T.C.

BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MATEMATİK BİLGİSAYAR ANABİLİM DALI
BİLGİ TEKNOLOJİLERİ BİLİM DALI

# 3D YEE MODELİNİN MATLAB İLE GERÇEKLENMESİ

(Yüksek Lisans Tezi)

Tezi Hazırlayan: **Hasan ÖVÜÇ**

İstanbul, 2010

T.C.

BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MATEMATİK BİLGİSAYAR ANABİLİM DALI
BİLGİ TEKNOLOJİLERİ BİLİM DALI

# 3D YEE MODELİNİN MATLAB İLE GERÇEKLENMESİ

(Yüksek Lisans Tezi)

Tezi Hazırlayan: **Hasan ÖVÜÇ**
Öğrenci No:
070862002

Danışman:
Yrd. Doç. Dr. *Turhan KARAGÜLER*

İstanbul, 2010

## YEMİN METNİ

Yüksek lisans tezi olarak sunduğum "3D Yee Modelinin MATLAB ile Gerçeklenmesi" başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere uygun şekilde tarafımdan yazıldığını, yararlandığım eserlerin tamamının kaynaklarda gösterildiğini ve çalışmamın içinde kullanıldıkları her yerde bunlara atıf yapıldığını belirtir ve bunu onurumla doğrularım. 15/06/2010

Aday: Hasan ÖVÜÇ

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

# YÜKSEK LİSANS TEZ SAVUNMA SINAVI SONUÇ TUTANAĞI

**Beykent Üniversitesi**
**Fen Bilimleri Enstitüsü Müdürlüğü'ne,**

Aşağıda tez adı belirtilen yüksek lisans öğrencisi. ..070862002..... no'lu ..... **Hasan ÖVÜÇ**'ün....01/07/2010... tarihinde yapılan tez savunma sınavı[1] sonucunda .90.....dakika süreyle sunduğu ve savunduğu tezi hakkında[2] oybirliğiyle/~~oyçokluğuyla,~~ Kabul/~~Red/Düzeltme~~(.....ay içinde) kararı verilmiştir.

Bilgilerinize saygılarımızla arz ederiz.

---

**AnabilimDalı**:. Matematik-Bilgisayar.....

**Programı** : Bilgi Teknolojileri.....

**Tez Başlığı**[3] : *3D Yee Modelinin MATLAB ile Gerçeklenmesi*

.....

.....

---

| **Tez Sınav Jürisi** | **Öğretim Üyesi** | **İmza** |
|---|---|---|
| **Danışman** | : .....Yrd.Doç.Dr. Turhan Karagüler | ..... |
| **Üye** | : .....Yrd.Doç.Dr. Gökhan Silahtaroğlu | ..... |
| **Üye** | : .....Yrd. Doç. Dr. Haluk Kul | ..... |

---

[1] Jüri üyeleri söz konusu tezin kendilerine teslim edildiği tarihten itibaren en geç bir ay içinde toplanarak öğrenciyi tez savunma sınavına alır. Belirlenen günde yapılamayan jüri toplantısı, katılanların hazırladığı bir tutanakla enstitü yönetimine bildirilir. Bu durumda jüri en geç onbeş gün içinde toplanarak adayı tez savunma sınavına alır. Tez savunma sınav süresi en az 45 dakikadır. Yüksek lisans tez savunma sınavı, tez çalışmasının sunulması ve bunu izleyen soru-yanıt bölümlerinden oluşur ve dinleyiciye açıktır. (Beykent Lisansüstü eğitim ve Öğretim Yönetmeliği-Madde30-3)

[2] Tez sınavının tamamlanmasından sonra jüri, tez hakkında "kabul", "düzeltme" veya "red" kararı verir. Jüri başkanı, jüri üyelerince imzalanmış sınav tutanağını, tez sınavını izleyen üç gün içinde ilgili enstitü yönetimine teslim eder. Tezi başarısız bulunan öğrencinin Enstitü ile ilişiği kesilir. Tezi hakkında düzeltme kararı verilen öğrenci en geç üç ay içinde gerekli düzeltmeleri yaparak ve yönetmelikte belirtilen usullere uygun olarak tezini aynı jüri önünde yeniden savunur. Bu savunma sınavında da tezi kabul edilmeyen öğrencinin enstitü ile ilişiği kesilir. (Beykent Lisansüstü eğitim ve Öğretim Yönetmeliği-Madde30-4)

[3] İleride doğabilecek aksaklıkların engellenmesi için tezin başlığının yazılması gerekmektedir.

# 3D YEE MODELİNİN MATLAB İLE GERÇEKLENMESİ

Tezi Hazırlayan: **Hasan ÖVÜÇ**

## ÖZET

Günümüzde MATLAB, mühendislik ve temel bilimin birçok problemi için en yaygın ve pratik programlama ortamlarından biri olarak kabul edilir. MATLAB genellikle 1D veya 2D modellemeler için ilk tercih olmasına karşın, problem boyutunun hayli yüksek değerlere çıktığı 3D ve 4D modellemelerde ise performans yavaşlığı nedeniyle aynı ilgiyi görmemektedir. Bu tezde, Zaman Uzayında Sonlu Fark (FDTD) metodu yardımıyla, daha önce FORTRAN dilinde kodlanan, hesaplı elektromanyetizmadan bir 3D problem (Kayıplı Dielektrik için 3D YEE Modeli) MATLAB aracılığıyla çözülmüştür. Başlangıçta herhangi bir iyileştirme yapılmaksızın düz MATLAB uyarlaması şeklinde çalıştırılan program, FORTRAN'da dokuz saniyede sonlanmışken MATLAB'da sekiz yüz saniyelik bir çalışma zamanı oluşmuştur. Ancak birtakım programlama ve MATLAB'a özgü tekniklerle, çalışma zamanı yüz saniyelere kadar indirilebilmiştir. Bu kabul edilebilir bir orandır. MATLAB'ın hazır yardımcı grafiksel ara yüzlerin varlığı ve programlama konusunda mevcut popüler dillere göre kolaylığı dikkate alındığında, kapsamlı projeler için de ilk tercih edilebilecek bir programlama ortamı olması beklenmektedir.

**Anahtar Kelimeler:** MATLAB ile Programlama, Zaman Uzayında Sonlu Fark (FDTD) Metodu, Hesaplamalı Elektromanyetizma

# SIMULATING 3D YEE MODEL WITH MATLAB

Presented By: **Hasan ÖVÜÇ**

## ABSTRACT

MATLAB is accepted as one of the most popular and practical programming environment for the problems of engineering and science. Usually, it is almost the first choice for the problems modelled in 1D and 2D. However, the problems involving 3D or 4D cases, MATLAB loses its popularity due to having rather poor computing run-time. In this work, a problem from the field of computational electromagnetism, 3D YEE Model of Lossy Dielectric, is fully coded by using MATLAB. The problem is discretized by means of Finite Difference Time Domain (FDTD) method. At the first stage, direct conversion from FORTRAN to MATLAB is tested and found cpu run time as about eight hundred seconds for MATLAB against six seconds of FORTRAN. However, later some modifications are carried out on the code and the cpu time is reduced to as low as around one hundred seconds which is reasonable considering the size of the problem of interest.

**Keywords**: Programming with MATLAB, FDTD, Computational Electromagnetism

# TEŞEKKÜR

Bu çalışmayı oluşturmamda sürekli desteği, değerli katkıları ve sabrından dolayı Yrd. Doç. Dr. Turhan KARAGÜLER'e teşekkür ederim.

# İÇİNDEKİLER

**Sayfa No**

# ŞEKİLLER LİSTESİ

# KISALTMALAR VE BİRİMLER DİZİNİ

| KISALTMA | TANIM | BİRİM |
|---|---|---|
| E | Elektrik Alan | Volt/metre |
| H | Manyetik Alan Şiddeti | Amper/metre |
| B | Manyetik Alan | Tesla |
| D | Deplasman Vektörü | Coulomb/metre$^2$ |
| J | Yüzeysel Akı Yoğunluğu | Amper/metre$^2$ |
| T | Zaman | Saniye |
| C | Işığın Boşluktaki Hızı | Metre/saniye |
| N, n | Zaman Atımı | |
| Curl | Rotasyon | |
| Div | Diverjans | |
| V | Hız | Metre/saniye |
| $\Lambda$ | Dalga Boyu | Metre |
| $\mu$ | Manyetik Geçirgenlik | Henry/metre |
| $\sigma$ | Elektriksel İletkenlik | $1/\Omega$ |
| $\rho$ | Hacimsel Yük Yoğunluğu | Coulomb/metre$^3$ |
| $\varepsilon$ | Dielektrik Geçirgenlik | Farad/metre |
| $\alpha$ | Sönüm Katsayısı | Neper/metre |
| $\beta$ | Faz Sabiti | Radian/metre |
| $\partial/\partial t$ | Zamana Bağlı Kısmi Türev | |

# 1 GİRİŞ

Bu tezde sayısal elektromanyetizmada özellikle yüksek frekans uygulamalarında kullanılan Yee modelinin MATLAB ile gerçeklenmesi amaçlanmıştır. Yee modelinin programlanmasında, mevcut uygulamalarda ağırlıklı olarak FORTRAN dili kullanılmıştır. Son yıllarda kullanımı yaygınlaşan ve standart yüksek seviye dillere karşı özellikle üstünlükleri olan MATLAB bilimsel hesaplama alanında en çok kullanılan araçlardan biridir. MATLAB programlama ortamının, sayısal elektromanyetizmanın komplike sayılan bir probleminin çözümü üzerinden sınanması yoluyla diğer programlama ortamlarıyla karşılaştırma yapılabilmesi hedeflenmiştir.

Çalışma, uygulamalı ve hesaplamalı elektromanyetizma problemlerinin çözümünde kullanımı en yaygın olan yöntemlerin başında gelen, Zaman Uzayında Sonlu Farklar (Finite Difference Time Domain- FDTD) yaklaşımını esas almıştır [1-3]. Yöntemin uygulandığı problem olarak en genel elektromanyetik alan uygulaması olan Kayıplı Dielektrik seçilmiştir [4-5]. Ayrıca problem üç boyutlu (3D) ve zaman değişkenli olarak modellendiğinden, yine olabilecek en karmaşık durum test edilmiştir.

Tezin ikinci bölümünde test problem ve bu problemi temsil amacıyla oluşturulan model tanıtılmıştır. Elektromanyetizmayı komple tanımlayan Maxwell Denklemlerinden yola çıkılarak probleme ilişkin elektrik ve manyetik alan denklemleri elde edilmiş olup, denklemlerin sonlu farklara ayrıştırılmış biçimleri ayrıca üretilmiştir. Zaman Uzayında Sonlu Farklar Metodu da bu bölümde tanıtılmıştır.

Tezin üçüncü bölümünde, problem için daha önce Kunz ve arkadaşları [6] tarafından kullanılan ve aynı zamanda daha bir çok araştırma çevresinde programlama ortamı olarak düşünülen FORTRAN dili ve tez için bu dile alternatif seçilen MATLAB skript dili tanıtılarak, avantaj ve dezavantajları tartışılmıştır. Dördüncü bölümde MATLAB diliyle elde edilen sonuçlar sunulmuş olup, MATLAB uygulamalarında önemli bir sorun olarak ortaya çıkan performans (hız), farklı denemelerle iyileştirilmeye çalışılmıştır. Bu cpu run-time

iyileştirmeleri karşılaştırılmalı olarak grafiklerle gösterilmiştir. Tezin son bölümünde tez kapsamında yapılan çalışmalar kısaca özetlenerek, gelecekte yapılabilecek ilaveler ayrıca tartışılmıştır.

## 1.1 FDTD Yöntemine Genel Bakış

FDTD yöntemi ilk olarak Kane S. Yee tarafından 1966 yılında yazılan "Numerical Solutions of Initial Boundary Value Problems Involving Maxwell's Equations in Isotropic Media" makalesinde kullanılmıştır [1]. Yöntem, esas olarak sonlu farklar ifadelerini her bir zaman adımı için elde ederek, standart alan konum değişimlerini zamana göre ifade eder. Bir başka deyişle, 3D bir probleme, zaman boyutunuda ekleyerek 4D bir problem çözümünü gerçekleştirir.

FDTD her yıl yaklaşık olarak dünya çapında iki binin üzerinde makaleye konu olmaktadır. FDTD karmaşık geometriler içerisindeki yalıtkan veya manyetik materyallerin elektromanyetik izdüşümlerini hesaplayabilir. FDTD'ye artan ilginin en büyük nedeni ise diğer yöntemler ile karşılaştırıldığında oldukça basit uygulanabilmesidir. Bu alandaki diğer teknikler ise FIT (Finite Integration Technique), TLM (Transmission Line Modeling), FETD (Finite-Element Time-Domain), PSTD (the Kabaspectral Time-Domain), MRTD (Multiresolution Time-Domain)'dir. Tüm bu tekniklerin verdiği sonuçlar başarılı ve doğrudur. FDTD ise içlerinde basit ve güçlü olması ile ayrılmaktadır.

FDTD modellemesi paralel olarak çalışan bilgisayarlara kolaylıkla uygulanabilir. Bu yöntem sayesinde her bilgisayar tüm sistem ile değil sadece ona en komşu olan diğer bilgisayar ile etkileşim içerisinde olmasının yeterli olmasıdır. Burada maliyeti sistemin hızı oluşturmaktadır. Sistemin hızı ise problem uzayının büyüklüğü ile doğru orantılıdır. Problem uzayı ne kadar büyük olursa problem çözümleri o kadar uzun olmakta ve masrafları arttırmaktadır. Bundan dolayı uzayı ve dalga boyunu maliyetlerden dolayı mümkün olduğu kadar küçük tutulması gerekmektedir. Örnek olarak, 100x100x100 hücreden oluşan bir uzaya dalga boyu on FDTD hücresi boyunda bir dalga oluşturduğumuzda bu uzay ancak on dalga boyu büyüklüğünde olabilmektedir.

FDTD problem sonuçları hacimsel olduğundan dolayı kolaylıkla ince plaka ve antenlere uygulanabilmektedir. Üç boyutlu çözüm imkanı özellikle MATLAB kullanılarak dalganın hareketini ve etkileşimlerini görsel olarak inceleme imkanı vermektedir.

## 1.2 Kullanım Alanları

FDTD sağladığı avantajları nedeniyle daha çok elektromanyetik modelleme, simülasyon ve analiz aracı olarak kullanılmaktadır. FDTD'nin yetenekleri şu şekilde sıralanabilir:

- Sistem rezonansları için merkezi geniş band yanıt tahminleri
- Sıralı üç boyutlu model geometrileri
- Herhangi yalıtkan bir obje ile tam yalıtkan, gerçek metal, düşük ve sıfır yapısal yalıtkanlığa sahip frekans bağımlı parametreler ile etkileşimlerinin modellenmesi (kayıplı yalıtkanlar, manyetik materyaller, eş yönsüz materyaller)
- Yakın alanlardan uzak alanlara taşınan her tip dalganın hesaplanmasında kullanılmaktadır
    - Dağılmış alanlar
    - Anten kalıpları
    - Radar kesitleri
    - Akımlar, güç yoğunluğu

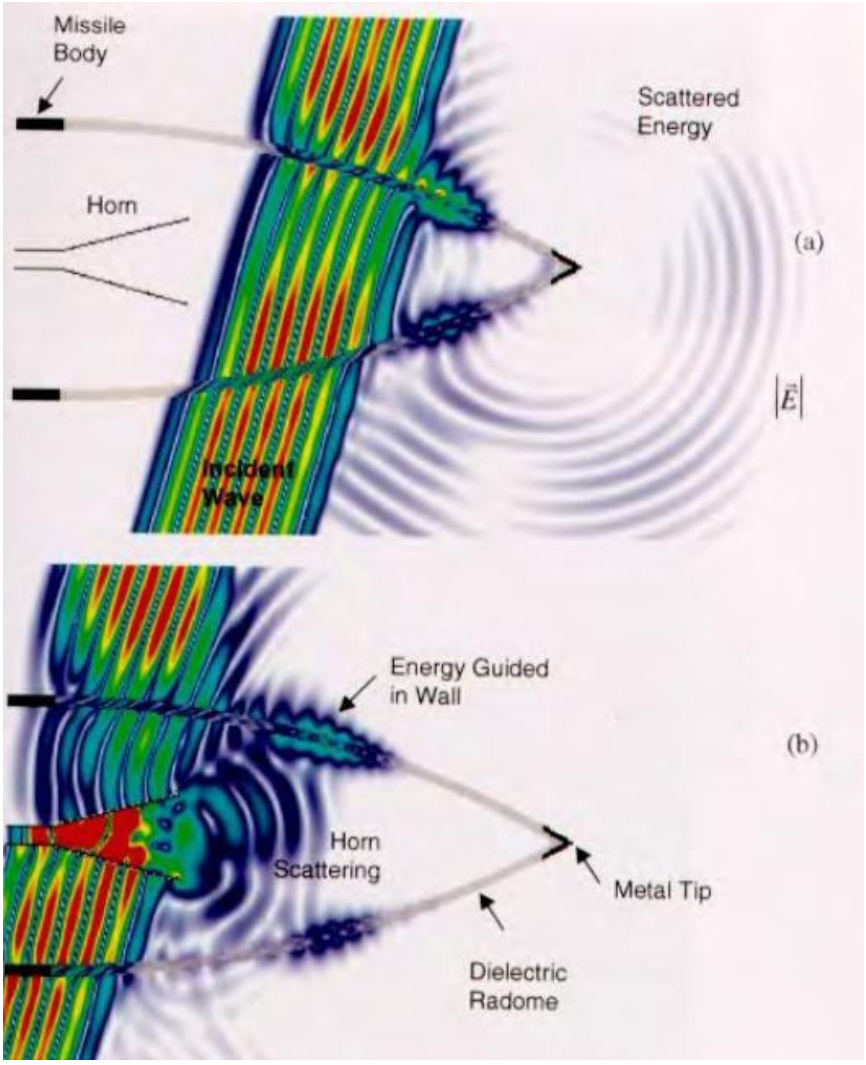
Bu yetenekler aşağıda sıralanan sistemlerde kullanılmaktadır;

- EMP (Elektromanyetik Darbe)
- HPM (Yüksek Enerjili Mikrodalga)
- Radar
- Lazerler

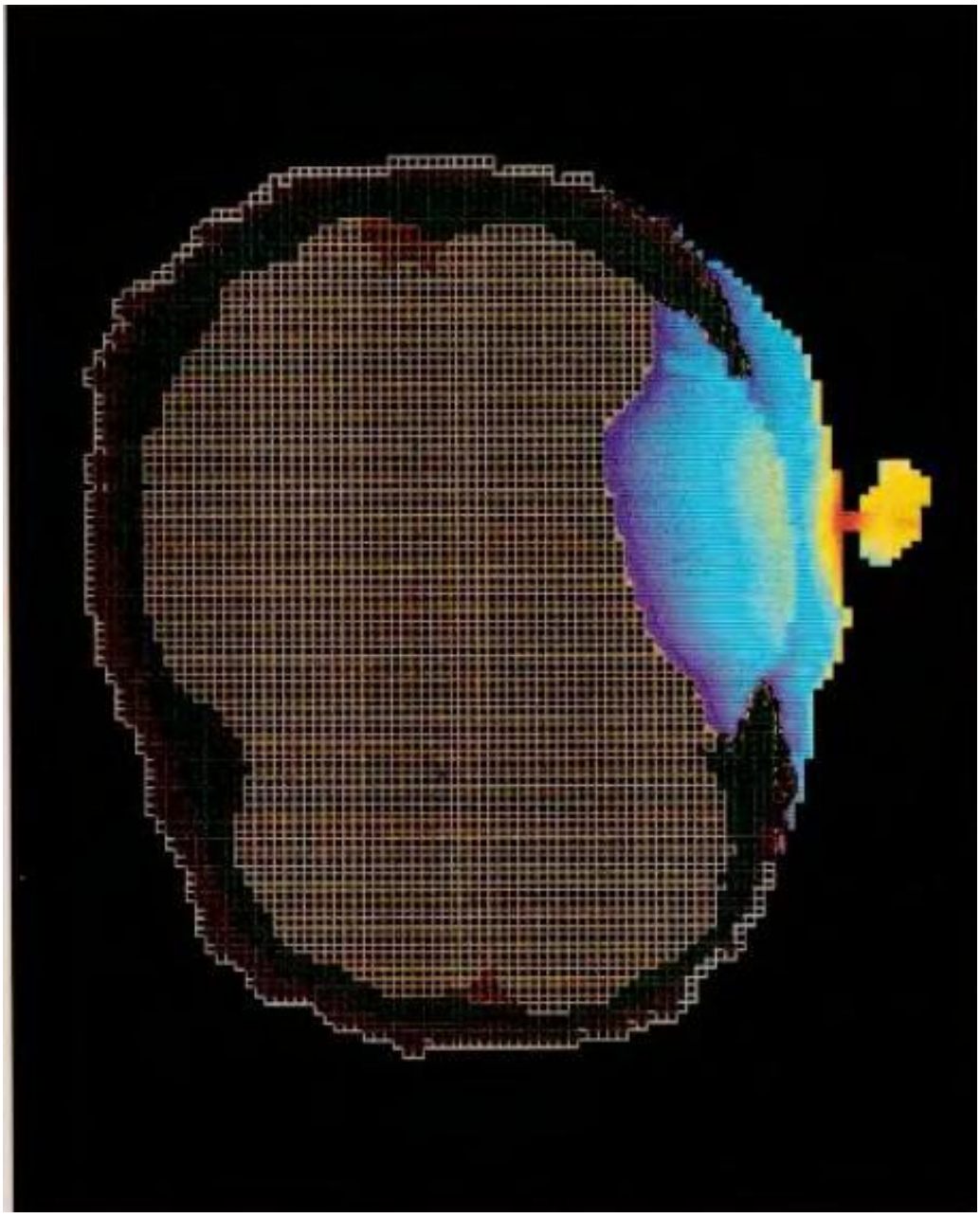
Aynı zamanda yukarıda bahsedilen sistemlerin atmosfer veya yeraltı ortamındaki canlı veya cansız, büyükten küçüğe her türlü obje üzerindeki etkileri simüle edebilir. Bu objeler:

- Aerosoller
- Sığınaklar
- Uçaklar
- İnsanlar
- Uydular
- Gömülü antenler

FDTD'in dağıtılmış alan hesaplaması, ilk olarak F-111 hayalet uçaklarının EMP (Elektro-Magnetic Pulse) alanında yüzey akımlarının düşürülmesi çalışmalarında kullanılmıştır. (Şekil 1.1)'de radar güdümlü bir füzenin FDTD yöntemiyle gönderilen dalganın hareketi incelenmektedir. (Şekil 1.2)'de insan beyninin cep telefonu kullanımı sırasında "SAR – Specific Absorption Rate" dağılımını gösteren FDTD simülasyonu gösterilmektedir.




