Ege Eğitim Dergisi 2002 (1), 2: 68-75

ÜÇ BOYUTLU GÖRÜNTÜLEMEDE 2D ve 3D API KULLANIMI

Aybars UĞUR*

ÖZET

Bu makalede ilk olarak üç boyutlu grafik programlamada yüksek düzeyli üç boyutlu uygulama geliştirme arayüzlerinin önemi anlatılmaktadır. Bu amaçla Java programlama dilinde, üç boyutlu etkileşimli bir küp nesnesi içeren bir program yapılmıştır. Aynı program Java 3D API kullanılarak bir kere daha gerçekleştirilmiştir. Her iki applet, applet'in yazılması sırasında harcanan çaba, kaynak kod satır sayısı ve işletim hızı gibi değişik ölçütler kullanılarak ölçülmüşlerdir. Sonuçta üç boyutlu grafik programlamada ve üç boyutlu görüntülemede ölçümlerden elde edilen değerlerin yardımıyla her iki yöntemin karşılaştırılması yapılmıştır.

SUMMARY

This paper describes the importance of high level three dimensional application programming interfaces in three dimensional graphics programming. Simple three dimensional interactive cube is implemented in Java programming language by using Java 2D API. After that, same program is implemented by using Java 3D API. These applets were measured by different views : such as effort required to build applet, lines of code (LOC) produced and execution speed. As a result, two methods in 3D graphics programming and viewing are compared by using values obtained from measurements.

Anahtar Kelimeler : Grafik programlama, 2D API, Java 2D, 3D API, Java 3D

1. Üç Boyutlu Nesnelerin Bilgisayarda Temsili

Günümüzde bilgisayar donanım ve yazılımlarının gelişmesi ve bilgisayar kullanıcılarının sayısının artması ile üç boyutlu bilgisayar grafikleri de daha çok kullanılır olmuştur. Her geçen gün e-ticaret alanından eğitim alanına sanal gerçeklikten bilimsel görselleştirmeye kadar birçok alanda üç boyutlu bilgisayar grafiklerinin kullanımı giderek yaygınlaşmaktadır. Kişisel bilgisayar ortamında bu tür programların yanında özellikle bilgisayar oyunları da üç boyut teknolojisinin ve donanımlarının geliştirilmesinde itici güç haline gelmiştir.

Grafiksel sahnelerde binalar, insanlar, taşıtlar, eşyalar, ağaçlar, çiçekler, hayvanlar, bulutlar, taşlar, tuğlalar, tahtalar, plastikler, kağıtlar, mermerler, çelikler, camlar, kumaşlar, lastikler, sular ve diğer sıvılar gibi değişik birçok nesne bulunabilir [4]. Farklı madde özelliklerine sahip nesneleri bilgisayarda tanımlamak için birden çok gösterim şekli vardır. Daha gerçekçi görüntüler oluşturmak için bu nesne özelliklerini doğru bir şekilde modellemek gerekir. Çokgenler ve ikinci dereceden yüzeyler, çokyüzlüler ve elipsoidlerin oluşturulmasında kullanılırlar ve kesin tanımları vardır. Spline yüzeyleri ve yapılandırma teknikleri uçakların kanat ve motorlarının ve eğrisel yüzeyler içeren diğer mühendislik yapılarının tasarımında kullanılırlar. Fraktal yapıları ve madde sistemleri bulut, ot gibi doğal nesnelerin doğru bir şekilde temsil edilebilmesini sağlarlar. Etkiyen kuvvet sistemlerini kullanan fiziksel tabanlı modelleme yöntemleri bir kumaş parçasının veya bir pelte küresinin katı olmayan hareketini tanımlar. "Octree" kodlamaları nesnelerin tıbbi görüntülerde olduğu gibi içsel görüntülerinin gösteriminde kullanılır. Bunlar dışında birçok üç boyutlu görüntüleme tekniği vardır. [4]

* Ege Üniversitesi, Bilgisayar Mühendisliği Bölümü, Bornova-İZMİR
e-posta : ugur@bornova.ege.edu.tr

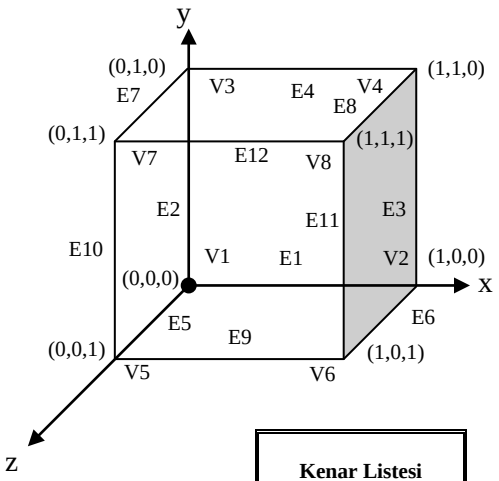
Katı nesnelerin gösterim düzenleri iki çeşittir: sınır gösterimleri (boundary representations) ve uzay bölümleme gösterimleri. Sınır gösterimi nesneyi, nesnenin içini dışından ayırmayı sağlayan yüzeyler dizisi şeklinde tanımlar. Çokgen yüzeyleri ve "Spline" parçalarını kullanır. Uzay bölümleme gösterimleri "Octree" gösteriminde olduğu gibi, nesneyi küçük katı parçalarına ayırarak tanımlar. Üç boyutlu değişik nesnelerin tanımlanmasında önemli bazı gösterim yöntemleri şunlardır :

