

**İSTANBUL TEKNİK ÜNİVERSİTESİ**  
**ELEKTRİK-ELEKTRONİK FAKÜLTESİ**

**ZAMANDA TERSİNE ÇEVİRME ALGORİTMALARI İLE  
KANSER TESPİTİ**

**LİSANS BİTİRME TASARIM PROJESİ**

**Cihan ÜRTEKİN**

**Alper Said SOYLU**

**ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ BÖLÜMÜ**

**MAYIS, 2019**



**İSTANBUL TEKNİK ÜNİVERSİTESİ**  
**ELEKTRİK-ELEKTRONİK FAKÜLTESİ**

**ZAMANDA TERSİNE ÇEVİRME ALGORİTMALARI İLE  
KANSER TESPİTİ**

**LİSANS BİTİRME TASARIM PROJESİ**

**Cihan ÜRTEKİN**  
**(040150601)**

**Alper Said SOYLU**  
**(040150728)**

**Proje Danışmanı: Prof. Dr. Funda AKLEMAN YAPAR**

**ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ BÖLÜMÜ**

**MAYIS, 2019**



İTÜ, Elektronik ve Haberleşme Mühendisliği Bölümü'nün ilgili Bitirme Tasarım Projesi yönergesine uygun olarak tamamen kendi çalışmamız sonucu hazırladığımız “ZAMANDA TERSİNE ÇEVİRME ALGORİTMALARI İLE KANSER TESPİTİ” başlıklı Bitirme Tasarım Projesi’ni sunmaktayız. Bu çalışmayı intihal olmaksızın hazırladığımızı taahhüt eder; intihal olması durumunda bitirme tasarım projesinin başarısız sayılacağını kabul ederiz.

**Cihan ÜRTEKİN**  
(040150601)

**Alper Said SOYLU**  
(040150728)

**Proje Danışmanı : Prof.Dr. Funda AKLEMAN YAPAR .....**



## **ÖNSÖZ**

Tez çalışmamızda her konuda bize destek olan hocamız sayın Prof. Dr. Funda AKLEMAN YAPAR'a çok teşekkür ederiz.

Mayıs 2019

Cihan ÜRTEKİN  
Alper Said SOYLU



## İÇİNDEKİLER

	<u>Sayfa</u>
ÖNSÖZ .....	vii
İÇİNDEKİLER .....	ix
KISALTMALAR .....	xi
SEMBOLLER .....	xiii
ÇİZELGE LİSTESİ.....	xv
ŞEKLİ LİSTESİ.....	xvii
ÖZET .....	xix
SUMMARY .....	xxi
<b>1. GİRİŞ .....</b>	<b>1</b>
1.1 Projenin Amacı.....	1
1.2 Literatür Araştırması .....	1
<b>2. SONLU FARKLARLA YAKLAŞIK TÜREV HESABI .....</b>	<b>3</b>
2.1 Giriş.....	3
2.2 İleri Fark Yaklaşımı .....	3
2.3 Geri Fark Yaklaşımı .....	4
2.4 Merkezi Fark Yaklaşımı.....	5
<b>3. FDTD METODU.....</b>	<b>7</b>
3.1 Giriş.....	7
3.2 Yee Algoritması .....	8
<b>4. TR ALGORİTMASI.....</b>	<b>11</b>
4.1 Giriş .....	11
4.2 TR Denklemlerinin 2D FDTD Denklemlerine Uygulanması .....	13
<b>5. TR ALGORİTMASININ NUMERİK UYGULAMALARI.....</b>	<b>15</b>
5.1 Giriş .....	15
5.2 TR Algoritmasının Adımları .....	16
5.3 Minimum Entropi Koşulu .....	17
5.4 Çözünürlük .....	18
5.5 Yapılan Çalışma .....	18
<b>6. KODLARIN GPU ÜZERİNDE (PARALEL) ÇALIŞTIRILMASI .....</b>	<b>31</b>
6.1 Giriş .....	31
6.2 CUDA Kütüphanesinin Kurulması .....	31
6.3 Matlab Ortamında Gerekli Eklentinin Kurulması.....	32
6.4 Kodların Düzenlenmesi .....	32
6.5 Kodların GPU Üzerinde Çalıştırılmasının Etkisi .....	33
<b>7. GERÇEKÇİ KISITLAR, SONUÇLAR VE ÖNERİLER .....</b>	<b>35</b>
7.1 Çalışmanın Uygulama Alanı .....	35
7.2 Gerçekçi Tasarım Kısıtları .....	35
7.2.1 Maliyet .....	35
7.2.2 Standartlar .....	35

7.2.3 Sosyal, çevresel ve ekonomik etki .....	35
7.2.4 Sağlık ve güvenlik riskleri.....	35
7.3 Sonuçlar.....	36
7.4 Geleceğe Yönelik Öneriler .....	36
<b>KAYNAKLAR.....</b>	<b>37</b>
<b>EKLER .....</b>	<b>39</b>
<b>ÖZGEÇMİŞ .....</b>	<b>59</b>

## **KISALTMALAR**

**FDTD** : Finite Difference Time Domain

**TR** : Time Reversal

**GHz** : Gigahertz



## **SEMBOLLER**

<b>p</b>	: Elektriksel yük yoğunluğu
<b>f</b>	: Kuvvet
<b>g</b>	: Hız
<b>t</b>	: Zaman
<b>S</b>	: Poynting vektörü



## **ÇİZELGE LİSTESİ**

	<b><u>Sayfa</u></b>
Çizelge 5.1 : Ortam parametreleri. ....	28
Çizelge 5.2 : Kaynak özellikleri. ....	28



## **ŞEKİL LİSTESİ**

	<b>Sayfa</b>
<b>Şekil 2.1</b> : Taylor seri açılımı .....	<b>3</b>
<b>Şekil 2.2</b> : İleri fark türev hesabı .....	<b>3</b>
<b>Şekil 2.3</b> : Geri fark türev hesabı .....	<b>4</b>
<b>Şekil 2.4</b> : Merkezi fark yaklaşımı .....	<b>5</b>
<b>Şekil 3.1</b> : Maxwell denklemleri .....	<b>6</b>
<b>Şekil 3.2</b> : FDTD algoritması için kullanılan Yee Hücresi .....	<b>7</b>
<b>Şekil 4.1</b> : FDTD-TR denklemleri .....	<b>11</b>
<b>Şekil 4.2</b> : 2 GHz merkez frekanslı örnek modüle Gauss Dalgası .....	<b>12</b>
<b>Şekil 5.1</b> : TR algoritması blok diyagramı .....	<b>14</b>
<b>Şekil 5.2</b> : Serbest uzayda dielektrik nesne 134. iterasyon .....	<b>18</b>
<b>Şekil 5.3</b> : Serbest uzayda dielektrik nesne ters varimax normu .....	<b>19</b>
<b>Şekil 5.4</b> : 28x28 düzlem array ile yapılan gözlem 135. iterasyon .....	<b>20</b>
<b>Şekil 5.5</b> : Düzlem array gözlem ters varimax normu .....	<b>21</b>
<b>Şekil 5.6</b> : Serbest uzayda iletken nesne 12. iterasyon .....	<b>22</b>
<b>Şekil 5.7</b> : Serbest uzayda iletken nesne ters varimax normu .....	<b>23</b>
<b>Şekil 5.8</b> : İki nesne ve iki kaynak ile gözlenen 115. iterasyona ait görüntü .....	<b>24</b>
<b>Şekil 5.9</b> : İki nesne gözlemine ait ters varimax normu .....	<b>25</b>
<b>Şekil 5.10</b> : Malzemeli ortamda iletken nesne gözlemi 100. iterasyon .....	<b>26</b>
<b>Şekil 5.11</b> : Malzemeli ortamda iletken nesne gözlemine ait ters varimax normu .....	<b>27</b>
<b>Şekil 6.1</b> : İşletim sistemine uygun CUDA kurulumu .....	<b>28</b>
<b>Şekil 6.2</b> : Matlab ortamında gerekli eklentinin kurulması .....	<b>29</b>
<b>Şekil 6.3</b> : Sırasıyla CPU ve GPU üzerinde çalışan ‘zeros’ fonksiyonu .....	<b>29</b>
<b>Şekil 6.4</b> : Normal dizinin GPU dizisine dönüştürülmesi .....	<b>30</b>
<b>Şekil A.1</b> : Algoritmanın 2 boyutlu olarak Matlab ortamında gerçekleşmesi .....	<b>35</b>
<b>Şekil A.2</b> : Yazılım ortamında malzeme eklenmesi .....	<b>36</b>



## **ZAMANDA TERSİNE ÇEVİRME ALGORİTMALARI İLE KANSER TESPİTİ**

### **ÖZET**

Mikrodalga görüntüleme son zamanlarda popülerleşen aktif bir araştırma sahasıdır. Mikrodalga görüntüleme iyonize edici olmayan ışının kullanımından dolayı özellikle tıbbi görüntülemede kullanımı tercih edilebilir. Mikrodalga görüntüleme verisinin sayısal hesaplama yöntemleri ile çözümlenmesi ve işlenmesi gereklidir. FDTD basit ve kullanışlı bir sonlu elemanlar yöntemidir. Zamanda Tersine Çevirme Algoritması kullanarak mikrodalga gözlem verisi ayrik uzayda geri yayılım yaptırılmasıyla nesne tespiti yapılmaktadır. Bu çalışmada üç boyutlu FDTD algoritması Matlab ortamında gerçeklenerek elektromanyetik dalga simülasyonu elde edilmiş ve gözlem kaynaklarından ölçülen elektrik alan verileri tersine çevrilen zamanda geri propagasyon yaptırılarak nesne konumu tespiti üzerine çalışılmıştır.



## **CANCER DETECTION WITH TIME REVERSAL ALGORITHMS**

### **SUMMARY**

Microwave imaging is a recently active study object. Microwave imaging, due to usage of non-ionizing radiation, can be preferred especially for medical imaging. Microwave data should be analysed and processed with numerical computation methods. FDTD is a simple and useful finite elements method. Using Time Reversal Algorithm, object detection is performed by back-propagation microwave observation data in discrete space. In this study, electromagnetic wave simulation is obtained implementing three dimensional FDTD algorithm in Matlab environment and detection of object position is studied by back-propagating electrical field data measured on observation sources in reversed time.



## **1. GİRİŞ**

### **1.1 Projenin Amacı**

Mikrodalga ile kanser tespiti alanında birçok mevcut yöntem ve çalışma mevcuttur. Tümör oluşumunun erken evrede tespiti kanser tedavisi açısından oldukça önemli olmakla birlikte, iyonize edici radyasyon tabanlı yöntemlerin çok fazla kullanılması sağlık açısından zararlıdır, bu yüzden sık aralıklarla kullanılması doğru değildir. İyonize edici olmayan elektromanyetik dalga yayılımı prensibine dayanan yöntemlerin geliştirilmesi, bu sorun için etkili bir çözümüdür.

Bu projede, ölçülen cisimden geri saçılan elektromanyetik alanlar kullanılarak cismin (tümörün) yerinin tespit edilmesini sağlayan zamanda tersine çevirme yönteminin, yazılımsal olarak gerçekleştirmesi üzerine çalışılmaktadır. Gerçeklenen yazılımın İki boyutlu ve üç boyutlu benzetim ortamlarında, ortama çeşitli materyaller eklenerek uygulanması ve test edilmesi amaçlanmaktadır. Ayrıca yöntemin verimliliğini artırmak ve yüksek doğruluk sağlamak amacıyla, test esnasında gözlemlenen sonuçlara göre uygun görülen değişikliklerle algoritmanın geliştirilmesi hedeflenmektedir.

### **1.2 Literatür Araştırması**

Son 10 yılda yapılan araştırmalar, mikrodalga meme kanseri tespitinin konvansiyonel X-ışını mamografisine başarılı bir seçenek olma potansiyeline sahip olduğunu göstermiştir. Mikrodalga görüntüleme, normal ve zararlı dokuların dielektrik özellikleri arasındaki farklılıklarını kullanarak bir insan vücudundaki kanser tümörlerini tespit etmek için iyonize olmayan elektromanyetik radyasyon kullanır. Meme kanseri tespiti için aktif mikrodalga görüntüleme teknikleri geniş ölçüde iki farklı kategoride sınıflandırılabilir: (i) tomografik yöntemler; (ii) geri saçılma yöntemleri. İki yaklaşım arasındaki temel fark, tomografik yöntemlerin, meme içinden iletilen mikrodalga

radasyonu ölçümllerine dayanmasıdır; geri saçılma yöntemleri, memedeki tümör dokularının yansımalarını kullanır. [1]

Geri saçılma görüntülemede, birkaç mikrodalga kaynak, homojen olmayan ortama dalga gönderir ve bunun sonucunda oluşan yansımalar ortamın etrafına yerleştirilmiş olan alıcılara gelir. Alıcılara gelen bu yansıma bilgileri kaydedilir. Zararlı meme tümörleri, sağlıklı meme dokularından dikkate alınır ölçüde farklı elektriksel özelliklere sahiptir.

Prensip olarak, tümörlerin yerlerinin geri saçılmalardan tespit edilmesi mümkündür. Bazı araştırmalar, sentetik silindirik ve düzlemsel antenleri içeren sayısal simülasyonlar ile 1 cm'den küçük çaplı küçük tümörlerin saptanması ve lokalize edilmesinin uygulanabilirliğini göstermektedir [2]. Bazı araştırmacılar daha yüksek çözünürlük elde etmek ve gürültüyü bastırmak için ultra geniş bantlı konfokal mikrodalga görüntüleme (CMI) kullanmıştır [3]. Bununla birlikte, difraktif olmayan ve düz çizgilerle hareket eden X-işinlarından farklı olarak, meme dokularında elektromanyetik mikrodalga yayılımı, refraksiyon ve çok yollu etkiler ile karakterize edilir, geri saçılan kanser darbe sinyali, iki veya daha fazla yolla alıcıya ulaşır [4]. Sonuç olarak, standart sinyal işleme algoritmaları, çok yollu yayılma nedeniyle iyi performans göstermez ve yeterince iyi bir çözünürlüğe sahip olan kanser tümörlerini doğru bir şekilde tanımlayamaz veya tespit edemez. Projemizde, çok kanallı yayılımı meme kanseri tespiti için avantajına kullanan zaman tersine çevirme sinyal işlemeye dayanan farklı bir geri saçılma görüntüleme algoritmasını kullanacağız.

## 2. SONLU FARKLARLA YAKLAŞIK TÜREV HESABI

### 2.1 Giriş

$y = f(x)$  ile gösterilen bir fonksiyonun bazı noktalardaki değerleri bilinirse, bu değerlerden yararlanılarak bir noktadaki türevin hesaplanması mümkündür. Benzer bir şekilde,  $y = f(x, y)$  gibi birden fazla değişkeni olan fonksiyonların da Taylor seri açılımından yararlanılarak kısmi türevleri yaklaşık bulunabilir.

Türev tanımından yola çıkılarak ileri fark, geri fark ve merkezi fark yöntemleriyle numerik türev hesabı açıklanabilir.

### 2.2 İleri Fark Yaklaşımı

Taylor seri açılımı:

$$f(x_i - h) = f_{i-1} = f_i - hf'_i + \frac{h^2}{2}f''_i + \frac{h^3}{6}f'''_i + \dots \quad (2.1)$$

$$f'_i = \frac{f_i - f_{i-1}}{h} - \frac{1}{2}hf''_i - \frac{1}{6}h^2f'''_i + \dots$$

**Şekil 2.1:** Taylor Seri Açılımı

Birinci terimden sonraki terimler ihmal edildiğinde,

$$f'_i = \frac{f_{i+1} - f_i}{h} + O(h) \quad (2.2)$$

**Şekil 2.2:** İleri Fark Türev Hesabı

İleri fark türev hesabı ifade elde edilmiş olur.

### 2.3 Geri Fark Yaklaşımı

Taylor serisi açılımı:

$$f(x_i + h) = f_{i+1} = f_i + hf'_i + \frac{h^2}{2}f''_i + \frac{h^3}{6}f'''_i + \dots \quad (2.3)$$

$$f'_i = \frac{f_{i+1} - f_i}{h} - \frac{1}{2}hf''_i - \frac{1}{6}h^2f'''_i - \dots$$

Birinci terimden sonraki terimler ihmal edildiğinde,

$$f'_i = \frac{f_i - f_{i-1}}{h} + O(h) \quad (2.4)$$

**Şekil 2.3:** Geri Fark Türev Hesabı

Geri fark türev hesabı elde edilmiş olur.

## 2.4 Merkezi Fark Yaklaşımı

Taylor serisi açılımı:

$$f(x_i + h) = f_{i+1} = f_i + hf'_i + \frac{h^2}{2}f''_i + \frac{h^3}{6}f'''_i + \dots \quad (2.5)$$

$$f(x_i - h) = f_{i-1} = f_i - hf'_i + \frac{h^2}{2}f''_i - \frac{h^3}{6}f'''_i + \dots$$

Taraf tarafa çıkarma yapılarak,

$$f'_i = \frac{f_{i+1} - f_{i-1}}{2h} - O(h) + \dots \quad (2.6)$$

**Şekil 2.4:** Merkezi Fark Yaklaşımı

Merkezi fark yaklaşımı bulunur.