Şekil 1.1 Radar Güdümlü Füze FDTD Simülasyonu [2]



Şekil 1.2 Cep Telefonu Kullanımını Gösteren FDTD Simülasyonu [2]

# 2 TEST PROBLEM VE MODEL

Çalışmanın uygulama alanı elektromanyetizma olup, modellenen test örneği olarak kayıplı dielektrik bölgeli elektromanyetik dalga problemi ele alınmıştır. Bu bölümde test problemini temsil eden diferansiyel denklemler ve sayısal çözümünde kullanılan ayrıklaştırılmış sonlu fark eşdeğerleri tanıtılmıştır. Tezin ana temasının programlama ortamı olması nedeniyle, test probleminin dinamiğine ilişkin kısımlar kısmen yüzeysel olarak ele alınmıştır. Bölümde ayrıca FDTD yöntemi ve kullanılan YEE modeli de ayrıca detaylandırılmıştır.

## 2.1 Yönetsel Denklem

Genel olarak elektromanyetik teori literatürde Maxwell Denklemleri olarak bilinen ve aşağıda verilen bir dizi diferansiyel eşitlik (2.1-2.4) ve bünye denklemleri (2.5-2.7) ile tam olarak ifade edilebilmektedir [7].

$$\mathrm{Curl}\,\overline{\mathrm{E}} = -\frac{\partial \overline{\mathrm{B}}}{\partial \mathrm{t}} \tag{2.1}$$

$$\mathrm{Curl}\overline{\mathrm{H}} = \overline{\mathrm{J}} + \frac{\partial \overline{\mathrm{D}}}{\partial \mathrm{t}} \tag{2.2}$$

$$\mathrm{div}\overline{\mathrm{B}} = 0 \tag{2.3}$$

$$\mathrm{div}\overline{\mathrm{D}} = \rho \tag{2.4}$$

Bu denklemlerde, $\vec{B}[T]$ Manyetik Alan, $\vec{H}[A/m]$ Manyetik Alan Şiddeti, $\vec{E}[V/m]$ Elektrik Alan, $\vec{D}[C/m^2]$ : Deplasman Vektörü, $\vec{J}[A/m^2]$ Yüzeysel

Akım Yoğunluğu, $\rho\left[C/m^3\right]$ Hacimsel Yük Yoğunluğu olup aşağıdaki Bünye Denklemleri ve Ohm Yasası ile birbiriyle ilişkilendirilmişlerdir.

$$\overline{B} = \mu\overline{H} \tag{2.5}$$

$$\overline{D} = \varepsilon\overline{E} \tag{2.6}$$

$$\overline{J} = \sigma\overline{E} \tag{2.7}$$

Yukarıda (2.5-2.7) denklemlerinde görülen yer ve ortamın özelliklerini belirleyen $\varepsilon$, $\mu$ ve $\sigma$ parametreleri sırasıyla dielektrik geçirgenlik [F/m], manyetik geçirgenlik [H/m] ve iletkenlik [1/Ω] olarak tanımlanmış olup, zamandan bağımsız oldukları varsayılmıştır.

Tüm bu denklemler aslında elektromanyetik teoriye ilşkin bilinen Faraday, Ampere, Ohm yasalarının Maxwell tarafından toparlanarak bir başka formda ifadesinden başka bir şey değildir. Bu denklemlere Maxwell denklemleri olarak adlandırılmasının en önemli nedeni ise denklem (2.2) yeralan $\frac{\partial \vec{D}}{\partial t}$ terimini bilinen Amper yasasına eklenmesidir. Bu eklenme sayesinde Elektrik Alanın zamanla değişiminin Manyetik Alan yaratımına yol açtığının bulunmuştur. Aynı şekilde denklem (2.1) ile ifade edilen ve Faraday Yasasının bir başka formu olan denklemdende, Manyetik Alanın zamanla değişiminin Elektrik Alan oluşturduğunun bilinmesi nedeniyle, bu alanların ortamda birbirlerini yaratarak yol almaları ve dolayısıyla dalga karakteri taşıdığı sonucuna varılmasıdır. Bu bulgunun sonuçları bilim ve teknoloji dünyasında son derece önemli bulunmuş olup, Maxwell'in bu denklem setini ifade ettiği ünlü notlarının [8], hemen sonrasında, elektromanyetik dalga yoluyla ses iletimi, sonralarında ise görüntü iletimi gibi günümüz dünyasının vazgeçilmez iletişim araçları olan telefon, radyo, televizyon gibi teknolojilerin önü açılmıştır.

Genel olarak, bu denklemlerden Elektrik Alan (E) veya Manyetik Alan (H) elimine edilince elektromanyetizmanın dalga denklemi E veya H cinsinden elde edilir. Örneğin E cinsinden elde edilen denklem (2.8), en genel durumu temsil eden Helmholtz denklemidir.

$$\Delta\overline{E} = \mu\sigma\frac{\partial\overline{E}}{\partial t} + \mu\varepsilon\frac{\partial^2\overline{E}}{\partial t^2}$$

(2.8)

Burada, "Δ", Laplasyan operatörü olup, Kartezyen Koordinat Sisteminde, Nabla operatörünün karesi olarak denklem (2.9)'daki gibi tanımlanmıştır.

$$\Delta=\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}+\frac{\partial^2}{\partial z^2}=\nabla.\nabla=\nabla^2$$

(2.9)

Laplasyan operatörü skalar veya vektörel büyüklüklere uygulanabilir. Örneğin E'nin vektörel büyüklük olmasından dolayı, her bir x,y,z bileşeni için ayrı ayrı yazılmalıdır. Helmholtz denklemi x bileşen için düzenlenirse (2.10);

$$\Delta E_x=\mu\sigma\frac{\partial E_x}{\partial t}+\mu\varepsilon\frac{\partial^2 E_x}{\partial t^2}$$

(2.10)

denklemi elde edilir. Çözümde üç boyutlu modelleme söz konusu olduğundan y ve z bileşenleri için yazılacak denklem (2.9) benzeri üçlü denklem seti bu çalışmada programlanan problem için genel yönetsel (governing) denklem olarak düşünülebilir.

Denklem (2.8) ve (2.9)'da görülebileceği gibi, Genel Dalga Denklemi (Helmholtz), çözümü oldukça karmaşık bir problemdir. Problemin analitik çözümünün elde edilmesi hemen hemen olanaksız olduğundan ancak sayısal yöntemlerle çözümü denenmektedir. Bundan dolayı bu tezde kullanılan FDTD metodu da bu yöntemlerden en etkin olanlarından biridir. Ancak uygulamada çoğu kez, kolaylığı göz önüne alındığında 1D Düzlem Dalga denklemi sayesinde de modellemeler yapılabilmekte ve birçok elektromanyetik dalga problemi çözülebilmektedir.

Örneğin z-yönünde ilerleyen bir elektromanyetik dalganın x-yönünde olduğu varsayılan E cinsinden (keza y- yönünde varsayılan H için de aynı ifade söz konusu), dalga denklemi (2.11) gibi elde edilebilir.

$$\frac{\partial^2 E_x}{\partial t^2} - v^2 \frac{\partial^2 E_x}{\partial t^2} = 0 \qquad (2.11)$$

Bu denklemde "$v$" dalganın hızını belirler ve boşlukta $v = \frac{1}{\sqrt{\varepsilon_0 \mu_0}}$ ile tanımlandığından değer olarak ışık hızını verir. Eğer problem kayıplı dielektrik bölge olursa, "$v$" 'nin yerine $\gamma = \sqrt{j\omega\mu(\sigma + jw\varepsilon)}$ veya $\gamma = \alpha + j\beta$ eşitliği ile ifade kompleks "$\gamma$" yazılarak (2.11) denklemi genelleştirilir. Burada "α" ve "β" fiziksel önemi olan sabitler olup sırasıyla sönüm katsayısı ve faz sabiti olarak adlandırılırlar.

Denklem (2.11)'in analitik çözümü, literatürde D'Alembert çözümü olarak bilinen $E = f(z - vt) + g(z + vt)$ burada "f" ve "g" herhangi fonksiyonlar olup, dalganın "+" ve "–" yönde "v" hızıyla ilerleyen bileşenlerini temsil ederler.

Bu çalışmada problem 3D olarak modellendiğinden, bir boyutlu kolaylaştırma işe yaramayacaktır.

## 2.2 FDTD Model

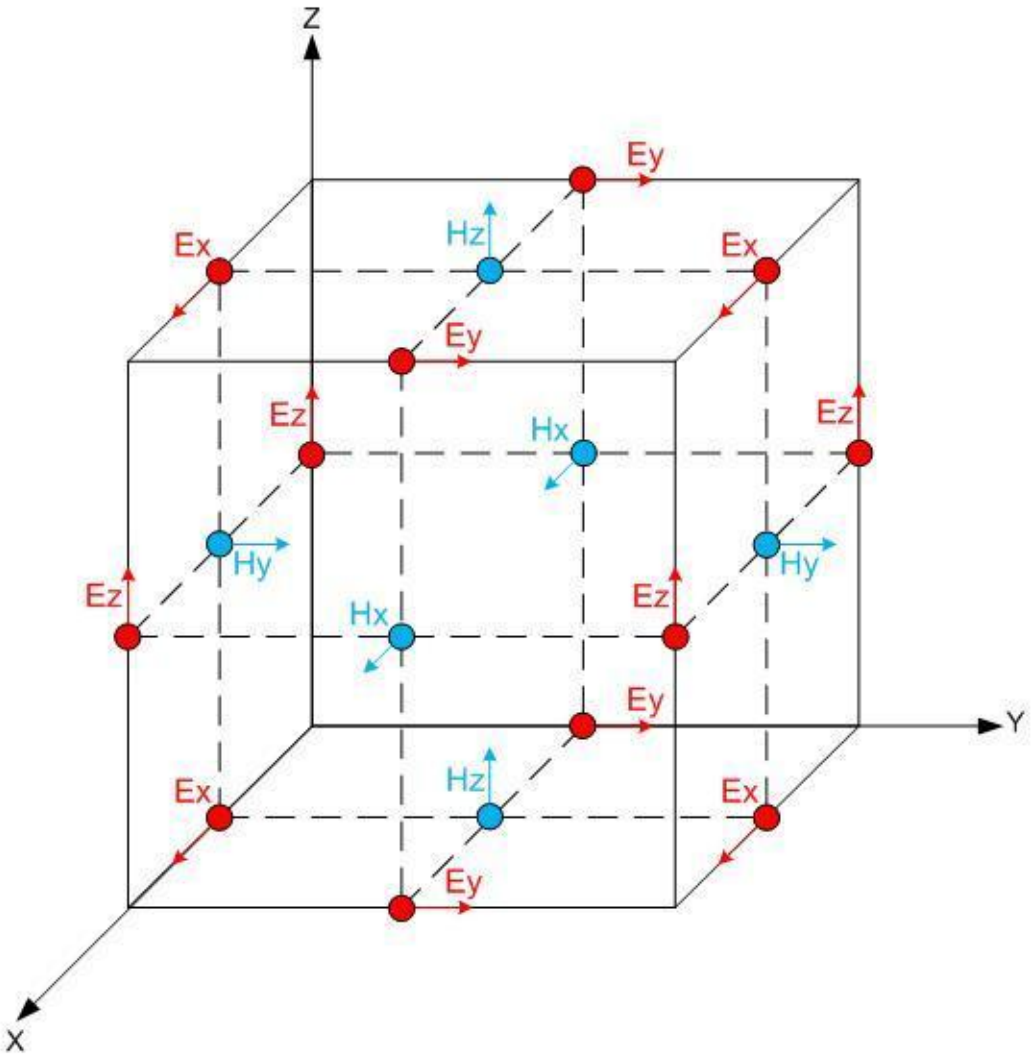
FDTD, sonlu farklar kavramını kullanarak Maxwell'in elektrik ve manyetik alan dağılım denklemleri de dahil, mühendislik ve temel bilimlerde karşılaşılan birçok problemi temsil eden diferansiyel denklemleri, uzay zaman bölgesinde çözen sayısal bir tekniktir. FDTD tekniğini merkezi farklar ortalamasını uzay zaman bölgesindeki iki Maxwell kırılım denklemine "Faraday ve Amper" uygulayarak, sonuç denklemleri üzerinden zaman adımlarında elektrik ve manyetik alan dağılımları hesaplanmaktadır.

Bu tezde seçilen problem modeli, kaynak [6]'da belirtilen Kunz'un problem modelini esas aldığından, FDTD modeli de Kunz ve arkadaşlarının kullandığı modelle örtüşmektedir. Programlama ortamı olarak FORTRAN yerine MATLAB'ın kullanılmasına karşın, Kunz ve arkadaşlarının modellerinde ve kodlarında

kullandıkları isimlendirmeler (fonksiyonlar, değişkenler, vs.) genellikle aynı tutulmuştur. Sonlu Fark denklemleri ise YEE hücresine göre elde edildiğinden, model için önce YEE hücre uzayının tanıtılması yerinde olacaktır.

## 2.3 Yee Hücre Uzayı ve Denklemler

Yee'nin hücre şeması bir dikdörtgenler ızgarasına benzemektedir [1]. (Şekil 2.1)'de görülebileceği üzere Elektrik elemanları köşelerde yer alırken manyetik elemanlar yüzeylerin merkezinde yer almaktadır. Bazı uygulamalarda YEE Hücresinin farklı örnekleri olsada, (Şekil 2.1)'de verilen hücre FDTD modellerin çoğunda aynen kullanılmıştır.




Şekil 2.1 Yee Hücre Şeması ve Elektrik-Manyetik Alan Bileşenleri Konumları [1]

YEE hücreleri I,J,K indeksleriyle tanımlanan bir uzaya yerleştirilmiştir. Bu uzayda $x = I\Delta x,\ y = \Delta x\ y,\ z = K\Delta z$ iken zamanı $t = n\Delta t$ olarak tanımlanır. Yee notasyonunda $E_z^n(I,J,K)t = n\Delta t$ zamanında $x = I\Delta x, y = J\Delta y, z = K\Delta z$ kartezyen koordinatlarındaki elektriksel alanın z bileşenini ifade etmektedir.

"I,J,K" Yee koordinatlarındaki $E_x^{scat,n}$ değeri için programlamada EXS(I,J,K) notasyonunu kullanılır. Aynı Yee hücresinde $H_y$ değeri için yani $H_y^{scat,n+1/2}$ için ise HYS(I,J,K) notasyonunu kullanılmaktadır. Zaman program içerisinde adımı n=N olarak ifade edilmektedir. Sonuçların doğruluğunu arttırmak amacıyla hesaplamalar zaman adımlarının yarı sürelerinde yapılmaktadır. Doğal bir sonuç olarak, FDTD yöntemi gereği manyetik ve elektriksel alan hesaplamaları n=N+1/2 zamanında olmaktadır. $E_x^{inc}$ için ise EXI(I,J,K) kullanılırken kayıplı elektriksel alan $\frac{\partial E_x^{inc}}{\partial t}$ için ise DEXI(I,J,K) notasyonu kullanılacaktır.

Maxwell denklemlerini Yee Hücre sistemine uyarlayabilmek için Faraday ve Amper yasasına aşağıdaki dönüşümler uygulanır.

$$\nabla \times E = -\mu \frac{\partial H}{\partial t} - \sigma_m H \tag{2.12}$$

$$\nabla \times H = -\varepsilon \frac{\partial E}{\partial t} + \sigma E \tag{2.13}$$

Maxwell'in Faraday Denklemi (2.12) ve Amper Denklemi (2.13) kartezyen koordinat sisteminde ifade edilirse (2.14-19) denklemleri elde edilir.

$$\frac{\partial H_x}{\partial t} = \frac{1}{\mu_x}\left(\frac{\partial E_y}{\partial z} - \frac{\partial E_z}{\partial y} - \sigma_{M_x} H_x\right) \tag{2.14}$$

$$\frac{\partial H_y}{\partial t} = \frac{1}{\mu_y}\left(\frac{\partial E_z}{\partial x} - \frac{\partial E_x}{\partial z} - \sigma_{M_y} H_y\right) \tag{2.15}$$

$$\frac{\partial H_z}{\partial t} = \frac{1}{\mu_z}\left(\frac{\partial E_x}{\partial x} - \frac{\partial E_y}{\partial z} - \sigma_{M_z} H_z\right) \tag{2.16}$$

$$\frac{\partial E_x}{\partial t} = \frac{1}{\varepsilon_x}\left(\frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} - \sigma_x E_x\right) \tag{2.17}$$

$$\frac{\partial E_y}{\partial t} = \frac{1}{\varepsilon_y}\left(\frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} - \sigma_y E_y\right) \tag{2.18}$$

$$\frac{\partial E_z}{\partial t} = \frac{1}{\varepsilon_z}\left(\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} - \sigma_z E_z\right) \tag{2.19}$$

YEE Uzayı koordinat sistemi ve "n" zaman adımlarını eklediğimizde denklem (2.20-25) ifadelerini elde ederiz.

Faraday Yasası Maxwell Düzeltilmiş Zaman ve Uzay uygulanmış hali;

$$H_x^{n+\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right) = H_x^{n-\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right) + \frac{\Delta t}{\mu(i,j+\frac{1}{2},k+\frac{1}{2})}$$

$$.\left[E_y^n\left(i,j+\frac{1}{2},k+1\right) - E_y^n\left(i,j+\frac{1}{2},k\right) + E_z^n\left(i,j,k+\frac{1}{2}\right) - E_z^n\left(i,j+1,k+\frac{1}{2}\right)\right] \tag{2.20}$$

$$H_y^{n+\frac{1}{2}}\left(i+\frac{1}{2},j,k+\frac{1}{2}\right) = H_y^{n-\frac{1}{2}}\left(i+\frac{1}{2},j,k+\frac{1}{2}\right) + \frac{\Delta t}{\mu(i+\frac{1}{2},j,k+\frac{1}{2})}$$

$$.\left[E_z^n\left(i+1,j,k+\frac{1}{2}\right) - E_z^n\left(i,j,k+\frac{1}{2}\right) + E_x^n\left(i+\frac{1}{2},j,k\right) - E_x^n\left(i+\frac{1}{2},j,k+1\right)\right] \tag{2.21}$$

$$H_z^{n+\frac{1}{2}}\left(i+\frac{1}{2},j+\frac{1}{2},k\right)=H_z^{n-\frac{1}{2}}\left(i+\frac{1}{2},j+\frac{1}{2},k\right)+\frac{\Delta t}{\mu(i+\frac{1}{2},j+\frac{1}{2},k)}$$

$$.\left[E_x^n\left(i+\frac{1}{2},j+\frac{1}{2},k\right)-E_x^n\left(i+\frac{1}{2},j,k\right)+E_y^n\left(i,j+\frac{1}{2},k\right)-E_y^n\left(i+1,j+\frac{1}{2},k\right)\right] \quad (2.22)$$

Maxwell'in Düzeltmiş Olduğu Amper Yasasına Zaman ve Konum uygulanmış hali [9];

$$E_x^{n+1}(i+\frac{1}{2},j,k)=\left[1-\frac{\sigma(i+{}^1\!/_2,j,k)\partial t}{\varepsilon(i+\frac{1}{2},j,k)}\right]E_x^n(i+\frac{1}{2},j,k)+\frac{\Delta t}{\varepsilon\left(i+\frac{1}{2},j,k\right)}$$

$$.\left[H_z^{n+\frac{1}{2}}\left(i+\frac{1}{2},j+\frac{1}{2},k\right)-H_z^{n+\frac{1}{2}}\left(i+\frac{1}{2},j-\frac{1}{2},k\right)+H_y^{n+\frac{1}{2}}\left(i+\frac{1}{2},j,k-\frac{1}{2}\right)-H_y^{n+1/2}\left(i+\frac{1}{2},j,k+\frac{1}{2}\right)\right] \quad (2.23)$$

$$E_y^{n+1}(i,j+\frac{1}{2},k)=\left[1-\frac{\sigma(i,j+\frac{1}{2},k)\partial t}{\varepsilon(i,j+\frac{1}{2},k)}\right]E_y^n(i,j+\frac{1}{2},k)+\frac{\Delta t}{\varepsilon\left(i,j+\frac{1}{2},k\right)}$$

$$.\left[H_x^{n+\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right)-H_x^{n+\frac{1}{2}}\left(i,j+\frac{1}{2},k-\frac{1}{2}\right)+H_z^{n+\frac{1}{2}}\left(i-\frac{1}{2},j+\frac{1}{2},k\right)-H_z^{n+1/2}\left(i+\frac{1}{2},j+\frac{1}{2},k\right)\right] \quad (2.24)$$

$$E_z^{n+1}(i,j,k+\frac{1}{2})=\left[1-\frac{\sigma(i,j,k+\frac{1}{2})\partial t}{\varepsilon(i,j,k+\frac{1}{2})}\right]E_z^n(i,j,k+\frac{1}{2})+\frac{\Delta t}{\varepsilon\left(i,j,k+\frac{1}{2}\right)}$$

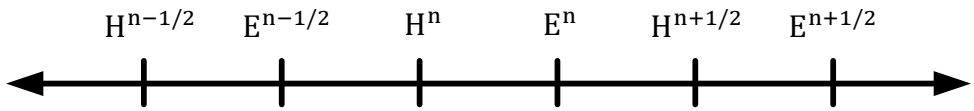
$$.\left[H_y^{n+\frac{1}{2}}\left(i+\frac{1}{2},j,k+\frac{1}{2}\right)-H_y^{n+\frac{1}{2}}\left(i-\frac{1}{2},j,k+\frac{1}{2}\right)+H_x^{n+\frac{1}{2}}\left(i,j-\frac{1}{2},k+\frac{1}{2}\right)-H_x^{n+1/2}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right)\right] \quad (2.25)$$

## 2.4 Kararlılık Koşulları

Doğru sonuçlar elde etmek için "Courant" tutarlılık koşuluna [10] uygun olarak hesaplama zaman aralıklarına dikkat edilmesi gerekir. "v" ışık hızı olup sabittir. Dolayısıyla verilen "$\Delta t$" değeriyle dalganın birim zamanda katedeceği yol bir hücrelik mesafeyi aşmamalıdır. Buradan hareketle kararlılık koşulu üç boyutlu kartezyen koordinat sistemi için denklem (2.26)'daki gibi tanımlanır. "v" sabit ve ışık hızıdır.

$$v \cdot \Delta t \leq \sqrt{\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2} + \frac{1}{(\Delta z)^2}} \qquad (2.26)$$

Daha doğru sonuçlar elde edebilmek için elektrik ve manyetik alanlar "leapfrog" yöntemiyler hesaplanır. "Leapfrog" yönteminde hesaplamalar "Δt/2" zaman aralıklarında sırayla tetiklenir. Zaman çizelgesi (Şekil 2.2)'de gösterilmiştir.