- Çokgen yüzey gösterimleri
- Eğriler ve eğrisel yüzey gösterimleri
- "Quadric" yüzey (küre, elipsoid, torus...) gösterimleri
- "Superquadrics" (süperelips, süperelipsoid)
- "Blobby" yüzey gösterimleri
- "Spline" gösterimler
- "Bezier" eğri ve yüzey gösterimleri
- "Sweep" gösterimler
- CSG (Constructive Solid Geometry = Katı Geometri Yapılandırma) Yöntemleri

Üç boyutlu grafik programları bu tür gösterim şekillerinin bir veya daha fazlasını kullanmaktadırlar. Üç boyutlu görüntülemenin anlatımından önce üç boyutlu nesnelerin bilgisayarda tanımlanmasına ilişkin yöntemlerden çokgen yüzeyli nesnelerin sınır gösterimi üzerinde duracağım. Diğer tür nesnelere ve nesnelerin tanımlanmasına ilişkin yöntemler de ayrıca araştırma konuları olmakla birlikte makalenin odaklandığı konu üç boyutlu görüntülemede iki ve üç boyutlu uygulama geliştirme arayüzlerinin kullanımının karşılaştırılması olduğu için isimleri verilerek geçilmiştir.

2. Çokgen Yüzeyli Nesnelerin Bilgisayarda Tanımlanması

Yüzeyleri çokgen olan üç boyutlu nesneler bilgisayarda köşe, kenar ve yüzey bilgileri ile tanımlanırlar. Bu bilgileri tutmak için köşe listesi, kenar listesi, çokgen yüzey listesi oluşturulur. Bu listelerin oluşturulmasında statik veya dinamik veri yapıları kullanılabilir. Köşe listesi, köşelerle birlikte köşelerin üç boyutlu koordinatlarını içerir. Kenar listesi, kenarları ve kenarları oluşturan köşeleri tutar. Çokgen yüzey listesi ise yüzeyleri oluşturan kenarları tutar. Basit bir küp şeklinin bilgisayarda tanımlanması için 8 köşenin koordinatlarının, 12 kenarı oluşturan köşelerin ve 6 yüzeyi oluşturan kenarların bilgilerinin tutulması gerekmektedir (Şekil 1). Bazı durumlarda ise kenar listesi kullanılmadan yüzey bilgileri doğrudan köşe bilgilerinden oluşturulur. Bu durumlarda sadece köşe listesi ve çokgen yüzey listesinin tutulması yeterlidir.



Köşe Listesi	
Köşe No	**Koordinatlar**
V1	(0,0,0)
V2	(1,0,0)
V3	(0,1,0)
V4	(1,1,0)
V5	(0,0,1)
V6	(1,0,1)
V7	(0,1,1)
V8	(1,1,1)

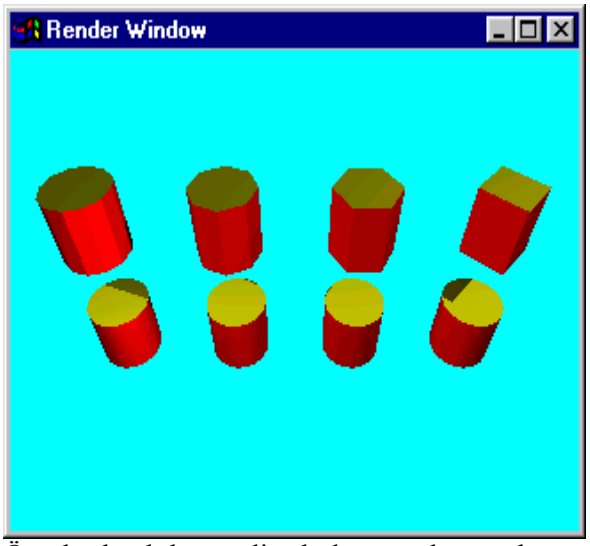
Kenar Listesi	
Kenar No	**Köşeler**
E1	(V1,V2)
E2	(V1,V3)
E3	(V2,V4)
E4	(V3,V4)
E5	(V1,V5)
E6	(V2,V6)
E7	(V3,V7)
E8	(V4,V8)
E9	(V5,V6)
E10	(V5,V7)
E11	(V6,V8)
E12	(V7,V8)

Çokgen Yüzey Listesi	
Yüzey No	**Kenarlar**
S1	E1,E2,E4,E3
S2	E1,E6,E9,E5
S3	E2,E5,E10,E7
S4	E3,E8,E11,E6
S5	E4,E7,E12,E8
S6	E9,E11,E12,E10