### 3. FDTD METODU

#### 3.1 Giriş

FDTD metodu elektromanyetik problemlerin çözümünde kullanılan, teorik ve pratik açıdan en basit yöntemdir [3]. Bu yöntem sayesinde elektromanyetik problemlerini doğrudan zaman domeninde analiz etmek mümkündür. FDTD, bir, iki ve üç boyutlu problemlere uyarlanabilir. FDTD yöntemi, Maxwell denklemlerindeki kısmi türev operatörlerinin merkezi farklılar dayalı sonlu karşılıkları ile değiştirilip doğrudan zaman ve konum domenlerinde numerikleştirilmesine dayanır [4].

Maxwell denklemleri elektrik ve manyetik alan vektörleri arasındaki uzamsal ve zamansal boytlardaki ilişkiyi ifade eder. Maxwell denklemleri ilgili hesaplamaların bilgisayarda gerçekleştirilebilmesi için numerik forma dönüştürülür [15].

$$\nabla \times \vec{E} = \frac{-\partial \vec{B}}{\partial t} \quad (3.1)$$

$$\nabla \times \vec{H} - \sigma \vec{E} = \frac{\partial \vec{D}}{\partial t} \quad (3.2)$$

$$\nabla \cdot \vec{D} = \rho \quad (3.3)$$

$$\nabla \cdot \vec{B} = 0 \quad (3.4)$$

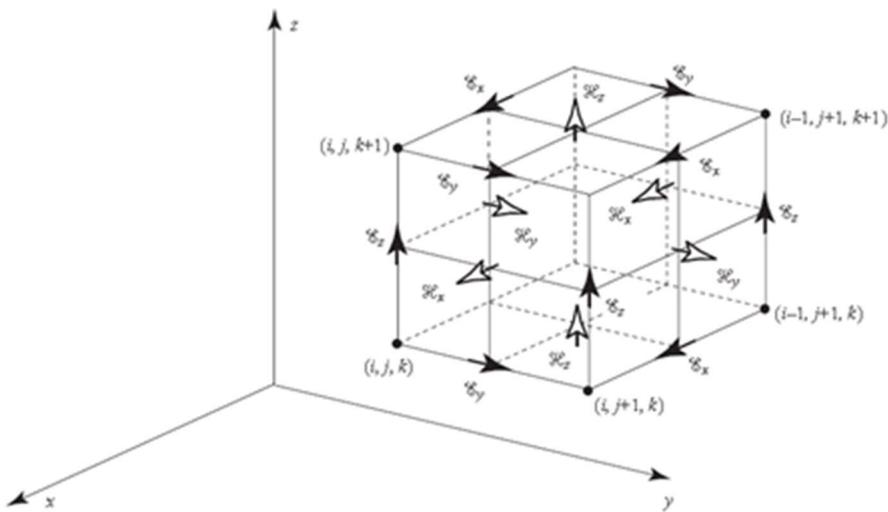
**Şekil 3.1:** Maxwell Denklemleri

### 3.2 Yee Algoritması

Elektromanyetik alanda FDTD hesabı ilk olarak 1996 yılında Kane Yee tarafından ikinci dereceden merkezi farklar yaklaşımı kullanılarak önerilen algoritma kullanılarak gerçekleştirilebilir [3]. Algoritmanın işleyışı söyle özetlenir:

1. Ampere ve Faraday denklemlerindeki tüm türevler sonlu farklar yaklaşımına göre değiştirilir. Zaman ve uzay ayriklaştırılır. Böylece elektrik ve manyetik alan hem zaman hem uzayda kesişmiş olur.
2. Elde edilen fark denklemleri iteratif forma dönüştürülerek güncelleme denklemleri elde edilir. Güncelleme denklemleri gelecek elektrik ve manyetik alan değerlerini önceki değerler cinsinden ifade eder.
3. Manyetik alan değerleri bir sonraki adım için hesaplanır.
4. Elektrik alan değerleri bir sonraki adım için hesaplanır.
5. Önceki iki adım istenilen adım sayısı kadar tekrarlanır.

Yee algoritmasında kullanılan üç boyuttaki ayrik elektrik ve manyetik alanları şekilde gösterilmiştir.



**Şekil 3.2:** FDTD algoritması için kullanılan Yee Hücresi

Gerçek bir problemde malzeme, her biri bir  $\mu$  ve  $\epsilon$  değerine sahip olan ve uygun bir şekilde boyutlandırılmış Yee Hücreleri'ne bölünerek kolayca analiz edilebilir. Burada  $\mu$ , birim hücredeki malzemenin manyetik geçirgenliğini,  $\epsilon$  ise malzemenin dielektrik sabitini gösterir. Alan elemanlarının hepsi için başlangıç değeri verilir. Daha sonra uygun bir cevap elde edilene kadar, alan denklemleri iteratif olarak hesaplanır.  $E$  ve  $H$  değerleri de güncelleştirilir. Ana döngü zaman düngüsüdür ve seçilen maksimum zaman adımı tamamlanıncaya kadar ana zaman döngüsü çalıştırılır. Eğer zaman  $\Delta t$  uzunlığında ayrık adımlara ayrılsa, şimdiki zamanda hesaplanan alan değerleri, önceki değerlere göre artar ve azalır [2].



## 4. TR ALGORİTMASI

### 4.1 Giriş

TR algoritması nedensel doğrultulu zamanda gözlenen sinyallerin ( $t' = -t$ ) dönüşümü yapılarak tersine çevrilen zamanda geri propagasyon yapılmasıyla kaynağa ulaşılmasını hedefler.

Fourier dönüşümü yapılmış  $E(t')$  denklemini ele alırsak:

$$E(t') := TE(t) := E(-t), t' := -t \quad (4.1)$$

$$E'(\omega) := TE(\omega) = \int_{-\infty}^{\infty} E(-t) e^{-i\omega t} dt \quad (4.2)$$

$$= \int_{-\infty}^{\infty} E(t) e^{-i\omega t} dt, \quad t \rightarrow -t \quad (4.3)$$

$$= E(-\omega) = E^*(\omega) \quad (4.4)$$

Zamanda çevrilmiş elektrik alan orijinalin komplex konjugesine eşittir. Bir düzlem dalga  $x$  doğrultusunda propagasyonlığında

$$E(x, \omega) = E(\omega) e^{-ikx}, \quad k = \omega \sqrt{\varepsilon \mu} \quad (4.5)$$

$\omega$  reel bir sayı ve zamanda çevrilmiş alan denklemi

$$E'(x, \omega) = E^*(\omega) e^{-ik^*x}, \quad k^* = \omega \sqrt{\varepsilon^* \mu^*} \quad (4.6)$$

(4.5) ve (4.6) denklemlerinde ve  $k$  ve  $k^*$  reel kısımlarının pozitif değerleri alındığında, ters doğrultuda sökümlenen dalga zamanda tersine çevrilerek genişleyen dalga olarak ortamda ilerler.

Zamanda tersine çevrilmiş dalgada  $\vartheta$  hızı ile ilerleyen elektronlar –  $\vartheta$  hızı ile ilerliyormuş gibi görünür. Aynı şekilde  $J_e$  elektrik akısı doğrultu değiştirmiştir. Elektriksel yükler ile aynı doğrultuda olan manyetik alan da doğrultusunu değiştirir. Zamanda çevrilmiş fiziksel parametreler  $\rho$  (elektriksel yük yoğunluğu),  $E$ ,  $J_m$ ,  $f$  (kuvvet) ve  $p$  (basınç) çift değerli olduğundan işaretlerini korurlar. Fakat  $\vartheta$  (hız),  $H$ ,  $J_e$ ,  $S$  (Poynting vektörü) tek değerli olduklarından zamanda tersine çevrildiklerinde işaretleri tersine çevrilir [12]. Sonuç olarak ( $t' = -t$ ) zamanda çevrilmiş parametreleri ifade ederse

$$\rho'(t') = \rho(-t) \quad (3.5) \quad E'(t') = E(-t) \quad (3.6) \quad J'_m(t') = J_m(-t) \quad (4.7)$$

$$\vartheta'(t') = -\vartheta(-t) \quad (3.8) \quad H'(t') = -H(-t) \quad (3.9) \quad J'_e(t') = -J_e(-t) \quad (4.8)$$

## 4.2 TR Denklemlerinin 2D FDTD Denklemlerine Uygulanması

Noktasal bir kaynak vakumda yayılım yaparsa ve zamanda tersine çevrilmiş FDTD denklemleri kullandığımı alanın bütün noktalarına uygularsak, dalga maksimum genliğe ulaştığı anda yakınsamaya başlar [13]. Bu amaçla (4.1) bölümündeki denklemleri FDTD denklemlerine uygulandığında zamanda tersine çevrilmiş FDTD-TR denklemleri elde edilir.

$$H_{x_{i+\frac{1}{2},j}}^{n+\frac{1}{2}} = H_{x_{i+\frac{1}{2},j}}^{n-\frac{1}{2}} + \frac{\Delta t}{\mu} \frac{E_{z_{i+\frac{1}{2},j+\frac{1}{2}}}^n - E_{z_{i+\frac{1}{2},j-\frac{1}{2}}}^n}{\Delta y} \quad (4.9)$$

$$H_{y_{i,j+\frac{1}{2}}}^{n+\frac{1}{2}} = H_{y_{i,j+\frac{1}{2}}}^{n-\frac{1}{2}} - \frac{\Delta t}{\mu} \frac{E_{z_{i+\frac{1}{2},j+\frac{1}{2}}}^n - E_{z_{i-\frac{1}{2},j+\frac{1}{2}}}^n}{\Delta x} \quad (4.10)$$

$$E_{z_{i+\frac{1}{2},j+\frac{1}{2}}}^{n+1} = \frac{1+\xi}{1-\xi} E_{z_{i+\frac{1}{2},j+\frac{1}{2}}}^n - \frac{1}{1-\xi} \frac{\Delta t}{\varepsilon} \frac{H_{y_{i+1,j+\frac{1}{2}}}^{n+\frac{1}{2}} - H_{y_{i,j+\frac{1}{2}}}^{n+\frac{1}{2}}}{\Delta x} \quad (4.11)$$

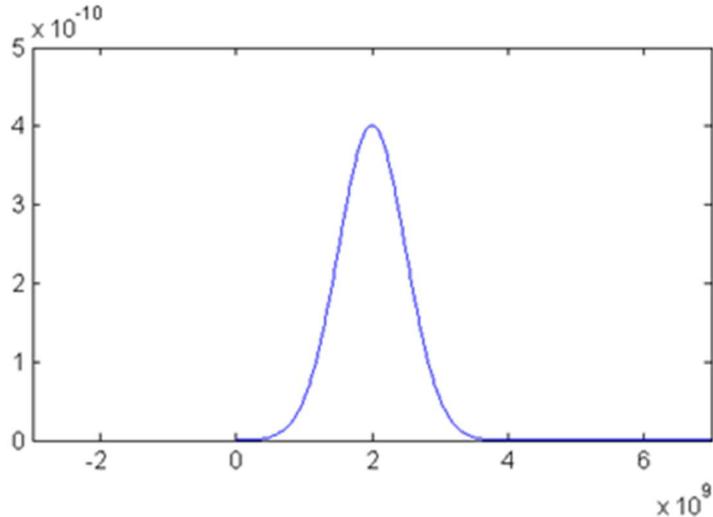
$$+ \frac{1}{1-\xi} \frac{\Delta t}{\varepsilon} \frac{H_{x_{i+\frac{1}{2},j+1}}^{n+\frac{1}{2}} - H_{x_{i+\frac{1}{2},j}}^{n+\frac{1}{2}}}{\Delta y}$$

$$\xi = \frac{\sigma \Delta t}{2\varepsilon} \quad (4.12)$$

**Şekil 4.1:** FDTD-TR Denklemleri

( $\Delta x = \Delta y = \Delta$ ) hücre boşluğunu ve  $\Delta t$  zamanda artışı göstermektedir.  $n' + \frac{1}{2}$  ifadesi geri yansımış dalga denkleminin bir sonraki adımı ile ilişkilidir.  $n'$  ve  $\Delta t$  FDTD algoritmasının iterasyon adımını simgeler ve bu nedenle pozitif değere sahiptirler. T dalganın geri yansıma başlangıç zamanı olarak kabul edilirse, TR algoritmasının zaman adımı  $t = T - n' \Delta t$  olarak bulunur.

TR algoritması çalışmalarının gösterimi için 2-GHz merkez frekanslı modüle Gauss dalgası kullanılmıştır.



**Sekil 4.2:** 2 GHz merkez frekanslı örnek modüle

Gauss Dalgası

4.9, 4.10, 4.11 ve 4.12 denklemleri iki boyutlu FDTD denklemleri üzerinden verilmiştir. Bu çalışmada ise üç boyutlu denklemler kullanılmış, çalışma üç boyutlu ortamda yapılmıştır. Bu da bilgisayar ortamında hafızaya ek yük bindirdiği için yapılan işlemlerin oldukça yavaşlamasına neden olmuştur. Bu yüzden ilerleyen süreçte deneyleri daha hızlı yapabilmek için kodları paralel işlemler olarak çalışıtmak üzerine araştırma yapılmıştır.

## **5. TR ALGORİTMASININ NUMERİK UYGULAMALARI**

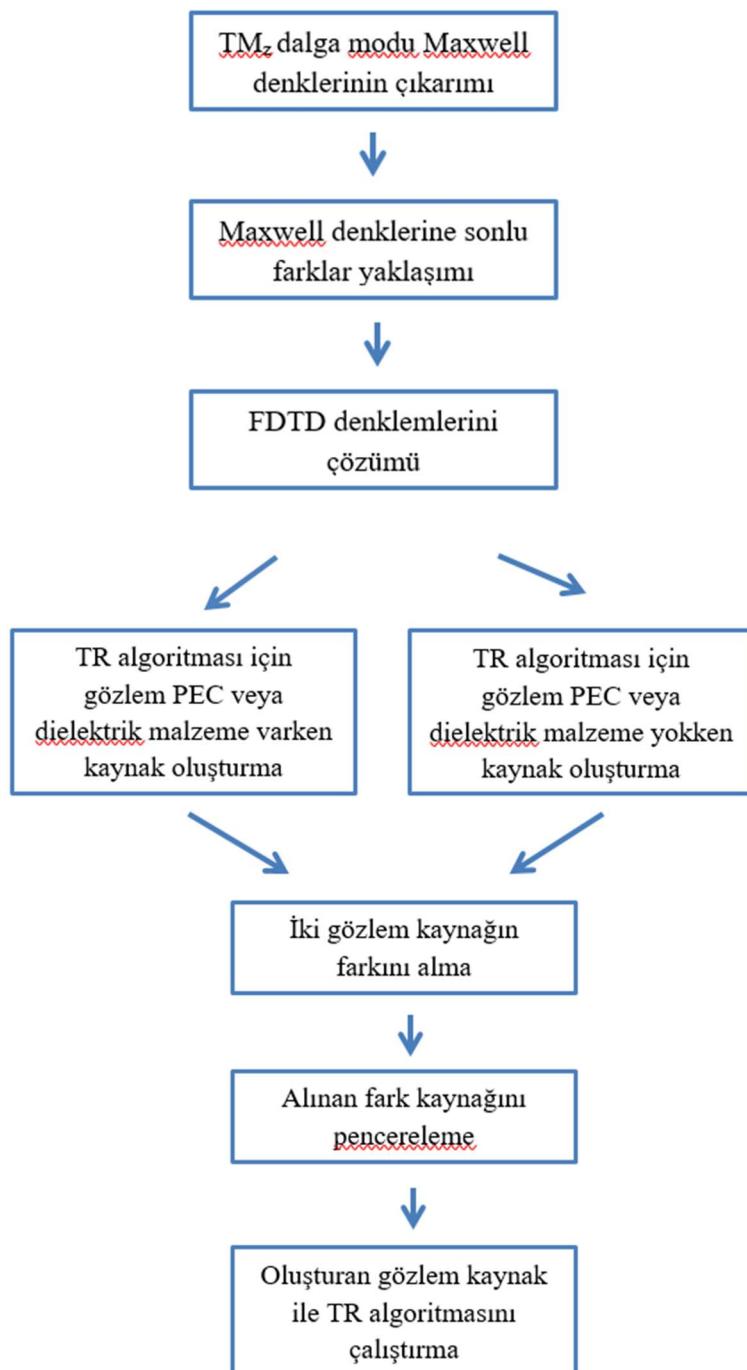
### **5.1 Giriş**

Ses dalgaları için kullanıldığından TR deneyleri algılanan işaretin zamanda tersine çeviren ve ortama geri propagasyon yapan dönüştürücülerin kullanılmasıyla uygulanır. Bu deneyler sonucunda homojen olmayan ve kayıplı ortamda zamanda tersine çevrilen dalganın hedef kaynağa doğru odaklandığı gözlemlenir [14]. FDTD yöntemi kullanılmasıyla TR algoritması doğrudan zaman domeninde uygulanabilmektedir [15].

Bu çalışmada TR algoritmasıyla ortam kayıplarının etkisinin kaldırıldığı görülmektedir. Bu sayede algoritma kayıplı ve kayıpsız ortamlarda kullanılabilir. Odaklama işlemi ortam parametreleri önceden bilinmeden de uygulanabilir [10].

