İYONKÜREDE DALGA YAYILIMI MODELLEMESİ İÇİN IŞIN İZLEME ALGORİTMALARININ GRAFİK İŞLEMCİ BİRİMLERİ İLE PARALEL İŞLENMESİ

GPU PARALLEL COMPUTING OF RAY TRACING ALGORITHMS FOR MODELING WAVE PROPAGATION IN THE IONOSPHERE

ATİLLA KAÇAR

DOÇ. DR. CENK TOKER
Tez Danışmanı

Hacettepe Üniversitesi
Lisansüstü Eğitim - Öğretim ve Sınav Yönetmeliği'nin
Elektrik ve Elektronik Mühendisliği Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ olarak hazırlanmıştır.

2018

ÖZET

İYONKÜREDE DALGA YAYILIMI MODELLEMESİ İÇİN IŞIN İZLEME ALGORİTMALARININ GRAFİK İŞLEMCİ BİRİMLERİ İLE PARALEL İŞLENMESİ

Atilla KAÇAR
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Danışmanı: Doç. Dr. Cenk TOKER
Nisan 2018, 70 sayfa

Kısa Dalga (KD) haberleşmesinde uzak mesafe sinyal iletimi iyonküre üzerinden gerçekleştirilmektedir. İyonküre, yeryüzünden yaklaşık 80 km ile 1100 km yükseklik arasında kalan alanı belirtir ve çoğunlukla iyonlar ve serbest elektronlardan oluşmaktadır. İyonküredeki bu plazma ortamı içinden geçen elektromanyetik dalgalarla bu plazma ortamı etkileşime girerek dalganın yönü, gücü, hızı ve polarizasyonu etkilenir. İyonküredeki elektron yoğunluğu coğrafi konum, zaman, yükseklik ve sıcaklık gibi birçok etmene bağlı olarak sürekli değişim göstermektedir.

KD haberleşmesinde vericiden çıkan sinyal iyonküreye girdikten sonra burada kırılım ve saçılım mekanizmaları yoluyla eğri bir yol üzerinden alıcıya ulaşır. Bu tez çalışması ile KD haberleşmesinde kullanılan ışın izleme algoritmalarının grafik işlemcilerle paralel programlanabilmesi hedeflenmiştir. Yapılan çalışmalarda ilk olarak, birbirinden bağımsız gönderilecek sinyallerin iyonküre içinde ışın izleme algoritmaları ile ilerleme işlemleri incelenmiştir. Sonrasında, kırılım ve saçılım modelleri paralel programlama için yeniden gözden geçirilip, yazılımsal faaliyetler gerçekleştirilmiştir. Bu faaliyetler sonucunda elde edilen verilerin tutarlılığı test edilip, süre ölçümleri yapılmıştır. Yapılan çalışmalar sonrasında grafik işlemciler ile paralel programlama faaliyetlerinden elde edilen kazanımlara değinilmiştir.

AnahtarKelimeler: KD haberleşme, ışın izleme, paralel programlama

ABSTRACT

GPU PARALLEL COMPUTING OF RAY TRACING ALGORITHMS FOR MODELING WAVE PROPAGATION IN THE IONOSPHERE

Atilla KAÇAR

Master Thesis, Department of Electrical And Electronics Engineering

Supervisor: Assoc. Prof. Cenk TOKER

April 2018, 70 pages

In long range short wave (SW) communications, the transmitted signal passes through the ionosphere. Ionosphere approximately covers the area between about 80 kms and 1100 kms above the earth's surface and it is mostly composed of ions and free electrons. This plasma medium in the ionosphere interacts with the electromagnetic wave passing through it and changes the path, strength, velocity and polarisation of the wave. Electron density in the ionosphere constantly changes depending on many factors such as geographical position, time, altitude and temperature.

In SW communication, the transmitted signal reaches the receiver by following a curved path which is determined by the diffraction and scattering mechanisms caused by the ionosphere. The aim of this thesis study is investigate propagation of the to parallelize the ray tracing algorithms used in the short wave communication with graphics processors. First, the propagation of the independently transmitted signals through the ionosphere is investigated by ray tracing algorithms. After that, diffraction and dispersion models are reviewed for parallel programming and software activities are performed. Finally, the consistency of the result derived from software activities are tested and time measurements are made. As a result, the achievements obtained from the parallel programming activities with the graphics processors are mentioned.

Key words: Short wave communication, ray tracing, parallel programming

TEŞEKKÜR

Tez çalışmam boyunca bilgi ve deneyimleri ile bana daima yol gösteren ve desteğini esirgemeyen tez danışmanım Sayın Doç. Dr. Cenk TOKER'e teşekkür ederim.

TÜBİTAK 112E568 ve 114E092 projesi kapsamında desteklenen IONOLAB-RAY çalışmalarını benimle paylaşan ve çalışmalarımda yardımcı olan Sayın Dr. Esra ERDEM'e teşekkür ederim.

Ülkemizin en büyük savunma sanayi firması olan ve lisansüstü çalışmalarımı destekleyen şirketim ASELSAN'a teşekkür ederim.

Hayatım boyunca hep arkamda durup bana destek olan, sevgilerini her zaman en derinden hissettiğim, canımdan çok sevdiklerim annem Şeyma, babam Menderes ve kız kardeşlerim Nurten ve Elif Nur'a en derin teşekkürlerimi sunarım.

Son olarak, hayatımın mihenk taşlarından birisi olan, yaşamıma anlam katan, yıllardır bir an olsun yanımdan ayrılmayan sevgili eşim Sinem'e sevgilerimi sunarım.

İÇİNDEKİLER

Sayfa

ŞEKİLLER

Sayfa

ÇİZELGELER

Sayfa

SİMGELER VE KISALTMALAR

ALU	Arithmetic Logic Unit
CPU	Central Processing Unit
DKY	Doğu Kuzey Yukarı koordinat sistemi
EBY	Enlem Boylam Yükseklik koordinat sistemi
GPS	Global Positioning System
GPU	Graphics Processing Unit
GS	Greenwich Saati
IGRF	International Geomagnetic Reference Field
IONORT	Ionospheric Ray Tracing
IRI	International Reference Ionosphere
KD	Kısa Dalga
LAT	Latitude
LON	Longitude
PHaRLAP	HF radio-wave raytracing toolbox
TEİ	Toplam Elektron İçeriği
YMYS	Yerküre Merkezli Yerküre Sabit koordinat sistemi
3B	3 Boyutlu

SÖZLÜK DİZİNİ

Grafik İşlemci Birimi	: Graphics Processing Unit
Greenwich Saati	: Universal Time (UT)
Havaküre	: Atmosfer
IONOLAB	: İyonosfer Araştırma Laboratuarı
İyonküre	: İyonosfer
Konsantrasyon	: Derişme
Merkezi İşlemci Birimi	: Central Processing Unit
Polarizasyon	: Kutuplanma
Ray	: Işın
Yanca	: Azimuth
Yayılım	: Propagation
Yükseliş	: Elevation
Yerküresel Konumlama Sistemi (YKS)	: Global Positioning System (GPS)

1. GİRİŞ

İyonküre, dünyayı çevreleyen atmosferin bir katmanı olup, güneşin etkisiyle iyonlaşmış gazlardan oluşmaktadır. Uzak mesafe haberleşmesi için kullanılan Kısa Dalga (KD) bandındaki haberleşme sistemlerinde vericiden çıkan sinyal, iyonkürede ilerlerken farklı elektron yoğunluğuna sahip bölgelerin içinden geçmektedir. İyonkürenin sahip olduğu elektron ortam yoğunluğu yükseklik, coğrafi konum, hava sıcaklığı, gece/gündüz durumu, mevsim, zaman gibi birçok etmene bağlı olarak değişmektedir [1]. Ortam yoğunluğundaki bu değişim, gönderilen dalganın maruz kaldığı yayılım mekanizmasının (kırılma, saçılma, yansıma) konum bağımlı olmasını getirmektedir. Ortam koşullarına bağlı olarak her kırılım ve saçılım sonrasında dalganın yönü değişerek ilerleyebilmekte ve vericiden çıkan hüzmenin bir kısmı alıcıya ulaşabilmekte, geri kalanı ise coğrafi olarak farklı bölgelere ulaşabilmekte veya uzaya kaçabilmektedir. İyonküredeki elektron yoğunluğu, sıcaklık, gece veya gündüz olması, hava durumu gibi birçok etmen modellenerek gönderilen sinyalin alıcıya ulaşma koşulları hesaplanabilmektedir [1].

İyonküre içerisindeki, yoğunluğu zaman ve konuma göre değişen iyonlaşmış gazların yaratmış olduğu karmaşık ortam nedeniyle elektromanyetik dalga yayılımı problemlerini çözmek oldukça zor olmaktadır. Mevcut bazı analitik ve nümerik yöntemler işlem yükü oldukça yüksek olan hesaplamalar içermektedir [2]. Tez kapsamında, IONOLAB [3] grubu çalışmaları kapsamında geliştirilen IONOLAB-RAY [4] isimli algoritmanın grafik işlemciler kullanılarak paralel programlanması sağlanmıştır. Bu tezde yapılan çalışmalar kapsamında, grafik işlemci üzerinde ışın izleme algoritması gerçeklenmiş ve iyonkürenin üç boyutlu küresel hücrelerden oluşturulmuş modeli, bu yazılıma girdi olarak verilerek daha gerçekçi sonuçlar elde edilmesi sağlanmıştır.

Işın izleme algoritmaları, belirli bir merkez ve yöne sahip doğrusal bir ışının üç boyutlu uzayda herhangi bir yoğunluğa girme noktasını ve girdikten sonra hangi yönde yoluna devam ettiğini hesaplamayı amaçlamaktadır. Işın izleme hesaplamaları genel olarak görüntü işleme, oyun ve görüntü modelleme işlemlerinde kullanılmaktadır. Tez çalışmaları kapsamında iyonkürede ışın izlemeye yönelik literatürdeki çalışmalar araştırılmıştır. PHaRLAP [5], IONORT ve IONOLAB-RAY çalışmaları incelenmiştir. IONOLAB-RAY'de dalga yayılım modeli için ışın izleme yöntemlerinden birisi olan Snell Yasası uygulanmaktadır. Snell Yasası içeriğinde kullanılan anahtar işlemlerden birisi kırılma indisi hesaplamasıdır. Bu kırılma indisi hesabında da Appleton-Hartree Eşitliği kullanılmaktadır.

Appleton-Hartree Eşitliğinde iyonkürenin fiziksel parametreleri dahil edilerek işlemler yapılmaktadır [6]. Geometrik optik yaklaşımı ile Snell Yasası'nın ışın izlemede kullanılması, algoritmalardaki işlem yükünün oldukça hafiflemesini ve işlem karmaşıklığının azaltılmasını sağlamaktadır. Bu avantajlarından dolayı IONOLAB-RAY algoritmasının tez kapsamında incelenip, paralelleştirilmesi çalışmaları yapılmıştır.

KD haberleşme sistemlerinde vericiden çıkan elektromanyetik dalganın ışınlar ile temsili düşünüldüğünde, verici ve alıcı arasında binlerce ışın ilerleyecek olup, her bir ışın için yukarıda bahsedilmiş olan işlemler uygulanmaktadır. Bu hesaplamaların yapılma süreleri modellenen iyonküre alanına ve gönderilen ışın sayısına göre değişim göstermektedir. IONOLAB-RAY'de her bir ışın için iyonkürede farklı elektron yoğunluklu ortama girildiğinde Snell Kuralları işletilerek yön hesabı yapılmakta ve sinyalin iyonkürede nereden çıktığı tespit edilmektedir. Gönderilmek istenen binlerce ışını benzetim yaparak ışın izleme işlemlerinden geçirmek ve hesaplamak, işlem ve işlemci yükü açısından bir hayli külfetli olabilmektedir. Sonucu görebilmek için beklenen uzun işlem süreleri karşılaşılan en büyük sorunlardan birisidir [4].

Işın izleme algoritmalarının iyonkürede birbirinden bağımsız ilerleyen ışınlara uygulanabilmesi nedeniyle bu işlemler çok verimli olarak çok çekirdekli işlemcilerde paralel programlanabilmektedir. Çok çekirdekli merkezi işlemci birimleri (CPU) ve grafik işlemci birimleri (GPU) bu konuda ön plana çıkan alternatiflerdendir [7]. Yapılan bu tez çalışmasında çok fazla çekirdeğe sahip olan grafik işlemci birimi kullanılmıştır. Bu grafik işlemci birimi ile binlerce işlem aynı anda yapılabilmektedir. Sahip olunan bu işlem gücü ve çekirdek sayısı, binlerce ışının aynı anda işlemlerden geçirilip, yayılımının hesaplanabilmesinin altyapısını oluşturmaktadır.

Grafik işlemciler son yıllarda giderek artan işlem güçleri ve çekirdek sayıları ile birçok paralel işleme alanında söz sahibi olmaya başlamıştır [8]. İlk olarak görüntü işleme alanında kullanılan ve bilgisayarların merkezi işlemci güçlerini rahatlatmak için kullanılan grafik işlemciler daha sonra birçok alanda kullanıma alınmıştır [9]. Büyük veri işleme, yapay zeka çalışmaları, derin öğrenme konusu, benzetim çalışmaları, sinyal işleme gibi konularda grafik işlemciler de kullanılmaktadır [10]. Ortalama bir CPU ile bir GPU karşılaştırıldığında en önemli farkın çekirdek sayıları olduğu görülmektedir. CPU'da 4, 8, 16, 24 gibi sınırlı sayıda çekirdekleri varken, GPU'larda çekirdek sayısı binlerle ifade edilmektedir. Örneğin; yapılmış olan bu tez çalışmasında kullanılan grafik işlemcisinin 2496 çekirdeği

bulunmaktadır. CPU çekirdeklerinin bire birde GPU çekirdeklerinden çok daha güçlü olması bilinen bir gerçektir. Fakat, birbirinden bağımsız binlerce işlem yapmak istenildiğinde GPU işlemcilerinin yüksek sayıda çekirdeğe sahip olması kullanıcılara büyük bir avantaj sağlamaktadır [11].

Yapılan bu tez çalışmasında ışın izleme algoritmalarının grafik işlemciler için uygunluğu görülüp, gerekli yerlerde de algoritmaların grafik işlemciye uygun hale getirilmesi sureti ile grafik işlemcilerin avantajlarından faydalanılmaya çalışılmıştır. Literatür çalışması ile başlayan araştırma sürecini ışın izleme algoritmalarını inceleme süreci takip etmiştir. Literatürdeki farklı ışın izleme algoritmaları arasından IONOLAB grubunda geliştirilmiş olan IONOLAB-RAY algoritmasının fonksiyonları kullanılmıştır [4]. Algoritmanın büyük çaplı benzetim çalışmaları için yeterince hızlı olmadığı ve binlerce ışının hesaplanması için oldukça fazla süre beklemek gerektiği tespit edilmiştir. Bu noktada algoritmaların paralellenebilirliği incelenip değerlendirilmiş ve yapılabilecek eniyileme çalışmaları ele alınmıştır.

Grafik işlemciler, merkezi işlemciler gibi dahil olduğu donanımın (bilgisayar, dizüstü bilgisayar, iş istasyonu vs.) bütün hafıza alanına erişememekte ve üzerinde işlem yapamamaktadır. Her bir grafik işlemcisinin kendi hafıza alanı olup, burada işlem yapabilmek için ilk önce verinin genel hafıza alanından bu alana transfer edilmesi gerekmektedir. Bu nedenle grafik işlemcinin hafıza alanına göre algoritma incelenmeli ve yazılım tasarımı yapılmalıdır. İnceleme işlemlerinden sonra tasarım aşamasına geçilerek, grafik işlemciler için hafıza alanı hesaplamaları ve yazılım mimarisi üzerine çalışmalar yapılmalıdır. Yapılan bu tez çalışmasında tüm bu nitelikler göz önüne alınarak tasarım yapılmış ve ardından yazılım faaliyetlerine geçilmiştir.

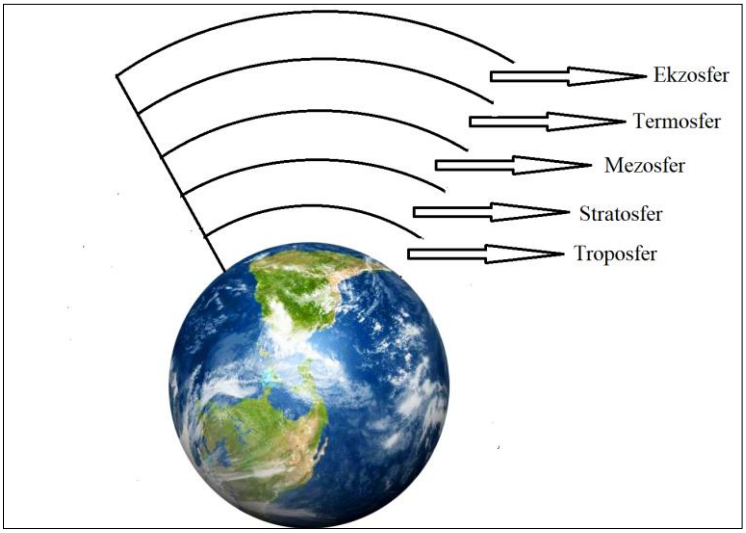
Elektron yoğunluğu verilmiş ve modellenmiş olan bir iyonküre bölgesinde, gönderilen ışının izlemiş olduğu yol, yansıyan ve yoluna devam eden ışın bilgileri, kırılım bilgileri ve son olarak düşüm bilgileri hesaplanarak doğruluk testleri yapılmıştır. MATLAB [12] ortamında kullanılan ışın izleme algoritmasının girdi verileri kaydedilerek yazılıma verilmiş, böylelikle hem yazılımda hem de MATLAB programında aynı veri ile aynı işlemler yapılarak aynı sonuçların görülmesi öngörülmüştür. Bir adet ışının işlemlerinin doğruluğu hesaplandıktan sonra 1024 adet sinyal için aynı işlemler yapılarak doğruluk testleri yapılmıştır. Bu aşamada hem MATLAB ortamının hem de grafik işlemcide çalıştırılmak üzere hazırlanmış olan CUDA [13] derleyicisiyle elde edilen yazılımın aynı sonuçları verdiği görülmüştür. Bu tez

kapsamında CUDA derleyicisi ile yazılan kodlar için CUDA kodu veya CUDA yazılımı şeklinde nitelendirmeler yapılmıştır. Veri seti ve işlemlerin doğruluk kontrolleri yapıldıktan sonra bu tezin asıl amaçlarından biri olan süre ölçüm faaliyeti gerçekleştirilmiştir. MATLAB'da yaklaşık 2700 saniyede (45 dakika) gerçekleşen bu işlemler, grafik işlemcide çalıştırılan CUDA yazılımı ile yalnızca 1.3 saniyede tamamlanmaktadır. İşlem süresinde yaklaşık iki bin kat iyileşme olduğu görülmüştür. Yüzbinlerce ışın kullanılarak yapılacak olan ışın izleme algoritmalarında benzetim süresi saatler sürecekken, grafik işlemcide çalıştırılacak CUDA yazılımı ile bu sürenin saniyeler mertebesinde yapılacağı görülmüştür. Ayrıca, IONOLAB-RAY algoritmasının C kodu yazılarak CPU'da işlem süresi ölçümleri yapılmıştır. CUDA yazılımının C kodundan yaklaşık on kat daha kısa işlem süresine sahip olduğu görülmüştür. Yapılan çalışmalar ve analizler kapsamında elde edilen tüm veriler literatür çalışmalarına atıf yapılarak ve görsel niteliklerle desteklenerek bu tez çalışması içerisinde ayrıntılarıyla sunulmuştur.

Tez çalışmaları kapsamında iyonkürenin içeriği, katmanları ve KD haberleşme sistemlerinde kullanım alanları ile ilgili bilgiler Bölüm 2'de verilmiştir. İyonküre üzerinde ışın izleme işlemleri için kullanılan araçlar ve Snell Yasası ile ilgili ayrıntılı bilgiler Bölüm 3'te ele alınmıştır. Işının yansıması, kırılması ve saçılması ile ilgili teorik bilgiler bu bölümde anlatılmıştır. Tez kapsamında yazılımsal faaliyetlerin üzerinde yapıldığı grafik işlemci birimleri ilgili ayrıntılı bilgiler Bölüm 4'te belirtilmiştir. Bu bölümde grafik işlemci birimlerinin mimari yapısı, paralel programlama ve merkezi işlemci birimleri ile grafik işlemci birimlerinin karşılaştırmaları ele alınmıştır. Bölüm 5'te, tez çalışmalarında kullanılan IONOLAB-RAY algoritmasına değinilmiş olup, yapılan çalışmalardaki analiz, tasarım ve yazılım gibi ana işlevler ayrıntılı bir şekilde açıklanmıştır. Gerçekleştirilen yazılımsal işlemlerin test ve süre ölçüm faaliyetleri ile yazılım iyileştirme faaliyetleri için kullanılan araçlara da bu bölümde yer verilmiştir. Bölüm 6'da tez kapsamında yapılan deneylere değinilmiştir. Bu bölümde yapılan test faaliyetleri, süre ölçüm çalışmaları ve grafiksel anlatımlarla tez çalışması desteklenmiştir. Bölüm 7'de ise elde edilen sonuçlar değerlendirilmektedir.

2. İYONKÜRE

İyonküre, yeryüzünden yaklaşık 80 km ile 1100 km yükseklik arasında bulunan bir havaküre katmanıdır. İyonküre, gazların güneş ışınları ile iyonize olduğu bir havaküre tabakası olarak da bilinmektedir. İyonküredeki elektron yoğunluğu yükseklik, konum, zaman, sıcaklık, mevsim, gece-gündüz durumu gibi birçok etmene bağlı olarak değişim göstermektedir.



Şekil 2.1 Havakürenin Katmanları [14].

KD haberleşme, navigasyon, ufuk ötesi radar vs. sistemlerinde iyonküredeki elektron yoğunluğunun bilinmesi çok önemlidir. Bu bölümde iyonkürenin yapısı ve iyonkürenin kullanım alanları ile ilgili bilgiler verilecektir.

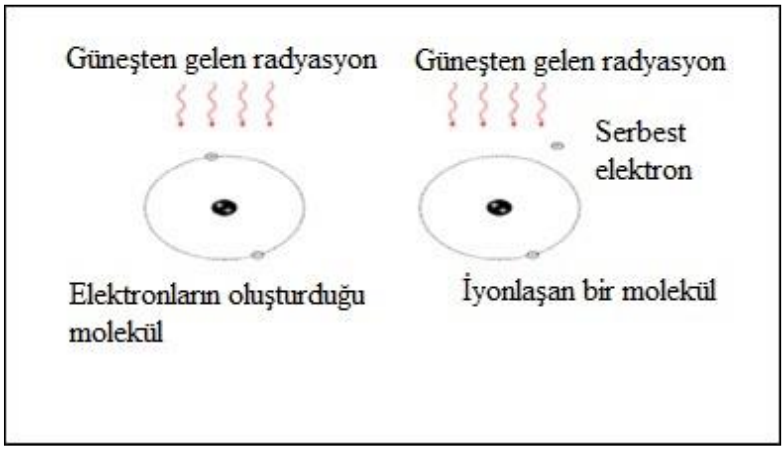
2.1. İyonkürenin Yapısı

Dünya'nın çevresini sarmakta olan gaz tabakasından oluşan havakürenin bir tabakasına iyonküre adı verilmiştir. Havaküre Şekil 2.1'de de görülen beş ana katmandan oluşmaktadır. Bu katmanlar sırasıyla aşağıdaki gibi açıklanmıştır.

- Troposfer: Yeryüzüne en yakın olan havaküre tabakasıdır. Gaz yoğunluğu olarak en yoğun olan katmandır. Bu katmanın kalınlığı 20 km'yi bulabilmektedir.
- Stratosfer: Traposfer ile yeryüzünden 50 km yüksekliğe kadar olan kalınlığa sahip katmandır. Havakürenin yaklaşık %20'si bu katmanda yer almaktadır [15].

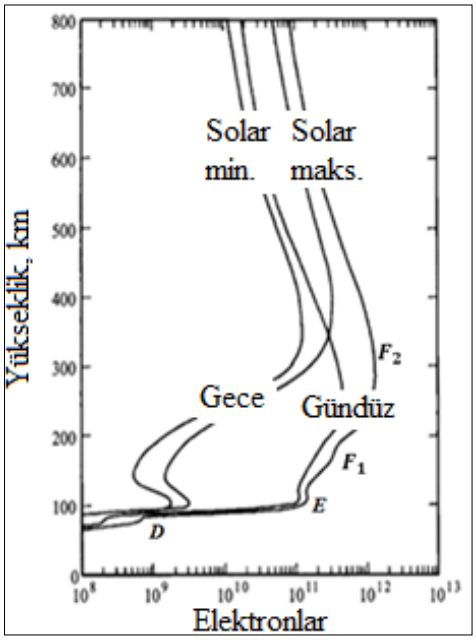
- Mezosfer: Stratosferden başlayıp 85 km'ye kadar ulaşan katmandır. Bu bölümde gazların iyonlaşmaya başladığı görülür. Ayrıca, Güneş'ten gelen zararlı ultraviyole ışınların tutulduğu bölümdür.
- Termosfer: Isılküre olarak da bilinen bu katman, gazların iyon halinde bulunduğu ve iyonlar arasında elektron alışverişinin oldukça fazla olduğu katmandır. Bu sebeple haberleşme sinyalleri ve radyo dalgaları bu katmanda çok iyi iletilir. Termosferin içerisinde yeryüzünden 80.km'den başlayıp 1100 km'ye kadar olan bölüme ise iyonküre adı verilir. Bu bölümde güneş ışınlarının etkisiyle iyonlaşan gazlar elektron yüklü hale gelmektedir.
- Ekzosfer: Havakürenin son katmanını oluşturan tabakadır. Yeryüzünden gönderilen yapay uyduların bulunduğu bölümdür.

İyonküre, gaz moleküllerinin güneş ışınlarının etkisiyle iyonize olduğu havaküre katmanlarından biridir. Havakürenin bu katmanında serbest iyonlar bulunmakta ve elektron yoğunluğu seviyesinin yüksekliği belirgin bir şekilde görülmektedir. Ayrıca Güneş'ten gelen yüksek enerjili parçacıklar bu katmanda gazların iyonlaşmasına neden olmaktadır. Güneş ışınlarıyla gelen yeterli yoğunluğa sahip bir radyasyon, molekül ya da atoma çarptığında enerji açığa çıkarıp Şekil 2.2'deki gibi serbest elektronun ortaya çıkmasına neden olabilmektedir. Güneş'ten gelmekte olan yüksek yoğunluklu güneş ışınları da bu şekilde iyonkürede bulunan gazların iyonize olmasına neden olmaktadır [16].



Şekil 2.2. İyonkürede iyonlaşma [17].

İyonküre temelde D, E ve F katmanları olarak nitelendirilen üç ana bölüme ayrılmıştır. Elektron yoğunluğuna bağlı olarak iyonküredeki bu katmanların olduğu bölümler Şekil 2.3'de görülmektedir. D katmanı yerküreden yaklaşık 80 km ile 100 km arasında bulunan bölümü tanımlamaktadır. Bu katmanda iyonlaşma iyonkürede en az olan bölgedir. İyonkürede yaklaşık 100 km ile 150 km'lik bölge arasında kalan alan ise E katmanı olarak nitelendirilmektedir. Son olarak yeryüzünden 150 km'den itibaren olan iyonküre katmanı ise F katmanı olarak nitelendirilmektedir. Geceleri tek katman olan F katmanı, gündüzleri F1 ve F2 olarak iki katmana ayrışmaktadır. F katmanında elektron iyonlaşması diğer katmanlara göre çok daha fazla olmaktadır [18].



Şekil 2.3. İyonkürede elektron yoğunluğu [19].

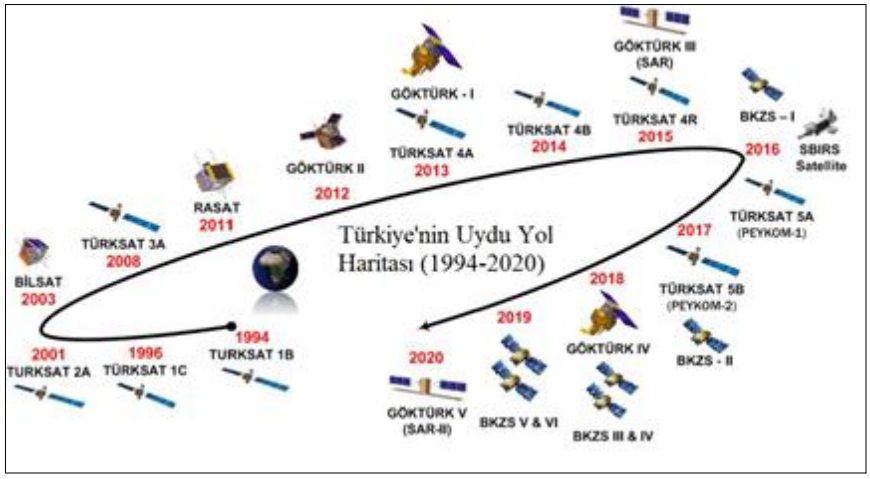
Yapısı ve içeriği açıklanmış olan iyonkürenin kullanıldığı başlıca kullanım alanları aşağıdaki bölümde başlıklar halinde ifade edilmiştir.

2.2. İyonkürenin Kullanım Alanları

İyonküre, haberleşme sistemlerinde önemli bir yere sahiptir. İyonkürede sinyal iletimi, elektron saçılımı ve dalga yayılımı gibi konular iyonküredeki ışın hareketlerini incelemeye yönelik konuları ifade etmektedir. Bir önceki bölümde yapısı incelenen iyonkürenin nerede hangi amaçlarla kullanıldığı aşağıdaki alt bölümlerde ele alınmıştır.

2.2.1. Uydu Haberleşmesi

Uydular dünya yörüngesine yerleştirilmiş, çoğunlukla haberleşme amacıyla kullanılan sistemlerdir. Uyduların haberleşme amaçlı kullanılması fikri ilk olarak İngiliz bilim adamı Arthur C. Clarke tarafından Mayıs 1945'te ortaya atılmıştır [20]. Yıllar içerisinde kullanımı giderek artan uydu haberleşme sistemleri hem askeri hem de sivil sistemler için günümüzde olmazsa olmaz bir konuma gelmiştir. Ülkemizde kullanılmakta olan ve gelecekte kullanıma alınması planlanan bazı uydular Şekil 2.4'de görülmektedir.



Şekil 2.4. Ülkemizin Uydu Haberleşme Kullanımı [21].

Kullanım yeri ve tasarımına bağlı olarak yeryüzünden farklı yükseklikte uydular bulunmaktadır. Uydulardan gönderilen sinyallerin, radyo dalgalarının yayılmasını etkileyecek düzeyde iyonlaşmanın bulunduğu iyonküreden geçmesi iyonküreyi uydu haberleşme sistemini etkileyen faktörlerden biri yapmaktadır. İyonküre üzerinden iletilen radyo dalgaları emilim, yansıma, kırılma, saçılma, polarizasyon gibi farklı zayıflatma mekanizmalarıyla karşılaşmaktadır. Bu nedenle iyi bir uydu haberleşme sistemi tasarımında iyonkürede radyo dalgasının saçılımı ve yayılımının da göz önünde bulundurulması gerekmektedir [22].

2.2.2. Seyrüsefer Sistemleri

Seyrüsefer sistemlerinin en bilinenlerinden olan GPS sistemi de iyonküreden etkilenen sistemlerdendir. GPS ile Dünya üzerinde bulunulan herhangi bir konum, Şekil 2.5'deki gibi uyduların sinyalleri kullanılarak bulunabilmektedir.

Şekil 2.5. GPS örnek çalışma görüntüsü [23].

Gazların iyonize olduğu iyonkürede bu iyonizasyonu sağlayan düzensiz ve beklenmedik güneş enerjisi, iyonkürenin GPS sinyalleri üzerindeki etkisinde değişiklikler yaratabilmekte ve havaküredeyaklaşık 100 km yükseklikteki alanda hatalara neden olabilmektedir. Ayrıca iyonküredeki elektron yoğunluk değişimi, elektromanyetik alan, güneş ışınları gibi radyo dalgalarının iyonkürede saçılımını etkileyen faktörler de GPS sistemlerini etkileyebilmektedir. Bu nedenle iyonküredeki değişikliklerin analizinin iyi yapılabilmesi yer bulma sistemlerinin daha doğru sonuçlar verebilmesini sağlamaktadır [18, 24].

2.2.3. Diğer Kullanım Alanları

KD haberleşme sistemleri uzun mesafeli haberleşmelerde kullanılmaktadır. Bu haberleşme altyapısını kullanıldığı alanlardan biri de büyükelçiliklerdir. Bu kullanımda alıcı ve gönderici antenlerin konumları birbirine bağlantılı bir konumda bulunmaktadır. Kullanılan bu antenler ile uzun mesafe haberleşmesi sağlanabilmekte olup, bir örnek anten görüntüsü Şekil 2.6'daki gibidir.

Şekil 2.6. Büyükelçiliklerde kullanılan bir anten örneği [25].

İyonkürenin etkin olduğu diğer kullanım alanları arasında askeri ve sivil haberleşme sistemleri göze çarpmaktadır. Vericiden gönderilmekte olan sinyal iyonkürede yol almaktadır. Işınım, kırılım, saçılım gibi etmenlerin iyonkürede radyo dalgasına etkisi sonrasında gönderilen radyo dalgası iyonküre içerisinde yol alıp alıcı pozisyonundaki antene ulaşmaktadır. Ufuk ötesi radarlar ve deniz haberleşmelerinde kullanılan askeri haberleşme sistemlerinden bazılarında sinyal iletimi iyonküre üzerinden gerçekleşmektedir. Bunların yanı sıra sivil uydu haberleşme sistemlerinde, amatör telsiz haberleşmelerinin bazılarında, yayıncılık sektörü haberleşmelerinde, mobil haberleşmelerde, uzun mesafeli uydu haberleşmeleri gibi birçok sivil haberleşme sistemlerinde de iyonküre üzerinden haberleşme kullanılmaktadır [26].