Arka Yüzey : S1
Alt Yüzey : S2
Sol Yan Yüzey : S3
Sağ Yan Yüzey : S4
Üst Yüzey : S5
Ön Yüzey : S6

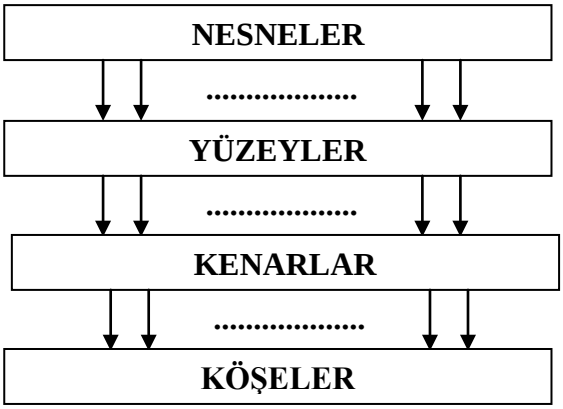
Çokgen Yüzey Listesi	
Yüzey No	**Kenarlar**
S1	E1,E2,E4,E3
S2	E1,E6,E9,E5
S3	E2,E5,E10,E7
S4	E3,E8,E11,E6
S5	E4,E7,E12,E8
S6	E9,E11, E12,E10

Şekil 1 : Kenar uzunluğu 1 birim olan bir küpün köşe listesi, kenar listesi ve çokgen yüzey listesi yardımı ile kenar tabanlı sınır temsili.



Örnek olarak bu tür listelerle tanımlanmış küp gibi üç boyutlu bir nesne ekranda gösterileceği zaman tüm yüzeyler dolaşılarak sıra ile her bir yüzey çokgen çizdiren komutlarla ekrana çizdirilir. Bir yüzeyi oluşturan kenar ve köşe bilgilerine çokgen yüzey listesi yardımı ile ulaşılır. Bu tür bilgilerin kullanılması ile üç boyutlu sahnelerin görüntülenmesi sırasında izlenen adımlar üç boyutlu görüntüleme konusunda anlatılacaktır.

Bir sahneyi oluşturan üç boyutlu nesnelerin bilgilerini içeren listeler değişik veri yapıları kullanılarak tutulabilirler. Nesneler, yüzeyler, kenarlar ve köşeler ayrı düzeyler olarak düşünülebilir (Şekil 2). Her bir nesneden kendisini oluşturan yüzeylere bağlantılar vardır. Yüzeylerden de kendilerini oluşturan kenar ve köşelere bağlantılar vardır. Kenarlardan da kendilerini oluşturan iki uç köşeye ulaşılabilmektedir. Her bir düzey ile alt düzeyler arasında bağlar olduğu gibi her bir düzeyin kendi içerisinde de bağlar vardır. Örnek olarak bir sahneyi oluşturan nesneler bağlı listelerle birbirine bağlanabilir.

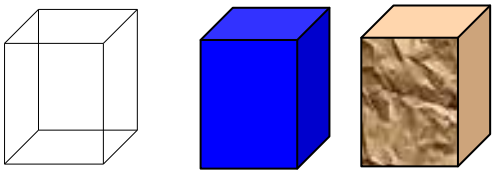


Şekil 2 : Çokgen yüzeyli nesnelerin bilgilerini içeren listelerin düzeyleri ve bağlantıları

Eğrisel yüzeyler de çokgensel yüzey yaklaşımı yapılarak çizdirilebilir. Örnek olarak bir silindir, çokgen ağı (polygon-mesh) şeklinde dilim dilim çizilen çokgenler yardımı ile oluşturulabilir (Şekil 3). Çokgen dilimlerinin sayısı çoğaltıldıkça köşeli parçalardan oluşan silindir gerçekten eğrisel düzgün bir silindir şekline gelecektir (4,6,8,12,36,72,180,360 dilim).

Şekil 3 : Çokgensel yüzey yaklaşımı ile, dilim sayısı değiştirilerek oluşturulmuş silindirler

Katı modelleme yapıldığında nesnelerin özellikleri (yapıldığı madde, yansıtıcılık ve geçirgenlik özellikleri, ...), yüzeylerin özellikleri (renk, doku, resim ..) ve kenarların özellikleri de veri yapılarına uygun şekilde depolanmalıdır. Nesnelerin bu tür özelliklerinin dikkate alınarak görüntülenmeleri katı gösterim (solid representation) olarak adlandırılır (Şekil 4-b). Nesnelerin yüzey yapısının sadece çerçevelerinden oluşan gösterim biçimine telkafes gösterim (wireframe representation) adı verilmektedir (Şekil 4-a). Sadece kenarlar çizildiğinden ve yüzeyleri oluşturan her bir noktanın renginin hesaplanması ve çizilmesi gerekmediğinden genelde önizlemelerde nesnelerin genel görüntüsünü veren telkafes gösterim kullanılır. Kenar listesinin tutulması telkafes görüntülerin doğrudan oluşturulmasını sağlar.



