oldukça başarılı sonuçlar vermektedir. Işının yansımasında olduğu gibi geçirgen nesnelerden geçmesi veya kırılması sonucu da yeni doğrultusu hesaplanır. Yeni doğrultu boyunca kesişim testleri tekrarlanır ve çarptığı yüzeylerin özelliklerine bağlı olarak yeniden yüzeyden yansır, yüzeyden geçer veya kırılır.

Geçirgenlik denince ışının kırılmadığı anlaşılmalıdır. Cam gibi ışığı kırmadan geçiren yüzeyleri modellemek oldukça kolaydır. Çünkü geçirgen yüzeylerden geçen ışının doğrultusu değişmemektedir.

Kırılmayı modellemek biraz daha zordur. Kırılma doğrultusu nesnelerin kırılma indislerine göre değişmektedir. Kırılan ışının doğrultusunu hesaplamada Snell kanunundan faydalanılacaktır. Snell kanununa göre ışının nasıl kırıldığı Şekil 1.20. 'de gösterilmiştir.



Şekil 1.20. Snell kanununa göre ışının kırılması

Snell kanununa göre bir ortamdan diğerine kırılarak geçerken ışının yüzey normaliyle yaptığı açtığı açı aşağıdaki formülle bulunur:

$$n_1 \text{Sin}(\theta_1) = n_2 \text{Sin}(\theta_2) \tag{1.16}$$

Burada n_1 birinci ortamın, n_2 de ışının kırılarak geçtiği ortamın kırılma indisidir. θ_1 ve θ_2 ise sırasıyla birinci ve ikinci ortamlarda ışının yüzey normaliyle yaptığı açıdır.

Gelen ışının doğrultusu i, yüzey normali n (ikisi de normalize edilmiş) olmak üzere kırılan ışının doğrultusu aşağıdaki ifade ile hesaplanabilir [2]:

$$\mathbf{t} = \mathbf{ri} + (w\text{-}k)\mathbf{n} \tag{1.17}$$

Burada t kırılan ışının (normalize edilmiş) doğrultusudur. R ortamların kırılma indislerinin oranıdır. Yani $r = n_1/n_2$ 'dir. w ve k aşağıda verilmiştir:

$$w = -(\mathbf{i}*\mathbf{n})r, \tag{1.18}$$

$$k = \sqrt{1+(w-r)(w+r)}\,. \tag{1.19}$$

1.7. Boyama (Shading)

Boyama (Shading) üretilecek görüntüdeki piksellerin rengini belirleme işlemidir. 3 ana boyama yöntemi vardır [13]:

1. Düz (Flat).
2. Gouraud.
3. Phong.

Bu yöntemler sırasıyla poligon, köşe noktaları ve piksel mertebesinde renk belirlerler. Düz boyamada üçgen için renk belirlenir ve üçgenin tamamı bu renge boyanır. Gouraud boyama [9] modeli üçgenin köşeleri için renk değerleri hesaplanıp aralarda kalan yerler bu değerlere göre belirlenir. Phong boyama [15] modeli üçgenin köşe noktalarının normalleri kullanılarak piksellerin renkleri belirlenir.

Tezde Phong boyama modeli uygulanmıştır. Phong boyama modeline göre herhangi pikselin rengi belirlenirken üç bileşen kullanılır:

1. Ambient.
2. Diffuse.
3. Specular.

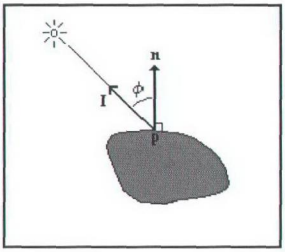
1.7.1. Ambient Bileşen

Standart ışın izleme modelinde doğrudan aydınlatma (direct illumination) göz önüne alınır. Yani ışınların yüzeye yalnızca ışık kaynağından geldiği varsayılır. Ama gerçekte o yüzeye başka yüzeylerden yansıyan ışınlar da gelmektedir. Örneğin bir odada masa ve onun üzerinde de bir lamba olsun. Lambadan yayılan ışınlar masaya çarpıp yansıyacak ve masanın altına ışın doğrudan gitmediği için orası gölgede kalacak. Işın izleme yöntemine göre bu bölgelerin koyu siyah olması gerekir. Ama gerçekte duvarlardan yansıyan ışınlar gereği tam siyah olmaz. Işın izlemede ortamdan kaynaklanan ışıklandırmayı işleme sokmak için bütün cisimler için sabit bir ambient renk değeri kullanılır. Eğer daha kaliteli görüntüler üretilmek istenirse genel aydınlatmanın (global illumination) kullanıldığı yani dolaylı ışıkların da değerlendirildiği radiosity gibi teknikler kullanılmalıdır.

Işık kaynağından bağımsız bir ambient bileşen kullanmak çoğu zaman kabul görmeyen bir yöntemdir. Her yere aynı renk değerinin eklenmesi 3B etkiyi azaltır. Yaygın olarak kullanılan bir yöntem headlight [11] kullanmaktır. Bu yöntem ambient bileşen olarak bakış noktasına yerleştirilmiş bir ışık kaynağı olduğunu varsayar. Gözlemci hareket ettikçe ışık kaynağı da hareket eder ve bütün yüzeyler değişen oranda ambient bileşene sahip olur. O nedenle yapılan çalışmalarda ambient bileşen olarak headlight kullanılmıştır.

1.7.2. Diffuse Bileşen

Boyamanın Diffuse bileşeni ışık kaynağı ile yüzeyler arasındaki etkileşimi fiziksel gerçeklere en yakın modelleyen parçasıdır. Diffuse bileşen Lambert kanununa dayanır. Buna göre yüzey üzerindeki herhangi bir noktanın diffuse bileşeni yüzey normali ile o noktadan ışık kaynağına doğru olan vektör arasındaki açının kosinüsüdür. Şekil 1.21.'de diffuse bileşenin geometrik gösterimi verilmiştir.



Şekil 1.21. Diffuse bileşen

Diffuse bileşenin ifadesi aşağıda verilmiştir:

$$i_{dif} = \mathbf{n} * \mathbf{I} = \cos \phi \tag{1.20}$$

Bu ifadede **n** ve **I** vektörleri normalize edilmiştir. O nedenle bu vektörlerin skaler çarpımları aralarındaki açının kosinüsünü verir. Çünkü vektörlerin boyu 1 birimdir. İfadeden de anlaşılacağı gibi diffüse bileşen yüzey normali ile ışık kaynağına olan vektörler arasındaki açının kosinüsü ile orantılıdır. ϕ açısı 0 derece için diffuse bileşen maksimum olur. ϕ açısı 90 derece olunca diffuse bileşen 0 olmaktadır. $\phi > \pi/2$ için yüzey ışık kaynağı tarafından aydınlatılmamaktadır. Yani arka yüz (back face) olmaktadır. Arka yüz kaldırma konusu anlatılırken bu konuda detaylı bilgi verilecektir.

Işık kaynağının diffuse rengi s_{dif} , yüzeyin diffuse rengi m_{dif} olduğu varsayıldığında pikselin diffuse renk bileşeni aşağıdaki gibi hesaplanır:

$$i_{dif} = (\mathbf{n}*\mathbf{I})\, m_{dif} \otimes s_{dif} \tag{1.21}$$

İfadedeki $\otimes$ sembolü renk bileşenlerinin ayrı ayrı çarpıldığını gösterir. Örneğin mavi ışık kaynağı için s_{dif} = (0.0, 0.0, 1.0) ve kırmızı yüzey için m_{dif} = (1.0, 0.0, 0.0) alınırsa $m_{dif} \otimes s_{dif}$ = (0.0, 0.0, 0.0) olmaktadır. Yani kırmızı yüzey mavi ışığı emer. Beyaz ışık

için s_{dif} = (1.0, 1.0, 1.0) 'dir. Işın izleme ile üretilen görüntülerde beyaz ışık kaynağı kullanılmıştır.

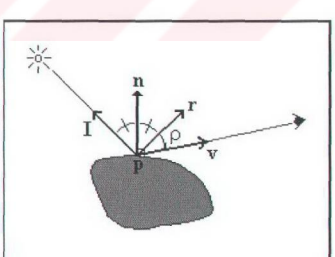
Diffuse bileşenin ifadesine dikkat edilirse bakış noktasının ifadede yer almadığı görülecektir. Yani diffuse bileşen bakış noktasından bağımsızdır (view-independent).

1.7.3. Specular Bileşen

Specular bileşenin amacı yüzeye parlak bir görünüm etkisi katmaktır. Yüzeylerde oluşan parlamalara 'highlight' denir. Highlightlar ışık kaynağının yeri ve yüzeylerin eğriselliğini anlamada gözlemciye yardımcı olurlar [14]. Specular bileşenin ifadesi aşağıda verilmiştir:

$$i_{spec} = (\mathbf{r} * \mathbf{v})^{mshi} = (\cos \rho)^{mshi} \qquad (1.22)$$

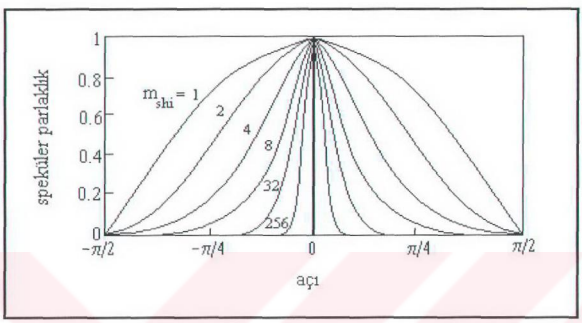
Burada v, yüzey üzerindeki **p** noktasından bakış noktasına olan vektör, **r** ise yüzeyden ışık kaynağına olan **I** vektörünün yüzey normalinden yansımasıdır. Bu vektörler Şekil 1.22.'de gösterilmiştir.



Şekil 1.22. Specular bileşen

Specular bileşene Phong ışıklandırma eşitliği (Phong lighting equation, Phong boyama ile karıştırılmamalıdır) denir.

Specular bileşen ifadelerinde vektörlerin skaler çarpımlarının üssü olan m_{shi} 'ye parlaklık parametresi (shininess parameter) denir. m_{shi} değeri artıkça nesneler üzerinde oluşan specular highlightların çapı azalmaktadır. Bu durum Şekil 1.23.' te gösterilmiştir.



Şekil 1.23. Parlaklık parametresinin değişimi

Yansıma vektörü r = **I** - 2(**n***I)**n** olduğu yansıma anlatılırken gösterilmişti. Eğer **n***I < 0 ise yüzeye ışık gelmez. O nedenle highlight hesaplanmamalıdır.

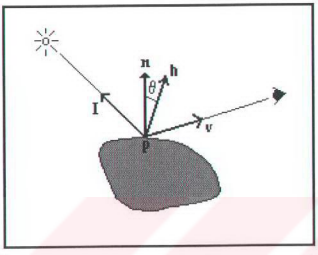
Specular bileşenin ifadesine bakıldığında r ile v 'nin aynı olduğu durumda maksimum değerde olduğu görülmektedir. Bu durumda eğer bakış noktasından yüzeye bakarsak ışık kaynağının yansıması doğrudan gözümüze gelir. Yansıyan ışın ile gözlemci arasındaki açı arttıkça specular bileşen de azalmaktadır. Diffuse bileşen gözlemciden bağımsız olduğu halde specular bileşen gözlemcinin konumuna bağımlıdır ve gözlemci hareket ettikçe highlightlar da yer değiştirmektedir.

Specular bileşenin Blinn tarafından değiştirilmiş hali yaygın olarak kullanılmaktadır. Blinn 'e göre specular bileşen aşağıdaki gibidir [3]:

$$\mathbf{i}_{spec} = (\mathbf{n} * \mathbf{h})^{mshi} = (\cos\theta)^{mshi} \qquad (1.23)$$

Burada **h** (half vector), **l** ve **v** arasındaki vektördür. **h** Şekil 1.24'te gösterilmiştir ve şöyle hesaplanır:

$$\mathbf{h} = (\mathbf{l} + \mathbf{v}) / \| \mathbf{l} + \mathbf{v} \| \tag{1.24}$$



Şekil 1.24. Blinn'in specular yaklaşımı

Diffuse bileşende olduğu gibi ışık kaynağının rengi ve yüzeyin rengi specular ifadeye eklendiğinde ifadenin son hali aşağıdaki gibi olur:

$$\mathbf{i}_{spec} = (\mathbf{n} * \mathbf{h})^{mshi}\, m_{dif} \otimes s_{spec} \tag{1.25}$$

Specular bileşen için Schlick'in farklı bir yaklaşımı vardır. Schlick'e göre specular bileşen aşağıdaki gibi hesaplanır [18]:

$$t = \cos \rho \tag{1.26}$$

$$\mathbf{i}_{spec} = t / (m_{shi} - t m_{shi} + t)\, \mathbf{m}_{dif} \otimes \mathbf{s}_{spec} \tag{1.27}$$

1.7.4. Aydınlatma Modeli

Bu bölümde buraya kadar anlatılan ambient, diffuse ve specular bileşenler biraaya getirilerek bir aydınlatma modeli ortaya konacaktır. Aydınlatma modelini yerel aydınlatma modeli (local lighting model) diye adlandırmak daha doğrudur. Çünkü burada yalnızca ışın kaynağından gelen ışınlar hesaba katılmıştır. Farklı yüzeylerden yansıyan ışınların ektisini aydınlatma modeline katabilmek için ambient bileşen kullanılmıştır. Sonuçta yerel aydınlatma modeli ifadesi aşağıdaki gibidir:

$$\mathbf{i}_{top} = \mathbf{i}_{amb} + \mathbf{i}_{dif} + \mathbf{i}_{spec} \tag{1.28}$$