İyonkürenin yapısı ve kullanım alanlarının belirtilmesinin ardından, iyonkürede dalga yayılımıyla ilgili bilgiler sıradaki bölümde verilmiştir. Ayrıca Bölüm 3'te tez kapsamında kullanılmış olan geometrik-optik ışın izleme yöntemi ile ilgili bilgiler verilmiş olup, iyonkürenin üç boyutlu modellenmesine ve literatürdeki çalışmalara değinilmiştir.

3. İYONKÜREDE IŞIN İZLEME

Bu bölümde, iyonkürede dalga yayılımının hesaplanmasında kullanılan elektron yoğunluk bilgisi ve kırılım indislerinin oluşturulması, ışın izleme yönteminin içeriği ve ışın izleme işlemleri ile ilgili yapılan literatür çalışmalarına yer verilmektedir. İlk olarak, yazılımda kullanılacak olan iyonküre parametrelerinin elde edildiği IRI-Plas yazılım aracından bahsedilmektedir. Daha sonra iyonkürede üç boyutlu ışın izleme işlemleri ayrıntılı olarak açıklanmaktadır. Literatürdeki çalışmalardan bazıları açıklanarak üçüncü bölüm tamamlanmıştır.

3.1. IRI-Plas

İyonkürede ışın izleme işlemlerinde en önemli parametrelerden ikisi, iyonkürenin fiziksel yapısı ile ilgili olan, konuma bağlı elektron yoğunluğu ve buna bağlı olarak da kırılma indisi bilgileridir. İyonkürenin fiziksel yapısı birçok farklı parametre ile ifade edilir ve bu parametrelerin hesaplanmasında farklı modellere dayanan araçlar geliştirilmektedir. Yapılan bu tez içeriğinde kullanılan kırılım indislerinin iyonkürenin fiziksel niteliklerini yansıtacak parametreler ile hesaplanması gerekmektedir. Tez kapsamında, iyonkürenin parametrelerinin hesaplanmasında IRI-Plas (International Reference Ionosphere – Plasmasphere) yazılım aracı kullanılmıştır.

IRI yazılım aracı, Uzay Araştırma Komitesi Uluslararası Radyo Bilimi Birliği (URSI) sponsorluğunda geliştirilmiştir. IRI modeli verileri, Yerküresel Konumlama Sistemi (YKS) uydularından, yerküresel iyonosonda ağından ve geri saçılımlı radarlardan sağlanmaktadır [4].

IRI-Plas, IRI modeline iyonkürenin 20.200 km yüksekliğine kadar olan plazma kısmının da dahil edilmesi ile ortaya çıkmıştır. Bu araç, kullanıcının ilgili tarih ve zaman için ölçüme dayalı TEİ veri girişi imkanı sağlamaktadır. Kullanıcı tarafından girilen bu veriler IRI-Plas içindeki modele beslenebilmektedir ve böylece mevcut iyonküreyi daha iyi yansıtan parametre değerleri çıktı olarak sunulabilmektedir. IRI-Plas'ın standart yükseklik çözünürlüğü, 80 km'den 500 km'ye 20'şer km, 500 km'den 1000 km'ye 50'şer km, 1.000 km'den 2000 km'ye 200'er km, 2.000 km'den 3.000 km'ye 500'er km, 3.000 km'den 10.000 km'ye 1000'er km ve 10.000 km'den 20.000 km'ye 2.000'er km adımlar ile verilmektedir [4]. Yapılan bu tez kapsamında, IRI-Plas aracı kullanılarak belirli bir tarih ve konumdaki

bölgenin iyonküre parametreleri hazırlanan grafik işlemcide çalıştırılacak olan CUDA yazılımına girdi verisi olarak sunulmaktadır [6, 27, 28, 29].

3.2. İyonkürede Snell Yasası ile Işın İzleme

Işın izleme, ışık kaynağından çıkan bir ışının hedef cisme gidene kadar ne şekilde hareket ettiğini göz önüne alarak herhangi bir cismin ya da mekanın görüntüsünü oluşturan bir grafik oluşturma yöntemidir. Işın izleme yöntemi optik, elektromanyetik, akustik, jeofizik gibi alanlarda dalga yayılım problemlerinin çözümünde yaygın olarak kullanılan bir çözüm yöntemidir.

Günümüzde görüş hattının ötesinde, uzak mesafeler ile haberleşme amacıyla kullanılan yöntemlerden biri de KD haberleşmedir. KD bandında yer alan sinyaller havaküredeyer alan katmanlardan biri olan iyonküreden çeşitli kırılma açıları ile ilerleyerek uzak mesafelere ulaşabilmektedir. İyonküre kendi içerisinde farklı iyon yoğunluğundaki katmanlardan oluşmaktadır. Bu nedenle KD bandındaki sinyaller, iyonküre tabakasında frekansa bağlı olarak farklı yüksekliklerden yansımaktadırlar [30].

KD radyo haberleşmelerinde dalganın iyonkürede ilerlediği yolun hesaplanmasında kullanılan yöntemlerden biri de ışın izleme modelidir. Işın izleme yöntemi fizik biliminde sabit olmayan ve değişebilen kırılım, saçılım ve yansıma özelliklerine sahip bölgeler içinde belirli dalganın ya da parçacıkların ilerleme yolunu hesaplamak için kullanılan bir yöntemdir. Işın izleme modeli fizik bilimi kapsamında değerlendirildiğinde bir geometrik optik yaklaşım olarak görülmektedir.

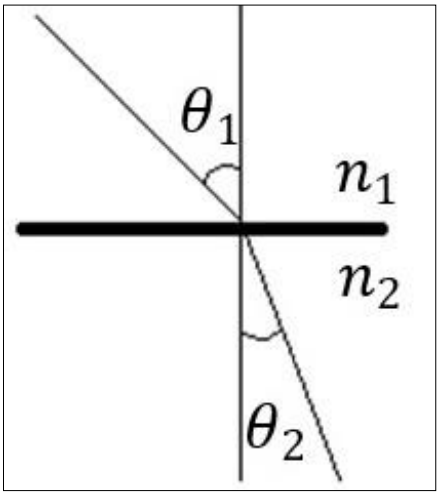
İyonkürede yol almakta olan bir dalga, iyonkürenin elektron yoğunluğu farklı bölgelerinde ilerlerken sıradan ve sıradışı dalgalar olmak üzere iki farklı dalga gibi yoluna devam etmektedir. Sıradan ve sıradışı dalgalar birbirlerinden bağımsız şekilde iyonkürede hareket edebilmekte ve birbirlerinden farklı zamanlarda farklı konumlara gidebilmektedirler. Bu iki dalga yolunun hesaplanmasında kullanılan bazı matematiksel formüller bazı yöntemlerde zor hesaplanma, karmaşık işlemler vs. nedeniyle modelleme yapılırken işlem yükü sorunları oluşturmaktadırlar.

İyonkürede ışın izleme modelini çözümleyebilmek adına Eikonal Denklemi, Hamiltonian Denklemleri ve Haselgrove Denklemleri, Booker Denklemi, WKB Yöntemi gibi çözüm yöntemleri ortaya çıkmıştır. Fakat bu yöntemlerde kullanılan karmaşık ve yoğun işlem

yükleri büyük bir sorun olarak görülmektedir. İşlem yüklerinin yoğun olması algoritmaların yavaş çalışmasına ve doğal olarak da ışın izleme modelinin yavaş sonuçlanmasına neden olmaktadır. İşlem yükü ve algoritmanın sonuçlanma süresi temel alındığında Snell Yasası diğer yöntem ve denklemlerden daha avantajlı durumda bulunmaktadır [4].

Tez kapsamında yapılan çalışmalarda işlem yükü ve sonuçlanma sürelerindeki avantajlarından dolayı geometrik optik yaklaşımı ile ışın izleme yöntemlerinden biri olan ayrıca frekansı modellemede en çok kullanılan yöntemlerden Snell Yasası kullanılmıştır [31].

Snell Yasası 1621 yılında Alman matematikçi Willebrord Snell tarafından ortaya atılmıştır. Işık ilerlerken kendisine en hızlı ilerleyebileceği istikameti seçmektedir. Işın hızı ise ışının farklı ortam koşullarına girmesine bağlı olarak değişmektedir. Şekil 3.1'de örnek bir görüntüsü verilen bir ortamdan farklı yoğunluktaki başka bir ortama geçiş yapan ışının yolunun hesaplanmasında Snell Yasası kullanılmaktadır. Snell Yasası, kesitler arasındaki kesişime gelme ve kırılma açısı arasındaki ilişkiyi vermektedir.



Şekil 3.1. Snell Yasası'nda Açı ve Kırılım.

Her iki ortamda, sırasıyla n_1 ve n_2 parametreleri ışının geçeceği iki farklı kırılma indisini ifade etmektedir. Kırılım ve yön değiştirmeden dolayı ışının ilerleyeceği yolun açısal değişimini de θ_1 ve θ_2 açıları belirtmektedir. θ_1 değeri ışının geliş açısının yüzey normaliyle yaptığı açıyı verirken, θ_2 açısı ışının n_2 ortamında yüzey normaliyle yaptığı açıyı belirtmektedir. Bu açı ve kırılma indisleri arasındaki ilişki

$$n_1 \sin\theta_1 = \mathrm{n}_2 \sin\theta_2 \quad (3.1)$$

ile ifade edilmektedir.

Bazı uygulamalarda iyonküre düzlemsel ya da küresel paralel katmanlardan oluşacak şekilde modellenebilmektedir. Bu katmaların kendi içinde düzgün dağılmış ve yön bağımsız olması kabulü yapılabilmektedir. Bu şartlar altında sinyalin iyonkürenin katmanlarında maruz kalacağı kırılma, yansıma ve iletim Snell Yasası ile hesaplanabilmektedir.

Snell Yasası'nın uygulanabilir olabilmesi için,

- İki bölge arasındaki sınır alan yüzeyinin dalga boyuna göre yeterince büyük olması,
- Gelmekte olan dalganın yüzeye sıfırdan farklı bir derece ile yaklaşması,
- İki bölge arasındaki sınırın pürüzsüz kabul edilmesi,

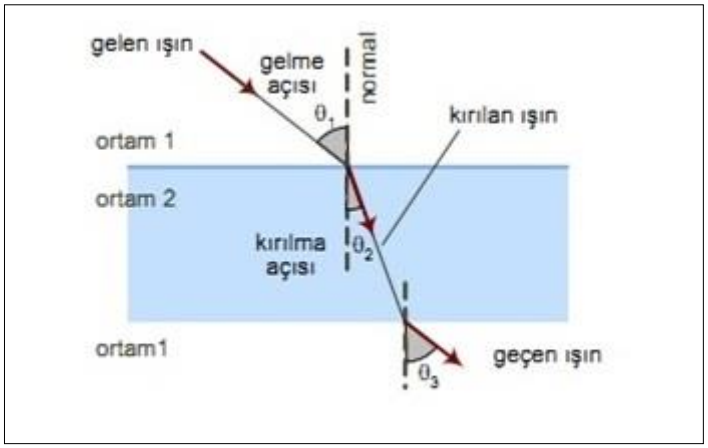
gerekmektedir.

İlerlemekte olan dalga sınıra ulaştığında burada kırılım veya yansıma gerçekleşmektedir. Elektron yoğunluğu çok olan ortamdan az olan ortama geçiş yaparken, kırılma açısının 90˚ olduğu andaki gelme açısına kritik açısı denmektedir. Geliş açısı kritik açıdan küçük olan dalga kırılarak ikinci ortama geçmekte ve ilerlemesine devam etmektedir. Kritik açıdan büyük olan geliş açılarında ise dalga yansıyarak geldiği ortama geri dönmektedir.

İyonkürenin düzgün dağılmış ve tek yönlü katmanlardan oluşacak şekilde modellendiği uygulamalarda Snell Yasası sıklıkla kullanılmaktadır [32]. Işığın farklı katmanlara geçişi sırasında yaşanan iletim, kırılım ve yansıma olayları ile ilgili aşağıdaki bölümlerde bilgi verilmiştir.

3.2.1. Işının Geçişi

Işının ilerleme hızı ortamdaki atomların, iyonların veya moleküllerin cinslerine ve konsantrasyonlarına da bağlı olup Şekil 3.2'de görülmektedir.



Şekil 3.2. Işının geçişi [33].

Bir ortamın kırılma indeksi onun ışın ile etkileşiminin bir ölçüsüdür ve

$$n_i = \frac{c}{v_i} \tag{3.2}$$

ile verilir. Burada, n_i, i frekansındaki kırılma indeksi, v_i ışının ortamdaki hızı ve c ışının vakumdaki hızıdır. Ortam yoğunluğu ile ilgili hesaplamada Denklem 3.2 kullanılır.

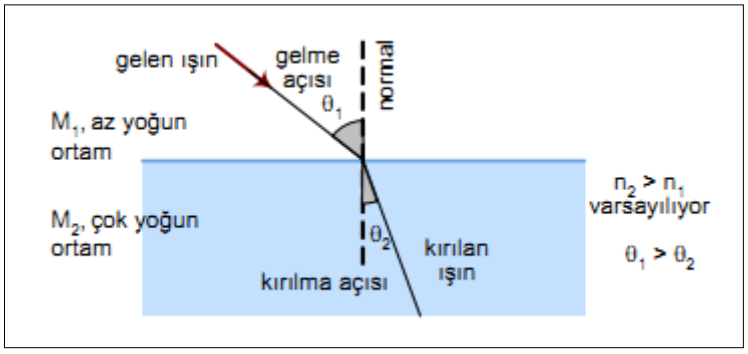
Geçişe neden olan etkileşim, ışının elektrik alanının değişmesiyle açıklanabilmektedir. Değişen elektrik alanı, ortamda bulunan taneciklerin bağlı elektronlarının dalgalanmalarını ve her bir taneciğin polarize olmasını sağlar. Işının sönümlenmemesi koşulu ile tanecikler polarizasyon enerjisini çok kısa bir süre (10^{-14} – 10^{-15} saniye) tutar ve hiçbir değişikliğe uğratmadan tekrar çıkarırlar, ardından tanecikler de orijinal durumlarına dönerler. Bu olayda herhangi bir enerji değişikliği olmadığından maddeden çıkan ışının frekansı değişmez fakat ilerleme hızı, tanecikler tarafından tutulma ve tekrar bırakılma sırasında geçen zaman nedeniyle biraz azalır. Dolayısıyla transmisyon olayı ara kademedeki atomların, iyonların veya moleküllerin dalgalandığı basamaklı bir işlem olarak tanımlanabilmektedir. Bir ortamda polarize olan her bir tanecikten çıkan ışının değişik yönlerde hareket edebileceği düşünülebilir. Taneciklerin küçük olması halinde, zıt yönlü ışınların birbirlerini yok etmesi sonucu orijinal ışık yolunun yönünden farklı yönlerde önemli derecede bir hareket görülmez. Taneciklerin, polimer molekülleri veya kolloidal tanecikler gibi büyük olması durumunda ise birbirini yok etme etkisi zayıflar ve ışınların bir kısmı farklı yönlerde hareket ederler, yani saçılırlar.

3.2.2. Işının Kırılması

Işın fiziksel yoğunlukları birbirinden farklı bir ortamdan diğerine geçerken, bu iki ortam arasındaki yoğunluk farkı nedeniyle aniden yön değiştirir. Şekil 3.3'de kırılım ile ilgili verilen görsel verilmiştir. Bir ışının ortam geçişi sonrasına yön değiştirmesine ışığın kırılması denir ve

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{n_2}{n_1} = \frac{v_1}{v_2} \tag{3.3}$$

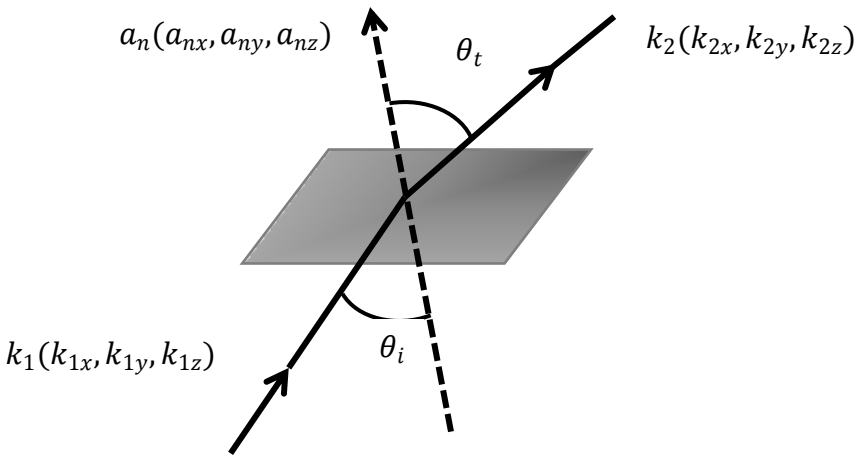
ile gösterilir. Şekil 3.3'deki M_1 ortamı vakum ise v_1 = c, n_1 =1'dir ve M_2 ortamının kırılma indeksi bu durumda iki açının sinüsleri oranına eşit olur. Ortam yoğunluğu ile ışınların kırılım açı ve hız bilgileri çözümlemesi Denklem 3.3 kullanılarak yapılmaktadır.



Şekil 3.3. Işının kırılması [33].

İyonküre üzerinde yol alan dalganın yayılım doğrultusunun hesaplanmasında kullanılan önemli değişkenlerden biri de iyonkürenin kırılma indisidir. Tez çalışması içerisinde, bahsetmiş olunan bu indisi hesaplamak için Appleton-Hartree Formülü kullanılmaktadır. Appleton-Hartree Formülü soğuk plazma ortamı için kırılma indisini veren matematiksel ifadedir. Bu ifade birbirlerinden bağımsız olarak E. V. Appleton, D. Hartree ve H. K. Lassen tarafından geliştirilmiştir. Bu nedenle bu denklem Appleton-Lassen Formülü olarak da bilinmektedir [6].

İki farklı bölge arasındaki katman sınırına gelen bir ışının geliş açısı kritik açıdan küçük olduğu durumda bu ışın kırılarak diğer bölge içine doğru ilerlemektedir. Kırılarak diğer bölgede yayılan ışının yayılma yön vektörü Şekil 3.4'deki geometrik yapıya göre hesaplanmaktadır [4].



Şekil 3.4. İyonküre katman sınır düzleminde kırılma [4].

Bu yapıya göre şu koşullar sağlanmalıdır:

- Katmanlar arasındaki sınırının yüzey normali, birinci katmandaki ışın yayılım yönü ve ikinci katmandaki ışın yayılım yönü aynı düzlem üzerinde olmalıdır. Buna göre

$$(a_n \times k_1) \cdot k_2 = 0 \tag{3.4}$$

  olmaktadır.
- Birinci katmandaki ışın yayılım yönü ile ikinci katmandaki ışın yayılım yönü arasındaki açı $\theta_t - \theta_i$ olmalıdır. θ_t açısı Denklem 3.1 ile hesaplanmaktadır. Bu durumda

$$k_1 \cdot k_2 = \cos(\theta_t - \theta_i) \tag{3.5}$$

  sağlanmalıdır.
- Katmanlar arasındaki sınırının yüzey normali ile ikinci katmandaki ışın yayılım yönü arasındaki açı θ_t olmalıdır. Buna göre

$$a_n \cdot k_2 = \cos(\theta_t) \tag{3.6}$$

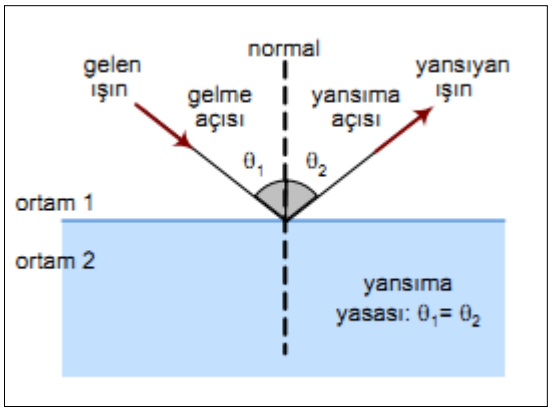
  denklemi elde edilmektedir.

Elde edilen bu üç denklemin (Denklem 3.4, Denklem 3.5, Denklem 3.6) beraber çözülmesi sonucunda ikinci katmandaki ışın yayılım yönü vektörünün üç bileşeni hesaplanmaktadır.

3.2.3. Işının Yansıması

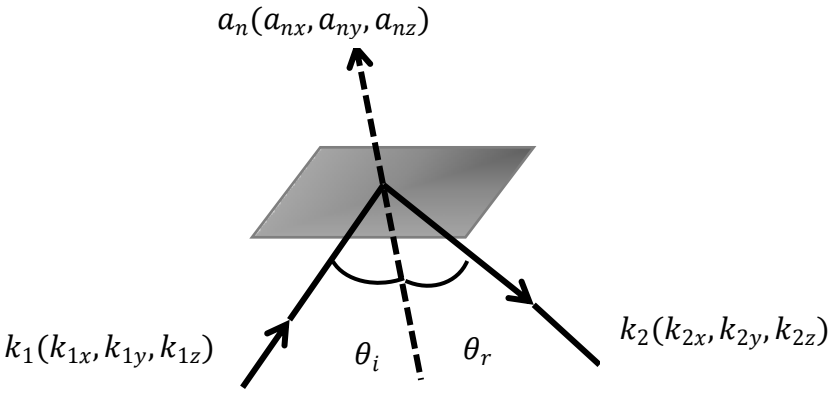
Yoğunluk indeksleri farklı olan iki ortam arasına giren ışının bir bölümü yansımaya uğrar. Ortamlar arasındaki yoğunluk indeks farkı ne kadar büyük olursa ışının yansıyan bölümü de

o kadar fazla olur. Yansıtıcı bir yüzeye gelen bir ışın, yüzeye dik olarak çizilen bir doğruya (normal) göre gelme açısına eşit bir yansıma açısıyla yansır. Yansıma yaşanan optik oluşum Şekil 3.5'de görülmektedir.



Şekil 3.5. Işının Yansıması [33].

Işının katman sınırına geliş açısı kritik açıdan büyük olduğu durumda ışın yansıyarak aynı katmanın içine doğru ilerlemektedir. Yansıyan ışının yayılma yön vektörü Şekil 3.6'daki geometrik yapıya göre hesaplanmaktadır.



Şekil 3.6. İyonküre katman sınır düzleminde yansıma [4].

Bu yapıya göre ise şu koşullar sağlanmalıdır:

- Katmanlar arasındaki sınırının yüzey normali, birinci katmandaki ışın yayılım yönü ve ikinci katmandaki ışın yayılım yönü aynı düzlem üzerinde olmalıdır. Buna göre

$$(a_n \times k_1) \cdot k_2 = 0 \tag{3.7}$$

olmaktadır.

- Birinci katmandaki ışın yayılım yönü ile ikinci katmandaki ışın yayılım yönü arasındaki açı $180° - (\theta_i + \theta_r)$ olmalıdır. θ_r açısı Snell Yasası'na göre θ_i açısına eşit olmaktadır. Bu durumda

$$k_1 \cdot k_2 = \cos(180° - 2\theta_i) \tag{3.8}$$

  sağlanmalıdır.
- Katmanlar arasındaki sınırının yüzey normali ile ikinci katmandaki ışın yayılım yönü arasındaki açı $180° - \theta_i$ olmalıdır. Buna göre

$$a_n \cdot k_2 = \cos(180° - \theta_i) \tag{3.9}$$

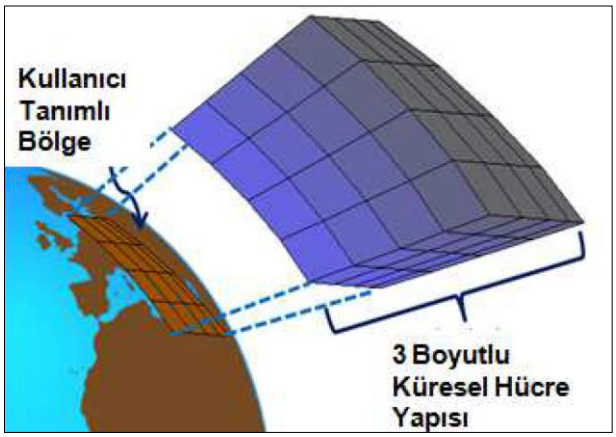
  denklemi elde edilmektedir.

Elde edilen bu üç denklemin (Denklem 3.7, Denklem 3.8, Denklem 3.9) beraber çözülmesi sonucunda ikinci katmandaki ışın yayılım yönü vektörünün üç bileşeni hesaplanmaktadır. Yapılan bu tez kapsamında kullanılan iyonkürede modellenen yüzeyler küp şeklinde varsayılmış, geçiş yüzeylerinin düz olduğunu kabul edilerek işlemler yapılmıştır.

3.3. İyonkürede 3 Boyutlu Model

Tez kapsamında yapılan çalışmalarda girdi verisi olarak alınan iyonküre elektron yoğunluğu ile ilgili model yapısı bu bölümde aktarılmaktadır. İyonkürenin yön bağımlı, düzgün dağılmayan ve zamana göre değişim gösteren yapısı üç boyutlu olarak modellenmiş şekilde incelenebilmektedir. İyonküre hücresel bölümlere ayrılarak hesaplamalarda kullanılmıştır. İyonküredeki elektron yoğunluğu modellemesi üç boyutlu hücresel yapıdan oluşturulmuş analiz ile tez çalışmasına girdi verisi olarak sunulmuştur. İyonküredeki elektron yoğunluk yapısı sadece yükseklikle değişim göstermeyip, enlem ve boylama göre de değişmesi nedeniyle üç boyutlu bu modelleme kullanılmıştır.

Üç boyutlu bu modellemede IONOLAB-RAY algoritması kapsamında geliştirilen araç kullanılmıştır. Kullanıcı incelemek istediği tarih, saat ve dakika bilgilerini ve diğer bilgileri bu araca girdi olarak girebilmektedir. Sonucunda iyonkürede üç boyutlu elektron modellemesi yapılmakta olup, ışın izleme algoritmasına girdi verisi üretilmiş olmaktadır. Böylece geliştirilmekte olan sinyal yayılım modelinin konuma ve zamana göre parametrik olarak modellenen iyonküre yapısına göre vereceği çıktıların değerlendirilmesine imkan sağlanmaktadır. Şekil 3.7'deki temsili modellenmiş iyonküre, girdi verisi olarak yazılım işlemlerinde kullanılmaktadır.



Şekil 3.7. İlgi alanı bölge üzerinde 3 boyutlu küresel ızgara modelinin gösterimi [4].

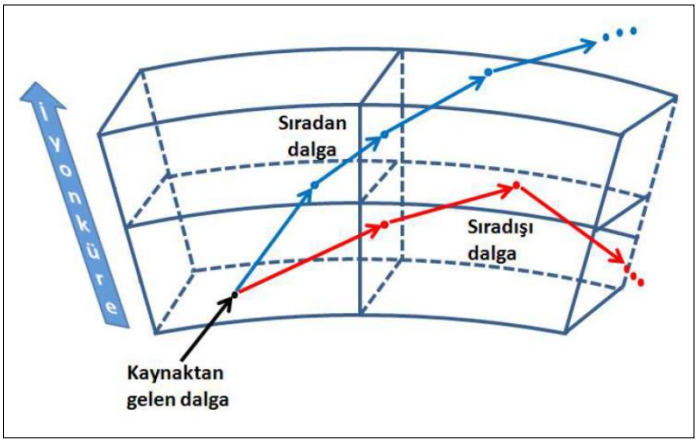
Tez kapsamında girdi verisi olarak kullanılan elektron yoğunluğunun modellenmesi, paralel düzlem katmanlardan oluşacak şekilde modellenmiştir. Katmanların yön bağımsız ve homojen olduğu kabul edilmiştir. Bu kabul doğrultusunda ışın izlemede Snell Yasası uygulanmıştır. Snell Yasası'nda kullanılan kırılma indisi değerleri, her bir katman için sabit olarak seçilen elektron yoğunluğu değerine karşılık sadeleştirilmiş Appleton-Hartree Formülü kullanılarak hesaplanmıştır. Basitleştirilmiş Appleton-Hartree Formülü'nde dünyanın manyetik alanı, elektronun dönme hareketi ve elektronun parçacıklarla çarpışma etkisi ihmal edilmiştir. Sonrasında konuma göre değişimin etkisini bir miktar yansıtmaya yönelik olarak katmanlar eksenler doğrultusunda döndürülmüş, ışının ilerleyişi izlenmiştir.

Girdi bilgisi olarak kullanılan belirli bir bölgenin elektron yoğunluğu öncelikli olarak her bir zaman adımına ve ızgara hücresine karşılık elektron yoğunluğu değerleri IRI-plas yazılım aracı kullanılarak sağlanmış, sadeleştirilmiş Appleton – Hartree ile kırılma indisi hesaplamaları yapılmıştır. Sonraki aşama olarak dünyanın manyetik alanı, elektronun dönme hareketi ve elektronun parçacıklarla çarpışma etkisinin de dahil edildiği Appleton - Hartree Formülü ile kırılma indisinin hesaplandığı durumda Snell Yasası'na dayanan ışın izleme algoritması uygulanmıştır. Geliştirilen algoritmanın MATLAB ortamındaki çıktıları ve grafik işlemcide çalıştırılan CUDA yazılımının çıktılarıyla karşılaştırılması tez çalışmaları kapsamında yapılmıştır.

Hesaplamalarda jeodezik koordinatlar, yerel koordinatlar ve YMYS koordinatları arasında gerekli dönüşümler yapılmakta, ışının ilerleme yönünün belirlenmesi, ışının yayılım vektörünün ile ilgili yükseklikteki küre ile kesişim konumunun hesaplanması ve dünyanın

manyetik alanı ile ışının ilerleme yönü arasındaki açı hesaplamaları ortak koordinat sistemi olarak YMYS'de yapılmaktadır.

Sıradan ve sıra dışı kırılma indisleri detaylı Appleton – Hartree Formülü'nün kullanılması ile hesaplanmaktadır. Bu durumda yazılım her adımda sıradan kırılma indisinin Snell Yasası'na uygulanması ile hesaplanan ışın izleme ve her adımda sıra dışı kırılma indisinin Snell Yasası'na uygulanması ile hesaplanan ışın izleme olacak şekilde iki parça olarak çalışabilecektir. Bu şekilde çalışan yazılımın çıktıları değerlendirilerek sıradan ve sıra dışı ışın yayılımının özellikleri incelenebilecektir. Şekil 3.8'de sıradan ve sıradışı dalgaların temsili olarak hücrelere ayrılmış iyonküre üzerindeki yayılımı yer almaktadır.



Şekil 3.8. Snell Yasası'nın 3 boyutlu küresel ızgara modelinde uygulanması [4].

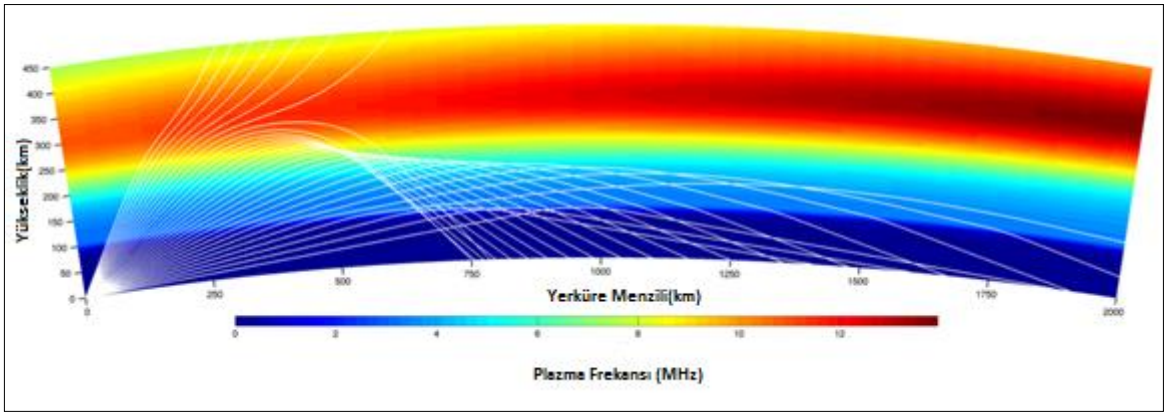
Tez kapsamında yapılan çalışmalarda hem sıradan hem de sıradışı dalga yolları hesaplanmıştır. Elde edilen deneysel sonuçlar sonucunda ışın dalga yolları aynı grafik üzerinde çizdirilmiş olup, sıradan ve sıradışı dalga yolları net olarak görülebilmektedir. Bu bölümün ardından IONOLAB-RAY algoritması ve muadil uygulama ve araçları ile ilgili bilgiler verilmiştir.

3.4. Geçmiş Çalışmalar

Bu bölümde tez kapsamında yapılmış olan faaliyetlerin literatürde benzerlerine ait bilgiler yer almaktadır. İyonkürede ışın izleme yöntemlerine ait farklı birkaç program bulunmaktadır. Bunlar arasından PHaRLAP, IONORT ve IONOLAB-RAY programları aşağıdaki alt bölümlerde ayrıntılarıyla açıklanmıştır.

