|  |  |  |
| --- | --- | --- |
|  | **KARADENİZ TEKNİK ÜNİVERSİTESİ**  **BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**  **BİLGİSAYAR GRAFİKLERİ-I DERS NOTLARI** | ktulogotr_01.jpg |

**IŞIN İZLEME (RAY TRACING)**

**1.Giriş**

Işın İzleme (Ray Tracing) yöntemi, bakış noktasından çıkan ve görüntü düzlemindeki piksellerin herbirinden geçen ışınlar ile 3D ortamdaki cisimler arasında yapılan kesişim testleri sonucu belirlenen en yakın cismin (eğer bu cisim ışını yansıtma özelliğine sahipse veya saydam olup ışın içinden doğrudan veya kırılarak geçiyorsa ondan yansıyıp/kırılıp çarptığı ilk cismin rengi de hesaba katılarak son) renginin bir boyama modeli (örneğin Phong boyama modeli) ile belirlenmesi olarak tanımlanabilir.

Işın vektörel bir büyüklük olduğu için öncelikle vektörel işlemler hakkında bilgiler verilecek ve ardından ilerleyen bölümlerde ışın izleme yöntemi detaylı olarak incelenecektir.

**2. Vektörel İşlemlerle İlgili Temeller**

Herhangi bir vektörün boyunu bulmak için (x,y,z) koordinatlarının karelerinin toplamının karekökü alınır. Örneğin **R=(0,6,8)** vektörünün boyu **|R|**= .

Herhangi bir vektörün boyunu **1** birim yapma işlemine “normalizasyon” denir ve işlem için (x,y,z) koordinatlarının her biri vektörün boyuna bölünür. Yukarıdaki **(0,6,8)** vektörü normalize edildiğinde bulunur. Normalize edilmiş vektörün boyu hesaplandığında olduğu görülür. Normalize edilmiş vektöre “birim vektör” denir. (Vektörel büyüklükler **koyu**, skaler büyüklükler normal font ile yazılacaktır).

Vektörler arasında vektörel ve skaler olmak üzere iki temel çarpım işlemi vardır. **R1** ve **R2** gibi iki vektörün sırasıyla vektörel ve skaler çarpımı aşağıdaki gibi yapılır:

**R1 x R2 = ( R1yR2z-R1zR2y, R1zR2x- R1xR2z, R1xR2y- R1yR2x )**

**R1 \* R2 = R1xR2x + R1yR2y + R1zR2z = |R1|\*|R2|\*Cos(β)**

Vektörel çarpım yüzey normalinin hesaplanmasında kullanılır. Yüzey normali yüzeye dik olan vektördür.

İki vektörün skaler çarpımında (x,y,z) koordinatları çarpılıp toplanır veya vektörlerin boylarının aralarındaki açının kosinüsüyle çarpımı olarak da hesaplanabilir. Dolayısıyla birim vektörlerin skaler çarpımı aralarındaki açının kosinüsünü verir. Skaler çarpım Phong boyamada diffuse ve specular renk bileşenlerinin hesaplanmasında kullanılır. Vektörel çarpım vektörel; skaler çarpım skaler bir değer döndürür.

Köşe noktalarının koordinatları **V0,V1,V2** şeklinde verilen üçgenin yüzey normali **N** vektörü, **(V1-V0)** ve **(V2-V0)** vektörlerinin vektörel çarpımı ile şöyle hesaplanır:

**V0=(0,40,120) V1=(30,-40,60) V2=(-30,-40,60)**

**(V1-V0)=(30,-80,-60) (V2-V0)=(-30,-80,-60)**

**N=(V1-V0)x(V2-V0)=(-80\*-60 - -60\*-80, -60\*-30 - 30\*-60, 30\*-80 - -80\*-30)**

**N = ( 0, 3600, -4800)**

**N** vektörü normalize edilirse **(0,0.6,-0.8)** elde edilir.

**3. Işının Tanımı ve Birincil Işınların Üretilmesi**

Başlangıç noktası ve doğrultuya sahip vektörel bir büyüklük olan **R** ışını:

**R = Ro + tRd**

olarak ifade edilir. Burada **Ro** ışının başlangıç noktası (**o**rigin), **Rd**de doğrultusudur (**d**irection). t ise ışının 3D uzayda **Rd** doğrultusu boyunca kaç birim gideceğini belirleyen skaler bir değerdir. Doğrultu vektörü **Rd** ‘nin hesaplanabilmesi için 2 noktaya ihtiyaç vardır. Bunlar **R1** ve **R2** olarak alınırsa **R1**’den **R2**’ye doğru olan doğrultu vektörü **Rd=R2-R1** ile hesaplanır. Işın izlemede doğrultu vektörleri birim vektör olmalıdır. Dolayısıyla **Rd** normalize edilerek boyu 1 birim yapılır.

Işın izlemede ilk adım başlangıç noktasından çıkıp 3D görüntü düzlemindeki piksellerin herbirinden geçecek olan birincil ışınların doğrultularının belirlenmesi işlemidir. Bunun için piksel koordinatlarından başlangıç noktasının koordinatı çıkarılır. Ardından normalize edilerek doğrultunun boyu 1 birim yapılır. Işınlar ile 3D ortamdaki cisimler arasında kesişim testleri yapılarak görüntü düzlemine hangi cismin şeklinin çizileceği belirlenir.

Şekil 1’den görüldüğü gibi ışınların 3D görüntü düzleminden geçtikleri pikseller ile en son ekranda üretilen görüntüdeki pikseller farklı koordinat sistemlerini kullanmaktadır. Örneğin bilgisayar ekranındaki 800x450 çözünürlükteki bir görüntünün (x,y) koordinatları sol üst köşede (0,0) sağ alt köşede de (799,449)’dur. Aynı çözünülürlükteki bir görüntü düzleminin sol üst köşesinin koordinatları (-400,224,500) olmalıdır (görüntü düzleminin bakış noktasına uzaklığı 500 birim alınmıştır). Dolayısıyla ekrandaki herhangi bir pikselin görüntü düzlemindeki karşılığını bulmak için bir dönüşüm yapmak gerekir. Ekrandaki 800x450 çözünürlüğünde bir görüntünün (x,y) koodinatlarının 3D Görüntü Düzlemi **GD(X,Y,Z)**’deki karşılığı aşağıdaki ifade ile bulunur:

**GD(X,Y,Z)** = ( x - 399, 224 - y, 500)

Yukarıdaki dönüşümün pratikte kullanılması pek tercih edilmez. Çünkü çizilecek görüntünün çözünürlüğü değiştirilmek istenildiğinde sadece 399, 224 gibi çözünürlüğe bağlı değerleri değiştirmek yeterli olmaz. Görüntü Düzlemine uzaklığın da değiştirilmesi gerekir. Dolayısıyla yalnızca çözünürlüğü değiştirerek dönüşüm imkanı sağlayan aşağıdaki ifade kullanılmalıdır:

**GD(X,Y,Z)** = ( 16\*x/799 - 8, 4.5 - y\*9/449, 10)

Burada görüntü düzlemi (16x9) boyunda seçilmiştir. Üretilecek görüntünün çözünürlüğü 800x450’den farklı mesela 1600x900 olduğunda sadece 799’u 1599 ve 449’u 899 yapmak yeterlidir.

**Şekil 1:** 2D Ekran ve 3D Görüntü Düzlemi Arasındaki İlişki

**4. Işın-Üçgen Kesişim Testi**

3D cisimler çoğunlukla üçgenler ile temsil edilirler. Burada anlatılacak olan ışın-üçgen kesişim testi iki aşamadan oluşmaktadır:

* Işın ile üçgenin tanımladığı yüzey arasında kesişim testi.
* Işın yüzey ile kesişiyorsa kesişim noktasının üçgenin içinde olup olmadığını belirleme.

