RASTGELE ŞEKİLLİ NESNELERİN RADAR KESİT ALANLARININ KESTİRİMİ İÇİN GRAFİK KULLANICI ARAYÜZLÜ UYGULAMA GELİŞTİRİLMESİ

DEVELOPMENT OF A GRAPHICAL USER INTERFACE (GUI) APPLICATION FOR RADAR CROSS SECTION (RCS) PREDICTION OF ARBITRARILY SHAPED OBJECTS

HALİT AYANLI

Hacettepe Üniversitesi
Lisansüstü Eğitim – Öğretim ve Sınav Yönetmeliğinin
ELEKTRİK ve ELEKTRONİK Mühendisliği Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

2007

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma jürimiz tarafından ELEKTRİK ve ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Başkan	:……
	Prof. Dr. Hüseyin Selçuk GEÇİM
Üye (Danışman)	:……
	Yard. Doç. Dr. Mehmet DEMİRER
Üye (Varsa İkinci Danışman)	:……
	Prof. Dr. Erdem YAZGAN
Üye	:……
	Yard. Doç. Dr. Ali Ziya ALKAR
Üye	:……
	Doç. Dr. Abdullah ÇAVUŞOĞLU

ONAY

Bu tez ……/……/…… tarihinde Enstitü Yönetim Kurulunca kabul edilmiştir.

Prof. Dr. Erdem YAZGAN
Fen Bilimleri Enstitüsü Müdürü

Aileme ...

RASTGELE ŞEKİLLİ NESNELERİN RADAR KESİT ALANLARININ KESTİRİMİ İÇİN GRAFİK KULLANICI ARAYÜZLÜ UYGULAMA GELİŞTİRİLMESİ

Halit AYANLI

ÖZ

Radar kesit alan kavramı elektronik harp (EH) alanında önem arz eden konulardandır. Bu konudaki ihtiyaçlardan biri de karmaşık cisimlerin radar kesit alanlarının, makul ölçülerde zaman ve kaynak harcanarak gerçeğe yakın biçimde hesaplanabilmesidir. RCS kestirim programları bu ihtiyaca yanıt olabilmek adına kullanılırlar. Bu tezde de, belirtilen ihtiyaca yönelik bir programın üretilmesi amaçlanmıştır.

Bahsedilen amaç doğrultusunda, RCS kavramı araştırılmış ve tezde bu konuyla ilgili özet bilgi verilmiştir. Bunun yanında, tez kapsamında üretilmiş olan ve RCSP adı verilen radar kesit alan kestirim programı ile ilgili tasarım ve kodlama detayları tezde sunulmuştur. Ayrıca programda kullanılan ya da yararlanılan yöntem ve kavramlar aktarılmıştır.

Programın tasarımında nesne tabanlı yaklaşım benimsenmiştir. Kod, C++ dili kullanılarak Visual Studio .NET ortamında geliştirilmiştir. Tasarımda kolay anlaşılırlık ile değişime ve gelişime elverişlilik temel prensip olarak alınmıştır. Program, STL formatında dosyalardan aldığı hedef cisim bilgilerini kullanarak istenilen analizleri yapmakta ve RCS kestirim sonuçlarını görsel olarak kullanıcıya sunmaktadır.

ANAHTAR SÖZCÜKLER: Radar kesit alan kestirimi, RKA, Stereolithography, STL, OpenGL, ışın takibi, grafik kullanıcı arayüzü, Visual Studio .NET, C++, UML, nesne tabanlı programlama.

Danışman: Yard. Doç. Dr. Mehmet DEMİRER, Hacettepe Üniversitesi, Elektrik ve Elektronik Mühendisliği Anabilim Dalı

DEVELOPMENT OF A GRAPHICAL USER INTERFACE (GUI) APPLICATION FOR RADAR CROSS SECTION (RCS) PREDICTION OF ARBITRARILY SHAPED OBJECTS

Halit AYANLI

ABSTRACT

Radar cross section concept is an important issue in electronic warfare (EW). One of the major necessities in this area is the computation of radar cross sections for complex targets. These computations should consume reasonable time and source, and yield realistic results. RCS prediction tools are used to satisfy this necessity. In this thesis, the aim is to develop such a program.

Intending for the stated purpose, RCS concept was searched and brief information is given in the thesis about the subject. Besides, the design and coding details of the radar cross section prediction program developed during this study is submitted. Moreover, the other concepts and methods used in development phase of this program named RCSP are explained.

The program was designed using object-oriented techniques, and the code was developed in Visual Studio .NET platform using C++ programming language. The basic principle in design has been the understandability and improvability. The RCSP program executes different analysis on the target object data given in STL file format depending on the given computational parameters. Then the prediction results are submitted visually to the user.

KEYWORDS: Radar cross section prediction, RCS, Stereolithography, STL, OpenGL, ray tracing, graphical user interface, Visual Studio .NET, C++, UML, object-oriented programming.

Advisor: Asst. Prof. Dr. Mehmet DEMİRER, Hacettepe University, Department of Electrical and Electronics Engineering

TEŞEKKÜR

Yazar, bu çalışmanın gerçekleşmesinde katkılarından dolayı, aşağıda adı geçen kişi ve kuruluşlara içtenlikle teşekkür eder.

Tez danışmanı sayın Yard. Doç. Dr. Mehmet DEMİRER ve eş danışman sayın Prof. Dr. Erdem YAZGAN tez çalışmasının gerçekleştirilmesi için gerekli ortamı hazırlamışlar, çalışmanın sonuca ulaştırılmasında ve karşılaşılan güçlüklerin aşılmasında yön gösterici olmuşlardır.

Başta Yazılım Mühendisliği müdürü olmak üzere MİKES A.Ş. (yazarın çalıştığı şirket) yöneticileri ve yazarın çalışma arkadaşları, tezin tamamlanması aşamasında hoşgörü göstererek manevi olarak destek olmuşlardır.

Yazarın ailesi ve arkadaşları, bu çalışmanın gerçekleştirilmesi sırasında hiçbir yardımdan ve fedakarlıktan kaçınmamış, maddi ve manevi destek olmuşlardır.

İÇİNDEKİLER DİZİNİ

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

σ	RCS
λ	Dalga boyu
3D	3-D Result, üç boyutlu sonuç ekranı
3DS	3D Studio
a	Cisim boyutu
A	Yüzey alanı
A-R-D	Area-Reflectivity-Directivity (alan-yansıtıcılık-yönlülük)
AN	Analysis, analiz ekranı
a_1, a_2	Ana eksen yüzey kavisi yarıçapları
ASCII	American Standard Code for Information Interchange
C	Küre merkez noktası
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
D	Directivity (yönlülük)
D	Işın doğrultusu
dB	Desibel
dBsm	Metrekare başına desibel
dll	Dynamic Link Library
DWG	Autocad drawing database
E	Elektrik alan
EH	Elektronik Harp
EW	Electronic Warfare
e_x, e_y, e_z	Kartezyen koordinat eksenleri yönündeki birim vektörler
G	Anten kazancı
gc	Garbage Collection
GLU	OpenGL Utility library
GLUT	OpenGL Utility Toolkit library
GO	Geometric Optics
GR	Graphs, grafik sonuç ekranı
I	Küre üzerinde nokta
IEEE	Institute of Electrical and Electronics Engineers
IR	Infrared (enfraruj)
k	Dalga numarası
M	Döndürme matrisi
MATLAB	Matrix Laboratory
N	Yüzey normal vektörü
OOP	Object-oriented programming (nesne tabanlı programlama)
OpenGL	Open Graphics Library
P	Güç
P	Üç boyutlu kartezyen uzayında nokta
PO	Physical Optics
r	Küre yarıçapı
R	Mesafe
R	Reflectivity (yansıtıcılık)
RCS	Radar Cross Section (bkz. RKA)
RCSP	RCS Predictor (bkz. RKAK)
RF	Radyo Frekansı
RGB	Red Green Blue

RKA	Radar Kesit Alan
RKAK	RKA Kestirici
RPM	Rapid Prototyping Machine
RX	Receiver, alıcı ekranı
S	Işın başlangıç noktası
SC	Scalars, sayısal sonuç ekranı
sr	Steradyan, katı açı birimi
SSM	Savunma Sanayii Müsteşarlığı
STL	Stereolithography
TG	Target, hedef ekranı
TX	Transmitter, verici ekranı
UML	Unified Modeling Language (birleşik modelleme dili)

EKLER DİZİNİ

Sayfa

1. GİRİŞ

Dünyamız insanlık tarihi boyunca savaşlara sahne olmuştur. Önceleri belki sadece barınacak yer ya da yiyecek bulma maksadıyla, sonraları çoğunlukla insanoğlunun bitmez tükenmez hırsı nedeniyle var olmuştur savaşlar. Ne yazık ki sadece tarih kitaplarında kalması arzu edilen savaşlar günümüzde de tüm hızıyla devam etmektedir.

Her ne kadar toplumlar savaşla karşı karşıya kalmayı istemese de bazen bu kaçınılmaz olabilmektedir. Bu durumda yapılabilecekler o toplumun uluslar arası düzeydeki gücüne bağlı olmaktadır. Bir ülkenin savaş karşısındaki yaptırımı caydırıcı bir askeri güç ile mümkündür. Yine ne yazıktır ki ülkeler arasındaki bu güç yarışı tüm hızıyla sürmekte ve ülkeler bütçelerinin büyük bir bölümünü silahlanmaya ve askeri yatırımlara ayırmaktadır. Hatta öyle ki, soğuk savaşın sona ermesiyle dünyanın daha güvenilir olacağı beklentisine rağmen devletler arasındaki güvenlik sorununun aşılamaması nedeniyle silahlanmaya harcanan para soğuk savaş dönemini aratmamaktadır [1]. Bu nedenle ülkemiz de bu yarış içinde yerini korumaya çalışmaktadır. Ancak ülke çıkarları gözetilirken yapılan harcamaların da makul seviyelerde tutulması önem taşımaktadır.

Günümüz savaş yaklaşımları geçmişe göre farklılıklar göstermektedir. Yakın zamanda meydana gelen savaşlarda da görüldüğü üzere, savaşlar teknolojinin sonuna kadar kullanıldığı olaylar haline gelmiştir. Hem saldırının etkinliğinde hem de savunmanın başarısında teknolojinin büyük payı olmuştur.

Savaşlarda teknoloji karşımıza daha çok elektronik harp kavramıyla (EH) çıkmaktadır. Aslında elektronik harp, yalnızca savaş zamanını değil barış zamanını da içeren bir faaliyetler bütünüdür. Elektronik harp faaliyeti, barış zamanında istihbarat toplanması ile başlayan ve savaş zamanında düşman silah sistemlerinin etkisiz hale getirilmesi ile ülkelerin silahlı kuvvetlerini beklenen zafere ulaştıran bir sürecin en önemli unsurlarındandır. Elektronik harp sistemleri pasif veya aktif şekilde, koruyucu veya elektronik saldırı amaçları ile kullanılabilen ve kara, hava ve deniz platformlarındaki saldırı ve savunma silah sistemlerinin etkinliğini artıran en önemli sistemlerdir [2].

Elektronik harp faaliyetleri sürecinin son aşamasında bulunan elektronik harp sistemlerinin, mevcut ve muhtemel tehditlere karşı etkili olabilmesi için tamamen milli ihtiyaçlar doğrultusunda milli istihbarat bilgilerine dayandırılarak milli olarak tanımlanması, geliştirilmesi, etkinliğinin denenmesi ve kullanılması gerekmektedir. Gelişmiş ülkelerdeki elektronik harp programları incelendiğinde, her ülkenin kendi coğrafyasına ve milli ihtiyaçlarına uygun projeleri milli firma ve kuruluşları tarafından geliştirilecek şekilde desteklediği ve bu projeler sonucunda elde edilen sistemlerin ise ülkelerin milli politika ve savunma stratejilerine uygun olduğu görülmektedir [2].

Yeni geliştirilen elektronik harp sistemleri, oldukça karmaşık ve yüksek teknoloji içeren sistemlerdir. Mevcut teknolojik imkanlarla, kullanıcının yabancılardan satın alıp kullandığı bu tür sistemlerin satıcı ülkelerce belirlenmiş bazı tehditlere karşı etkin olamaması mümkündür. Bu şekilde tedarik edilen elektronik harp sistemlerinin sınırlandırılmış kabiliyetlere sahip olması, bu kabiliyetlerin bir kriz ortamında daha da sınırlandırılması veya iş görmez hale getirilmesi mümkün olabilecektir [2]. Ülkemiz tarihinde de bu duruma örnek teşkil edecek olaylar Kıbrıs konusunda yaşanmıştır. Savunma Sanayii Müsteşarlığı'nın (SSM) internet sitesinde [3] bu durum şöyle ifade edilmiştir:

"1964 yılında Kıbrıs bunalımı sırasında, müttefik ülkelerden alınan savunma teçhizatının Türkiye'nin ulusal çıkarları doğrultusunda kullanılması ihtiyacı hasıl olmuş; ancak başta Amerika Birleşik Devletleri (ABD) olmak üzere, bazı müttefik ülkelerce çıkarılan engeller sebebiyle savunma ihtiyaçlarının karşılanmasında diğer ülkelere mutlak bağımlı hale gelinmesinin sakıncaları kuşkuya yer bırakmayacak şekilde gözler önüne serilmiştir."

Bu nedenle, ülkelerin ihtiyaç duyacakları gelişmiş elektronik harp sistemlerinin tam bağımsız olarak kullanılabilmesi ancak milli olarak geliştirilmeleri ile mümkün olacaktır [2].

Yukarıdaki elektronik harp tanımına bakıldığında, temelde istihbarat toplama ve silah sistemlerini etkisiz hale getirme işleri göze çarpar. Bu işler ise düşmana görünmemeyi gerekli kılmaktadır. Elektronik harp çerçevesinde, görmek ve görünmemek kavramları radarlar ve karıştırıcıları işaret eder. Bir elektronik harp

sistemi, radarları vasıtasıyla görmeye, karıştırıcıları vasıtasıyla da kendisini görmeye çalışan diğer sistemlere görünmemeye çalışır. Böyle bir sistemin başarısından anlaşılması gereken, radar sistemlerinin hedefleri algılamadaki etkinlikleri ve karşı tedbir sistemlerinin ise radarları karıştırmadaki etkinlikleridir. Bu nedenle, gerek radar gerekse hedef konumunda olan platformların başarılarını etkileyen temel faktörlerden biri, kısaca cisimlerin radar sinyallerini yansıtma kabiliyetlerinin ölçüsü diyebileceğimiz Radar Kesit Alan (RKA veya Radar Cross Section, RCS) kavramıdır.

Buraya kadar anlatılanlar ışığında özetle şunlar söylenebilir. Savaş kavramı biz istemesek bile var olacaktır; o halde bu olaydan en az düzeyde etkilenmek için çalışmak gerekir. Bu çalışmalarda yapılan harcamaları ise makul seviyelerde tutmak gereklidir. Savunma alanındaki teknolojik gelişmeleri yakından takip etmek ve özellikle elektronik harp konusunda milli teknolojiler üretmek zorunluluğu da unutulmamalıdır. Elektronik harp sistemlerinin başarılarını etkileyen unsurlardan biri de cisimlerin RCS ölçüleridir.

Bu tez çalışması, yukarıda belirtilen gerekleri de göz önünde bulundurarak, elektronik harp alanında önem arz eden doğru RCS hesaplamalarının kabul edilebilir miktarlarda zaman ve kaynak harcanarak yapılabilmesi için milli bir RCS kestirim programının üretilmesini amaçlamaktadır. Konunun genişliği ve hedeflenen RCS analizinin zorluğu ve karmaşıklığı da dikkate alındığında, böyle bir tezin oldukça kapsamlı olacağı görülmektedir. Öyle ki, bu tür programlar bazı şirketler tarafından kalabalık grupların uzun süren çalışmaları neticesinde geliştirilmiş bulunmakta ya da halen geliştirilmektedir. Bu programlar yoğun bir emeğin ürünü olup piyasada yüksek rakamlarla alıcı bulmaktadırlar. İlgilenenler için ekler bölümünde böyle örnekler sunulmuştur (EK 1). Yine ekler bölümünde, yapılan benzer tez çalışmalarından örnekler de yer almaktadır (EK 2). Verilen tez örneklerinin bu tezden başlıca farkları kapsamlarında ve kullanılan geliştirme ortamlarındadır.

Dolayısıyla, bu tez çalışması bahsi geçen biçimde bir program geliştirilmesini amaçlamakla birlikte, daha sonra üzerinde çalışılmayı gerekli kılan bir ürün ortaya koymaktadır. Bu nedenle tezde üretilen her türlü yapı, kod ve ürün için kolay anlaşılabilirlik asıl hedef olmuştur. Tasarım adımlarında kullanılan yapılar olası

değişiklik ve eklemelere müsaade edecek biçimde tasarlanmıştır. Geliştirme ortamı olarak kullanımı yaygın olan Microsoft Visual Studio .NET 2003 ve programlama dili olarak da yüksek seviye dillerden C++ kullanılmıştır. Tüm kod parçaları açık olup grafik kütüphanesi dışında her türlü kod bu tez kapsamında üretilmiştir.

Geliştirilen uygulamada öncelikli olarak kullanıcı görsel arayüzü kolaylığı hedef alınmıştır. Kullanıcının belirli bir dosya biçimi ile hedef cismin üç boyutlu şekil bilgilerini programa girdi olarak sağlaması mümkündür. Kullanıcının farklı radar tipleri ve pozisyonları, farklı hedef şekilleri gibi etkenlere bağlı analizleri farklı detay ve çözünürlükte yapabilmesine imkan sağlanmıştır. Sonuçlar ise tablo, çizim ve grafikler şeklinde kullanıcıya sunulmakta ve kayıt edilebilmelerine imkan verilmektedir.

Tezin bundan sonraki bölümleri şu şekildedir. Bölüm 2'de RCS kavramı açıklanacak ve hesaplama yöntemleri hakkında bilgi verilecektir. Bölüm 3'te tez sonucunda geliştirilen RCS kestirim programı ile ilgili başlıklar yer alacaktır. Sırasıyla, programda kullanılan yardımcı unsurlar ve kavramlar ile ilgili bilgiler, program tasarım bilgileri ve kullanım bilgileri sunulacaktır. Son olarak bölüm 4'te sonuç ve öneriler sunulacaktır.

Bu tez ile ilgili her türlü bilgiye www.ayanli.com/yukseklisans adresinden erişilebilir.

2. RADAR KESİT ALAN KAVRAMI

Giriş bölümünde de belirtildiği gibi, elektronik harp temelde istihbarat toplama ve silah sistemlerini etkisiz hale getirme işlerini kapsamaktadır. Bu nedenle elektronik harp, görünmeden hareket etmeyi zorunlu kılar. Görünmekten kasıt ise düşman silah sistemleri tarafından varlığınızın tespit edilmesidir.

Elektronik harp sistemleri pek çok farklı şekilde algılama yapabilir. Algılamada RF (radyo frekansı), IR (enfraruj), lazer, ses dalgaları gibi sinyaller kullanılabilir. Fakat bunlardan en yaygın olarak kullanılanı, pek çok farklı ortam koşulunda ve uzak mesafelerde güvenilir bir algılama sağlıyor olması nedeniyle RF sinyallerdir [4] [5].

Radarlar, vericilerinden çıkan elektromanyetik sinyallerinin alıcılarına erişen yansımalarıyla cisimleri algılarlar. Radar sinyalleri için hedef konumunda olan cisimlerin ise, algılanmamaları için radara en az seviyede sinyal yansıtmaları gereklidir. Bu durumda görünmemeyi başarmış olurlar. İşte cisimlerin bu sinyal yansıtma özellikleri onların radar imzalarını (radar signature) oluşturur. Cisimlerin radar imzalarının bilinmesiyle, bir radara hangi konumda yaklaşılırsa cismin algılanmayacağı da kestirilmiş olur. Aynı şekilde, bir cismin radar imzası biliniyorsa ve bu gerekenin üzerinde bir yansıtmayı işaret ediyorsa, o cisim için imza küçültme (signature reduction) yoluna gidilmesi ihtiyacı doğmuş olur. Bu nedenlerle cisimlerin radar imzaları büyük önem taşımaktadır.

Radar imzasının temelini cismin radar kesit alanı (radar cross section – RCS) oluşturur. RCS için farklı kaynaklarda şu tanımlar yapılmaktadır. [6]'da, bir cismin gelen elektromanyetik dalgayı ne mertebede yansıttığının tanımıdır denilmiştir. [7]'de ise, bir hedefin radar sinyallerini radar alıcısının yönünde yansıtabilme kabiliyetinin ölçüsü olarak tanımlanmıştır. Yine aynı kaynakta, birim katı açı (steradyan, sr) başına hedeften radar yönünde yayılan geri saçınım (backscatter) gücünün hedef tarafından alınan güç yoğunluğuna oranı olarak da tanım bulmuştur. Ayrıca RCS'nin, sinyal dalga boyunun cismin boyutlarına oranla küçük olduğu durumda, hedef cisimden alınan yansıma miktarının aynısını verebilecek bir iletken kürenin fiziksel kesit alanına eşit olacağı da [6]'da belirtilmiştir.

Bu tanımlarının yanı sıra, RCS için kavramsal diğer bir tanım da [7]'ye göre şöyledir. Yayılan tüm enerjinin hedef üzerine düşmeyeceği gerçeği

düşünüldüğünde, RCS en iyi biçimde üç faktörün çarpımı şeklinde ifade edilebilir. Bunlar izdüşümsel geometrik kesit alan, yansıtıcılık oranı ve yönlülük oranıdır. Burada izdüşüm alanı fiziksel olarak radar tarafından görülen hedef kesitini ifade etmektedir. Yansıtıcılık oranı, hedef tarafından saçılan gücün yine hedef tarafından alınan güç içindeki yüzdesidir [7]. Hedefin geri kalan gücü soğurduğu varsayılır. Yönlülük oranı ise, hedefin radar yönünde yansıttığı gücün, geri yansımanın tüm yönlerde eşit miktarda (isotropic, eşyönlü) olması durumunda radarda alınması beklenen güce oranıdır [7].

Belirtilenler bir eşitlik altında toplanacak olursa, σ radar kesit alanını göstermek üzere, şu denklem elde edilir:

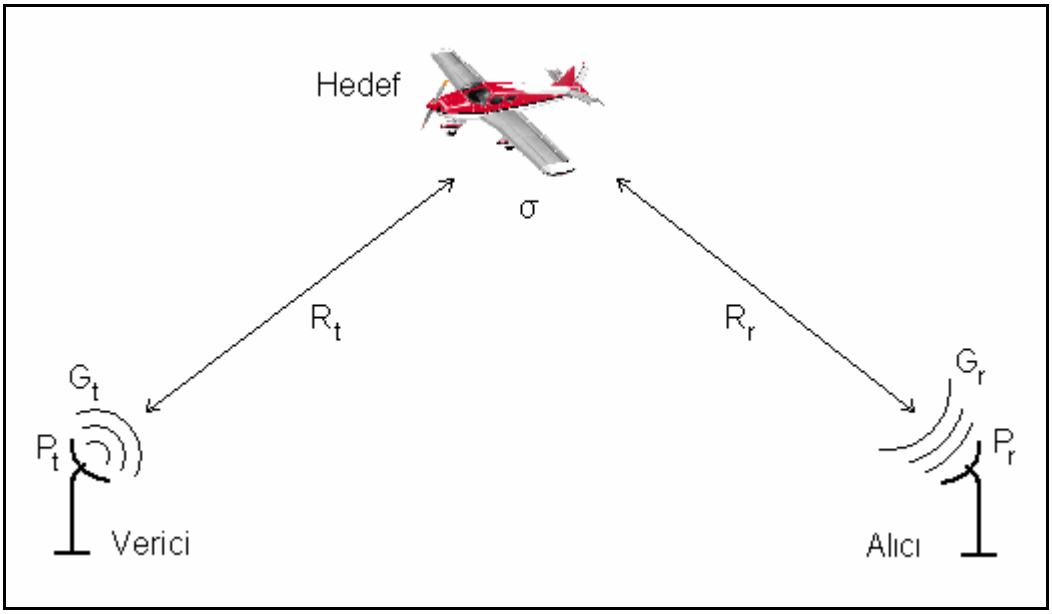
σ = izdüşümsel geometrik kesit alan × yansıtıcılık oranı × yönlülük oranı (2.1)

Diğer yandan, radar menzil denklemi (radar range equation) de RCS ile ilgili bilgiler içerir. Radar menzil denklemi Eş. 2.2'de verilmektedir. Burada alt indislerden t vericiyi (transmitter) ve r alıcıyı (receiver) göstermek üzere, P güç, R uzaklık, G ise anten kazancını simgeler. Radar sinyalinin dalga boyu λ ile, hedef cismin RCS'si ise yine σ ile gösterilmiştir. Şekil 2.1'de bu eşitliği açıklayan bir senaryo da mevcuttur.

$$\frac{P_r}{P_t} = \sigma\left(\frac{\lambda}{4\pi R_t R_r}\right)^2 \frac{G_t G_r}{4\pi} \tag{2.2}$$

Alıcı ve vericinin aynı yerde olmasıyla tek durağanlı (monostatik), farklı yerde olmasıyla ise çift durağanlı (bistatik) durum oluşur. Her iki durumda da yukarıdaki eşitlik geçerlidir, ancak bistatik durumda uzaklıklar için verilen alt indisler önemini yitirir ve iki uzaklık değeri birbirine eşit alınır.

Eş. 2.2'de güç birimi watt, uzaklık ve dalga boyu birimi metre alındığında, anten kazancı da birimsiz olduğundan, RCS birimi m^2 olacaktır. Ancak RCS değerleri büyük aralıklarda değişim gösterdiği için (Çizelge 2.1) genellikle logaritmik skala kullanılır ve dBsm (dB per square meter, metrekare başına dB) birimi ile ifade edilir. Bu değer ise $10\log_{10}(\sigma)$ ile hesaplanır.



Şekil 2.1. Örnek senaryo üzerinde radar menzil denklemi parametrelerinin gösterimi.

Çizelge 2.1. Bazı hedefler için tipik RCS değerleri [8].

Hedef	**RCS (m^2)**	**RCS (dBsm)**
Otomobiller	100	20
Yolcu uçakları	40	16
Savaş uçakları	2 – 6	3 – 7.78
Yetişkin insanlar	1	0
Kuşlar	0.01	-20
Böcekler	0.00001	-50
Gelişmiş hayalet uçaklar	0.000001	-60

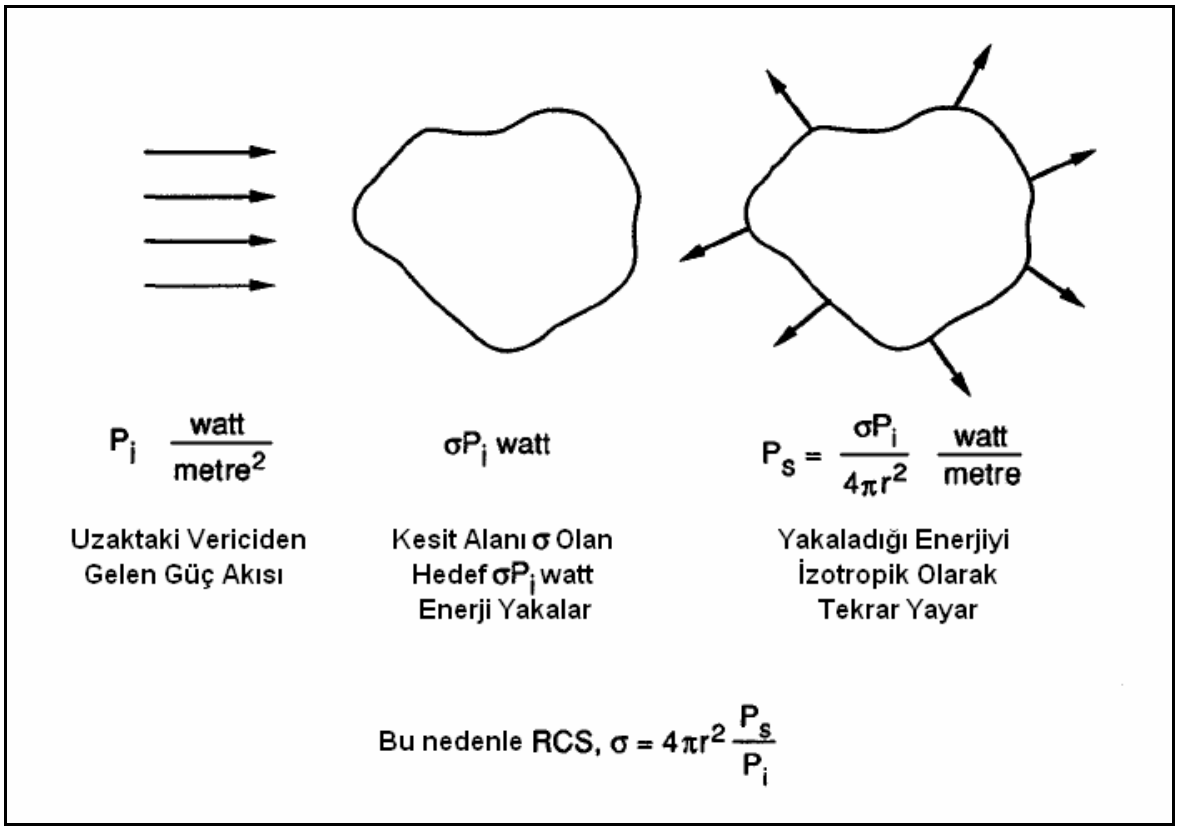
$$P_r = \left(\frac{P_t G_t}{4\pi R_t^{\,2}}\right)(\sigma)\left(\frac{1}{4\pi R_r^{\,2}}\right)\left(\frac{G_r \lambda^2}{4\pi}\right) \tag{2.3}$$

Eş. 2.3. radar menzil denkleminin tekrar düzenlenmiş halidir ve sembollerin anlamları Eş. 2.2 ile aynıdır. Bu dizilimi ile bakıldığında, ilk terim hedef mesafesindeki radar sinyali güç yoğunluğunu vermektedir. RCS bir alan ifadesi olduğundan, gelen sinyalin alana bağlı güç yoğunluğu ile çarpıldığında hedefin ne kadar güç almışçasına yansıtma yapacağı bulunur. İlk iki terim çarpımı hedeften yansımış gücü gösterir. Üçüncü terim de eklendiğinde, alıcı mesafesindeki radar güç yoğunluğu çıkar. Yine aynı mantıkla, antenin alan ifadesi diyebileceğimiz

açıklık (antenna aperture) değeriyle bu yoğunluk çarpıldığında alıcı tarafından algılan güç bulunmuş olur [5].

Buradan hareketle, RCS değerinin alıcı ve verici güçleri ya da mesafelerine bağlı olmadığı söylenebilir. Çünkü denklemde yer alan RCS dışındaki ifadeler, alıcı antenin açıklığının etkisi ile [9]'da belirtilen elektromanyetik dalganın boşlukta ters kare kanununa (inverse square law) göre yayılması ve güç yoğunluğunun azalmasının etkisini yansıtmaktadır. RCS'nin bu denklemdeki rolü sadece hedefin yansıttığı sinyal miktarını belirlemektir.

RCS için bir diğer tanım da [10]'da, hedeften belirli bir yönde saçılan birim katı açı başına düşen gücün, yine belirli bir yönden saçıcı üzerine gelen yüzey dalgasındaki birim alan başına düşen güce oranının 4π katı olarak yapılır [11]. Bu tanım yukarıda [7]'ye göre verilen tanım ile benzerdir. Bu tanımın karşılığı Şekil 2.2'de ve Eş. 2.4'te gösterilmiştir. [11]'de verilen bir başka eşitlik de, saçılan gücün ölçüldüğü uzaklığın sonsuza yaklaştığı durumdaki gelen ve saçılan elektrik alanlarının büyüklükleri ile RCS değerini ilişkilendirmiştir (Eş. 2.5).



Şekil 2.2. RCS için sezgisel tanım ([11], fig. 3.1'den değiştirilerek).

$$\sigma = 4\pi \frac{\frac{P_s}{4\pi}}{\frac{P_i}{4\pi R^2}} = 4\pi R^2 \frac{P_s}{P_i} \tag{2.4}$$

$$\sigma = \lim_{R\to\infty} 4\pi R^2 \frac{|E_s|^2}{|E_i|^2} \tag{2.5}$$

Yapılan tanımlamalar ışığında RCS'nin

- Vericinin ve alıcının hedefe göre konumlarına,
- Vericinin ve alıcının polarizasyonlarına,
- Hedefin şekline ve malzeme özelliklerine,
- Hedefin açısal duruşuna,
- Sinyalin dalga boyuna

bağlı olduğu söylenebilir [11].

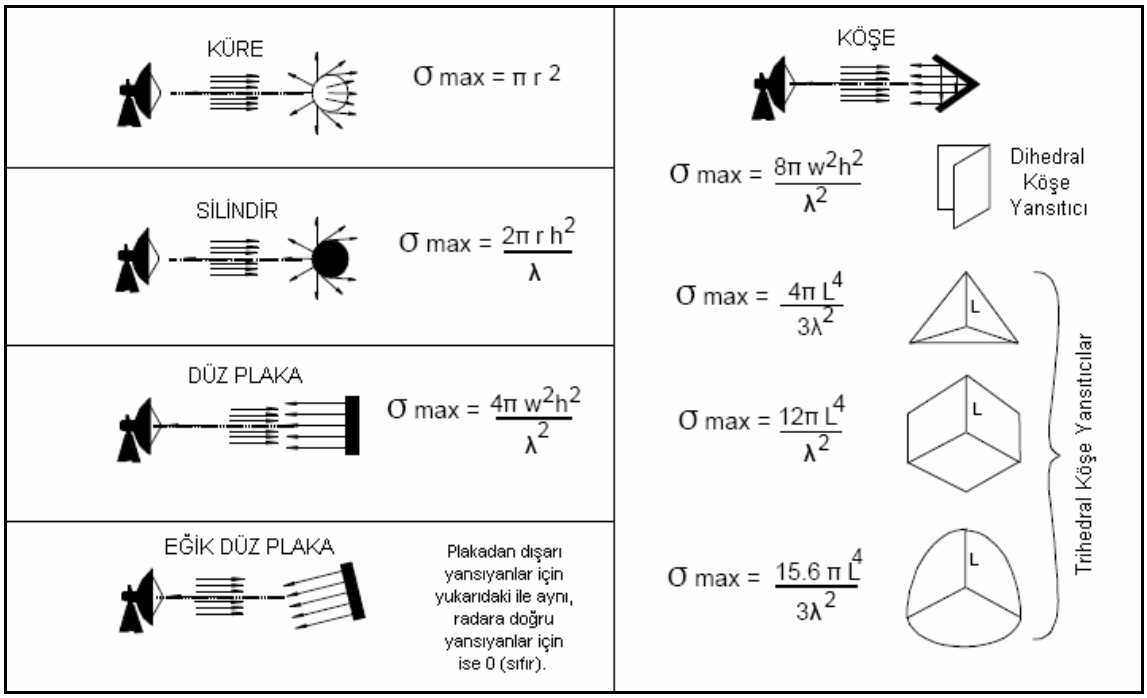
Bazı basit geometrik şekiller için RCS değeri analitik olarak hesaplanabilmektedir. Ancak bu hesaplamalar da çeşitli ön şartların gerçekleştiği durumlarda geçerli olabilmektedir. Örneğin bir düz plaka (flat plate) için RCS değeri, plaka normal vektörü yönünde aydınlatıldığı durumda, plakanın yüzey alanının karesi ve frekansın karesi ile orantılıdır. Eğer plaka, yüzey normali radar bakış açısının dışında kalacak şekilde yerleştirilmiş ise iki farklı durum oluşabilir. Plakanın bir çift kenarı radar doğrultusuna dik ise RCS değeri plakanın kenar uzunluğunun karesine bağlıdır ve frekanstan bağımsızdır. Aksi durumda ise, köşelerden gelen yansımalar alınacağından RCS değeri boyuttan bağımsız olur ve frekansın karesiyle ters orantılıdır [11]. Şekil 2.3'te bazı örnek şekiller ve RCS eşitlikleri verilmiştir. Ayrıca [12] s.369'daki Tablo R15'te daha geniş bir liste mevcuttur.