3.4.1. PHaRLAP

PHaRLAP, Australya'nın araştırma kurumlarından biri olan Defence Science and Technology Organisation (DSTO) tarafından kısa dalga bandında olan dalga yayılımını modelleyen yazılımlardan biridir. Bu uygulama, iki boyutlu ışın izlemeden üç boyutlu manyetik-iyonik ışın izlenmesine kadar çeşitli birçok ışın izleme çözümleri sunmaktadır. İki boyutlu ışın izleme yöntemi ile yapılan PHaRLAP modellemesi örneği Şekil 3.9'da görülmektedir.



Şekil 3.9. PHaRLAP'ta iki boyutlu modelleme örneği.

PHaRLAP'ta iki boyutlu ışın izlemede Coleman denklemleri kullanılmaktadır. Üç boyutlu ışın izleme ise Haselgrove denklem setinin çözülmesi ile uygulanmaktadır. Haselgrove denklemlerini küresel koordinatlarda çözen Johns ve Stephenson kodu yerine işlem karmaşıklığını azaltmak amacıyla denklem setinin kartezyen koordinatlarındaki halini çözen kod uygulanmaktadır. Kullanılan kırılma indisi ise Appleton-Hartree Eşitliğidir. İyonkürenin parametrelerinin hesaplanmasında IRI kullanılmaktadır ve IRI modelinin getirdiği kısıtlar PHaRLAP yazılımına yansımaktadır. PHaRLAP yazılımının işlem yükü fazla olup, işlem süresi uzun sürmektedir [4].

3.4.2. IONORT

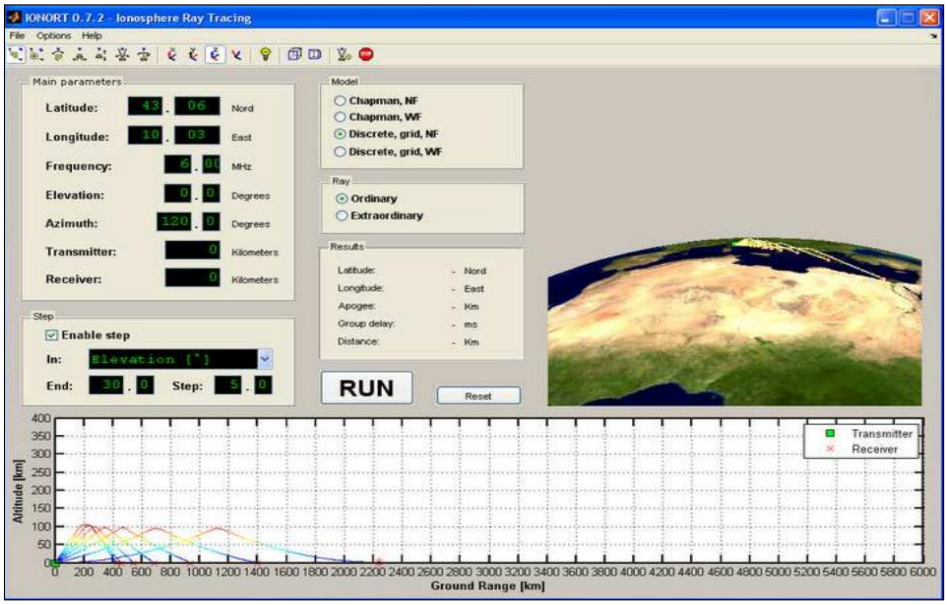
Işın izleme algoritmalarının iyonküre üzerinde uygulanmış bir diğer algoritma da IONORT isimli yazılımsal araçtır. Bu yazılım aracı, iyonküredeki kısa dalga haberleşmesinde iki ya da üç boyutlu hesaplama işlemlerinde kullanılmaktadır. Hesaplamalar sonucunda çıkan ışın izlerini 2 ya da 3 boyutlu olarak görselleştirebilen bir araçtır. Hesaplama sırasında temel olarak 6 tane birinci dereceden diferansiyel denklem çözümlemesi yapmakta olup, bağımsız

bir grup integral işlemleri de yapılmaktadır. IONORT, iyonküreyi modellerken elektron yoğunluğunu hesaplamada jeomanyetik yoğunluk, nötr parçacıklı elektron çarpışmasından frekans modellemesi gibi hala geçerliliği olan terimlerden istifade eder.

IONORT programı, ışın izleme algoritmalarının birinci dereceden difarensiyel denklem çözümlerini temel alan Hamiltonian formülasyonunu yermerkezli küresel koordinat sistemi ile çözümlemeyi baz almaktadır. Program 3 ana blok işlemden oluşmaktadır.

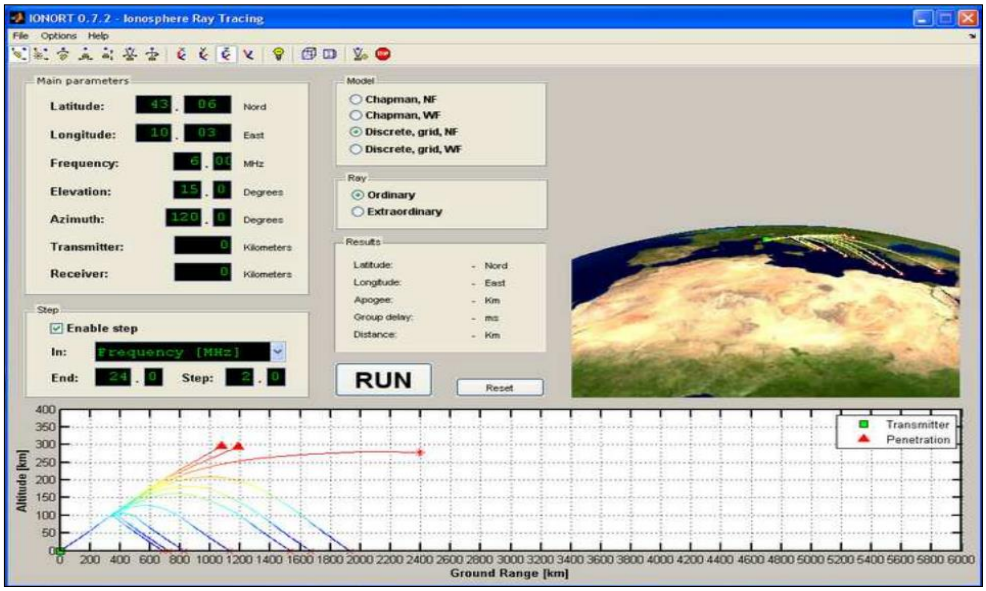
a) Kullanıcı arayüzünden grafiksel girdi bilgisi,
b) Algoritmanın işlemesi,
c) Kullanıcı arayüzünden grafiksel çıktı bilgisinin sunulması.

Kullanıcı arayüzünden grafiksel girdi bilgisinin oluşturulduğu ilk blokta yapılan işlemler MATLAB programı üzerinde gerçekleştirilmektedir. Algoritmanın gereksinim duyduğu hem veri hem de parametre değerleri Şekil 3.10'da ekran görüntüsü verilen bu bölümde oluşturulmaktadır.



Şekil 3.10. IONORT Arayüzü, Girdi verisi oluşturma.

Kalan işlemlerin kodu FORTRAN [34] dilinde yazılmış olup, matematiksel işlemler burada yapılmaktadır. Şekil 3.11'de de görüldüğü gibi son aşamada yine bir arayüz ile birlikte elde edilen sonuçlar ekrana yansıtılmaktadır.



Şekil 3.11. IONORT Arayüzü, Çıktı görüntüsünün oluşturulması.

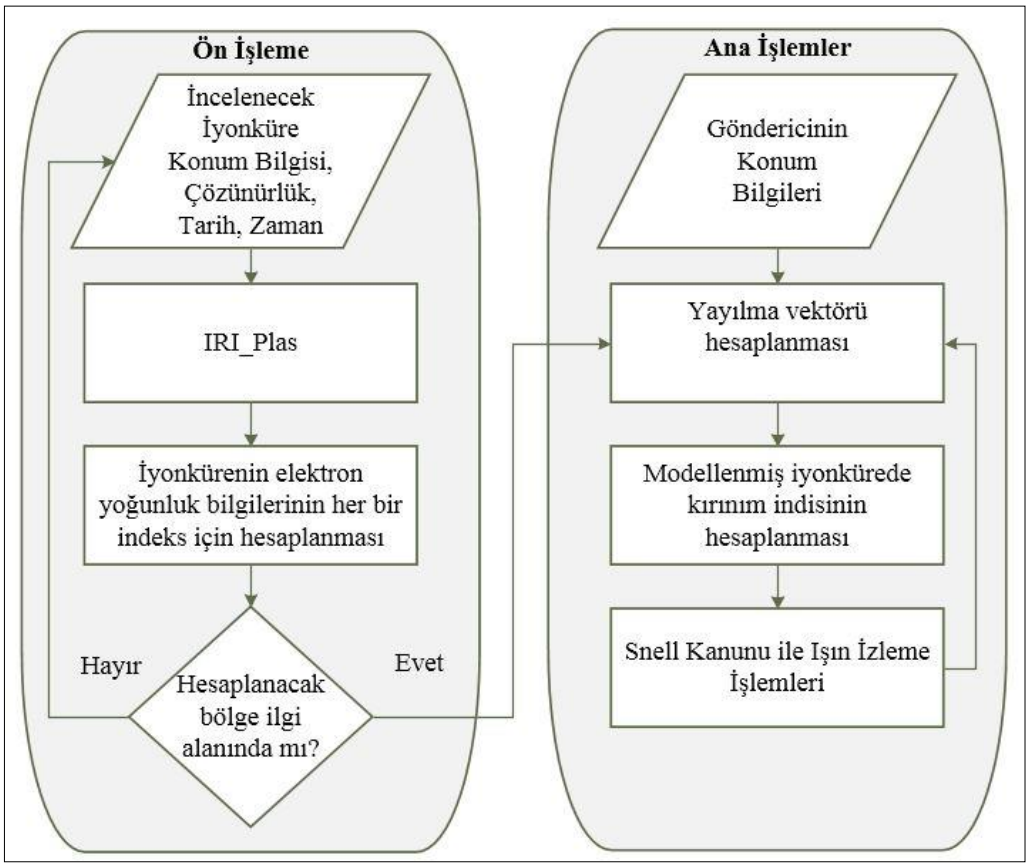
Arayüz görüntüleri yukarıdaki şekillerde verilen IONORT aracında kırılma indisinin nasıl yapıldığı açıklanmamış olup, grafiksel gösterimlerin kısıtlı olduğu bilinmektedir. IONORT yazılım aracı da tanıtıldıktan sonra tez çalışmalarımızın ana konusu olan IONOLAB-RAY algoritmasına aşağıdaki bölümde değinilmiştir.

3.4.3. IONOLAB-RAY

KD bandında sinyal yayılımını modellemek üzere IONOLAB-RAY algoritması geliştirilmiştir. Algoritma iki aşamadan oluşmaktadır. Bunlardan birincisi ön koşum aşaması, ikincisi ana koşum aşamasıdır. Ön koşum aşamasında kullanıcı tarafından girilen tarih, saat ve bölge sınırları dahilinde IRI-Plas yazılımı kullanılarak iyonküre parametreleri hesaplanmaktadır. Bu parametreler kullanılarak kırılma indisinin önceden hesaplanabilecek bileşenleri elde edilmektedir. Ön koşum aşamasında oluşturulan üç boyutlu küresel iyonküre geometrik modelinde enlem ve boylam adım çözünürlüğü kullanıcı tarafından belirlenebilmektedir. Kırılma indisinin hesaplanmasında kullanılacak parametreler modüler olarak ön koşum aşamasında hesaplanmaktadır. IONOLAB-RAY algoritmasında gömülü olan bir diğer modül IGRF modülüdür. Bu modül sayesinde senaryonun koşturulacağı konum ve zamana bağlı olarak dünyanın manyetik alanı hesaplanmaktadır.

Ana koşum senaryo girdilerinde kaynağın konumu, yönelim yanca ve yükseliş açıları ve çalışma frekansı tanımlanmaktadır. Kaynak noktasının konum ve yönelimi ile başlayarak iyonküre üç boyutlu küresel ızgara modelinde Snell Yasası'na dayanan ışın izleme

uygulanmaktadır. Snell Yasası'nın uygulanmasında kırılma indisi her adımda gelen ışının yönü ile dünyanın manyetik alanı arasındaki açı arasındaki ilişkiden ya da ilgili bileşen verileri kullanılarak hesaplanmaktadır [35]. Ana koşum kullanıcı tarafından girilecek parametre seti ile çoklu olarak koşturulabilmektedir. Böylece çıktılar karşılaştırmalı olarak analiz edilebilmektedir. IONOLAB-RAY algoritma akış şeması Şekil 3.12'deki gibidir.



Şekil 3.12. IONOLAB-RAY algoritması akış şeması.

Ön işleme ve ana işlemler olmak üzere 2 farklı koşum işleminden oluşan IONOLAB-RAY algoritmasında ana işlemler kısmı, ışınların işlendiği ve yayılım yollarının hesaplandığı bölüm olarak değerlendirilmektedir. Ana işlemler bölümünün işlem yükü, ön işlemlerden çok daha ağırdır. Yapılmış olan bu tez kapsamında IONOLAB-RAY algoritmasında ana işlemlerin yapıldığı bölüme odaklanılmış olup, bu kısmın paralel programlanması için çalışmalar gerçekleştirilmiştir. Bir sonraki bölümde grafik işlemciler, CPU ve GPU mimari farkları ve CUDA ile paralel programlama ile ilgili ayrıntılı bilgiler verilmektedir.

4. İŞLEMCİLER

Günümüzde pek çok alanda bilgisayar kullanımı oldukça yaygınlaşmıştır. Başlangıçta sadece bazı hesaplamalar yapmak için kullanılan bilgisayarlar, kullanım alanları genişledikçe insan hayatında daha çok yer edinmişler, iletişim yetenekleri geliştikçe de insanla daha çok ortamı paylaşır hale gelmişlerdir. Bu bağlamda görüntünün güzel olması pek çok uygulama için yeterli koşul olmasına rağmen, güncel uygulamaların büyük kısmında gerçekçilik de gereklidir. Örneğin oyunlar, özel animasyonlar ve tamamen animasyonlardan oluşan filmler, bilgisayar dünyasında oldukça popülerdir. Görüntünün gerçekçi ve güzel olmasında ekran kartı kuşkusuz büyük önem taşımaktadır. Ekran kartları kendi işlemcisi ile belleği olan grafik komutlarını direkt destekleyen donanımlardır. Görüntü oluşturmada donanımın verdiği desteğe ek olarak, kaliteyi arttırmak adına yapılan yazılım çalışmaları da bulunmaktadır. Bu çalışmalar arasında uygulanılabilirliğinin yüksek olması ve fiziki gerçekliğe oldukça yakın olması nedeni ile ışın izleme yöntemi yaygın olarak kullanılmaktadır. Işın izleme yöntemi, bilgisayar ortamında üç boyutlu, fotoğraf kalitesine çok yakın görüntü üretimi için kullanılan, oldukça basit ve gerçekçi bir yöntemdir. Üç boyutlu görüntü oluşturmak için kullanılabilen birçok alternatif yöntemin bulunmasına rağmen, ışın izleme yönteminin tercih edilmesinin nedeni, bu yöntemin gerçek dünyada ışığın evreyle olan etkileşimini modellemeye çalışmasıdır.

Açıklanmış olunan bu yöntemin kısa dalga haberleşmesinde de sıklıkla kullanıldığı görülmektedir. İyonkürede modellenmiş olan yüzeyde dalga ilerlerken adım adım Snell Yasası'ndan yararlanılarak, ışın izleme yöntemi uygulanmıştır [35].

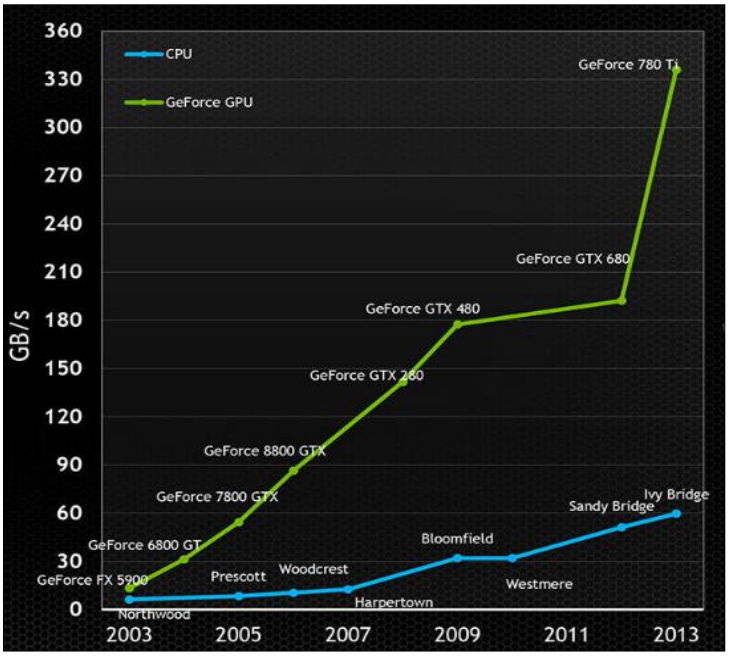
4.1. Grafik İşlemci Birimleri

Grafik işleme birimi (GPU), merkezi işleme biriminde (CPU) giderek artan işlem yükünün azaltılması amacıyla tasarlanmıştır. İki ve üç boyutlu görüntülerin işlenmesi, ekrana yüksek çözünürlükte görüntü sunulması gibi nedenler Şekil 4.1'de görülen GPU'ların ortaya çıkış amacı olmuştur. İlk olarak CPU'nun yapmış olduğu grafik işleme işlevleri GPU'da yaptırılarak hem CPU'nun işlem yükü azaltılmış hem de sistemin genel işleyişi hızlandırılmıştır. Bu iki birim arasındaki görev paylaşımından elde edilen performans kazancı ekran kartları ve grafik işleme üniteleri üzerinde günümüzde devam eden kapsamlı çalışmalar yapılmasına sebep olmuştur. Günümüzde mobil telefonlar, bilgisayarlar ve oyun

konsolları gibi alanlarda özelliklerinden faydalanılan GPU'lar, büyük ve karmaşık verilerin paralel işlenmesi gibi genel amaçlı hesaplamalarda da kullanılmaktadır [36].

Şekil 4.1. Grafik İşlemci Birim, Nvidia Tesla [37].

Veri boyutu fazla olan işlemlerin hesaplanmasında grafik işlemci biriminin önemli avantajları bulunmaktadır. CPU ve GPU arasındaki fark, görevleri nasıl işledikleridir. CPU ardışık seri işlem için optimize edilmiş birkaç çekirdekten oluşurken, GPU çok sayıda işi eşzamanlı olarak yürütmek için tasarlanmış binlerce daha küçük, daha verimli çekirdekten oluşur. Bu sayede büyük boyutlara sahip verilerin işlenmesi, paralel programlama yapılarak CPU'ya göre çok daha kısa sürede tamamlanmaktadır. Paralel programlama yapabilme kabiliyeti nedeniyle büyük verilerin işlenmesinde grafik işlemci üniteleri tercih edilmektedir. Yıllar içinde oldukça çarpıcı şekilde gelişen grafik işlemci birimlerinin günümüzdeki kullanımı giderek artmakta olup, Şekil 4.2'de CPU mimarilerinin ve GPU üreticilerinden birisi olan Nvidia [39]'nın GeForce ekran kartlarındaki yıllar içerisinde gelişen Gigabayt/saniye analizi görülmektedir.

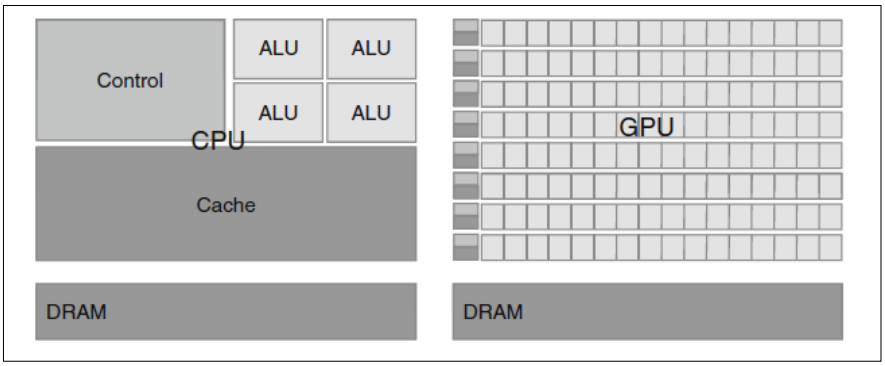


Şekil 4.2. GPU ve CPU Gelişimi [39].

Nvidia'nın grafik kartları ile CPU mimarilerinin gelişimi yıllara göre kıyaslandığında GPU gelişiminin CPU'ya göre çok daha hızlı olduğu görülebilmektedir. Gelişimin bu denli hızlı ve etkili olması, GPU'ların kullanım alanlarının artmasına neden olmaktadır.

4.3. GPU ve CPU Mimarisi

GPU ve CPU işleme birimini birbirlerinden ayıran en önemli özellik, GPU'da bulunan çekirdek sayısının CPU'dakinden çok daha fazla olmasıdır. Şekil 4.3'te de görüldüğü üzere 4 çekirdekli CPU mimarisinde bulunan 4 adet aritmetik-mantık (ALU) birimine karşılık GPU'da yüzlerce aritmetik-mantık birimi bulunmaktadır.



Şekil 4.3. CPU ve GPU Mimarisi [40].

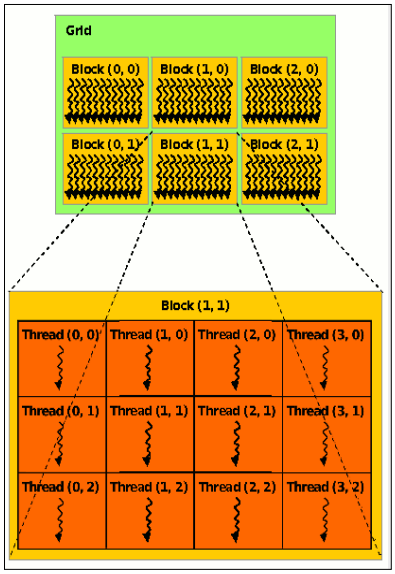
GPU'da aynı anda aritmetik-mantık birimlerine veriler gönderilerek binlerce iş parçacığı (thread) aynı anda kullanılıp eş zamanlı olarak bu birimlerin çalışması sağlanır. CPU'da ise aynı anda komuta edilen iş parçacığı sayısı birkaç tane ile sınırlıdır. CPU ve GPU arasındaki bu mimarisel farkın sebebi, işleme birimlerindeki transistor kullanımının farklı tasarımlara sahip olmasıdır. CPU'larda transistorlar yönergelerin yeniden düzenlenmesini gerçekleştiren tamponlar, rezerve istasyonları, tahmin mekanizmaları ve büyük tampon bellek gibi bileşenlerden oluşur. Bu bileşenler, tek bir iş parçacığının yürütülmesini hızlandırmak için tasarlanmıştır. GPU'da ise transistorlar işlemci dizileri, çoklu işlem donanımı, paylaşılan bellek ve birden fazla bellek denetleyicilere atanmıştır. Bu özellikler belirli bir iş parçacığının işletilmesinin hızlandırılması için değil, yonganın üzerinde bulunan onbinlerce iş parçacığının aynı anda yürütülmesi için tasarlanmıştır ve böylece iş parçacıkları arasında iletişim kolaylaştırılarak yüksek bellek bant genişliği sağlanmıştır.

İki mimari arasındaki bir diğer fark da bellek gecikmesidir. CPU'da bellek gecikmesi, geniş arabellekler ve tahmin mekanizmaları ile yönetilir. Bunlar tasarım üzerinde büyük yer kaplarlar ve genellikle güç tüketimi açısından savrukturlar. GPU ise gecikmeyi binlerce iş parçacığını bir anda destekleyerek yönetir. Eğer herhangi bir iş parçacığı bellekten yük bekliyorsa, GPU hiçbir gecikmeye sebep olmadan derhal başka bir işe geçiş yapar. CPU biriktirme ara belleğini, genel bellek erişim gecikmesini azaltarak performansı arttırmak amacıyla kullanır. GPU ise ara belleği (ya da yazılımla yönetilen paylaşılan belleği) bant genişliğini arttırmak için kullanır.

4.3. CUDA ile Paralel Programlama

Grafik işlemciler, merkezi işlemci birimlerine göre çok daha fazla işlemci çekirdeğine sahiptirler. Örneğin Intel i7 işlemcisi 4 çekirdeğe sahip iken, Nvidia Tesla K20 grafik işlemcisi 2496 çekirdeğe sahiptir. Her ne kadar grafik işlemcilerinin her bir çekirdeğinin hızı, merkezi işlemci birimlerinin çekirdek hızından çok daha düşük olsa da çok sayıda çekirdek grafik işlemcileri paralel programlamaya daha uygun hale getirmektedir. Veri paralelliğinde de, görev (task) paralelliğinde de çok sayıda grafik işlemci çekirdeği avantaj sağlamaktadır. Yapılacak işlemlerin de paralel programlamaya uygun olma durumu kritik bir rol oynamaktadır. Yapılacak işlemlerin paralel programlamaya uygun olması da paralel programlama yapılabilmesi açısından en kritik durumlardandır. Eğer programlamada kullanılacak veri seti ve algoritma, paralel işlenebilir hale getirilebilirse, her bir işlem grafik

işlemci mimarisinde yer alan blok ve iş parçacığı (thread) kavramlarına dağıtılarak paralel işlenebilir. GPU blok thread mimarisiyle ilgili yapı Şekil 4.4'de görülmektedir.



Şekil 4.4. GPU'da Blok-Thread Yapısı [8].

Grafik işlemcilerde oluşturulan çekirdek fonksiyonlarında blok ve iş parçacığı (thread) yapıları kullanılarak her blokta işlenecek iş parçacığı sayısı ayarlanabilmekte, böylelikle fonksiyon tamamlama hızı ve paralel işlem sayısı belirlenebilmektedir. Şekil 4.4'de ızgara-blok-iş parçacığı hiyerarşisi gösterilmiştir. Her blok içerisinde yer alan thread'ler birbirlerinden bağımsız olarak çalışabilmektedirler. Bloklar da aynı şekilde basit bir anlatımla bağımsız çalışma gruplarını temsil etmektedirler.

Fonksiyonların çağrımında blok ve thread kullanım şekli aşağıda verilen örnekteki gibidir:

- kernelFonksiyon <<< blok, thread >>>(float *fIn);
  - *thread:* İşlenecek eleman sayısına göre belirlenebilen, her blokta kaç adet thread çalışması gerektiğini gösteren değerdir. Grafik işlemci kartının özelliklerine ve mimarisine göre buraya verilecek değerin performansa etkisi değişmektedir. Tez kapsamında bu değer 256 olarak belirlenmiş ve CUDA yazılımında kullanılan bütün fonksiyonlarda bu değer kullanılmıştır.
  - *blok:* İşlenecek eleman sayısına göre ayarlanmış blok sayısıdır. Bu sayı belirlenmiş thread sayısına göre değişmektedir. Bu örnek işlem için, işlenecek veri boyu 1024 olarak verilmiş olup, 256 adet thread sayısı belirlendiğinde blok sayısı da 4 olarak görülmektedir.

- CUDA koduyla çalışan fonksiyon içeriği;

```
__global__ void kernelFonksiyon ( float *fIn)
{
        int idx = blockIdx.x*blockDim.x + threadIdx.x;
        fIn [idx] = idx;
}
```

  - *blockIdx.x:* İşlem yapılırken hangi bloğun çalıştığını belirten indeks bilgisidir. Bu bilgi ile hangi blokta işlem yapıldığı anlaşılır. Bu örnek kapsamında blockIdx.x değeri 0 ile 4 arasında değerler almaktadır.
  - *blockDim.x:* İşlemler esnasında kullanılan blokların boylarını, her bir bloğun içerdiği thread sayısını belirtir. Verilmiş olan bu örnek için *blockDim.x* değeri 256 olarak görülecektir.
  - *threadIdx.x:* İşlem yapılırken hangi thread'in çalıştığını belirten index bilgisidir. Bu bilgi ile hangi thread'in işlem yaptığı anlaşılır. Verilmiş olan bu örnek kapsamında threadIdx.x değeri 0 ile 256 arasında görülecektir.
  - *idx:* Hesaplanan blokId ve threadId değerleri ile belirlenen indeks değeridir. Bu değer girdi verisindeki hangi elemanın işleneceğini göstermektedir.
- kernelFonksiyon çalıştırıldıktan sonra elde edilen çıktı bilgisi şu şekilde olur;
  - fIn[0] = 0
  - fIn[1] = 1
  - fIn[2] = 2
  - fIn[3] = 3
  - fIn[4] = 4

    .

    .

    .
  - fIn[1020] = 1020
  - fIn[1021] = 1021
  - fIn[1022] = 1022
  - fIn[1023] = 1023.

5. IONOLAB-RAY ALGORİTMASININ GRAFİK İŞLEMCİLERE UYARLANMASI

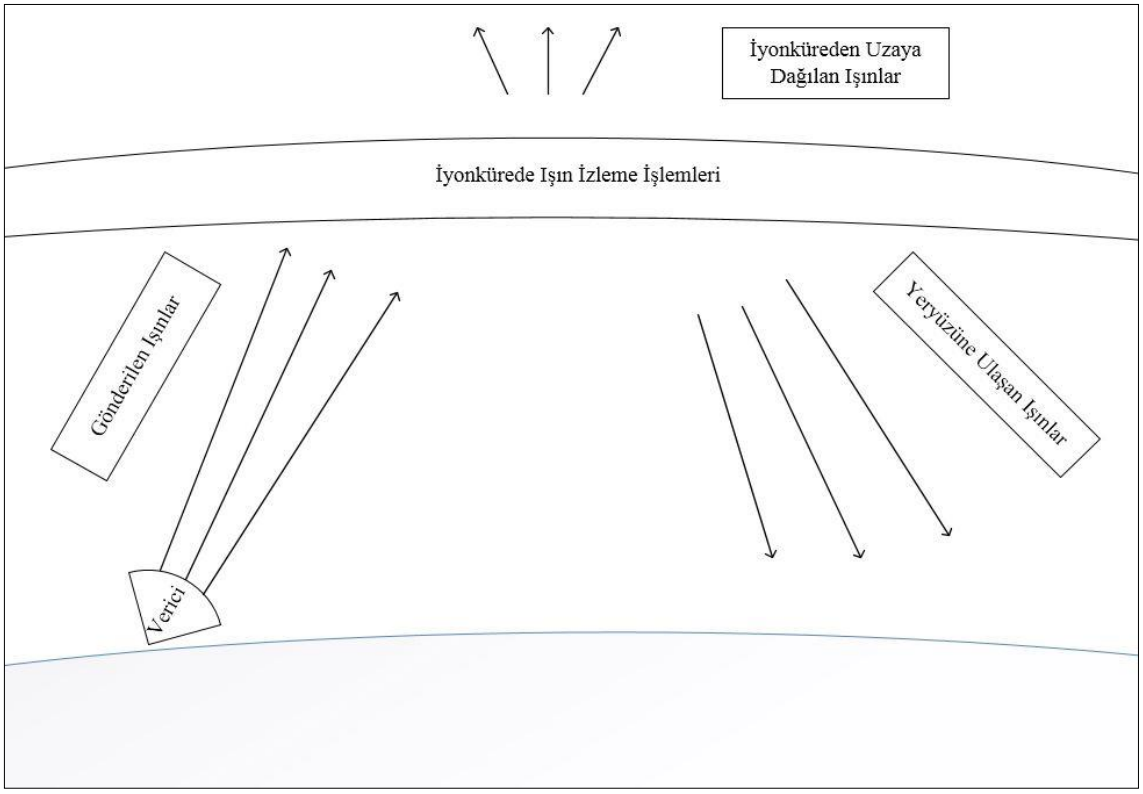
Bu bölümde tez kapsamında ışın izleme işlemlerinin grafik işlemciler kullanılarak paralel programlanabilmesi adına yapılmış olan faaliyetlerden bahsedilmiştir. Işın izleme yönteminin grafik işlemcilerde çalıştırılan CUDA yazılımı ile paralel programlanması için ilk olarak kullanılacak algoritmanın paralel programlanabilirliğinin analiz edilmesi gerekmektedir. Algoritmanın paralel işleme uygunluğu bu bölümde değerlendirilmiştir. Paralel olmayan bazı bölümlerin farklı yöntemlerle paralelleştirmesi sağlanmıştır. Sonrasında tez kapsamında kullanılan grafik işlemcinin niteliklerine uygun bir tasarım çalışması gerçekleştirilmiştir. Grafik işlemciler için kısıt kabul edilen grafik işlemcisi hafıza alanının algoritma işlemleri için yeterliliği analiz edilmiştir. Algoritma içerisinde fonksiyon grupları ayrıştırılarak bir akış şeması çıkartılmıştır. Elde edilen analiz ve tasarım çalışmaları sonrasında MATLAB programı ile çalışmalar gerçekleştirilmiştir. MATLAB kodu ve test verisi bulunan IONOLAB-RAY algoritması yine MATLAB programı üzerinde çalıştırılarak gözlemler gerçekleştirilmiştir.

CUDA, grafik işlemci birimlerini üreten en önemli ve en büyük firmalardan biri olan Nvidia'nın bir ürünüdür. Yapılmış olan bu tez kapsamında paralel programlama yazılım faaliyetleri gerçekleştirilmesinde bu ürün kullanılmıştır. Yazılım faaliyetlerinin gerçekleştirilmesinin ardından grafik işlemcide çalıştırılacak CUDA kodunun eniyilemesi için çalışmalar yapılmıştır. Son olarak MATLAB sonuçları ile grafik işlemcide çalıştırılan CUDA kodunun sonuçları karşılaştırılarak ölçüm ve analiz çalışmaları yapılmıştır. Aşağıdaki alt bölümlerde sırasıyla yapılmış olan çalışmalar ayrıntılı olarak anlatılmıştır.