a) telkafes b) katı

Şekil 4 : Telkafes gösterim ve katı gösterim

3. Üç Boyutlu Nesnelerin Ekranda Görüntülenmesi

Üç boyutlu nesnelerin ekranda görüntülenmesi iki boyutlu nesnelerin ekranda görüntülenmesine oranla çok daha zordur. Bunun nedeni, görüntü yüzeylerinin iki boyutlu olmasıdır. Üç boyutlu nesnelerin koordinatlarının iki boyutlu koordinatlara dönüştürülmesi gerekmektedir. Üç boyutlu uzayda bakış noktası ve yönü, arkada kalan nesneler ve nesnelerin arkada kalan yüzeyleri gibi faktörleri de göz önünde bulundurmak gerekmektedir. [4][8] Buna göre üç boyutlu nesneler içeren bir sahnenin görüntülenmesi için izlenen işlem sırasının basitleştirilmiş hali aşağıdaki gibidir (Şekil 5):

1) Nesneler tanımlandıkları yerdeki başlangıç noktasına (orijine) göre ölçeklendirilir (scaling) ve döndürülürler (rotation). Ölçeklendirme ve döndürme değerleri nesnelerin uzaydaki büyüklüğünü ve yönünü belirler. Bu adım sonunda nesneler istenilen boyut ve açıya getirilmiş olurlar.

2) Nesneler yer değiştirme (translation) işlemi ile tanımlandıkları uzaydan gerçek uzaydaki konumlarına taşınırlar.

$$z_{ortalama} = \frac{1}{n}\sum_{i=0}^{n-1} z_i$$

3) Bakış noktasının orijine taşınması ile nesneler, bakış noktasının başlangıç noktası kabul edildiği uzaydaki konumlarına getirilirler. Bu tüm nesneler için bir yer değiştirme işlemidir.

4) Bakış yönünün z eksenine alınması ile tüm nesnelerin uzaydaki bir kameranın veya bir gözün bakış yönüne göre yerleştirilmesi sağlanır. Bu tüm nesneler için bir döndürme işlemidir.

5) Üç boyutlu nesnelerin iki boyutlu görüntü yüzeyinde görüntülenmek üzere izdüşümlerinin alınması ile z koordinatları ortadan kaldırılır. İzdüşüm uzayın boyutunu azaltma işlemidir. Nesnelerin boyutlarının göreceli oranlarını koruyarak değişik yer ve bakış açılarından bakılarak elde edilen görüntülerden yararlanan mimari ve mühendislik uygulamalarında kullanılan paralel izdüşüm yönteminin yanında görüntülerde derinlik hissi uyandıran perspektif izdüşüm gibi daha gerçekçi yöntemler de kullanılabilir.

6) Görünmeyen yüzeylerin ortadan kaldırılması ile görünen yüzeyler belirlenir. Bakış noktasından görünmeyen yüzeylerin ekranda görüntülenmemesi istenmektedir. Nesnelerin arka yüzeyleri ve diğer nesnelerin arkasında kalan nesneler görünmemelidir. Yüzey normali arkaya bakan nesnelerin belirlenmesi ile arka yüzeyler ortadan kaldırılır. Diğer nesnelerin arkasında kalan nesnelerin ortadan kaldırılması için ise değişik yöntemler kullanılabilir. Bunlardan birisi, tüm nesnelerdeki görüntülenecek tüm yüzeylerin bakış noktasına göre uzakta olandan yakına olana doğru sıralanmasıdır. Böylece yakında olan nesneler daha sonra çizileceklerinden ekranda diğerlerinin üzerini kaplarlar. Üçgen veya dörtgen

şeklindeki çokgenleri uzaklıklarına göre sıralamak için kullanılan en basit yöntem köşelerinin z koordinatlarının derinlik ortalamalarını kullanarak yapılır. Fakat bu yöntem karmaşık nesneler ve nesne grupları için gerçek sonuçları vermez. Ortalama z değeri, n köşeli bir çokgende i köşe numarası ve z_i ilgili köşenin z değeri olmak üzere şu şekilde hesaplanır :

7) İki boyutlu görüntü yüzeyine aktarılmış ve görünecek olan nesneler için bir çeşit ölçeklendirme yapılarak gerçek dünyadaki pencerenin ekran penceresine eşlenmesi sağlanır. Bu işlem gerçek koordinatlardan araç koordinatlarına dönüşüm olarak adlandırılır.

