

50609

T.C.
EGE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
10.2700.0000.011

ÜÇ BOYUTLU
ÇİZİM ve ANİMASYON

YÜKSEK LİSANS TEZİ

Hazırlayan
Aybars UĞUR

Danışman
Doç. Dr. Seyit GÜLER

Bu tez

Prof. Dr. Halil Şengonca
Doç. Dr. Seyit Güler
Doç. Dr. Ahmet Kaşlı
'dan oluşan jüri tarafından oybirliği ile kabul edilmiştir.

Bornova - İZMİR
1996

ÖZET

ÜÇ BOYUTLU ÇİZİM ve ANİMASYON

UĞUR, Aybars

Yüksek Lisans Tezi, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Seyit Güler

Ağustos 1996, 103 sayfa

Bu tezde, kişisel bilgisayarlar üzerinde, iki ve üç boyutlu çizim ve animasyonlar hazırlanmasını sağlayan bir program tasarlanıp geliştirilmiştir.

İki boyutlu animasyonlarda, birbirinden az değişiklikler içeren bir dizi resim arka arkaya görüntülenmektedir. Bitmap formatındaki resimler de, programın basit çizim fonksiyonlarını kullanan kısmı yardımı ile oluşturulabilmekte ya da değiştirilebilmektedir.

Üç boyutlu nesneler, yaygın olarak kullanılan, nesnelerin düzlem, çokgen yüzey bilgilerini içeren sınır temsili yöntemi yardımı ile modellenmektedir. Üç boyutlu katı nesnelerin, iki boyutlu görüntü yüzeylerinde temsili ve animasyonu için de, üç boyutlu dönüşümler, izdüşüm teknikleri (paralel izdüşüm, perspektif izdüşüm), arka ve görünmeyen yüzeylerin elenmesi yöntemleri kullanılmaktadır.

Anahtar Kelimeler : Çizim, animasyon, bitmap, katı nesne, sınır temsili, üç boyutlu dönüşümler, paralel izdüşüm, perspektif izdüşüm, arka yüzeylerin elenmesi, görünmeyen yüzeylerin elenmesi.

ABSTRACT
THREE DIMENSIONAL DRAWING and ANIMATION

UĞUR, Aybars
MSc in Computer Eng.
Supervisor: Assoc. Prof. Dr. Seyit Güler
August 1996, 103 pages

In this thesis, a program for preparing two and three dimensional drawings and animations on personal computers has been designed and developed.

A series of slightly changed bitmap images are displayed in rapid succession for two dimensional animation. Bitmap images could be constructed or modified by the program part using simple drawing functions.

Three dimensional objects are modelled by a common method which is called boundary representation contains a set of plane, polygon surfaces of objects. Three dimensional transformations, projection techniques (parallel projection, perspective projection), back-face removal and hidden-surface removal methods are used for representing and animating three dimensional solid objects on two dimensional viewing surface.

Keywords : Drawing, animation, bitmap, solid object, boundary representation, three dimensional transformations, parallel projection, perspective projection, back-face removal, hidden-surface removal.

TEŞEKKÜR

Tez çalışmasını yöneten ve başta kaynak ve bilgi olmak üzere her konuda destek veren danışmanım Doç. Dr. Seyit Güler'e teşekkür ederim.

Tez çalışması süresince gösterdikleri yakın ilgi ve sağladıkları olanaklar için, Bilgisayar Mühendisliği Bölüm Başkanı Prof.Dr. Şaban Eren'e teşekkür ederim.

Tezin düzeltme aşamasındaki önerileri için Prof. Dr. Halil Şengonca ve Doç. Dr. Ahmet Kaşlı'ya teşekkür ederim.

Başta Bilgisayar Mühendisliği Bölümü Araştırma Görevlileri olmak üzere, bu tezin ortaya çıkmasında, doğrudan veya dolaylı olarak katkıda bulunan, konu ile ilgili kaynakların bulunmasında yardımcı olan, değerli fikir ve önerilerini esirgemeyen herkese teşekkür ederim.

Tez çalışması ile ilgili yazılımın oluşturulması ve tezin yazılması süresince verdikleri destek ve gösterdikleri anlayıştan dolayı, aileme teşekkür ederim.

İÇİNDEKİLER

	<u>Sayfa</u>
ÖZET	iv
ABSTRACT	v
TEŞEKKÜR	vi
ŞEKİLLER DİZİNİ	xi
SİMGELER ve KISALTMALAR DİZİNİ	xiii
 1. GİRİŞ	 1
 2. ANİMASYON	 4
2.1 Animasyon Nedir ?	4
2.2 El Yapımı Animasyon ve Bilgisayar Animasyonu	5
2.2.1 El yapımı animasyon	5
2.2.2 Bilgisayar animasyonu	6
2.3 Animasyonun Kısa Tarihi	8
2.4 Animasyonun Kullanıldığı Alanlar	10
2.4.1 Bilimsel çalışma ve araştırmalar	10
2.4.2 Mühendislik	11
2.4.3 Eğitim ve öğretim	11
2.4.4 Film endüstrisi	11
2.4.5 Reklamcılık	12
2.4.6 Sanat	13
2.4.7 Spor	13
2.4.8 Bilgisayar Oyunları ve Sanal Gerçeklik	13
2.5 Animasyonun ve Bilgisayar Animasyonunun Yararları	14
2.6 Gelecekte Animasyon	16
 3. WINDOWS	 18
3.1 Windows Nedir ?	18
3.2 Pencere Nedir ?	18
3.3 Pencere elemanları	18
3.4 Nesneye Yönelik Programlama (Object Oriented Programming) ..	19

İÇİNDEKİLER (devam)

Sayfa

4. BİLGİSAYAR GRAFİKLERİ	20
4.1 İki Boyutlu Dönüşüm İşlemleri	20
4.1.1 Yer değiştirme	21
4.1.2 Ölçeklendirme	23
4.1.3 Döndürme	25
4.2 Gerçek Koordinatlardan Araç Koordinatlarına Dönüşüm	26
4.3 Üç Boyutlu Bilgisayar Grafikleri	27
4.3.1 Üç boyutlu koordinat sistemi	28
4.3.2 Üç boyutlu görüntüleme teknikleri	29
4.4 Üç Boyutlu Dönüşüm İşlemleri	33
4.4.1 Yer değiştirme	34
4.4.2 Ölçeklendirme	34
4.4.3 Döndürme	36
4.5 Katı Modelleme	38
4.6 Üç Boyutlu Nesnelerin Sınır Temsili	38
4.7 Bilgisayar Grafiklerinde Veri Yapıları	40
4.8 Vektör İşlemlerine Giriş	41
5. BORLAND C++ ile WINDOWS PROGRAMLAMA	43
5.1 Windows Programlama Temelleri	43
5.2 Çoklu-Ortam (Multimedia).....	44
5.3 API (Uygulama Programlama Arayüzü)	44
5.4 OWL (Nesne Pencereleeri Kitaplığı)	45
5.5 IDE (Bütünleşik Geliştirme Ortamı)	45
5.6 GDI (Grafik Araç Arayüzü)	46
5.6.1 DC (Araç Durumu)	46
5.6.2 GDI koordinat sistemi	47
5.6.3 GDI fonksiyonları ile çizim	47
5.6.4 Bitmap	50

İÇİNDEKİLER (devam)

Sayfa

6. ANİMASYON TEKNİKLERİ	52
6.1 Sprite (karakter) Animasyonu	52
6.2 Sil ve Yeniden Çizdir Tekniği	52
6.3 Bitmap'i Ekran Dışında Hazırlama Tekniği	53
7. ÇİZİM ve ANİMASYON PROGRAMI	54
7.1 Menü Seçenekleri	55
7.2 File (Dosya) Seçeneği	55
7.2.1 BMP (bitmap) alt seçeneği	55
7.3 ClipBoard (Pano) Seçeneği	60
7.3.1 Copy (Kopyala) alt seçeneği	60
7.3.2 Paste (Yapıştır) alt seçeneği	61
7.3.3 Clip (Kesme) alt seçeneği	61
7.3.4 Selectall (Tümünü Seç) alt seçeneği	61
7.3.5 Cls (Ekranı Temizle) alt seçeneği	62
7.4 Draw (Çizim) Seçeneği	62
7.4.1 Line (Çizgi) alt seçeneği	64
7.4.2 Rectangle (Dikdörtgen) alt seçeneği	66
7.4.3 Ellipse (Elips) alt seçeneği	66
7.4.4 Point (Nokta) alt seçeneği	66
7.4.5 FloodFill (İçini Doldur) alt seçeneği	67
7.5 Attributes (Nitelikler) Seçeneği	67
7.6 Animate3D (3D Animasyon) Seçeneği	68
7.6.1 Load (Yükle) alt seçeneği	68
7.6.2 Draw3D (Çizdir) alt seçeneği	75
7.6.3 DelShapes (Şekilleri Sil) alt seçeneği	75
7.6.4 Wireframe (Telkafes) alt seçeneği	76
7.6.5 Parallel (Paralel) alt seçeneği	76
7.7 Help (Yardım) Seçeneği	76

İÇİNDEKİLER (devam)

Sayfa

8. PROGRAM YARDIMI ile HAZIRLANAN ÖRNEK	
ANİMASYONLAR	77
8.1 Windows Ortamından Alınan Resimlerin Arka Arkaya Gösterilmesi	77
8.2 Animasyon Formatlarından Parçalanmış Resimlerin Arka Arkaya Gösterilmesi	80
8.3 Baştan Oluşturulan Resimlerin Arka Arkaya Gösterilmesi	82
8.4 Üç Boyutlu Nesnelerin Çizim ve Animasyonu	84
9. KULLANILAN VERİ YAPILARI ve GÖRÜNTÜLEME	93
10. SONUÇ ve ÖNERİLER	98
YARARLANILAN KAYNAKLAR	101
ÖZGEÇMİŞ	103

ŞEKİLLER DİZİNİ

<u>Şekil</u>		<u>Sayfa</u>
4.1	Yer değiştirme	22
4.2	Ölçeklendirme	23
4.3	Döndürme	26
4.4	Paralel İzdüşüm	30
4.5	Perspektif İzdüşüm	31
4.6	Arka Yüzeylerin Elenmesi	32
4.7	Kenarları 1 birim olan küp nesnesi	39
4.8	Kenarları 1 birim olan küp nesnesinin sınır temsili	39
7.1	Animation Platform uygulama programının penceresi	54
7.2	File (Dosya) seçeneğinin alt seçenekleri	55
7.3	File (Dosya) - BMP alt seçeneğinin alt seçenekleri	56
7.4	File - BMP - Open alt seçeneğinin çalıştırılması sonucu ekrana gelen diyalog kutusu	56
7.5	File - BMP - Save alt seçeneğinin çalıştırılması sonucu ekrana gelen diyalog kutusu	57
7.6	File - BMP - Animate alt seçeneğinin çalıştırılması sonucu ekrana gelen diyalog kutusu	59
7.7	BMP animasyonu için kullanılan bir dosya örneği	60
7.8	Clipboard (Pano) seçeneğinin alt seçenekleri	60
7.9	Draw (Çizim) seçeneğinin alt seçenekleri	62
7.10	Draw seçeneğinin alt seçenekleri yardımı ile oluşturulan iki boyutlu temel çizimler	63
7.11	Attributes (Nitelikler) seçeneğinin alt seçenekleri	67
7.12	Color Diyalog Kutusu	68
7.13	Animate3D seçeneğinin alt seçenekleri	68
7.14	Animate3D - Load alt seçeneğinin alt seçenekleri	69
7.15	About Diyalog Kutusu	76

ŞEKİLLER DİZİNİ (devam)

<u>Şekil</u>	<u>Sayfa</u>
8.1 İki boyutlu resimlerin arka arkaya gösterilmesini sağlayan bir animasyon dosyası	78
8.2 İki boyutlu resimlerin arka arkaya gösterilmesi ile elde edilen bir animasyon örneğinin resim kareleri	79
8.3 İki boyutlu resimlerin arka arkaya gösterilmesi ile elde edilen bir animasyon örneğinin resim kareleri	80
8.4 İki boyutlu resimlerin arka arkaya gösterilmesini sağlayan bir animasyon dosyası	81
8.5 İki boyutlu resimlerin arka arkaya gösterilmesi ile elde edilen bir animasyon örneğinin resim kareleri	81
8.6 Bir insanın yürümesinin aşamalarını gösteren animasyonun resimleri	82
8.7 Bir insanın yürümesinin aşamalarını gösteren animasyonun resim parçaları	83
8.8 Bir insanın yürümesinin aşamalarını gösteren animasyon dosyası	84
8.9 İşlem merkezi ortasında olan üç boyutlu küp nesnesinin, veri yapısına yüklenmesini sağlayan diyalog kutusu	85
8.10 İşlem merkezi ortasında olan üç boyutlu küp nesnesinin tanımlandığı "cubecntr.shp" dosyası	86
8.11 Üç boyutlu nesnelerin ekrana çizilmesi	87
8.12 Üç boyutlu nesnelere ilişkin animasyon (senaryo) dosyasını yüklemek için kullanılan diyalog kutusu	88
8.13 Üç boyutlu birkaç nesnenin çizilmesi ile oluşan görüntü	92
9.1 Çokgen düzlemsel yüzeyler şeklinde tanımlanan katı nesneler	93
9.2 Çokgen düzlemsel yüzeyler şeklinde tanımlanan katı nesneler (ayrıntılı)	95
9.3 Üç boyutlu nesnelerin görüntülenmesi sırasında gerçekleştirilen işlemler	96

SİMGELER ve KISALTMALAR DİZİNİ

Simgeler

Acıklama

x_F	Sabit noktanın x koordinatı.
y_F	Sabit noktanın y koordinatı.
S_x	x eksenindeki ölçeklendirme faktörü.
S_y	y eksenindeki ölçeklendirme faktörü.
S_z	z eksenindeki ölçeklendirme faktörü.
T_x	x eksenindeki yer değiştirme.
T_y	y eksenindeki yer değiştirme.
T_z	z eksenindeki yer değiştirme.
ϕ (fi)	Yatay eksen ile yapılan açı.
θ (teta)	Döndürme açısı.

Kisaltmalar

Acıklama

2D (Two Dimensional)	İki Boyutlu.
3D (Three Dimensional)	Üç Boyutlu.
API (Application Programming Interface)	Uygulama Programlama Arayüzü.
BMP (Bitmap)	Dikdörtgen grafik nesnesinin veri dizisi.
CAD (Computer Aided Design)	Bilgisayar Destekli Tasarım.
DDB (Device Dependent Bitmap)	Araç Bağımlı Bitmap.
DIB (Device Independent Bitmap)	Araç Bağımsız Bitmap.
GDI (Graphics Device Interface)	Grafik Araç Arayüzü.
IDE (Integrated Development Env.)	Bütünleşik Geliştirme Ortamı.
OLE (Object Linking Embedding)	Nesne Bağlama ve Katma.
OWL (Object Windows Library)	Nesne Pencereleeri Kitaplığı.

1. GİRİŞ

Bilgisayar yazılım ve donanımlarında son yıllarda görülen gelişmeler, bilgisayar grafikleri ve animasyon işlemlerindeki nitelik ve hızı oldukça arttırmıştır. Bu şekilde, günlük yaşamdan değişik bilim dallarına kadar her alanda etkisini göstermeye başlayan bilgisayar grafikleri ve animasyona duyulan ilgi ve gereksinim giderek artmakta, bu konular, standartların oluşması ve yeni kavramların eklenmesi ile her geçen gün zenginleşmektedir.

Bu projenin uygulama kısmının, yaygın olarak kullanılan ve nitelik ve hızlarında son bir yılda meydana gelen iyileşme nedeni ile IBM uyumlu bilgisayar donanımı üzerinde, yine yaygın olarak kullanılan ve gelişmesini sürdüren Windows programlama ile yapılmasına karar verilmiştir. C programlama dilinin de üstünlüklerinden yararlanmak amacı ile, Windows 3.1 üzerinde çalışan Borland C++ 4.0 sürümü kullanılmıştır. Grafik ile ilgili programlarda, kullanıcı arayüzünün niteliği oldukça önemlidir. Bu nedenle, bu tür programların grafik tabanlı işletim sistemleri üzerinde ya da grafik tabanlı uygulamaları destekleyen kütüphane ve fonksiyonları içeren arayüz veya ortamlarda yazılması yararlıdır.

Kullanılan Donanım	
İşlemci ve Hız	486 DX 2 80 Mhz
Bellek	8 Mbyte RAM
Sabit Disk	420 Mbyte
Ekran	SVGA Renkli

Kullanılan Yazılım	
İşletim Sistemi	DOS 6.0
Arabirim	Windows 3.1
Programlama Dili	BC ++ 4.0

Bu tez projesi süresinde animasyon konusu, interneti de içine alan çeşitli kaynaklardan araştırılmış, geleneksel animasyon ve bilgisayarlı animasyon arasındaki fark ve benzerlikler, bilgisayarlı animasyonun günümüzde geldiği nokta ve yararları incelenmiştir. Ayrıca, yukarıda belirtilen donanım ve yazılım üzerinde, iki ve üç boyutlu basit çizim ve animasyonlar yapılmasını sağlayan bir arayüz geliştirilmiş ve grafik ve Windows programlama konularında deneyim kazanılmıştır.

Oluşturulan arayüzün içerdiği nokta, çizgi, dikdörtgen, elips gibi basit şekillerin çizilmesini sağlayan çizim fonksiyonları ile, iki boyutlu resimler oluşturulabilmektedir. Bu

şekiller, içi boş ya da dolu ve renkli olarak çizilebildiği gibi, içleri sonradan da boyanabilmektedir. Oluşturulan bu resimler, ikincil bellek ortamlarına, bitmap resim formatı şeklinde kaydedilebilmektedir. Hazır olarak alınan veya program yardımı ile hazırlanan resimler ekrana getirilerek, üzerlerinde değişiklikler yapılabilmektedir. Pano aracılığı ile, diğer uygulamalar ile resim alışverişi de olasıdır. İkincil bellek ortamlarında bulunan bitmap resim formatındaki dosyaların arka arkaya gösterilmesi ile, iki boyutlu animasyonlar oluşturulabilmektedir.

Üç boyutlu nesneler, çokgen yüzeyler kümesi olarak bilgisayarda sunulmuştur. Sınır Temsili (Boundary Representation veya B-Representation) adı verilen bu yöntem yardımı ile dosyadan, köşe, kenar ve çokgen yüzey bilgileri okunan üç boyutlu nesneler, ekranda gösterilebilmektedir. Katı nesnelerin arka yüzeyleri veya başka nesnelerin arkasında olan katı nesnelerin yüzeylerinin ekranda görüntülenmemesi gerekmektedir. Bunlar için, Arka Yüzeyleri Eleme (Back-Face Removal) Yöntemi ve Görünmeyen Yüzeyleri Eleme (Hidden-Surface Removal) yöntemlerinden Derinlik Sıralaması Yöntemi (Depth-Sorting Method) kullanılmıştır. Üç boyutlu bir nesnenin, iki boyutlu görüntü yüzeyinde sunulması için, Paralel İzdüşüm (Parallel Projection) ve Perspektif İzdüşüm (Perspective Projection) yöntemlerinin her ikisinin de kullanılabilmesi sağlanmıştır. Ayrıca, bakış noktasının yeri ve yönü de değiştirilebilmektedir. Üç boyutlu nesnelerin yerlerinin ve durumlarının belirlenmesinde ve animasyonlarında, geometrik dönüşüm işlemlerinden (yer değiştirme, ölçeklendirme, döndürme) yararlanılmıştır. Nesneler, yüzey renkleri ile gerçekçi bir şekilde çizilebildikleri gibi, çerçevesini oluşturan çizgiler şeklinde de (wireframe) çizilerek, hızlı bir şekilde görüntülenebilmektedir. Çizgiler şeklinde gösterim, nesnenin yüzey bilgilerini içermediği için, tam bir tanım olmasa da, animasyonların önizlemesinde kullanılabilir.

Nesne kelimesi, tez metni içerisinde iki farklı anlamda kullanılmıştır: Nesneye yönelik programlamada nesne, veri yapıları ve onlar üzerinde işlemler yapan fonksiyonlardan oluşan bir soyut veri tipidir. Bilgisayar Grafiklerinde ise, iki boyutlu veya üç boyutlu resim parçaları, nesne olarak adlandırılır.

Bölüm 2'de, animasyon kavramı, animasyonun tarihçesi ve gelişimi, kullanıldığı alanlar ve önemi üzerinde durulmuştur. Bölüm 3'te, arayüzün yapıldığı Windows ortamı kısaca tanıtılmıştır. Bölüm 4'te, bilgisayar grafikleri konusu ele alınmış, iki ve üç boyutlu

koordinat sistemleri incelenmiş, üç boyutlu nesnelerin bilgisayarda sunulması ve ekranda görüntülenmesi yöntemleri anlatılmıştır.

Bölüm 5'te, Windows programlama ile ilgili genel kavramlar ve grafik programlama üzerinde durulmuştur. Bölüm 6'da animasyon teknikleri anlatılmıştır. Bölüm 7'de, Borland C++ üzerinde geliştirmiş olduğum arayüz tanıtılmıştır. Bölüm 8'de, bu arayüz üzerinde çizim ve animasyonların nasıl oluşturulacağı anlatılmış ve hazırlanan örneklerle yer verilmiştir. Bölüm 9'da ise, üç boyutlu katı nesnelerin bilgisayardaki sunumunda kullanılan veri yapıları ve bu nesnelerin bilgisayar ekranında görüntülenmesi için, sırası ile yapılması gereken işlemler üzerinde durulmuştur.



2. ANİMASYON

2.1 Animasyon Nedir ?

Animasyon, film resimlerini ve bilgisayarda oluşturulmuş görüntüleri yapay olarak canlandırma, yani hareket ediyor gibi görünen resimler oluşturma işlemidir. Oluşan bu hareketli görüntülere de animasyon denilmektedir.

Bir parça film şeridine bakıldığında, durağan resimlerden oluşan bir ortam görülür. Bilimsel bir bakış açısı ile animasyon, gözün çalışma düzeneği temeline dayanır. Gözün ağ tabakasının koruyucu özelliği (ağtabakanın aydınlanması kesildiğinde ışık duyumunun hemen bitmemesi), belirli bir orandan (saniyede 18-24) daha hızlı olan hareketlerde devreye girerek hareketin bulanık görülmesine yol açar. Retinaya (gözün ağ tabakası) gelip kaydedilen bir tek resim, beyinde daha uzun süre tutulur. Belirli en az süre (50 milisaniye = 1 saniye / 20 resim) içinde ikinci bir resmin görünmesi ile beyin, tutmaya devam ettiği son resimle yenisini birleştirebilir. Bu şekilde, ardarda gelen resimlerin gösterilmesi ile, tüm resimler birbirleri ile kaynaşır. Ardarda gelen resimler arasındaki küçük değişimler sonucunda, sürekli bir hareket izlenimi oluşur. Standart 35 mm'lik filmlerde, saniyede 24 resim gösterilir. Bu nedenle, görüntüde, farkedilir bir titreme olmaz (Televizyon saniyede 30 resim göstermektedir. Saniyede 24 resim formatında yapılan yayınlar ile arasındaki uyumsuzluk, her 4'üncü resim iki kere gösterilerek yapılan çevrim sonucu giderilir) (Fox and Waite, 1984).

Animasyonda hareket eden nesnelerin hızları, tüm hareket için yapılan çizimlerin sayısına ve ardarda gelen resimlerde nesnenin koordinatları arasındaki uzaklığa bağlıdır. Örnek olarak, zıplayan bir top bir resimde en altta, sonrakinde en üstte ise hızlı bir şekilde zıpladığı anlaşılır (Fox and Waite, 1984).

İki saatlik bir film elde edebilmek için $172\ 800$ ($24 \text{ resim} * 2 * 60 * 60$) resim karesi elde etmek gerekecektir. Bilgisayarlar kullanılmadan önce, animasyona dayalı filmler için her karenin çizilmesi, boyanması ve fotoğraflanması insanlar tarafından el ile yapılıyordu. Buradan da anlaşılacağı gibi, bilgisayarlar animasyon konusunda da bir çığır açmıştır. (Fox and Waite, 1984)

2.2 El Yapımı Animasyon ve Bilgisayar Animasyonu

2.2.1 El yapımı animasyon

Geleneksel olarak, bilgisayar kullanmadan üretilen el yapımı animasyon, temel olarak birkaç adımda elde edilir (Fox and Waite, 1984) :

Adım 1 : Sahne ve olayların ayrıntılı anlatımının yer aldığı senaryonun yazımı.

Adım 2 : Filmin önemli sahnelerinden oluşan konu tahtasının (story board) hazırlanması. Konu tahtası, birkaç resim yada yüzlerce resim içerebilir.

Adım 3 : Filmdeki her harekete ilişkin anahtar resimlerin çizilmesi. Anahtar resimler, ilgili hareketin başlangıç ve bitişini (uç noktalar) gösteren resimlerdir. Örnek olarak, konu tahtasında ata binme gösterilecekse, çizgi film karakterinin ata binmesi sırasında gerçekleştirdiği her işlem için bir anahtar resim çizilebilir. Atın yanında durduğu resim ve kolunu atın üzerine attıktan sonrayı gösteren resim gibi. Bu resimler, filmdeki önemi nedeni ile baş animatörler tarafından seçilir (Glassner, 1984).

Adım 4 : Anahtar resimlerden, ara resimlerin elde edilmesi. Ara resimler, resim kopyalayıp çizmeye dayandığı için, sanatsal yetenekten çok çizim yeteneği olan kişiler tarafından oluşturulur. Ayrıca ara resimler, sıradüzensel olarak, paralel bir şekilde birden çok kişi tarafından aynı anda çalışılarak da tamamlanabilir. Adımlar içerisinde en fazla zaman alan kısım budur. Çizgi film karakterinin, kolunu atın üzerine atmasını gösteren ardışık iki anahtar resmin arasındaki resimler bu aşamada elde edilir. Bu işlemin süresini kapsayacak ve düzgün geçişi sağlayacak sayıda ara resim üretmek gerekir.

Adım 5 : Resimlerin filme alınması. Tamamlanan resimler, özel olarak hazırlanmış bir düzleme konularak tek tek filme alınır. Kamera yeri ve aydınlatma özellikleri değiştirme gibi birtakım yöntemlerle bu aşamada da çeşitli etkiler elde edilebilir.

Ayrıca, 4. adımda oluşturulan resimler taslak çizgiler şeklinde hazırlanıp, boyamadan önce filme alınıp, hareketler nitelik sınavasından geçirilebilir. Hatalar giderilinceye kadar, çizgi (kalem) sınavası yinelenir.

Not : Ses, anahtar resimlerin oluşturulmasında belirleyici olacağı için bu adımdan önce kaydedilir.

Her resim, saydamlar üzerine çizilmiş olan parçaların birleştirilmesi ile oluşturulursa, çizilecek toplam resim sayısı azaltılır. Örnek olarak, çizgi film karakterinin kol resmini içeren saydam, tüm yürüme animasyonları sırasında, değişik zeminlerde, konumu ya da açısı değiştirilerek yeniden kullanılabilir.

2.2.2 Bilgisayar animasyonu

Bilgisayar yardımı ile yapılan animasyona, bilgisayar animasyonu (bilgisayarlı animasyon) denilir. Bilgisayar kullanılarak yapılan animasyonlarda temel olan, modellenmiş cisimlere hareket dizileri uygulayarak gerçeğe yakın görüntüler elde etmektir. Karmaşık matematik formülleri kullanılarak bilgisayarda elde edilen gölgelendirilmiş, ışıklandırılmış üç boyutlu resimler, bilgisayar yardımı ile canlandırılır. Aynı zamanda, verilen iki anahtar resim arasındaki resimlerin oluşturulma işlemi de bilgisayar tarafından yapılarak büyük kazanç sağlanır.

Bilgisayar animasyonu kullanılan teknolojinin niteliğine göre iki kısımda incelenebilir:

- *Yüksek teknoloji ürünü bilgisayarlar ile üretilen animasyon* : Kullanılan donanım ve yazılım, animasyon için özel olarak hazırlanmıştır, oldukça pahalı ve niteliklidir. Özellikle film yapımında yararlanılmaktadır. Silicon Graphics'in iş istasyonları bu tür donanıma örnek olarak verilebilir. Perspektif, görünen yüzeyler, boyama, tonlama ve aydınlatma hesaplarında oldukça hızlı sonuçlar verirler.

- *Düşük fiyatlı bilgisayarlar ile üretilen animasyon* : Evlerde kullanılan genel amaçlı kişisel bilgisayarlarda bile üretilebilen animasyondur. İki boyutlu animasyonlar ya da karmaşık olmayan üç boyutlu animasyonlar için, kişisel bilgisayarlar yeterlidir.

Yeni teknolojilerin üretilmesi ve yeni yöntemlerin bulunması ile birlikte, animasyon giderek yaygınlaşmakta ve önem kazanmaktadır. Örnek olarak, 10 yıl önce, çözünürlüğü 1024*1024 nokta olan ve 16.7 milyon (2^{24}) rengi destekleyen bir bilgisayar animasyon sistemi 160 000 \$ iken, bugün evlerde kullanılan ve maliyeti birkaç bin doları geçmeyen bilgisayarlar, renk sayısı ve çözünürlük bakımından aynı özellikleri içermektedir (Fox and Waite, 1984).

Bilgisayar, tüm animasyonun üretiminde kullanılabildiği gibi, el yapımı animasyonlar için yardımcı bir araç olarak da kullanılmıştır. Bilgisayara alınan el yapımı çizimler kolaylıkla düzeltilebilir, renklendirilebilir ve üzerlerinde değişiklikler yapılabilir. Ara resimler, anahtar resimler ve gerekli bilgiler bilgisayara yüklenerek, bilgisayar tarafından otomatik olarak elde edilebilir (Üç boyutlu animasyonlarda ara resimlerin elde edilmesi, iki boyutlulara göre daha kolaydır. Çünkü, iki boyutlu olarak verilen üç boyutlu iki resim arasındaki geçişlerde üçüncü boyutlara ilişkin bilgiler tanımsız olduğu için bilgisayarın hesaplamaları yanlış olabilir). El yapımı animasyonlarda karşılaşılan, üst üste konulan saydam parçalarının yol açtığı renk değişimi problemi de bilgisayar yardımı ile otomatik olarak çözülür. Çünkü, bilgisayarda, en üste çizilen resmin rengi, daha önceden çizilmiş resimlerin renklerinden etkilenmez.