5.1. Analiz ve Tasarım

Bu bölümde tez kapsamında kullanılacak olan ışın izleme algoritmasının analiz ve tasarım çalışmaları gerçekleştirilmiştir. Algoritmanın paralel kullanılabilirliği üzerinde incelemeler yapılıp, bazı bölümlerde düzeltme ve değiştirmeler yapılarak işlemlerin paralel programlamaya uygun hale getirilmesi sağlanmıştır. Temel olarak incelendiğinde IONOLAB-RAY algoritmasında ışınların ilerleyişi Şekil 5.1'deki gibi görülmektedir.

Vericiden gönderilen sinyallere iyonkürede ışın izleme işlemleri uygulanarak ışınların dalga yayılım yolları hesaplanmaktadır.

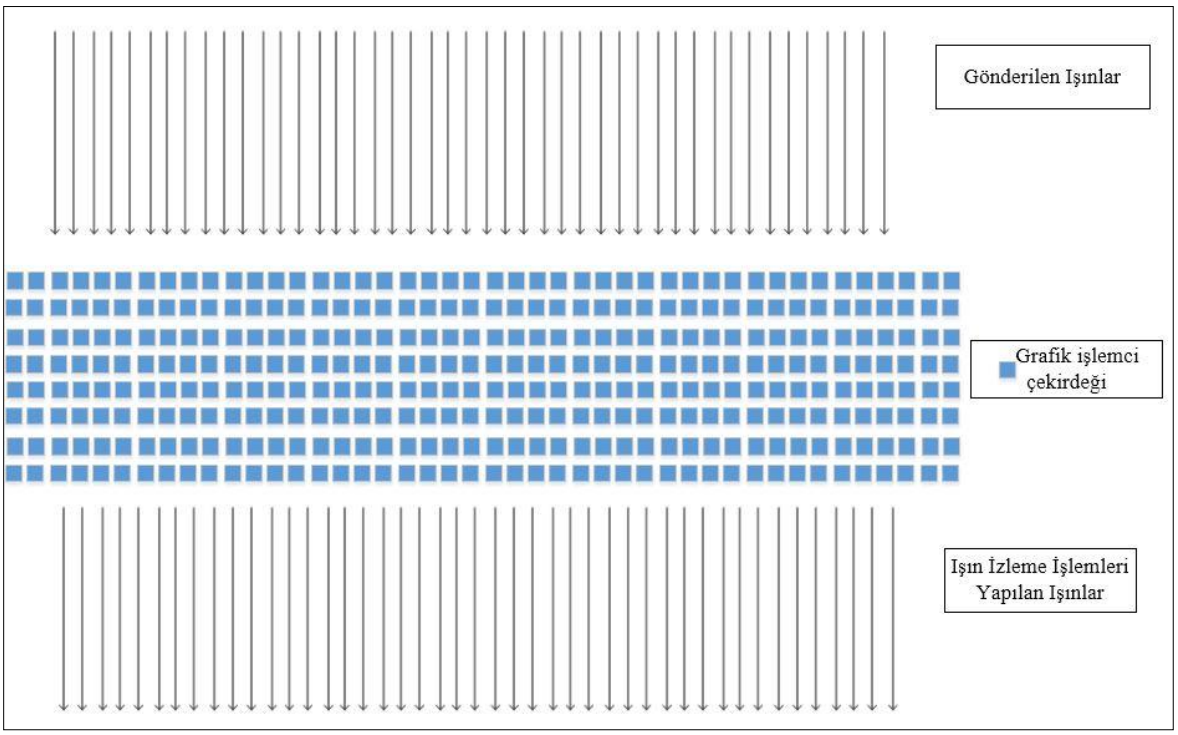


Şekil 5.1. IONOLAB-RAY algoritmasının genel işleyişi.

IONOLAB-RAY algoritmasında gönderilecek olan ışınlar sırayla teker teker hesaplanmaktadır. Şekil 5.1'deki gibi aynı anda birbirlerinden bağımsız gönderilip paralel bir şekilde ışın yolları hesaplanabilecek olan sinyaller IONOLAB-RAY algoritmasında sıralı bir şekilde hesaplanmaktadır. Ayrıca, iyonküre analizinde, ufuk ötesi radarların başarı oranlarının artırılmasında, KD haberleşme sistemlerindeki analiz ve benzetim çalışmalarında binlerce sinyalin kısa sürede işlenme gereksinimi bulunmaktadır. Hem ışınların dalga yollarının birbirlerinden bağımsız hesaplanması hem de kısa sürede binlerce ışının dalga yollarının hesaplanması gereksinimi, IONOLAB-RAY algoritmasında sıralı şekilde yapılan dalga yolu hesaplama işlemlerinin grafik işlemcilerde çalıştırılan CUDA yazılımı ile paralel programlanmasının önünü açan etmenler olarak değerlendirilmiştir.

Grafik işlemcilerde çalıştırılan CUDA yazılımında grafik işlemcinin sahip olduğu CUDA çekirdekleri kullanılabilmektedir. Bu çekirdeklerin sayısı kullanılan grafik işlemciye göre değişebilmektedir. Bu tezde yapılan çalışmalar kapsamında kullanılan grafik işlemcinin

CUDA çekirdeği sayısı 2496'dır. Yapılan çalışmalar dahilinde, IONOLAB-RAY algoritmasında ışınların teker teker algoritmaya beslenip ışın izleme işlemleri uygulanması yerine aynı anda birçok ışının farklı grafik işlemci çekirdeklerine beslenip işlenmesi şeklinde bir tasarım düşünülmüştür. Her bir ışın için farklı grafik işlemci çekirdeğinde ışın izleme işlemleri uygulanarak, binlerce sinyalin herbirinin farklı grafik işlemci çekirdeklerinde aynı anda işlenmesi sağlanmıştır. Bağımsız ışınların, bağımsız grafik işlemci çekirdeklerinde ışın izleme işlemlerini yürütmesi görsel olarak Şekil 5.2'de sunulmuştur.



Şekil 5.2. GPU ile IONOLAB-RAY algoritması

Bu tezde yapılan çalışmalar kapsamında, modellenen ve elektron yoğunluğu bilinen bir iyonküre bölgesinde, farklı verici parametrelerine sahip ışınların, farklı grafik işlemci çekirdeklerinde aynı anda paralel işlenmesi sağlanmıştır. Ayrıca, ışınların iyonkürede kırılarak ilerlemesi, farklı elektron yoğunluğuna sahip ortamlarda yön değiştirmesi gibi işlemlerin, grafik işlemcilerde çalıştırılması için kullanılan CUDA yazılımı ile yapılabilirliği de incelenmiştir.

Işının vericiden çıkış yönünü tanımlayan yanca ve yükseliş açıları ile frekansa bağlı olarak çok sayıda farklı ışının tanımlanması, her bir ışın için yapılan hesaplamalarda ışınların sıklıkla farklı ortamlara geçişlerinin olması ve bu ortam geçişleri sırasında birçok kontrol

mekanizmasının çalışıyor olması GPU kullanımlarında bir dezavantaj olarak değerlendirilmiştir. Çünkü bir GPU çekirdeğinin işlem gücü bir CPU çekirdeğinin işlem gücünden düşüktür, yazılımda kontrol mekanizmalarının çok olması GPU'lar için bir dezavantaj oluşturmaktadır. CUDA ile paralel programlamada başarılı olunabilinmesi için ışınların birbirlerinden bağımsız işlenebilir olması ve binlerce ışının kısa sürelerde işlenebilir olma gereksinimi, bahsi geçen dezavantajı tolere edebilir düzeyde olmalıdır. Bu tasarım kıstasları temel alınarak inceleme çalışmaları gerçekleştirilmiş olup, IONOLAB-RAY algoritmasının MATLAB kod analizi ile çalışmalara devam edilmiştir.

- IONOLAB-RAY algoritması girdi olarak;
  - Frekans bilgisi,
  - Enlem bilgisi,
  - Boylam bilgisi,
  - Yükseklik bilgisi,
  - Yükseliş açı bilgisi,
  - Yanca açı bilgisi,

verilerini kullanmaktadır. IONOLAB-RAY algoritmasına girdi olarak kullanılan parametreler analiz edildiğinde yukarıdaki değerlerin kullanıldığı ve bu değerlerin modellenmiş iyonküre üzerinde vericiye ait bilgiler olduğu görülmektedir. Bu parametrelere sahip olan ışınlar iyonküre üzerinde, daha önceki bölümlerde bahsedilmiş olunan Snell Yasası prensibiyle kırılım ve yansıma gerçekleştirerek ilerlemektedir. Algoritmaya belirli bir iyonküre alanının elektron yoğunluk bilgileri ve kırılma inidisi için kullanılacak girdi bilgileri verildiğinde, ışın izleme yöntemi ile ışınların yönelimi hesaplanabilmektedir. Vericiye ait girdi bilgileri, çıktı bilgileri ve tanımları aşağıda belirtilmiştir.

Frekans: Gönderilecek ışının frekans bilgisidir.

Enlem: Vericinin yerküre üzerindeki enlem bilgisidir.

Boylam: Vericinin yerküre üzerindeki boylam bilgisidir.

Yükseklik: Vericinin yerküre üzerinde, deniz seviyesinden ne kadar yüksekte olduğunun bilgisidir.

Yükseliş: Gönderilecek olan ışının yükseliş açı bilgisidir.

Yanca: Gönderilecek olan ışının yanca açı bilgisidir.

Sıradan ve sıradışı dalga: Işın izleme işlemlerinde farklı kırılma indisleri ile hesaplanan sıradan ve sıradışı dalga bilgileridir.

Çıktı: Işın izleme işlemleri ile hesaplanan sıradan ve sıradışı dalga bilgileri, gerçekleşen her bir kırılım, saçılım ve dağılım işleminde hafızada tutulur. Bu tutalan bilgiler ışının izlediği yolu çizdirebilmek için kullanılır.

- IONOLAB-RAY algoritması girdi parametreleriyle beslendikten sonra ışın izleme işlemlerinin gerçekleştiği ana işlemler bölümü çalıştırılmaktadır. Bu bölümde yapılan hesaplamalarda ilerleyen dalga için sırasıyla;
  - DKY'ten YMYS'ye çevirim, (referans noktası için)
  - DKY'ten YMYS'ye çevirim, (ışın için)
  - Birim vektör hesabı,
  - Küresel kesişim vektör hesabı,
  - YMYS'den EBY'ye çevirim,
  - Frekans bulma,
  - Döngü içerisinde;
    - DKY'ten YMYS'ye çevirim, (referans için)
    - DKY'ten YMYS'ye çevirim, (frekans için)
    - Yüzeyin normalinin hesaplanması,
    - Appleton-Hartree Formülü'nün uygulaması,
    - Yayılım vektörünün hesaplanması,
    - Yön bulma işlemleri,
    - Yüzey kesişim vektör hesabı,
    - Küresel kesişim vektör hesabı,
    - Küresel çevrimlerin yapılması,
    - Işının hangi yöne gideceğinin açısal olarak bulunması,
    - Appleton-Hartree Formülü'nün uygulaması devam ettirilerek,

    ışın izleme tamamlanana kadar döngü sürer.
  - Çıkan ışın dalga yoluna ait bilgiler ilgili yerlerde tutulması,
  - İstenirse ışının ilerleyiş görüntüsü grafiksel olarak çizdirilmesi,

işlemleri yapılmaktadır.

- IONOLAB-RAY algoritmasının analizi sonrası yazılımsal çalışmalar yapılmıştır. Işın izleme işlemleri gerçekleşmesi için gerek duyulan bütün sabit parametreler yazılıma değişmez olarak eklenmiştir. IONOLAB-RAY'de incelenen fonksiyonların herbiri için CUDA kodunda çalışacak fonksiyon tanımı yapılmıştır. Bu fonksiyonlardan bazıları ve işlevleri aşağıdaki maddelerde belirtilmiştir.
  - *txtOku:* Yazılımına girdi bilgisi şeklinde verilecek olan parametrelerin düzenlenebildiği, konfigürasyon dosyasını okuyup ilgili değişkenlere aktaran fonksiyonu ifade etmektedir. Burada okunup, kullanılacak olan konfigürasyon dosyasının içeriklerine dair bilgiler Ek-1'de verilmiştir.
  - *gpu_Acilis*: Gerekli hafıza alanı açma işlemlerinin ve dosyadan okuma işlemlerinin yapıldığı fonksiyondur.
  - *gpu_main*: Yazılımın başladığı ana işlev fonksiyonudur.
  - *gpu_fOrdinary*: Sıradan dalga için hesaplamaların yapıldığı fonksiyondur.
  - *gpu_fExtraOrdinary*: Sıradışı dalga için hesaplamaların yapıldığı fonksiyondur.
  - *geo2ecef*: Kartezyen koordinatlarda, jeodezik koordinat sisteminden YMYS koordinat sistemine geçiş işlemlerinin yapıldığı fonksiyondur.
  - *spheroidgeo2ecef*: Küresel koordinatlarda, jeodezik koordinat sisteminden YMYS koordinat sistemine geçiş işlemlerinin yapıldığı fonksiyondur.
  - *enu2ecef*: DKY koordinat sisteminden YMYS koordinat sistemine geçiş işlemlerinin yapıldığı fonksiyondur.
  - *ecef2lla*: YMYS koordinat sisteminden EBY koordinat sistemine geçiş işlemlerinin yapıldığı fonksiyondur.
  - *lla2ecef*: EBY koordinat sisteminden YMYS koordinat sistemine geçiş işlemlerinin yapıldığı fonksiyondur.
  - *ecef2enu*: YMYS koordinat sisteminden DKY koordinat sistemine geçiş işlemlerinin yapıldığı fonksiyondur.
  - *fpropagationVector*: Yayılım vektörünün hesaplandığı fonksiyondur.
  - *fpropagationDirection*: Yayılım yönünün hesaplandığı fonksiyondur.
  - *DosyalarınKaydedilmesi* : Hesaplanmış olan sıradan ve sıradışı dalga yollarına ait bilgilerin ayrı ayrı dosyalara kaydedildiği bölümlerdir.
  - *gpu_Kapanis* : Hafıza alanlarının serbest bırakılması gibi kapanış işlemlerinin yapıldığı fonksiyondur.
- Grafik işlemcilerde çalıştırılan CUDA yazılımı ile paralel programlama adına yapılmış olan temel çalışma Şekil 5.2'deki gibi her bir ışının farklı grafik işlemci çekirdeğinde

yapılması üzerinedir. 1024 adet ışının aynı anda işlenebilmesi için bir tasarım yapılmıştır. Her bir thread için bir ışın verilerek ışın izleme işlemleri gerçekleştirilmiştir. 1024 ışın için 4 blok, 256 thread kullanılarak işlemler yapılmış ve sonuç olarak 1024 ışının ışın dalga yolu hesaplanmıştır.

- Yazılımda yapılan işlemlerle sıradan ve sıradışı dalga için çıktılar üretilmektedir. Elde edilen çıktı bilgileri ışının izleyeceği yolu göstermektedir. Bu bilgiler ile elde edilen sonuçlar Bölüm 6'da grafiksel olarak gösterilecektir. Hem sıradan hem de sıradışı dalga biçimleri için elde edilen çıktılar aşağıda maddeler halinde belirtilmiştir. Bunlar:
  - Işının iyonkürede kaç kez kırılım, saçılım ve dağılıma uğradığı bilgisi,
  - Işının her bir kırılım, saçılım ve dağılım noktasında DKY koordinat sistemine göre konum bilgileri,
  - Işının her bir kırılım, saçılım ve dağılım noktasında EBY koordinat sistemine göre konum bilgileri,
  - Işının her bir kırılım, saçılım ve dağılım noktasında YMYS koordinat sistemine göre konum bilgileridir.

Hem IONOLAB-RAY algoritmasının analizine hem de CUDA yazılımında yapılmış olan tasarım faaliyetine ait bilgiler bu bölümde aktarılmıştır. Grafik işlemcilerde çalıştırılan CUDA yazılımında yapılan işlemler, ışın izlemede ışınların bağımsız incelenebilirliği sebebiyle farklı grafik işlemci çekirdeklerinde farklı ışınların işlenmesi şeklinde gerçeklenmiştir. Bu çalışmalara ek olarak, CUDA ile yazılacak kodun kalitesini ve performansını artırabilmek amacıyla öncelikle algoritmanın C kodunun yazılması planlanmıştır. C kodunun yazılmasının ardından CUDA kodu yazımına geçişin daha tutarlı ve ölçülebilir olduğu değerlendirilmiştir. C kodu ile yapılan süre ölçüm faaliyetlerine Bölüm 6'da değinilmiştir. Analiz ve tasarım çalışmaları hakkında verilen kapsamlı bilgilerin ardından sırasıyla MATLAB çalışmaları, CUDA yazılımı ve eniyileme faaliyetleri ile ilgili bilgiler verilmiştir.

5.2. MATLAB Çalışmaları

Tez çalışmasında kullanılan IONOLAB-RAY algoritmasının iyonkürenin 3 boyutlu küresel ızgara modelinde ışın izleme işlemleri aşağıdaki adımlardan oluşmaktadır:

1. İşlemlere dahil olan coğrafi bölgenin sınırlarını belirleyen enlem, boylam ve yükseklik alt/üst değerleri girilir,

2. Izgara yapısının çözünürlüğünü belirleyen enlem, boylam ve yükseklik adım aralıkları girilir,
3. Analizi yapılmak istenen boyutta tarih ve saat:dakika değerleri girilir,
4. Çalışma frekansı girilir,
5. Çalışılacak bölgedeki elektron yoğunluğu, iyonların yoğunlukları, güneş lekesi sayısı ve güneş yayılım akısı değerleri tutulur,
6. Algoritmadaki Appleton – Hartree Formülü'nde ışının hücreye geliş açısına bağlı olmayan tüm bileşenleri hesaplanır,
7. Işının kaynak noktasının konumu (enlem, boylam, yükseklik) ve ışının çıkış açıları kullanıcı tarafından girilir,
8. Kaynağın konumuna karşılık IGRF çalıştırılır. Dünyanın manyetik alanı bileşenleri (YMYS koordinatlarında) elde edilir,
9. İlerleyen ışının yayılım vektörü hesaplanır,
10. Işının yayılım vektörü ile bir sonraki yükseklikten geçen kürenin kesişim noktası belirlenir,
11. Işının yayılım vektörü ile bir sonraki yanal yüzeyin kesişim noktası belirlenir,
12. Işının hangi yüzey ile daha yakında kesiştiği kontrol edilerek ışıma yolu üzerindeki bir sonraki nokta elde edilir,
13. Kesişim noktası çıktı olarak ilgili matriste tutulur. Eğer kesişim noktası yeryüzüne temas etmemiş ise 17. Maddeye gidilir,
14. Sınır bölgesi için geliş açısı ve daha önce hesaplanan ilgili veriler kullanılarak Appleton – Hartree Formülü ile kırılma indisi hesaplanır,
15. Bu kesişimin olduğu yüzey ile yansıma / iletim açısı Snell Yasası ile hesaplanır,
16. 12. Maddeye gidilir,
17. Kesişim noktalarının tutulduğu matris grafik haline getirilir ve ışının izlediği yol bu grafik üzerinden takip edilir [4].

Algoritmada ilk on adımda verilen hesaplamalar ön koşum olarak düşünülebilir. Kullanıcının ilgi alanı olan tüm coğrafi bölge ve zaman değerleri için bu kısım önceden koşturularak çıktıları kaydedilebilir. Sonrasında kaynağın konumu ve ışının çıkış açısına göre çıktılar kısa koşum süreleri ile elde edilebilir [4].

IONOLAB-RAY algoritması yukarıda belirtilen sıra ile işlemlerini yapmaktadır. Tez çalışması kapsamında her bir adım kontrollü olarak beslenip, sonuçları analiz edilmiştir. Her bir alt fonksiyonun giriş ve çıkışına kesme noktaları koyarak fonksiyonun grafik işlemcide

çalıştırılan CUDA yazılımı ile koordine olup olmadığı incelenmiştir. Yazılımın doğruluğunu ispatlayabilmek adına birçok kontrol kodu yazılmıştır. Kontrol amaçlı olarak bu kodlarda, MATLAB programından üretilen çıktı verisi ile CUDA kodundan üretilen çıktı verisinin çıkan sonuçları okunup, karşılaştırılabilmektedir. Burada yapılmış olan nümerik doğrulama yöntemi ile ilgili ayrıntılı bilgiler Bölüm 6'da verilmiştir.

Yapılmış olan bu tez çalışmaları kapsamında IONOLAB-RAY kodu ile grafik işlemcide çalıştırılan CUDA kodunun çıktıları karşılaştırdıktan sonra süre ölçümü yapılmıştır. Sıradan dalga ve sıradışı dalganın hesaplanmasında IONOLAB-RAY algoritmasının MATLAB'da alınan süre ölçümlerinde, MATLAB'ın sağlamış olduğu "tic-toc" fonksiyonları kullanılmıştır. CUDA yazılımındaki ölçümlerde ise Ek-3 içerisinde görülen süre ölçüm fonksiyonları kullanılmıştır. CUDA yazılımında ölçülecek olan toplam süreye dahil edilen işlemler şu şekildedir:

- Kullanılacak verilerin CPU'dan GPU'ya kopyalanma süresi,
- Grafik işlemciden çalıştırılan CUDA yazılımında ışın izleme işlemlerinin yapılmış olduğu ana işlemlerin toplam süresi.

MATLAB'da tek bir ışının sıradan ışın izleme işlemlerinin yapılması için geçen süre yaklaşık 2.5 saniyedir. Bu süre, iyonkürede incelenecek alanın büyüklüğüne, ışının ne kadar kırılım ve yansıma yapacağı gibi değerlere bağlı olarak artıp azalabilmektedir. 2.5 saniye olan süre sıradan ışın hesabı için yapılan işlemlerin süresi iken, sıradışı ışın hesabındaki işlem süresi yaklaşık 2.4 saniye olarak ölçülmektedir. Yalnızca sıradan ışın hesaplaması temel alındığında, elde edilmiş olan yaklaşık 2.5 saniyelik işlem süresi sadece bir ışının ışın izleme yöntemiyle yolunun çözümlenmesi anlamına gelmektedir. Sadece deneme ve test amaçlı yapılan bu tek ışın içeren çalışmadan yola çıkarak asıl testlerde ihtiyaç duyulan binlerce ışının MATLAB'da hesaplanmasının ne kadar uzun süre alacağı görülmektedir.

- 100 ışın için tahmini MATLAB işleyiş süresi ≈ 250 saniye.
- 1000 ışın için tahmini MATLAB işleyiş süresi ≈ 40 dakika.
- 10000 ışın için tahmini MATLAB işleyiş süresi ≈7 saat.
- 100000 ışın için tahmini MATLAB işleyiş süresi ≈ 3 gün.
- 1000000 ışın için tahmini MATLAB işleyiş süresi ≈ 28 gün.

Bu süre ölçüm bilgileri tezin ilerleyen aşamalarında oldukça faydalı olmuştur. Elde edilen bu bilgiler, yazılımda gelinen noktanın verimliliği veya kullanılabilirliliği gibi konuları

MATLAB sonuçlarıyla kıyaslayabilmek için elde edilen en önemli verilerden biri olmaktadır.

5.3. Yazılım Çalışmaları

Algoritmanın analizi, incelenmesi ve tasarım çalışmalarının ardından yazılımsal faaliyetlere gerçekleştirilmiştir. Yapılmış olan bu yazılımsal tez çalışmaları, Nvidia firmasının Tesla K20 isimli ekran kartı üzerinde gerçekleştirilmiştir. Bu karta ait başlıca donanımsal bilgiler aşağıdaki gibidir:

- 3.52 Tflops tek kayan nokta çözünürlükte işlemci performansı,
- 1.33 Tflops çift çözünürlükte işlemci performansı,
- 2496 CUDA çekirdeği,
- 5 GB DDR5 grafik işlemci hafızası,
- 208 GB/s hafıza band genişliği.

CUDA yazılımı içerisinde grafik işlemciye ait diğer bilgiler yazılım çalışmaya başladığında ekrana yazdırılacak şekilde ayarlanmıştır. Grafik işlemciye ait blok, thread sayısı ve yazmaç sayısı gibi bilgilerin yer aldığı ekran görüntüsü Ek-2'de yer almaktadır.

Özellikleri yukarıda da belirtilmiş olan bu grafik işlemci kartı üzerinde yaklaşık 5GB'lık hafıza alanı göz önünde bulundurularak bir hafıza analizi yapılmıştır. Grafik işlemciler direkt olarak bilgisayarın hafızasına erişemedikleri için yapılan bu hafıza alanı planlaması önemlidir. Bilgisayarın kendi hafıza alanı 512 GB olsa bile, grafik işlemcide kullanılabilir alan yalnızca kullanılacak grafik işlemcinin kendi hafızasıyla doğru orantılıdır. Yapılmış olan bu tez çalışmaları da bu yaklaşık 5 GB'lık bu grafik işlemci alanına göre planlanarak gerçekleştirilmiştir. Algoritmaya girdi verisi olarak kullanılan frekans, enlem, boylam ve yükseliş bilgilerinin kullanıldığı vektörler için hem merkezi işlemci biriminde hem de grafik işlemci biriminde maksimum vektör boyları için ayrı ayrı hafıza alanı ayırması yapılmıştır. Sonrasında hem incelenecek iyonküre bilgilerinin hem de vericiye ait bilgilerin girilebildiği konfigürasyon dosyası hazırlanmıştır. Bu konfigürasyon parametreleri ile yapılan alan ayrılması işlerinin ardından, algoritmada kullanılacak girdi bilgilerinin okunması işlemleri yapılmıştır. Grafik işlemcide çalıştırılan CUDA yazılımında kullanılacak olan girdi bilgileri aşağıda maddeler halinde verilmiştir:

- *Freakans bilgisi*: Kaynaktan çıkan, dalganın frekansı, frekans dizisi, (MHz cinsinden)

- *Enlem bilgisi*: Kaynağın konum enlemi, (° "derece" cinsinden)

- *Boylam bilgisi*: Kaynağın konum boylamı, (° "derece" cinsinden)

- *Yükseklik bilgisi*: Kaynağın konum yüksekliği, (m cinsinden)

- *Yükseliş açı bilgisi*: Dalganın kaynaktan çıkış doğrultusu yükseliş açısı, (° "derece" cinsinden)

- *Yanca açı bilgisi*: Dalganın kaynaktan çıkış doğrultusu yanca açısı, (° "derece" cinsinden)

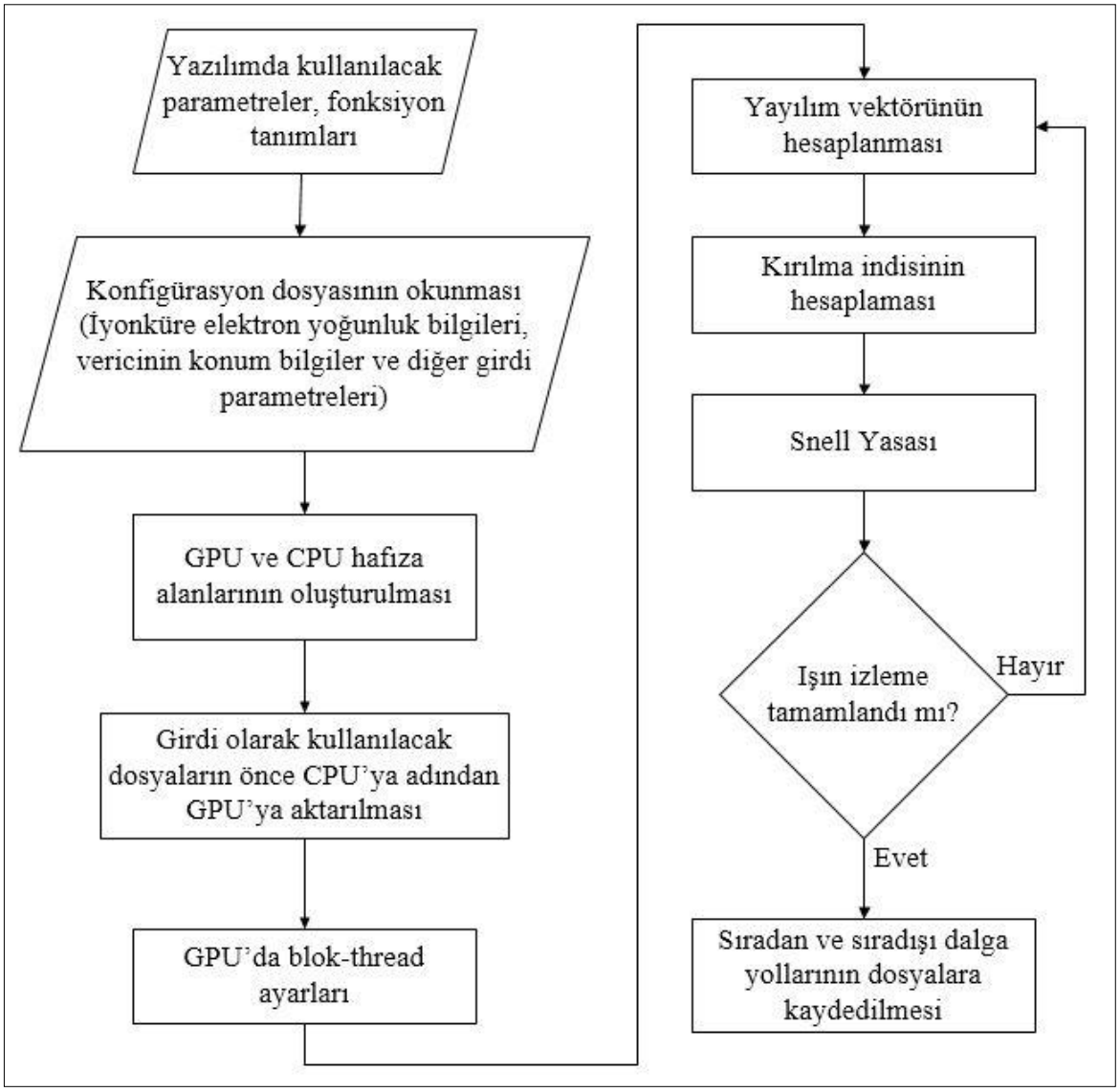
- *Ön işlem sonrası girdi bilgileri*: IRI-Plas yazılımı ile elde edilen, kırılma indisi için gereken tüm iyonküre parametrelerinin bulunduğu bilgileri içermektedir.

Tüm bu girdi bilgileri yazılımda önce merkezi işlemci hafızasına ardından grafik işlemci hafızasına alındıktan sonra paralel programlama için gerekli blok/thread ayarlaması yapılmaya başlanmıştır. Hesaplanacak olan her bir ışının ışın izlemesini bir thread yapacak şekilde paralel programlama uygulamasına ön tasarım kısmında karar verilmiştir. Bu karar ve planlama sonrasında algoritmaya girecek ışın sayısına göre CUDA yazılımında kaç blok ve kaç thread kullanılması gerektiği hesaplanmıştır. Test ve yazılım doğrulama faaliyeti kapsamında 1024 adet ışın için hesaplama yapılmış olup CUDA yazılımında,

- Her blokta 256 adet thread,

- 1024/256 = 4 adet blok,

kullanılmıştır. Bu durumda 4 adet blok kullanılıp, her bir blok içerisinde 256'şar thread görevlendirilerek her bir ışının, grafik işlemcide çalıştırılacak CUDA yazılımı ile paralel programlaması gerçekleştirilmiştir. Bu belirlenen değerler grafik işlemci kartının donanımsal nitelikleri bakımından sınır değerleri içerisinde kalmaktadır. Bahsedilen bu ayarlamalar yapıldıktan sonra, CUDA yazılımında Şekil 5.3'te gösterilen blok diyagramdaki işlemler takip edilerek IONOLAB-RAY algoritması tamamlanmıştır.



Şekil 5.3. Grafik işlemci için hazırlanan blok şema.

Grafik işlemcide çalıştırılan CUDA yazılımda 1024 ışının tamamına ışın izleme algoritması uygulanmıştır. 4 blok ve 256 thread kullanılarak grafik işlemci çekirdekleri çalıştırılmıştır. Hem global hem de lokal CUDA fonksiyonları yazılım içerisinde sıkça kullanılmıştır.

- *Global CUDA fonksiyonları:* Bu fonksiyonlar doğrudan merkezi işlemci üzerinden ulaşılabilen ve kullanılabilen fonksiyonlardır. Örneğin;
  __global__void enu2ecef_Device(float *fInputs, float *fOutputs).

- *Lokal CUDA fonksiyonları:* Bu fonksiyonlara doğrudan merkezi işlemci üzerinden ulaşılamayan fonksiyonlardır. Bu fonksiyonlar sadece başka bir CUDA fonksiyonu içerisinden ulaşılarak, çalıştırılabilirler. Örneğin;

__device__void_appleton_Device_v2(float *fInputs, float *fOutputs).

MATLAB üzerinde çalışan bütün fonksiyonlar için CUDA yazılımında lokal ya da global fonksiyonlar oluşturulmuştur. Bütün bu fonksiyonların girdi ve çıktı parametrelerinin MATLAB girdi ve çıktılarıyla aynı olmasına dikkat edilerek yazılım faaliyetleri gerçekleştirilmiştir. Sonuçların ve yazılımın doğrulanmasının ardından işlemlerin süre ölçüm faaliyetlerine geçilmiştir. Ek-3 içerisinde CUDA yazılımında gerçekleştirilen süre ölçüm faaliyeti ile alakalı yapılan ölçüm işlemi yer almaktadır. Ayrıca, MATLAB içerisinden girdi dosyalarının nasıl yazılıma besleneceği ve CUDA yazılımının çalışması sonucunda oluşan çıktı dosyalarının nasıl inceleneceğine dair ayrıntılı bilgiler Ek-6 içerisinde yer almaktadır.