Birinci aşama için üçgenin tanımladığı yüzeyin denklemini çıkarmak gerekmedir. Bilindiği gibi yüzey denklemi Ax+By+Cz+D=0 ‘dır. Burada **(A,B,C)** yüzey normalidir. Yukarıda **V0,V1,V2** şeklinde verilen üçgenin yüzey denklemini çıkaralım:

Daha önce üçgenin normali **N=(0,0.6,-0.8)** olarak hesaplanmıştı. Dolayısıyla **N=(A,B,C)** biliniyor. Yüzeyin üzerinde olduğu için yüzey denklemini sağlayacağından üçgenin köşe noktalarından herhangi biri D ‘nin hesabı için kullanılabilir. Dolayısıyla Ax+By+Cz+D=0 ‘daki **(x,y,z)** yerine köşe noktalarından herhangi birinin mesela **V0** ‘ın **(x,y,z)** ‘sini yazıp sıfıra eşitlersek D ‘yi :

Ax + By + Cz + D = 0

0\*0 + 0.6\*40 + -0.8\*120 + D = 0

D = 72

olarak buluruz. Dolayısıyla yüzey denklemi 0.6y - 0.8z + 72 = 0 ‘dır.



**Şekil 2:** Alan Testi

Işın yüzey ile kesişiyorsa üçgenin köşe noktaları gibi ışının yüzey üzerindeki koordinatları da yüzey denklemini sağlamalıdır. Dolayısıyla şöyle yazabiliriz:

A(**Rox**+t**Rdx**)+B(**Roy**+t**Rdy**)+C(**Roz**+t**Rdz**)+D=0

Yukarıdaki denklem t ‘ye göre düzenlenirse:

t>0 ise ışın yüzey ile kesişiyor demektir.

t<0 ise ışın yüzey ile kesişmiyor demektir.

t=∞ ise yani ise ışın yüzeye paralel demektir.

Örnek olarak **Ro=(0,0,0)** başlangıç noktasından **Rd=(0,0,1)** doğrultusu boyunca giden **R** ışınının bu yüzey ile kesişip kesişmediğini t hesabı ile belirleyelim:

Işının yüzey üzerindeki koordinatları **R=Ro+**t**Rd=(0,0,0)+**90**(0,0,1)=(0,0,90)** olarak bulunur. Böylece kesişim testinin I. Aşaması tamamlanmış oldu.

II. aşamada **(0,0,90)** noktasının üçgenin içinde olup olmadığına karar verilmelidir. Bunun için değişik yöntemler denenebilir. Burada “alan testi” yöntemi kullanılacaktır. Buna göre Şekil 2‘den de görüldüğü gibi yukarıda hesaplanan kesişim noktasından üçgenin köşelerine doğrular çizerek 3 alt üçgen oluşturulur. Bu alt üçgenlerin alanları toplamı büyük üçgenin alanına eşitse kesişim noktası üçgenin içinde demektir.

**V0,V1,V2** üçgeninin alanı |**(V1-V0)x(V2-V0)**|ile hesaplanabilir. **R=(0,0,90)** noktası kullanılarak oluşturulan s1, s2 ve s3 alt üçgenlerin alanları ve **V0,V1,V2** üçgeninin S alanını hesaplandığında S=6000, s1=1500, s2=1500 ve s3=3000 çıkar. S=s1+s2+s3 olduğundan kesişim noktası üçgenin içindedir.