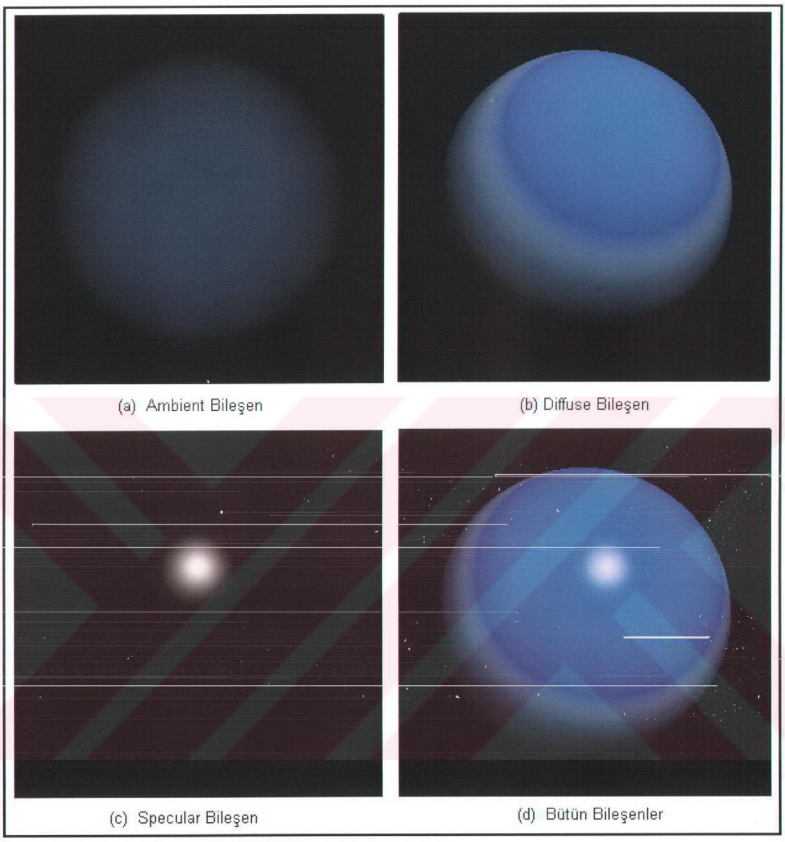
Gerçekte ışık nesneleri aralarındaki uzaklığın karesiyle ters orantılı olarak aydınlatır. Ters orantılı katsayı d olmak üzere d 'nin ifadesi aşağıdaki gibidir:

$$d = 1/ \| \mathbf{s}_{pos} - \mathbf{p} \|^2 \tag{1.29}$$

Bu ifadede $\mathbf{s}_{pos}$ ışık kaynağını yeri, $\mathbf{p}$ de aydınlatılacak noktadır. d katsayısı aydınlatma modeline eklendiğinde son hali aşağıdaki gibi olur.

$$\mathbf{i}_{top} = \mathbf{i}_{amb} + d(\mathbf{i}_{dif} + \mathbf{i}_{spec}) \tag{1.30}$$

Aydınlatma modelinde kullanılan bileşenler Şekil 1.25. (a), (b), (c)'de gösterilmiştir. Şekil 1.25.(d)'de de bileşenlerin tamamı kullanılarak elde edilen görüntü verilmiştir.



Şekil 1.25. Aydınlatma modelinin bileşenleri

1.8. Doku Kaplama (Texture Mapping)

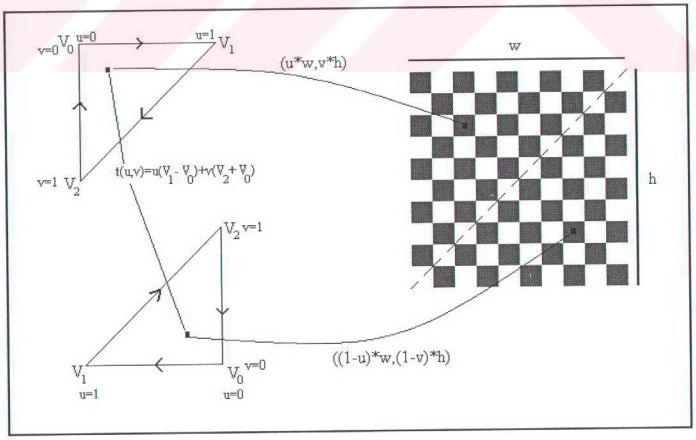
1.8.1. Düzlemsel Yüzey Üzerine Doku Kaplama

Işın izleme ile doku kaplama yapmak için Tomas Möler'in ışın-üçgen kesişim testinden faydalanılmıştır. Bilindiği gibi Möller'in yöntemine göre kesişim testi

yapıldığında yalnızca kesişim noktasına uzaklık olan t değeri değil aynı zamanda kesişim noktasının barisentrik koordinatları da hesaplanabiliyordu. Barisentrik koordinatlar ve üçgenin köşe noktaları kullanılarak üçgen üzerinde herhangi bir noktaya gidebilmek için aşağıdaki ifade kullanılıyordu:

$$t(\mathbf{u},\mathbf{v}) = V_0 + \mathbf{u}(V_1 - V_0) + \mathbf{v}(V_2 - V_0)$$

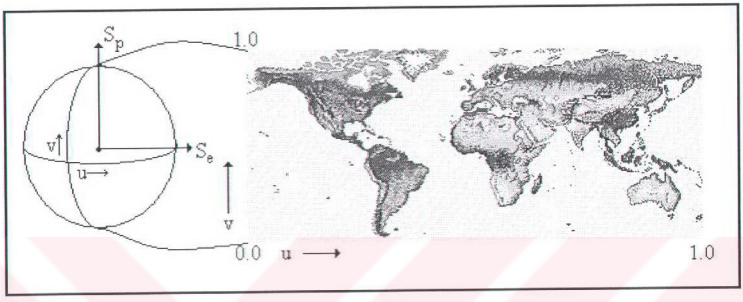
Üçgenin köşe noktaları uygun sırada tutulursa barisentrik koordinatlarla doku kaplama için kullanılan resim dosyasındaki doğru piksel koordinatları rahatlıkla bulunabilir. Bunun için üç varsayım yapılmıştır. Birincisi üzerine doku kaplama yapılacak yüzey düzlemseldir. İkincisi yüzey iki dik üçgenden oluşmaktadır. Köşe noktalarının dizilişi V_0, V_1 ve V_2 olmalıdır ve dik kenarları birleştiren köşe noktası V_0 olmalıdır. Bu varsayımlar kullanarak doku kaplama şu şekilde yapılır: Doku kaplamada kullanılacak resmin genişliği w, yüksekliği h olduğunda, üzerine doku kaplama yapılacak olan yüzeye ait dik üçgenlerden birincisi için resim dosyasındaki koordinatları bulmak için (u*w,v*h) formülü kullanılır. İkinci dik üçgen için de ((1-u)*w,(1-v)*h) formülü kullanılır. Şekil 1.26.'da barisentrik koordinatlarla doku kaplamanın nasıl yapıldığı gösterilmektedir.



Şekil 1.26. Barisentrik koordinatlarla doku kaplama

1.8.2. Küre Üzerine Doku Kaplama

Küre üzerine doku kaplamanın nasıl yapıldığı Şekil 1.27.'de gösterilmiştir.



Şekil 1.27. Küre üzerine doku kaplama.

Şekil 1.27'den görüldüğü gibi küre üzerine doku kaplama yapılırken $\mathbf{S_p}$ ve $\mathbf{S_e}$ vektörlerinden yararlanılmaktadır. $\mathbf{S_p}$ vektörü kürenin merkezinden kuzeye doğru olan vektör; $\mathbf{S_e}$ vektörü ise kürenin merkezinden ekvatora doğru olan vektördür. Doku kaplama yaparken $\mathbf{S_p}$ ve $\mathbf{S_e}$ vektörleri normalize edilerek kullanılır. Bu vektörler ve küre üzerindeki kesişim noktası bilindiğinde küre üzerine doku kaplama yapmak oldukça kolaydır. Şekilde u ve v parametrelerinin resim ve küre üzerinde nasıl değiştiği gösterilmiştir. Dikkat edilirse u, resim üzerinde yatay olarak ilerlerken, küre üzerinde ekvatora paralel olarak ilerlemekte; v de resim üzerinde dikey olarak ilerlerken, küre üzerinde ekvatora dik meridyen olarak ilerlemektedir [5].

Kesişim noktasının normali $\mathbf{S_n}$ olduğunda küre üzerine doku kaplama yapmak için v eksenin boyunca $\mathbf{S_p}$ ile $\mathbf{S_n}$; u eksenin boyunca da $\mathbf{S_e}$ ile $\mathbf{S_n}$ arasındaki açıya bakılır. Küre üzerinde hesaplanan u ve v değerlerinin resim üzerindeki piksel koordinatlarını bulmak için u ve v sırasıyla resmin genişliği ve uzunluğuyla çarpılmalıdır. Küre üzerine doku kaplama için yalancı dilde yazılmış kod aşağıda verilmiştir [7]:

1: $\phi = \arccos(-\mathbf{S_p} * \mathbf{S_n})$

2: $v = \phi / \pi$

3: $\theta = ((\mathbf{S_e} * \mathbf{S_n}) / \sin(\pi)) / (2*\pi)$

4: if($(\mathbf{S_p} \times \mathbf{S_e}) * Sn > 0$) then $u = \theta$

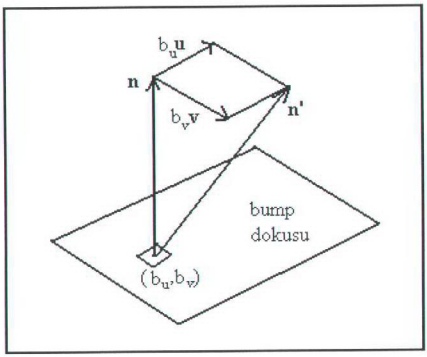
5: else $u = 1 - \theta$

1.8.3. Bump Mapping ile Doku Kaplama

Blinn [2] tarafından 1978'de ortaya atılan bump mapping, doku kaplamada kullanılacak resim dosyasından faydalanılarak üretilecek görüntünün yüzey normalini değiştirmeye dayanır. Kullanılan bump dokusu ile yüzey normali değiştirilerek üretilen görüntünün yüzeyinde sanki tümsekler çukurlar varmış gibi görülür. Bump mapping çok sayıda poligon gerektiren hayvanların kas yapısı, giysiler gibi şeyleri modelleyebilir [16].

1.8.3.1. Heightfield Yöntemiyle Bump Mapping

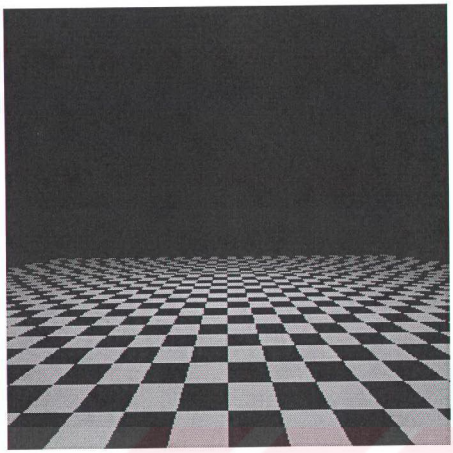
Bu yönteme göre bump mapping ile doku kaplamada kullanılacak dokuya ait her bir monokrom renk bir yüksekliği temsil eder. Dokudaki beyaz renk yüksek alanı; siyah renk de alçak alanı temsil eder. Şekil 1.28.'de gösterilen, yüzey normalini değiştirmede kullanılacak olan u ve v değerlerini bulmak için bu monokrom renk değerleri kullanılır. Dokudaki komşu sütunların farlı alınarak u değerleri; satırların farkı alınarak da v değerleri hesaplanır [17].



Şekil 1.28. Heightfield ile bump mapping

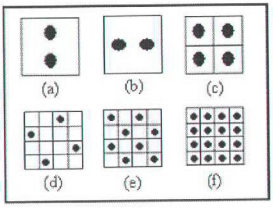
1.9. Aliasing ve Antialiasing

Doku kaplama yaparken örnekleme eksikliği sonucu üretilen görüntülerde bozulmalar olabilmektedir. Klasik ışın izleme yönteminde her bir pikselden bir tane ışın yollanır. Üzerine doku kaplama yapılacak yüzey piksel çözünürlüğünden daha yüksek bir çözünürlükte renk değişimini gerektiriyorsa veya pikselden geçen birincil ışın pikselden küçük üçgenlerle kesiştiğinde üretilecek görüntüde bozulmalar olacaktır. Örneğin bir yüzey üzerine satranç tahtası dokusu kaplanmak istensin. Eğer yollanan ışın hesaplanan doku koordinatları sonucu satranç tahtası üzerinde siyah renkten beyaz geçildiği kenara denk geliyor ve bu kenar için ışının geçtiği piksele doku üzerinde 1 'den fazla piksel karşılık geliyorsa üretilen görüntüde kenarlarda bozulmalar olmaktadır. Doku kaplama yapılacak yüzey bakış noktasından uzaklaştıkça bu bozulmalar daha da artmakta ve üretilen görüntü daha da bozulmaktadır. Görüntülerdeki bu tür bozulmalara aliasing etkisi denir. Bunun bir örneği Şekil 1.29.'da verilmiştir.