Şekil 2.2 "Leapfrog" Yöntemi Zaman Çizelgesi

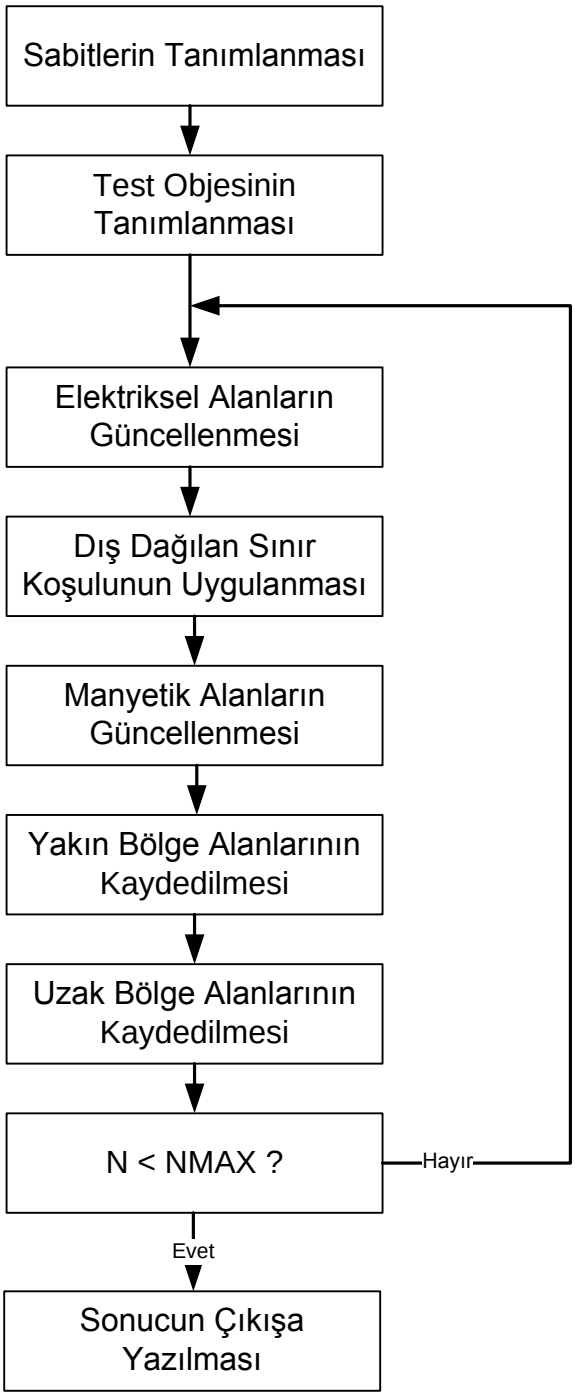
"Nyquist" örnekleme limitine [11] göre en ideal dalga boyu "$\lambda = 2\Delta x$" olmalıdır. En doğru sonuçlar yaklaşık olarak dalgaboyu başına dört hücre geldiğinde elde edilmiştir.

## 2.5 Program Akışı

Bir önceki bölümde tanıtılan model ve denklemlerin programlanmasında aşağıdaki adımlar uygulanmıştır. Bu adımlar aynı zamanda bir işaret akış diyagramı olarak (Şekil 2.3)'de gösterilmiştir.

- Ana program
  - N index'ine kadar olan zaman adımlarını ilerletir
- Problem uzayının tanımlanması
  - Problem uzayı büyüklüğünün tanımlanması

        - Her bir boyut için tanımlanacak hücre sayısı belirtilir
        - Hücre büyüklüğünün tanımlanması (Δx, Δy, Δz)
    - Δt'nin hesaplanması
    - Sabit çarpanların hesaplanması
- Test objesinin tanımlanması
    - IDONE(I,J,K) dizisi ile test objesi tanımlanır. Verilen "I,J, K" koordinatlarındaki tam sayı değer ile objenin tam iletken, kayıplı dielektrik, boşluk ya da herhangi bir karışık materyal olmasına göre test objesi tanımlanmış olur
- Elektrik ve Manyetik alan algoritmalarının uygulanması
    - Verilen koordinata ve materyal özelliğine (boşluk, kayıplı dielektrik, tam iletken) göre bir önceki zamanda kendisinin ve en yakın komşusunun alan niceliğine göre E ve M alanları hesaplanır
- Dış radyasyon sınır koşulunun uygulanması
- Verilerin kaydedilmesi
- E ve M alan verilerini, akımları ve diğer nicelikleri FDTD hesaplama uzayındaki dizilere kaydeder
- Uzak alan dönüşümü
    - Kapalı yüzeyde objenin etrafındaki teğetsel E ve M akımlarını değerlendirir ve uzak bölgedeki ilgili saçılan ya da dağılan alanları hesaplar

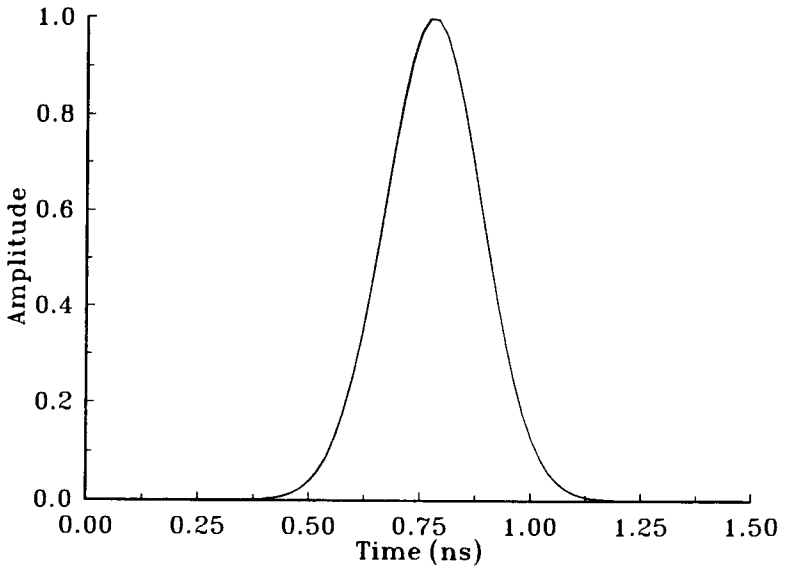



Şekil 2.3 Program Akışı

## 2.6 Gaussian Atımı

Hücre büyüklüğü kullanılan frekans büyüklüğüne göre karar verilir. Yüksek frekanslarda doğru sonuçlar elde edebilmek için mümkün olduğunca küçük hücrelerin kullanılması gerekmektedir. Maksimum zaman adımı değeri, courant tutarlılık koşuluna göre karar verilir. Deney sırasında Gaussian dalga formu (Şekil 2.4) kullanılmıştır. Gaussian dalga formu düzgün bir "cosine" dalga

formuna sahip olduğundan dolayı programı sağlıklı çalışması açısından önemlidir. FDTD geniş dalga boylarında çalışabilmektedir. Hücrenin boyutlarına karar verirken dalga boyunun kaç hücre büyüklüğünde olacağı önem arz etmektedir. Dalga boyu ne kadar az hücre ile ifade edilirse o kadar doğru sonuçlar elde edilecektir. Fakat hiçbir zaman bir hücreden az olmamalıdır.




Şekil 2.4 Gaussian Atımı (Courant tutarlılık limitinde bir santimetre kenarlı kübik küre hücresinde ve bir zaman atımı için kullanılan gaussian atımı grafiği $\boldsymbol{\beta = 32}$ alınmıştır)

FDTD Programı elektriksel ve manyetik alan update algoritmaları içerisindeki fonksiyonlar ile dalganın hedef alana etkisini dahil etmektedir.

## 2.7 Materyal Değişkeni

Milyonlarca hücrenin bulunduğu bir uzaya göre az sayıda farklılık gösteren materyal tipleri için tüm uzaya uygulamak çok fazla hafıza tüketmesine neden olabilir. Bundan dolayı farklı tip materyalleri tanımlamak için IDONE-IDSIX dizilerini kullanılır. Materyal dizileri program içersinde "I, J, K" ile belirtilen Yee hücresinin köşegen, kenar ortalarında ve yüzey merkezlerinde bulunan elektriksel veya manyetiksel alanların "x, y, z" yönlerine bağlı olarak ve herbiri için materyal özelliğini saklamaktadır. Programda tanımlı ortam değişkeni tamsayı değerleri aşağıdaki listede verilmiştir. Materyal değişkeni dizisi homojen

özellikler göstermeyen nesnelerin elektromanyetik dalga ile etkileşimini noktasal olarak tüm hücreler için yaptığımız elektrik ve manyetik alan hesaplamalarında kullanılır. Sonuç olarak, FDTD yöntemini kullanarak toprak veya insan beyni gibi homojen olmayan malzemelerin elektromanyetik dalga etkileşimlerini inceleme fırsatı elde edilmiş olur.

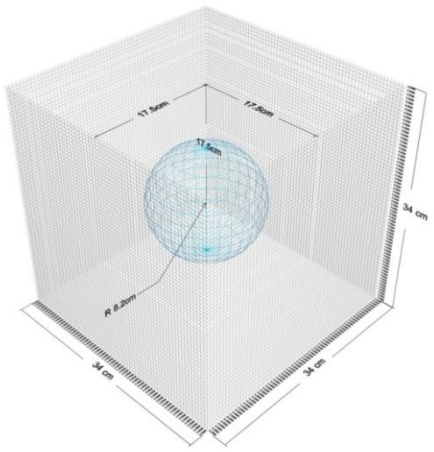
| | |
|---|---|
| IDONE(I,J,K) = 0 | Boşluk |
| IDONE(I,J,K) = 1 | Tam iletken |
| IDONE(I,J,K) = 2 | Kayıplı Yalıtkan |

Şekil 2.5 Ortam Materyal Tipleri

FDTD programı içerisinde IDONE(I,J,K), IDTWO(I,J,K) ve IDTHRE(I,J,K) dizileri EXS, EYS ve EZS hesaplamaları için kullanılmaktadır. Ortama bağlı HXS, HYS ve HZS değerlerini hesaplamak için ise IDFOUR(I,J,K), IDFIVE(I,J,K) VE IDSIX(I,J,K) dizilerini kullanılmaktadır.

## 2.8 YEE Hücrelerini Kullanarak Obje Yaratılması

FDTD program içerisinde merkez koordinatları (17,5-17,5-17,5) olan ve yarıçapı 8,2 olan bir küre (Şekil 2.6) benzer şekilde yaratılmaktadır.




Şekil 2.6 Problem Uzayındaki Lossy Dielektrik Nesne

NX, NY ve NZ uzay sınırlarını, R kürenin yarıçapını, SC (x,y,z) koordinatları üzerinde olmak üzere kürenin merkezini, MTYPE materyal tipini göstermektedir. Aşağıda gösterildiği gibi döngü koşulunu sağlayan her noktanın

materyal tipi dizi üzerinde kayıplı dielektrik olarak kaydedilir. Materyal dizisi oluşturulurken tüm değerleri sıfır olarak kaydedildiğinden dolayı geri kalan noktalar için materyal tipi sıfır-boşluk ortamı olarak kalmaya devam edeceklerdir.

```
for I=1:NX,
 for J=1:NY,
  for K=1:NZ,
       R=sqrt((I-SC)^2+(J-SC)^2+(K-SC)^2);
       if (R <= RA),
        IDONE(I,J,K)=MTYPE;
        IDONE(I,J,K+1)=MTYPE;
        IDONE(I,J+1,K+1)=MTYPE;
        IDONE(I,J+1,K)=MTYPE;
        IDTWO(I,J,K)=MTYPE;
        IDTWO(I+1,J,K)=MTYPE;
        IDTWO(I+1,J,K+1)=MTYPE;
        IDTWO(I,J,K+1)=MTYPE;
        IDTHRE(I,J,K)=MTYPE;
        IDTHRE(I+1,J,K)=MTYPE;
        IDTHRE(I+1,J+1,K)=MTYPE;
        IDTHRE(I,J+1,K)=MTYPE;
       end
  end
 end
end
```

## 2.9 Boşluk Ortamında Elektriksel Alanların Güncellenmesi

Tam iletken ortamdaki elektriksel alanları zaman bağımlı olarak program içerisinde denklem (2.29)'da gösterildiği şekliyle hesaplanır.

$$EXS(I,J,K) = EXS(I,J,K) + {\Delta t}/{\varepsilon_0}\left[{\text{HZS(I,J,K)-HZS(I,J-1,K)}}/{\Delta Y} - \left({\text{(HYS(I,J,K)-HYS(I,J,K-1)}}/{\Delta Z}\right)\right] \quad (2.29)$$

Ortam boşluk olduğundan dolayı $\Delta t/\varepsilon_0$ ve $\Delta t/\mu_0$'ın etkisi yoktur. Program içerisinde denklem (2.30)'da gösterildiği gibi kullanılır.

```
EXS(I,J,K)=EXS(I,J,K)+(HZS(I,J,K)-HZS(I,J-1,K))*DTEDY-
(HYS(I,J,K)-HYS(I,J,K-1))*DTEDZ;                    (2.30)
```

## 2.10 Boşluk Ortamında Manyetik Alanların Güncellenmesi

Materyal tipi kayıplı dielektrik olan kürede herhangi bir manyetik materyal içermediğinden dolayı sadece boşluk içerisindeki manyetik alanları güncelliyoruz. Tam iletken ve kayıplı dielektrik ortamlarında manyetik alan hesaplamaları yapılmamaktadır. Bundan dolayı bu koordinatlardaki manyetik alan değerleri sıfır olarak kalacaktır. Boş ortam için denklemimiz denklem (2.31)'de gösterilmiştir.

$$\mathrm{HYS(I,J,K)} = \mathrm{HYS(I,J,K)} + {}^{\Delta t}/_{\mu_0}\left[\left({}^{\mathrm{EZS(I+1,J,K)-EZS(I,J-1,K)}}/_{\Delta X}\right) - \left({}^{\mathrm{(EXS(I,J,K+1)-EXS(I,J,K)}}/_{\Delta Z}\right)\right] \tag{2.31}$$

$H_y$denklem (2.32)'de hesaplanmaktadır.

HYS(I,J,K)=HYS(I,J,K)-(EXS(I,J,K+1)-EXS(I,J,K))*DTMDZ...

+(EZS(I+1,J,K)-EZS(I,J,K))*DTMDX (2.32)

DTMDX ve DTMDZ sabit olup program başlangıcında hesaplaması yapılmıştır. Örnek olarak, DTMDZ "DT/(XMU0*DELZ)" ifadesiyle hesaplanmaktadır. Aynı ifadenin DTMDX için hesaplanırken DELZ yerine DELX ifadesi gelmektedir.

## 2.11 Tam İletken Ortamda Elektriksel Alanların Güncellenmesi

Boşluk için "**σ**" sonsuz olduğundan dolayı $\mathrm{E^{scat}} = -\mathrm{E^{inc}}$ olacaktır. Program içerisindeki gösterimi ise EXS(I,J,K)=-EXI(I,J,K) şeklindedir.

## 2.12 Kayıplı Dielektrik Ortamda Elektriksel Alanların Güncellenmesi

Kayıplı dielektrik ortam için $\varepsilon \neq \varepsilon_0$ olduğu varsayılır.

$$E_x^s(I,J,K)^n = E_x^s(I,J,K)^{n-1}\left[\frac{\varepsilon}{\varepsilon+\sigma\Delta t}\right] - \left[\frac{\sigma\Delta t}{\varepsilon+\sigma\Delta t}\right] E_x^i(I,J,K)^n - \left[\frac{(\varepsilon-\varepsilon_0)\Delta t}{\varepsilon+\sigma\Delta t}\right] E_x^i$$

$$+\frac{H_z^s(I,J,K)^{n-\frac{1}{2}} - H_z^s(I,J-1,K)^{n-\frac{1}{2}}}{\Delta Y}\left[\frac{\Delta t}{\varepsilon+\sigma\Delta t}\right] + \frac{H_y^s(I,J,K)^{n-\frac{1}{2}} - H_y^s(I,J,K-1)^{n-\frac{1}{2}}}{\Delta Z}\left[\frac{\Delta t}{\varepsilon+\sigma\Delta t}\right] \quad (2.33)$$

Kayıp dielektrik ortam elektriksel ve manyetik alan hesaplamalarında denklem (2.33) için sabit çarpanlardan ECRLY "$\Delta t/(\varepsilon + \sigma\Delta t)\Delta y$" denklem (2.34)'de program içerisindeki karşılığı gösterilmiştir.

$$ECRLY(M) = \frac{DT}{(EPS(M)+SIGMA(M)*DT)*DY} \quad (2.34)$$

Kayıplı yalıtkan ortam içerisindeki $E_x^s$ hesaplamak için denklem (2.35) kullanılır.

$$EXS(I,J,K) = EXS(I,J,K) * ESCTC(IDONE(I,J,K)) - EINCC(IDONE(I,J,K)) * EXI(I,J,K) - EDEVCN(IDONE(I,J,K)) * DEXI(I,J,K) + (HZS(I,J,K) - HZS(I,J-1,K)) * ECRLY(IDONE(I,J,K)) - (HYS(I,J,K) - HYS(I,J,K-1)) * ECRLZ(IDONE(I,J,K)) \quad (2.35)$$

## 2.13 Radyasyon Sınır Koşulları

Hesaplama uzayı radyasyon sınır koşulları ile sınırlandırılmıştır. Sınır koşuluna ulaşan dalga dağılacak, soğurulacak veya geri yansıyacaktır. Bu olayın etkilerinin hesaplamalara dahil edilmesi gerekmektedir. Sınır koşullarının gelen dalgaya etkisini hesaplamak için Mur Absorbing Boundary yöntemi kullanılır [12]. Mur Absorbing Boundary iki seviyeden oluşmaktadır. Birinci seviyede, zamanda

bir adım geri giderken lokasyonda da bir adım önceki hücrenin elektrisel alanını hesaplanır. İkinci seviye zamanda ise zamanda iki adım öncesin de lokasyonda iki hücre öncesinin elektriksel alanı hesaplanmaktadır. Sınır koşulu x=0 olsun, y=jΔy ve z=(k+1/2)Δz lokasyonu için $E_z$ 'nin birinci seviye hesaplaması denklem (2.36)'de gösterilmiştir. İkinci seviye sınır koşulunun x=0'da uygulanması denklem (2.37)'de gösterilmiştir.

$$E_z^{n+1}\left(0,j,k+\frac{1}{2}\right) = E_z^{n}\left(1,j,k+\frac{1}{2}\right) + \frac{c\Delta t-\Delta x}{c\Delta t+\Delta x}\left(E_z^{n+1}\left(1,j,k+\frac{1}{2}\right) - E_z^{n}\left(0,j,k+\frac{1}{2}\right)\right) \tag{2.36}$$

$$\begin{aligned} E_z^{n+1}\left(0,j,k+\frac{1}{2}\right) &= -E_z^{n-1}\left(1,j,k+\frac{1}{2}\right) + \frac{c\Delta t-\Delta x}{c\Delta t+\Delta x}\left(E_z^{n+1}\left(1,j,k+\frac{1}{2}\right)\right. \\ &+ \left(E_z^{n-1}\left(0,j,k+\frac{1}{2}\right) + \frac{2\Delta x}{c\Delta t+\Delta x}E_z^{n}\left(0,j,k+\frac{1}{2}\right)\right. \\ &+ E_z^{n}\left(1,j,k+\frac{1}{2}\right) \\ &+ \frac{\Delta x(c\Delta t)^2}{2(\Delta y)^2(c\Delta t+\Delta x)}.\left(E_z^{n}\left(1,j+1,k+\frac{1}{2}\right)\right. \\ &- 2E_z^{n}\left(0,j,k+\frac{1}{2}\right) + E_z^{n}\left(1,j-1,k+\frac{1}{2}\right) \\ &+ E_z^{n}\left(1,j+1,k+\frac{1}{2}\right) - 2E_z^{n}\left(1,j+\frac{1}{2}\right) \\ &+ E_z^{n}\left(1,j-1,k+\frac{1}{2}\right) + \frac{\Delta x(c\Delta t)^2}{2(\Delta z)^2(c\Delta t+\Delta x)}.E_z^{n}\left(0,j,k+\frac{3}{2}\right) \\ &- 2E_z^{n}\left(0,j,k+\frac{1}{2}\right) + E_z^{n}\left(0,j,k-\frac{1}{2}\right) + E_z^{n}\left(1,j,k+\frac{3}{2}\right) \\ &- 2E_z^{n}\left(1,j,k+\frac{1}{2}\right) + E_z^{n}\left(1,j,k-\frac{1}{2}\right) \end{aligned} \tag{2.37}$$

Birinci seviye Mur soğurulması ZY yönündeki radyasyon için $E_x^s$ hesaplanması denklem (2.38)'de gösterilmiştir. CYD, CYY, CYFXD ve CYFZD program içersinde sabit olduklarından dolayı hesaplamaları daha önceden yapılmıştır.

```
EZS(I,1,K)=EZSY1(I,2,K)+CYD*(EZS(I,2,K)-EZSY1(I,1,K))      (2.38)
```

İkinci Seviye Mur Soğurulması ZY yönündeki radyasyon için $E_x^s$ hesaplanması denklem (2.39)'de gösterilmiştir.

```
EZS(I,1,K)=-EZSY2(I,2,K)+CYD*(EZS(I,2,K)+EZSY2(I,1,K))...
   +CYY*(EZSY1(I,1,K)+EZSY1(I,2,K))...
   +CYFXD*(EZSY1(I+1,1,K)-2.*EZSY1(I,1,K)+EZSY1(I-1,1,K)...
   +EZSY1(I+1,2,K)-2.*EZSY1(I,2,K)+EZSY1(I-1,2,K))...
   +CYFZD*(EZSY1(I,1,K+1)-2.*EZSY1(I,1,K)+EZSY1(I,1,K-1)...
   +EZSY1(I,2,K+1)-2.*EZSY1(I,2,K)+EZSY1(I,2,K-1))          (2.39)
```

# 3 PROGRAMLAMA ARAÇLARI

3GL, Üçüncü Jenerasyon Diller, kullanımı daha zor olan düşük seviyeli assembler benzeri dillerin üzerine bir katman gibi eklendiklerinden dolayı, çoğu zaman yüksek seviyeli diller olarak adlandırılır. 3G diller, programları oldukça hızlı bir şekilde makine koduna çevirebilmektedirler. 3G etkili bir şekilde kullanabilmek için iyi bir programlama bilgisi ve deneyimine ihtiyaç vardır.