8) Son aşama, nesnelerin ekranda görüntülenmesidir. Yüzeylerin çizimi çokgen gibi basit iki boyutlu nesnelerin çizilmesini sağlayan çizim fonksiyonları yardımı ile gerçekleştirilir. Nesnelerin telkafes (wireframe) şeklinde sadece kenarlarının çizilmesi isteniyorsa çizgiler veya içi boş çokgenler, katı (solid) bir nesne şeklinde renkli olarak çizilmesi isteniyorsa içi dolu çokgenler kullanılır. Nesnelerin yüzeylerine renk, desen ve doku verilebilir.
Bilgisayar grafiklerinde gerçekçi görüntüler oluşturmak için yukarıda anlatıldığı gibi nesnelerin arka ve görünmeyen yüzeylerinin elenmesi ve perspektif izdüşüm gibi yöntemler kullanılarak tam olarak temsil edilmesi dışında ışın izleme (ray tracing) ve ışıma (radiosity) yöntemlerinin de kullanılması gerekmektedir. Bir nesne üzerinde görülen renkleri ve aydınlatma etkilerini (ışık yansımaları, saydamlık, gölgelendirme gibi konular da dahil) modellemek karmaşık bir işlemdir; fiziksel ve psikolojik yönleri vardır. Temelde aydınlatma etkileri, nesne yüzeylerinin elektromanyetik enerji etkileşimini dikkate alan modellerle tanımlanır. Işık, insan gözüne ulaştığında, gerçek ortamda görülenleri tanımlayan algılama işlemleri başlar. Fiziksel aydınlatma modellerinde, nesnelerin türlerinin, ışık kaynaklarına ve diğer nesnelere göre nesnelerin yerlerinin, ışık kaynaklarının özelliklerinin bilinmesi gerekir. Nesneler, saydam olmayan maddelerden yapılmış olabilecekleri gibi, çok veya az saydam da olabilirler. Parlak veya tam tersine donuk yüzeyli olabilecekleri gibi, değişik yüzey desenlerine de sahip olabilirler. Nesneler gibi, ışık kaynaklarının da değişik konum, renk ve nitelikleri olabilir. [4]

4. İki Boyutlu Uygulama Geliştirme Arayüzü Kullanılarak Üç Boyutlu Bir Küp Nesnesinin Bilgisayarda Oluşturulup Görüntülenmesi

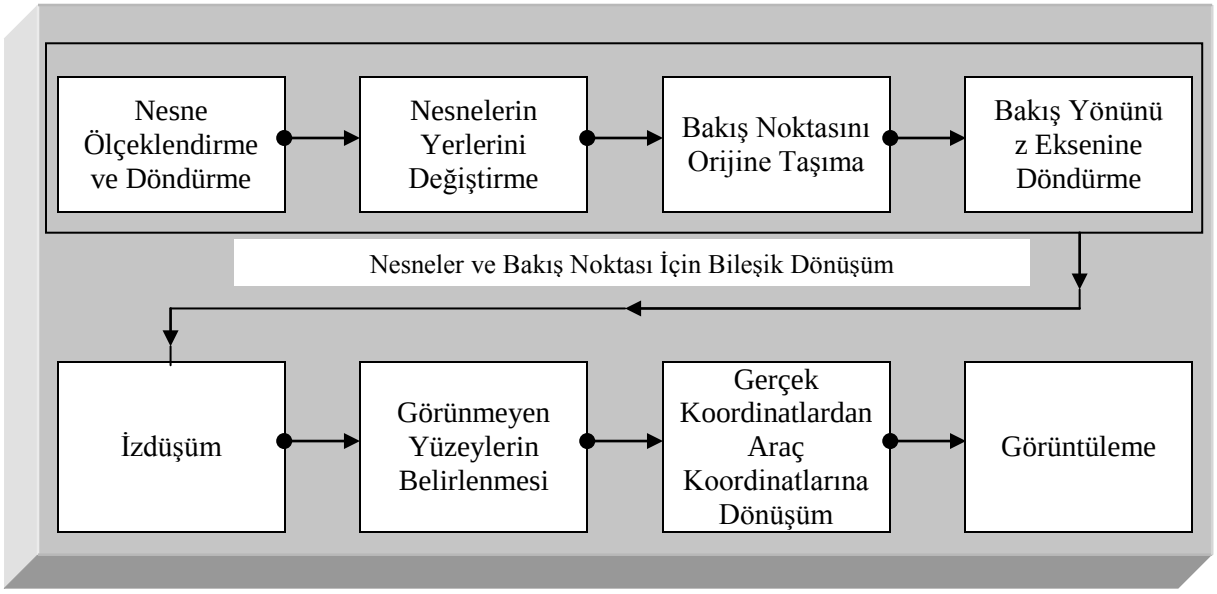
Java 2D kullanarak geliştirdiğim bir applet'e internet üzerinden erişilerek bir küp üzerinde etkileşimli olarak klavye veya düğme takımı yardımı ile dönüşüm işlemleri (taşıma, döndürme, ölçeklendirme) yapılabilmekte ve perspektif, paralel görüntüler oluşturulabilmekte ve küp telkafes (wireframe) veya sabit gölgelendirmeli katı (solid) bir şekle getirilebilmektedir. Hazırladığım programa "bornova.ege.edu.tr/~ugur/kup3d.html" adresinden ulaşılabilmekte ve kullanılabilmektedir.