1024 adet ışının sıradan ışın izleme işlemlerinin yapılması için CUDA yazılımı ile ölçülen süre yaklaşık 1.3 saniyedir. Aynı sayıda ışının MATLAB'da birebir aynı işlemlerden geçmesi ise yaklaşık 40 dakika sürmektedir. MATLAB algoritmasının hızı ile grafik işlemcide çalıştırılan CUDA kodunun işleme hızını mukayese ettiğimizde grafik işlemcide çalıştırılan CUDA kodunun yaklaşık iki bin kat daha hızlı olduğu görülmektedir.

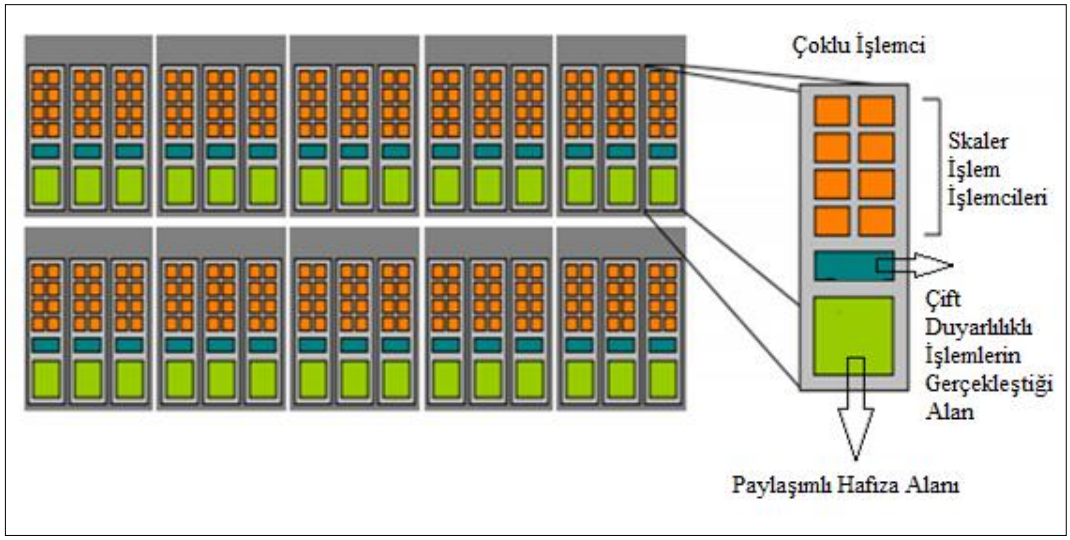
Sıradan dalga için ışın izleme işlemleri yukarıda aktarıldığı gibi yapılmakta olup, bahsedilmiş olan bu işlemlerin bir benzeri de sıradışı dalgaların hesaplanmasında kullanılmıştır. Sıradışı dalgalar için farklı kırılma indisleri kullanılarak sıradan dalgalardan farklı ve bağımsız hesaplamalar yapılmaktadır. Sıradışı dalga yolu hesaplamasında kullanılan CUDA fonksiyonu da sıradan dalga yolu hesaplaması için hazırlanan CUDA fonksiyona benzerdir. Sadece kırılım indislerinde hesaplama yöntemleri farklıdır. Sıradışı dalga yolu işlem süreleri ölçüldüğünde de sıradan dalga yolu hesaplama süresindeki gibi yaklaşık olarak iki bin katlık bir farkın olduğu görülmektedir. CUDA yazılımıyla GPU'da çalışma süresi ve MATLAB'ta ölçülen işlem sürelerine ek olarak C kodu ile yazılıp CPU'da çalıştırılan algoritmanın işlem süresi sonuçları da değerlendirilmiştir. Bu üç farklı süre ölçümünün karşılaştırmaları Bölüm 6 içerisinde tablolar halinde verilmiştir.

5.4. Eniyileme

Tez kapsamında grafik işlemcide çalıştırılacak CUDA yazılımında yapılmış olan eniyileme işlemleri bu bölümde açıklanmaktadır. CUDA yazılımlarında aşağıda belirtilen bazı temel eniyileme işlemleri uygulanmaktadır.

- *Blok-thread eniyilemesi:* Tek boyutlu blok/thread yapısı kullanılabildiği gibi iki boyutlu blok/thread yapısı da kullanılabilmektedir. Kartın özelliklerine bağlı olarak avantajları değişebilen bu değerler için en uygun değerin tek boyutlu blok/thread yapısında thread için 256 olduğu görülmüştür. Bu sebepten 4 blok, 256 thread ile işlemler gerçekleştirilmiştir.

- *Paylaşımlı hafıza (shared memory) kullanımı:* Paylaşımlı hafıza olarak belirtilen grafik işlemci hafıza alanı, yazmaçların sürekli grafik işlemci hafızasına gitmektense daha yakında ve erişimleri daha hızlı olan yerden veriyi çekmelerini sağlar. Bu sayede yapılan işlemler çok daha hızlı gerçekleşir. Eğer bir veri bir CUDA fonksiyonu içerisinde birden fazla kez kullanılacaksa o verinin ana hafızadan paylaşımlı hafızaya alınıp orada işlenmesi, işlem hızını oldukça iyileştirecek faktörlerden biridir. Paylaşımlı hafızanın donanımsal yerleşimi Şekil 5.4'te görülmektedir.



Şekil 5.4. Paylaşımlı hafıza alanı donanımsal görünümü [38].

Grafik işlemcide çalıştırılacak CUDA kodunun fonksiyonları içerisinde örnek kullanım şekli ise şu şekildedir:

```
__global__void enu2ecef_Device2(float *fX_East, float *fLat0)
{
    int i = threadIdx.x + blockIdx.x * blockDim.x;
    unsigned int tid=threadIdx.x;>>> Paylaşımlı alanda veri indeksi
    __shared__float sdatafLat0[256];  >>> Paylaşımlı hafıza alanı kullanım şekli
    sdatafLat0[tid]  = fLat0[i];>>> Ana hafızadan paylaşımlı hafıza alanına veri gönderimi
```

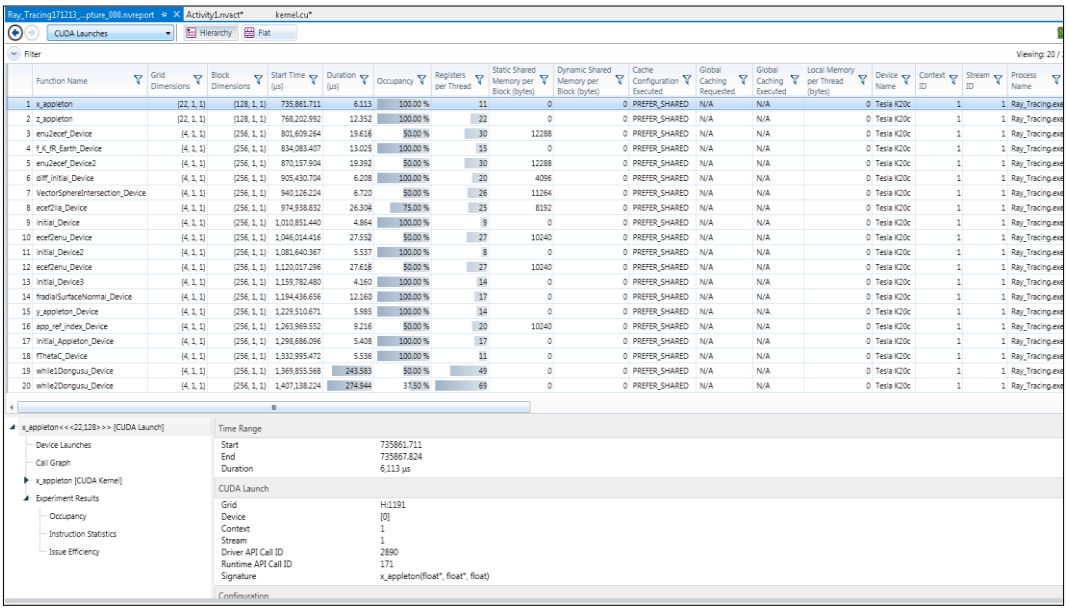
```
    sdatafLat0 [tid] = sdatafLat0[tid] *(PI_SAYISI)/180 *        sinf(sdatafLat0[tid]*
(PI_SAYISI)/180);   >>> Paylaşımlı alan üzerindeki işlemler
}
```

- *Lokal fonksiyon kullanımı:* Alt fonksiyonlar üzerinden aynı thread'ler kullanılarak lokal CUDA fonksiyonları kullanılarak eniyilemesi sağlanmıştır.
- *Özdeğer fonksiyonlarının(intrinsic funciton) kullanımı:* Bu fonksiyonlar merkezi işlem biriminde kullanılan matematik fonksiyonların yerine CUDA kodlarında kullanılan muadilleridir. Bu işlemler daha hızlı olarak grafik işlemcide çalıştırılacak CUDA kodunda yapılabilmektedir. Örnekler;

  Fonksiyon – Matematik Kütüphanesi Adı – Özdeğer Fonksiyon Adı

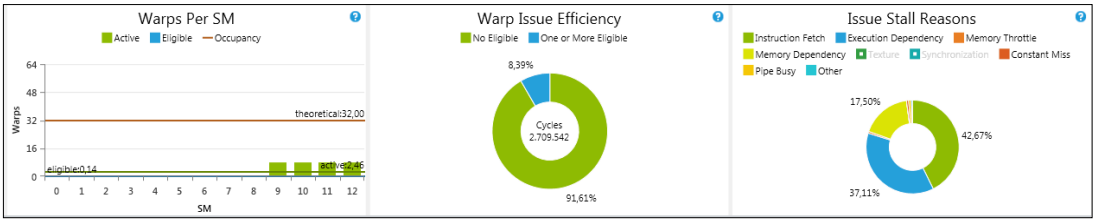
Sinüs	-	sin(x)	-	__sinf(x)
Logaritma	-	logf(x)	-	__logf(x)
Kuvvet Alma	-	powf(x,y)	-	__powf(x,y)
Exponansiyel	-	exp(x)	-	__expf(x)
Kosinüs	-	cos(x)	-	__cosf(x)
Tanjant	-	tan(x)	-	__tanf(x)

- *Cuda kütüphanelerinin kullanımı:* Nvidia'nın sunmuş olduğu hazır kütüphaneler bu tez kapsamında ihtiyaç duyulan fonksiyonlarda kullanılmak için oldukça idealdir. Algoritma içerisinde uygun olduğu görülen yerlerde bu kütüphane fonksiyonları kullanılarak algoritmanın eniyilenmesi sağlanmıştır. Örnek kütüphaneler; cufft, cublas.
- *Kernel fonksiyonlarını birleştime:* Uygun işlemlerin birleştirilerek tek bir fonksiyon içerisinde yapılması hem veri okuma/yazma işlemlerinden feragat edilmesini sağlayarak hem de paylaşımlı alandaki verinin alandan ayrılmadan kullanılmaya devam edilmesini sağlayarak işlemlerin daha hızlı yapılmasını sağlamaktadır. Tez kapsamında yapılan çalışmalarda bu durum sıklıkla göz önünde bulundurarak olabildiğince fonksiyonların birleştirilmesine çalışılmıştır.
- *Nvidia Nsight* [41] *ve Visual Profiler* [42] *araçlarının kullanımı:* Bu araçlar yazılımın analizini basit ve seri bir şekilde ayrıntılı olarak sunmaktadır. Tez kapsamında yazılmış olan CUDA yazılımı bitirildikten sonra, bu araçlar vasıtasıyla analizler yapılarak iyileştirme faaliyetleri gerçekleştirilmiştir. Bu araçlar sayesinde Şekil 5.5'te örnek verileri görülen hafıza darboğazı, yüzdesel grafik işlemcinin verimli kullanımı ve paylaşımlı hafıza alanı kullanımı gibi birçok analiz verisi incelenebilmektedir.



Şekil 5.5. Nsight analiz aracı ile CUDA yazılımının kod analizi.

Ayrıca bahsedilen bu analiz aracı ile birçok grafiksel analiz de Şekil 5.6'de görülen örnekteki gibi yapılabilmektedir. Bu avantajlar ve kullanım kolaylığı göz önünde bulundurulduğunda aracın oldukça kullanışlı ve işe yarar olduğu söylenebilmektedir.



Şekil 5.6. Verimliliklerle alakalı grafiksel görünümler.

Maddeler halinde bahsedilmiş olan bu eniyileme faaliyetleri yazılım geliştirmenin her aşamasında kullanmıştır. Bu işlemler gerçekleştirilerek yazılımın işlem süresi yaklaşık %50 oranında iyileştirilmiştir. İlk versiyonu ile yaklaşık 2.2 saniye civarı süren grafik işlemcilerde çalıştırılacak CUDA yazılımında algoritma işlem süresi, iyileştirmeler sonrasında 1.2 saniye düzeylerine çekilmiştir. Yapılmış olan kodlama ve eniyileme çalışmalarının ardından veri setleri ile deneysel çalışmalar gerçekleştirilmiştir. Bölüm 6'da yazılımsal doğrulama, grafiksel analiz ve süre ölçüm faaliyetleri ayrıntılı olarak anlatılmıştır.

6. GRAFİK İŞLEMCİLER İLE IŞIN İZLEME UYGULAMALARI

Bölüm 5'te anlatılmış olan grafiksel işlemci kullanılarak yapılan hesaplamalar sonucunda yazılımın öncelikle doğru çalışıp çalışmadığı değerlendirilmiştir. Yazılımın MATLAB kodu ile uyumlu çalıştığı ve aynı sonuçları ürettiği görüldükten sonra elektron yoğunluğu belirli olan ve girdi olarak verilen bölgede ışın izleme algoritması çalışması yapılmıştır. CUDA yazılımı ile üretilen sonuçlar, IONOLAB-RAY algoritmasının sonuçları ile karşılaştırılarak doğru sonuçların üretilip üretilmediği tespit edilmeye çalışılmıştır. İki farklı matematiksel doğrulama yöntemi kullanılmaktadır. İlk doğrulama faaliyetinde iki çıktı verisinin bütün elemanları birbirleriyle oranlanır ve iki farklı sonuç verisinin de birbirlerine ne kadar yakınsadığı tespit edilmeye çalışılır, MATLAB kodları ile yapılan bu kontrol işlemi Denklem 6.1'deki gibidir. Normal şartlarda iki çıktı bilgisinin de birbirine çok benzer sonuçlar üretmesi ve oranlanma sonuçlarının 1'e yakın değerler alması beklenir. Bütün çıktı bilgileri için bu doğrulamanın yapılmasının ardından, çıkan oran değerinin 1'den çıkarılması ve mutlak değerinin alınmasıyla birlikte karşılaştırılan iki değerin oransal farkı bulunur. Hesaplanan çıktı bilgilerinde maksimum oran farkının ne olduğunu görebilmek için ise hesaplanan bütün oransal fark değerlerinin maksimumunu bulmak gereklidir. CUDA yazılımı ile IONOLAB-RAY algoritmasının maksimum oransal farkının hesaplanmasında bu yöntem kullanılmıştır ve

$$\max(\ \mathrm{abs}\left(1-\frac{\mathrm{CUDA\_Sonuçları}}{\mathrm{IONOLAB\_Sonuçları}}\right)\ ) \tag{6.1}$$

ile gösterilir. $\mathrm{CUDA\_Sonuçları}$ ile grafik işlemcide çalıştırılan CUDA yazılımından elde edilen çıktı bilgileri ifade edilirken, $\mathrm{IONOLAB\_Sonuçları}$ ile IONOLAB-RAY algoritması ile oluşan çıktı bilgileri ifade edilmiştir. Mutlak değeri alınmış oransal hata bilgisi çizdirilerek hangi çıktı verisinde ne kadarlık oransal hatalar olduğu görülebilmektedir.

Bir diğer matematiksel doğrulama işlemi ise sayılar arasında oran değil, fark hesabının yapılması ile gerçekleştirilir ve

$$\max(\ \mathrm{abs}(\mathrm{CUDA\_Sonuçları}-\mathrm{IONOLAB\_Sonuçları})\ ) \tag{6.2}$$

ile gösterilir. Karşılaştırılacak iki çıktı bilgi setinin değerleri birbirlerinden çıkartılıp, mutlak değerleri alınır ve maksimum fark hatasının ne kadar olduğu Denklem 6.2'deki formül ile hesaplanır. Fark hatasında da, mutlak değeri alınmış fark hata bilgisi çizdirilerek hangi çıktı verisinde ne kadarlık fark hatası olduğu görülebilmektedir.

CUDA yazılımı ile üretilen çıktıların matematiksel olarak IONOLAB-RAY algoritması ile aynı sonuçları verdiğinin görülmesinin ardından, grafik işlemcide çalıştırılan CUDA yazılımının sonuçları grafiksel olarak çizdirilerek ışının izlemiş olduğu yol görsel olarak takip edilmiştir. Hem sıradan hem de sıradışı dalga için çıkan sonuçlar MATLAB ortamında çizdirilmiştir. Çizdirilen bu noktalar DKY ve EBY koordinat sisteminde elde edilen bilgiler ile yapılmıştır. Ayrıca görsel olarak karşılaştırma yapılabilmesi için hem MATLAB çıktıları hem de grafik işlemcide çalıştırılan CUDA yazılımından üretilen çıktılar birlikte çizdirilerek yazılımın doğruluğu grafiksel olarak da görülmüştür. Tezin bu bölümünde, yapılmış olan uygulamalar ve elde edilen sonuçlar ve görseller aşağıdaki alt bölümlerde sunulmaktadır. Dört uygulamada da algoritmanın girdi bilgisi olarak ihtiyaç duyduğu iyonkürenin elektron yoğunluğu, modellenmiş iyonkürenin koordinat aralıkları ve diğer parametreleri şu şekildedir:

- Enlem aralığı : 25 derece ile 45 derece arası,
- Boylam aralığı : 25 derece ile 35 derece arası,
- Tarih : 25 Mart 2011,
- Saat : 12:00 GS.

Bu tarih ve saat için belirtilen koordinat aralıklarında iyonküredeki elektron yoğunluğu bilgileri, hazırlanmış olan grafik işlemcide çalıştırılan CUDA yazılımına ve IONOLAB-RAY algoritmasına beslenmiştir. Belirtilen enlem ve boylam aralıkları arasında istenilen bir bölgeden çıkan ışının iyonkürede ilerleyerek nasıl bir yol çizeceği ve nereye iniş yapacağı aşağıdaki alt başlıklarda verilen örneklerde incelenmiştir. Yukarıda ayrıntıları verilen bu veri setiyle Örnek 1, Örnek 2, Örnek 3 ve Örnek 4'deki çalışmalar yapılmıştır. Örnek 5, Örnek 6 ve Örnek 7 çalışmalarında farklı bir veri seti kullanılmış olup, ayrıntıları Örnek 5'in içeriğinde verilmiştir. Ayrıca, Örnek 5'te kullanılan veri setinin aynı gün, farklı saatlerde alınmış veri setleriyle Örnek 8 ve Örnek 9 çalışmaları yapılmıştır. Yapılan deneylerin tamamında kullanılan grafik işlemci kartı Nvidia Tesla K20'dir. Bu grafik kartının 2496 kendine has çekirdeği bulunmaktadır. Karşılaştırma yapılan MATLAB kodunun çalıştığı sistem ise Intel i5 işlemciye sahip olup 4 çekirdekli ve 3.30 Ghz işlem gücündedir. Grafik işlemcide çalıştırılan CUDA yazılımında süre ölçümü alma işlemi Ek-3 içerisinde verilmiştir. IONOLAB-RAY algoritmasının MATLAB'da alınan süre ölçümlerinde ise MATLAB üzerinde çalıştırılan "tic-toc" fonksiyonları ile alınmıştır [12]. Ayrıca, MATLAB süre ölçümleri tek çekirdekte MATLAB çalıştırılırken alınmıştır.

6.1. Örnek 1 – İlk Veri Seti ile CUDA Yazılımı Doğrulama

Hem MATLAB hem de grafik işlemcide çalıştırılan CUDA yazılımında verici ile ilgili girdi olarak kullanılan parametreler şu şekildedir:

- Frekans : 16.5 Mhz,
- Enlem : 39.5 derece,
- Boylam : 32 derece,
- Yanca : 0 derece,
- Yükseliş: 30 derece,
- Tarih : 25 Mart 2011,
- Saat : 12:00 GS.

Verici ile ilgili verilen bu değerler, elektron yoğunluğu bilinen ilgili alan için hem MATLAB'a hem de grafik işlemcide çalıştırılan CUDA yazılımına girdi bilgisi olarak verilmektedir. Yukarıda verilmiş olan girdi bilgileriyle işlemler yapıldığında MATLAB'da sıradan dalga için elde edilen çıktı verilerinde DKY, YMYS ve EBY koordinat düzlemine ait değerler yer almaktadır. MATLAB'da elde edilen sonuçlar IONOLAB-RAY algoritmasının sonuçlarıdır.

MATLAB sonuçlarının ardından, sıradan dalga için grafik işlemcide çalışması için hazırlanan CUDA yazılımı çalıştırılmıştır. MATLAB ile aynı girdi parametreleri ile beslenen yazılımın sonucunda elde edilen sonuçlar ile CUDA yazılımının sonuçları karşılaştırılmıştır. Nümerik sonuçların karşılaştırılması için kullanılan yöntemlerle ilgili ayrıntılı bilgiler Bölüm 6'da verilmiştir. Burada kullanılan yöntem iki sonuç verisinin herbir elemanının birbirlerinden farkının maksimum olduğu yerin bulunmasıdır. Bu yöntem ile hem MATLAB hem de CUDA yazılımı sonuçları arasında maksimum farkın olduğu yer ve maksimum fark bilgisi tespit edilmiştir. Bu hesaplamalarda maksimum farkın 10^{-7} düzeylerinde olduğu görülmüştür. Bu düzeyde görülen hatanın nedeni MATLAB ve CUDA yazılımının farklı çözünürlükte çalışıyor olmasından kaynaklanmaktadır. Doğal ve sonuca etki etmeyecek düzeyde olan bu hata değeri, CUDA yazılımı ile grafik işlemcide çalıştırılan programın, IONOLAB-RAY algoritması ile eşdeğer sonuçlar ürettiğini göstermektedir.

CUDA yazılımı ve MATLAB'da çalıştırılan IONOLAB algoritmasının sonuçlarının aynı olduğu görüldükten sonra iki çıktı verisi de ayrı ayrı grafiksel olarak çizdirilmiştir. Çizdirilen

bu grafikler DKY koordinat sistemine göre elde edilen sonuçların görselleştirilmiş halleridir. Bu görseller Ek-3 bölümünde verilmiştir.

Elde edilen matematiksel ve görsel sonuçlardan da anlaşılacağı üzere grafik işlemcide çalıştırılan CUDA yazılım sonucu ile MATLAB sonuçları birbirleriyle tutarlıdır. DKY koordinat sistemine ait sonuçların yanı sıra EBY ve YMYS koordinat sistemine ait sonuçlar da analiz edilmiştir. EBY ve YMYS koordinat sistemine ait sonuç değerleri de grafik işlemcide çalıştırılan CUDA yazılımı sonucu ile MATLAB sonuçlarının tutarlı olduğunu göstermektedir. Yapılan ilk 4 örnek uygulamada DKY koordinat sistemine ait sonuçlara daha çok değinildiği için, EBY ve YMYS koordinat sistemi sonuçlarının matematiksel ve grafiksel doğrulamasına yer verilmemiştir. Elde edilen veriler karşılaştırılıp, grafik işlemcide çalıştırılan CUDA yazılımının ve MATLAB kodunun doğru çalıştığı görüldükten sonra grafiksel gösterimlerle sonuçlar değerlendirilmiştir.

İlk örnek ile yapılan çalışmaların hem matematiksel sonuçları hem de grafiksel sonuçları ayrıntılı bir şekilde karşılaştırılmıştır. Elde edilen bu veriler ışığında grafik işlemcide çalıştırılan CUDA yazılımının MATLAB'da çalıştırılan algoritma ile bire bir sonuçlara çok yakın değerlerde sonuçlar ürettiği tespit edilmiştir. Yapılan bu incelemeye ek olarak işlem süreleri ile ilgili de çalışmalar yapılmıştır. Hem MATLAB'da bir ışın için geçen işlem süresi hem de CUDA yazılımında 1024 ışın hesaplamasında geçen işlem süresi ölçülmüştür. MATLAB, C kodu, CUDA yazılımı ile yapılan süre ölçüm faaliyetlerine ait bilgiler Çizelge 6.1'de verilmiştir. Bu tabloda 1024 adet hem sıradan hem de sıradışı ışınların işlem süreleri yer almaktadır.

Çizelge 6.1. Örnek 1 – Süre ölçümleri.

Süre Ölçüm Altyapısı	**1024 sıradan ışının hesaplaması için geçen süre**	**1024 sıradışı ışının hesaplaması için geçen süre**
MATLAB	200 saniye	190 saniye
C kodu	3 saniye	3.5 saniye
CUDA yazılımı	0.2 saniye	0.22 saniye

Aynı sayıda aynı girdi bilgileri ile işlemler yapıldığında grafik işlemcide çalıştırılan CUDA yazılımının MATLAB'da çalıştırılan IONOLAB-RAY algoritmasından yaklaşık olarak 1000 kat daha hızlı olduğu görülmektedir.

6.2. Örnek 2 – Nümeriksel Doğrulama ve Grafiksel Çalışma

Hem MATLAB'a hem de grafik işlemcide çalıştırılan CUDA yazılımına girdi olarak kullanılan parametreler şu şekildedir:

- Frekans : 12 Mhz,
- Enlem : 42.5 derece,
- Boylam : 35 derece,
- Yanca : 0 derece,
- Yükseliş: 30 derece,
- Tarih : 25 Mart 2011,
- Saat : 12:00 GS.

Bu girdi parametreleri de Örnek 1'deki gibi aynı şekilde hem MATLAB'daki IONOLAB algoritmasına hem de grafik işlemcide çalıştırılan CUDA yazılımına girdi olarak verilmiştir. MATLAB ve grafik işlemcide çalıştırılan CUDA yazılımından çıkan sonuçlar grafiksel olarak değerlendirilmiştir. Sıradan ve sıradışı dalga yolları aynı grafiklerde çizdirilerek sonuçlar görülmüştür. Elde edilen bu grafikler Ek-4'te verilmiştir.

Ayrıca, MATLAB ve CUDA yazılımından elde edilen ışınların matematiksel bilgileri de karşılaştırılmıştır. İki sonuç vektörü arasında hatanın en fazla olduğu yerde, veri değerleri arasındaki farkın 1.3×10^{-7} olduğu görülmüştür. Bu değer MATLAB ve CUDA yazılımının çalışmakta olduğu çözünürlük farkından kaynaklanmakta olup, doğal karşılanmaktadır.

Nümerik kontrollerden ve grafiksel gösterimlerden sonra süre ölçüm işlemleri gerçekleştirilmiştir. Elde edilen işlem süresi bilgileri Çizelge 6.2'de verilmiştir.

Çizelge 6.2. Örnek 2 – Süre ölçümleri.

Süre Ölçüm Altyapısı	**1024 sıradan ışının hesaplaması için geçen süre**	**1024 sıradışı ışının hesaplaması için geçen süre**
MATLAB	385 saniye	315 saniye
C kodu	4.5 saniye	4 saniye
CUDA yazılımı	0.4 saniye	0.35 saniye

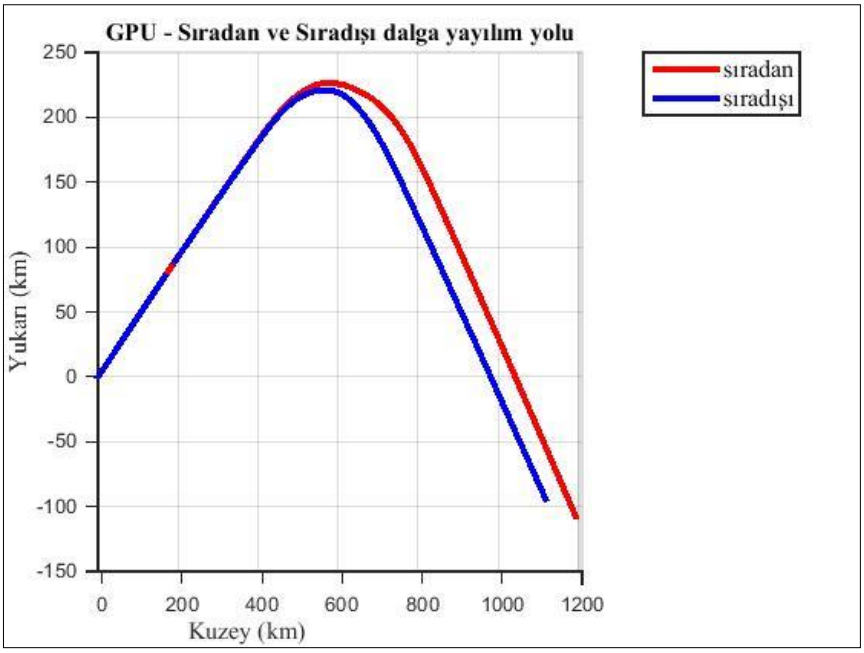
Elde edilen süreler değerlendirildiğinde grafik işlemcide çalıştırılan CUDA yazılımının MATLAB'da yapılan işlemleri yaklaşık 950 kat daha hızlı gerçekleştiği görülmektedir.

6.3. Örnek 3 – Sıradan ve Sıradaşı Dalga Hesaplaması ve Süre Ölçümü

Hem MATLAB hem de grafik işlemcide çalıştırılan CUDA yazılımında girdi olarak kullanılan parametreler şu şekildedir:

- Frekans : 10 Mhz,
- Enlem : 37.5 derece,
- Boylam : 30.5 derece,
- Yanca : 0 derece,
- Yükseliş: 30 derece,
- Tarih : 25 Mart 2011,
- Saat : 12:00 GS.

Yukarıda belirtilen verici bilgileri ve veri setiyle yapılan çalışma sonucunda hem MATLAB hem de grafik işlemcide çalıştırılan CUDA yazılımının sonuçları değerlendirilmiştir. Grafik işlemcide çalıştırılan CUDA yazılımının çalıştırılması sonucunda elde edilen dalga yayılım yollarının DKY koordinat sistemindeki görüntüsü Şekil 6.1'deki gibidir. Bu grafikte hem sıradan hem de sıradışı ışın için dalganın yayılım yolu görülmektedir.



Şekil 6.1. CUDA yazılımı çıktısı ile elde edilen sıradan ve sıradışı dalganın yayılım yolu.

Çalışılan bu veri seti için de süre ölçüm faaliyetleri yapılmış olup, elde edilen süre bilgileri Çizelge 6.3'deki gibidir.

Çizelge 6.3. Örnek 3 – Süre ölçümleri.

Süre Ölçüm Altyapısı	**1024 sıradan ışının hesaplaması için geçen süre**	**1024 sıradışı ışının hesaplaması için geçen süre**
MATLAB	550 saniye	450 saniye
C kodu	6 saniye	6.2 saniye
CUDA yazılımı	0.50 saniye	0.45 saniye

1024 ışının işlem süreleri karşılaştırıldığında, MATLAB ile sıradan ve sıradışı ışının dalga yollarının hesabı için toplamda yaklaşık 1100 saniyelik işlem süresi ölçülürken, grafik işlemcide çalıştırılan CUDA yazılımında sıradan ve sıradışı ışının dalga yollarının hesaplanmasında geçen toplam süre yaklaşık 0.95 saniye olarak görülmektedir. Bu durumda grafik işlemcide çalıştırılan CUDA yazılımının MATLAB'dan yaklaşık 1200 kat daha hızlı çalıştığı hesaplanmaktadır.