Bilgisayarda, üç boyutlu gerçekçi görüntüler elde edebilmek için, görünmeyen yüzeyler çıkarılarak perspektif izdüşümleri alınan nesnelerin ışın izleme gibi yöntemler ile gölgelendirilmesi, yani yüzeylerine bakınca görünecek ışık miktarının hesaplanması gerekmektedir. Herbir adımda, pekçok nokta için toplama, çıkartma, çarpma ve bölme gibi çok sayıda matematiksel işlemler yapılır. Bu yüzden, yüksek hızlı işlemciler, hesaplamaların daha hızlı yapılmasını sağlayacaktır. Bilgisayar grafikleri ve animasyonlar, bilgisayarda çok yer tutacağından, ana bellek ve ikincil bellek miktarının çok olması da önemlidir. Kişisel bilgisayarlarda, Windows'ta ve Windows Programlama için kullanılan ortamlarda rahat çalışabilmek, en azından 486 veya Pentium işlemci ve 8 Mbyte RAM gerektirir. Yüksek çözünürlükte aynı anda 2^{24} (yaklaşık 16.7 milyon) renk gösterebilen 24 bitlik görüntü kartlarının kullanılması ile de gerçek renge çok yakın tonlar elde edilecektir.

2.3 Animasyonun Kısa Tarihi

Animasyon tekniklerinin kullanıldığı makinalar, 175 yıldan fazla bir süredir vardır. Thaumatrope, 1820'lerin ortalarında İngiliz doktor John Paris tarafından keşfedilmiştir. Bu, iki yüzünde farklı resimler olan bir diskin, iki yanından bağlı ipler yardımı ile hızlı bir şekilde çevrilmesi sonucu, iki resmi aynı anda görme temeline dayanır. Örnek olarak, bir madalyonun, bir yüzüne kafes, bir yüzüne papağan resmi çizilip hızlıca çevrilirse papağan, kafes içerisindeymiş gibi görülür (Fox and Waite, 1984).

Gözün ağ tabakasının görüntüyü saklama olgusu çok uzun zamandır bilinmektedir. Belçikalı fizikçi Plateau, buna dayanarak 1832'de Fenakistiskop adlı aygıtı yapmıştır. Fenakistiskop, bir kulba takılı daire şeklinde çentikli bir diskin döndürülmesi temeline dayanır. Diskin üzerinde, herbiri animasyonun bir karesine denk gelen bir dizi çizilmiş resim vardır. Bir aynanın önüne konulan dairenin döndürülmesi sırasında, çentikleri arasından bakılması ile animasyon görüntülenir. İngiliz William G. Horner'ın bundan az bir değişikliklerle 1834'te gerçekleştirdiği Zootrop (Wheel of Life) dünya çapında ün kazanmıştır. Zootrop, içine resimlerin çizildiği dönen bir silindirdir. Bu silindir döndürüldüğünde, resimler dairesel yüzeyindeki yarıklardan bakılarak görülür. Her iki aygıt da yansıtma konusunda yetersizdir, ekrana yansıtılan resimler bir an için ışık alabilmektedir ve ancak çok az miktarda desen ile çalışmaya uygundur (Fox and Waite, 1984).

Sinema kameralarının keşfinden çok önce Eadweard Muybridge, koşan bir atı görüntülemek için koşuyoluna bir dizi (12 tane) hareketsiz fotoğraf makinası (fotoğraf 1816'da icat edilmiştir) yerleştirmişti. At koşarken, her bir makineye bağlanmış ipi koparıyor ve böylece deklanşörü çalıştırıyordu. İlk denemelerini 1872'de yapmış olmakla birlikte, bir atın yürüyüşlerinin her bir evresini daha sonra rötuş yapmadan fotoğraf halinde kaydetmeyi 1878'de başardı. Muybridge daha sonra, resimlerin olduğu dış kenarı dönebilen dairesel camlar kullanarak Zoopraxiscope'u (Zoopraxiscope) geliştirmiştir. Aydınlatılan disk döndürüldüğünde, bir turu yarım saniye kadar süren tekrarlı hareket dizisi elde edilmektedir (Fox and Waite, 1984).

Praksinoskop, Zootrop'un kertiklerinin camlar ile değiştirilmiş halidir. Mucit Emile Reynaud bu aracın, resimleri ekrana yansıtan bir çeşidini bulmuştur. Film karelerinin, yarı

saydam uzun kağıt şeritlere çizilmesinden faydalanarak, dünyanın ilk sinema tiyatrosunu 1892'de Paris'te açmıştır (Fox and Waite, 1984).

1880'lerde selüloitin ortaya çıkışı, film dünyasındaki gelişmeleri hızlandırmıştır. Sinemanın bulunuşu, 1895'te hareketin çözümlenmesi ve filmlerin gösterimi gibi sinemanın iki işlevini yerine getiren bir aygıt yapan Lumiere kardeşlere mal edilir. O güne kadar birçok araştırmacı, bu iki işlevi gerçekleştirebilmek için çalışmışlardır.

Animasyon üretiminin eskiden beri kullanılan diğer bir popüler yolu da Kineograph'tır. Özel kartlara çizilen resimlerin üstüste konularak bağlanması, sonra baş parmak ile hızlıca açılması esasına dayanır. Patenti 1868 yılında alınmış olmasına rağmen, çok daha eskiden beri bilinen bir yöntemdir (Fox and Waite, 1984).

İlk çizgi filmler 1906'da Amerikalı James Stuart Blackton ve 1908'de Fransız Emile Cohl tarafından yapılmıştır. Emile Cohl, beyaz kağıt üzerindeki karakalem çizimlerini fotoğraflamıştır. Negatiflerini kullanarak siyah zeminde beyaz şekiller göstermiştir. Birkaç yıl içinde, animasyon tekniklerinin gelişmesi ile yeni çizgi filmler yapılmış ve çizgi film karakterleri üretilmiştir.

1920'lerin başında zemin çizimi, animasyon hareketlerinden ayrı bir iş haline gelmiştir. Ayrı uzmanlar tarafından tamamlanan zeminler üzerine, animasyon ile uğraşan kişilerin yaptıkları resimler konulmuştur. Sonra da, resimlerin ana hatlarının çizimi ile renklendirme işlemleri birbirinden ayrılmıştır. 1960'lı yıllara kadar pekçok unutulmaz çizgi film hazırlanmış ve çizgi film televizyonun vazgeçilmez bir parçası olmuştur. 1960'lı yıllardan itibaren ilk bilgisayarlı animasyonların üretimine başlanmıştır.

Star Wars filminde pilot brifingindeki "Death Star" simülasyonu için bilgisayar animasyonu kullanılmıştır. Walt Disney tarafından üretilen TRON adlı filmde, bilgisayar animasyonundan geniş ölçüde yararlanılmıştır (1982). On beş dakika süren bu animasyonlar için tasarlanan araçlar ve gölgelendirmeler oldukça başarılıdır. "Star Trek II - The Wrath of Khan" filminde bilgisayar animasyonu kullanılan 1 dakikalık bölüm, Lucasfilm Ltd.'nin grafik ustaları tarafından 5 aylık bir sürede üretilmiştir. Gezegen yüzeyi, yerçekilleri ve etkiler gerçekçi bir şekilde tasarlanmıştır (Fox and Waite, 1984).

2.4 Animasyonun Kullanıldığı Alanlar

Animasyon tekniklerinin kullanıldığı alanlar, her geçen gün daha da artmaktadır. Günümüzde, animasyonun en çok kullanıldığı alanlar şunlardır (Fox and Waite, 1984) :

Bilimsel çalışma ve araştırmalar
Mühendislik
Eğitim ve öğretim
Film endüstrisi
Reklam
Sanat
Spor
Bilgisayar oyunları ve Sanal Gerçeklik

2.4.1 Bilimsel çalışma ve araştırmalar

Animasyon, başta uzay ve tıp ile ilgili olmak üzere pekçok bilim dalında, çalışma ve araştırmalarda kullanılmaktadır.

Uzay çalışmaları: Uzaya ilişkin fizik yasaları ve hareketler bilgisayara yüklenir. Sonra, uzay araçlarının yörüngeleri bulunur, uzay araçlarının içindeymiş veya dışından bakıyormuş gibi görüntüler elde edilir. Kalkış, iniş, çarpışma gibi olaylar ile ilgili animasyonlar oluşturularak uzay araçlarını tehlikeye atmadan, yapılan hesaplamaların doğruluğu sınanır. Gök cisimlerinin hareketlerinin animasyonları da yapılan araştırmaların görselleştirilmesi açısından yararlıdır.

Tıbbi araştırmalar: Kemiklerin, kasların, organların, vücuttaki sistemlerin üç boyutlu olarak görselleştirilmesi ve hareket ettirilmesi bu konudaki araştırmalarda kullanılır; hasta ya da yaralıların durumları konusunda daha net bilgilerin elde edilmesini sağlar.

2.4.2 Mühendislik

Mühendisler ve tasarımcılar, karmaşık hareketlerin görüntülenmesinde animasyondan yararlanırlar. Parçaların tasarımında, değişik açılardan bakılması, kesitlerinin alınması ve değişik büyüklüklerde görüntülenmesi olası hataların önüne geçilmesini sağlayacaktır. Şehirlerin, yolların, binaların ve evlerin tasarımında da animasyon kullanılmaktadır. Bir ev tasarlandıktan sonra, yapımına başlanmadan önce, bilgisayar animasyonu yardımı ile çevresinde ve içinde dolaşılabilen, herhangi bir noktadan günün herhangi bir saatindeki güneş ışığı yansımalarına kadar görüntülenebilmektedir. Kara, deniz ve hava araçlarını tasarlayan kişiler, değişik rüzgar ve sıcaklık derecelerine göre tasarladıkları parçaların durumlarını da görme şansını elde edebilirler.

2.4.3 Eğitim ve öğretim

Öğrencilerin okudukları konuları canlı olarak bilgisayarda görmeleri öğrenmelerini kolaylaştıracaktır. Örnek olarak animasyon, fizik dersinde cisimlerin hareketleri, hızları, ivmeleri ve problemlerin çözümleri konusunda kullanılabilir. Kimya derslerinde kimyasal reaksiyonlar, biyoloji derslerinde hücre bölünmeleri, canlıların hareketleri, matematikte vektörler, coğrafyada dünyanın hareketleri, tarihte göçler animasyon yardımı ile anlatılabilir. Birtakım araçların çalışma şekilleri, üç boyutlu kesitleri alınarak gösterilebilir. Yabancı dil öğrenen kişiler sesli animasyonlardan yararlanabilirler.

Kara, deniz, hava ve uzay araçlarının simülatörleri, insanların tehlikeye atılmadan, bu araçların kullanımını öğrenmelerini çok ucuza sağlaması açısından oldukça yararlıdır. Öğrenciler, gerçek araçlara binmeden önce, simülatörleri üzerinde deneyim kazanırlar.

2.4.4 Film endüstrisi

Çizgi film ve bilim-kurgu filmlerinin yapımında bilgisayarlı animasyonların önemi giderek artmaktadır ve son yıllarda birçok filmde etkili bir şekilde kullanılmıştır. Jurassic Park, Total Recall, Robocop, Terminator 2, Mask, Toy Story buna örnek olarak verilebilir.

Jurassic Park, maket ve bilgisayarda modellenmiş dinazorları başarılı bir şekilde birlikte kullanması açısından, filmlerde bilgisayar kullanımını daha farklı bir düzeye getirmiştir. Animasyon ekibi filmdeki dinazor görüntülerinin bilgisayarda gerçekçi şekilde oluşturulabilmesi için önce bir dinazor iskeletini üç boyutlu olarak görüntülemişlerdir. Bu iskeletin üzerinde kas ve en üstte de deri tanımlanmıştır. Dinazorların hareketlerini gerçekçi bir biçimde yansıtabilmek için -ki bu animasyonun en önemli amaçlarından birisidir-, kasların ve derinin davranışları incelenerek en iyiye ulaşılmaya çalışılmıştır. Böylece, dinazorların insan kovalamaları, mutfak sahnesinde koşma, zıplama ve ayaklarının kayması gibi zor hareketler oldukça başarılı bir şekilde gerçekleştirilmiştir (Özgüç, 1995).

Total Recall filminde ise insan iskeleti modellendikten sonra gerçek aktörlerin hareketleri incelenerek, yürüme, koşma, dövüşme gibi hareket dizileri iskelet modellerine uygulanmıştır. Örnek olarak, X-ışını panosunun arkasındaki iskeletlerin hareketi oldukça gerçekçi bir görüntüyle verilmiştir (Özgüç, 1995).

Robocop 2 filminde ise, bilgisayarla oluşturulmuş bir yüz kullanılmıştır. Gerçek bir insanın yüzü kullanılarak alınan veriler üzerinde parametrik fonksiyonlar kullanılarak, yüzdeki mimikler gerçekleştirilmeye çalışılmıştır (Özgüç, 1995).

Tamamı ile bilgisayarda yapılmış ilk çizgi film Toy Story 1995'te Amerika'da gösterime girmiştir. Bugüne kadar bütün çizgi filmlerinde çizgi ustalarını kullanan Disney'in bilgisayarla yaptırdığı ilk çizgi filmin, sinema tarihinde önemli bir dönüm noktası olacağına kesin gözle bakılmaktadır. Daha önceki çizgi filmlerde bir tek sahne için bile yüzlerce çizim yapılması gerekirken, yeni teknoloji ile sahnelerin bir tek tuşa basılarak değiştirilebilmesi, Disney yetkilileri tarafından ilerde yapılacak atılımların başlangıcı olarak görülmektedir.

2.4.5 Reklamcılık

Bilgisayar animasyonunun reklamcılık alanında ilk kullanım şekillerinden birisi, bir binaya yerleştirilmiş bilgisayar denetimli lambalar yardımı ile yazılar geçirilmesi ve hareketli resimler oluşturulmasıdır. Günümüzde ise, özellikle televizyon reklamlarında, film niteliğinde bilgisayar animasyonlarından yararlanılmaktadır.

2.4.6 Sanat

İki boyutlu resimlerin bilgisayar yardımı ile çizilmesinin pekçok üstünlükleri vardır. Örnek olarak, bir anda, bir evin tüm duvarları, 16 milyondan çok daha fazla renk içeren paletten alınan renge boyanabilir. Bu rengi, standart sanat malzemesi ile elde etmek çok zor olabilir. İlgili tüm alanları duyarlı bir şekilde boyamak uzun sürer ve yeni bir renk vermek olanaksızlaşabilir. Bilgisayar ile birlikte, sanatçılar yeni bir gösterim aracı bulmuşlardır. Bilinen sanat dallarında yardımcı bir araç olarak kullanılmakla birlikte, sanata yeni boyutlar da getirmiştir. Örnek olarak, başka bir araç ile kolaylıkla elde edilemeyecek olan üç boyutlu resimler, bilgisayar yardımı ile kolaylıkla çizilebilir. Animasyon yardımı ile, bu şekil hareket ettirilir, döndürülür, ölçeklendirilir ya da üzerinde başka bir işlem uygulanır. İki farklı resim arasındaki geçişin tüm aşamaları bilgisayar aracılığı ile gösterilebilir.

2.4.7 Spor

Animasyon, atletizmden tenise, golften futbola kadar pekçok spor dalında, sporcuların başarımlarını arttırmak için kullanılabilir. Koşucuların hareketleri animasyon haline getirilir. Hatalı hareketler, adım uzunlukları, başarılı ve başarısız koşucuların davranışları gibi birtakım parametreler incelenerek geliştirici çözümler bulunur.

2.4.8 Bilgisayar Oyunları ve Sanal Gerçeklik

Gelişen donanım ve yazılımlar ile birlikte elde edilen gerçekçi bilgisayar oyunları, günümüzde oldukça yaygınlaşmıştır. Bilgisayar oyunları sayesinde, eğitimde kullanılan simülatörlerin benzerleri eğlence amaçlı olarak da kullanılır. Bir arabanın, geminin, uçağın, helikopterin ya da uzay aracının içine binip, onu kullanmak, hızını değiştirmek, dışarıdan araca, aracın içinden dışarıya bakmak olasıdır. Herhangi bir ortamda dolaşma, değişik araçlar kullanma, değişik yönlerde hareket etme, maceralara atılma ya da satrançtan futbola kadar çeşitli konularda oyuncu olma şansı bilgisayar oyunları ve sanal gerçeklik aracılığı ile evlere kadar girmiştir.

2.5 Animasyonun ve Bilgisayar Animasyonunun Yararları

Bilgisayar yardımı ile elde edilen animasyonun el yapımı animasyon üzerine üstünlükleri :

- Nitelik, kolaylık, hız, düşük maliyet :

Bilgisayar yardımı ile oluşturulan resimlerdeki nesnelerin koordinatları doğru olarak hesaplanır ve kolaylıkla ekrana yansıtılır. Daha geniş bir renk paletinden kolaylıkla yararlanmak olasıdır. Bu resimlerin animasyonunda ise, kullanılan veri ve bilgiler yardımı ile kolaylıkla istenilen hareketler verilebilir ve otomatik olarak ilgili hesaplamalar yapılır ve sonraki veya önceki ara resimler elde edilebilir. Hesaplanan değerler, istenilen şekilde ekrana yansıtılır. El üretimi animasyonlarda kullanılan farklı araçların işlevini, bilgisayar, otomatik olarak, hızlı ve nitelikli bir şekilde yerine getirir. Bu nedenle işlemler kolaylaşır, daha az kişi, daha az zaman ve çaba harcayarak, daha ucuza, bilgisayar ortamının tüm üstünlüklerinden yararlanarak animasyonları tamamlarlar. Animasyonu izleyip, resimleri değiştirmek de zor değildir. Bilgisayarlar, animatörlerin elle çok zor yapabilecekleri veya hiç yapamayacakları çizimler için kullanılırlar. Animatörler, iki boyutlu şekillerin çiziminde iyi de olsalar, üç boyutlu şekiller ve bunların hareketleri en deneyimli ellere ve gözlere bile güç gelebilir. Üç boyutlu sanal bir ortamda, bir kameranın, sürekli ürettiği perspektif görüntüleri elle yapabilmek olanaksızdır. Tanımlanmış üç boyutlu nesneler etrafında kamera hareket ederken veya nesneler kendi kendilerine hareket ederlerken, bilgisayar tüm yoğun ve karmaşık matematiksel işlemleri yapabilir. Işık etkileri de bir görüntü karesi içinde kolaylıkla çizilebilir; ancak video veya sinemada olduğu gibi bir kareden diğer bir kareye geçilirken bu etkiler gerçek dışı olarak değişmemelidir. Bilgisayarlar bunun için gerekli olan boyama, tonlama ve aydınlatma işlemlerini kolaylıkla yapabilirler.

Animasyonun üstünlükleri ve önemi :

a) Gerçek yaşamda elde edilemeyecek hareketlerin oluşturulmasını sağlamıştır (Gerçek yaşamda görüntüsünü alamayacağımız nesnelerin animasyonu ve gerçek yaşamda görüntüsü alınsa bile istenilen hareketin verilemeyeceği nesnelerin animasyonu).

Bir uzay aracını belirli bir amaçla uzaya göndermeden önce, gerekli fizik yasaları ve hareketlerin bilgisayara yüklenmesi ile, o ana kadar denenmemiş olan bir işlem görselleştirilip ve önceden sınanabilir.

Günümüzde nesli tükenmiş canlıları (dinozor) ya da gerçek yaşamda bilim-kurgu filmlerindeki nesneleri (robotlar, canlılar ...) görüntülemek olanaksızdır. Akıcı bir halde, her türlü şekle girebilen bir bilim-kurgu öğesinin maket yardımı ile bile oluşturulup gerçekçi bir görünüm ile hareketlendirilmesi hemen hemen olanaksızdır. Bu gibi, yani gerçek aktörün olmadığı durumlarda, yapay aktörlerin animasyonları kullanılır. Yerinde duran bir heykelin ya da eski dönemlerde yapılmış resim ya da kabartmaların bilgisayar yardımı ile hareketlendirilmesi de olasıdır.

b) Gerçek yaşamda elde edilebilecek hareketlerde, gerçekleştirildiğinde can ve mal kaybı gibi tehlikeli sonuçlar doğurabilecek risklerin önüne geçmiş, maliyetleri düşürmüş, işlemleri hızlandırmıştır.

Bir filmde, uçak kazası sahnesinde gerçek veya model uçak kullanmak yerine, bilgisayar ortamında gerçekleştirmek. Gerçek uçak kullanıldığında, doğabilecek risklerin, para ve zaman kaybının yanında, istenilen şekilde gerçekleşmemesi durumunda, aynı aşamalardan yeniden geçmek gerekecektir. Bilgisayarda hazırlanan görüntü ve hareket dizilerinde ise, bunu sağlamak birkaç parametrenin değiştirilmesi ile kolayca yapılabilecektir.

Otomobil firmalarının, üretim aşamasındaki otomobillerin rahatlığını ve bazı koşullardaki dayanıklılığını ölçmek için, bilgisayarda modellerini oluşturup belirlenen hareketleri ve kaza sırasında olabilecek durumları modeller üzerinde uygulayarak ürüne ilişkin bir fikir edinmeleri de bu madde kapsamına girer.

Bir filmde, balinanın açık denize ulaşması için yüksek ve kayalık bir dalgakıran üzerinden denize atlaması gerekmektedir. Ancak, bir balina için o yükseklikteki bir kayanın üzerinden atlamak oldukça zordur. Ayrıca açık denize ulaşan evcil bir balina geriye döndürülemeyebilir. Bu yüzden, bilgisayarla bir balina modeli oluşturulup, bu model kullanılarak bir animasyon hazırlanmıştır. Balinanın sudan çıkma sahnesinin görüntülenmesi için önce bir maket sudan çıkarılmış, bunun arkasına animasyon

eklenmiştir. Maket balina, yeterince esnek ve oluşturulan görüntü çok gerçekçi olmayacağından sahnenin tümünde maket bir balina kullanılması uygun bulunmamıştır.

Okullarda verilen derslere ilişkin konuların daha iyi anlaşılabilmesini sağlayacak olan animasyon paketleri, daha esnek bir şekilde, bilgisayar ortamının getirdiği üstünlüklerden da yararlanarak, maketlerden daha az bir maliyetle elde edilebilir.

c) En tehlikeli işlemleri bile görsel olarak tekrarlamayı olası hale getirerek, eğitim ve öğretimde başarıyı artırmıştır.

Gerçek bir uçağın kalkması, bir süre havada kalıp tekrar inmesi çok maliyet gerektirir. Pilot eğitimlerinde, öğrenilmesi gereken temel konular ile ilgili denemelerde bu işlem defalarca yinelenir. Simülatörler, bu maliyeti ortadan kaldırdığı gibi, risksiz bir şekilde defalarca denemeyi de sağlar. Değişik hava koşullarında, değişik uçaklarla ve değişik parametreler ile denemeler yapılır. Ayrıca kalkışı başaran bir pilot adayı, inişte başarısız oldu ise, kalkış yapmadan, simülatör üzerinde doğrudan iniş çalışması da yapabilir. Yaptığı hataları değişik bakış açılarından izleyerek inceleyebilir.

2.6 Gelecekte Animasyon

- Gelecekte, kişisel bilgisayarlar üzerinde animasyonlar ve filmler kolaylıkla yapılabilecek, yapay aktörler oluşturulabilecektir.

- Ekran çözünürlüğü, renk sayısı ve hız (ekran, işlemci, bellek) gibi faktörlerdeki artış, bilgisayar animasyonunun niteliğini yani gerçekçiliğini artırabilecektir.

- Nesneleri, bilgisayarda sunabilmek için, yeni çevre aygıtları ve yöntemler bulunabilecektir.

- Nesnelere ilişkin, çok geniş, üç boyutlu, bilgi kütüphaneleri oluşturulacak ve bunlar yaygınlaşarak, tekrar kullanımı sağlanabilecektir.

- Bilgisayar animasyonunun kullanıldığı alan türleri ve kullanan kişi sayısı artabilecek ve daha profesyonelce kullanılmaya başlanabilecektir.

- Bilgisayar animasyonunun, bilgisayarla ilgili olan veya olmayan alanlarla etkileşimi sonucu, her alanda gelişmeler olabilecektir.



3. WINDOWS

3.1 Windows Nedir ?

Windows, öğrenmesi ve kullanımı kolay, bilgisayar ve çevre birimleri ile iletişimi sağlayan, grafik tabanlı bir kullanıcı arabirimidir. Yeni çıkan sürümleri ile, tam bir işletim sistemi haline getirilmiştir.

3.2 Pencere Nedir ?

Çerçeve içine alınmış dikdörtgen bir ekran alanına pencere adı verilir. Pencere, kullanıcı ile uygulama arasında görsel bir arayüz oluşturur. Pencere, uygulamanın denetimi altında olup, boyut ve şeklini de içine almak üzere herşeyi uygulama tarafından denetlenir. Windows üzerinde, aynı anda birden çok uygulama çalıştırılabilir. Çok görevlilik adı verilen bu özellik, pencereler yardımı ile kolay bir hale getirilmiştir. Windows'un donanım bağımsız altyapısında da pencerelerden yararlanılmıştır. Windows, uygulamaların fiziksel ekrana doğrudan erişmelerine izin vermez, bütün işlemler pencerenin o andaki alanı içinde, boyutuna bağlı olarak yapılır. Pencere, boyutları da kolaylıkla değiştirilebilir. Windows'ta OLE (Object Linking and Embedding), yani Nesne Bağlama ve Katma desteği ile, uygulamalar arasında bağlantı kurulur.

3.3 Pencere elemanları

Pencere Çerçevesi : Windows'un tüm pencerelerinin etrafında, onu sınırlayan ve diğerlerinden ayıran bir çerçeve vardır.

Başlık Çubuğu : İlgili pencerenin üzerinde, pencere başlığının ortalanmış olarak yer aldığı çubuktur.

Denetim Kutusu ve Denetim Menüsü : Denetim Kutusu, pencerelerin sol üst köşesinde bulunur ve fare imleci onun üzerinde iken farenin tuşu tıklandığında Denetim Menüsü ortaya çıkar.

Menü Çubuğu : Menü seçenekleri, menü çubuğu üzerinde yer alır. Seçenekler, komut listelerini içerir.

Kaydırma Çubuğu : Tümü pencere içine sığmayan belgelerin ya da resimlerin görünebilmesi için kaydırma yapılabilmesini sağlar.

3.4 Nesneye Yönelik Programlama (Object Oriented Programming)

Nesneye yönelik yaklaşımdaki temel düşünce çok basittir. İnsanlar, dünyayı nesneler kümesi şeklinde algırlar. Örnek olarak, otomobile bakan kişi, özel bir atomlar yığını değil, otomobili görür. Otomobil, herbiri nesne olan, gövde, motor, kapı, tekerlek, gibi farklı nesnelere bölünebilir (Murray and Pappas, 1990).

Windows programlama, nesneye yönelik programlamadır. Nesneye yönelik programlamada nesne, veri yapıları ve onlar üzerinde işlemler yapan fonksiyonlardan oluşan bir soyut veri tipidir. Bir grafik nesnesi, veri ve verinin sunumu ile gösterilir. Pencere elemanları içinde değinilen menüler, başlık çubukları, denetim kutuları, kaydırma çubukları, grafik nesnelerinden bazılarıdır. Diğer bazı grafik nesneleri de şunlardır :

Simgeler (Icon) : Simge, kullanıcıya belirli bir işlemi, düşünceyi ya da ürünü hatırlatmaya yarayan küçük bir sembol şeklindeki grafik nesnesidir.

İmleçler (Cursor) : Grafik şeklindeki imleç, bulunduğu yere göre değişik şekiller alarak, yapılabilecek işlemler konusunda bir fikir verir.

İleti Kutuları (Message Box) : Yaygın grafik nesnelerinden birisi de ileti kutularıdır. İleti kutuları, başlık ve ileti içeren pencerelerdir. Kullanıcıya ileti ulaştırılmasını sağlarlar.

Diyalog Kutuları (Dialog Box) : Özellikle kullanıcıdan girdi almak için kullanılan pencerelerdir.

Yazıtipleri (Font) : Bir karakter seti kümesini tanımlar. Yazıtipleri, değişik görüntülerde yazı oluşturmak için kullanılır.

Bitmap'ler (Bitmap) : Noktalar cinsinden belleğe kaydedilmiş görüntü parçasıdır, resmin ikili gösterimidir. Yani, dikdörtgen grafik nesnesini sunan veri dizisidir.

Fırçalar (Brush) : Boyamak ve önceden tanımlanmış şekilde bölge doldurmak için kullanılan bir grafik nesnesidir.

Kalemler (Pen) : Çizgi çizmek için kullanılır. Üç temel bileşeni vardır : Kalınlığı, stili ve rengi.

4. BİLGİSAYAR GRAFİKLERİ

4.1 İki Boyutlu Dönüşüm İşlemleri

İki boyutlu basit şekilleri, özellikleri ile birlikte çizmeyi sağlayan program parçaları ile, değişik görüntüler oluşturulabilir. Bununla birlikte bazı uygulamalarda, şekiller üzerinde çeşitli işlemler yapmak gerekebilir. Şekilleri ekrana yerleştirmek için büyütülmesi veya küçültülmesi istenebildiği gibi, resmin görüntüsünü sınamak için, parçaların göreceli yer ve boyutlarının yeniden düzenlenmesi de istenebilir. Animasyon uygulamalarında, ekrandaki nesnelerin sürekli hareket etmesi sağlanmalıdır. Bütün bunlar, şekillerin koordinat noktaları üzerinde, uygun geometrik dönüşüm işlemlerinin yapılması ile elde edilir. Temel dönüşüm işlemleri, yer değiştirme, ölçeklendirme ve döndürmedir (Hearn and Baker, 1986).

Temel dönüşümlerin değişik kombinasyonlarını kullanan pekçok uygulamalar vardır. Tanımlanmış şekillerden oluşan bir resim, her şeklin ölçeklendirilmesi, döndürülmesi, yer değiştirilmesi ve uygun resim içi yerleştirmelerle oluşturulabilir. Bu dönüşümlerde, bir anda bir işlem yapılır. Önce, nesneyi tanımlayan koordinatlar ölçeklendirilir, ardından ölçeklendirilmiş koordinatlar döndürülür, en son olarak döndürülmüş koordinatlar istenen yere hareket ettirilir. Sonuç koordinatlarını hesaplamanın daha etkin bir yolu, başlangıç koordinatlarından matris yöntemleri yardımı ile doğrudan elde etmektir. Bu durumda temel dönüşüm işlemleri, matris formunda gösterilir. Noktalar tekdüze koordinatlar şeklinde gösterilerek dönüşüm denklemleri matrislere yazılır. Yani, iki boyutlu koordinat konumu (x,y) için bir üçlü

$$\begin{bmatrix} x_h & y_h & w \end{bmatrix} \text{ gerekir} \quad (x_h = x \cdot w \quad y_h = y \cdot w \quad \text{olmak üzere}).$$

w parametresi, dönüşüm sınıfının durumuna göre, 0 dışında bir değer alır. İki boyutlu dönüşümlerde w = 1'dir. Dolayısı ile, her iki boyutlu koordinat konumu (x,y), doğrudan tekdüze koordinat biçiminde gösterilebilir :

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \text{ (w'nun diğer değerleri üç boyutlu dönüşümlerde kullanılır)}$$

$$\begin{aligned} \text{Noktanın yeni konumu (Dönüşümden sonraki)} : \quad P' &= \begin{bmatrix} x' & y' & 1 \end{bmatrix} \\ \text{Noktanın ilk konumu (Dönüşümden önceki)} : \quad P &= \begin{bmatrix} x & y & 1 \end{bmatrix} \end{aligned}$$

Nesneler, koordinat noktalar kümesi ile tanımlanırlar. Örnek olarak, bir çizginin tanımlanması için, iki uç noktasının koordinatlarını belirtmek yeterlidir. Geometrik dönüşüm işlemleri, bu noktaların yeni koordinatlarını hesaplamakta kullanılan program parçalarına yerleştirilebilir (Hearn and Baker, 1986).