Yazdığım applet'in ilk sürümünde daha çok kişi tarafından anlaşılmasının kolay olması açısından köşe, kenar ve yüzey listelerini tutmak için daha esnek ve uygun olan bağlı listeler kullanmak yerine dizi kullandım. Bir küp nesnesinin 8 köşesinin x,y,z koordinatları v[8][3] matrisinde tutulmaktadır. İlgili atamaları yapan metot aşağıda verilmiştir :

```
public void set_vertices()
 {
  v[0][0] = 0 ; v[0][1] = 0 ; v[0][2] = 0;
  v[1][0] = 0 ; v[1][1] = 0 ; v[1][2] =  1;
  v[2][0] = 0 ; v[2][1] =  1; v[2][2] = 0;
  v[3][0] = 0 ; v[3][1] =  1; v[3][2] =  1;
  v[4][0] =  1; v[4][1] = 0 ; v[4][2] = 0;
  v[5][0] =  1; v[5][1] = 0 ; v[5][2] =  1;
  v[6][0] =  1; v[6][1] =  1; v[6][2] = 0;
  v[7][0] =  1; v[7][1] =  1; v[7][2] =  1;
 }
```

Kübün 12 kenarının uç köşelerini belirten atamaları yapan metot aşağıda verilmiştir :

```
public void set_edges()
 {
  e[0][0]  = 0; e[0][1]  = 1;
  e[1][0]  = 0; e[1][1]  = 2;
```



Şekil 5 : Üç Boyutlu Nesnelerin Görüntülenmesi Sırasında Gerçekleştirilen İşlemler (View Pipeline)

```
  e[2][0]  = 0; e[2][1]  = 4;
  e[3][0]  = 7; e[3][1]  = 3;
  e[4][0]  = 7; e[4][1]  = 5;
  e[5][0]  = 7; e[5][1]  = 6;
  e[6][0]  = 6; e[6][1]  = 2;
  e[7][0]  = 6; e[7][1]  = 4;
  e[8][0]  = 1; e[8][1]  = 3;
  e[9][0]  = 1; e[9][1]  = 5;
  e[10][0] = 2; e[10][1] = 3;
  e[11][0] = 4; e[11][1] = 5;
 }
```

Kübün 6 yüzeyinin 4'er kenarını belirten atamaları yapan metot aşağıda verilmiştir :

```
public void set_surfaces()
{
  s[0][0] = 0;  s[0][1] = 8; s[0][2] = 10; s[0][3] = 1;
  s[1][0] = 0;  s[1][1] = 2; s[1][2] = 11; s[1][3] = 9;
  s[2][0] = 10; s[2][1] = 3; s[2][2] = 5;  s[2][3] = 6;
  s[3][0] = 5;  s[3][1] = 4; s[3][2] = 11; s[3][3] = 7;
  s[4][0] = 2;  s[4][1] = 1; s[4][2] = 6;  s[4][3] = 7;
  s[5][0] = 8;  s[5][1] = 9; s[5][2] = 4;  s[5][3] = 3;
 }
```

Kübün tüm köşe noktaları için dönüşüm işlemlerini gerçekleştiren metot aşağıda verilmiştir. Küp nesnesinin tüm köşelerine uygulanacak işlemin dönüşüm matrisi x,y,z eksenlerindeki taşıma, ölçeklendirme ve döndürme parametre değerlerini içeren parameter dizisi kullanılarak elde edildikten sonra tüm köşe noktalarının konumlarını gösteren vektörlerle tek tek çarpılır. Köşe noktalarının dönüşümden sonraki konumlarının değerleri (ekran değerleri) bulunmuş olur. Perspektif izdüşümü alınarak görüntülenmesi gerektiğinde ayrıca perspektif izdüşüm dönüşüm işleminden de geçirilmektedir.

```
public void transform_all_points()
 {
   int i, j;
   double t_4[] = new double[4];
   double i_4[] = new double[4];
   double t_16[] = new double[16];
   transform(t_16, parameter);

   for(i=0; i<8; ++i)
   {
    t_4[0] = v[i][0];
    t_4[1] = v[i][1];
    t_4[2] = v[i][2];
    t_4[3] = 1;
    mult(t_4, t_16, i_4);

    if (perspective_flag==false)
    { t_v[i][0] = i_4[0];
      t_v[i][1] = i_4[1]; } else
    { t_v[i][0] = i_4[0]*(zprp-zvp)/(zprp-i_4[2]);
      t_v[i][1] = i_4[1]*(zprp-zvp)/(zprp-i_4[2]);
    }

   }
 }
```