4GL Dördüncü Jenerasyon Diller daha az prosedür içermektedirler. Kod yazımları insan diline yakındır. Kullanıcı için kod yazımı ve okunması daha kolay anlaşılır. 4G dillerin çalıştırılmadan önce yorumlanmaya ihtiyacı olduğundan dolayı 3G diller ile karşılaştırıldığında daha yavaş çalışmaktadırlar [13].

## 3.1 MATLAB Ortamı

MATLAB yüksel seviyeli bir script dilidir. Uygulama geliştiricisi programı çalıştırmak için derlemek zorunda değildir. Değişkenleri programa sıkı koşullar altında tanıtma zorunluluğu bulunmamaktadır. Program geliştirme esnasında "low-level" hafıza yönetimi yapılmasına gerek yoktur. Bu özellikleri ile MATLAB kısa zamanda uygulama geliştiricinin karmaşık problemlerin üstesinden gelebilmesini sağlar.

MATLAB sistemi iki ana parçadan oluşur. Masaüstü Araçları ve Geliştirme Ortamı: masaüstü ve komut penceresi, editör, "debugger", "code analyzer", "help browser", "workspace", dosyalar ve diğer araçları içerir. Mathematiksel Fonksiyon Kütüphanesi: "sin", "cosine" gibi temel fonksiyonlardan matris "eigenvalues", "bessel functions", "fast fourier transforms" gibi kompleks aritmetik hesaplamalara kadar çeşitli içerik bulunduran bir kütüphanedir [14].

MATLAB programları ve fonksiyonları editör aracılığıyla okunabilir durum ve komutlar içeren ".m" uzantılı M-dosyalarına yazılır. Komut satırına dosyanın ismi yazılarak çağrılır. Fonksiyon adı ile dosya adı aynı olmak zorundadır. MATLAB kullanıcıları matematiksel operasyon, algoritma geliştirme, programlama, 2D ve 3D grafik çizimi, modelleme, simülasyon, grafik arayüz

oluşturma, veri giriş çıkışı, matris işlemleri ve kullanıcı arayüzleri yaratabilirler [15].

MATLAB, matematiksel istatistik, optimizasyon, "neural network", "fuzzy", işaret ve görüntü işleme, kontrol tasarımları, yöneylem çalışmaları, tıbbi araştırmalar, finans ve uzay araştırmaları gibi çok çeşitli alanlarda kullanılmaktadır. MATLAB kullanıcıya hızlı bir analiz ve tasarım ortamı sağlar.

Tanımlı fonksiyonları kullanıcılara karmaşık sorunların çözülmesinde yardımcı olur. Bir kullanıcı standart "toolbox"'ları kullanarak sinyal işleme, resim işleme, kontrol üniteleri, paralel işlemler, komünikasyon ve nöron ağları ile ilgili problemleri "toolbox"'lardaki hazır fonksiyonları kullanarak çözebilir. MathWorks'ün sitesinde dünya üzerindeki MATLAB kullanıcılarının kendi problemleri için geliştirdikleri fonksiyon, program ve arayüzleri ücretsiz paylaşabilmektedir. Bu özelliği sayesinde her kullanıcı ihtiyacı olan modülleri internet üzerinden bulabilmekte ve kendi sistemine uyarlayabilmektedir.

SIMULINK eklentisiyle MATLAB görselden matematiğe geçiş yaklaşımını benimsemiştir. Mühendisler mantıksal süreçlerini sürükle bırak yöntemi ile oluşturabilmektedir. Oluşturulan sistem matematiksel olarak denenebilmektedir. Bu yöntem ile bir otobüsün süspansiyonunu, bir rüzgar türbinini veya dijital ses işlemcilerini dizayn etmek mümkündür.

MATLAB üzerinde çalıştığı platformun desteklediği herhangi bir görüntü arabirimiyle birlikte çalışabilmektedir. Kendine özel görüntü ara birimi istememektedir.

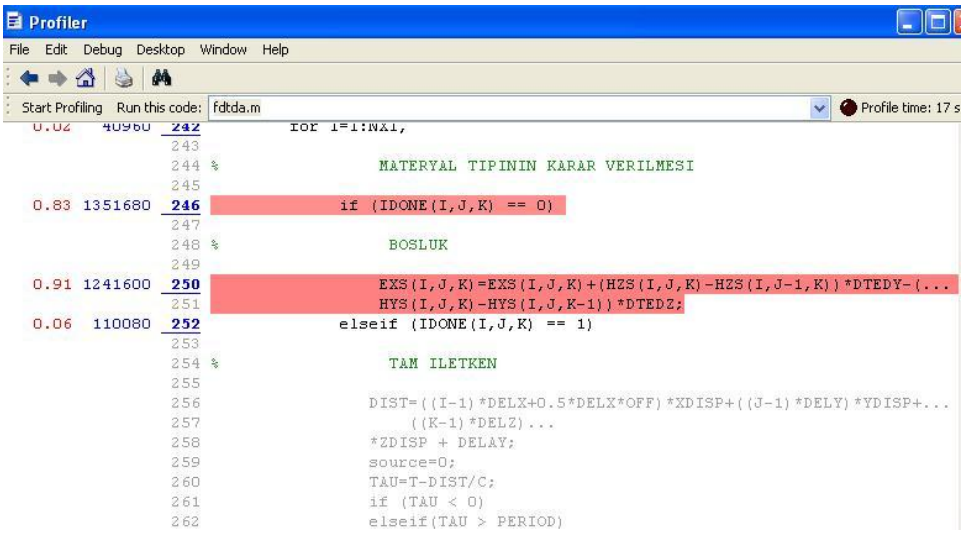
### 3.1.1 MATLAB Tarihçesi

MATLAB tescilli bir marka olup Matrix Laboratory'nin kısaltılmış halidir. C ve Pascal gibi yüksek seviyeli bir dildir. 1970'lerin sonunda Cleve Moler tarafından geliştirilmiştir. "The MathWorks" firması pazarlama ve destek tarafında faaliyet göstermektedir. Script dili FORTRAN 90'a benzemektedir.

### 3.1.2 MATLAB "Debugger"'ı ve "Profiler"'ı

MATLAB görsel "debugger"'ı program akışını geçici olarak durdurabilmektedir. Durdurulma anında ortam değişkenlerini gösterebilme, program adımlarını ileri veya geri hareket ettirebilme ve sadece ilgili satırları çalıştırma imkanı yaratabilmektedir.

MATLAB Profiler özelliği uygulama geliştiriciye program performansı hakkında bilgi verir. Profiler penceresi içerisinden program çağrılır. Program bittiğinde fonksiyon ve ifadelerin kaç kez çağırıldığı, işlem süreleri ve kullanılmayan ifade ve satırlar hakkında rapor üretir. Profiler kullanılmadan yapılan denemelerde programın dört kat daha hızlı çalıştığı görülmüştür.




Şekil 3.1 EXS Boşluk Alan Güncellemesi İle İlgili FOR Döngüsü

### 3.1.3 MATLAB "GUI"'si – Grafik Kullanıcı Arayüzü

Kullanıcılar MATLAB son kullanıcı arayüzlerini temelde beş nedenden dolayı tercih etmektedir.

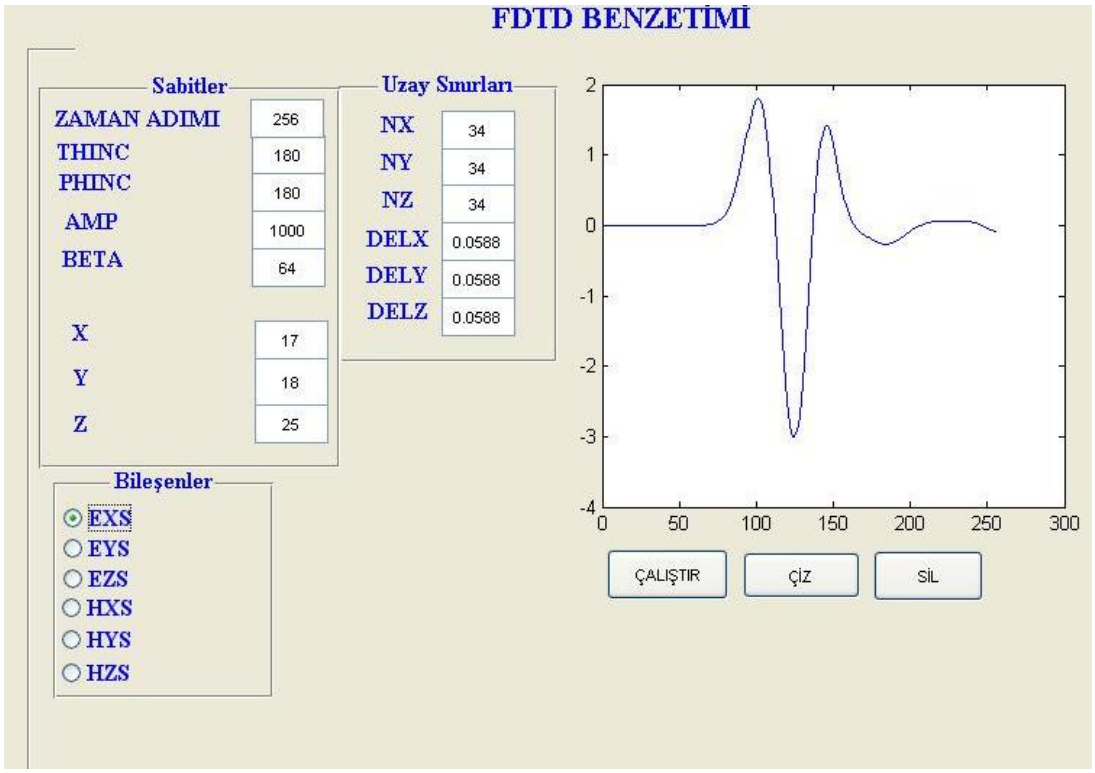
- Yüksek seviyeli skript temelli geliştirme yapılabilmesi
- MATLAB'in matematiksel hesaplama gücü ile uygulamaların uyumunun sağlanması
- İşletim sistemi bağımsız uygulamalar geliştirilebilmesi
- Son kullanıcı ile etkileşimli uygulamalar geliştirilebilmesi
- Gerçek zamanlı sonuçlar gösterilebilmesi

MATLAB "GUI"'lerinde "object-orient" nesne tabanlı yapılar kullanılmaktadır. MATLAB "event handling" olay başarımı yaklaşımına sahiptir. Uygulama geliştirici GUI üzerinde bir obje yarattığında bu işlem ilgili ".m" dosyasında "handle graphic object" otomatik olarak yaratılır. Son kullanıcının "GUI" üzerinde yaptığı işlemler "graphic handle" object tarafından yakalanır. Bu işlemler için ilgili "handle graphic object"'e bağlı "callback" fonksiyonundaki tanımlamalar gerçekleştirilir. Son kullanıcı "GUI" uygulamalarında anlık olarak görselin veya verinin üzerinde çalışabilir.

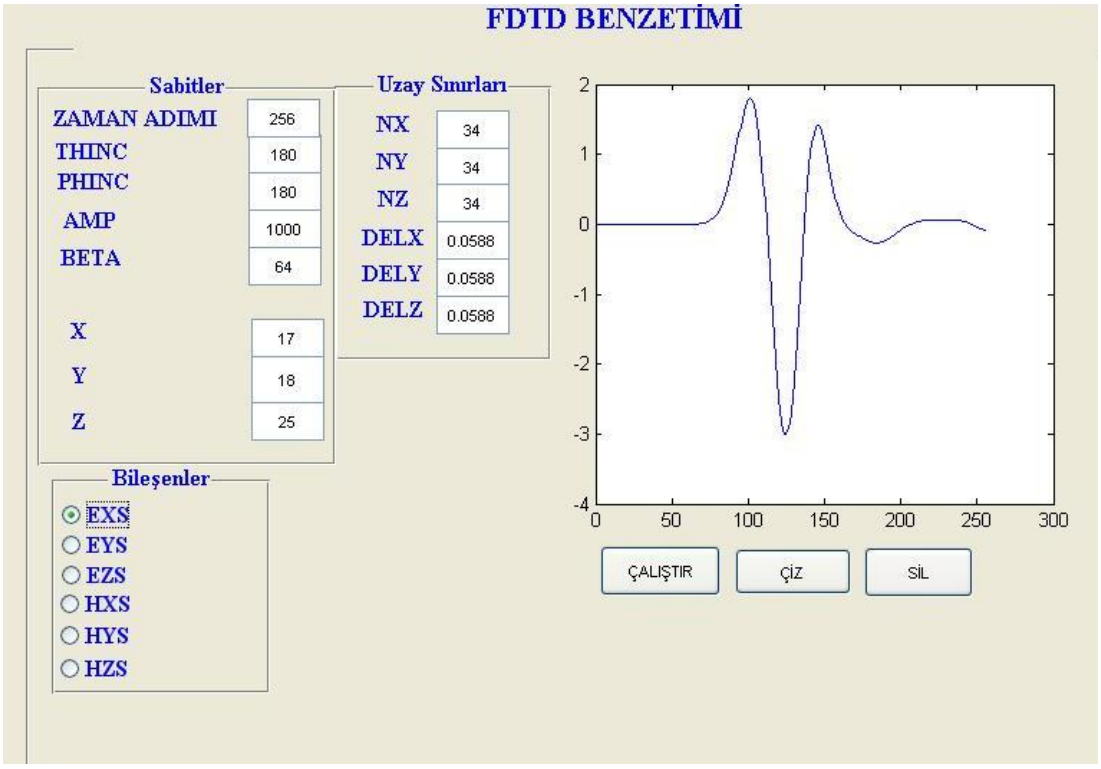
Kullanıcı "GUIDE" alt programı vasıtasıyla sürükle bırak yöntemini kullanarak *radio button, text box, slider, run button, panel, axes ve static text* gibi objeler ile arayüz tasarlayabilir. "GUIDE" ilgili objenin özelliklerini göstererek font, renk, görünürlük, saydamlak, katman ve aksiyon tanımlamaları yapılmasına imkan sağlar. İşlem sonuçlarını "GUI" içersinde tanımlanabilen "axes" objesi içerisinde tablo, grafik veya film olarak gösterilebilir.

"GUI" fonksiyonları lokal değişkenler ile çalışmak durumundadır. Bunun gerekçesi çoklu "GUI" yapılarından kaynaklanmaktadır. Uygulama geliştirici çoklu "GUI" geliştirirken sistematiğini kaybetmemesi için değişkenler fonksiyonlara özel tanımlanmaktadır. Dolayısıyla diğer fonksiyonlar tarafından kullanılan bir değişken olduğunda her bir fonksiyonun başında global değişken olarak değişkenin tanımlanma zorunluluğunu ortaya çıkmaktadır.

MATLAB FDTD uygulaması son kullanıcını rahatlıkla kullanabilmesi için geliştirilen "GUI" ekran görüntüsü (Şekil 3.2), (Şekil 3.3)'de gösterilmiştir. Bu uygulama ile son kullanıcı program içeriğini detaylı olarak inceleme zorunluluğu hissetmemektedir. Uygulama son kullanıcının en çok ihtiyaç duyacağı değişkenler, uzay sınırları ve hedef sonuç grafik seçimlerini son kullanıcının belirleyebileceği şekilde sunmaktadır. Hedef sonuç grafiği arka planda çalışan FDTD programı sayısal sonuçlarını elde edildikten sonra grafik biçiminde ekranın sağ tarafında göstermektedir. Kullanıcı çalıştır butonuyla programın arka planda hesaplamaları başlatmasını sağlar. Çiz butonu ile elde edilen verilerin grafik ekranda gösterilmesi sağlanmaktadır. Sil butonuyla grafik ekranda üretilen değerler temizlenir. Eğer sil butonu çalıştırılmadan yeni değerler hesaplanır ve çizdirilirse çizimler aynı anda gösterilmiş olacaktır.




Şekil 3.2 FDTD EXS Sonuç Grafiği




Şekil 3.3 FDTD EZS Sonuç Grafiği

### 3.1.4 MATLAB "MEX" Desteği

"MEX" (MATLAB Executable) altyapısı ile diğer yazılım dillerinin özel kütüphanelerini kullanabilmek ve "C, FORTRAN, JAVA" ile derlenen programların daha yüksek performansına sahip olabilmek için kullanılmaktadır. "C, JAVA, FORTRAN" dilleri MATLAB kadar veriler için geniş hafıza kullanmamaktadır. Özellikle döngülerde her bir döngü sırasında diziye tekrar yorumlamadığından dolayı daha hızlı çalışmaktadır. Fonksiyonların MATLAB tarafından kullanılabilmesi için MATLAB'ın desteklediği derleyicilerin kullanılması gerekmektedir.

"MEX"'i kullanabilmek için ana fonksiyonun ismi "mexfunction" olarak isimlendirilir. "Mexfunction" dört adet giriş argümanı durmaktadır. Herhangi bir dönüş argümanı yoktur. Giriş argümanları;

1. Çıkış parametreleri sayısı (nlhs parametresi)
2. Çıkış parametreleri için bir pointer bilgisi (plhs parametresi)
3. Giriş parametreleri sayısı (nrhs parametresi)
4. Giriş parametreleri için bir pointer bilgisi (prhs parametresi)

C dilinde yazılan fonksiyonlarda çağrılar MATLAB'e özel isimleri "mx" veye "mex" olan rutinler ile başlamak zorundadır. Microsoft ortamında C programlarının derlenebilmesine yönelik MATLAB'ın geliştirdiği "lcc" derleyicisi MATLAB kurulumu ile birlikte sisteme dahil olmaktadır [16].

```
MEX fonksiyonları:
        C/C++ Fonksiyon Girişi:
        "mexfunction"(int nlhs, mxArray *plhs[ ],
        int nrhs, const mxArray *prhs[ ]) { . }

        FORTRAN Girişi:
        SUBROUTINE "MEXFUNCTION"( NLHS, PLHS, NRHS, PRHS)
```

### 3.1.5 MATLAB Component Run-Time Ortamı

MATLAB "Component Run-Time" Ortamı MATLAB kurulu olmayan bir windows platformunda çalıştırılabilir hale getirilmiş MATLAB programlarını kullanabilmeyi amaç edinmektedir. Çalışma düzeni "Java Run Time" ortamına benzemektedir. İlk defa kullanılacak ise "mbuild –setup" komutu ile sistemde bulunan derleyicilerin tanıtılması ve opsiyonel özelliklerinin MATLAB'a bildirilmesi gerekmektedir. Bunu yapabilmek için MATLAB ".m" dosyasının "MCC" alt programı ile derlenmesi gerekmektedir [17]. Derlenme sonucunda iki adet dosya yaratılır. Birinci dosya çalıştırılabilir dosyadır, ikinci dosya ise "CTF Component Technology File" dosyasıdır. "CTF" içerisinde sıkıştırılmış ve şifrelenmiş ".m" dosyaları ile ".m" dosyasının çalıştırılabilmesi için gerekli olan diğer dosyalar bulunur. "CTF" şifreleme algoritması olarak "AES" kullanır. Şifrelenmiş dosyalar simetrik iki adet bin yirmi dört bit "RSA" anahtarı ile korunmaktadır. MATLAB programı "MCR" yüklü bir makinede ilk defa çalıştırıldığında "CTF" dosyası yeni bir klasör içerisine açılır. Böylelikle ".m" dosyası çalışmak için ihtiyaç duyduğu tüm dosyaları bu klasörden çağırarak kullanır.