Daha karmaşık şekiller temelde küre, silindir ya da düzlem plaka gibi basit şekillerin bir araya gelmesiyle oluşmaktadır. Ancak bu karmaşık şekillerde RCS değerinin hesaplanmasını daha da zor hale getiren farklı davranışlar ortaya çıkmaktadır. Bu davranışlara sebep olan mekanizmalar [11]'de bölüm 6.1'de önemlerine göre şu şekilde sıralanmıştır:

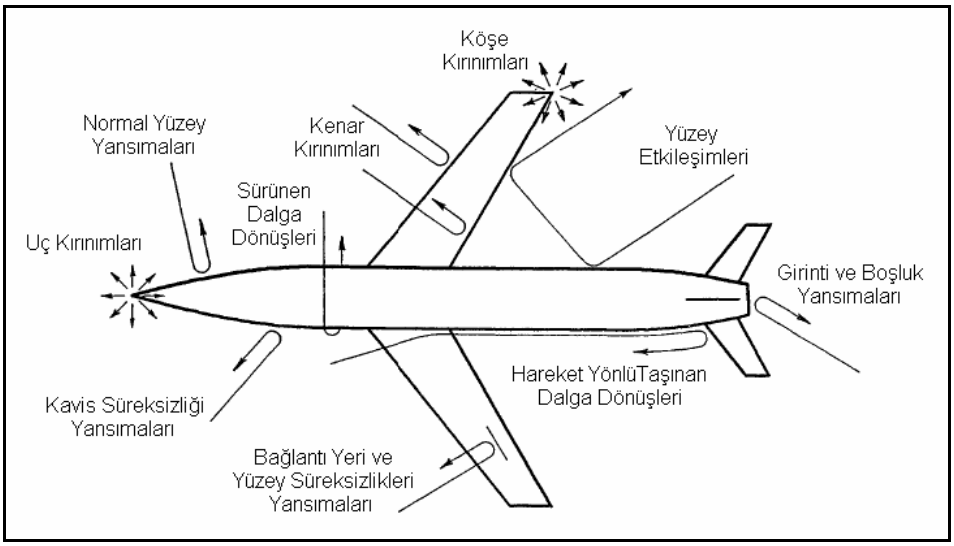
- Girinti ve cepler (reentrant structures)
- Normal yüzey yansımaları (specular scattering)

- Hareket yönlü taşınan dalgalar (traveling wave echoes)
- Kenar ve köşe kırınımları (edge and vertex diffraction)
- Sürünen dalgalar (creeping waves)
- Yüzey etkileşimleri (surface interactions)
- Yüzey süreksizlikleri (surface discontinuities)

Bu mekanizmaların nasıl oluşabildiği Şekil 2.4'te görsel olarak sunulmuştur.

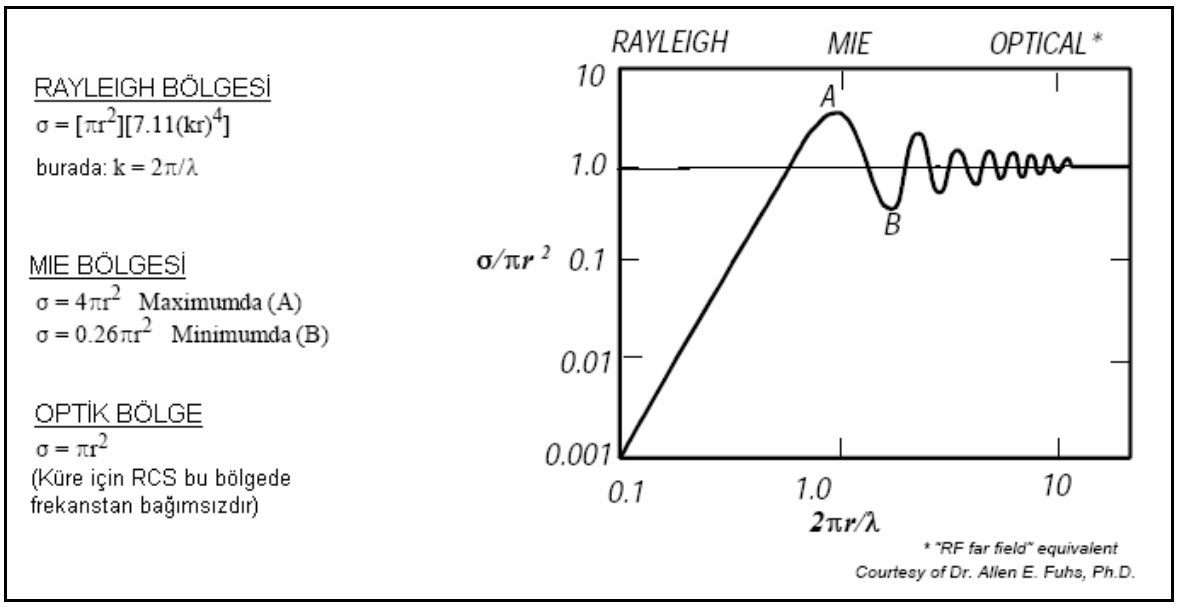


Şekil 2.3. Basit şekiller için RCS eşitlikleri ([7], fig. 3'ten değiştirilerek).



Şekil 2.4. Karmaşık şekillerde yansıma mekanizmaları ([11], fig. 6.1, p.227'den değiştirilerek).

Yukarıda bahsedilen tarzda karmaşık etkiler, frekansa bağlı olarak da değişkenlik gösterebilmektedir. Örneğin yüzeye teğet olarak hareket eden sürünen dalgalar, cismin boyutu ile sinyal dalga boyu arasındaki orana göre, geri dönüşlerinde normal yüzey yansımalarını arttırıcı ya da azaltıcı etki yapabilirler. Bu nedenle, frekansa bağlı olarak farklı saçınım rejimlerinden bahsedilir. Bunlar Rayleigh bölgesi, rezonans bölgesi ve optik bölge diye isimlendirilmiştir. Rayleigh bölgesi, $ka<1$ (a: cisim boyutu, k: dalga numarası, $k=2\pi/\lambda$) olan bölgedir ve saçınımda uyarılmış çift kutuplu momentler (induced dipole moments) baskındır. Rezonans bölgesi Mie adıyla da anılır. Burada yüzey yansımaları yanında sürünen dalgalar da saçınımda rol oynar ve $1<ka<10$ olmaktadır. Yüksek frekans bölgesinde, $10<ka$ olduğunda yüzey yansımaları en baskın hale gelir. Bu bölgeye de Optik bölge denilmiştir [11]. Şekil 2.5'te iletken küre örneği için bu bölgeler gösterilmektedir.



Şekil 2.5. İletken küre için RCS bölgeleri ve eşitlikleri ([7], fig.7'den değiştirilerek).

Optik bölgede frekans değerleri yüksektir ve bağımsız saçınım merkezleri kavramı geçerlilik kazanır [11]. Diğer bir deyişle, bu bölgede kolektif etkileşimler oldukça küçük olduğundan, cisimlerin birden fazla bağımsız saçınım merkezinin birleşiminden oluştuğu kabul edilebilir [13].

Cisimlerin gerçek RCS değerlerini bulmak için sıkça kullanılan yöntemlerden biri, cismin küçük oranda bir maketi üzerinde laboratuar ortamında ölçümlerin alınmasıdır. Bu işlemde kullanılan sinyal dalga boyu da oranlı olarak değiştirilir ve deneyde bulunan sonuçlar bazı hesaplamalar sonrasında gerçek değerlere çevrilir. Çevresel etkileri ortadan kaldırmak için de yansıtmasız yüzeylerden oluşan test odaları oluşturulur. Ayrıca cismin sabitlendiği kaideler de yansımaya etki etmeyecek şekillerde ve uygun malzemelerden yapılırlar. Tüm bunlar yoğun emek isteyen çalışmalardır. Bunların yanında, sadece hesaplamalara dayanan kestirim yöntemleri daha az çalışma gerektirir. Ancak bu hesaplamalar da yukarıda bahsedilen çeşitli sebeplerle çok karmaşık olabilmektedir.

Gerçek dünyada kullanılan radar sinyalleri ve hedef konumundaki cisimler ele alındığında, dalga boyunun cismin boyutları yanında küçük kaldığı görülür. Örneğin bir savaş uçağı için ortalama boyutlar 15 – 20 metredir. Kullanılan sinyal frekansı 1 GHz olduğunda dalga boyu 0.3 metre olur ve $ka=2\pi 15/0.3=100\pi$ değeri optik bölgeyi gösterir. 10 GHz sinyal ile $ka>10$ olması için $a>0.15\pi$ olması yeterli olacaktır. Buradan hareketle, RCS değeri kestirimi için, yüksek frekans bölgesi varsayımlarına dayanan ve daha basit hesaplamalar gerektiren teknikler geliştirilmiştir. [11], bölüm 5'te bu tekniklerden bazılarını şu başlıklarla vermektedir:

- Geometrik optik (geometric optics, GO)
- Fiziksel optik (physical optics, PO)
- Geometrik kırınım teorisi (geometric theory of diffraction, GTD)
- Fiziksel kırınım teorisi (physical theory of diffraction, PTD)
- Eş akımlar metodu (method of equivalent currents, MEC)
- Tekdüze kavuşmazlık teorisi (uniform asymptotic theory, UAT)
- Artan uzunluk kırınım katsayısı (incremental length diffraction coefficient, ILDC)

Adı geçen tekniklerden her birinin farklı avantajları olabildiği gibi yetersiz kaldıkları konular da olabilir. Örneğin geometrik optik tekniği en basit hesaplamalara sahiptir, ancak sonsuz frekans ve sıfır dalga boyu (hedef cismin boyutlarının çok büyük olması durumu) varsayımından yola çıktığı için sonsuz RCS değerlerinin hesaplanmasına neden olabilmektedir. Yoğun integral hesaplamaları gerektiren

teknikler ise daha doğru sonuçlar vermelerine karşın, hem zaman hem de bilgisayar kaynakları yönünden yetersiz kalabilmektedir ([5], bölüm II.E). Fiziksel optik de oldukça başarılı sonuçlar veren ancak yansıma açısının artmasıyla doğruluğu azalan tekniklerdendir [11]. Bu ve benzer nedenlerle, birbirini tamamlayıcı şekilde birden fazla tekniğin bir arada kullanılması da sık karşılaşılan yöntemlerdendir.

Geometrik optik, ışık kavramını temel olarak almaktadır. Optik kuramının kırınım (ışının ortam değiştirmeksizin cisimlerin kenar ya da köşelerini yalayarak geçmesi durumunda yön değiştirmesi, diffraction) kuralını yok sayarak, ışınların sadece doğrusal izlerde hareket ettiğini kabul eden bir uygulamasıdır. Bu varsayım, ışık dalga boyu yeterince küçük olduğunda, dolayısıyla ışının izlediği yolun sadece yansıma (ışının bir engel tarafından yönünün değiştirilerek geldiği ortama döndürülmesi, reflection) ve kırılma (ışının ortam değiştirirken yön değiştirmesi, refraction) kuralları ile açıklanabileceği durumda geçerlidir [14].

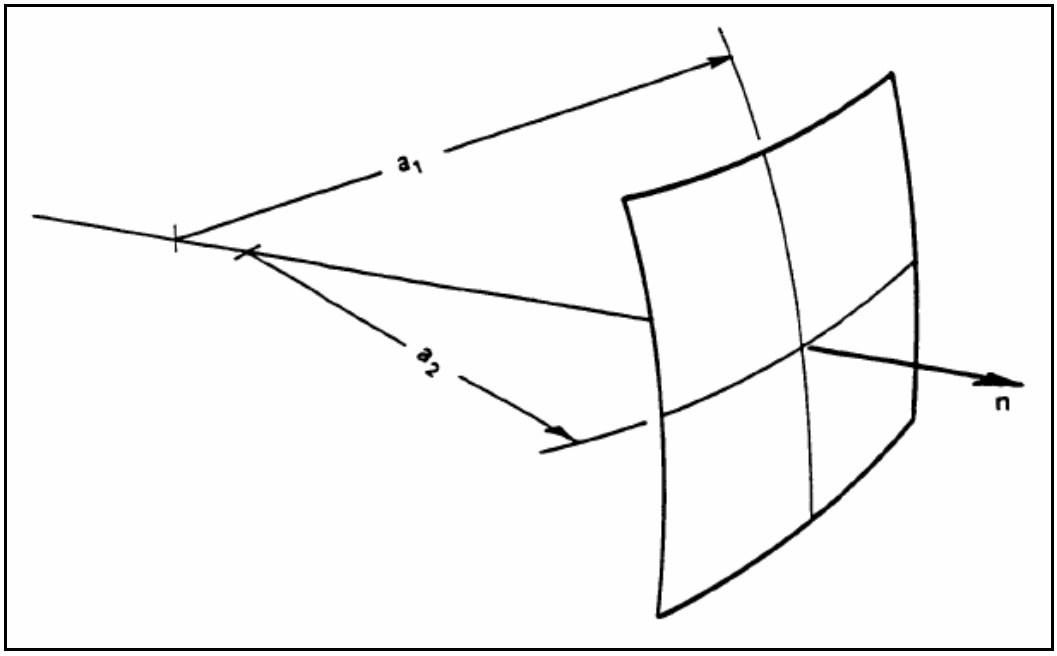
GO teorisinde RCS, yüzey yansıma noktasındaki kavisin bölgesel yarıçapını (local radius of curvature) içeren çok basit formüllerle ifade edilir. Ancak bu formül, yüzey kavisinin yarıçapı sonsuz olduğunda çalışmaz. Örneğin düzlem plaka ya da silindir için yüzeye dik bakış açılarında bu durum gözlenir. Burada genellikle fiziksel optik gibi diğer yöntemlere başvurulur [11].

GO, dalga boyunun çok küçük olmasına izin verilen bir ışın takibi (ray tracing) işlemidir. Burada enerjinin ışın denilen çok ince tüpler boyunca yayıldığı varsayılır. Bu ışın, farklı kırılma indislerine sahip iki ortamı ayıran bir yüzeye çarptığında Snell yasasına ([15]) göre yansıyan ve kırılan (ikinci ortama aktarılan) ışınlar olarak ikiye ayrılır. Bu iki ışının büyüklüklerinin normalize edilmesiyle elde edilen katsayılar kullanılarak yapılan bazı hesaplamalar ([11], bölüm 5.1 ve 5.2) sonucunda Eş. 2.6 elde edilir. Burada a_1 ve a_2, yansıma noktasındaki ana eksen yüzey kavisi yarıçaplarını gösterirken σ yine RCS'dir. Eşitliği tanımlamak üzere [16]'da aslı sunulan şekil aşağıda verilmiştir (Şekil 2.6).

$$\sigma = \pi a_1 a_2 \tag{2.6}$$

Bulunan eşitlikte RCS'nin frekanstan bağımsız olduğu görülür, ancak bu bir yaklaştırma değeridir ve dalga boyu çok küçükken yani cisimler elektriksel olarak

büyükken geçerlidir. Kullanılan kavis yarıçapları kesinlikle yansıma noktasında hesaplanmalıdır. Ayrıca bu eşitlik monostatik ve bistatik radar durumları için de geçerlidir [11].



Şekil 2.6. Çift kavisli yüzey için ana eksen yüzey kavisi yarıçapları ([17], fig. 2.9)

3. RADAR KESİT ALAN KESTİRİM PROGRAMI

Tezin amacı doğrultusunda, radar kesit alan kestirimi için grafik kullanıcı arayüzlü bir bilgisayar programı geliştirilmiştir. Radar Cross Section Predictor (RCSP veya Radar Kesit Alan Kestirici, RKAK) adı verilen bu program, kullanıcıların muhtemel istekleri ve ihtiyaçları göz önünde tutularak tasarlanmıştır. RCSP programı ile farklı şekillerdeki cisimler için RCS kestirimleri yapılabilmektedir. Kullanıcı, farklı değişkenlere bağlı olarak yapabileceği analizlerin sonuçlarını görsel olarak alabilmekte ve bunları kaydedebilmektedir.

Bu bölümde, öncelikle RCSP programında kullanılan yardımcı unsurlardan bahsedilecektir. Bu unsurlar, girdi dosyası biçimi olarak kullanılan stereolithography (STL) formatı, görsel ekranlarda kullanılan OpenGL grafik kütüphanesi (open graphics library), geometrik optik teorisiyle birlikte kullanılan ışın takibi algoritması ve hesaplama ile çizim aşamalarında faydalanılan üç boyutlu uzay matematik denklemleridir. Daha sonra, programın tasarım bilgileri, kod yapısı ve kullanılan hesaplama yöntemleri sunulacaktır. Son olarak, programın kullanımı örnek ekran görüntüleri ile açıklanacaktır.

3.1. Yardımcı Unsurlar

RCSP programı tasarımı tamamen özgün olup tez kapsamında yapılmıştır. Programda kullanılan her türlü kod parçası ve veri yapısı da özgün olarak tasarlanarak kodlanmıştır. Bundan amaçlanan, tüm koda hakimiyeti sağlamak, her türlü değişikliğe kolaylık ve imkan tanımak ve katkısı öngörülemeyen olası yabancı kodları en aza indirmektir. Ancak bazı durumlarda, var olan açık kaynaklı kodları kullanmak kaçınılmaz olmuştur. Örneğin her türlü üç boyutlu şeklin çiziminde, böyle bir kütüphane olan OpenGL kullanılmıştır. Programda, kullanıcıdan alınacak şekil bilgileri için genel kabul gören ve yaygın biçimde kullanılan formatlardan biri seçilmiştir. RCS kestirim hesaplamaları sırasında, mevcut teorilerden ve algoritmalardan faydalanılmıştır. İhtiyaç duyuldukça üç boyutlu uzay matematiği denklemlerine başvurulmuştur. Tüm bu konularda tasarıma faydalı olacak şekilde kullanılmış olan bilgiler yardımcı unsurlar olarak nitelendirilmiş ve aşağıdaki alt başlıklarda sunulmuştur.

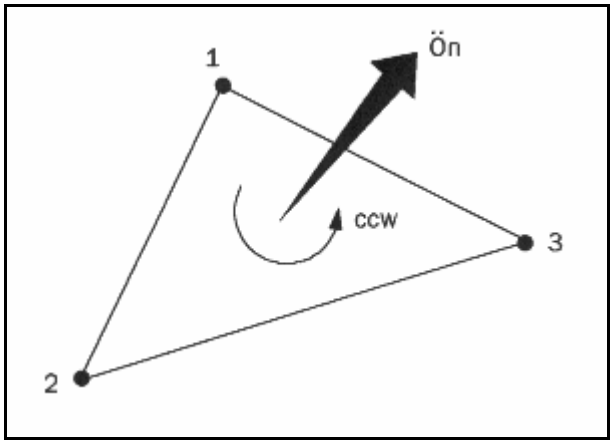
3.1.1. Stereolithography (STL) formatı

RCSP programının amacı herhangi bir cismin RCS değerini kestirebilmektir. O halde öncelikle programa ilgilenilen cismin şekil bilgileri aktarılabilmelidir. Bu bilgiler çok farklı biçimlerde olabilir. Kullanıcı, hedef cismin şeklini tanımlamak için bir bilgisayar destekli tasarım (computer aided design – CAD) programı kullanmak durumundadır. CAD programı olarak piyasada var olan programlardan (Solidworks, Autodesk 3ds Max, Autodesk Mechanical Desktop, vs.) herhangi biri seçilebilir. Dikkat edilecek husus, bu program aracılığı ile hazırlanan bilgilerin RCSP programının girdi formatında kaydedilebilmesi ya da bu formata çevrilebilmesidir. Bu sayede, hazırlanan bilgiler RCS kestiriminde kullanılabilecektir.

Temel olarak CAD programı çıktı formatları benzer özellikler taşırlar. Fakat kullanılacakları alana göre içerdikleri bilgi miktarı değişebilir. Örneğin 3ds formatı daha sıklıkla oyun ve internet alanlarında karşımıza çıkar ve bu nedenle yüzey kaplaması ya da ışıklandırma gibi görsel ayrıntılar içerir. Bu formatlar, yüzeyleri küçük parçalar şeklinde tanımlarlar ve bunları köşe (vertex), nokta, çizgi, kenar, ağ (mesh), yüzeycik (facet) gibi temel yapılarla belirtirler.

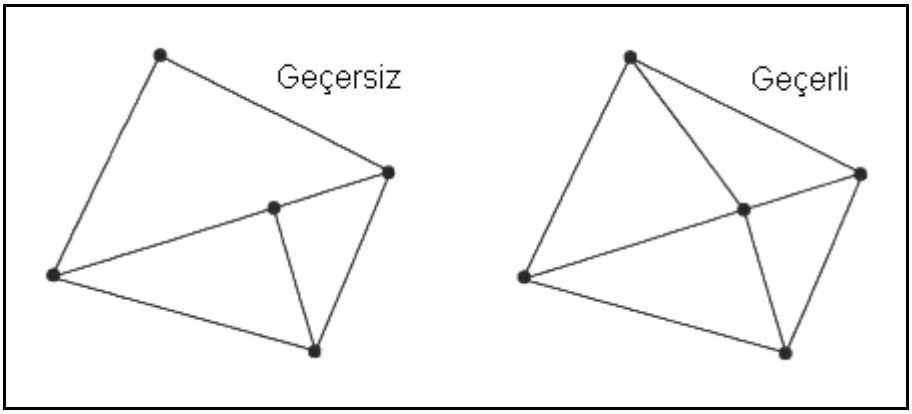
Mevcut formatlar arasında neredeyse en basit olanı stereolithography (STL) formatıdır. Hızlı ilkörnekleme makinelerinde (rapid prototyping machines – RPM) ve bilgisayar destekli üretimde (computer aided manufacturing – CAM) kullanılır. Kullanım yeri ve amacı itibariyle sadece cismin yüzeyini tanımlaması yeterli olduğundan fazla bilgi içermez. Binary ve ASCII olmak üzere iki farklı biçimi vardır [18].

STL formatı, üç boyutlu yüzey geometrisinin üçgensel gösterimine dayanır. Yüzeyler, yüzeycik adı verilen küçük üçgenlere ayrılarak tanımlanırlar. STL dosyası, bu üçgenlerin köşelerinin koordinatları ile yüzeyciğin normal vektörünü içerir. Her yüzeycik için üç köşe noktasının koordinatları, üç boyutlu kartezyen koordinat sistemine göre verilir. Bu köşelerin veriliş sırası, sağ el kuralına göre yüzeyciğin normal vektörünün yönünü belirler (Şekil 3.1). Bu yön, yüzeyin baktığı yön ya da ön yüz olarak tabir edilir. Normal vektörü de üç boyutlu kartezyen koordinat sistemine göre verilir. Toplam olarak bir yüzeycik için ((3 köşe koordinatı + 1 normal vektörü) × 3 kartezyen sistemi elemanı) = 12 sayı verilmiş olur [19].



Şekil 3.1. Yüzeycik için köşeler ve normal vektörü ([19], fig. 1).

Köşe-köşeye kuralına göre, bir üçgen yüzeycik komşu üçgen yüzeycik ile iki köşe paylaşmak zorundadır. Bir başka deyişle, bir yüzeycik köşesi başka bir yüzeycik kenarı üzerinde olamaz. Bu kural da Şekil 3.2'de gösterilmiştir .

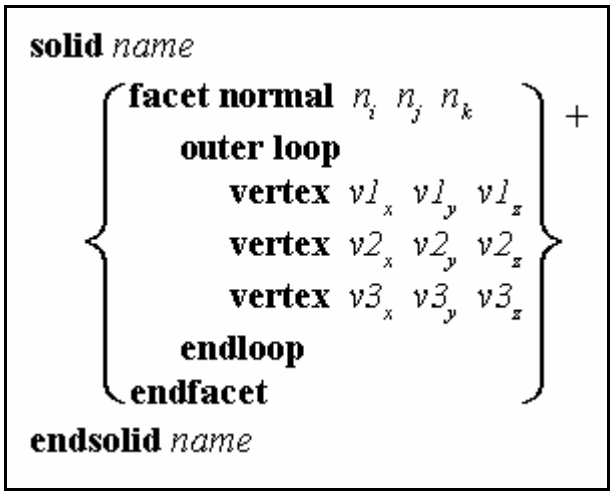


Şekil 3.2. Yüzeycik için köşe-köşeye kuralı ([19], fig. 2).

STL formatının iki farklı biçiminden biri ASCII formatıdır. Bu biçimde, yüzeycikler ve koordinat verileri kolay anlaşılır şekilde sıralanmıştır. ASCII formatı, herhangi bir metin işleme programı ile açılıp okunabilir. Bu sayede şekil üzerinde ufak değişiklikler yapmak daha kolay olabilmektedir. Dosya içinde, verilmek istenen her veri öncesinde uygun bir anahtar kelime kullanılır. Bu kullanım nedeniyle dosya boyutu içerdiği yüzeycik verisine oranla oldukça büyük olabilmektedir. ASCII formatı, asıl olarak CAD arayüzlerini test etme amacıyla geliştirilmiş bir formattır.

ASCII formatında bir dosyanın içerdiği şekil bilgileri, "solid" ve "endsolid" anahtar kelimeleri arasında yer alır. İsteğe bağlı olarak bu anahtar kelimelerden sonra

şekil adı tanımlayıcı olarak yazılabilir. Arada ise, birbirini takip eder biçimde her bir yüzeyciğin bilgileri yer alır. Burada son yüzeyciği "endsolid" kelimesi belirleyeceği için yüzeycik sayısında bir sınırlama yoktur. Bir yüzeycik için 7 satırlık alan kullanılır. Bunlardan ilki, "facet normal" anahtar kelimesi ve ardından sıralanmış kartezyen sistemi bileşenlerini gösteren üç sayıdan oluşur. Verilen normal vektörü birim vektördür ve sıralanan üç sayı boyu 1 olacak şekilde bir vektör tanımlamaktadır. İkinci satır "outer loop" anahtar kelimesidir. Bunu takip eden üç satırda, üçgen yüzeycik için köşe olan noktaların koordinatları verilir. Bu satırların içeriği de "vertex" anahtar kelimesinin ardından üç sayı gelecek şekildedir. Sonraki satırlardaki "endloop" ve "endfacet" anahtar kelimeleriyle bir yüzeycik için ayrılan alan sonlanır. Burada verilen her türlü koordinat verisi, tek duyarlıklı yüzen nokta (single precision floating point) sayı formatındadır. ASCII formatı Şekil 3.3'te özetlenmiştir. Şekildeki $\{\ldots\}^+$ ifadesi ardışık olarak tekrar edebilecek bölgeyi simgelemektedir.



Şekil 3.3. ASCII format STL dosyası içeriği [19].

Örnek bir cisim için oluşturulmuş ASCII formatındaki STL dosyası EK 3'te sunulmuştur.

STL formatının diğer biçimi olan ikili biçim (binary format), küçük dosya boyutu nedeniyle daha yaygın kullanılır. Ancak herhangi bir programla içeriği hakkında bilgi edinmek ya da değişiklik yapmak mümkün değildir, çünkü sadece formata uygun olarak gruplandığında anlam kazanan sayılar yığını şeklindedir. Binary formatta farklı olarak bir başlık kısmı bulunur. 80 byte uzunluğundaki bu alanın

kullanımı kesin kurallara bağlanmamıştır. Ardından gelen 4 byte ile (IEEE long integer formatına göre) yüzeycik sayısı verilir. ASCII formattan farklı olarak yüzeycik sayısı sınırlıdır ve yüzeycik bilgilerinden önce verilmiştir. Buradan sonra, verilen yüzeycik sayısı kadar tekrar eden ve her biri 50 byte uzunluğunda olan yüzeycik bilgisi alanları gelir. Her bir yüzeycik için toplam 12 koordinat bileşeni sıralanır. Bunların da her biri 4 byte uzunluğunda olup tek duyarlıklı yüzen nokta (single precision floating point) sayı formatındadır. Ardından gelen son 2 byte ise öznitelik alanı olarak ayrılan bölgenin kaç byte olduğunu gösterir. Ancak bu alan genellikle 0 değerinde bırakılmıştır. Binary formatı da Şekil 3.4'te özetlenmiştir. Şekildeki $\{\ldots\}^+$ ifadesi yine ardışık olarak tekrar edebilecek bölgeyi simgelemektedir.

Bytes	Data type	Description	
80	ASCII	Header. No data significance.	
4	unsigned long integer	Number of facets in file	
4	float	*i* for normal	+
4	float	*j*	
4	float	*k*	
4	float	*x* for vertex 1	
4	float	*y*	
4	float	*z*	
4	float	*x* for vertex 2	
4	float	*y*	
4	float	*z*	
4	float	*x* for vertex 3	
4	float	*y*	
4	float	*z*	
2	unsigned integer	Attribute byte count	

Şekil 3.4. Binary format STL dosyası içeriği [19].

Tezde üretilen RCSP programı, girdi olarak ASCII formatta STL dosyasını kabul etmektedir. Ancak programın modüler tasarımı sayesinde, eklenecek birkaç kod parçası ile diğer dosya formatlarının da girdi olarak tanıtılmaları mümkündür.

3.1.2. OpenGL grafik kütüphanesi

Tez kapsamında üretilen RCSP programı, görsel arayüzü açısından kullanıcının ihtiyaçlarına yanıt vermesi beklenen bir programdır. Bu nedenle arayüz, kullanım kolaylığı yanında görsel yönden de tatmin edici olacak şekilde tasarlanmıştır. Arayüz elemanları, kod geliştirme ortamı olarak kullanılan Microsoft Visual Studio .NET platformunun sunduğu standart elemanlardan seçilmiştir. Ancak bu platformdaki grafik kütüphanesi üç boyutlu çizimler için yetersiz kaldığından başka grafik kütüphanesi kullanma zorunluluğu doğmuştur.

Grafik kütüphanelerinin temel işlevi, bilgisayar yazılımları ile bilgisayarın grafik donanımı arasındaki köprüyü kurmaktır. Bilgisayarlarda, farklı firmalar tarafından üretilmiş farklı grafik donanımları bulunabilir. Bu donanımlar kendilerine has sürücü programları aracılığı ile işletim sistemleriyle standart olarak haberleşebilmektedir. Grafik kütüphaneleri, bu sürücü programları kullanarak grafik donanımına erişimi sağlayan ve bu işleri bir üst seviye kullanıcı için platformdan bağımsız hale getiren yapılardır. Örneğin bir grafik kütüphanesi, ekran üzerinde bir nokta çizme işlemini, her işletim sistemi ya da her grafik donanımı için standart olarak kullanılabilecek bir fonksiyonla halledilebilir hale getirir. Bu sayede üst seviye kullanıcının platformdan bağımsız olarak kullanacağı tek bir fonksiyon sağlamış olur. Arka planda ise, o fonksiyon donanım ve yazılım bilgilerine göre karar vererek nokta çizme işlemini çok farklı biçimlerde gerçekleştiriyor olabilir. Bu açıdan, grafik kütüphaneleri üst seviye kullanıcı için standart fonksiyonlar sağlayan ve donanımla köprü oluşturan yapılar olarak görülebilir.

Üç boyutlu dünyada yarışan grafik kütüphanelerinden ikisi öne çıkmaktadır. Bunlardan biri Microsoft destekli olan Direct3D, diğeri ise OpenGL (open graphics library)'dir. Tezde OpenGL kütüphanesi kullanılmıştır. Tezdeki sınırlı kullanımlarının yanında bu kütüphanelerle yapılabilecekler çok daha fazladır. Bilgisayar oyunlarındaki artan detaylar ve animasyon filmlerdeki başarılı sahneler bunlara örnek olabilir. Bu bölümde OpenGL kütüphanesinin tezde kullanılmış

özellikleri yanı sıra, gerekli görüldüğünde önemli başka ayrıntılar da bütünlüğü sağlamak adına belirtilecektir.

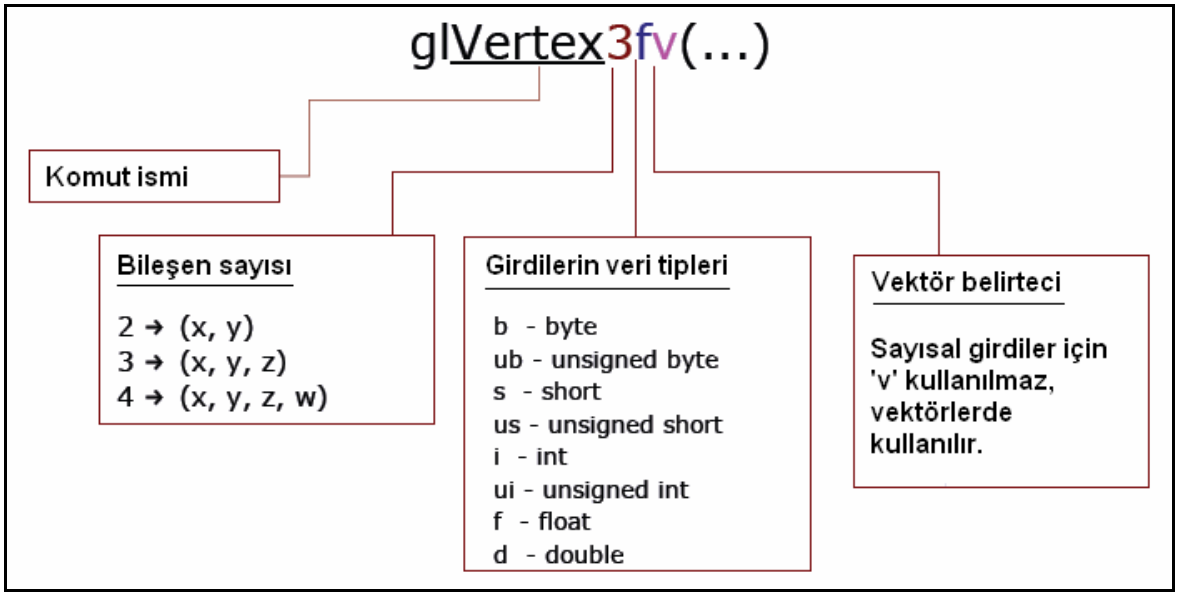
RCSP programında, kullanıcının bir hedef cismi girdi olarak vermesiyle işlem başlar. Burada kullanıcının istediği cismi yüklediğinden emin olması için gerekli olan ilk görsel adım yer alır ve cisim farklı açılardan çizilerek kullanıcıya sunulur. Girilen cismin gösterilmesinin ardından, kullanıcı RCS analizi için gerekli pozisyon ayarlamalarını yapmak isteyecektir. Burada da grafik işlemlerindeki ikinci adım olarak görüntünün hareket ettirilmesi yer alır. Son olarak da analiz sonuçlarının aktarılması için görüntüde renklendirme ihtiyacı doğar. Bu üç adım, tez kapsamında OpenGL kütüphanesinden kullanılmış olan fonksiyonların özetidir.

OpenGL arayüzünde 150 civarında fonksiyon bulunur. Bu fonksiyonlar ile temel işlemler yerine getirilebilir. Ancak bu kütüphane üst seviyede komutlar sağlamaz. Örneğin bir küp ya da küre çizmek için bile fonksiyonlar bulunmaz. Bunun yerine, kullanıcıya ilkel geometrik şekilleri (nokta, doğru ve çokgenler) çizebileceği arayüzler sağlanmıştır. Ayrıca OpenGL donanımdan bağımsız arayüzler sağlamak adına threading (iş parçalama) ya da windowing (pencereleme) gibi mekanizmalar da sunmaz. Bu ve benzeri ek özellikler, OpenGL kütüphanesi üzerine inşa edilen OpenGL Utility Library (GLU) ve Utility Toolkit (GLUT) kütüphaneleri ile kullanıcılara sunulmuştur ([20], /chapter01.html). OpenGL ve GLU fonksiyonları için kısa açıklamalar [21]'de bulunabilir.

OpenGL fonksiyonları genel bir biçime uygun olarak isimlendirilmişlerdir (Şekil 3.5). Bu biçimde fonksiyon ismi ve alacağı değişken sayısı ile tipleri belirtilir. Bazı fonksiyonların birden fazla değişken tipi ile işlem yapabilen uyarlamaları vardır ve uygun son ekler kullanılarak bunlar birbirinden ayrılabilmektedir. Bu isimlendirme sayesinde kullanıcı fonksiyon hakkında ilk bakışta fikir sahibi olabilmektedir. Kullanılan son ekler ve belirttikleri veri tipleri Çizelge 3.1'de verilmiştir.

OpenGL çalışma prensibi açısından bir durum makinesidir (state machine). Fonksiyonları aracılığı ile bu makinenin durumları değiştirilir ve bir dahaki değişime kadar o durum korunur. Örneğin çizim yapılan renk bir durum değişkenidir. Mavi değerinin bir kez verilmesinin ardından başka bir değer verilene kadar tüm çizimlerde mavi olarak kullanılır. Durum değişkenleri renk örneğindeki

gibi farklı değerler almak zorunda değillerdir. Bazıları için geçerli ve geçersiz şeklinde iki konum mevcuttur. Bu değişkenler “glEnable(…)” ve “glDisable(…)” komutları ile kontrol edilirler. Örneğin ışıklandırma bu şekilde devreye alınır ya da etkisiz kılınır. Durum değişkenlerinin çoğunun değerini “glGet…()” ya da “glIs…()” fonksiyonları ile sorgulamak mümkündür.



Şekil 3.5. OpenGL fonksiyonlarının isimlendirme biçimi ([22]’den değiştirilerek).

Çizelge 3.1. OpenGL fonksiyon son ekleri ve veri tipleri ([20], /chapter01.html, table 1-1’den değiştirilerek).

Son ek	Veri tipi	C dilindeki karşılığı	OpenGL tip tanımı
b	8-bit tamsayı	signed char	GLbyte
s	16-bit tamsayı	short	GLshort
i	32-bit tamsayı	int / long	GLint, GLsizei
f	32-bit yüzen nokta	float	GLfloat, GLclampf
d	64-bit yüzen nokta	double	GLdouble, GLclampd
ub	8-bit işaretsiz tamsayı	unsigned char	GLubyte, GLboolean
us	16-bit işaretsiz tamsayı	unsigned short	GLushort
ui	32-bit işaretsiz tamsayı	unsigned int / unsigned long	GLuint, GLenum, GLbitfield

Grafik kütüphanesi, grafik donanımı ile çalışırken kendisine uygun bir aygıt bağlamı (device context) ve resmetme bağlamı (rendering context) kullanır. Bu bağlamlar, başka programlar tarafından paylaşılıyor olabilir. En basit yaklaşımla, ekran üzerindeki pikselleri birçok program paylaştığından bir programın piksel üzerinde sürekli işlem yapması mümkün değildir. Bir program, ekrana istediği

görüntüyü çizdikten sonra da ihtiyaç duyulduğunda o görüntünün ekranda olmasından sorumludur. Örneğin, çalışmakta olan bir internet tarayıcısının penceresi ekranda iken onun üzerine gelecek şekilde bir metin işlemcisi de çalıştırılabilir. Bu durumda metin işlemcisi ekran kaynağını kullanmaya başlar. Bu pencere kapatıldığında ise internet tarayıcısı tekrar aktif olacaktır. Ancak tarayıcı penceresinin kullandığı pikseller metin işlemcisi tarafından kullanılmış ve değiştirilmiş olabilir. Bu nedenle, kendi ihtiyaç duyduğu şekli ile ekranı yeniden oluşturmak internet tarayıcısının görevidir.