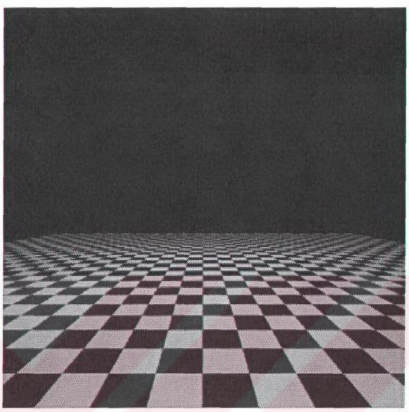
Şekil 1.29. Aliasing etkisi

Antialiasing görüntülerdeki aliasing etkisini kaldırma işlemidir. En basit yöntem üretilecek görüntünün her bir pikseli için daha fazla örnekleme yapmaktır. Bir piksel için 1 'den fazla örnekleme yapmaya supersampling denir. Işın izlemede supersampling yapmak için görüntü düzlemindeki her bir pikselden 1'den fazla ışın yollanır. Pikselden geçen ışınlar için hesaplanan renk değerlerinin ortalaması alınarak o pikselin rengi belirlenir. Çeşitli piksel supersampling türleri Şekil 1.30.'da verilmiştir [13]:



Şekil 1.30. Bazı supersampling türleri

Şekil 1.29.'daki görüntünün Şekil 1.30.(c)'deki gibi supersampling yapılarak yani bir piksel için dört örnek alınarak üretilmiş hali Şekil 1.31.'de verilmiştir:



Şekil 1.31. Antialiasing

1.10. Işın İzlemeyi Hızlandırma Yöntemleri

Işın izleme ile üretilen 3B görüntüler çok kalite olsa da görüntü üretimi çok zaman almaktadır. Işın izlemenin en çok zaman aşaması kesişim testleridir. O nedenle kesişim testlerini hızlandırmaya yönelik yöntemler geliştirilmiştir. Bu yöntemlerin çoğu kesişim testleri öncesi 3B görüntüyü oluşturacak poligonlar üzerinde bazı testler yaparak kesişim testlerinde kullanılacak poligon sayısını indirgemeye yönelik yöntemlerdir.

1.10.1. Arka-Yüz Kaldırma (Backface Culling)

Bakış noktasından bir küreye bakıldığını varsayılsın. Bu kürenin neredeyse yarısı gözlemci tarafından görülmemektedir. Hatta gözlemci küreye yaklaştıkça görünmeyen yüzeylerin oranı daha da artmaktadır. Normalde bakış noktasından görünmesi imkansız

olan yüzeyler kesişim testleri öncesi belirlenip kaldırılırsa testlere harcanan zaman oldukça azalır.

Bakış nokrasından bakıldığında görülmesi imkansız olan yani gözlemciye ters olan yüzeylere arka-yüz (backface) denir. Arka-yüz kaldırma yapmak için yüzey normalinden faydalanılır.

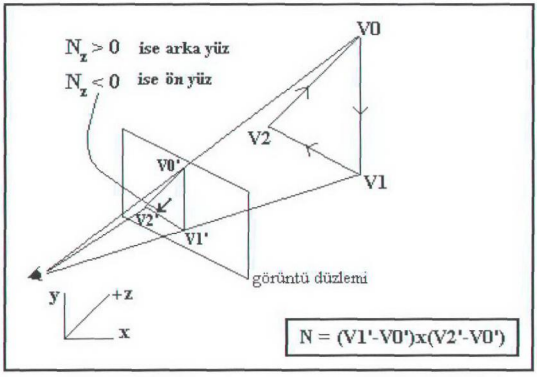
1.10.1.1. Vektörel Çarpımla Arka-Yüz Kaldırma

Bu yönteme göre öncelikle poligonun görüntü düzlemine izdüşümü alınır. İzdüşümü alınmış poligonun normali hesaplanır.

Eğer poligonun z bileşeni sıfırdan küçükse o poligon arka-yüzdür.

Eğer poligonun z bileşeni sıfırdan büyükse o poligon ön-yüzdür.

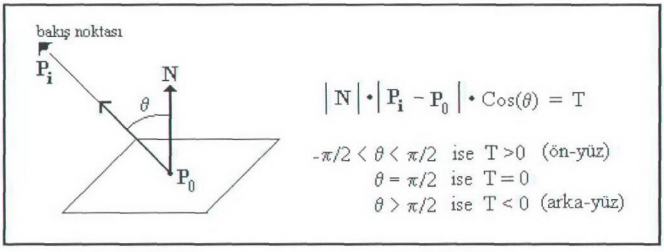
Yalnız burada dikkat edilmesi gereken bir nokta vardır: Poligonun üçgen olduğu varsayılsın. Bilindiği gibi köşeleri V_0, V_1 ve V_3 olan üçgenin normali $N = (V_1 - V_0) \times (V_2 - V_0)$ ifadesi ile hesaplanıyordu. Hesaplanan normalin z bileşenine göre üçgenin arka-yüz olup olmadığına karar verebilmek için üçgenin köşe noktalarının sırasına dikkat edilmelidir. Yukarıda verilen arka-yüz kaldırma şartı üçgenin köşeleri saat yönünün tersi sırada alındığı ve -z-ekseni görüntü düzlemine doğru olduğu durumda geçerlidir. Eğer köşeler saat yönünde seçilirse bu sefer +z ekseni görüntü düzlemine doğru olmalıdır. Şekil 1.32. 'de arka-yüz kaldırma gösterilmiştir. Şekilde üçgenin köşelerinin saat yönünde sıralandığı ve +z-ekseninin görüntü düzlemine doğru olduğu varsayılmıştır. Yapılan çalışmalarda üçgenlerin köşe noktaları hep saat yönünde alınmıştır.



Şekil 1.32. Vektörel çarpımla arka-yüz kaldırma

1.10.1.2. Skaler Çarpımla Arka-Yüz Kaldırma

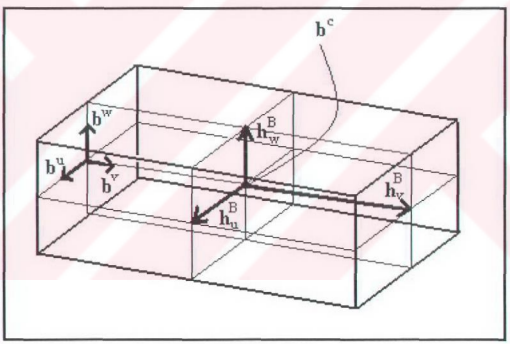
Bu yönteme göre yüzey üzerindeki herhangi bir noktadan bakış noktasına doğru olan vektör belirlenir. Bu vektörle yüzey normalinin skaler çarpımı yapılır. Yüzey normali ile bakış noktasına olan vektörlerin normalize edilmiş olduğu varsayıldığında bu vektörlerin skaler çarpımı aralarındaki açının kosinüsünü verir. Bu eğer açı $\pi/2$'den büyükse skaler çarpım negatif çıkar. Buradan yüzeyin arka-yüz olduğu anlaşılır. Pozitif değerler için yüzey bakış noktasından görünür yani ön-yüz (frontface) dür. Şekil 1.33.'te bu yönteme göre arkan yüz kaldırma gösterilmiştir.



Şekil 1.33. Skaler çarpımla arka-yüz kaldırma

1.10.2. Çevreleyen Hacim Yöntemi (Bounding Volume)

Çevreleyen hacim yönteminde 3B nesneleri içine alan ve kesişim testinin hızlı olması için onlardan daha basit 3B geometrik şekiller kullanılır. Dikdörtgen prizma yaygın olarak kullanılmaktadır. Çevreleyen hacim olarak dikdörtgen prizma kullanıldığında ışın-nesne kesişim testi öncesi o nesneyi çevreleyen dikdörtgen prizma ile ışın arasında kesişim testi yapılır. Eğer ışın prizma ile kesişiyorsa o zaman 3B nesne ile kesişip kesişmediği test edilir. Böylece üretilecek bütün görüntü için o nesne için yalnızca etrafını çevreleyen dikdörtgen prizma ile kesişen ışınlar için kesişim testi yapılır. Böylece gereksiz ışın testleri azaltılmış olur. Çevreleyen hacim dikdörtgen prizma Şekil 1.34.'te gösterilmiştir [8].



Şekil 1.34. Dikdörtgen prizma

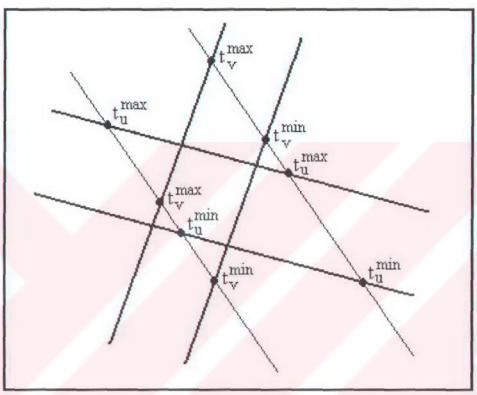
Şekil 1.34'teki $\mathbf{b}^c$ prizmanın merkezidir. $\mathbf{b}^u$, $\mathbf{b}^v$ ve $\mathbf{b}^w$ normalize edilmiş yön vektörleridir. Prizmanın merkezinden yüzeylerine uzaklıklar da h^B_u, h^B_v ve h^B_w ile gösterilmiştir.

Işın-Prizma kesişim testinde diğer kesişim testlerinde olduğu gibi ışının prizmaya t uzaklığı hesaplanmaya çalışılır. Prizma 3 çift birbirine paralel yüzeyden oluşmaktadır. Bu yüzey çiftlerlerine slab denir. Her bir slab için u, v ve w eksenleri için minimum ve maksimum t değeri vardır. Şekil 1.35'te uv ekseni için minimum ve maksimum t değerleri

prizma üzerinde gösterilmiştir. Bu değerler t^{min} ve t^{max} ile gösterilirse t^{min}_i, t^{max}_i, $\forall i \in \{u,v,w\}$ 'dir. Kesişim testi için aşağıdaki ifade ile t^{min} ve t^{max} değerleri hesaplanır [12].

$$t^{min} = \max(t^{min}_u, t^{min}_v, t^{min}_w)$$

$$t^{max} = \min(t^{max}_u, t^{max}_v, t^{max}_w)$$



Şekil 1.35 u-v ekseni için min. ve max. t değerleri

Eğer $t^{min} <= t^{max}$ ise ışın prizmaya kesişiyordur. Aksi halde kesişmez. Işın-prizma kesişimi için yalancı dilde yazılmış bir kod aşağıda verilmiştir [13]:

Işın_Prizma_Kesişimi(o,d,B)

1: $t^{min} = -\infty$

2: $t^{max} = +\infty$

3: $\mathbf{p} = \mathbf{b}^c - \mathbf{o}$

4: foreach $i \in \{u,v,w\}$

5: $e = \mathbf{a}^i * \mathbf{p}$

6: $f = \mathbf{a}^i * \mathbf{d}$

```
7:          if( | f | > ε )
8:              t1 = (e+hi) / f
9:              t2 = (e-hi) / f
10:             if(t1 > t2) swap(t1, t2)
11:             if(t1 > t^min) t^min = t1
12:             if(t2 < t^max) t^max = t2
13:             if(t^min > t^max) return (REJECT,0)
14:             if (t^max < 0 ) return (REJECT,0)
15:         else if((-e-hi)>0 | | (-e+hi)<0 ) return (REJECT,0)
16:     if(t^min > 0) return (INTERSECT, t^min)
17:     else return (INTERSECT, t^max)
```

Kodun 7. satırındaki if(| f | > ε if(| f | > ε)) ifadesiyle ışının slablara paralel olup olmadığı test edilir. Burada $\varepsilon = 10^{-20}$ seçilebilir. 13. satırdaki test sonucu return çalışırsa ışın prizmayla kesişmiyor; 14. satırda ise prizma ışının gerisinde demektir. 15. satıra gelindiğinde ışınının prizmaya paralel olduğu biliyor yalnız prizmanın içinde mi dışında mı olduğu test edilmelidir. 15. satırdaki test bunu yapmaktadır.

Küre, çevreleyen hacim olarak yaygın kullanılır. Çevreleyen hacim olarak küre kullanıldığında öncelikle ışın-çevreleyen küre kesişim testi yapılır. Eğer ışın küreyle kesişiyorsa çevrelediği nesneler için kesişim testleri yapılır. Işın-küre kesişim testi daha önce anlatıldığından burada tekrar bahsedilmeyecektir.

1.11. Paralel Bilgisayarlarla Işın İzleme

Işın izleme yöntemi ile 3B görüntü üretimi oldukça fazla hesaplama isteyen bir iştir. Algoritmik etkinliğin hız açısından yetersiz kaldığı noktada hız için başka çözümler bulunmalıdır. Hızı arttıracak en basit çözüm paralel bilgisayarlardan ya da işlemcilerden oluşan sistemler kullanmaktır [20].

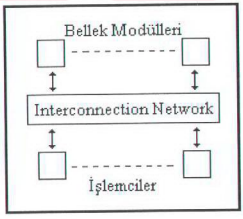
1.11.1. Paralel Bilgisayar Türleri

Paralel bilgisayarları ortak bellek kullanarak haberleşen işlemcilerden oluşan sistemler ve mesajlaşarak haberleşen bilgisayarlardan oluşan sistemler olmak üzere iki ana gruba ayırmak mümkündür.

1.11.1.1. Paylaşımlı Bellekli Çoklu-İşlemcili Sistem

Paylaşımlı Bellekli Çoklu-İşlemcili Sistem (Shared Memory Multiprocessor System) Şekil 1.36.'dan da görüldüğü gibi işlemciler ile bellek modüllerinin bir interconnection network ile birbirlerine bağlanmasıyla oluşturulmuş bir sistemdir. Bu sistemde tek adres uzayı vardır. Her bir bellek lokasyonu için tek adres değeri vardır. Bütün işlemciler bu adres değerini kullanarak istediği bellek bölgesine erişebilir.