"MCR" kullanmanın avantajlarını şu şekilde sıralayabiliriz;

1. "GUI" uygulamaları için idealdir
2. Her son kullanıcı için MATLAB kurulmasına gerek yoktur. Lisans maliyet avantajı sağlar
3. Son kullanıcının MATLAB bilgisine sahip olması gerekmemektedir
4. Son kullanıcı uygulamanın gizli kalması gereken kod detaylarına erişemez

MATLAB Component Runtime çalışma ortamının bazı kısıtlamaları bulunmaktadır. Uygulama geliştiricinin yazmış olduğu ".m" dosyasının mutlaka bir fonksiyon olması gerekmektedir. "Signal Processing Toolbox" fonksiyonları gibi gömülü fonksiyonları desteklememektedir. Uygulamanın çalışabilmesi için hedef sistemde mutlaka MCR yüklü olması gerekmektedir.

### 3.1.6 MATLAB TZ (Tam Zamanlı) Hızlandırıcı

MATLAB 6.5 versiyonundan önce özellikle döngü kullanımında uygulamalarda ciddi yavaşlık gözlenmekteydi. MATLAB programı çalıştırabilmek için programı ilk olarak kaba koda çevirmesi gerekmektedir. Kaba kodundaki her talimat "interpreter" tarafından yorumlanır. Daha sonra her kaba kodu satırı sırasıyla "interpreter" tarafından çalıştırılır. MATLAB 6.5 ile birlikte MATLAB yapısal olarak uygulama geliştiricinin vermiş olduğu döngü ve değişkenleri performans arttırıcı şekilde yorumlama yeteneğine kavuşmuştur. TZ program içerisinde ilk defa karşılaştığı ve program boyunca yapısal değişiklik göstermeyen değişken ve nesneler için bir kez yorumlama yapması yeterli olmaktadır. Derleyici, değişken veya nesne değiştiğinde programı tekrar yorumlaması gerekmektedir. Sabit nesne ve değişkenleri her seferinde tekrar derleme ve hafıza da yer ayırmak zorunda kalmayacağından dolayı program hızlanmış olacaktır.

MATLAB TZ, programcıların döngü işlemlerini vektörize yani matris operasyonları haline getirme zorunluluğunu ortadan kaldırmıştır. Vektörize etme işlemi TZ'den önce uygulama geliştiricisini performans anlamında negatif etkileyen bir unsurdu. FDTD programında olduğu gibi daha öncesinde yazıldığı dilin "FORTRAN" döngü yetenekleriyle rahatlıkla yapılan işlemler MATLAB ortamında ciddi yavaşlatıcı etkilere sahip olmaktaydı. Aşağıdaki örnek döngünün TZ kullanılmadan nasıl vektörize edildiği gösterilmiştir.

$a_n = n, b_n = 1000 - n$ ve n=1,...,1000 için $ssum = \sum_{n=1}^{1000}(a_n b_n)$ hesaplayalım.

Döngü kullanıldığında;

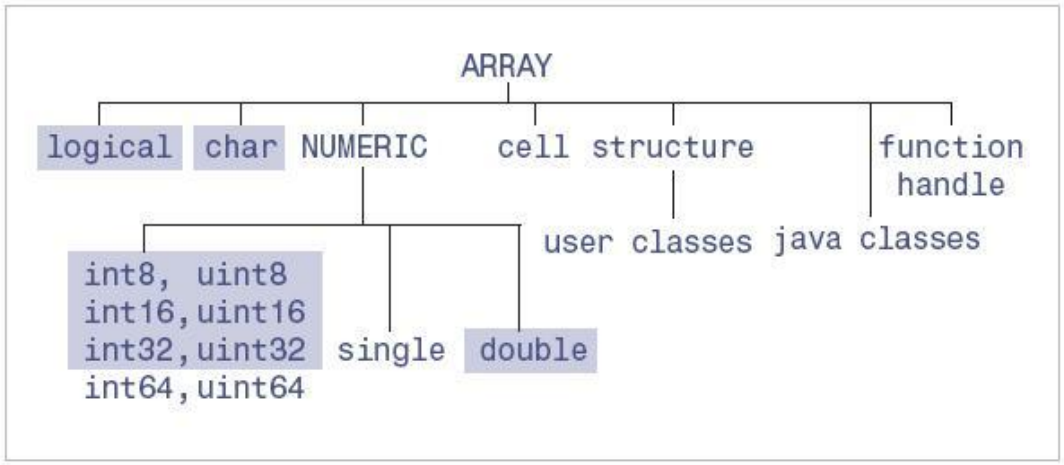
```
a = 1:1000;
b = 1000 - a;
ssum=0;
        for n=1:1000
                ssum = ssum +a(n)*b(n);
        end
```

Vektörize edilmiş haliyle;

```
Ssum=a*b;
```

MATLAB FDTD programında “I, J, K” indekslerine sahip YEE uzayında her bir “I”, “J” ve “K” noktası için ayrı matematiksel operasyonlar yapıldığından dolayı dizi operasyonları yapılamamıştır. Çünkü işlem sırasında dizi elemanları sürekli olarak değişmektedir. Örnek olarak; A dizisinin birinci elemanı B dizisinin birinci elemanı ile çalışırken, A dizisinin ikinci elemanı B dizinin beklenen ikinci elemanı yerine üçüncü elemanı ile çalışması gerekebilmektedir. Bu operasyonları yapabilmek için dizilerin elemanlarına özel kurallar tanımlanabilmelidir veyahut dizilerin farklı indeks yapısına sahip kopyalarını oluşturup bu kopyalarda işlemler yapıldıktan sonra tekrar asıl diziye bu işlem sonuçlarının aktarılması gerekecektir. Bu operasyonu oldukça karmaşık hale getireceğinden dolayı büyük uygulamarda karmaşıklıklara ve hatalara yol açabilir.

MATLAB TZ özelliğinin kısıtlamaları bulunmaktadır. Bunlar dizilerin üç boyuttan daha büyük olamaması, döngülerinin “scalar” işlemler yapma zorunluluğu, (Şekil 3.4)’de gösterilen veri tiplerinin kullanılıyor olması, program tarafından diğer M-File veya “MEX” fonksiyonlarının çağrılmaması, platformun “Intel X86” işlemciye sahip olması ve yoğun vektör işlemlerinin bulunmaması gerekmektedir.



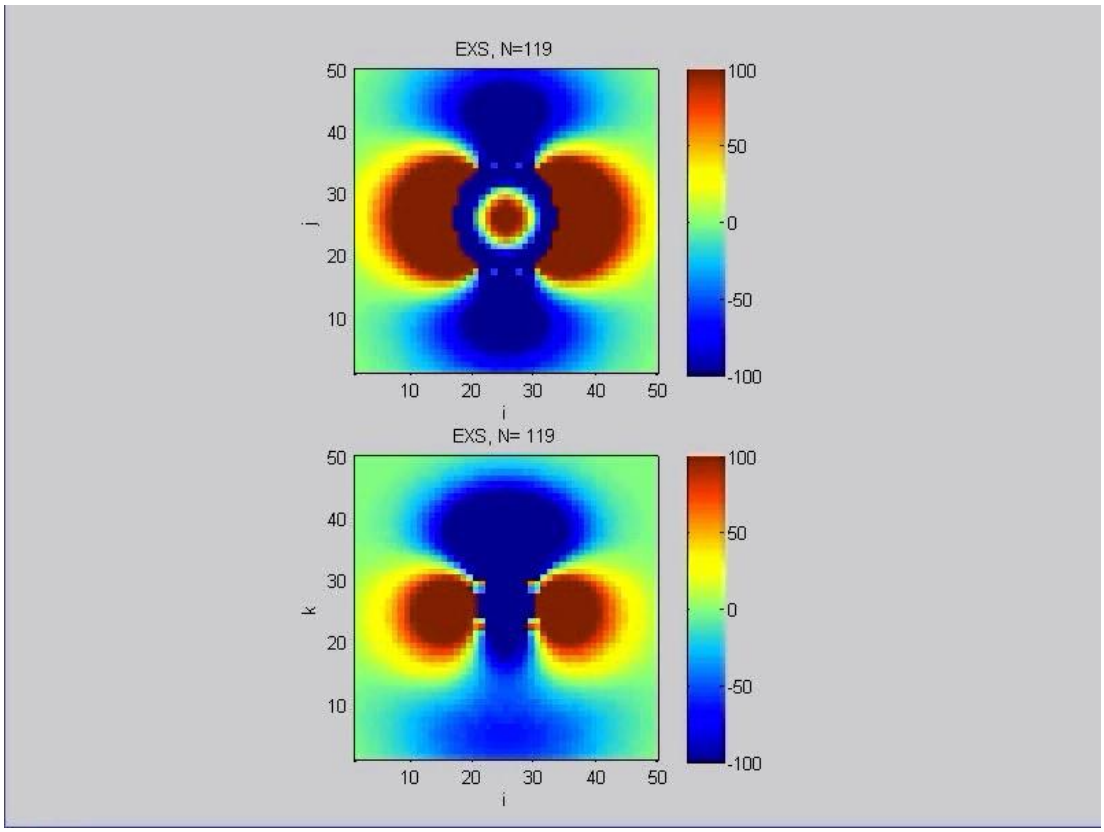

Şekil 3.4 MATLAB TZ’nin Tarafından Desteklenen Veri Tipleri (Gölgeli Olanlar) [18]

MATLAB TZ kurulumdan itibaren aktif olarak gelir. Bu özelliği aktif etmek ya da kapatmak için “feature accel on/off” komutları kullanılır. TZ performansa etki eden diğer bir unsurdur. TZ kapatıldığında kırk zaman adımı için MATLAB FDTD programı çalışması yüz seksen yedi bin dokuz yüz otuz beş

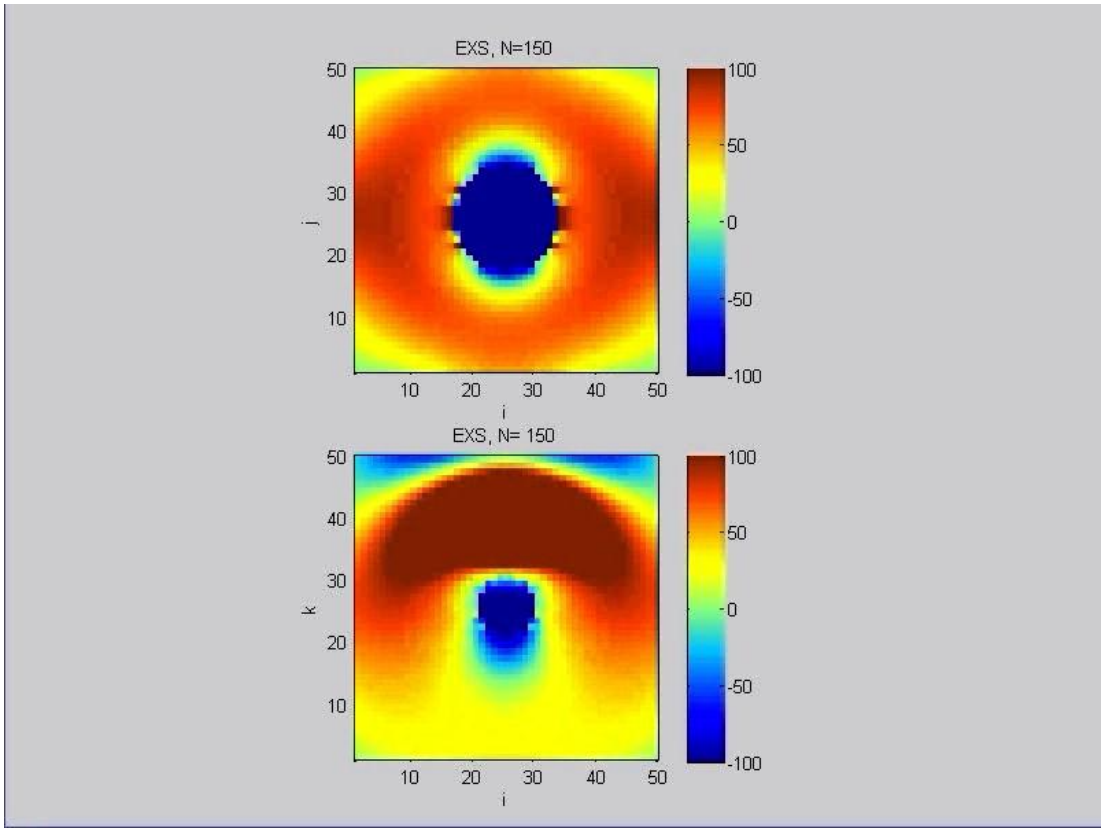
saniye sürmüştür. Aynı zaman adımı değerleri için TZ aktif edildiğinde ise bu süre on altı bin yedi yüz doksan dokuz saniye olarak gerçekleşmiştir. Bu denemenin sonucunda TZ'nin performansı negatif yönde etkilediği anlaşılmıştır.

### 3.1.7 MATLAB Film ve Grafik Uygulamaları

MATLAB yüksek seviyeli grafik rutinlerini kullanır. Bu rutinler temel olarak verinin gösterilmesi, noktasal grafikler, kutuplu koordinat grafikler, bar ve histogram grafikler, "contour" grafikler, yüzey grafikler, "mesh" grafikler ve animasyonların yaratılmasında kullanılır. MATLAB ile iki boyutlu ve üç boyutlu grafikler elde edilebilir. Uygulama geliştirici grafiklerin boyutlarını, renklerini, ve etiketleri üzerinde çalışma yapabilir. MATLAB'in "object oriented" grafik özellikleri sayesinde son kullanıcı etkileşimli olarak anlık grafikler üzerinde düşük seviyeli değişiklikler yapabilir veya grafiğin görselini istediği şekilde tasarlayabilir. Görsel (Şekil 3.5) ve (Şekil 3.6)'da FDTD programı ile EXS alan dağılımının film fonksiyonu ile hareketli "avi" görüntüsünden birer an gösterilmiştir [19]. Birinci şekil (Şekil 3.5)'deki görüntü yedinci saniyeye aittir. İkinci şekil (Şekil 3.6)'daki görüntü onuncu saniyeye aittir. Bu ekran görüntüsünde "ij" düzlemi "xy" düzlemini ifade ederken ik düzlemi "xz" düzleminde üç boyutlu elektromanyetik dalganın nesne ile çarpışması simule etmektedir. Test probleminin sayısal sonuçlarını anında bir animasyona aktarabilme olanağı MATLAB'ı öne çıkaran bir özelliktir.




Şekil 3.5 FDTD Film Saniye 7 [20]




Şekil 3.6 FDTD Film Saniye 10

MATLAB temelde diziler ile çalışmaktadır. Diziler ile bir resmin her pikselini koordinat ve özelliklerini ifade etmek mümkündür. Çoğu resim iki boyutlu diziler ile ifade edilebilmektedir. Bir I dizisindeki I(x,y) ifadesi I resminin x satırında ve y kolonunda bulunan ilgili renk değerini ifade etmektedir. RGB biçimindeki resimler üç boyutludur. Birinci indeks kırmızı renk değerini, ikinci indeks yeşil renk değerini, üçüncü indeks mavi indeks değerini saklar. MATLAB resim gösterimi için "double", "uint16" ve "uint8" veri tiplerini desteklemektedir. I dizisinin bir resim olduğunu düşünürsek sırasıyla I(x,y,R), I(x,y,G) ve I(x,y,B) sorgusunun cevapları "x,y" koordinatlarında ki pikselin "R,G,B" renk değerlerini verecektir. MATLAB temel resim formatları olan BMP(Microsoft Windows Bitmap), GIF(Graphics Interchange Files), HDF(Hierarchical Veri Format), JPEG(Joint Photographic Experts Group), PCX(Paintbrush), PNG(Portable Network Graphics), TIFF(Tagged Image File Format) ve XWD(X Windows Dump)'ı formatlarını desteklemektedir.

### 3.1.8 MATLAB Kullanımının Dezavantajları

Matris operasyonlarında değişkenler çalışma esnasında tiplerini değiştirebilmeleri bir avantaj iken değişkenler için ayrılan fazla hafıza performans anlamında bir dezavantaja dönüşebilmektedir.

Yorumlanmış bir dildir. Önce yorumlanmaktadır, sonra derlenmektedir. Bu da performans sorunlarına yol açmaktadır. MATLAB programında döngü kullanımı programı yavaşlatmaktadır. Döngü yerine "Mathworks" firması vektör işlemlerini önermektedir. Vektör işlemleri ile aynı döngü işlemi yirmi kat daha hızlı yapılabilmektedir. Fakat her işlemi kolaylıkla vektör işlemlerine çevirmek döngü kullanımı kadar kolay olmamaktadır.

MATLAB, C veya FORTRAN programlarına nazaran lisans yönünden maliyeti daha yüksek olabilmektedir. Bu da iş çevreleri için maliyeti arttıran bir unsurdur. Kullanımını ve piyasaya dağılımını yavaşlatmaktadır. Buna karşın öğrenciler için ücretsiz eğitim sürümleri bulunmaktadır.

## 3.2 FORTRAN Programlama Ortamı

FORTRAN elli yılı aşkın süredir kullanılmaktadır. Günümüzde süper bilgisayarları ve yüksek hızda matematiksel çözüm gerektiren uygulamalar halen FORTRAN dili ile programlanmaktadır. Çoğunlukla (hava tahminleri, akışkanlar dinamiği) gibi simülasyonlarda kullanılmaktadır.

1950'li yıllara gelindiğinde programcılar açısından "assembler" dilinin kullanarak program yazımında oldukça ciddi zorluklar yaşanmaktaydı. Makine dillerinin kullanıcı tarafından anlaşılma zorluğu ve mantıksal süreçlerin uygulanabilirliğinin zorluğundan dolayı daha zor problemleri "assembler"'ın anlayabileceği komutlara çevirebilecek ara programlar geliştirilmesi düşünüldü. IBM firması bünyesinde John Backus tarafından yönetilen bir ekip tarafından FORTRAN (Mathematical FORmula TRANslation kelimelerinin kısaltılmışıdır) programlama dili 1957 yılında IBM 704 bilgisayarları için geliştirildi. FORTRAN ile birlikte matematik işlemleri daha kolay programlanabilir hale gelmiştir. FORTRAN daha hızlı ve etkin bir dil olmuştur. [21]

Zaman içerisinde FORTRAN ile yazılan programlarda yazım farklılaştırmaları oluşmaya başladı. Yazılım hatalarından korunmak ve herkesin ortak bir dil kullanmasını sağlamak amacıyla tarihte ilk defa ANSI (American National Standarts Institute) bir programlama dilinin kullanım ve yapısal özelliklerinin sınırlarını çizmiştir. 1966 yılında yapılan bu çalışma ile FORTRAN 66 ortaya çıkmıştır [22]. Benzer çalışmalar zaman içerisinde FORTRAN 77, FORTRAN 90, FORTRAN 95 ve FORTRAN 2003 için de yapılmıştır.

### 3.2.1 FORTRAN Kullanımının Dezavantajları

FORTRAN özellikle matematik alanında yazılan kütüphaneler ile ön plana çıkmaktadır. LAPACK (Linear Algebra Pack) kütüphanesi 1992 yılında FORTRAN 90 için geliştirilmiştir. Diğer programlama dilleri benzer çalışmaları yaptılarsa da FORTRAN kadar başarılı olamamışlardır.

FORTRAN 90 HPFF (High Performance FORTRAN Forum) tarafından geliştirilen versiyonu ile paralel hesaplama özelliğini kazanmıştır. Bu özellik bir veri modeli kullanmaktadır. Bu model sayesinde bir diziyi ilgilendiren işlemler

birden fazla işlemciye eş zamanlı dağıtılabilmektedir. Bu da işlemleri oldukça hızlandırmaktadır. Paralel işleme özelliklerini kullanabilmek için programın yazımı sırasında paralel işlem olarak yapılması istenen döngüler için derleyicinin anlayabileceği farklı bir yazım yöntemi kullanılır.

Alt yordam argümanları değer yerine referans ile verilmektedir. Bu da verinin korunmasını zorlaştırmaktadır. Veri kapsamı sınırlıdır. "COMMON" ve "LOCAL" bloklarında değişkenler tanımlanabilmektedir. Bunların haricinde bir veri kapsamı bulunmamaktadır. Bazen aynı işlemi başka bir değişken için yapabilmek için bir alt yordam ya da programı tekrar klonlamak gerekebilmektedir.

Döngü koşulları sınırlıdır. Çoğu zaman döngü akışını sağlamak için "GOTO" durumu kullanılabilmektedir. FORTRAN FDTD programında, programcılıkta pek tercih edilmeyen "GOTO" durumu pek çok yerde kullanılmıştır. Grafik çizme ve arayüz oluşturma özellikleri bulunmamaktadır.

FORTRAN'ın sıkı yazım kuralları bulunmaktadır. Örnek olarak, eskiden kullanılan hafıza kartlarının yapısından dolayı program satırlarının yedi ile yetmişinci satırlar arasında yazılması veya yeni satır kullanma zorunlulukları gibi kurallar verilebilir. Mevcut bir FORTRAN uygulamasıyla çalışırken farklı derleyeciler yazım kuralları ile ilgili farklı hatalar üretebilmektedir. Hata mesajları çoğunlukla hatayla tam ilgili olmaması ya da çeşitli hatalar için benzer hataları vermelerinden dolayı basit sorunları çözmek başlangıç için zaman alıcı olabilmektedir.