OpenGL ile çalışıldığında da yukarıdaki durum oluştuğundan, OpenGL'i kullanarak ekrana görüntü yükleyen program bu görüntüyü tazeleme (refresh) işlemini de yapar. Sürekli kaynak kullanımını önlemek için bu tazeleme işlemi zamana bağlanarak gerçekleştirilir. Tazeleme işlemi, ekrana görüntünün aktarıldığı anı belirtir. Tazelemeyi her an yapmak mümkündür, ancak tazeleme sırasında ekranda yarım çizilmiş görüntüler görmemek için gönderilen görüntünün tamamlanmış olduğu anlar tazeleme için tercih edilmelidir. Devingen bir ortamda bu zamanlamayı yakalamak oldukça zor olacaktır. O halde görüntünün tamamlanmış olduğunu başka mekanizmalarla garanti etmek gerekecektir.

OpenGL bu sorunu çift arabellek (double buffer) kullanarak çözer. Çift arabellek mekanizmasında, belleklerden biri kullanıcının isteklerini yerine getirip saklarken diğeri ekrana aktarılan görüntüyü tutar. Görüntünün tamamlanmasının ardından kullanıcı "SwapBuffers(…)" fonksiyonu ile arabellekleri değiştirir. Böylece ekranda daima istenilen görüntüler gözlenir.

OpenGL fonksiyonları kullanılarak yapılacak her türlü değişiklik ekrana değil, değiştirilmek üzere seçilen belleğe işlenir. Kullanıcı bir görüntü oluşturmak için öncelikle belleği temizlenmelidir. Aksi durumda bu bellekte kalan bilgilerin karışıklık yaratması muhtemeldir. Belleğin temizlenmesi, tüm sahnenin arka plan renginde boyanması şeklinde olur. Bunun için "glClearColor(r,g,b,a)" fonksiyonu kullanılarak bir renk ataması yapılır. Ardından "glClear(…)" fonksiyonuna GL_COLOR_BUFFER_BIT değişken değeri geçirilerek temizleme gerçekleştirilir. Bu fonksiyona ayrıca GL_DEPTH_BUFFER_BIT değişkeni de geçirilerek z-buffer denilen bellekten derinlik bilgileri de temizlenmelidir. Bu bilgiler, cisimler birbiri üzerine çizildiğinde derinlik ölçümüne göre hangisinin nerelerde diğerinden önde

olacağını ve hangisinin ekrana aktarılacağını belirler. Bu nedenle temizlenmesi önemlidir. Ancak bu ölçümün “glEnable(GL_DEPTH_TEST)” komutu ile geçerli kılınmış olması da gerekir.

Bellek temizlendikten sonra, tüm görüntü yeniden oluşturulmalıdır. Sonrasında ise yine arabellekler değiştirilecek ve görüntü ekrana gönderilecektir. Görüntü oluşturma işleminin, aynı fonksiyonlar kullanılarak kendini tekrarlaması durumunda görüntü listesi (display list) denilen OpenGL yapısı kullanılabilir.

Görüntü listeleri, başlangıcı ve bitişi belli olan bir grup OpenGL komutunun saklanarak tekrar çalıştırılmalarına imkan tanıyan yapılardır. Listeler için öncelikle “glGenLists(…)” ile bir tanıtma numarası alınır. Liste başlangıcı için ise bu numara ve “glNewList(…)” komutu kullanılır. Liste içindeki komutlar sıralandıktan sonra “glEndList(…)” ile liste sonlandırılır. Listenin kullanılması ise “glCallList(…)” komutu ile olur. Oluşturulan görüntü listeleri kullanıldıklarında, sakladıkları komutlar o an çağrılmışçasına işlem görürler. Ancak bu işlemlere müdahale edilemez ve görüntü listesi bir defa oluşturulduktan sonra içerdiği komutlar değiştirilemez. Bu nedenle, görüntü listeleri kullanılırken dikkatli olunmalıdır. Sık değişen görüntüler için liste kullanmak yerine tazelemeyi her defasında komutları yeniden çağırarak yapmak yerinde olur. Performans artışı beklenen durumlar ise çok sayıda komutun çalıştırılması gereken ancak komutlardaki değişkenlerin korunduğu durumlardır.

Bu konuda dikkat çekilmesi gereken önemli bir nokta da, görüntü listelerine giren komutların kullandığı değişkenleri nasıl algıladıklarıdır. Örneğin bir renk ayarlama komutu “glColor(a,b,c)” şeklinde görüntü listesinde kullanılmış ise, liste bu komutu o anki değişken değerleri ile saklayacaktır. Listenin yeniden kullanıldığı anda değişkenler a, b ve c’nin yeni değerleri ne olursa olsun, saklanan komut eski değerlerle “glColor(0,0,1)” gibi işlem görecektir. Bu nedenle kullanılan değişken değerlerinin değişmesi durumunda tüm listenin yeniden oluşturulması gerekecektir. Ancak bu işlem görüntü listelerinin kullanımını oldukça sınırlayacağından tercih edilmez. Bunun yerine, iç içe görüntü listeleri tanımlanarak sorun çözülür. En dıştaki görüntü listesi, alt seviyedeki listeleri sırasıyla çağırma işini yapar. Değişkenlere bağlı olarak yenilenmesi gereken

komutlar için sadece ilgili alt liste tekrar oluşturulur ve diğer alt listeler aynen kullanılmaya devam eder. Bu sayede performans artışı sağlanır.

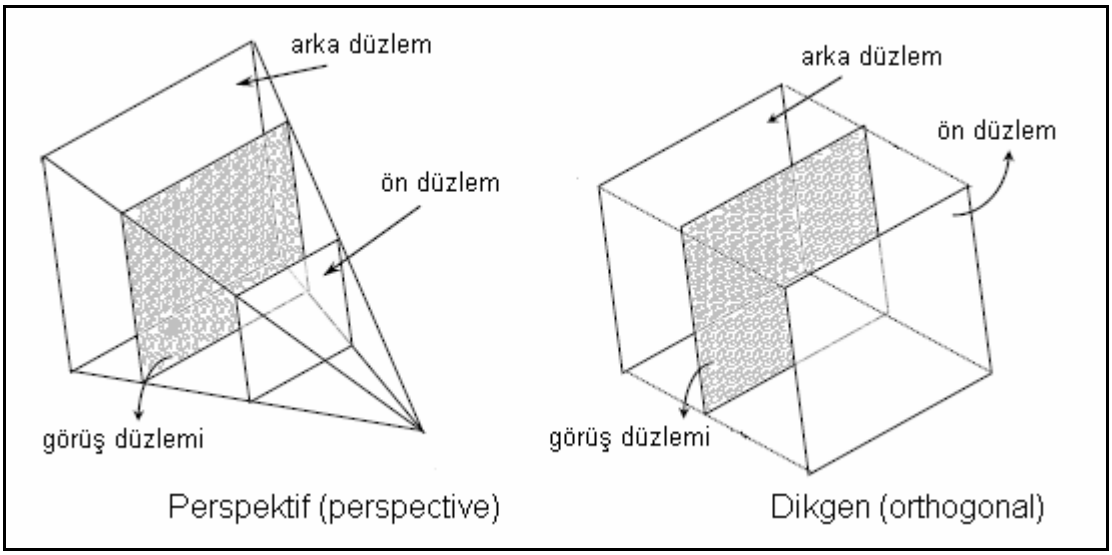
Örneğin bir araba için dört tekerlek çizilecekse, bir tekerleği çizmek için gereken komutlar görüntü listesi şeklinde saklanarak dört defa bu liste çağrılabilir. Ancak yukarıda belirtilen özellik (değişken değerlerinin ilk halleri ile listeye aktarılması ve değiştirilememesi) nedeniyle bu tekerlekler aynı pozisyonlara çizilmiş olacaktır. Benzer şekilde, bir cisim için hazırlanan komutlar, cismin yer değiştirmesi neticesinde yeniden çizilmesi ihtiyacı doğduğunda geçersiz hale gelecektir. Bu durum OpenGL koordinat sistemi özelliği ile sorun olmaktan çıkarılır, çünkü bu koordinat sisteminde dönüşüm matrisleri sayesinde cisimler aynı komutlarla farklı pozisyonlara çizilebilmektedir.

OpenGL koordinat sisteminde, üç adet dönüşüm matrisi bulunur. Bunlar GL_MODELVIEW, GL_PROJECTION ve GL_TEXTURE matrisleridir. GL_MODELVIEW matrisi ile iki farklı dönüşüm kontrol edilir. Bunlardan biri, sanal dünya tabir edebileceğimiz tüm sahneyi, görünür alan koordinatlarına taşıyan görüş alanı (viewing) dönüşümüdür. Diğeri ise, sanal dünya içinde farklı cisimlerin birbirlerine göre konumlarını belirlemeye yarayan modelleme (modeling) dönüşümüdür. GL_PROJECTION matrisi ile izdüşüm dönüşümü kontrol edilir. GL_TEXTURE matrisi ise cisimlerin yüzey kaplamaları için kullandıkları dokuların koordinat dönüşümü içindir. Tüm matrisler ilk kullanımlarından önce özdeşlik (identity) matrisine eşitlenerek etkisiz hale getirilirler. Bu halleriyle, kullanılan komutlara etkileri olmaz. Sonrasında ise istenilen dönüşümler bazı komutlar aracılığı ile uygulanarak matrislerin etkileri değiştirilir.

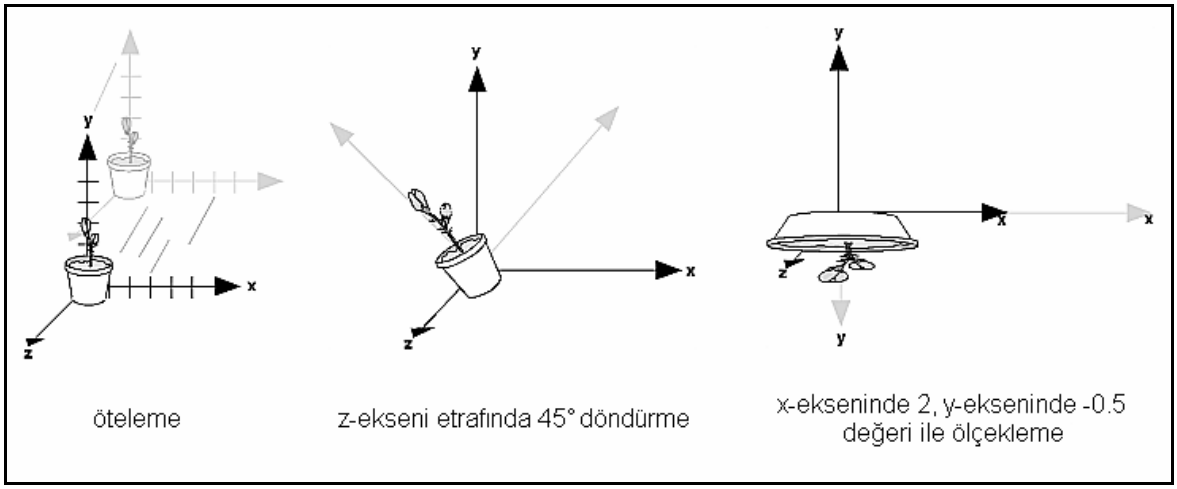
GL_PROJECTION matrisi, üç boyutlu uzaydan iki boyutlu uzaya geçişi sağlar. Bunun için iki seçenek perspektif (perspective) ve dikgen (orthogonal) uzaydır. Sırasıyla bu seçenekleri geçerli kılmak için “glFrustum(…)” ve “glOrtho(…)” komutları kullanılır. “glFrustum(…)” yerine GLU kütüphanesinden “gluPerspective(…)” komutu da kullanılabilir. Farklı iki durum için oluşan görüş alanı Şekil 3.6’da verilmiştir.

GL_MODELVIEW matrisi, üç boyutlu uzayda cisimlerin konumlarını belirlemeye yarar. Bu amaçla kullanılan OpenGL dönüşüm komutları, öteleme için

"glTranslate(…)", döndürme için "glRotate(…)" ve ölçekleme için "glScale(…)"dir (Şekil 3.7). Bunlar modelleme dönüşümleri olarak adlandırılabilir. Bunun yanı sıra, görüş alanı dönüşümü olarak "gluLookAt(…)" komutu kullanılabilir. Bu komut da sanal dünya içindeki kamera pozisyonunu belirleyerek görünecek alanı tanımlar.



Şekil 3.6. İki farklı GL_PROJECTION matris seçeneği ve oluşturdukları görüş alanları ([23], /cgViewing.pdf'den değiştirilerek).

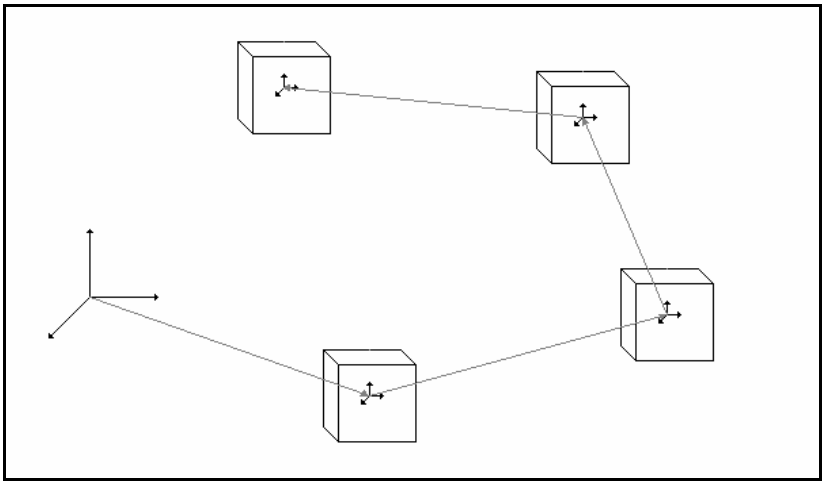


Şekil 3.7. Modelleme dönüşümü örnekleri ([20], /chapter03.html, figure 3-5, 3-6 ve 3-7'den değiştirilip birleştirilerek).

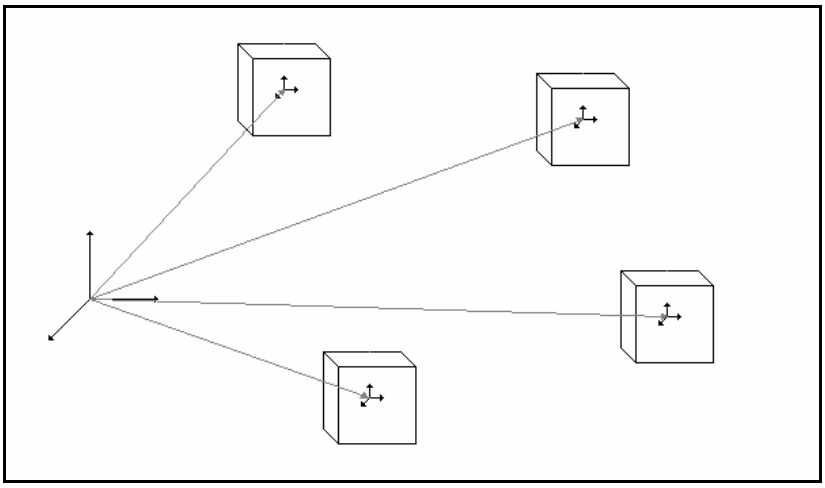
Örneğin sanal dünya koordinatlarına göre (1,0,0) konumuna bir nokta çizmek için iki yol izlenebilir. Nokta çizme komutuna (1,0,0) bilgisi verilebilir; ya da modelleme dönüşümü uygulanarak koordinat eksenleri istenilen konuma getirildikten sonra nokta çizme komutu (0,0,0) bilgisi verilerek çalıştırılır. Bu şekildeki kullanımları

sayesinde modelleme dönüşümleri cisimlerin pozisyonlarını belirlemede etkili olur. Yukarıda bahsedilen dört tekerleğin aynı konuma çizilmesi problemi de bu dönüşümlerle aşılır. Her tekerleğin çizilmesinden önce bu dönüşümler uygulanarak, aynı komutlar ve aynı koordinat bilgileri ile çizilmiş tekerleklerin farklı konumlara yerleştirilmesi sağlanır. Burada dikkat edilmesi gereken husus, dönüşümlerin hangi hiyerarşiye göre uygulandığıdır. Şöyle ki, bir tekerleği çizmek üzere uygulanmış bir dönüşümün ardından diğer tekerleği çizmek için uygulanacak dönüşüm farklıdır. Birincisi sanal dünya koordinatlarını istediği konuma taşıdığından, ikinci dönüşüm yeni koordinatları kendi istediği pozisyona taşıyacaktır. Bu hiyerarşik yapı Şekil 3.8'de gösterilmiştir. Diğer bir yöntem ise dönüşümleri uygulayıp istenilen çizimi yaptıktan sonra ters dönüşümle koordinatları eski pozisyona taşımaktır. Böylece çizilecek her cisim için uygulanacak dönüşüm, doğrudan cismin konumunu verecektir. Bu hiyerarşi ise Şekil 3.9'da gösterilmiştir.

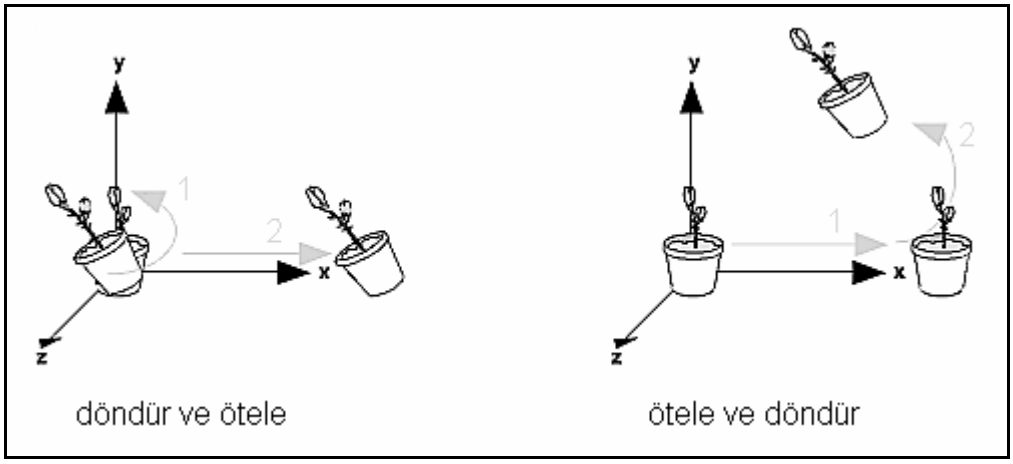
Modelleme dönüşümleri için ters dönüşüm uygulamak her zaman kolay olmayabilir, çünkü genellikle birden fazla dönüşüm birlikte kullanılarak yeni konum belirlenmiştir. Bu durumda, dönüşümlerin değişme özelliğinin olmadığına (Şekil 3.10) ve dönüşümlerin ters dönüşümdeki uygulama sırasına dikkat edilmelidir. OpenGL matris siteminin bir özelliği bu geri dönüşümün daha kolay yapılabilmesine imkan vermektedir. Bu özellik matris yığını (stack) kullanılmasıdır. OpenGL için o anda geçerli olan matris, yığının tepesindekidir. Kullanıcı istediği dönüşümleri uygulamadan önce, en tepedeki aktif matrisi yığında bir alt kademeye göndererek saklayabilir. Daha sonra kendi dönüşümlerini uygulayarak çizimlerini tamamlamasının ardından sakladığı matrisi geri çağırır. Bu şekilde ters dönüşümleri uygulamasına gerek kalmadan kendinden önceki koordinatları geri getirmiş olur. Matrisleri saklamak için "glPushMatrix(…)", geri çağırmak içinse "glPopMatrix(…)" komutları kullanılır.



Şekil 3.8. Modelleme dönüşümlerinin birbiri ardına uygulandığı hiyerarşik yapı.



Şekil 3.9. Modelleme dönüşümlerinin ters dönüşümlerden sonra uygulandığı hiyerarşik yapı.



Şekil 3.10. Modelleme dönüşümlerinin farklı uygulama sırasına göre sonuçları ([20], /chapter03.html, figure 3-4'ten değiştirilerek).

Durum değişkenleri ve konum gibi özellikler ayarlandıktan sonra, çizim komutları sıralanır. OpenGL çizimlerinde nokta, doğru ve çokgen gibi ilkel geometrik şekiller kullanılır. Bu şekillerden hangisinin çizileceği "glBegin(…)" fonksiyonuna girilen çizim kipi değişkeni ile belirtilir. Burada kullanılabilecek değişken değerleri ve bu değerler ile ifade edilen geometrik şekiller aşağıdaki Çizelge 3.2 ile özetlenmiştir. Kullanıcı, "glBegin(…)" ile açtığı komut bölgesini "glEnd()" ile kapatmalıdır. Bu komut bölgesinde sıralanacak "glVertex(…)" komutları ile de şekil çizilir. "glVertex(…)" komutları, seçilen çizim kipine göre işlenir. Örneğin GL_POINTS seçilmiş ise her "glVertex(…)" için bir nokta çizilir. GL_TRIANGLES seçilmiş ise, art arda kullanılan üç "glVertex(…)" komutu bir üçgen çizilmesine neden olur. GL_TRIANGLE_FAN seçilmiş ise de, ilk "glVertex(…)" komutu tüm üçgenler için ortak noktayı belirtecek şekilde yorumlanır ve ardından gelen "glVertex(…)" komutlarından ardışık olarak alınan her komut ikilisi için bir üçgen oluşturulur. Bu kip ile, n "glVertex(…)" komutu kullanılarak n-2 üçgen oluşturulmuş olur. Bu kipler ile oluşturulabilecek örnekler Şekil 3.11'de gösterilmektedir. Şekildeki Vn indisleri, n kullanım sıralarını belirtmek üzere "glVertex(…)" komutlarının belirttiği noktaları göstermektedir.

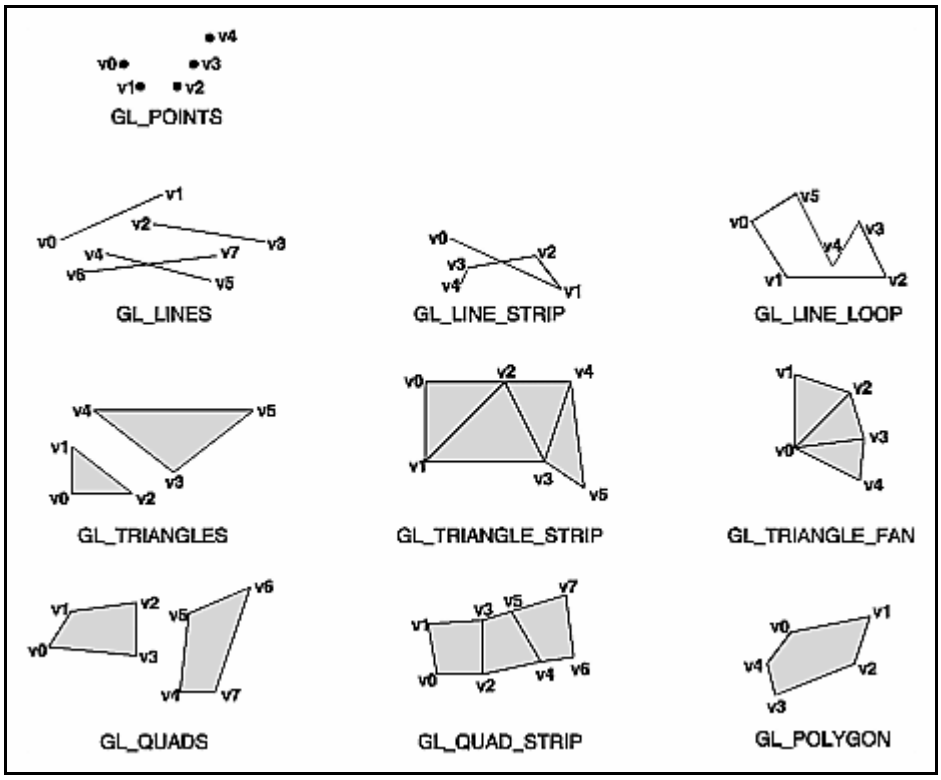
Çizim bölgesi olarak tabir edilen "glBegin(…)" ve "glEnd()" komutları arasındaki bölgede kullanılabilecek OpenGL komutları Çizelge 3.3'te verilenlerle sınırlıdır. Bunlar dışında kalan komutlar kullanıldıkları takdirde hata oluşturur ya da öngörülemez davranışlar gösterirler.

Çizelge 3.2. Geometrik ilkel şekiller için OpenGL kipleri ve anlamları ([20], /chapter02.html, table 2-2'den değiştirilerek).

Kip değişkeni değeri	**Değişkenin anlamı**
GL_POINTS	Tek tek noktalar
GL_LINES	Nokta çiftlerinden elde edilen doğru parçaları
GL_LINE_STRIP	Seri halde bağlı doğru parçaları
GL_LINE_LOOP	Son ve ilk noktanın da bağlandığı seri doğru parçaları
GL_TRIANGLES	Nokta üçlülerinden elde edilen üçgenler
GL_TRIANGLE_STRIP	Birbirine bağlı üçgenler şeridi
GL_TRIANGLE_FAN	Birbirine bağlı üçgenler yelpazesi
GL_QUADS	Nokta dörtlülerinden elde edilen dörtgenler
GL_QUAD_STRIP	Birbirine bağlı dörtgenler şeridi
GL_POLYGON	Basit dışbükey çokgen

Çizelge 3.3. “glBegin(…)” ve “glEnd()” komutları arasındaki çizim bölgesinde kullanılabilecek OpenGL komutları ([20], /chapter02.html, table 2-3’ten değiştirilerek).

glVertex	glColor	glIndex
glNormal	glTexCoord	glEdgeFlag
glMaterial	glArrayElement	glEvalCoord
glEvalPoint	glCallList	glCallLists



Şekil 3.11. İlkel geometrik şekillerle oluşturulan örnekler ([20], /chapter02.html, figure 2-7).

Buraya kadar aktarılanlar, bir OpenGL görüntüsü oluşturmak için yeterlidir. Ancak elbette ki bu geniş kütüphane ile yapılabilecekler bunlarla sınırlı değildir. Bu konuda daha detaylı bilgi edinmek isteyen okuyucular için önerilebilecek kaynaklar [20] ve [24]’tür.

Tez kapsamında, bu anlatılanlara ek olarak windowing özelliğine ihtiyaç duyulduğundan, Microsoft Visual Studio .NET pencereleri ile OpenGL bağlamlarını eşleştirebilecek bir yapı kullanılmıştır. Bu yapıda, program penceresinde bir standart .NET Form elamanı olan Panel nesnesi kullanılır. Bu

nesnenin işleyicisi (handler) kullanılarak kendisini ebeveyn olarak görecek sanal bir pencere oluşturulur. Bu sanal pencerenin aygıt ve resmetme bağlamları da yaratılarak kullanıma hazır hale getirilir. Bu şekilde, ana pencere içindeki bir elemanın yavru (child) alanına ait bağlamlarda yapılan değişiklikler, ebeveyn alan sayesinde ekrana taşınmış olur. Bu yöntem [25]'ten alınarak uyarlanmış ve tezde kullanılmıştır.

3.1.3. Işın takibi (ray tracing) algoritması

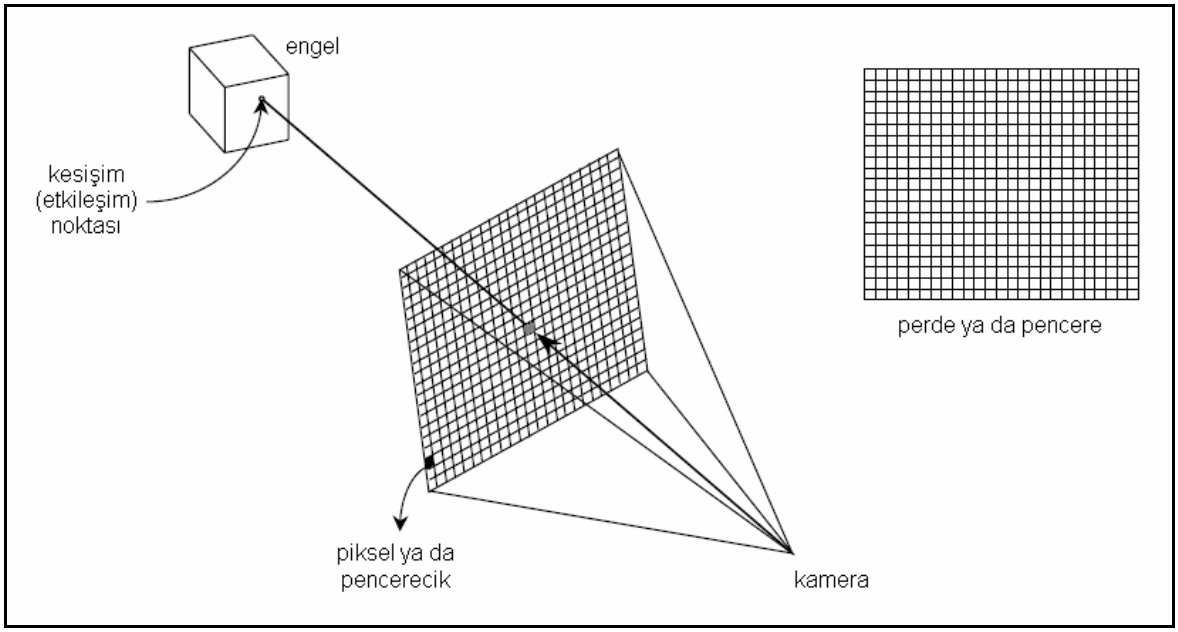
Işın takibi, geometrik optik alanından genel bir tekniktir. Bu teknikte ışığın aldığı yol, ışık ışınlarının optik yüzeylerle etkileşimlerinin takip edilmesiyle modellenir. Işın takibi tekniğinden, kamera merceği, teleskop, dürbün ve mikroskop gibi optik sistemlerinin tasarımında yararlanılır. Ancak burada ele alınacak haliyle, üç boyutlu (3D) bilgisayar grafiklerinin oluşturulmasında kullanılan bir resmetme (rendering) yönteminin adıdır [26].

Bilgisayar grafikleri önce matematiksel olarak hesaplanır ve modellenirler. Bu modelleme, arzu edilen görüntü kalitesi ya da detayına bağlı olarak farklı yöntemlerle yapılabilir. Tarama hattı resmetmesi (scanline rendering), ışın taksimi (ray casting) ve ışın takibi (ray tracing) sık kullanılan örneklerdir. Tarama hattı resmetmesi yönteminde, çokgen ya da piksel tabanlı değil satır tabanlı çalışılır. Önce resmedilecek tüm çokgenler en yukarıdaki y koordinatlarına göre sıralanırlar. Sonra, oluşturulacak görüntünün en yukarıdaki satırından başlanarak her bir satır için, sıralamanın en başındaki çokgenlerle kesişme durumu kontrol edilerek görüntü taranır. Görüntüde alt satırlara inildikçe yukarıda kalan çokgenler listeden çıkarılarak işleme devam edilir [27]. Işın taksimi yöntemi ise ışın takibi tekniğinin kısaltılmış hali gibidir. Bu teknikteki farklılık, ışının bir engel ile etkileşiminden sonraki davranışının takip edilmemesidir. Bu nedenle ışın takibi ve ışın taksimi isimleri kullanılmaktadır. Bu teknik kullanıldığında, yansımaların, kırılmaların ve gölgelemelerin kusursuzca resmedilmesi ihtimali ortadan kalkar. Ancak bu tipteki detayları, doku kaplama gibi yöntemlerle bir dereceye kadar taklit etmek mümkün olabilmektedir [28].

Görüntünün oluşması, bir kaynaktan çıkan ışığın çeşitli engellerle etkileşimi sonrasında göz ya da kameraya ulaşmasıyla olur. Bu doğal olay ışın takibi tekniğinin de temelidir. Ancak bu teknikte ışınların takibi kaynaktan kameraya

doğru değil, kameradan kaynağa doğru yapılır. Bu sayede, kaynaktan çıktığı halde engeller nedeniyle kameraya ulaşması mümkün olamayacak ışınların takibi yapılmamış ve işlem yükü hafifletilmiş olur. Kameradan kaynağa doğru yapılan takibin ışının hareketine ters yönde olmasından hareketle, bu tekniğe ters ışın takibi (backwards ray tracing) de denilmektedir.

Oluşturulacak iki boyutlu bilgisayar görüntüsü, resmedilen üç boyutlu dünyanın bir perde üzerine düşen ya da bir pencereden izlenebilen hali olarak düşünülebilir. Oluşturulacak görüntü ise piksellerden ibaret olacaktır. O halde her bir piksel, üç boyutlu dünyaya açılan pencerenin bir bölümünü ifade edebilir. Görüntünün oluşturulması ise, bu pencereciklerden nelerin görülebildiğini bulma işlemine dönüşür. Bu amaçla her bir pencerecikten kameraya gelecek ışının yansıtıcısı bulunmaya çalışılır, çünkü bu yansıtıcı kameranın gördüğü cisimdir (Şekil 3.12).

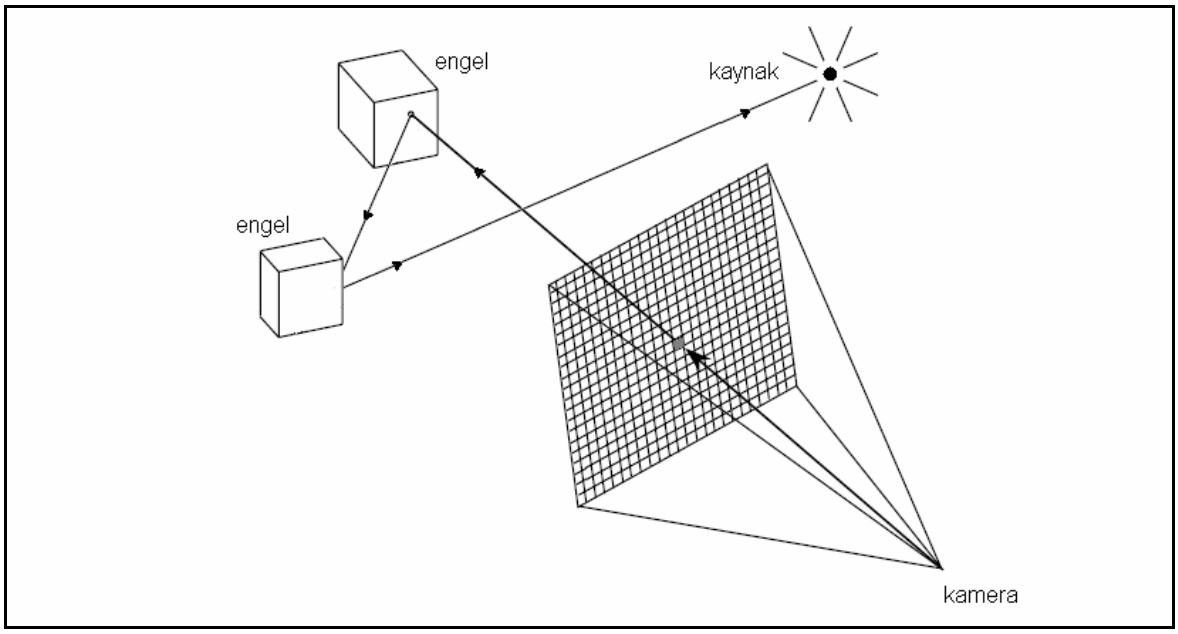


Şekil 3.12. Işın takibi tekniğinde kamera, engel ve pencerecik kavramları ([23], /cgHLHSR.pdf'den değiştirilerek).

İşte bu noktada, takibi yapılmak üzere sanki kameradan çıkıyormuşçasına bir sanal ışın gönderilir. Göz ışını da denilen bu sanal ışının gerçek bir ışını ifade edebilmesi için, bir ışık kaynağına ulaşması gerekir. Yolculuğuna kamerada başlayan sanal ışın bir kaynakta son bulmalıdır ki bu ışın gerçek ışının ters yönde takip edilmiş hali olsun. Takibi yapılan ışının gerçek ışına dönüşmesi, bakılan pencerecikten görülen cismi de belirlemiş olur. O pencereciğe karşılık gelen piksel

rengi de kameradan gönderilen ışının ulaştığı engel ve ışık kaynağının renkleri ile belirlenir.