Paylaşımlı Bellekli Çoklu-İşlemcili Sistemde işlemciler ortak bellek modülleri aracılığıyla haberleşirler. Veriler ortak olarak kullanıldığı için bu tip sistemlerin programlanması oldukça kolaydır. Yalnız bütün belleğe bütün işlemcilerin hızlı olarak erişmesini sağlamak donanımsal olarak oldukça zordur.



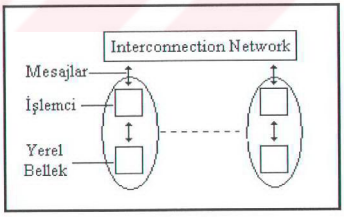
Şekil 1.36. Paylaşımlı bellekli çoklu-işlemcili sistem

1.11.1.1. Mesaj-Geçmeli Çoklu-Bilgisayar Sistemi

Paylaşımlı Bellekli Çoklu-İşlemcili Sistem özel olarak tasarlanmış bir bilgisayar sistemidir. O nedenle maliyeti oldukça yüksektir. Bu sistemin alternatifi Şekil 1.37.'de de görüldüğü gibi bilgisayarları bir interconnection network ile birbirine bağlamaktır. Her bilgisayarın, başkasının erişemeyeceği kendi işlemcisi ve belleği vardır. Bu sistemde bellek bilgisayarlar arasında dağıtılmış olduğundan her bir bilgisayarın kendi adres uzayı vardır. Bilgisayarlar network üzerinden mesaj yollayarak haberleşirler.

Bu sistemin en önemli avantajı ölçeklemenin yani sisteme yeni bilgisayarlar eklemenin çoklu işlemcili sisteme göre çok daha kolay olması ve böylece sisteme eklenen her yeni bilgisayar sayesinde performansın sürekli artmasıdır. Ayrıca network aracılığıyla birbirine bağlı bilgisayarlar üzerinde doğrudan uygulanabilir bir sistemdir.

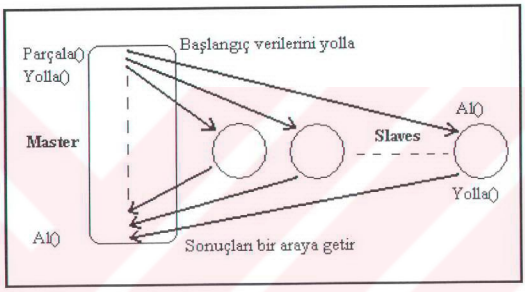
Kısa zamanda daha hızlı işlemcilerin piyasaya çıkmasıyla birlikte bir çok özel tasarlanmış çoklu-işlemcili sistemlerin ömrü de çok kısalmaktadır. Yeni bir bilgisayarın işlemcisi m işlemcili bir çoklu-işlemcili sistemdeki işlemcilerden m kere daha hızlıysa maddi açıdan bakıldığında yeni bilgisayarı kullanmak gerektiği aşikardır. Ayrıca bilgisayarlardan oluşan networke yeni bir bilgisayar eklemek oldukça kolaydır. O nedenle çalışmalar Mesaj-Geçmeli Çoklu-Bilgisayar Sistemi üzerinde yapılmıştır [1].



Şekil 1.37. Mesaj-geçmeli çoklu-bilgisayar sistemi

1.11.2. Paralel Bilgisayarları Programlamada Kullanılan Yaklaşım

Paralel bilgisayarları programlarken Master-Slave yaklaşımı kullanılmıştır. Master-Slave yaklaşımı Şekil 1.38.'de gösterilmiştir. Master süreç problemi alt parçalara ayırır ve her bir Slave sürece hesaplaya başlaması için ihtiyaç duyduğu başlangıç verilerini yollar. Slave'ler hesaplamayı bitirince sonuçları Master sürece yollar ve o da sonuçlar bir araya getirir [6].



Şekil 1.38. Master-Slave yaklaşımıyla paralel programlama

2. YAPILAN ÇALIŞMALAR

2.1. Giriş

Çalışmalarda öncelikle ışın izleme yöntemiyle 3B görüntü üretimi gerçeklenmiştir. Çeşitli ışın-üçgen kesişim yöntemleri denenmiştir. Görünmeyen yüzeylerin kaldırılması yapılmıştır. Doku kaplama yapılmıştır. Doku kaplamada karşılaşılan aliasing problemi supersampling yapılarak çözümlenmiştir. Piksel rengini belirlemede Phong aydınlatma modeli kullanılmıştır. Buna göre her bir pikselin ambient, diffuse ve specular bileşeni hesaplanmıştır.

Yansıyan ışınlar modellenmiş ve farklı derinlik değerleri için yansıma ile görünen nesneler hatasız modellenmiştir. Ayrıca kırılma da fiziksel gerçeklere uygun bir şekilde (kırılma indisleri de hesaba katılarak) modellenmiştir. Yansıyan ve kırılan ışınlar için ışın izleme algoritması özyinelemeli olarak yansıma ve kırılma doğrultuları için yeniden koşturumuştur. Bu yapısıyla algoritma genellik taşımaktadır.

Paralel programlada Master-Slave yaklaşımı modellenmiştir. Ana bilgisayar (Master) işi parçalara ayırıp diğer bilgisayarlara (Slave) yollamış ve onlardan gelen sonuçları bir araya getirmiştir. Multithreaded programlama yapılmış ve buna göre ana bilgisayarda her bir bilgisayar için bir thread başlatılmıştır. Paralel hesaplama yapılırken diğer bilgisayarlar threadlerle kontrol edilmiştir. Networkteki bilgisayar sayısı arttırıldıkça speedupın da lineere yakın oranda arttığı gözlemlenmiştir.

2.2. Özyinelemeli Işın İzleme Programının Tasarlanması

Özyinelemeli ışın izleme algoritması birincil ışınların üretilmesiyle başlar. Birincil ışınlar bakış noktasından çıkan ve görüntü düzlemindeki piksellerin her birinden geçen ışınlardır. Yollanan birincil ışınlar ile 3B nesneler arasında kesişim testleri yapılır ve kesişimler için t uzaklık değerleri hesaplanır. Bu t değerleri sıralanarak en yakın kesişim bulunur. Böylece görünmeyen yüzeylerin kaldırılması işlemi de yapılmış olur. En yakın nesnenin rengi kullanılarak birincil ışının geçtiği pikselin rengi için aydınlatma modelinin ambient, diffuse ve specular bileşenleri hesaplanır. Bu nesne ışığı yansıtan veya kıran

özelliğe sahip ise yansıma veya kırılma doğrultuları hesaplanır ve birincil ışınlar için yapılan işlemler özyinelemeli (recursive) olarak tekrarlanır. Birincil ışınlar, yansıyan veya kırılan ışınlar için bulunan renk değerleri, toplamları 1 olan katsayılarla çarpılarak genel aydınlatma modeli ile pikselin gerçek rengi belirlenir. Böylece piksel üzerinde yalnızca birincil ışının ilk çarptığı yüzeyin değil aynı zamanda yansıyarak veya kırılarak da çarptığı yüzeylerin renkleri de görünür.

2.2.1. Birincil Işınların Üretilmesi, Kesişim Testleri ve Gölge Testi

Birincil ışınların üretilmesi için başlangıç noktasının ve görüntü düzleminde geçtiği pikselin koordinatlarının bilinmesi yeterlidir. Yapılan çalışmalarda genellikle başlangıç noktası (0,0,0) olarak alınmıştır. Görüntü düzlemi 10x10 birim alınmıştır. Bakış noktasının görüntü düzleminin merkezine dik olarak 12 birim geride olduğu başka bir deyişle görüntü düzleminin merkezinin koordinatlarının (0,0,12) olduğu varsayılmıştır. Burada neden 11 veya 13 değil de 12 diye bir soru akla gelebilir. 12 rakamının seçilmesinin özel bir nedeni vardır. Fiziksel olarak gözün görüş açısı ϕ aşağıdaki ifade ile hesaplanır [1]:

$$\phi = 2\arctan(w/2d)$$

Bu ifadede w nesnenin görüş doğrultusuna dik genişliği, d ise nesnenin uzaklığıdır. 36mm görüntü genişliğine (frame size) ve 50mm lens uzaklığına sahşp kamera için bu açı $\phi = 2\arctan(36/(2*50)) = 39.6$ derece olmaktadır. Üretilen görüntüler için görüntü düzlemi 10x10 birim alındığında yani w=10 olduğunda 39.6 derecelik görüş açısı elde edebilmek için görüntü düzleminin bakış noktasından yaklaşık 12 birim ileride olması gerekmektedir.

Üretilecek görüntünün çözünürlüğü 400x400 olduğunda piksel koordinatları (10*X/400 - 5, 5 – 10*Y/400, 12) ifadesiyle hesaplanır.

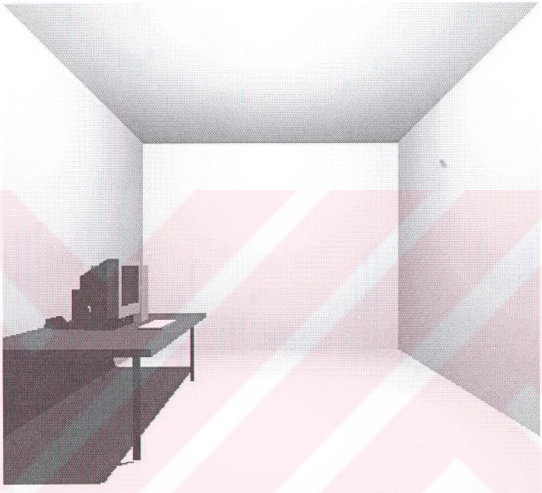
Birincil ışının üretilmesi onun doğrultusunu belirlemek demektir. Bunun için piksel koordinatlarından bakış noktasının koordinatlarının farkı alınır ve bu fark vektörü normalize edilir. Birincil ışınların doğrultuları belirlenince sıra kesişim testlerine gelir. Burada öncelikle ışın-üçgen kesişim testi ile üretilen görüntüler verilecektir.

Işın-üçgen kesişim testlerinde 3 yöntem kullanılmıştır. Bunlar:

1. Alan Hesabı.

2. Jordan Curve Teoremi.
3. Möller'in Yöntemi.

Bu üç yöntem ile Şekil 2.1.'deki 400x400 çözünürlükte görüntü için hesaplama zamanları Tablo 2.1.'de verilmiştir.



Şekil 2.1. Işın-üçgen kesişim testlerini test için üretilen görüntü

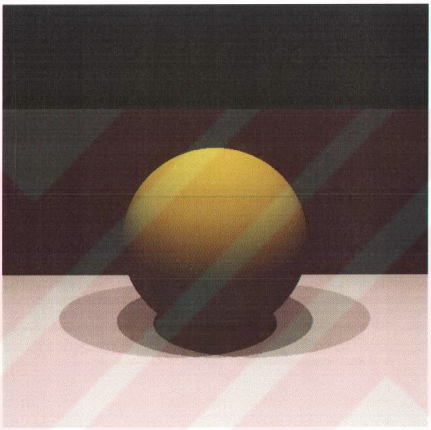
Tablo 2.1. Testlerdeki hesaplama süreleri

Poligon sayısı	Işık kaynağı sayısı	Möllerin Yöntemi	Jordan Curve T.	Alan Hesabı
112	1	73.09 sn.	119.2 sn.	131.77 sn.

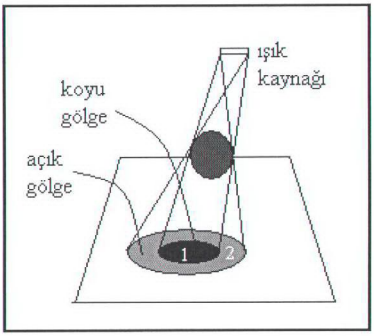
Şekil 2.1.'deki görüntüye dikkat edilirse masa, monitor ve kasanın gölgelerinin de doğru olarak üretildiği görülür. Herhangi bir yüzeyin gölge kalıp kalmadığının belirlenmesi için yüzeyden ışık kaynağına gölge test etme ışınları yollanmıştır. Eğer bu ışın ışık kaynağına ulaşmadan başka bir nesneye çarpıyor ve bu nesne ile yüzey arasındaki

uzaklık yüzey ile ışık kaynağındaki uzaklıktan küçükse yüzey bu nesnenin gölgesinde kalıyor demektir. Aydınlatma modelinin diffuse ve specular bileşenleri hesaplanırken öncelikle gölge testi yapılır. Eğer nesne gölgede kalıyor ise bu bileşen değerleri sıfır olur. Böylece gölgede kalan yüzey için yalnızca ambient bileşen hesaba katılır.

1 'den fazla ışık kaynağı kullanıldığında değişik tonlarda gölgeler elde edilir. Şekil 2.2.'de 3 tane ışık kaynağının aydınlattığı kürenin değişik tonlardaki gölgeleri görülmektedir.