## 5.2 TR Algoritmasının Adımları

FDTD analizi yapılan medyuma TR yöntemi tatbik edilerek dalga zamanda geri propagasyon yapılır. Zamanda tersine çevrilen işaret önce yakınsayarak hedef kaynağa odaklandıktan sonra tekrar ıraksamaya başlar.



**Şekil 5.1:** TR Algoritması Blok Diyagramı

Şekilde kullanılan TR algoritmasının aşamaları blog diyagramı olarak gösterilmektedir. TR algoritması ortamda nesne varken ve yokken çalıştırılıp iki kere gözlem kaynak kaydedilir. Bu işlemin amacı geri gönderilecek dalganın sadece faz farkı olmuş elektrik alanları içermesini sağlamaktır. Çünkü fokuslama işlemi faz farkı olan elektrik alan bileşenleri sayesinde olmaktadır.

### 5.3 Minimum Entropi Koşulu

Geri propagasyon dalga optimum yakınsamayı sağladıkten çok kısa bir süre sonra tekrar ilerlemeye devam eder. Optimum fokuslanma yakınsayan ve iraksayan dalganın arasındaki çok kısa bir süredir. İdeal durumda, ortam şartları bilinirken, optimum fokuslanma zamanda çevrilmiş dalganın maksimum yaptığı optimum zaman anında olduğundan uzaysal – zamansal uyumlu süzgeç ile optimum ana ulaşılır [14]. Fakat uygulamada sadece alıcı dizi kullanıldığında ve ortamin özellikleri bilinmediğinden bu yöntem yanlış sonuçlar verebilir.

Elektromanyetik dalga kaynağına tekrar geldiğinde hedef konumunda dik bir maksimum yapması diğer yerlerde küçük değerler olması beklenir. Yüksek genlikli resimler düşük minimuma sahip olacağından minimum entropi koşulu optimum zamanı bulmak için kullanılır [10]. Entropi hesabı için ters varimax norm hesaplanır [16]. Elektrik alan tüm noktalarda bulunur ve tüm iterasyon adımları gerçekleştirilirken FDTD TR model ile birlikte ters varimax normu da hesaplanır.

$$R(E_x^n) = \frac{\left[ \sum \sum \sum_{i,j,k} E_x^{n^2}(i,j,k) \right]^2}{\sum \sum \sum_{i,j,k} E_x^{n^4}(i,j,k)} \quad (5.1)$$

(i, j, k) birim hücre koordinatlarını, n FDTD TR algoritmasının zaman adımlarını ifade eder. Varimax normun minimum olduğu değer FDTD TR algoritmasının çıktısını belirlemektedir.

## 5.4 Çözünürlük

Elektromanyetik dalganın çözünürlüğü çıkış sinyalinin zaman genişliğine bağlıdır [10]. Nesnenin tespiti çıkış sinyalinin dalga boyu ile merkez frekansı ile orantılıdır. İyi bir çözünürlük yüksek frekanslarda ve geniş bantlarda sağlanmaktadır [10]. Bu durum donanımsal gerçekleme açısından kısıtlama getirmektedir. Bu nedenle yüksek çözünürlük elde etmek amacıyla verici sinyalinden bağımsız olarak, zamanda tersine çevrilmek için kaydedilmiş dalgayı merkez frekansı geri gönderilecek dalganın maksimumunda olan çok kısa bir Gauss dalgasıyla çarparak dalga pencerelenir.

$$g_i(t) = e^{((t-tp)/\tau)^2} \quad (5.2)$$

denklemi her alıcı için geri yansıma işleminin giriş sinyalini ifade etmektedir. İfadenin maksimum olduğu zamanı  $tp$  simgelemektedir. Tau Dirac zaman fonksyonun bant genişliğini belirleyen parametredir. Benzer bir çalışma iteratif TR çalışmalarının ultrasonik uygulamalarında çoklu nesne tespiti için gerçekleştirılmıştır [17].

## 5.5 Yapılan Çalışma

Elektromanyetik dalganın hareketini göstermek amacı ile 1 GHz merkez frekanslı diferansiyel Gauss dalgası kullanılmıştır. Noktasal kaynak  $177 \times 177 \times 177$  bir FDTD hücrende propagasyon yapmaktadır. Ortama materyal konulmayıp serbest uzay ve ortamın sınırları mükemmel iletken yapıldı. Ortama bağıl manyetik geçirgenlik sabiti 10 olan bir cisim yerleştirildi. Noktasal dipol diferansiyel Gauss kaynağından yayınlanan dalga yayılım yaptırılarak 58 gözlem noktasında elektrik alanın her iterasyonda gözlenen değerleri kaydedildi. Gözlem noktaları Gauss kaynağıyla aynı düzlemede bulunacak şekilde konumlandırıldı. Aynı işlem cismin bulunmadığı durum için de tekrarlandı. Daha sonra TR algoritması çalıştırılarak boş ortamda gözlem noktalarından işaretler zamanda tersine çevrilerek geri propagasyon yapıldı. Elektrik alanın ters varimax normu her iterasyonda hesaplanarak tepe noktası tespit edilerek cismen konumu bulundu. Varimax normunun tepe yaptıktan sonraki ilk yerel minimum yaptığı yerde görüntü alındı. Fakat sinyaldeki küçük dalgalanmalar göz ardı edildi.

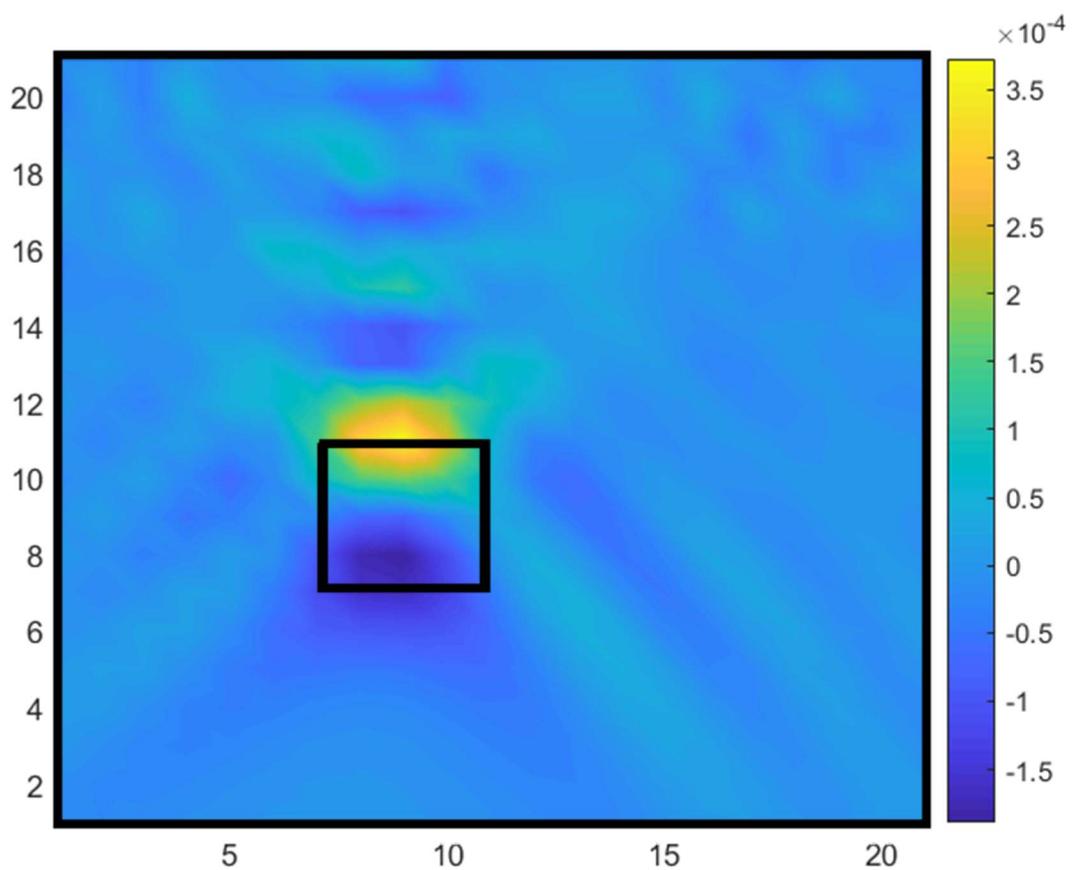
Çeşitli konfigürasyonlarda testler hesaplamalar tekrarlanarak testler yapıldı. Öncelikler boş uzayda dielektrik obje ile test yapıldı ve cisim konumu doğru olarak bulundu. Bu testte lineer dipol array gözlem kaynağı kullanıldı. Ayrıca aynı test düzlem array ile de aynı test tekrarlandı ve isabetli sonuç alındı. Bundan sonra lineer array tersine kaynak kullanarak serbest uzayda iletken nesne tesbiti yapıldı. Daha sonra ortama iki ayrı nesne konulup iki ayrı kaynaktan propagasyon yapılarak test yapıldı ve cisimlerin konumları yaklaşık olarak görüntülendi. Son olarak malzeme içinde yerleştirilen iletken nesne görüntülendi. Malzemenin cismin görüntüsünü saptırdığı gözlemlendi.

**Çizelge 5.1:** Ortam Parametreleri

Ortam hücre sayısı	177 x 177 x 177
Cisim hücre sayısı	4 x 4 x 4
Birim hücre ebatları	2.8 x 2.8 x 2.8 mm
Birim zaman adımı	5.39 ps
Ortam ebatları	0.5 x 0.5 x 0.5 m
Tümör ebatları	1.1 x 1.1 x 1.1 cm

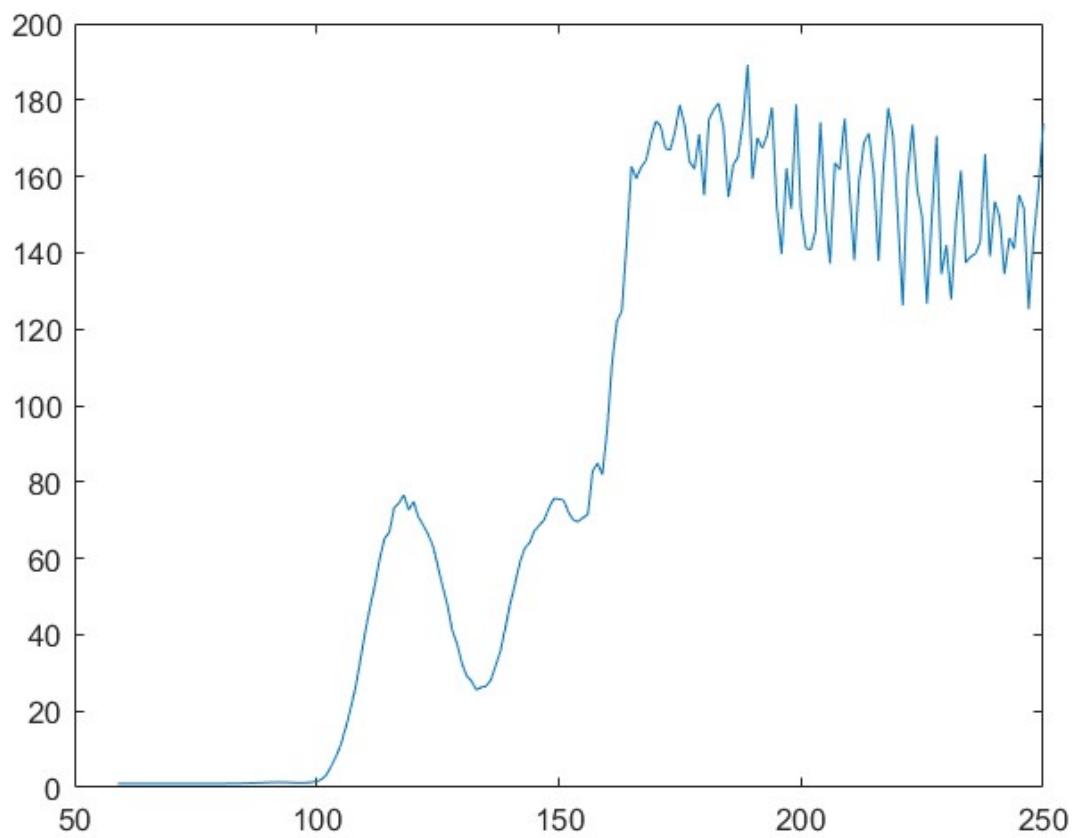
**Çizelge 5.2:** Kaynak Özellikleri

Merkez frekansı	1 GHz
Band genişliği	2 GHZ
tw	8.6339e-11
t0	2.5902e-10



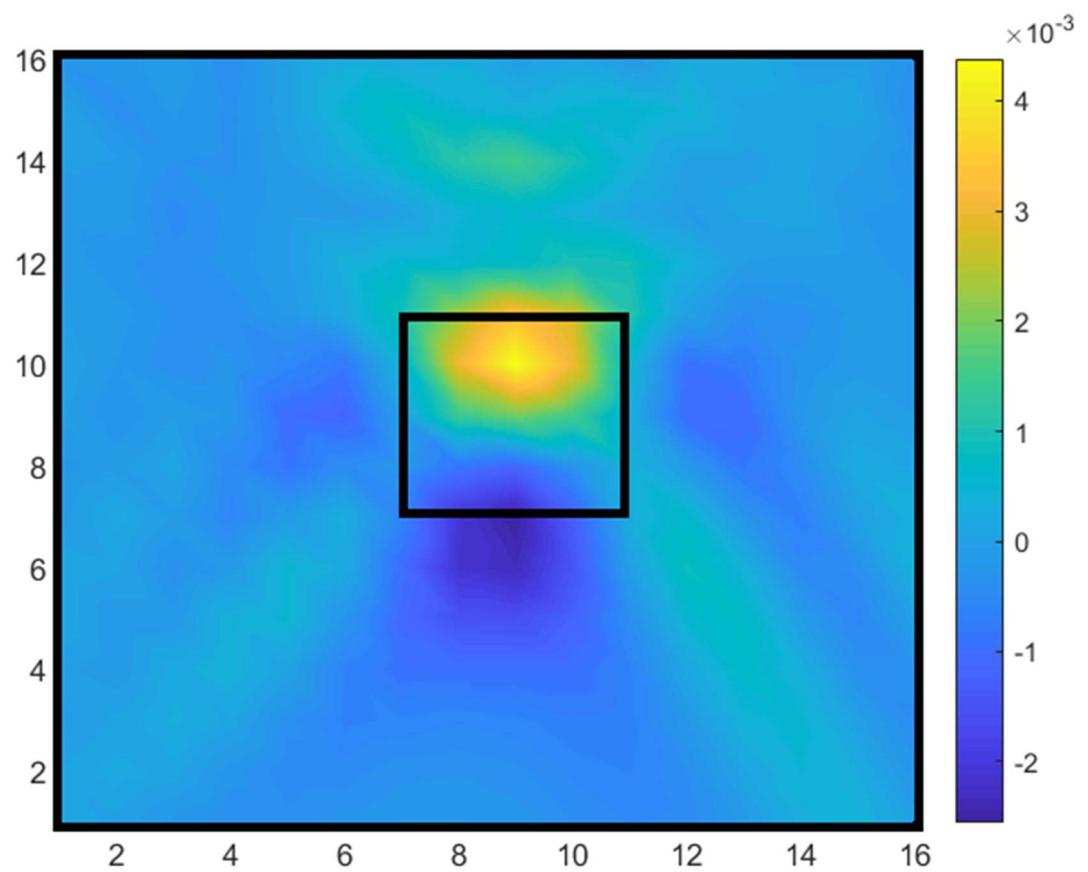
**Şekil 5.2:** Serbest Uzayda Dielektrik Nesne 134. İterasyon

Serbest uzayda dielektrik nesne, 28 dipolden oluşan lineer gözlem kaynağı kullanıldı. Gözlem işaretini pencerelenerek ters propagasyon yapıldı. Görüntü 134. İterasyonda gözlemlendi. Şekildeki görüntü kesittir.