Işın takibi algoritması ise ayrıntıda şu şekilde işler. Yukarıda anlatıldığı üzere, her bir pencerecikten ışın gönderilerek işlem yapılmaktadır. Bu nedenle algoritma, oluşturulacak görüntüdeki piksel sayısı kadar tekrarlanan bir yapıdadır. Bir piksel için ise, gönderilen sanal ışının bir engelle karşılaşıp karşılaşmayacağına bakılır. Bunun için üç boyutlu sahnedeki tüm cisimlerle (engeller ve ışık kaynakları) kesişim testi yapılır. Eğer gönderilen sanal ışın hiçbir cisimle kesişmiyorsa, o piksele ışın ulaşmayacağından piksel karanlık olarak renklendirilecek demektir. Bu noktada diğer piksel için işleme geçilir. Gönderilen ışın bir engelle kesişiyorsa, işlem devam eder. Engelle kesişen sanal ışının, bu etkileşim sonrasında hangi yönde yoluna devam edeceği hesaplanır ve o yönde yeni bir ışın gönderilir. Engelden çıkan bu ışın için de sahnedeki tüm cisimlerle kesişme testi yapılarak yukarıdaki adımlar tekrarlanır. Işının son olarak hiçbir cisimle kesişmemesi durumunda piksel karanlık olur. Ancak ışın bir ışık kaynağı ile kesiştiğinde, piksel için ışık kaynağı ve engelin renklerine bakılarak bir renk atanır (Şekil 3.13).



Şekil 3.13. Işın takibi tekniğinde sanal ışının izlediği yol.

Bu algoritmadaki en zaman alıcı işlem, cisimlerle ışınların kesişme testlerinin yapılmasıdır. Hiç bir engelle karşılaşılmasa dahi, tüm pikseller için birer defa ışın gönderilip kesişme testi yapılacağından bu testin süresi daha da önem

kazanmaktadır. Bu testler, matematiksel olarak tanımlanabilen geometrik şekillerdeki cisimler için kolay olabilmektedir. Örneğin küre şeklindeki bir cisim için bu işlem [26]'ya göre şu şekilde yapılabilir. Küre için r yarıçapı, C merkez noktasını ve I küre yüzeyi üzerindeki bir noktayı göstermek üzere Eş. 3.1'deki tanım yapılabilir. Işın için de S başlangıç noktasını ve D doğrultusunu göstermek üzere t değişkenine bağlı olarak Eş. 3.2'deki tanım yapılır. Bu iki denklem birlikte çözüldüğünde Eş. 3.3 elde edilir (V=S–C). Bu denklemin çözümünün olabilmesi için Eş. 3.4 sağlanmalıdır. Görüldüğü gibi ışın ile kürenin kesişim testi basit bir büyüklük kontrolü ile yapılabilmektedir. Ancak çoğunlukla üç boyutlu sahnede yer alan şekiller matematiksel olarak tanımlanamamakta ve testlerin bu kadar kolay yapılmasına imkan vermemektedir. Bu nedenle ışın takibi tekniği oldukça zaman alıcı olabilmektedir. Ancak çok başarılı biçimde yansıma ve gölgeleme gibi ayrıntıları resmedebilmesi nedeniyle de bilgisayar grafikleri dünyasında sık kullanılmaktadır.

$$|I - C|^2 = r^2 \tag{3.1}$$

$$S + tD = 0 \tag{3.2}$$

$$|V - tD|^2 = r^2 \tag{3.3}$$

$$(2V \bullet D)^2 - 4(D^2)(V^2 - r^2) \tag{3.4}$$

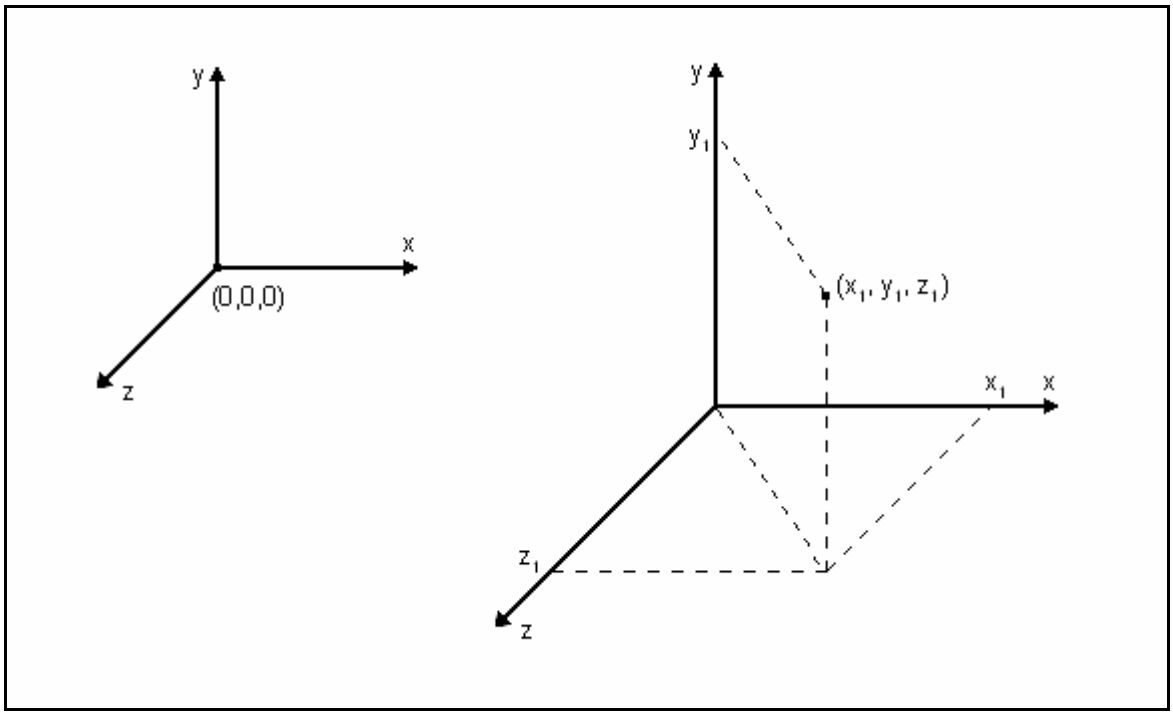
Işın takibi konusunda ekstra bilgi [29]'da bulunabilir. Ayrıca [30]'da bulunan etkileşimli örnek, algoritmanın görsel bir sunumunu içermektedir.

Tezde, radar ışınlarının verici, hedef ve alıcı arasında izlediği yolun bulunmasında ışın takibi algoritmasından faydalanılmıştır. Bu algoritmadan farklı olarak, tek bir sinyal kaynağı olması nedeniyle ışınlar olağan yönlerinde vericiden alıcıya doğru takip edilmişlerdir. Ayrıca amaç bir görüntü oluşturmaktan ziyade hedef cisim üzerindeki radar yansımalarını bulmak olduğundan, hedefteki tüm yüzeycikler hem piksel hem de engel gibi değerlendirilmiştir. Sırasıyla tüm yüzeycikler için radar vericisinden ışın gönderilmiş ve ardından bu ışının tüm diğer yüzeyciklerle kesişimleri test edilerek radar vericisine erişimi değerlendirilmiştir.

3.1.4. Üç boyutlu uzayda geometrik denklemler

Tezin kapsamı ve ihtiyaçlarının gereği olarak, üç boyutlu cisimlerin uzaydaki şekillerini, konumlarını veya hareketlerini tanımlayan geometrik denklemlere ihtiyaç duyulmuştur. Hedef cismin ekrana çizilmesi, görüntünün hareket ettirilmesi, radar sinyallerinin cisimle olan etkileşimi gibi birçok adımda kullanılan geometrik tanımlar ve denklemler bu bölümde kısaca açıklanacaktır.

Üç boyutlu kartezyen koordinat sisteminde birbirini dik olarak kesen x, y, z ana eksenleri bulunur. Bu eksenlerin kesiştikleri yer başlangıç noktası veya orijin diye adlandırılır. Bu uzaydaki bir nokta ise, bulunduğu konumun üç eksen üzerindeki izdüşüm değerleri ile belirlenir (Şekil 3.14). Örneğin (2,4,-5) gösterimi, x eksenindeki izdüşümü başlangıç noktasına 2 birim uzaklıkta olan bir noktayı ifade eder. Noktanın alanı, hacmi, uzunluğu, yönü gibi özellikleri tanımsızdır.



Şekil 3.14. Üç boyutlu kartezyen koordinat sistemi ve nokta tanımı.

Uzaydaki herhangi farklı iki noktayı birleştiren çizgiye doğru parçası denir. Doğru ise bu doğru parçasının uçları noktalarla sınırlandırılmamış ve sonsuza uzanan halidir. $P_1=(p_{1x},p_{1y},p_{1z})$ ve $P_2=(p_{2x},p_{2y},p_{2z})$ noktalarından geçen doğru üzerindeki $P=(p_x,p_y,p_z)$ noktalarının denklemi Eş. 3.5'teki gibi verilebilir. Burada k sabitinin

değişimi ile doğru üzerinde farklı P noktaları elde edilir. Noktaların koordinatlarını kullanarak aynı doğru denklemi Eş. 3.6'daki haliyle de yazılabilir.

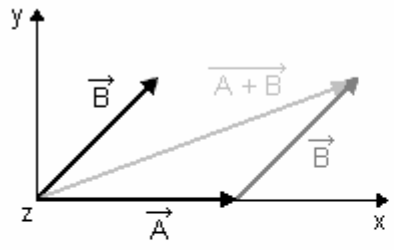
$$P = P_1 + k(P_2 - P_1) \tag{3.5}$$

$$\frac{p_x - p_{1x}}{p_{2x} - p_{1x}} = \frac{p_y - p_{1y}}{p_{2y} - p_{1y}} = \frac{p_z - p_{1z}}{p_{2z} - p_{1z}} = k \tag{3.6}$$

Geometrideki tanımıyla ışın, sadece bir yönde sonsuza uzanan diğer ucu ise bir nokta ile sınırlı olan doğrudur. Vektör ise yönlü doğru parçası olarak tanımlanabilir. Başlangıç ve bitiş noktaları olduğundan büyüklüğü bellidir ve sonsuz değildir. Başlangıç noktası A ve bitiş noktası B olan V vektörü Eş. 3.7'deki şekillerde ifade edilebilir.

$$\overrightarrow{V} = \overrightarrow{AB} = [AB] = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} b_x - a_x \\ b_y - a_y \\ b_z - a_z \end{bmatrix} \tag{3.7}$$

Vektörler, başlangıçları orijine taşınarak da gösterilebilirler. Vektörlerin bu şekilde taşınabilmesi, toplama ve çıkarma işlemlerinde uç uca ekleme yönteminin uygulanabilmesine imkan verir. Vektörler üzerindeki toplama ya da çıkarma işlemi, x, y ve z bileşenlerinin büyüklükleri toplanarak ya da çıkarılarak yapılır (Şekil 3.15).



$$\vec{A} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \quad \vec{B} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

$$\overrightarrow{A+B} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \begin{bmatrix} a_x + b_x \\ a_y + b_y \\ a_z + b_z \end{bmatrix}$$

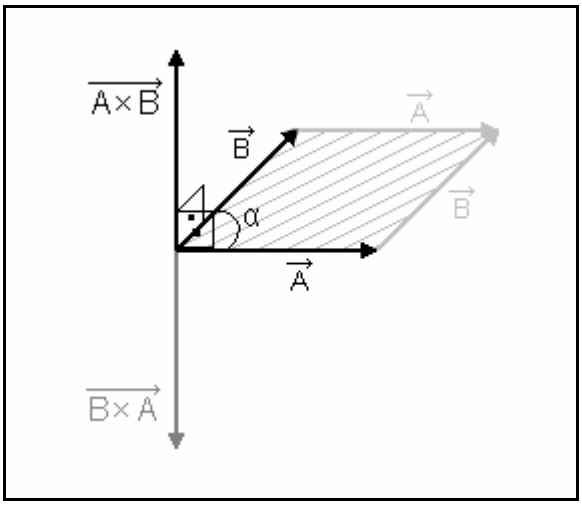
$$\overrightarrow{A-B} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} - \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \begin{bmatrix} a_x - b_x \\ a_y - b_y \\ a_z - b_z \end{bmatrix}$$

Şekil 3.15. Vektörlerde toplama ve çıkarma işlemleri.

Vektörlerde çarpma işlemi ise iki farklı şekilde tanımlanmıştır. Nokta çarpım, içsel çarpım ya da diğer adıyla yönsüz çarpım (dot product, inner product, scalar product) sayısal sonuç verirken, çapraz çarpım ya da vektör çarpım (cross product, vector product) denilen işlem sonuçta yine bir vektör verir. Doğrultuları arasındaki açı α olan iki vektör için nokta çarpım Eş. 3.8'deki gibi bulunur. Bu eşitlik kullanılarak, koordinatları bilinen iki vektör arasındaki açı da hesaplanabilir. Aynı iki vektör için çapraz çarpımın bulunuşu ise Eş. 3.9'dadır. Buradaki e_x, e_y ve e_z koordinat eksenleri yönündeki, n ise çarpım vektörü yönündeki birim vektörlerdir. Çapraz çarpımdan elde edilen vektör çarpılan iki vektöre de diktir ve yönü sağ el kuralına göre bulunur. Ayrıca bu çarpım vektörünün büyüklüğü, iki vektörün kenarlarını oluşturduğu paralelkenarın alanına eşittir (Şekil 3.16).

$$\overrightarrow{A} \bullet \overrightarrow{B} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \bullet \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = a_x b_x + a_y b_y + a_z b_z = \left| \overrightarrow{A} \right| \left| \overrightarrow{B} \right| \cos\alpha \tag{3.8}$$

$$\overrightarrow{A} \times \overrightarrow{B} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \times \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \det \begin{bmatrix} e_x & e_y & e_z \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{bmatrix} = n \left| \overrightarrow{A} \right| \left| \overrightarrow{B} \right| \sin\alpha \tag{3.9}$$



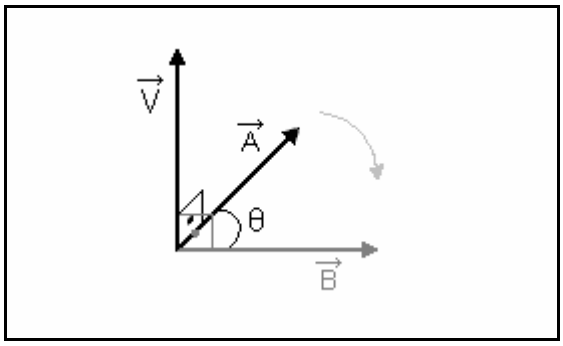
Şekil 3.16. Vektörlerde çapraz (vektör) çarpım işlemi.

Bir vektörün, başka bir vektör ile işaret edilen doğrultu etrafında belli bir açı kadar döndürülmesi işlemi Şekil 3.17'de gösterilmiştir. Bu işlemin sonucu, farklı yönde yeni bir vektör olacaktır. Döndürülen vektörün büyüklüğünün etkilenmemesi için

döndürme eksenini tanımlayan vektör birim vektör olmalıdır. Döndürülecek vektör $A=(a_x,a_y,a_z)$, dönme ekseni birim vektörü $V=(v_x,v_y,v_z)$ ve dönme açısı θ ile gösterilmek üzere M döndürme matrisi ile bu işlemin sonucu olan $B=(b_x,b_y,b_z)$ vektörü Eş. 3.10'da verilmiştir [31].

$$M = \begin{bmatrix} (1-\cos\theta)v_x v_x + (\cos\theta) & (1-\cos\theta)v_y v_x + (\sin\theta)v_z & (1-\cos\theta)v_z v_x + (\sin\theta)v_y \\ (1-\cos\theta)v_x v_y + (\sin\theta)v_z & (1-\cos\theta)v_y v_y + (\cos\theta) & (1-\cos\theta)v_z v_y + (\sin\theta)v_x \\ (1-\cos\theta)v_x v_z + (\sin\theta)v_y & (1-\cos\theta)v_y v_z + (\sin\theta)v_x & (1-\cos\theta)v_z v_z + (\cos\theta) \end{bmatrix}$$

$$\overrightarrow{B} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} M \qquad (3.10)$$



Şekil 3.17. Bir vektörün başka bir vektör etrafında döndürülmesi işlemi.

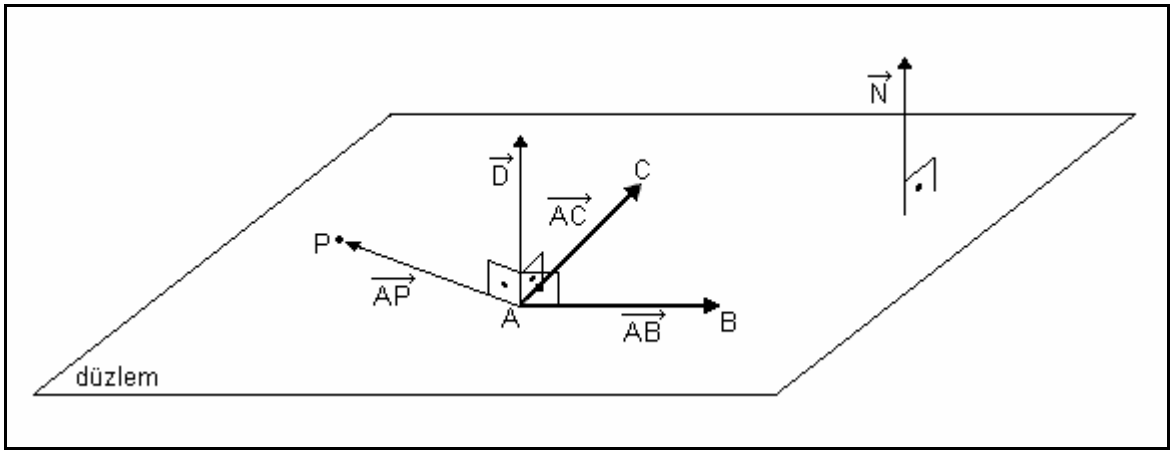
Uzaydaki üç noktayı içine alan yüzeye düzlem denir. Daha çok bilinen tabiri ile, uzayda üç nokta bir düzlem belirtir. Yüzeyin normal vektörü ve yüzeyden bir nokta ile de düzlem tanımlanabilir. Düzlem için büyüklük kavramı yoktur, sonsuza uzanır. A, B ve C noktalarının belirlediği düzlem denklemini bulmak için vektörlerden faydalanılır. [AB] ve [AC] vektörleri düzlem içinde kaldığından bunların çapraz çarpımları yüzeye dik olan bir vektör verecektir. Bu dik vektör ile yüzey içindeki herhangi bir vektörün nokta çarpımı da sıfır olacaktır. Şekil 3.18'de de gösterilen bu özellikler kullanılarak düzlem denklemi Eş. 3.11'deki gibi bulunur. Ayrıca, D vektörünü hesaplamadan da, bilinen bir N normal vektörü varsa bu vektör de yüzeydeki herhangi bir vektöre dik olacağından Eş. 3.12 kullanılarak düzlem denklemi bulunabilir.

$$[AB]=\begin{bmatrix} b_x-a_x \\ b_y-a_y \\ b_z-a_z \end{bmatrix},\ [AC]=\begin{bmatrix} c_x-a_x \\ c_y-a_y \\ c_z-a_z \end{bmatrix} \text{ ve } \overrightarrow{D}=[AB]\times[AC] \text{ olmak üzere}$$

$$\overrightarrow{D}\bullet[AP]=\begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix}\bullet\begin{bmatrix} p_x-a_x \\ p_y-a_y \\ p_z-a_z \end{bmatrix}=d_xp_x+d_yp_y+d_zp_z-\left(d_xa_x+d_ya_y+d_za_z\right)=0 \qquad (3.11)$$

$$\overrightarrow{N}=\begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \text{ olmak üzere}$$

$$\overrightarrow{N}\bullet[AP]=\begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix}\bullet\begin{bmatrix} p_x-a_x \\ p_y-a_y \\ p_z-a_z \end{bmatrix}=n_xp_x+n_yp_y+n_zp_z-\left(n_xa_x+n_ya_y+n_za_z\right)=0 \qquad (3.12)$$



Şekil 3.18. Düzlemi tanımlayan nokta ve vektörler.

Düzlem, sonsuza kadar uzanarak uzayı iki parçaya böler. Bir noktanın uzayın hangi kısmında kaldığının bulunması için düzlem denkleminde P noktası yerine konulması gerekir. Bu şekilde çözülen denklem =0 sonucunu verirse nokta düzlemin içindedir. Sonuç >0 çıkarsa nokta düzlemin önünde, <0 çıkarsa da arkasındadır. Düzlemin normal vektörünün işaret ettiği yön, ön tarafı kabul edilir.

Düzlemin bir doğru parçası ile kesişimi de iki denklemin (Eş. 3.12 ile Eş. 3.5) birlikte çözülmesiyle bulunur. Bu çözüm sonucunda aşağıdaki Eş. 3.13 elde edilir. Buradaki k değeri, kesişme noktasının doğru parçasının başlangıç noktasına

uzaklığını belirtir. Eğer 0<k<1 ise kesişim noktası doğru parçası üzerindedir ve düzlem ile doğru parçası kesişmektedir. Eşitlikteki payda değeri sıfır olduğunda ise k sonsuz olacağından düzlem ile doğru parçası paraleldir ve kesişmezler [32].

$$k = \frac{n_x p_{1x} + n_y p_{1y} + n_z p_{1z} - \left(n_x a_x + n_y a_y + n_z a_z\right)}{n_x\left(p_{1x} - p_{2x}\right) + n_y\left(p_{1y} - p_{2y}\right) + n_z\left(p_{1z} - p_{2z}\right)} \tag{3.13}$$

Burada belirtilen geometrik kavram ve eşitlikler tez kapsamında ihtiyaç duyuldukça kullanılmış olanlardır. Bunun ötesinde ayrıntılı bilgi, diğer kaynakların yanı sıra [33], [34] ve [35]'ten erişilecek bağlantılarda bulunabilir.

3.2. Tasarım Bilgileri

Bölüm 3.1'in girişinde de bahsedildiği gibi, RCSP programı tasarımı tamamen özgün olup tez kapsamında yapılmıştır. Programda kullanılan her türlü kod parçası ve veri yapısı da özgün olarak tasarlanarak kodlanmıştır. Bundan amaçlanan, tüm koda hakimiyeti sağlamak, her türlü değişikliğe kolaylık ve imkan tanımak ve katkısı öngörülemeyen olası yabancı kodları en aza indirmektir.

Aşağıdaki alt başlıklarda tasarım yaklaşımı, geliştirme platformu, kod yapısı, kod etkileşimleri ve işleyişi gibi tasarım bilgileri verilecektir.

3.2.1. Tasarım yaklaşımı

RCSP programı tasarımında nesne tabanlı programlama (object-oriented programming, OOP) yaklaşımı uygulanmıştır. Nesne tabanlı yaklaşım, programlama dünyasında yaygın olarak kullanılmaktadır. Programlama tarihine bakıldığında, fonksiyon ya da akış bazlı (function based, flow based) yöntemlerden sonra geldiği ve en son eğilimlerden biri olduğu görülür. Bu yaklaşımda, tasarım nesneler üzerinden yapılır. Nesnelerin hangi durumlarda nasıl tepkiler vereceği ya da verebileceği tanımlanır. Bu şekilde olaylara değil nesnelere önem yüklenmiş olur. Nesneleri kullanan diğer yapılar da, nesnenin kabiliyeti çerçevesinde bir işi yapacağını bilir ancak nasıl yapacağı ile ilgilenmemiş olur. Bu sayede de bilgilerin sadece bilmesi gereken yapılara açık olduğu bir soyutlama (abstraction) ortamı da kurulmuş olur.

Nesneler bir defa tasarlanıp kodlandıklarında farklı yerlerde tekrar kullanılabilirler. Örneğin bir otomobil nesnesinin tasarımında, bu otomobilin ihtiyaç duyacağı nesnelerden olan motor ya da tekerlek daha önce başka bir nesne için tasarlanmış olabilir. Bu tekrar kullanılabilirlik (reusability) özelliği sayesinde nesneler küçük değişikliklerle farklı yerlerde benzer işleri görebilir.

Nesnelerle tasarım yapmanın bir avantajı da farklı nesnelerin tasarımlarının farklı kişilerce yapılabilmesidir. Bu yaklaşımda önemli olan diğer nesnelerin varlığı ve sundukları fonksiyonlar olduğundan, tasarlanan nesnenin tasarımı sadece bunlardan etkilenir. Örnekteki otomobil nesnesinin çalışmasını tasarlayan kişi bu konuya hakim şekilde işini sürdürürken, sürücü nesnesinin tasarımcısı sadece otomobilin sürücüye sunacağı fonksiyonları bilerek tasarımına devam edebilir. Bu sayede tasarım iş yükü paylaştırılmış ve zaman kazanılmış olur.

Gerçek hayatta da işler nesne tabanlı modele uygun yürümektedir. Örneğin otomobilin anahtarını çeviren kullanıcı, motorda neler olduğuyla ilgilenmez. Onun yaptığı otomobil nesnesine çalış komutunu vermekten ibarettir. Ancak bir tamirci için otomobil nesnesinin sağladığı komutlar daha farklı olabilir. Burada da nesneler arası soyutlama örneği görülür. Nesne tabanlı yaklaşımın gerçek hayata uygunluğu nedeniyle, birçok problem için doğal çözüm yöntemi nesne tabanlı olmaktadır.

Nesne tabanlı modelde nesnelerin (object) ait oldukları ve onların özelliklerini belirleyen yapılara sınıf (class) denir. Sınıfların belli işler için kullanıma açtıkları üye değişkenleri (member variables) ve üye fonksiyonları (member functions) vardır. Program çalışırken bu sınıflara ait nesneler zamanla yaratılıp yok edilebilirler. O an aktif halde bulunan nesneler birbirleri üzerinden fonksiyonlar çağırıp işlemler yaptırabilirler.

Sınıflar dışında kalan bir yapı da sayım listesidir (enumeration). Bu yapı da nesneler arası bilgi geçirme gibi işlemlerde bilginin kodlayıcının anlayacağı biçimde sınıflandırmasını sağlar. Örneğin bir cismin dört yöne hareket ettirilmesi mümkün ise, bu cisme hareket yönü bilgisi 1, 2, 3, 4 gibi rakamlar yerine SAG, SOL, UST, ALT gibi tanımlarla geçirilir. Bu tanımlar ise sayım listelerinde değerleri belirtilerek tanımlanmış olmalıdır.

3.2.2. Geliştirme platformu

Yapılan literatür taramasında, bu tez benzeri çalışmaların MATLAB kullanılarak hayata geçirildiği görülmüştür. MATLAB, başlangıç amacı sınırlı olsa da günümüz itibari ile birçok farklı işin yapılmasında kullanılan gelişmiş bir platformdur. Sinyal ya da görüntü işleme gibi işlerde özelleşmiş kütüphaneleri bulunduğu gibi basit matematik hesaplar için de sunduğu fonksiyonları vardır. Ayrıca grafik kullanıcı arayüzü oluşturulmasına ve hatta oluşturulan kodun ayrı bir program haline getirilmesine de imkan sunmaktadır. Her ne kadar taşınabilir başlı başına programlar üretse de, bu programların çalıştırılmasında bazı dinamik bağlantı kütüphanelerine (dynamic link library, dll) ihtiyaç duyulmaktadır. Bu nedenle, MATLAB ile üretilen bir programın, ileride çıkarılan yeni sürümlerle çalışmaması ya da dinamik kütüphane problemi yaşaması ihtimali vardır. MATLAB ile üretilen programların bilinen bir diğer eksikliği de hızlarıdır. MATLAB kütüphaneleri birçok işi yapabilecek seviyede tasarlandıklarından, hızdan feragat edilmiştir. İşte bu bahsedilen nedenlerle, programlama platformu olarak MATLAB kullanılmamasına karar verilmiştir.

RCSP programında tasarım ve geliştirme ortamı olarak Microsoft Visual Studio .NET 2003 kullanılmıştır. Visual Studio ailesinin 6.0 versiyonundan sonra çıkan .NET platformu birçok kütüphaneyi de beraberinde getirmektedir. Bu .NET standardı kütüphaneler sayesinde, üretilen kodun anlaşılırlığı artmıştır. Ayrıca .NET Form arayüzü elemanları kullanıldığından, RCSP ekranları da kolay anlaşılır hale gelmiştir. Bu platformun sağladığı kolaylıklar sayesinde kod üretimi süreci de kısalmıştır. Programlama dili olarak, nesne tabanlı programlamayı destekleyen yüksek seviye dillerden biri olması nedeniyle C++ tercih edilmiştir. Yine .NET platformunun bir özelliği olarak sunulan atık toplama (garbage collection, __gc) özelliği de kullanılmış, bu sayede programdan kaynaklanabilecek olası hafıza kaçaklarının (memory leakage) önüne geçilmiştir.

3.2.3. Kod yapısı

Tamamı tez kapsamında üretilen kodlar, teze özgü isimlendirme sistemine uygun olarak hazırlanmıştır. Bu isimlendirme sisteminde her bir sınıf adı “Crcsp” ön eki ile başlar. Bu sayede, kod içinde farklı kütüphanelerden kullanılan nesneler ile özgün olarak üretilen sınıflara ait nesneler ayırt edilebilir. Sayım listelerine de

“Ercsp” ön eki getirilir. Ayrıca standart bir yöntem olan, üye değişken ve fonksiyonların isimlerine “m” ön eki ile başlanılması yöntemi de kullanılmıştır. Bu kullanım da, kod içinde kullanılan bir değişkenin bir nesneye mi ait olduğu yoksa geçici bir değişken mi olduğu konusunda bilgi vericidir. Kodlamada uyulan bir başka kural da fonksiyon isimlerinin büyük harflerle başlamasıdır. Bu isimler, her bir kelime yine büyük harfle başlayacak şekilde devam eder.

Tekrar kullanılabilirliğin genel bir uygulaması olarak, kodlanan her bir sınıf için o sınıfın adına uygun isimle bir dosya yaratılmıştır. Birden fazla sınıfın benzer amaçlarla üretildiği durumlarda ise bu sınıflar ortak amacı gösterir biçimde isimlendirilmiş olan tek bir dosyada toplanmıştır. Her iki durumda da dosya isimlerinin “rcsp” ön eki ile başlamasına dikkat edilmiştir.

Grafik kullanıcı arayüzü kullanılan programlar için iki ayrı kod bölümü bulunması uygundur. Bu bölümlerden biri arayüz ilgili işlemleri yapar. Diğer bölüm ise programdan beklenen asıl işlemlerle ilgilenir. Arayüz bölümünün yapması gereken, kendisine kullanıcı tarafından verilen bilgiler ışığında diğer bölümden uygun fonksiyonları işletmektir. Ana işlem bölümü de arayüz bölümünün ihtiyaç duyduğu fonksiyonları sağlamakla yükümlüdür. Örneğin bir hesap makinesi programında, arayüz bölümü kullanıcıya rakamları ve işaretleri gösterip bunlardan birinin seçilmesini sağlar. Ana işlem bölümü de kendisinin yapacağı işlemlerde kullanacağı sayıları öğrenmek için bir fonksiyon sunar. Bu sayede, asıl işi yapan bölüm aynı kalmak üzere farklı arayüzlerle programın yeniden derlenmesi mümkün olabilir.

RCSP programında, Microsoft Visual Studio .NET platformunun sunduğu proje tiplerinden “Windows Forms Application” kullanılmıştır. Bu proje tipinin arayüz bölümü olarak sunduğu Form sınıfı ise tezde benimsenen isimlendirme standardına uygun olarak CrcspForm şeklinde kullanılmıştır. Arayüz bölümünde, geliştirme ortamının sunduğu görsel form tasarım aracı kullanılarak standart olarak sunulan görsel elamanlardan oluşan bir form hazırlanmıştır. Bu formun ayrıntılı ekran görüntüleri kullanım bilgileri bölümünde (Bölüm 3.3) verilecektir. Ana işlem bölümü olarak da CrcspMain sınıfı tasarlanmıştır. Arayüz her türlü işlemi bu sınıf aracılığı ile yaptırır ve dolayısıyla CrcspMain sınıfının

sunduklarından başka bilgiye sahip değildir. Kodun diğer bölümleri ise CrcspMain sınıfının kullanımında ve denetiminde olan kısımlardır.

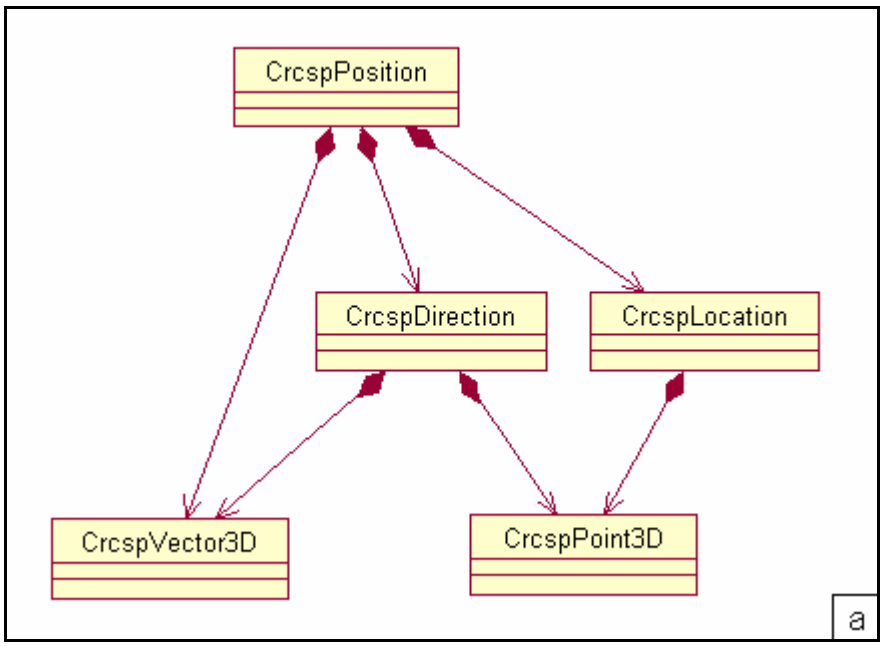
C++ dilinde kodlar .h (ya da .hpp) ve .cpp uzantılı dosyalarda bulunur. Tüm RCSP kodu da Çizelge 3.4'te listelenen isimlerdeki .h ve .cpp dosyalarında yer almaktadır. Dosyalarda tanımlanmış olan sınıflar ve sayım listeleri de aynı çizelgede belirtilmiştir.

Çizelge 3.4. RCSP kodunun yer aldığı dosyalar ile bu dosyalarda tanımlamış sınıflar ve sayım listeleri.

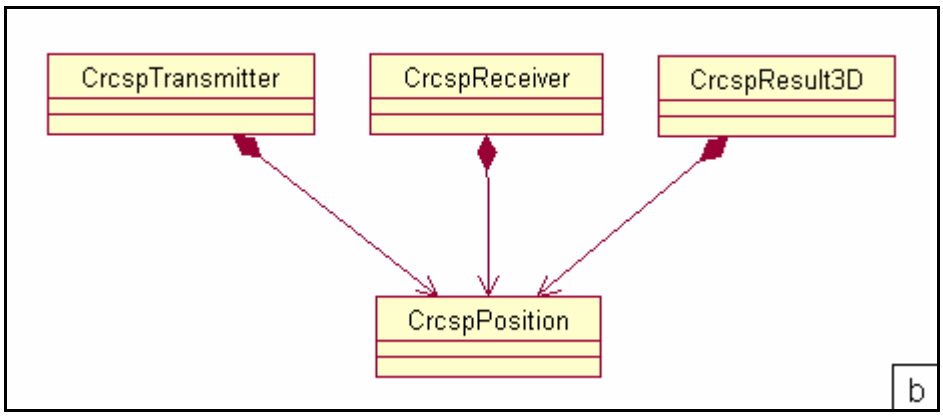
Dosya adı (.h ve .cpp)	**Sınıflar**	**Sayım listeleri**
rcspArrayList	CrcspArrayList, CrcspFacetArrayList, CrcspVertexArrayList	–
rcspColorScale	CrcspColorRgb, CrcspColorScale	–
rcspCommon	CrcspDirectivityTable	ErcspPredefinedPositions, ErcspDirectivityFunction, ErcspDimensionUnit
rcspFacet	CrcspFacet	–
rcspForm	CrcspForm	–
rcspGeometrics	CrcspPoint3D, CrcspVector3D, CrcspDirection3D, CrcspLocation3D, CrcspPosition3D, CrcspPlane3D	–
rcspMain	CrcspMain	ErcspGlWinId, ErcspMotionIndex, ErcspTxRxViewType
rcspOpenglWindow	CrcspOpenglWindow	–
rcspRcsCalculator	CrcspCallbackFunc, CrcspCalculatorOuts, CrcspRcsCalculator	ErcspRadarType, ErcspAnalysisType
rcspReceiver	CrcspReceiver	–
rcspResult3d	CrcspResult3d	–
rcspStlFileParser	CrcspStlFileParser	–
rcspTarget	CrcspTarget	–
rcspTransmitter	CrcspTransmitter	–
rcspVertex	CrcspVertex	–

3.2.4. Kod etkileşimleri ve işleyişi

Kod işleyişi, nesne tabanlı programın gereği olarak nesnelerin birbiri üzerinden ihtiyaç duydukları fonksiyonları çağırmaları şeklinde olmaktadır. Burada bir nesnenin hangi nesnelerle haberleşebileceği önemlidir. Kurulan bu hiyerarşik yapı Şekil 3.19 ile daha kolay anlaşılabilir. Şekilde gösterilen kutucuklar sınıfları tanımlamaktadır. Aralardaki bağlantılar da standart UML (unified modeling language) gösterimine uygun olarak nesne sahipliği ve kullanımı bilgilerini vermektedir.