Üç boyutlu kübün ekranda görüntülenmesini gerçekleştiren metot aşağıda verilmiştir. Kübün 6 yüzeyinden her birinin 4 köşesinin koordinatları saat yönünün tersine sırayla bulunarak yüzey ekrana çizilir. İlgili yüzeyin

normali kameranın yönüne göre arka yüzeyse çizdirilmez. Nesne katı olarak görüntülenecekse yüzeyleri Java'nın fillPolygon metodu [3][5] ile içi ilgili renkle dolu bir çokgen olarak çizdirilir. Telkafes olarak görüntülenecekse Java'nın drawPolygon [3][5] metodu ile çizdirilir.

```
public void paint(Graphics g)
  {

   int i,j,k;
   Dimension d;
   d = getSize();
   showStatus("Copyright by Aybars UGUR,
[Ctrl,shift] yön tuslari veya PgUp PgDown ");
   for(i=0;i<6;++i)
   {
    temp_v[0] = e[s[i][0]][0];
    temp_v[1] = e[s[i][0]][1];
    temp_v[2] = e[s[i][1]][0];
    if(temp_v[2]==temp_v[0])
    {
     temp_v[0]=temp_v[1]; temp_v[1]=temp_v[2];
temp_v[2]=e[s[i][1]][1];
    } else
    if(temp_v[2]==temp_v[1])
    {
     temp_v[2]=e[s[i][1]][1];
    } else;
    temp_v[3] = e[s[i][2]][0];
    if(temp_v[3]==temp_v[2])
     temp_v[3] = e[s[i][2]][1];
    for(j=0;j<4;++j)
    {
     xValues[j] = (int)t_v[temp_v[j]][0];
     yValues[j] = d.height-(int)t_v[temp_v[j]][1];
    }
    if(solid_flag)
    {
     g.setColor(colors[i%6]);
     if( ((double) xValues[0]*(yValues[1]-
yValues[2])+
         (double) xValues[1]*(yValues[2]-
yValues[0])+
         (double) xValues[2]*(yValues[0]-
yValues[1])) < 0)
      g.fillPolygon(xValues,yValues,4);
    } else
    {
     g.setColor(Color.black);
     g.drawPolygon(xValues,yValues,4);
    };
   }
}
```

5. Üç Boyutlu Uygulama Geliştirme Arayüzü Kullanılarak Üç Boyutlu Bir Küp Nesnesinin Bilgisayarda Oluşturulup Görüntülenmesi

Java 3D kullanarak geliştirdiğim bir applet'e internet üzerinden erişilerek katı bir küp üzerinde etkileşimli olarak fare yardımı ile dönüşüm işlemleri (taşıma, döndürme, ölçeklendirme) yapılabilmektedir :

```
// Bir dönüşüm grubu oluşturulup özelliklerinin belirlenerek sisteme eklenmesi
    TransformGroup objTrans = new TransformGroup();
    objTrans.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);
    objTrans.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
    objRoot.addChild(objTrans);

// Üç boyutlu bir küp nesnesinin oluşturulması ve Java 3D
görüntü ağacına yerleştirilmesi
    objTrans.addChild(new ColorCube(0.4));
    BoundingSphere bounds =
      new BoundingSphere(new Point3d(0.0,0.0,0.0), 100.0);
// Kullanıcının fare kullanarak nesnenin dönüşünü kontrol
etmesini sağlayan düğümün oluşturulması
    MouseRotate behavior1 = new MouseRotate();
    behavior1.setTransformGroup(objTrans);
    objTrans.addChild(behavior1);
    behavior1.setSchedulingBounds(bounds);
// Kullanıcının fare kullanarak nesneyi z ekseninde taşımasını
sağlayan düğümün oluşturulması
    MouseZoom behavior2 = new MouseZoom();
    behavior2.setTransformGroup(objTrans);
    objTrans.addChild(behavior2);
    behavior2.setSchedulingBounds(bounds);
// Kullanıcının fare kullanarak nesneyi (x,y) eksenlerinde
taşımasını sağlayan düğümün oluşturulması
    MouseTranslate behavior3 = new MouseTranslate();
    behavior3.setTransformGroup(objTrans);
    objTrans.addChild(behavior3);
    behavior3.setSchedulingBounds(bounds);
```

6. SONUÇ

Aynı işlevi yapan applet'in biri iki boyutlu (Java 2D API) diğeri üç boyutlu (Java 3D API) olmak üzere iki farklı uygulama geliştirme arayüzü kullanılarak geliştirilmesi ile karşılaştırılması Tablo 1'de verilmiştir.

Tablo 1 : Geliştirilen applet için Java 2D ve Java 3D API'lerinin karşılaştırılması

Kriter	Java 2D	Java 3D
Süre Analiz,Tasarım,Kodlama	300 dakika	15 dakika
Hatalardan Arındırma Süresi	110 dakika	2 dakika
Kaynak Kod Satır Sayısı	449 satır	98 satır
Programlamadaki Zorluk ve Önşartlar	Güçlü bir matematik ve programlama alt yapısı gerektiriyor	Java 3D API özelliklerini bilmek gerekiyor
Kullanım Kolaylığı	Kullanıcılar yapısına kolay alışıyor	Java 3D standart tuş ve fare özellikleri ile evrensel
Hız	İyi	İyi

Burada üç boyutlu grafik programlamada 2D API ve 3D API kullanımı arasındaki

karşılaştırma yapılmıştır. Analiz, tasarım, kodlama ve hatalardan arındırma süreleri ve üretilen programların kaynak kod satır sayıları ölçülmüştür. Üretilen programlar kullanım kolaylığı ve hız açılarından karşılaştırılmışlardır. Farklı düzeydeki programcıların benzeri bir programı geliştirmek için harcayacağı süre ve kodlayacağı satır sayıları da farklı olacaktır. Fakat, bu tablodan da çıkan üç boyutlu grafik programlamada yüksek düzeyli uygulama geliştirme arayüzlerinin kullanımının yararlı olduğu sonucunu değiştirecek bir durum ortaya çıkmayacaktır. Daha zor bir yazılımın geliştirilmesi durumunda da iki boyutlu ve üç boyutlu uygulama geliştirme arayüzleri kullanımının getireceği çaba, maliyet ve kalite farkının daha da açılacağı kesindir.