Işın-Üçgen kesişim testi yapan Intersect() adlı method aşağıdadır:

|  |
| --- |
| float Intersect(Vertex Ro, Vertex Rd)  {  Vertex normal;  Vertex R;  float S, s1, s2, s3;  normal = (V1-V0).CrossProduct(V2-V0);  float D = -(V0 \* normal);  float t = -(Ro \* normal + D) / (normal \* Rd);  if (t > 0)  {  R = Ro + t \* Rd;  S = (V1 - V0).CrossProduct(V2 - V0).Length();  s1 = ( R - V0).CrossProduct(V2 - V0).Length();  s2 = (V1 - V0).CrossProduct( R - V0).Length();  s3 = (V1 - R).CrossProduct(V2 - R).Length();  float fark = (float)Math::Abs(S - (s1 + s2 + s3));  float epsilon = 0.005F;  if (fark <= epsilon) return t; else return 0;  }  else return 0;  } |

**5. Görünmeyen Yüzeylerin Kaldırılması (Hidden Surface Removal)**

Görüntü düzleminden geçen ışınlardan herhangi biri gittiği doğrultu boyunca 1’den fazla cisimle kesişiyorsa ekrana bunlardan görüntü düzlemine en yakın olanı çizilmelidir. Bunun için kesişim testleri ile hesaplanan t uzaklıkları sıralanır ve en küçük t uzaklığına sahip cismin görüntüsü çizilir. Diğer cisimler görünmeyen yüzey olarak adlandırılırlar.

**U0( 0, 30, 40) U1(40, -30, 120) U2 (-40, -30, 120)**

**Y0(-50, 30, 124) Y1(50, 30, 124) Y2 ( 0, -30, 44)**

**Z0(-30, 0, 37) Z1(30, 40, 117) Z2 ( 30, -40, 117)**

Yukarıda sırasıyla kırmızı, yeşil ve mavi renklere sahip **U0,U1,U2** üçgeni **Y0,Y1,Y2** üçgeni ve **Z0,Z1,Z2** üçgeninin köşe noktalarının koordinatları verilmiştir. Başlangıç noktası **Ro=(0,0,0)** ‘dan çıkan ve görüntü düzleminde **(0,0,5)** noktasındaki pikselden geçen ışın ile bu üçgenler arasında ışın-üçgen kesişim testi yöntemine göre tU=80, tY=84, tZ=77 uzaklık değerleri hesaplanır. Uzaklıklar sıralandığında mavi renkli Z üçgeninin bakış noktasına daha yakın olduğu görülür. Dolayısıyla ışının geçtiği piksel mavi renge boyanır. Kırmızı renkli U ve yeşil renkli Y üçgenleri görüntü düzleminde **(0,0,5)** noktasındaki pikselden görünmeyen üçgenlerdir.

**Şekil 3:** Işın-Küre Kesişim Testi

**6. Işın-Küre Kesişim Testi**

Kesişim testine ışının başlangıç noktası **Ro**’dan kürenin merkezine **l=c–Ro** vektörü hesaplanarak başlanır. Sonra üç tane skaler değer s**=l\*Rd**, l2**=l\*l** ve r2=r\*r şeklinde hesaplanır. Burada s, **l** vektörünün boyunun **Rd** üzerine izdüşümüdür. Çünkü **Rd** birim vektör olduğundan **l\*Rd** aynı zamanda **|l|\*Cos(β)** demektir. l2, **l** vektörünün r2 de yarıçapın boyunun karesidir. Eğer (s<0 && l2>r2) şartı sağlanıyorsa Şekil 3-(a)‘daki durum söz konusudur yani kesişim yok demektir. s<0 olsa bile l2<r2 ise yani ışının başlangıç noktası kürenin içinde ise Şekil 3-(b) ‘deki gibi her zaman kesişim vardır. s>0 iken hangi durumlarda kesişim olduğunu belirlemek üzere m2=l2-s2 hesaplanır. m2>r2 ise Şekil 3-(c)‘deki gibi kesişim yoktur. m2<=r2 ise kesişim vardır ve l2<r2 ise ışın Şekil 3-(d)’deki gibi içten, l2>r2 ise de Şekil 3-(e)‘deki gibi dıştan küreyle kesişiyor demektir. q=sqrt(r2-m2) hesaplanır. Dıştan kesiştiğinde ışının küreye uzaklığı t=s–q ile içten kesiştiğinde t=s+q ile hesaplanır. Işın-küre kesişim fonksiyonu Intersect() aşağıda verilmiştir:

float Intersect(Vertex **Ro**, Vertex **Rd**)