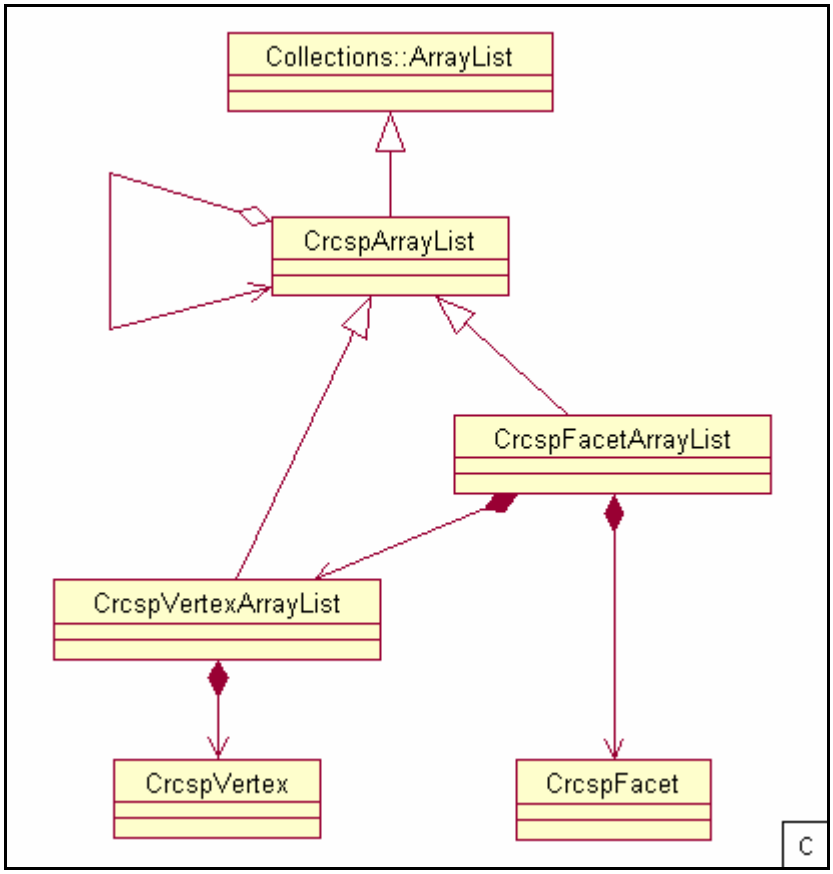


1. Konum sınıfı etkileşimleri.

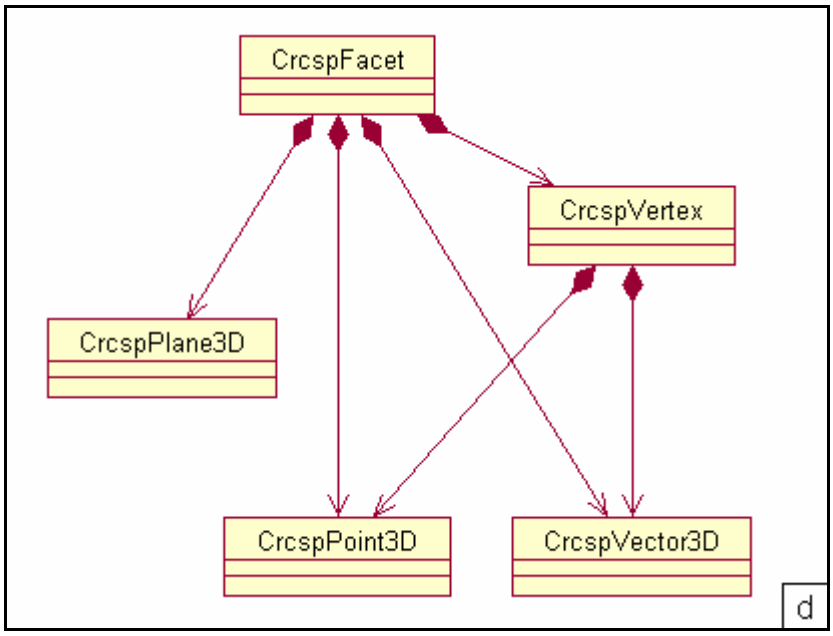


2. Verici, alıcı ve 3D sonuç sınıfları etkileşimleri.

Şekil 3.19. Sınıflar ve aralarındaki etkileşim.

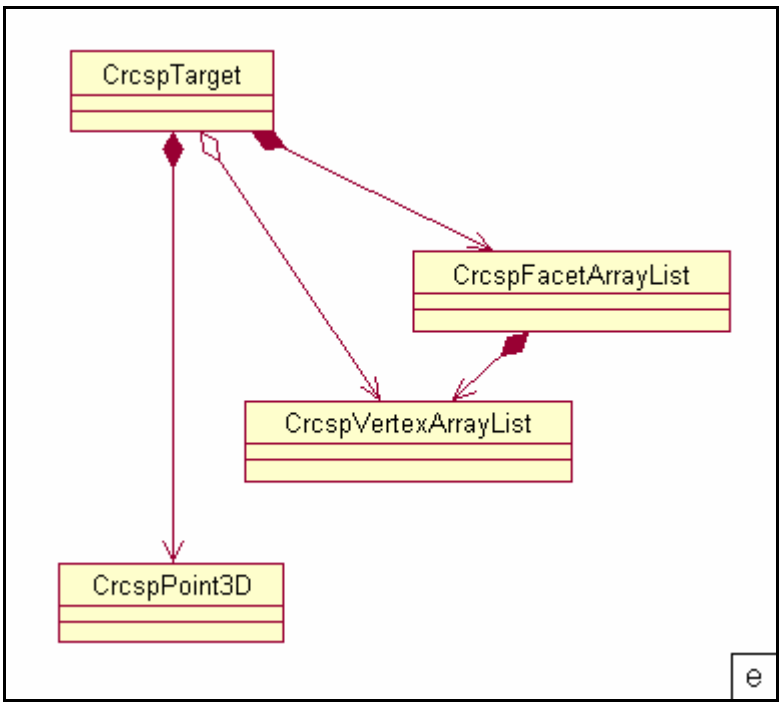


3. Yüzeycik ve köşe liste sınıfları etkileşimleri.

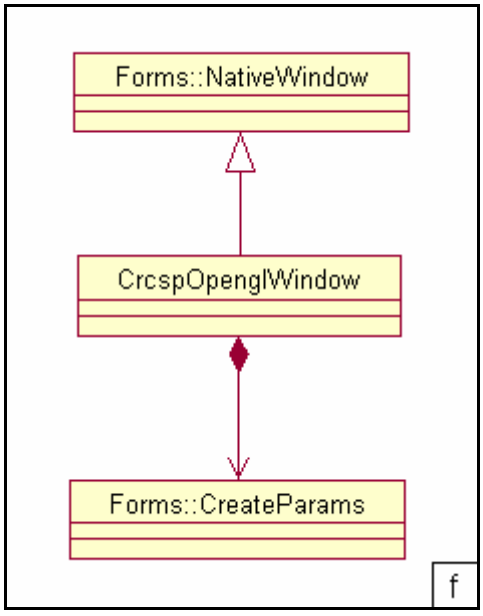


4. Yüzeycik ve köşe sınıfları etkileşimleri.

Şekil 3.19. devam ediyor.

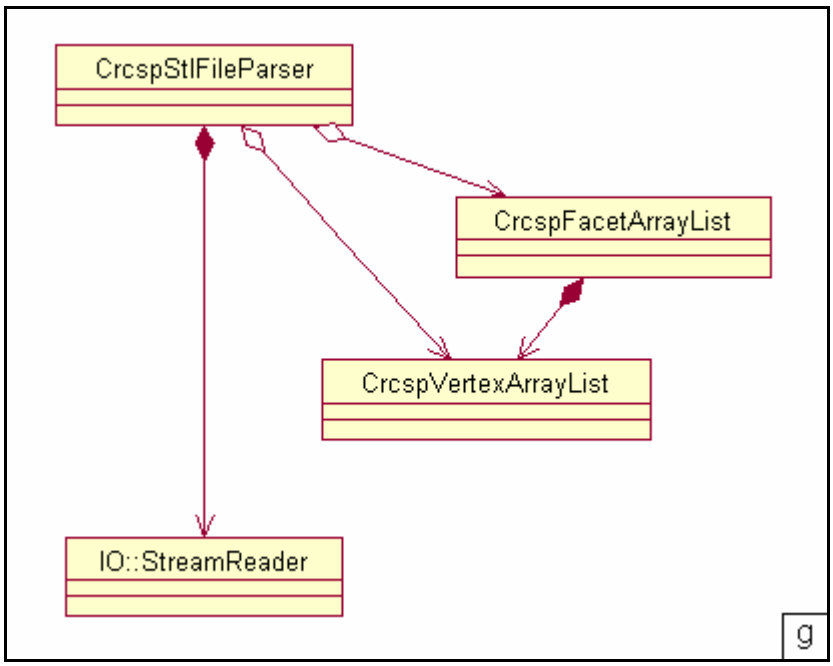


5. Hedef sınıfı etkileşimleri.

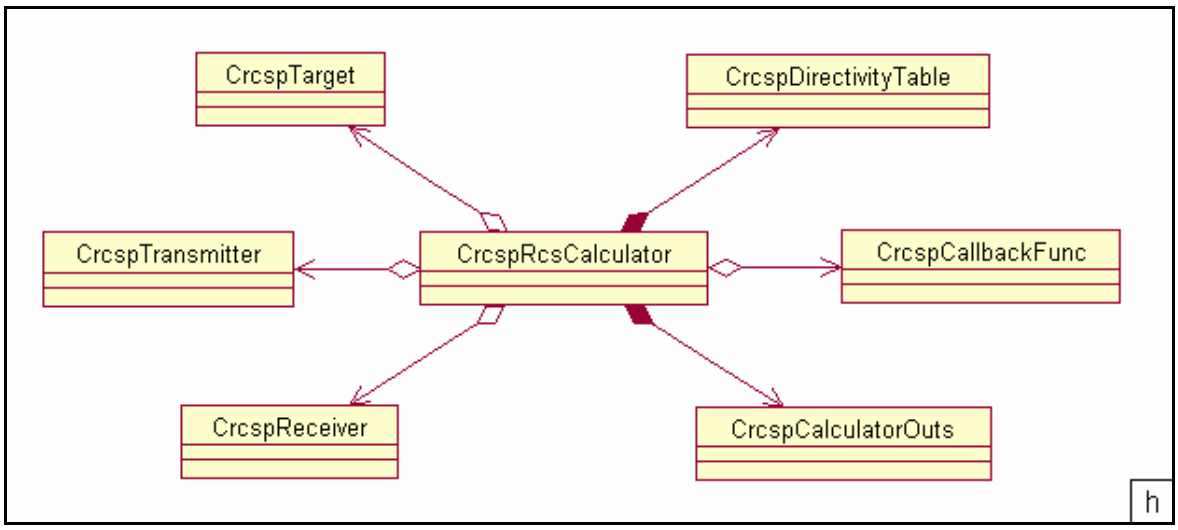


6. OpenGL çizim sınıfı etkileşimleri.

Şekil 3.19. devam ediyor.

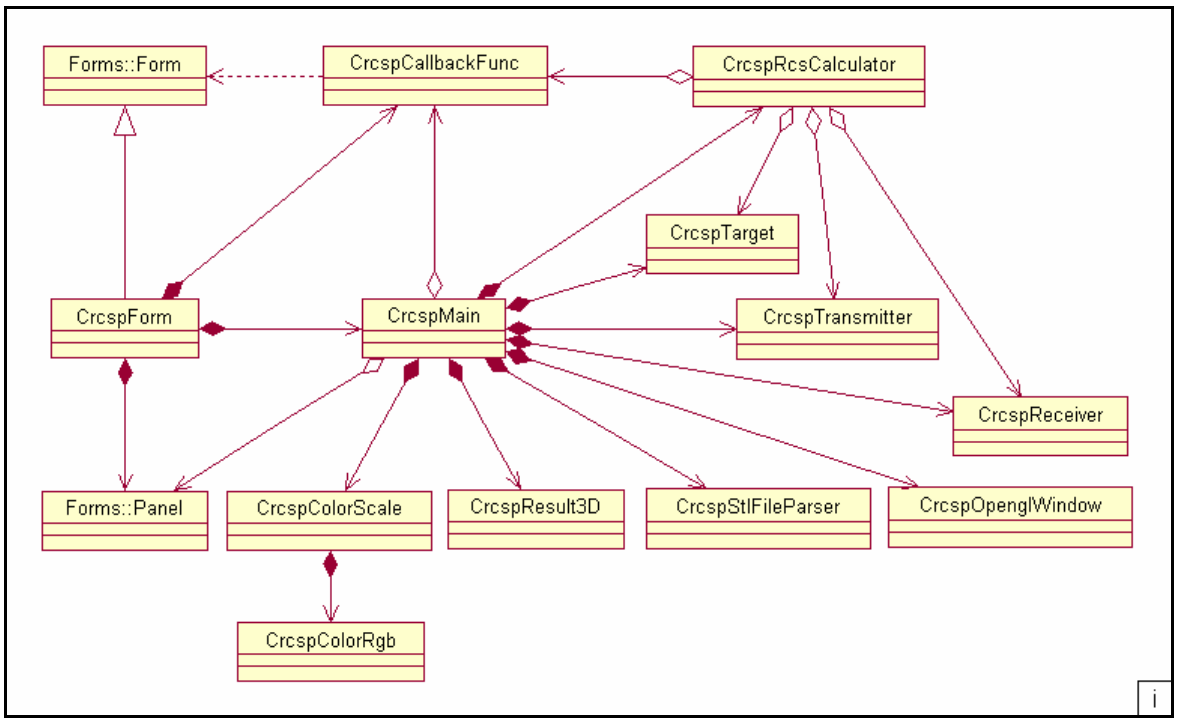


7. STL dosya okuma sınıfı etkileşimleri.



8. RCS hesaplama sınıfı etkileşimleri.

Şekil 3.19. devam ediyor.



9. Ana işlem ve arayüz sınıfları etkileşimleri.

Şekil 3.19. devam ediyor.

Programda, bir çok durumda üç boyutlu kartezyen uzayındaki konumların kullanılması gerekmiştir. Bu konum bilgisi bir sınıf içinde toplanarak daha fazla bilgi içerecek hale getirilmiştir. CrcspPosition3D sınıfı bu amaçla üretilmiştir ve her türlü cisim için konum bilgisini tutar (bkz. Şekil 3.19.a). Tam olarak cismin yerini, baktığı yönü ve yukarı yön vektörünü içermektedir. Bu bilgilere alıcı, verici, hedef gibi cisimlerin konumu ile üç boyutlu görüntülerdeki kamera konumunu belirlemede ihtiyaç duyulur.

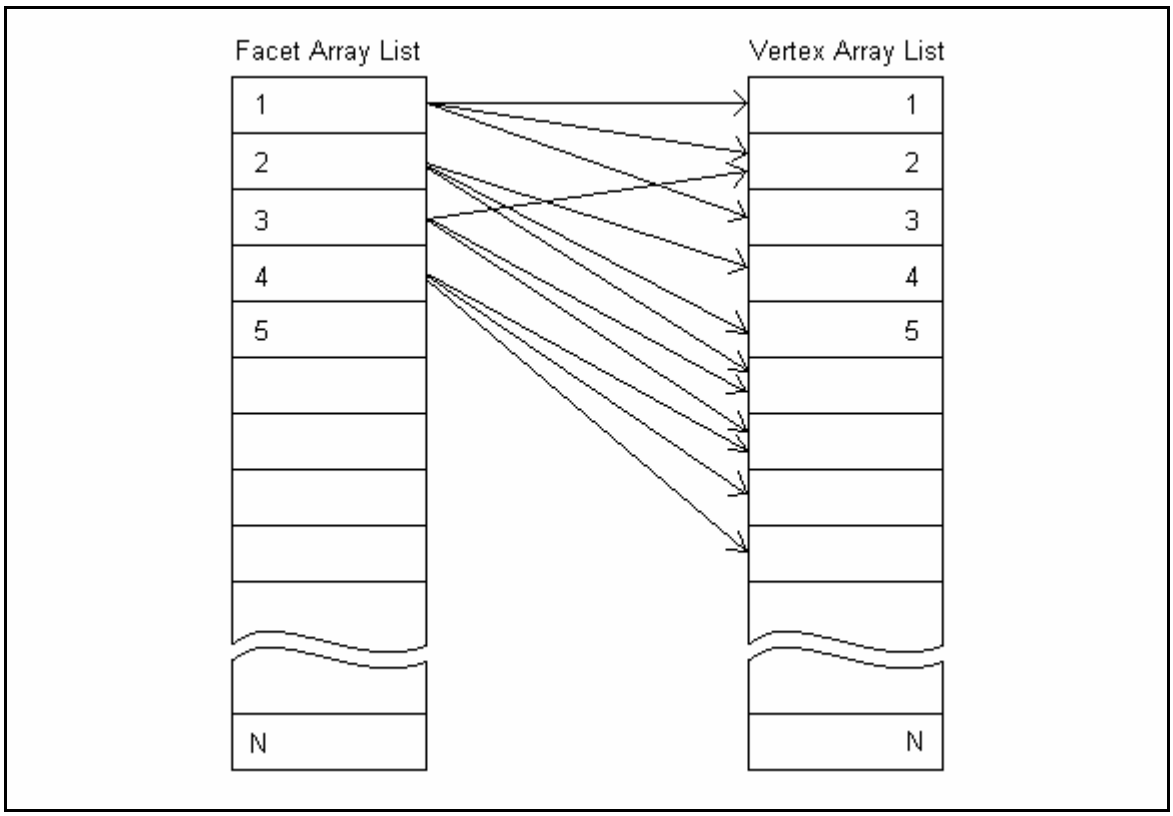
CrcspTransmitter ve CrcspReceiver sınıfları, radar vericisi ve alıcısı için gerekli olan bilgileri tutar (bkz. Şekil 3.19.b). Bunlar ilk tasarımda konum olarak sınırlı kalmıştır ancak alıcı ve verici ile ilgili polarizasyon gibi bilgilerin de tutulması gerektiğinde hazır bir veri yapısı olarak kullanılabilir.

CrcspArrayList sınıfı bir soyut sınıftır. Bu sınıftan üretilmiş bir nesne bulunmamakta, ancak bu sınıftan türeyen başka sınıflara ait nesneler kullanılmaktadır. CrcspArrayList sınıfının kendisi de aslında türemiş bir sınıftır. Mevcut .NET kütüphanelerinde bulunan Collections::ArrayList sınıfını baz olarak

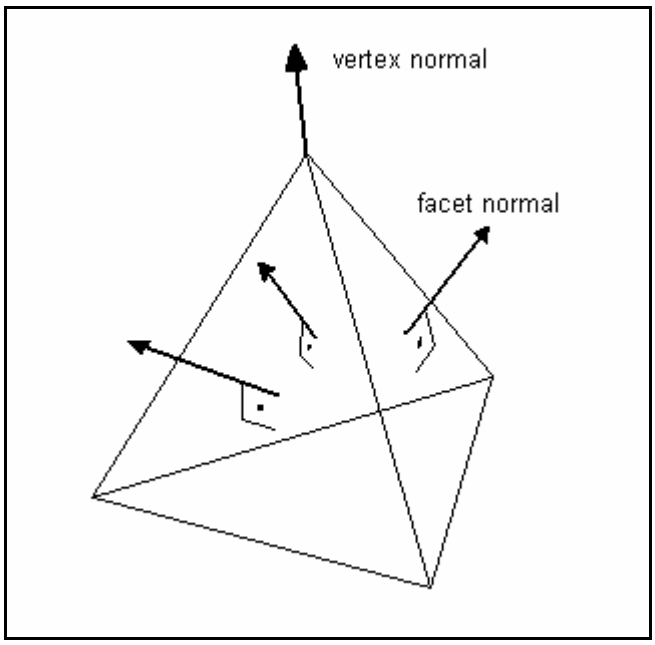
alır. Bu sayede listeye ekleme, listeden çıkarma, listeyi sıralama gibi işlemler hazır kütüphaneden kullanılmış ve hata payı azaltılmıştır. RCSP programı için ihtiyaç duyulan iki liste sınıfı CrcspFacetArrayList ve CrcspVertexArrayList sınıflarıdır. Bunlar, programa girdi olarak verilen ve hedef cismin yüzey bilgilerini içeren STL dosyasından (bkz. Bölüm 3.1.1) gelen bilgileri tutarlar (bkz. Şekil 3.19.c). Hedef cisim için STL dosyasında yüzeyciklerden oluşan bir liste verilmiştir. Yüzeyciklerin her biri içinse üç adet köşe koordinatı verilmiştir. Bu dosya yapısındaki bilgi tekrarını bertaraf etmek için ikili liste yapısı kullanılmıştır. Burada CrcspVertexArrayList tekrarlanmayan biçimde köşeleri belirlerken, CrcspFacetArrayList de yüzeycik bilgilerini diğer listeye referanslar vererek tutar.

Liste sınıflarında tutulan nesneler CrcspFacet ve CrcspVertex sınıflarındandır. Listeler arasındaki referans verme işlemi de bu sınıflara ait nesneler üzerinden olur. Buradaki yapıda, CrcspFacet sınıfından bir nesnenin, CrcspVertex sınıfından üç nesne için referans bilgisini tutması esastır (bkz. Şekil 3.19.d). Böylece yeni üretilmiş ya da başka yüzeycikler tarafından kullanımda olan köşe nesnelerinden üçünü kullanan bir yüzeycik nesnesi oluşturulur (Şekil 3.20). CrcspFacet sınıfının tuttuğu diğer bilgiler ise yüzey normal vektörü, yüzeycik merkezinin koordinatı, yüzeyciğin içinde bulunduğu düzlem denklemi ve yüzeyciği oluşturan kenar vektörleridir. CrcspVertex sınıfının tuttuğu bilgi, o köşenin koordinatları ve köşe normalinden ibarettir. Köşe normali, o köşeyi paylaşan yüzeyciklerin normallerinin vektörel ortalamasıdır (Şekil 3.21). Bu bilgi OpenGL ile yapılan çizimlerde cismin yüzey devamlılığını sağlayabilmek için kullanılmaktadır. OpenGL ile bir yüzeycik çizilirken, her köşe için bu normal vektörü değerleri verilerek işlem yapılır.

CrcspTarget sınıfı hedef ile ilgili olan verileri tutar (bkz. Şekil 3.19.e). Şekil bilgisi yukarıda açıklandığı biçimde yüzeycik ve köşe listeleri halinde saklanır. Ayrıca hedef cismin uzandığı en büyük ve en küçük koordinat değerleri de burada tutulur. Daha sonra bu değerler kullanılarak cismin merkezi koordinat eksenleri merkezine oturtulur. Bu sınıfın sakladığı diğer bilgi de yüzey özellikleridir. Tüm cisim için aynı yüzey özelliği tanımlanmakta olup bu bilgiler kullanıcı arayüzünden alınmaktadır. Daha sonra RCS hesaplanırken de yüzeyin sinyal yansıtma oranı ya da saçınım katsayısı gibi özellikler bu sınıftan sorularak kullanılır.



Şekil 3.20. Yüzeycik listesi ile köşe listesi arasındaki yapı.



Şekil 3.21. Köşe normali ve yüzeycik normali kavramları.

OpenGL ile ilgili ayarlamaların yapıldığı ve bazı OpenGL fonksiyonlarının paketlenerek sunulduğu sınıf CrcspOpenglWindow'dur. Bu sınıf [25]'ten alınan proje örneğine göre üretilmiştir. Baz olarak kullanılan Forms::NativeWindow

sınıfının özellikleri sayesinde, OpenGL aygıt bağlamı ve resmetme bağlamlarını kullanarak rcspForm arayüzünde çizim yapmaya imkan verir. Bu sınıf, sanal bir form oluşturmakta ve bunun için de Forms::CreateParams sınıfından bir nesne kullanmaktadır (bkz. Şekil 3.19.f).

CrcspStlFileParser sınıfı, RCSP programına girdi olan STL dosyasından bilgi ayıklanması işini yapar. Girdi dosyasının adı kullanıcı arayüzünden girildiğinde rcspMain nesnesine bilgi gelir. Daha sonra CrcspStlFileParser dosyayı okumak üzere hafızaya yükler. Eğer dosya uygun bir STL formatına sahipse okuma yapılır. Aksi durumda işlem yapılamaz. Burada, benzer şekilde sınıflar kullanarak başka tipte dosyaların da okunmasına imkan verecek eklemeler yapılabilir. Dosya okunmaya başlandıktan sonra bir hata ile karşılaşılırsa işlem tamamlanamaz. CrcspStlFileParser, dosya tipinin ASCII ya da binary olmasına göre uygun kodu çalıştırmaktadır. Dosyadan okunan hedef cisim bilgileri, bu sınıfa geçici olarak sunulmuş CrcspArrayList işaretçileri (pointer) aracılığı ile tutulan listelere eklenir (bkz. Şekil 3.19.g). Köşe bilgileri listeye eklenirken liste kontrol edilir ve aynı köşeye ait bilginin liste içinde tekrar yer alması önlenir. Bir köşe zaten listede bulunuyorsa yeni bir yüzeycik tarafından da kullanılıyor demektir. Bu durumda köşe normali de o köşeyi paylaşan yüzeyciklerin normalleri kullanılarak güncellenir. Burada geçici işaretçilerle erişilerek içleri doldurulan listeler aslında CrcspTarget nesnesinin sahip olduğu nesnelerdir. Dosya okuyucu işini bitirdiğinde, hedef cisim nesnesine yeni şekil bilgileri girilmiş olur.

CrcspRcsCalculator sınıfı en son işlemi yapan bölümdür. Etkileşimleri, verilen tüm bilgiler ışığında hesaplamaları tamamlaması için tasarlanmıştır (bkz. Şekil 3.19.h). Kullanıcı arayüzünden istek geldiği zaman rcspMain tarafından çalıştırılır. Çalışması sırasında yapılan ve kalan iş miktarını bildirmek üzere CrcspCallbackFunc nesnesi aracılığı ile arayüzle haberleşir. Hesaplamalarda ihtiyaç duyacağı bilgiler gerek CrcspMain üzerinden gerekse CrcspForm nesnesine verilen geçici işaretçi ile doğrudan kendisine bildirilir. Bu bilgiler hesaplamanın hangi pozisyonlar için yapılacağı, kaç farklı konumda bilgi toplanacağı ya da ne kadar hassasiyet beklendiği gibi kriterlerdir. Bunun yanında hedef, verici ve alıcı nesnelerinin geçici işaretçileri de hesaplamalarda kullanılmak üzere bu sınıfa verilir. Hesaplamalar sırasında hedefin yüzey özellikleri de dikkate

alınır. CrcspTarget ile verilen yüzey saçınım özelliğinin sayısal değerlere dönüştürülmesi için CrcspDirectivityTable sınıfı kullanılır. Hesaplamalar sonrasında sonuçların arayüze aktarılabilmesi için de CrcspCalculatorOuts veri yapısı kullanılır. Bu yapıda, hesaplama yapılan konum vektörü ile o konumdan alınan sonuç listelenir. Arayüz, işlem tamamlandığında bu veri yapısından okuduğu değerler ile sonuç grafiklerini ve tablolarını oluşturur.

CrcspDirectivityTable sınıfı, yüzeyin saçınım özelliğini taklit etmek için kullanılmıştır. Bu sınıf, farklı yönlülük fonksiyonları (directivity function) için 0-90 derecelik açı bölgesinde yüzeyin yansıtma oranlarını veren tablolardan oluşur. Buradaki açı değeri, o yüzeyden sinyalin Snell yasasına (bkz. [15]) göre yansıması gereken yön ile gerçek yansımanın olduğu varsayılan yön arasında kalan açıdır (Şekil 3.22). Diğer bir deyişle yüzeyden ana eksen yansıması yapan sinyal doğrultusu ile ölçüm yapılan doğrultu arasındaki farktır. Örneğin yüzey normalinden aydınlatılmışsa beklenen yansıma yönü de normal doğrultusudur. Ancak ölçüm alınmak istenen doğrultu yüzey ile 45 derecelik açı yapıyorsa, bu konumdan okunacak yansıma değeri yönlülük fonksiyonunun 45 için verdiği değer olur.

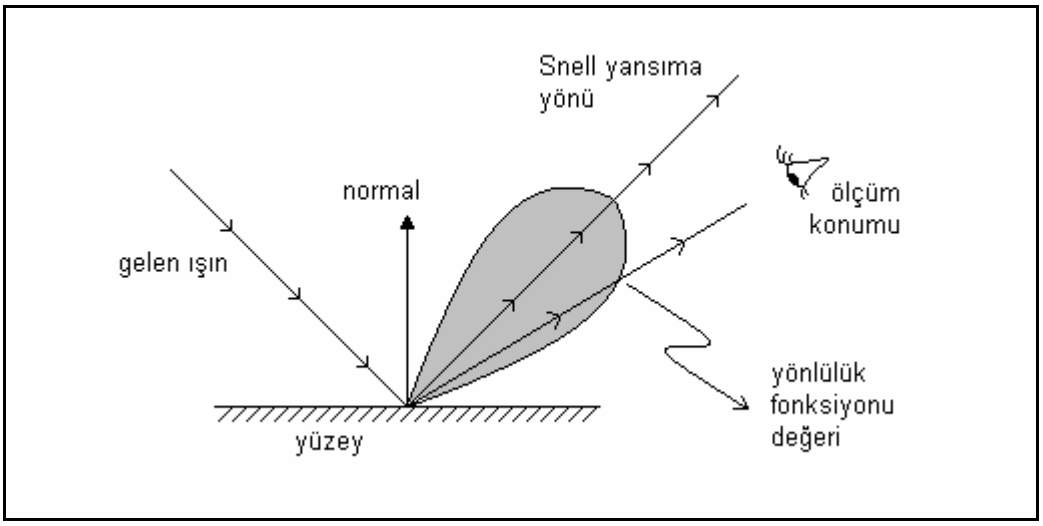
Yönlülük fonksiyonu olarak

- Impulse (vuru)
- Square (kare)
- Triangular (üçgen)
- Gaussian (Gauss)
- Cosine (kosinüs)
- Sinc (sinüs kardinal)

kullanılmıştır. Bu fonksiyonların on seviye farklı yüzey saçınım katsayıları için aldıkları değerler de EK 4'te verilmiştir.

Bir yüzey için yüzey saçınım katsayısı yüksek seçilmişse o yüzey pürüzlü yapıya sahip demektir. Böyle yüzeylerde gelen sinyalin her yönde yansıtılma olasılığı vardır. Bu durumda yönlülük fonksiyonundan okunan değerler büyük açılar için bile kabul edilebilir seviyenin üzerinde olur. Ancak yüzey pürüzsüz seçilmişse ve saçınım katsayısı küçük ise, büyük açılar için yönlülük değerleri çok düşük olur. Buradan bulunacak değerler o yüzeyden dönecek sinyal seviyesini

etkileyeceğinden (bkz. Eş. 2.1) RCS hesabında kullanılır. Yüzey için yönlülük katsayısının yanı sıra soğurma katsayısı da tanımlanmıştır. Bu katsayı, yüzde olarak o yüzeyin gelen sinyalin ne kadarını soğurduğunu ne kadarını ise yansıttığını verir. Bu değer de RCS'yi etkiler.

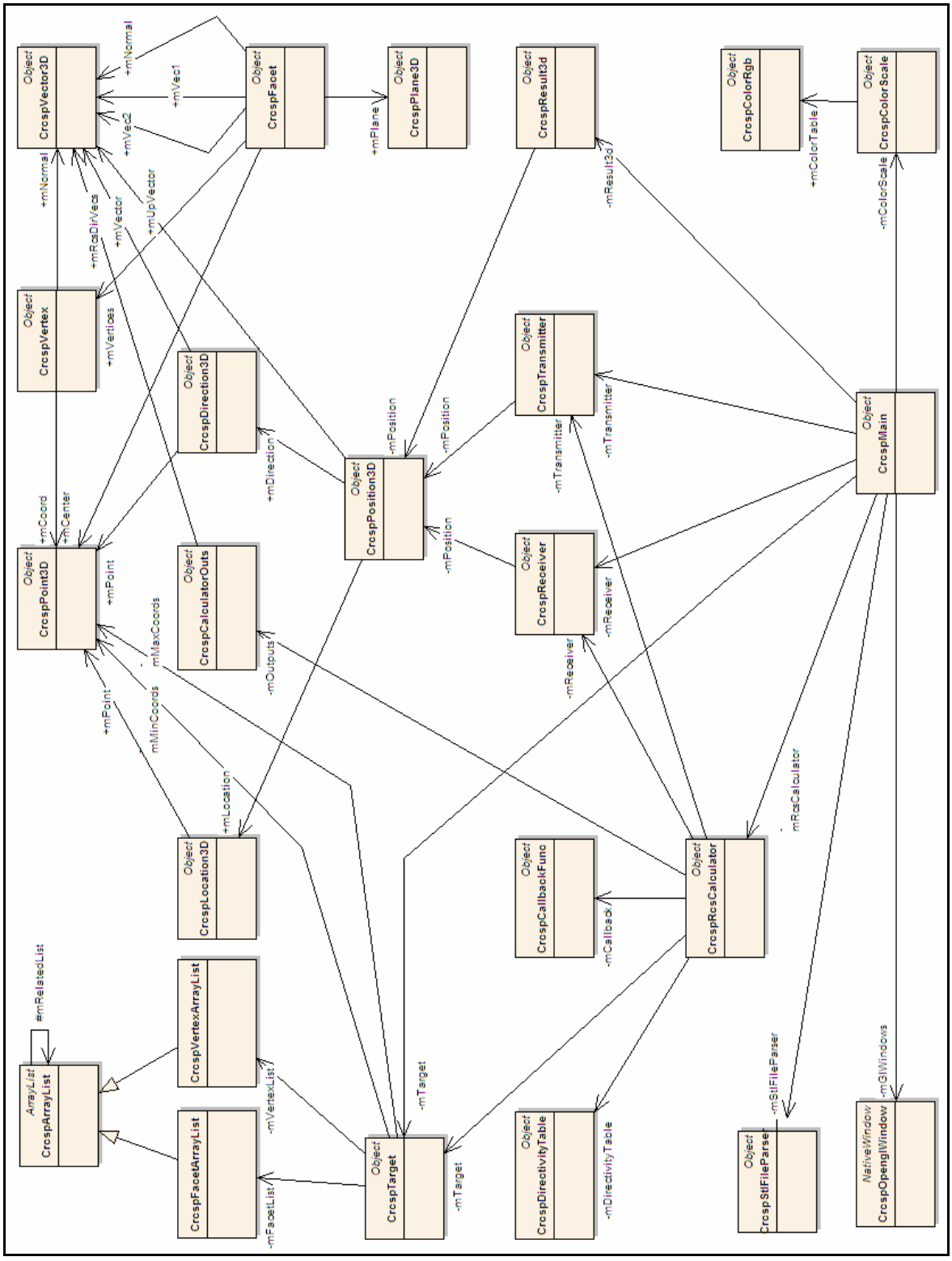


Şekil 3.22. Yönlülük fonksiyonu kavramı.

Buraya kadar bahsedilen arayüz dışındaki tüm sınıf ve yapılar CrcspMain sınıfı kontrolündedir. CrcspMain nesnesi arayüz tarafından yaratılır ve tüm diğer gerekli sınıfları yaratarak onların nesnelerini tutar (bkz. Şekil 3.19.i). Bu sınıfın temel işlevi, arayüz ile diğer birimler arasındaki iletişimin sağlanması ve birimlerce yürütülecek işlerin kontrol edilmesidir. Bu nedenle neredeyse her sınıf ile iletişim içindedir.

Tüm bu sınıflar arasındaki etkileşimi ve sınıfların sahip oldukları üye değişken ve üye fonksiyonları, detaylı olarak sunan bir kod özeti hazırlanmıştır. Bu özetin çıkarılmasında, Sparx System firmasının bir ürünü olan Enterprise Architect 6.5 programı kullanılmıştır. Bu program, en son ürün halindeki çalıştırılabilir RCSP dosyasını alarak dosya içindeki bilgilerden tasarım yapısını çıkarabilmektedir. Bu sayede, başlangıçtaki tasarım beklentilerinin oluşturulan program tarafından karşılanıp karşılanamadığı da görülmektedir. Ayrıca son ürüne ait bir çok bilgiyi barındıran bir doküman da üretilmiş olmaktadır. Oluşturulan özet, Şekil 3.19'daki UML gösterimlerine benzer fakat daha detaylı çizimler (Şekil 3.23) içeren bir internet sayfası halindedir. Bu sayfada sınıflar hakkında bilgiler de

listelenebilmektedir. Bu özet dokümanına, tezin giriş bölümünde belirtilen internet adresinden (www.ayanli.com/yukseklisans) erişilebilir.



Şekil 3.23. Tüm sınıflar arası etkileşim.

Bu bölümde verilmiş olan bilgiler, RCSP programının tasarımını ve işleyişini özetlemektedir. Program kullanımına dayalı bir sonraki bölümdeki (Bölüm 3.3) anlatım sırasında da yeri geldikçe tasarım ve kod işleyişi konularında açık kalan noktalara değinilecektir.