Yazılım ölçümü yazılımın kalitesini anlamak ve yazılım geliştiricilerin verimliliğini belirlemek gibi amaçlarla yapılır. Yazılım ölçütleri fiziksel ölçütler gibi iki çeşittir : Doğrudan ölçüler ve dolaylı ölçüler. Boyut uyarlı ölçütler, yazılım mühendisliği çıktısı ve kalitesinin doğrudan ölçülmesinde kullanılır. Fonksiyon uyarlı ölçütler dolaylı ölçütlerdir. İnsan uyarlı ölçütler yazılım geliştiren kişilerle ilgilidir. Yazılım mühendisliği işlemindeki doğrudan ölçüler maliyet ve harcanan çabadır. Ürünün doğrudan ölçüleri arasında üretilen kaynak kodun satır sayısı, işletim hızı, belirli bir süre içinde tespit edilen hata sayısı sayılabilir. Ürünün dolaylı ölçüleri fonksiyonelliği, kaliteyi, karmaşıklığı, etkinliği, güvenilirliği ve diğer özellikleri içerir. Verimlilik ölçütleri yazılım mühendisliği işleminin çıktısına odaklanır, kalite ölçütleri müşteri gereksinimlerinin (yazılımın kullanılabilirliği) uygunluğunun göstergesidir. Teknik ölçütler ise yazılımın mantıksal karmaşıklığı gibi yazılımın karakterine odaklanır.

İki boyutlu uygulama geliştirme arayüzlerinin kullanılması ile üç boyutlu nesneler bilgisayarda temsil edilip görüntülenebilmektedir. Ancak üç boyutlu nesnelerin iki boyutlu ekran üzerinde görüntülenebilmesi için karmaşık bir işlem sırasının izlenmesi gerekmektedir. Bu nedenle, yazılımlarında veya programlarında üç boyutlu bilgisayar grafiklerini kullanmak isteyen profesyonel ve amatör kişi veya şirketler için bu yolu izlemek çok uygun olmamaktadır. Ayrıca birçok programda tek tür gösterim de yeterli olmayacağından birçok temsil modelinin programlanması da gerekmektedir. Bütün bu nedenler işlemleri zorlaştırıp kaliteyi düşürmekte, yazılım maliyetlerini artırmakta ve yazılımların piyasaya sürümünü geciktirmektedir. Bu tür programları yazmak iyi bir matematik ve programlama altyapısı da gerektirmektedir. Üç boyutlu grafik ve animasyonlar içeren yazılımlar geliştirmek için kullanılan diğer bir yol üç boyutlu uygulama geliştirme arayüzleri (3D API) kullanmaktır. Alt düzeyli 3D API'ler hız sağlarken, yüksek düzeyliler yazılım geliştiricilerin sadece yazılımın içeriğine odaklanmalarını sağlar. [1][2][6][7] Nesnelerin temsili ve görüntülenmesi gibi konularda analiz, tasarım ve kodlama yapılmasına gerek kalmaz. Yüksek düzeyli 3D API'lerin önemi giderek artmaktadır ve gelişimlerinin hızlanması gerekmektedir. Yeni geliştirilmekte olan yazılımlarda kullanım oranları artmıştır ve geliştirilecek olan yazılımlarda daha yoğun ve etkin olarak kullanılacaklardır.

KAYNAKLAR

[1] Brown, K., Petersen, D., "Ready-to-Run Java 3D, with plug-and-play code, 1998.
[2] Day, B., "3D Graphics Programming in Java : Part 1, Java 3D, Javaworld", Aralık 1998.
[3] Deitel, H.M., Deitel, P.J., "Java How to Program", Third Edition, Prentice-Hall, 1999.
[4] Hearn, D., Baker, M.P., "Computer Graphics C Version", Prentice-Hall, 1997.
[5] Knudsen, J., "Java 2D Graphics", O'Reilly, 1999.
[6] Sowizral, H.A., Nadeau, D.R., Bailey, B.J, "Introduction to Programming with Java 3D", San Diego Supercomputer Center, University of California at San Diego, June 3rd 1998.
[7] Sun Microsystems, Inc., "Java 3D API Collateral", The Fourth Generation of 3D Graphics APIs Has Arrived – A New Generation of 3D API Emerges, 29 Temmuz 1999.
[8] Watt, A., "3D Computer Graphics", Third Edition, Addison-Wesley, 2000