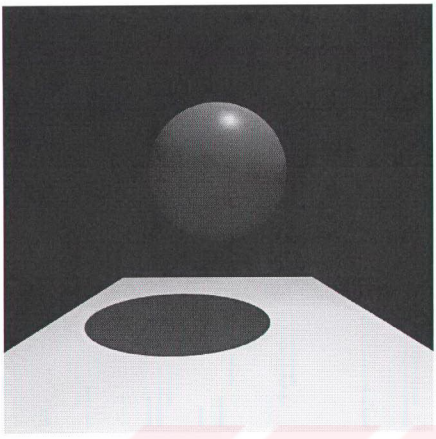


Şekil 2.2. Farklı gölgeler (3 ışık kaynağı için)

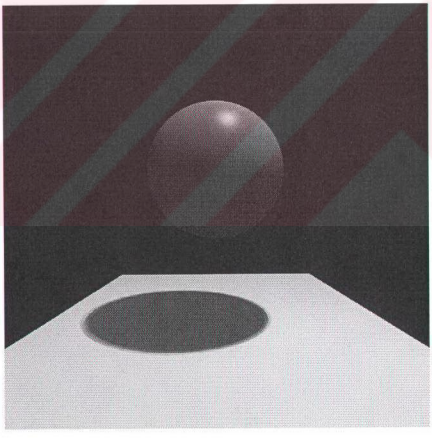


Şekil 2.3. Gölge modeli

Şekil 2.2.'de kullanılan ışık kaynakları noktasal olduğu için keskin gölgeler (hard shadows) elde edilmiştir. Bu durumda gölge Şekil 2.3.'te '1' ile gösterilen alan gibi olur. Poligonal ışık kaynağı kullanıldığında ise gölgelerin ortası koyu kenarları ise açık renkte olur. Bu durumda gölge Şekil 2.3.'te '2' ile gösterilen alan gibi olur. Doğru gölgeler noktasal olmayan ışık kaynağı kullanıldığında elde edilir. Eğer noktasal ışık kaynağı kullanılmışsa poligonal ışık kaynağındaki gibi yumuşak gölgeler (soft shadows) elde etmek için noktasal ışık kaynağı poligonal ışık kaynağının köşeleri üzerinde gezdirilir ve köşeler için üretilen renk değerlerinin ortalaması alınır. Böylece '1' ile gösterilen yumuşak gölgeler elde edilir. Şekil 2.4.'te noktasal ışık kaynağıyla elde edilen görüntü, ardından Şekil 2.5.'te ise anlatılan teknikle gölgelerin yumuşatılmış hali görülmektedir.



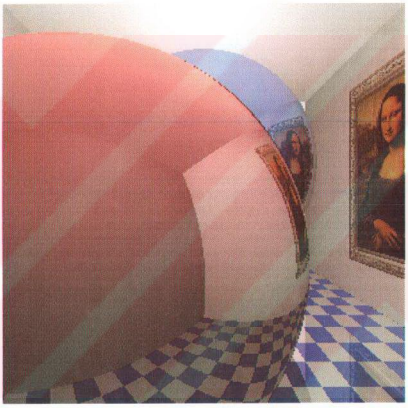
Şekil 2.4. Keskin gölge (hard shadow)



Şekil 2.5. Yumuşak gölge (soft shadow)

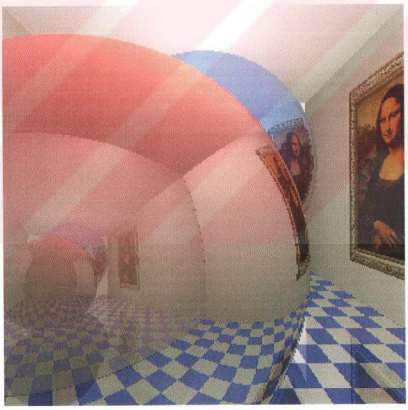
2.2.2. Yansıma ve Kırılma ile Görünen Nesnelerin Modellenmesi

Yansıtıcı özelliğe sahip yüzeylerde yansıma ile görünen nesneleri belirlemek için yansıma doğrusu hesaplanır. Işın yeni doğrultu boyunca ilerlerken çarptığı ilk yüzeyin görüntüsü yansıdığı yüzeyde görünür. Eğer yansıyan ışın yine yansıtıcı özelliğe sahip bir nesneye çarparsa yeniden yansıma doğrultusu hesaplanır ve aynı işlemler yapılır. Çalışmalarda geliştirilen recursive kod sayesinde istenilen sayıda yansıma sonucu görünen nesnenin görüntüsü birincisinde görülebilmektedir. Şekil 2.6., Şekil 2.7. ve Şekil 2.8.'de farklı derinlik değerleri için yansımalar gösterilmiştir.



Şekil 2.6. Yansıma, derinlik=1

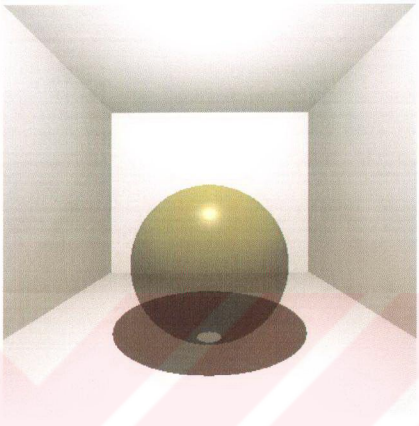
Şekil 2.7. Yansıma, derinlik=2



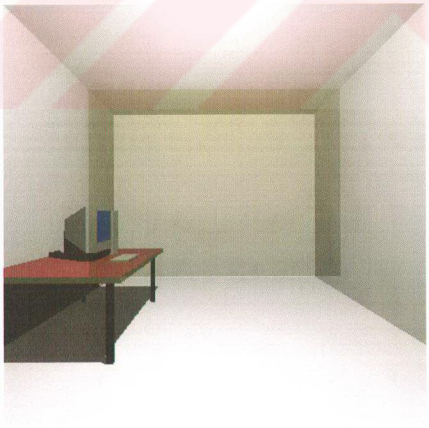
Şekil 2.8. Yansıma, derinlik=3

Kırılmada da yansımada olduğu gibi kırılma doğrultusu hesaplanır ve ışın yeni doğrultu boyunca yollandığında ilk çarğtığı nesnenin görüntüsü birincisinde görünür. Eğer nesne cam gibi ışığı kırmadan iletiyorsa yeni doğrultu hesaplamaya gerek yoktur. Şekil 2.9.'da ışını kırmadan ileten saydam bir küre verilmiştir. Şekil 2.10. ve Şekil 2.11.'de

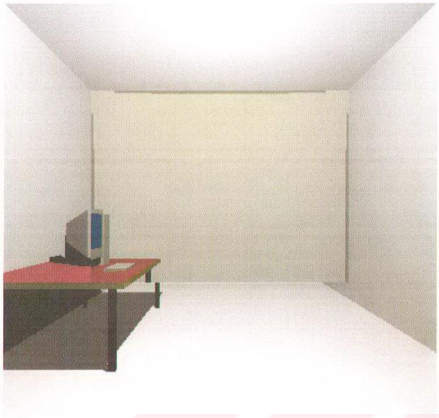
sırasıyla ışını kırmadan ileten ve $n=n_1/n_2 = 1/1.33$ (n' ler kırılma indisleri) oranında kıran yüzey verilmiştir.



Şekil 2.9. Saydam küre



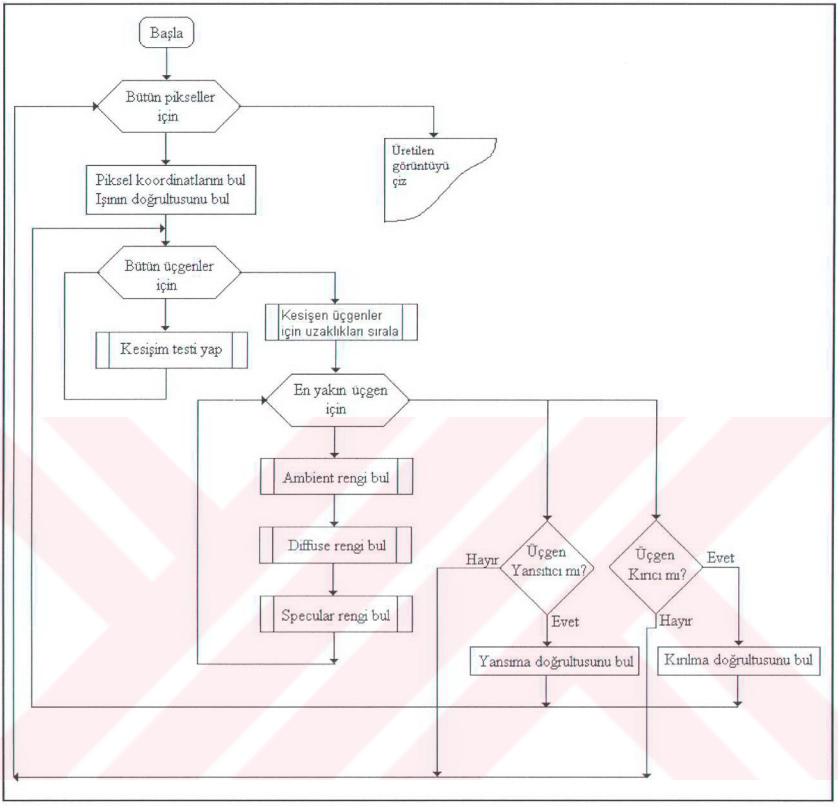
Şekil 2.10. Saydam yüzey



Şekil 2.10. Kırılgan yüzey, n_1=1.0, n_2=1.33

Özyinelemeli Işın İzleme Programının Akış Diyagramı

Özyinelemeli ışın izleme programının akış diyagramı Şekil 2.11.'de verilmiştir. Üretilecek görüntüyü oluşturan nesnelerin üçgenlerden oluştuğu varsayılmıştır. Akış diyagramından da görüldüğü gibi ilk adım piksel koordinatlarının bulunmasıdır. Bu koordinatlar ve başlangıç noktasının koordinatları kullanılarak birincil ışınların doğrultusu belirlenir. Başlangıç noktası ve doğrultusu belirlenmiş ışın ile görüntüyü oluşturacak üçgenler arasında kesişim testleri yapılır. Bulunan t uzaklık değerleri sıralanarak en yakın kesişim noktası bulunur. En yakın üçgen için aydınlatma modelinin bileşenleri hesaplanır. Eğer üçgen ışını yansıtıcı ve kırıcı özelliğe sahipse yansıma ve kırılma doğrultuları hesaplanır ve birincil ışınlar için yapılan işlemler tekrarlanır.



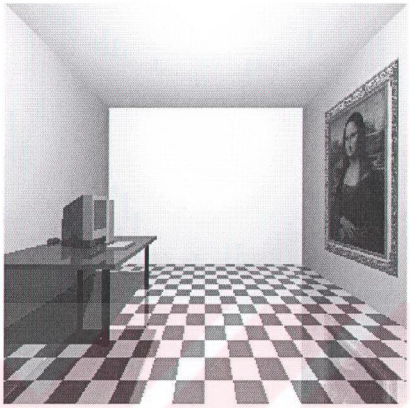
Şekil 2.11. Özyinelemeli ışın izleme programının akış diyagramı

2.4. Düzlemsel Yüzey ve Küre Üzerine Doku Kaplama

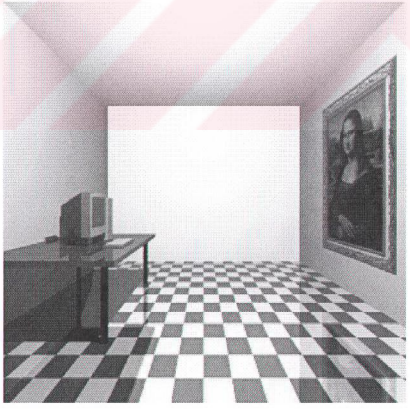
Düzlemsel yüzey üzerine doku kaplama yaparken barisentrik koordinatlardan yararlanılmıştır. Şekil 2.12.'de bir doku kaplama örneği verilmiştir.

Şekil 2.12.'ye dikkat edilirse doku kaplama yapılmış yüzey üzerinde örnekleme eksikliğinden kaynaklanan bozulmalar olmaktadır. Bu bozulmalar supersampling yapılarak düzeltilmişitir. Supersamling 'de her bir piksel için 4 örnekleme yapılmış ve pikselin

gerçek renk değerini belirlemek için örneklemelerin ortalaması alınmıştır. Sunuçta elde edilen görüntü Şekil 2.13.'te verilmiştir.

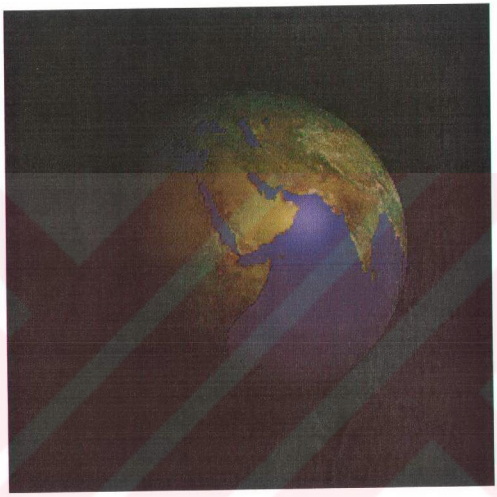


Şekil 2.12. Yüzey üzerine doku kaplama



Şekil 2.13. Supersampling ile bozulmaların düzeltilmesi

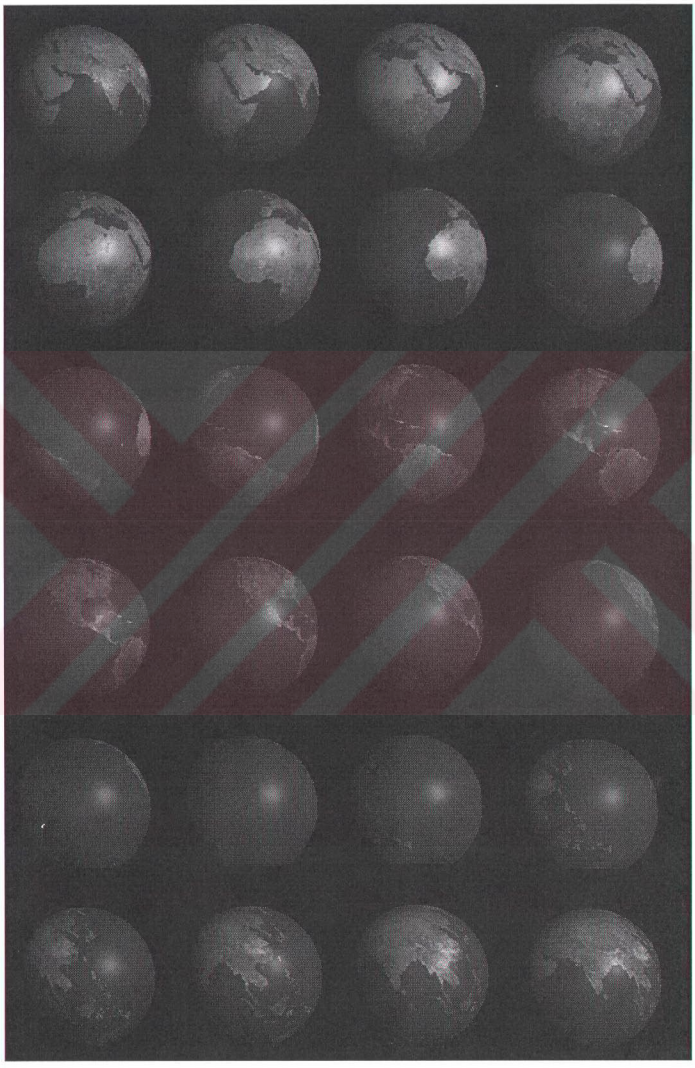
Küre üzerine doku kaplama için genel bilgilerde anlatılan yöntemle u ve v parametreleri hesaplanmıştır. Daha sonra bulunan u ve v değerleri doku kaplamada kullanılan resim dosyasının genişliği ve yüksekliği ile çarpılarak elde edilen koordinatlardaki renk değeri ışın izleme ile hesaplanan kesişim noktasının doku bilgisi olarak kullanılmıştır. Şekil 2.14. 'te küre üzerine doku kaplama örneği verilmiştir.