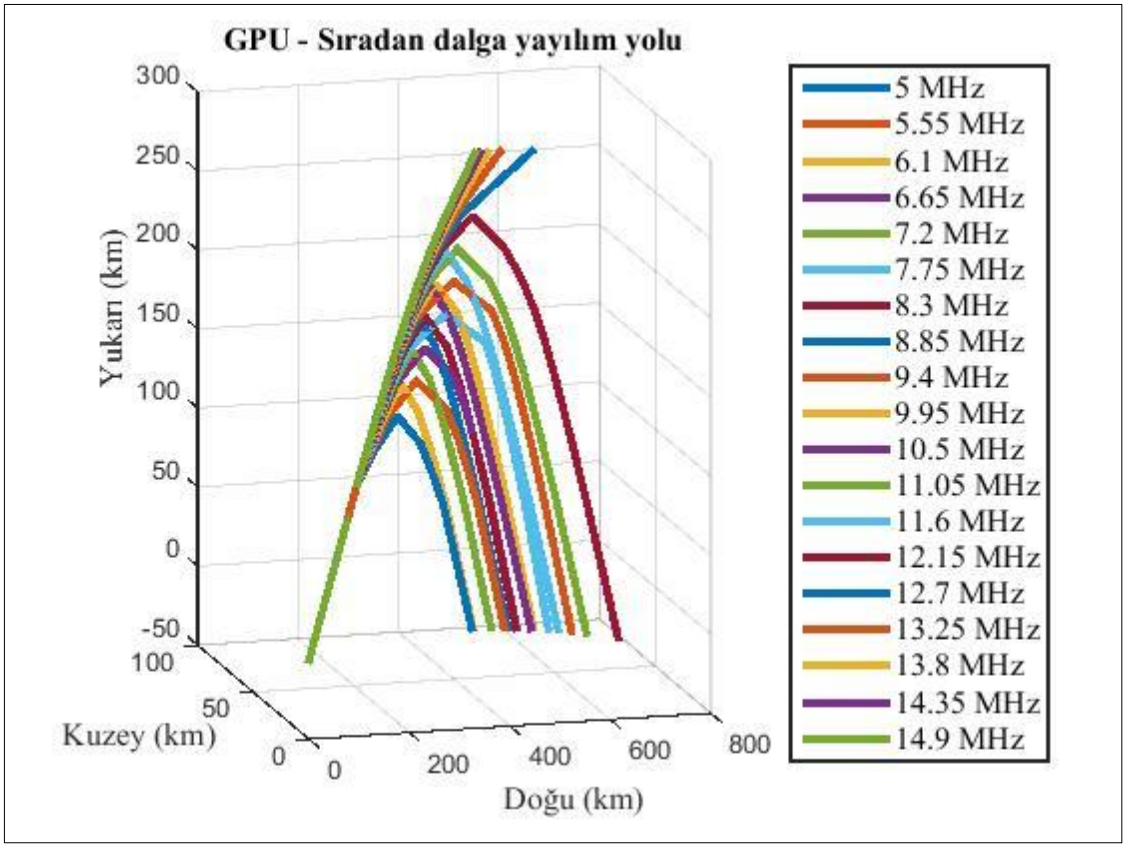
6.4. Örnek 4 – Frekans Değişiminin Etkisi

Bu uygulamada, grafik işlemcide çalıştırılan CUDA yazılımında 1024 ışının dalga yolu hesabı yapılırken her bir ışının frekans bilgisi değiştirilerek oluşan frekansa bağımlı ışın yolları incelenmiştir. İncelenen veri setinde, 1024 ışın için 0.01 Mhz frekans adımları ile hesaplama yapılmıştır. Kullanılmış olan veri setinin girdi parametreleri şu şekildedir:

- Frekans : 5 Mhz ile 15.24 Mhz arası,
- Enlem : 37.5 derece,
- Boylam : 30.5 derece,
- Yanca : 0 derece,
- Yükseliş: 30 derece,
- Tarih : 25 Mart 2011,
- Saat : 12:00 GS.

Grafik işlemcide çalıştırılan CUDA yazılımı ile elde edilen frekans değişimine göre sıradan ışınların dalga ilerleyiş şekli Şekil 6.2'de verilmiştir. Yapılan bu deney setinde 1024 adet

ışının aynı grafikte gosterimi çok karışık olacağı için yaklaşık 0.5 MHz'lik frekans adımlarıyla elde edilen sonuçlar gösterilmiştir.



Şekil 6.2. CUDA yazılımı ile elde edilen frekans bağımlı sıradan dalga yolları.

Bu veri setinden elde edilen süre ölçüm sonuçları ise Çizelge 6.4'te verilmiştir.

Çizelge 6.4. Örnek 4 – Süre ölçümleri.

Süre Ölçüm Altyapısı	**1024 sıradan ışının hesaplaması için geçen süre**	**1024 sıradışı ışının hesaplaması için geçen süre**
MATLAB	850 saniye	800 saniye
C kodu	6 saniye	5.8 saniye
CUDA yazılımı	0.60 saniye	0.50 saniye

Şekil 6.2'de de görüldüğü gibi, yüksek frekans değerlerindeki ışınların ulaşabildiği nokta düşük frekanslı ışınlardan daha uzakta olabilmektedir. Bu örnekten daha uzaklara ışın gönderebilmek için gönderilecek ışının yüksek frekans değerlikli olması gerekliliği çıkarımı yapılabilmektedir. Ayrıca, ışınların frekanslarının artmasıyla birlikte ışın izleme

işlemlerinde izlenen yolun uzamasından dolayı ışınların daha fazla kırılıma uğradığı ve işlemsel yükün arttığı süre ölçümlerinden anlaşılmaktadır.

6.5. Örnek 5 – İkinci Veri Seti ile Enlem Değişiminin Etkisi

Bu uygulamada diğer ilk 4 Örnek'ten farklı bir veri seti kullanılmıştır. Kullanılmış olan iyonkürenin elektron yoğunluk bilgisi, modellenmiş iyonkürenin koordinat aralıkları ve diğer parametreleri şu şekildedir:

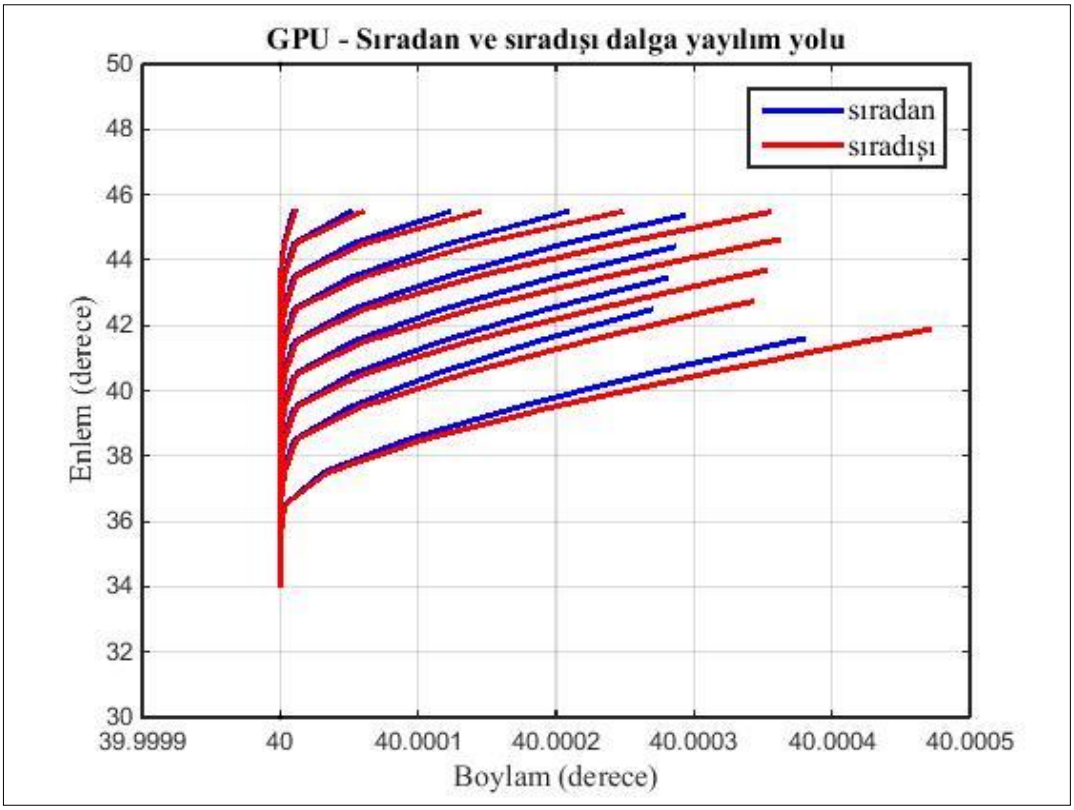
- Enlem aralığı : 30 derece ile 45 derece arası,
- Boylam aralığı : 25 derece ile 45 dececе arası,
- Tarih : 15 Ocak 2015,
- Saat : 4:30 GS.

Bu veri setine ek olarak Örnek 8 ve Örnek 9'da aynı tarihte fakat sırasıyla 11:00 GS ve 00:00 GS'de alınmış olan veri setleri ile de çalışma yapılmış olup, gün içerisinde iyonküredeki değişimin ışın izlemeye olan etkisi analiz edilmiştir.

Bu örnekte, enlem bilgisinin yaklaşık 0.01 derece değişimiyle hesaplanan ışınların ışın yolları çizdirilmiştir. Frekans, boylam ve yükseklik değerlerin sabit tututalarak 32.0 dereceden 0.01 adımlarla 42.24 dereceye kadar toplam 1024 farklı ışının hesaplaması yapılmıştır. Kullanılan verici girdi parametreleri şu şekildedir:

- Frekans : 11 Mhz,
- Enlem : 32.0 derece ile 42.24 derece arası,
- Boylam : 40 derece,
- Yanca : 0 derece,
- Yükseliş: 30 derece,
- Tarih : 15 Ocak 2015,
- Saat : 4:30 GS.

Bu deney ile farklı enlem başlangıç değerine sahip ışınların ulaştığı konum bilgileri Şekil 6.3'te görülmektedir.



Şekil 6.3. CUDA yazılımı ile elde edilen enlem bağımlı sıradan ve sıradışı dalga yollarının enlem ve boylam ekseninde görüntüsü.

Şekil 6.3'te sıradan ve sıradışı dalga yolları aynı görselde görülmektedir. Boylamda kuzeye doğru bir eğilim gözlenmekle birlikte boylam aralığı çok dar olduğu için bu hareket oldukça sınırlıdır. Son olarak süre ölçüm faaliyeti gerçekleştirilmiş olup, Çizelge 6.5'te görülmektedir.

Çizelge 6.5. Örnek 5 – Süre ölçümleri.

Süre Ölçüm Altyapısı	**1024 sıradan ışının hesaplaması için geçen süre**	**1024 sıradışı ışının hesaplaması için geçen süre**
MATLAB	2400 saniye	2300 saniye
C kodu	14 saniye	12 saniye
CUDA yazılımı	1.2 saniye	1.10 saniye

Bu deney setiyle birlikte işlem yapılacak iyonküre ilgi alanı ve işlem çözünürlüğü değişmiştir. Grafik işlemcide çalıştırılan CUDA yazılımı, tek duyarlılıklı yerine çift duyarlılıklı işlem yapmaya başlamıştır. İyonkürede farklı alan incelemesi ve çözünürlük değişimi nedeniyle CUDA yazılımında bir miktar artma görülmektedir. Buna rağmen, grafik

işlemcide çalıştırılan CUDA yazılımının MATLAB'dan yaklaşık 2000 kat daha hızlı bir şekilde sonuçlara ulaşabildiği görülmektedir.

6.6. Örnek 6 – Boylam Değişiminin Etkisi

Bir diğer deneysel çalışmada ise, boylam derecesi her bir ışın için değişen ışınların yolları hesaplanarak çizdirilmiştir. Deneyde kullanılmış olan parametreler şu şekildedir:

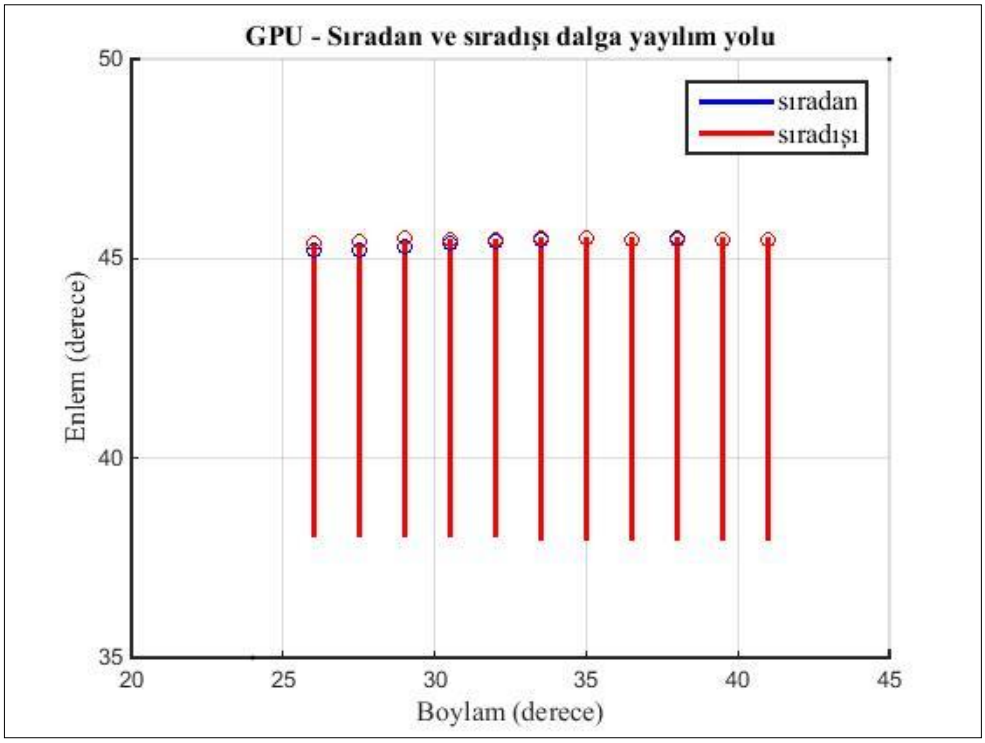
- Frekans : 9 Mhz,
- Enlem : 38 derece,
- Boylam : 26.0 derece ile 41.36 derece arası,
- Yanca : 0 derece,
- Yükseliş : 30 derece,
- Tarih : 15 Ocak 2015,
- Saat : 4:30 GS.

Boylamda 0.015 derece adımlarla farklı 1024 boylam başlangıç noktası için ışın izleme işlemleri yapılmış olup, sonucunda oluşan yollar 1'er boylam derece aralıklarıyla çizdirilmiştir. Hangi boylam derecesi ile ışın gönderimi yapıldığında ışınların nerelere ulaştığı Şekil 6.4'te görülmektedir. Ayrıca bu çalışma ile alınan süre ölçümleri Çizelge 6.6'da verilmiştir.

Çizelge 6.6. Örnek 6 – Süre ölçümleri.

Süre Ölçüm Altyapısı	**1024 sıradan ışının hesaplaması için geçen süre**	**1024 sıradışı ışının hesaplaması için geçen süre**
MATLAB	2300 saniye	1900 saniye
C kodu	12 saniye	11 saniye
CUDA yazılımı	1.1 saniye	1.00 saniye

Süre ölçümleri değerlendirildiğinde Örnek 5'teki ölçümler ile benzerlikler taşıdığı görülmektedir. Örnek 5'te enlem başlangıç bilgisi değişirken, Örnek 6'da boylam başlangıç bilgisi değişmektedir.



Şekil 6.4. Grafik işlemcide çalıştırılan CUDA yazılımı ile hesaplanan boylam bağımlı sıradan ve sıradışı dalga yollarının enlem ve boylam ekseninde görüntüsü.

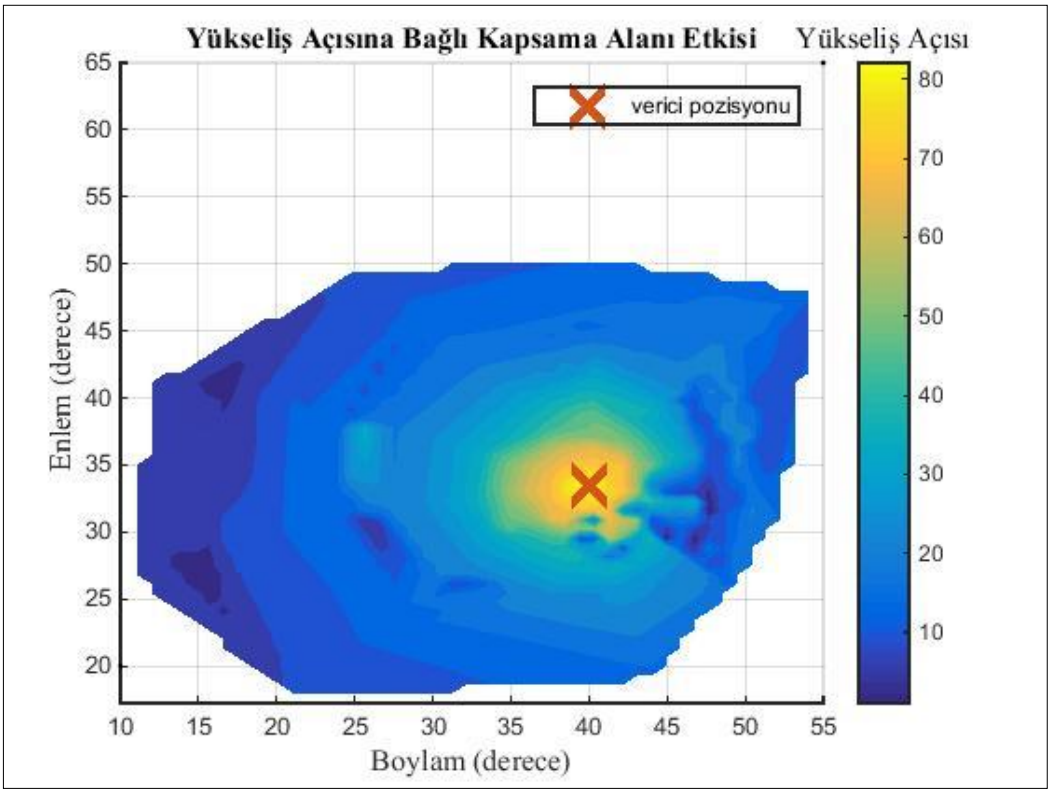
Şekil 6.4'te hem sıradan hem de sıradışı dalga yolları farklı boylam başlangıç noktaları için aynı grafikte çizdirilmiş olup, bu deney seti için sıradışı dalga yollarının sıradan dalga yollarından belirgin şekilde farklı olmadığı görülmüştür.

6.7. Örnek 7 – Yükseliş ve Yanca Değişiminin Etkisi

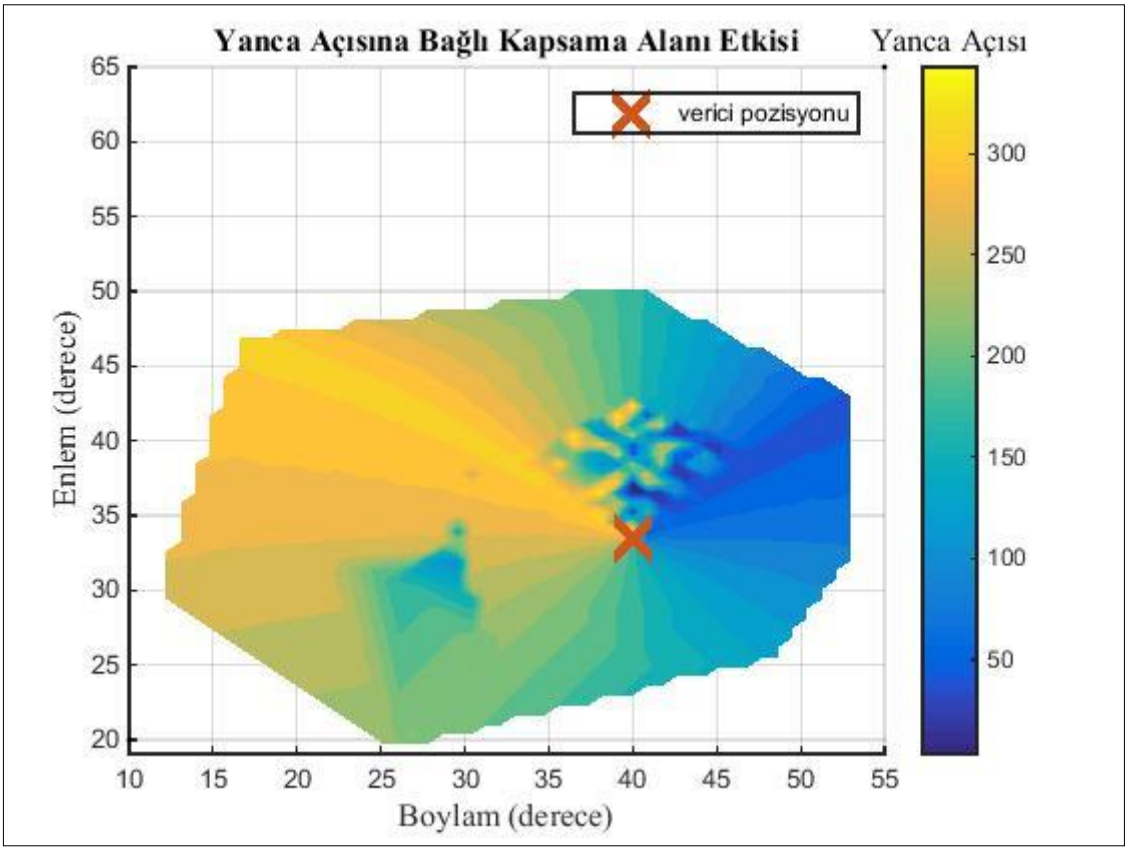
Tez kapsamında yapılmış olan bu deneysel çalışmada frekans, enlem ve boylam başlangıç bilgileri her bir ışın için sabit verilmiştir. 1'er derecelik yükseliş açı değişiminde 0 derece ile 360 derece arasında yanca değerleri değiştirilerek ölçümler alınmıştır. Yanca açı değeri farklı verilerek 1024 farklı ışının hesaplamaları yapılmıştır. Bu son deneyde kullanılmış olan parametreler şu şekildedir:

- Frekans : 5 Mhz,
- Enlem : 33.4 derece,
- Boylam : 40 derece,
- Yükseliş : 5 derece ile 90 derece arası,
- Yanca : 0 derece ile 360 derece arası,
- Tarih : 15 Ocak 2015,
- Saat : 4:30 GS.

Bu deney setinde her bir ışın hesaplanırken yanca değişimi 30 derece olarak belirlenmiştir. 0 derece ile 360 derece arasındaki yanca değişiminde, yükseliş bilgisi sabit tutulmuştur. Toplam 90 adet farklı yükseliş bilgisinden elde edilen sonuçlar topluca aynı grafik üzerinde değerlendirilmiştir. Bu deneydeki amaç, sabit verici noktasından farklı yanca ve yükseliş değerlerinde ışınların nereye gidebileceğini ve iyonkürenin farklı yönlerdeki etkisini gözlemleyebilmektir. Sonuçta oluşan 1024 adet farklı parametre ile hesaplanmış ışının, enlem ve boylam düşüş noktalarının yükseliş açısına göre kapsama alanı Şekil 6.5'teki gibi görülmektedir. Yanca açısına göre ışınların düşüş noktalarının kapsama alanı ise Şekil 6.6'daki gibi görülmektedir.Bu deneyde 5 ile 90 derece arasında yükseliş açı değişimi 1'er derece aralıklarla yapılmıştır. Grafikte de her yükseliş açı değişiminde ışının 0-360 derece yanca açı değişiminde nereye düştüğü farklı renklerle çizdirilmiştir. Her bir düşüş noktası ise birbirine bağlantılı şekilde çizdirilmiştir. Kısıtlı sayıda farklı renk kullanılabildiği için yaklaşık her on yükseliş derecesinde bir aynı renkler kullanılmıştır. Burada dikkat edilmesi gereken husus, yükseliş açı değeri düşük iken ışının daha uzağa; yükseliş açı değeri 90 dereceye yakınsadıkça ışınların düşüş noktasının vericiye yakın olmaya başlamasıdır.



Şekil 6.5. CUDA yazılımı ile elde edilen yükseliş ve yanca bağımlı sıradan dalga yollarının yükseliş açısı değişimine göre kapsama alanı görüntüsü, 04:30 GS.



Şekil 6.6. CUDA yazılımı ile elde edilen yükseliş ve yanca bağımlı sıradan dalga yollarının yanca açısı değişimine göre kapsama alanı görüntüsü, 04:30 GS.

Bu deneyde yükseliş açısının küçük olduğu anlarda gönderilen ışının çok daha uzak bölgelere gidebildiğini gözlemlenirken, yükseliş açısının artmaya başlamasıyla beraber gönderilen ışınların düştüğü konum bilgilerinin vericiye doğru yakınsadığı gözlemlenmektedir.

Yine bu deney seti ile frekans bağımlı olarak ışınların düşüş noktaları da incelenmiştir. 2, 5, 10, 15, 20 ve 30 MHz frekanslı ışınlar ile yükseliş ve yancadaki değişimin ışınların düşüş noktalarına etkisi değerlendirilmiştir. Bu değerlendirme yapılırken yükseliş açısında 10'ar derecelik aralıklar yancada ise 0-360 derece arası taranmıştır. Farklı frekans değerleriyle elde edilen bu analiz ile ilgili ayrıntılı grafiksel bilgiler Ek-5'te bulunmaktadır. Bu grafiklerde düşük frekans değerlerinde ışınların nispeten daha yakın noktalara gittiği görülmektedir. Frekans değerleri arttıkça ışınların daha uzak noktalara düşüş yaptığı, fakat frekans arttırımının bir süre sonra ışınların dalga yollarına fazla bir etki göstermediği analiz edilmiştir.

6.8. Örnek 8 – Üçüncü Veri Seti ile Yükseliş ve Yanca Değişiminin Etkisi

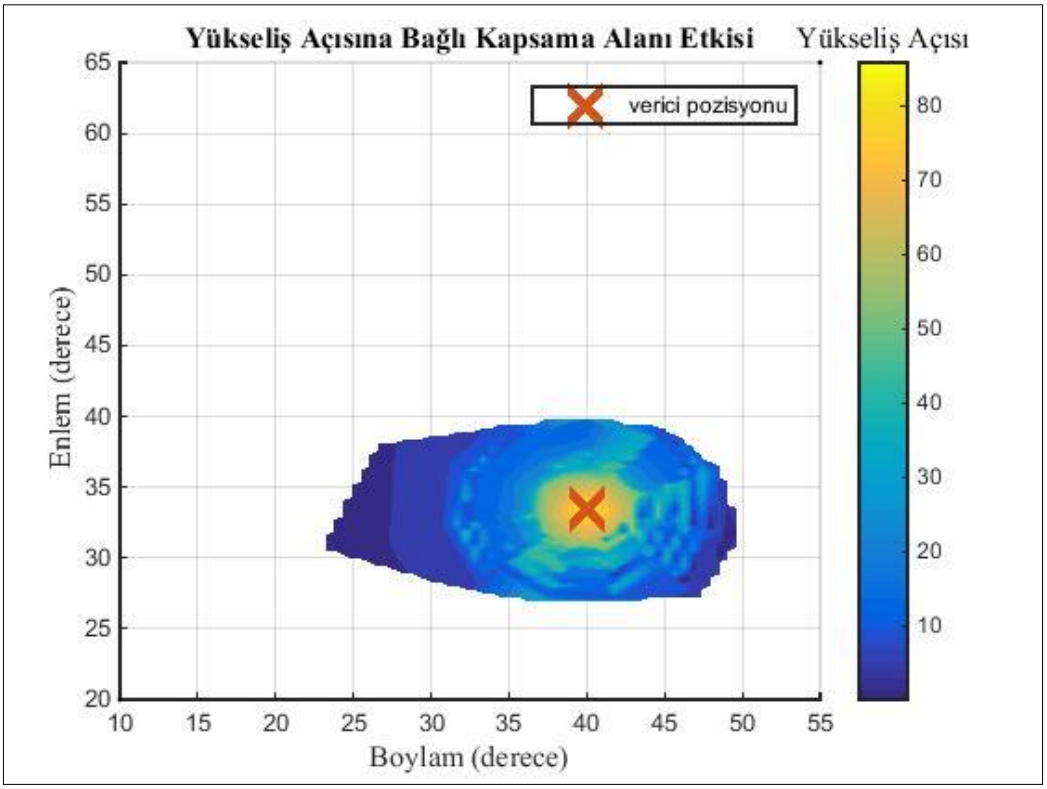
İyonkürenin elektron yoğunluk bilgisinin gün içinde değişimini analiz edebilmek adına bu deney setinde Örnek 7'de kullanılan deney setinin aynı gün, fakat farklı saatte alınmış iyonküre elektron yoğunluk bilgileri kullanılmıştır. Kullanılmış olan veri seti şu şekildedir:

- Enlem aralığı : 30 derece ile 45 derece arası,
- Boylam aralığı : 25 derece ile 45 dececе arası,
- Tarih : 15 Ocak 2015,
- Saat : 11:00 GS.

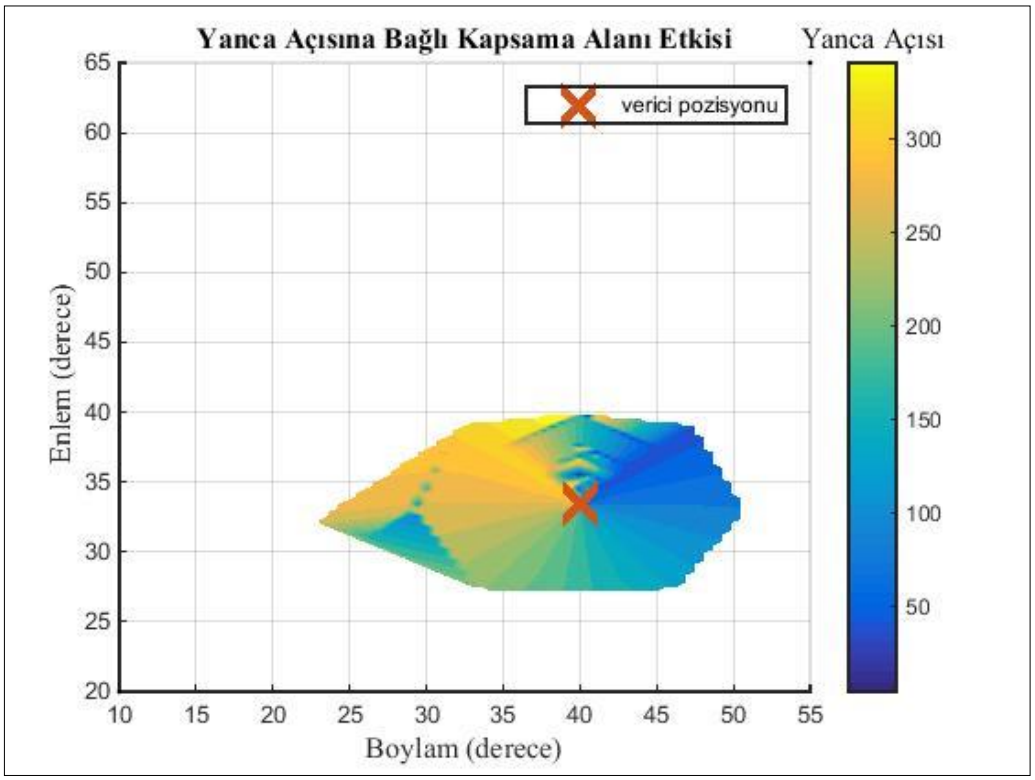
Vericiye ait başlangıç bilgileri ise şu şekildedir:

- Frekans : 5 Mhz,
- Enlem : 33.4 derece,
- Boylam : 40 derece,
- Yükseliş : 5 derece ile 90 derece arası,
- Yanca : 0 derece ile 360 derece arası,
- Tarih : 15 Ocak 2015,
- Saat : 11:00 GS.

Bu deney setinde de Örnek 7'deki gibi her bir ışın hesaplanırken yanca değişimi 30 derece olarak belirlenmiştir. 0 derece ile 360 derece arasındaki yanca değişiminde, yükseliş bilgisi sabit tutulmuştur. Toplam 90 adet farklı yükseliş bilgisinden elde edilen sonuçlar topluca aynı grafik üzerinde değerlendirilmiştir. Bu deneydeki amaç ise aynı gün, farklı saatlerde iyonküredeki değişimin ışın izlemeye olan etkisinin incelenmesidir. 11:00 GS için, sıradan dalga ışın yollarının grafik işlemcide çalıştırılan CUDA yazılımı sonuçları Şekil 6.7 ve Şekil 6.8'deki gibi görülmektedir.



Şekil 6.7. CUDA yazılımı ile elde edilen yükseliş ve yanca bağımlı sıradan dalga yollarının yükseliş açısı değişimine göre kapsama alanı görüntüsü, 11:00 GS.



Şekil 6.8. CUDA yazılımı ile elde edilen yükseliş ve yanca bağımlı sıradan dalga yollarının yanca açısı değişimine göre kapsama alanı görüntüsü, 11:00 GS.

Bu grafikler incelendiğinde ışınların düşüş noktalarının, güneş yaklaşık olarak tepede olduğunda 04:30 GS'de alınan ölçümlere göre vericiye daha yakın noktalarda sönümlendiği yorumu yapılabilmektedir.

6.9. Örnek 9 – Dördüncü Veri Seti ile Yükseliş ve Yanca Değişiminin Etkisi

Yapılan son deney setinde deney setinde Örnek 7'de kullanılan deney setinin aynı gün, farklı saatte alınmış iyonküre elektron yoğunluk bilgileri kullanılmıştır. Kullanılmış olan veri seti şu şekildedir:

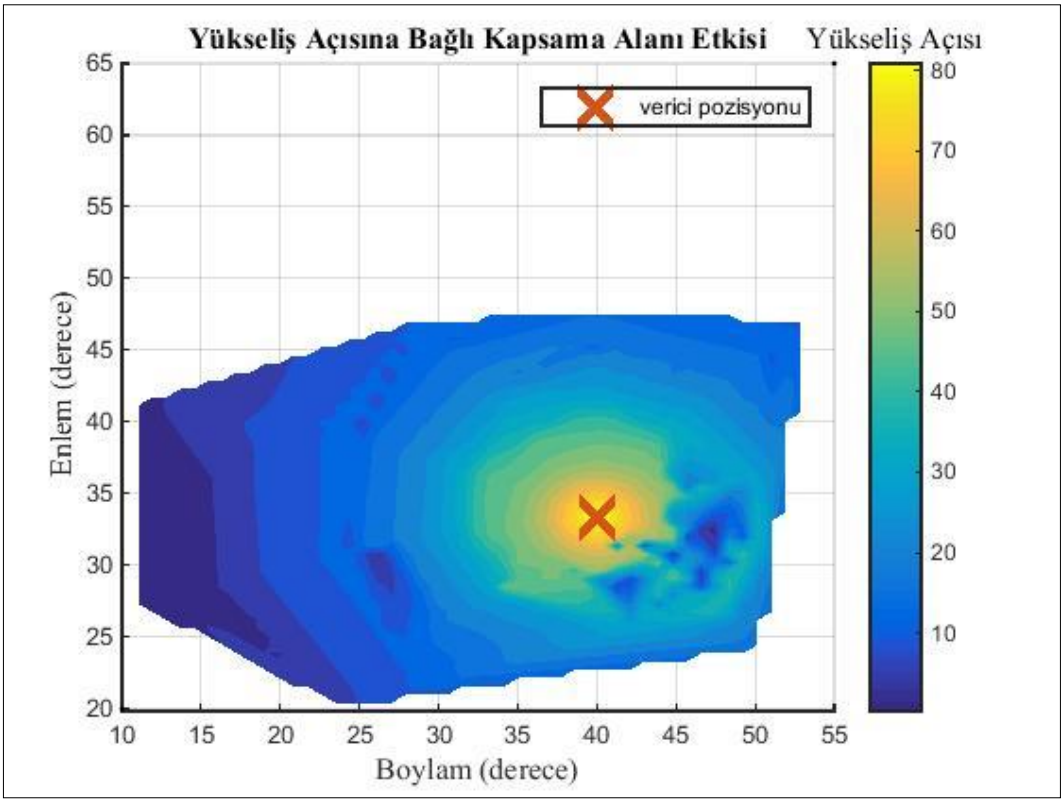
- Enlem aralığı : 30 derece ile 45 derece arası,
- Boylam aralığı : 25 derece ile 45 dececе arası,
- Tarih : 15 Ocak 2015,
- Saat : 00:00 GS.