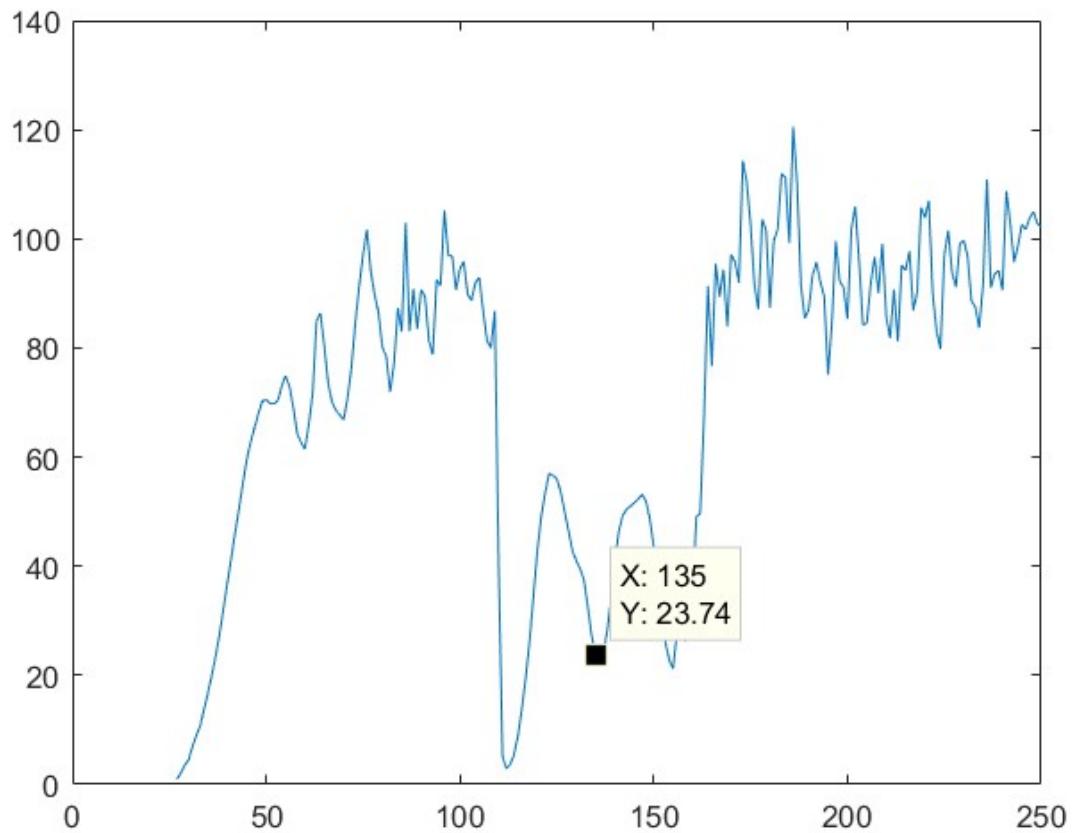


**Şekil 5.3:** Serbest Uzayda Dielektrik Nesne Ters Varimax Normu

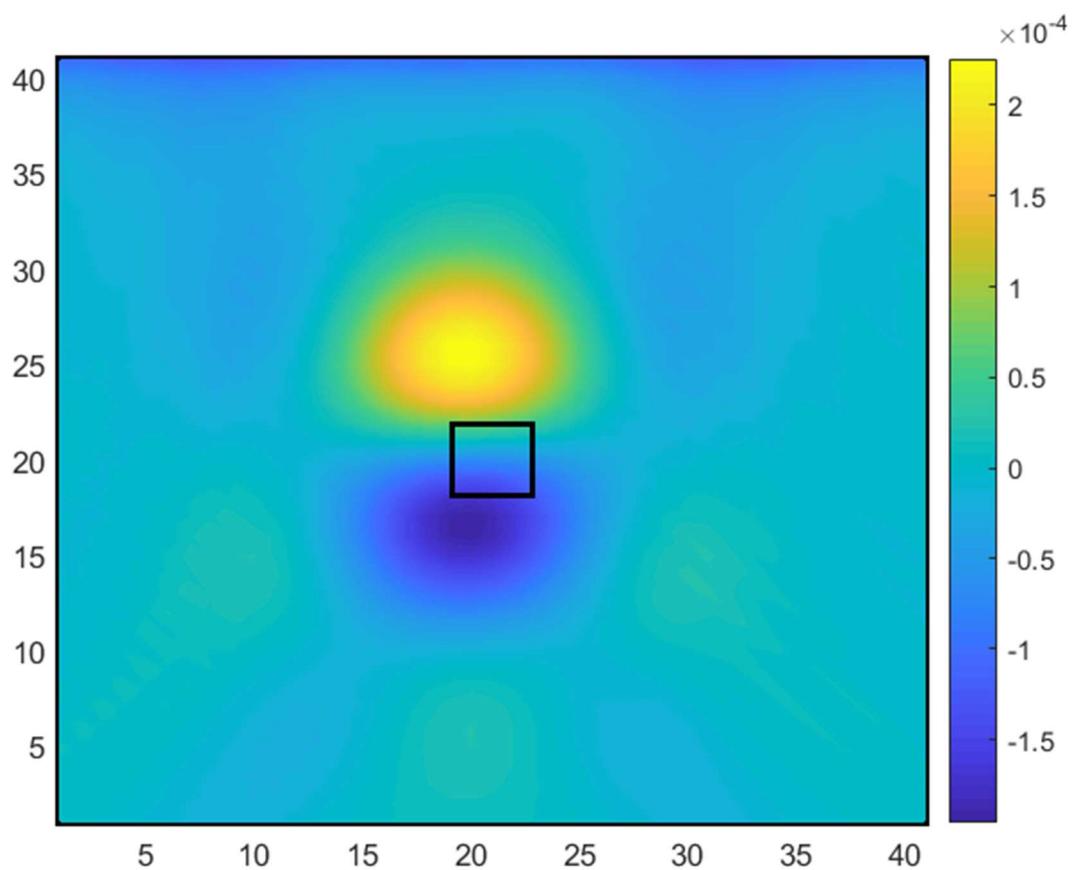
Şekil 5.3, konfigürasyona ait ters varimax normunu göstermektedir.



**Sekil 5.4:** 28x28 Düzlem Array ile Yapılan Gözlem 135. İterasyon

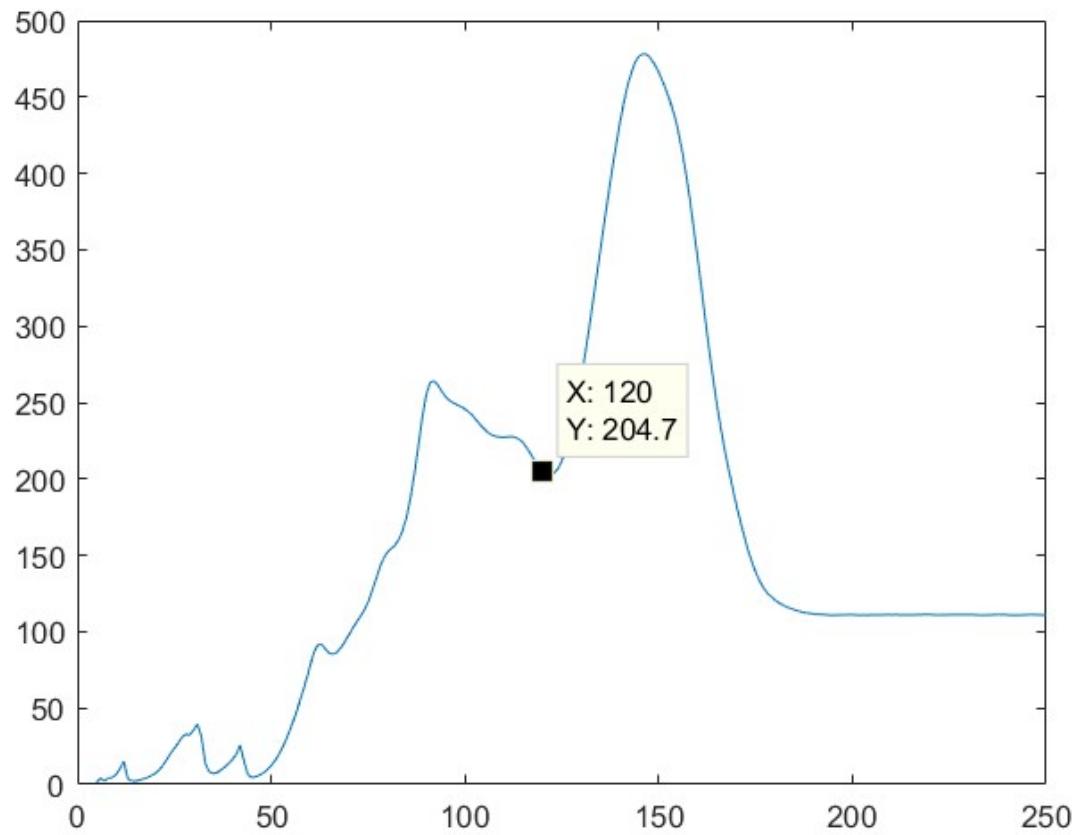


**Şekil 5.5:** Düzlem Array Gözlem Ters Varimax Normu

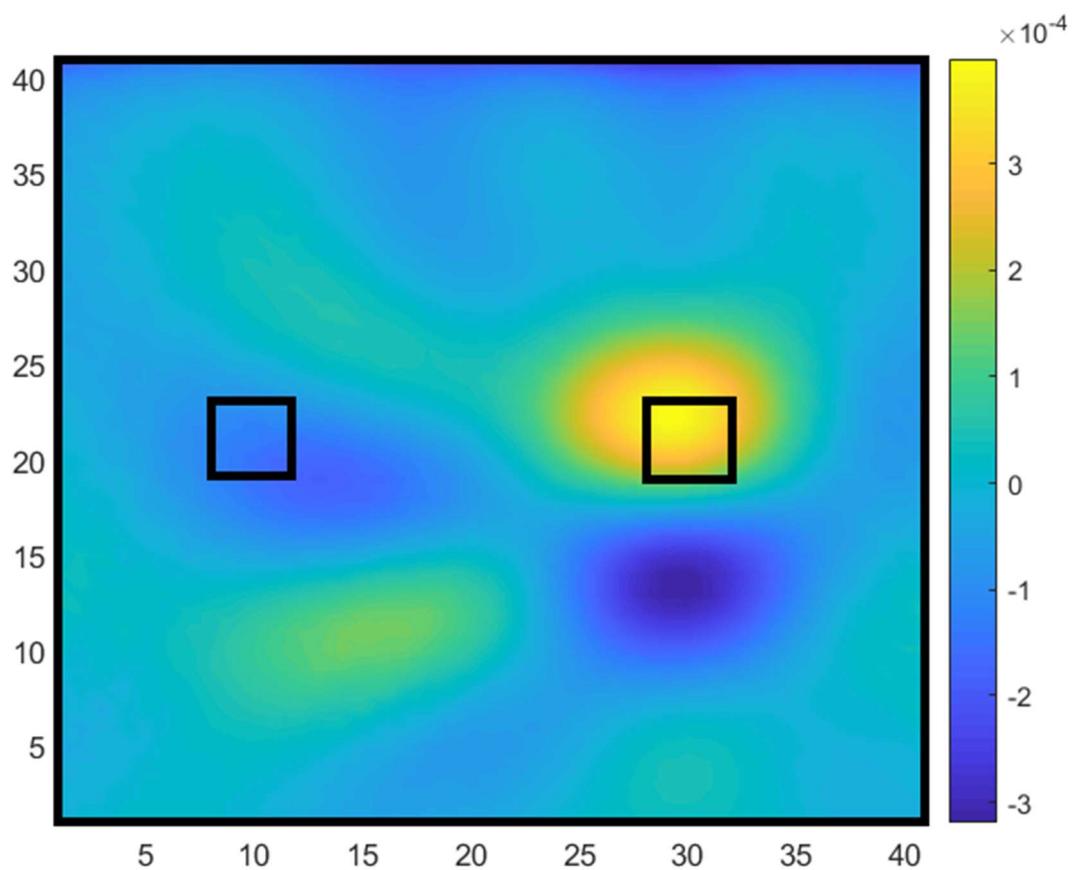


**Şekil 5.6:** Serbest Uzayda İletken Nesne 12. İterasyon

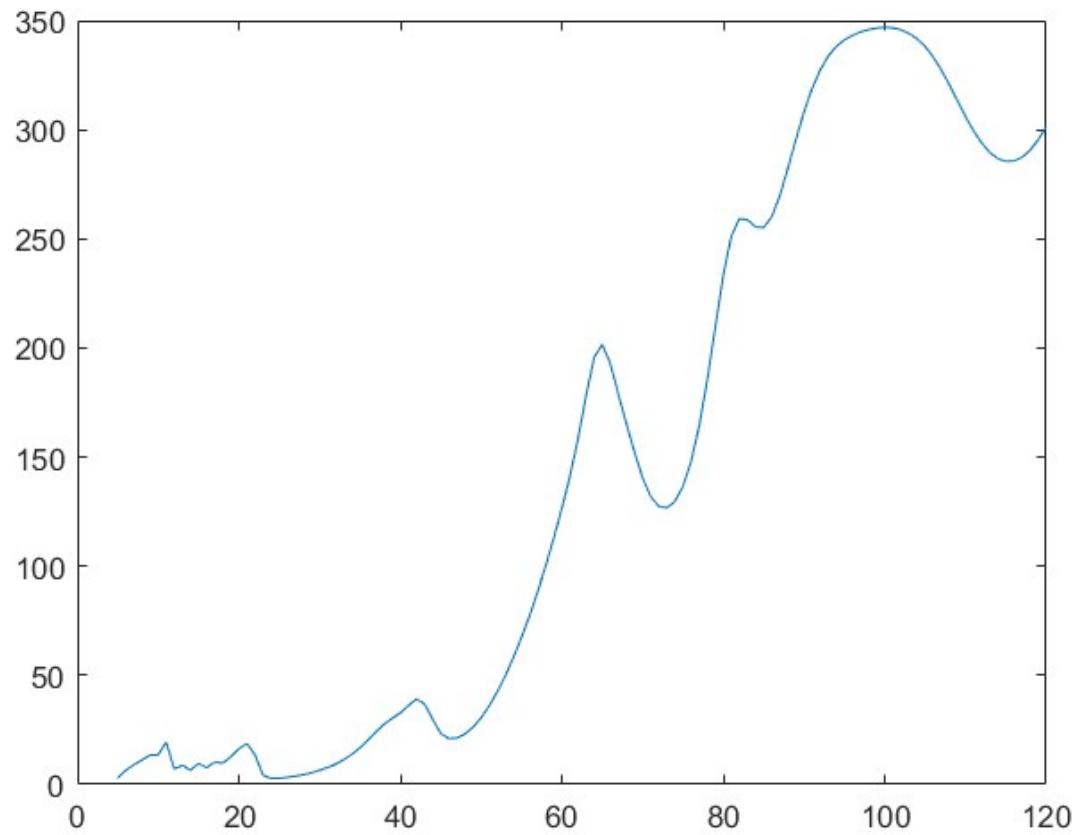
Serbest uzayda iletken nesne 28 dipol lineer array anten ile gözlenen 12. iterasyona ait görüntü şekilde yer almaktadır.



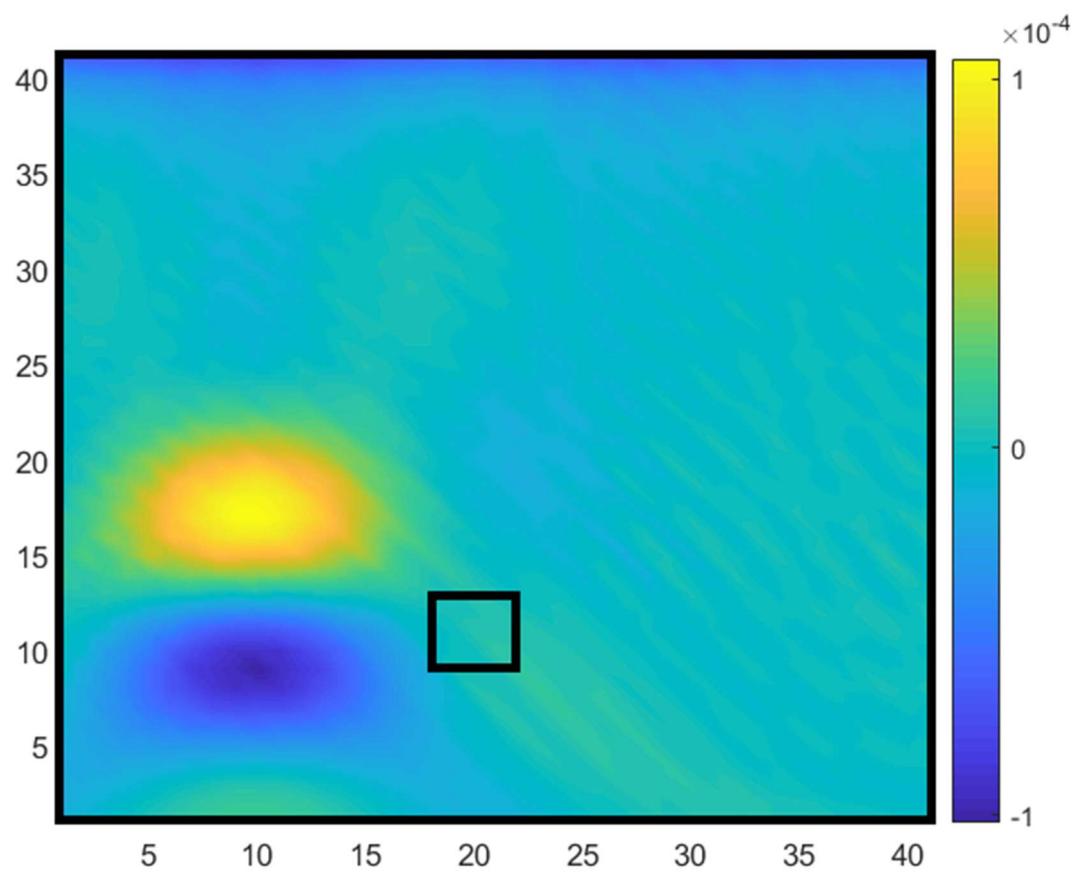
**Şekil 5.7:** Serbest Uzayda İletken Nesne Ters Varimax Normu