3.3. Kullanım Bilgileri ve Çalışma Şekli

Tez kapsamında üretilen programdan beklenen nihai çıktı RCS değeridir. Kullanıcı bu sonuca erişmek için birkaç adımdan oluşan işlemleri gerçekleştirmek durumundadır. Bu işlemler, kullanım kolaylığını esas alacak ve karışıklığa sebep olmayacak biçimde gruplanarak kullanıcıya sunulmuş olmalıdır. RCSP programında bu gruplama sekme (tab) yapısı kullanılarak yapılmıştır. Sekme, aynı ekranı farklı durumlar için ihtiyaç duyulan farklı görsel elemanlar ile kullanmayı sağlayan yapıdır. Kullanılan sekmeler, yine kolaylık sağlaması açısından işlem sırasına uygun olacak biçimde program ekranının üst bölümünde sıralanmıştır. Program açılışında, ilk sekme aktif olur. Kullanıcı her bir sekmede yapması gerekenleri bitirdikten sonra bir sonraki sekmeye geçerek işlemi sürdürür.

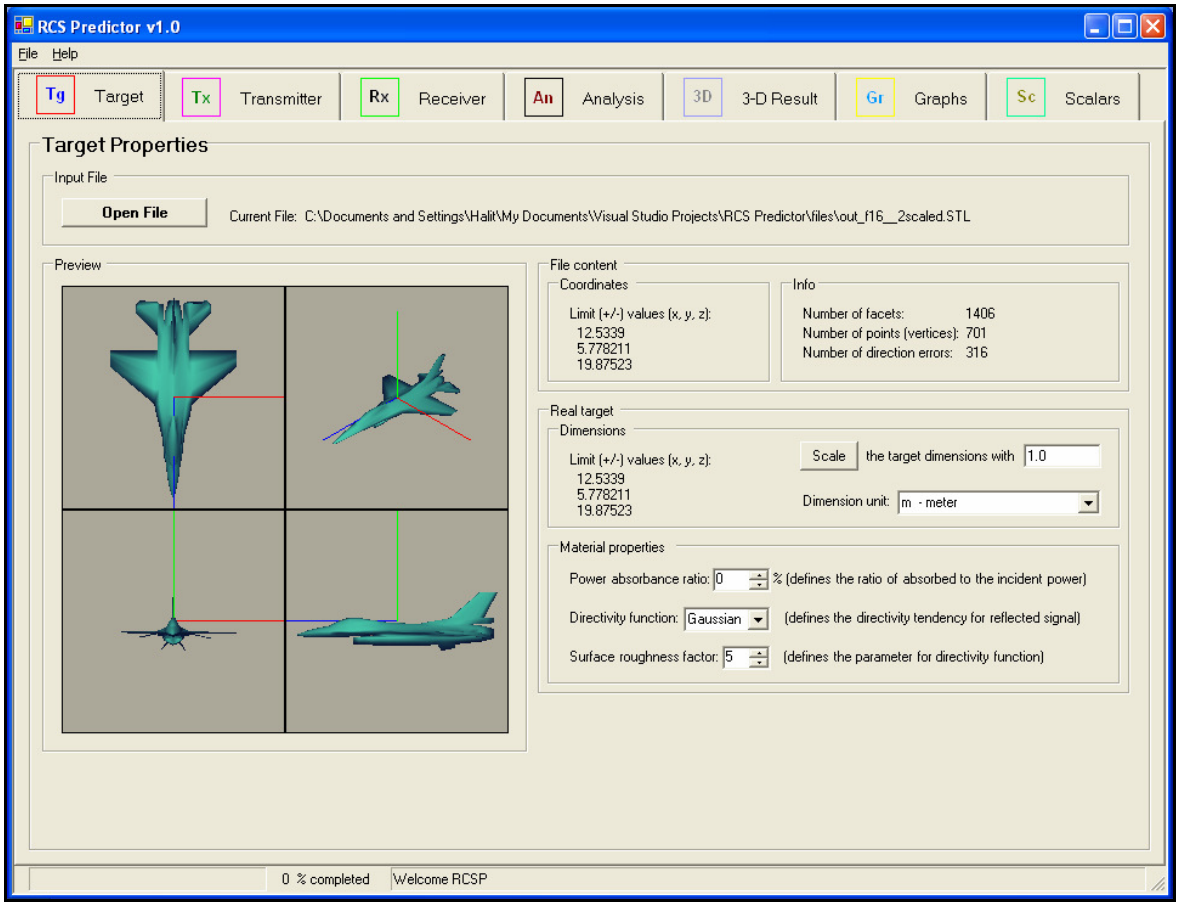
Sekmelerin isimleri sırasıyla

- Hedef için “Target (TG)”
- Verici için “Transmitter (TX)”
- Alıcı için “Receiver (RX)”
- Analiz için “Analysis (AN)”
- Üç boyutlu sonuç için “3-D Result (3D)”
- Grafik sonuç için “Graphs (GR)”
- Sayısal sonuç için “Scalars (SC)”

şeklindedir. Burada kullanılan kısaltmalar, kod içinde de o sekmeye ait değişkenlerin isimlerinde ön ek olarak kullanılmıştır. Bu sayede değişkenlere kod içinde kolay erişim sağlanmıştır.

Aşağıdaki alt başlıklarda her bir sekme için ekran görüntüleri verilmektedir. Bu başlıklar altında RCSP programı kullanımı, kullanıcı tarafından yapılması gerekenler, programdan beklenenler ve bunların nasıl yerine getirildiği gibi konularda açıklamalar yer alacaktır.

3.3.1. TG ekranı



Şekil 3.24. TG ekranı.

RCSP programı açıldığında, kullanıcının ilk gördüğü ekrandır. Bundan sonraki tüm adımlarda kullanılacak olan hedef cisim burada tanımlanır. Bunun için kullanıcının yapması gereken, daha önceden CAD programlarından birini kullanarak hazırlamış olduğu çizimin uygun formattaki bilgileri içeren dosyasını programa tanıtmaktır. “Open File” butonuna basıldığında açılan pencereden bu dosya işaret edilerek yükleme yapılır. Dosya yükleme işlemi sorunsuz olarak tamamlandığında, kullanıcıya bilgi vermek amacıyla o an yüklü bulunan dosyanın yeri ve adı “Current File: …” şeklinde ekrana yerleştirilir.

Dosya yükleme işleminde, yukarıda bahsedilen CrcspStlFileParser sınıfı çalışır. Sonuç olarak çıkan yapı, yüzeycik ve köşe listeleri doldurulmuş olan bir hedef cisim nesnesidir. Yine bilgi vermek amacıyla, listelerdeki toplam yüzeycik ve köşe sayısı da ekrana yansıtılır. Ayrıca, cismin üç ana koordinat ekseni boyunca kaç birim uzunluğunda olduğu bilgisi de verilir. Bu limit değerler, istenirse kullanıcı

tarafından “Scale” butonu ile ölçeklenebilmektedir. Hedef cismin dosyasından okunan her türlü sayı birimsizdir. Bu sayıların ne ifade ettiğini kullanıcı bilmek ve programa bildirmek durumundadır. Örneğin yüklenen bir uçak şekli için limit değerler 15-20 mertebelerinde ise birimin metre olması muhtemeldir. Mil ya da santimetre gibi birimler bu sayılar ile anlamsız olacaktır. Yine aynı şekil için limitler 150-200 şeklinde verilmişse, kullanıcının 0.1 katsayısını ile ölçekleme yapma ihtiyacı duyacağı söylenebilir.

Yüklenen hedef cisim için yüzey bilgileri de tanımlanmalıdır. Yüzey maddesinin sinyali soğurma yüzdesi (absorbance ratio) 0-100 aralığında bir değerdir. Düşük soğurma oranı, yüksek yansıtma oranı demektir. Bu değer RCS hesabına doğrudan çarpan olarak etki eder. Cisim yüzeyi için saçınım özelliği de belirtilmelidir. Bu özellik yönlülük fonksiyonu ve saçınım katsayısı ile belirlenir. Bu fonksiyonun ve katsayının etkileri yukarıda CrcspDirectivityTable açıklanırken verilmiştir.

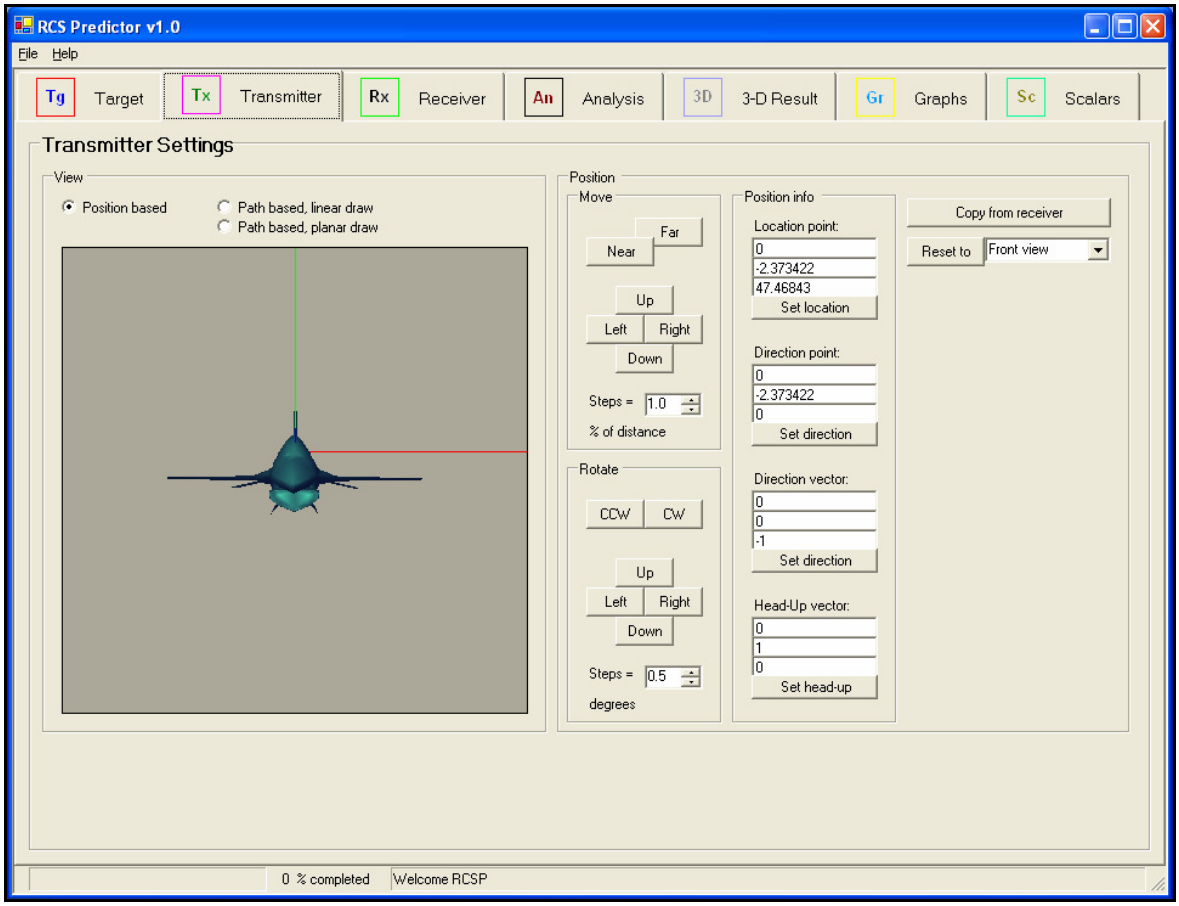
Bu ekranda dört adet de OpenGL görüntüsü bulunur. Bu görüntüler, yüklenen hedef cismin perspektif, üst, ön ve yan olmak üzere dört farklı açıdan çizilmiş şekilleridir. Bunlar üzerinde kullanıcının işlem yapmasına izin verilmez. Görüntülerin amacı, kullanıcıya yüklediği dosya içeriği hakkında ön bilgi vermektir. Yanlış bir cismin üzerinde işlem yapılmasını önlemeye yönelik bir özelliktir.

Yeni bir hedef cisim dosyası yüklendiğinde, diğer sekmelerce kullanılan ve daha önce yapılan hesaplamalardan kalmış görüntüler ve değişkenler de varsayılan ilk değerlerine çekilir.

Kullanıcı, dosyadan hedef cismi yükleyip gerekli ayarlamaları yaptıktan sonra radar alıcı ve verici ayarlarını yapmak üzere bir sonraki sekmeye geçer.

3.3.2. TX ekranı

Bu sekme radar vericisinin özelliklerini belirlemek ve ayarlamaları yapmak içindir. Daha önce de belirtildiği gibi, ilk tasarımda bu özellikler sadece konum ile sınırlı kalmıştır. Bu nedenle sekme ekranında yalnızca konum ile ilgili ayarlamalar vardır.



Şekil 3.25. TX ekranı.

Her yeni sicim yüklendiğinde, verici konumu da varsayılan değere çekilir. Bu varsayılan değer, cismin perspektif görüntüsünü verecek şekilde olan konumdur.

Ekranda yer alan OpenGL görüntüsü, hedef cismin radar vericisi konumundan görünen halidir. Bu görüntü üzerinde yer alan seçeceklerle konum bazlı ya da rota bazlı görünümler arasında geçiş yapılabilir. Konum bazlı görüntüde, kamera radar vericisinin konumundan görüntü verir. Rota bazlı görüntüde ise, vericinin konumuna bağlı olarak izlemesi beklenen dairesel yörünge ile bu yörüngeye göre hedefin durumu çizilir. Dairesel rota düzleminin normali, vericinin üst yön vektörü olacak şekilde yörünge oluşturulmuştur. Bu iki farklı görüntü seçeneği ile verici ve hedef arasındaki konumlama daha açık anlaşılmış olur.

Kullanıcı, verici konumunu sekiz farklı yönde yapabileceği hareketle belirler. Bu hareketler iki gruptadır. “Move” grubunda taşıma, “Rotate” grubunda döndürme işlemleri vardır. Taşıma işlemleri sola, sağa, yukarı, aşağı, uzağa ya da yakına olabilmektedir. Döndürme işlemleri de sola, sağa, üste, alta, saat yönünün tersine

ya da saat yönüne doğru yapılabilir. Her iki grup hareket için de kullanıcı adım aralığını yani hareket çözünürlüğünü “Steps” değeri ile belirleyebilmektedir.

Konum bilgisi CrcspPosition3D sınıfı ile tutulduğundan bu sınıfa ait bilgileri içerir. Bunlar yukarıda bahsedildiği gibi yer noktası, yön noktası ve vektörü ile üst yön vektörüdür. Verici için tutulan bu bilgiler, metin kutularında kullanıcıya sunulmuştur. Kullanıcı konum değiştirmek için konumlama butonlarını kullanabileceği gibi bu metin kutularında değişiklik yaparak da konum belirleyebilir.

Bunun yanında, kullanıcının konum belirlemek için başvurabileceği bir diğer yöntem de kopyalama ve sıfırlamadır. Kopyalama için kullanılan kaynak alıcı konumudur. Kullanıcı, alıcı ve vericiyi aynı konuma getirmek için bu özelliği kullanabilir. Sıfırlama için ise daha önceden belirlenmiş konumlar kullanılır. Bunlar standart olan perspektif, üst, alt, ön, arka, sol yan ve sağ yan görüntülerini veren konumlardır.

OpenGL görüntüsü üzerine sağ tıklandığında açılan yardımcı menü ile de konum ayarlamaları yapılabilmektedir. Bu menüde bulunan seçenekler ekranda var olan konumlama elamanları ile yapılabilecek tüm hareketleri sağlar. Ayrıca kullanımı hızlandırmak adına klavye kısayol tuşları da tanımlanmıştır. Bu sayede kullanıcı klavyeden gireceği tuş birleşimleri ile konum değişiklikleri yapabilmektedir. Bu kısayollar Çizelge 3.5’te gösterildiği şekildedir.

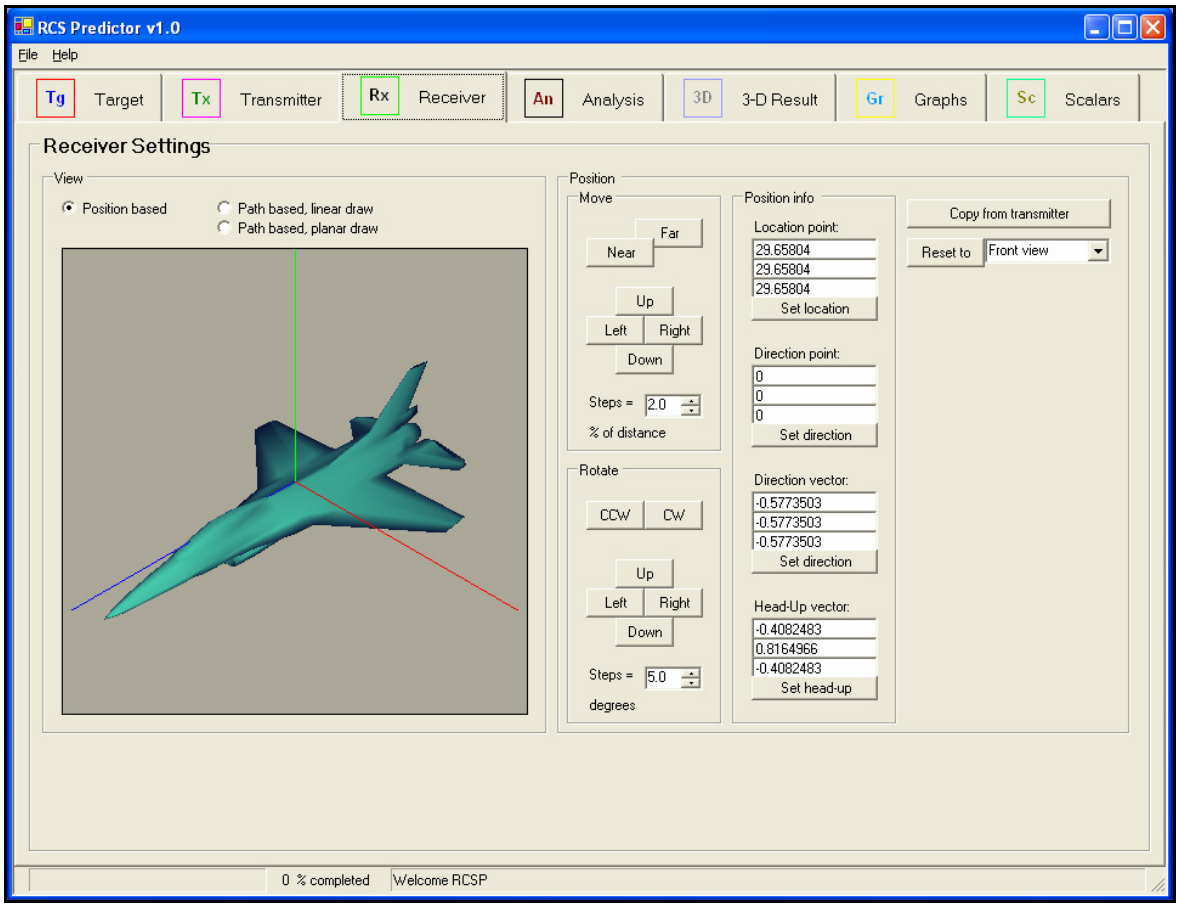
Çizelge 3.5. Konum değişiklikleri için klavye kısayol tuşları.

Hareket türü	**Hareket yönü**	**Kısayol tuşu**
Taşıma	Sola	CTRL+Sol ok
	Sağa	CTRL+Sağ ok
	Yukarı	CTRL+Üst ok
	Aşağı	CTRL+Alt ok
	Uzağa	CTRL+ALT+Üst ok
	Yakına	CTRL+ALT+Alt ok
Döndürme	Sola	ALT+Sol ok
	Sağa	ALT+Sağ ok
	Üste	ALT+Üst ok
	Alta	ALT+Alt ok
	Saat yönünün tersine	CTRL+ALT+Sol ok
	Saat yönüne	CTRL+ALT+Sağ ok

Çizelge 3.5. devam ediyor.

Kopyalama	Verici konumundan	CTRL+ALT+T
	Alıcı konumundan	CTRL+ALT+R
Sıfırlama	Perspektif	CTRL+ALT+S
	Üst	CTRL+ALT+W
	Alt	CTRL+ALT+X
	Ön	CTRL+ALT+Z
	Arka	CTRL+ALT+E
	Sol yan	CTRL+ALT+A
	Sağ yan	CTRL+ALT+D

3.3.3. RX ekranı

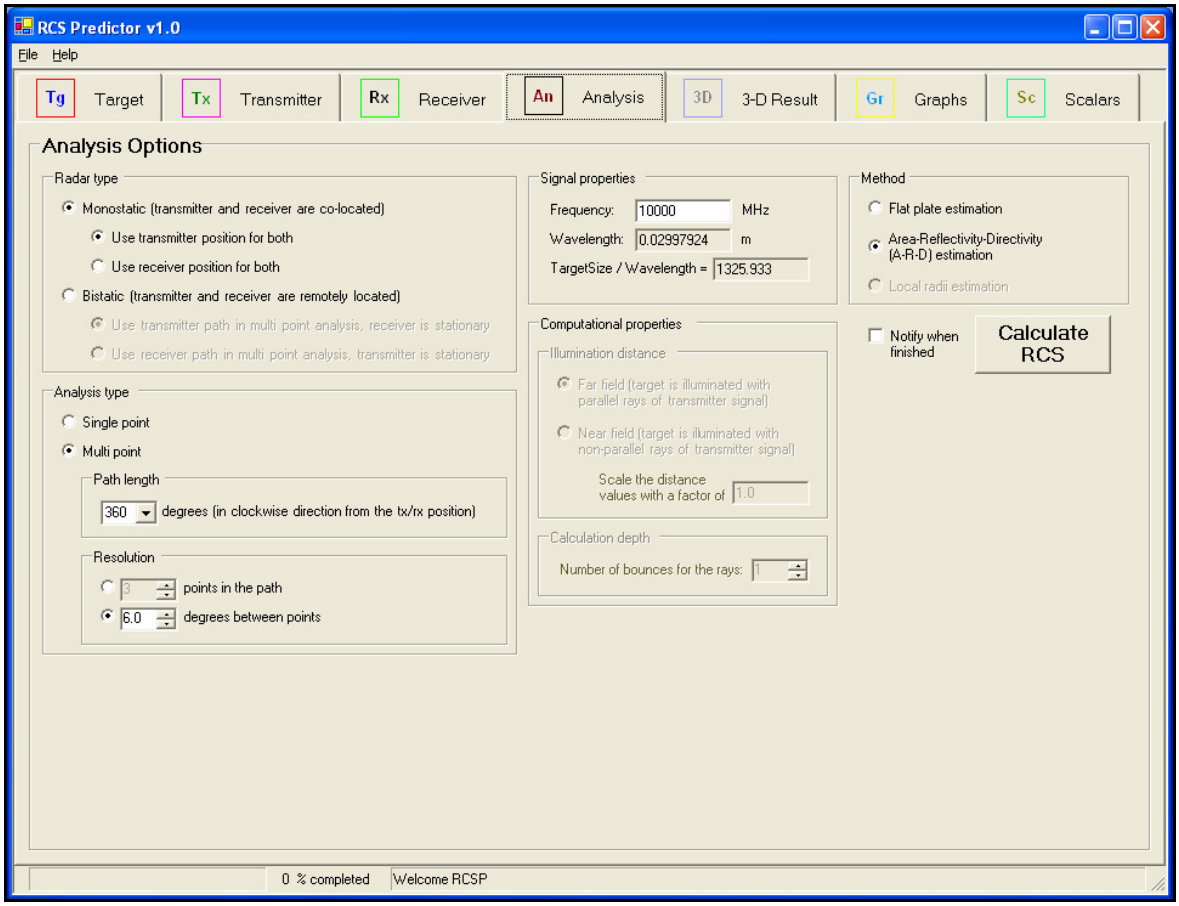


Şekil 3.26. RX ekranı.

Bu ekran görsel olarak TX ekranı ile aynı özellikleri taşır. Kullanımı ve çalışması da verici ekranıyla benzerdir.

Kullanıcı, verici ve alıcı ekranlarındaki ayarlamaları yaptıktan sonra analiz ayarlarını yapmak üzere bir sonraki sekmeye geçmelidir.

3.3.4. AN ekranı



Şekil 3.27. AN ekranı.

Kullanıcının en son ayarlamaları yaparak RCS analizini başlatacağı sekme AN ekranıdır.

Bölüm 2'de de belirtildiği gibi, radar sistemleri verici ve alıcının bulundukları yere göre iki şekilde adlandırılırlar. Verici ile alıcı aynı yerde ise tek durağanlı (monostatik), farklı yerde ise çift durağanlı (bistatik) radar adı verilir. RCSP programı da bu iki durumu göz önüne alarak çalıştığından, bu ekranda kullanıcıdan radar tipini belirtmesi istenir. Eğer tek durağanlı radar seçilmiş ise, verici ve alıcının paylaştığı konum da sorulur. Bunun için kullanıcı daha önceki sekmelerde tanımladığı verici konumu ya da alıcı konumundan birinin kullanılmasını isteyebilir. Çift durağanlı radar seçilmesi durumunda da iki ayrı

yerde konumlanmış olan verici ve alıcıdan hangisinin hareketli kabul edileceği sorulur. Buradaki hareketlilik kavramı, çok noktadan analiz yapılacağı zaman geçerlilik kazanacaktır.

Kullanıcı, analizin sadece belirttiği konumdan ve tek noktada yapılmasını isteyebilir. Diğer bir seçeneği de çok noktadan analiz yapılmasını istemektir. Bunun için analiz tipi grubunda yer alan “Single point” ve “Multi point” seçeneklerden birini işaretler. Tek nokta analizi istenmişse, verici ve alıcı için tek ya da çift durağanlılık durumu da gözetilerek belirtilen konumlar kullanılır. Analiz noktası olarak ya verici ve alıcının mevcut konumları ya da tek durağanlılık için belirtilen ortak konum kullanılacaktır. Çok noktada yapılacak analiz için ise bu analiz noktalarının tanımlanması gerekir.

Çok noktalı analizde kullanılacak noktalar kümesinin bir rota üzerinde olması gerekmektedir. Rota tanımı da, belirtilen bir konumdan başlanarak dairesel yörünge üzerinde alınacak uzunluğu belli yol olarak yapılabilir. Rotanın başlangıç konumu tek durağanlı sistem için bellidir, çünkü verici ve alıcı aynı yerdedir. Çift durağanlı sistem içinse yukarıda belirtilen hareketlilik seçeneğine bakılır. Hareketli olması istenen birimin konumu kullanılarak rota başlangıcı belirlenir. Başlangıç noktasını belirten bilgi, kullanılan konumun “Location::Point” alanındaki yer bilgisinden alınmaktadır. Rota yörüngesi dairesel olacaktır ve merkezini seçilen rota başlangıç konumunun içerdiği “Direction::Point” alanındaki yönelme noktası belirler. Başlangıç noktasından geçecek şekilde bu merkez etrafında sayısız yörünge çizilebilir. Bu yörüngelerden hangisinin kullanılacağına da yine başlangıç konumunun içerdiği “UpVector” alanındaki üst yön bilgisine bakılarak karar verilir. Bu üst yön vektörü, rota yörünge düzleminin normal vektörünü tayin eder. Bu şekilde tek bir dairesel yörünge tanımlanmış olur.

Tanımlanan dairesel yörüngede ne kadar yol alınacağı da kullanıcı tarafından belirlenir. Örneğin kullanıcı simetrik bir şekil ile ilgilendiğini bildiğinde, bu şeklin sadece bir yarısında analiz yapmak isteyebilir. Hedef cismin sadece bir bölgesinden alınacak RCS değerlerine odaklanmış olabilir. Bu türden ihtiyaçlar nedeniyle, tanımlanan yörünge üzerinde alınacak yol kullanıcının seçimine bırakılmıştır. Bu seçim, “Path length” altındaki alana girilecek açı değeri ile belirtilir. Bu açı değeri, belirlenen yörünge üzerinde başlangıç konumundan

itibaren saat yönünde gidilerek ne kadar yol alınacağını anlatır. Sunulan seçenekler 30, 45, 60, 90, 120, 180 ve 360 derecedir.

Çok noktada analiz için, yapılan rota seçiminin yanı sıra bu rota üzerindeki analiz noktaları da belirlenmelidir. Seçilen rota, dairesel bir yayı belirtmektedir. Kullanıcı, bu yay üzerinde kaç noktada işlem yapılacağını ya da işlem noktaları arasında kaçar derecelik açı farkı (0.5 derecenin katları şeklinde) olacağını belirtebilir. Bu iki yöntemden biri uygulanarak, yapılacak çok noktalı analizin açısal çözünürlüğü belirlenmiş olur.

Yapılan radar tipi ve analiz tipi seçimlerinden sonra, kullanıcı çalışılacak radar frekansını da programa bildirir. Frekansa bağlı RCS değerleri hesaplamasında kullanılacak bu değer, dalga boyuna çevrilerek de arayüzde kullanıcıya gösterilir. Burada, seçilen hedef cismin boyutlarından en uzunu ile dalga boyu arasındaki oran da arayüze aktarılır. Amaç, kullanıcının bu oranı göz önünde tutarak RCS analiz sonuçlarının doğruluk payı hakkında fikir yürütebilmesini sağlamaktır. RCS analizleri yüksek frekans tekniklerini içerdiğinden, bu boyut dalga boyu oranı yüksek iken hesaplanan değerler gerçeğe daha yakın olacaktır.

AN ekranında görülen “Computational properties” (hesaplama özellikleri) grubu, ileriye dönük olarak eklenmesi düşünülen alanları içermektedir. Bu alanların RCSP programının ilk versiyonunda hesaplamaya etkisi yoktur. Yakın/uzak alan (near/far field) kavramı, radar verici ve alıcısı için seçilen konumun algılanmasını belirleyecektir. Böylece radar sinyallerinin hedef cisim üzerine bir nokta kaynaktan çıkıyormuşçasına düşürülmesi ya da tüm sinyallerin paralel varsayılması gibi farklı durumlar oluşturulabilecektir. Ayrıca, sinyallerin alıcıya ulaşmadan önce kaç yüzeyden yansımalarına imkan verileceği de bir değişkene bağlanacaktır. Bu sıçramaların (bouncing) da takibi ile, hedef cisim üzerindeki köşe yansıtıcılarının (corner reflector) katkıları daha etkin olarak hesaba dahil edilebilecektir.

RCS değerini farklı yöntemler kullanarak ve farklı varsayımlara dayanarak kestirmek mümkündür. RCSP programında da, birden fazla yöntem ile RCS kestirimi yapılması ihtiyacı doğabileceği düşünülerek “Method” grubu altında farklı yöntemler listelenmiştir. Sonraki üretimlerde bu alana yeni yöntemler eklenebilir. Bu sayede kullanıcı farklı seçenekler arasından duruma en uygun olan ve en iyi

kestirimi veren yöntemi bulabilecektir. Programın bu sürümünde “Flat plate estimation” (düz plaka tahmini), “Area-Reflectivity-Directivity (A-R-D) estimation” (alan-yansıtıcılık-yönlülük tahmini) ve “Local radii estimation” (bölgesel yarıçaplar tahmini) olmak üzere üç yöntem listelenmiştir. Bunlardan düz plaka ve A-R-D tahminleri için hesaplamalar gerçekleştirilmektedir. Bölgesel yarıçaplar tahmini ise GO (geometrik optik) yaklaşımına göre yapılabilecek hesaplamalar için eklenmiş fakat işlemleri gerçekleştirilmemiştir. Bölgesel yarıçaplar tahmini için öngörülen, her yüzeyciğin köşe normallerine göre bulunacak eğriliklerini dikkate alan ve Eş. 2.6’ya dayanarak hesaplamaları yapan bir yöntem kullanılmasıdır.

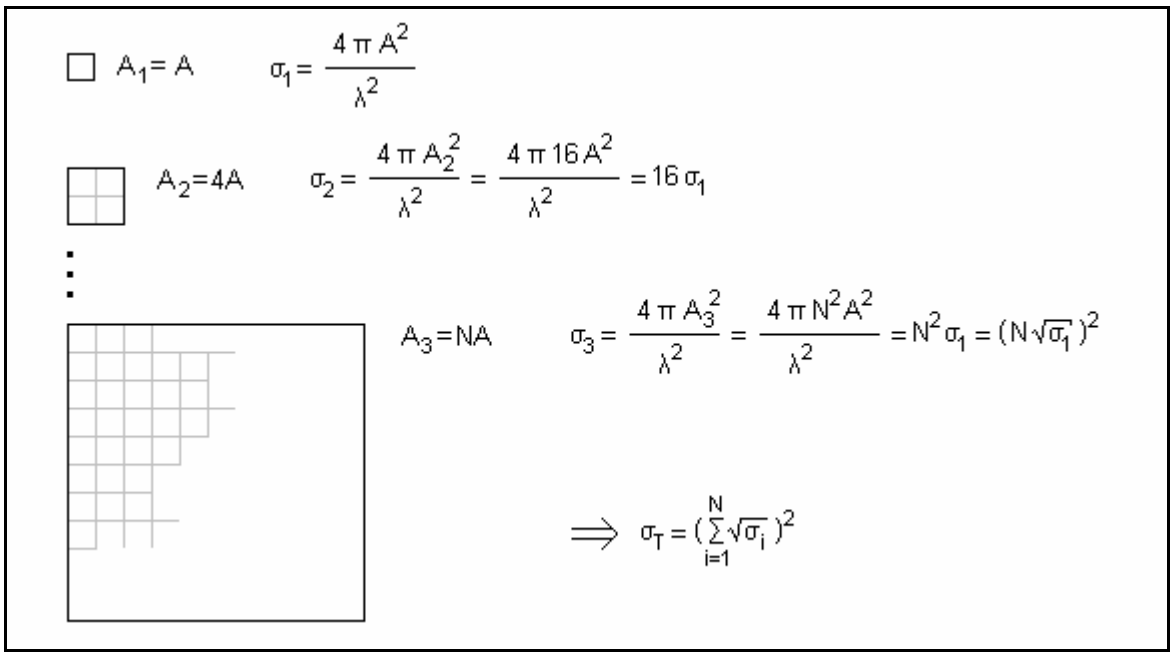
Programda, düz plaka tahmini için hesaplamalar gerçekleştirilmiştir. Yöntemde, düz plaka için Şekil 2.3’te verilmiş olan denklem kullanılmaktadır (Eş. 3.14). Ayrıca yüksek frekans yaklaşımına dayanarak, her bir yüzeycik ayrı bir saçınım merkezi olarak varsayılmıştır. Aşağıdaki Şekil 3.28, bir düz plakanın RCS değeri ile o plakayı oluşturan küçük plakaların RCS değerleri arasındaki ilişkiyi göstermektedir. Buna göre, ayrık saçınım merkezleri varsayımında, o merkezlerden elde edilen RCS değerlerinin önce kare kökleri bulunmuş sonra toplam değerinin ikinci kuvveti alınmıştır.

RCSP programında düz plakadan farklı olarak yüzeycikler üçgenlerden oluşmaktadır. Programdan beklenen bir yaklaştırma hesabı olduğundan denklemlerde üçgen alanı kullanılmasında bir sakınca görülmemiştir. Ayrıca yüzeyin aydınlanma açısına göre etkin alanı hesaplanarak kullanılmıştır. Bir yüzey, aydınlanma açısına bağlı olarak gelen sinyal yoğunluğunun farklı miktarlarını yakalayacaktır. Etkin alan ise bu aydınlatmanın kaynağı tarafından algılanan yüzey alanıdır ve gerçek alan ile aydınlatma açısının kosinüsü çarpılarak bulunur (Şekil 3.29).

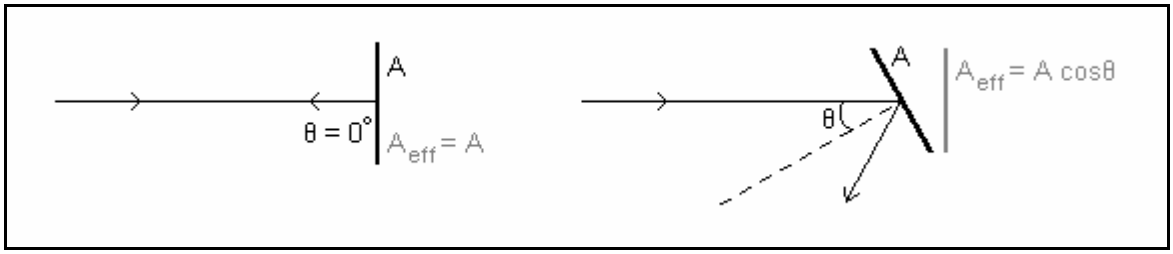
Sonuç olarak bu yaklaşımlar ve varsayımlar ışığında, hedef cismin RCS değeri Eş. 3.15’teki gibi bulunabilir. Ayrıca yüzey için verilen yansıtma (ya da soğurma) ve saçınım katsayıları da temelde o yüzeyin yakaladığı ve yaydığı gücü etkilediğinden etkin alan kavramı gibi hesaba katılabilir. Bu durumda R (reflectivity) yansıtma oranını ve D (directivity) yönlülük fonksiyonu değerini göstermek üzere toplam RCS değeri için Eş. 3.16 elde edilir. RCSP programında

da, düz plaka tahmini seçeneği işaretlendiğinde kullanılan hesaplama yöntemi bu şekildedir.