Şekil 2.14. Küre üzerine doku kaplama

Şekil 2.14. 'teki küre y ekseni etrafında 15 'er derece döndürülerek Şekil 2.15. 'teki görüntüler elde edilmiştir. Daha sonra bu görüntüler birleştirilerek animasyona çevrilmiştir. 24 görüntü, kürenin 360 derece dönmesi anlamına geldiğinden üretilen animasyonda bu 24 görüntü sürekli art arda gösterilerek kürenin devamlı dönmesi sağlanmıştır.

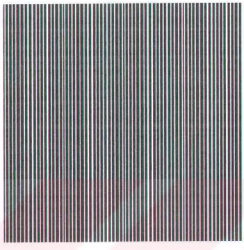
Bilindiği gibi küre üzerine doku kaplama yapmak için S_p ve S_e vektörleri kullanılıyordu. Animasyonda küre y ekseni boyunca döndürüldüğü için rotasyon matrisini S_e vektörü ile çarpmak yeterlidir.



Şekil 2.15. Animasyon için doku kaplama ile üretilen görüntüler

2.5. Bump Mapping ile Doku Kaplama

Heightfield yöntemi uygulanarak bump mapping ile doku kaplama yapılmıştır. Şekil 2.16.'daki bump dokusu kullanılarak bump mapping ile elde edilen görüntü Şekil 2.17.'de verilmiştir.



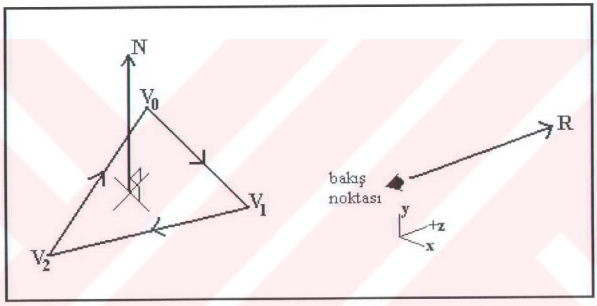
Şekil 2.16. Bump dokusu

Şekil 2.17. Bump mapping örneği

2.6. Işın İzlemeyi Hızlandırma Yöntemleri

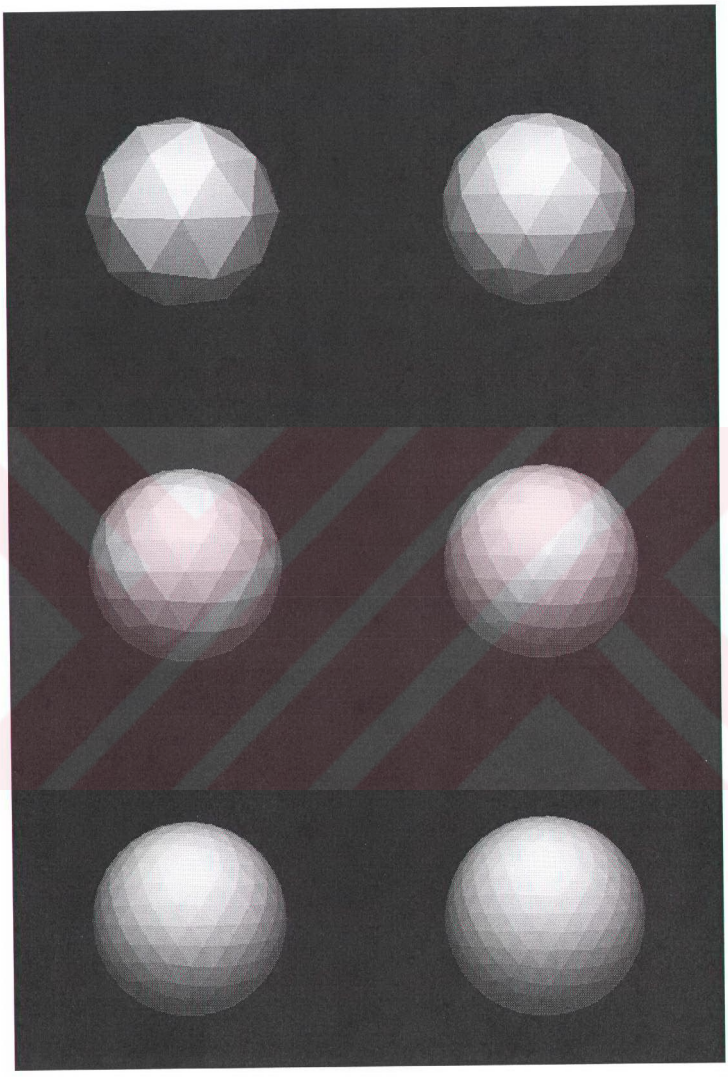
2.6.1. Arka-Yüz Kaldırma

Işın izleme ile 3B görüntüler üretirken üçgenlerin köşe noktalarının sırasının sol el kuralına göre yani saat yönünde olmasına dikkat edilmiştir. Bu durumda yüzey normali yüzeyin görünen yüzünde dik vektör olur. Şekil 2.18.'de verilen üçgenin köşe noktalarının sıralanışı saat yönünde ve koordinat eksenleri de sol el kuralına göre seçildiğinde yüzey normalinin nasıl olacağı gösterilmiştir.



Şekil 2.18. Sol el kuralına göre üçgenin köşe noktalarını sıralanışı

Arka yüz kaldırma teknikleri kullanılarak görünmesi imkansız olan yüzeyler kaldırılmıştır. Arka yüz kaldırmanın ışın izlemeyi ne kadar hızlandırdığını test etmek için 3D Studio Max 4 ortamında çizdirilen farklı (sayıda üçgenlerden oluşan) küreler kullanılmıştır. Bunun için 3D Studio Max 4'te çizdirilen küreler **.ASE** uzantılı dosya formatında kaydedilmiştir. Bu dosyada küreyi oluşturan üçgenlerin köşe noktalarının koordinatları tutulmaktadır. Bir program yazılarak bu koordinatlar okunmuş ve ışın izleme programının işleyebileceği veriyapısına dönüştürülmüştür. Test için kullanılan küreler Şekil 2.19.'da, elde edilen sonuçlar da Tablo 2.1.'de gösterilmiştir.



Şekil 2.19. Farklı sayıda üçgenlerden oluşan küreler

Tablo 2.1. Arka yüz kaldırma sonuçları

	GEÇEN ZAMAN (sn.)	
Kürenin üçgen sayısı	**Arka-yüz kaldırma yok**	**Arka-yüz kaldırma var**
80	14.47	12.53
180	32.89	28.14
320	61.23	49.97
500	101.91	87.47
720	175.89	140.92
980	260.38	224.89

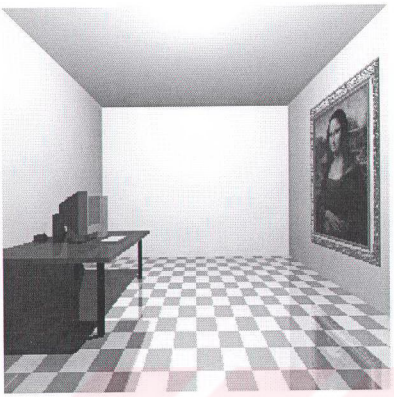
2.6.2 Çevreleyen Hacim Kullanma

Çevreleyen hacim yöntemi ile hızlandırmada çevreleyen küre ve dikdörtgen prizma kullanılmıştır. Şekil 2.19.'daki küreler arka yüz kaldırma yapılarak ve çevreleyen hacim küre kullanılarak elde dilen sonuçlar Tablo 2.2.'de gösterilmiştir.

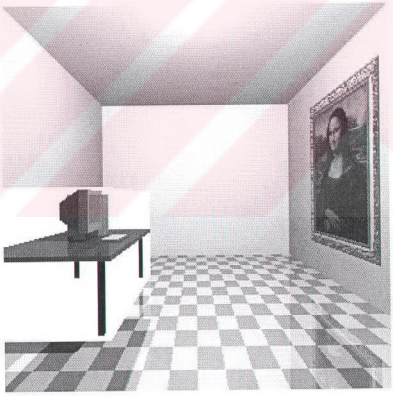
Tablo 2.2. Çevreleyen küre sonuçları

	GEÇEN ZAMAN (sn.)		
Kürenin üçgen sayısı	**Arka-yüz kaldırma yok**	**Arka-yüz kaldırma var**	**Arka-yüz k. Çevreleyen küre**
80	14.47	12.53	8.58
180	32.89	28.14	16.84
320	61.23	49.97	29.58
500	101.91	87.47	51.05
720	175.89	140.92	81.02
980	260.38	224.89	127.55

Şekil 2.20.'de çevreleyen hacim olarak dikdörtgen prizma kullanılarak üretilen görüntü verilmiştir. Görüntüde masa ve bilgisayarı çevreleyen bir dikdörtgen prizma kullanılmıştır. Dikdörtgen prizma Şekil 2.21.'de verilmiştir.



Şekil 2.20. Çevreleyen hacim yöntemiyle üretilmiş görüntü



Şekil 2.21. Çevreleyen hacim dikdörtgen prizma

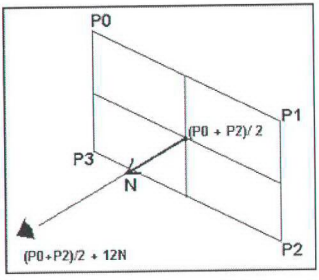
Şekil 2.20.'deki 300x300 piksel çözünürlüğündeki görüntünün çevreleyen hacim yöntemi kullanılmadan üretilmesi için geçen zaman 43.8 saniyedir. Çevreleyen hacim yöntemi kullanılarak üretildiğinde bu zaman 36.7 saniyeye düşmektedir. Çözünürlük 600x600 olduğunda değerler sırasıyla 172.83 saniyeye 144.73 saniye olmaktadır.

2.7. Etkileşimli Işın İzleme

Işın izleme ile görüntü üretimi etkileşimli hale getirilmiştir. Buna göre kullanıcı üretilen görüntü içinde klavyedeki tuşlarla gezinebilmektedir. Gezinirken ileri-geri gidebilmekte; sağa sola dönebilmektedir.

Etkileşimli ışın izlemenin gerçeklenebilmesi için kullanıcının hareket türüne bağlı olarak (ilerleme, dönme) görüntü düzlemi ve bakış noktasının yeni koordinatları belirlenmelidir. Etkileşimli ışın izlemede görüntü düzlemi, dört köşe noktasının koordinatları kullanılarak ifade edilmiştir. Yani ilerleme veya dönme işlemleri yapılırken görüntü düzleminin yeni yerini belirlemede bu köşe noktalarının koordinatlarını kullanmak yeterli olmuştur. Şekil 2.22'den de görüldüğü gibi köşe noktaları için görüntü düzleminin sırasıyla sol üst köşe noktasından başlanarak saat yönünde hareket edilerek P_0, P_1, P_2, P_3 terimleri kullanılmıştır.

Dönme işlemi yapılırken, öncelikle görüntü düzleminin köşe noktalarından bakış noktasının koordinatları çıkarılmış, ardından dönme işlemi için gereken rotasyon matrisi ile yeni koordinatlar çarpılıp bakış noktasının koordinatları tekrar eklenmiştir. Böylece bakış noktasının etrafında dönme gerçeklenmiştir. Görüntü düzleminin dönme sonrası koordinatları bulunduktan sonra bakış noktasının yeni koordinatları hesaplanmalıdır. Bakış noktası her zaman görüntü düzleminin merkezine işaret ettiğinden görüntü düzleminin koordinatları değişince bakış noktasının koordinatları da değişir. Bakış noktasının yeni koordinatlarını bulmak için öncelikle yeni görüntü düzleminin merkezinin koordinatları belirlenmelidir. Görüntü düzleminin merkezi $(P_0 + P_2) / 2$ veya $(P_1 + P_3) / 2$ ile belirlenir. Ardından görüntü düzleminin yeni normali hesaplanır. Çünkü rotasyon sonucu normal değişmiştir. Bakış noktası görüntü düzleminin merkezine dik konumda olduğundan hesaplanan yeni normali bakış noktasının görüntü düzlemine uzaklığı olan 12 ile çarpılarak görüntü düzleminin merkezine eklenmiş ve bakış noktasının yeni koordinatları bulunmuştur.