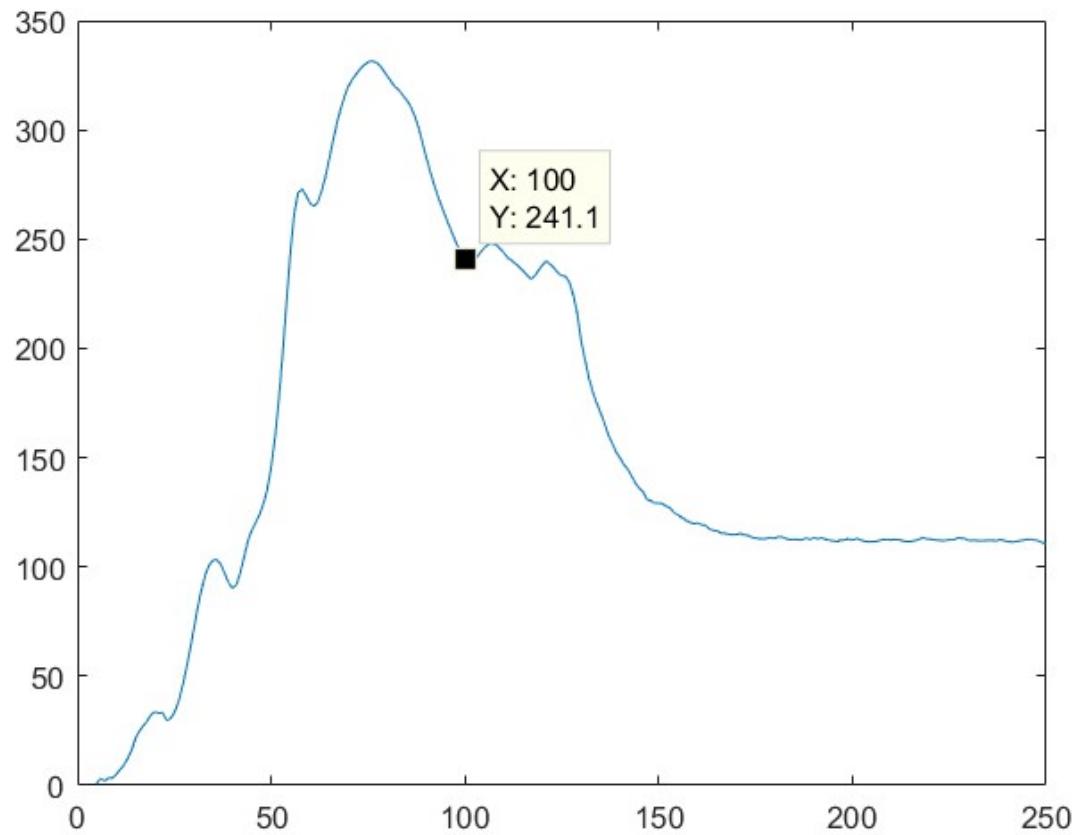
**Şekil 5.8:** İki Nesne ve İki Kaynak ile Gözlenen 115. İterasyona Ait Görüntü



**Şekil 5.9:** İki Nesne Gözlemine Ait Ters Varimax Normu



**Şekil 5.10:** Malzemeli Ortamda İletken Nesne Gözlemi 100. İterasyon



**Şekil 5.11:** Malzemeli Ortamda İletken Nesne Gözlemine Ait Ters Varimax Normu



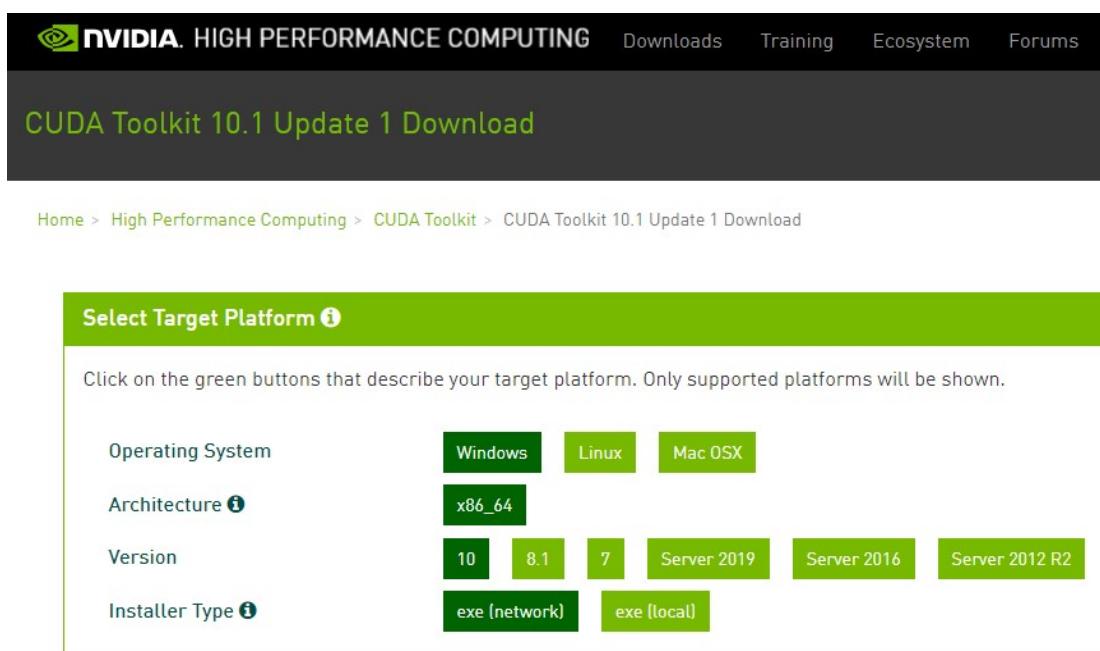
## 6. KODLARIN GPU ÜZERİNDE (PARALEL) ÇALIŞTIRILMASI

### 6.1 Giriş

Benzetim ortamı üç boyutlu olduğu için, simülasyon yapmak hafızaya büyük bir yük bindirmekte, CPU üzerinde çalışma yapıldığında simülasyon oldukça fazla zaman almaktadır. Bu yüzden kodları GPU üzerinde paralel işlemler halinde çalıştırmanın yolu araştırılmıştır.

### 6.2 CUDA Kütüphanesinin Kurulması

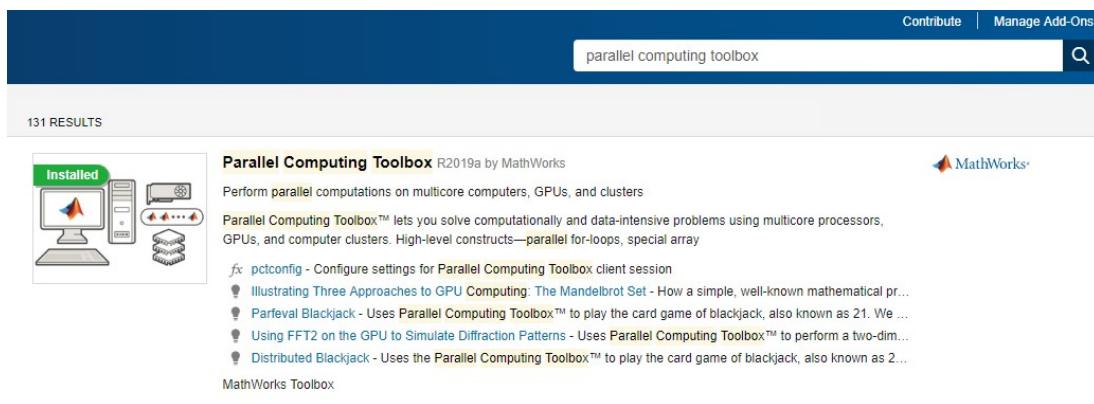
Kodların GPU üzerinde, yani ekran ekran kartı üzerinde çalıştırılabilmesi için “Nvidia” ekran kartı kullanan bir bilgisayara ihtiyaç vardır. Gerekli bilgisayar temin edildikten sonra herhangi bir arama motoruna “CUDA” yazıp aratılarak Nvidia’ya ait web sitesine ulaşılabilir. Bu web sitesinden bilgisayın işletim sistemine uygun yükleme versiyonu seçilerek CUDA kütüphanesi bilgisayara kurulabilir.



Şekil 6.1: İşletim Sistemine Uygun CUDA Kurulumu

### 6.3 Matlab Ortamında Gerekli Eklentinin Kurulması

Cuda kurulduktan sonra Matlab ortamında “Add-Ons”, ardından “Get Add-Ons” yolu izlenerek eklentilerin olduğu sayfaya ulaşılmalı, arama çubuğu, “Parallel Computing Toolbox” yazarak arama yapılmalı ve bulunan eklenti kurulmalıdır.



**Şekil 6.2:** Matlab Ortamında Gerekli Eklentinin Kurulması

### 6.4 Kodların Düzenlenmesi

Yazılan kodlar, GPU üzerinde çalışacak şekilde yeniden düzenlenmelidir. Bunun için yapılması gereken, tüm dizileri (array), “gpuArray” formatına dönüştürmektir. Örneğin, Matlab’de sıkça kullanılan “zeros” ve “ones” dizi oluşturma fonksiyonları yerine, “gpuArray.zeros” ve “gpuArray.ones” kullanılmalıdır.

```
%EM field dimensions                                %EM field dimensions
Hx = zeros(nx,ny,nz);                            Hx = gpuArray.zeros(nx,ny,nz);
Hy = zeros(nx,ny,nz);                            Hy = gpuArray.zeros(nx,ny,nz);
Hz = zeros(nx,ny,nz);                            Hz = gpuArray.zeros(nx,ny,nz);
Ex = zeros(nx,ny,nz);                            Ex = gpuArray.zeros(nx,ny,nz);
Ey = zeros(nx,ny,nz);                            Ey = gpuArray.zeros(nx,ny,nz);
Ez = zeros(nx,ny,nz);                            Ez = gpuArray.zeros(nx,ny,nz);
```

**Şekil 6.3:** Sırasıyla CPU ve GPU üzerinde çalışan ‘zeros’ fonksiyonu

Benzer şekilde, “array” formatında oluşan değişkenler de “gpuArray” formatına dönüştürülmelidir.

```

for n=1:l:n_iter
    %magnetic field derivatives
    Hxy = diff(Hx,1,2);
    Hxz = diff(Hx,1,3);
    Hzx = diff(Hz,1,1);
    Hzy = diff(Hz,1,2);
    Hyx = diff(Hy,1,1);
    Hyz = diff(Hy,1,3);

for n=1:l:n_iter
    %magnetic field derivatives
    Hxy_cpu = diff(Hx,1,2);
    Hxz_cpu = diff(Hx,1,3);
    Hzx_cpu = diff(Hz,1,1);
    Hzy_cpu = diff(Hz,1,2);
    Hyx_cpu = diff(Hy,1,1);
    Hyz_cpu = diff(Hy,1,3);
    Hxy = gpuArray(Hxy_cpu);
    Hxz = gpuArray(Hxz_cpu);
    Hzx = gpuArray(Hzx_cpu);
    Hzy = gpuArray(Hzy_cpu);
    Hyx = gpuArray(Hyx_cpu);
    Hyz = gpuArray(Hyz_cpu);

```

**Şekil 6.4:** Normal dizinin GPU dizisine dönüştürülmesi

Gerekli dönüşümler tamamlandıktan sonra kod çalıştırılırsa, eğer CUDA kütüphanesi başarılı bir şekilde kurulmuşsa Matlab otomatik olarak kütüphaneyi tanıyacak ve kodları GPU üzerinde çalıştıracaktır.

## 6.5 Kodların GPU Üzerinde Çalıştırılmasının Etkisi

Kodların GPU üzerinde paralel olarak çalıştırılması, çalışma hızında önemli bir etki yapmıştır. Kodlar sırasıyla CPU ve GPU üzerinde, 250 iterasyon çalıştırılarak test edilmiştir. CPU üzerinde çalışan kodların 250 iterasyonu 358 saniyede tamamladığı bilgisayarda, GPU üzerinde çalışan kodlar 250 iterasyonu 40 saniyede tamamlamıştır. Bu teste göre kodların GPU üzerinde çalıştırılması, işlem hızını yaklaşık 9 katına çıkarmıştır.



## **7. GERÇEKÇİ KISITLAR, SONUCLAR VE ÖNERİLER**

### **7.1 Çalışmanın Uygulama Alanı**

Tıbbi görüntüleme üzerine geliştirilen bu projenin uygulama alanı, başta kanser tespiti olmak üzere biyomedikal alandır.

### **7.2 Gerçekçi Tasarım Kısıtları.**

#### **7.2.1 Maliyet**

Proje yazılımsal bir projedir. Gerekli yazılımlar ya okul tarafından lisanslı olarak ya da açık kaynak olarak temin edilecektir. Herhangi bir masrafi bulunmamaktadır.

#### **7.2.2 Standartlar**

Bu proje, biyomedikal uygulamalar ve yazılımsal produktlere ilişkin IEEE, EU ve TSE standartlarına uygun bir şekilde gerçekleştirilecektir. Ayrıca, hiçbir kaynak atıf yapılmaksızın kullanılmayacaktır. Sürecin bir takım çalışması halinde gerçekleştirilmesine de önem verilecektir.

#### **7.2.3 Sosyal, çevresel ve ekonomik etki**

Proje kanser teşhis konusunda gelişmelere öncü olabilme potansiyelini taşımaktadır. Bu proje toplumda insan sağlığının gelişmesine olumlu yolda etki sağlayacaktır. İnsan nüfusunun daha sağlıklı bir yaşam sürmesine vesile olması beklenmektedir.

#### **7.2.4 Sağlık ve güvenlik riskleri**

Ofis koşullarında karşılaşılabilcek ve sağlık ve güvenlik riskleri dışında bir tehlike öngörülmektedir.

### **7.3 Sonuçlar**

Bu tezde, FDTD ve TR algoritmaları birleştirilip nesne tespitinde kullanılacak bir algoritma geliştirilmiştir. TR algoritması için kaydedilen gözlem kaynak çizgi halinde kaydedilerek test edilmiştir. Sonuçlar FDTD TR algoritmasının nesne tespitini konumsal olarak doğru yaptığı göstermektedir. Algoritma tıbbi görüntüleme gibi çeşitli uygulama alanlarında kullanım potansiyeline sahip ve kolay modellenebilir bir yöntemdir. Paralel hesaplama teknikleri kullanarak hesaplama hızı ve kapasitesi artırlabilir.

### **7.4 Geleceğe Yönelik Öneriler**

Algoritma tıbbi görüntüleme gibi çeşitli uygulama alanlarında kullanım potansiyeline sahip ve kolay modellenebilir bir yöntemdir. Paralel hesaplama teknikleri kullanarak işlem süresi kısaltılabilir ve daha yüksek örnekleme frekanslarında çalışılabilir.

## KAYNAKLAR

- [1] **Yee, K.S.**, (1966). Numerical solution of initial boundary value problems involving Maxwell's equations, *IEEE Trans. Antennas and Propagat*, AP-14(3), 302-307.
- [2] **Erol, Y. ve Balık, H.H.**, (2001). Zaman Domeninde sonlu farklar metodu ile tek boyutlu yapılarda elektromanyetik dalga yayılımının simülasyonu, *Ulusal Bilişim – Multimedya Konferansı*, Elazığ.
- [3] **Schneider, J. B.**, (2016). Understanding the Finite-Difference Time-Domain Method, from <http://bbs.hwr.com.cn/downebd/90845d1368122665-ufdtd.pdf>
- [4] **Sevgi, L.**, (1999). Elektromanyetik Problemler ve Sayısal Yöntemler, Birsen Yayınevi, İstanbul.
- [5] **Dalgiç, H.A.**, (2014). Yüksek frekans deniz radarının zaman uzayı sonlu farklar yöntemi ile modellenmesi, *Yüksek Lisans Tezi*, Gebze İleri Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü, Gebze.
- [6] **Yavuz, M.E.**, (2008). Time Reversal Based Signal Processing Techniques for Ultrawideband Electromagnetic Sensing in Random Media, *Doktora Tezi*, The Ohio State University, Ohia, USA.
- [7] **Foschini G.J. ve Gans, M.J.**, (1998). On limits of wireless communication in a fading environment when using multiple antennas, *Wireless Personal Commun.*, 6(3), 311-315.
- [8] **Paulraj, A.J., Gore, D.A., Nabar, R.U. ve Bolcskei, H.**, (2004). An overview of MIMO communications – a key to gigabit wireless, *Proc. IEEE*, 92(2), 198-218.
- [9] **Paulraj, A.J., ve Boon, C.N.**, (1998). Space-time modems for wireless personal communications, *IEEE Pers. Commun.*, 5(1), 36-48.
- [10] **Kosmas, P. Ve Rappaport, C.M.**, (2005). Time Reversal With the FDTD Method for Microwave Breast Cancer Detection, *IEEE Transactions on Microwave Theory and Techniques*, 53(7), 2317-2323.
- [11] **Başaran, S.C. ve Çakır, G.**, Zaman Domeninde Sonlu Farklar Yöntemi İle Mikroserit Hatlara Uygulanan MUR ve Dispersive (DBC) Sınır Koşullarının Analizi, from [http://www.emo.org.tr/ekler/d1f77dc3b4264ef\\_ek.pdf](http://www.emo.org.tr/ekler/d1f77dc3b4264ef_ek.pdf)
- [12] **Altman, C. ve Suchy, K.**, (1991). Reciprocity, Spatial Mapping and Time Reversal in Electromagnetics, Springer, Amsterdam.
- [13] **Sorrentino, R., Rosseli, L. ve Mezzanote, P.**, (1993). Time-reversal in finite difference time domain method, *IEEE Microw. Guided Wave Lett.*, 3-11, 402-404.