{

Vertex **l** = **Center** - **Ro**;

float s = **l** \* **Rd**;

float l2 = **l** \* **l**;

float r2 = Radius \* Radius;

if (s < 0 && l2 > r2) return 0;

float s2 = s \* s;

float m2 = l2 - s2;

if (m2 > r2) return 0;

float q = (float)Math::Sqrt(r2 - m2);

if (l2 > r2) return s - q;

else return s + q;

}

**7. Phong Boyama Modeli**

Işınlar ile üçgenler/kürelerle kesişim testleri yapılıp görüntü düzlemine en yakın olan üçgen/küre belirlendiğinde bu üçgenin/kürenin rengi doğrudan ilgili piksele setlenmez. Çünkü ışık kaynağının o noktayı ne oranda aydınlattığı dikkate alınmalıdır. Phong boyama modeline göre ışık kaynağına bağlı olarak ambient, diffuse ve specular olmak üzere üç renk bileşeni, toplamları 1 olan katsayılarla çarpılıp toplanarak pikselin son rengi belirlenir.

**7.1. Diffuse Renk Bileşeni**

Diffuse katsayı Şekil 4-(a)’daki gibi yüzey normali ile ışık kaynağına doğru olan vektör skaler çarpılarak belirlenir. Her ikisinin de birim vektör olduğu varsayıldığında skaler çarpımları aralarındaki açının kosinüsünü verir. Negatif değerler için yüzey ışık kaynağı tarafından aydınlatılmıyor demektir. Işık kaynağı yüzeye tam dik vuruyorsa yani yüzey normali ile aralarındaki açı 0 derece ise Cos(0)=1 olduğundan maksimum aydınlatma; açı 90 derece ise de Cos(90)=0 minimum (sıfır) aydınlatma söz konusudur. Hesaplanan diffuse katsayı ile yüzeyin rengi çarpılarak Phong boyama modelinin diffuse renk bileşeni belirlenir.

Işık kaynağının koordinatları **(0,40,0)** olduğu durumda yüzey normali **(0,1,0)** olan kırmızı renkli bir üçgen üzerindeki **(0,0,30)** noktasının diffuse renk bileşenini hesaplayalım:

Işık kaynağına doğru olan vektör **(0,40,0)-(0,0,30)=(0,40,-30)**‘dur. Normalize edildiğinde **(0,0.8,-0.6)** olur. Diffuse katsayı **(0,0.8,-0.6)\*(0,1,0)=**0.8 çıkar. Kırmızı renk (255,0,0), hesaplanan diffuse katsayı ile çarpılırsa diffuse renk bileşeni (204,0,0) olarak bulunur.

**7.2. Specular Renk Bileşeni**

Specular katsayı Şekil 4-(b)’deki gibi ışık kaynağından yüzeye doğru olan vektörün yüzey normalinden yansıma vektörü ile bakış noktasına doğru olan vektör arasındaki açının belli bir skaler değer kadar kuvveti alınarak hesaplanır. Hesaplanan specular katsayı ile ışık kaynağının rengi çarpılarak Phong boyama modelinin specular renk bileşeni belirlenir.



**Şekil 4:** Diffuse ve Specular Katsayı Hesabı

Şekil 4-(b)’deki specular katsayı hesabındaki **reflected** yansıma vektörünün hesaplanışı Şekil 5’te gösterilmiştir. Burada gelen ışının doğrultusu **Rd**, yüzey normali **N**, olduğu durumda yansıma vektörü **Rd-2Rd\*N\*N** ile hesaplanır.