Şekil 2.22. Etkileşimli ışın izleme

İlerleme işlemi yapılırken her tuşa basıldığında 10 birim ileri veya geri gidildiği varsayılmıştır. İlerleme işleminde görüntü düzleminin yeni koordinatları bulunurken öncelikle normal hesaplanır. Çünkü dönme yapılarak normal değişmiş olabilir. Hesaplanan normal 10 ile çarpılarak görüntü düzleminin köşelerinde çıkarılır. Ardından dönme işleminde anlatıldığı gibi bakış noktasının yeni koordinatları hesaplanır.

Etkileşimli ışın izlemede her tuşa basıldığında bakış noktası ve görüntü düzleminin köşe noktaları P_0, P_1, P_2, P_3'ün koordinatları değiştiğinden yeni koordinatlar için yollanan birincil ışınların doğrultuları da değişmektedir. Bilindiği gibi herhangi bir birincil ışının doğrultusunun hesaplanabilmesi için bakış noktasının ve ışının geçtiği pikselin koordinatlarının bilinmesi gerekmektedir. Dönme veya ilerleme işlemleri için bakış noktasının yeni koordinatlarının nasıl hesaplandığı yukarıda anlatılmıştı. Yeni piksel koordinatları hesaplanırken ise yeni P_0, P_1, P_2, P_3'ün koordinatlarından faydalanılır. Örneğin üretilecek görüntü 400x400 çözünürlükte olduğunda görüntünün x. satır, y. sütunun 3B piksel koordinatları şu ifade ile bulunur:

$$Piksel(x,y) = P_0 + ((P_1-P_0)/400)*x + ((P_3-P_0)/400)*y$$

Şekil 2.23.'te etkileşimli ışın izleme ile üretilen görüntüler verilmiştir. Her hareket için yalnızca görüntü düzlemi ve bakış noktasının koordinatları değiştirildiğinden diğer bir değişle görüntüyü oluşturan üçgenlerin koordinatları değişmediğinden oda içerisinde ne kadar dolaşılırsa dolaşılsın üretilen görüntülerde herhangi bir bozulma olmamaktadır.

Şekil 2.23. Etkileşimli ışın izleme ile üretilen görüntüler

2.8. Paralel Işın İzleme

Işın izleme yöntemi ile görüntü üretiminde pikseller birbirinden bağımsız hesaplanabilmektedir. Bu özelliği ile paralel hesaplama için çok uygun bir yöntemdir. Paralel programlamada Master-Slave yaklaşımı kullanılmış ve yerel ağ üzerinde önce statik iş tahsisi yapılmıştır. Karmaşık görüntülerde statik iş tahsisi yük dengesizliklerine neden olabilmektedir. Ayrıca ağ heterojen ise yani farklı hızlardan oluşan bilgisayarlardan oluşmuş ise statik iş tahsisi yük dengesizliğini daha da arttırmaktadır. Karmaşık görüntüler ve heterojen ağda olabilecek yük dengesizlikleri işlemci çiftliği modeli gerçeklenip dinamik iş tahsisi ve böylece yük dengeleme yapılarak çözümlenmiştir.

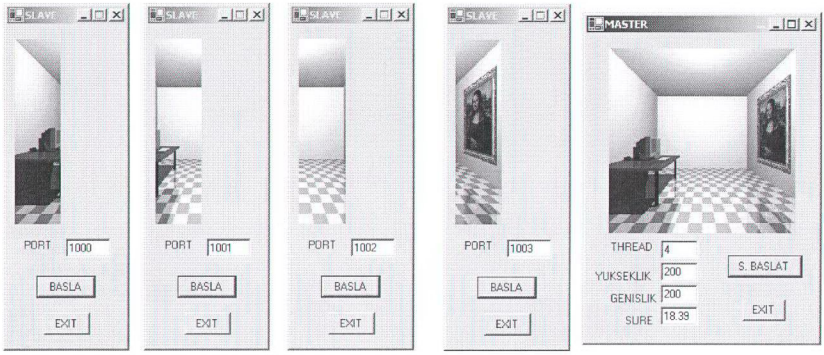
2.8.1. Master-Slave Yaklaşımı ile Paralel Programlama

Çalışmada paralel programlama için Master-Slave yaklaşımı kullanılmıştır. Bu yaklaşımda Ana bilgisayar (Master) işi parçalara böler ve her bir bilgisayara (Slaves) bir iş verir. İşlerinin bitiren bilgisayarlar sonuçları ana bilgisayara yollarlar. Ana bilgisayar da onları bir araya getirir.

Işın izlemede herhangi bir pikselin renk bilgisi diğerinden bağımsız hesaplanabildiğinden paralel hesaplamaya çok yatkın bir yöntemdir. Geliştirilen Master program aracılığıyla ağ üzerindeki bilgisayarlara iş tahsisi yapılıp onlardan gelen sonuçlar bir araya getirilerek yüksek speedup değerlerinde görüntüler üretilmiştir.

Master program kullanıcıya bilgisayar sayısını sorar ve her bir bilgisayar için bir thread başlatır. Thread her bir bilgisayar için ayrı bir portu dinlemeye başlar. Ağ üzerindeki herhangi bir bilgisayar (Slave) dinlenen portlardan birinden bağlantı kurduğunda ona görüntünün hangi parçasını üreteceğine dair piksel koordinatlarının bilgisini yollar. Bu bilgiyi alan bilgisayar ilgili görüntüyü üretir ve Master programın koştuğu ana bilgisayara yollar.

Şekil 2.24. 'te Master program ve Slave programlar görülmektedir.



Şekil 2.24. Master program ve Slave programların görüntüsü

2.8.2. Paralel Etkileşimli Işın İzleme

Etkileşimli ışın izleme Master-Slave yaklaşımıyla paralel programlama ile birleştirilmiş ve Paralel Etkileşimli Işın İzleme yapılmıştır. Buna göre ana bilgisayar basılan tusa göre (*W* ileri gitme, *A* sola dönme, *S* geri gitme, *D* sağa dönme tuşlarıdır) yerel ağdaki bilgisayarlara tus ve üretecekleri görüntü bilgisini yollamaktadır. Bütün bilgisayarlardan gelen sonuçları bir araya getirip ekrana çizmektedir. Kullanıcı her tuşa basışında aynı işlemler tekrarlanmaktadır.

2.8.3. İşlemci Çiftliği Modeli ile Dinamik İş Tahsisi ve Yük Dengeleme

Yerel ağdaki bilgisayarların heterojen olduğu durumlarda yük dengelemesi yapmak kaçınılmazdır. Ayrıca ışın izlemede herhangi bir pikselin renk bilgisinin hesaplanması için geçen süre diğerinden farklı olabilmektedir. Görüntünün karmaşıklığı arttıkça ve özellikle hızlandırma yöntemi olarak çevreleyen hacim kullanıldığında yük dengesizlikleri olabilmektedir. İşte bu iki durum için işlemci çiftliği (processor farm) modeli kullanılarak dinamik iş tahsisi ve böylece yük dengelemesi yapılmıştır.

İşlemci çiftliği modelinde ana bilgisayar işi parçalara ayırıp bir iş havuzu (task pool) oluşturmaktadır. Yerel ağdaki her bir bilgisayar bu havuzdan iş istemekte ve işi

tamamlayıp sonuçları yollayınca havuzdaki bütün işler tamamlanana dek yeni bir iş isteyip onunla ilgilenmektedir.

Yapılan uygulamada iki bilgisayar üzerinde işlemci çiftliği modeli uygulanarak dinamik iş tahsisi ve yük dengelemesi yapılmıştır. Ana bilgisayar üretilecek görüntüyü istenilen parçaya bölmekte ve ağ üzerinde paralel çalışan bilgisyarlar istekte bulundukça onlara yeni yeni işler tahsis etmektedir. Geliştirilen özel mekanizmayla aynı anda iki bilgisayarın iş istediğinde iş bunlardan birine verilmektedir. Bunun için özel bir değişken kullanılmış ve bu değişkene eşzamanlı erişim kullanılan programlama dilinin (Visual C#) sağladığı **lock(this){.....}** yapısı ile engellenmiştir. lock emri ile {.....} arasındaki işlemler aynı anda tek bir thread tarafından yapılabilir. Herhangi bir değişken ya da kod bloğunun aynı anda tek bit thread tarafından işlenmesi hızı azaltmaktadır. Ayrıca üretilecek görüntüyü paralel bilgisayarlardan çok daha fazla parçalara bölmek haberleşme zamanın artmasına neden olmaktadır. O nedenle iş havuzundaki (task pool) iş sayısı iyi seçilmelidir.

3. BULGULAR ve TARTIŞMA

3.1. Giriş

Yapılan çalışmalarda ışın izleme yöntemi ile gerçeğe yakın kalitede görüntü üretimi gerçeklenmiştir. Işın izleme ışının yansıma, kırılma gibi davranışlarını fiziksel esaslara uygun olarak modelleyebilmektedir. Yapılan çalışmalarda yansıyan veya kırılan ışınlar için gerçek fiziksel esaslar kullanılmıştır. Gerçek hayattaki fiziksel olayların modellenmesinde ışın izleme yöntemini kullanmak mümkündür.

Işın izleme ile görüntü üretimi piksel mertebesinde gerçeklendiğinden üretilen görüntülerin kalitesi yüksektir. Piksel mertebesinde çalışma ile görünmeyen yüzeylerin kaldırılması, gölgelerin belirlenmesi hatasız yapılmıştır.

Üçgenlerin köşe noktalarının sıralanışlarına dikkat edilerek düzlemsel yüzey üzerine doku kaplarken barisentrik koordinatları kullanmak yeterli olmuştur.

Işın izlemenin piksel mertebesinde hassas olması üretilen görüntülerin hesaplama zamanlarını arttırmaktadır. Bu dezavantajı ve herhangi bir pikselin diğerinden bağımsız hesaplanabilme avantajı ışın izleme ile karmaşık görüntüler üretirken paralel hesaplama yapılmasını gerektirir.

Işın izleme ile görüntü üretirken herhangi bir pikselin renk bilgisini hesaplama zamanı diğerlerinden farklı olabilir. Bunun sebebi ilgili pikselden geçen ışının kesiştiği nesnelerin sayısı ve fiziksel özellikleridir. Arkayüz kaldırma veya çevreleyen hacim yöntemleri bu zaman farkını artttırmaktadır. Görünmeyen yüzeyleri kaldırırken kesişim testlerinde bulunan t değerleri sıralanmalıdır. Kesişim sayısı ve dolayısıyla t değerlerini sıralama zamanları pikselden piksele değişebilmektedir. Eğer kesişen nesne ışığı yansıtıcı veya kırıcı özelliğe sahipse yansıyan veya kırılan ışının doğrultusu bulunup yeni ışın için kesişim testi yapmak gerekmektedir. Birbirinden farklı özelliklerde nesnelerden oluşan görüntüler üzerilirken ışınların çarptığı nesnelerin yansıtıcı/kırıcı özelliğe sahip olup olmaması ışının geçtiği pikselin rengini belirleme zamanını değiştirmektedir. Işın izlemenin bu özelliği paralel ışın izleme yapılırken değişik tekniklerin kullanılmasını gerektirebilmektedir. Özellikle statik iş tahsisi yapıldığında bazı görüntüler üretilirken yük dengesizlikleri olabilmektedir. Ayrıca ağdaki bilgisayarların farklı hızlarda olması da yük dengesizliğinin bir nedenidir. Bunun için yapılan çalışmalarda yük dengesizliğini

minimuma indirmek için İşlemci Çiftliği (Processor Farm) modeli gerçeklenmiş ve dinamik iş tahsisi ile yük dengelemesi yapılmıştır.

Bu bölümde önce paralel etkileşimli ışın izlemede statik iş tahsisi yapılarak üretilen görüntüler için hesaplanan speedup değerleri ardından işlemci çiftliği modeli uygulanarak dinamik iş tahsisiyle yük dengelemesi sonucu hesaplanan speedup değerleri verilecektir.

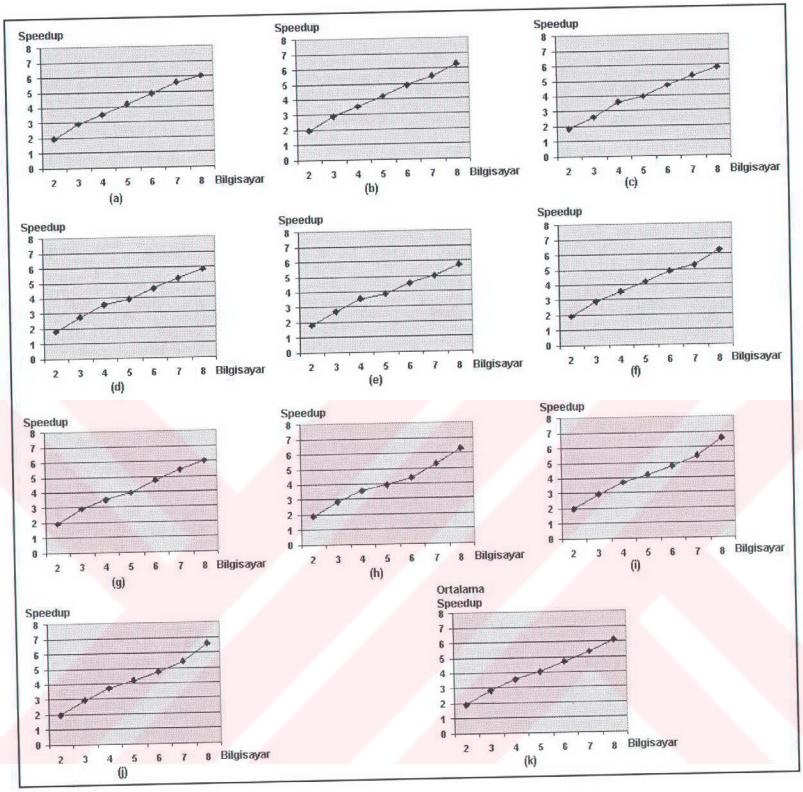
3.2. Paralel Etkileşimli Işın İzleme

Statik iş tahsisi ile paralel etkileşimli ışın izleme yapılmıştır. Bunun için öncelikle P4 2.4 GHz işlemcili, 256 MB Ram'i olan bilgisayarlar 10/100 Mbps hızında 8-Port Fast Ethernet Switch'e bağlanarak bir yerel ağ kurulmuştur.