Günümüzde internet üzerinde ya da kütüphanelerde FORTRAN üzerine yazılmış kitap, makale veya paylaşılan çalışmalar bulmak oldukça zordur. Grafik desteği MATLAB kadar güçlü değildir. Uygulama geliştirme sırasındaki kod yazımı sıkı kurallara bağlanmıştır. Derleme sırasında karşılaşılan hatalar MATLAB kadar kolay analiz edilememektedir. Diğer diller ile birlikte kullanılabilirlik desteği bulunmamaktadır. Ücretli ve ücretsiz her platform için derleyeciler bulunmasına rağmen derleyici performansları ve kullanılabilirlikleri değişkenlik göstermektedir.

# 4 FDTD UYGULAMA BAŞARIM KARŞILAŞTIRMASI

"AMD Athlon 64 X2 Dual Core Processor 3800+" işlemci, 2 GB RAM ve linux 2.6.32-22-generic kernel platformunda altı virgül kırk beş saniye sürmüştür. Yapılan ölçüm sonuçları kabul edilebilir düzeylerdedir. Çalışma uzayı büyütültüğünde uygulamanın performansında orantısal düşmeler gözlemlenmektedir.

Aynı koşullar altında FDTD kodunu MATLAB programında çalıştırıldığında (Şekil 4.1)'deki sonuçlar elde edilmiştir. MATLAB ortamına uyarlanan aynı FORTRAN uygulaması çalıştırıldığında toplam çalışma süresi yüz yirmi saniye olarak gerçekleşmiştir.


Şekil 4.1 MATLAB Performans Sonuçları

FORTRAN ve MATLAB uygulamaları arasında on beş kat hız farkı oluşmaktadır. Uygulamada performans arttırıcı aksiyonlar alınmadan önce aradaki hız farkı yüz kata kadar çıkabilmekteydi. MATLAB hızlandırmak için yapılan aksiyonlar daha çok fonksiyon yapısı, değişken yapısı ve döngü yapılarında yoğunlaşmıştır.

MATLAB'da alt fonksiyonlar ayrı dosyalar içerisinde yazılabilmektedir. Dosya ismi "fonksiyon_adı.m" şeklinde kullanılmaktadır. Ana program fonksiyonları "function [out1, out2, ...] = myfun(in1, in2, ...)" şeklinde çağırır.

Eğer ana fonksiyonda kullanılan değişken ve sabitler alt fonksiyon tarafından istenecek ise ilgili değişken ve sabitler hem ana fonksiyon hem de alt fonksiyonda global olarak tanımlanmaktadır. Global değişken ve sabitlerin fonksiyonlar tarafından çağırılması performans sorunlarına yol açar. Bundan dolayı yaklaşım olarak ana fonksiyon içerisinde alt fonksiyonları tanımlamak daha performanslı olmaktadır. Bu yöntem ile değişken ve sabitleri global olarak MATLAB'a bildirilmesi zorunluluğu ortadan kalkmaktadır. Her durumda "MEX" ya da gömülü hazır bulunan fonksiyonların haricindeki fonksiyon çağrılması performansı negatif yönde etkilemektedir.

FDTD MATLAB uygulaması dört boyut üzerinde çalışmaktadır. İlk üç boyut koordinat sistemi iken dördüncü boyut zaman adımlarını ifade etmektedir. Zamana bağlı olarak üç boyutlu dizilerdeki elektriksel ve manyetik alan değişkenleri materyal geometri dizisinden faydalanarak hesaplamalarını yapmaktadır. FORTRAN ortamında bu işlemi yapabilmek için programcılıkta sık kullanılan for döngüsünden yararlanılmıştır. Döngüler yerine matris operasyonlarının yapılması önerilmektedir. Matris operasyonları çoğu kaynakta vektör işlemleri anlamında kullanılmaktadır. Vektörel işlemlerin açık bir şekilde daha yüksek performansa sahip olduğu başarıyla test edilmiştir. MATLAB dizilerden değer çağırmak ve alt fonksiyona giriş göndermek için aynı dizim kurallarını kullanmaktadır. Bir A dizisindeki "I, J, K" indeksindeki değer "A(I,J,K)" şeklinde çağırılırken, bir B alt fonksiyonuna I ve J girişlerinin gönderilmesi "B(I,K)" şeklinde yapılmaktadır. Bu durum uygulamanın okunmasında zorluk yaratmaktadır.

Matris operasyonlarında matris'in program başında "zeros" ya da "ones" "built-in" fonksiyonlarından biriyle tanımlanmasını önerilmektedir. Döngüler içerisinde sürekli büyüyen veya program içerisinde çeşitli veri girişi olan diziler programın başında yeteri kadar hafıza ayrılması performansı olumlu yönde etkileyeceği varsayılmaktadır. Yapılan denemelerde bu performans artışı gözle görülür biçimde hissedilmemiştir.

Özel "Sparse" fonksiyonu bir dizi ifadesidir. Dizinin sıfır haricindeki değerlerinin saklar. İfadeler için ilgili değerler harici sıfırdır ifadesini kullanır. Faydası değer girilmemiş olan yani sıfır tanımlı elemanlar için hafıza ayırmaması

ve işlem yapmamasıdır. Bu özelliği sayesinde matris operasyonlarını hızlanmasına katkıda bulunmaktadır. FDTD programında "sparse" dizi kullanımı mümkün olmamıştır. Bu durumun iki nedeni vardır. Birincisi, FDTD programında kayıplı dielektrik küre oluşturulurken dizilerin sıfır değerli elemanlarına ihtiyaç duyar. İkinci ise çok boyutlu dizilerde indeksleme özelliği sadece tam diziler için sağlanmaktadır. Elektrik ve manyetik alan dizilerinin güncellenmesi sırasında indekse ihtiyaç duyulduğundan dolayı "sparse" diziler kullanılamamaktadır.

Yapılan çalışmalar neticesinde FORTRAN uygulamasının MATLAB uygulamasından daha hızlı çalıştığı görülmüştür. Bunun en büyük nedeni döngü yapısı konusunda FORTRAN'ın daha hızlı, verimli ve etkin olmasıdır. MATLAB uygulamasının yavaşlığı incelendiğinde E.Alan güncellemesinde "if" durum cümlesi ve buna bağlı komutlarda yavaşlık olduğu anlaşılmıştır.

# 5 SONUÇ VE DEĞERLENDİRME

Öngörülen FDTD programının FORTRAN koduna en yakın haliyle MATLAB ortamında modellenmesinde karşılaşılan en önemli sorun çalışma zamanının sekiz yüz saniye kadar sürmesi olmuştur. Bu süre bu boyuttaki bir problemin çözümü için oldukça yüksektir. Performans problemi kaynakların önerileri ışığında adım adım iyileştirilmiştir. Döngüler, koşullu durumlar, fonksiyon sayıları, global değişken tanımları, TZ ve "profiler" kullanımının performansa negatif etkisi görülmüş ve bu alanlarda iyileştirme yapılarak sekiz kat hız artışı sağlanmıştır.

Gelecek açısından bakıldığında döngüler ve koşullu durumlar tamamıyla programdan kaldırılır veya bir "MEX" dosyasına çevrilirse programın FORTRAN'ın performansına yaklaşması beklenebilir. Diğer bir performans arttırıcı yapılabilecek çalışma ise programın paralel sunucular üzerinden çalıştırılması olacaktır. FDTD yöntemi paralel çalışan bilgisayar mimarilerine uygulanabilir bir yöntemdir. MATLAB'ın parallel toolbox özellikleri kullanılarak programın hızının daha da arttırabileceği gözükmektedir.

MATLAB grafik özellikleri ile bu problemde elektromanyetik alan içerisinde bulunan nesnenin tanımı 2D ortamlar için resimler üzerinden yapılabilir. Herhangi beyaz arka planlı 2D bir obje resmi bir diziye aktarılarak bu dizi içerisinde beyaz renk koduna sahip olmayan piksellerin materyal tipi, çalıştığımız örnekte olduğu gibi kayıplı dielektrik verilerek, farklı nesnelerin elektromanyetik dalgayla etkileşimleri incenenebilir.

Genelde bilimsel hesaplama alanındaki 3D ve zaman dahil edilince 4D kabul edilen problemlerin modellenmesi ve kodlanmasında FORTRAN dili hala yaygın olarak kullanılmaktadır. Keza MATLAB, bilimsel hesaplama alanında en çok kullanılan ortam olmasına karşın, performans yavaşlığı nedeniyle 3D veya 4D modellemeler için tercih edilmemektedir. Bu tezde FDTD tekniğini kullanılarak 3D kayıplı dielektrik problemi MATLAB ortamıyla çözülmüş ve MATLAB'ın performansı iyileştirilmiştir.

# KAYNAKLAR


[1] Yee, K. S.:”"Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media". IEEE Trans. On Antennas and Propagation vol.14 pp:302–307 **(1966).**

[2] Allen Taflove and Susan C. Hagness.:“Computational Electrodynamics: The Finite-Difference Time-Domain Method, 2nd edition”.. Artech House Publishers. ISBN 1-58053-076-1 **(2000)**.

[3] Dennis L. Sullivan, “ Electromagnetic Simulation using the FDTD Method”, IEEE Press. **(2000)**.

[4] Clayton R. Paul & Syed A. Nasar “Introduction to Electromagnetic Fields” McGraw-Hill, **(1987)**.

[5] Stephen D. Gedney, “An Anisotropic PML Absorbing Media for the FDTD Simulation of Fields in Lossy and Dispersive Media”, Electromagnetics, Volume 16, pages 399 – 415**(4 Haziran 1996)**.

[6] Kunz, K. S.; Luebbers, R. J.: “The Finite Difference Time Domain Method for Electromagnetics” CRC Press, **(1993)**.

[7] D”avi”d K. Cheng, “Field and Wave Electromagnetics”, Pearson Education, **(1989)**.

[8] James Clerk Maxwell.: “A Treatise on Electricity and Magnetism”, Oxford University Press, **(1891)**.

[9] Taflove, A.; Brodwin, M. E.:“Numerical solution of steady-state electromagnetic scattering problems using the time-dependent maxwell's equations”, **(1974)**.

[10] R. Courant, K. F. (March 1967). On the partial difference equations of mathematical physics. IBM Journal.

[11] Nyquist, H.: “Certain topics in telegraph transmission theory”, **(1928).**

[12] Mur, G.: “Absorbing boundary conditions for the finite difference approximation of the time-domain electromagnetic-field equations”, IEEE, **(1981)**.

[13] Accelerating MATLAB, [http://www.mathworks.com/company/newsletters/digest/sept02/accel_MATLAB.pdf] **(2010)**.

[14] Palm, W. J. :Introduction to MATLAB 7 for Engineers**(2004)**.

[15] İnan, Y. A.: MATLAB Temel Seviye Semineri **(2010)**.

[16] MEX Files Guide, [http://www.mathworks.com/support/tech-notes/1600/1605.html] **(Mayıs 2010)**.

[17] J.Chapman, S.: "MATLAB Programming for Engineers. Thomson" **(2008)**.

[18] "Supported and Compatible Compilers – Release 2010a" Mathworks, [http://www.mathworks.com/support/compilers/R2010a/index.html], (Mayıs 2010).

[19] Susan C. Hagness.:"3-D FDTD code with PEC boundaries" Department of Electrical and Computer Engineering University of Wisconsin-Madison **(Şubat 2000)**.

[20] Övüç, Hasan.:3D FDTD Video Simulation http://video.yahoo.com/watch /7649337/20320775] **(2010)**.

[21] Michael, M.; John, R.; Malcolm, C.: "FORTRAN 95/2003", Oxford, **(2004)**.

[22] The Fortran 66 Standard, [http://www.fh-jena.de/~kleine/history/languages/ /ansi-x3Dot9-1966-Fortran66.pdf], **(01 Haziran 2010).**