**Şekil 5:**  Yansıma Vektörü Hesabı

**7.3. Ambient Renk Bileşeni**

Bir odada masa sehpa gibi eşyaların altları gibi ışık kaynağı tarafından doğrudan aydınlatılmayan yüzeyler vardır. Buralara duvarlardan veya diğer eşyalardan yansıyan ışıkların vurması nedeniyle az da olsa aydınlatılmaktadırlar. Phong modelinin ambient renk bileşeni bu dolaylı aydınlatmayı modeller ve cismin kendi rengi 0..1 arası bir katsayı ile çarpılarak ambient renk bileşeni hesaplanır.

**8. Yansıma (Reflection), Saydamlık (Transparency) ve Kırılma (Refraction)**

Işını yansıtan yüzeyler için yansıyan ışının doğrultusunun (reflected direction) **Rd-2Rd\*N\*N** ile hesaplanacağından specular bileşene ait **reflected** vektörü hesaplanırken bahsedilmişti. Işını yansıtma özelliğine sahip cisim için Şekil-5’teki gibi yansıma doğrultusu hesaplandıktan sonra yeni doğrultu boyunca yollanan ışın ile cisimler üzerinde kesişim testleri yapılarak en yakın cisim belirlenir. Onun rengi 0..1 arası bir katsayı ile çarpılarak yansıma ile görülen cismin rengi hesaplanır.

Saydam cisimlerde ışın cismin içinden geçerken doğrultusu değişmez. Yani cismin içinden geçen ışının doğrultusu (transmitted direction) gelen ışının doğrultusu **Rd** ile aynı alınıp yukarıdaki işlemler tekrarlanır.

Işın saydam bir küreden geçerken (ışının başlangış noktasının kürenin dışında olduğu varsayıldığından) küreyle önce dıştan sonra da içten kesişerek yoluna devam eder. Işın küreyle dıştan kesiştikten sonra aynı doğrultu boyunca içten kesişecek şekilde tekrar yollandığında ışın-küre kesişim testi t uzaklığı hesabında:

if (l2 > r2) return s - q;

else return s + q;

ışının başlangıç noktası küre üzerinde yani l2 = r2 olduğundan normalde else kısmının koşması beklenir ama ışın-üçgen alan testindeki gibi l2 ve r2 float sayılar olduğundan çoğu zaman birebir aynı olmazlar. Bazen l2>r2 bile olabilir ve t=s-q olarak hatalı hesaplanacağından küre üzerinde gürültülere sebep olur. Yukarıdaki kodun else kısmının koşmasını garanti etmek için if (l2 > r2 + 0.001) olarak güncellenir. if ((int)l2 > (int)r2) de aynı işlevi görür.

Kırılma (Refraction) ile ilgili belgeye [buradan](http://ceng2.ktu.edu.tr/~cakir/files/grafikler/Refraction.pdf) erişebilirsiniz.

**9. Gölge Testi**

Herhangi bir yüzeyin başka bir yüzeyin gölgesinde kalıp kalmadığını belirlemek için Şekil 6’da gösterildiği gibi yüzey üzerindeki kesişim noktasından ışık kaynağına doğru gölge test etme ışını yollanır. Bu ışın ile üçgenler/küreler arasında kesişim testleri yapılarak en yakın üçgen/küre belirlenir. Hesaplanan uzaklık değeri ışık kaynağına olan uzaklıktan küçük ise yüzey bu üçgenin/kürenin gölgesinde kalıyor demektir.