4.1.1 Yer değiştirme

Yer değiştirme, bir nesnenin bir doğru üzerinde bir noktadan diğer noktaya getirilmesidir. Koordinatları (x,y) olan bir noktanın, (x',y') olan yeni bir noktaya gitmesi, T_x ve T_y , yer değiştirme uzaklıkları eklenerek sağlanır (Hearn and Baker, 1986) :

$$x' = x + T_x \quad y' = y + T_y$$

Yer değiştirme uzaklık çifti (T_x, T_y) , yer değiştirme vektörü diye adlandırılır. Temel dönüşüm denklemleri, tekdüze formda gösterilen koordinat noktaları ile, 3×3 'lük dönüşüm matrislerinin çarpımı şeklinde de ifade edilir (Hearn and Baker, 1986) :

$$\text{Yer değiştirme için Dönüşüm Matrisi} = T(T_x, T_y) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ T_x & T_y & 1 \end{bmatrix}$$

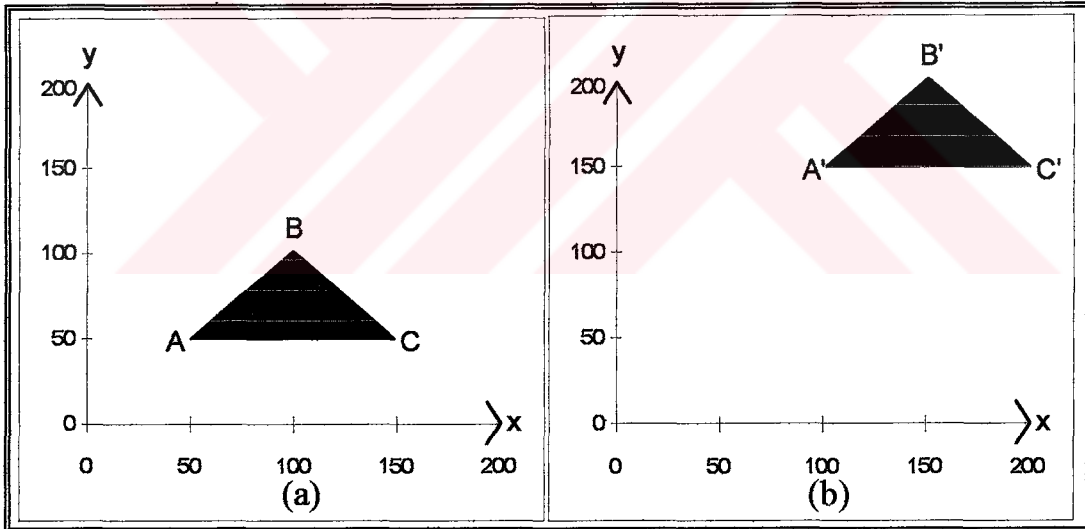
$$P' = P \cdot T(T_x, T_y) \text{ veya,}$$

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ T_x & T_y & 1 \end{bmatrix}$$

Dönüşümlerdeki vektörleri, sütun (kolon) biçiminde gösteren kaynaklar da vardır. Bu durumda, yer değiştirme için dönüşüm işlemi şu şekilde değiştirilecektir :

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & Tx \\ 0 & 1 & Ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Çokgenler, köşelerinin koordinatlarına, belirlenen yer değiştirme uzaklıkları eklenerek yer değiştirirler. Bir üçgenin yer değiştirmesi Şekil 4.1'de gösterilmiştir. Daire ve elips gibi, köşesi olmayan şekiller, merkezlerinin yeri değiştirilip, aynen çizilerek hareket ettirilir. Yer değiştirme uzaklıkları, herhangi bir gerçel sayı olabilir (pozitif, negatif, 0).



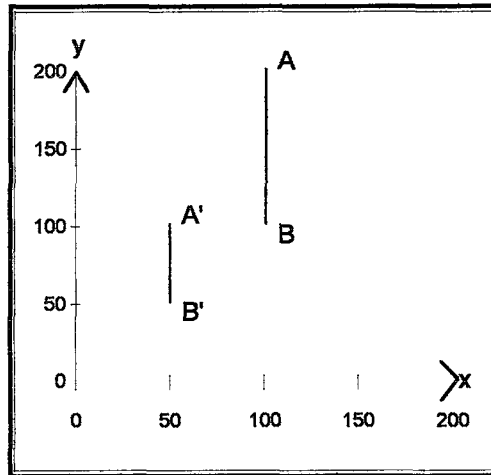
Şekil 4.1 : Yer değiştirme. (a) Üçgen nesnesinin ilk konumundaki köşe noktaları : A(50,50), B(100,100), C(150,50). (b) Üçgen nesnesinin yer değiştirme uzaklıkları $T_x=50$ ve $T_y=100$ olmak üzere, yer değiştirmesinden sonraki köşe noktaları : A'(100,150), B'(150,200), C'(200,150).

4.1.2 Ölçeklendirme

Bir nesnenin boyutunu değiştirmek için yapılan dönüşüm işlemine, ölçeklendirme denilir. Dönüştürülmüş (ölçeklendirilmiş) koordinatları (x',y') elde etmek için, çokgenin uç noktalarının koordinat değerleri (x,y) , ölçeklendirme faktörleri (S_x, S_y) ile çarpılır (Hearn and Baker, 1986) :

$$x' = x \cdot S_x \quad y' = y \cdot S_y$$

Ölçeklendirme faktörü S_x , nesneyi x , S_y ise y doğrultusunda ölçeklendirir. S_x ve S_y , herhangi pozitif sayılar olabilir. Nesnelerin boyutunu küçültmek için 1'den küçük, büyümek için 1'den büyük değerler vermek gerekir. S_x ve S_y için 1 değerinin verilmesi, nesnelerin boyutunu değişmeden bırakır. S_x ve S_y için aynı değerlerin verilmesi, ölçeklendirilmiş nesnelerin, göreceli oranlarının aynı kalmasını sağlar ki buna düzgün ölçeklendirme denilir (Şekil 4.2). S_x ve S_y 'ye farklı değerler verilmesi, birkaç temel şeklin ölçeklendirme yolu ile değiştirilmesinden oluşan resimlerin tasarımında kullanılır (Hearn and Baker, 1986).



Şekil 4.2 : Ölçeklendirme. $[AB]$ doğru parçası, $S_x = S_y = 1/2$ olacak şekilde ölçeklendirilerek, $[A'B']$ doğru parçası elde ediliyor. AB uzaklığı $|AB|$, $A'B'$ uzaklığının $|A'B'|$ iki katıdır. Doğru parçası, önceki boyutunun yarısına indirilmiştir.

$$\text{Ölçeklendirme için Dönüşüm Matrisi} = S(S_x, S_y) = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P' = P \cdot S(S_x, S_y) \text{ veya,}$$

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Ölçeklendirilmiş nesnelerin yerini, sabit nokta denilen ve ölçeklendirme işleminden sonra değişmeden kalan koordinatı seçerek denetleyebiliriz. Sabit noktanın koordinatları (x_F, y_F) , şeklin köşesinde, kenarında, merkezinde veya dışında olabilir. Çokgen, her köşesinin sabit noktaya olan uzaklıklarının ölçeklendirilmesi ile, sabit noktaya göre ölçeklendirilir (Hearn and Baker, 1986) :

$$x' = x_F + (x - x_F)S_x, \quad y' = y_F + (y - y_F)S_y$$

Bu denklemlerdeki deyimleri yeniden düzenleyerek, seçilen sabit bir noktaya göre ölçeklendirilmiş koordinatları elde edebiliriz.

$$x' = x \cdot S_x + (1 - S_x)x_F$$

$$y' = y \cdot S_y + (1 - S_y)y_F$$

Bunun üstünlüğü, $(1 - S_x)x_F$ ile $(1 - S_y)y_F$ 'nin nesnenin tüm noktaları için sabit olmasıdır.

Yer değiştirme işleminde olduğu gibi, ölçeklendirmede de nesneler, ekran koordinat sınırlarının dışına taşabilir. Bu sınırların dışındaki nesneler, kullanılan sistemin durumuna göre kırpılır veya çarpılır (Hearn and Baker, 1986).

4.1.3 Döndürme

Nesnenin noktalarının dairesel yol üzerinde hareket ettirilmesi işlemine döndürme denilir. Bu tür dönüşüm, çokgenin her köşesinin döndürülme miktarını belirten döndürme açısı ile tanımlanır (Hearn and Baker, 1986).

Şekil 4.3'te görüldüğü gibi, (x',y') noktasının, (x,y) noktasından uzaklığı, θ (teta) döndürme açısı ile belirtilir. ϕ (fi), noktanın yatay eksen ile yaptığı açıdır. Bir noktayı döndürmek için kullanılan dönüşüm denklemlerini, dik üçgenin köşeleri ve kenarları arasındaki ilişkilerden çıkarabiliriz. Trigonometrik eşitliklerle birlikte r , noktanın orijinden uzaklığı olmak üzere (Hearn and Baker, 1986) :

$$x' = r \cos(\phi + \theta) = r \cos\phi \cos\theta - r \sin\phi \sin\theta$$

$$y' = r \sin(\phi + \theta) = r \sin\phi \cos\theta + r \cos\phi \sin\theta$$

$$x = r \cos\phi, y = r \sin\phi \text{ olmak üzere :}$$

$$x' = x \cos\theta - y \sin\theta$$

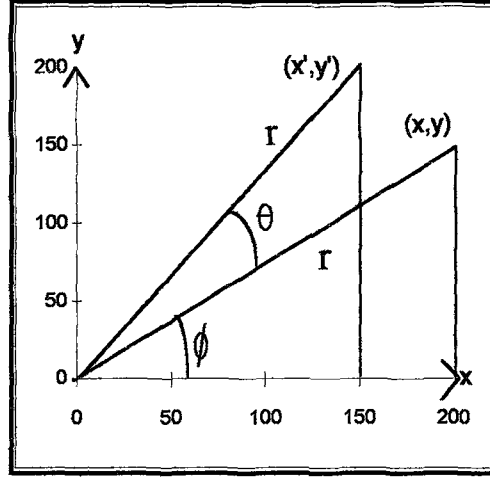
$$y' = y \cos\theta + x \sin\theta$$

Saat hareketinin tersi yönündeki dönmeler pozitifdir yani θ açısının negatif değerleri, nesneleri saat, pozitif değerleri saatin tersi yönünde döndürmeyi sağlar (Şekil 4.3).

$$\text{Döndürme için Dönüşüm Matrisi} = R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P' = P \cdot R(\theta) \text{ veya,}$$

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix} = \begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Şekil 4.3 : Döndürme. (x,y) noktasının, orijine göre, θ döndürme açısı ile döndürülmesi sonucu geldiği konum (x',y').

Nesneleri, seçilen herhangi bir döndürme noktası (x_R, y_R) etrafında döndürmek için denklemler şu şekilde değiştirilebilir (Hearn and Baker, 1986) :

$$x' = x_R + (x - x_R) \cos \theta - (y - y_R) \sin \theta$$

$$y' = y_R + (y - y_R) \cos \theta + (x - x_R) \sin \theta$$

Döndürme noktası, nesnenin içinde veya dışında olabilir. İçinde tanımlanırsa, dönme etkisi, nesnenin bir iç nokta etrafında dönmesi; dışında tanımlanırsa, nesnenin tüm noktalarının, merkezi döndürme noktası olan dairesel bir yol etrafında hareket ettirilmesi biçiminde gözlenir (Hearn and Baker, 1986).

Döndürme hesaplamaları, herbir nokta için pekçok trigonometrik fonksiyon içerdiği için uzun sürer.

4.2 Gerçek Koordinatlardan Araç Koordinatlarına Dönüşüm

Görüntülenecek alan içindeki geometrik şekillerin, ekran gibi araçların koordinat sistemindeki boyutları, gerçekte olduklarından farklı olabilmelidir. Gerçek bir bina ekrandaki görüntüsünden defalarca büyük, bir molekül ise defalarca küçüktür (Ammeraal, 1992). Bu şekilde, değişik büyüklükteki alanlar ekranda görüntülenebilir. Bunun için,

nesnelerin koordinatlarının, gerçek koordinat sisteminden ekran koordinat sistemine eşlenmesi gerekmektedir. $P(x_w, y_w)$ konumunda olan bir noktanın, ekrandaki $P'(x_v, y_v)$ konumuna eşlenmesi için, aşağıdaki orantıların korunması gerekmektedir (Hearn and Baker, 1986) :

$$\frac{x_w - x_{w \min}}{x_{w \max} - x_{w \min}} = \frac{x_v - x_{v \min}}{x_{v \max} - x_{v \min}} \quad \text{ve} \quad \frac{y_w - y_{w \min}}{y_{w \max} - y_{w \min}} = \frac{y_v - y_{v \min}}{y_{v \max} - y_{v \min}}$$

Bu formüllerden, x_v ve y_v şu şekilde hesaplanır :

$$x_v = \frac{x_{v \max} - x_{v \min}}{x_{w \max} - x_{w \min}} (x_w - x_{w \min}) + x_{v \min}$$

$$y_v = \frac{y_{v \max} - y_{v \min}}{y_{w \max} - y_{w \min}} (y_w - y_{w \min}) + y_{v \min}$$

Aynı dönüşümden geçecek koordinatlarda, işlemleri hızlandırmak için, tüm nesnelerde aynı olan S_x ve S_y , bir kere hesaplanarak aşağıdaki formül de kullanılabilir :

$$x_v = S_x * (x_w - x_{w \min}) + x_{v \min}$$

$$y_v = S_y * (y_w - y_{w \min}) + y_{v \min}$$

4.3 Üç Boyutlu Bilgisayar Grafikleri

Üç boyutlu bilgisayar grafikleri pekçok alanda kullanılmaktadır. Tasarımların bilgisayar yardımı ile yapılabilmesini sağlayan CAD (Computer Aided Design = Bilgisayar Destekli Tasarım), en büyük kullanım alanlarından biridir. Üç boyutlu bilgisayar grafikleri yardımı ile ayrıca, grafikler çizilebilir; fiziksel, ekonomik vs. sistemlerin modelleri oluşturulabilir.

Üç boyutlu bilgisayar uygulamaları, varolan (gerçek) nesnelerin gösterimi veya yeni şekillerin tasarımı diye iki farklı sınıfa ayrılabilir. Varolan üç boyutlu bir nesnenin

gösterimi, onu bilgisayarda göstermek için oluşturulan yaklaşık tanımıdır. Örnek olarak, katı bir nesne, çerçeve (iskelet) çizgileri veya düzgün yüzeyler şeklinde sunulabilir. Bilgisayar destekli tasarım uygulamalarında ise amaç, yeni üç boyutlu şekiller oluşturmaktır. Otomobil, gemi, uçak, uzay aracı gövdeleri, makina parçaları, elektrik, elektronik devreleri, binalar, yüzey desenleri, belirli tasarım kısıtlarına ulaşmaya kadar yeniden düzenlenir. Üç boyutlu uygulamaların hepsinde, üç boyutlu nesnelerin tanımları, üç boyutlu gerçek dünya koordinat sisteminde tanımlanıp, iki boyutlu ekran koordinatlarına dönüştürülür. İki boyutlu uygulamalarda ise sadece, gerçek koordinat düzleminde araç koordinat düzlemine çevrim vardır.

Genelde model, yapay olarak oluşturulmuş, başka bir nesneyi kolaylıkla gözlemlemeyi sağlayan bir nesnedir (Mäntylä, 1988). Gözlemlemeyi olası hale getirmek için, bina, gemi, araba gibi üç boyutlu fiziksel nesnelerin, göreceli boyutları ve genel görünüşleri korunarak fiziksel modelleri oluşturulur (bunlar gerçek boyutlarında değildir). Kimyada, molekül içindeki atomların, diğerleri ile olan göreceli dizilişleri de, molekül modelleri yardımıyla gözlemlenebilir (moleküllerin bunlar dışında pekçok özelliklerinin olması önemli değildir). Bilim ve mühendisliğin değişik alanlarında yaygın olarak kullanılan matematiksel modeller, sayısal veri ve denklemler aracılığı ile modellenmiş bir olayın davranışlarını sunar. Bilgisayar üzerinde hazırlanan modellere bilgisayar modeli denilir ve bunlar, gerekli olan verileri içerir. Mühendislik çizimlerini de içine alan pekçok alanda kullanılır. Geometrik modelleme, nesne modelinin geometrik kısmını içeren bir alt kümesidir. Fiziksel nesnenin tam matematiksel tanımının oluşturulma işlemidir. "Bakış noktasından, nesnelerin hangi parçaları görülebilir ve ne renktedir" gibi geometrik problemler, bu konunun bir parçasıdır. Katı modelleme, geometrik modellemenin bir dalı olup, fiziksel katı nesnelerin tam sunumunu içerir.

4.3.1 Üç boyutlu koordinat sistemi

Üç boyutlu nesneleri tanımlamak için, 3 boyutlu (3D) koordinat sistemini kullanmak gerekir. Bu sistemi anlamamanın en kolay yolu, 2 boyutlu (2D) Kartezyen Koordinat Sistemini incelemektir. İlk olarak, orijin adı verilen başlangıç noktasını belirlemek gerekir. Yatay olan x , dikey olan y eksenidir ve bu eksenler birbirine diktir. Artık, 2D bir yüzey üzerindeki herhangi bir nokta, koordinatları ile tanımlanabilir. Noktanın x eksenine boyunca olan uzaklığı x , y eksenine boyunca olan uzaklığı y koordinatı olarak adlandırılır ve noktanın

konumu (x,y) şeklinde gösterilir (Barkakati, 1993). Genel olarak kullanılan 3D Kartezyen Koordinat Sistemi, 2 boyutlunun genişletilmiş halidir. 3D Sistem, 2 boyutlu sisteme, x ve y eksenlerinin oluşturduğu düzleme dik olan z ekseninin eklenmiş halidir. Pozitif x eksenini sağa, pozitif y eksenini yukarıya doğru giderken, z eksenini iki durumda olabilir : Sayfadan öne doğru geliyorsa Sağ El Koordinat Sistemi (Sağ elin parmakları pozitif x ekseninden pozitif y eksenine doğru uzanırken, diğerlerine dik olan başparmağın gösterdiği yön z eksenini olur); Sayfadan arkaya doğru gidiyorsa Sol El Koordinat Sistemi olarak adlandırılır. Sağ el koordinat sistemi, 3D grafik ve mühendislik uygulamalarında kabul edilmiş bir standarttır.

4.3.2 Üç boyutlu görüntüleme teknikleri

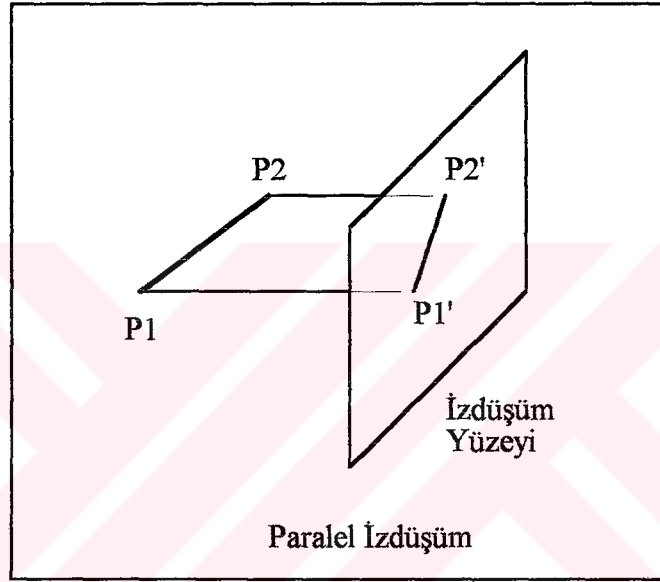
İzdüşüm, uzayın boyutunu azaltan dönüşüm işlemidir. Grafikte kullanılan izdüşümler, üç boyutlu uzayı iki boyuta indirerek, üç boyutlu görüntülerin iki boyutlu görüntü yüzeylerinde sunulmasını sağlar. Görüntü yüzeyleri genelde düzlem şeklindedir. Nesneler üzerindeki noktalardan çıkan izdüşüm doğrularının düzlem ile kesiştiği noktalar, izdüşümün oluşmasını sağlar.

Üç boyutlu nesnelerin, iki boyutlu görüntü yüzeylerindeki sunumu sırasında, nesnelerin arka yüzeyleri, ya da diğer nesnelerin arkasında kalan nesneler görülmeyecektir. Gerçekçi görüntülerde, uzakta olan nesnelerin, yakında olanlardan küçük görünmesi istenebilir. Bu tür işlemleri gerçekleştirmek üzere, çeşitli teknikler geliştirilmiştir. İzdüşüm için kullanılan iki farklı yöntem paralel izdüşüm ve perspektif izdüşümdür. Görünmeyen yüzeylerin elenmesine yönelik kullanılan algoritmalarda, başlıca iki yaklaşım vardır : Nesne uzayına dayalı, görüntü uzayına dayalı. Nesne uzayını gözönüne alan yöntemlerde nesneler ve nesne parçalarından görünmez olanlar işaretlenip çizilmezler. Görüntü uzayı algoritmalarında ise, izdüşüm düzlemindeki her nokta incelenerek, görünürlüğüne karar verilir.

4.3.2.1 Paralel İzdüşüm (Parallel Projection)

Katı bir nesnenin, nesne yüzeyleri üzerindeki noktalardan, düzlemsel görüntü (izdüşüm) yüzeyine paralel doğrular uzatılıp izdüşümleri alınarak iki boyutta gösterimine

paralel izdüşüm denilir. Nesnelere, değişik yer ve bakış açılarından bakılarak, değişik yüzeyleri gözlemlenebilir. Bu teknik, nesnelerin göreceli oranlarını koruyarak, değişik yer ve bakış açılarından bakılarak elde edilen görüntülerden yararlanan mimari ve mühendislik uygulamalarında kullanılır. İzdüşüm yönü, izdüşüm düzlemine dik ise, dönüşüm denklemleri doğrudandır. Herhangi bir (x,y,z) noktası için, xy düzleminde olan görüntü yüzeyindeki izdüşüm noktası (x_p,y_p,z_p) şu şekilde elde edilir (Şekil 4.4): $x_p=x$, $y_p=y$, $z_p=0$ (Hearn and Baker, 1986).

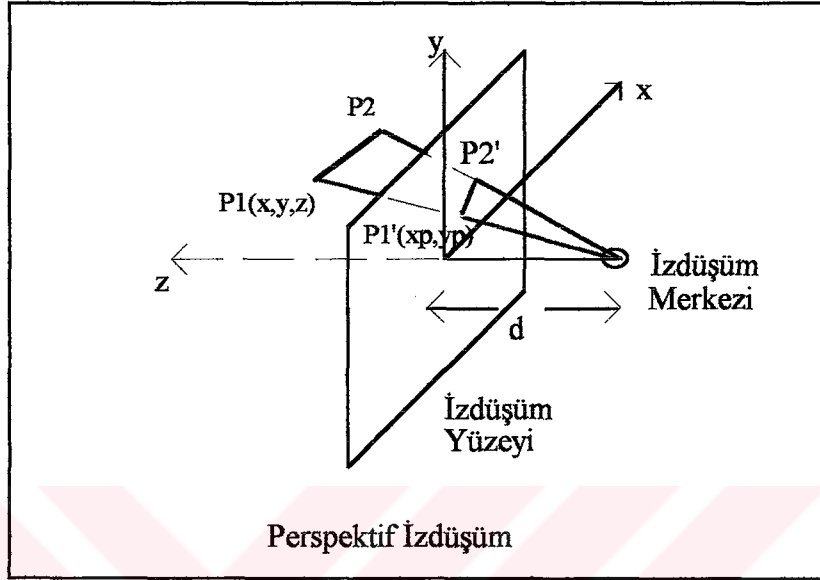


Şekil 4.4 : Paralel İzdüşüm'de, $[P_1P_2]$ doğru parçası, izdüşüm yüzeyinde $[P_1'P_2']$ doğru parçası şeklinde görülür.

4.3.2.2 Perspektif İzdüşüm (Perspective Projection)

İki boyutlu görüntü yüzeylerinde, derinlik görünümü uyandıran bir yöntemdir. Paralel izdüşümdeki paralel doğruların yerini, nesnenin noktalarından çıkıp izdüşüm merkezi ya da bakış noktası denilen bir noktaya yaklaşan doğrular almıştır. Böylece, bakış noktasına uzak olan nesneler, yakın olanlardan daha küçük görünür (Watt, 1990). Göz ve kameraların resimleri oluşturma yöntemine daha yakın olduğu için, daha gerçekçidir. Şekil 4.5'te izdüşüm merkezi, negatif z ekseninde, izdüşüm düzleminden d uzaklığında geridedir. İzdüşüm merkezi için herhangi bir nokta kullanılabilir ama, z ekseninde olması, dönüşüm

işlemlerini kolaylaştıracaktır. Perspektif izdüşümde, izdüşüm merkezinin sonsuzda yer aldığı durum, paralel izdüşümdür (Hearn and Baker, 1986).



Şekil 4.5 : Perspektif İzdüşüm'de, [P₁P₂] doğru parçası, izdüşüm yüzeyinde [P₁'P₂'] doğru parçası şeklinde görülür.

$$x_p = x * \left(\frac{d}{z+d} \right) = x * \left(\frac{1}{z/d+1} \right)$$

$$y_p = y * \left(\frac{d}{z+d} \right) = y * \left(\frac{1}{z/d+1} \right)$$

$$z_p = 0$$

Üç boyutlu koordinat gösteriminde, perspektif dönüşüm (Hearn and Baker, 1986) :

$$\begin{bmatrix} x_h & y_h & z_h & w \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1/d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$(w = z/d + 1 \text{ olmak üzere}) \begin{bmatrix} x_p & y_p & z_p & 1 \end{bmatrix} = \begin{bmatrix} x_h/w & y_h/w & z_h/w & 1 \end{bmatrix}$$

4.3.2.3 Arka Yüzeylerin Elenmesi (Back-Face Removal)

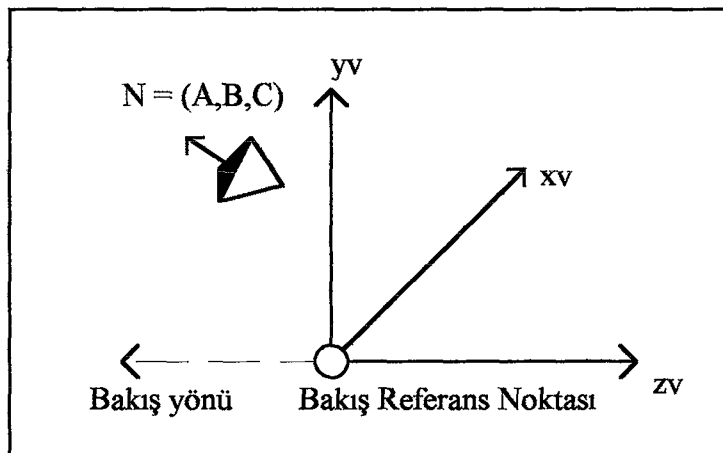
Nesnelerin arka yüzeylerini nesne uzayı yaklaşımı ile belirleyen, düzlem denklemine dayalı basit bir yöntemdir (Hearn and Baker, 1986) :

$$Ax + By + Cz + D = 0 \text{ (Düzlem Denklemi).}$$

Sağ el koordinat sisteminde tanımlanmış herhangi bir (x',y',z') noktası, aşağıdaki koşulu sağlarsa, düzlem içindedir (Hearn and Baker, 1986) :

$$Ax' + By' + Cz' + D < 0$$

Eğer (x',y',z') noktası bakış noktası ise, yukarıdaki koşulu sağlayan düzlemler arka yüzeydir ve bakış noktasından görünmez. Yüzey normal vektörüne bakılarak, basit bir arka yüzey sınaması yapılabilir. Normal vektörü, (A,B,C) kartezyen bileşenlerini içerir. Normal vektörünün C bileşeni, bakış yönüne paraleldir. Eğer $C < 0$ koşulu sağlanırsa, normal vektörü bakış noktasından uzaklaşıyor demektir ve arka yüzeydedir (Şekil 4.6) (Hearn and Baker, 1986).



Şekil 4.6 : Arka Yüzeylerin Elenmesi.

4.3.2.4 Görünmeyen Yüzeylerin Elenmesi (Hidden-Surface Removal)

Görünmeyen yüzeylerin elenmesinde, görüntü uzayı ve nesne uzayı yaklaşımlarını kullanmak olasıdır. Her iki yaklaşımı da kullanan Derinlik Sıralaması Yöntemi (Depth-Sorting Method), kolay ve az sayıda yüzeyden oluşan (birkaç bine kadar) görüntüler için oldukça etkindir. Aşağıdaki temel fonksiyonları yapar (Hearn and Baker, 1986) :

1. Yüzeyler, azalan derinlikte sıralanır (uzaktan yakına doğru).
2. En uzaktakinden başlanarak çizilir.

Sıralama işlemi nesne uzayında, çokgen yüzeylerin çizimi görüntü uzayındadır.

Bu yönteme Ressam Algoritması (painter's algorithm) da denmektedir. Yağlı boya resimlerin yapılmasında, önce arka plan renkleri boyanır. Sonra uzaktaki nesneler çizilir. Sonra da öndeki nesneler, arkadaki nesnelerin üzerine çizilir. Her boya katmanı, bir önceki katmanı kaplar. Benzer teknikle, Derinlik Sıralamasında, görüntü yüzeyinden uzaklığına göre yüzeyler sıralanır. Uzak yüzeylerin yoğunluk değerleri eklenir. Bu yoğunluk değerleri üzerine, daha yakındakilerin yoğunluk değerleri eklenir (Hearn and Baker, 1986).