# EKLER

## EK-1: FDTD MATLAB Programı

```
function fdtda
% ÇALIŞMA ALANININ TEMİZLENMESİ
clear

% DOSYALAR
F10 = fopen('NZOUT3D.txt','w+');
F17 = fopen('DIAGS3D.txt','w+');

% SABİTLER VE DEĞİŞKENLER

NX=34;
NY=34;
NZ=34;
NX1=NX-1;
NY1=NY-1;
NZ1=NZ-1;
NTEST=4;
NSTOP=40;
DELX=1./17;
DELY=1./17;
DELZ=1./17;
THINC=180;
PHINC=180;
ETHINC=1;
EPHINC=0;
AMP=1000;
BETA=64;
EPS0=8.854E-12;
XMU0=1.2566306E-6;
T=0;

% DİZİLERİN OLUŞTURULMASI
IDONE=zeros(NX,NY,NZ);
IDTWO=zeros(NX,NY,NZ);
IDTHRE=zeros(NX,NY,NZ);
EXS=zeros(NX,NY,NZ);
EYS=zeros(NX,NY,NZ);
EZS=zeros(NX,NY,NZ);
HXS=zeros(NX,NY,NZ);
HYS=zeros(NX,NY,NZ);
HZS=zeros(NX,NY,NZ);
STORE=zeros(1,4);
ESCTC=zeros(1,9);
EINCC=zeros(1,9);
EDEVCN=zeros(1,9);
ECRLX=zeros(1,9);
ECRLY=zeros(1,9);
ECRLZ=zeros(1,9);
EPS=zeros(1,9);
SIGMA=zeros(1,9);
EYSX1(1:4,1:NY1,1:NZ1)=zeros(4,33,33);
EYSX2(1:4,1:NY1,1:NZ1)=zeros(4,33,33);
EZSX1(1:4,1:NY1,1:NZ1)=zeros(4,33,33);
EZSX2(1:4,1:NY1,1:NZ1)=zeros(4,33,33);
EXSY1(1:NX1,1:4,1:NZ1)=zeros(33,4,33);
EXSY2(1:NX1,1:4,1:NZ1)=zeros(33,4,33);
EZSY1(1:NX1,1:4,1:NZ1)=zeros(33,4,33);
EZSY2(1:NX1,1:4,1:NZ1)=zeros(33,4,33);
EXSZ1(1:NX1,1:NY1,1:4)=zeros(33,33,4);
EXSZ2(1:NX1,1:NY1,1:4)=zeros(33,33,4);
EYSZ1(1:NX1,1:NY1,1:4)=zeros(33,33,4);
EYSZ2(1:NX1,1:NY1,1:4)=zeros(33,33,4);

for L=1:9,
        ESCTC(L)=0;
        EINCC(L)=0;
        EDEVCN(L)=0;
        ECRLX(L)=0;
        ECRLY(L)=0;
```

```
        ECRLZ(L)=0;
end

% SİMÜLASYON ORTAMININ KURULMASI

MTYPE=2;
RA=8.2;
SC=17.5;

for I=1:NX,
for J=1:NY,
for K=1:NZ,
            R=sqrt((I-SC)^2+(J-SC)^2+(K-SC)^2);
if (R <= RA),
              IDONE(I,J,K)=MTYPE;
              IDONE(I,J,K+1)=MTYPE;
              IDONE(I,J+1,K+1)=MTYPE;
              IDONE(I,J+1,K)=MTYPE;
              IDTWO(I,J,K)=MTYPE;
              IDTWO(I+1,J,K)=MTYPE;
              IDTWO(I+1,J,K+1)=MTYPE;
              IDTWO(I,J,K+1)=MTYPE;
              IDTHRE(I,J,K)=MTYPE;
              IDTHRE(I+1,J,K)=MTYPE;
              IDTHRE(I+1,J+1,K)=MTYPE;
              IDTHRE(I,J+1,K)=MTYPE;
end
end
end
end

% IDXXX DİZİ DEĞERLERİNİN KONTROLÜ

for I=1:NZ,
for J=1:NY,
for K=1:NX,
if((IDONE(I,J,K) >= 10) || (IDTWO(I,J,K) >= 10) || ...
                    (IDTHRE(I,J,K) >= 10))
                fprintf (F17,'HATA: GECERSIZ MATERYAL TIPI\n');
                fprintf (F17,'LOKASYON:');
                fprintf (F17,'%d %d %d','I','J','K','\n');
end
end
end
end

% KURULUM

C=1.0/sqrt(XMU0*EPS0);
PI=4.0*atan(1.0);

% DT'NIN "COURANT" TUTARLILIK KOŞULUNA GÖRE HESAPLANMASI

DTXI=C/DELX;
DTYI=C/DELY;
DTZI=C/DELZ;
DT=1./sqrt(DTXI^2+DTYI^2+DTZI^2);

% BETA TARAFINDAN BELİRLENEN "ALPHA DECAY" ORANININ HESAPLANMASI

ALPHA=(1./(BETA*DT/4))^2;

BETADT = BETA*DT;
PERIOD = 2*BETADT;

% OFFSET DEGERİ

OFF=1.0;

% GELEN DALGA İÇİN "COSINUS" YÖNLERİNİN HESAPLANMASI

COSTH=cos(PI*THINC/180);
SINTH=sin(PI*THINC/180);
COSPH=cos(PI*PHINC/180);
SINPH=sin(PI*PHINC/180);

% GELEN DALGA KOMPONENTLERININ GENLİĞİNİN HESAPLANMASI
```

```
AMPX=AMP*(ETHINC*COSTH*COSPH-EPHINC*SINPH);
AMPY=AMP*(ETHINC*COSTH*SINPH+EPHINC*COSPH);
AMPZ=AMP*(-ETHINC*SINTH);

% X, Y, Z HUCRE YERLEŞTİRMELERİ İÇİN BAĞIL UZAYSAL GECİKMENİN BULUNMASI

XDISP=-COSPH*SINTH;
YDISP=-SINPH*SINTH;
ZDISP=-COSTH;

% TÜM DİELEKTRİK MATERYALLER İÇİN AYNI GEÇİRGENLİĞİN SAĞLANMASI

for I=1:9,
        EPS(I)=EPS0;
        SIGMA(I)=0;
end

% MATERYAL TİPLERİNE GÖRE "EPS" VE "SIGMA" DEĞERLERİNİN TANIMLANMASI

EPS(2)=4*EPS0;
SIGMA(2)=0.005;

% DENKLEMLER İÇERSİNDE DÜZENLİ OLARAK KULLANILACAK SABİTLERİN HESAPLANMASI

% BOŞLUK HESAPLAMALARI İÇİN

 DTEDX=DT/(EPS0*DELX);
 DTEDY=DT/(EPS0*DELY);
 DTEDZ=DT/(EPS0*DELZ);
 DTMDX=DT/(XMU0*DELX);
 DTMDY=DT/(XMU0*DELY);
 DTMDZ=DT/(XMU0*DELZ);

% KAYIPLI DIELEKTRIKLER İÇİN

for I=2:9,
        ESCTC(I)=EPS(I)/(EPS(I)+SIGMA(I)*DT);
        EINCC(I)=SIGMA(I)*DT/(EPS(I)+SIGMA(I)*DT);
        EDEVCN(I)=DT*(EPS(I)-EPS0)/(EPS(I)+SIGMA(I)*DT);
        ECRLX(I)=DT/((EPS(I)+SIGMA(I)*DT)*DELX);
        ECRLY(I)=DT/((EPS(I)+SIGMA(I)*DT)*DELY);
        ECRLZ(I)=DT/((EPS(I)+SIGMA(I)*DT)*DELZ);
end

% DALGANIN UZAYDA DÜZGÜN OLARAK YAYILABİLMESİ İÇİN MAKSİMUM UZAYSAL
% GECİKMENİN BULUNMASI

DELAY=0;
if (XDISP <= 0)
    DELAY=DELAY-XDISP*NX1*DELX;
end
if (YDISP <= 0)
    DELAY=DELAY-YDISP*NY1*DELY;
end
if (ZDISP <= 0)
    DELAY=DELAY-ZDISP*NZ1*DELZ;
end

% DIŞ RADYASYON SINIR KOŞULU SABİTLERİNİN HESAPLANMASI

CXD=(C*DT-DELX)/(C*DT+DELX);
CYD=(C*DT-DELY)/(C*DT+DELY);
CZD=(C*DT-DELZ)/(C*DT+DELZ);
CXU=CXD;

% İKİNCİ SEVİYE ORDB SABİTLERİNİN HESAPLANMASI

CXX=2*DELX/(C*DT+DELX);
CYY=2*DELY/(C*DT+DELY);
CZZ=2*DELZ/(C*DT+DELZ);

CXFYD=DELX*C*DT*C*DT/(2.*DELY*DELY*(C*DT+DELX));
CXFZD=DELX*C*DT*C*DT/(2.*DELZ*DELZ*(C*DT+DELX));
CYFZD=DELY*C*DT*C*DT/(2.*DELZ*DELZ*(C*DT+DELY));
CYFXD=DELY*C*DT*C*DT/(2.*DELX*DELX*(C*DT+DELY));
CZFXD=DELZ*C*DT*C*DT/(2.*DELX*DELX*(C*DT+DELZ));
```

```
CZFYD=DELZ*C*DT*C*DT/(2.*DELY*DELY*(C*DT+DELZ));


fprintf (F17,'UZAY %d %d %d KADAR HÜCRE VE X,Y,Z YÖNLERİNDEDİR\n\n',NX,NY,NZ);
fprintf (F17,'HÜCRE BÜYÜKLÜĞÜ DELX %10.6f, DELY %10.6f, DELZ %10.6f
METREDİR\n\n',DELX,DELY,DELZ);
fprintf (F17,'ZAMAN ADIMI %12.6E SANİYEDİR, MAKSIMUM ZAMAN ADIM SAYISI %d
DIR\n\n',DT,NSTOP);
fprintf (F17,'GELEN GAUSSIAN ATIM GENLİĞİ %6.0f V/M, DECAY FACTOR ALPHA=%12.3E, WIDTH
BETA=%6.0f\n\n',AMP,ALPHA,BETA);
fprintf (F17,'GELEN YÜZEY DALGA POLARİZASYONU:\nBAĞIL ELEKTRIK ALAN THETA
KOMPONENTI %4.1fBAĞIL ELEKTRIK ALAN PHI KOMPONENTI %4.1f\n\n',ETHINC,EPHINC);
fprintf (F17,'THETA DAN GELEN YÜZEY DALGASI ACISI %d\nDERECE %d\n\n',THINC,PHINC');
fprintf (F17,'GELEN EX GENLİĞİ %d V/M\nGELEN EY GENLİĞİ%d V/M\nGELEN EZ GENLİĞİ%d
V/M\n\n',AMPX,AMPY,AMPZ);
fprintf (F17,BAĞIL UZAYSAL GECIKME = %d\n\n',DELAY);

for N=1:NSTOP,

%     EX DAĞILAN ALANININ HESAPLANMASI

for K=2:NZ1,
for J=2:NY1,
for I=1:NX1,

%     MATERYAL TİPİNİN KARAR VERİLMESİ

if (IDONE(I,J,K) == 0)

%     BOŞLUK

                EXS(I,J,K)=EXS(I,J,K)+(HZS(I,J,K)-HZS(I,J-1,K))*DTEDY-(...
                HYS(I,J,K)-HYS(I,J,K-1))*DTEDZ;
elseif (IDONE(I,J,K) == 1)

%     TAM İLETKEN

                DIST=((I-1)*DELX+0.5*DELX*OFF)*XDISP+((J-1)*DELY)*YDISP+...
                    ((K-1)*DELZ)...
                *ZDISP + DELAY;
                source=0;
                TAU=T-DIST/C;
if (TAU < 0)
elseif(TAU > PERIOD)
else
                    source=exp(-ALPHA*((TAU-BETADT)^2));
end

                EXI=AMPX*source;
                EXS(I,J,K)=-EXI;

else

%      KAYIPLI DİELEKTRİK

                DIST=((I-1)*DELX+0.5*DELX*OFF)*XDISP+((J-1)*DELY)*...
                    YDISP+((K-1)*DELZ)*ZDISP + DELAY;
                source=0;
                dsrce=0;
                TAU=T-DIST/C;

if (TAU < 0)
elseif(TAU > PERIOD)
else
                    source=exp(-ALPHA*((TAU-BETADT)^2));
                    dsrce=exp(-ALPHA*((TAU-BETADT)^2))*(-2*ALPHA*...
                        (TAU-BETADT));
end

%      GELEN DALGANIN X KOMPONENTİ
                exi=AMPX*source;

%      DİELEKTRİKLER ICIN GELEN DALGANIN ZAMAN TÜREVİ
                dexi=AMPX*dsrce;

                EXS(I,J,K)=EXS(I,J,K)*ESCTC(IDONE(I,J,K))-...
                     EINCC(IDONE(I,J,K))*exi-...
```

```
                    EDEVCN(IDONE(I,J,K))*dexi+(HZS(I,J,K)-...
                    HZS(I,J-1,K))*ECRLY(IDONE(I,J,K))...
-(HYS(I,J,K)-HYS(I,J,K-1))*ECRLZ(IDONE(I,J,K));
end
end
end
end

%       EY DAĞILAN ELEKTRİK ALANININ HESAPLANMASI

for K=2:NZ1,
for J=1:NY1,
for I=2:NX1,

%       MATERYAL TİPİNİN KARAR VERİLMESİ

if (IDTWO(I,J,K) == 0)

%     BOŞLUK

            EYS(I,J,K)=EYS(I,J,K)+(HXS(I,J,K)-HXS(I,J,K-1))*DTEDZ...
                    -(HZS(I,J,K)-HZS(I-1,J,K))*DTEDX;
elseif (IDTWO(I,J,K) == 1)

%     TAM İLETKEN

                DIST=((I-1)*DELX)*XDISP+((J-1)*DELY+0.5*DELY*OFF)*YDISP+...
                    ((K-1)*DELZ)*...
                ZDISP + DELAY;
                source=0;
                TAU=T-DIST/C;
if (TAU < 0)
elseif(TAU > PERIOD)
else
                    source=exp(-ALPHA*((TAU-BETADT)^2));
end

                EYI=AMPX*source;
                EYS(I,J,K)=-EYI;
else

%      KAYIPLI DİELEKTRİK

                source=0;
                dsrce=0;
                DIST=((I-1)*DELX)*XDISP+((J-1)*DELY+0.5*DELY*OFF)*YDISP...
                    +((K-1)*DELZ)*...
                ZDISP + DELAY;
                TAU=T-DIST/C;

if (TAU < 0)
elseif(TAU > PERIOD)
else
                    source=exp(-ALPHA*((TAU-BETADT)^2));
                    dsrce=exp(-ALPHA*((TAU-BETADT)^2))*(-2*ALPHA*...
                        (TAU-BETADT));
end

%       GELEN DALGANIN Y KOMPONENTİ
                eyi=AMPY*source;
%       DIELEKTRIKLER İÇİN GELEN DALGANIN ZAMAN TÜREVİ
deyi=AMPY*dsrce;

                EYS(I,J,K)=EYS(I,J,K)*ESCTC(IDTWO(I,J,K))-...
                     EINCC(IDTWO(I,J,K))*eyi-...
                     EDEVCN(IDTWO(I,J,K))*deyi+(HXS(I,J,K)-...
                     HXS(I,J,K-1))*ECRLZ(IDTWO(I,J,K))-(HZS(I,J,K)-...
                     HZS(I-1,J,K))*ECRLX(IDTWO(I,J,K));
end
end
end
end

%       EZ DAĞILAN ALANININ HESAPLANMASI

for K=1:NZ1,
for J=2:NY1,
```

```
for I=2:NX1,

%     MATERYAL TIPININ KARAR VERILMESI

if (IDTHRE(I,J,K) == 0)

%     BOŞLUK

                    EZS(I,J,K)=EZS(I,J,K)+(HYS(I,J,K)-HYS(I-1,J,K))*...
                        DTEDX-(HXS(I,J,K)-HXS(I,J-1,K))*DTEDY;
elseif (IDTHRE(I,J,K) == 1)

%     TAM İLETKEN
                    DIST=((I-1)*DELX)*XDISP+((J-1)*DELY)*YDISP+((K-1)*...
                        DELZ+0.5*DELZ*OFF)*ZDISP + DELAY;
                    source=0;
                    TAU=T-DIST/C;
if (TAU < 0)
elseif(TAU > PERIOD)
else
                        source=exp(-ALPHA*((TAU-BETADT)^2));
end

                    EZI=AMPX*source;
                    EZS(I,J,K)=-EZI;
else

%      KAYIPLI DİELEKTRİK
                    DIST=((I-1)*DELX)*XDISP+((J-1)*DELY)*YDISP+((K-1)*...
                        DELZ+0.5*DELZ*OFF)*ZDISP + DELAY;
                    dsrce=0;
                    source=0;
                    TAU=T-DIST/C;

if (TAU < 0)
elseif(TAU > PERIOD)
else
                        source=exp(-ALPHA*((TAU-BETADT)^2));
                        dsrce=exp(-ALPHA*((TAU-BETADT)^2))*(-2*ALPHA*...
                            (TAU-BETADT));
end

%      GELEN DALGANIN Z KOMPONENTİ
                    ezi=AMPZ*source;

%      DİELEKTRİKLER İÇİN GELEN DALGANIN ZAMAN TÜREVİ
                    dezi=AMPZ*dsrce;

                    EZS(I,J,K)=EZS(I,J,K)*ESCTC(IDTHRE(I,J,K))-...
                    EINCC(IDTHRE(I,J,K))*ezi-...
                    EDEVCN(IDTHRE(I,J,K))*dezi+(HYS(I,J,K)-...
                    HYS(I-1,J,K))*ECRLX(IDTHRE(I,J,K))-(HXS(I,J,K)-...
                    HXS(I,J-1,K))*ECRLY(IDTHRE(I,J,K));
end
end
end
end

%   YX İÇİN İLK SEVIYE ORBC UYGULANMASI

for K=2:NZ1,
            J=1;
            EYS(1,J,K)=EYSX1(2,J,K)+CXD*(EYS(2,J,K)-EYSX1(1,J,K));
            EYS(NX,J,K)=EYSX1(3,J,K)+CXU*(EYS(NX1,J,K)-EYSX1(4,J,K));
            J=NY1;
            EYS(1,J,K)=EYSX1(2,J,K)+CXD*(EYS(2,J,K)-EYSX1(1,J,K));
            EYS(NX,J,K)=EYSX1(3,J,K)+CXU*(EYS(NX1,J,K)-EYSX1(4,J,K));
end
for J=2:NY1-1,
            K=2;
            EYS(1,J,K)=EYSX1(2,J,K)+CXD*(EYS(2,J,K)-EYSX1(1,J,K));
            EYS(NX,J,K)=EYSX1(3,J,K)+CXU*(EYS(NX1,J,K)-EYSX1(4,J,K));
            K=NZ1;
            EYS(1,J,K)=EYSX1(2,J,K)+CXD*(EYS(2,J,K)-EYSX1(1,J,K));
            EYS(NX,J,K)=EYSX1(3,J,K)+CXU*(EYS(NX1,J,K)-EYSX1(4,J,K));
end
```

```
%       KALAN YÜZEYLER İÇİN İKİNCİ SEVİYE ORBC UYGULANMASI
for K=3:NZ1-1,
for J=2:NY1-1,
             EYS(1,J,K)=-EYSX2(2,J,K)+CXD*(EYS(2,J,K)+EYSX2(1,J,K))...
                  +CXX*(EYSX1(1,J,K)+EYSX1(2,J,K))+CXFYD*(EYSX1(1,J+1,K)...
                  -2.*EYSX1(1,J,K)+EYSX1(1,J-1,K)+EYSX1(2,J+1,K)-2.*...
                  EYSX1(2,J,K)+EYSX1(2,J-1,K))+CXFZD*(EYSX1(1,J,K+1)-2.*...
                  EYSX1(1,J,K)+EYSX1(1,J,K-1)+EYSX1(2,J,K+1)-2.*...
                  EYSX1(2,J,K)+EYSX1(2,J,K-1));
             EYS(NX,J,K)=-EYSX2(3,J,K)+CXD*(EYS(NX1 ,J,K)+EYSX2( 4,J,K))...
                  +CXX*(EYSX1(4,J,K)+EYSX1(3,J,K))+CXFYD*(EYSX1(4,J+1,K)...
                  -2.*EYSX1(4,J,K)+EYSX1(4,J-1,K)+EYSX1(3,J+1,K)-2.*...
                  EYSX1(3,J,K)+EYSX1(3,J-1,K))+CXFZD*(EYSX1(4,J,K+1)-2.*...
                  EYSX1(4,J,K)+EYSX1(4,J,K-1)+EYSX1(3,J,K+1)-2.*...
                  EYSX1(3,J,K)+EYSX1(3,J,K-1));
end
end

%       HESAPLANAN DEĞERLERİN KAYDEDİLMESİ
for K=2:NZ1,
for J=1:NY1,
              EYSX2(1,J,K)=EYSX1(1,J,K);
              EYSX2(2,J,K)=EYSX1(2,J,K);
              EYSX2(3,J,K)=EYSX1(3,J,K);
              EYSX2(4,J,K)=EYSX1(4,J,K);
              EYSX1(1,J,K)=EYS(1,J,K);
              EYSX1(2,J,K)=EYS(2,J,K);
              EYSX1(3,J,K)=EYS(NX1,J,K);
              EYSX1(4,J,K)=EYS(NX,J,K);
end
end

%   ZX İÇİN BİRİNCİ SEVİYE ORDB UYGULANMASI

for K=1:NZ1,
        J=2;
        EZS(1,J,K)=EZSX1(2,J,K)+CXD*(EZS(2,J,K)-EZSX1(1,J,K));
        EZS(NX,J,K)=EZSX1(3,J,K)+CXU*(EZS(NX1,J,K)-EZSX1(4,J,K));
        J=NY1;
        EZS(1,J,K)=EZSX1(2,J,K)+CXD*(EZS(2,J,K)-EZSX1(1,J,K));
        EZS(NX,J,K)=EZSX1(3,J,K)+CXU*(EZS(NX1,J,K)-EZSX1(4,J,K));
end
for J=3:NY1-1,
        K=1;
        EZS(1,J,K)=EZSX1(2,J,K)+CXD*(EZS(2,J,K)-EZSX1(1,J,K));
        EZS(NX,J,K)=EZSX1(3,J,K)+CXU*(EZS(NX1,J,K)-EZSX1(4,J,K));
        K=NZ1;
        EZS(1,J,K)=EZSX1(2,J,K)+CXD*(EZS(2,J,K)-EZSX1(1,J,K));
        EZS(NX,J,K)=EZSX1(3,J,K)+CXU*(EZS(NX1,J,K)-EZSX1(4,J,K));
end

%   KALAN YUZEYLER ICIN IKINCI SEVIYE ORBC UYGULANMASI

for K=2:NZ1-1,
for J=3:NY1-1,
             EZS(1,J,K)=-EZSX2(2,J,K)+CXD*(EZS(2,J,K)+EZSX2(1,J,K))...
                 +CXX*(EZSX1(1,J,K)+EZSX1(2,J,K))...
                 +CXFYD*(EZSX1(1,J+1,K)-2.*EZSX1(1,J,K)+EZSX1(1,J-1,K)...
                 +EZSX1(2,J+1,K)-2.*EZSX1(2,J,K)+EZSX1(2,J-1,K))...
                 +CXFZD*(EZSX1(1,J,K+1)-2.*EZSX1(1,J,K)+EZSX1(1,J,K-1)...
                 +EZSX1(2,J,K+1)-2.*EZSX1(2,J,K)+EZSX1(2,J,K-1));
             EZS(NX,J,K)=-EZSX2(3,J,K)+CXD*(EZS(NX1,J,K)+EZSX2(4,J,K))...
                 +CXX*(EZSX1(4,J,K)+EZSX1(3,J,K))...
                 +CXFYD*(EZSX1(4,J+1,K)-2.*EZSX1(4,J,K)+EZSX1(4,J-1,K)...
                 +EZSX1(3,J+1,K)-2.*EZSX1(3,J,K)+EZSX1(3,J-1,K))...
                 +CXFZD*(EZSX1(4,J,K+1)-2.*EZSX1(4,J,K)+EZSX1(4,J,K-1)...
                 +EZSX1(3,J,K+1)-2.*EZSX1(3,J,K)+EZSX1(3,J,K-1));
end
end

%   HESAPLANAN DEGERLERİN KAYDEDİLMESİ

for K=1:NZ1,
for J=2:NY1,
           EZSX2(1,J,K)=EZSX1(1,J,K);
           EZSX2(2,J,K)=EZSX1(2,J,K);
           EZSX2(3,J,K)=EZSX1(3,J,K);
```

```
          EZSX2(4,J,K)=EZSX1(4,J,K);
          EZSX1(1,J,K)=EZS(1,J,K);
          EZSX1(2,J,K)=EZS(2,J,K);
          EZSX1(3,J,K)=EZS(NX1,J,K);
          EZSX1(4,J,K)=EZS(NX,J,K);
end
end

%   ZY ICIN BIRINCI ORBC UYGULANMASI

for K=1:NZ1,
        I=2;
        EZS(I,1,K)=EZSY1(I,2,K)+CYD*(EZS(I,2,K)-EZSY1(I,1,K));
        EZS(I,NY,K)=EZSY1(I,3,K)+CYD*(EZS(I,NY1,K)-EZSY1(I,4,K));
        I=NX1;
        EZS(I,1,K)=EZSY1(I,2,K)+CYD*(EZS(I,2,K)-EZSY1(I,1,K));
        EZS(I,NY,K)=EZSY1(I,3,K)+CYD*(EZS(I,NY1,K)-EZSY1(I,4,K));
end
for I=3:NX1-1,
        K=1;
        EZS(I,1,K)=EZSY1(I,2,K)+CYD*(EZS(I,2,K)-EZSY1(I,1,K));
        EZS(I,NY,K)=EZSY1(I,3,K)+CYD*(EZS(I,NY1,K)-EZSY1(I,4,K));
        K=NZ1;
        EZS(I,1,K)=EZSY1(I,2,K)+CYD*(EZS(I,2,K)-EZSY1(I,1,K));
        EZS(I,NY,K)=EZSY1(I,3,K)+CYD*(EZS(I,NY1,K)-EZSY1(I,4,K));
end

%       KALAN YÜZEYLER İÇİN İKİNCİ SEVİYE ORBC UYGULANMASI

for K=2:NZ1-1,
for I=3:NX1-1,
           EZS(I,1,K)=-EZSY2(I,2,K)+CYD*(EZS(I,2,K)+EZSY2(I,1,K))...
               +CYY*(EZSY1(I,1,K)+EZSY1(I,2,K))...
               +CYFXD*(EZSY1(I+1,1,K)-2.*EZSY1(I,1,K)+EZSY1(I-1,1,K)...
               +EZSY1(I+1,2,K)-2.*EZSY1(I,2,K)+EZSY1(I-1,2,K))...
               +CYFZD*(EZSY1(I,1,K+1)-2.*EZSY1(I,1,K)+EZSY1(I,1,K-1)...
               +EZSY1(I,2,K+1)-2.*EZSY1(I,2,K)+EZSY1(I,2,K-1));
         EZS(I,NY,K)=-EZSY2(I,3,K)+CYD*(EZS(I,NY1,K)+EZSY2(I,4,K))...
             +CYY*(EZSY1(I,4,K)+EZSY1(I,3,K))...
             +CYFXD*(EZSY1(I+1,4,K)-2.*EZSY1(I,4,K)+EZSY1(I-1,4,K)...
             +EZSY1(I+1,3,K)-2.*EZSY1(I,3,K)+EZSY1(I-1,3,K))...
             +CYFZD*(EZSY1(I,4,K+1)-2.*EZSY1(I,4,K)+EZSY1(I,4,K-1)...
             +EZSY1(I,3,K+1)-2.*EZSY1(I,3,K)+EZSY1(I,3,K-1));
end
end

%       HESAPLANAN DEĞERLERİN KAYDEDİLMESİ

for K=1:NZ1,
for I=2:NZ1,
           EZSY2(I,1,K)=EZSY1(I,1,K);
           EZSY2(I,2,K)=EZSY1(I,2,K);
           EZSY2(I,3,K)=EZSY1(I,3,K);
           EZSY2(I,4,K)=EZSY1(I,4,K);
           EZSY1(I,1,K)=EZS(I,1,K);
           EZSY1(I,2,K)=EZS(I,2,K);
           EZSY1(I,3,K)=EZS(I,NY1,K);
           EZSY1(I,4,K)=EZS(I,NY,K);
end
end

%   XY İÇİN BİRİNCİ SEVİYE ORBC UYGULANMASI

for K=2:NZ1,
        I=1;
        EXS(I,1,K)=EXSY1(I,2,K)+CYD*(EXS(I,2,K)-EXSY1(I,1,K));
        EXS(I,NY,K)=EXSY1(I,3,K)+CYD*(EXS(I,NY1,K)-EXSY1(I,4,K));
        I=NX1;
        EXS(I,1,K)=EXSY1(I,2,K)+CYD*(EXS(I,2,K)-EXSY1(I,1,K));
        EXS(I,NY,K)=EXSY1(I,3,K)+CYD*(EXS(I,NY1,K)-EXSY1(I,4,K));
end
for I=2:NX-1,
        K=2;
        EXS(I,1,K)=EXSY1(I,2,K)+CYD*(EXS(I,2,K)-EXSY1(I,1,K));
        EXS(I,NY,K)=EXSY1(I,3,K)+CYD*(EXS(I,NY1,K)-EXSY1(I,4,K));
        K=NZ1;
        EXS(I,1,K)=EXSY1(I,2,K)+CYD*(EXS(I,2,K)-EXSY1(I,1,K));
```

```
        EXS(I,NY,K)=EXSY1(I,3,K)+CYD*(EXS(I,NY1,K)-EXSY1(I,4,K));
end

%   KALAN YÜZEYLER İÇİN İKİNCİ SEVİYE ORBC UYGULANMASI

for K=3:NZ1-1,
for I=2:NX1-1,
            EXS(I,1,K)=-EXSY2(I,2,K)+CYD*(EXS(I,2,K)+EXSY2(I,1,K))...
                +CYY*(EXSY1(I,1,K)+EXSY1(I,2,K))...
                +CYFXD*(EXSY1(I+1,1,K)-2.*EXSY1(I,1,K)+EXSY1(I-1,1,K)...
                +EXSY1(I+1,2,K)-2.*EXSY1(I,2,K)+EXSY1(I-1,2,K))...
                +CYFZD*(EXSY1(I,1,K+1)-2.*EXSY1(I,1,K)+EXSY1(I,1,K-1)...
                +EXSY1(I,2,K+1)-2.*EXSY1(I,2,K)+EXSY1(I,2,K-1));
         EXS(I,NY,K)=-EXSY2(I,3,K)+CYD*(EXS(I,NY1,K)+EXSY2(I,4,K))...
             +CYY*(EXSY1(I,4,K)+EXSY1(I,3,K))...
             +CYFXD*(EXSY1(I+1,4,K)-2.*EXSY1(I,4,K)+EXSY1(I-1,4,K)...
             +EXSY1(I+1,3,K)-2.*EXSY1(I,3,K)+EXSY1(I-1,3,K))...
             +CYFZD*(EXSY1(I,4,K+1)-2.*EXSY1(I,4,K)+EXSY1(I,4,K-1)...
             +EXSY1(I,3,K+1)-2.*EXSY1(I,3,K)+EXSY1(I,3,K-1));
end
end

%       HESAPLANAN DEĞERLERİN KAYDEDİLMESİ

for K=2:NZ1,
for I=1:NX1,
            EXSY2(I,1,K)=EXSY1(I,1,K);
            EXSY2(I,2,K)=EXSY1(I,2,K);
            EXSY2(I,3,K)=EXSY1(I,3,K);
            EXSY2(I,4,K)=EXSY1(I,4,K);
            EXSY1(I,1,K)=EXS(I,1,K);
            EXSY1(I,2,K)=EXS(I,2,K);
            EXSY1(I,3,K)=EXS(I,NY1,K);
            EXSY1(I,4,K)=EXS(I,NY,K);
end
end

%       XZ İÇİN BİRİNCİ SEVİYE ORBC UYGULANMASI

for J=2:NY1,
        I=1;
        EXS(I,J,1)=EXSZ1(I,J,2)+CZD*(EXS(I,J,2)-EXSZ1(I,J,1));
        EXS(I,J,NZ)=EXSZ1(I,J,3)+CZD*(EXS(I,J,NZ1)-EXSZ1(I,J,4));
        I=NX1;
        EXS(I,J,1)=EXSZ1(I,J,2)+CZD*(EXS(I,J,2)-EXSZ1(I,J,1));
        EXS(I,J,NZ)=EXSZ1(I,J,3)+CZD*(EXS(I,J,NZ1)-EXSZ1(I,J,4));
end
for I=2:NX1-1,
        J=2;
        EXS(I,J,1)=EXSZ1(I,J,2)+CZD*(EXS(I,J,2)-EXSZ1(I,J,1));
        EXS(I,J,NZ)=EXSZ1(I,J,3)+CZD*(EXS(I,J,NZ1)-EXSZ1(I,J,4));
        J=NY1;
        EXS(I,J,1)=EXSZ1(I,J,2)+CZD*(EXS(I,J,2)-EXSZ1(I,J,1));
        EXS(I,J,NZ)=EXSZ1(I,J,3)+CZD*(EXS(I,J,NZ1)-EXSZ1(I,J,4));
end

%       KALAN YÜZEYLER İÇİN İKİNCİ SEVİYE ORBC UYGULANMASI

for J=3:NY1-1,
for I=2:NX1-1,
            EXS(I,J,1)=-EXSZ2(I,J,2)+CZD*(EXS(I,J,2)+EXSZ2(I,J,1))...
                +CZZ*(EXSZ1(I,J,1)+EXSZ1(I,J,2))...
                +CZFXD*(EXSZ1(I+1,J,1)-2.*EXSZ1(I,J,1)+EXSZ1(I-1,J,1)...
                +EXSZ1(I+1,J,2)-2.*EXSZ1(I,J,2)+EXSZ1(I-1,J,2))...
                +CZFYD*(EXSZ1(I,J+1,1)-2.*EXSZ1(I,J,1)+EXSZ1(I,J-1,1)...
                +EXSZ1(I,J+1,2)-2.*EXSZ1(I,J,2)+EXSZ1(I,J-1,2));
         EXS(I,J,NZ)=-EXSZ2(I,J,3)+CZD*(EXS(I,J,NZ1)+EXSZ2(I,J,4))...
             +CZZ*(EXSZ1(I,J,4)+EXSZ1(I,J,3))...
             +CZFXD*(EXSZ1(I+1,J,4)-2.*EXSZ1(I,J,4)+EXSZ1(I-1,J,4)...
             +EXSZ1(I+1,J,3)-2.*EXSZ1(I,J,3)+EXSZ1(I-1,J,3))...
             +CZFYD*(EXSZ1(I,J+1,4)-2.*EXSZ1(I,J,4)+EXSZ1(I,J-1,4)...
             +EXSZ1(I,J+1,3)-2.*EXSZ1(I,J,3)+EXSZ1(I,J-1,3));
end
end

%       HESAPLANAN DEĞERLERİN KAYDEDİLMESİ
```

```
for J=2:NY1,
for I=1:NX1,
            EXSZ2(I,J,1)=EXSZ1(I,J,1);
            EXSZ2(I,J,2)=EXSZ1(I,J,2);
            EXSZ2(I,J,3)=EXSZ1(I,J,3);
            EXSZ2(I,J,4)=EXSZ1(I,J,4);
            EXSZ1(I,J,1)=EXS(I,J,1);
            EXSZ1(I,J,2)=EXS(I,J,2);
            EXSZ1(I,J,3)=EXS(I,J,NZ1);
            EXSZ1(I,J,4)=EXS(I,J,NZ);
end
end

%       YZ İÇİN BİRİNCİ SEVİYE ORBC UYGULANMASI

for J=1:NY1,
        I=2;
        EYS(I,J,1)=EYSZ1(I,J,2)+CZD*(EYS(I,J,2)-EYSZ1(I,J,1));
        EYS(I,J,NZ)=EYSZ1(I,J,3)+CZD*(EYS(I,J,NZ1)-EYSZ1(I,J,4));
        I=NX1;
        EYS(I,J,1)=EYSZ1(I,J,2)+CZD*(EYS(I,J,2)-EYSZ1(I,J,1));
        EYS(I,J,NZ)=EYSZ1(I,J,3)+CZD*(EYS(I,J,NZ1)-EYSZ1(I,J,4));
end
for I=3:NX1-1,
        J=1;
        EYS(I,J,1)=EYSZ1(I,J,2)+CZD*(EYS(I,J,2)-EYSZ1(I,J,1));
        EYS(I,J,NZ)=EYSZ1(I,J,3)+CZD*(EYS(I,J,NZ1)-EYSZ1(I,J,4));
        J=NY1;
        EYS(I,J,1)=EYSZ1(I,J,2)+CZD*(EYS(I,J,2)-EYSZ1(I,J,1));
        EYS(I,J,NZ)=EYSZ1(I,J,3)+CZD*(EYS(I,J,NZ1)-EYSZ1(I,J,4));
end

%      KALAN YÜZEYLER İÇİN İKİNCİ SEVİYE ORBC UYGULANMASI

for J=2:NY1-1,
for I=3:NX1-1,
            EYS(I,J,1)=-EYSZ2(I,J,2)+CZD*(EYS(I,J,2)+EYSZ2(I,J,1))...
                +CZZ*(EYSZ1(I,J,1)+EYSZ1(I,J,2))...
                +CZFXD*(EYSZ1(I+1,J,1)-2.*EYSZ1(I,J,1)+EYSZ1(I-1,J,1)...
                +EYSZ1(I+1,J,2)-2.*EYSZ1(I,J,2)+EYSZ1(I-1,J,2))...
                +CZFYD*(EYSZ1(I,J+1,1)-2.*EYSZ1(I,J,1)+EYSZ1(I,J-1,1)...
                +EYSZ1(I,J+1,2)-2.*EYSZ1(I,J,2)+EYSZ1(I,J-1,2));
          EYS(I,J,NZ)=-EYSZ2(I,J,3)+CZD*(EYS(I,J,NZ1)+EYSZ2(I,J,4))...
              +CZZ*(EYSZ1(I,J,4)+EYSZ1(I,J,3))...
              +CZFXD*(EYSZ1(I+1,J,4)-2.*EYSZ1(I,J,4)+EYSZ1(I-1,J,4)...
              +EYSZ1(I+1,J,3)-2.*EYSZ1(I,J,3)+EYSZ1(I-1,J,3))...
              +CZFYD*(EYSZ1(I,J+1,4)-2.*EYSZ1(I,J,4)+EYSZ1(I,J-1,4)...
              +EYSZ1(I,J+1,3)-2.*EYSZ1(I,J,3)+EYSZ1(I,J-1,3));
end
end

%       HESAPLANAN DEĞERLERİN KAYDEDİLMESİ

for J=1:NY1,
for I=2:NX1,
            EYSZ2(I,J,1)=EYSZ1(I,J,1);
            EYSZ2(I,J,2)=EYSZ1(I,J,2);
            EYSZ2(I,J,3)=EYSZ1(I,J,3);
            EYSZ2(I,J,4)=EYSZ1(I,J,4);
            EYSZ1(I,J,1)=EYS(I,J,1);
            EYSZ1(I,J,2)=EYS(I,J,2);
            EYSZ1(I,J,3)=EYS(I,J,NZ1);
            EYSZ1(I,J,4)=EYS(I,J,NZ);
end
end

    T=T+DT/2;

%      HX DAĞILAN ALANININ HESAPLANMASI

for K=1:NZ1,
for J=1:NY1,
for I=2:NX1,
                HXS(I,J,K)=HXS(I,J,K)-(EZS(I,J+1,K)-EZS(I,J,K))*DTMDY...
                    +(EYS(I,J,K+1)-EYS(I,J,K))*DTMDZ;
end
end
```

```
end

%       HY DAĞILAN ALANININ HESAPLANMASI

for K=1:NZ1,
for J=2:NY1,
for I=1:NX1,
                HYS(I,J,K)=HYS(I,J,K)-(EXS(I,J,K+1)-EXS(I,J,K))*DTMDZ...
                    +(EZS(I+1,J,K)-EZS(I,J,K))*DTMDX;
end
end
end

%       HZ DAĞILAN ALANININ HESAPLANMASI


for K=2:NZ1,
for J=1:NY1,
for I=1:NX1,
                HZS(I,J,K)=HZS(I,J,K)-(EYS(I+1,J,K)-EYS(I,J,K))*DTMDX...
                    +(EXS(I,J+1,K)-EXS(I,J,K))*DTMDY;
end
end
end

    T=T+DT/2;

%       VERİLERİN KAYDEDİLMESİ

if (N ~= 1)

%       ÖLÇME YAPILACAK HÜCRENİN BELİRLENMESİ

for NPT=1:NTEST,
            I=IOBS(NPT);
            J=JOBS(NPT);
            K=KOBS(NPT);

%       DAĞILAN ALANLAR

if (NTYPE(NPT) == 1)
                STORE(NPT)=EXS(I,J,K);
end
if (NTYPE(NPT) == 2)
                STORE(NPT)=EYS(I,J,K);
end
if (NTYPE(NPT) == 3)
                STORE(NPT)=EZS(I,J,K);
end
if (NTYPE(NPT) == 4)
                STORE(NPT)=HXS(I,J,K);
end
if (NTYPE(NPT) == 5)
                STORE(NPT)=HYS(I,J,K);
end
if (NTYPE(NPT) == 6)
                STORE(NPT)=HZS(I,J,K);
end

%       AKIM DÖNGÜLERİ
%       X YÖNLÜ AKIM

if (NTYPE(NPT) == 7)
                STORE(NPT)=0;
for KK=K:K+1,
for JJ=J:J+1,
                        STORE(NPT)=STORE(NPT)+(-HYS(I,JJ,KK)+...
                            HYS(I,JJ,KK-1))*DELY+(HZS(I,JJ,KK)-...
                            HZS(I,JJ-1,KK))*DELZ;
end
end
end

%       Y YÖNLÜ AKIM

if (NTYPE(NPT) == 8)
                STORE(NPT)=0;
```

```
for KK=K:K+1,
for II=I:I+1,
                        STORE(NPT)=STORE(NPT)+(-HZS(II,J,KK)+...
                            HZS(II-1,J,KK))*DELZ+(HXS(II,J,KK)-...
                            HXS(II,J,KK-1))*DELX;
end
end
end

%       Z YÖNLÜ AKIM

if (NTYPE(NPT) == 9)
                    STORE(NPT)=0;
for JJ=J:J+1,
for II=I:I+1,
                        STORE(NPT)=STORE(NPT)+(-HXS(II,JJ,K)+...
                            HXS(II,JJ-1,K))*DELX+(HYS(II,JJ,K)-...
                            HYS(II-1,JJ,K))*DELY;
end
end
end
end
else

%   ÖLÇÜLECEK DEĞERLERİN SEÇİLMESİ
%           1 = EXS (ELEKTRİK ALANIN DAĞILAN X KOMPONENTİ)
%           2 = EYS (ELEKTRİK ALANIN DAĞILAN Y KOMPONENTİ)
%           3 = EZS (ELEKTRİK ALANIN DAĞILAN Z KOMPONENTİ)
%           4 = HXS (MANYETİK ALANIN DAĞILAN X KOMPONENTİ)
%           5 = HYS (MANYETİK ALANIN DAĞILAN Y KOMPONENTİ)
%           6 = HZS (MANYETİK ALANIN DAĞILAN Z KOMPONENTİ)
%           7 = IX (H NIN DİKDORTGEN DÖNGÜSÜNDEN GEÇEN AKIMIN X KOMPONENTİ)
%           8 = IY (H NIN DİKDORTGEN DÖNGÜSÜNDEN GEÇEN AKIMIN Y KOMPONENTİ)
%           9 = IZ (H NIN DİKDORTGEN DÖNGÜSÜNDEN GEÇEN AKIMIN Z KOMPONENTİ)

        NTYPE(1)=1;
        NTYPE(2)=1;
        NTYPE(3)=5;
        NTYPE(4)=5;

%   TEST YAPILACAK HÜCRELERİNİN KOORDİNATLARININ VERİLMESİ

        IOBS(1)=17;
        JOBS(1)=18;
        KOBS(1)=25;
        IOBS(2)=17;
        JOBS(2)=18;
        KOBS(2)=18;
        IOBS(3)=17;
        JOBS(3)=18;
        KOBS(3)=24;
        IOBS(4)=17;
        JOBS(4)=18;
        KOBS(4)=17;

        NPTS=NTEST;

%       ÇIKIŞ DOSYASININ GİRİŞ SATIRININ YAZILMASI

        fprintf (F10,'%d %d %d %d %d %d %d',DELX,DELY,DELZ,DT,NSTOP,NPTS);
        fprintf (F10,'\n');
        fprintf (F17,'HESAPLAMALAR YAPILDI VE DOSYAYA %d LOKASYONLARINDA
KAYDEDİLDİ\n',NPTS);

%       NTYPE VERİ HATASININ KONTROLÜ

for NPT=1:NPTS,
fprintf (F17,'%d %d %d %d %d %d\n',NPT,NTYPE(NPT),IOBS(NPT),JOBS(NPT),KOBS(NPT));
if (IOBS(NPT) >= NX)
                    fprintf (F17,...
'IOBS, JOBS VEYA KOBS DEĞERİNDE HATA VAR %d ÇALIŞMA DURDURULDU\n',NPT);
end
if (IOBS(NPT) <= 1)
                    fprintf (F17,...
'IOBS, JOBS VEYA KOBS DEĞERİNDE HATA VAR %d ÇALIŞMA DURDURULDU\n',NPT);
end
if (JOBS(NPT) >= NY)
```

```
                fprintf (F17,...
'IOBS, JOBS VEYA KOBS DEĞERİNDE HATA VAR %d ÇALIŞMA DURDURULDU\n',NPT);
end
if (JOBS(NPT) <= 1)
                fprintf (F17,...
'IOBS, JOBS VEYA KOBS DEĞERİNDE HATA VAR %d ÇALIŞMA DURDURULDU\n',NPT);
end
if (KOBS(NPT) >= NZ)
                fprintf (F17,...
'IOBS, JOBS VEYA KOBS DEĞERİNDE HATA VAR %d ÇALIŞMA DURDURULDU\n',NPT);
end
if (KOBS(NPT) <= 1)
                fprintf (F17,...
'IOBS, JOBS VEYA KOBS DEĞERİNDE HATA VAR %d ÇALIŞMA DURDURULDU\n',NPT);
end
end
for NPT=1:NTEST,
if ((NTYPE(NPT) >= 10) || (NTYPE(NPTS) <= 0))
fprintf (F17,ÖRNEKLEME NOKTASI İÇİN NTYPE HATASI %d ÇALIŞMA DURDURULDU\n',NPT);
end
end


end
for II=1:NPTS,

%   VERİLERİN ÇIKIŞ DOSYASINA YAZILMASI

             fprintf (F10,'%d ',STORE(II));
end
    fprintf (F10,'\n');
end
T=NSTOP*DT;

fprintf (F17,'ÇIKIŞ ZAMANI %14.7f SANİYE VE ZAMAN ADIMI SAYISI %d'DİR.',T,NSTOP);

fclose(F10);
fclose(F17);

end
```