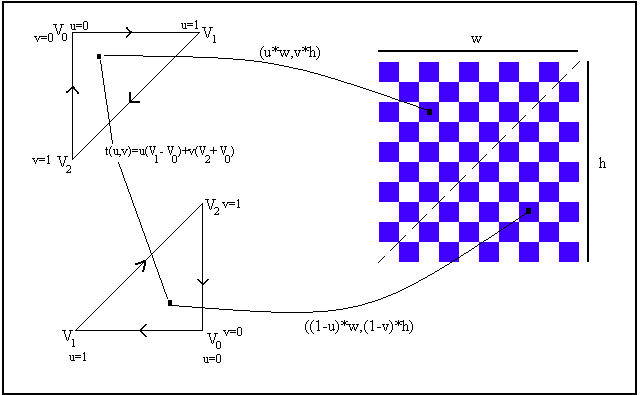
**Şekil 6 :** Gölge Test Etme Işını

**10. Düzlemsel Yüzey Üzerine Doku Kaplama**

Işın izleme ile doku kaplama yapmak için Tomas Möler’in ışın-üçgen kesişim testinden faydalanılmıştır. Möller’in yöntemine göre kesişim testi yapıldığında yalnızca kesişim noktasına uzaklık olan t değeri değil aynı zamanda kesişim noktasının barisentrik koordinatları da hesaplanabilir. Barisentrik koordinatlar ve üçgenin köşe noktaları kullanılarak üçgen üzerinde herhangi bir noktaya gidebilmek için aşağıdaki ifade kullanılır:

t(u,v) = V0 + u(V1-V0) + v(V2-V0)

Üçgenin köşe noktaları uygun sırada tutulursa barisentrik koordinatlarla doku kaplama için kullanılan resim dosyasındaki doğru piksel koordinatları rahatlıkla bulunabilir. Bunun için bazı varsayımlar yapılmıştır. Birincisi üzerine doku kaplama yapılacak yüzey düzlemseldir. İkincisi yüzey iki dik üçgenden oluşmaktadır. Köşe noktalarının dizilişi V0, V1 ve V2 olmalıdır ve dik kenarları birleştiren köşe noktası V0 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 7‘de barisentrik koordinatlarla doku kaplamanın nasıl yapıldığı gösterilmektedir.



Şekil 7: Barisentrik koordinatlarla doku kaplama

11. Arkayüz Kaldırma (Backface Culling)

Arkayüz olan yüzey (veya üçgen) bakış noktası ile arasında başka yüzeyler olmasa bile ters durduğu için görünmeyen yüzeydir. Yüzeyin veya üçgenin ters durması ne demektir? Y=0 yüzeyi üzerinde aşağıda köşe noktaları **U0,U1,U2** olarak verilmiş üçgeni düşünelim.

**U0( 0, 0, 40) U1(40, 0, -40) U2 (-40, 0, -40)**

Bu üçgenin +Y ve –Y eksenlerine bakan iki farklı yüzü vardır. Üçgenin yüzey normali hesaplanırsa **(0, 6400, 0)** bulunur. Normalize edilirse **(0,1,0)** olur. Şimdi köşe noktalarının koordinatlarını farklı sırada yazalım:

**U0(40, 0, -40) U1( 0, 0, 40) U2 (-40, 0, -40)**

Tekrar yüzey normali hesaplanırsa **(0,-6400,0)** bulunur. Normalize edilirse bu sefer **(0,-1,0)** olur. İlk hesaplanan yüzey normali **(0,1,0)** +Y eksenine doğru şimdiki ise **(0,-1,0)** –Y eksenine doğru çıktı. Dolayısıyla köşe noktalarının sırası değişince yüzey normalinin doğrultusu da değişmektedir. 3D kartezyen koordinat sisteminde Z ekseni (0,0,0) ‘dan ileriye doğru pozitif artıyor ise (+Z) köşe noktaları saat yönünde (ClockWise-CW) negatif olarak artıyor ise (-Z) saat yönünün tersi sırada (CounterClockWise-CCW) tanımlanmalıdır. Bu kurallara sırasıyla sol el ve sağ el kuralı denir. Sol el kuralında sol elin 4 uzun parmağı +X eksenini gösterirken +Y eksenini gösterecek şekilde katlandığında baş parmağın doğrultusu +Z eksenini gösterir. Aynı işlem sağ el ile yapıldığında baş parmak yine +Z ‘i gösterir. DirectX 3D kartezyen koordinatları sol el; OpenGL de sağ el kuralına göre belirler.