4.4 Üç Boyutlu Dönüşüm İşlemleri

Üç boyutlu yer değiştirme, ölçeklendirme ve döndürme yöntemleri, iki boyutlu yöntemlerin, z eksenini de gözönüne alınarak genişletilmiş şeklidir. Yer değiştirme için, üç boyutlu yer değiştirme vektörü kullanılır. Ölçeklendirme de üç ölçeklendirme faktörü ile yapılır. Üç boyutlu dönüşüm işlemlerinden döndürme, daha az doğrudandır. Geometrik dönüşüm denklemleri, iki boyutta olduğu gibi, dönüşüm matrisleri yardımı ile gösterilir. Bir dizi dönüşüm, matrislerin birleştirilmesi ile tek matriste gösterilir (Hearn and Baker, 1986).

4.4.1 Yer deęiřtirme

Üç boyutlu tekdüze koordinat gösteriminde, $P(x,y,z)$ koordinatından $P(x',y',z')$ koordinatına

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix}$$

matris işlemi yardımı ile hareket ettirilir. T_x , T_y , T_z parametreleri, eksenlerdeki yer deęiřtirme uzaklıklarıdır. Bu matris işlemi ile (Hearn and Baker, 1986) :

$$x' = x + T_x \quad y' = y + T_y \quad z' = z + T_z \quad \text{denklemleri gerekleřtirilir.}$$

Bir nesnenin yeri, tanımlı her noktası üzerinde dönüşüm işlemi uygulanarak deęiřtirilir. Çokgen yüzeyler kümesi řeklinde gösterilen nesnenin yerinin deęiřtirilmesi de, her yüzeyinin her köşesindeki koordinat deęerlerinin yerlerinin deęiřtirilmesi ile saęlanır. Köşelerin yer deęiřtirmiş koordinat deęerlerinin kümesi, nesnenin yeni yerini tanımlar. Nesneyi eski yerine getirmek için $-T_x$, $-T_y$, $-T_z$ ile elde edilen yer deęiřtirme matrisi uygulanır. Bu, ters yönde bir yer deęiřtirme olup, yer deęiřtirme matrisi ile tersinin çarpımı birim matrisi verir (Hearn and Baker, 1986).

4.4.2 Öleklendirme

Üç boyutlu öleklendirme için kullanılan matris işlemi (orijine göre) :

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

şeklindedir. Ölçeklendirme parametreleri S_x , S_y , S_z , pozitif değerler alabilirler. Bu matris işlemi, yeri $P(x,y,z)$ 'de olan bir noktayı, yeri $P(x',y',z')$ olan duruma gelecek şekilde ölçeklendirir ve ölçeklendirme denklemleri (Hearn and Baker, 1986):

$$x' = x \cdot S_x \quad y' = y \cdot S_y \quad z' = z \cdot S_z \quad \text{şeklindedir.}$$

Bir nesne, tüm noktalara uygulanan ölçeklendirme ile ölçeklendirilir ve orijine göre hareket eder. Ölçeklendirme parametrelerinin hepsi eşit değilse, nesnenin göreceli boyutları da değişir. Düzgün ölçeklendirme ($S_x = S_y = S_z$), nesnenin orjinal şeklini korur (Hearn and Baker, 1986).

Sabit bir $P(x_F, y_F, z_F)$ noktasına göre ölçeklendirme, üç dönüşüm işleminin ardarda uygulanması ile gerçekleştirilir.

- 1) Sabit noktayı orijine taşıma
- 2) Ölçeklendirme
- 3) Sabit noktayı eski konumuna geri getirme

$$T(-x_F, -y_F, -z_F) \cdot S(S_x, S_y, S_z) \cdot T(x_F, y_F, z_F)$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_F & -y_F & -z_F & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x_F & y_F & z_F & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_F & -y_F & -z_F & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ x_F & y_F & z_F & 1 \end{bmatrix}$$

$$= \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ (1-S_x)x_F & (1-S_y)y_F & (1-S_z)z_F & 1 \end{bmatrix}$$

Ters ölçeklendirme matrisi, ölçeklendirme parametreleri S_x , S_y , S_z yerine $1/S_x$, $1/S_y$, $1/S_z$ karşılıkları yerleştirilerek oluşturulur ve noktayı eski haline geri döndürür. Ölçeklendirme matrisi ile tersinin çarpımı birim matrisi verir (Hearn and Baker, 1986).

4.4.3 Döndürme

Bir nesneyi döndürebilmek için, nesnenin döndürüleceği döndürme eksenini ve döndürme açısının miktarı belirlenmelidir. İki boyutlu uygulamalarda döndürme eksenini, xy düzlemine dik olur. Üç boyutta ise, döndürme eksenini herhangi bir uzaysal yönde olabilir. En kolay döndürme eksenleri, koordinat eksenlerine paralel olanlardır. Herhangi bir döndürme eksenindeki dönme, x, y, z eksenlerindeki döndürme işlemlerinin bileşimi ile sağlanır. Saatin tersi yönündeki dönmeler, pozitif dönme açıları ile sağlanır (eksenin pozitif tarafından orijine doğru bakıldığında). Üç boyutlu z ekseninde dönme denklemleri, iki boyutlu denklemlerin hemen hemen aynısıdır.

$$x' = x \cos\theta - y \sin\theta$$

$$y' = x \sin\theta + y \cos\theta$$

$$z' = z$$

θ parametresi, dönme açısıdır. Tekdüze koordinat yapısında, üç boyutlu z eksenine göre döndürme denklemlerinin gösterimi şu şekildedir. (Hearn and Baker, 1986) :

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Diğer iki eksen etrafında döndürme denklemleri, z eksenini etrafında döndürme için kullanılan parametrelerin, dairesel permütasyonu ile elde edilir.

x eksenini etrafında döndürme için x yerine y, y yerine z, z yerine x kullanılarak denklemler elde edilir (Hearn and Baker, 1986) :

$$y' = y \cos \theta - z \sin \theta$$

$$z' = y \sin \theta + z \cos \theta$$

$$x' = x$$

Tekdüze koordinat yapısı :

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

y eksenini etrafında döndürme :

$$z' = z \cos \theta - x \sin \theta$$

$$x' = z \sin \theta + x \cos \theta$$

$$y' = y$$

y eksenini etrafında döndürmenin matris gösterimi :

$$\begin{bmatrix} x' & y' & z' & 1 \end{bmatrix} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{bmatrix} \cos \theta & 0 & -\sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ters döndürme matrisi, θ dönme açısı yerine $-\theta$ konularak oluşturulur. Dönme açısının negatif değerleri, saat yönünde dönmeyi sağlar ve herhangi bir döndürme matrisi, tersi ile çarpılırsa birim matrisi verir. Dönme açısının işaretinin değişiminden yalnızca sinüs fonksiyonu etkilendiği için ters matris, satır ve sütunların yerlerinin değiştirilmesi ile elde edilebilir. Böylece, bir döndürme matrisinin tersi, transpozu alınarak elde edilir ($R^{-1} = R^T$). Bu yöntem, bileşik döndürme matrisleri için de geçerlidir (Hearn and Baker, 1986).

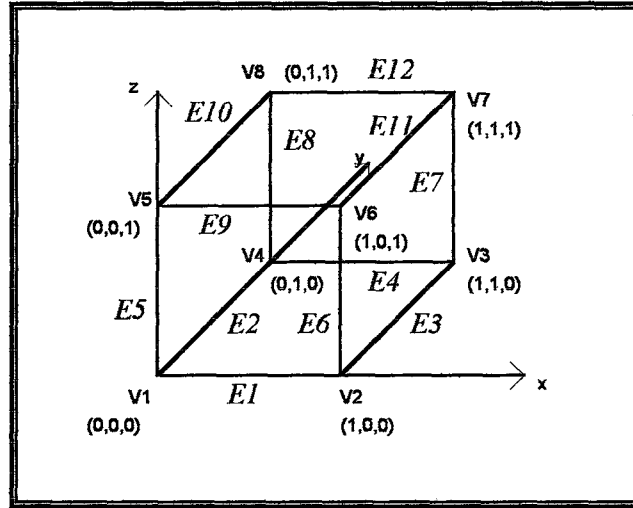
4.5 Katı Modelleme

Katı nesneler, kenarlarını oluşturan çizgiler yardımı ile ya da yüzey modelleme teknikleri ile sunulabilir. Üçüncü modelleme gösterimi, katı modelleme olup, nesnenin tam ve belirli bir tanımını verir, üzerinde işlemler yapmayı sağlar. Sınırlarının şekli yanında, resmin iç ve dış bölgelerini de tanımlar. Bu özelliği, nesnenin, kenarlarını oluşturan çizgiler yardımı ile sunumunda görülen, nesnenin yüzeylerinin hangileri olduğunun anlaşılamaması gibi belirsizlikleri giderir. Yüzey modelleri de, nesnenin tam tanımını içermez. Çünkü uzaydaki noktalar, nesnelerin içinde veya dışında olma durumlarına göre sınıflandırılmaz; sadece sınırları içindeki noktalar sözkonusudur. Katı modeller oluşturmak için kullanılan sunumlardan birisi olan Sınır Temsili (Boundary Representation) bir sonraki konuda incelenmiştir. Katı modelleme kuramında geometrik bilgi, çokyüzlü nesnelerin köşe koordinatları (x,y,z) gibi şekil tanımlama parametrelerini içerir. Katı modellemede komşuluk topolojisi ise, değişik geometrik bileşenler arasındaki bağlantıyı tanımlar ve nesnenin farklı parçaları arasındaki ilişkisel bilgidir (Anand, 1993).

4.6 Üç Boyutlu Nesnelerin Sınır Temsili

Üç boyutlu nesneleri modellemenin bir yolu, nesnenin sınırlarını oluşturan yüzeyleri tanımlamaktır. Gerçekçi nesneler, eğrisel yüzeyler içerseler de, eğrisel yüzeyleri de çokgenler yardımı ile yaklaşık olarak elde etmek olasıdır. Küp gibi basit nesneler, doğal olarak yüzeyler ile sınırlandırılmış olmasına rağmen, silindir ve küre gibi eğrisel yüzeyler, ancak çok sayıda düzlemsel yüzeylere bölünerek modellenebilir (Hearn and Baker, 1986).

Bir çokgen, köşe noktaları yardımı ile tanımlanır. Üç boyutlu uzayda bir nokta da, x , y , z koordinatları ile tanımlanır ve (x,y,z) şeklinde gösterilir. Şekil 4.7'deki, tüm kenarları 1 birim olan küpün (üç boyutlu bir nesnedir), Şekil 4.8'de, geometrik veri tabloları ile (köşe listesi, kenar listesi ve çokgen yüzey listesi) kenar tabanlı sınır temsili görülmektedir. Köşe listesindeki her köşe, üç boyutlu uzayda (x,y,z) bileşenleri ile tanımlanır. Nesnenin tanımlı her kenarı, kenar listesinde görüldüğü gibi, birer doğru parçası olduğu için, iki uç köşesinden oluşmaktadır. Çokgen yüzey listesinde de, nesnenin sınırlarını oluşturan tüm yüzeyler, ikiden fazla kenarın kapalı bir alan oluşturacak şekilde ve sıralı olarak belirtilmesi ile tanımlanır. Küp, dikdörtgen yüzeylerden oluştuğu için, her yüzeyi 4'er kenardan oluşur.



Şekil 4.7 : Kenarları 1 birim olan küp nesnesi.

Köşe Listesi		Kenar Listesi		Çokgen Yüzey Listesi	
Köşe No	Koordinatlar	Kenar No	Köşeler	Çokgen No	Kenarlar
V1	0,0,0	E1	V1,V2	S1	E1,E3,E4,E2
V2	1,0,0	E2	V1,V4	S2	E5,E9,E6,E1
V3	1,1,0	E3	V2,V3	S3	E6,E11,E7,E3
V4	0,1,0	E4	V3,V4	S4	E7,E12,E8,E4
V5	0,0,1	E5	V1,V5	S5	E8,E10,E5,E2
V6	1,0,1	E6	V2,V6	S6	E9,E10,E12,E11
V7	1,1,1	E7	V3,V7		
V8	0,1,1	E8	V4,V8		
		E9	V5,V6		
		E10	V5,V8		
		E11	V6,V7		
		E12	V7,V8		

Şekil 4.8 : Kenarları 1 birim olan küp nesnesinin, köşe listesi, kenar listesi ve çokgen yüzey listesi yardımı ile kenar tabanlı sınır temsili.

Geometrik veri tabloları, çokgen yüzeylerin uzaysal yerini belirleyen sınır koordinatlarını içerir. Ayrıca, nitelikler veya nitelik tabloları ekleyerek, yüzeylere renk ve benzeri bilgiler atamak da olasıdır.

Kenar tabanlı sınır temsiliinde, kenar bilgileri de yer aldığı için, kenarlara nitelik vermeyi ve kenarlar üzerinde işlemler yapmayı kolaylaştırır. Eğrisel yüzeyler için de, kesişim kenarları sadece bir kere tanımlanarak tekrardan kaçınılmış olur. Diğer bir sınır temsili modeli olan köşe tabanlı sınır temsiliinde ise kenar listesi yoktur. Çokgen yüzey listesindeki kenarlar yerine de, yüzeyi oluşturan köşeler sıra ile verilir.

4.7 Bilgisayar Grafiklerinde Veri Yapıları

Veri yapıları, mühendislikten çok, bilgisayar bilimleri ile ilgili de olsa, bilgisayar grafiklerinde kullanılan geometrik verinin, etkin kullanım için nasıl düzenleneceğinin bilinmesi önemlidir. Bir nesnenin bilgisayar modelinde üç tür bileşen vardır : Veri, algoritma ve yapı. Veri, model tanımının en temel elemanları olan sayısal değerleri, karakterleri, komutları içerir. Algoritmalar, verinin nasıl işleneceğini, yapı ise nasıl düzenleneceğini gösterir. Yapı iyi tasarlanırsa, algoritma yapılandırmasına duyulan gereksinim azalır. Verinin yapılandırılması, veri tabanının önemli bir konusu olup, sonraki kullanımlar için önemlidir. Bu, özellikle bilgisayar grafiklerinde ve tasarım sistemlerinde doğrudur. Çünkü, çok sayıda verinin çok yüksek hızlarda işlenmesi gerekmektedir. CAD uygulamalarındaki veri tabanlarının büyük bir bölümü, geometrik verileri içerir (Anand, 1993).

Bilgisayar grafiklerinde kullanılan en yaygın veri tabanları, ilişkisel veri tabanları, sıradüzensel veri tabanları ve ağ veri tabanlarıdır. Bunlar, çeşitli veri tipleri ve veri yapılarından oluşur. Karakterler, tamsayılar, gerçel sayılar çok sıklıkla kullanılan veri tipleridir. Diğer veri tiplerinin birleştirilmesi ile oluşan yapılar, dizinler, bağlaçlı listeler, ve ağaçlar da, grafik programlarında önemli yer tutarlar (Anand, 1993).

4.8 Vektör İşlemlerine Giriş

Vektör, belirlenmiş iki nokta arasındaki yönlü doğru parçasıdır. Vektör toplama ve çarpma işlemleri, bilgisayar grafiklerinde yoğun olarak kullanılır. Vektör toplama, iki vektörün karşılık gelen elemanları toplanarak gerçekleştirilir.

Vektör Toplama :

$$V_1 = (x_1, y_1, z_1) \quad V_2 = (x_2, y_2, z_2)$$

$$V_1 + V_2 = (x_1 + x_2, y_1 + y_2, z_1 + z_2)$$

Vektör çarpımı ise iki yol ile yapılabilir :

Skaler Çarpım : Sonuçta, bir sayısal değer elde edilir. Vektörlerin arasına "." konularak gösterilir.

$$V_1 \cdot V_2 = x_1 x_2 + y_1 y_2 + z_1 z_2$$

Bir vektörün kendisi ile skaler çarpımının karekökü vektörün uzunluğunu verir.

$$|V| = (V \cdot V)^{1/2} = (x^2 + y^2 + z^2)^{1/2} \quad (\text{Pisagor Teoremi})$$

İki vektörün skaler çarpımı, aralarındaki açılardan yararlanarak da yazılabilir.

$$V_1 \cdot V_2 = |V_1| |V_2| \cos \theta$$

Bu durumda skaler çarpımın, bir vektörün uzunluğu ile, diğer vektörün izdüşüm uzunluğunun çarpımını temsil ettiği anlaşılır.

İki vektörün vektörel çarpımı ile ($V_1 \times V_2$), orjinal iki vektöre dik, diğer bir vektör elde edilir ve sağ elin parmakları, V_1 'den V_2 'ye doğru kıvrılırken, bunlara dik olan başparmağın gösterdiği yöndedir.

$$\mathbf{V}_1 \times \mathbf{V}_2 = u \, |\mathbf{V}_1| \, |\mathbf{V}_2| \sin\theta$$

$$\mathbf{V}_1 \times \mathbf{V}_2 = \begin{vmatrix} \mathbf{u}_x & \mathbf{u}_y & \mathbf{u}_z \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{vmatrix}$$

Vektörel çarpım için kullanılan bu determinant formu, bileşenleri ile birlikte sonuç vektörü üretir : $(y_1z_2 - z_1y_2, z_1x_2 - x_1z_2, x_1y_2 - y_1x_2)$.



5. BORLAND C++ ile WINDOWS PROGRAMLAMA

Windows, kullanıcılara grafik tabanlı bir arayüz sunar. Uygulamalarında, grafik ve animasyonlar kolaylıkla kullanılabilir. Grafik tabanlı olmayan arayüzler, grafik ve resimlerle çalışmak için pek uygun değildir. Uygulama geliştirenler, 3 boyutlu grafik ve animasyon gibi konularda birbirleri ile yarışır hale gelmişlerdir. Windows API, resim çizme ve gösterme olanakları sunmakla birlikte, animasyon için aynı kolaylıkları sağlamaz. Animasyonu doğrudan yapmaması nedeni ile programcılar, nitelikli animasyonlar oluşturabilmek için çok sayıda konuyu öğrenmek zorundadırlar : 2 ve 3 boyutlu grafiklerle ilgili işlemler, animasyon teknikleri, resim dosya biçimleri, veri yapıları gibi... Ayrıca, C++ dilinin özelliklerinden yararlanmak amacı ile, uygulamalarda, grafik ve animasyon işlemlerinde büyük kolaylıklar sağlayan C++ sınıfları ile de çalışmayı öğrenmeleri gerekir.

5.1 Windows Programlama Temelleri

DOS'taki geleneksel sıralı (sequential) programlama stiline alışkın olan kişiler, Windows'un olay merkezli (event-driven) programlamasını anlamakta zorlanabilirler. Windows Programlamasını öğrenmek için, olay merkezli programlamayı öğrenmek gerekir. Windows Programları, program kodunun yazılma sırası ile değil, olayların oluşu ile denetlenir. Olay merkezli programlama, olan olaylar ile ilgili iletilerin (mesaj) üretimi ve işlenmesi çerçevesinde çalışır. Örnek olarak farenin veya klavyenin bir tuşunun basılı olması ya da farenin hareket ettirilmesi durumunda bir ileti gönderilir. Windows programcısı, bu iletilerle ilgili işlemleri denetleyebilecek düzeneği kurabilmelidir. Bu yapı pekçok üstünlüğü de beraberinde getirir. Örnek olarak, sabit disk ile ilgili bir hata olduğunda, program nerede olursa olsun bunu gideren fonksiyonlara atlar ve gerekeni yapar. Her fonksiyonda bu tür hataların olup olmadığını tekrar tekrar denetlemesi gerekmez.

5.2 Çoklu-Ortam (Multimedia)

Bütünleştirilmiş ses ve hareketli görüntülerle çalışmayı sağlayan Çoklu-Ortam'ın birkaç yıllık bir geçmişi olmasına rağmen, CD-ROM ansiklopedi ve sözlükleri, iş sunumları, ses ve hareketli görüntüler içeren oyunlar gibi pekçok uygulama yazılmıştır. CD-ROM ve ses kartı Çoklu-Ortamın en önemli iki bileşenidir. Şu anda DOS altında çalışan çok sayıda Çoklu-Ortam ürünü olmakla birlikte, sağladığı olanaklardan dolayı Windows altında çalışan uygulamaların sayısı giderek artmaktadır.

5.3 API (Uygulama Programlama Arayüzü)

Windows uygulamalarının kullanımı kolaydır fakat uygulamaları geliştiren yazılımcılar, karmaşık API ile çalışmak durumundadır. API, Windows uygulamaları yazmayı sağlayan fonksiyonlar bütünüdür. Windows API, 600'den fazla fonksiyon içerir. Bunlardan çok azını içeren bir uygulama yazılsa bile, uygulamanın hangilerini ne şekilde gerektirdiğini bulmak zordur. Windows programlama, geleneksel programlama yapısından farklı yaklaşımlar da içerir. Fakat Windows Programlamanın üstünlükleri çok daha fazladır (Barkakati, 1993) :

- Windows uygulamaları makina bağımsızdır. Uygun olan her ekranda çalışır, her yazıcıdan çıktı alınabilir. Fakat makina bağımsız olması bir kötülüğü de beraberinde getirir. DOS tabanlı grafik programları, örnek olarak hareketli görüntüler içeren oyunlar, video bağdaştırıcısının özel bir moduna göre yazılır (256 renkli 320X200 çözünürlük 13h VGA). Programcı, bağdaştırıcının tüm özelliklerinden yararlanır. Örnek olarak 8 bitlik 256 renk içeren 320X200'lük bir resim 64K'ya sığar. Dolayısı ile, Windows altındaki bir animasyon daha yavaş olacaktır.

- Windows'ta önceden tanımlanmış kullanıcı arayüzü bileşenleri (menüler, diyalog kutuları vs.) programlamayı kolaylaştırır. Bunlar ile ilgili işlemleri tasarlayıp kodunu yazmak ve çalıştırmak gerekmez. Arayüz tasarımına ve oluşturmaya çok az vakit harcanır.

- Windows, grafik çizmeyi ve yazı yazmayı sağlayan GDI grafik arayüzünü içerir. GDI sayesinde programcı, donanım kurulumu ile uğraşmak zorunda kalmaz.

- Windows'ta bellek yönetimi ile çok fazla uğraşmak gerekmez.

5.4 OWL (Nesne Pencereleeri Kitaphğı)

C, 1990'lı yılların başında, Windows uygulamaları yazmak için tercih edilen bir programlama dili olmuştur. Daha sonra C++ popülaritesini artırmış olmakla birlikte, C++ programından windows fonksiyonlarını çağırmak, onları standart C kütüphanesinden çağırmak kadar kolay değildir. Çünkü derleyici, Windows fonksiyonlarını çağırırken, özel nesne kodu üretmek zorundadır ve Windows, fonksiyonlarına değer gönderirken farklı bir yöntem izlemektedir. OWL, Windows uygulamaları yazmayı kolaylaştıran C++ sınıfları kütüphanesidir. OWL sınıflarının birincil amacı, Windows uygulamaları için tam bir çatı oluşturmastır. Çatı denmesinin nedeni, sınıfların tüm bileşenlerinin, iskelet program parçaları halinde tüm Windows uygulamalarına oturtulabilmesidir.

5.5 IDE (Bütünleşik Geliştirme Ortamı)

Windows uygulamaları IDE ve Project Manager (Proje Yöneticisi) ile kurulur. Proje'ler üç çeşit dosya içerirler :

- C, C++ kaynak kodlarını içeren dosya(lar)
- Modül tanımlama (module definition) dosyası (Linker'a, uygulamanın kod ve veri segmentleri, uygulama program yığınının boyutuna ilişkin bilgiler verir)
- Kaynak (Resource) dosyası veya dosyaları (Pekçok Windows uygulaması, simge, imleç, bitmap, menü, diyalog kutuları gibi kaynakları içerir. Bu kaynaklar, Resource Workshop gibi kaynak editörleri ile yazılır ve derlenir). Kaynaklar, EXE veya DLL (Dynamic Link Library) içine yüklenmiş sadece okunabilir verilerdir. Kaynaklar, genelde uygulamanın görünen yüzüdür ve uygulama kodu içinde değil, bağımsız oluşturulur. Böylece, uygulama ile kaynak oluşturmaları aynı anda yürütülebilir. Ayrıca, örnek olarak menüde bir değişiklik yapınca, tüm uygulama kodunu değil sadece kaynaklardaki değişiklikleri derlemek yetecektir.

5.6 GDI (Grafik Araç Arayüzü)

GDI Windows'un grafik çıktı fonksiyonlarını içerir. GDI, Windows programını, ekran, yazıcı gibi çıktı araçlarından soyutlamak için tasarlanmıştır. Çıktı alınacağı zaman GDI fonksiyonları çağrılır ve bu fonksiyonlar, gerçek grafik çıktıları için gerekli araç sürücülerine erişirler. Fiziksel araçlar üzerinde çıktı oluşturmalarının yanısıra, iki yarı-arac üzerine de çıktı üretebilir (Bitmap ve Metafile). Bitmap, dikdörtgen şeklindeki nokta dizisi, GDI ise yüklenmiş fonksiyon işlemleri dizisidir. Bitmap'ler grafik ve animasyon işlemlerinde oldukça yararlıdır. GDI fonksiyonları 4 sınıfa ayrılabilir :

- Çizgi, dikdörtgen ve elips gibi grafik nesnelerini çizmeyi sağlayan vektör çizim fonksiyonları.
- Resimlerle ilgili işlemlerde kullanılan Bitmap fonksiyonları.
- Pencerele yazı yazdırmayı sağlayan yazı çıktı fonksiyonları.
- Görüntü bağdaştırıcısı tarafından desteklenen renklerden yararlanmayı sağlayan palet yönetim fonksiyonları.

5.6.1 DC (Araç Durumu)

Grafik çıktısı üretmek için gereken tüm bilginin tutulduğu veri yapısıdır. Örnek olarak DC, arka plan rengi, kalem, doldurma deseni, yazıtipi gibi grafik niteliklerini içerir ve grafik ve yazıların görünüşünü denetler. Tüm GDI fonksiyonları, DC'ye değişken olarak handle adı verilen özel bir tanımlayıcı isterler. DC kullanırken, ona erişmek için uygun bir GDI fonksiyonu çağrılır, çizim için DC kullanılır ve daha sonra DC serbest bırakılır. DC, kalem, fırça, bitmap, arka plan ya da yazı rengi gibi çizim nitelikleri ve font bilgilerini içerir ve bunlar için önceden belirlenmiş değerlerini atar. DC'nin birincil kullanım nedeni grafik çıktısı üretmektir. Grafik çıktısı üretiminde şu işlem sırası izlenir :

1. DC almır.
2. Grafik nitelikleri ayarlanır yani DC içerisindeki çizim nesneleri seçilir.
3. İlgili GDI çizim fonksiyonları çağrılarak çizim yapılır.
4. DC serbest bırakılır.

Çizim nesneleri şunlardır :

Kalem (Pen) : Dikdörtgen, elips ve çokgen çerçevesi ile çizgi görünüşünü denetler.

Fırça (Brush) : İçi doldurulmuş çizimler yapmak için doldurma desenini temin eder.

Yazıtipi (Font) : Yazı çıktılarının şekil ve boyutunu belirler.

Palet (Palette) : Renk dizisidir.

Bitmap : Resim çiziminde kullanılır.

DC, bir anda, her tip grafik nesnesinin bir kopyasını bulundurabilir. SelectObject fonksiyonu, DC içine grafik nesnesi seçmekte kullanılır. Silmek için ise DeleteObject fonksiyonu kullanılır.

5.6.2 GDI koordinat sistemi

GDI, fiziksel ve mantıksal olmak üzere, iki koordinat sistem yaklaşımını destekler. Aracın fiziksel koordinat sistemi sabittir. Ekrandaki pencere için fiziksel koordinat sisteminin başlangıç noktası, pencerenin sol üst köşesidir. Pozitif x ekseni sağa doğru, pozitif y ekseni aşağıya doğru artar. Mantıksal koordinat sistemi birden çok seçenek içinden seçilebilir. Windows, herhangi bir grafik çıktısı vermeden önce, bu koordinatları fiziksel koordinatlara eşler. Bütün GDI çizim fonksiyonları, mantıksal koordinatları argüman olarak kabul eder. Eşleme modu (mantıksal koordinat sisteminin fiziksel koordinat sistemine göre ölçeklendirilmesi), Windows'un desteklediği eşleme tiplerini tanımlar. Araç koordinat sistemi de 3'e ayrılır : a) Ekran koordinat sistemi, b) Tüm pencere koordinat sistemi, c) Kullanıcı alanı koordinat sistemi.

5.6.3 GDI fonksiyonları ile çizim

GDI, nokta, çizgi, dikdörtgen, çokgen, elips gibi çok sayıda çizim fonksiyonunu içerir.

5.6.3.1 Nokta çizimi

SetPixel komutu ile, ekranın x,y koordinatlarına, belirtilen renkteki bir nokta, "SetPixel(x,y,renk);" komutu ile konulabilir. Renk, 32 bitlik bir değer olup, 3 baytı, rengin kırmızı (r)(red), yeşil (g)(green) ve mavi (b)(blue) bileşenlerinin değerlerini gösterir. RGB makrosu ile, renk, şu şekilde tanımlanır : "RGB(r,g,b)" (r,g ve b 0 ile 255 arasında, rengin yoğunluğunu gösteren değer olmak üzere). Birden çok noktadan oluşan resimleri ekranda göstermek için, SetPixel yerine bitmap fonksiyonları kullanmak, çok daha hızlı sonuç verecektir.

Form 1 : TColor SetPixel(int x, int y, TColor color);

Form 2 : TColor SetPixel(const TPoint& p, TColor color);

5.6.3.2 Çizgi çizimi

Çizgi çizmek için, iki farklı GDI fonksiyonu kullanılır. P1(x1,y1) den P2(x2,y2) noktasına çizim için (P2 noktası hariç) şu komutlar kullanılır :

MoveTo(x1,y1);

LineTo(x2,y2);

MoveTo, seçilmiş olan kalem, ekranın (x1,y1) noktasına götürür ve LineTo, bu noktadan (x2,y2) noktasına, belirtilen kalemin nitelikleri ile çizimi sağlar.

Birden çok çizgi parçası çizmek için Polyline fonksiyonu kullanılır. Çizgi parçalarının köşe noktaları bir dizinde saklanarak çizim sağlanır.

TPoint points[3];

points[0].x = x0; points[0].y = y0;

points[1].x = x1; points[1].y = y1;

points[2].x = x2; points[2].y = y2;

dc.Polyline(points,3);

Kapalı bir çokgen için ise, ilk noktanın koordinatları, en son nokta olarak dizine eklenir :

```
TPoint points[4];
points[0].x = x0; points[0].y = y0;
points[1].x = x1; points[1].y = y1;
points[2].x = x2; points[2].y = y2;
points[3].x = x0; points[3].y = y0;
dc.Polyline(points,3);
```

Form 1 : **BOOL MoveTo(int x, int y);**

Form 2 : **BOOL MoveTo(const TPoint& point);**

Form 3 : **BOOL MoveTo(const TPoint& point, TPoint& oldPoint);**

Form 1 : **BOOL LineTo(int x, int y);**

Form 2 : **BOOL LineTo(const TPoint& point);**

BOOL Polyline(const TPoint* points, int count);

5.6.3.3 Kapalı şekillerin çizimi

GDI, kapalı şekillerin çerçevesini etkin kalem biçiminde çizer ve içlerini etkin fırça biçimi ile doldurur.