$\sigma = \frac{4\pi A^2}{\lambda^2}$, (A: alan, m^2; λ: dalga boyu, m) (3.14)



Şekil 3.28. Düz plakanın küçük yüzeyler toplamı şeklinde gösterimi ve aralarındaki RCS ilişkisi.



Şekil 3.29. Aydınlanma açısına göre etkin yüzey alanı bulunması.

$$\sigma = \left(\sum_{i=1}^{N}\sqrt{\sigma_i}\right)^2 = \left(\sum_{i=1}^{N}\sqrt{\frac{4\pi}{\lambda^2}A_{eff\,i}{}^2}\right)^2 = \frac{4\pi}{\lambda^2}\left(\sum_{i=1}^{N}A_{eff\,i}\right)^2 = \frac{4\pi}{\lambda^2}\left(\sum_{i=1}^{N}\left(A_i\cos\theta_i\right)\right)^2 \quad (3.15)$$

$$\sigma_T = \frac{4\pi}{\lambda^2}\left(\sum_{i=1}^{N}\left(RD_iA_i\cos\theta_i\right)\right)^2 = \frac{4\pi}{\lambda^2}\left(R\sum_{i=1}^{N}\left(D_iA_i\cos\theta_i\right)\right)^2 \quad (3.16)$$

RCSP programında gerçekleştirilen diğer yöntem de A-R-D tahminidir. Bu tahmin için de Eş. 2.1'e dayanılarak hesaplamalar yapılmıştır. Yöntem, düz plaka yöntemi ile benzerlikler içermektedir, fakat farklı olarak kare alma işlemi ve $4\pi/\lambda^2$ açıklık değeri kullanılmamaktadır. Burada, yüzeycik alanı tarafından yakalanacak güç ve bu gücün ilgilenilen yönde saçılacak oranı ön plana çıkmaktadır. Bu güç miktarına denk saçılımı sağlayabilecek alan ise RCS olarak değerlendirilmektedir. A-R-D yönteminde de yüksek frekansa dayanarak yapılan bağımsız ayrık saçınım merkezleri varsayımı benimsenmiştir. Buna göre, RCS tahmini için Eş. 3.17'de verilen denklem kullanılmıştır. R ve D yine yansıtma oranı ile yönlülük fonksiyonu değerini göstermektedir.

$$\sigma_T = \left(\sum_{i=1}^{N} \left(RD_i A_i \cos\theta_i \right) \right) = \left(R \sum_{i=1}^{N} \left(D_i A_i \cos\theta_i \right) \right) \tag{3.17}$$

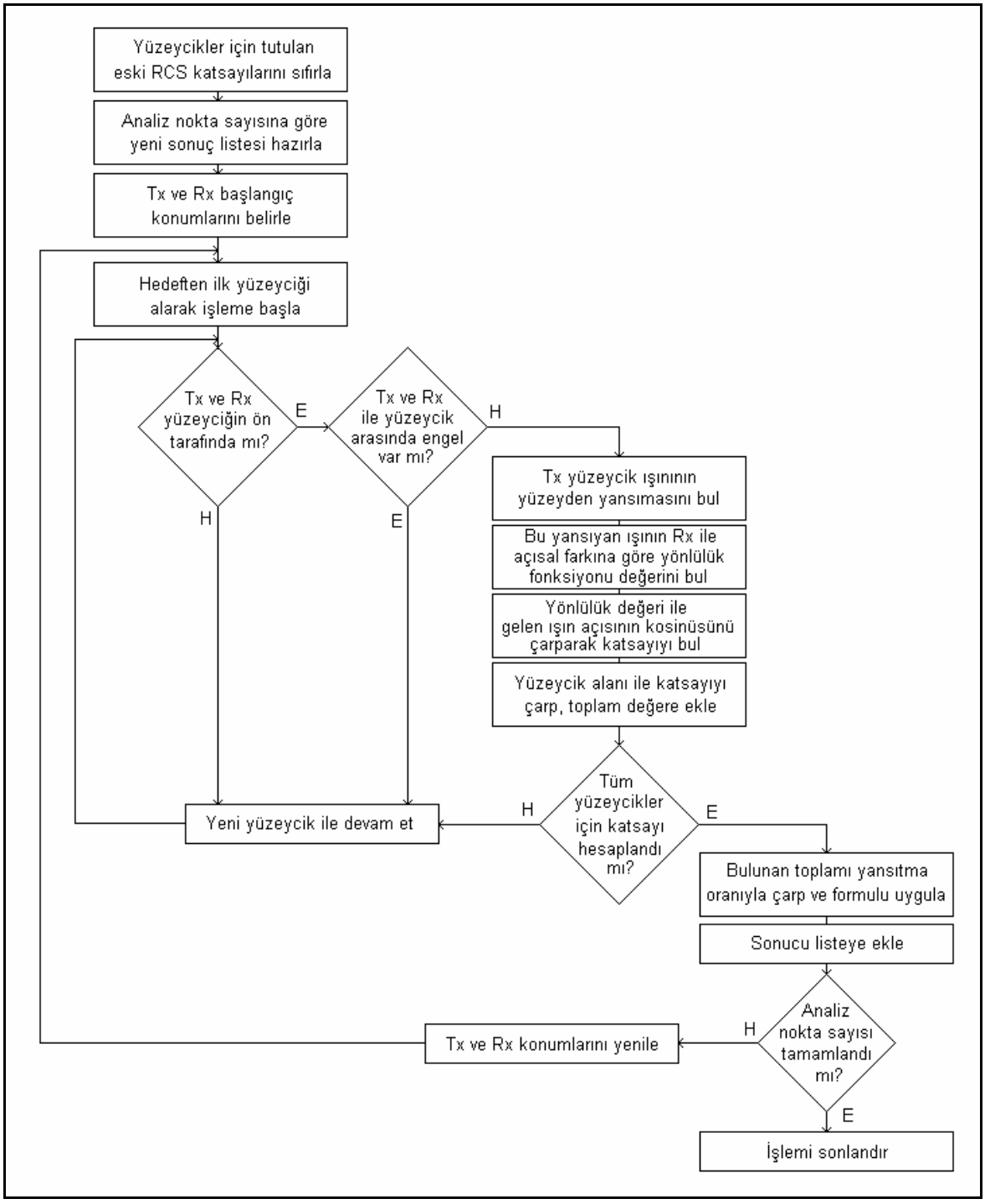
Kullanıcı, bu ekranda yaptığı ayarlamaların ardından "Calculate RCS" butonuna basarak hesaplama işlemini başlatır. Bu sırada işlemin tamamlanan kısmının yüzdesi arayüz ile kullanıcıya bildirilir. İşlem devam ederken arayüzdeki kontrollerin çalışması engellenmiştir. Bu nedenle kullanıcı başlattığı hesaplamanın bitirilmesini beklemek durumundadır. Kullanıcı "Notify when finished" seçeneğini işaretlemişse işlem tamamlandığında sesli ve yazılı bir mesajla uyarılır. Bu mesajda, gerçekleştirilen işlemin ne kadar zaman aldığı da belirtilmektedir.

RCS hesaplama işlemi başlatıldığında RCSP programında sırasıyla şunlar olmaktadır. Öncelikle, radar tipine bakılarak verici ve alıcı için konum değişkenleri oluşturulur. Hedef cismin yüzeycik sınıflarına ait alanlarda tutulan eski RCS bilgileri varsa temizlenir. Yapılacak hesaplamanın açısal çözünürlüğüne ve toplam nokta sayısına göre yeni bir sonuç listesi oluşturularak hesaplama aşamasına başlanır. Bu aşama, analiz istenen her konum için tekrarlanan bir döngüden oluşmaktadır. Döngüde yapılan işlem, başka bir fonksiyona hesaplatılan RCS değerini uygun birime çevirerek sonuç listesine yazmak ve ardından gerekiyorsa verici ya da alıcının konum değişikliğini yapmaktan ibarettir. Verilen tek bir konum için RCS hesaplamasını yapan fonksiyon ise hedef cismin tüm yüzeycikleri için tekrarlanan bir döngüyle çalışır. Bu döngüde, her bir yüzeycik için bulunan RCS katsayısı ile yüzeycik alanı çarpılarak sonuçlar toplanır. Yapılan işlem, her yüzeyciğin farklı saçınım merkezi kabul edilmesine dayalı olarak RCS değerlerinin

toplanmasıdır. Buradan çıkan toplam diğer katsayılarla çarpıldıktan sonra, arayüzde seçilmiş olan hesaplama yöntemine göre Eş. 3.16 veya Eş. 3.17'ye uygun olarak işlenir. Ancak buradaki değere uygulanacak işlem, istenilen hesaplama yönteminde bağlı olarak değişiklik gösterebilir. Düz plaka tahmini için değerin karesi alınarak $4\pi/\lambda^2$ ile çarpılır. A-R-D tahmini içinse başka işlem uygulanmaz.

Yukarıda anlatılan hesaplama döngüsü içindeki RCS katsayısı bulma işlemi, bir yüzey için belirli bir verici ve alıcı konumuna göre yapılır. Bu katsayının hesaplandığı fonksiyonda yapılan ilk kontrol, verici ve alıcı konumlarının yüzeyciğe göre yerleşimleridir. Eğer verici ya da alıcı, yüzeyciğin içinde bulunduğu düzlemin ön tarafında değilse işleme devam edilmez. Verici ve alıcı konumları, yüzeycik normalinin işaret ettiği yönde olmalıdır. Aksi durumda yüzeyin vericiden sinyal alması ya da alıcı yönünde sinyal saçması mümkün olamayacaktır. Bu durumda, yüzeyciğin RCS değerine de etkisi yoktur. Yapılan bu ilk kontrol, çoklu sinyal yansımalarının ya da sıçramalarının (multiple bouncing) da hesaba katıldığı durumda geçersiz olacaktır. İlk kontrolün ardından, verici ve alıcı ile o yüzeycik arasında bir engel olup olmadığına bakılır. Engel olması durumunda yüzeycik vericiye göre gölgelenmiştir ya da alıcıyı göremiyordur. Bu durumda da RCS değerine etkisi sıfır olur. İkinci kontrol de geçilmişse yüzeyciğin saçınımı hesaplanabilir durumdadır. Verici ile yüzeycik merkezi arasındaki ışının yüzeycik normali ile yaptığı açı kullanılarak Snell yasasına (bkz. [15]) göre yüzeyden yansıması beklenen ışın elde edilir. Ancak alıcı konumu bu ışın üzerinde olmayabilir. Bu nedenle, yüzeycik merkezi ile radar alıcısı arasında sanal ikinci bir ışın oluşturulur. Bu iki ışın arasındaki açıya göre yönlülük fonksiyonu değeri bulunur. Burada yapılan ise, yüzeyden alıcı yönünde saçılacak ışının gücünü bir olasılık dağılım fonksiyonuna bağlı olarak bulmaktır (bkz. Şekil 3.22). Fonksiyonun dönüş değeri olan RCS katsayısı da bu yönlülük değeri ile aydınlanma açısının kosinüsü çarpılarak bulunur. Üzerinde işlem yapılan yüzeycik için bu RCS katsayısı varsa o yüzeyciğe ait daha önceki katsayılarla toplanarak kaydedilir. Tüm konumlar için yapılan işlemler tamamlandığında, her yüzeycik için kaydedilmiş bu değerler analiz yapılan konum sayısına bölünerek normalleştirilir. Bu sonuçlar da yüzeyciklerin üç boyutlu sonuç ekranındaki renklerini belirlemektedir.

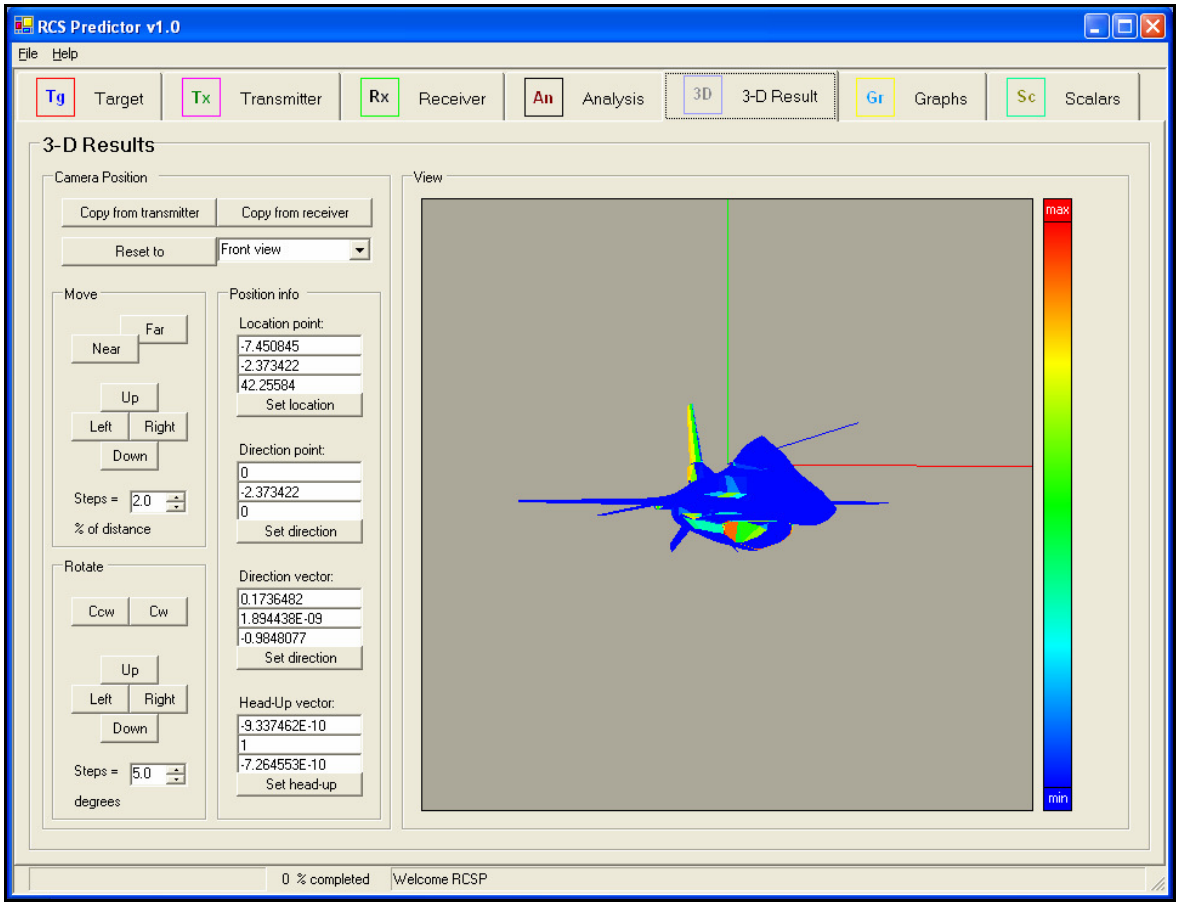
Detayları yukarıda anlatılan analiz adımlarının akış şeması aşağıda Şekil 3.30'da özetlenmiştir.



Şekil 3.30. Analiz adımları akış şeması.

Hesaplama işleminden sonraki adım ise sonuçların incelenmesidir. Bu amaçla kullanıcı son üç sekmeden birine geçebilir.

3.3.5. 3D ekranı



Şekil 3.31. 3D ekranı.

Kullanıcının analiz sonuçlarını inceleyebileceği ekranlardan biri 3D sekmesindedir. Bu sekmede bulunan elemanlar TX ve RX ekranlarındakilerle benzerdir. Burada da hedef cismin üç boyutlu görüntüsünün çizildiği bir OpenGL alanı ve bu alandaki görüntünün kontrolü için gerekli yönlendirme elemanları vardır. Yönlendirme elemanları, Çizelge 3.5'te belirtilen hareketleri sağlarlar. Bu elemanların kullanımları ve etkileri TX ekranında bahsedilen biçimdedir.

Bu ekrandaki çizim alanında, sayısal sonuçlar yerine görsel sonuçlar sunulmaktadır. Bundan amaçlanan, kullanıcıya hedef cismin yansıtma karakterini verebilmektir. Kullanıcı, cismi farklı açılardan inceleyerek hangi kısımların diğerlerine göre daha fazla sinyal yansıtma oranına sahip olduğunu görebilmektedir.

Şekil üzerinde, bu yansıtma oranlarının ayırt edilebilmesi için farklı renk kodları kullanılmıştır. Kullanılan renk kodları, şeklin sağ yanındaki alanda belirtilmiştir. Buna göre en yüksek yansımayı kırmızı renk, en düşük yansımayı ise mavi renk temsil etmektedir. Ayrıca RCSP kodunda seçilecek bir değişkene bağlı olarak bu renk skalası siyah-beyaz olarak da kullanılabilir. Ancak bu değişiklik arayüzden yapılmamaktadır ve kodun derlenerek programın yeniden üretilmesini gerektirir.

Çizimde, hedef cisim için verilen yüzeycik yapısı kullanılmaktadır. bu yüzeyciklerin her biri için AN ekranında bahsedilen biçimde bir RCS katsayısı hesaplanmaktadır. Bu katsayı yüzeyciğin alanından bağımsızdır ve dolayısıyla tüm yüzeycikler için normalleştirilmiş bir değer olarak kullanılabilir. Üç boyutlu sonuç görüntüsü çizilirken de yüzeycikler bu değerlere göre renk skalasından seçilen karşılıklarla gösterilirler. Bu yöntem ile, farklı büyüklükte alanlara sahip yüzeycikler arasında alanlarına değil yönlülük ve yansıtma durumlarına bakılarak renklendirme yapılmış olmaktadır.

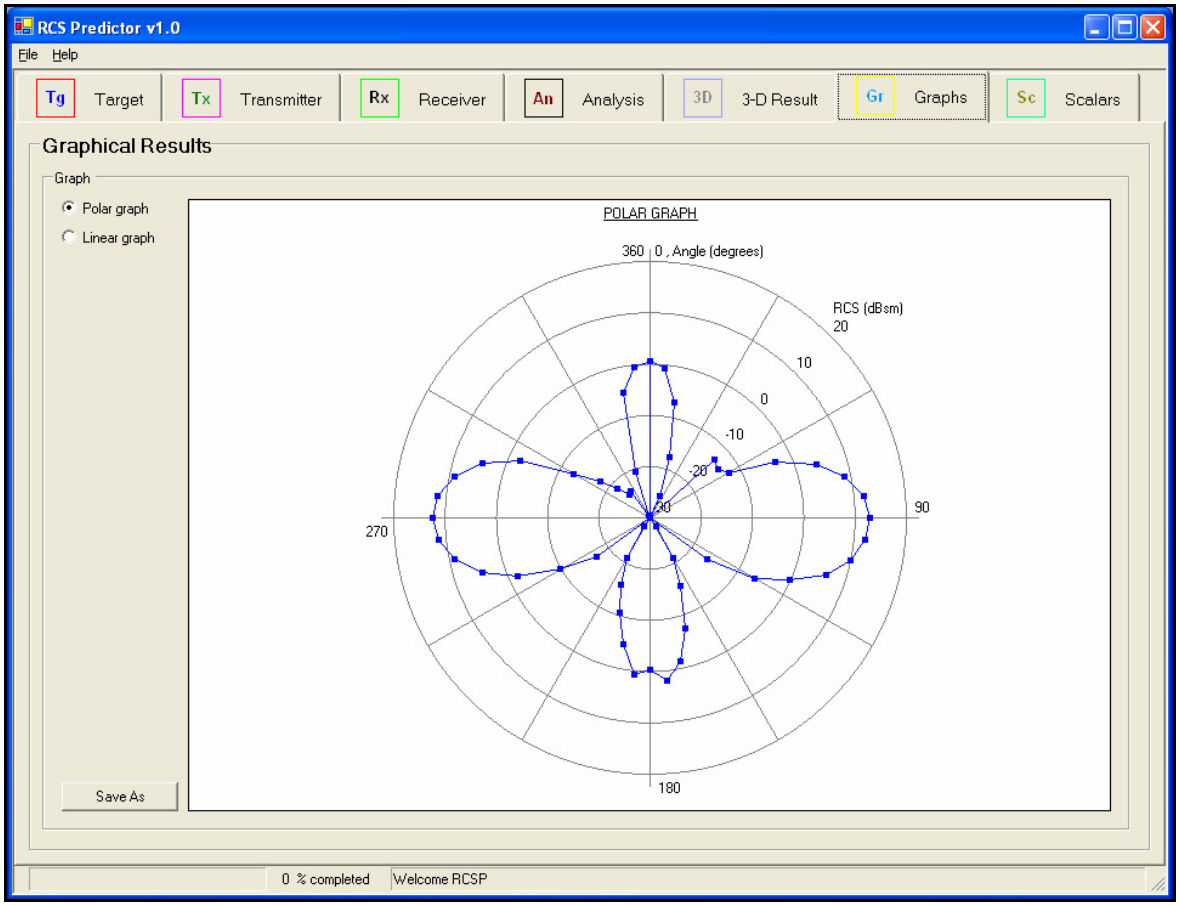
3.3.6. GR ekranı

Bu ekran, sonuçların grafiklerle gösterildiği yerdir. Kullanıcıya sonuçları inceleyebileceği iki ayrı grafik seçeneği sunulmuştur. Bunlar kutupsal (polar) ve doğrusal (linear) grafiklerdir. Her iki gösterimde de değişken ekseni açıları değer ekseni ise RCS değerlerini içermektedir. Açı ekseni değerleri, analiz aşamasında kullanılan rotanın başlangıcı 0 (sıfır) dereceyi gösterecek ve saat yönündeki hareket pozitif açıları verecek şekilde sıralanmıştır. RCS ekseni ise en yüksek ve en düşük değeri içine alacak şekilde dinamik olarak ayarlanmaktadır.

RCS değerlerini göstermek için genel olarak kullanılan grafik şekli kutupsaldır. Kutupsal gösterimde açı ekseni her zaman 0-360 dereceyi kapsayacak şekilde çizilir, ancak analiz için ne kadarlık bir rota yayı kullanılmışsa o bölgede RCS değerleri verilir. Örneğin analiz için 180 derecelik bir yay seçilmişse, kutupsal grafiğin sağ yanında ölçüm sonuçları yer alırken sol yarısı boş olacaktır. RCS değerleri için kullanılan eksen ise, içten dışa doğru artan değerleri gösteren halkalar şeklindedir. Burada en iç ve en dış halkanın değeri dinamik olarak ayarlanmaktadır ve her bir halkanın gösterdiği değer sağ üst bölgesinde yazılmıştır. Doğrusal grafik gösteriminde, değer ekseni gibi açı ekseni de dinamiktir. Analiz için kullanılan yayın açısı, grafik eksenine yayılarak gösterilir. Bu

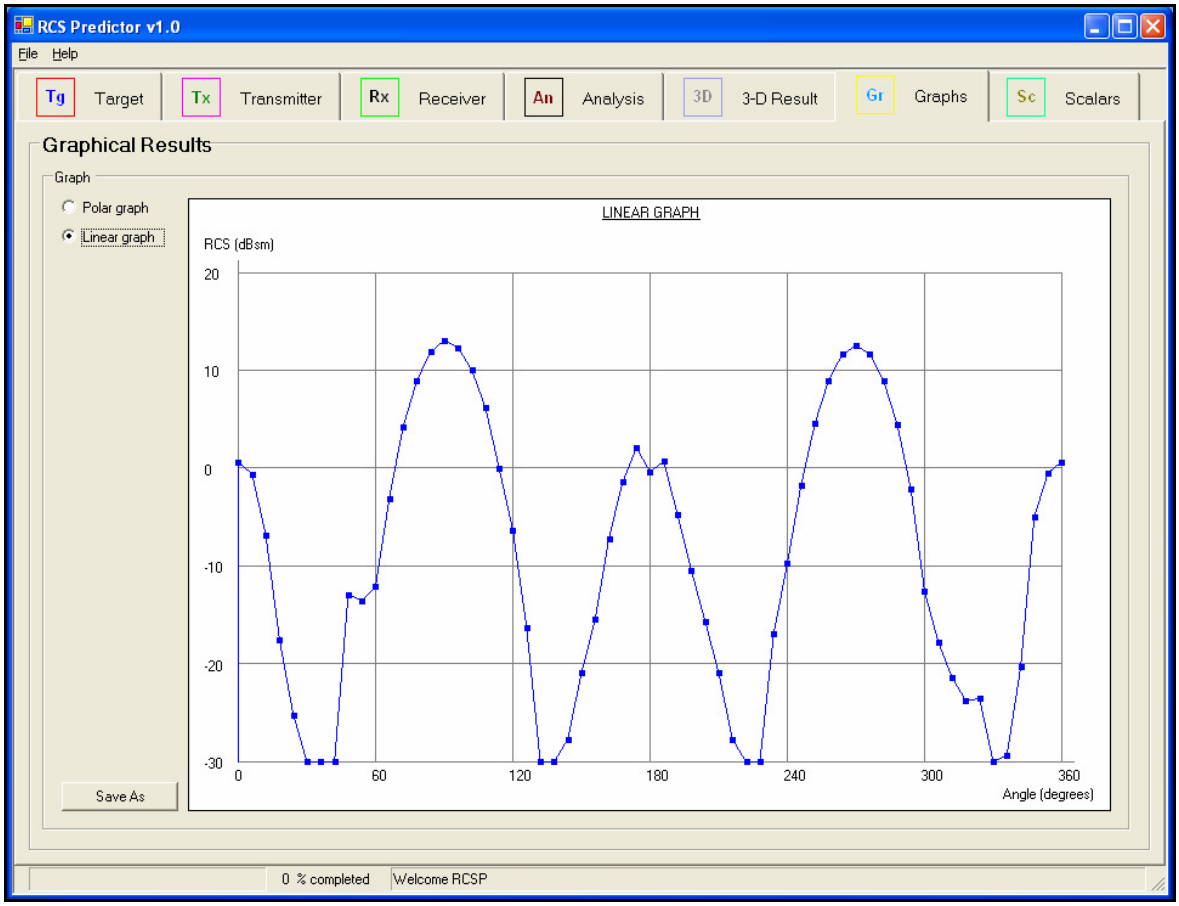
sayede kullanıcı, küçük bir rotada yaptığı analizin sonuçlarını daha geniş bir grafikte inceleyebilecektir. Kutupsal gösterimle de, cismin o rota düzlemi üzerindeki yansıtma davranışını daha kolay biçimde algılayacak ve analizle bağdaştırabilecektir.

Kullanıcı, ekranın sol tarafındaki “Polar Graph” ve “Linear Graph” kontrolleri ile iki grafik tipi arasından seçimini yapabilir. Seçili durumdaki grafiği de “Save As” butonuna basarak ayrı bir resim dosyası olarak kaydedebilir. Bu resim dosyası içeriği ekrandaki grafik bölgesinden ibaret olacaktır. Analize konu olan cisim ve analiz değişkenleri hakkında bir bilgi resim dosyasına eklenmeyecektir. Bu sebeple, kaydedilen resim dosyasının uygun biçimde isimlendirilmesi veya uygun yerde saklamasından kullanıcı sorumludur.



1. Kutupsal (polar) grafik.

Şekil 3.32. GR ekranı.



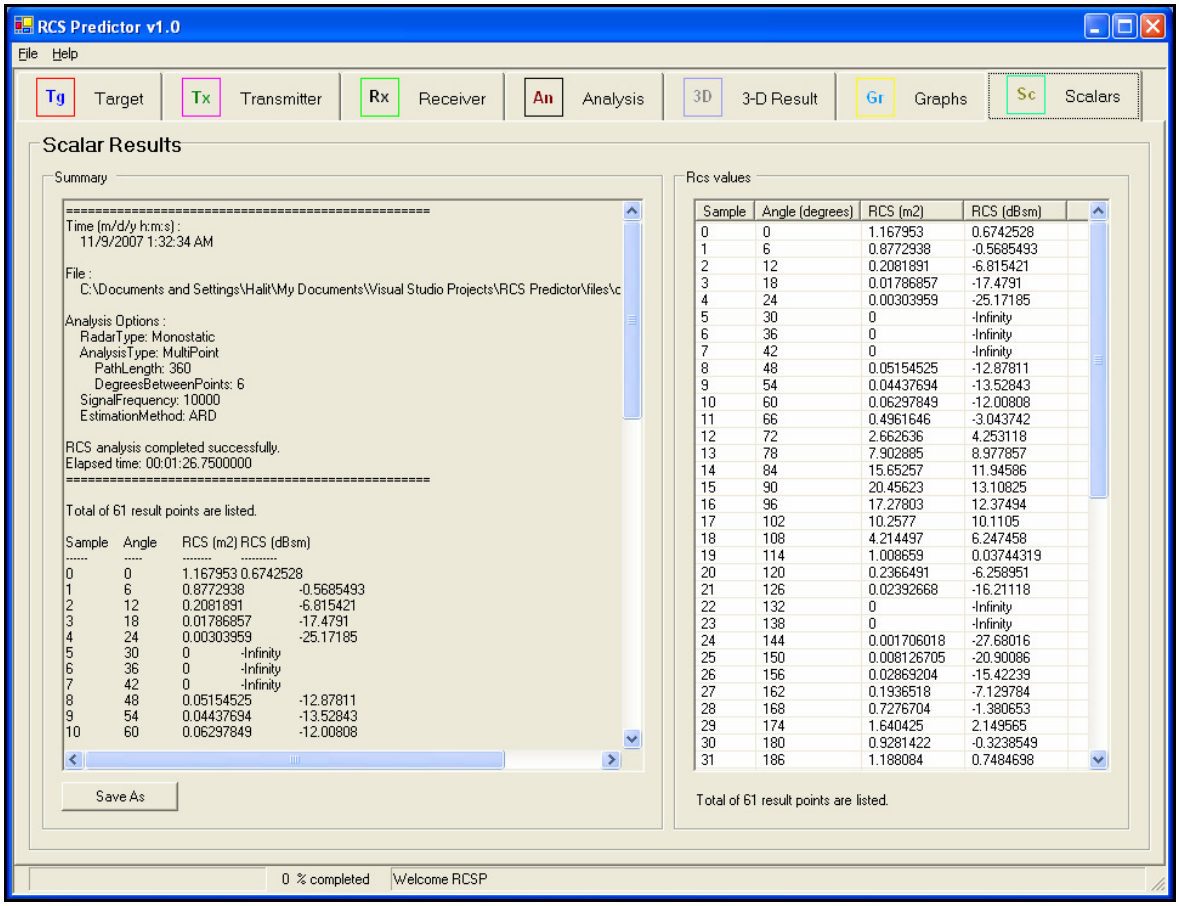
2. Doğrusal (linear) grafik.

Şekil 3.32. devam ediyor.

3.3.7. SC ekranı

Yapılan analizin sonuçları ve bu analize ait bir özet raporu bu sekmede sunulmaktadır. Kullanıcı yaptırdığı analizin özetini ekranın solunda görebilir. Analizin yapıldığı tarih ve zaman bilgileri, hangi dosya ile çalışıldığı, hangi analiz değişkenlerinin seçili olduğu ve hesaplamaların ne kadar sürede tamamlandığı gibi bilgiler bu özette yer alır. Analiz sonuçları ise sağ tarafta ayrıca verilmiştir. Burada da hesaplama yapılan her konum için rota başlangıç noktasına göre açı farkı ile birlikte o konumda elde edilen RCS değeri (m^2 ve dBsm olarak) listelenir. Bu değerler grafik ekranındaki çizimlerde kullanılan verilerdir.

Kullanıcı “Save As” butonu aracılığı ile bu ekrandaki özeti ve sonuç verilerini içeren bir metin dosyasını kaydedebilir. Kaydedilen dosyanın uygun biçimde isimlendirilmesi veya uygun yerde saklaması yine kullanıcının sorumluluğundadır.



Şekil 3.33. SC ekranı.

4. SONUÇLAR VE ÖNERİLER

Bu bölümde, tez çalışmasının sonuçları değerlendirilecektir. Ayrıca, gelecekte tez konusu ile ilgili olarak yapılabilecek çalışmalara yönelik öneriler de sunulacaktır.

4.1. Sonuçlar

Giriş bölümünde de bahsedildiği gibi, bu tez çalışmasında elektronik harp alanında önem arz eden RCS hesaplamalarının kabul edilebilir miktarlarda zaman ve kaynak harcanarak yapılabilmesi için milli bir RCS kestirim programının üretilmesi amaçlanmıştır. Bu amaç doğrultusunda yapılan çalışmada konu detaylı olarak incelenmiş ve bu alandaki kaynaklardan faydalanılmıştır. Pek çok farklı ayağı bulunan bu çalışmada hem RCS kavramına yönelik hem de programlama ve tasarım detaylarına yönelik tecrübe ve kazanımlar elde edilmiştir.

Tez çalışmasında, bazı eksikleri bulunmakla birlikte tatmin edici sonuçlar veren bir program üretilmiştir. RCSP adı verilen bu programının ilk versiyonunda toplam 30 farklı dosyaya yayılmış olarak yaklaşık 10200 satır kod bulunmaktadır. Toplam satır sayısının yaklaşık yüzde 40'lık kısmı geliştirme ortamı tarafından otomatik olarak üretilen kodlardan oluşmaktadır.

RCSP programı, STL formatındaki dosyalardan hedef cisim bilgilerini okuyarak bu cisim üzerinde farklı parametrelere bağlı analizleri gerçekleştirebilmekte ve RCS kestirimi sonuçlarını kullanıcıya sunabilmektedir. Görsel kullanıcı arayüzü, kolay kullanım sağlamakta ve anlaşılabilirliği arttırmaktadır. Kullanıcı, üzerinde çalışılan hedef cismin üç boyutlu görüntülerini inceleyebilmektedir. Ayrıca analiz sonuçlarını hem grafik hem de sayısal olarak inceleyip kaydedebilmektedir.

Bölüm 3.3 altındaki başlıklarda verilen ekranlara ait şekillerde, örnek bir hedef cisim için yapılan işlemler görülebilir. Bu şekillerde bir F-16 için yapılan analizden kesitler sunulmuştur. Cisim STL dosyasında yaklaşık 1400 yüzeycik ile tanımlanmıştır. Hedef cismin boyutları ise 13-20-6 metre mertebelerindedir. Monostatik radar tipi seçilen analizde, uçağın burun ucu başlangıç konumu olarak alınmış ve uçağın etrafında bir tam tur atacak şekilde bir rota belirlenmiştir (eğim açısı (pitch): 0 derece, ufuk açısı (azimuth): 0-360 derece). Bu rota üzerinde analiz noktaları arasında 6'şar derecelik aralıklar bırakılarak toplam 61 ölçüm değeri elde edilmiştir. Hesaplamalar A-R-D tahmini yöntemine göre yaptırılmıştır.

Bu örnek cisim için elde edilen RCS grafikleri beklenen biçimde değişimler göstermektedir. Şöyle ki, uçağın motor boşluğundan alınan yansımanın en yüksek olduğu, o bölgedeki yüzeyciklerin turuncu-yeşil renginden anlaşılmaktadır (bkz. Şekil 3.31). Ayrıca kanat ucunda ve arka yönlerde de yansımanın fazla olduğu gözlenmektedir (bkz. Şekil 3.32). Bu sonuçlar bir uçak için beklenen tipik RCS grafiklerine ([7], figure 5 ve [36]) uymaktadır. Örnekte elde edilen RCS değerlerinin ortalaması ise 4 dBsm'dir. Bu da kullanılan hedef için kabul edilebilir bir değerdir.

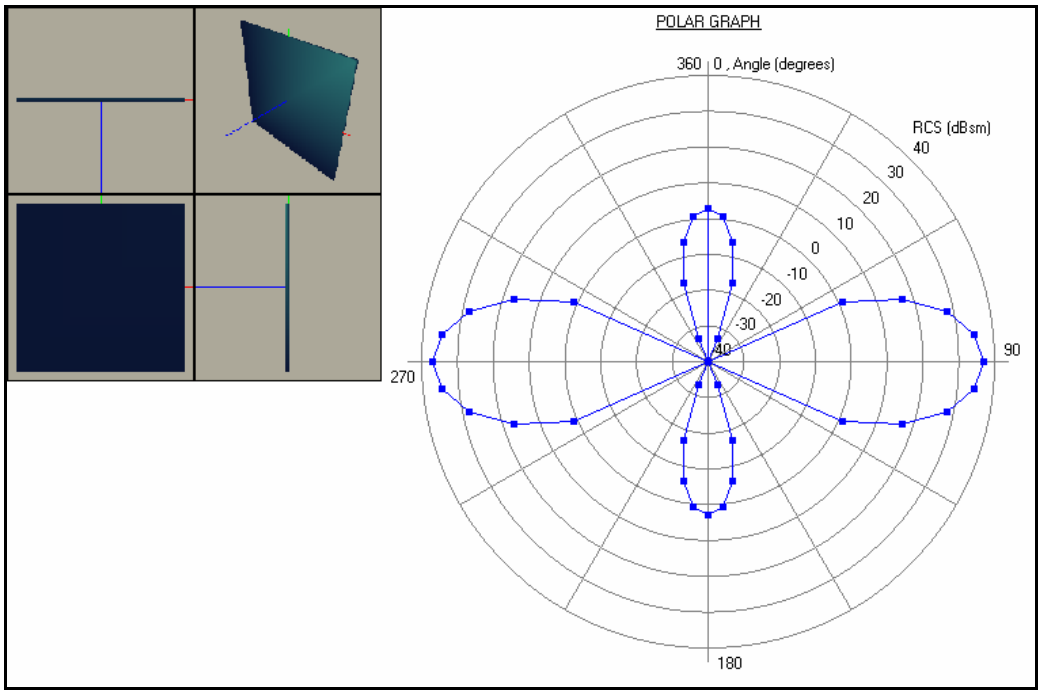
EK 5'te, [36]'da [37]'den alınarak verilmiş olan ve farklı uçaklar için ortalama RCS değerlerini gösteren Çizelge E5. sunulmuştur. Buna göre, RCSP programından elde edilen RCS sonuçları grafik olarak doğru görünmekte ve beklenene yakın değerler içermektedir.