Paralel etkileşimli ışın izleme ile üretilecek görüntüler önce bir bilgisayarda ardından ağ üzerinde sırasıyla 2, 3, 4, 5, 6, 7 ve 8 bilgisayar paralel çalıştırılarak üretilmiş ve hesaplama zamanları kullanırak speedup değerleri hesaplanmıştır. Görüntülerin üretilme zamanları Tablo 3.1.'de, speedup değerleri Tablo 3.2.'de ve speedup eğrileri de Şekil 3.1.'de verilmiştir.

Tablo 3.1. Paralel etkileşimli ışın izleme hesaplama zamanları (sn.)

Tuş	1 Bil	2 Bil	3 Bil	4 Bil	5 Bil	6 Bil	7 Bil	8 Bil
W	70.83	36.28	24.41	20.17	16.88	14.5.	12.64	11.81
W	70.38	36.11	24.38	20.06	16.78	14.36	12.86	11.22
D	64.95	35.05	23.45	18.08	16.48	13.94	12.27	11.08
W	64.14	35.23	23.44	18.09	16.48	13.94	12.22	11.03
W	62.53	34.47	23.06.	17.7	16.31	13.83	12.47	10.92
A	66.89	34.62	23.19	18.92	16.02	13.72	12.73	10.75
A	68.67	35.38	23.66	19.67	17.47	14.47	12.61	11.42
W	68.67	36.06	24.36	19.27	17.53	15.64	12.98	10.97
W	69.66	35.36	23.83	18.81	16.53	14.7	12.88	10.59
A	68.05	35.06	23.47	19.53	16.84	15.19	12.64	10.89



Şekil 3.1. Paralel etkileşimli ışın izleme ile üretilen görüntüler için speedup eğrileri

Şekil 3.1.(a)..(j) arasındaki speedup eğrilerileri her bir tuş (W,A,S,D) için elde edilen eğrilerdir. Şekil 3.1.(k)'da ise bu eğrilerin ortalaması verilmiştir. Bu şekilden de görüldüğü gibi speedup değerleri bilgisayar sayısı arttırıldıkça lineere yakın oranda artmaktadır.

Tablo 3.2. Paralel etkileşimli ışın izleme için speedup değerleri

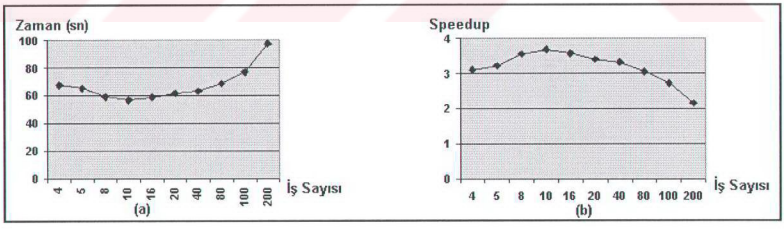
Tuş	2 Bil	3 Bil	4 Bil	5 Bil	6 Bil	7 Bil	8 Bil
W	1.95	2.9	3.51	4.2	4.88	5.6	6
W	1.95	2.89	3.51	4.19	4.9	5.47	6.27
D	1.85	2.6	3.6	3.94	4.66	5.29	5.82
W	1.82	2.74	3.55	3.89	4.6	5.25	5.82
W	1.81	2.71	3.53	3.83	4.52	5.01	5.73
A	1.93	2.88	3.54	4.18	4.88	5.25	6.22
A	1.94	2.9	3.49	3.93	4.75	5.45	6.01
W	1.9	2.82	3.56	3.92	4.39	5.29	6.26
W	1.97	2.92	3.7	4.21	4.74	5.41	6.58
A	1.94	2.9	3.48	4.04	4.48	5.38	6.25

3.3. İşlemci Çiftliği Modeli ile Dinamik İş Tahsisi ve Yük Dengeleme

Kurulan ağ üzerinde statik iş tahsisi yapıldıktan sonra yalnız 4 bilgisayar paralel çalıştırılarak dinamik iş tahsisi ile yük dengeleme yapılmış ve Tablo 3.3.'te verilen sonuçlar bulunmuştur. Hesaplama zamanı ve speedup eğrisi Şekil 3.2.'de verilmiştir.

Tablo 3.3. Dinamik iş tahsisi ile yük dengeleme sonuçları

iş sayısı	4	5	8	10	16	20	40	80	100	200
zaman	75.4	73.06	66.86	64.77	66.56	69.56	71.02	76.49	84.84	105.31



Şekil 3.2. Dinamik iş tahsisi ile yük dengeleme için hesaplama zamanı ve speedup eğrileri

Tablo 3.3.'teki iş sayısı 4 için bulunan hesaplama zamanı ve speedup değeri statik iş tahsisi içindir. Çünkü yerel ağdaki bilgisayar sayısı 4'tür. Diğer iş sayısı ve elde edilen sonuçlar dinamik iş tahsisi sonuçlarıdır. Statik iş tahsisi ile dinamik iş tahsisini karşılaştırmak için bir arada verilmiştir. Şekil 3.2.(a)'dan da görüldüğü gibi iş sayısı 40

olana kadar hesaplama zamanı statik iş tahsisinin altındadır. Yalnız speedup değeri Şekil 3.2.(a)'den de görüldüğü gibi iş sayısı 10 olana kadar artmakta sonra düşmeye başlamaktadır. O nedenle bu uygulama için optimum iş sayısını 10 olduğu ortaya çıkmaktadır.

Şekil 3.3. Dinamik iş tahsisi ile yük dengeleme yapılarak üretilen görüntü

4. SONUÇLAR

Çalışmada öncelikle ışın izleme ile 3B görüntü üretimi gerçeklenmiştir. Işın-nesne kesişim testleri ve özellikle ışın-üçgen kesişim testi yöntemleri karşılaştırmalı olarak incelenmiştir.

Geliştirilen özyinelemeli ışın izleme algoritması ile yasıyan ve kırılan ışınlar modellenmiştir. Algoritma özyinelemeli olduğundan yansıma ve kırıma sayısı istenildiği kadar arttırılabilmektedir. Böylece ışın birinci yüzeyden birkaç kez yansıyıp/kırılıp en son çarptığı yüzeyin rengi birinci yüzeyde görülebilmektedir.

Her bir pikselin rengi Phong'un boyama modeline göre belirlenmiştir. Ambient, diffuse ve specular bileşenler doğru olarak hesaplanmıştır.

Üretilen görüntülerin kalitesini arttırmada yaygın olarak kullanılan doku kaplama tekniği ve özellikle yüzey normalini değiştirerek o yüzeyin kullanılan dokuya göre batık/kabarık parçalardan oluşmuş izlenimini veren bump mapping ile doku kaplama yapılmıştır. Aliasing problemi supersampling yapılarak çözümlenmiştir.

Arka-yüz kaldırma ve çevreleyen hacim yöntemleri kullanılarak ışın izleme hızlandırılmıştır. Bunun için arkayüz kaldırmanın doğru olarak gerçeklenebilmesi için 3B görüntüyü oluşturacak bütün üçgenlerin köşe noktalarının saat yönünde sıralanmış olarak tutulmasına dikkat edilmiştir..

Etkileşimli ışın izleme yapılmıştır. Buna göre kullanıcı üretilen 3B ortamda istediği gibi hareket edebilmektedir. Her bir hareket için yalnızca görüntü düzlemi ve bakış noktasının yeni koordinatları hesaplanmaktadır. 3B nesnelerin koordinatları değişmediğinden 3B ortamda ne kadar gezinme yapılırsa yapılsın üretilen görüntülerde bozulma olmamaktadır.

Master-Slave yaklaşımı ile paralel ışın izleme yapılmıştır. Homojen ağ üzerinde statik iş tahsisi yapıldığında speedup'ın lineere yakın olarak arttığı görülmüştür.

İşlemci çiftliği modeli gerçeklenmiştir. Böylece dinamik iş tahsisi ile yük dengelemesi yapılabilmektedir.

5. ÖNERİLER

Çalışmalarda ışın izleme ile üretilen görüntüler doğrudan aydınlatma (direct illumination) veya yerel aydınlatma (local illumination) esasına dayanmaktadır. Buna göre herhangi bir yüzey doğrudan ışık kaynağı tarafından aydınlatılıyorsa onun diffuse ve specular bileşenleri hesaplanır. Ama gerçek hayatta durum böyle değildir. Işık kaynağından çıkıp başka yüzeylerden yansıyan ışınlar da o yüzeyi dolaylı yoldan aydınlatmaktadır. Bu şekilde yapılan aydınlatma dolaylı aydınlatma (indirect illumination veya global illumination) denir. Dolaylı aydınlatma yapılarak çok daha kaliteli görüntüler üretmek mümkündür.

Çalışmalar Windows işletim sistemi altında Visual C# programlama dili kullanılarak yapılmıştır. Paralel hesaplama soket programlama ile gerçeklenmiştir. TCP/IP'nin yavaş bir protokol olması geliştirilen uygulamalar için bir dezavantajdır. Paralel hesaplama için tasarlanmış ve genelde Unix işletim sistemi altında çalışan PVM (Parallel Virtual Machines) platformu kullanılmamıştır. Paralel hesaplama için tasarlanmış bir yazılım geliştirme olduğundan PVM kullanılarak daha etkin paralel programlama yapmak mümkündür.

6. KAYNAKLAR

1. Athas, W.,C., ve Seitz C.,L., "Multicomputers: Message-Passing Concurrent Computers", Computer, 21(8), 1988, 9-24.

2. Bec, ve Xavier, "Faster Refraction Formula, and Transmission Color Filtering", Ray Tracing News, 10(1), 1997.

3. Blinn, J.,F., "Models of Light Reflection for Computer Synthesized Pictures", ACM Computer Graphics, 192-198, 1977.

4. Blinn, J.,F., "Simulation of Wrinkled Surfaces", Computer Graphics, Ağustos 1978, 286-292.

5. Blinn, J.,F., ve Newell, M.,E., "Texture and Reflection in Computer Generated Images", Communications of the ACM ,19(10), 1976, 542-547.

6. Braunl, T., Parallel Programming An Introduction, Prentice Hall, London, 1993.

7. Glassner, A., An Introduction To Ray Tracing, Ninth printing, Morgan Kaufmann Publishers, San Francisco, 2002.

8. Gottschalk, ve Stefan, "Collision Queries using Orinented Bounding Boxes", Ph.D. Thesis, Department of Computer Science, University of North Carolina at Chapel Hill, 1999.

9. Gouraud, H., "Continuous Shading of Curved Surfaces", IEEE Transactions on Computers, C-20, 1971, 623-629.

10. Greene, ve Ned, "Environment Mapping and Other Applicaitons of World Projections", IEEE Computer Graphics and Applications, 6(11), 1986, 21-29.

11. International Standard ISO/IEC 14772-1:1997 (VRML).

12. Kay, T.,L., ve Kajiya, J.,T., "Ray Tracing Complex Scenes", Computer Graphics, Ağustos 1986.

13. Möller, T., ve Haines E., Real-Time Rendering, Second Edition, A. K. Peters, 2002.

14. Möller, T., ve Trumbore B., "Fast Minimum Storage Ray-Triangle Intersection", Journal of Graphics Tools, 2(1), 1977, 21-28.

15. Phong, Bui Tuong, "Illumination for Computer Generated Pictures", Communicaitons of the ACM, 18(6), 1975, 311-317.

16. Sander, Pedro V., John Snyder, Steven J. Gortler, ve Hugues Hoppe, "Texture Mapping Progressive MEshes", Computer Graphics, Ağustos 2001, 409-416.

17. Schlag, J., "Fast Embossing Effect on Raster Image Data", Graphics Gems IV, Academic Press, 1994, 433-437.

18. Schlick, C., "A Fast Alternative to Phong's Specular Model", Graphics Gems IV, Academic Press, 1994, 385-387.

19. Stone, J.,E., "An Efficient Library For Parallel Ray Tracing And Animation", University of Missouri,1998.

20. Wilkinson, B., Allen, M., Parallel Programming, Prentice Hall, New Jersey, 1999.

ÖZGEÇMİŞ

Ömer ÇAKIR 1978'de Trabzon'da doğdu. İlk ve orta öğrenimini Tabzon'da yaptı. 1996'da KTU Bilgisayar Mühendisliği Bölümü'nü kazandı. 2001 yılında KTÜ Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Ana Bilim Dalında Yüksek Lisans eğitimine başladı. Aynı yıl Bilgisayar Mühendisliği Bölümü'ne Araş. Gör. alarak atandı. 2002 yılında Öğr. Gör. oldu. Halen bu görevine devam etmektedir. Yabancı dil olarak İngilizce bilmektedir.