- [14] **Fink, M., Cassereau, D., Derode, A., Prada, C., Roux, P., Tanter, M., Thomas, J.L. ve Wu, F.**, (2000). Time-reversed acoustics, *Rep. Prog. Phys.*, 63, 1933-1995.
- [15] **Taflove, A. ve Hagness S.C.**, (2000). Computational Electrodynamics: The Finite-Difference Time-Domain Method, MA: Artech House, Boston.
- [16] **Xu, X. Miller, E.L. ve Rappaport, C.M.**, (2003). Minimum entropy regularization in frequency-wavenumber migration to localize subsurface objects, *IEEE Trans. Geosci. Remote Sens.*, 41, 1804-1812.
- [17] **Montaldo, G., Tanter, M. ve Fink, M.**, (2004). Real time inverse filter focusing through iterative time reversal, *J. Acoust. Soc. Amer.*, 115, 768-775.

## EKLER

### EKA Kodlar

```
1      %TMz Polarization
2      %physical constants
3      c      = 2.998e8;
4      eta0  = 120*pi;
5      mu0   = pi*4e-7;
6      eps0  = 1e-9/(36*pi);
7      %environment parameters
8      width  = 1;
9      height = 1;
10     tw     = 1e-9/pi;%
11     t0     = 4*tw;
12     %discretization parameters
13     dx = 0.005;
14     dy = 0.005;
15     nx = width/dx;
16     ny = height/dy;
17     dt  = 0.95/(c*sqrt(dx^2+dy^2));
18     %calculation parameters
19     n_iter = 1000;
20     c1 = dt/(mu0*dy);
21     c2 = dt/(mu0*dx);
22     c3 = dt/(eps0*dx);
23     c4 = dt/(eps0*dy);
24     %initialization
25     Hx = zeros(nx,ny);
26     Hy = zeros(nx,ny);
27     Ez = zeros(nx,ny);
28     %iteration
29     for n=1:n_iter
30         %Maxwell Equations (TMz)
31         Ezx = diff(Ez,1,1);
32         Ezy = diff(Ez,1,2);
33         Hx(2:nx-1,2:ny) = Hx(2:nx-1,2:ny) - c1*Ezy(2:nx-1,:);
34         Hy(2:nx,2:ny-1) = Hy(2:nx,2:ny-1) + c2*Ezx(:,2:ny-1);
35         Hxy = diff(Hx,1,2);
36         Hyx = diff(Hy,1,1);
37         Ez(2:nx-1,2:ny-1) = Ez(2:nx-1,2:ny-1) + c3*Hyx(2:nx-1,2:ny-1) - c4*Hxy(2:nx-1,2:ny-1);
38         %Gaussian Source
39         f(n)= exp(-(n*dt-t0)^2/(tw^2))/dy;
40         Ez(round(nx/2),round(ny/2)) = Ez(round(nx/2),round(ny/2)) + f(n);
41         %Neuman Condition
42         Ez(:,2) = -Ez(:,1);
43         Ez(2,:) = -Ez(1,:);
44         Ez(:,ny-1) = -Ez(:,ny);
45         Ez(nx-1,:) = -Ez(nx,:);
46         %display
47         %n = n + 1;
48         pcolor(Ez);
49         shading interp;
50         drawnow
51     end
```

Şekil A.1: Algorimanın 2 boyutlu olarak Matlab ortamında gerçekleşmesi

```

1   %material
2   adipose = 10;
3   tumor   = 60;
4   mx = 3 * ny / 8;
5   my = 0;
6   mz = 0;
7   mw = nx / 4; % width
8   mh = ny / 4; % height
9   ml = nz / 4; % length
10  al = ny / 2;
11  eps = ones(nx,ny,nz) * eps0;
12  for i=1:1:nx
13    for j=1:1:ny
14      for k=1:1:nz
15        % adipose tissue is located under z < al
16        if (k<al)
17          eps(i,j,k) = eps0 * adipose ;
18        end
19        if (i>mx && i<(mw+mx) && j>my && j<(mh+my) && k>mz && k<(ml+mz))
20          eps(i,j,k) = eps0 * tumor;
21        end
22      end
23    end
24  end

```

**Sekil A.2:** Yazılım ortamında malzeme eklenmesi

```

TR.m  WithoutTumor.m  WithTumor.m  +
1 %physical constants
2 - clear all;
3 - close all;
4 - load 'withtumor.mat';
5 - load 'withouttumor.mat';
6
7 - c0      = 2.998e8;
8 - eta0   = 120*pi;
9 - mu0    = pi*4e-7;
10 - eps0   = 1e-9/(36*pi);
11 %box dimensions
12 - width   = 0.5; % 30cm
13 - height  = 0.5;
14 - length  = 0.5; % 1cm
15 %source parameters
16 - f0      = 1e9; % GHz
17 - band    = 2e9;
18 - tw      = sqrt(-log(0.1)/(pi*band)^2);%1e-8/pi;
19 - t0      = 4*tw;
20 %spatial discretization
21 - adipose = 5;
22 - tumor   = 10;
23 - sigma   = 5;
24 - epsr   = tumor;
25 - w      = 2 * pi * band;
26 - k      = (w/c0)*sqrt(epsr-lj*sigma/(w*eps0));
27 - beta   = real(k);
28 - c      = w / beta;
29 - lambda = c/f0;
30 - dxmax  = lambda / 20;
31 - dx     = dxmax;
32 - dy     = dx;
33 - dz     = dx;
34 - nx     = round(width/dx);
35 - ny     = round(height/dy);
36 - nz     = round(length/dz);
37
38 %source position
39 - srcx  = round(nx / 2);
40 - srcy  = round( 3 * ny / 4);
41 - srcz  = round(nz / 2);
42

```

```

43 % material
44 - eps = ones(nx,ny,nz) * eps0; %* adipose;
45 - sigma = zeros(nx,ny,nz);% * f0 * le-9 * 0.5 - 0.5;
46 %temporal discretization
47 - dt = 0.99/(c0*sqrt(dx^2+dy^2+dz^2));
48
49 - recl = trec - rec;
50 - tau = 100e-12;
51 - [foo,tp] = max(abs(recl),[],2);
52 - for k=1:nrec
53 -     rekn(k,:) = exp(-((dt*(1:l:n_iter)-tp(k)))/tau).^2) .* recl(k,:);
54 - end
55 - % hold on
56 - % plot(rec(15,:))
57 - % plot(exp(-((dt*(1:l:n_iter)-tp(15)))/tau).^2))
58 - % draw now
59 - %
60 - % while 1
61 - % end
62
63
64 %EM field dimensions
65 - Hx = zeros(nx,ny,nz);
66 - Hy = zeros(nx,ny,nz);
67 - Hz = zeros(nx,ny,nz);
68 - Ex = zeros(nx,ny,nz);
69 - Ey = zeros(nx,ny,nz);
70 - Ez = zeros(nx,ny,nz);
71 %iteration
72 - i = 0;
73 - for n=1:l:n_iter
74 -     %magnetic field derivatives
75 -     Hxy = diff(Hx,1,2);
76 -     Hxz = diff(Hx,1,3);
77 -     Hzx = diff(Hz,1,1);
78 -     Hzy = diff(Hz,1,2);
79 -     Hyx = diff(Hy,1,1);
80 -     Hyz = diff(Hy,1,3);
81 -     %electric field maxwell equations
82 -     epsi = eps(:,2:end-1,2:sz-1);
83 -     ksi = (dt * sigma(:,2:end-1,2:sz-1)) ./ ( 2 * epsi );
84 -     c2 = (1./(1+ksi)).*(dt./epsi);
85 -     c1 = (1-ksi)./(1+ksi);
86 -     Ex(:,2:end-1,2:end-1) = c1.*Ex(:,2:end-1,2:sz-1) - c2.*((1/dy)*Hzy(:,1:end-1,2:end-1) - (1/dz)*Hyz(:,2:ny-1,1:end-1));
87

```

```

88 -      epsi = eps(2:end-1,:,2:end-1);
89 -      ksi = (dt * sigma(2:end-1,:,2:end-1)) ./ ( 2 * epsi );
90 -      c2 = (1./(1+ksi)).*(dt./epsi);
91 -      cl = (1-ksi)./(1+ksi);
92 -      Ez(2:end-1,:,2:end-1) = cl.*Ey(2:end-1,:,2:end-1) - c2.*((1/dz)*Hxz(2:end-1,:,1:end-1) - (1/dx)*Hzx(1:end-1,:,2:end-1));
93 -
94 -      epsi = eps(2:end-1,2:end-1,:);
95 -      ksi = (dt * sigma(2:end-1,2:end-1,:)) ./ ( 2 * epsi );
96 -      c2 = (1./(1+ksi)).*(dt./epsi);
97 -      cl = (1-ksi)./(1+ksi);
98 -      Ez(2:end-1,2:end-1,:) = cl.*Ez(2:end-1,2:end-1,:) - c2.*((1/dx)*Hyx(1:end-1,2:end-1,:) - (1/dy)*Hxy(2:end-1,1:end-1,:));
99 -
100 - %TR sources
101 - for k=1:nrec
102 -     Ez(recdx * k, recy, recz) = Ez(recdx * k, recy, recz) + recn(k, n_iter-n+1);
103 - end
104 - %Ez(recx, recdy , recz)
105 - %rec(l,n_iter-n)
106 - %electric field derivatives
107 - Exy = diff(Ex,1,2);
108 - Exz = diff(Ex,1,3);
109 - Ezx = diff(Ez,1,1);
110 - Ezy = diff(Ez,1,2);
111 - Eyx = diff(Ey,1,1);
112 - Eyz = diff(Ey,1,3);
113 -
114 - %magnetic field maxwell equations
115 - Hx(:,l:end-1,l:end-1) = Hx(:,l:end-1,l:end-1) + (dt/(mu0*dy))*Ezy(:, :, l:end-1) - (dt/(mu0*dz))*Eyz(:, :, l:end-1,:);
116 - Hy(l:end-1,:,l:end-1) = Hy(l:end-1,:,l:end-1) + (dt/(mu0*dz))*Exz(l:end-1,:, :) - (dt/(mu0*dx))*Ezx(:, :, l:end-1);
117 - Hz(l:end-1,l:end-1,: ) = Hz(l:end-1,l:end-1,: ) + (dt/(mu0*dx))*Eyx(:, l:end-1,: ) - (dt/(mu0*dy))*Exy(l:end-1,:,:);
118 -
119 - %display
120 - if (mod(n,10)==0)
121 -     slice(:,:,)=Ez(60:100,round(ny/2)-20:round(ny/2)+20,srcz);
122 -     pcolor(slice.');
123 -     colorbar;
124 -     shading interp
125 -     drawnow
126 - end
127 - i = i+1;
128 - disp(i)
129 -
130 - R(n) = varimax_norm(Ez(60:100,round(ny/2)-20:round(ny/2)+20,srcz));
131 - end
132 -
133 - figure;plot(R)
134 -
135 - function R = varimax_norm(Ez)
136 -     R = sum(sum(sum(Ez.^2)))^2 / sum(sum(sum(Ez.^4)));
137 - end
138 -

```

```

TR.m WithoutTumor.m WithTumor.m +
1 %physical constants
2 - clear all;
3 - close all;
4 - c0 = 2.998e8;
5 - eta0 = 120*pi;
6 - mu0 = pi*4e-7;
7 - eps0 = 1e-9/(36*pi);
8 %box dimensions
9 - width = 0.5; % cm
10 - height = 0.5;
11 - length = 0.5; % cm
12 %source parameters
13 - f0 = 1e9; % GHz
14 - band = 2e9;
15 - tw = sqrt(-log(0.1)/(pi*band)^2);%1e-8/pi;
16 - t0 = 4*tw;
17 %spatial discretization
18 - adipose = 1; %5;
19 - sigma = 5;
20 - epsr = 10;
21 - w = 2 * pi * band;
22 - k = (w/c0)*sqrt(epsr-lj*sigma/(w*eps0));
23 - beta = real(k);
24 - c = w / beta;
25 - lambda = c/f0;
26 - dxmax = lambda / 20;
27 - dx = dxmax;
28 - dy = dxmax;
29 - dz = dxmax;
30 - nx = round(width/dx);
31 - ny = round(height/dy);
32 - nz = round(length/dz);
33 %source position
34 - srcx = round(nx / 2);
35 - srcy = round( 3 * ny / 4);
36 - srcz = round(nz / 2);
37 %material
38 - al = 0;
39
40 - eps = ones(nx,ny,nz) * eps0 ;
41 - sigma = zeros(nx,ny,nz);%*f0 * 1e-9 * 0.5 - 0.5;
42 - for i=1:1:nx
43 -   for j=1:1:ny
44 -     for k=1:1:nz
45 -       % adipose tissue is located under z < al
46 -       if (k<al)
47 -         eps(i,j,k) = eps0 * adipose;
48 -         sigma(i,j,k) = f0 * 1e-9 * 0.5 - 0.5;
49 -       end
50 -     end
51 -   end
52 - end

```

```

53 %time discretization
54 - dt = 0.99/(c0*sqrt(dx^2+dy^2+dz^2));
55
56 - tw=16*dt;
57 - t0=3*tw;
58
59 - n_iter = 250;
60 %receivers
61 - nrec = round(nx / 3)-1;
62 - recdx = round(nx / nrec);
63 - recy = srcy-20;
64 - recz = srcz;
65 - rec = zeros(nrec,n_iter);
66 %EM field dimensions
67 - Hx = zeros(nx,ny,nz);
68 - Hy = zeros(nx,ny,nz);
69 - Hz = zeros(nx,ny,nz);
70 - Ex = zeros(nx,ny,nz);
71 - Ey = zeros(nx,ny,nz);
72 - Ez = zeros(nx,ny,nz);
73 %iteration
74 - i = 0;
75 - for n=1:n_iter
76 %magnetic field derivatives
77 - Hxy = diff(Hx,1,2);
78 - Hxz = diff(Hx,1,3);
79 - Hzx = diff(Hz,1,1);
80 - Hzy = diff(Hz,1,2);
81 - Hyx = diff(Hy,1,1);
82 - Hyz = diff(Hy,1,3);
83
84 %electric field maxwell equations
85 - epsi = eps(:,2:end-1,2:sz-1);
86 - ksi = (dt * sigma(:,2:end-1,2:sz-1)) ./ ( 2 * epsi );
87 - c2 = (1./(1+ksi)).*(dt./epsi);
88 - cl = (1-ksi)./(1+ksi);
89 - Ex(:,2:end-1,2:end-1) = cl.*Ex(:,2:end-1,2:sz-1) + c2.*((1/dy)*Hzy(:,1:end-1,2:end-1) - (1/dz)*Hyz(:,2:ny-1,1:end-1));
90
91 - epsi = eps(2:end-1,:,2:end-1);
92 - ksi = (dt * sigma(2:end-1,:,2:end-1)) ./ ( 2 * epsi );
93 - c2 = (1./(1+ksi)).*(dt./epsi);
94 - cl = (1-ksi)./(1+ksi);
95 - Ey(2:end-1,:,:2:end-1) = cl.*Ey(2:end-1,:,:2:end-1) + c2.*((1/dz)*Hxz(2:end-1,:,1:end-1) - (1/dx)*Hzx(1:end-1,:,2:end-1));
96
97 - epsi = eps(2:end-1,2:end-1,:);
98 - ksi = (dt * sigma(2:end-1,2:end-1,:)) ./ ( 2 * epsi );
99 - c2 = (1./(1+ksi)).*(dt./epsi);
100 - cl = (1-ksi)./(1+ksi);
101 - Ez(2:end-1,2:end-1,:) = cl.*Ez(2:end-1,2:end-1,:) + c2.*((1/dx)*Hyx(1:end-1,2:end-1,:) - (1/dy)*Hxy(2:end-1,1:end-1,:));
102