Vericiye ait bilgiler ise yine Örnek 7 ve Örnek 8'deki gibi olup yalnızca saat bilgisi değişmiştir. Aşağıdaki gibidir:

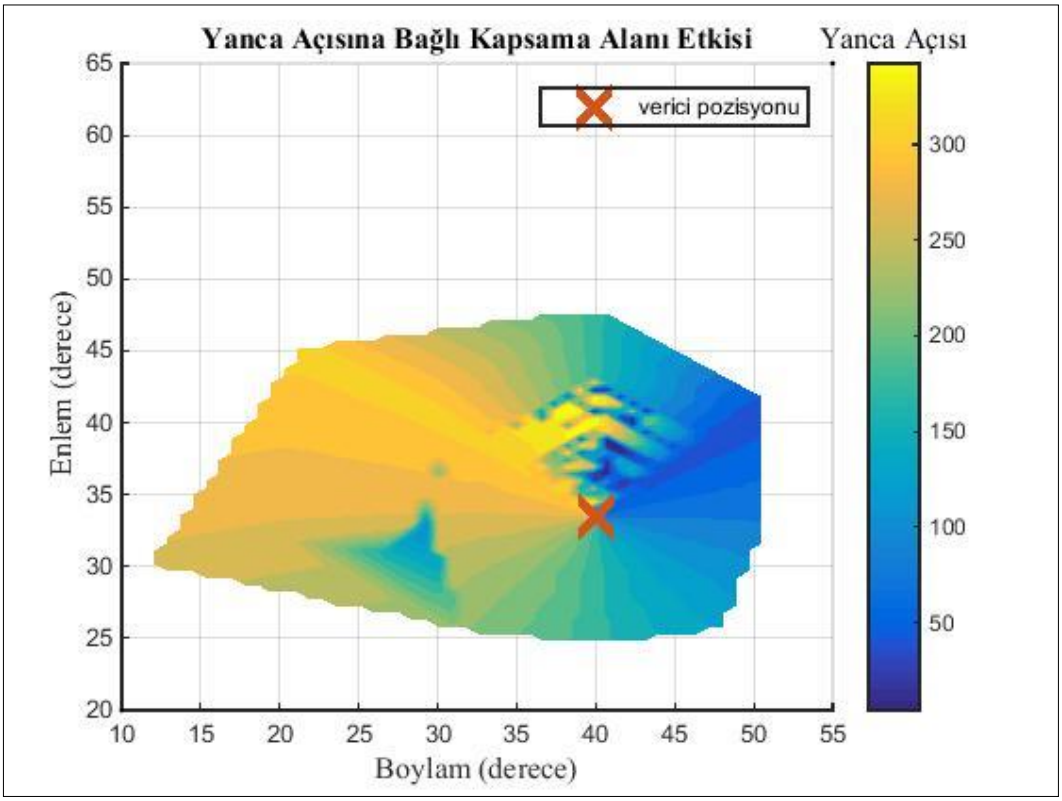
- Frekans : 5 Mhz,
- Enlem : 33.4 derece,
- Boylam : 40 derece,
- Yükseliş : 5 derece ile 90 derece arası,
- Yanca : 0 derece ile 360 derece arası,
- Tarih : 15 Ocak 2015,
- Saat : 00:00 GS.

Yükseliş ve yancadaki değişim aralıkları Örnek 7 ve Örnek 8'de belirtilenlerin aynısıdır. Sadece farklı saatlerin iyonküredeki eletron yoğunluğuna etkisi ve ışın izlemeye etkisini inceleyebilmek adına bu şekilde yapılmıştır.

Yapılan bu son deney setinde grafik işlemcide çalıştırılan CUDA yazılımının sonuçları çizdirilmiştir. Sıradan dalga yollarının yanca ve yükseliş değişimine göre düşüş noktalarının grafiksel gösterimleri Şekil 6.9 ve Şekil 6.10'daki gibidir.



Şekil 6.9. CUDA yazılımı ile elde edilen yükseliş ve yanca bağımlı sıradan dalga yollarının yükseliş açısı değişimine göre kapsama alanı görüntüsü, 00:00 GS.



Şekil 6.10. CUDA yazılımı ile elde edilen yükseliş ve yanca bağımlı sıradan dalga yollarının yanca açısı değişimine göre kapsama alanı görüntüsü, 00:00 GS.

Bu görüntüler incelendiğinde iyonkürede elektron yoğunluk bilgileri 00:00 GS ve 04:30 GS'de alınan veri setlerinde ışınların düşüş noktalarının birbirine benzer olduğu görülmektedir. Son 3 örnek alınan süre ölçüm bilgilerinin yaklaşık değerleri Çizelge 6.7'deki gibidir.

Çizelge 6.7. Örnek 7, 8, 9 – Süre ölçümleri.

Süre Ölçüm Altyapısı	**1024 sıradan ışının hesaplaması için geçen süre**	**1024 sıradışı ışının hesaplaması için geçen süre**
MATLAB	2700 saniye	2600 saniye
C kodu	15 saniye	14.7 saniye
CUDA yazılımı	1.3 saniye	1.25 saniye

Yukarıdaki süre ölçümlerinde görüldüğü üzere grafik işlemcide çalıştırılan CUDA yazılımı MATLAB'dan yaklaşık 2000 kat daha hızlı çalışmaktadır. Son 3 örnek çalışma değerlendirildiğinde iyonkürenin saat bazlı elektron yoğunluk bilgisinin ışın izlemeye etkisi, CUDA yazılımı ile toplamda yalnızca yaklaşık 4 saniyede incelenebilmiştir.

7. SONUÇLAR

Günümüzde giderek artmakta olan işlemci gücü ihtiyaçları alternatif çözüm yollarının ortaya çıkmasını sağlamıştır. Yakın zamanda en göze çarpan teknolojik gelişmelerden biri de grafik işlemci birimleridir. Son yıllarda analiz, benzetim, oyun, görüntü işleme vs. birçok konuda grafik işlemciler yoğunlukla kullanılmaya başlanmıştır. Yapılacak olan işlem, algoritma vs. içeriğine bağlı olarak merkezi işlemci birimlerinden çok daha iyi sonuçlar verebilen grafik işlemci birimleri, paralel programlama alanında gelişim göstermektedir. Yapılan bu tez kapsamında kısa dalga haberleşme sistemlerinde ışın izleme yöntemlerinin paralel programlanması üzerine eniyileme çalışmaları yapılmıştır. İyonkürede ilerlemekte olan kısa dalga bandındaki ışınların birbirlerinden bağımsız hesaplanabilirlikleri, paralel programlamayı kullanma fikrini doğurmuştur. Bu fikirden esinlenerek grafik işlemciler ile ışın izleme işlemlerinin ortak paydada buluşabileceği fikri oluşmuştur. Literatür araştırmasıyla başlayan tez çalışmalarında, kısa dalga haberleşme sistemleri ve iyonkürede ışın izleme yöntemleri araştırılmıştır. Gerek işlem yükünün emsallerine göre daha basit olması, gerekse de algoritma hesaplama sürelerinin daha kısa olması nedeniyle ışın izleme yöntemlerinden geometrik-optik yaklaşıma sahip Snell Yasası ile ışın izleme yöntemi üzerinde daha çok durulmuştur. Bu yöntemin literatür taraması yapılıp, literatürde ışın izleme adına kullanılmakta olan alternatif çözüm yolları incelenmiştir.

Işın izleme işlemlerinde IONOLAB-RAY [4] çalışması üzerine yoğunlaşılmış olup, test aşamalarında kullanılmak üzere veri setleri ve MATLAB algoritma kodları Dr. Esra ERDEM'den temin edilmiştir. Algoritmanın analizi yapılıp, grafik işlemciler için uygun olup olmadığı değerlendirilmiştir. Yapılan inceleme ve analiz işlemlerinin ardından, grafik işlemcilerde geometrik-optik ışın izleme yönteminin olumlu sonuçlar verebileceği sonucuna varılmıştır.

Grafik işlemci birimlerinde donanımsal çekirdek sayısı binlerce olabilmektedir. Birbirinden bağımsız ve paralel olarak çalışma prensibiyle binlerce grafik işlemci çekirdeğinde farklı veriler işlenebilmektedir. İyonkürede gönderilmek istenen birbirinden bağımsız binlerce ışının analizi değerlendirildiğinde, bu işlemlerin grafik işlemcilerde çalıştırılan CUDA yazılımında yapılması algoritmanın donanım mimarisine uyumluluğu açısından önemli bir gösterge olmaktadır. Bu sayede hem işlem yükünün paralelleştirilerek azaltılması hem de işlem süresinde iyileşme hedeflenmiştir. Tez kapsamında algoritma analizinin ardından grafik işlemcide çalıştırılacak CUDA derleyicisi ile yazılım faaliyetlerine geçilmiştir.

NVIDIA grafik işlemci kartlarında Tesla K20 ekran kartı ile çalışmalar yapılmıştır. Bu kart üzerinde NVIDIA'nın sunmuş olduğu ücretsiz kod derleme aracı olan CUDA derleyicisi kullanılmıştır. Bu derleyici üzerinde yazılım geliştirme faaliyetleri gerçekleştirilmiştir. Yazılım faaliyetleri birçok alt aşamaya ayrılmış olup beşinci bölümde bu faaliyetlerden ayrıntılı olarak söz edilmiştir.

Grafik işlemcide çalıştırılan CUDA kodu geliştirme safhasında belirlenen her alt fonksiyon için doğruluk ve süre ölçüm faaliyetleri gerçekleştirilmiştir. İyonkürede dalga yayılımında sıradan ve sıradışı olmak üzere iki farklı dalga yayılımı oluşmakta olup, bu iki dalga için de ayrı ayrı fonksiyonlar yazılmıştır. Kullanıcı, analiz yapmak istediği bölgenin elektron yoğunluğunu ve girdi bilgilerini yazılıma besleyip, yazılımın çalıştırılması sonucunda çıkan koordinat bilgilerini görebilmektedir. Yazılımın çıktısı olarak sunulan koordinat bilgileri, ışının iyonkürede modellenen bölgelerden geçerken elektron yoğunluğu farklılığından dolayı yapmış olduğu kırılımlardan dolayı değişiklik göstermektedir. Elde edilen yazılım çıktı bilgileri hem matematiksel hem de grafiksel olarak incelenmiştir. Örnek çalışmalarda, MATLAB algoritmalarına ve grafik işlemcide çalıştırılan CUDA yazılımına aynı girdi bilgileri beslenerek sonuçlarında oluşan çıktı bilgilerinin birbirleriyle uyumlu olduğu görülmüştür.

Yazılımın tamamlanmasının ardından yazılımda iyileştirme çalışmaları gerçekleştirilmiş olup, işlemlerin CUDA yazılımı ile daha da hızlanması sağlanmıştır. İyileştirme kapsamında NVIDIA'nın sunmuş olduğu iyileştirme kütüphaneleri ve analiz araçları da kullanılmıştır.

Grafik işlemcilerde çalıştırılan CUDA derleyicisi ile paralel programlama yapılarak yazılmış olan IONOLAB-RAY algoritmasının MATLAB kodu içeriğinde tek bir ışının dalga modeli hesaplanabilmektedir. Grafik işlemcilerde çalıştırılan CUDA yazılımında ise 1024 adet ışının paralel olarak işlenebilmesi adına bir altyapı hazırlanmıştır. MATLAB algoritmasının, C kodunun ve CUDA yazılımının işlem süreleri karşılaştırılmıştır. Yapılan örnek çalışmalarda süre ölçüm faaliyetleri ile elde edilen sıradan dalga için ölçümler Çizelge 7.1'de belirtilmiştir.

Çizelge 7.1. Genel süre ölçümleri.

Deney Seti Numarası	MATLAB ile işlem süresi	C kodu ile işlem süresi	CUDA yazılımı ile işlem süresi
1	200 saniye	3 saniye	0.2 saniye
2	385 saniye	4.5 saniye	0.4 saniye
3	550 saniye	6 saniye	0.5 saniye
4	850 saniye	6 saniye	0.6 saniye
5	2400 saniye	14 saniye	1.2 saniye
6	2300 saniye	12 saniye	1.1 saniye
7	2800 saniye	15 saniye	1.33 saniye
8	2500 saniye	14.5 saniye	1.25 saniye
9	2700 saniye	14.8 saniye	1.3 saniye

Çözünürlüğün artırıldığı 5.deney ve sonrası baz alındığında MATLAB'da 1024 ışının ışın izlemesi yaklaşık 2700 saniye sürmektedir. Buna karşılık, grafik işlemcide çalıştırılan CUDA yazılımı ile 1024 ışının işlenme süresi yaklaşık 1.3 saniye olarak ölçülmüştür. Bu iki ölçüm değerlendirildiğinde, aynı girdi verisi verildiğinde CUDA yazılımı ile paralel programlama yapılarak MATLAB algoritmasından yaklaşık iki bin kat daha hızlı işlem yapılabildiği görülmüştür. CUDA yazılımı ile C kodu karşılaştırıldığında ise CUDA yazılımının C kodundan yaklaşık on kat daha hızlı sonuç verdiği görülmektedir. Ölçülen işlem sürelerinin farklılıkları, 1024 ışının CUDA yazılımında paralel bir şekilde işlenebilmesinden kaynaklanmaktadır. MATLAB'da ve C kodunda ise 1024 ışın sırayla işleniyor olduğundan, herhangi bir paralellik söz konusu değildir. MATLAB algoritmalarında paralellik sağlansa bile bu genelde dört çekirdekli merkezi işlemciler kullanıldığı için yaklaşık dört kat süre kazancı anlamına gelmektedir. Bu durumda bile grafik işlemci yazılımı, dört çekirdekte yapılan MATLAB algoritmalarının işlem süresinden yaklaşık beş yüz kat daha verimli olmaktadır. Tez çalışmalarında elde edilen işlem süreleri analiz edildiğinde, grafik işlemcide çalıştırılan CUDA yazılımının çok daha verimli olduğu görülmektedir.

Çalışmalar kapsamında elde edilen sıradan ve sıradışı dalgaların dalga yolları, grafiksel olarak da incelenmiştir. Bu incelemeler sonrasında, kaynaktan giden ışının nereye ulaştığı analiz edilebilmektedir. Çeşitli frekans, enlem, boylam, yükseklik ve açı değerleri gibi farklı parametreler verilerek elektron yoğunluğu bilinen bir bölgede gönderilen ışının nerelere

ulaşabileceği analiz edilebilmektedir. Bu analizi hem doğru hem de çok kısa sürede gerçekleştirebilecek olmak, iyonkürede gerçekleştirilecek kısa dalga haberleşmesinde çok önemlidir. Örnek 7, Örnek 8 ve Örnek 9 çalışmaları ile iyonkürede aynı gün içinde oluşan değişime göre ışın yayılım yollarının 4 saniye içerisinde analiz edilebileceği gösterilmiştir. Tez çalışmaları kapsamında paralel programlama ile ışın izlemenin hem doğru yapılabilmesi hem de çok kısa sürede yapılması sağlanarak hedeflenen başarımlara ulaşılmıştır.

Yapılmış olan bu tez çalışması ile birlikte iyonkürede kısa dalga haberleşme benzetim ve analizlerinin çok daha kısa sürede yapılmasının önü açılmıştır. Mevcut uygulamalar ile oldukça uzun işlem süreleri ile yapılan benzetim ve analizlerin, paralel programlama ile çok daha kısa sürede gerçeklenebilmesi sağlanacaktır. Tez çalışmasının başarımını daha da artırabilmek için yapılan bu işlemlerin bir araç haline getirilmesi ve kullanıcı kontrolünün daha da kolaylaştırılması ile daha verimli hale getirmesi sağlanabilir. Algoritmalara ve kişisel isteklere göre şekillendirebilecek olan bu çalışma, iyonkürenin etkilerinin değerlendirilmesi, yön bulma ve ufuk ötesi radarların başarımlarının artırılmasına yönelik vs. birçok alanda kullanılabilir niteliktedir.

KAYNAKLAR

[1] K. Davies, *Ionospheric Radio*. The Institution of Engineering and Technology, Michael Faraday House, Six Hills Way, Stevenage SG1 2AY, UK: IET, **1990**.

[2] S. S. Kirby, L. V. Barkner, and D. M. Stuart, *Studies of the Ionosphere and Their Application to Radio Transmission*, Bur. Stand. J. Res., vol. 12, no. January, pp. 15–51, **1934**.

[3] IONOLAB, http://www.ionolab.org/ (Şubat, **2018**).

[4] E. Erdem, *İyonkürede Elektromanyetik Dalga Yayılım Modeli ve Benzetimi,* Doktora Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, **2017.**

[5] Cervera, M.A., Harris, T.J., Modeling ionospheric disturbance features in quasi-vertically incident ionograms using 3-D magnetoionic ray tracing and atmospheric gravity waves, *Journal Of Geophysical Research: Space Physics*, 119, 431–440, Jan, **2014**.

[6] C. S. Gillmor and J. R. Spreiter, *Discovery of the Magnetosphere*, Hist. Geophys., vol. 7, **1997**.

[7] M. A. Çolak, N. Erdoğan, *Grafik Kartı Üzerinde Paralel Hızlandırılmış Işın İzleme*, 2. Ulusal Yüksek Başarımlı ve Grid HEsaplama Konferansı, s. 67-73, **2010**.

[8] J. Sanders and E. Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*, Pearson Education Inc., Boston, **2010.**

[9] J. Nickolls and W. J. Dally, *The GPU Computing Era*, IEEE Micro, vol. 30, no. 2, pp. 56–69, Mar. **2010**.

[10] D. B. Kirk and W. W. Hwu, *In Praise of Programming Massively Parallel Processors: A Hands-on Approach*, Morgan Kaufmann Publishers, London, **2010**.

[11] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, and K. Skadron, *A Performance Study of General-Purpose Applications on Graphics Processors Using CUDA*, University of Virginia, Department of Computer Science, USA, **2008**

[12] MATLAB programı, https://www.mathworks.com/products/matlab.html (Ocak, **2018**).

[13] CUDA derleyici programı, https://developer.nvidia.com/cuda-downloads (Ocak, **2018**).

[14] Havaküre katmanları, http://www.gorgoda.com/wp-content/uploads/2009/01/dunya.jpg (Şubat, **2018**).

[15] Havaküre katmanları, https://www.mgm.gov.tr/genel/sss.aspx?s=atmosfer (Ocak, **2018**).

[16] S. Bora, *Ionosphere and radio communication*, Resonance, vol. 22, no. 2, pp. 123–133, **2017**.

[17] İyonkürede iyonlaşma, https://www.electronics-notes.com/articles/antennas-propagation/ionospheric/ionosphere.php (Ocak, **2018**).

[18] M. S. M. and B. V. P. I. Abba, W. A. W. Z. Abidin, T. Masri, K. H. Ping, *Ionospheric Effects on Gps Signal in Low- Low - Latitude Region : a*, Niger. J. Technol., vol. 34, no. 3, pp. 523–529, **2015**.

[19] İyonkürede elektron yoğunluğu, https://books.google.com.tr/books?id=ODvvCAAAQBAJ&pg=PA5&dq=ionosphere+layers&hl=tr&sa=X&ved=0ahUKEwi4vq_6uKLYAhVGMZoKHQuhBWEQ6AEIODAC#v=onepage&q=ionosphere layers&f=false (Ocak, **2018**).

[20] Uydu haberleşmesi, http://www.itu.int/itunews/manager/display.asp?lang=en&year=2008&issue=03&ipage=Arthur-Clarke (Ocak, **2018**).

[21] Türkiye'nin uydu haritası, http://1.bp.blogspot.com/-AAyKSQ_o3pg/TxEVZG3B5jI/AAAAAAAAA4s/sPw2MK9OoaY/s1600/uydular.png (Ocak, **2018**).

[22] E. Network and P. Testbed, *Introduction to Satellite Communications Technology for NPen*, NAS Technical Report NAS-04-009, August, **2004**.

[23] Gps, https://www.dunyaatlasi.com/wp-content/uploads/2017/12/konum-belirleme-gps-nedir-nasil-calisir.jpg (Ocak, **2018**).

[24] S. Dubey, R. Wahi, and A. K. Gwal, Ionospheric effects on GPS positioning, *Advances in Space Research*, vol. 38, no. 11. pp. 2478–2484, **2006**.

[25] Kısa dalga haberleşme anteni, https://www.electronics-notes.com/articles/antennas-propagation/log-periodic-lpda-antenna/log-periodic-basics.php (Ocak, **2018**)

[26] N. M. Maslin, *HF Communications: A Systems Approach*, Springer US, **1987**.

[27] F. Arikan, S. Shukurov, H. Tuna, O. Arikan, and T. L. Gulyaeva, *Performance of GPS slant total electron content and IRI-Plas-STEC for days with ionospheric disturbance*, Geod. Geodyn., vol. 7, no. 1, pp. 1–10, **2016**.

[28] B. W. Reinisch and D. Bilitza, *Karl Rawer's life and the history of IRI*, Adv. Sp. Res., vol. 34, no. 9, pp. 1845–1850, **2004**.

[29] D. Bilitza, *International Reference Ionosphere 2000 of ionospheric It was and the F peak down to*, *Radio Sci.*, vol. 36, no. 2, pp. 261–275, **2001**.

[30] M. Kabasakal and C. Toker, *Ionogram generation in HF band by using three dimensional ray tracing*, 25th Signal Processing and Communications Applications Conference (SIU), pp. 1–4, **2017**.

[31] Paul Lutus, *Optical and Ray Tracing Mathematics*, Optical Ray Traser, **2014**.

[32] J. Peatross and M. Ware, *Physics of Light and Optics*, Brigham Young University, **2008**.

[33] Işının kırılması, http://www.bayar.edu.tr/besergil/1_emi_ozellikler.pdf (Ocak, **2018**).

[34] Fortran programı, http://groups.engin.umd.umich.edu/CIS/course.des/cis400/fortran/fortran.html (Ocak, **2018**).

[35] E. Erdem and F. Arikan, IONOLAB-RAY: A wave propagation algorithm for anisotropic and inhomogeneous ionosphere, TUBITAK-*Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 253906, pp. 1712–1723, **2017**.

[36] A. Kaçar, H. İ. Şahin, *NVIDIA Grafik İşlemci Mimarileri ve Paralel Programlama*, TTEK Aselsan A.Ş., Ankara, **2014**.

[37] Tesla grafik kartı, http://www.nvidia.com.tr/page/home.html (Ocak, **2018**).

[38] T. Lanfear, *Inside Kepler*, Nvidia Corporation, **2012**.

[39] Grafik işlemcilerin gelişimi, https://www.enterprisetech.com/2014/11/17/nvidia-doubles-tesla-gpu-accelerators/ (Ocak, **2018**)

[40] D. B. Kirk and W. W. Hwu, *Programming Massively Parallel Processors*, 2. edition, Morgan Kaufmann publishers, **2010**.

[41] Nvidia Visual Studio programı, https://developer.nvidia.com/nvidia-nsight-visual-studio-edition (Ocak, **2018**).

[42] Nvidia Visual Profiler programı, https://developer.nvidia.com/nvidia-visual-profiler (Ocak, **2018**).

EKLER

Tez kapsamında yapılmış olan faaliyetleri desteklemek için bazı görsel nitelikler aşağıdaki bölümlerde sunulmuştur.

Ek-1. Konfigürasyon Parametrelerinin Tanımları

Konfigürasyon parametreleri ile CUDA yazılımı beslenecek olup, bu parametre dosyasının adı "konfigurasyon.txt"dir. Bu dosyanın içeriğinde her satırda kullanılacak parametreler CUDA yazılımında ayrı ayrı çeklip değerlendirilmiştir. Bu parametrelerin yer aldığı dosyada her bir satırın sırasıyla içeriğine ait bilgiler aşağıdaki gibidir. Ayrıca, girilecek olan bu girdi parametreleri için tam sayı isteniyor ise integer, ondalık sayı isteniyorsa float ifadeleri kullanılmıştır.

1.Satir = Latitude değerlerinin dosya linki.
2.Satir = Longitude değerlerinini dosya linki.
3.Satir = Height değerlerinin dosya linki.
4.Satir = Lexico değerlerinin dosya linki.
5.Satir = m_i değerlerinin dosya linki.
6.Satir = n_n değerlerinin dosya linki.
7.Satir = NE değerlerinin dosya linki.
8.Satir = T_h değerlerinin dosya linki.
9.Satir = F değerlerinin dosya linki.
10.Satir = XX değerlerinin dosya linki.
11.Satir = YY değerlerinin dosya linki.
12.Satir = ZZ değerlerinin dosya linki.
13.Satir = İncelenecek iyonkürede kaç adet latitude olduğu bilgisi – integer.
14.Satir = İncelenecek iyonkürede kaç adet latitude oldugu bilgisi – integer.
15.Satir = İncelenecek iyonkürede kaç adet height oldugu bilgisi - integer.
16.Satir = İncelenecek iyonkürede kaç adet lexico oldugu bilgisi, satır sayısı - integer .
17.Satir = İyonkürede maksimum kaç adet kırılım yapılabilmesi için ayarlanabilir boy, açılacak hafıza alanlarını belirler – integer.
18.Satir = Vericiden çıkacak ışının frekans baslangıç değeri - float (10MHz icin 10 girilir.).
19.Satir = Vericiden çıkacak ışının frekans değişimi olacak mı bilgisi ,1 ya da 0 - integer.

20.Satir = Vericiden çıkacak ışının frekans artırım değeri, 1024 ışın icin eğer 19.satır 1 ise burası aktif olur. Her ışında bu değer kadar ilerler – float.

21.Satir = Vericiden çıkacak ışının Latitude baslangıç değeri - float.

22.Satir = Vericiden çıkacak ışının Latitude değişimi olacak mı bilgisi ,1 ya da 0 - integer.

23.Satir = Vericiden çıkacak ışının Latitude artırım değeri, 1024 ışın için eğer 22.satır 1 ise burası aktif olur. Her–ışında bu değer kadar ilerler - float.

24.Satir = Vericiden çıkacak ışının Longitude baslanğıç değeri - float.

25.Satir = Vericiden çıkacak ışının Longitude değişimi olacak mı bilgisi ,1 ya da 0 – integer.

26.Satir = Vericiden çıkacak ışının Longitude artırım değeri, 1024 ışın için eğer 25.satır 1 ise burası aktif olur. Her ışında bu değer kadar ilerler – float.

27.Satir = Vericiden çıkacak ışının Azimuth baslangıç degeri – float.

28.Satir = Vericiden çıkacak ışının Azimuth değişimi olacak mı bilgisi ,1 ya da 0 – integer.

29.Satir = Vericiden çıkacak ışının Azimuth artırım değeri, 1024 ışın için eğer 25.satır 1 ise burası aktif olur. Her ışında bu değer kadar ilerler – float.

30.Satir = Vericiden çıkacak ışının Elevation baslangıç değeri – float.

31.Satir = Vericiden çıkacak ışının Elevation değişimi olacak mı bilgisi ,1 ya da 0 – integer.

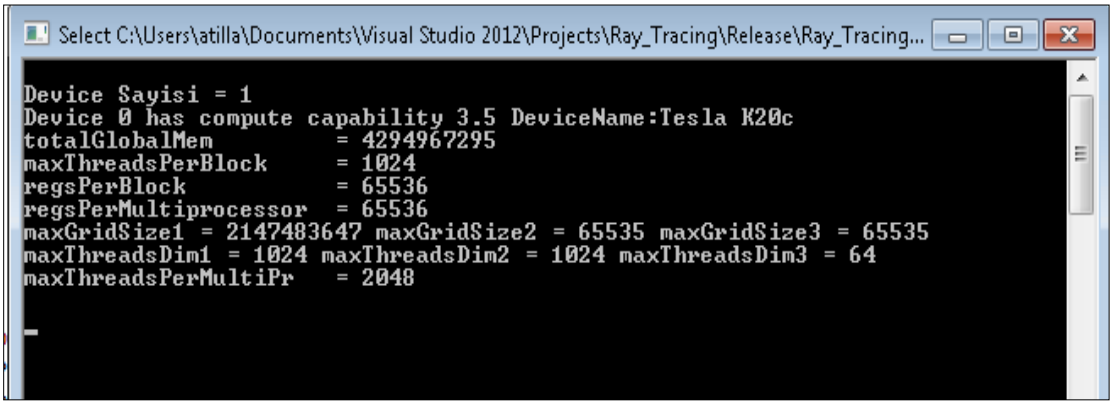
32.Satir = Vericiden çıkacak ışının Elevation artırım değeri, 1024 ışın için eğer 25.satır 1 ise burası aktif olur. Her ışında bu değer kadar ilerler – float.

33.Satir = Kullanılacak azimuth ve elevation değerleri dosyadan mı alınsın, yoksa satır 27 ile 32 arası–mı geçerli olsun kontrolüdür. - integer (1 ya da 0).

34.Satir = Vericiden çıkacak ışının Azimuth değerlerinin dosya linki.

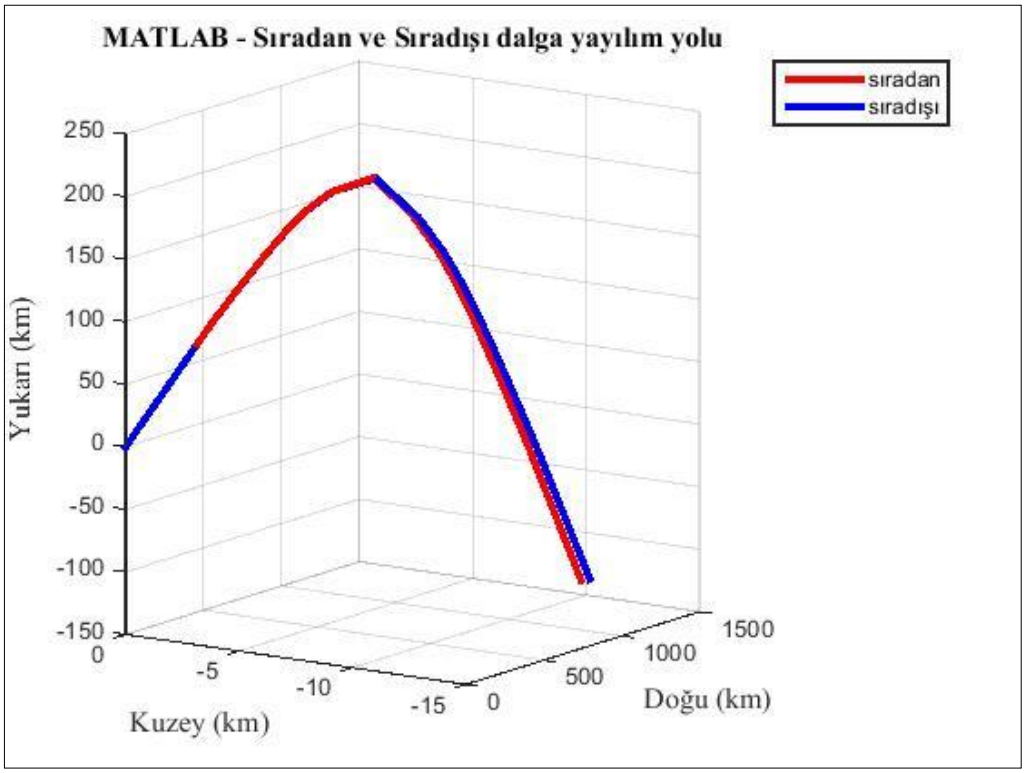
35.Satir = Vericiden çıkacak ışının Elevation değerlerinin dosya linki.