Dikdörtgen : **Rectangle(x1,y1,x2,y2)** sol üst köşesi (x1,y1), sağ alt köşesi (x2,y2) olan bir dikdörtgen çizer.

Form 1 : **BOOL Rectangle(int x1, int y1, int x2, int y2);**

Form 2 : **BOOL Rectangle(const TPoint& p1, const TPoint& p2);**

Form 3 : **BOOL Rectangle(const TPoint& point, const TSize& s);**

Form 4 : **BOOL Rectangle(const TRect& rect);**

Elips : `Ellipse(x1,y1,x2,y2)` Karşı köşeleri $(x1,y1)$ ve $(x2,y2)$ noktaları ile sınırlandırılmış dikdörtgen içine sığan elipsi çizer.

Form 1 : `BOOL Ellipse(int x1, int y1, int x2, int y2);`

Form 2 : `BOOL Ellipse(const TPoint& p1, const TPoint& p2);`

Form 3 : `BOOL Ellipse(const TPoint& point, const TSize& size);`

Form 4 : `BOOL Ellipse(const TRect& rect);`

Çokgen : `Polygon` komutu ile çizilir. Dizinde tanımlanan ilk nokta ile son noktayı, kapalı bir şekil oluşturmak üzere otomatik olarak birleştirir.

`BOOL Polygon(const TPoint* points, int count);`

5.6.4 Bitmap

Resimdeki her nokta, bitmap'te bir veya daha fazla bite karşılık gelir. Monokrom resimlerde, bitmap'teki her bir bit (siyah ve beyaz ya da 0 ve 1 olma durumuna göre) bir noktaya karşılık gelir. Renkli resimlerde ise, bir nokta için, o noktanın rengini belirtmek üzere birden fazla bit bilgisi kullanmak gerekir. Bir nokta için 1 (monokrom), 4 (16 renk), 8 (256 renk), 24 (yaklaşık 16.7 milyon renk) bit bilgi tutulabilir.

DDB (Device Dependent Bitmap = Araç bağımlı bitmap) : Görüntü aracı için birtakım kesin tanımlamalar yapmaktadır.

DIB (Device Independent Bitmap = Araç bağımsız bitmap) : Windows 3.0 ile gelmiş olup DDB'den daha esnek bir yapısı vardır. Bitmap bilgisini araç bağımsız bir şekilde, eski DDB bilgisine renk paleti ekleyerek yükler.

Bitmap, resim yapılabilecek bellek sahası olarak düşünülebilir. `BitBlt` ve `StretchBlt` fonksiyonları, bitmap'i görüntü aracına kolaylıkla kopyalayabilir. Bitmap'ler, hareketli görüntüler içeren uygulamalarda da oldukça yararlıdır. Örnek olarak küçük bir resmi

ekranda dolařtırmak iin resim bitmap řeklinde kaydedilip, gerektiğinde BitBlt fonksiyonu ile ekrana kopyalanır.

Bitmap kullanımının iki byk ktlğ vardır. Birincisi, boyutuna baėlı olarak bitmap'ler, kestirilemeyen bir biimde belleėin byk bir kısmını kullanır (Oliver et al., 1993). Gsterilen her nokta iin bellekte belirli bir yer harcanır. Aynı bitmap'i renkli ekranda gstermek, monokrom ekrandakinden daha fazla yer kaplar. Bitmap'lerin diėer bir ktlğ sadece resmi iermeleridir. rnek olarak, otomobilin bitmap sunumunda, tekerlek, kapı gibi deėiřik paralarına eriřilemez.



6. ANİMASYON TEKNİKLERİ

Eski Disney filmleri, herbir görüntünün özel olarak çizildiği geleneksel animasyon yaklaşımı ile hazırlanmışlardır. Bu tür animasyona "frame animation" veya "cel animation" da denilmektedir. Bu konu içerisinde, geleneksel olmayan ve bilgisayarlarda kullanılan animasyon teknikleri kısaca tanıtılacaktır (Barkakati, 1993).

6.1 Sprite (karakter) Animasyonu

Sprite'lar, etkileşimli video oyunlarında, oyunun parçası olan karakterlerdir. Oyuncu, oyun kolu, fare ya da klavye gibi bir girdi aracı ile bu karakterleri zemin üzerinde hareket ettirebilir. Video oyun makinaları, bu karakterlerin hareketlerini sağlayan donanımı içerir. IBM uyumlu PC'lerde ise görüntü donanımı Sprite'ları desteklemediği için, kullanılan teknikler yazılıma dayanmak durumundadır (Barkakati, 1993).

6.2 Sil ve Yeniden Çizdir Tekniği

Bir nesneyi hareket ettirmenin en kolay yolu, eski konumundaki halini silip yeni konumunda, yeni hali ile tekrar çizdirmektir. Hızlı bir şekilde çizilip yeniden yazılırsa, nesne ekran üzerinde hareket ediyor gibi görülür. Bu yöntemin en büyük kötülüğü, silinip tekrar çizdirilen resim parçalarının ekranda bulunduğu yerde bir titreme olmasıdır. Bunu önlemenin bir yolu, birden çok görüntü sayfasının desteklendiği durumlarda, görüntü sayfalarını arka arkaya göstermektir. Çoklu görüntü sayfalarında, etkin sayfa gösterilirken, o anda görünmeyen sayfalar çizilerek hazırlanır. Sonra, etkin ve görünmeyen görüntü sayfaları yer değiştirilerek, hazırlanmış olan sayfa görüntülenir. Animasyona devam etmek için bu işlem döngüsü tekrarlanır. Silicon Graphics gibi yüksek teknoloji ürün bilgisayarlarda, bellek takası (buffer swapping) adı verilen bu tekniği destekleyecek donanım bulunmaktadır (Barkakati, 1993).

Fakat, pekçok PC görüntü bağdaştırıcısı, yüksek çözünürlükteki görüntü modlarında, birden çok görüntü sayfasını desteklemez. Daha da önemlisi, Windows, birden çok görüntü

sayfasını desteklemez. Bu yüzden, Windows'ta titreşimsiz animasyonlar oluşturmak için başka bir yaklaşıma gereksinim duyulur (Barkakati, 1993).

6.3 Bitmap'i Ekran Dışında Hazırlama Tekniği

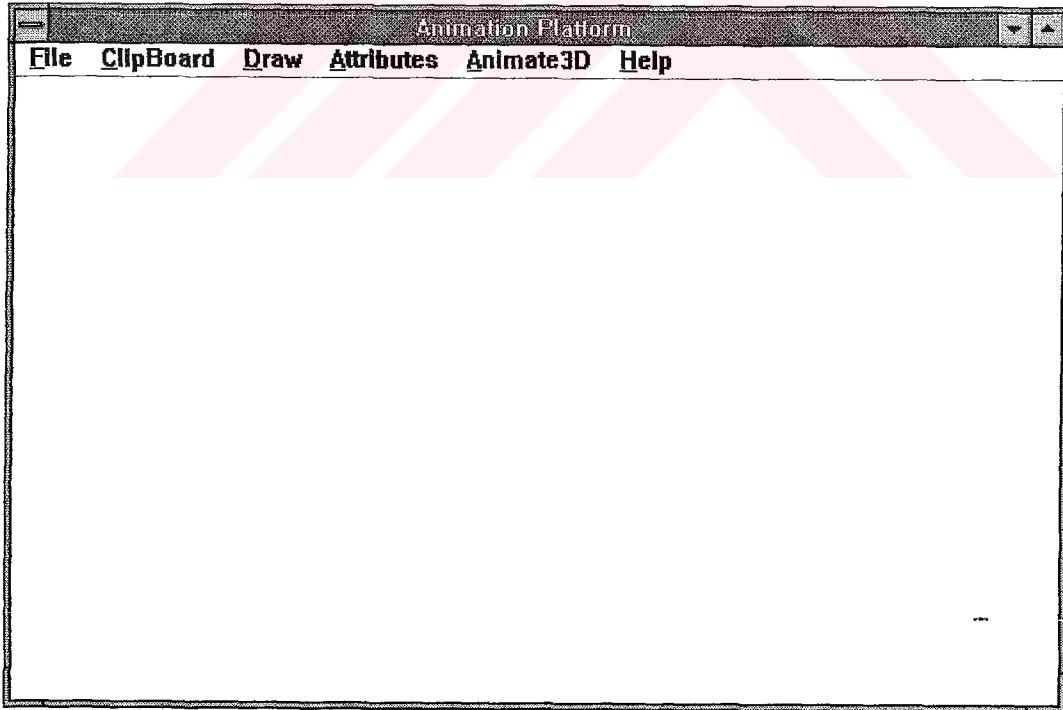
Animasyonda ekran titreşimi, tüm çizim işlemlerinin görünür olmasından kaynaklanmaktadır. Resmin bir parçası bir süre için ortadan kaybolur ve sonra yeniden ortaya çıkar. İki görüntü sayfası kullanıldığında, bu işlemler görünmeyen sayfada yapıldığı için titreşim ortadan kalkar. Hazırlanmış sayfa, etkin olan sayfa ile yer değiştirerek aniden görüntülenir. Aynı mantık ile, ekran dışında hazırlanıp aniden gösterilen resimler ile de titreşim önlenir. Windows, ekran dışı bitmap çizimini destekler ki bu, görüntülenecek sayfayı oluşturmak için ideal bir ortamdır. Ekran dışında hazırlanmış olan resim, BitBlt fonksiyonu ile ekrana kolaylıkla aktarılabilir (Barkakati, 1993).

7. ÇİZİM ve ANİMASYON PROGRAMI

Kişisel bilgisayarlarda, Windows ortamında BorlandC++ 4.0 kullanılarak, iki ve üç boyutlu basit çizim ve animasyonlar yapılmasını sağlayan bir arayüz geliştirilmiştir. Program için, C++ programlama dilinde yaklaşık olarak 2500 satır kod yazılmıştır. Arayüzün oluşturulması sırasında, C++, windows programlama, grafik ve animasyon programlama kitaplarından yararlanılmıştır. Program, Windows ortamında çalışabilen bir hale getirilmiştir. Programın tüm parçaları, sabit disk ya da ağ ortamına kopyalandıktan sonra, "drawanim.exe" çalıştırılabilir veya yeni bir program ögesi olarak grup pencerelerinden birisine eklenebilir. Program, fare yardımı ile hareket ettirilen ekran imleci,



bu ekleme işleminden sonra oluşan simgesi üzerine getirilerek çalıştırılır. Program çalıştırıldığında, Animation Platform başlığı altında, Şekil 7.1'de görülen pencere açılır.



Şekil 7.1 : Animation Platform uygulama programının penceresi.

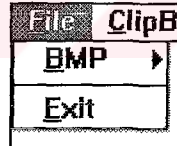
7.1 Menü Seçenekleri

Windows ortamında komutlar, menülerde listelenir. Uygulama penceresinin üst tarafındaki menü çubuğunda, o uygulamaya ilişkin seçenekler yer alır. Her seçenek altında, onun ile ilgili alt seçenekler bulunabilir. Bu sayede, istenilen bir komutun nerede olduğu kolaylıkla bulunabilir. Windows ortamında, herhangi bir uygulama programının menü çubuğu üzerindeki seçeneklere, değişik yollar ile ulaşılabilir. İlk yöntem, fareyi hareket ettirerek, ekran imlecinin, menü üzerindeki ilgili seçeneğin kapladığı alan üzerinde herhangi bir yere getirilerek, farenin sol tuşuna basılmasıdır. İkinci yöntem ise, ilgili seçenekteki kelimenin üzerindeki altı çizili olan harfe, klavyedeki "Alt" tuşuna basılı iken basılmasıdır.

7.2 File (Dosya) Seçeneği

File seçeneğinin altında 2 alt seçenek vardır: BMP ve Exit (Şekil 7.2).

BMP (bitmap)'ler ile ilgili işlemleri içeren alt seçenekleri vardır. Exit seçildiğinde ise, Animation Platform uygulama penceresi kapatılarak programdan çıkılır.



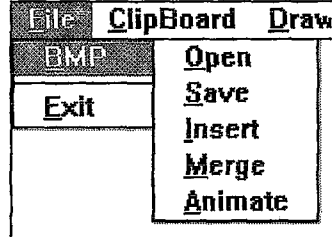
Şekil 7.2 : File (Dosya) seçeneğinin alt seçenekleri

7.2.1 BMP (bitmap) alt seçeneği

Bitmap formatındaki dosyaların uzantıları "BMP" olduğu için, ilgili komutlar BMP alt seçeneği altında toplanmışlardır.

BMP alt seçeneği seçildiği zaman, yanında basamaklı bir menü açılır. Bu menüde, 5 seçenek vardır : Open, Save, Insert, Merge, Animate (Şekil 7.3).

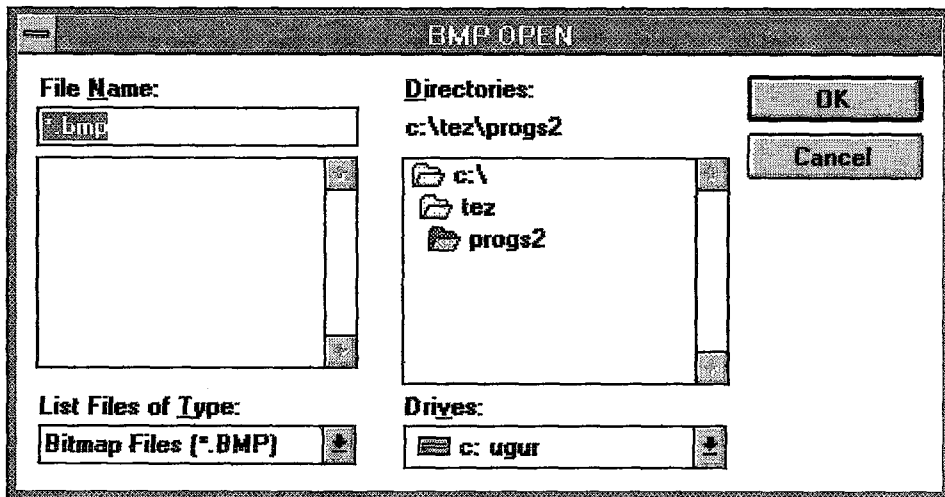
Bu komutların herhangi biri seçildiğinde, ekrana ilk olarak ilgili bir diyalog kutusu gelir. Diyalog kutuları, liste kutuları, metin kutuları ve komut düğmeleri gibi parçalardan oluşur. Fare ve klavye kullanılarak, bu sahalarda seçimler yapılabilir ve bilgi girilebilir. Yapılan işlemi onaylamak için "OK" (Tamam) komut düğmesine basılır. İşlemden vazgeçildi ise, "Cancel" (İptal) komut düğmesine basılır.



Şekil 7.3 : File (Dosya) - BMP alt seçeneğinin alt seçenekleri.

7.2.1.1 Open (Aç) alt seçeneği

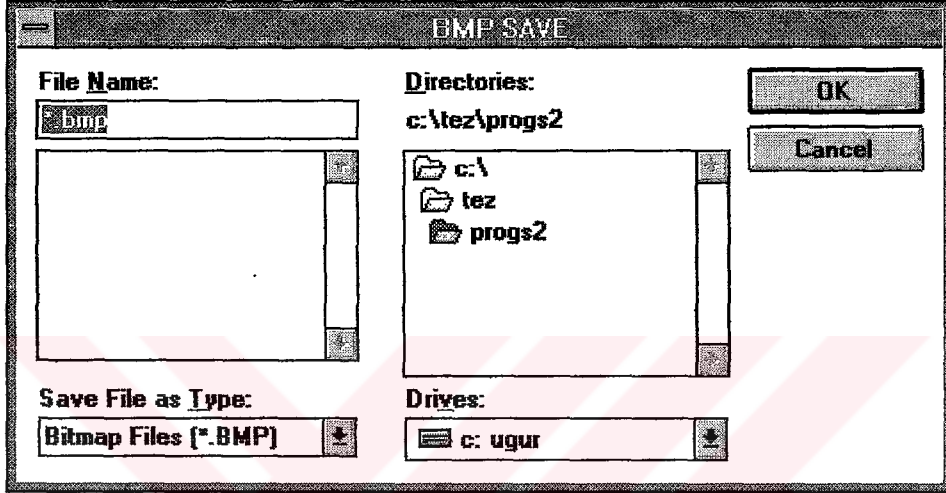
İkincil bellek ortamlarından alınan bitmap resim formatındaki dosyaların ekranda görüntülenmesini sağlar. Komut çalıştırıldığında çıkan BMP OPEN Diyalog Kutusu'nda, ekranda görüntülenecek bitmap dosyasının yeri (sürücü, dizin) ve adı verilerek onaylanır (Şekil 7.4).



Şekil 7.4 : File - BMP - Open alt seçeneğinin çalıştırılması sonucu ekrana gelen diyalog kutusu.

7.2.1.2 Save (Kaydet) alt seçeneği

Ekrandaki resmin, bitmap resim formatında, ikincil bellek ortamlarına kaydedilmesini sağlar. Komut çalıştırıldığında çıkan BMP SAVE Diyalog Kutusu'nda, ekrandaki resmin kaydedileceği bitmap dosyasının yeri (sürücü, dizin) ve adı verilerek onaylanır (Şekil 7.5).



Şekil 7.5 : File - BMP - Save alt seçeneğinin çalıştırılması sonucu ekrana gelen diyalog kutusu.

7.2.1.3 Insert (Ekle) alt seçeneği

İkincil bellek ortamlarından alınan bitmap dosyalarının, ekrandaki resmin herhangi bir yerine eklenmesini sağlar. Komut çalıştırıldığında çıkan BMP OPEN Diyalog Kutusu'nda, ekranda görüntülenecek bitmap dosyasının yeri (sürücü, dizin) ve adı verilerek onaylanır. Farenin sol tuşuna basılı tutularak fare hareket ettirilir. Bu sırada eklenecek resim, arka plandaki resmi bozmadan hareket etmektedir ve ekran imleci resmin sol üst köşesindedir. Resim uygun konuma gelince, farenin sol tuşu serbest bırakılır ve resim eklenir. Farenin sol tuşu basılı iken, resmin eklenmesinden vazgeçilirse, farenin sağ tuşuna basılır.

Resimler, daha küçük resim parçalarından oluşabilir. Bu seçenek sayesinde, resim parçaları hazırlanır ve bunlar, benzeri resimleri oluşturmakta kullanılabilir. Birden çok resim parçasını karmaşık bir resmi oluşturmak için kullanmayı sağlaması yanında, aynı resmin kopyalarını da resme eklemek olasıdır.

7.2.1.4 Merge (Birleřtir) alt seeneęi

İkincil bellek ortamlarından alınan bitmap dosyalarının belirli bir bölümünün (siyah renk olmayan kısımlar), ekrandaki resmin herhangi bir yerine eklenmesini sağlar. Insert alt seeneęindeki gibi alıřtırılır.

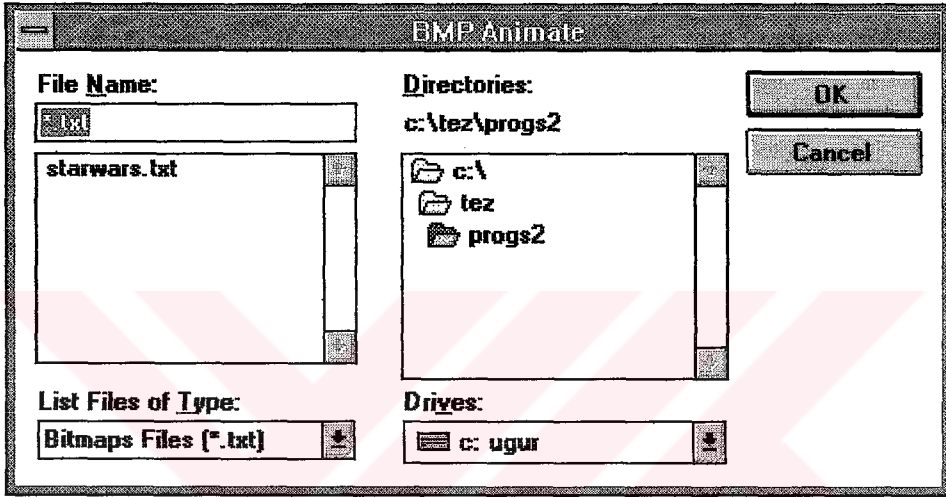
Animasyon işlemlerinde önemli bir yer tutar. Insert seeneęinin geliştirilmiş bir şeklidir. Resimlerdeki her karakter, dikdörtgen şeklinde deęildir. Karakterler, dikdörtgen içerisinde belirli bir yeri kaplarlar. Bitmap'in içindeki dięer bölgeleri siyah hale getirip, ekrandaki resmin üzerine, sadece eklenecek bitmap üzerindeki karakterlere ilişkin bilgileri eklemek olası hale gelir.

Böylece, örnek olarak insan yürüyüşünün safhalarının resimleri siyah zemin üzerinde oluşturulup, ayrı ayrı bitmap dosyaları halinde kaydedilir. Sonra herbiri, istenilen bir zemin üzerine, ilgili yerlere, Merge seeneęi ile eklenerek tekrar kaydedilirse, herhangi bir ortamda insan yürüyüşünü gösteren animasyon parasının resimleri oluşturulmuş olur. Siyah zemin üzerinde çizilmiş olan resimler tekrar kullanılabilir. Örnek olarak, aynı resimler başka bir zemin üzerine eklenerek, başka bir ortamdaki yürüme animasyonu oluşturulabilir.

Bir zemin üzerine birden fazla karakter ekleyerek daha karmaşık animasyonlar elde edilebilir. Bir insanın yürüyüşünün aşamalarındaki deęişik durumlara göre, baş, gövde, kol ve bacakların durumunu gösteren daha küçük resimler oluşturulup ayrı ayrı kaydedilirse, daha büyük resimleri oluşturmak için, tekrar tekrar aynı paraların çizilmesi gerekmez. Burada anlatılan yöntemler, geleneksel çizgi film hazırlamada kullanılan tekniklere benzemektedir. Fakat, bilgisayarda resim oluşturmak çok daha kolaydır. Çizimler ve kapalı alanların doldurulması, ek bir araç gerektirmeden, kolaylıkla ve bilgisayar ortamının getirdięi tüm üstünlüklerinden yararlanarak yapılmaktadır. Ayrıca, üst üste konulan resimlerde, önceden konulan resimlerin renkleri, sonradan aynı yere konulan resimlerin renklerini bozmamaktadır. Böylece, saydamlarda yaşanan problemler, bilgisayar ortamında otomatik olarak çözülür.

7.2.1.5 Animate (Canlandır) alt seçeneği

İkincil bellek ortamlarında bulunan bitmap dosyalarını arka arkaya ekrana getirerek, iki boyutlu animasyonlar oluşturulmasını sağlar. Komut çalıştırıldığında çıkan BMP Animate Diyalog Kutusu'nda, açılacak animasyon dosyasının (uzantısı .txt) yeri (sürücü, dizin) ve adı verilerek onaylanır (Şekil 7.6).



Şekil 7.6 : File - BMP - Animate alt seçeneğinin çalıştırılması sonucu ekrana gelen diyalog kutusu.

Bu animasyon dosyasının ilk satırı, arka arkaya gösterilecek olan bitmap dosyaları arasındaki bekleme miktarını belirtir. Sonraki satırlarda ise her satır bir tane içermek üzere, sıra ile gösterilecek olan bitmap dosyalarının yer ve adları yer almaktadır. Bu dosya, herhangi bir ASCII yazı editörü ile kolaylıkla hazırlanabilir. İyi bir animasyon için, saniyede 24 civarında resmin arka arkaya gösterilmesi gerekmektedir. Uygulamanın çalıştırıldığı bilgisayar ve ekranın hızı, animasyondaki resimlerin boyutu, renk sayısı, animasyonun hızını etkilemektedir. Bazı durumlarda, büyük boyutlu ve gerçekçi resimlerin animasyonunda yeterli hız sağlanamaz. Bazı durumlarda da, resimler çok hızlı geçer. Bu durumda, 0'dan büyük bir tamsayı şeklinde ifade edilebilen bekleme miktarı ayarlanır. Uygulamanın çalıştırıldığı bilgisayarın durumuna göre bekleme miktarı değiştirilerek, birkaç denemede istenen değer verilebilir. Bekleme miktarı, program içerisinde FOR döngüsünün dönme sayısı olarak ele alınmıştır (Şekil 7.7).

starwars.txt dosyası

1000000

c:\install\avis\t0190001.bmp

c:\install\avis\t0190002.bmp

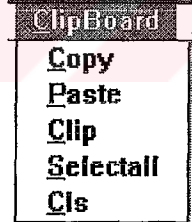
c:\install\avis\t0190003.bmp

c:\install\avis\t0190004.bmp

Şekil 7.7 : BMP Animasyonu için kullanılan bir dosya örneği. İlk satırda, arka arkaya gösterilecek bitmap dosyaları arasındaki bekleme miktarı, sonraki satırlarda ise, bu bitmap dosyalarının isimleri yer almakta. Bu örnekte, 4 tane bitmap dosyası arka arkaya görüntülenecektir.

7.3 ClipBoard (Pano) Seçeneği

Pano seçeneğinin altında 5 alt seçenek vardır : Copy, Paste, Clip, Selectall, CIs (Şekil 7.8). Bunlar, Windows'un panosu ile ilgili resim alışverişini sağlayan ve resimlerle ilgili yararlı işlemleri yapan komutlardır.



Şekil 7.8 : ClipBoard (Pano) seçeneğinin alt seçenekleri

7.3.1 Copy (Kopyala) alt seçeneği

Ekranda olan resmi Windows'un panosuna kopyalar. Böylece resmi, pano aracılığı ile başka uygulamalara geçirmek olası hale gelir.

7.3.2 Paste (Yapıştır) alt seçeneği

Windows'un panosunda olan resmi ekrana kopyalar. Böylece, bitmap formatı dışındaki resimleri de -bitmap formatına çevirmeden- başka bir program aracılığı ile görüntüleyerek ya da kopyalayarak programa geçirmek olası hale gelir. Windows'taki tüm ekranın (PrintScreen tuşu), ya da etkin pencerenin (Alt+PrintScreen) panoya alınması sayesinde de, Windows'tan görüntü yakalanıp uygulamaya geçirilebilir.

7.3.3 Clip (Kesme) alt seçeneği

Ekranda olan resmin, dikdörtgen şeklinde belirli bir bölümünü keserek, diğer kısımlarını siler.

Komut çalıştırıldığında, ekran imleci, resmin kullanımda kalacak dikdörtgen şeklindeki parçasının 4 köşesinden birisine getirilerek farenin sol tuşuna basılır. Bu durumda iken ekran imleci, fare yardımı ile seçilecek dikdörtgen bölgenin karşıt köşesine hareket ettirilir ve farenin sol tuşu serbest bırakılır. Fare hareketi sırasında, her değişimde, seçili bölgenin etrafındaki çerçeve boyutları da değişecektir. Böylece, istenen kısmı belirlemek kolaylaşacaktır. Bu işlem sonucunda, belirlenmiş olan kısım (farenin sol tuşu bırakıldığı anda, yani çerçevenin içinde kalan kısım) pencerenin sol üst köşesine kaydırılarak, resmin diğer kısımları silinecektir. Farenin sol tuşuna basılı durumda iken, sağ tuşuna da basarak işlemi iptal etmek de olasıdır.

Kesme işlemi, resim parçalarının elde edilmesinde oldukça yararlıdır. Örnek olarak, tez metninin bu bölümünde yer alan resimlerin tümünün düzenlenmesinde (ilgili ekran parçalarının kesilmesinde) Animation Platform programının kendisi kullanılmıştır.

7.3.4 Selectall (Tümünü Seç) alt seçeneği

Tüm pencere çizim alanının, çizim için kullanılmasını sağlar.

Komut çalıştırıldığında, çizim alanı pencere boyutlarına gelmektedir. Değişik boyutlarda resimler oluşturmak için kullanılır. Ayrıca, bir bitmap dosyası ekrana getirildikten sonra, pencere boyutları onun kapladığı alandan büyük de olsa, çizim bölgesi bitmap'in boyutları kadar olacaktır. Clip, yani kesme işleminden sonra da, çizim alanı pencere boyutlarından küçük olacaktır. Resmi genişletmek, resmin çevresine eklemeler yapmak için, tüm pencere alanının kullanıma geçmesini sağlayan bu komut kullanılır. Ayrıca, Windows ortamında pencere boyutları da kolayca değiştirilebildiğinden, istenen boyutta resimler hazırlamak kolaylaşacaktır. Fakat, pencere boyutu değiştiğinde, eğer yeni boyutun belirlediği tüm alanın kullanılması isteniyorsa, tekrar bu komut seçilmelidir.

7.3.5 CIs (Ekranı Temizle) alt seçeneği

Ekranı temizlemek için kullanılır. Birtakım çizimler yapıldıktan sonra, yapılan çizimler hatalı olduğunda, yeni bir çizime başlamak için, pratik bir şekilde ekranın temizlenmesini, silinmesini yani beyaz renkli bir zemine dönüşmesini sağlar.

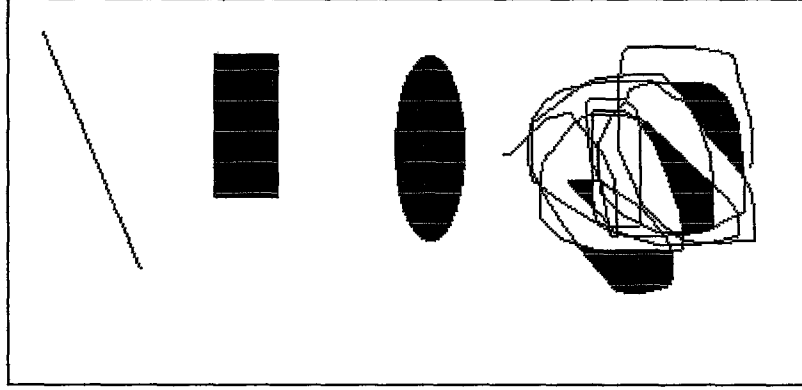
7.4 Draw (Çizim) Seçeneği

Draw seçeneğinin altında 5 alt seçenek vardır : Line, Rectangle, Ellipse, Point, Floodfill (Şekil 7.9). Pencere üzerinde, iki boyutlu temel çizim ve boyama işlemlerinin yapılmasını sağlarlar.