```

```

103
104 %gaussian source
105 f(n) = sin(2*pi*f0*n*dt)*exp(-(n*dt-t0)^2/(tw^2))/dy;
106 f_ = -2*(n*dt-t0)/tw*exp(-(n*dt-t0)^2/(tw^2))/dy;
107 Ez(srcx,srcy,srcz) = Ez(srcx,srcy,srcz) + f(n);
108 %Ezn(n)=Ez(srcx,srcy,srcz);
109
110 %electric field derivatives
111 Exy = diff(Ex,1,2);
112 Exz = diff(Ex,1,3);
113 Ezx = diff(Ez,1,1);
114 Ezy = diff(Ez,1,2);
115 Eyx = diff(Ey,1,1);
116 Eyz = diff(Ey,1,3);
117
118 %magnetic field maxwell equations
119 Hx(:,1:end-1,1:end-1) = Hx(:,1:end-1,1:end-1) - (dt/(mu0*dy))*Ezy(:,:,1:end-1) + (dt/(mu0*dz))*Eyz(:,:,1:end-1,:);
120 Hy(1:end-1,:,:1:end-1) = Hy(1:end-1,:,:1:end-1) - (dt/(mu0*dz))*Exz(1:end-1,:,:,:) + (dt/(mu0*dx))*Ezx(:,:,1:end-1,:);
121 Hz(1:end-1,1:end-1,:) = Hz(1:end-1,1:end-1,:) - (dt/(mu0*dx))*Eyx(:,:,1:end-1,:) + (dt/(mu0*dy))*Exy(1:end-1,:,:,:);
122
123 for k=1:nrec
124 rec(k,n) = Ez(recdx * k, recy, recz);
125 end
126
127 %display
128 if (mod(i,5)==0)
129 slice(:,:,)=Ez(:,:,srcz);
130 pcolor(slice');
131 colorbar;
132 shading interp
133 drawnow
134 end
135 i = i+1;
136 disp(i)
137 end
138 close all
139 hold on
140 for k=1:nrec
141 plot(rec(k,:))
142 end
143
144 save('withouttumor.mat','rec','nrec','n_iter','recy','recdx','recz')
145

```

```

TR.m × WithoutTumor.m × WithTumor.m × +
1 %physical constants
2 - clear all;
3 - close all;
4 - c0 = 2.998e8;
5 - eta0 = 120*pi;
6 - mu0 = pi*4e-7;
7 - eps0 = 1e-9/(36*pi);
8 %box dimensions
9 - width = 0.5; % cm
10 - height = 0.5;
11 - length = 0.5; % cm
12 %source parameters
13 - f0 = 1e9; % GHz
14 - band = 2e9;
15 %tw = sqrt(-log(0.1)/(pi*band)^2);%le-8/pi;
16 %spatial discretization
17 - adipose = 1; %5;
18 - tumor = 10;
19 - sigma = 5;
20 - epsr = tumor;
21 - w = 2 * pi * band;
22 - k = (w/c0)*sqrt(epsr-lj*sigma/(w*eps0));
23 - beta = real(k);
24 - c = w / beta;
25 - lambda = c/f0;
26 - dxmax = lambda / 20;
27 - dx = dxmax;
28 - dy = dxmax;
29 - dz = dxmax;
30 - nx = round(width/dx);
31 - ny = round(height/dy);
32 - nz = round(length/dz);
33 %source position
34 - srcx = round(nx / 2);
35 - srcy = round( 3 * ny / 4);
36 - srcz = round(nz / 2);
37 %material
38 - mw = 4; % width
39 - mh = 4; % height
40 - ml = 4; % length
41 %mx = nx / 4-10-mw/2;
42 - mx = nx / 2-10-mw/2;
43 - my = ny / 2-mh/2;
44 - mz = nz / 2-ml/2;
45
46 - al = 0;
47
48 - eps = ones(nx,ny,nz) * eps0 ;
49 - sigma = zeros(nx,ny,nz);%*f0 * 1e-9 * 0.5 - 0.5;

```

```

50 -   for i=1:1:nx
51 -     for j=1:1:ny
52 -       for k=1:1:nz
53 -         % adipose tissue is located under z < al
54 -         if (k<al)
55 -           eps(i,j,k) = eps0 * adipose;
56 -           sigma(i,j,k) = f0 * le-9 * 0.5 - 0.5;
57 -         end
58 -         if (i>mx && i<(mw+mx) && j>my && j<(mh+my) && k>mz && k<(ml+mz))
59 -           eps(i,j,k) = eps0 * tumor;
60 -           sigma(i,j,k) = f0 * le-9 - 0.5;
61 -         end
62 -       end
63 -     end
64 -   end
65 -   %time discretization
66 -   dt = 0.99/(c0*sqrt(dx^2+dy^2+dz^2));
67 -
68 -   tw=16*dt;
69 -   t0=3*tw;
70 -
71 -   n_iter = 250;
72 -   %receivers
73 -   nrec = round(nx / 3)-1;
74 -   recdx = round(nx / nrec);
75 -   recy = srcy-20;
76 -   recz = srcz;
77 -   rec = zeros(nrec,n_iter);
78 -   %EM field dimensions
79 -   Hx = zeros(nx,ny,nz);
80 -   Hy = zeros(nx,ny,nz);
81 -   Hz = zeros(nx,ny,nz);
82 -   Ex = zeros(nx,ny,nz);
83 -   Ey = zeros(nx,ny,nz);
84 -   Ez = zeros(nx,ny,nz);
85 -   %iteration
86 -   i = 0;
87 -   for n=1:1:n_iter
88 -     %magnetic field derivatives
89 -     Hxy = diff(Hx,1,2);
90 -     Hxz = diff(Hx,1,3);
91 -     Hzx = diff(Hz,1,1);
92 -     Hzy = diff(Hz,1,2);
93 -     Hyx = diff(Hy,1,1);
94 -     Hyz = diff(Hy,1,3);
95 -

```

```

96 %electric field maxwell equations
97 - epsi = eps(:,2:end-1,2:nz-1);
98 - ksi = (dt * sigma(:,2:end-1,2:nz-1)) ./ ( 2 * epsi );
99 - c2 = (1./(1+ksi)).*(dt./epsi);
100 - cl = (1-ksi)./(1+ksi);
101 - Ex(:,2:end-1,2:end-1) = cl.*Ex(:,2:end-1,2:nz-1) + c2.*((1/dy)*Hzy(:,1:end-1,2:end-1) - (1/dz)*Hyz(:,2:ny-1,1:end-1));
102 -
103 - epsi = eps(2:end-1,:,2:end-1);
104 - ksi = (dt * sigma(2:end-1,:,2:end-1)) ./ ( 2 * epsi );
105 - c2 = (1./(1+ksi)).*(dt./epsi);
106 - cl = (1-ksi)./(1+ksi);
107 - Ey(2:end-1,:,2:end-1) = cl.*Ey(2:end-1,:,2:end-1) + c2.*((1/dz)*Hxz(2:end-1,:,1:end-1) - (1/dx)*Hzx(1:end-1,:,2:end-1));
108 -
109 - epsi = eps(2:end-1,2:end-1,:);
110 - ksi = (dt * sigma(2:end-1,2:end-1,:)) ./ ( 2 * epsi );
111 - c2 = (1./(1+ksi)).*(dt./epsi);
112 - cl = (1-ksi)./(1+ksi);
113 - Ez(2:end-1,2:end-1,:) = cl.*Ez(2:end-1,2:end-1,:) + c2.*((1/dx)*Hyx(1:end-1,2:end-1,:) - (1/dy)*Hxy(2:end-1,1:end-1,:));
114 -
115 %gaussian source
116 %f(n) = sin(2*pi*f0*n*dt)*exp(-(n*dt-t0)^2/(tw^2))/dy;
117 - f(n) = -2*(n*dt-t0)/tw*exp(-(n*dt-t0)^2/(tw^2))/dy;
118 - Ez(srcx,srcy,srcz) = Ez(srcx,srcy,srcz) + f(n);
119 - %Ezn(n)=Ez(srcx,srcy,srcz);
120 -
121 %electric field derivatives
122 - Exy = diff(Ex,1,2);
123 - Exz = diff(Ex,1,3);
124 - Ezx = diff(Ez,1,1);
125 - Ezy = diff(Ez,1,2);
126 - Eyx = diff(Ey,1,1);
127 - Eyz = diff(Ey,1,3);
128 -
129 %magnetic field maxwell equations
130 - Hx(:,1:end-1,1:end-1) = Hx(:,1:end-1,1:end-1) - (dt/(mu0*dy))*Ezy(:,1:end-1) + (dt/(mu0*dz))*Eyz(:,1:end-1,:);
131 - Hy(1:end-1,:,1:end-1) = Hy(1:end-1,:,1:end-1) - (dt/(mu0*dz))*Exz(1:end-1,:,:) + (dt/(mu0*dx))*Ezx(:,1:end-1);
132 - Hz(1:end-1,1:end-1,:) = Hz(1:end-1,1:end-1,:) - (dt/(mu0*dx))*Eyx(:,1:end-1,:) + (dt/(mu0*dy))*Exy(1:end-1,:,:);
133 -
134 - for k=1:l:nrec
135 - rec(k,n) = Ez(recdx * k, recy, recz);
136 - end
137

```

```

138 %display
139 - if (mod(i,5)==0)
140 -     slice(:,:,)=Ez(:,:,srcz);
141 -     pcolor(slice');
142 -     colorbar;
143 -     shading interp
144 -     drawnow
145 - end
146 - i = i+1;
147 - disp(i)
148 - end
149 -
150 - close all
151 - hold on
152 - for k=1:l:nrec
153 - plot(rec(k,:))
154 - end
155 -
156 - trec = rec;
157 - save('withtumor.mat','trec','nrec','n_iter','recy','recdx','recz')
158

```

## EK B Paralel (GPU Üzerinde) Çalışan Kodlar

```
TR_prl.m  WithoutTumor_prl.m  WithTumor_prl.m  + ]
```

```
1 %physical constants
2 - clear all;
3 - close all;
4 - load 'withtumor.mat';
5 - load 'withouttumor.mat';
6
7 - c0      = 2.998e8;
8 - eta0   = 120*pi;
9 - mu0    = pi*4e-7;
10 - eps0   = 1e-9/(36*pi);
11 %box dimensions
12 - width   = 0.5; % 30cm
13 - height  = 0.5;
14 - length  = 0.5; % 1cm
15 %source parameters
16 - f0      = 1e9; % GHz
17 - band    = 2e9;
18 - tw      = sqrt(-log(0.1)/(pi*band)^2);%le-8/pi;
19 - t0      = 4*tw;
20 %spatial discretization
21 - adipose = 5;
22 - tumor   = 10;
23 - sigma   = 5;
24 - epsr    = tumor;
25 - w       = 2 * pi * band;
26 - k       = (w/c0)*sqrt(epsr-lj*sigma/(w*eps0));
27 - beta    = real(k);
28 - c       = w / beta;
29 - lambda  = c/f0;
30 - dxmax   = lambda / 20;
31 - dx      = dxmax;
32 - dy      = dx;
33 - dz      = dx;
34 - nx      = round(width/dx);
35 - ny      = round(height/dy);
36 - nz      = round(length/dz);
37
38 %source position
39 - srcx   = round(nx / 2);
40 - srcy   = round( 3 * ny / 4);
41 - srcz   = round(nz / 2);
42
43 % material
44 - eps = gpuArray.ones(nx,ny,nz) * eps0; %* adipose;
45 - sigma = gpuArray.zeros(nx,ny,nz);% * f0 * le-9 * 0.5 - 0.5;
46 %temporal discretization
47 - dt    = 0.99/(c0*sqrt(dx^-2+dy^-2+dz^-2));
```

```

49 -     recl = trec - rec;
50 -     tau = 100e-12;
51 -     [foo,tp] = max(abs(recl),[],2);
52 -     for k=1:l:nrec
53 -         recn(k,:)=exp(-((dt*((1:l:n_iter)-tp(k))/tau).^2).*recl(k,:));
54 -     end
55 -     % hold on
56 -     % plot(rec(15,:))
57 -     % plot(exp(-((dt*((1:l:n_iter)-tp(15))/tau).^2))
58 -     % draw now
59 -     %
60 -     % while l
61 -     % end
62 -
63
64     %EM field dimensions
65 -     Hx = gpuArray.zeros(nx,ny,nz);
66 -     Hy = gpuArray.zeros(nx,ny,nz);
67 -     Hz = gpuArray.zeros(nx,ny,nz);
68 -     Ex = gpuArray.zeros(nx,ny,nz);
69 -     Ey = gpuArray.zeros(nx,ny,nz);
70 -     Ez = gpuArray.zeros(nx,ny,nz);
71 -     %iteration
72 -     i = 0;
73 -     for n=1:l:n_iter
74 -         %magnetic field derivatives
75 -         Hxy_cpu = diff(Hx,1,2);
76 -         Hxz_cpu = diff(Hx,1,3);
77 -         Hzx_cpu = diff(Hz,1,1);
78 -         Hzy_cpu = diff(Hz,1,2);
79 -         Hyx_cpu = diff(Hy,1,1);
80 -         Hyz_cpu = diff(Hy,1,3);
81 -         Hxy = gpuArray(Hxy_cpu);
82 -         Hxz = gpuArray(Hxz_cpu);
83 -         Hzx = gpuArray(Hzx_cpu);
84 -         Hzy = gpuArray(Hzy_cpu);
85 -         Hyx = gpuArray(Hyx_cpu);
86 -         Hyz = gpuArray(Hyz_cpu);
87 -         %electric field maxwell equations
88 -         epsi = eps(:,2:end-1,:,nz-1);
89 -         ksi = (dt * sigma(:,2:end-1,2:nz-1)) ./ ( 2 * epsi );
90 -         c2 = (1./(1+ksi)).*(dt./epsi);
91 -         cl = (1-ksi)./(1+ksi);
92 -         Ex(:,2:end-1,2:end-1) = cl.*Ex(:,2:end-1,2:nz-1) - c2.*((1/dy)*Hzy(:,1:end-1,2:end-1) - (1/dz)*Hyz(:,2:ny-1,1:end-1));
93 -
94
95 -         epsi = eps(2:end-1,:,2:end-1);
96 -         ksi = (dt * sigma(2:end-1,:,2:end-1)) ./ ( 2 * epsi );
97 -         c2 = (1./(1+ksi)).*(dt./epsi);
98 -         cl = (1-ksi)./(1+ksi);
99 -         Ey(2:end-1,:,2:end-1) = cl.*Ey(2:end-1,:,2:end-1) - c2.*((1/dz)*Hxz(2:end-1,:,1:end-1) - (1/dx)*Hxx(1:end-1,:,2:end-1));
100

```

```

101 -
102 -     epsi = eps(2:end-1,2:end-1,:);
103 -     ksi = (dt * sigma(2:end-1,2:end-1,:)) ./ ( 2 * epsi );
104 -     c2 = (1./(1+ksi)).*(dt./epsi);
105 -     c1 = (1-ksi)./(1+ksi);
106 -     Ez(2:end-1,2:end-1,:) = c1.*Ez(2:end-1,2:end-1,:) - c2.*((1/dx)*Hyx(1:end-1,2:end-1,:) - (1/dy)*Hxy(2:end-1,1:end-1,:));
107 -
108 - %TR sources
109 - for k=1:rec
110 -     Ez(recdx * k, recy, recz) = Ez(recdx * k, recy, recz) + recn(k, n_iter-n+1);
111 - end
112 - %rec(l,n_iter-n)
113 - %electric field derivatives
114 - Exy_cpu = diff(Ex,1,2);
115 - Exz_cpu = diff(Ex,1,3);
116 - Ezx_cpu = diff(Ez,1,1);
117 - Ezy_cpu = diff(Ez,1,2);
118 - Eyx_cpu = diff(Ey,1,1);
119 - Eyz_cpu = diff(Ey,1,3);
120 - Exy = gpuArray(Exy_cpu);
121 - Exz = gpuArray(Exz_cpu);
122 - Ezx = gpuArray(Ezx_cpu);
123 - Ezy = gpuArray(Ezy_cpu);
124 - Eyx = gpuArray(Eyx_cpu);
125 - Eyz = gpuArray(Eyz_cpu);
126 -
127 - %magnetic field maxwell equations
128 - Hx(:,l:end-1,l:end-1) = Hx(:,l:end-1,l:end-1) + (dt/(mu0*dy))*Ezy(:, :, l:end-1) - (dt/(mu0*dz))*Eyz(:, :, l:end-1,:);
129 - Hy(l:end-1,:,l:end-1) = Hy(l:end-1,:,l:end-1) + (dt/(mu0*dz))*Exz(l:end-1,:,:) - (dt/(mu0*dx))*Ezx(:, :, l:end-1);
130 - Hz(l:end-1,l:end-1,:) = Hz(l:end-1,l:end-1,:) + (dt/(mu0*dx))*Eyx(:, :, l:end-1) - (dt/(mu0*dy))*Exy(:, :, l:end-1,:);
131 -
132 - %display
133 - if (mod(n,10)==0)
134 -     slice(:,:,)=Ez(60:100,round(ny/2)-20:round(ny/2)+20,srcz);
135 -     pcolor(slice.');
136 -     colorbar;
137 -     shading interp
138 -     drawnow
139 - end
140 - i = i+1;
141 - disp(i)
142 -
143 - R(n) = varimax_norm(Ez(60:100,round(ny/2)-20:round(ny/2)+20,srcz));
144 - end
145 -
146 - figure;plot(R)
147 -
148 - function R = varimax_norm(Ez)
149 -     R = sum(sum(sum(Ez.^2)))^2 / sum(sum(sum(Ez.^4)));
150 - end
151 -