Yüzey normalinin doğrultusunun köşe noktalarının sırasına bağlı olarak değişmesi ile arkayüz olması arasında ne ilişki var? 3D cisimlerin görüntüleri çizilirken onların renkleri doğrudan ekrana basılmaz. Son renk değeri belirlenirken ışık kaynağının konumu da dikkate alınır. O cisim ışık kaynağından az miktarda veya çok miktarda ışık almasına bağlı olarak rengi belli katsayılarla çarpılarak değiştirilip ekrana basılır. Bu katsayılardan biri diffuse diğeri de specular katsayılardır. Her ikisi de belirlenirken yüzey normali kullanılır. Örneğin diffuse katsayı yüzey normali ile bakış noktasına doğru olan vektörlerin skaler çarpımı ile bulunur.

Yukarıda ilk tanımlanan U üçgeni içindeki **(0,0,0)** noktasının diffuse katsayısını hesaplayalım. Bunun için ışık kaynağının koordinatları bilinmelidir. Işık kaynağı **(0,60,80)** noktasında olsun. Işık kaynağına doğru olan vektör **(0,60,80)-(0,0,0)=(0,60,80)** olur. Normalize edilirse **(0,0.6,0.8)** bulunur. Diffuse kaysayıyı bulmak için bu vektör ile yüzey normali skaler çarpılırsa 0.6 bulunur. Aynı işlemler köşe noktalarının sırası değiştirilerek normali **(0,-1,0)** olan üçgen için yapılırsa bu sefer -0.6 bulunur. Renk değerinin negatif olması imkansız olduğundan -0.6 diffuse katsayı olarak kullanılamaz. Başka bir deyişle bu yüzey ışık kaynağı tarafından aydınlatılmıyor demektir. Demek ki yüzey normalinin doğrultusuna bağlı olarak yüzey ışık alır veya almaz. Benzeri şekilde yüzey normali ile bakış noktasına doğru olan vektörler skaler çarpılınca sonuç negatif çıkıyorsa yüzeyin bakış noktasına göre görülmesi imkansız demektir. Yani arkayüzdür. Böylece arkayüz belirlemede kullanılan birinci yöntem açıklanmış oldu. Demek ki herhangi bir yüzeyin arkayüz olup olmadığını belirlemek için o yüzey üzerindeki herhangi bir noktadan (üçgen için köşe noktalarından herhangi biri) bakış noktasını doğru olan vektör ile yüzey normali skaler çarpılır. Sonuç sıfırdan küçükse bu yüzey arkayüzdür. Mesela yukarıda ikinci olarak tanımlanan U üçgeni bakış noktası **(0,60,80)** alındığında arkayüz olmaktadır. Bakış noktasına doğru olan vektör ile yüzey normalini **skaler çarparak arkayüz kaldırma** Şekil 8’de gösterilmiştir.

|  |  |
| --- | --- |
|  |  |
| **Şekil 8:** Skaler çarpımla arkayüz kaldırma | **Şekil 9:** Vektörel çarpımla arkayüz kaldırma |

Arkayüz kaldırmada kullanılan diğer bir yöntemde üçgenin görüntü düzlemine izdüşümü alınır ve normali hesaplanır. Eğer normalin Z bileşeni 0’dan büyük ise arkayüzdür. İzdüşüm sonrası vektörel çarpımla normal hesaplandığından bu yöntem **vektörel çarpımla arkayüz kaldırma** olarak isimlendirilecektir. Vektörel çarpımla arkayüz kaldırma Şekil 9’da gösterilmiştir. Herhangi bir noktanın benzer üçgenler yardımıyla görüntü düzlemine perspektif dönüşüm ile izdüşümünün nasıl yapıldığı Şekil 10’da gösterilmiştir.



**Şekil 10:** Perspektif Dönüşüm ile İzdüşüm

12. Etkileşimli Işın İzleme

Etkileşimli Işın İzleme ile ilgili belgeye [buradan](http://ceng2.ktu.edu.tr/~cakir/files/grafikler/Interactive_Ray_Tracing.pdf) erişebilirsiniz.