Draw	Attribut
<u>L</u> ine	
<u>R</u> ectangle	
<u>E</u> llipse	
<u>P</u> oint	
<u>F</u> loodfill	

Şekil 7.9 : Draw (Çizim) seçeneğinin alt seçenekleri

Şekil 7.10'da, bu alt seçenekler yardımı ile oluşturulmuş bir ekran parçası görülmektedir.



Şekil 7.10 : Draw seçeneğinin alt seçenekleri yardımı ile oluşturulan iki boyutlu temel çizimler. Soldan sağa, çizgi, içi dolu dikdörtgen, elips ve noktalardan oluşturulmuş çizimin belli bölgelerinin içinin doldurulması ile elde edilen şekil.

Windows uygulama programlarının önemli bir kısmını, ileti döngüsü oluşturur. Amacı, Windows tarafından gönderilen iletileri almak ve işlemektir. Herhangi bir Windows uygulaması başlatıldığında gönderilen iletiler, okunup işleninceye kadar ileti kuyruğunda tutulur. Programda, farenin sol tuşu basılı iken ve basılı değilken, farenin sağ tuşuna basılı iken ve fare hareketinde yapılması gereken işlemler, fonksiyon şeklinde hazırlanmıştır. Etkin olan araç (çizgi, dikdörtgen, elips, nokta, içini doldurma, kesme, ...), ilgili araca ilişkin alt seçeneğin çalıştırılması ile belirlenmektedir. Fare ile ilgili fonksiyonlardan birine girildikten sonra yapılacak işlemler, etkin araca göre ayrı ayrı hazırlanmıştır. Örnek olarak, etkin olan araç çizgi ise çizgi, dikdörtgen ise dikdörtgen çizilmesi gerektiğinden, program bu kısımlarda ayrı dallara sapacak şekilde yazılmıştır. Yapılan çizim işleminden sonra, pencere içindeki resim belleğe kaydedilmekte ve böylece pencere boyutu ve yerinin değişmesi ya da uygulamalar arası geçişte görüntünün kaybolması gibi durumlarda, pencerenin tekrar görüntülenmesi gerektiğinde kullanılmaktadır. Bu programın oluşturulmasında, özellikle BC4\EXAMPLES\OWL\OWLAPPS dizini altındaki örneklerden yararlanılmıştır. Çizgi çizmeyi sağlayan DRAW adlı program geliştirilerek, son haline kadar çeşitli aşamalardan geçilmiştir. Menü, simge, bitmap gibi her çeşit Windows kaynaklarının oluşturulmasını sağlayan RESOURCE WORKSHOP adlı program kullanılarak, menüye değişik özellikleri içeren seçenek ve alt seçenekler eklenmiş, seçenek çıkarma ve düzenlemeler yapılmıştır. Pencereye eklenen menünün cevap

tablosunda, oluşturulan menü elemanının numarasına karşılık bir üye fonksiyon atanmış ve sonra da bu fonksiyonun içeriği programa eklenmiştir. Onay konulabilen seçeneklerin oluşturulmasında bu işlemlere birkaç ek daha yapılmaktadır. Diğer örnek programlardan da, pano ile ilgili işlemlerin yapıldığı program parçaları, dosya açma kapama ya da renk belirleme gibi amaçlarla kullanılan hazır diyalog kutularının veya yenilerinin oluşturulduğu ve işlendiği program parçaları alınarak programa eklenmiştir. Konuların anlaşılması için Windows programlama ile ilgili kitaplardan ve BorlandC++ 4.0'ın Help desteğinden yararlanılmıştır.

7.4.1 Line (Çizgi) alt seçeneği

Pencere içerisine, iki boyutlu çizgi çizilmesini sağlar. Çizgi rengi, bu komut çalıştırılmadan önce, Attributes - Color - Pen alt seçeneği kullanılarak belirlenir.

Komut çalıştırıldıktan sonra, sırası ile şu işlemler yapılır : Fare yardımı ile ekran imleci, çizim alanı içerisinde, çizilecek çizginin uç köşelerinden birisine getirilerek farenin sol tuşuna basılır. Tuş bırakılmadan, fare hareketi ile ekran imleci, çizginin diğer ucuna getirilir. Farenin hareketi sırasında çizginin aldığı şekil görülmekte ve her durumda, orjinal resmin ilgili parçası saklanarak, orjinal resmin bozulması da önlenmektedir. Çizgi istenen duruma gelince, farenin hareketi durdurulur ve farenin sol tuşu serbest bırakılır. Ekranda oluşan çizginin iptal edilmesi için, farenin sol tuşu bırakılmadan sağ tuşuna basılabilir. Farenin her hareketi sırasında çizginin arka plan üzerindeki şekliyle görülmesi ve arka planın işlem boyunca korunması olay merkezli programlamada şu şekilde sağlanmıştır :

- (Farenin sol tuşuna basıldığında yapılacak işlemler)

Sol tuşuna basılma durumunu gösteren değişken 0 ise

- a) 1 yapılarak sonradan gelen farenin sol tuşuna basılı iletilerinin buraya girmesi engellenir.
- b) Kalem ve Fırça tanımlanır.
- c) Ekran imlecinin koordinatları alınır. Çizginin, bu çizim boyunca değişmeyecek olan bir köşesinin koordinatlarıdır.
- d) Fare, sol tuşuna basıldıktan sonra hareket etmemiştir. Bunun ile ilgili değişken güncellenir.

- (Fare hareket ettirildiğinde yapılacak işlemler)

Sol tuşuna basılma durumunu gösteren değişken 1 ise (basılı ise)

- a) Farenin sol tuşuna basıldıktan sonraki ilk hareket ise, bunun ile ilgili değişken güncellenir, değilse bir önceki hareket sırasında alınan arka plan görüntüsü, pencerenin ilgili yerine konulur (ilk hareket ise, zaten orjinal resim üzerine çizim yapılmamıştır; önceden hareket yapıldı ise de, son çizimin çizilmiş olduğu bölge çizimden önce belleğe kaydedildiği için, resmin ilgili bölgesine tekrar getirilerek orjinal resim elde edilir).
- b) Farenin sol tuşuna basıldığında alınan imlecin konumu ile, şu anki imleç konumu arasındaki dikdörtgen bölge belleğe kaydedilir ve bu iki konum arasına, Windows GDI fonksiyonları yardımı ile çizgi çizilir.

- (Farenin sol tuşu serbest iken yapılacak işlemler)

Sol tuşuna basılma durumunu gösteren değişken 1 ise (basılı ise)

- a) Resim belleğe alınır. Böylece, ekranda oluşacak olan değişimlerde, bellekte olan bu resim ekrana tekrar kopyalanarak, kaybolması önlenir. Bu sırada, başka herhangi bir işlem yapılmadığı için, son çizgi ekranda kalmış ve çizim yapılmıştır. Farenin sol tuşuna basılma durumunu gösteren değişken 0 yapılır.

- (Farenin sağ tuşuna basıldığında yapılacak işlemler)

Sol tuşuna basılma durumunu gösteren değişken 1 ise (basılı ise)

Fare, sol tuşuna basıldıktan sonra hareket ettirildi ise

Bir önceki fare hareketi sırasında alınan arka plan görüntüsü, pencerenin ilgili yerine konularak, hiç çizim yapılmadığı durum yani çizimden önceki orjinal resim elde edilir. Farenin sol tuşuna basılma durumunu gösteren değişken 0 yapılır.

7.4.2 Rectangle (Dikdörtgen) alt seçeneği

Pencere içerisine, iki boyutlu dikdörtgen çizilmesini sağlar. Çerçeve rengi, bu komut çalıştırılmadan önce, Attributes - Color - Pen; iç doldurma rengi, Attributes - Color - Brush alt seçeneği kullanılarak belirlenir.

Çizimin yapılmasında kullanılan yöntem, Line komutundakine çok benzemektedir. Yalnız, çizginin uç noktaları yerine, dikdörtgenin karşıt uç köşeleri belirlenmekte ve farenin her hareketinde çizgi yerine kapalı bir dikdörtgen çizilmektedir.

7.4.3 Ellipse (Elips) alt seçeneği

Pencere içerisine, iki boyutlu elips çizilmesini sağlar. Çerçeve rengi, bu komut çalıştırılmadan önce, Attributes - Color - Pen; iç doldurma rengi, Attributes - Color - Brush alt seçeneği kullanılarak belirlenir.

Çizimin yapılmasında kullanılan yöntem, Line komutundakine çok benzemektedir. Yalnız, çizginin uç noktaları yerine, elipsi içine alan en küçük dikdörtgenin karşıt uç köşeleri belirlenmekte ve farenin her hareketinde çizgi yerine kapalı bir elips çizilmektedir.

7.4.4 Point (Nokta) alt seçeneği

Farenin hareketi sırasında -sol tuşuna basılı iken-, geçtiği yerlere nokta koyarak, düzgün olmayan, eğrisel şekiller çizilmesini sağlar. Çizim rengi, bu komut çalıştırılmadan önce, Attributes - Color - Pen alt seçeneği kullanılarak belirlenir.

Buradaki yöntem, Line komutundakinden farklıdır. Fare yardımı ile ekran imleci, çizim alanı içerisinde, çizilecek şeklin bir ucuna getirilerek farenin sol tuşuna basılır. Tuş bırakılmadan, fare yardımı ile ekran imleci hareket ettirilerek, bu yol üzerinde çizim yapılması sağlanır. Farenin sol tuşu bırakıldığı zaman, çizim de bitecektir. Çizime başlanmadan, tüm çizim alanı belleğe kaydedilmektedir. Farenin sol tuşu bırakılmadan sağ tuşuna basılırsa, bellekteki bu resim ekrana kopyalanarak, yapılan son çizim iptal edilir.

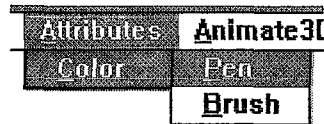
7.4.5 FloodFill (İçini Doldur) alt seçeneği

Kapalı bir bölgeyi belirli bir renk ile doldurur. Taslak olarak sınırları belirlenmiş resim parçalarının ya da kesişen bölgelerin renklendirilmesinde oldukça yararlıdır.

Komut çalıştırılmadan önce, kapalı alanın hangi renk ile doldurulacağını belirtmek için, Attributes - Color - Brush alt seçeneği kullanılır. Kapalı bölgenin sınırlarını belirten renk ise, Attributes - Color - Pen alt seçeneğinden alınır. Komut çalıştırılınca, fare hareketi ile ekran imleci kapalı bölgenin sınırlarını belirten renkte sınırı olan bölgenin içinde herhangi bir noktaya getirilerek sol tuşuna basılır. Bu işlemden sonra bu alan otomatik olarak doldurma rengine boyanır.

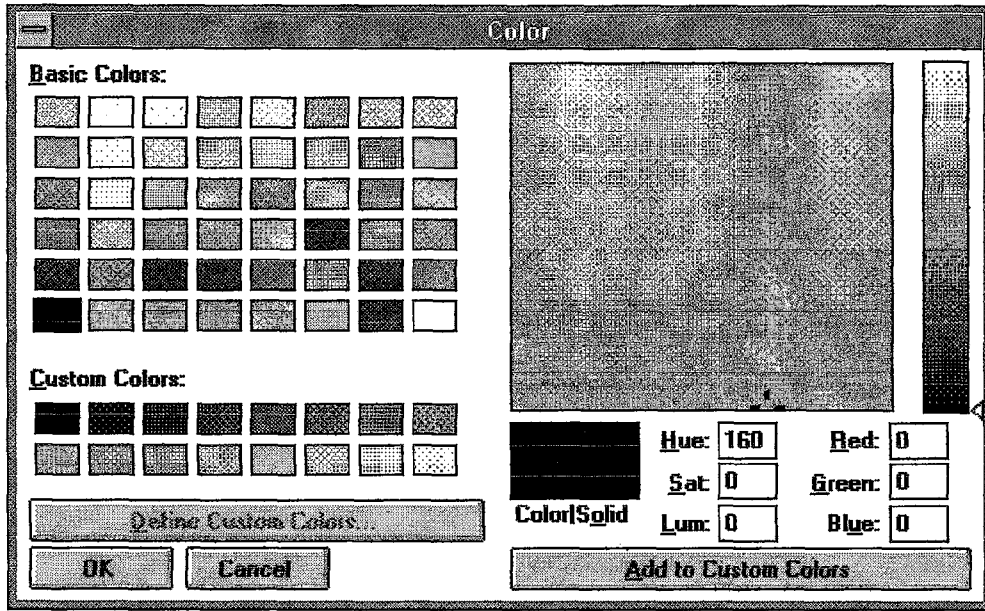
7.5 Attributes (Nitelikler) Seçeneği

Attributes seçeneğinin altında tek alt seçenek vardır : Color (Şekil 7.11). Color alt seçeneğinin de iki alt seçeneği vardır : Pen, Brush (Şekil 7.11). Bu alt seçenekler, Draw seçeneğindeki iki boyutlu temel çizimlerde kullanılan renkleri belirlemede kullanılırlar. Pen yardımı ile belirlenen renk, nokta ve çizgi rengi, dikdörtgen, elips ve kapalı alanın sınır rengi olarak kullanılır. Brush yardımı ile belirlenen renk ise, dikdörtgen, elips ve kapalı alanın iç doldurma rengidir. Kısaca Pen, Windows'taki kalem rengi, Brush ise Windows'taki fırça rengidir.



Şekil 7.11 : Attributes (Nitelikler) seçeneğinin alt seçenekleri.

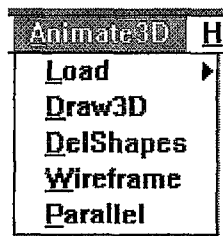
Pen ya da Brush seçildiğinde, çıkan diyalog kutusundan (Şekil 7.12) renk seçmek ve yeni renkler oluşturmak olasıdır.



Şekil 7.12 : Color Diyalog Kutusu

7.6 Animate3D (3D Animasyon) Seçeneği

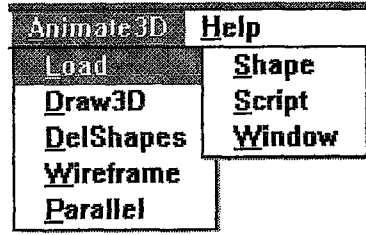
Animate3D seçeneğinin altında, 5 alt seçenek vardır: Load, Draw3D, DelShapes, Wireframe, Parallel (Şekil 7.13).



Şekil 7.13 : Animate3D seçeneğinin alt seçenekleri.

7.6.1 Load (Yükle) alt seçeneği

Bu alt seçeneğin altında üç alt seçenek vardır : Shape, Script, Window (Şekil 7.14).



Şekil 7.14 : Animatc3D - Load alt seçeneğinin alt seçenekleri.

Bu alt seçenekler, üç boyutlu nesnelerin ve bunlara ilişkin hareket bilgilerinin dosyalardan okunmasını sağlarlar.

7.6.1.1 Shape (Şekil) alt seçeneği

Bu komut çalıştırıldığında, ekrana çıkan diyalog kutusundan, üç boyutlu nesne tanımlarının yapıldığı dosyanın seçilmesi sağlanır. Bu dosya geçerli ise, dosyadaki bilgiler, çizim sırasında kullanılacak veri yapısına yüklenir. Kenar uzunluğu 100 birim olan küp nesnesinin bilgilerinin yer aldığı örnek "cube.shp" dosyası, şu şekildedir :

```

8 // Nesnenin toplam köşe sayısı (Bir sonraki satırdan itibaren okunacak satır sayısı)
0 0 0 // 0 numaralı köşenin (x,y,z) koordinatları
100 0 0 // 1 numaralı köşenin (x,y,z) koordinatları
0 100 0 // 2 numaralı köşenin (x,y,z) koordinatları
100 100 0 // 3 numaralı köşenin (x,y,z) koordinatları
0 0 100 // 4 numaralı köşenin (x,y,z) koordinatları
100 0 100 // 5 numaralı köşenin (x,y,z) koordinatları
0 100 100 // 6 numaralı köşenin (x,y,z) koordinatları
100 100 100 // 7 numaralı köşenin (x,y,z) koordinatları
12 // Nesnenin toplam kenar sayısı (Bir sonraki satırdan itibaren okunacak satır sayısı)
0 1 // 0 numaralı kenarın uç köşe numaraları
0 2 // 1 numaralı kenarın uç köşe numaraları
1 3 // 2 numaralı kenarın uç köşe numaraları
2 3 // 3 numaralı kenarın uç köşe numaraları
0 4 // 4 numaralı kenarın uç köşe numaraları

```

1 5 // 5 numaralı kenarın uç köşe numaraları
 2 6 // 6 numaralı kenarın uç köşe numaraları
 3 7 // 7 numaralı kenarın uç köşe numaraları
 4 5 // 8 numaralı kenarın uç köşe numaraları
 4 6 // 9 numaralı kenarın uç köşe numaraları
 5 7 // 10 numaralı kenarın uç köşe numaraları
 6 7 // 11 numaralı kenarın uç köşe numaraları
 6 // Nesnenin toplam yüzey sayısı (Bu sayının iki katı satır okunacaktır)
 0 2 3 1 // 0 numaralı yüzeyi oluşturan kenar numaraları (sıra ile)
 0 0 0 // 0 numaralı yüzeyin 0 ile 255 arasındaki r,g,b renk bileşenleri (siyah renk)
 2 5 10 7 // 1 numaralı yüzeyi oluşturan kenar numaraları (sıra ile)
 255 0 0 // 1 numaralı yüzeyin 0 ile 255 arasındaki r,g,b renk bileşenleri (kırmızı renk)
 0 4 8 5 // 2 numaralı yüzeyi oluşturan kenar numaraları (sıra ile)
 0 255 0 // 2 numaralı yüzeyin 0 ile 255 arasındaki r,g,b renk bileşenleri (yeşil renk)
 1 6 9 4 // 3 numaralı yüzeyi oluşturan kenar numaraları (sıra ile)
 0 0 255 // 3 numaralı yüzeyin 0 ile 255 arasındaki r,g,b renk bileşenleri (mavi renk)
 3 7 11 6 // 4 numaralı yüzeyi oluşturan kenar numaraları (sıra ile)
 255 255 0 // 4 numaralı yüzeyin 0 ile 255 arasındaki r,g,b renk bileşenleri (sarı renk)
 8 9 11 10 // 5 numaralı yüzeyi oluşturan kenar numaraları (sıra ile)
 255 0 255 // 5 numaralı yüzeyin 0 ile 255 arasındaki r,g,b renk bileşenleri (mor renk)
 1 1 1 // Nesnenin ölçeklendirme faktörleri
 0 0 0 // Nesnenin döndürme açıları (derece cinsinden)
 0 0 0 // Nesnenin yer değiştirme uzaklıkları

Örnek dosyaya gözönüne alınacak olursa, ilk satırda üç boyutlu nesnenin köşe sayısı verilmekte ve izleyen satırlarda 0. köşeden, (toplam köşe sayısı - 1) 'e kadar, köşelerin (x,y,z) koordinatları gelmektedir. C dilinde diziler, 0. elemandan başladığı için, tüm dizi kullanımlarındaki numaralandırma sistemlerinde karışıklık olmaması amacıyla ilk elemanlar 0 numaralı eleman olarak kabul edilmiştir. Örnek olarak 0. köşe burada ilk köşe olup, son köşenin numarası da toplam köşe sayısının bir eksiği olmaktadır. Her satırda, (x,y,z)'ye karşılık gelen birer tane gerçel sayı ve aralarında birer boşluk bulunmak zorundadır.

Sonra, nesnenin toplam kenar sayısı verilmekte ve izleyen satırlarda 0. kenardan, (toplam kenar sayısı - 1) 'e kadar kenarların uç noktalarının köşe numaraları gelmektedir.

Kenarlar doğru parçaları olduğu için, iki uç noktaları vardır. Bu nedenle, her satırda iki pozitif tamsayı ve aralarında birer boşluk bulunmak zorundadır.

Sonra, nesnenin toplam yüzey sayısı verilmekte ve izleyen satırlarda 0. yüzeyden (toplam yüzey sayısı - 1) 'e kadar, yüzeyleri oluşturan kenarların numaraları ve renkleri verilmektedir. Her yüzey için iki satır bilgi bulunduğundan, dosyanın bu bölümünde, toplam yüzey sayısının iki katı kadar satır vardır. İlgili yüzeye ait bilgilerin ilk satırında, o yüzeyi sınırlayan kenarların numaraları yer alır. Bu satırlarda, 3 ya da 4 kenar numarası (aralarında birer boşluk bulunmak üzere) verilmektedir. Eğer 3 kenar varsa o yüzey üçgen, 4 kenar varsa dörtgen şeklindedir. Daha çok kenardan oluşan yüzeyler ise, parçalara ayrılarak, onları oluşturan üçgen ya da dörtgen parçaları ile temsil edilebilir. Dosyaya koymadan önce bu işlemin yapılması gerekmektedir. Bu şekilde yapılmasının nedeni, bilgisayarda oluşturulan veri yapısındaki karmaşıklığı azaltmaktır. Dosyanın daha esnek olarak, 4'ten çok kenarlı yüzeyleri de içermesi sağlanabilirdi. Dörtten çok kenarlı nesneleri, kendisini oluşturan üçgenlere parçalayan bir program parçası yazılabilirdi. İlgili yüzeye ait bilgi içeren ikinci satırda ise, o yüzeyin renginin r(red), g(green), b(blue), yani kırmızı, yeşil ve mavi renk bileşenlerinin 0 ile 255 arasında tamsayı yoğunluk değerleri yer almaktadır (aralarında birer boşluk olmak üzere). Tüm renkler, bu temel renklerin belirli oranlarda karışımından elde edilebilir. Örnek olarak, yoğun bir sarı renk elde etmek için, kırmızı ve yeşil bileşenler için 255, mavi için ise 0 değerleri verilir. Temel üç rengin belirli oranlarda karışımlarının hangi renklere karşılık geldikleri, programda kullanılan Color Diyalog Kutusu içindeki Red, Green, Blue değerleri değiştirilerek kolaylıkla gözlenebilir.

Küp, doğal olarak dörtgen şeklindeki çokgen yüzeylerin arasında kalan bir hacim olduğu için, örnekte her yüzey için 4 kenar verilmiştir. Üçgen tanımlamak için, dosyada dördüncü kenar numarası olarak -1 verilmelidir.

Dosya hazırlanırken, program tarafından üç boyutlu nesnenin geçerli bir nesne olup olmadığına denetimi yapılmaması nedeni ile tüm elemanları doğru verilmelidir. Programa ek yapılarak, kenarlara veya köşelere de renk değerleri atanabilirdi. Fakat bu kısımlar, programı ve dosyaları uzatacağı ve çok gerekli olmadığı için yapılmamıştır.

Son üç satırda ise, her eleman gerçel sayı ve her satırdaki elemanların arasında birer boşluk olmak üzere sıra ile, nesnenin ölçeklendirme faktörleri, nesnenin döndürme açıları

(derece cinsinden), nesnenin yer deęiřtirme uzaklıkları verilir. Böylece, nesneler uzayda, herhangi bir büyüklükte (ölçeklendirme faktörleri yardımı ile), herhangi bir açıda (döndürme açıları yardımı ile), herhangi bir konuma (yer deęiřtirme uzaklıkları yardımı ile) yerleřtirilebilir.

Buna benzer dosyalar yazılarak, piramit gibi üç boyutlu nesneler tanımlanabildięi gibi, iki boyutlu çokgenler de oluşturulabilir. Örnek olarak, bir dikdörtgen, 4 köře, 4 kenar ve 2 yüzey (arka ve ön) bilgisi ile tanımlanabilir. Silindir, koni, küre gibi eğrisel yüzeyler de, küçük üçgen veya dörtgen gibi parçalara bölünerek, yaklaşık olarak tanımlanabilir. Fakat, çok sayıda köře, kenar ya da yüzey bilgisi hesaplanması ve verilmesi gerekeceğinden bu dosyanın hazırlanması uzun sürecektir. Bu řekilde eğrisel nesnelere ilişkin dosyaların oluşturulmasını saęlayan program parçaları yazılarak bu işlem de kolaylařtırılacaktır.

Programda, toplam olarak en fazla 10 tane üç boyutlu nesnenin veri yapısına yüklenmesine izin verilmektedir. Bu yüzden, 10 nesne yüklendikten sonra bu seçeneğın çalıştırılması durumunda, daha fazla nesne yüklenmeyecektir ve kullanıcı, veri yapısına daha fazla ek yapılamayacağı konusunda bilgilendirilecektir. Program başladığında veya DelShapes komutu verildikten sonra, veri yapısında herhangi bir nesne yoktur. Yüklenen ilk nesneye 0 numarası, sonraki her yüklenen nesneye, sıra ile 9'a kadar (dahil) numara verilmektedir.

7.6.1.2Script (Senaryo) alt seçeneęi

Bu komut çalıştırıldığında, ekrana çıkan diyalog kutusundan, veri yapısına yüklenmiş üç boyutlu nesnelerin hareket bilgilerini içeren dosyanın seçilmesi saęlanır. Bu dosya satır satır işlenerek, geçerli olanlar, veri yapısında ve ekranda gerekli deęişikliklerin yapılmasını saęlar. Satır sayısı konusunda sınırlama yoktur. Her satırdaki ilk eleman işlem no olup, işlem türünü belirten bir tamsayıdır.

İşlem No	İşlem Türü
0	Ölçeklendirme
1	Döndürme
2	Yer Değiştirme
3	Yer Değiştirme (Kamera)
4	Döndürme (Kamera)
5	İzdüşüm
10	Görüntüle

İşlem numarası, 0, 1, 2 olan satırların biçimi şöyledir (t tamsayı, r gerçel sayı olmak üzere) :

İşlem No(t) dönüşümx(r) dönüşümy(r) dönüşümxz(r) şekil no(t) incr(t)

dönüşümx : x eksenindeki dönüşüm

dönüşümy : y eksenindeki dönüşüm

dönüşümxz : z eksenindeki dönüşüm

şekil no : işlemlerin gerçekleştirileceği nesnenin veri yapısındaki numarası

incr : dönüşümx, dönüşümy ve dönüşümxz'de verilen sayıların, kamera parametrelerine etki şekli. 0 olması durumunda atama, 1 olması durumunda artış olacaktır.

Örnek :

Elimizde, kenarları 100 birim olan bir küp nesnesi olsun. Bu şekil, 0. nesne olarak veri yapısına yüklensin. Senaryo için okutulan dosyadan

0 2 2 2 0 0

şeklinde bir satırın anlamı incelenecek olursa :

İşlem numarası 0 olduğu için, ölçeklendirme dönüşümü yapılacaktır. Sırası ile x, y ve z eksenlerindeki ölçeklendirme faktörleri okunacaktır. Bu işlemler, sadece 0 numaralı nesne için yapılacaktır ve en son elemandan da, ölçeklendirme faktörlerine atama yapılacağı sonucu çıkmaktadır. Dolayısı ile, bu satır sonucunda, daha önceki ölçeklendirme faktörleri ne olursa olsun kenarları 200 birim olan bir küp nesnesi oluşacaktır. Satır

0 2 2 2 0 1

şeklinde olsaydı, ölçeklendirme faktörleri atanmayıp ilgili nesne üzerindeki öncekilerine ilave edilecekti. Yani, ölçeklendirme faktörleri 3,3,3 olan nesnenin (kenarları 300 birim olan küp nesnesi) -bu satırın işlenmesi ile birlikte- ölçeklendirme faktörleri 5,5,5 olacaktı (kenarları 500 birim olan küp nesnesi).

İşlem numarası, 3, 4 olan satırların biçimi de şöyledir (t tamsayı, r gerçel sayı olmak üzere) :

İşlem No(t) dönüşümx(r) dönüşümy(r) dönüşümxz(r) incr(t)

dönüşümx : x eksenindeki dönüşüm

dönüşümy : y eksenindeki dönüşüm

dönüşümxz : z eksenindeki dönüşüm

incr : dönüşümx, dönüşümy ve dönüşümxz'de verilen sayıların, kamera parametrelerine etki şekli. 0 olması durumunda atama, 1 olması durumunda artış olacaktır.

Bunun öncekilerden farkı, işlemlerin gerçekleştirileceği nesnenin veri yapısındaki numarasının verilmemesidir. Farklılığın nedeni ise, kamera hareketi olması nedeni ile, bu işlemlerin veri yapısında yüklü olan tüm nesnelere uygulanmasıdır. Kameranın yeri ve yönü değiştirilmektedir.

İşlem numarası 5 olan satırlarda ise, İşlem No'dan sonra 0 veya 1 gelebilir. 1 olursa İzduşüm türü paralel, 0 olursa perspektif olarak ele alınır.

İşlem numarası 10 olan satırlarda başka eleman yoktur. Sadece, tüm veri yapısındaki nesnelerin ekranda görüntülenmesi sağlanır. Böylece, dönüşüm faktörleri değiştirilerek elde edilen görüntü, bu işlemlerin yapılmasından sonra ekranda görüntülenir.

Program boyunca, senaryo dosyaları defalarca yüklenebilir. Yüklenen değişik nesneler üzerinde, aynı senaryo dosyaları da yüklenerek benzeri senaryolar, farklı yapıda nesneler üzerinde de denenebilir. Aynı senaryo dosyalarının arka arkaya çalıştırılmasında da bir sorun olmaz. Gözönüne alınması gereken konu, senaryo içinde, nesneler ile kameranın durumu arasındaki ilişkiyi uygun durumda tutmaktır.

7.6.1.3Window(Pencere) alt seçeneği

Bu komut çalıştırıldığında, ekrana çıkan diyalog kutusundan, gerçek koordinat sistemi penceresi ve araç penceresine ilişkin bilgilerin yer aldığı dosya (.win uzantılı) seçilir.

Bu dosyalar toplam iki satır olup, her satırdaki elemanların arasında birer boşluk olmak üzere, sıra ile, gerçek koordinat sistemi penceresi ve araç penceresinin köşe noktalarının (x,y) koordinatlarını içerir. İlk satırdaki gerçek koordinat sistemi penceresinin elemanları gerçel sayı olup ilk ikisi, dikdörtgen pencerenin en küçük (x,y) değerli köşesinin (xwmin,ywmin), sonraki ikisi ise en büyük değerli (x,y) köşesinin (xwmax,ywmax) koordinatlarıdır. İkinci satırdaki araç penceresinin elemanları tamsayı olup, gerçek koordinat sistemi penceresinin ekrandaki pencerede hangi noktalara eşleneceğini gösterir. İlk iki elemanı, dikdörtgen pencerenin en küçük (x,y) değerli köşesinin (xvmin,yvmin), sonraki ikisi ise en büyük değerli (x,y) köşesinin (xvmax,yvmax) koordinatlarıdır. Bu alt seçeneğin nesnelerin veri yapılarına yüklendiği dosyadan ayrı tutulmasının nedeni, yüklenen birden çok nesne için farklı boyutta koordinat sistem pencereleri tanımlanmasının nesneler arasında açabileceği uyumsuzlukları önlemektir.