```

```

TR_prl.m WithoutTumor_prl.m WithTumor_prl.m +
1 %physical constants
2 - clear all;
3 - close all;
4 - c0 = 2.998e8;
5 - eta0 = 120*pi;
6 - mu0 = pi*4e-7;
7 - eps0 = 1e-9/(36*pi);
8 %box dimensions
9 - width = 0.5; % cm
10 - height = 0.5;
11 - length = 0.5; % cm
12 %source parameters
13 - f0 = 1e9; % GHz
14 - band = 2e9;
15 - tw = sqrt(-log(0.1)/(pi*band)^2);%1e-8/pi;
16 - t0 = 4*tw;
17 %spatial discretization
18 - adipose = 1; %5;
19 - sigma = 5;
20 - epsr = 10;
21 - w = 2 * pi * band;
22 - k = (w/c0)*sqrt(epsr-lj*sigma/(w*eps0));
23 - beta = real(k);
24 - c = w / beta;
25 - lambda = c/f0;
26 - dxmax = lambda / 20;
27 - dx = dxmax;
28 - dy = dxmax;
29 - dz = dxmax;
30 - nx = round(width/dx);
31 - ny = round(height/dy);
32 - nz = round(length/dz);
33 %source position
34 - srcx = round(nx / 2);
35 - srcy = round( 3 * ny / 4);
36 - srcz = round(nz / 2);
37 %material
38 - al = 0;
39
40 - eps = gpuArray.ones(nx,ny,nz) * eps0 ;
41 - sigma = gpuArray.zeros(nx,ny,nz);%*f0 * 1e-9 * 0.5 - 0.5;
42 - for i=1:1:nx
43 - for j=1:1:ny
44 - for k=1:1:nz
45 - % adipose tissue is located under z < al
46 - if (k<al)
47 -     eps(i,j,k) = eps0 * adipose;
48 -     sigma(i,j,k) = f0 * 1e-9 * 0.5 - 0.5;
49 - end
50 - end
51 - end
52 - end

```

```

53      %time discretization
54      dt = 0.99/(c0*sqrt(dx^2+dy^2+dz^2));
55
56      tw=16*dt;
57      t0=3*tw;
58
59      n_iter = 250;
60      %receivers
61      nrec = round(nx / 3)-1;
62      recdx = round(nx / nrec);
63      recy = srcy-20;
64      recz = srcz;
65      rec = gpuArray.zeros(nrec,n_iter);
66      %EM field dimensions
67      Hx = gpuArray.zeros(nx,ny,nz);
68      Hy = gpuArray.zeros(nx,ny,nz);
69      Hz = gpuArray.zeros(nx,ny,nz);
70      Ex = gpuArray.zeros(nx,ny,nz);
71      Ey = gpuArray.zeros(nx,ny,nz);
72      Ez = gpuArray.zeros(nx,ny,nz);
73      %iteration
74      i = 0;
75      for n=1:l:n_iter
76          %magnetic field derivatives
77          Hxy_cpu = diff(Hx,1,2);
78          Hxz_cpu = diff(Hx,1,3);
79          Hzx_cpu = diff(Hz,1,1);
80          Hzy_cpu = diff(Hz,1,2);
81          Hyx_cpu = diff(Hy,1,1);
82          Hyz_cpu = diff(Hy,1,3);
83          Hxy = gpuArray(Hxy_cpu);
84          Hxz = gpuArray(Hxz_cpu);
85          Hzx = gpuArray(Hzx_cpu);
86          Hzy = gpuArray(Hzy_cpu);
87          Hyx = gpuArray(Hyx_cpu);
88          Hyz = gpuArray(Hyz_cpu);
89
90          %electric field maxwell equations
91          epsi = eps(:,2:end-1,2:sz-1);
92          ksi = (dt * sigma(:,2:end-1,2:sz-1)) ./ ( 2 * epsi );
93          c2 = (1./(l+ksi)).*(dt./epsi);
94          cl = (1-ksi)./(l+ksi);
95          Ex(:,2:end-1,2:end-1) = cl.*Ex(:,2:end-1,2:sz-1) + c2.*((1/dy)*Hzy(:,1:end-1,2:end-1) - (1/dz)*Hyz(:,2:ny-1,1:end-1));
96
97          epsi = eps(2:end-1,:,2:end-1);
98          ksi = (dt * sigma(2:end-1,:,2:end-1)) ./ ( 2 * epsi );
99          c2 = (1./(l+ksi)).*(dt./epsi);
100         cl = (1-ksi)./(l+ksi);
101         Ey(2:end-1,:,:2:end-1) = cl.*Ey(2:end-1,:,:2:end-1) + c2.*((1/dz)*Hxz(2:end-1,:,1:end-1) - (1/dx)*Hzx(1:end-1,:,2:end-1));
102

```

```

103 -
104 -      epsi = eps(2:end-1,2:end-1,:);
105 -      ksi = (dt * sigma(2:end-1,2:end-1,:)) ./ ( 2 * epsi );
106 -      c2 = (1./(1+ksi)).*(dt./epsi);
107 -      c1 = (1-ksi)./(1+ksi);
108 -      Ez(2:end-1,2:end-1,:) = c1.*Ez(2:end-1,2:end-1,:) + c2.*((1/dx)*Hyx(1:end-1,2:end-1,:)) - (1/dy)*Hxy(2:end-1,1:end-1,:));
109 -
110 -      %gaussian source
111 -      %f(n) = sin(2*pi*f0*n*dt)*exp(-(n*dt-t0)^2/(tw^2))/dy;
112 -      f(n) = -2*(n*dt-t0)/tw*exp(-(n*dt-t0)^2/(tw^2))/dy;
113 -      Ez(srcx,srcy,srcz) = Ez(srcx,srcy,srcz) + f(n);
114 -      %Ezn(n)=Ez(srcx,srcy,srcz);
115 -
116 -      %electric field derivatives
117 -      Exy_cpu = diff(Ex,1,2);
118 -      Exz_cpu = diff(Ex,1,3);
119 -      Ezx_cpu = diff(Ez,1,1);
120 -      Ezy_cpu = diff(Ez,1,2);
121 -      Eyx_cpu = diff(Ey,1,1);
122 -      Eyz_cpu = diff(Ey,1,3);
123 -      Exy = gpuArray(Exy_cpu);
124 -      Exz = gpuArray(Exz_cpu);
125 -      Ezx = gpuArray(Ezx_cpu);
126 -      Ezy = gpuArray(Ezy_cpu);
127 -      Eyx = gpuArray(Eyx_cpu);
128 -      Eyz = gpuArray(Eyz_cpu);
129 -
130 -      %magnetic field maxwell equations
131 -      Hx(:,1:end-1,1:end-1) = Hx(:,1:end-1,1:end-1) - (dt/(mu0*dy))*Ezy(:,:,1:end-1) + (dt/(mu0*dz))*Eyz(:,:,1:end-1,:);
132 -      Hy(1:end-1,:,1:end-1) = Hy(1:end-1,:,1:end-1) - (dt/(mu0*dz))*Exz(1:end-1,:,:) + (dt/(mu0*dx))*Ezx(:,:,1:end-1,:);
133 -      Hz(1:end-1,1:end-1,:) = Hz(1:end-1,1:end-1,:) - (dt/(mu0*dx))*Eyx(:,:,1:end-1) + (dt/(mu0*dy))*Exy(1:end-1,:,:);
134 -      for k=1:l:nrec
135 -          rec(k,n) = Ez(recdx * k, recy, recz);
136 -      end
137

```

```

138      %display
139 -      if (mod(i,5)==0)
140 -          slice(:,:,)=Ez(:,:,srcz);
141 -          pcolor(slice');
142 -          colorbar;
143 -          shading interp
144 -          drawnow
145 -      end
146 -      i = i+1;
147 -      disp(i)
148 -  end
149
150 -  close all
151 -  hold on
152 -  for k=1:l:nrec
153 -      plot(rec(k,:))
154 -  end
155
156 -  save('withouttumor.mat','rec','nrec','n_iter','recy','recdx','recz')
157

```

```

TR_prl.m  WithoutTumor_prl.m  WithTumor_prl.m  +
1 %physical constants
2 clear all;
3 close all;
4 c0 = 2.998e8;
5 eta0 = 120*pi;
6 mu0 = pi*4e-7;
7 eps0 = 1e-9/(36*pi);
8 %box dimensions
9 width = 0.5; % cm
10 height = 0.5;
11 length = 0.5; % cm
12 %source parameters
13 f0 = 1e9; % GHz
14 band = 2e9;
15 %tw = sqrt(-log(0.1)/(pi*band)^2);%le-8/pi;
16 %spatial discretization
17 adipose = 1; %5;
18 tumor = 10;
19 sigma = 5;
20 epsr = tumor;
21 w = 2 * pi * band;
22 k = (w/c0)*sqrt(epsr-lj*sigma/(w*eps0));
23 beta = real(k);
24 c = w / beta;
25 lambda = c/f0;
26 dxmax = lambda / 20;
27 dx = dxmax;
28 dy = dxmax;
29 dz = dxmax;
30 nx = round(width/dx);
31 ny = round(height/dy);
32 nz = round(length/dz);
33 %source position
34 srcx = round(nx / 2);
35 srcy = round( 3 * ny / 4);
36 srcz = round(nz / 2);
37 %material
38 mw = 4; % width
39 mh = 4; % height
40 ml = 4; % length
41 %mx = nx / 4-10-mw/2;
42 mx = nx / 2-10-mw/2;
43 my = ny / 2-mh/2;
44 mz = nz / 2-ml/2;
45
46 al = 0;
47
48 eps = gpuArray.ones(nx,ny,nz) * eps0 ;
49 sigma = gpuArray.zeros(nx,ny,nz);%*f0 * le-9 * 0.5 - 0.5;

```

```

50 -    for i=1:1:nx
51 -        for j=1:1:ny
52 -            for k=1:1:nz
53 -                % adipose tissue is located under z < al
54 -                if (k<al)
55 -                    eps(i,j,k) = eps0 * adipose;
56 -                    sigma(i,j,k) = f0 * 1e-9 * 0.5 - 0.5;
57 -                end
58 -                if (i>mx && i<(mw+mx) && j>my && j<(mh+my) && k>mz && k<(ml+mz))
59 -                    eps(i,j,k) = eps0 * tumor;
60 -                    sigma(i,j,k) = f0 * 1e-9 - 0.5;
61 -                end
62 -            end
63 -        end
64 -    end
65 -    %time discretization
66 -    dt = 0.99/(c0*sqrt(dx^2+dy^2+dz^2));
67 -
68 -    tw=16*dt;
69 -    t0=3*tw;
70 -
71 -    n_iter = 250;
72 -    %receivers
73 -    nrec = round(nx / 3)-1;
74 -    recdx = round(nx / nrec);
75 -    recy = srcy-20;
76 -    recz = srcz;
77 -    rec = gpuArray.zeros(nrec,n_iter);
78 -    %EM field dimensions
79 -    Hx = gpuArray.zeros(nx,ny,nz);
80 -    Hy = gpuArray.zeros(nx,ny,nz);
81 -    Hz = gpuArray.zeros(nx,ny,nz);
82 -    Ex = gpuArray.zeros(nx,ny,nz);
83 -    Ey = gpuArray.zeros(nx,ny,nz);
84 -    Ez = gpuArray.zeros(nx,ny,nz);
85 -    %iteration
86 -    i = 0;
87 -    for n=1:1:n_iter
88 -        %magnetic field derivatives
89 -        Hxy_cpu = diff(Hx,1,2);
90 -        Hxz_cpu = diff(Hx,1,3);
91 -        Hzx_cpu = diff(Hz,1,1);
92 -        Hzy_cpu = diff(Hz,1,2);
93 -        Hyx_cpu = diff(Hy,1,1);
94 -        Hyz_cpu = diff(Hy,1,3);
95 -        Hxy = gpuArray(Hxy_cpu);
96 -        Hxz = gpuArray(Hxz_cpu);
97 -        Hzx = gpuArray(Hzx_cpu);
98 -        Hzy = gpuArray(Hzy_cpu);
99 -        Hyx = gpuArray(Hyx_cpu);
100 -       Hyz = gpuArray(Hyz_cpu);
101

```

```

102 %electric field maxwell equations
103 - epsi = eps(:,2:end-1,2:nz-1);
104 - ksi = (dt * sigma(:,2:end-1,2:nz-1)) ./ ( 2 * epsi );
105 - c2 = (1./(1+ksi)).*(dt./epsi);
106 - cl = (1-ksi)./(1+ksi);
107 - Ex(:,2:end-1,2:end-1) = cl.*Ex(:,2:end-1,2:nz-1) + c2.*((1/dy)*Hzy(:,1:end-1,2:end-1) - (1/dz)*Hyz(:,2:ny-1,1:end-1));
108 -
109 - epsi = eps(2:end-1,:,:2:end-1);
110 - ksi = (dt * sigma(2:end-1,:,:2:end-1)) ./ ( 2 * epsi );
111 - c2 = (1./(1+ksi)).*(dt./epsi);
112 - cl = (1-ksi)./(1+ksi);
113 - Ey(2:end-1,:,:2:end-1) = cl.*Ey(2:end-1,:,:2:end-1) + c2.*((1/dz)*Hxz(2:end-1,:,:1:end-1) - (1/dx)*Hzx(1:end-1,:,:2:end-1));
114 -
115 - epsi = eps(2:end-1,2:end-1,:);
116 - ksi = (dt * sigma(2:end-1,2:end-1,:)) ./ ( 2 * epsi );
117 - c2 = (1./(1+ksi)).*(dt./epsi);
118 - cl = (1-ksi)./(1+ksi);
119 - Ez(2:end-1,2:end-1,:) = cl.*Ez(2:end-1,2:end-1,:) + c2.*((1/dx)*Hyx(1:end-1,2:end-1,:) - (1/dy)*Hxy(2:end-1,1:end-1,:));
120 -
121 %gaussian source
122 - f(n) = sin(2*pi*f0*n*dt)*exp(-(n*dt-t0)^2/(tw^2))/dy;
123 - f(n) = -2*(n*dt-t0)/tw*exp(-(n*dt-t0)^2/(tw^2))/dy;
124 - Ez(srcx,srcy,srcz) = Ez(srcx,srcy,srcz) + f(n);
125 - %Ezn(n)=Ez(srcx,srcy,srcz);
126 -
127 %electric field derivatives
128 - Exy_cpu = diff(Ex,1,2);
129 - Exz_cpu = diff(Ex,1,3);
130 - Ezx_cpu = diff(Ez,1,1);
131 - Ezy_cpu = diff(Ez,1,2);
132 - Eyx_cpu = diff(Ey,1,1);
133 - Eyz_cpu = diff(Ey,1,3);
134 - Exy = gpuArray(Exy_cpu);
135 - Exz = gpuArray(Exz_cpu);
136 - Ezx = gpuArray(Ezx_cpu);
137 - Ezy = gpuArray(Ezy_cpu);
138 - Eyx = gpuArray(Eyx_cpu);
139 - Eyz = gpuArray(Eyz_cpu);
140 -
141 %magnetic field maxwell equations
142 - Hx(:,1:end-1,1:end-1) = Hx(:,1:end-1,1:end-1) - (dt/(mu0*dy))*Ezy(:,1:end-1) + (dt/(mu0*dz))*Eyz(:,1:end-1,:);
143 - Hy(1:end-1,:,1:end-1) = Hy(1:end-1,:,1:end-1) - (dt/(mu0*dz))*Exz(1:end-1,:,1:end-1) + (dt/(mu0*dx))*Ezx(:,1:end-1,:);
144 - Hz(1:end-1,1:end-1,:) = Hz(1:end-1,1:end-1,:) - (dt/(mu0*dx))*Eyx(:,1:end-1,:) + (dt/(mu0*dy))*Exy(1:end-1,:,:);
145

```

```

146 - for k=1:l:nrec
147 -     rec(k,n) = Ez(recdx * k, recy, recz);
148 - end
149
150     %display
151 - if (mod(i,5)==0)
152 -     slice(:,:,1)=Ez(:,:,srcz);
153 -     pcolor(slice');
154 -     colorbar;
155 -     shading interp
156 -     drawnow
157 - end
158 - i = i+1;
159 - disp(i)
160 end
161
162 - close all
163 - hold on
164 - for k=1:l:nrec
165 -     plot(rec(k,:))
166 - end
167
168 - trec = rec;
169 - save('withtumor.mat','trec','nrec','n_iter','recy','recdx','recz')
170

```

## ÖZGEÇMİŞ

<b>Ad-Soyad</b>	: Cihan ÜRTEKİN
<b>Doğum Tarihi ve Yeri</b>	: 02.12.1995 / Tekirdağ
<b>E-posta</b>	: cihanurtekin@gmail.com

Cihan Ürtekin 1995 yılında Tekirdağ, Türkiye'de doğdu. Lise eğitimini 2009-2013 yılları arasında Tekirdağ Fen Lisesi'nde tamamladı. 2015-2019 yılları arasında İstanbul Teknik Üniversitesi, Elektronik ve Haberleşme Mühendisliği bölümünde öğrenim gördü.

## **ÖZGEÇMİŞ**

**Ad-Soyad** : Alper Said SOYLU  
**Doğum Tarihi ve Yeri** : 13.03.1995 / Ankara  
**E-posta** : soylu15@itu.edu.tr

Alper Said SOYLU 1995 yılında Ankara, Türkiye'de doğdu. 2013 yılında Sakarya Fatih Anadolu Lisesi'nden mezun oldu. 2015-2019 yılları arasında İstanbul Teknik Üniversitesi, Elektronik ve Haberleşme Mühendisliği bölümünde öğrenim gördü.