## EK-2: FDTD MATLAB GUI Programı

```
function varargout = fdtd-gui(varargin)
% Tüm fonksiyonlar bulunmamaktadır. Kritik öneme sahip tekrar         etmeyen
fonksiyonlar örnek olarak ifade edilmiştir.
% fdtd-gui M-file for fdtd-gui.fig
%      fdtd-gui, by itself, creates a new fdtd-gui or raises the existing
%      singleton*.
%
%      H = fdtd-gui returns the handle to a new fdtd-gui or the handle to
%      the existing singleton*.
%
%      fdtd-gui('CALLBACK',hObject,eventVeri,handles,...) calls the local
%      function named CALLBACK in fdtd-gui.M with the given input arguments.
%
%      fdtd-gui('Property','Value',...) creates a new fdtd-gui or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before fdtd-gui_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to fdtd-gui_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIVERI, GUIHANDLES

% Edit the above text to modify the response to help fdtd-gui

% Last Modified by GUIDE v2.5 02-Jun-2010 12:20:32

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
'gui_Singleton',  gui_Singleton, ...
'gui_OpeningFcn', @fdtd-gui_OpeningFcn, ...
'gui_OutputFcn',  @fdtd-gui_OutputFcn, ...
'gui_LayoutFcn',  [] , ...
'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before fdtd-gui is made visible.
function fdtd-gui_OpeningFcn(hObject, eventveri, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventveri  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user veri (see GUIVERI)
% varargin   command line arguments to fdtd-gui (see VARARGIN)
global runstate
runstate=0;

% Choose default command line output for fdtd-gui
handles.output = hObject;

% Update handles structure
guiveri(hObject, handles);

% UIWAIT makes fdtd-gui wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = fdtd-gui_OutputFcn(hObject, eventveri, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventveri  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user veri (see GUIVERI)

% Get default command line output from handles structure
```

```
varargout{1} = handles.output;

% --- Executes on button press in buttonRUN.
function buttonRUN_Callback(hObject, eventveri, handles)
% hObject    handle to buttonRUN (see GCBO)
% eventveri  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user veri (see GUIVERI)
global runstate
NSTOP=str2double(get(handles.txtNSTOP,'String'));
THINC=str2double(get(handles.txtTHINC,'String'));
PHINC=str2double(get(handles.txtAMP,'String'));
AMP=str2double(get(handles.txtAMP,'String'));
BETA=str2double(get(handles.txtBETA,'String'));
% set(handles.textEXS,'String','RESULT');
% v2=str2double(get(handles.txtY,'String'));
% z=v1^3+v2^2;
fdtda(NSTOP,THINC,PHINC,AMP,BETA);
runstate=1;

% --- Executes on button press in pushDraw.
function pushDraw_Callback(hObject, eventveri, handles)
% hObject    handle to pushDraw (see GCBO)
% eventveri  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user veri (see GUIVERI)
global runstate
if (runstate~=0)
    load('NZOUT3D.txt');
    x=NZOUT3D(:,1);
    plot(x);
    runstate = 1;
end
```

## Ek-3: FDTD MATLAB Film Programı

```
%*********************************************************************%     FİLM GİRİŞ
%*********************************************************************

tview(:,:)=EXS(:,:,5);
sview(:,:)=EXS(:,13,:);
set(gcf, 'position', [400 250 800 600]);

subplot('position',[0.15 0.55 0.7 0.37]),pcolor(tview');
shading flat;
caxis([-100 100]);
colorbar;
axis image;
title(('EXS,N=0'));
xlabel('i');
ylabel('j');

subplot('position',[0.15 0.08 0.7 0.37]),pcolor(sview');
shading flat;
caxis([-100 100]);
colorbar;
axis image;
title(('EXS, N=0'));
xlabel('i');
ylabel('k');

rect=get(gcf,'Position');
rect(1:2)=[0 0];

M=filmin(NSTOP/2,gcf,rect);

% ÇERÇEVE'LERIN MATRIX'E KAYDEDİLMESİ

timestep=int2str(N);
tview(:,:)=EXS(:,:,25);
sview(:,:)=EXS(:,18,:);

subplot('position',[0.15 0.55 0.7 0.37]),pcolor(tview');
shading flat;
caxis([-100 100]);
colorbar;
axis image;
title(['EXS, N=',timestep]);
xlabel('i');
ylabel('j');

subplot('position',[0.15 0.08 0.7 0.37]),pcolor(sview');
shading flat;
caxis([-100 100]);
colorbar;
axis image;
title(['EXS, N= ',timestep]);
xlabel('i');
ylabel('k');

nn=N;
M(:,nn)=getçerçeve(gcf,rect);
```

# ÖZGEÇMİŞ
# HASAN ÖVÜÇ

20 Nisan 1977 tarihinde İzmit'de doğdu. İlk, orta ve liseyi İzmit ili Gölcük ilçesinde tamamladı. Boğaziçi Üniversitesi Meslek Yüksek Okulundan 2000 yılında mezun oldu. Anadolu Üniversitesi Açık Öğretim Fakültesi İşletme bölümünden 2006 yılında mezun oldu. 2007 yılında Beykent Üniversitesi Fen Bilimleri Enstitüsü Bilgi Teknolojileri bölümünde yüksek lisans eğitimine başladı. 2010 yılı Şubat ayında Ankara ilinde askerlik görevini tamamladı. 1998-2002 yılları arasında takım yöneticisi olarak Superonline firmasında çalıştı. 2002-2005 yılları arasında sistem ve ağ yöneticisi olarak Tradesoft firmasında çalıştı. 2005-2009 yılları arasında Tellcom firmasında sistem yöneticisi olarak görev aldı.

Yabancı dili İngilizce olup, 2001 yılından beri evlidir.