-500 -500 500 500 // Gerçek koordinat sistemi penceresinde görüntülenecek olan kısım
0 0 300 300 // Gerçek koordinat sistemi penceresinin, ekranda kaplayacağı ve eşleneceği dikdörtgen alanın köşe noktaları.

7.6.2 Draw3D (Çizdir) alt seçeneği

Bu komutun çalıştırılması ile, veri yapılarına yüklenmiş olan nesneler en son halleri ile ekranda görüntülenirler.

7.6.3 DelShapes (Şekilleri Sil) alt seçeneği

Bu komutun çalıştırılması ile, veri yapısına yüklü olan nesneler silinir. Böylece, programdan çıkılmadan, veri yapısına yeni nesneler eklenebilir. Burada tüm köşe, kenar ve yüzey bilgileri iptal edilerek bellekten silinmektedir.

7.6.4 Wireframe (Telkafes) alt seçeneği

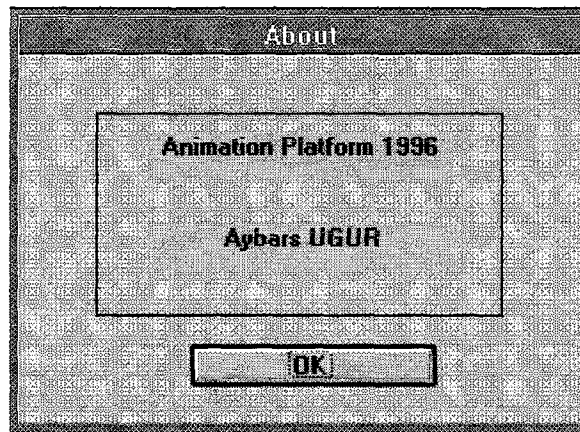
Bu, onaylı bir alt seçenektir. Seçenek üzerine bir kere basılarak yanında onay işareti belirmesi sağlanır. Tekrar basıldığında ise, onay işareti kalkar. Onaylı olduğunda, ekrana çizilecek olan şekillerin sadece sınırlarını oluşturan çizgiler çizilir. Onaysız olduğunda ise yüzey bilgileri de dahil edilir.

7.6.5 Parallel (Paralel) alt seçeneği

Bu, onaylı bir alt seçenektir. Seçenek üzerine bir kere basılarak yanında onay işareti belirmesi sağlanır. Tekrar basıldığında ise, onay işareti kalkar. Onaylı olduğunda, ekrana çizilecek olan şekiller paralel izdüşümleri alınarak, onaysız olduğunda ise perspektif izdüşümleri alınarak çizilecektir.

7.7 Help (Yardım) Seçeneği

Help seçeneğinin altında tek alt seçenek vardır : About. Bu komutun çalıştırılması ile açılan bilgilendirme diyalog kutusunda programın adı, yapıldığı yıl, yapan kişinin ismi ve "OK" komut düğmesi yer alır (Şekil 7.15). Bu diyalog kutusunun kapatılması için "OK" komut düğmesine basılır.



Şekil 7.15 : About Diyalog Kutusu

8. PROGRAM YARDIMI ile HAZIRLANAN ÖRNEK ANİMASYONLAR

Bu bölümde, program yardımı ile hazırlanan örnek animasyonlar üzerinde durulacaktır.

8.1 Windows Ortamından Alınan Resimlerin Arka Arkaya Gösterilmesi

Windows ortamında, başka bir program aracılığı ile gösterilen animasyonları, hazırlanan drawanim programı aracılığı ile de göstermek ve resimler üzerinde değişiklikler yapıp geliştirmek de olasıdır. Bu işlem için öncelikle, bu animasyonun birkaç defa çalıştırılması gerekmektedir. Bu sırada, herbir resim panoya kaydedilip, drawanim programının (ya da gerekirse başka bir program kullanılarak) ClipBoard-Paste(Yapıştır) seçeneği ile program penceresine alınıp, Bitmap olarak, File-BMP-Save(Kaydet) seçeneği ile ikincil bellek ortamına kaydedilmelidir. Gerekirse, ClipBoard-Clip (Kesme) seçeneği ile, alınan ekranın sadece gerekli bölümleri bırakılarak kayıt yapılmalıdır. Bu işlem, animasyonu oluşturan resim karelerinin tümünün düzenlenip ikincil bellek ortamına kaydedilme işlemi tamamlanıncaya kadar sürdürülmelidir. Eğer, ikincil bellek ortamına kaydedilirken resimleri sıralama işlemi yapılmadı ise, resimler numara verilerek başka isim ile sıralı olarak kopyalanmalıdır. Sonra herhangi bir ASCII editör yardımı ile, animasyon resim kareleri arasındaki bekleme süresini ve sıra ile gösterilecek resimlerin isimlerini içeren bir dosya (.txt uzantılı) hazırlanmalıdır. Şekil 8.1'de, bu şekilde hazırlanmış bir dosya görülmektedir. Bu animasyonu çalıştırmak için yapılması gereken tek işlem, File-BMP-Animate (Canlandır) seçeneği yardımı ile, txt uzantılı bu dosyayı çalıştırmaktır. Bu dosyanın ilk satırında, arka arkaya gösterilecek olan resimler arasındaki bekleme süresi yer almaktadır. Yavaş çalışan bilgisayarlarda, özellikle büyük boyutlu ve çok renkli resimlerde, 0 olması yararlıdır. Bu bekleme süresi, her resimden önce FOR döngüsünün 0'dan bu sayıya kadar çalıştırılması ile gerçekleştirilir. Bu program hızlı bilgisayar ortamlarında gösterilirken, dosyadaki bu sayı büyütülerek gerekli hız düzenlemeleri yapılabilir. Sonraki satırlarda ise, arka arkaya gösterilecek olan resimlerin isimleri, sürücü ve dizin isimleri ile birlikte verilmiştir. Örnekte de görüldüğü gibi bu animasyonda, arka arkaya 18 farklı resim gösterilmektedir. Bazı bölümlerin veya tüm animasyonun tekrarlı olarak gösterilmesi isteniyorsa, ASCII yazı editörlerinin kopyala-yapıştır özelliği kullanılarak, ilgili satırlar dosyanın sonuna eklenmelidir.

zoe.txt dosyası

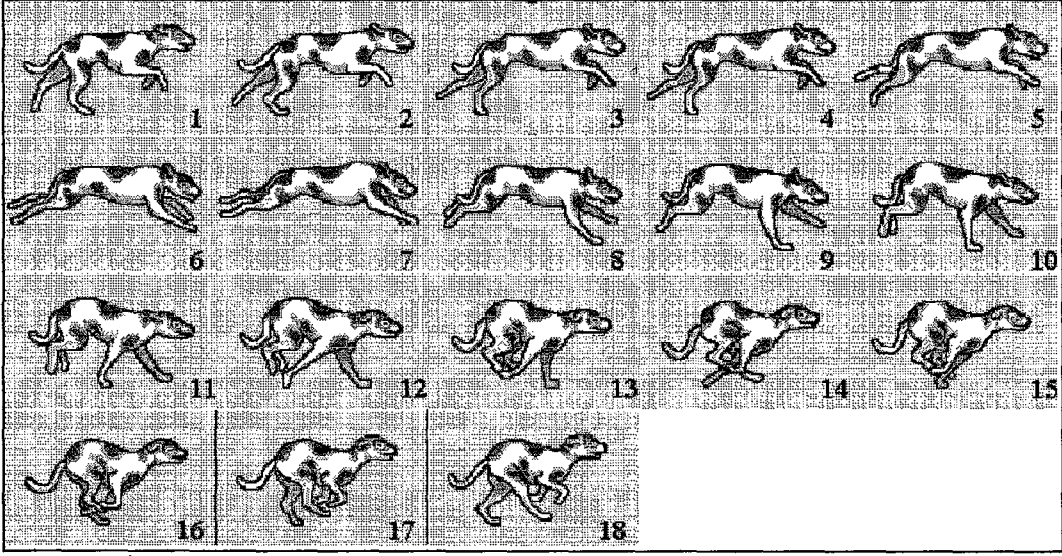
0

c:\drawanim\examples\zoe\zoe1.bmp
 c:\drawanim\examples\zoe\zoe2.bmp
 c:\drawanim\examples\zoe\zoe3.bmp
 c:\drawanim\examples\zoe\zoe4.bmp
 c:\drawanim\examples\zoe\zoe5.bmp
 c:\drawanim\examples\zoe\zoe6.bmp
 c:\drawanim\examples\zoe\zoe7.bmp
 c:\drawanim\examples\zoe\zoe8.bmp
 c:\drawanim\examples\zoe\zoe9.bmp
 c:\drawanim\examples\zoe\zoe10.bmp
 c:\drawanim\examples\zoe\zoe11.bmp
 c:\drawanim\examples\zoe\zoe12.bmp
 c:\drawanim\examples\zoe\zoe13.bmp
 c:\drawanim\examples\zoe\zoe14.bmp
 c:\drawanim\examples\zoe\zoe15.bmp
 c:\drawanim\examples\zoe\zoe16.bmp
 c:\drawanim\examples\zoe\zoe17.bmp
 c:\drawanim\examples\zoe\zoe18.bmp

Şekil 8.1 : İki boyutlu resimlerin arka arkaya gösterilmesini sağlayan bir animasyon dosyası.

Şekil 8.2'de zoe1.bmp'den zoe18.bmp'ye kadar, birbirinden az değişiklikler içeren resimler görülmektedir. Bu şekil de, programın File-BMP-Insert(Ekle) seçeneği yardımı ile düzenlenmiştir. Bu resimlerin arka arkaya gösterilmesi ile, resimlerdeki canlının hareket ediyor gibi görünmesi sağlanacaktır.

Bu resimler üzerine, programın çizim fonksiyonlarını kullanarak ekler yapılabilir. Örnek olarak, koşunun daha gerçekçi bir ortamda gerçekleştiğini göstermek için, her resim karesinin önüne çimen ve taş resmi eklenmiştir. Bunun için sırasıyla şu işlemler yapılmıştır:



Şekil 8.2 : İki boyutlu resimlerin arka arkaya gösterilmesi ile elde edilen bir animasyon örneğinin resim kareleri.

a) Çimen ve taş içeren bir resim çizilmiş (Şekil 8.3'te sağ alt köşedeki resim) ve bu, File-BMP-Save(Kaydet) seçeneği ile "stone.bmp" olarak ikincil bellek ortamına kaydedilmiştir.

i) İlk olarak, Attributes-Brush ile fırça rengi siyah olarak seçilmiş ve Draw-Floodfill ile, beyaz renkli olan tüm ekran siyaha boyanmıştır.

ii) Clipboard-Clip ile, bu resmin boyutu belirlenmiştir. Diğer resimlerin üzerine eklenecek bir resim parçası olduğu için boyutu küçük tutulmuştur.

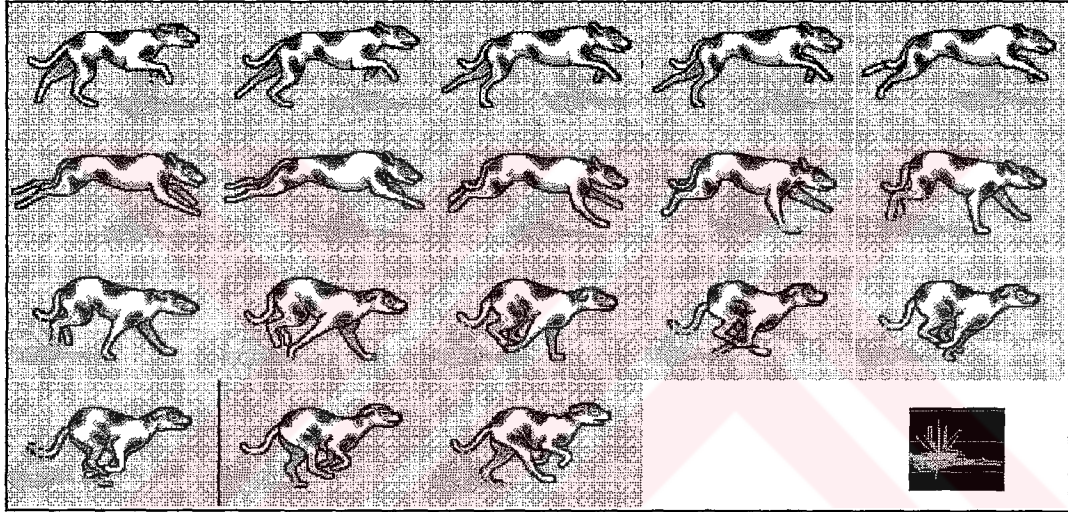
iii) Attributes seçeneğinin Pen ve Brush alt seçenekleri yardımı ile herbir nesne için kalem ve fırça rengi belirlenmiş ve Draw-Line(Çizgi), Draw-Ellipse(Elips) ve Draw-Point(Nokta) gibi çizim fonksiyonları kullanılarak bu nesneler çizilerek oluşturulmuştur. Burada, çimenler çizilirken kalem rengi olarak yeşil seçilmiş ve Draw-Point ile çizim yapılmış, toprak çizilirken kalem rengi gri olarak belirlenmiş ve Draw-Point ile çizim yapılmıştır. Taşlar çizilirken de, içlerinin doldurulması için fırça rengi olarak kahverengi seçilmiş ve küçük elipsler çizilerek oluşturulmuştur.

iv) File-BMP-Save ile, oluşturulan bu resim "stone.bmp" olarak kaydedilmiştir.

b) Her resim karesi ekrana tek tek alınarak, her birinin uygun konumlarına (animasyonun fiziksel kurallara göre gerçekçi görünmesini sağlayacak şekilde) "stone.bmp" resmi eklenerek tekrar ikincil bellek ortamına kaydedilmişlerdir. Resimleri birleştirme işlemi, ilgili resim karesinin pencereye File-BMP-Open (Aç) alt seçeneği

yardımları ile getirilmesinden sonra, File-BMP-Merge (Birleştir) alt seçeneği yardımı ile "stone.bmp" resminin uygun konuma eklenmesi ile gerçekleştirilir. "stone.bmp" oluşturulurken, zemin renginin siyah yapılması sayesinde, resmin sadece taş ve çimenleri gösteren kısmının resimlere eklenmesi sağlanmıştır. "stone.bmp" nin küçük boyutlarda olması, fare ile resimler üzerinde alacağı konumu belirlerken esneklik ve hız sağlamıştır.

Bu işlemlerden sonra ise, animasyon dosyası oluşturulmuş ve File-BMP-Animate ile çalıştırılmıştır.



Şekil 8.3 : İki boyutlu resimlerin arka arkaya gösterilmesi ile elde edilen bir animasyon örneğinin resim kareleri.

8.2 Animasyon Formatlarından Parçalanmış Resimlerin Arka Arkaya Gösterilmesi

AVI, MPEG gibi çok kullanılan animasyon formatlarındaki resim karelerini ayrı dosyalar halinde kaydetmeyi sağlayan programlar bulunmaktadır. Displ89 gibi Internet aracılığı ile elde edilebilecek bu tür programlar yardımı ile, önceden hazırlanmış olan animasyonlar, kendilerini oluşturan durağan resim parçalarına ayrılabilir. Bu işlem sırasında, Drawanim programının bu resim dosyalarını kolaylıkla kullanabilmesi için, Windows BMP formatında kaydedilmeleri uygun olacaktır. Bundan sonra yapılması gereken işlem, bu dosyaların isimlerini içeren animasyon dosyasının oluşturulup, Drawanim programının File-BMP-Animate seçeneği yardımı ile çalıştırılmasıdır. Örnek

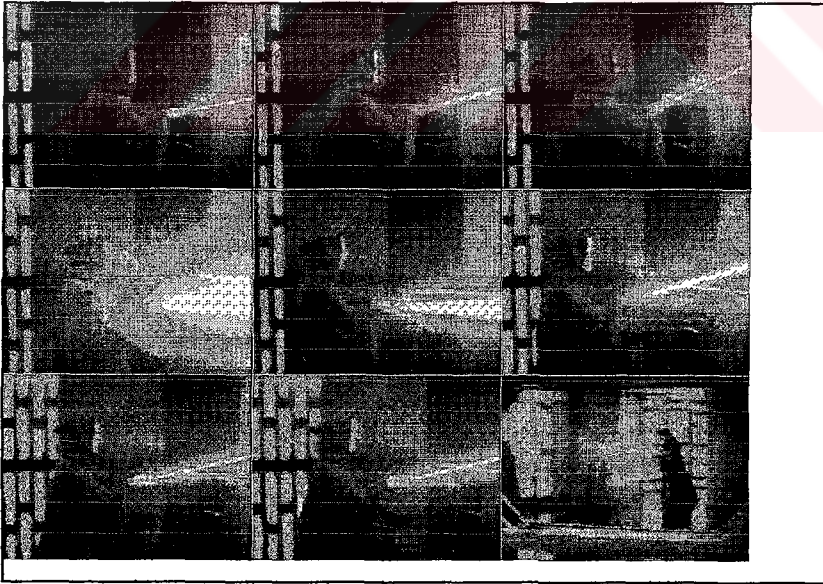
olarak, star wars filminin bir bölümünü içeren bir AVI animasyonu, resim karelerine ayrılarak arka arkaya gösterilmiştir. Şekil 8.4'te animasyon dosyası ve Şekil 8.5'te bu animasyonun resimleri görülmektedir. Bu animasyonlarda, tüm renkleri görebilmek için, Windows ekranında görüntülenebilen renk sayısını 16'nın üzerine çıkartmak gerekecektir (256 veya 65536 gibi).

starwars.txt dosyası

200000

c:\drawanim\examples\starwars\t0190001.bmp
 c:\drawanim\examples\starwars\t0190002.bmp
 c:\drawanim\examples\starwars\t0190003.bmp
 c:\drawanim\examples\starwars\t0190004.bmp
 c:\drawanim\examples\starwars\t0190005.bmp
 c:\drawanim\examples\starwars\t0190006.bmp
 c:\drawanim\examples\starwars\t0190007.bmp
 c:\drawanim\examples\starwars\t0190008.bmp
 c:\drawanim\examples\starwars\t0190009.bmp

Şekil 8.4 : İki boyutlu resimlerin arka arkaya gösterilmesini sağlayan bir animasyon dosyası.

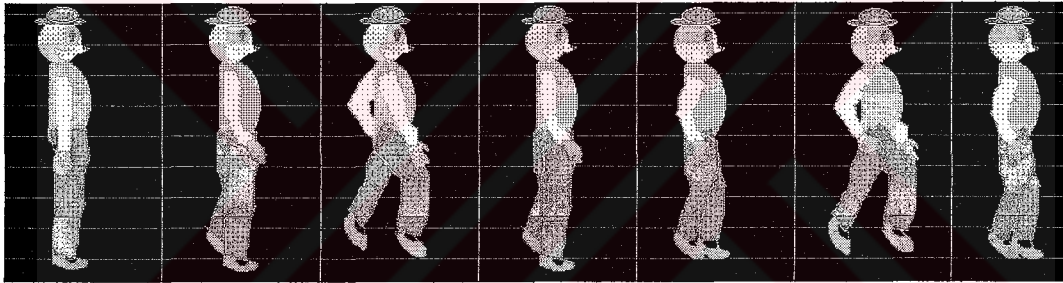


Şekil 8.5 : İki boyutlu resimlerin arka arkaya gösterilmesi ile elde edilen bir animasyon örneğinin resim kareleri.

8.3 Baştan Oluşturulan Resimlerin Arka Arkaya Gösterilmesi

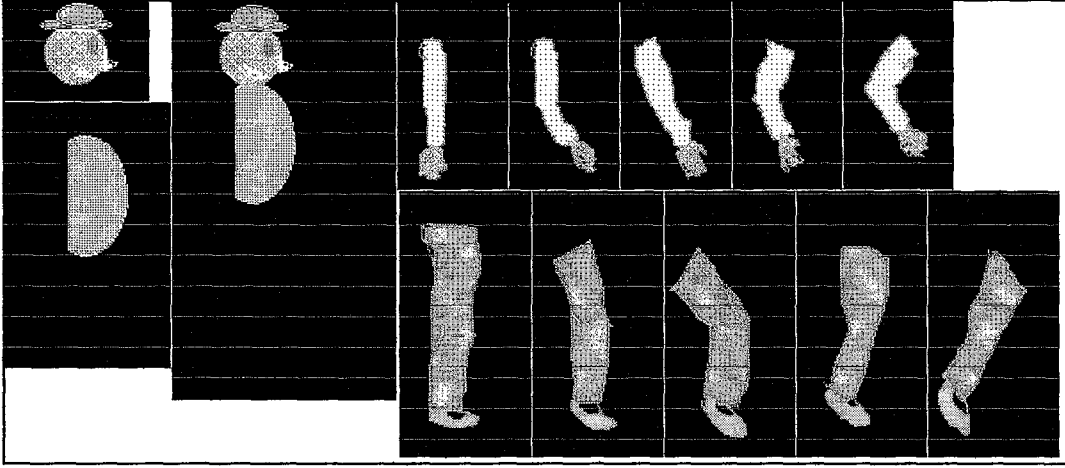
Animasyonlar, sadece programın çizim komutlarından yararlanılarak oluşturulan resimlerin arka arkaya gösterilmesi ile de oluşturulabilmektedir. Bu bölümde, bir insanın yürümesinin aşamalarını gösteren örnek bir animasyonun oluşturulması adım adım anlatılacaktır :

1) Animasyonu oluşturacak resimlerin sayısının belirlenmesi. Basit bir animasyon yapılacağı gözönüne alınarak, 7 resmin yeterli olacağı düşünülmüştür (Şekil 8.6). Bu şekilde, 1. resim durma anını, 2., 3., 4. resim sağ ayağını öne atıp sol ayağını onun yanına getirişini, 5., 6., 7. resim sol ayağını öne atıp sağ ayağını onun yanına getirişini göstermektedir.



Şekil 8.6 : Bir insanın yürümesinin aşamalarını gösteren animasyonun resimleri.

2) Bu ara resimlerin tasarlanıp bilgisayarda oluşturulması. Bu resimlerin oluşturulmasından önce, tekrar kullanılabilir parçaların tespiti yapılmalıdır. Örnek olarak 2. resim ile 4., 5. resim ile 7. aynıdır. Toplam olarak 5 resim kullanılacaktır. Basit bir animasyonda baş ve gövdenin hareket etmediği varsayılarak, birer kere tasarlanıp çizilmiş ve kaydedilmişlerdir. Büyük 5 resmi oluştururken, bu parçalar, File-BMP-Merge seçeneği yardımı ile, resmin üzerine tek tek eklenebilir. Aynı şekilde, kol ve bacak için de, 5'er farklı resim hazırlanmıştır. Toplam olarak 12 resim parçası kullanılarak, 5 resim elde edilmiştir (Şekil 8.7). Baş ve gövde birleştirilerek kaydedilmiş ve 5 resim için 5 ayrı birleştirme yerine 1 birleştirme yapılması sağlanmıştır.



Şekil 8.7 : Bir insanın yürümesinin aşamalarını gösteren animasyonun resim parçaları.

Büyük resmi oluşturmak üzere küçük resimler eklenirken, eklenecek resmin büyüklüğü ile doğru orantılı olacak şekilde, resmin konumunu belirlerken, işlemler yavaşlayacaktır. Bu nedenle, parçaların, kendilerini içlerine alabilecek en küçük boyutlu dikdörtgen bitmap resim alanlarına kaydedilmesi uygun görülmüştür. Siyah zemin üzerine, arkada kalan parçadan öne doğru yerleştirme işlemi (yağlı boyadakinine benzer şekilde) yapılmıştır. Her 5 resim için de, önce arkada kalan kol ve ardından arkada kalan bacak ve gövde-baş bileşimi eklenmiştir. Sonra da, yine sırası ile öndeki bacak ve kol eklenmiştir. Parçalar tekrar kullanılmasa idi, baş ve gövde 5'er kere, kol ve bacaklar da 10'ar kere çizileceğinden, 30 tane parça çizmek gerekecekti. 12 resim parçası kullanılarak, 18 resim parçasının çiziminden tasarruf edilmiş ve $18/30 = 60\%$ daha az iş yapılmıştır.

3) Resimlerin gösterim hızını ve hangi resmin hangi sırada gösterileceğini belirleyen animasyon dosyasının hazırlanması (Şekil 8.8). Farklı 5 resim, arka arkaya 7 resmin gösterilmesi ile yürüme animasyonunu oluşturacaktır. Üçüncü resimden sonra ikinci, beşinci resimden sonra dördüncü tekrar gösterilecektir.

Bu resimlerin taslakları el ile oluşturulup, tarayıcı yardımı ile, bilgisayar ortamında bitmap olarak saklanabilir. Sonra, boyama ve düzenleme yapılarak daha nitelikli animasyonlar da elde edilebilir.

walking.txt dosyası

2000000

c:\drawanim\examples\walking\man1.bmp

c:\drawanim\examples\walking\man2.bmp

c:\drawanim\examples\walking\man3.bmp

c:\drawanim\examples\walking\man2.bmp

c:\drawanim\examples\walking\man4.bmp

c:\drawanim\examples\walking\man5.bmp

c:\drawanim\examples\walking\man4.bmp

Şekil 8.8 : Bir insanın yürümesinin aşamalarını gösteren animasyon dosyası.

Bu resimler arka arkaya gösterilip, birden çok adım atan bir insan animasyonu oluşturulabilir veya farklı zeminlerde gösterilerek de tekrar kullanılabilir.

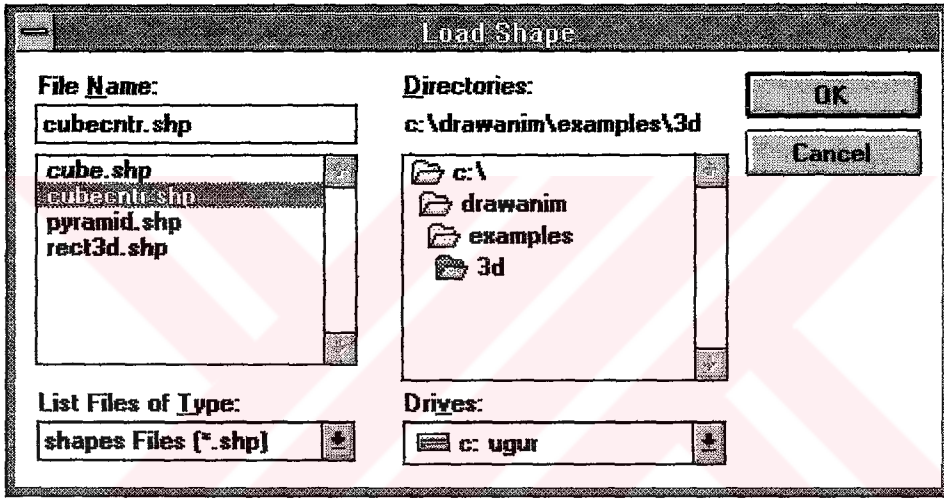
8.4 Üç Boyutlu Nesnelerin Çizim ve Animasyonu

Üç boyutlu nesnelerin animasyonu için özellikle Animate3D seçeneğinin alt seçeneklerinden yararlanılır. Ayrıca, animasyon sırasında oluşan görüntüler (her bir resim karesi), clip ve iki boyutlu çizim seçenekleri yardımı ile düzenlenip bitmap resim formatında, ikincil bellek ortamlarına kaydedilebilir. Kaydedilen bu resimler, File-Load-Animate alt seçeneği yardımı ile arka arkaya gösterilebilir. Bu sayede, üç boyutlu resimler üzerinde iki boyutlu çizim seçeneklerini kullanarak iki ve üç boyutlu animasyonlar birleştirilmiş olur. İki boyutlu resimler arka arkaya gösterildiği için, bu yöntemde, üç boyutlu animasyondaki hesaplamaların yol açtığı gecikmenin ve her bir yüzeyin çizilmesi sırasında meydana gelen titreşimin de önüne geçilmiş olur.

Üç boyutlu nesnelerin animasyonu için üç tür dosya hazırlamak gerekmektedir. İlk dosya türü, veri yapılarına yüklenecek nesne bilgilerinin yer aldığı .shp uzantılı dosyalardır. Temel olarak nesnelerin, köşe, kenar ve yüzey bilgileri ile, uzaydaki yer, açı ve büyüklüklerinin bilgilerini içerir. İkinci dosya türü, gerçek koordinat sistemi ve araç koordinat sistemi arasındaki eşlemeyi yapmak üzere, bunların boyutlarını içeren pencere dosyalarıdır. Bu dosyalar ".win" uzantılıdır. Üçüncü dosya türü ise, ".scr" uzantılı olan ve

nesnelerin ve kameranın hareket bilgilerini içeren senaryo dosyalarıdır. Nesnelerin yer değiştirmesini, ölçeklendirilmesini ve döndürülmesini sağlar.

Daha önceki bölümlerde biçimleri anlatılan bu dosyalar hazırlandıktan sonra, sıra bunların çalıştırılmasına gelmiştir. İlk önce, Animate3D-Load-Shape seçeneği yardımı ile çıkan diyalog kutusundan ".shp" uzantılı bir dosya okunarak küp, piramit, dikdörtgen gibi üç boyutlu bir nesne, veri yapısına yüklenir (Şekil 8.9). Bu seçenek tekrar çalıştırılarak, başka dosyalardan başka nesnelerin (10 taneye kadar) veri yapısına yüklenmesi de olasıdır.



Şekil 8.9 : İşlem merkezi ortasında olan üç boyutlu küp nesnesinin, veri yapısına yüklenmesini sağlayan diyalog kutusu.

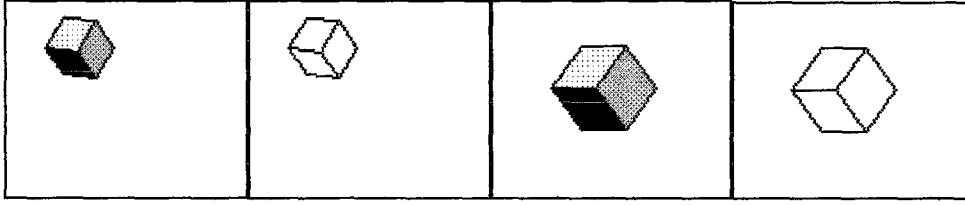
İşlem merkezi ortasında olan üç boyutlu küp nesnesinin, sınır temsili yolu ile tanımlandığı örnek "cubecnt.shp" dosyası Şekil 8.10'da görülmektedir.