Aşağıdaki Çizelge 4.1'de, basit ve karmaşık yapıdaki bazı cisimler için yapılan analizlere ait bilgiler verilmiştir. Analizlerin tümünde monostatik radar tipi seçilmiş ve cisim etrafında 360 derecelik bir rota üzerinde yine 6'şar derecelik aralıklarla ölçüm yapılmıştır. Tüm diğer parametreler varsayılan açılış değerlerinde bırakılmıştır. Düz plaka ve küre için Şekil 2.3'te verilen eşitliklerle hesaplanan değerler beklenen RCS olarak belirtilmiştir. Ortalama RCS bilgisi ise RCSP programı analiz sonuçlarından elde edilmiştir. Bu çizelgeye göre de sonuçlar kabul edilebilir değerlere sahiptir. Ancak programda tanımlı yöntemler ile küre için yeterli doğrulukta sonuçlar alınamamıştır.

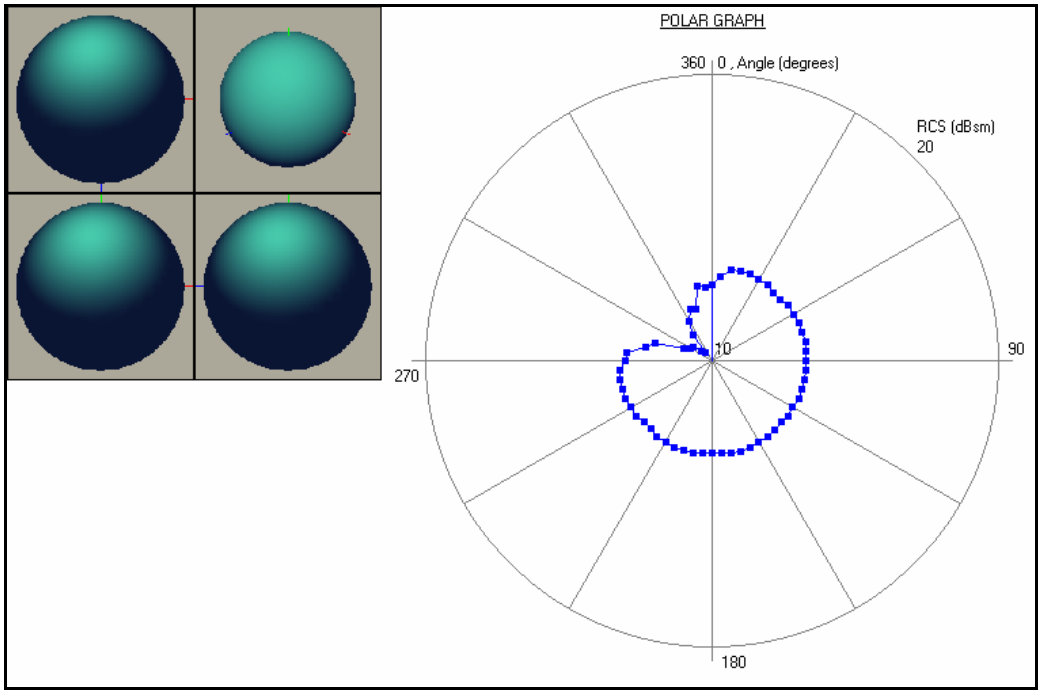
Çizelge 4.1. Bazı farklı örnekler için analiz bilgileri.

Hedef cisim	**Tahmin yöntemi**	**Ölçülen RCS (dBsm)**	**Beklenen RCS (dBsm)**
Düz Plaka (w=h=1m)	FlatPlate	Maks. 37 (bkz. Şekil 4.1)	Maks. 41
Küre (r=1m)	FlatPlate	13 (bkz. Şekil 4.2)	5
Örnek Füze ([12], figure R70)	FlatPlate	Min. -40, maks. 20 (bkz. Şekil 4.3)	Min. -15, maks. 30
F-16 (h=6m, w=14m, l=20m)	A-R-D	4 (bkz. Şekil 3.32)	7 (bkz. Çizelge E5.1)

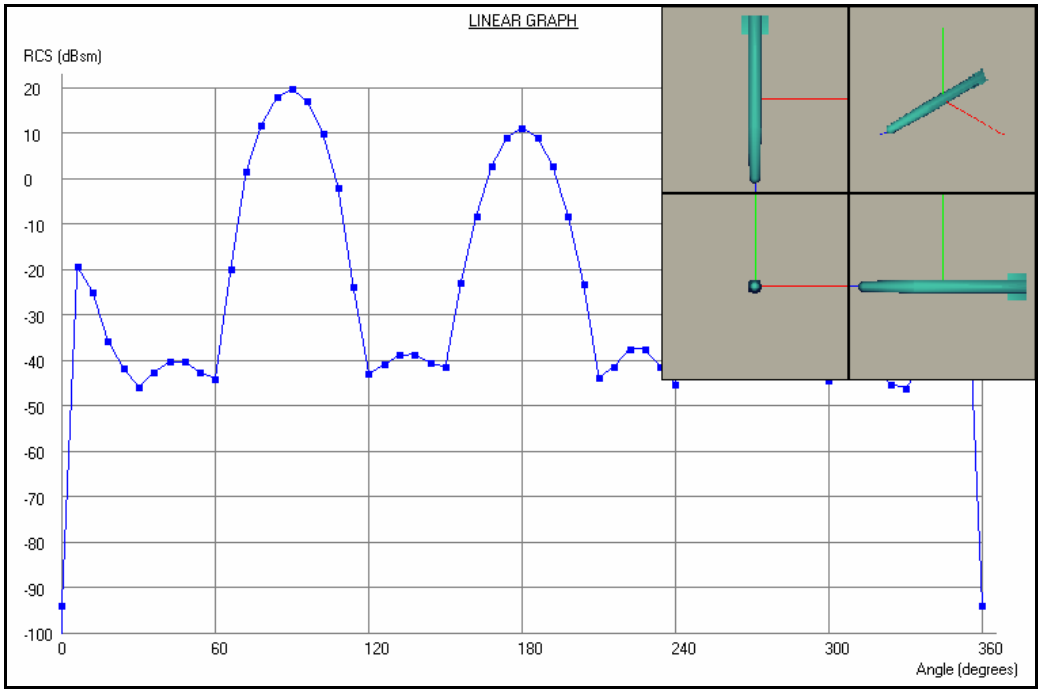
Sonuç olarak, tezin amaçları doğrultusunda, beklendiği şekilde çalışan ancak üzerinde iyileştirmelere ihtiyaç duyulabilecek bir program üretilmiştir. Bu iyileştirmelere örnek olması ve ileride yapılacak çalışmalara ışık tutması amacıyla bazı öneriler takip eden bölümde sunulmuştur.



Şekil 4.1. Düz plaka için analiz sonucu.



Şekil 4.2. Küre için analiz sonucu.



Şekil 4.3. Örnek füze için analiz sonucu.

4.2. Öneriler

RCSP programı tasarım ve kod yapısı olarak değişikliğe ve geliştirmeye açıktır. Bu çalışma üzerine yenilikler getirmek ya da iyileştirmeler yapmak isteyenler için uygun bir yapı mevcuttur. Programa yönelik birkaç iyileştirme önerisi de aşağıda belirtilmektedir. Yeni çalışmalarda kullanılabileceği düşüncesi ile programa ait kaynak kodu da EK 6'da sunulmuştur.

RCSP programı, girdi olarak STL dosya formatındaki verileri kabul etmektedir. Bu format birçok CAD programı tarafından üretilebilmektedir ve yaygın olarak kullanılmaktadır. Ancak gerektiğinde farklı formatta hazırlanmış verileri de okuyabilmesi RCSP için bir avantaj olacaktır. Bu nedenle, STL formatının yanı sıra 3DS veya DWG gibi farklı biçimdeki verilerin de girdi olarak kabul edilmesi için eklemeler yapılabilir. Bunun için yapılması gereken, ana yapıya uygun olarak "CrcspStlFileParser" sınıfına benzer biçimde sınıflar üretip programa eklemektir.

Mevcut versiyonda, yüklenen hedef cisim için tüm yüzeyciklerle ilgili hesaplamalarda kullanılmak üzere bir tek yünlülük fonksiyonu tanımlanmaktadır. Bunun gibi saçınım ve soğurma katsayıları da tüm cisim için ortaktır. Analiz edilecek hedef cismin farklı yüzey malzemesi ve kaplamalarının daha iyi taklit

edilebilmesi amacıyla bu değişkenler yüzeycikler için ayrı ayrı tanımlanabilir. Bu işlem fazladan hafıza kullanımına neden olabilecektir ancak doğru bir tasarımla bu ihtiyaç en alt seviyede tutulabilir. Ayrıca, ilk versiyonda tanımlı halde bulunan yönlülük fonksiyonları ve değerleri zenginleştirilebilir.

RCS hesaplama metodu olarak mevcut versiyonda kullanılan yöntemler, beklenen grafikleri sağlamakla birlikte elde edilen değerler bakımından eksik kalmaktadır. Bu yöntemlere ek olarak sunulacak yeni yaklaşımlar doğru sonuçların bulunmasını kolaylaştıracaktır. Örnek olarak Fiziksel Optik yaklaşımına göre hesaplamaların yapılacağı bir metot mevcut analizlere eklenebilir.

Programın ilk versiyonunda arayüze eklenmesine rağmen işlevselliği bulunmayan bazı özelliklere işlevsellik kazandırılabilir. Örneğin konum bilgilerinin metin kutularına değer girilerek değiştirilebilmesi sağlanabilir. AN ekranı altında bahsedilen yakın/uzak alan kavramlarına ya da ışın sıçrama sayısına yönelik kontroller eklenebilir.

RCS hesaplama algoritmasına göre, yüzeyciklerin yansıttığı ışınların bulunmasında ya da gölgelenme durumlarının kontrolünde yüzeycik merkez noktası kullanılmaktadır. Bu durum bazı yüzeycikler için RCS etkilerinin olması gerekenden küçük hesaplanmasına neden olabilir. Başka bir seçenek olarak yüzeycik merkezi yerine köşe noktalarının her biri ile gölgeleme testi yapılabilir ve yansıyan sinyaller bu noktalara göre bulunabilir. Ancak bu durumda hesaplama süresinin oldukça artabileceği de göz ardı edilmemelidir.

Mevcut RCSP programı, ortalama 1000-1500 yüzeycik içeren hedef cisimler için yapılan analizleri birkaç dakikanın altındaki sürelerde tamamlamaktadır. Fakat kullanılan ışın takibi algoritması benzeri yöntem, artan yüzeycik sayısına bağlı olarak uzun zaman gerektirebilmektedir. Bu hesaplama süresinin azaltılması için iki öneri sunulabilir. Birincisi, paralel işlemci kullanımıdır. RCS hesaplamasında çok noktalı analiz istenmişse farklı konumlar için benzer işlemler tekrarlanmaktadır. O halde bu işlemlerin eş zamanlı olarak farklı kaynaklarda tamamlanıp bir kaynakta birleştirilmesi hesaplama sonucunu etkilemeyecektir. Bu yöntemle hesaplama süresi çok noktada yapılan analizler için düşürülebilecektir. İkinci öneri ise algoritmanın gerçekleştirilmesi sırasında kullanılan yönteme

yöneliktir. Mevcut algoritmada belli bir verici ve alıcı konumu için tüm yüzeycikler üzerinden RCS katsayısı hesaplanmaktadır. Bu hesaplama sırasında da verici ve alıcının o yüzeyciğin hangi yanına düştüğü ve yüzeycik ile aralarında engel olup olmadığı kontrol edilmektedir. Değişken olmayan verici ya da alıcı konumu için de bu işlemlerin tekrarlanması fazladan zaman kaybına neden olmaktadır. Bu nedenle, buradaki işlemlerin her konum değişiminde bir defa yapılması ve yüzeycik sınıfı içinde yeni eklenecek alanlarda tutulması hesaplama süresini azaltacaktır.

Kullanıcı, analiz değişkenlerini belirleyip hesaplamayı başlattıktan sonra programa müdahale edememektedir. Bu amaçla arayüzdeki tüm kontroller hesaplama işlemi bitene kadar etkisiz hale getirilmektedir. Aksi durumda olaylara (event) bağlı çalışmakta olan arayüzde kilitlenmeler olabilmektedir. Ancak bu uygulama kullanıcının başlattığı bir analizi yarıda keserek sonlandırmasına izin vermemektedir. Gelişmiş bir programda olması gereken, bu işlemden "Cancel" butonu ile vazgeçilebilmesidir. Arka planda hesaplamalar devam ederken arayüzün hala aktif durumda olması ise iki işlemin farklı iş parçacıklarında (thread) gerçekleştirilmesi ile mümkün olacaktır. Visual Studio .NET kütüphanelerinin sunduğu "thread" yapısı ve sınıfları kullanılarak bu özellik RCSP programına kazandırılabilir.

RCSP programı arayüzünde kullanılan dil İngilizce'dir. Kullanılan pek çok terimin daha kolay anlaşılması için bu dil seçilmiştir. Ancak konuya ilişkin terimlerin tamamı için yaygın olarak kullanılan Türkçe karşılıklarının bulunması durumunda arayüz dilinin Türkçe'ye çevrilmesi ve hatta dil seçiminin kullanıcıya bırakılması uygun olacaktır.

KAYNAKLAR

[1] Nogayeva, A., 2007, ABD'den silaha büyük kaynak, Cumhuriyet gazetesi Strateji eki (16 Temmuz, 2007).

[2] Yıldız, H., Milli Elektronik Harp Yaklaşımı ve Teknolojik Öncelikler, http://www.turkishdefense.net/reference/HY/hy0001.htm (13 Ağustos, 2007).

[3] Savunma Sanayii Müsteşarlığı resmi internet sitesi, http://www.ssm.gov.tr/TR/savunmasanayiimiz/Pages/Tarihce.aspx (13 Ağustos, 2007).

[4] Scleher, D. C., 1999, Electonic Warfare in the Information Age, Artech House.

[5] Chatzigeorgiadis, F., 2004, Development of Code for a Physical Optics Radar Cross Section Prediction and Analysis Application, M.S. Thesis, Naval Postgraduate School, Monterey, California.

[6] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Radar_cross_section (14 Ağustos, 2007).

[7] Radar Cross Section, http://www.earth2.net/parts/mugu/rcs.pdf (25 Eylül, 2005).

[8] Antenna Measurements, RCS Measurements and Measurements on Pulsed Signals with Vector Network Analyzers R&S ZVM, R&S ZVK, Application note, http://www.rohde-schwarz.com/www/downcent.nsf/ANFileByANNoForInternet/95C92F360122C68BC1256F0B0028C147/$file/1EZ52_0E.pdf (08 Haziran, 2005).

[9] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Free_space_loss (14 Temmuz, 2007).

[10] IEEE Standard Dictionary of Electrical and Electronics Terms, 3rd ed., ANSI/IEEE Std 100, 1984.

[11] Knott, E. F., Shaeffer, J. F., Tuley, M. T., 2004, Radar Cross Section (2nd Edition), SciTech Publishing.

[12] Barton, D. K., Leonov, S. A., 1998, Radar technology encyclopedia, Artech House

[13] Millimeter Wave Technology Inc. product brochure, Signature Management: ARTEMISIA, http://www.mwt-materials.com/ARTEMISIA-RCS%20demo/ARTEMISIA%20Brochure.pdf (8 Haziran, 2005).

[14] Eric Weisstein's World of Science, a Wolfram web source, Physics division, http://scienceworld.wolfram.com/physics/GeometricOptics.html (21 Haziran, 2005)

[15] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Snell's_law (31 Ağustos, 2007)

[16] Skolnik, M. I., 1990, Radar Handbook (2nd Edition), McGraw-Hill.

[17] Losier, M., 1997, Investigation Into the Significance of Geometry on the Radar Cross Section of a Ship, M.S. Thesis, Royal Military College of Canada, Kingston, Ontario.

[18] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/STL_(file_format) (1 Eylül, 2007).

[19] Fabbers.com, Your Digital Fabrication Portal, http://www.ennex.com/~fabbers/StL.asp (1 Eylül, 2007).

[20] OpenGL Programming Guide, The Official Guide to Learning OpenGL, Version 1.1, http://www.glprogramming.com/red/ (3 Nisan, 2007).

[21] Universität Freiburg – Institut für Informatik, Reisert, M., http://lmb.informatik.uni-freiburg.de/people/reisert/opengl/doc/opengl.html (15 Mayıs, 2007).

[22] Reisert, M., Setia, L., 2004, An Introduction to Computer Graphics using OpenGL, Sommer Campus 2004 Kurs, http://lmb.informatik.uni-freiburg.de/people/reisert/opengl/openGLKurs.pdf (15 Mayıs, 2007).

[23] The Centre for Development of Advanced Computing, Mumbai, Computer Graphics Course CG-2004, Lecture Notes, http://sarathi.cdacmumbai.in/education/apgdst/cgfac/lectures/ (15 Mayıs, 2007).

[24] NeHe Neon Helium Productions, OpenGL Tutorials, http://nehe.gamedev.net (1 Eylül, 2007).

[25] The Code Project, Creating an OpenGL view on a Windows Form, http://www.codeproject.com/managedcpp/OpenGLViewWinForms.asp (27 Mart, 2007).

[26] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Ray_tracing (5 Şubat, 2007).

[27] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Scanline_rendering (5 Şubat, 2007).

[28] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Ray_casting (5 Şubat, 2007)

[29] ACM SIGGRAPH Education Committee, http://www.siggraph.org/education/materials/HyperGraph/raytrace/rtrace0.htm (7 Eylül, 2007).

[30] ACM SIGGRAPH Education Committee, http://www.siggraph.org/education/materials/HyperGraph/raytrace/rt_java/raytrace.html (7 Eylül, 2007).

[31] Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Rotation_matrix (26 Nisan, 2007).

[32] Homepage of Paul Bourke at University of Western Australia, http://local.wasp.uwa.edu.au/~pbourke/geometry/planeline/ (21 Temmuz, 2007).

[33] International Education Software, Manipula Math Applet Collections, http://www.ies.co.jp/math/products/index.html (8 Eylül, 2007).

[34] Oregon State University, Department of Mathematics, Undergraduate Resources for Calculus http://www.math.oregonstate.edu/home/programs/undergrad/CalculusQuestStudyGuides/vcalc/dotprod/dotprod.html (8 Eylül, 2007).

[35] Paul's Online Math Notes at Lamar University, http://tutorial.math.lamar.edu/Classes/CalcIII/EqnsOfPlanes.aspx (8 Eylül, 2007).

[36] Aerospaceweb.org, http://www.aerospaceweb.org/question/electronics/q0168.shtml (30 Mayıs, 2005).

[37] Spick, M., 2000, Brassey's Modern Fighters: The Ultimate Guide to In-Flight Tactics, Technology, Weapons, and Equipment, Pegasus Publishing.

EKLER

EK 1. RCS ANALİZİ ALANINDAKİ TİCARİ PROGRAMLAR

- **Program:** Lucernhammer MT

  Firma: Tripoint Industries, Inc.

  Internet: http://lucernhammer.tripointindustries.com/lucern_brief.html

  Tanımı: Lucernhammer MT is a high-frequency (or asymptotic) radar cross section (RCS) calculation engine. It is designed around the high-resolution triangle mesh CAD model, or "facet file." It will read Xpatch/ACAD compatible facet and edge files directly with no modifications. Lucernhammer MT also has a built-in edge generator which allows it to generate the diffracting edge list on the fly if desired.

- **Program:** Epsilon™

  Firma: Roke Manor Research Ltd.

  Internet: http://www.roke.co.uk/epsilon/

  Tanımı: Epsilon™ is a software tool designed to predict the Radar Cross Section (RCS) of a target directly from its geometrical description. Epsilon™ is an analysis tool which provides support for radar system modeling. It allows a designer to evaluate the Radar Signature of his design and to investigate how that signature relates to the shape and orientation of the design.

- **Program:** Artemisia

  Firma: Millimeter Wave Technology Inc.

  Internet: http://www.mwt-materials.com/ARTEMISIA-RCS%20demo/ARTEMISIA%20Brochure.pdf

  Tanımı: –

- **Program:** SIG2D™

  Firma: Science Applications International Corporation (SAIC).

  Internet: http://www.saic.com/products/software/sig2d/

  Tanımı: SIG2D™ (2-Dimensional Signature Prediction/Radar Cross Section Analysis Code) is a surface integral computer code for radar cross section analysis of objects that exhibit two-dimensional translational symmetry.

- **Program:** SIGLBC™

  Firma: Science Applications International Corporation (SAIC).

  Internet: http://www.saic.com/products/software/siglbc/

  Tanımı: SIGLBC™ (3-Dimensional Signature Prediction/Radar Cross Section Analysis Code Using Moment-Method Approach) rigorously computes the RCS (Radar Cross Section) of systems/sub-systems and in-situ antenna patterns using generalized surface integral equations. SIGLBC has been developed with focus on the RCS designer. The formulation is selected to be robust in an environment of complex 3D shapes, optimization and fast incremental updates have been a paramount concern from the onset, and the code has been engineered for easy implementation on high speed platforms.

- **Program:** Xpatch® Electromagnetic Simulation Software

  Firma: Science Applications International Corporation (SAIC).

  Internet: http://www.saic.com/products/software/xpatch/

  Tanımı: The Xpatch® toolkit is a set of prediction codes and analysis tools that use the shooting-and-bouncing ray (SBR) method to predict realistic far-field and near-field radar signatures for 3D target models.

- **Program:** RiKA

  Firma: C2Tech

  Internet: http://www.ctech.com.tr/rika.htm

  Tanımı: RiKA (Radar izi Kestirim ve Analiz) is a radar signature prediction tool based-on high frequency electromagnetic techniques. Design of stealthy platforms (such as planes, ships, and land vehicles) with low radar cross-section (RCS) values is a costly process. RiKA works on a CAD model of a structure to predict its radar signature. It relies mainly on geometrical optics, physical optics and first order diffraction techniques to accurately predict the RCS value, the range profile, scattering centers of a platform for given frequencies and look angles. Material coatings on the platform are also taken into account. RiKA, by virtue of an added option, can be used in calculating the radiation pattern of high frequency antennas mounted on the platform.

EK 2. YAPILAN BENZER TEZ ÇALIŞMALARI

- **Tez adı:** Development of code for a physical optics radar cross section prediction and analysis application

  Yazar adı: Filippos Chatzigeorgiadis

  Yılı: 2004

  Öz: The significance of the Radar Cross Section (RCS) in the outcome of military engagements makes its prediction an important problem in modern Electronic Warfare. The POFACETS program, previously developed at the Naval Postgraduate School (NPS), uses the Physical Optics method to predict the RCS of complex targets, which are modeled with the use of triangular facets. The program has minimum computer resource requirements and provides convenient run–times. This thesis upgraded, enhanced and expanded the functionalities and capabilities of the POFACETS program. The new functionalities were implemented by upgrading the Graphical User Interface and model database, allowing the creation of models with an unlimited number of facets, providing capabilities for the automatic creation of models with standard geometric shapes, allowing the combination of existing target models, providing capabilities for sharing target models with commercial CAD programs, and creating new display formats for RCS results. The new computational capabilities include the development of a user–updateable database of materials and coatings that can be applied to models in one or multiple layers, and the computation of their effects on the models' RCS. Also implemented are the computations of the ground's effect on the RCS, and the exploitation of symmetry plane s in models, in order to decrease run–time for RCS prediction.

- **Tez adı:** Implementation of physical theory of diffraction for radar cross section calculations

  Yazar adı: Alper Kürşat Öztürk

Yılı: 2002

Öz: A computer program which uses the Physical Theory of Diffraction (PTD) method to calculate the Radar Cross Section (RCS) of perfectly conducting tar- gets with arbitrary shape is developed. Given an arbitrary surface, it is first meshed using planar triangles. The area of each triangle is restricted to be smaller than 0:005¸2 in order to obtain a good approximation to the actual surface. After meshing, Physical Optics (PO) surface integral is numerically evaluated over the whole surface. If the surface has edges or wedges, diffractions originating from these edges play a significant role in the overall scattered field. This part of the diffracted field is calculated using PTD-EEC method. Calculation of the edge currents is made possible by canonically modeling the arbitrary-shaped edge. If the surface of the scatterer has thin wires attached to it, then the thin wire scattering formulation in the literature is applied. Expressions for scattering mechanism on a straight wire are based on diffraction, attachment, reflection and launch. The results get sufficiently accurate especially for electrically large bodies.

- **Tez adı:** Radar cross section of complex targets

  Yılı: 1989

  Yazar adı: Youssef, N.N.

  Öz: A summary of the development and verifications of a computer code, RECOTA (return from complex target), developed at Boeing Aerospace for calculating the radar cross section of complex targets is presented. The code utilizes a computer-aided design package for modeling target geometry in terms of facets and wedges. It is based on physical optics, physical theory of diffraction, ray tracing, and semi empirical formulations, and it accounts for shadowing, multiple scattering and discontinuities for monostatic calculations.

- **Tez adı:** Radar cross sections of standard and complex shape targets

  Yılı: 1974

  Yazar adı: Sohel, M. S.

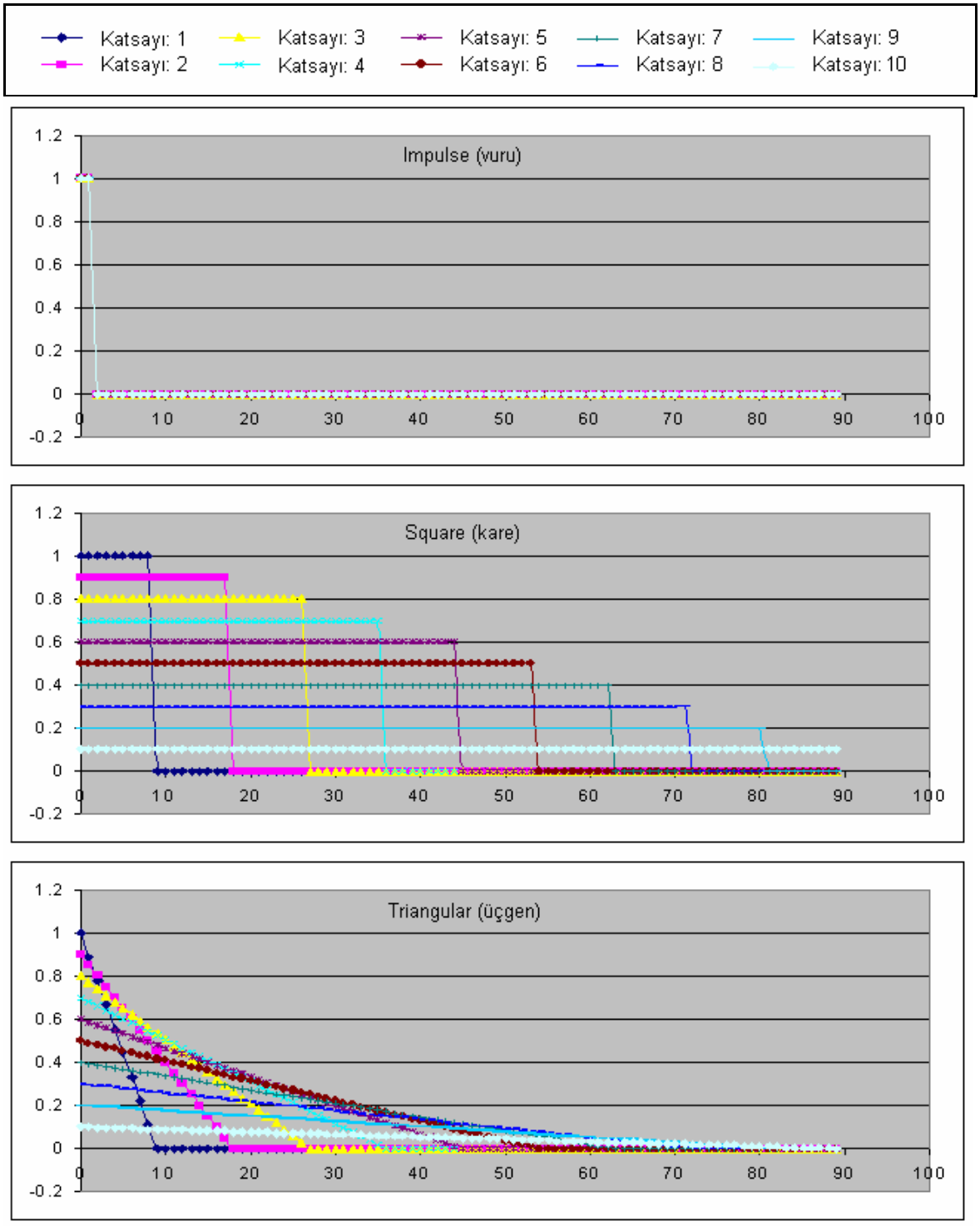
  Öz: The theoretical, analytical, and experimental results are described for radar cross sections (RCS) of different-shaped targets. Various techniques for predicting RCS are given, and RCS of finite standard targets are presented. Techniques used to predict the RCS of complex targets are made, and the RCS complex shapes are provided.

EK 3. ASCII FORMAT STL DOSYASI ÖRNEĞİ

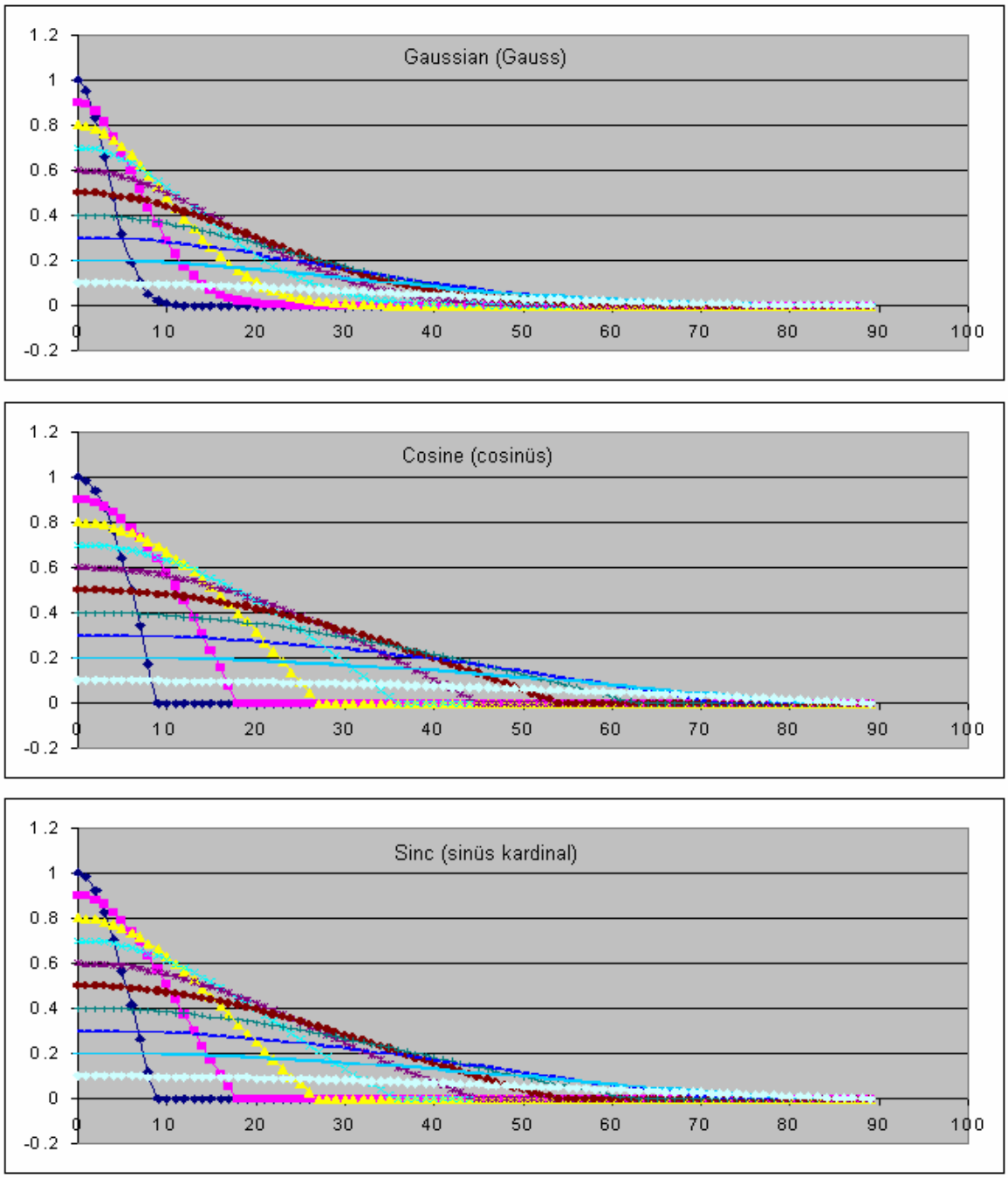
Bir köşesi kartezyen koordinat sistemi merkezinde olup kenarları eksenlere paralel ve 1 birim uzunluğunda olan basit bir küp şeklini tanımlamak için oluşturulmuş ASCII formatındaki STL dosyası içeriği şu şekildedir:

```
Solid SimpleCube
  facet normal 1 0 0
    outer loop
      vertex 1 1 1
      vertex 1 0 1
      vertex 1 0 0
    endloop
  endfacet
  facet normal 1 0 0
    outer loop
      vertex 1 1 0
      vertex 1 1 1
      vertex 1 0 0
    endloop
  endfacet
  facet normal -1 0 0
    outer loop
      vertex 0 0 0
      vertex 0 0 1
      vertex 0 1 1
    endloop
  endfacet
  facet normal -1 0 0
    outer loop
      vertex 0 1 0
      vertex 0 0 0
      vertex 0 1 1
    endloop
  endfacet
  facet normal 0 1 0
    outer loop
      vertex 1 1 1
      vertex 1 1 0
      vertex 0 1 0
    endloop
  endfacet
  facet normal 0 1 0
    outer loop
      vertex 0 1 1
      vertex 1 1 1
      vertex 0 1 0
    endloop
  endfacet
  facet normal 0 -1 0
    outer loop
      vertex 0 0 0
      vertex 1 0 0
      vertex 1 0 1
    endloop
  endfacet
  facet normal 0 -1 0
    outer loop
      vertex 0 0 1
      vertex 0 0 0
      vertex 1 0 1
    endloop
  endfacet
  facet normal 0 0 1
    outer loop
      vertex 1 1 1
      vertex 0 1 1
      vertex 0 0 1
    endloop
  endfacet
  facet normal 0 0 1
    outer loop
      vertex 1 0 1
      vertex 1 1 1
      vertex 0 0 1
    endloop
  endfacet
  facet normal 0 0 -1
    outer loop
      vertex 0 0 0
      vertex 0 1 0
      vertex 1 1 0
    endloop
  endfacet
  facet normal 0 0 -1
    outer loop
      vertex 1 0 0
      vertex 0 0 0
      vertex 1 1 0
    endloop
  endfacet
endsolid
```

EK 4. YÖNLÜLÜK FONKSİYONLARININ FARKLI YÜZEY SAÇINIM KATSAYILARI İÇİN ALDIKLARI DEĞERLER



Şekil E4.1 Yönlülük fonksiyonlarının farklı yüzey saçınım katsayıları için aldıkları değerler.



Şekil E4.1 devam ediyor.

EK 5. ÖRNEK UÇAKLAR İÇİN ORTALAMA RCS DEĞERLERİ

Çizelge E5.1. Bazı uçaklar için ortalama RCS değerleri ([36]).

Uçak adı	**RCS (dBsm)**
F-15 Eagle	26
F-4 Phantom II	20
B-52 Stratofortress	20
Su-27	12
B-1A	10
F-16 Fighting Falcon	7
B-1B Lancer	0.09
F-18E/F Super Hornet	0
Rafale	0
Typhoon	-3
AGM-86 ALCM	-6
BGM-109 Tomahawk	-13
SR-71 Blackbird	-18.5
F-22 Raptor	-22
F-117 Nighthawk	-25
B-2 Spirit	-28.5
AGM-129 ACM	-30
Boeing Bird of Prey	-70

EK 6. RCSP PROGRAMI KAYNAK KODU

RCSP programı kaynak kodu Visual Studio .NET projesi olarak aşağıdaki CD'de bulunmaktadır.

ÖZGEÇMİŞ

Adı Soyadı : Halit AYANLI

Doğum Yeri : Eskişehir

Doğum Yılı : 1979

Medeni Hali : Bekar

Eğitim ve Akademik Durumu

Ortaokul	:	1991 – 1995	Eskişehir Kılıçoğlu Anadolu Lisesi
Lise	:	1995 – 1998	Eskişehir Fatih Fen Lisesi
Lisans	:	1998 – 2003	Orta Doğu Teknik Üniversitesi
			Elektrik ve Elektronik Mühendisliği Bölümü

Yabancı Dil : İngilizce

İş Tecrübesi

2004 – …	MİKES (Mikrodalga Elektronik Sistemler) A.Ş.
	Yazılım Mühendisi