Ek-2. Kullanılan Grafik İşlemcinin Özellikleri

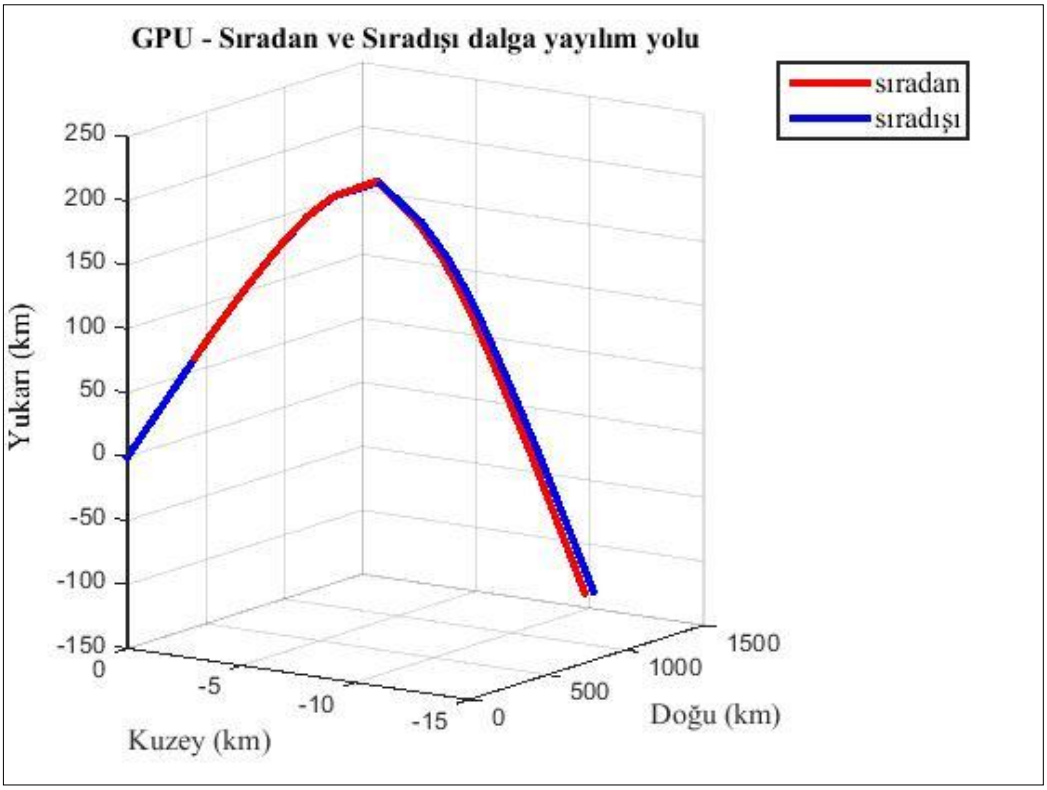


Şekil Ek-2.1. Grafik işlemciye ait bilgiler.

Ek-3. Elde Edilen Grafikler ve CUDA ile Süre Ölçümü



Şekil Ek-3.1. MATLAB çıktısı ile elde edilen sıradan ve sıradışı dalganın yayılım yolu.

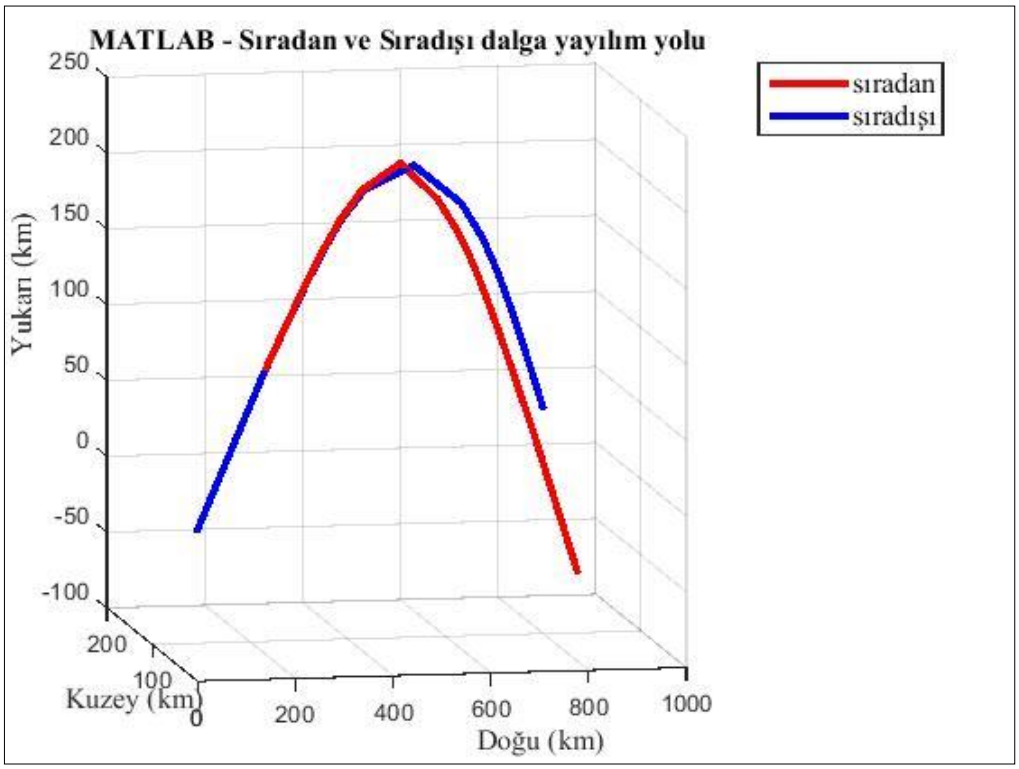


Şekil Ek-3.2. GPU'da çalıştırılan CUDA yazılımının çıktısı ile elde edilen sıradan ve sıradışı dalganın yayılım yolu.

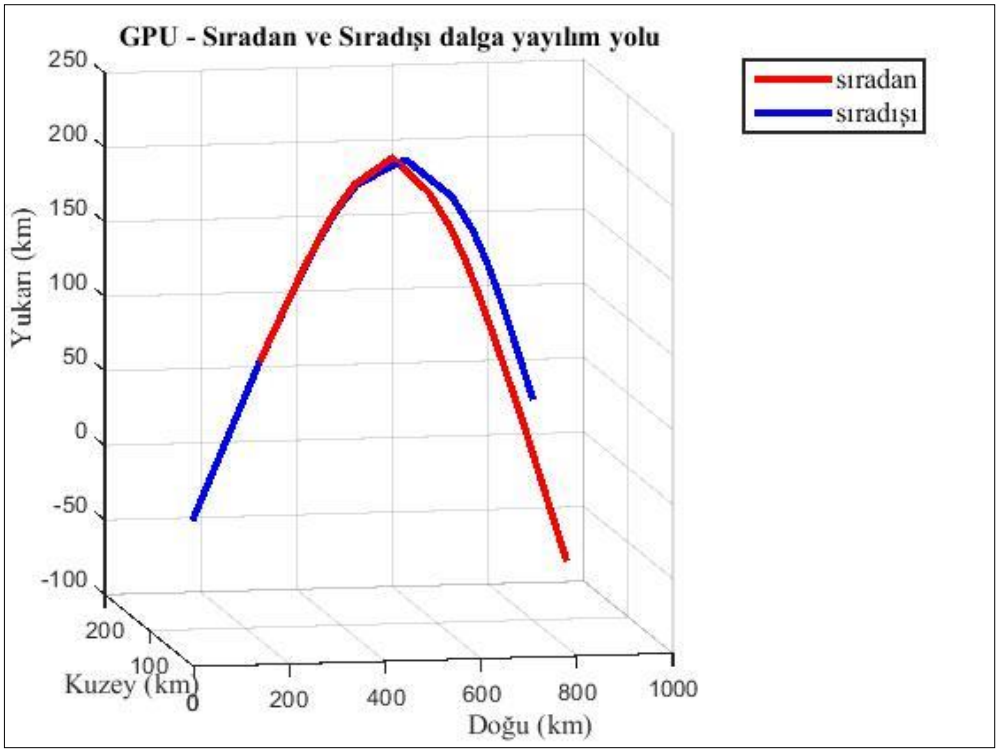
```
cudaThreadSynchronize();
cudaEventRecord( start, 0 );

gpu_fOrdinary(fTfrequency, fTLatitude, fTLongitude, fTheight, fTrans_zen, fTAzimuth);

cudaEventRecord( stop, 0 );
cudaEventSynchronize( stop );
cudaEventElapsedTime( &time, start, stop );

printf("gpu_fOrdinary: %e miliseconds\n", (double)(time));
```

Şekil Ek-3.3. Grafik işlemcide çalıştırılan CUDA kodu ile süre ölçümü.

Ek-4. Sıradan ve Sıradışı Dalga Yayılım Yolları

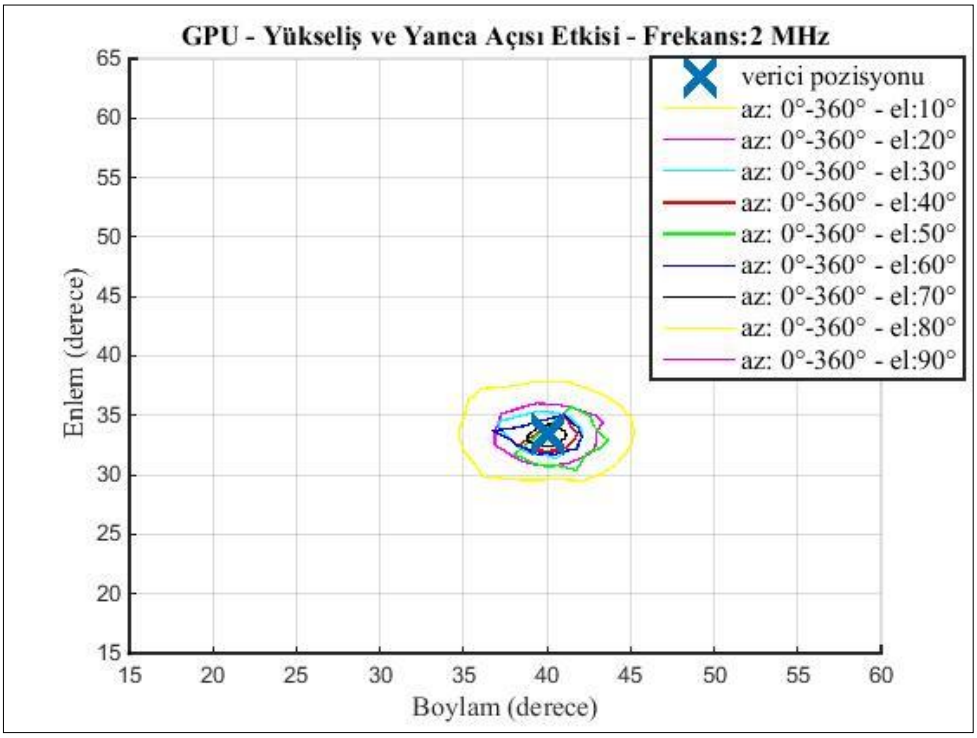


Şekil Ek-4.1. MATLAB çıktısı ile elde edilen sıradan ve sıradışı dalganın yayılım yolu.

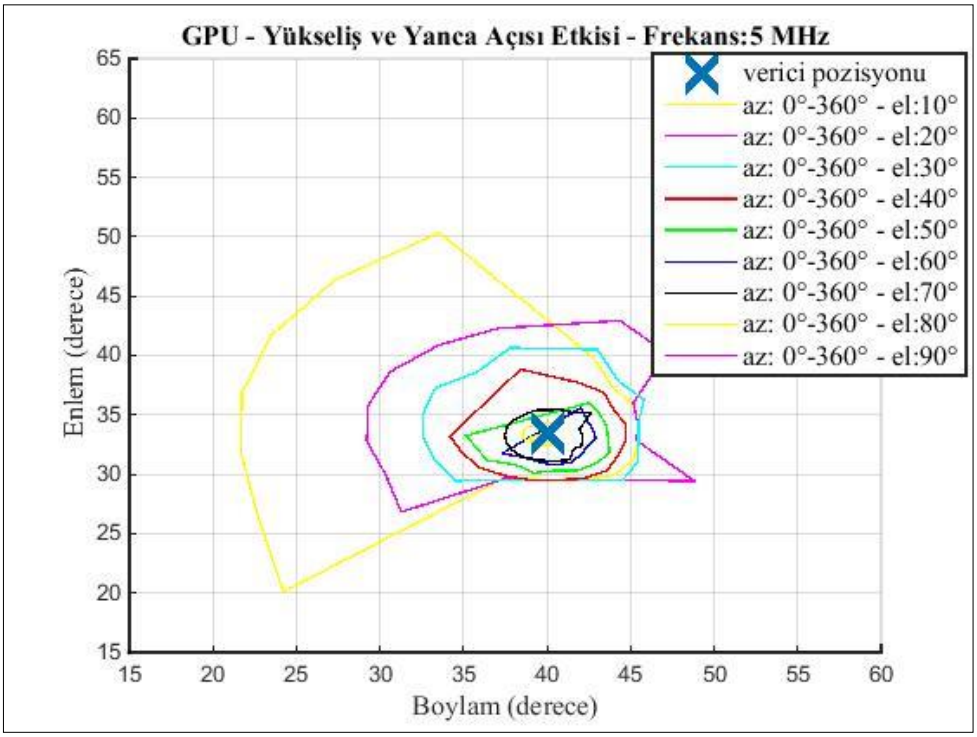


Şekil Ek-4.2. GPU'da çalıştırılan CUDA yazılımının çıktısı ile elde edilen sıradan ve sıradışı dalganın yayılım yolu.

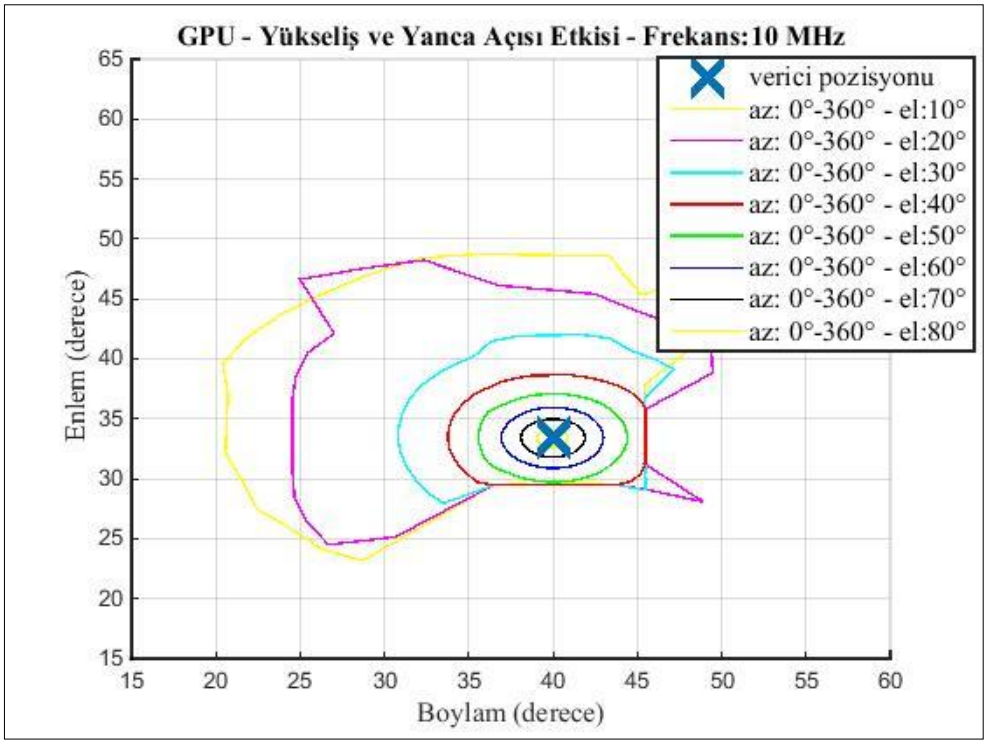
Ek-5. Frekans Bağımlı Olarak Işınların Yayılım Yolları



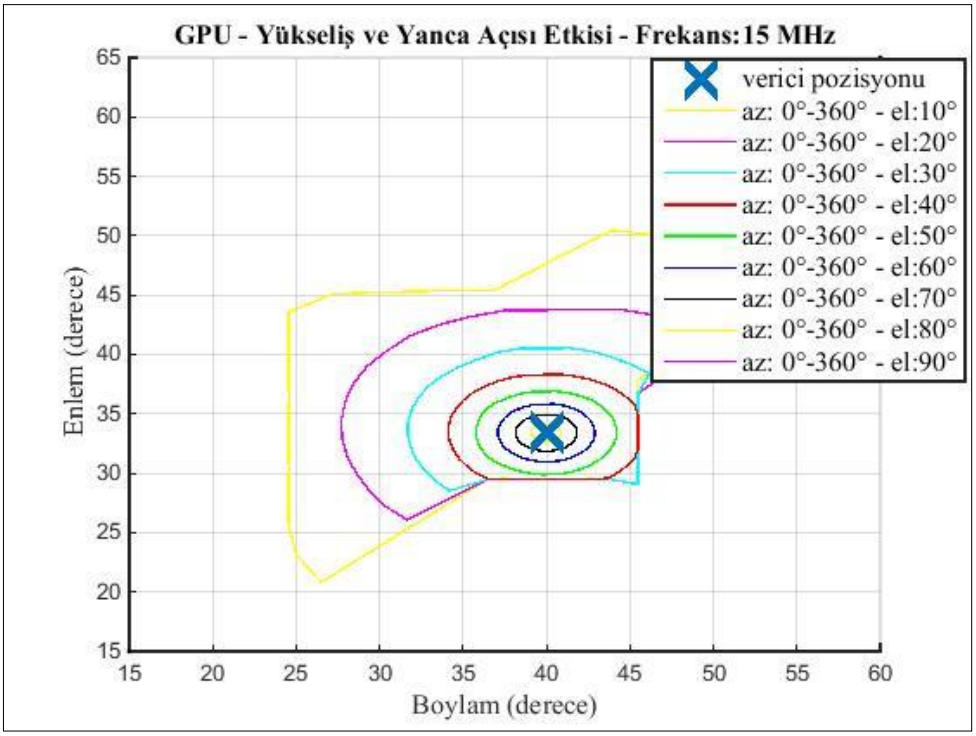
Şekil Ek-5.1. 2 MHz frekans ile yükseliş ve yanca bağımlı sıradan dalga yollarının görüntüsü.



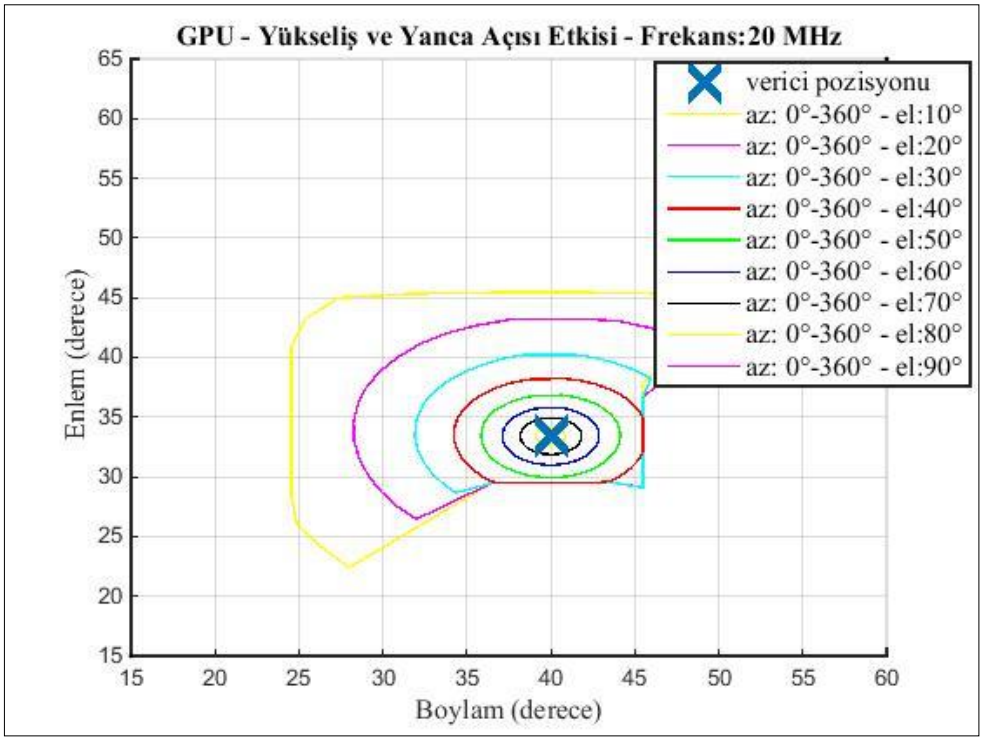
Şekil Ek-5.2. 5 MHz frekans ile yükseliş ve yanca bağımlı sıradan dalga yollarının görüntüsü.



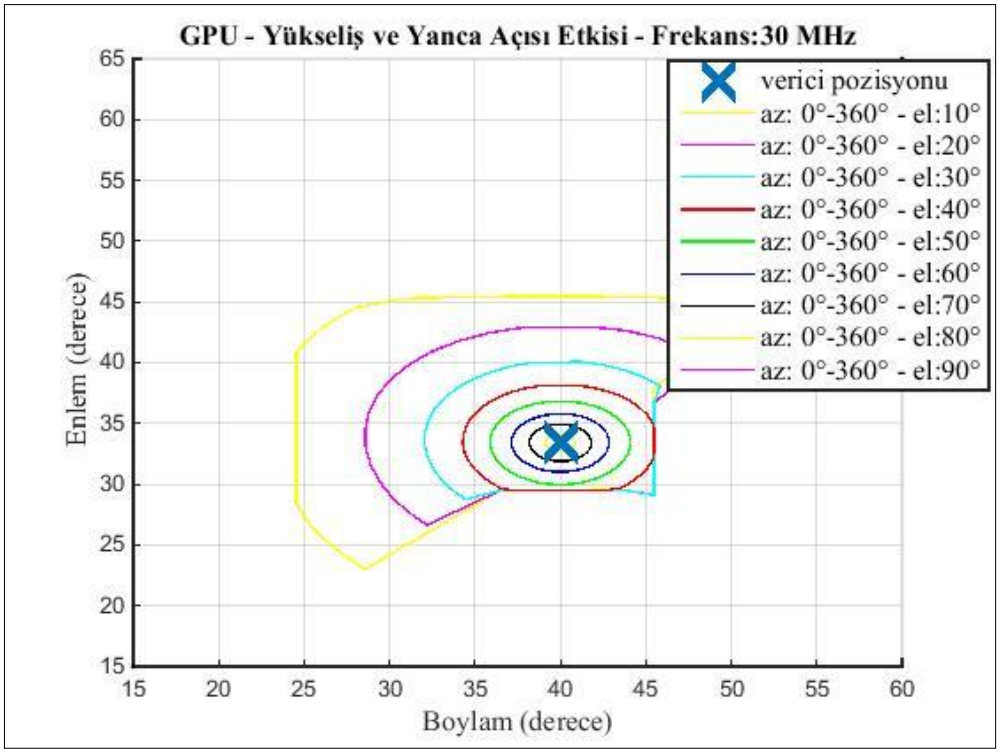
Şekil Ek-5.3. 10 MHz frekans ile yükseliş ve yanca bağımlı sıradan dalga yollarının görüntüsü.



Şekil Ek-5.4. 15 MHz frekans ile yükseliş ve yanca bağımlı sıradan dalga yollarının görüntüsü.



Şekil Ek-5.5. 20 MHz frekans ile yükseliş ve yanca bağımlı sıradan dalga yollarının görüntüsü.



Şekil Ek-5.6. 30 MHz frekans ile yükseliş ve yanca bağımlı sıradan dalga yollarının görüntüsü.

Ek-6. CUDA yazılımının kullanımı

IONOLAB-RAY algoritması içerisinden girdi bilgilerinin CUDA yazılımına beslenir hale getirilmesi, CUDA yazılımının çalıştırılması ve sonuç dosyalarının okunup incelenmesi ile ilgili bilgiler bu bölümde verilmiştir. Örnek-9'un hazırlanması için yapılan işlemler aşağıda verilmiştir. Sırasıyla aşağıdaki işlemlerin yapılması ile hazırlanan CUDA yazılımının sonuçları görülebilmektedir.

- IONOLAB-RAY algortimasında incelenecek iyonküre ile ilgili bilgiler girdi dosyaları olarak kaydedilecektir. Örnek-9'daki veri seti için kullanım şu şekildedir;
  `fmainprocess(5,33.4,40,0,30,0,'AtillaTest_0000.mat', 0)`
- MATLAB'da "kayitdosyasi.m" adında girdi dosyalarının kaydedilmesi için gerekli fonksiyonların olduğu kod çalıştırılacaktır.
  Kayitdosyasi.m içeriği şağıdaki gibidir.

```
fid = fopen('Gpu_Inputs\height.bin', 'w', 'l');
fwrite(fid, height.', 'int');
fclose(fid);

fid = fopen('Gpu_Inputs\lat.bin', 'w', 'l');
fwrite(fid, lat.', 'int');
fclose(fid);

fid = fopen('Gpu_Inputs\lexico_values.bin', 'w', 'l');
aa= lexico_values.';
fwrite(fid, aa(:), 'int');
fclose(fid);

fid = fopen('Gpu_Inputs\long.bin', 'w', 'l');
fwrite(fid, lon.', 'int');
fclose(fid);

fid = fopen('Gpu_Inputs\m_i.bin', 'w', 'l');
fwrite(fid, (m_i).', 'double');
fclose(fid);

fid = fopen('Gpu_Inputs\n_n.bin', 'w', 'l');
fwrite(fid, (n_n).', 'double');
fclose(fid);

fid = fopen('Gpu_Inputs\NE.bin', 'w', 'l');
fwrite(fid, (NE).', 'double');
fclose(fid);

fid = fopen('Gpu_Inputs\T_h.bin', 'w', 'l');
fwrite(fid, (T_h).', 'double');
fclose(fid);

fid = fopen('Gpu_Inputs\XX.bin', 'w', 'l');
fwrite(fid, (XX).', 'double');
fclose(fid);

fid = fopen('Gpu_Inputs\YY.bin', 'w', 'l');
fwrite(fid, (YY).', 'double');
```

```
    fclose(fid);

    fid = fopen('Gpu_Inputs\ZZ.bin', 'w', 'l');
    fwrite(fid, (ZZ).', 'double');
    fclose(fid);

    fid = fopen('Gpu_Inputs\F.bin', 'w', 'l');
    fwrite(fid, (F).', 'double');
    fclose(fid);

    azim_set   = (0:33:360); %incelenecek azimuth ve elevation degerleri
ayarlanir
    for i=1:1024
       elevations(i) =  floor((i-1)/size(azim_set,2));
       azimuths(i)   = azim_set(mod(i-1,size(azim_set,2))+1);
    end

    fid = fopen('Gpu_Inputs\azimuths.bin', 'w', 'l');
    fwrite(fid, azimuths, 'double');
    fclose(fid);

    fid = fopen('Gpu_Inputs\elevations.bin', 'w', 'l');
    fwrite(fid, elevations, 'double');
    fclose(fid);
```

- Daha sonra bu kaydedilen dosyaların adres bilgileri ve diğer girdi parametreleri, Ek-1'de belirtilen konfigürasyon dosyasına hem link hem de diğer bilgiler olarak girilecektir. Örnek 9 için hazırlanan konfigürasyon dosyası aşağıdaki gibidir;

```
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/lat.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/long.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/height.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/lexico_values.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/m_i.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/n_n.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/NE.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/T_h.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/F.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/XX.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/YY.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/ZZ.bin
16
21
421
141456
1500
5
0
0
33.4
0
0.01
40.0
0
0
0.0
0
0.35
25.0
0
0.087
45
25
45
```

```
30
1
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/azimuths.bin
c:/Users/atilla/Documents/Ray_Tracing/Gpu_Inputs/elevations.bin
```

- CUDA yazılımının çalıştırılması için “Ray_Tracing.exe” çift tıklanacaktır. Birkaç saniye içerisinde işlenmiş olan 1024 adet ışının hem sıradan hem de sıradışı ışın yolları exe dosyasının çalıştığı dizinde oluşacaktır.
- Çıktı olarak oluşan dosyalar 1024 adet farklı ışının sıradan ve sıradışı yollarına ait bilgilerini içermektedir. MATLAB üzerinde bu veriler incelenebilmektedir. Örneğin; EBY koordinat sistemine göre oluşan sıradan dalga yollarına ait bilgiler incelenmek istenirse aşağıdaki işlemler yapılmalıdır.

```
figure;
fid1 = fopen('C:\Users\akacar\Desktop\Ordinary_llaX_Out.bin', 'r', 'l');
fid2 = fopen('C:\Users\akacar\Desktop\Ordinary_llaY_Out.bin', 'r', 'l');
fid3 = fopen('C:\Users\akacar\Desktop\Ordinary_llaZ_Out.bin', 'r', 'l');
fid4 = fopen('C:\Users\akacar\Desktop\Ordinary_iPIndex_Out.bin', 'r', 'l');
hold on
grid on

ordlla_X_gpu = fread(fid1, 'double');
fclose(fid1);
ordlla_Y_gpu = fread(fid2, 'double');
fclose(fid2);
ordlla_Z_gpu = fread(fid3, 'double');
fclose(fid3);
ordHeight_Gpu = fread(fid4, 1024, 'int32');
fclose(fid4);
oHeight_Gpu =ordHeight_Gpu(:)+1;

for i=1:1024
    aa = ordlla_X_gpu(i:1024:end);
    oLLA_X_gpu{i} = aa(1:oHeight_Gpu(i));
    aa = ordlla_Y_gpu(i:1024:end);
    oLLA_Y_gpu{i} = aa(1:oHeight_Gpu(i));
    aa = ordlla_Z_gpu(i:1024:end);
    oLLA_Z_gpu{i} = aa(1:oHeight_Gpu(i));

    k=0;
    if(oHeight_Gpu(i)>1)
        for j=2:oHeight_Gpu(i)
                if((oHeight_Gpu(i)>0) & ((abs((oLLA_X_gpu{i}(j))-
(oLLA_X_gpu{i}(j-1))) > 3) || ((oLLA_Y_gpu{i}(j))<10 ||
(oLLA_Y_gpu{i}(j))>55) || (oLLA_X_gpu{i}(j)<0) || (oLLA_Y_gpu{i}(j)<0)))
                    k = oHeight_Gpu(i)-j+1;
                    bos = j:1:oHeight_Gpu(i);
                    break;
                end

        end
    end
    if(k)
        oLLA_X_gpu{i}(bos) = [];
        oLLA_Y_gpu{i}(bos) = [];
        oLLA_Z_gpu{i}(bos) = [];
    end
    oHeight_Gpu(i)=oHeight_Gpu(i)-k;
    clear bos
end
```

Buraya kadar EBY koordinat sistemi ile oluşturulan sonuçlar okunmuştur. Sonuçların iyonküre analiz alanı içinde olup olmadığı ve negatiflik kontrolü yapılmıştır. Bundan sonra aşağıdaki işlemler ile 1024 adet ışının bazıları ekrana çizdirilmektedir.

```
k=1;
hold on;
ElPoints=[];
AzPoints=[];
for i=1:1:1024
        gpuSonucOrd.freq.trans_lat.trans_long.trans_height.trans_elev.trans_azi.n
        _points = [ iterasyon 0];
        gpuSonucOrd.freq.trans_lat.trans_long.trans_height.trans_elev.trans_azi.p
        _LLA= zeros(1,iterasyon,3);
        gpuSonucOrd.freq.trans_lat.trans_long.trans_height.trans_elev.trans_azi.p
        _LLA(:,:,1) = oLLA_X_gpu{i}(1:iterasyon);
        gpuSonucOrd.freq.trans_lat.trans_long.trans_height.trans_elev.trans_azi.p
        _LLA(:,:,2) = oLLA_Y_gpu{i}(1:iterasyon);
        gpuSonucOrd.freq.trans_lat.trans_long.trans_height.trans_elev.trans_azi.p
        _LLA(:,:,3) = oLLA_Z_gpu{i}(1:iterasyon);
        isim = sprintf('az: 0°-360° - el:%d°',floor((i-1)/size(azim_set,2)));
        xx(k) = oLLA_X_gpu{i}(iterasyon);
        yy(k) = oLLA_Y_gpu{i}(iterasyon);
        k = k+1;
        if(~mod(i,size(azim_set,2)))
            str = [isim];
             xx(k) = xx(1);
             yy(k) = yy(1);
            ElPoints=[ElPoints; yy', xx', elevations(i)*ones(12,1)];
            AzPoints=[AzPoints; yy', xx', azimuths(1:12)'];
            k=1;
        end
end

figure
[X,Y] = meshgrid(linspace(min(ElPoints(:,1)),max(ElPoints(:,1)),n),
linspace(min(ElPoints(:,2)),max(ElPoints(:,2)),n));
hold on; grid on;
Z=griddata(ElPoints(:,1),ElPoints(:,2),ElPoints(:,3),X,Y);
contourf(X,Y,Z,20); acibar = colorbar;
title(acibar,'Yükseliş Açısı', 'FontName', 'Times New Roman', 'FontSize',12);
xlabel('Boylam (derece)', 'FontName', 'Times New Roman', 'FontSize',12)
ylabel('Enlem (derece)', 'FontName', 'Times New Roman', 'FontSize',12)
zlabel('Yükseklik (km)', 'FontName', 'Times New Roman', 'FontSize',12)
title('Yükseliş Açısına Bağlı Kapsama Alanı Etkisi','FontName', 'Times New
Roman', 'FontSize', 12);
hArray(1)=plot(40,33.4,'X','LineWidth',10,'MarkerSize',20);
strArray{1} = ['verici pozisyonu'];
legend(hArray(1),strArray{1}, 'verici pozisyonu','FontName', 'Times New Roman',
'FontSize', 12);
plot([10 55],[20 65],'.k','LineWidth',0.00000001); % eksen ayarlaması
```

Bu işlemler sonucunda CUDA yazılımı ile elde edilen 1024 adet ışının ışın yolu IONOLAB-RAY algoritmasına göre okunmuştur. Bu ışın yolları istenen aralıklarla grafiksel olarak çizdirilebilir. Yukarıda verilen örnek fonksiyonlar ile Örnek-9'da çizdirilen kapsama alanı görüntüsü oluşmaktadır.