8	8 köşenin x,y,z koordinatları
-50 -50 -50	
50 -50 -50	
-50 50 -50	
50 50 -50	
-50 -50 50	
50 -50 50	
-50 50 50	
50 50 50	
12	12 kenarın köşeleri
0 1	
0 2	
1 3	
2 3	
0 4	
1 5	
2 6	
3 7	
4 5	
4 6	
5 7	
6 7	
6	6 yüzeyin kenarları ve renkleri
0 2 3 1	
0 0 0	
2 5 10 7	
255 0 0	
0 4 8 5	
0 255 0	
1 6 9 4	
0 0 255	
3 7 11 6	
255 255 0	
8 9 11 10	
255 0 255	
1 1 1	
45 45 0	
200 -200 500	Ölçeklendirme parametrelerinin değerleri 1. Kendi boyutunda, kenar uzunluğu 100. x ve y eksenlerinde 45 derece döndürülmüş, z ekseninde döndürülmemiş. merkezi, x=200, y=-200, z=500 noktasına alınmış.

Şekil 8.10 : İşlem merkezi ortasında olan üç boyutlu küp nesnesinin tanımlandığı "cubecntr.shp" dosyası.

Gerçek koordinat sistemi ve araç koordinat sistemi penceresinin boyutlarının değiştirilmesi için de, Animate3D-Load-Window alt seçeneğine basıldığında ortaya çıkan diyalog kutusundan ".win" uzantılı pencere dosyası seçilir.

Bu iki dosya türünün ana belleğe yüklenmesinden sonra, Animate3D-Draw3D alt seçeneği yardımı ile, bu üç boyutlu nesne ekrana çizilebilir (Şekil 8.11). Animate3D-Wireframe onaylı durumdayken sadece kenarları çizilir, onaylı değilken ise, yüzey renkleri ile birlikte yüzeyleri çizilerek nesnenin şekli daha iyi anlaşılır. Animate3D-Parallel onaylı iken, bakış noktası sonsuzda kabul edilerek şeklin daha düzgün çıkması, onaylı değilken z ekseninde verilen noktada kabul edilerek daha gerçekçi çıkması sağlanır.



Şekil 8.11 : Üç boyutlu nesnelerin ekrana çizilmesi. Soldan sağa, a) Wireframe ve Parallel'in onaysız olduğu durum b) Wireframe'in onaylı, Parallel'in onaysız olduğu durum c) Wireframe'in onaysız, Parallel'in onaylı olduğu durum d) Wireframe ve Parallel'in onaylı olduğu durum. Vcri yapılarına yüklenmiş olan küp nesnesini, Animate3D-Draw3D yardımı ile, ekranda değişik şekillerde görüntülenmek olasıdır.

Nesnelerin yüklenmesinden sonra, ekrana doğrudan çizdirmek yerine, animasyon da yapılabilir. Bunun için, Animate3D-Load-Script seçeneğinden çıkan diyalog kutusu yardımı ile (Şekil 8.12), ".scr" uzantılı animasyon dosyasının seçilmesi gerekmektedir. Bu dosya, sonuna kadar satır satır okunarak, nesnelere ve kameraya ilişkin hareket bilgileri işlenmektedir.

Dosyada, önce, işlemler arasındaki bekleme miktarı verilmekte ve sırası ile, nesnenin x,y,z eksenlerindeki boyutu 4'er katına çıkarılmaktadır. Sonra, y eksenini etrafında 18 kere, 10'ar derece döndürülerek, her bir adımda ekranda görüntülenmektedir. İşlem merkezi ağırlık merkezi olarak tanımlandığı için (orijin), y eksenini de merkezinden geçmektedir. Bu aşamada toplam olarak 180 derece döndürülmüştür. Sonra, aynı anda x ve y eksenlerinde 10'ar derece döndürülerek, y ekseninde 40 birim yukarıya alınmaktadır. Bu işlem 9 kere tekrarlanarak, her bir adımda, küp nesnesi ekranda görüntülenmektedir.

10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 0 10 0 0 1 // 0 nolu nesne, tanımlandığı y eksenini etrafında 10 derece döndürülüyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 1 10 10 0 0 1 // 0 nolu nesne, x,y eksenlerinde 10'ar derece döndürülüyor
 2 0 40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim yukarıya alınıyor
 10 // Son durum ekrana çiziliyor
 0 4.2 4.2 4 0 0 // 0 nolu nesne x ve y eksenlerinde, boyutunun 4.2 katı yapıyor

[illegible]

10 // Son durum ekrana çiziliyor

1 -10 -10 0 0 1 // 0 nolu nesne, x,y eksenlerinde -10'ar derece döndürülüyor

2 0 -40 0 0 1 // 0 nolu nesne, y ekseninde 40 birim aşağıya alınıyor

10 // Son durum ekrana çiziliyor



Döndürme



Döndürerek yer değiştirme



Ölçeklendirme



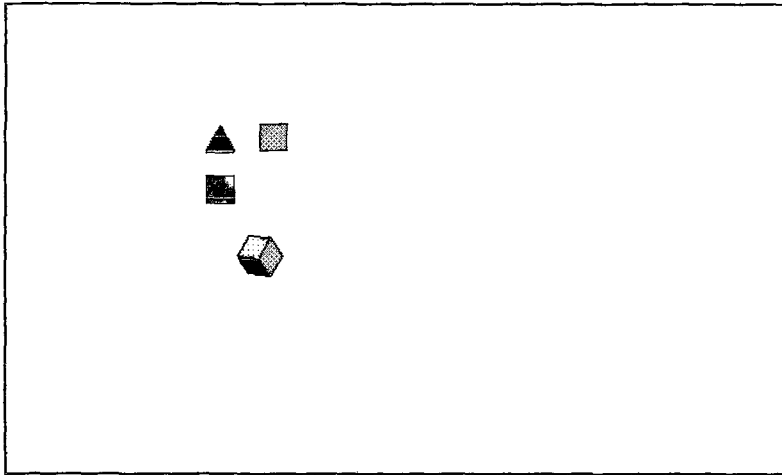


Döndürerek yer deęiřtirme



Animasyon dosyasının yüklenmesinden önce arka plana bitmap resmi getirilerek veya yeni bir resim oluşturularak üç boyutlu nesnelerin animasyonunun arka plan resminin önünde yapılması da olasıdır.

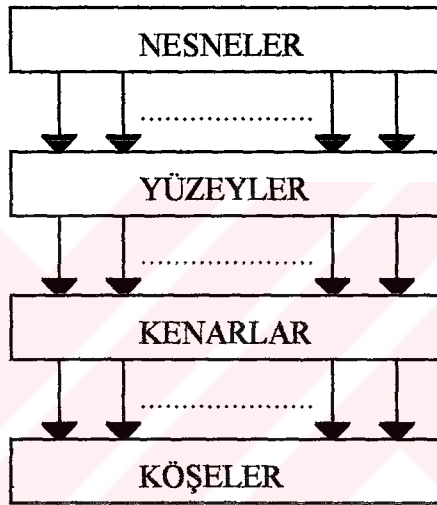
Başta da belirtildięi gibi, birden fazla nesneyi veri yapılarında tutup ekranda görüntülemek ve hareket ettirmek de olasıdır. İki tane küp nesnesi, bir piramit ve bir dikdörtgenden oluşan bir görüntü Şekil 8.13'te görölmektedir.



Şekil 8.13 : Üç boyutlu birkaç nesnenin çizilmesi ile oluşan görüntü.

9. KULLANILAN VERİ YAPILARI ve GÖRÜNTÜLEME

Üç boyutlu nesneler, kendilerini oluşturan yüzeyler, kenarlar ve köşeler şeklinde düzey düzey tanımlanmışlardır. Aynı düzeyli elemanlar ve bir alt düzeydeki elemanlar arasında bağlantılar kurulmuştur (Şekil 9.1). Örnek olarak, nesneler düzeyindeki nesnelerden her birisi için, işaretçiler yardımı ile, kendilerini oluşturan yüzeylere bağlantılar kurulmuştur. Benzer bağlantılar, yüzeyler ile yüzeylerin sınırlarını belirleyen kenarlar ve kenarlar ile kenarları oluşturan köşeler arasında da vardır.



Şekil 9.1 : Çokgen düzlemsel yüzeyler şeklinde tanımlanan katı nesneler

Program içerisinde, katı nesneler ile ilgili olarak hazırlanan veri yapıları şu şekildedir :

Köşeler için : "struct vertex { float x, y, z, w } vlist[256]; ". "vlist" değişkeni, 256 tane vertex (köşe) yapısı içeren bir dizidir. Dolayısı ile, aynı anda belleğe alınarak yüklenebilecek olan nesnelerin toplam köşe sayısı 256'dır. İstenirse, programın birkaç yerinde yapılacak değişiklikler ile bu sayı kolaylıkla artırılabilir. Bir köşe yapısı, tekdüze gösterimde (w), köşenin x, y ve z koordinatlarını içerir. Nesneler eklendikçe, köşe listesinin sonunu gösteren değişken, ilgili nesnelerin köşe sayısı kadar artırılacak ve her bir köşe dizisine eklenecektir. İlk köşe, 0. köşedir.

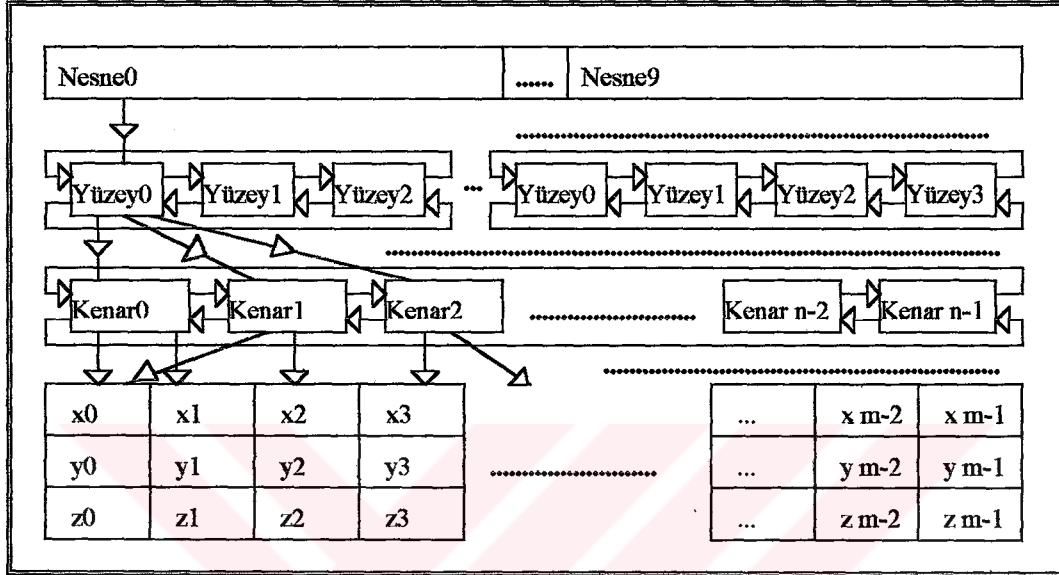
Kenarlar için : "struct proenode { int nume; struct proenode *left, *right; int v[2]; } *stproe;". Tüm nesnelerin tüm kenarları, çift yönlü bağlaçlı liste olarak birbirine bağlanmıştır. Böylece, kenar bilgileri dinamik olarak kullanılabilir. Kenar bilgileri, ancak belleğe yeni nesneler yüklendiğinde eklenecektir ve gerektiği kadar bellek kullanılacaktır. Tüm nesnelerin silinmesi durumunda, tüm kenarlar bellekten silinip tekrar kullanılır hale gelecektir. "stproe", kenarların başlangıcını gösteren işaretçidir. Sola ve sağa gidilerek diğer kenarlara ulaşılabilir. "left" ve "right" ise, yine proenode (primitive object edge node) yapısındaki, soldaki ve sağdaki bellek bölgelerini gösteren işaretçilerdir. "nume", ilgili kenarın numarasıdır. Nesneler eklendikçe, kenar listesine, ilgili nesnelerin kenarları sayısınca kenar eklenecektir. İlk kenar 0. kenardır. Dinamik olarak, istenildiği kadar kenar eklemek olasıdır. "v[2]", kenarın iki uç noktasını gösteren köşelerin, 256 boyutundaki köşe dizisindeki yerlerini (indis) içerir.

Yüzeyler için : "struct profnode { int numf; struct profnode *left, *right; struct proenode *edge[4]; float zvalue; int v[4]; int r,g,b; } *stprof;". Bir nesneyi oluşturan tüm yüzeyler, kendi içerisinde, çift yönlü bağlaçlı liste olarak birbirine bağlanmıştır. Diğer nesnelerin yüzeyleri ile aralarında bağlantı yoktur. "stprof", ilgili nesnenin yüzeylerinin başlangıcını gösteren işaretçidir. "left" ve "right" yardımı ile sola ve sağa gidilerek diğer yüzeylerine ulaşılabilir. Her yüzey, 3 veya 4 kenardan (en fazla 4) oluştuğu için, kenar yapısındaki bellek bölgelerini gösteren "edge[4]", 4 elemanlı bir dizi olarak ele alınmıştır. "v[4]", üçgen veya dörtgenin köşelerinin köşe dizisindeki yerlerinin indisini tutar. "numf", ilgili yüzeyin numarasıdır. Her nesneye ait yüzeyler 0'dan başlamak üzere kendi içinde numaralanırlar. "zvalue", o yüzeyin köşelerinin z değerlerinin ortalaması alınarak hesaplanan yaklaşık derinliğini verir. "r", "g", "b" ise, o yüzeyin renginin kırmızı, yeşil ve mavi bileşenlerinin 0 ile 255 arasındaki yoğunluk değerleridir.

Nesneler için : "struct prosnode { float sprms[9]; struct profnode *firstface; } stprosl[10];". Programda, "stprosl" değişkeni 10 tane nesneyi alabilen bir dizidir. Her bir elemanı, yapı olarak, 9 tane dönüşüm değeri (yer değiştirme, ölçeklendirme ve döndürme için 3'er) ve kendisini sınırlayan yüzeylerin yer aldığı çift yönlü bağlaçlı listenin başlangıç adresini içerir.

Şekil 9.2'de, bu veri yapıları daha ayrıntılı bir şekilde gösterilmiştir. Bu şekilden, üç boyutlu nesnelere ilişkin bilgilerin bilgisayar belleğindeki tutulma biçimi, daha kolay

görülebilmektedir. Programda, Load-Shape(Şekil) alt seçeneği yardımı ile, dosyalarda bulunan nesne bilgileri veri yapılarına yüklenirler. Sırası ile köşe bilgileri, kenar bilgileri, yüzey bilgileri yüklenerek aralarındaki bağlantılar kurulur.

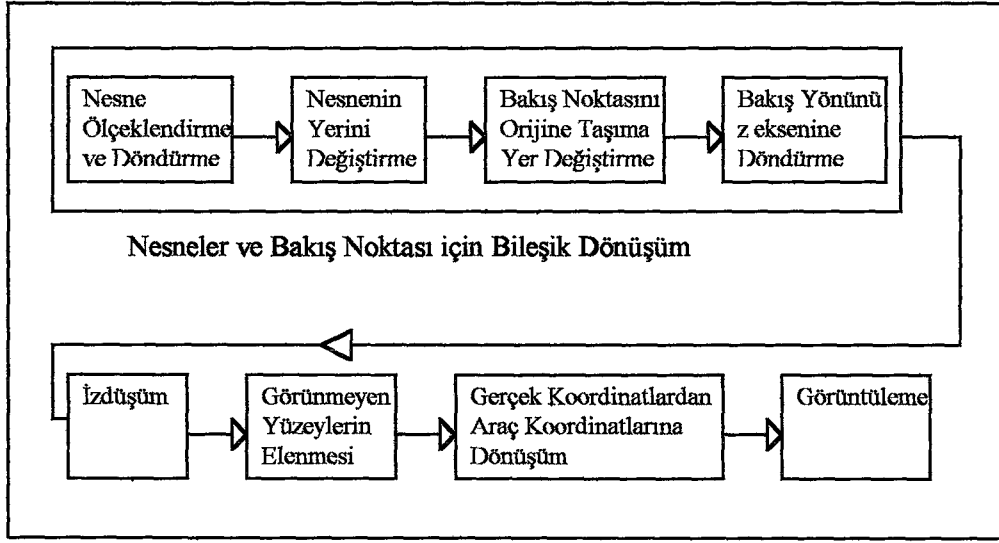


Şekil 9.2 : Çokgen düzlemsel yüzeyler şeklinde tanımlanan katı nesneler (ayrıntılı).

Algoritmaların daha etkin ve hızlı duruma getirilebilmesi için, veri yapıları, bağlantılar ve herbir düzeyin içerdiği veri çeşitlerinde birtakım değişiklikler yapılabilir.

Nesneleri görüntülemek için, bu veri yapılarındaki bilgiler kullanılarak, algoritmalar yardımı ile birtakım işlemlerin uygulanması gerekmektedir. Nesneleri görüntülemek için yapılması gereken işlemleri sırası ile gösteren görüntüleme hattı, Şekil 9.3'te gösterilmiştir.

Tüm nesnelerin uzaydaki yerlerini (yer değiştirme), açı (döndürme) ve büyüklüklerini (ölçeklendirme) belirleyen değerler, nesne düzeyinde ayrı ayrı yer almaktadır. Bakış noktası ve yönü ise, ekrana gelecek olan, dolayısı ile veri yapısında tutulan tüm nesneler için aynıdır. Bu iki değer grubu birleştirilerek, tüm nesnelerin tüm köşeleri için bileşik bir dönüşüm uygulanır. Böylece, bakış noktasına göre gerçek uzaydaki konumları belirlenmiş olur. Üç boyutlu nesnelerin görüntülenmesi sırasında izlenen adımlarda şu işlemler yapılır (Burger and Gillies, 1989) :



Şekil 9.3 : Üç boyutlu nesnelerin görüntülenmesi sırasında gerçekleştirilen işlemler.

1) Nesneler, tanımlandıkları andaki orijine göre ölçeklendirilir ve döndürülür. Burada kullanılan ölçeklendirme ve döndürme değerleri ayrı ayrı olup, nesne dizisinde her nesne için ayrı ayrı tutulurlar. Bu adım sonucunda nesneler, istenilen boyut ve açıya getirilmiş olurlar.

2) Uzaydaki yerleri değiştirilir. Burada kullanılan yer değiştirme değerleri ayrı ayrı olup nesne dizisinde her nesne için ayrı ayrı tutulurlar. Bu adım sonucunda nesneler, 1. adımdakine ek olarak, uzayda istenen yere getirilmiş olurlar.

3) Bakış noktası orijine taşınır. Bu, tüm nesneler için bir yer değiştirme işlemidir. Dolayısı ile bu değerler, nesneler düzeyinden de, katı nesneler için kullanılan veri yapılarının diğer düzeylerinden de ayrı olarak tutulurlar. Bu adım sonucunda nesneler, bakış noktasının orijin kabul edildiği bir uzaydaki konumlarına gelirler.

4) Bakış yönü z eksenine alınır. Bu, tüm nesneler için bir döndürme işlemidir. Dolayısı ile bu değerler, nesneler düzeyinden de, katı nesneler için kullanılan veri yapılarının diğer düzeylerinden de ayrı olarak tutulurlar. Bu adımda nesneler, gerçek uzayda, bir kamera veya bir gözün bakış yeri ve bakış yönüne göre yerleşmiş olurlar.

5) Nesneler için izdüşüm alınır. Nesnelerin z değerleri kaldırılarak paralel izdüşüm veya nesnelerin koordinatları bir formülden geçirilerek gerçekçi perspektif izdüşüm elde edilir. Bu adım sonunda üç boyutlu nesneler, iki boyutlu görüntü yüzeylerine yansıtılmış olurlar. İki boyutlu yüzeyde temsil edildiklerinden, z değerleri ortadan kalkmıştır.

6) Görünmeyen yüzeylerin elenmesi gerçekleştirilir. Görünmeyen yüzeyler görüntülenmeyecektir. Arka yüzeylerin elenmesi yöntemi ile, normali arkaya doğru bakan yüzeyler ortadan kaldırılır. görüntülenecek tüm nesnelerin tüm yüzeyleri, arkada olandan öne doğru sıralanarak da, görünmeyen yüzeylerin elenmesi otomatik olarak sağlanır. Üçgen veya dörtgen şeklindeki çokgenleri sıralamak için kullanılan en basit yöntem, köşelerinin z koordinatlarının derinlik ortalamalarını kullanarak sıralamaktır. Fakat bu yöntem, karmaşık nesneler ve nesne grupları için gerçek sonuçları vermez. Yüzeylerin köşelerinin derinlik ortalamaları, n tane köşe için, köşe numarası i ve ilgili köşenin z değeri z_i olmak üzere şu şekilde hesaplanır :

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

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 koordinatlara dönüşüm" diye adlandırılır.

8) Nesneler ekranda görüntülenir. Yüzeylerin çizimi, çokgen gibi basit iki boyutlu nesnelerin çizilmesini sağlayan çizim fonksiyonları yardımı ile gerçekleştirilir. Nesnelerin sadece kenarlarının çizilmesi isteniyorsa çizgiler veya içi boş çokgenler, katı bir nesne şeklinde renkli olarak çizilmesi isteniyorsa, içi dolu çokgenler kullanılır. Renk bilgileri, veri yapılarında, yüzeyler düzeyinde, her yüzey için, kırmızı, yeşil ve mavi yoğunluk bileşenleri olarak tutulmaktadır.

10. SONUÇ ve ÖNERİLER

1. Bilgisayar yazılım ve donanımlarında, son yıllarda görülen gelişmeler, bilgisayarın grafik ve animasyon konularında kullanılmasını sağlamıştır. Bunun sonucunda, animasyon üretimi kolaylaşmış, hızlanmış, nitelikli bir hale gelmiş ve maliyetler düşmüştür. Kişisel bilgisayar ortamında bile, iki ve üç boyutlu çizim ve animasyonlar kolaylıkla hazırlanabilmektedir. Yakın gelecekte, üç boyutlu işlemlerde kullanılan kartların yaygınlaşması ve daha da gelişmesi sayesinde, nitelik ve hız daha da artacaktır. Bir yılda bile büyük değişimler yaşanan bilgisayar grafikleri konusunda çalışan kişilerin, bu gelişmeleri iyi bir şekilde izlemeleri ve en yeni donanım ve yazılımları kullanmaları büyük önem taşımaktadır.

2. Programlama ortamları, iki ve üç boyutlu çizim ve animasyonlar hazırlamayı sağlayan uygulama programlarının (arayüzlerin) yazılmasını kolaylaştıracak şekilde gelişmektedir. Yazılım Mühendisliğindeki ilerlemeler ve bilgisayar grafiklerinin Nesneye Yönelik Programlama yaklaşımı temeline oturtulması da bunu desteklemektedir. Bilgisayar Grafikleri konusunda çalışan kişilerin, yeni çıkan programlama ortamlarının veya yeni sürümlerinin (uyarlama) bu konuya getirdiği kolaylıkları incelemeleri büyük önem taşımaktadır. Çünkü, bir önceki sürümünde onlarca satır kaynak kod gerektiren işlemler, yeni sürümde bir komut ile kolaylıkla yapılabilmektedir. Programlama ortamlarının kullanıcıya yakın olanakları, yardım desteği, hata ayıklama özellikleri, getirdikleri esneklikler, hazır kütüphane fonksiyonları, kullanıcı arayüzü ve grafik fonksiyonları, kişisel bilgisayarlarda grafik programlamayı kolaylaştırmıştır. Bir programlama ortamını kullanmaya başlayacak olan kişinin ilgili kaynakları ve örnekleri elde ederek incelemesi, gereksiz işler yapılmasının önüne geçecektir.

3. Animasyondan yani hareketli görüntülerden pekçok alanda yararlanılmaktadır. Bunlardan en önemlileri : Bilimsel çalışma ve araştırmalar, mühendislik, eğitim ve öğretim, film endüstrisi, reklamcılık, sanat, spor, bilgisayar oyunları ve sanal gerçeklik olarak sayılabilir.

4. Animasyon, gerçek yaşamda elde edilemeyecek veya elde edilse bile risk ve yüksek maliyete neden olabilecek hareketli görüntülerin, bu kötülükleri ortadan kaldıracak şekilde elde edilmesini sağlamaktadır. Örnek olarak, sanal olarak bilgisayar ortamına aktarılmış

bir şehirde dolaşmak, yapımına başlanmadan önce binaların içini gezmek, ortamda değişiklikler yaparak sınamak gibi çok sayıda işlem gerektiren değişik konularda yeni ufuklar açmıştır. Film endüstrisinde de, son yıllarda bilgisayar animasyonu kullanılarak yapılan filmler ilgi görmeye başlamıştır.

5. Bir nesnenin bilgisayar modeli, veri, algoritma ve yapı olmak üzere üç bileşenden oluşur. Yapı iyi tasarlanırsa, algoritmalar daha hızlı ve etkin bir duruma getirilebilir. Buradan da anlaşılabileceği gibi, bilgisayar grafikleri, kendi konu alanı dışında, Programlama Dilleri, Yazılım Mühendisliği, Algoritmalar, Veri Yapıları konularında yeterli düzeyde bilgi sahibi olmayı da gerektirmektedir. Ayrıca, üç boyutlu koordinat sistemi, vektörler, matrisler gibi matematiksel işlemler de yoğun olarak kullanılmaktadır.

6. Animasyon oluşturmak için hazırlanan uygulama programları, grafiksel arayüzler üzerine oturtulmalı, bulunduğu ortamın özelliklerinden yararlanabilmelidir. Örnek olarak, Nesne Bağlama ve Katma özelliği sayesinde, hazırlanmış olan diğer uygulama programları ile veri alışverişi yapılarak onların özelliklerinden de yararlanılabilmektedir. Böylece, hazırlanan program, bitmap dışındaki resim formatlarından da yararlanabilmektedir. Ayrıca, hazırlanan bu uygulama programının menü özellikleri tanıtılırken, hazırlanan resimler de, yine uygulama programında düzenlenip panoya ve oradan da kelime işlemciye aktarılmıştır. İki boyutlu ve üç boyutlu animasyonların aynı ortam üzerinde yapılması programa esneklik getirmiştir. Örnek olarak, üç boyutlu görüntüler üzerinde, iki boyutlu çizim fonksiyonları ile değişiklikler yapılabilmektedir. Temel çizim fonksiyonları, pencere silme, pencere boyutu değiştirme, resmin belli bir bölümünü kesme gibi işlemler de bir grafik paketinde olmalıdır.

7. Nesneler, kendilerini oluşturan yüzeyler, kenarlar ve köşeler şeklinde düzey düzey veri yapısına yüklenmektedirler. Nesnelerin ekranda görüntülenebilmesi için, nesnelere ve kameraya ait, yer değiştirme, ölçeklendirme ve döndürme gibi dönüşüm işlemlerinin bileşimini içeren matrisler oluşturulur. Görüntülenecek nesnelerin köşe noktalarına uygulanan bu dönüşüm işlemlerinden sonra, isteğe bağlı olarak, paralel veya perspektif izdüşüm uygulanır. Sonra, görünmeyen yüzeylerin elenmesi ve gerçek koordinatlardan araç koordinatlarına dönüşüm işlemleri gerçekleştirilerek, iki boyutlu çizim fonksiyonları yardımı ile nesneler pencereye çizilir. Uygulama programında, bir seçenek yardımı ile, ekrandaki görüntünün perspektiften paralele veya paralelden perspektife çevrilmesi

sayesinde, aralarındaki farklar kolaylıkla görülebilmektedir. Aynı şekilde, telkafes gösterimi ile, yüzeylerin renkli olarak gösterimi arasında da geçiş yapılabilmektedir.



YARARLANILAN KAYNAKLAR

Ammeraal, L., 1989, Programs and Data Structures in C, John Wiley & Sons, Great Britain, 206p.

Ammeraal, L., 1992, Programming Principles in Computer Graphics, John Wiley & Sons, USA, 233p.

Anand, V.B., 1993, Computer Graphics and Geometric Modeling for Engineers, John Wiley & Sons, Inc., USA, 407p.

Barkakati, N., 1993, Imaging and Animation for Windows, Sams Publishing, USA, 423p.

Burger, P. and Gillies, D., 1989, Interactive Computer Graphics (Functional, Procedural and Device Level Methods), Addison Wesley, Great Britain, 504p.

Deitel, H.M. and Deitel, P.J., 1994, C:How to Program, Prentice-Hall, USA, 926p.

Fox, D. and Waite, M., 1984, Computer Animation Primer, McGraw-Hill, USA, 501p.

Glassner, A.S., 1984, Computer Graphics User's Guide, Howard W. Sams & Co.,Inc., USA, 239p.

Grady, M.T., 1989, Turbo C! Programming Principles and Practices, McGraw-Hill, 396p.

Hearn, D. and Baker, M.P., 1986, Computer Graphics, Prentice-Hall, 352p.

Mäntylä, M., 1988, An Introduction to Solid Modelling, Computer Science Press, Inc., USA, 401p.

Miller, L.H. and Quilici, A.H., 1986, Programming in C, John Wiley & Sons, USA, 426p.

Murray III, W. H. and Pappas C. H., 1990, Windows Programming : An Introduction, McGraw-Hill, USA, 651p.

Oliver, D., Anderson, S., McCord, J., Gumas, S., Zigon, B., 1993, Tricks of the Graphics Gurus, Sams Publishing, USA, 894p.

Özgüç, B., 1995, Bilim ve Teknik, Sayı 328, Bilgisayar Grafik Yazılımı, Görüntünün Matematiği, Sayfa 18-25.

Schildt, H., 1989, Born to Code in C, McGraw-Hill, USA, 525p.

Schildt, H., 1995, C++ The Complete Reference Second Edition, McGraw-Hill, 912p.

Schildt, H., Pappas, C. H., Murray III, W. H., 1994, Osborne Windows Programming Series, Vol 1 : Programming Fundamentals, McGraw-Hill, USA, 649p.

Schildt, H., Pappas, C. H., Murray III, W. H., 1994, Osborne Windows Programming Series, Vol 2 : General Purpose API Functions, McGraw-Hill, USA, 964p.

Uğur, A., 1993, Grafik-İstatistik Paket Programı, Lisans Tezi, Ege Üniversitesi Bilgisayar Mühendisliği Bölümü, İzmir, 68s.

Watt, A., 1990, Fundamentals of Three-Dimensional Computer Graphics, Addison Wesley, Great Britain, 430p.

ÖZGEÇMİŞ

Adı Soyadı : Aybars UĞUR
Doğum Tarihi ve Yeri : 1972, İZMİR
Uyruğu : Türkiye Cumhuriyeti

Eğitim

1989 - 1993 EGE Üniversitesi Bilgisayar Mühendisliği Bölümü (Ort = 73/100)
1986 - 1989 İzmir İnönü Lisesi (Ort = 9.51/10, Lise 1.'liği)
1983 - 1986 İzmir İnönü Lisesi (Orta Bölüm)
1979 - 1983 Ertuğrulgazi İlkokulu

EGE Üniversitesi Bilgisayar Mühendisliği Bölümünde Araştırma Görevlisi olarak görev yapmaktayım (18-02-1994'ten itibaren).